Musical robots have already inspired the creation of worldwide robotic dancing contests, as RoboCup-Junior's Dance, where school teams, formed by children aged eight to eighteen, put their robots in action, performing dance to music in a display that emphasizes creativity of costumes and movement. This book describes and assesses a user-customizable framework for robot dancing edutainment applications. The proposed architecture enables the definition of choreographic compositions, which result on a conjunction of reactive dancing motions in real-time response to multi-modal inputs. These inputs are shaped in the form of three rhythmic events, different dance floor colors, and the awareness of the surrounding obstacles. This architecture was applied to a Lego-NXT humanoid robot dancing on a real-world dance stage. We report on an empirical evaluation over the overall robot dance performance made to a group of students after a set of live demonstrations. This evaluation validated the framework's potential application in edutainment and its ability to sustain the interest of the general audience by offering a reasonable compromise between musical-synchrony, variability and animacy.

João Lobato Oliveira

João L. Oliveira received the MSc degree in Electrical and Computers Engineering from the Faculty of Engineering of the University of Porto (FEUP) in 2008. Since 2008 he is a PhD student in Informatics Engineering at FEUP and a research member of LIACC and INESC Porto. His main research is in Autonomous Robot Dancing and Audio Beat Tracking.

978-3-659-22225-2

Autonomous Robot Dancing

João Lobato Oliveira

Autonomous Robot Dancing

A Customizable Robot Dancing Framework based on Multi-Modal Events

LAP LAMBERT Academic Publishing

Lobato Oliveira

João Lobato Oliveira

Autonomous Robot Dancing

João Lobato Oliveira

Autonomous Robot Dancing

A Customizable Robot Dancing Framework based on Multi-Modal Events

LAP LAMBERT Academic Publishing

Impressum / Imprint
Bibliografische Information der Deutschen Nationalbibliothek: Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.

Bibliographic information published by the Deutsche Nationalbibliothek: The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available in the Internet at http://dnb.d-nb.de.

Coverbild / Cover image: www.ingimage.com

Verlag / Publisher:
LAP LAMBERT Academic Publishing
ist ein Imprint der / is a trademark of
AV Akademikerverlag GmbH & Co. KG
Heinrich-Böcking-Str. 6-8, 66121 Saarbrücken, Deutschland / Germany
Email: info@lap-publishing.com

Herstellung: siehe letzte Seite /
Printed at: see last page
ISBN: 978-3-659-22225-2

Zugl. / Approved by: Porto, Faculty of Engineering of the University of Porto, MSc Thesis in Electrical and Computers Engineering, 2008

Acknowledgments

First of all, I would like to express my gratitude to my supervisors, Prof. Dr. Luis Paulo Reis and Prof. Dr. Fabien Gouyon, for all their support and guidance in the development of this project, helping me whenever needed and making all this work possible. I would also like to thank Luis Gustavo Martins, from INESC Porto, for his explanations on Marsyas operation and functionality.

Finally, but not least, I wish to thank all my family for their lifetime support, Ana for her love and patience, and all my friends for their friendship and encouragement throughout the years, in especial Gustavo Meirinhos and Pedro Vieira for helping me with the robot design and construction, and Pedro Allegro for his companionship and assistance.

Table of Contents

List of Figures

List of Tables

Acronym List

A/D	Analog/Digital
AI	Artificial Intelligence
API	Application Programming Interface
BT	Bluetooth
CD	Complex Difference
CI	Chaotic Itinerancy
CM	Complex Mixtures
COM	COMmunications serial port
CoP	Center of Pressure
CPG	Central Pattern Generator
CPU	Central Processing Unit
D2K	Data-to-Knowledge
DAC	Digital-to-Analog Converter
DAT	Dynamic Attending Theory
dB	Decibels
DC	Direct Current
DoF	Degrees-of-Freedom
DP	Dynamic Programming
EEM	Entrainment Ensemble Model
FEUP	Faculty of Engineering of the University of Porto
FFT	Fast Fourier Transform
FIR	Finite Impulse Response
FNN	Feed-Forward Neural Network
FP	False Positives
FPS	Frames per Second

GUI	Graphical User Interface
HFC	High-Frequency Content
Hi-Fi	High-Fidelity
HMM	Hidden Markov Model
HRI	Human-Robot Interaction
HW	Hamming Window
I^2C	Inter-Integrated Circuit
IBI	Inter-Beat Interval
ICA	Independent Component Analysis
IDE	Integrated Development Environment
IMIRSEL	International Music Information Retrieval Systems Evaluation Laboratory
INESC	Institute for Systems and Computer Engineering
I/O	Input/Output
IP	Internet Protocol
ISMIR	International Symposium on Music Information Retrieval
LCD	Liquid Crystal Display
LED	Light-Emitting Diode
LFO	Learning-From-Observation
LIACC	Laboratory of Artificial Intelligence and Computer Science
M[ε]X	Musical Expression
M2K	Music-to-Knowledge
MARSYAS	Music Analysis, Retrieval and Synthesis for Audio Signals
Max/MSP	Max/Max Signal Processing
MFC	Microsoft Foundation Class
MIDI	Musical Instrument Digital Interface
MIEEC	Integrated Masters in Electrical and Computer Engineering
MIR	Music Information Retrieval
MIREX	Music Information Retrieval Evaluation eXchange
MoCap	Motion Capture

MPMA	Multiple Paired Inverse/Forward Model
NL	Negative Log-Likelihood
NP	Non-Pitched Percussion
NWPD	Normalized Weighted Phase Deviation
OS	Operating System
PC	Personal Computer
PD	Phase Deviation
PN	Pitched Non-Percussion
PP	Pitched Percussion
PPL	Phase-Locked Loops
QoM	Quantity of Motion
RAM	Random Access Memory
RCD	Rectified Complex Domain
RNNPB	Recurrent Neural Network with Parametric Bias
ROC	Receiver Operating Characteristic
SMI	Silhouette Motion Image
SMS	Spectral Modeling Synthesis
SOL	Sound Onset Labellizer
STFT	Short Time Fourier Transform
SVM	Support Vector Machine
TFR	Time-Frequency and Time-Scale Representation
TGA	Topological Gesture Analysis
TP	True Positives
UDP	User Datagram Protocol
USB	Universal Serial Bus
UTM	Unit of Telecommunications and Multimedia
WAV	WAVEform Audio Format
WPD	Weighted Phase Deviation
WRM	Wavelet Regularity Modulus
XML	Extensible Markup Language
ZMP	Zero-Moment Point

Chapter 1

Introduction

Musical robots are increasingly present in multidisciplinary edutainment areas thrilling fanciers worldwide with ensemble performances with professional dancers and musicians[1] (Weinberg, 2007b), and being active intervenients in pedagogical and therapeutic scenarios (Kozima, Michalowski, & Nakagawa, 2008). They already inspired the creation of worldwide robot dancing contests where school teams, formed by students of various ages, program their robots for dancing to music in a display that emphasizes creativity of costumes and movement (RoboCupJunior, 2008). Although these robotic systems undeniably demonstrate personality they typically lack from musical awareness and animacy, with pre-programmed deaf robots or dancing robots strictly tuned to music with no human control.

In this book we describe a user-customizable framework for robot dancing edutainment applications. Contrasting to other approaches, the developed system supplies a flexible interface for defining choreographic compositions for Lego-NXT-based dancing robots in reactive response to external multi-modal events. In order to assure an autonomous and expressive behavior, the developed system explored the rhythmic

[1] See when ASIMO robot conducted the Detroit Symphony Orchestra in a performance of Mitch Leigh's "The Impossible Dream" from the Man from La Mancha (on May 13th, 2008)[1]. Online at: http://www.autoblog.com/tag/asimo+orchestra/.

phenomenon beyond music, which is composed of a succession of note-events that generally makes people move (Bello, et al., 2005). To parse these musical rhythmic events from polyphonic audio signals on-the-fly we implemented a real-time onset detection algorithm based on the signal's spectral flux. In addition, on top of the onset detection we applied an adaptive peak-picking algorithm to retrieve three levels of rhythmic intensity. In combination with these rhythmic events, the framework deals with external inputs in the form of sensorial events, such as floor colors and obstacles. Such multi-modal support enables the creation of more variable and dynamic dancing sequences, while assuring the avoidance of obstacles. On top of the system, a user-interface gives high-level control over the musical analysis and the Lego-NXT robot's sensorimotor parameters. Moreover, it provides an online visualization of the detected note-onsets for the calibration of the performed onset detection.

From an educational point of view this framework provides an intuitive environment for learners and children to experiment the creation of their own dancing behaviors, by generating robot dancing motions in response to multi-modal events. Its autonomy and its basis on Lego robots, allied to the use of an amusing aesthetics, enable the generation of varied and expressive dance performances capable of entertaining vast audiences, of various ages.

To validate our approach, and considering such applications, a vast audience formed by students ranging from 6 to 17 years old empirically evaluated the developed robotic system. Their judgment suggested that our implementation served its edutainment purposes but, due to hardware limitations, is still far from the requested variety of human-inspired movements and musical-synchrony. Nevertheless, the proposed architecture can be used as a plausible platform for robot dancing contests such as (RoboCupJunior, 2008) and (ROBO-ONEEntertainment, 2008).

This chapter aims to contextualize our research in autonomous robot dancing, presenting our motivations towards robot dancing based on multi-modal events, and our methodology in consideration to the used tools and chosen architecture. As a starting point to this book, we introduce some basic thoughts and definitions followed by the main objectives involved.

1.1 Motivations

The motivations which supported the topic of this work were drawn from a conjunction of factors. To better describe the approached research topics and in order to motivate the reader to follow our approach through this book, in this section we summarize the scientific and technological context of this work and introduce the hosting institutions and the author's personal trajectory before enrolling into this project. As one last point of interest, we introduce some of the inspirational thoughts from the literature beyond the conception of this work.

1.1.1 Scope

More and more "Dancing Robots" and "Human-Robot Musical Interaction" are becoming very common terms. In an increasing number of research labs around the world (especially in Japan), researchers follow a quest to find the perfect solution to achieve a rhythmic perceptive and interactive dancing robot.

Dance represents a form of non-verbal communication in social rhythmic interactions, serving the increasingly importance given to Human-Robot Interaction (HRI) and by the human natural need of keeping relationships. This kind of interaction can be achieved by social intelligent robotic agents that can map intermodal rhythms, perceived from the environment, to dance motions; by imitation and improvisation while following the musical rhythm.

Robot Dancing is an interdisciplinary theme which increasingly enthusiasms fanciers worldwide, while assisting people's life, being further enjoyable. As the term points out, it embraces several topics such as *dance*, *rhythm*, *rhythmic perception*, *multi-modality*, *autonomous robotic systems*, *sensorimotor synchronization*, *reactive and anticipatory behavior*, *rhythmic entrainment and embodiment*, *and human-robot social interaction*.

1.1.2 Research at LIACC and INESC Porto

This research was carried out at the Laboratory of Artificial Intelligence and Computer Science (LIACC), under the supervision of Prof. Dr. Luis

Paulo Reis, in association with the Institute for Systems and Computer Engineering (INESC) of Porto, under the supervision of Prof. Dr. Fabien Gouyon.

LIACC was created in 1988 to promote the collaboration of researchers that were separately working in the fields of Computer Science and Artificial Intelligence in different Faculties. LIACC aims to help solve general computer science problems, from security to software reliability. These hard, real-world problems can only be solved in the long term by combining the power of formal methods with more technology-oriented approaches and were used as a frame of reference in defining the LIACC short-term goals. Since June 2007 LIACC's activities are organized around four research groups: Advanced Programming Systems, Distributed Artificial Intelligence and Robotics, Formal Models of Computation, and Language, Complexity and Cryptography.

INESC Porto is an institution created to act as an interface between the academic world, the world of industry and services, as well as the public administration, in the framework of the Information Technologies, Telecommunications and Electronics. Its activities range from research and development, to technology transfer, consulting and advanced training. Its main research areas of interest are Telecommunications and Multimedia, Power Systems, Manufacturing Systems Engineering, Information and Communication Systems, and Optoelectronics.

Among the various areas of expertise within these institutions, our work was specially related to Autonomous Robotic Systems, covered by Distributed Artificial Intelligence and Robotics, in LIACC; and Automatic Rhythm Description, covered by the Telecommunications and Multimedia (UTM) unit, in INESC.

1.1.3 Personal Trajectory

This book reports all the research and work developed during the last five months (i.e., February to July of 2008) of the 5th year of the Integrated Masters in Electrical and Computer Engineering (MIEEC) at the Faculty of Engineering of the University of Porto (FEUP). It represents the final project in an academic cycle of five years that came to an end. A cycle

through which I cultivated the skills and knowledge needed for the development of such work.

Everything else came, since early years, from the natural dancing activities in complex interactive environments (e.g., discos and festivals). Activities that promoted curiosity on all the processes involved in the rhythmic perception of music and its subsequent embodiment in the form of dance.

1.1.4 Thoughts from the Literature

"The goal of AI has been characterized as both the construction of useful intelligent systems and the understanding of human intelligence…trying to build truly intelligent autonomous robots." (Brooks, 1991b, p. 1).

"Intelligent systems are decomposed into independent and parallel activity producers which all interface directly to the world through perception and action…in a behavioral reactive manner." (Brooks, 1991a, p. 1).

"The intelligence does not come from a set of rules that describe "how music works." Rather, the intelligence comes from continuous transformations of the signal, and the way that these transformations are manifested as musical behaviors (…) There is no function in which it is decided what is the "right thing to do" as the signal is being processed." (Scheirer, 2000, p. 75).

"The computational model and the psychoacoustic experiment play overlapping and complementary roles in advancing our knowledge about the world." (Scheirer, 2000, p. 66).

"A computational theory of music cognition indeed, any computational theory of perception or cognition in any modality must be validated through comparison to meaningful experimental data. (…) The model builder might have implemented a large lookup table and listed all the

stimulus patterns and the appropriate responses." (Desain & Honing, 1992, p. 5).

Resembling human perception and behavior (as human intelligent capabilities), the research program we envisioned aims to develop a rhythmic perceptual dancing robotic system that would generate dance performances with a reasonable compromise between musical-synchrony, animacy and variability, ultimately enhancing the long-term interest of the general audience. The resulting dance should be tested in a real-world environment and empirically compared to human dance performances.

1.2 Aims and Outline

"*The fundamental slicing up of an intelligent system is in the orthogonal direction dividing it into activity producing subsystems. Each activity (pattern of interactions with the world), or behavior producing system individually connects sensing to action... The advantage of this approach is that it gives an incremental path from very simple systems to complex autonomous intelligent systems*." (Brooks, 1991a).

The main objective of this book is to create a user-customizable framework which may be used in the control of a dancing humanoid robot to perform seemingly autonomous dance movements in synchrony to the musical rhythm, without former knowledge of the music. Ultimately, by additionally combining external sensorial events, the proposed framework should enable an interesting long-term relationship between general human audience and the artificial dancing agent, by enabling a compromise between *musical-synchrony*, *variability*, and *animacy*, while exhibiting an autonomous but controllable behavior.

For a list of publications about the work described in this book see (Oliveira, Gouyon, & Reis, 2008a), (Oliveira, 2008b), and (Oliveira, Reis, Faria, & Gouyon, 2012). Abstracts and electronic versions of these

publications, and future information about this work, are available in http://paginas.fe.up.pt/~ee03123/.

1.3 Methodology and Tools

The implementation of the proposed robot dancing system required the use of a conjunction of software applications, which worked together to develop, experiment, and ultimately control the dancing robotic platform. In this section we present the main used tools and their role in the implementation of the proposed robotic system.

1.3.1 Marsyas

Marsyas (Music Analysis, Retrieval and Synthesis for Audio Signals)[2] is an open source software framework for rapid prototyping and experimentation with audio analysis and synthesis with specific emphasis to music signals and Music Information Retrieval (MIR) (Tzanetakis & Cook, 2000). Its basic goal is to provide a general, extensible and flexible architecture that allows easy experimentation with algorithms and provides fast performance that is useful for developing real-time audio analysis and synthesis tools. A variety of existing building blocks that form the basis of most published algorithms in Computer Audition are already available as part of the framework and extending the framework with new components/building blocks is straightforward. It has been designed and written by George Tzanetakis with help from students and researchers from around the world. Marsyas has been used for a variety of projects in both academia and industry.

1.3.2 Microsoft Visual Studio – Visual C++

Microsoft Visual Studio[3] is the main Integrated Development Environment (IDE) from Microsoft. It can be used to develop console and

[2] Available at http://marsyas.info/.
[3] Microsoft Visual Studio Developer Center at http://msdn.microsoft.com/en-us/vstudio/default.aspx.

Graphical User Interface (GUI) applications along with Windows Forms applications, web sites, web applications, and web services in both native code as well as managed code for all platforms supported by Microsoft.

Visual C++ is Microsoft's implementation of the C and C++ compiler and associated languages services and specific tools for integration with the Visual Studio IDE. It can compile either in C mode or C++ mode.

1.3.3 MATLAB

MATLAB[4] is a numerical computing environment and programming language. Created by MathWorks, MATLAB allows easy matrix manipulation, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs in other languages.

1.3.4 Lego Mindstorms NXT

Lego Mindstorms NXT[5] is a programmable robotic kit designed by Lego (see Figure 1). It is composed of a brick-shaped computer, named NXT brick, containing a 32-bits microprocessor, flash and RAM (Random Access Memory) memory, a 4 MHz 8-bit microcontroller and a 100x64 LCD (Liquid Crystal Display) monitor. This brick supports up to four sensorial inputs and can control up to three servo-motors. It also provides a proper user interface displayed through the LCD and controlled with its four buttons, and a 16 kHz speaker. Lego NXT supports USB (Universal Serial Bus) 2.0 connection to a personal computer (PC) and supports Bluetooth wireless communication, for remote control and data exchange. It offers many sensing capabilities through its *ad hoc* sensors. In the scope of this project, we configured our robot with a color sensor, to detect and distinguish visible colors, and an ultrasonic sensor, capable of obstacle detection, retrieving the robot's distance to obstacles in inches or centimeters.

---

[4] MATLAB official web site at http://www.mathworks.com/.
[5] For more information consult http://mindstorms.lego.com/eng/default.aspx.

Figure 1 - Lego NXT brick and some of its sensors and servo-motors.

1.3.5 NXT Remote API

The NXT Remote API[6] is a C++ library, designed by Anders Søborg. It enables the remote control of the Lego NXT brick over Bluetooth, using any C++ compiler on a Windows OS (Operating System) machine (the library can be also used with Pocket PCs running Windows Mobile using MS Visual Embedded C++). This API is decomposed in nine classes, which make it possible to:

- Open and close Bluetooth connections with multiple NXT units;
- Control the motors;
- Send and receive messages from the NXT;
- Read sensor values in both mode-dependent and raw;
- Set Brick name, get battery level, read firmware version, etc;
- Control the NXT speaker;
- Play sound files;
- Start and stop on-brick programs;
- Use compass and sonar sensors;
- Communicate with I^2C (Inter-Integrated Circuit) sensors;
- Direct commands for the PCF8591 A/D (Analog/Digital) converter;
- Direct commands for the PCF8574 I/O (Input/Output) Chip.

[6] For more information and download consult http://www.norgesgade14.dk/index.php.

1.3.6 Methodology

In this section we describe our methodology beyond the conception of the proposed framework by interconnecting all present tools towards the proposed goal. This framework was decomposed into three sub-modules, according to their functional nature:

- **Music Analysis Module:** module responsible for the rhythmic analysis of the musical input on-the-fly based on a real-time onset detection function.

- **Robot Control Module:** module responsible for the robot control by combining the multi-modal inputs and deciding on the robot dancing output.

- **Human Control Module:** module responsible for the user interface through which the user has a deterministic role by defining the robot's and the onset function's parameters and all the dance movements to be performed by the robot.

To clarify our method we summarize it through a diagram, depicted in Figure 2. It is decomposed in a series of functional blocks each one developed or executed with a correspondent tool or set of tools, as follows:

- The *Music Analysis Module* was essentially based on Marsyas (v0.23), by combining a set of signal processing blocks to develop a real-time onset detection function. The resultant output, in the form of rhythmic events, is sent to the *Robot Control Module* via UDP/IP (User Datagram Protocol/Internet Protocol) sockets.

- The NXT Remote API (v0.3) was embedded in our *Robot Control Module* to remotely receive sensing (i.e., ultrasonic and color sensors) information from the robot and send motor outputs correspondent to the generated dance movements. The bi-directional communication with the robot was achieved via Bluetooth.

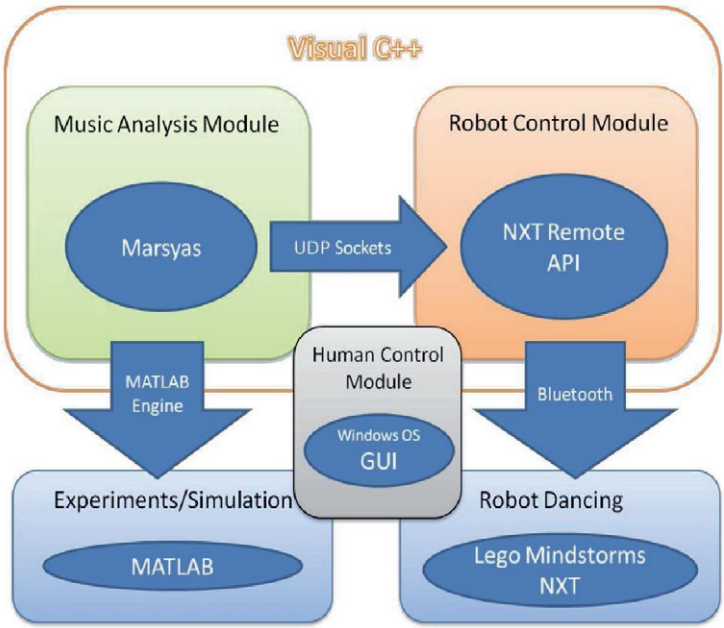


Figure 2 – Block diagram of the proposed methodology.

- Visual Studio (2008) was the foremost used software of our work. Through this IDE we integrated all the existing algorithms, namely Marsyas' onset detection function and the NXT Remote API, and programmed all the remnant code for the implementation of the proposed framework. The framework's user interface, which constituted the *Human Control Module*, was also designed through this IDE by recurring to the Microsoft Foundation Class (MFC) library.

- In this project, MATLAB (R2008a) was used as a plotting platform for aiding the calibration of the implemented *Music Analysis Module*.

- Based on Lego Mindstorms NXT, we built a humanoid-like robot using two NXT bricks that controls six servo motors (one for each leg and each arm, one for a rotating hip and one for the head) and the two

referred sensors: color and ultrasonic sensor. This resulted in the robotic agent tested in the proposed robot dancing framework.

1.4 Book Structure

This book is organized in five chapters, the first of which is this introduction to this work's motivation, aims, outline, methodology and tools. The following Chapter 2 presents some related work from recent literature on the areas of *audio note-onsets detection* and *dancing robots*, and presents some relevant definitions for a comprehensive reading of this book. Chapter 3 describes our implementation's approach, with focus on the designed system architecture. Chapter 4 describes the calibration of the integrated onset detection function, and the setup for live demonstrations of the generated robot dance performance. It ends with an empiric evaluation of the implemented system and an overall discussion of the empiric results after proper statistic analysis. Finally, Chapter 5 concludes this book by summarizing general conclusions and proposing a path for future work.

Chapter 2

Related Work

This chapter presents some related work in all topics of interest to this book. It is divided in two main distinct sections: Section 2.1 – *Audio Note-Onsets Detection* and Section 2.2 – *Dancing Robots*. A third Section 2.3 – *A Step Further* defines our approach in contrast to the formerly reviewed.

2.1 Audio Note-Onsets Detection

Many musical applications require the accurate detection of the onset-times of notes in musical signals. Audio note-onset detection (or simply audio onset detection) is one of the most classic research areas of Music Information Retrieval, and several approaches have been proposed so far. The task of note-onset detection is typically defined as finding the starting time of each musical note in a piece of music, where a musical note is not restricted to those having a clear pitch or harmonic partials. Although this seems a trivial task, in polyphonic music, where a set of notes (i.e., chords) might occur almost simultaneously, the definition of onsets starts to become blurred (Dixon, 2006).

In order to clarify the concept of onset time and introduce the process of its detection for any application, based on (Bello, et al., 2005) we define the concepts of *transients*, *onsets* and *attacks*. Due to the importance of distinguishing the similarities and differences between these key concepts

for an understandable reading of this book, we begin this section by presenting some fundamental considerations to audio note-onset detection.

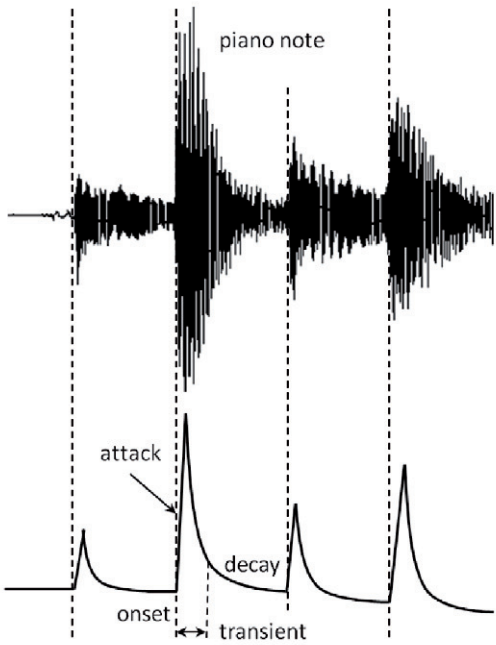


Figure 3 - "Attack", "transient", "decay", and "onset" in a series of piano notes.

Figure 3 shows, in the simple of four piano notes, how one could differentiate these notions. The *attack* of a musical note is the time interval during which the amplitude envelope of the note increases up to its maximum (Bello, et al., 2005). *Transients* are short intervals during which the signal evolves quickly in a relatively unpredictable way after reaching the maximum amplitude of the note (Bello, et al., 2005). In acoustics, the "transient often corresponds to the period during which the excitation is applied and then damped, leaving only the slow decay at the resonance frequencies of the body" (Bello, et al., 2005). The *onset* of a musical note typically matches the starting point of the transient, or the earliest time at which the transient can be reliably detected (Bello, et al., 2005). These are the time-points which note-onset detection algorithms attempt to retrieve. All these considerations assume the time resolution of the human hear of

10 ms, below which it cannot distinguish two simultaneous transients as two individual musical notes.

In Section 2.1.1 – *Basic Definitions and General Architecture of Onset Detection Algorithms* we review the basic concepts behind note-onset detection and introduce a general categorization of onset detection algorithms. In Section 2.1.2 – *Audio Onset Detection Functions: A Literature Review*, we present a review on some of the research made on this area, up to the year of 2007. In Section 2.1.3 – *Results' Analysis and Comparison*, we conclude the review in this area of research by presenting a comparison of the results achieved by some of the most prominent approaches, also up to the year of 2007.

2.1.1 Basic Definitions and General Architecture of Onset Detection Algorithms

As illustrated in Figure 4, onset detection algorithms are normally split into three components (Bello, et al., 2005): the *pre-processing* of the original audio signal for improving the performance of the subsequent stages; the *reduction* of the pre-processed signal, which represents the actual onset detection function as a representation of the changing state of a musical signal, typically at a lower sampling rate; and the *peak-picking* algorithm applied upon the detection function to retrieve the actual onset times. Frequently, the pre-processing stage is ignored in order to simplify the algorithm.

Following (Bello, et al., 2005)'s work, we decomposed this section in a sub-section dedicated to each of these onset detection stages.

1.4.1.1 Note-Onset Detection: Pre-Processing

Pre-processing implies the low-level manipulation of the signal's waveform (e.g., by isolating different frequency bands) in order to accentuate/attenuate various aspects of the signal to be analyzed (Bello, et al., 2005). This is considered an optimization procedure which intimately depends on the application of the implemented note-onset detection method.

Attending to the literature (particularly (Bello, et al., 2005)), this review is decomposed in two processes that appear to be of particular relevance to the pre-processing of onset detection approaches: *multiple bands* and *transient/steady-state separation*.

The pre-processing based on *multi bands* (e.g., by using filter-banks) is normally used to satisfy the needs of specific applications that require the onset detection in individual sub-bands to complement global estimates, as a way of increasing the robustness of a given onset detection method (Bello, et al., 2005).

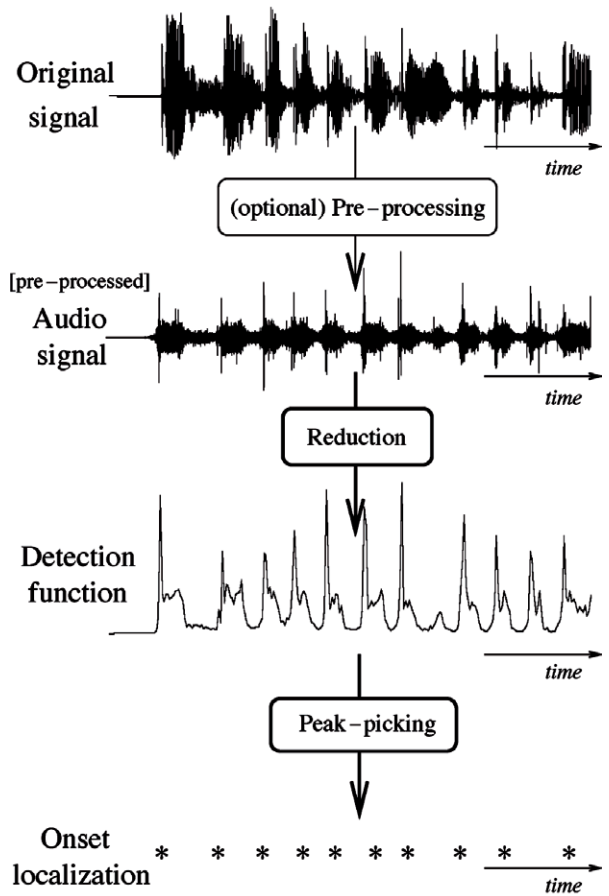


Figure 4 - Flowchart of a standard onset detection algorithm (Bello, et al., 2005).

The process of *transient/steady-state separation* is usually associated with the modeling of music signals, by means of sinusoidal models (e.g., "additive synthesis" and spectral modeling synthesis (SMS)), which represent an audio signal as a sum of sinusoids with slowly varying parameters. These models may also consider the residual of the synthesis method as a Gaussian white noise filtered with a slowly varying low-order filter (Bello, et al., 2005). Due to the irrelevance of the pre-processing stage in the scope of this book a literature review shall not be presented.

2.1.1.1 Note-Onset Detection: Reduction

In the context of note-onset detection, *reduction* consists of the process of transforming the audio signal into a highly sub-sampled function which salients the transients in the original signal or sub-bands of the original signal (after pre-processing) (Bello, et al., 2005). It consists in the basis of the onset detection approach beyond a wide class of onset detection methods, which will be the main focus of our review in this context (see Section 2.1.2).

Based on (Bello, et al., 2005), (Dixon, 2006), and (Duxbury, Bello, Davies, & Sandler, 2003b), we decomposed this reduction analysis in six different classes, according to the nature of the methods (i.e., according to their basis on signal features, of different kinds, or statistics): *Temporal Methods*, *Energy-Based* (*Spectral Weighting*) *Methods*, *Phase-Based Methods*, *Complex Methods*, *Time-Frequency and Time-Scale Methods (TFR)*, and *Statistical Methods*.

- **Temporal Methods:** An onset usually occurs attached to an increase of the signal's amplitude in time (Bello, et al., 2005). Early temporal methods of onset detection used a detection function, *E(n)*, which follows the amplitude envelope of the signal, constructed by low-pass filtering the signal, *x(n)*:

$$E(n) = \frac{1}{N}\sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1}|x(n+m)|\omega(m)\,, \tag{2.1}$$

where *ω(m)* is an *N*-point window or smoothing kernel, centered at *m=0*. Distinctively, one can compute the local energy, rather than the amplitude, by squaring, instead of rectifying, each sample, *n*, of the signal:

$$E(n) = \frac{1}{N}\sum_{m=-\frac{N}{2}}^{\frac{N}{2}-1}[x(n+m)]^2\omega(m)\ . \tag{2.2}$$

This reduction method, without later adjustments, is not usually suitable for reliable onset detection by peak-picking. An improvement, included in some standard onset detection algorithms, is to work with energy derivative along time, in order to locate sudden rises in energy. This process is commonly combined with a pre-processing based on *filter-banks* or *transient/steady-state separation*.

- **Energy-Based (Spectral Weighting) Methods:** The occurrence of a musical note always leads to a burst in the signal's energy, which can be flatter or sharper typically depending on the absence or presence of percussion in the musical piece. Based on this notion, looking into the energy of the signal across time is a straightforward and efficient metric by which to detect certain types of note onsets, especially percussive (Duxbury, Bello, Davies, & Sandler, 2003b). Consider the time-frequency representation of a signal by computing the Short Time Fourier Transform (STFT). By recurring to a finite-length sliding window, *ω(m)*, (e.g., Hamming Window (HW)), it can be calculated as:

$$STFT = X(n,k) = \sum_{m=-\frac{N}{2}}^{\frac{N}{2}-1} x(hn+m)\omega(m)e^{-\frac{2j\pi mk}{N}}\ , \tag{2.3}$$

where *x(mh)* is the time-domain signal, *k = 0, 1,..., N-1* is the index of each frequency bin, and *n* is the frame number. One of the simplest methods to compute an energy-based onset detection

function may be produced by calculating the first derivative of the L_2-norm squared energy of a frame of the signal, *E(m)*, given by:

$$E(m) = \sum_{q=(m-1)h}^{mh} \sum_{k=0}^{N-1} |X(q,k)|^2 , \quad (2.4)$$

where h is the hop size, m the hop number and q is the integration variable. Another variant for computing an energy-based onset detection function would be to calculate the L_1-norm of the difference between magnitude spectra. When restricted to positive changes (to emphasize onsets rather than offsets) and summed across all frequency bins, this results in the onset function commonly known as the *spectral flux* (or *spectral difference*), *SF*, which is given by (Dixon, 2006) and (Masri, 1996):

$$SF(n) = SF_1(n) = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} H(|X(n,k)| - |X(n-1,k)|) , \quad (2.5)$$

where $H(x) = (x + |x|)/2$ is the half-wave rectifier function.

- **Phase-Based Methods:** According to Fourier analysis, a signal can be represented by a group of sinusoidal oscillators with time-varying amplitudes, frequencies and phases, which tend to have stable amplitudes and frequencies during steady-state (Duxbury, Bello, Davies, & Sandler, 2003b). Hence, one can calculate the phase of the k^{th} oscillator at any given time-frame, n, from:

$$\begin{aligned} \varphi_k(n-1) - \varphi_k(n-2) = \varphi_k(n) - \varphi_k(n-1) \Leftrightarrow \\ \Leftrightarrow \Delta\varphi_k(n) = \varphi_k(n) - 2\varphi_k(n-1) + \varphi_k(n-2) \cong 0 , \end{aligned} \quad (2.6)$$

where $\varphi_k(n)$ represents the k^{th} frequency bin of the n^{th} time-frame from the STFT of the audio signal (i.e., the signal's phase unwrapping). This implies that the *phase deviation*, *PD*, between the target and real phase values can be calculated as (Bello & Sandler, 2003) and (Duxbury, Bello, Davies, & Sandler, 2003b):

$$PD = d_{\varphi} = princarg[\Delta\varphi_k(n)] \,, \tag{2.7}$$

where the *princarg* operator maps the $\Delta\varphi_k(n)$ angle to the $[-\pi, \pi]$ range. Hence, d_{φ} will tend to zero if the phase value is accurately predicted and will deviate from zero otherwise, which is the case for most oscillators during attack transients.

- **Complex Methods:** Considering that energy-based onset detection function favor strong percussive onsets while phase-based approaches emphasize soft onsets, one might combine phase and energy information, both more reliable at opposite ends of the frequency axis, into onset detection functions of complex nature (Bello, Duxbury, Davies, & Sandler, 2004).

  For computing a complex-domain onset detection function one might consider a simpler measure of the spread of the distribution calculated as the mean absolute phase deviation:

$$\varsigma_p(n) = \frac{1}{N}\sum_{k=1}^{N}|\Delta\varphi_k(n)| \,. \tag{2.8}$$

Yet, this function is susceptible to phase distortion and to noise introduced by the phases of components with no significant energy (Bello, et al., 2005). As an alternative, (Bello, Duxbury, Davies, & Sandler, 2004) introduced an approach that works with Fourier coefficients in the complex domain, summed across the frequency-domain to generate the following *complex difference*, *CD**, onset detection function:

$$CD^*(n) = \sum_{k=1}^{N} \Gamma_k(n) \,, \tag{2.9}$$

where the k^{th} spectral bin is quantified by calculating the Euclidean distance, $\Gamma_k(n)$, between the observed $X_k(n)$ and that predicted by the previous frame, $\hat{X}_k(n)$:

$$\Gamma_k(n) = \left\{ \left|\hat{X}_k(n)\right|^2 + |X_k(n)^2| - 2\left|\hat{X}_k(n) X_k(n) \cos\left(\Delta\varphi_k(n)\right)\right| \right\}^{\frac{1}{2}} . \quad (2.10)$$

This expression is derived from the following complex considerations:

$$\begin{cases} \hat{S}_k(n) = \hat{X}_k(n) e^{j\hat{\phi}_k(n)} \\ S_k(n) = X_k(n) e^{\phi_k(n)} \end{cases} , \quad (2.11)$$

where $\hat{\phi}_k(n) = princarg[2\varphi_k(n-1) - \varphi_k(n-2)]$. (2.12)

In a simpler equivalent way, (Dixon, 2006) defined his own *CD* function as:

$$CD(n) = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} |X(n,k) - X_T(n,k)| , \quad (2.13)$$

where the target value, $X_T(n, k)$, is estimated by assuming constant amplitude and rate of phase change:

$$X_T(n,k) = |X(n-1,k)| e^{\psi(n-1,k)+\psi'(n-1,k)} . \quad (2.14)$$

- **Time-Frequency and Time-Scale Methods:** Alternatively to the former methods, one may also compute onset detection functions based on time-scale or time-frequency representations. The most notable algorithm based on this approach is the *wavelet regularity modulus*, *WRM*, which is a local measure of the regularity of the signal given by (Daudet, 2001):

$$WRM = \sum_{(j,k)\in\vec{o}[i]} 2^{js} d_{j,k} , \quad (2.15)$$

where $d_{j,k}$ are the wavelet coefficients, $\beta[i]$ is the full branch leading to a given small-scale coefficient, $d_{1,i}$, and s is a free parameter used to emphasize certain scales. This model reveals to be an effective onset

detection function, where increases of WRM represents the existence of large, transient-like coefficients in the branch, *β[i]*, (Bello, et al., 2005).

A more detailed review on this type of approach is presented in Section 2.1.2.

- **Statistical Methods:** Onset detection based on statistical methods assumes that the signal can be described by some model of probability (Bello, et al., 2005). This type of approach looks for abrupt changes in the signal and registers their likely times in a probabilistic manner, by recurring to likelihood measures or Bayesian model selection criteria.

  Based on (Bello, et al., 2005), probabilistic approaches can be decomposed in *model-based change point detection* methods (e.g., (Jehan, 1997)), through which change points are detected when the given likelihood ratio surpasses a fixed threshold (looking for an instantaneous switch between two distinct models); and in approaches based on "surprise signals" (e.g., (Abdallah & Plumbley, 2003)), which look for surprising moments relative to a single global model. The latter are based on a detection function that uses a global ICA (Independent Component Analysis) model to trace of the *negative log-likelihood*, *NL*, of the signal given its recent history:

$$NL = -\log p(\mathrm{x}) = -\sum_{i=1}^{N} \log p_i(s_i) + \log|\mathrm{A}|\,, \tag{2.16}$$

  where s is obtained from x using $\mathrm{s} = \mathrm{A}^{-1}\mathrm{x}$, $p_i(\cdot)$ is the assumed or estimated probability density function of the i^{th} component of s, and $|\mathrm{A}|$ is the determinant of an N x N basis matrix, A.

  A more detailed review on this type of approach is presented in Section 2.1.2.

2.1.1.2 Note-Onset Detection: Peak-Picking

The actual timings of the note-onsets are inferred from the onset detection function by finding local maxima in this function using *peak-picking* algorithms, typically based in thresholding approaches subject to a set of constraints. This thresholding should be carefully defined due to its high impact on the ultimate results, specifically on the ratio of false

positives (reported detections where no onset exists) to false negatives (missed detections) (Dixon, 2006). Following (Bello, et al., 2005), we decomposed this process into two consecutive procedures: *thresholding* followed by *peak-picking*.

- **Thresholding:** The definition of proper thresholds should take into consideration the effective separation of event-related peaks from non-event-related ones. This definition should also be intimately dependent on the application in respect to the undesirability of false positives and/or false negatives (Dixon, 2006). Following (Bello, et al., 2005), one may decompose thresholding approaches into two main classes: *fixed thresholding* and *adaptive thresholding*.

  Fixed thresholding defines onsets as peaks where the detection function, *d(n)*, exceeds the threshold, i.e., $d(n) \geq \delta$, where δ is a positive constant. Although very simple, this approach is inefficient in the presence of dynamic music signals, tending to miss onsets, generally in quiet passages, while over-detecting during the loud ones (Bello, et al., 2005). This invokes the use of a signal *adaptive threshold*, $\delta[n]$, generally computed as a smoothed version of the detection function. This smoothing can be linear, e.g., using a low-pass FIR-filter:

$$\delta[n] = \delta + \sum_{i=0}^{M} a_i d(n-i) \,, \tag{2.17}$$

  with $a_0=1$; or non-linear, e.g., using the square of the detection function itself:

$$\delta[n] = \delta + \lambda \sum_{i=-M}^{M} \omega_i d^2(n+i) \,, \tag{2.18}$$

  where λ is a positive constant and $\{\omega_i\}_{i=-M\ldots M}$ is a smooth window.

  Alternatively, in order to reduce the fluctuations, due to the presence of large peaks, the thresholding can be defined in percentiles, based, for instance, in the local median (Bello, et al., 2005):

$$\delta[n] = \delta + \lambda \,\text{median}\{|d(n-M)|, \ldots, |d(n+M)|\} \,. \tag{2.19}$$

  Similarly to the former, (Dixon, 2006) used an *adaptive thresholding* function defined as:

$$\delta[n] = \max\left(d(n), \delta * \delta[n-1] + (1-\delta)d(n)\right). \quad (2.20)$$

- **Peak-Picking:** Based on this decomposition, peak-picking can be reduced to identifying local maxima above the defined threshold (Bello, et al., 2005). As a representative example we will describe (Dixon, 2006)'s approach where each onset detection function, *d(n)*, is normalized to have a mean of 0 and a standard deviation of 1. In his scheme, a peak at time $t = \frac{nh}{r}$ is considered an onset if it fulfills the following three conditions:

$$\begin{aligned} &1)\ d(n) \geq d(k)\,, \ \text{for all k such that } n-\omega \leq k \leq n+\omega \\ &2)\ d(n) \geq \frac{\sum_{k=n-m\omega}^{n+\omega} d(k)}{m\omega+\omega+1} + \delta \\ &3)\ d(n) \geq \delta[n] \end{aligned} \quad , \quad (2.21)$$

where $\omega=3$ is the size of the window used to find a local maximum, $m = 3$ is a multiplier so that the mean is calculated over a larger range before the peak, δ is the threshold above the local mean which an onset must reach, and $\delta[n]$ is the used threshold function, given by eq. (2.20).

For a review of a number of peak-picking algorithms for audio signals see (Kauppinen, 2002).

2.1.2 Audio Onset Detection Functions: A Literature Review

Earlier algorithms developed for onset detection focused mainly on the variation of the signal energy envelope in the time domain (Lacoste & Eck, 2007).

Based on the (instantaneous short-term) spectral structure of the signal, Masri (Masri, 1996) proposed a *high frequency content* (HFC) function with a linear frequency dependent weighting, $W_k = |k|$, which linearly weights each bin's, k, contribution in proportion to its frequency, and is given by:

$$HFC = \frac{1}{N}\sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} W_k |X_k(n)|^2 \,. \quad (2.22)$$

The *HFC* function produces sharp peaks during attack transients and is notably successful when faced with percussive onsets, where transients are well modeled as bursts of white noise (Bello, et al., 2005). In a more general approach, based on changes in the spectrum, Masri formulated the detection function as a "distance" between successive short-term Fourier spectra, treating them as points in an *N*-dimensional space. Based in this criterion, he developed the *spectral flux*, *SF*, onset detection method, which calculates the spectral difference using the L_1-norm of the difference between magnitude spectra (see eq. (2.5)).

Later, (Duxbury, Sandler, & Davies, 2002) used a pre-processing scheme based on a constant-Q conjugate quadrature filter-bank to separate the signal into five sub-bands. By using this approach, the authors developed a hybrid approach that considers energy changes in high-frequency bands and spectral changes (i.e., *spectral flux*) in lower bands. In order to calculate this *spectral flux*, the authors proposed the use of the L_2-norm of the rectified difference:

$$SF_2(n) = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} \{H(|X(n,k)| - |X(n-1,k)|)\}^2 \,, \tag{2.23}$$

where $H(x) = (x + |x|)/2$ is a half-wave rectifier. This rectification has the effect of counting only those frequencies where there is an increase in energy, and is intended to emphasize onsets rather than offsets. By implementing a multiple-band scheme, the approach effectively avoids the constraints imposed by the use of a single reduction method, while having different time resolutions for different frequency bands.

(Scheirer, 1998) demonstrated that much information from the signal can be discarded while still retaining its rhythmical aspect. On a set of test musical pieces, Scheirer filtered out different frequency bands using a filter-bank as pre-processing. He extracted the energy envelope for each of those bands using rectification and smoothing. Finally, with the same filter-bank, he modulated a noisy signal with each of those envelopes and merged everything by summation. With this approach, rhythmical information was retained. Yet, in another experiment he showed that if the envelopes are

summed before modulating the noise, a significant amount of information about the rhythmical structure is lost. This experiment alerted for the care needed when discarding signal content for not losing any of its rhythmic information.

(Klapuri, 1999) used the psychoacoustical model developed by Scheirer to develop a robust onset detector, propounding the difference of the log spectral power in bands as a more psychoacoustically relevant feature related to the discrimination of intensity (and simulating the ear's perception of loudness). Hence, to get better frequency resolution, he employed a pre-processing consisting of a filter-bank of 21 filters. The author points out that the smallest detectable change in intensity is proportional to the intensity of the signal, which means that $\Delta I/I$ is a constant, where I is the signal's intensity. Therefore, instead of using $(d/dt)A$ where A is the amplitude of the envelope, he used $\frac{1}{A}\left(\frac{d}{dt}A\right) = \frac{d}{dt}\log(A)$. This provided more stable onset peaks and allowed lower intensity onsets to be detected. Later, (Klapuri, Eronen, & Astola, 2006) used the same kind of pre-processing and won the ISMIR (International Symposium on Music Information Retrieval) 2004 tempo induction contest[7].

(Jehan, 1997), also motivated by psychoacoustical factors, formed an onset detection function by taking power in Bark bands and applying a spectral masking correction based on spreading functions, familiar from the perceptual coding of audio, and post masking with half cosine convolution.

In contrast to Scheirer's and Klapuri's approaches, (Bello & Sandler, 2003) took advantage of phase information (i.e., *phase deviation*) to track the onset of a note. They found that, at steady state, oscillators tend to have predictable phase. This is not the case at the time of an onset, which allowed the decrease in predictability to be used as an indication of note onset. To measure this effect, they collected statistics about the phase acceleration, as estimated by eq. (2.7). To detect the onset, different statistics were calculated across the range of frequencies, including mean,

[7] For more information consult http://www.ismir.net/.

variance, and kurtosis. These provided an onset trace, which could be analyzed by standard peak-picking algorithms, equally considering all frequency bins, *k*.

Advocating that the energy of the signal is concentrated within the bins which contain the partials of the currently sounding tones, (Dixon, 2006) developed an improvement in the former phase-based detection function, by proposing weighting the frequency bins, *k*, by their magnitude. Based on this rationale, Dixon built a new onset detection function, called the *weighted phase deviation* (WPD), and defined by:

$$WPD(n) = \frac{1}{N}\sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} |X(n,k)\varphi\,''(n,k)| \,. \tag{2.24}$$

This is similar to the complex functions (analyzed below), in which the magnitude and phase are considered jointly, but with a different manner of combination. The author further proposed another option to define a weighted phase deviation function, but in a normalized way (NWPD), where the sum of the weights is factored out to give a weighted average phase deviation:

$$NWPD(n) = \frac{\sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} |X(n,k)\varphi\,''(n,k)|}{\sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} |X(n,k)|} \,. \tag{2.25}$$

Subsequently, Bello *et al.* presented three studies, (Bello, Duxbury, Davies, & Sandler, 2004), (Duxbury, Bello, Davies, & Sandler, 2003a), and (Duxbury, Bello, Davies, & Sandler, 2003b), on the combined use of energy and phase information for the detection of onsets in musical signals, in the form of complex domain methods, and developed a set of *complex domain*, *CD*, algorithms. They showed that by combining phase and energy approaches it is possible to achieve a more robust onset detection function, enjoying from both performances: energy-based onset detection functions perform well for pitched and non-pitched music with significant percussive content, while phase-based approaches provide better results for strongly

pitched signals and are less robust to distortions in the frequency content and to noise. This was corroborated by the experimental results achieved for a large range of audio signals.

(Dixon, 2006) has also proposed an improvement to these complex methods (specifically given by eq. (2.9)), by trying to resolve their absence in distinguishing increases from decreases in the amplitude of the signal (i.e., onsets from offsets). They formulated such resolution in the form of a *rectified complex domain*, *RCD*, function, by applying a half-wave rectification to a spectral flux-based function, which grants exclusive consideration to increases in energy in spectral bins. Therefore, the *RCD* is basically the incorporation of this rectification in a *CD* method, being described as follows:

$$RCD(n) = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}} RCD(n,k)\ , \tag{2.26}$$

where

$$RCD(n,k) = \begin{cases} |X(n,k) - X_T(n,k), & \text{if} |X(n,k)| \geq |X(n-1,k)| \\ 0, & \text{otherwise} \end{cases}. \tag{2.27}$$

By rather approaching a time-frequency/time-scale analysis, (Daudet, 2001) developed a transient detection based on a simple dyadic wavelet decomposition of the residual signal. This transform, using the Haar wavelet[8], was chosen for its simplicity and its good time localization at small scales. The method takes advantage of the correlations across scales of the coefficients. The significance of full-size branches of coefficients, from the largest to the smallest scale, can be quantified by a regularity modulus, which is a local measure of the regularity of the signal (see eq. (2.15)).

[8] The Haar wavelet is the first known wavelet and was proposed in 1909 by Alfréd Haar. It is the simplest possible wavelet, but has the advantage of not being continuous and therefore not differentiable.

By considering probability models instead, we shall again refer the work of (Jehan, 1997), which also developed a statistical scheme based on the comparison between two auto-regressive models of the signal, forming a model-based change point detection method. Like other similar approaches, he uses two parametrical Gaussian statistical models, $\mathcal{A}$, $\mathcal{B}$, where their log-likelihood ratio, s, is defined as:

$$s = log \frac{p_{\mathcal{B}}(x)}{p_{\mathcal{A}}(x)}. \tag{2.28}$$

This function assumes that s will change sign at some unknown time, due to the signal convergence from model $\mathcal{A}$ to model $\mathcal{B}$. In Jehan's approach, both models' parameters and the signal's change points, s, are then optimized to maximize the log-likelihood ratio between the probability of having a change at these points and the probability of not having an onset at all. Change points are detected when this likelihood ratio surpasses a fixed threshold.

In a distinct statistic way, based on "surprise signals", (Abdallah & Plumbley, 2003) introduced the *negative log-likelihood using an ICA* model scheme, given by eq. (2.16). Their approach was developed on the notion of an observer which builds a model of a certain class of signals, such that it is able to make predictions about the likely evolution of the signal as it unfolds in time. Such an observer will be relatively surprised at the onset of a note because of its uncertainty about when and what type of event will occur next. However, if the observer is in fact reasonably familiar with typical events (i.e., the model is accurate), that surprise will be localized to the transient region, during which the identity of the event is becoming established. Thus, a dynamically evolving measure of surprise, or onset (novelty), can be used as a detection function.

Ultimately, some recent models, despite being rare, approach onset detection as a problem of supervised (machine) learning. (Davy & Godsill, 2002), also based on time-frequency and time-scale analysis, developed an

audio segmentation algorithm using a Support Vector Machine (SVM)[9] that classify spectrogram frames into being probable onsets or not. The SVM was used to find a hypersurface delimiting the probable zone from the less probable one. Unfortunately, no clear tests were made to outline the performance of the model.

(Kapanci & Pfeffer, 2004) also used SVM on a set of frame features to estimate if there is an onset between two selected frames. By using this function in a hierarchical structure, they were able to find the position of onsets. Their approach was mainly focused on finding onsets in signals with slowly varying changes over time, such as solo singing.

(Marolt, Kavcic, & Privosnik, 2002) used a neural network approach for note onset detection. Their model used the same kind of pre-processing as Scheirer's in (Scheirer, 1998), with a filter-bank of 22 filters. An integrate-and-fire network was then applied separately to the 22 envelopes. Finally, a multi-layer perception was applied on the output to accept or reject the onsets. Results were good but the model was only applied to mono-timbral piano music.

In a similar approach, regarding the use of neural networks, (Lacoste & Eck, 2005) and (Lacoste & Eck, 2007) most notably developed two onset detection algorithms that have participated in the MIREX (Music Information Retrieval Evaluation eXchange) 2005 Audio Onset Detection contest[10], yielding the best and second best performance. Both proposed algorithms first classify frames of a spectrogram into onset or non-onset by using a feed-forward neural network. From the classification of each frame, their model extracts the onset times by recurring to a simple peak-picking algorithm based on a moving average. The first version of their algorithm, SINGLE-NET, is comprised of a time-space transform (spectrogram) which is in turn treated with a feed-forward neural network (FNN), and the resulting trace is fed into a peak-picking algorithm to find onset times. The second version of their algorithm, MULTI-NET, repeats the SINGLE-NET

[9] SVMs are a set of related supervised learning methods used for classification and regression. They belong to a family of generalized linear classifiers. A tutorial on SVM has been produced by C.J.C Burges, at http://research.microsoft.com/~cburges/papers/SVMTutorial.pdf.

[10] For additional information consult the results of the Audio Onset Detection task at MIREX 2005 in http://www.music-ir.org/mirex/wiki/2005:Audio_Onset_Detection_Results.

variant multiple times, by applying different hyper-parameters. A tempo detection algorithm is run on each of the resulting FNN outputs, and the SINGLE-NET and tempo-detection outputs are then combined using a second neural network. With this work, they concluded that a supervised learning approach to note onset detection performs well and warrants further investigation.

With so many possible algorithms for the detection of musical note-onsets in an audio signal now published, which some are referred above, research questions are turning to the comparative evaluation of such methods (Collins, 2005). Therefore, below we present some approaches for the comparative evaluation of onset detection methods.

Bello *et al.*, in (Bello, et al., 2005), presented a tutorial on onset detection in music signals where they reviewed, categorized, and compared some of the most commonly used techniques for onset detection, also presenting possible enhancements, and providing some guidelines for choosing the appropriate method for a given application. The discussed methods were based on several predefined signal features, namely the signal's amplitude envelope (wavelet methods), spectral magnitudes and phases (spectral and phase-based methods), time-frequency representations (temporal methods); and methods based on probabilistic signal models (statistical methods). The achieved results are discussed in the following Section 2.1.3.

Based on (Bello, et al., 2005) work, Collins, in (Collins, 2005), and then Dixon, in (Dixon, 2006), sought to extend their results and reviewed them, as a benchmark for comparison.

Collins, besides reviewing and extending their work, also explored the potential of psychoacoustically motivated models such as those of (Klapuri, 1999) and (Jehan, 1997), referred above. In this research, the author investigated 16 detection functions, including a number of novel and recently published models.

(Dixon, 2006) complemented and extended (Bello, et al., 2005)'s work by introducing new onset detection functions, already referred, and by testing the new methods alongside independent implementations of a subset

of the published methods on the same data set and on a second data set which is two orders of magnitude larger. They restricted their comparison to methods based on short term spectral coefficients, which are the most widely used methods, and the most successful according to the MIREX 2005 Audio Onset Detection evaluation. The achieved results are also discussed in the following Section 2.1.3, in comparison to the previous ones.

2.1.3 Results' Analysis and Comparison

In signal processing, onset detection is an active research area leading to worldwide contests as the Audio Onset Detection contest featured by the MIREX annual competition — an ISMIR member since 2005. These contests aim to find the more efficient onset detection function at retrieving the time locations at which all musical events in a recording begin. We begin this subsection by presenting the main difficulties beyond the evaluation of note-onset detection models and then introducing some methods to overcome this difficulty with the use of tools for building a set of reference onset times. Next, based on (Bello, et al., 2005) and (Dixon, 2006), we present a benchmarking comparison between some of the referred reduction models (see Section 2.1.1.2), and emphasize the ones with best performance. We finish this comparison by introducing some guidelines for choosing the right onset detection function according to its application.

2.1.3.1 Methodology and Tools for the Evaluation of Automatic Onset Detection Algorithms

The main difficulty with the evaluation of any MIR task is that of obtaining large data sets where the ground truth is manually annotated by human expert musicians. The case of annotating the precise timings of onset locations by hand is a laborious and error-prone task (Dixon, 2006).

A second problem consists of robustly comparing the results obtained from different algorithms according to their accuracy on finding the correct onset times, within a given tolerance window (typically of 50ms) (Dixon, 2006). To solve this problem, MIR researchers typically measure the accuracy of note-onset detection algorithms using the *F*-measure, often

graphically depicted through a Receiver Operating Characteristic (ROC) curve, by combining *precision*, *P*, and *recall*, *R*, into a single value representing the optimal point in the ROC curve, as follows:

$$P = \frac{c}{c+f^{+}} \ , \tag{2.29}$$

$$R = \frac{c}{c+f^{-}} \ , \tag{2.30}$$

$$F = \frac{2PR}{P+R} = \frac{2c}{2c+f^{+}+f^{-}} \ , \tag{2.31}$$

where c is the number of correct detections, f^{+} is the number of false positives and f^{-} is the number of false negatives.

A third, and last, problem that raises when evaluating and comparing onset detection algorithms is related to the consideration of a single or multiple onsets when these are played very close together (e.g., when a chord is played on a piano or guitar). The MIREX Audio Onset Detection evaluation addresses this issue by complementing the standard counts of correct detections, false positives and false negatives, with counts of merged onsets (i.e., two onsets detected as a single onset) and double onsets (i.e., a single onset recognized as two) (Downie, West, Ehmann, & Vincent, 2005).

By addressing these issues, we describe two distinct frameworks to improve the performance of the evaluation of algorithms for automatic note onset detection in music signals.

(Leveau, Daudet, & Richard, 2004) developed a carefully designed software tool, called SOL (Sound Onset Labellizer), which combines the three most used hand-label methods (i.e., signal plot, spectrogram, and listening to signal slices), to provide an user-interface to construct the set of reference onset times and cross-validate it amongst different expert listeners. With this application, the authors have objected to build a common methodology and a common annotation tool, which in turn can be used to build a common database of onset-annotated files. In order to be

shared by the widest community they freely disposed this software and files online[11].

In order to enable, coordinate and evaluate submissions to MIREX, a software framework was developed by (Downie, West, Ehmann, & Vincent, 2005) in association with the IMIRSEL team. Their final solution is based in the Data-to-Knowledge (D2K) Toolkit and is included as part of the Music-to-Knowledge (M2K) Toolkit[12] (both implemented in JAVA). M2K modules are connected by an itinerary based in XML (Extensible Markup Language) which describes the particular process flow for each evaluation task. These frameworks are flexibly customizable to suit the specific topologies of all participants' submissions. Hence, they represent a significant advance over traditional evaluation frameworks in general MIR.

2.1.3.2 Benchmarking

Following the previous analysis, in this subsection we present (Bello, et al., 2005)'s and (Dixon, 2006)'s experimental results that compare the most relevant onset detection approaches, described in Section 2.1.2.

To test every relevant scheme, they have both used the same mono data set of 44.1 KHz and 16 bit sound files, with reference onsets marked up by hand by a single expert. The tests were composed of 4 sets of short excerpts from a range of instruments, classed into the following groups (Bello, et al., 2005):

- NP — non-pitched percussion, such as drums (119 onsets);
- PP — pitched percussion, such as piano and guitar (577 onsets);
- PN — pitched non-percussion, in this case solo violin (93 onsets);
- CM — complex mixtures from popular and jazz music (271 onsets).

In (Bello, et al., 2005), the onset labeling was accomplished mostly by hand, which is a lengthy and inaccurate process, especially for complex recordings such as pop music, typically including voice, multiple

[11] Available at http://perso.telecom-paristech.fr/~grichard/ISMIR04/.
[12] M2K is an open-source initiative and is freely available from http://music-ir.org/evaluation/m2k.

instruments and post-production effects. A small subsection of the database corresponds to acoustic recordings of piano music, generated through MIDI (Musical Instrument Digital Interface), which removes the error introduced by hand-labeling. In a way to allow for its inaccuracy, the authors considered correct matches the ones which match the annotated onsets within a 50ms window. (Bello, et al., 2005, p. 11~Table I) presents the achieved results for all compared methods, according to the characteristics of each: Spectral Weighting Methods — *Spectral Flux* (*SF*) (Masri, 1996), and *High-Frequency Content* (*HFC*) (Masri, 1996), given by eq. (2.5); Phase-Based Methods — *Phase Deviation* (*PD*) (Duxbury, Bello, Davies, & Sandler, 2003a), given by eq. (2.7); Time-Frequency and Time-Scale Methods — *Wavelet Regularity Modulus* (*WRM*) (Daudet, 2001), given by eq. (2.15); and Statistical Methods — *Negative Log-Likelihood* (*NL*) (Abdallah & Plumbley, 2003), given by eq. (2.16). For the sake of a fair comparison between the detection functions, the authors opted to use a common post-processing and peak-picking technique. Peak-picking was accomplished using the moving-median adaptive threshold method, based on (Rodet & Jaillet, 2001). However, the performance for each detection function could be improved by fine tuning the peak-picking algorithm for each specific onset detection function. (Bello, et al., 2005, p. 10~Figure 7) presents the results achieved in (Bello, et al., 2005, p. 11~Table I) in the form of a ROC curve, comparing the performance of each tested scheme. To compose this curve, all peak-picking parameters (e.g., filter's cutoff frequency, λ) were held constant, except for the threshold, δ, which was varied to trace out the performance curve. Better performance is indicated by a shift of the curve upwards and to the left. The optimal point on a particular curve can be defined as the closest point to the top-left corner of the axes, where the error is at its minimum.

An overview of every analyzed method when applied to different types of audio signals (i.e., violin, piano, and pop music) can be observed in (Bello, et al., 2005, pp. 7-8~Figure 4, Figure 5, Figure 6).

Following (Bello, et al., 2005)'s considerations, by reading the different optimal points we can retrieve the best set of results for each onset detection method. In overall, the *negative log-likelihood* performed the

best, with a mean accuracy of 90.6±4.7%, followed by the *HFC*, with a mean 90.0±7.0%, the *spectral flux*, with a mean 83.0±4.1%, the *phase deviation*, with a mean 81.8±5.6%, and, finally, the *wavelet regularity modulus*, with a mean 79.9±8.3%.

Based on (Bello, et al., 2005), one might also withdraw additional conclusions by analyzing the shape of each curve, as it contains useful information about the properties of each method. The *negative log-likelihood* seems the most appealing for most applications by remaining closer to the top-left corner of the ROC curve despite the class of instruments, while producing little noise. The *HFC* was able to identify most (95%) of the annotated onsets in all excerpts while only falsely identifying 10% of noise peaks as onsets (i.e., 10% of false positives). The *wavelet regularity modulus* revealed similar performance but it seems more prone to false positives. Finally, the *spectral flux* and the *phase deviation*, as temporal-domain methods, seemed to deliver a smoother onset detection function, which minimized noise that would potentially result in false positive detections.

According to (Bello, et al., 2005, p. 11~Table I), and also referring (Bello, et al., 2005)'s discussion, we present a performance analysis depending on the type and quality of the input signal. Notably, the *negative log-likelihood* was the best performing onset detection function, working generally well for all classes of instruments in the data sets. The *HFC* seems to be best suited for highly percussive sounds and complex mixtures, in the presence of drums (i.e., percussion). On the other hand, the *phase deviation* seems best suited for pitched sounds (both PP and NP), richer in tonal information, while being poor at analyzing percussive sounds and complex mixtures. The *wavelet regularity modulus* was only effective when dealing with simple percussive sounds. Ultimately, the *spectral flux* seems moderate for all classes of instruments, slightly under-performing *phase deviation* for pitched sounds and *HFC* for more percussive and complex sounds.

Following these results, we present Dixon's analysis, from (Dixon, 2006). In his experiments, the ground-truth data was used in advance to select optimal values of α (positive thresholding constant) and δ (the

threshold above the local mean which an onset must reach). Similarly to (Bello, et al., 2005), Dixon considered an onset to be correctly detected if it matches a ground-truth onset time within a 50 ms tolerance window. However, the author did not penalize merged onsets, considering that the analyzed data contained many simultaneous or almost simultaneous notes (and the author was not attempting to recognize the type of notes). Also contrarily to (Bello, et al., 2005), Dixon, in (Dixon, 2006, p. 5~Table 1), presented the comparative results in function of their *precision*, *recall*, and *F*-measure, as respectively given by eq. (2.29), eq. (2.30), and eq. (2.31). All methods' parameters were chosen to maximize *F*.

In this context, and in order to compare his results with (Bello, et al., 2005)'s, Dixon re-tested the *spectral flux* (*SF*), the *phase deviation* (*PD*), and also tested his proposed models, *WPD*, *NWPD*, and *RCD*, in contrast to their former versions compiled in (Bello, et al., 2005) (respectively, *PD*, *PD*, and *CD*).

Therefore, (Dixon, 2006, p. 5~Table 1) presents his own comparative results, based on the same data sets used in (Bello, et al., 2005), for 8 different onset detection functions: Energy-Based Methods — *Spectral Flux* (*SF** from (Bello, et al., 2005) and *SF* from (Dixon, 2006)); Phase-Based Methods — *Phase Deviation* (*PD** from (Bello, et al., 2005) and *PD* from (Dixon, 2006)); *Weighted Phase Deviation* (*WPD*), from (Dixon, 2006); *Normalized Weighted Phase Deviation* (*NWPD*), from (Dixon, 2006); Complex Methods — *Complex Domain* (*CD*), from (Bello, Duxbury, Davies, & Sandler, 2004) and (Duxbury, Bello, Davies, & Sandler, 2003b); and *Rectified Complex Domain* (*RCD*), from (Dixon, 2006).

By analyzing (Dixon, 2006, p. 5~Table 1), it is observable that there are some large discrepancies between Bello's results (marked with * — *SF** and *PD**) and Dixon's own implementations of the same functions (i.e., *SF* and *PD*).

Referring (Dixon, 2006)'s analysis, and akin to (Bello, et al., 2005)'s results, the *SF** and *SF* revealed better performance on percussive sounds (i.e., on PP data) than on the PN and NP data sets. Yet, Dixon's *SF* showed much smaller performance differences among the 3 tested data sets than

Bello's *SF**, although this difference may be justified by the use of a more uniform peak-picking function. Greater differences were even evident between Dixon's *PD* and Bello's *PD** onset detection functions. Contrarily to *PD*, *PD** achieved overall results closer to *WPD* and *NWPD*. This fact was latter justified by (Bello, et al., 2005) to be the result of applying a weighting scheme to *PD**, which was not applied to *PD*. Nevertheless, the *WPD* and *NWPD* functions revealed significant improvements over the *PD* function, although no conclusions could be withdrawn on the overall improvement of applying normalization on the WPD function (i.e., by using the *NWPD* function), since the *NWPD* was slightly better for the PP and CM data but slightly worse for the remaining data. Finally, the *RCD* function slightly outperformed *CD* in the overall, although the difference was not significant.

In summary, some of Dixon's results (Dixon, 2006) contradicted Bello's (Bello, et al., 2005) and suggested that similar performance might be obtained, in the overall, with a magnitude-based (e.g., *spectral flux*), a phase-based (e.g., *weighted phase deviation*) or a complex domain (e.g., *complex difference*) onset detection function. Yet, since the tested data sets were small and not sufficiently general, Dixon did not draw further conclusions about the differences between these methods, except to state that *spectral flux* has the advantage of being the simplest and fastest algorithm.

As conclusion, one should refer that all the discussed results (from (Bello, et al., 2005) and (Dixon, 2006)), while depicting a general trend in the behavior of the tested approaches, are not absolute due to the signal dependencies of the methods, and to the chosen peak-picking and post-processing algorithms. The hand-labeling of the ground truth onsets used in the evaluation could also be ambiguous and subjective, especially in complex mixtures, which might have slightly compromised the results on this type of data.

2.1.3.3 Onset Detection Functions: Comparative Analysis and Application of the Methods

When picking the most accurate method for a specific application the general rule of thumb is that one should choose the method with minimal

complexity that satisfies the requirements of that application, in a balance of complexity between pre-processing, construction of the detection function, and peak-picking (Bello, et al., 2005).

What follows is a summary discussion of the merits of different reduction approaches and some guidelines to find the appropriate method for a specific application, with an emphasis on the ones that have been previously compared, founded mainly on (Bello, et al., 2005)'s and (Dixon, 2006)'s results.

We decomposed this general discussion in six different methods, according to Section 2.1.1.2:

- **Temporal Methods (e.g., (Klapuri, 1999)):** these reveal the highest simplicity in a computational perspective. However, they depend on the existence of clearly identifiable amplitude increases, which are only present in highly percussive events in simple audio signals. Either way, temporal-based onset detection functions, relying on the amplitude of the signal, are typically inefficient when facing amplitude modulations (i.e., vibrato, tremolo) or the overlapping of energy produced by simultaneous sounds. Ultimately, these methods present low precision in the time-localization of the detected onsets.

  Bello *et al.* consider these methods especially adequate to very percussive (PP, e.g., drums) musical signals (Bello, et al., 2005).

- **Energy-Based (Spectral Weighting) Methods (e.g., (Masri, 1996), (Duxbury, Sandler, & Davies, 2002)):** from these we shall refer the commonly used *HFC* (Masri, 1996). It is particularly efficient when applied to percussive signals but less robust when facing low-pitched and non-percussive events, due to the occurrence of energy changes at low frequencies, which are then de-emphasized by the weighting. This problem can be overcome by using *spectral difference* (*spectral flux*) methods such as the L_1-norm of the difference between magnitude spectra, given by eq. (2.5), (Masri, 1996), or the L_2-norm of the rectified spectral difference, given by eq. (2.23), (Duxbury, Sandler, & Davies, 2002), as these can respond to changes in the distribution of spectral

energy, as well as the total, in any part of the spectrum. However, the *spectral flux* only relies on magnitude information.

Bello *et al.* consider the *SF*, similarly to *PD*, especially adequate to strongly pitched transients (Bello, et al., 2005).

- **Phase-Based Methods (e.g., (Bello & Sandler, 2003), (Duxbury, Bello, Davies, & Sandler, 2003a), (Dixon, 2006)):** these were designed in order to compensate the shortcomings of the former approaches. We shall refer the spread of the distribution of *phase deviations* (*PD*), given by eq. (2.7), (Duxbury, Bello, Davies, & Sandler, 2003a), which are successful at detecting low and high-frequency tonal changes regardless of their intensity. Yet, they are especially susceptible to phases of noisy low-energy components, and to phase distortions common of complex audio signals. The *WPD*, given by eq. (2.24), and the *NWPD*, given by eq. (2.25), both proposed in (Dixon, 2006), are both very significant improvements of the *PD* function, but the normalization is only an improvement of the *WPD* in the presence of PP or CM data, while for pitched PN and NP data (Dixon, 2006) observed a slight degradation in the performance of the results.

  As referred, Bello *et al.* also consider the *PD* especially adequate to strongly pitched transients (Bello, et al., 2005).

- **Complex Methods (e.g., (Bello, Duxbury, Davies, & Sandler, 2004), (Duxbury, Bello, Davies, & Sandler, 2003b), (Dixon, 2006)):** In the complex domain, both phase and amplitude information work together to offer generally more robust onset detection approaches. These approaches are both straightforward to implement, and computationally cheap. Besides, they prove effective for a large range of audio signals. Some complex domain algorithms, like the *complex difference* (*CD*), given by eq. (2.9) (Duxbury, Bello, Davies, & Sandler, 2003b), and performing better on the lower frequency components of the spectrum, may be beneficial to incorporate them within a multi-resolution approach. This has the advantage that high frequency noise bursts may be used to improve time localization of hard onsets. The *RCD*, given by

eq. (2.26), as proposed by (Dixon, 2006), revealed to offer small performance improvements to *CD*, with non-significant differences.

Bello *et al.* consider the *CD* a good choice to any application in general, at the cost of a slight increase in computational complexity, when compared to *PD* or *SF* (Bello, et al., 2005).

- **Time-Frequency and Time-Scale Methods (e.g., (Davy & Godsill, 2002), (Daudet, 2001)):** As an exemplar TRF method we shall refer the *wavelet regularity modulus*, given by eq. (2.15), (Daudet, 2001). It can be used to precisely localize onset events in a theoretical resolution down to two samples of the original signal, which is typically better than the ear's resolution in time. However, this alternative resolution imposes a much less smooth detection function (requiring some post-processing to remove spurious peaks) and an increase in algorithmic complexity.

  Bello *et al.* consider the *WRM* especially useful, possibly in combination with another method, to applications requiring a precise detection of the onset times (Bello, et al., 2005).

- **Statistical Methods (e.g., (Abdallah & Plumbley, 2003)):** Probabilistic models provide a more general theoretical view of the analysis of onsets. If the model is adequate, then robust detection functions for a wide range of signals can be produced. An example is the *surprise-based method using ICA* to model the conditional probability of a short segment of the signal, calculated as the difference between two *negative log-likelihoods*, *NL*, (Abdallah & Plumbley, 2003), as given by eq. (2.16). This adaptive statistical model grants the most precise onset detection, but imposes a potentially computational-expensive and time-consuming training process to fit the parameters of the model to a given training set.

  Bello *et al.* considered the *NL* as the best option, due to its best overall accuracy and less dependence on a particular choice of parameters, if a high computational load is acceptable, and a suitable training set is available (Bello, et al., 2005).

Contesting some of (Bello, et al., 2005)'s results and conclusions, (Dixon, 2006) argued that the *SF*, *PD*, and the *CD* onset detection functions achieved similar higher level of performance, where *SF* was proven to be the best approach, offering the best tradeoff between accuracy and computational demands.

Based on Dixon's conclusions, and given the real-time requirements of our robot dancing system, we used Marsyas to implement the *spectral flux* method for the integrated note-onset detection function, as given by eq. (2.5). This was the basis of the implemented low-level rhythmic perception model integrated in our system's *Music Analysis Module* (see Chapter 3).

2.2 Dancing Robots

Interaction through dance is expected to improve the quality of symbiosis between robots and people in natural environments. Since human emotions have a close relationship to music and dancing, the research of these expressive non-verbal languages applied to robots may enhance the robots' sociability, opening communication channels besides spoken language. Besides, robotics has even become a mode of artistic expression by means of embodied mechanical actions with aesthetical designs.

This subsection presents the solutions achieved by some of the most notable researchers on the area of robotic dancing and rhythmic interactive robots.

The first dance-expressive robots set back to the 80s through robotic art performances, where choreographers and cinematographers explored the emotional and aesthetic dimension of robot movement into theatre and movie characters. The first cinematic robot dancing performance was introduced by Bob Roger's musical film "Ballet Robotique" (Figure 5a)), where a team of industrial robots on an automobile assembly line where programmed to dance together in synchrony to music recorded by London's Royal Philharmonic Orchestra. Since then, many

cinematographic and theatrical pieces were shot with differently shaped robots expressively performing dance with humans, in artistic scenes. The first most famous cinematic dancing robot was the one featured by Short Circuit's hero Johnny 5 (in Figure 5b)), back to 1986, which could already express emotions and body movements, controlled by a puppeteer wearing a telemetry suit. Other remarkable artistic robot dancing performances were presented by Jean-Marc Matos in the "Talos and Koïné" theatre piece, the Apostolos' "Mars Suite" human-robot choreography, the Artemis Moroni's "Foreseen Variations" installation, and by the music-theatrical performances of the Omnicircus from Frank Garvey and Chico MacMurtrie's Amorphic Robot Works. More recent artistic dancing robots include Giles Walker's Robot Pole Dancers (Figure 5c)) and Lapassarad Thanaphant's dancing robot waiter at Hajime Robot Restaurant (in Figure 5d)).

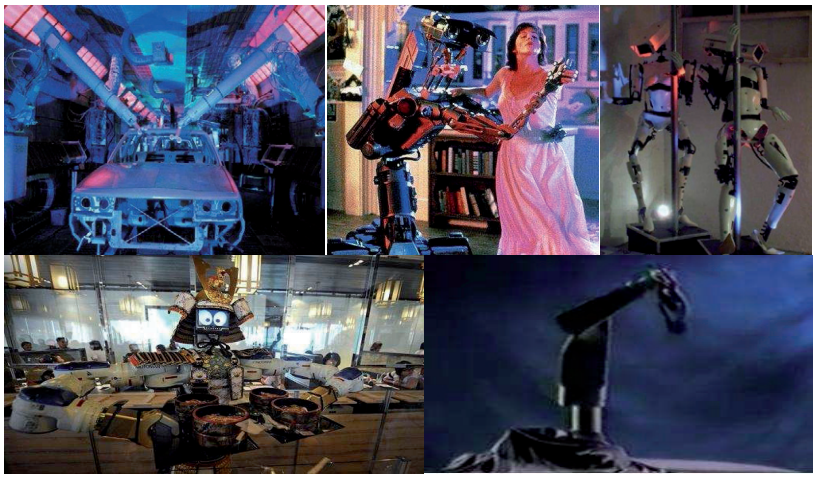


Figure 5 – Artistic dancing robots: **a)** Bob Roger's film "Ballet Robotique" [top-left]; **b)** Short Circuit's movie character Johnny 5 [top-middle]; **c)** Giles Walker Robot Pole Dancers [top-right]; **d)** Hajime restaurant's dancing robot waiter [bottom-left]; **e)** Apostolos' "FreeFlight" dancing industrial robot [bottom-right].

In the line of the former, Apostolos *et al.* presented a comprehensive study exploring the aesthetics of (industrial) robotic movements for robot choreography while artistically comparing different robot designs (see

Figure 5e)) (Apostolos, 1988), (Apostolos, Littman, Lane, Handelman, & Gelfand, 1996).

Later, by deeply exploring the paradigms of embodied interactions between humans and robots in artistic settings such as museum exhibitions (see Figure 6a)), theatre, and musical installations, Camurri *et al.* designed an interactive multi-modal architecture for emotional dancing agents (Camurri & Coglio, 1998). Their paradigm consisted of generating emotional outputs, through specific robot actions, such as motion, navigation, and light effects; and audio commands, through music composition and speech; in Reaction and Rationalization to multi-modal inputs acquired from human movement, musical context, and the robot emotional internal state. Their architecture was tested with children in a museum exhibition by recurring to a mobile robot, ActiveMedia Pioneer 1, dressed and equipped with on-boar remote-controlled loudspeakers, infrared localization sensors, and a Saphira navigation system.

Suzuki *et al.* extended this concept by introducing four musical platforms to create multimodal artistic environments for collaborative human-robot music and dance performances (Suzuki & Hashimoto, 2004). For supporting the interaction, their robotic system integrated acoustic and visual environmental inputs, and the robot's own movement, to dynamically react with motion while producing sound and music according to the context of the performance. All integrating modules, along with a flexible GUI, were interconnected through a MIDI network, which provided means for communicating and exchanging data.

The four human–machine multimedia-multimodal settings, depicted in Figure 6b), consisted of: *i)* a Reactive Audiovisual Environment, as an environment-oriented musical system where human movement and environmental sound produces contextualized sound and music in real-time (Figure 6b), top-left); *ii)* a Visitor Robot, that extends the former setting by generating robot motion and navigation within the interactive space according to the external audio-visual stimuli (Figure 6b), bottom-left); *iii)* an iDance installation, as an object-oriented setup through which the robot reacts to direct physical contact provided by a human performer (Figure 6b), top-right); and *iv)* a MIDItro setting, which combines all of the former

modes by reacting to the performer's voice, handclap, gesture, and direct physical contact (Figure 6b), bottom-right).

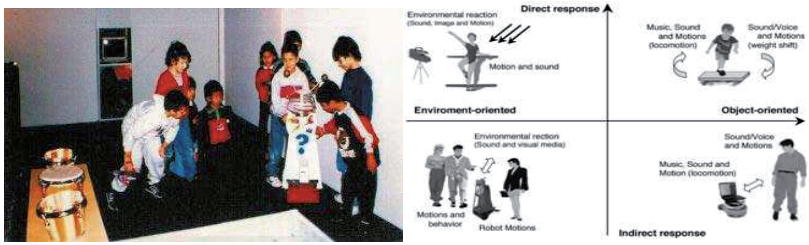


Figure 6 – Multimedia-multimodal human-robot installations: **a)** Museum exhibition at "Città dei Bambini" concerning children-robot interaction involving music, movement, dance, and a mobile robot (Camurri & Coglio, 1998) [left]; **b)** Four human–machine multimedia-multimodal settings (Suzuki & Hashimoto, 2004) [right]: Reactive Audiovisual Environment [top-left], Visitor Robot [bottom-left], iDance [top-right], and MIDItro [bottom-right].

Focusing on the most recent approaches concerning robot dancing, we may start to refer some of the existing commercial edutainment toy robots with embedded choreography editors and high-level motion controllers. The most known for their dance orientation are the RoboSapien robots created by WooWee Robotics (in Figure 7a)) — RoboSapien, RoboSapien V2, FemiSapien, and RoboSapien RS Media (WooWeeRobotics, 2008). These robots' dancing behaviors may be designed through visual programming environments such as Robo-Go Choreographer, and users may control and trigger their dance creations, through the RobotDance application, by recurring to voice commands, to the Nintendo's Wii Remote, or to the RoboRemote. Other similar low-cost edutainment humanoid platforms, with *ad hoc* motion editors for easily composing continuous point-to-point dancing sequences, include Kondo Kagaku and Speecys robots, as well as Hitec's Robonova-I (HitecRobotics, 2008) and Aldebaran Robotics' robot NAO (Gouaillier, et al., 2008). Other platforms already used for robot dancing include programmable robotic kits such as Lego Mindstorms NXT, as used in this work, and Robotis' Bioloid.

Other trendy dancing toy-robots include the SegaToys' iPets (SegaToys, 2008), iCat, iDog (Figure 7b)), iFish, iCYPenguin, and iSpin, and the USB Dancing Robot (Gizmodo, 2008) (in Figure 7c)). These Tamagotchi-like plastic pets were especially designed for being connected to an iPod (or

other MP3 player) or placed next to a loud speaker. These little robots come in a variety of colors and are "fed" with music by interacting with it through flashing LEDs (Light-Emitting Diodes) in time to the music and bopping/moving to the beat. Some of them even display mood feelings given the type of music and the level of music deprivation.

Figure 7 – Commercial dance-oriented robots: **a)** WowWee's RobotSapien Family (WooWeeRobotics, 2008) [left]; **b)** iDog Pup (SegaToys, 2008) [middle]; **c)** USB Dancing Robot (Gizmodo, 2008) [right].

Back to the academia, Nakazawa, Nakaoka *et al.*, from Tokyo University, presented an approach for designing a biped robot, HRP-2, that could imitate the spatial trajectories of complex motions recurring to a Motion Capture (MoCap) system (see Figure 8) (Nakaoka, et al., 2007), (Nakaoka, et al., 2005), (Nakaoka, 2003). To do so, they developed a Learning-From-Observation (LFO) training method that enabled a robot to acquire situated knowledge from observing human demonstrations, by relying on predesigned task models which represent only the actions that are essential to mimicry. This method was applied in the performance of Japanese traditional folk dances imitating a female human dancer. For such, and because the leg and upper body have different purposes (the leg motions stably supported the robot body, while the upper-body motions expressed dancing patterns) as well as different motor constraints, the authors applied different strategies to design task models for leg and upper-body motions. These two motion types were concatenated and adjusted in the final stage. Finally, to generate executable motion, while considering balance and stability issues, the authors applied a dynamic filter to compensate the Zero-Moment Point (ZMP) and the yaw-axis moment of

the robot, and conducted skill refinement to resolve other kinematic problems such as self-collision.

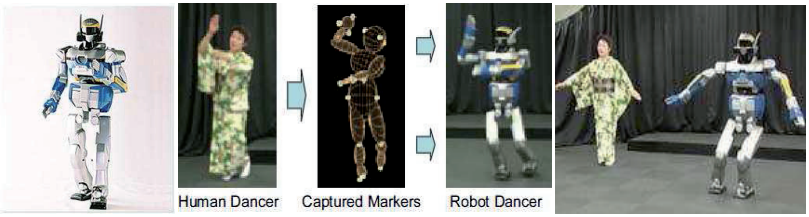


Figure 8 - HRP-2 learning from observation (Nakaoka, et al., 2005): **a)** HRP-2 humanoid robot [left]; **b)** Mapping human dancing movements onto HRP-2 [middle]; **c)** HRP-2 imitating an Aizu Bandaisan Japanese dance performance [right].

Despite the flexibility of motion generation, the referred approach was not able to autonomously adjust the timing of the robot's dancing motion while interacting with the auditory environment, i.e., while listening to music. To overcome this limitation, later, (Shiratori, Kudoh, Nakaoka, & Ikeuchi, 2007) modeled the proper modifications to the generated upper body dance motion in way that it would follow the speed of the played music (see Figure 9). Their model was based on the insights that high frequency components are gradually attenuated with the increase of the music's speed, and that important stop motions are preserved even when high frequency components are attenuated. For that purpose, and to satisfy the joint limitations of the robot, the authors analyzed motion data at varying musical speeds by using a hierarchical motion decomposition technique – the hierarchical B-spline. All motion data was captured at 120 fps (frames per second) by an optical motion capture system, produced by Vicon, and each joint angle was calculated and converted into a 3D logarithmic space using quaternion algebra.

Their final tests were also performed in HRP-2 with which they achieved successful upper-body robot dance performances at different speeds, while satisfying the criteria for balance maintenance.

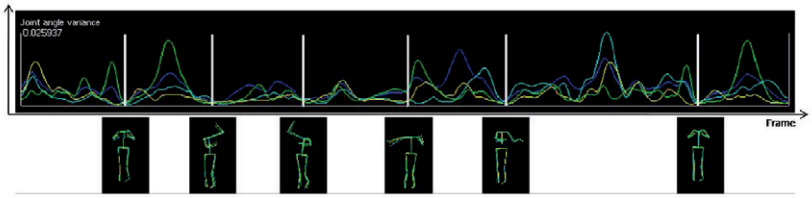


Figure 9 – Joint angle variance on an Aizu Bandaisan dance performance at different speeds (Shiratori, Kudoh, Nakaoka, & Ikeuchi, 2007): original speed (green), 1.2 times faster (yellow), and 1.5 times faster (light blue); with dark blue line representing the joint angle variance along all motion sequences. Below are the preserved dance key-poses among all musical speeds.

Still focused on the imitation of human dance movements by learning, Tidemann *et al.* taught a robot to dance the YMCA (in Figure 10) by applying the self-organization of a connectionist modular architecture for motor learning and control (Tidemann & Öztürk, 2007). Their online learning by imitation approach was based on a recurrent neural network architecture that used Multiple Paired Inverse/Forward Models (MPMA) to acquire and generate matching dancing motor primitives. The MPMA generated robot motor commands, in the form of joint velocities, to match the joint angles of the demonstrator, acquired via motion capture.

Figure 10 – Robotic simulation of YMCA key-poses by imitating a human performer (Tidemann & Öztürk, 2007).

For more educational purposes, Tanaka *et al.* from Sony, used QRIO (see Figure 11) for developing a dancing robot that could interact with children (Tanaka, Fortenberry, Aisaka, & Movellan, 2005), (Tanaka & Suzuki, 2004). Their implementation used a posture mirroring method for generating dance motions relying on cyclic repetitions of sympathetic and dynamic behaviors, based on an Entrainment Ensemble Model (EEM). To keep a synchronous interaction with children the authors used a "Rough but Robust Imitation" visual system through which QRIO mimics the detected human movements, in an entrainment process.

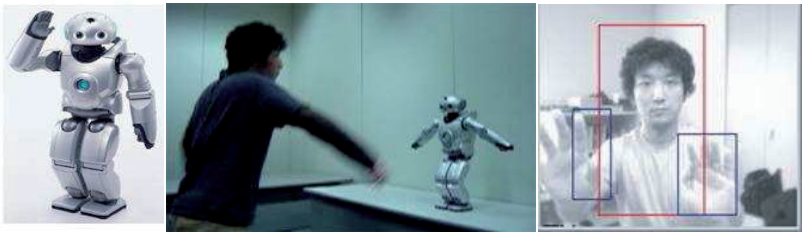


Figure 11 – Sony entertainment robot QRIO (Tanaka & Suzuki, 2004): **a)** QRIO humanoid robot [left]; **b)** QRIO reacting to human movement by following its rhythm and shape [middle]; **c)** A set of moving-regions obtained by QRIO's cameras [right].

In the line of iPets, (Aucouturier & Ogai, 2007) used a robot designed by ZMP, called MIURO (see Figure 12), for dancing while playing music from an embedded iPod. For generating dancing behaviors in a seemingly autonomous manner the authors designed basic dynamics through a special type of chaos (chaotic itinerancy (CI)) for the robot to exhibit a variety of periodic motion styles alternating from detached independent movements to others strongly attached to the musical rhythm. The robot motor commands were generated in real-time by converting the output from a neural network that processes a pulse sequence corresponding to the beats of the music. Each neuron was (biologically) inspired in the FitzHugh-Nagumo model to generate a chaotic itinerant dancing behavior among low-dimensional local attractors of higher dimensional chaos. The resulting dancing revealed a strong compromise between musical-synchrony and autonomy which ultimately enhanced the long-term interest of human audiences.

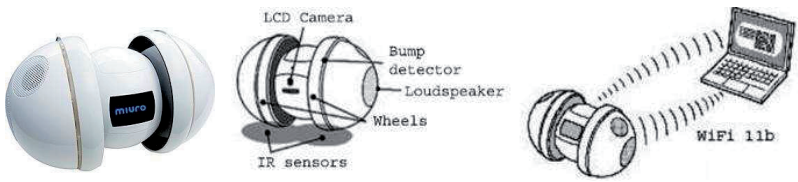


Figure 12 - The MIURO robot dancing platform (Aucouturier & Ogai, 2007): **a)** MIURO white edition [left]; **b)** robot's constitution as a two-wheeled musical player equipped with an iPod mp3 player interface and a set of loudspeakers [middle]; **c)** Wheel velocities can be controlled in real-time through wireless communication with a computer [right].

On the same year, Sony launched a similar egg-shaped robot named Rolly (see Figure 13) as a conventional mp3 portable player with musically-synchronous dancing moves and light effects (Sony, 2008). Besides its two 1.2 W speakers, Rolly is equipped with several motors for rotation and spinning, two LED bands, two movable "arms", and an accelerometer for vertical orientation. Its audio analysis functions extract musical components such as beat, meter, voice and pitch, which are used for different reactions. Proprietary choreographical software allows users to create their own dancing routines or download extra dancing schemes from Sony's repository. Ultimately, a built-in Bluetooth module supports music streaming directly from other devices like PCs, mobiles and Hi-Fi (High-Fidelity) systems.

Figure 13 – Sony's Rolly (Sony, 2008): **a)** Rolly white edition [left]; **b)** Dancing music player mode of operation [right].

(Burger & Bresin, 2007), and (Burger, 2007) used the Lego Mindstorms NXT kit to design a robot, named M[ε]X (from Musical EXpression) — in Figure 14 — which expressed movements to display emotions embedded in the audio layer, in both live and recorded music performances. Their robot had constraints of sensors and motors, so the emotions (happiness, anger and sadness) were implemented by taking into account only the main characteristics of musicians' movements. For evaluating if the robot movements supported the intended emotions without recurring to *ad hoc* musical stimuli, the authors designed two behavioral evaluation scenarios with subjects that experienced and assessed robot emotions with and without music.

Figure 14 - The M[ε]X emotional expressive robot (Burger, 2007).

Focused on a more active robotics perspective, towards human interaction, (Takeda, Hirata, & Kosuge, 2007) proposed a dance partner robot, which was named MS DanceR, consisting of a platform for realizing effective human-robot coordination with physical interaction. MS DanceR, as illustrated in Figure 15, consists of an omni-directional mobile robotic interface which moves along dance-step trajectories during a ballroom dance, including a force/torque sensor (Body Force Sensor) to realize compliant physical interaction between the robot and a human, by measuring the user leading force moment. For estimating the next intended dance step according to the human lead at the transition, a Step Estimator module uses a set of Hidden Markov Models (HMM) to stochastically model the time series of the leading force-moment. According to step transition rules, based on motion constraints, the Motion Generator module generates cooperative robot dancing movements in a close-loop physical interaction for accompanying the human dance partner.

Figure 15 - The dance partner robot (Takeda, Hirata, & Kosuge, 2007): **a)** A force-torque sensor between the robot's upper and lower body measures the human leading force-moment [left]; **b)** MS DanceR two-colors layout [middle]; **c)** An omni-directional mobile base uses special wheels to move along dance-step trajectories [right].

Considering rhythmicity as a holistic property of social interaction, (Michalowski, Sabanovic, & Michel, 2006), (Michalowski, Kozima, & Sabanovic, 2007), and (Michalowski & Kozima, 2007) investigated the role of rhythm and synchronism in human-robot interactions, and their application in pedagogical and therapeutical scenarios. For this purpose, the authors applied perceptive techniques and generated social rhythmic behaviors in non-verbal interactions through dance between Keepon (Michalowski, Kozima, & Sabanovic, 2007), (Michalowski & Kozima, 2007), a yellow puppet-like robot with 4 DoF (see Figure 16a) and Figure 16b)), or Roillo (Michalowski, Sabanovic, & Michel, 2006), a robotic virtual platform (see Figure 16c)), and children. For perceiving rhythm over different modalities, their robotic system integrated musical signal-processing techniques to detect the musical tempo, hand-clapping, or drum-beats; and computer-vision methods, and accelerometers (and pressure sensors) to enable the perception of repetitive movements by peoples' heads, arms, or bodies.

In (Michalowski, Kozima, & Sabanovic, 2007), Keepon attended to both auditory and visual data, and extracted movement by computing the average optical flow in a region of interest of an incoming video stream. In (Michalowski & Kozima, 2007), Keepon recurred to auditory stimuli and spatial sensing by using a battery-powered three-axis accelerometer, with wireless Bluetooth data transfer, implanted in a toy, that detected rhythmic movements by finding magnitude peaks, after applying a zero-crossing or low-pass filtering the retrieved data. These peaks were then treated as "beats" in the same way as musical beats or visual movement direction changes, as in (Michalowski, Kozima, & Sabanovic, 2007).

Ultimately, a Max/MSP (Max/Max Signal Processing) object, sync~, received the stream of multi-modal beats and produced an oscillator synchronized with the given tempo. This oscillator drove a stream of commands that cyclically moved Keepon's bobbing and rocking degrees-of-freedom. At last, a sequencer was used to record aligned streams of beats, sensor data, and motor commands for later playback and behavioral analysis.

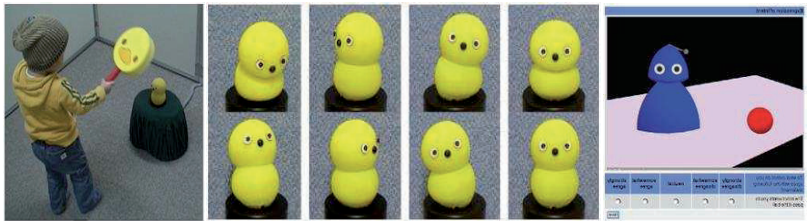


Figure 16 – Rhythm and Synchrony in human-robot interactions: **a)** Keepon dancing with a child (Michalowski & Kozima, 2007) [left]; **b)** Keepon's body motions with its 4 DoF (Michalowski & Kozima, 2007) [middle]; **c)** Roillo requesting the ball using a deictic gesture (Michalowski, Sabanovic, & Michel, 2006) [right].

By extending robot dancing to multi-robot choreography, (Park, Kim, Lee, Yoo, & Kim, 2007) presented an heterogeneous robot dancing team, RoboBees, constituted by one HSR-VII (HanSaRam 7) and several Robonova humanoids (see Figure 17). Their system performed synchronized dance movements among the team of robots by generating and combining both periodic motions, through an online pattern generator, and aperiodic motions, through an offline pattern generator. For generating and stabilizing the online periodic motions, the authors introduced a time-domain passivity compliance control system for changing the initial planned trajectories accordingly.

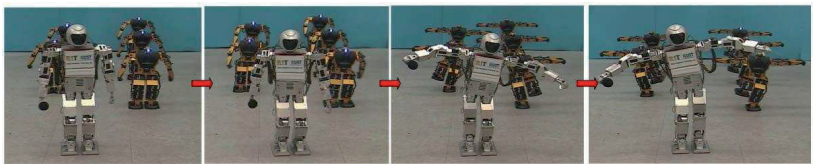


Figure 17 - Synchronized aperiodic dancing motions among a team of robots (Park, Kim, Lee, Yoo, & Kim, 2007).

(Ellenberg, Grunberg, Oh, & Kim, 2008) also used Robonova as their humanoid robot dancing platform – see Figure 18. In their implementation, they applied a real-time beat tracker, based on Klapuri's algorithm (Klapuri, Eronen, & Astola, 2006), for synchronizing the robot's dance motion to the beat of the music. Their application coordinated the robot's dancing by choosing a random series of gestures, from a motion library, and linearly interpolating them, point-to-point, for generating dancing

sequences. The robot's balance was manually assured *a priori* with a careful design of the integrated dancing motions. A graphical user interface, shown in Figure 18b), was also designed to let users choreograph gesture sequences, which extended the system towards a basic choreographic tool.

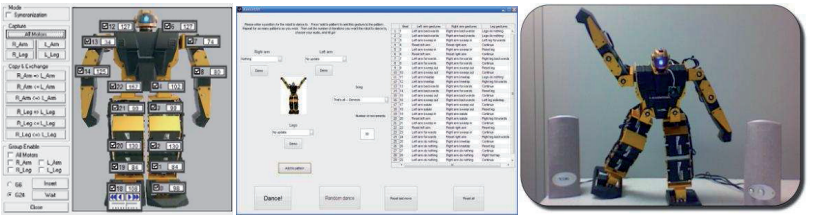


Figure 18 – Dancing Robonova (Ellenberg, Grunberg, Oh, & Kim, 2008): **a)** Motion editor [left]; **b)** Application GUI [middle]; **c)** Demonstration screen-shot [right].

By using a similar humanoid platform (see Figure 19a)), Shinozaki *et. al* investigated the role of robots in entertainment (Shinozaki, Oda, Tsuda, Nakatsu, & Iwatani, 2006) and designed a robot dancing system to perform dancing sequences concatenated from short dance motions (named dance units), adapted from real human performances (Shinozaki, Iwatani, & Nakatsu, 2007). For such, the authors collaborated with a professional dancer to extract around sixty hip-hop dancing units, which were converted into humanoid poses, using a motion editor. By linearly interpolating those dancing primitives, using a neutral posture between them, the authors could perform a huge amount of dance variations, representing a repertoire of generic hip-hop dancing.

Later, the authors developed a real-time audio beat tracker in order to generate beat-synchronous robot dancing sequences (Nakahara, et al., 2009). The detected beats conducted the linear interpolation between key-poses, to be accomplished within the duration of the current Inter-Beat-Interval (IBI). Additionally, the system also supported symbolic audio signals inputted from a MIDI keyboard. Through this device the user could control both the musical tempo and the musical strength, which, respectively, controlled the velocity of motion execution and the range of movements.

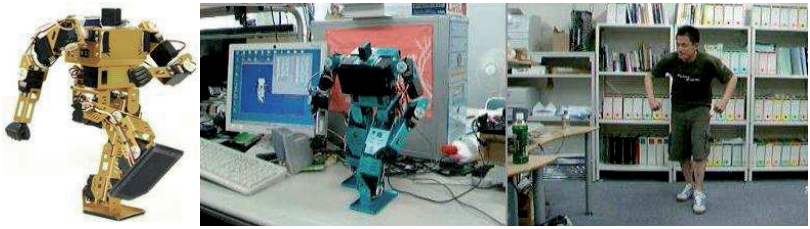


Figure 19 – Generating humanoid motions for entertainment (Shinozaki, Iwatani, & Nakatsu, 2007): **a)** Tai-Chi humanoid robot motion; **b)** Adapting a hip-hop dancing unit from a human dancing posture to the robot.

Concerned with robotic anthropomorphism, Or designed the first flexible spine humanoid robot for belly dancing performances (Or, 2006), (Or, 2009) (see Figure 20). Inspired by the rhythmic movements commonly exhibited in lamprey locomotion, the author simulated belly-dancing by replicating the lamprey's Central Pattern Generator (CPG) for developing a control architecture for the robot spine's high degree-of-freedom. The lamprey CPG follows a connectionist model consisting of 100 interconnected copies of a segmental oscillator with eight neuron units each. These receive global and local excitations from the lamprey brain stem, to, respectively, control the frequency of oscillation of the CPG, and altering the inter-segmental phase lag.

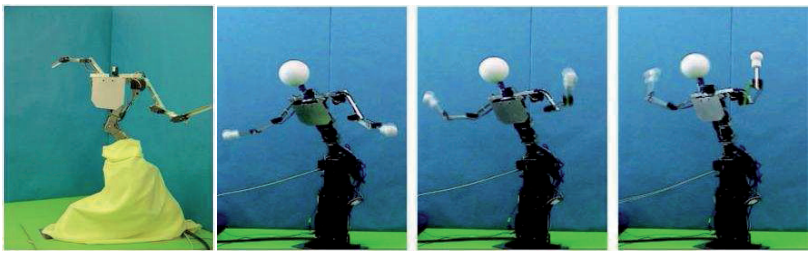


Figure 20 – Flexible spine humanoid belly dancer (Or, 2009): **a)** The Waseda Belly Dancer no. 1 (WBD-1) [left]; **b)** WBD-2 performing emotional belly dancing with flexible and undulatory movement [right].

By varying the global and extra excitations from the brain stem as well as the plane of movements, the proposed lamprey CPG model could generate plausible output patterns, which could be used for all the possible motions of the human spine. By designing the control for basic spine

movements, such as (lateral) flexion, hyperextension, and twisting; and complex movements, such as chest circle and undulation (see Figure 20b)), Or was able to produce full-body flexible spine emotional humanoid robots, which could dance more naturally and at a lower cost than conventional research-type humanoid robots.

Distinctively, Schöllig *et. al* proposed a control algorithm to enable flying vehicles to perform musically-synchronous rhythmic movements (Schöllig, Augugliaro, Lupashin, & D'Andrea, 2010). The proposed algorithm synchronized the side-to-side motion of a quadrocopter (in Figure 21c)) with the musical beat and melody, while stabilizing the vehicle in the air, contradicting its non-linear and unstable dynamics. The 2D side-to-side quadrocopter oscillation was defined in the xz-pane by a cascade controller design. The actual synchronization was achieved by using concepts from Phase-Locked Loops (PLL), as depicted in Figure 21b), in order to reach the quadrocopter side-to-side outermost positions with the beat. A feedback strategy was used to keep the vehicle in phase with the musical beats (pre-processed from the music). Additionally, a feed-forward component was added to achieve fast adaptation to frequency (beat) and amplitude (melody) changes.

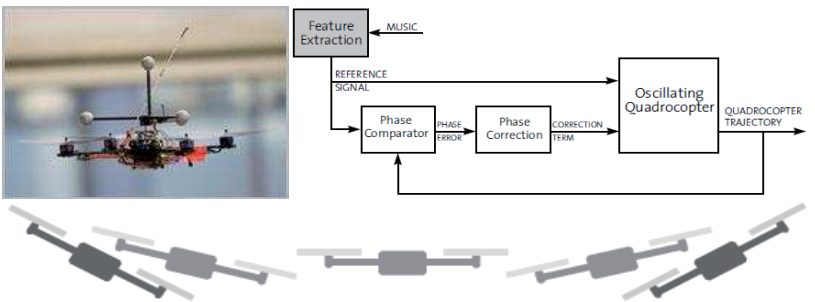


Figure 21 – Quadrocopter motion synchronized to music (Schöllig, Augugliaro, Lupashin, & D'Andrea, 2010): **a)** Quadrocopter [top-left]; **b)** Overall system architecture based on PPL for beat-synchronous quadrocopter motion [top-right]; **c)** Side-to-side 2D quadrocopter motion [bottom].

Ultimately, we may refer worldwide robot dancing contests where school teams and research groups program their robotic creations to dance in creative displays of costumes, movement and music. The most

emblematic competitions are presented by RoboCup Junior's Dance (RoboCupJunior, 2008) (Figure 22a)), ROBO-ONE GATE Dance Competition (ROBO-ONEEntertainment, 2008) (Figure 22b)), and by the Austrian's Hexapod Dancing Championship (UAS, 2008) (Figure 22c)).

Figure 22 – Robot dancing contests: **a)** RoboCup Junior's Dance (RoboCupJunior, 2008) [left]; **b)** ROBO-ONE GATE Dance Competition (ROBO-ONEEntertainment, 2008) [middle]; **c)** Hexapod Dancing Championship (UAS, 2008) [right].

2.3 A Step Further

Most of the referred musical robotic systems lack from flexibility and human control, presenting mainly reactive robots manifestly stiffed to their pre-programmed functions (i.e., through a fixed sequence of motor commands). Their musical perceptive systems are typically mere applications of existing models, with the dance movements being rendered to a given piece of music by adapting the execution speed of the dance motion sequence to the musical tempo (which is, in many approaches, extracted *a priori* from the musical signal.

These approaches have merits, with a notable convincing effect of musical-synchrony, but typically fail at sustaining long-term interest, since the dance repertoire of the robot is rapidly exhausted and frequent patterns begin to reoccur without any variation.

Improving these absences, by developing a customizable framework in which users have a deterministic role, by flexibly defining the robot choreography through selected individual dance movements that are able to react in real-time to multi-modal external events, seems the perfect start to give a step further to robot dancing applications that would exhibit a compromise between *musical-synchrony*, *variability*, and *animacy*.

Chapter 3

System Architecture

The implemented robot dancing system architecture was firstly published in (Oliveira, Gouyon, & Reis, 2008a) and fully described in (Oliveira, 2008b). This system is composed of a humanoid robotic agent (see Figure 23 and Figure 24), built with two Lego Mindstorms NXT kits[13]; a hand-made dance environment, composed of a multi-color floor and a covering wall to delimit the dance space (see Figure 25); and a robot dancing control software constituted by three modules (see Figure 26): *Music Analysis Module*, *Robot Control Module*, and *Human Control Module*.

The proposed architecture generates reactive robot dancing behaviors in response to multi-modal events formed by *i)* three rhythmic events: *Low*, *Medium* or *Strong* onsets; and two sensorial event classes defined by ii) the stepped color: *Blue*, *Yellow*, *Green*, *Red*; and *iii)* the proximity to a surrounding obstacle: *OK*, *Too Close*. By playing with these inputs a user can, through a proper interface, flexibly define a set of dance moves, which are sequenced during the dance performance. Contrasting to some other approaches, every body movement, as their progression during the dance, is produced by the robot in an autonomous way without former knowledge of the music. Besides, the proposed framework abdicates from strict *musical-*

[13] For the Lego Mindstorms NXT complete kit set constitution and overview (i.e., pieces and correspondent references), consult http://www.peeron.com/inv/sets/8527-1.

synchrony in favor of sustaining the long-term interest of the general audience by promoting *variability* and *animacy* to the robot dance performance.

A video displaying an overview of the framework's functionalities is available in (Oliveira, Reis, & Gouyon, 2008c).

In this chapter we fully describe the proposed robot dancing system architecture. This description is depicted in three sections: Section 3.1 – *Dancing Robotic Agent*, Section 3.2 – *Dance Environment*, and Section 3.3 – *Dancing Control System*.

3.1 Dancing Robotic Agent

Following the idea that humanly shaped robots greatly provide the anthropomorphism requested by natural interactions, and that dance, as a bodily language, requires a physical body, our dancing robotic agent was designed as a humanoid with six degrees-of-freedom (DoF), as illustrated in Figure 24. For its conception, we used two Lego Mindstorms NXT kits, each composed of an NXT-brick (i.e., a brick-shape automaton) with Bluetooth support, and each connected to three servo-motors. In total, the six motors control two legs, which form an omni-directional base, two arms, the head (along with a spinning fan), and a rotating hip. In addition, we connected a color sensor to our robot, for detecting and distinguishing visible colors, and an ultrasonic sensor for obstacle detection. Ultimately, to increase the animacy and amusement of our robot's aesthetics, we dressed it with a red skirt that spins with the robot hip while dancing (see Figure 23b)).

This humanoid robot's design allowed the definition of 14 distinct dance movements, defined as BodyPart-Movement ("L" to the Left, "R" to the Right, or "Alternate" to combine alternated movements): *Legs-RRotate*, *Legs-LRotate*, *Head-RRotate*, *Head-LRotate*, *Body-RRotate*, *Body-LRotate*, *RArm-RRotate*, *RArm-LRotate*, *LArm-RRotate*, *LArm-LRotate*, *2Arms-RRotate*, *2Arms-LRotate*, *2Arms-RAlternate*, and *2Arms-LAlternate*.

Table 1 – Designed robot dancing movement's description: correspondent motor(s) and rotational direction.

Movement	Motor	Sign (Direction)
Legs-RRotate	RLeg	+
	LLeg	-
Legs-LRotate	RLeg	-
	LLeg	+
Head-RRotate	Head	+
Head-LRotate	Head	-
Body-RRotate	Body	+
Body-LRotate	Body	-
RArm-RRotate	RArm	+
RArm-LRotate	RArm	-
LArm-RRotate	LArm	+
LArm-LRotate	LArm	-
2Arms-RRotate	RArm	+
	LArm	+
2Arms-LRotate	RArm	-
	LArm	-
2Arms-RAlternate	RArm	+
	LArm	-
2Arms-LAlternate	RArm	-
	LArm	+

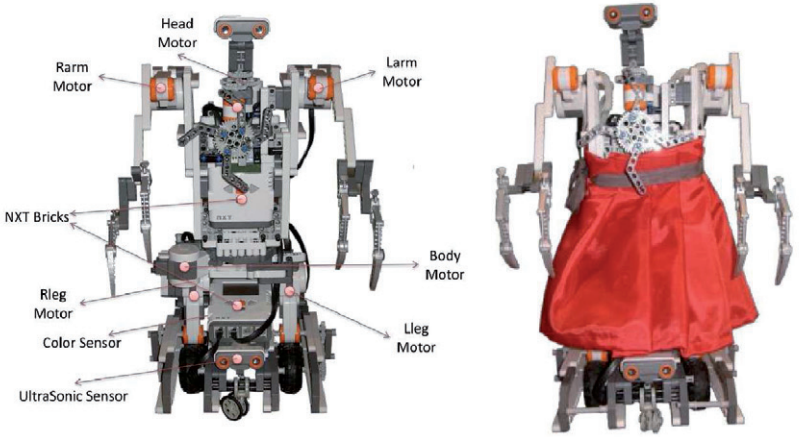


Figure 23 – Lego NXT humanoid robot: **a)** Robot's sensorimotor constitution [left]; **b)** Robot dancing outfit [right].

Attending to Figure 23 and Figure 24, Table 1 maps each movement to its correspondent motor (i.e., actuator), and the defined rotational direction.

3.2 Dance Environment

The designed dance environment (see Figure 25) was intended to simulate a real world environment for creating a realistic context to experiment the robot dancing, comparable to real human dancing. It incorporated a multi-color floor, for inducing dance variations dependent on the stepped color, and a covering wall to delimit the dance space. The dance floor was created with four paperboards (one red, one yellow, one green, and one blue) and it was surrounded with polystyrene to fulfill the walls.

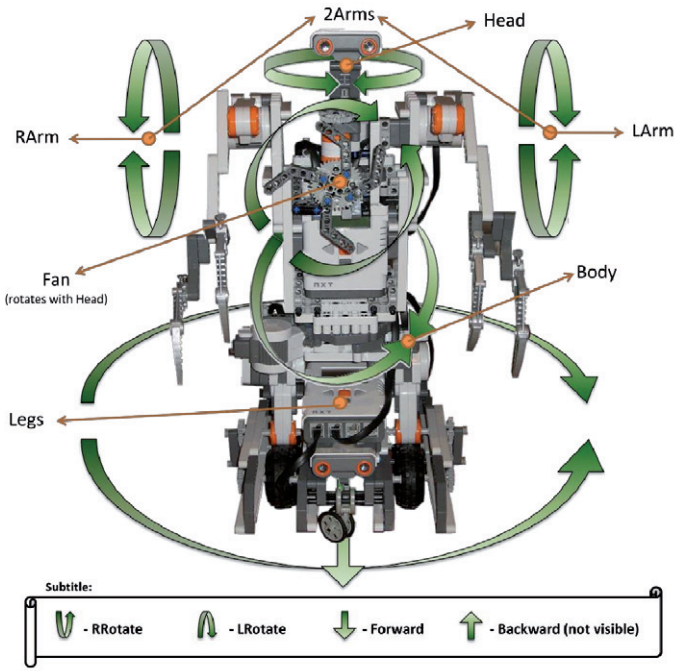


Figure 24 – Dancing robot's degrees-of-freedom.

Figure 25 – Real-world dance environment.

3.3 Dancing Control System

The modular architecture of the proposed system was designed to control the robot dancing tuned to multi-modal events while supplying flexible human control. As illustrated in Figure 26, this implementation was composed of three control modules. Initially, the *Music Analysis Module* applies a real-time onset detection algorithm to detect musical rhythmic events (i.e., note-onsets), at three defined levels of magnitude. These events are then sent in real-time, via UDP/IP sockets, to the *Robot Control Module*. By combining these rhythmic events with sensorial events, received from the robot's sensors, this module decides on the motor commands that are sent to the robot via Bluetooth to control its dancing (see Figure 29). Above the former two, a *Human Control Module*, composed of a Graphical User Interface, enables flexible user control over the system behavior. It provides a control panel for the configuration of the analysis' parameters, and an interface for the dance sequence composition. To keep the parallelism of behaviors and the demanded real-time

sensorimotor processing, all these modules run in a multi-threading architecture (see Figure 27). This architecture submits each thread (i.e., Module's function) to a time-division multiplexing (i.e., "time slicing"), in very much the same way as the parallel execution of multiple tasks.

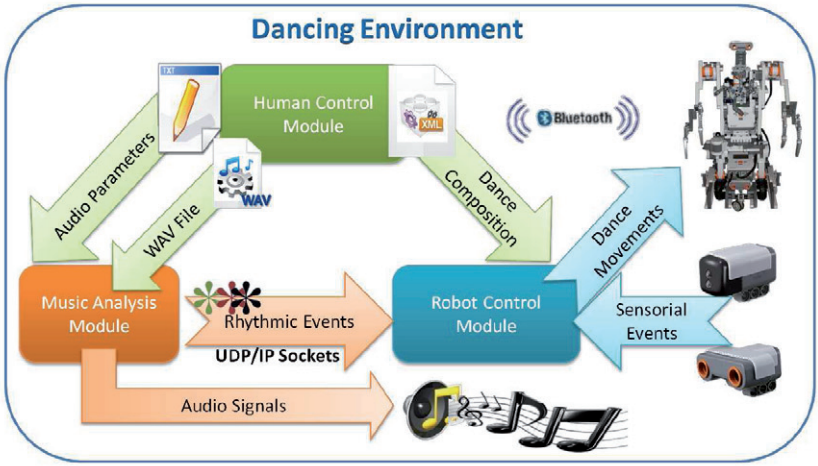


Figure 26 – Dancing control system modular architecture.

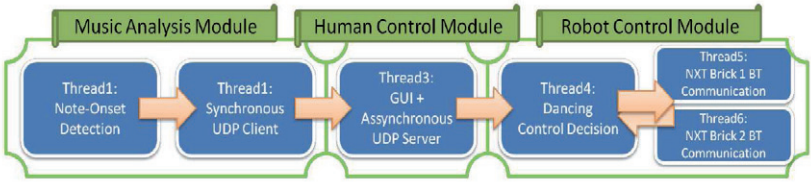


Figure 27 –Multithreading processing architecture between the three control modules.

3.3.1 Music Analysis Module

Music is generically an event-based phenomenon for both performer and listener, formed by a succession of sounds and silence organized in time. We nod our heads or tap our feet to the rhythm of a piece, by focusing our attention in the successive occurrence of notes in a musical piece (Bello, et al., 2005). In dance, body movements emerge as a natural response to such musical rhythmic events.

To retrieve these rhythmic events from music, on-the-fly, we focused our musical analysis on the detection of the musical onset times through a

real-time onset detection function, over music data composed of polyphonic audio signals.

This real-time onset detection function was designed in Marsyas based on the signal's *spectral flux*, as proposed by (Dixon, 2006), and given by eq. (2.5) (see Section 2.1.1.2). As depicted in Figure 28, in Marsyas such implementation comprised the combination of a set of functional blocks (i.e., classes), named `MarSystems`[14], and each described as follows:

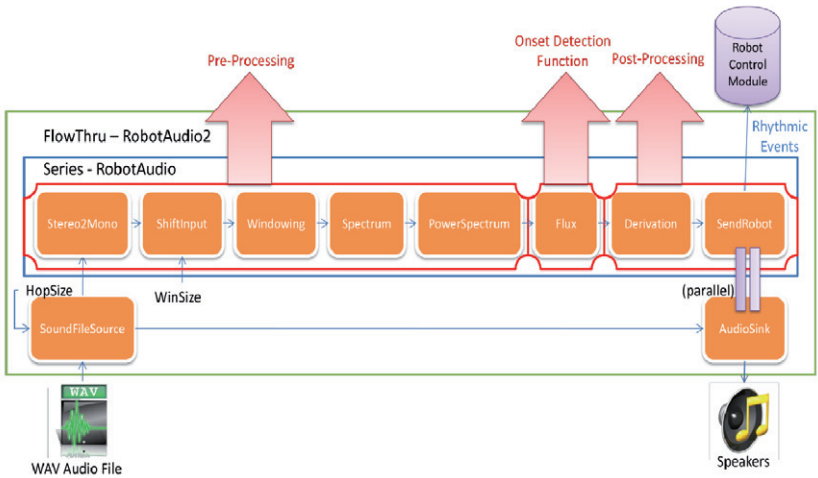


Figure 28 – Block diagram of the spectral flux's onset detection function implemented in Marsyas.

- **MarSystem:** This composite represents the abstract base class for any type of system. Basically a `MarSystem` takes as input a vector of float numbers (entitled `realvec`) and produces a new vector (possibly with different dimensionality). Hence, `MarSystems` are the core processing blocks of Marsyas, which perform some kind of signal transformation.

- **Series:** This structure combines a series of `MarSystem` objects to a single `MarSystem`. It corresponds to executing the system objects one after the other in sequence.

[14] Find the user manual of Marsyas at http://marsyas.info/docs/manual.

- **FlowThru:** This composite combines a series of `MarSystem` objects to a single `MarSystem`, sequentially executed one after the other, but forwards the original composite input to the output. This structure grants that the sound file is simultaneously analyzed and reproduced, and that the `SendRobot` (*Music Analysis Module* output) and `AudioSink` (speakers output) occur at (almost) exactly the same time (marked as "parallel" in Figure 28). This composite `MarSystem` assures the musical-synchrony of the performed dance to the reproduced audio.

- **SoundFileSource:** This `MarSystem` represents the first functional block, which consists of reading and loading the chosen audio WAV (WAVEform Audio Format) file input to be further analyzed.

- **Stereo2Mono:** This class converts the stereo input file to a mono output, in order to simplify the signal processing.

- **ShiftInput:** This block overlaps each consecutive frame of the signal in order to grant a smoother analysis. Its output emerges in the form of overlapped windows of the input signal, with their size adjusted by the defined `WinSize` (i.e., windows size). The analysis step is called hop size and equals to the frame size minus the overlap (typically 10 ms). The hop size (assigned as `HopSize`) defines the data granularity. In general, more overlap will give more analysis points and therefore smoother results across time, but the computational expense is proportionately greater.

- **Windowing:** Windowing of a simple waveform causes its Fourier transform to have non-zero values (commonly called leakage) at frequencies other than ω. It tends to be worst (highest) near ω and least at frequencies farthest from ω. Windowing in the time domain results in a "smearing" or "smoothing" in the frequency domain. This implementation used a Hamming Window (HW), due to its moderation. The Hamming window does not have as much side-lobe suppression as

other windowing functions (like e.g., Blackman), but its main lobe is narrower. Its side-lobes "roll off" very quickly versus frequency. This window is in the family known as "raised cosine". Its equation is described as follows:

$$HW(n) = 0.53836 - 0.46164\, cos(\frac{2\pi n}{N-1})\,. \tag{3.1}$$

- **Spectrum:** A periodic signal can be defined either in the time domain, as a function, or in the frequency domain, as a spectrum. In order to transform the signal from time to frequency domain a Fourier Transform shall be applied. Its discrete version is defined as follows:

$$X(f) = \sum_{n=-\infty}^{\infty} x[n]\, e^{-i2\pi \frac{f}{f_s} n}\,, \tag{3.2}$$

where $f = \frac{\omega}{2\pi}$ represents the frequency, ω the angular frequency, $f_s = \frac{1}{T_s}$ the sampling frequency, t the time domain and *x[n]* represents the samples of the signal *x(t)*, given by:

$$x(t) = ze^{i\omega t} = z(cos(\omega t) + isen(\omega t))\,, \tag{3.3}$$

$$z = |z|e^{i\theta} = |z|(\cos(\theta) + isen(\theta))\,, \tag{3.4}$$

$$\theta = \omega t + \phi\,, \tag{3.5}$$

where ϕ is the phase offset.

Therefore, this block applies the Fast Fourier Transform (FFT) to compute the complex spectrum (with *N/2+1* points) of each input window (given by the former block). Its output is an *N*-sized column vector (where *N* is the size of the input audio vector and *N/2* is the Nyquist bin), in the following format:

[Re(0), Re(N/2), Re(1), Im(1), Re(2), Im(2), ..., Re(N/2-1), Im(N/2-1)] .

A signal's spectrum, *X(n, k)*, is then displayed by a plot of the Fourier coefficients as a function of the frequency index, where the FFT is defined as:

$$X(n,k) = \sum_{m=-\frac{N}{2}}^{\frac{N}{2}-1} x(hn+m)HW(m)e^{-\frac{2\pi i}{N}mk},\ k = 0,\dots,N-1\,, \quad (3.6)$$

where *n* is the frame number, *k* the frequency bin, *h* the hop size, and *N* the window size, which are parameters already defined in the former blocks.

Any aspect of the signal can now be retrieved from its audio spectrum.

- **PowerSpectrum:** The power spectrum of a signal, $PS_{dB}[n]$, also referred as the energy/power spectral density, represents the contribution of every frequency of the spectrum to the power of the overall signal. It is useful because many signal processing applications, such as onset detection, are based on frequency-specific modifications of the musical signal. Hence, this class computes the magnitude/power of the complex spectrum (in decibels (dB)), by taking *N/2+1* complex spectrum bins and processing the corresponding *N/2+1* decibel's real values. Its function is described as follows:

$$PS_{dB}[n] = 10\log_{10}(E[n])\,, \quad (3.7)$$

where $E[n]$ is the energy/power of the signal, given by:

$$E[n] = \sum_{n=n_1}^{n_2} |S(n,k)|^2\,. \quad (3.8)$$

Therefore, the given output data of this block, at each frame, represents the power spectrum, or contribution of every frequency to the power of the original signal, for a given window.

- **(Spectral) Flux:** This block outputs the actual onset detection function, which is given by eq. (2.5). As observed, in Section 2.1.1.2, the *spectral flux* measures the change in magnitude in each frequency bin, *k*, given by the `PowerSpectrum`, restricted to the positive changes and summed across all frequency bins.

- **Derivation:** This functional block retrieves only the crescent `Flux` output to emphasize onsets rather than offsets. For this purpose, this block subtracts the *n* frame of the signal's *spectral flux*, *SF[n]*, to its *n-1* one, *SF[n-1]*:

$$Drv[n] = SF[n] - SF[n-1] \, . \tag{3.9}$$

- **SendRobot:** This last block comprises our peak-picking function and UDP client. It applies peak-picking with an adaptive thresholding algorithm that distinguishes three rhythmic events according to their magnitude, in the form of *Strong*, *Medium* and *Soft* onsets. These are sent to the *Robot Control Module* via UDP sockets.

  At time intervals, *i*, of 5 frames, the peak-picking algorithm looks for the highest onset detected so far, *mPP*, through the following function:

$$mPP[i] = max\,(mPP[i-1], Drv[n]) \, . \tag{3.10}$$

  Due to the potential inconsistencies in the beginning of some music data, the calculation of the first *mPP* waits approximately 2.5s from the beginning of the musical analysis before starting to be computed.

  In order to distinguish the three referred rhythmic events, the integrated adaptive thresholding algorithm, *SR(x)*, is defined as follows:

$$SR(x) = \begin{cases} Strong & , \quad \text{if} \quad x > \delta_3 \\ Medium & , \quad \text{if} \quad \delta_2 < x < \delta_3 \\ Soft & , \quad \text{if} \quad \delta_1 < x < \delta_2 \\ Silence & , \quad \text{if} \quad x \leq \delta_1 \end{cases} \quad , \tag{3.11}$$

where, $$\begin{cases} \delta_1 = thres_1 * PP[i] \\ \delta_2 = thres_2 * PP[i] \\ \delta_3 = thres_3 * PP[i] \end{cases}, 0 < thres_l < 1 . \quad (3.12)$$

The values of $\texttt{thres}_1$, $\texttt{thres}_2$, and $\texttt{thres}_3$, are constants which can be flexibly assigned through the system's user interface (see Section 3.3).

- **AudioSink:** This class consists of a real-time *audio sink* based on RtAudio, a Microsoft produced adaptive wide-band speech codec. It is responsible for sending its output to the speakers in order to reproduce the given audio file, as music.

Given this decomposition, some last considerations shall be presented in respect to classic decomposition of audio note-onset algorithms (see Section 2.1):

- **Pre-Processing:** The pre-processing (see Section 2.1.1.1) of our onset detection scheme consists of the series composed of the following blocks: `Stereo2Mono`, `ShiftInput`, `Windowing`, `Spectrum`, and `PowerSpectrum`. These processes properly prepare the analyzed audio signal to be processed by the implemented s*pectral flux* onset detection function.

- **Onset Detection Function:** This function (see Section 2.1.1.2) is fully determined by the `Flux` functional block, which measures the variation of the energy between consecutive frames in order to detect the required onsets.

- **Post-Processing:** The post-processing (see Section 2.1.1.3) is represented by the *Derivation* and `SendRobot` blocks, which are responsible for selecting the onsets from the former onset detection function. It consists of a peak-picking algorithm, which finds local maxima in the detection function; and an adaptive thresholding

algorithm, which defines the respective levels of magnitude of the detected onsets according to $\texttt{thres}_1$, $\texttt{thres}_2$, and $\texttt{thres}_3$.

The proper calibration of the *Music Analysis Module*'s parameters, namely the `WinSize`, `HopSize`, `thres1`, `thres2`, and `thres3`, shall be discussed in the next chapter through experimentation and discussion of the respective results.

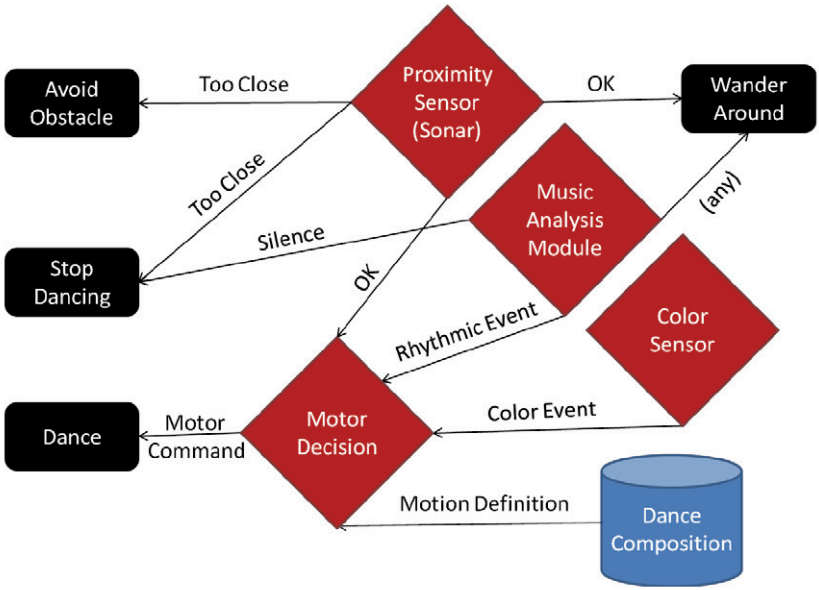


Figure 29 – *Robot Control Module*'s dancing control decision algorithm.

3.3.2 Robot Control Module

Rhythm is the key component that forms the symbiotic relationship between dance and music, dating back to prehistoric times. Body movements and music are closely linked in a dynamic relationship between acting and listening, cause and effect (Lee, Enke, Borchers, & de Jong, 2007).

In order to replicate such reactive behavior, the implemented *Robot Control Module* is responsible for controlling the robot to perform dance movements in musical-synchrony to the detected rhythmic events received

from the *Music Analysis Module* on-the-fly, and variably according to the received sensorial events and pre-defined dance compositions (see Section 3.3.3).

In Figure 29, we present a flow chart describing the implemented robot dancing decision algorithm.

As observable, the robot's body movement reacts to a conjunction of stimuli formed by four rhythmic events, namely: *Low*, *Medium*, *Strong* onsets, or *Silence*; and two types of sensorial events defined by the detected color: *Blue*, *Yellow*, *Green*, *Red*; and by the proximity to an obstacle: *Too Close*, *OK(Distance)*.

In order to assure the color variation during dance performance, while enforcing a variable behavior, the robot wanders around the dance floor while performing its pre-assigned dance movements.

All the dance movements, defined for each conjunction of rhythmic plus sensorial events, as the velocity of their execution, can be flexibly assigned in the *Human Control Module*, through a proper user interface (see Section 3.3.3). Each full dance composition can be saved in a proper XML file (see Appendix B).

The bi-directional interaction between this module and the robot is achieved via Bluetooth, thanks to the NXT Remote API (see Section 1.3.5) through the following classes (see Figure 30):

- **Serial Class:** This class is responsible for the Bluetooth communication. It ensures the connection (through the `connect()` function) between this module and each of the robot's NXT bricks, through pre-defined Communications Serial Ports (COM).

- **Brick Class:** Through this class we can retrieve and set any information related to the NXT brick (`name`, `battery level`, `firmware version`, and `start` or `stop` programs). This class was used to retrieve the battery level (`battery_level()` function) in order to check (and consequently assure) the correct connection to each brick and the Bluetooth connection state.

- **Motor Class:** This class is responsible for controlling every robot's motors (i.e., the robot's actuators). It sets the defined motor direction and speed (with the `on(speed)` function) and can retrieve the number of performed rotations (with `get_rotation()`).

- **Sonar Class:** This class is the sensor class for 9 Volts sensors like the ones we used (i.e., a color sensor and an ultrasonic sensor). It retrieves the values given by each sensor (with the `distance()` function) in the chosen scale (centimeters for the ultrasonic distance and color reference for the detected color). The ultrasonic values are given from 0-255 cm and the color values from 0-17 (see Appendix A for the color number chart).

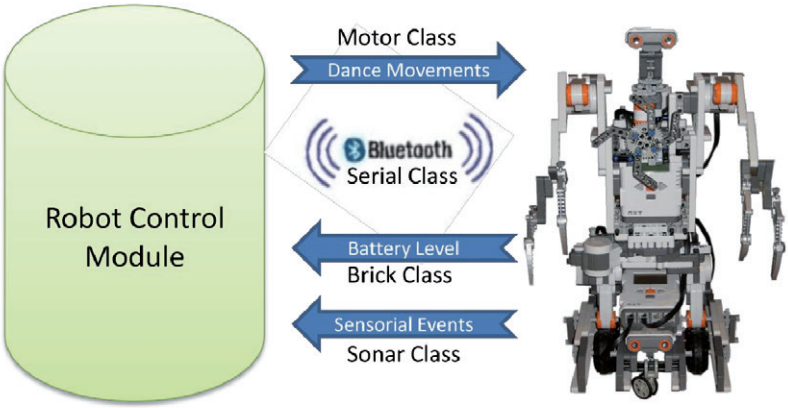


Figure 30 – Bi-directional communication between the *Robot Control Module* and the robot, through four NXT Remote API classes: `Motor`, `Serial`, `Brick`, and `Sonar`.

3.3.3 Human Control Module

The *Human Control Module* is positioned on top of the whole system giving the user a higher flexible control over the robot dancing behavior. This module consists of a user graphical interface composed of two blocks: a *Robot Control Panel* and a *Dance Composition Menu*. The *Robot Control Panel* (see Figure 32) is a user-definable control interface where one can set the Bluetooth and sensorimotor connection ports, with one or two NXT

bricks; pick the audio file to be analyzed and reproduced, and define the correspondent music analysis parameters (which can be possibly saved in a proper text file). The *Dance Creation Menu* (see Figure 31) enables the user to flexibly define each individual dance movement in correspondence to a given rhythmic and color event; as well as their velocity of execution: *High*, *Medium*, *Low*, *None*. The resulting dance can be saved in a proper XML file and imported into the system *a posteriori*. Hence, the user has some control over the whole system's behavior by flexibly defining the robot choreography, through a set of dance movements to be executed during performance; by selecting the audio data to be reproduced and analyzed; and by setting the threshold parameters for calibrating the music analysis. In addition, we included a real-time plotting interface (based on MATLAB) that enables the visualization of the detected note-onsets on-the-fly for the proper calibration of the music analysis.

The implemented GUI, and its interaction with the former modules, is respectively represented in Figure 31, Figure 32, and Figure 33. In consideration to them, Table 2 briefly describes each control component marked in Figure 31 and Figure 32.

Figure 31 - Dance Creation GUI.

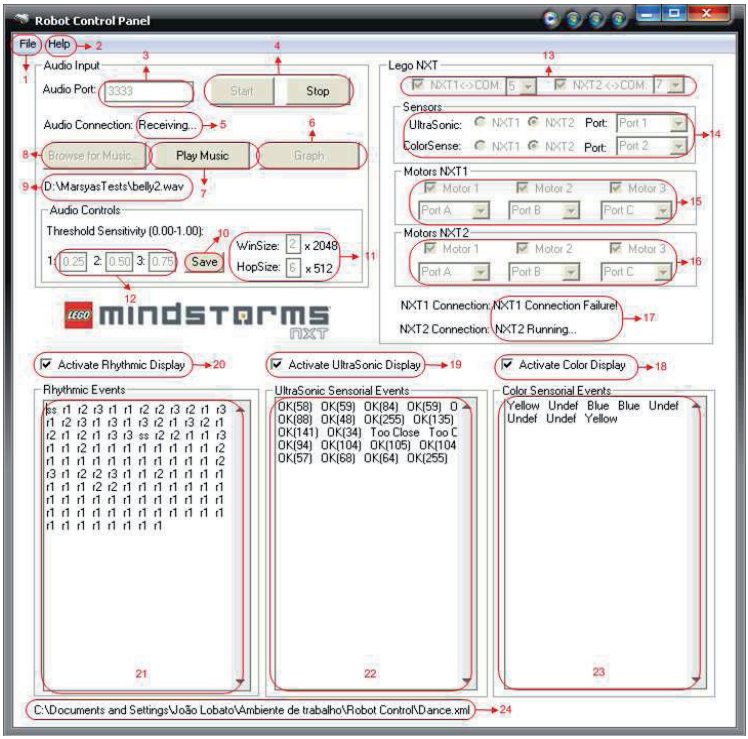


Figure 32 – Robot Control Panel GUI.

Table 2 - *Human Control Module* GUI: components description.

Nr.	Control Component	Description
1	**Control Panel File Menu**	This menu is decomposed in three separators: *Load Dance File*, which opens the Windows explorer in order to choose and subsequently load a created XML dance file; *Dance Creation…*, which opens the Dance Creation interface; and *Exit*, to quit the application.
2	**GUI Help Menu**	This menu contains the *About…* separator which presents some information about the author and the software version.

3	**Audio Port**	In this text box the user can define the UDP client socket port opened by the correspondent socket server (i.e., the *Music Analysis Module*), in order to transmit the audio parameters and receive the given rhythmic events. By default it was defined as 3333.
4	**Start/Stop Buttons**	These buttons are responsible for starting or stopping the robot control. The start button initializes the UDP connection with the *Music Analysis Module* and the Bluetooth connection with each NXT brick, loading every defined parameter. When ready, the system waits for the selection of *simulation* or *performance* mode of operation (by pressing the respective button – see below). The *stop* button stops the robot (if it was dancing), stops the connection with the *Music Analysis Module*, and clear all previously defined parameters.
5	**Audio Connection Status**	This text box shows the status of the UDP connection with the *Music Analysis Module*.
6	**Graph Mode (Simulation)**	This button allows for the simulation of the onset detection function, given the assigned parameters. This simulation is represented through a MATLAB plot (see Chapter 4) which points out the peak-picking of each detected rhythmic event (*Soft*, *Medium*, or *Strong* onset).
7	**Musical Mode (Performance)**	This button activates the actual performance of the robot, through its pre-defined dance movements in reaction to each individual conjunction of rhythmic and sensorial events.
8	**Audio (WAV) File Explorer**	This button opens the Windows explorer in order to choose the intended audio WAV file, to be analyzed and parallelly reproduced.

9	**Audio (WAV) File Path**	This text box shows the path of the chosen audio file.
10	**Save Audio Parameters Text File**	This button saves the audio thresholding parameters (`thres`$_1$, `thres`$_2$, and `thres`$_3$ — see Section 3.3.1), defined manually in (12), in a proper text (.txt) file, with the name of the audio file. Each audio file should have its own parameters' text file.
11	**`WinSize` and `HopSize` Parameters**	These text boxes show the defined values of `WinSize` and `HopSize`, to be sent to the *Music Analysis Module*, and allow their manual manipulation.
12	**Thresholding Parameters**	These text boxes show the loaded values of `thres1`, `thres2`, and `thres3` (from the correspondent audio parameters text file), to be sent to the *Music Analysis Module*, and enable their manual manipulation.
13	**Lego NXT Bricks (Check + BT COM Port)**	In this area the user can check the state of connection to each NXT brick, and define their correspondent Bluetooth serial COM port.
14	**Sensors Control (Brick + Port)**	In this area the user can define the sensors connected to each NXT brick and the correspondent NXT ports on which the sensors are connected.
15	**Motors Control Brick 1 (Check + Ports)**	In this area the user can check which motors, from NXT brick 1, shall be controlled, and define the ports to which they are connected.
16	**Motors Control Brick 2 (Check + Ports)**	In this area the user can check which motors, from NXT brick 2, shall be controlled, and define the ports to which they are connected.
17	**Lego NXT bricks Connection Status**	These text boxes show the current connection status to each defined NXT brick.

18	**Color Display Activation**	This checkbox activates or deactivates the display of the detected color events, received from the color sensor.
19	**UltraSonic Display Activation**	This checkbox activates or deactivates the display of the ultrasonic events, received from the ultrasonic sensor.
20	**Rhythmic Display Activation**	This checkbox activates or deactivates the display of the detected rhythmic events, received from the *Music Analysis Module*.
21	**Rhythmic Display**	If the `Rhythmic Display Activation checkbox` is activated, this text box displays the received rhythmic events: `r1` (Soft), `r2` (*Medium*), `r3` (*Strong*), `ss` (*Silence*).
22	**UltraSonic Display**	If the `UltraSonic Display Activation checkbox` is activated, this frame displays the received ultrasonic events: `Too Close`, `OK(Distance)`.
23	**Color Display**	If the `Color Display Activation checkbox` is checked, this text box displays the received color events: `Blue`, `Yellow`, `Red`, `Green`.
24	**Dance File Path**	This text box shows the path of the chosen XML dance file.
25	**Dance Creation File Menu**	This menu is decomposed in three separators: *New*, which resets the *Dance Creation* interface to the default values; *Load Dance File…*, which opens the Windows explorer in order to choose and subsequently load a previously created XML dance file, to be further analyzed or altered; and *Save* and *Save As…*, to respectively save the created dance in the opened dance file or in a new one.
26	**Dance Movement**	To choose the intended dance movement from

	ComboBox	the list of movements (with 14 distinct movements – see Section 3.1).
27	**Movement Speed ComboBox**	To define the velocity of execution of the correspondent dance movement, from four levels: `High`, `Medium`, `Low`, or `None` speed.

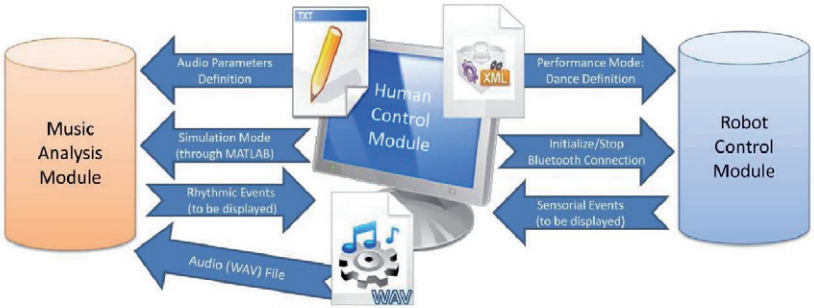


Figure 33 – *Human Control Module* bi-directional interaction with the *Music Analysis Module* and the *Robot Control Module*.

3.4 Conclusions

This chapter described our robot dancing system, the designed dance environment and described the dancing control architecture, depicting each of its constituent modules.

Given this system's architecture, in the next chapter we present some experiments and its correspondent results in order to calibrate the implemented real-time onset detection function. We conclude the next chapter, by reporting on an empirical evaluation by a group of students over the overall robot dance performance after a set of live demonstrations.

Chapter 4

Experiments and Results

Our experiments focused on performance and efficiency tests related to the integrated real-time note-onset detection and on an empiric evaluation of the robot dance performance. These experiments are described in this chapter through two distinct sections: Section 4.1 – *Real-Time Note-Onset Detection Calibration* and Section 4.2 – *Assessment of the Robot Dance Performance*.

4.1 Real-Time Note-Onset Detection Calibration

In this section, we describe the calibration made to the implemented real-time note-onset detection algorithm (see Section 3.3.1). We decomposed these experiments in two sub-sections: Section 4.1.1 – *Note-Onset Detection Post-Processing* and Section 4.1.2 – *Thresholding Parameters Settings*. All tests were performed with aid of the system's *simulation mode* (see Section 3.3.3), thanks to Marsyas' MATLAB engine capabilities.

4.1.1 Note-Onset Detection Post-Processing

In order to smooth the response of the implemented onset detection function down to the main onsets, we experimented applying a Butterworth

low-pass filter to the `Flux` output, tested with different coefficient values (see Figure 34 – `Filter` block).

Digital Butterworth are FIR (Finite Impulse Response) filters characterized by a magnitude response that is maximally flat in the pass-band and monotonic[15] in the overall. These filters sacrifice roll-off steepness for monotonicity in the pass- and stop-bands, being essentially smooth.

The gain, *G(ω)*, of an *n*-order Butterworth low pass filter is given in terms of its transfer function, *H(s)*, (output-input ratio, $\frac{V_o(s)}{V_i(s)}$, where $s=j\omega$):

$$G^2(\omega) = |H(j\omega)|^2 = \left|\frac{V_o(j\omega)}{V_i(j\omega)}\right|^2 = \frac{G_0^2}{1+\left(\frac{\omega}{\omega_c}\right)^{2n}} , \tag{4.1}$$

where *n* is the order of the filter; ω_c the cutoff frequency[16], which must be a number between 0 and 1 (1 corresponds to the Nyquist frequency, π radians per sample); and G_0 is the DC (Direct Current) gain (gain at zero frequency).

Marsyas processes this filtering through a generic filter transfer function defined by the coefficients, *b* and *a*, with the length of *n+1* row vectors, and coefficients in descending powers of *z*, as follows:

$$H(z) = \frac{V_o(x)}{V_i(x)} = \frac{b(1)+b(2)z^{-1}+\cdots+b(n+1)z^{-n}}{1+a(2)z^{-1}+\cdots+a(n+1)z^{-n}} . \tag{4.2}$$

In order to retrieve the correspondent Butterworth coefficients (a_i, b_i), to each chosen *n* and ω_c values, we used the MATLAB `butter(n,ωc)` function.

It can be seen, as shown in Figure 35a), that as *n* approaches infinity, the gain becomes a rectangle function and frequencies below ω_c will be passed

[15] Monotonicity is a property of certain types of digital-to-analog converter (DAC) circuits. In a monotonic DAC, the analog output always increases or remains constant as the digital input increases. Monotonicity is an important characteristic in many communication applications where DACs are used.

[16] Cutoff frequency is that frequency where the magnitude response of the filter is $\sqrt{1/2}$.

with gain G_0, while frequencies above ω_c will be suppressed. For smaller values of n the cutoff will be less sharp (Wikipedia, 2008).

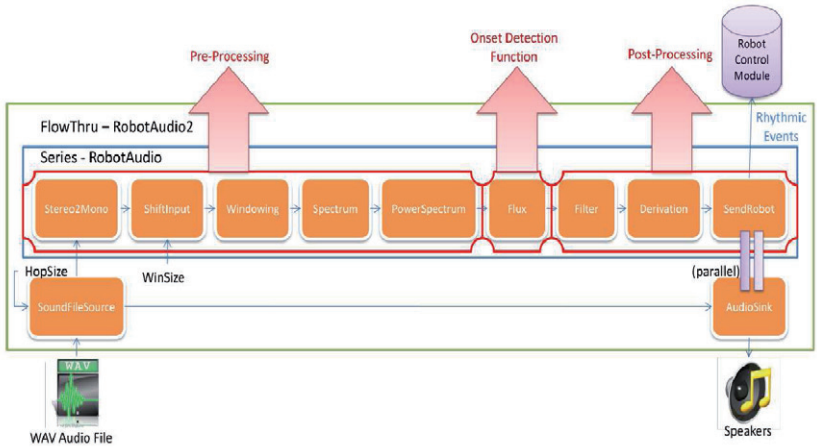


Figure 34 – Onset detection model in Marsyas with filtering (`Filter` block).

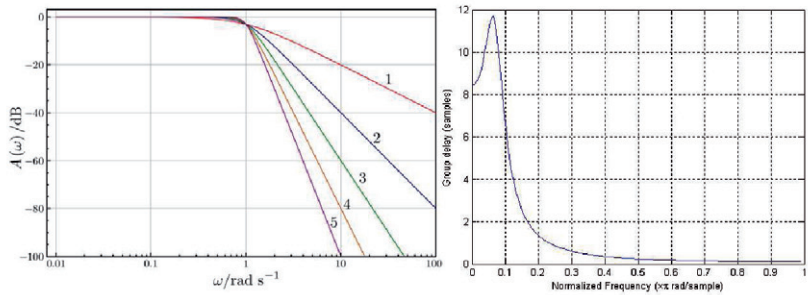


Figure 35 – Butterworth low-pass filter (Wikipedia, 2008): **a)** gain plot for $n = 1$ to $n = 5$. Note that the slope is 20n dB/decade where n is the filter order [left]; **b)** group delay of a third order filter (i.e., $n = 3$) with $\omega_c = 0.075$ [right].

Considering these definitions, in Figure 36 we present the `Filter` block output results achieved for a different group of coefficient values, in response to the same musical input.

As observable, by increasing n and decreasing the cutoff frequency, ω_n, the signal became smoother, starting to salient only the main onsets (i.e.,

signal peaks). However, this generated a group delay[17] (see Figure 35b)) that increased with the increase of this smoothing. The minimum acceptable coefficient values ($\omega_c = 0.075$ and $n = 3$ — see Figure 36b)) created a delay of almost 12 frames, which corresponds to approximately 0.8 seconds, due to fs_{Flux} = 14.36 Hz. In addition to the whole process's natural time consumption, this represents a considerably high delay facing the requirements.

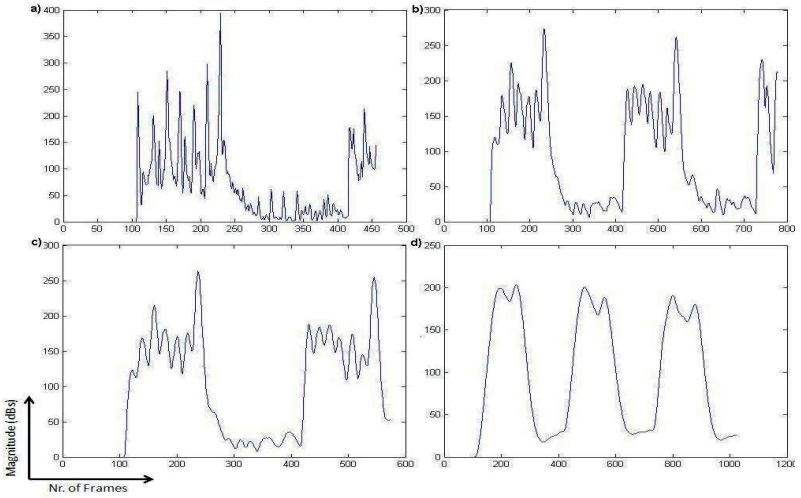


Figure 36 – Butterworth low-pass filter output for different coefficient values: **a)** $\omega_c = 0.28$ and $n = 2$; **b)** $\omega_c = 0.075$ and $n = 3$; c) $\omega_c = 0.075$ and $n = 4$. d) $\omega_c = 0.02$ and $n = 4$.

In a way to bypass this issue we decided to substitute the filter with a slight increase of the window size and hop size (i.e., `WinSize` and `HopSize`). By experimenting different pairs of values, in response to the same musical input, as shown in Figure 37, we agreed to set these parameters to `WinSize` = 4096 and `HopSize` = 3072. Although it obscures some signal content underneath, these parameters provide more efficient onset detection with no delay imposed in the process. These can

[17] The group delay is defined as the derivative of the phase with respect to angular frequency and is a measure of the distortion in the signal introduced by phase differences for different frequencies. It can be seen that there are no ripples in the gain curve in either the pass-band or the stop-band.

be further manually changed through the system's user interface (see Section 3.3.3).

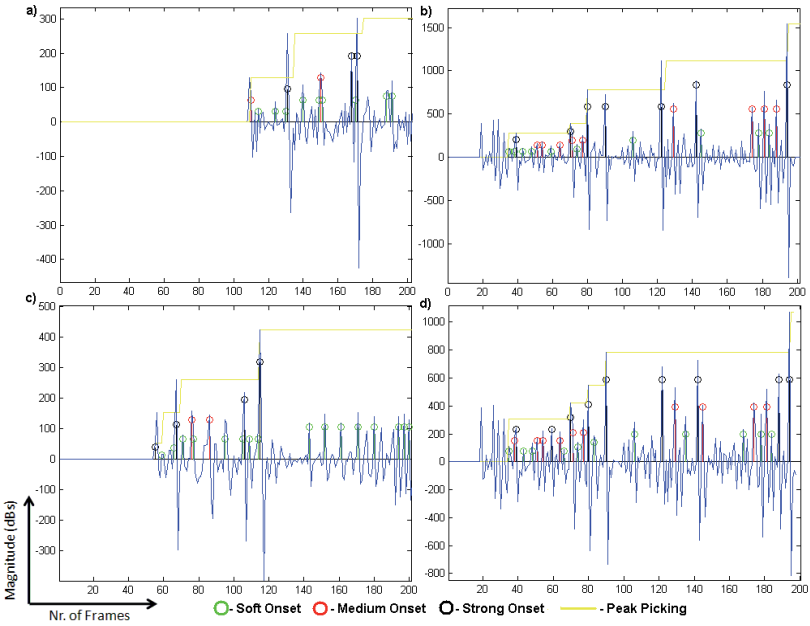


Figure 37 – *Music Analysis Module* output for different pairs of win size and hop size values: **a)** `WinSize` = 2048 and `HopSize` = 512; **b)** `WinSize` = 2048 and `HopSize` = 3072; **c)** `WinSize` = 4096 and `HopSize` = 1024; **d)** `WinSize` = 4096 and `HopSize` = 3072.

4.1.2 Thresholding Parameters Settings

For the sake of more meaningful and uniform (and eventually autonomous) definition of the pick-peaking threshold parameters, we perform a set of tests with diverse music styles. Based on (Bello, et al., 2005), we tested different music styles according to a range of musical instruments classed into the following four groups (see Section 2.1.3.2): NP, PP, PN, and CM. Due to the inherent differences of the four considered instrumental classes, we were compelled to consider different thresholding parameters for each class of musical piece in order to normalize the onset detection accordingly. By taking advantage of the system's *simulation mode* (see Section 3.3.3), Table 3 presents the optimal threshold values

achieved for each of the considered music styles. Figure 38 depicts the resultant onset detection for an excerpt of each tested instrument class.

As described in Section 3.3.3, all these parameters can be individually saved in a dedicated text file for each tested musical piece.

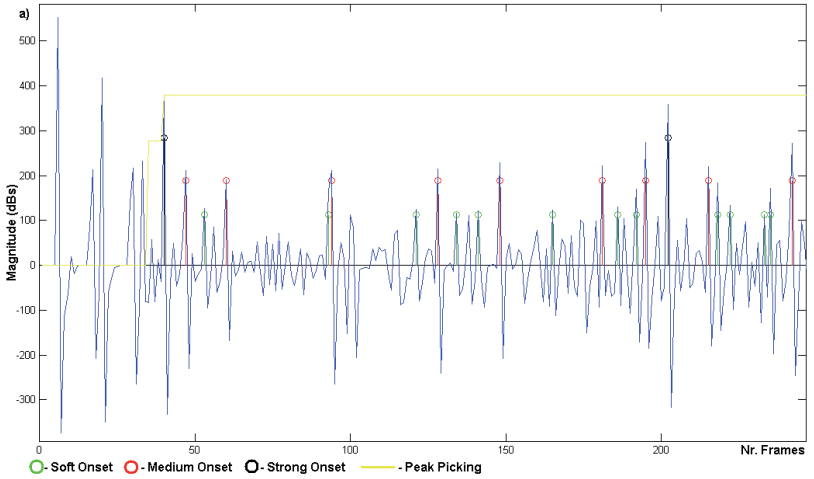
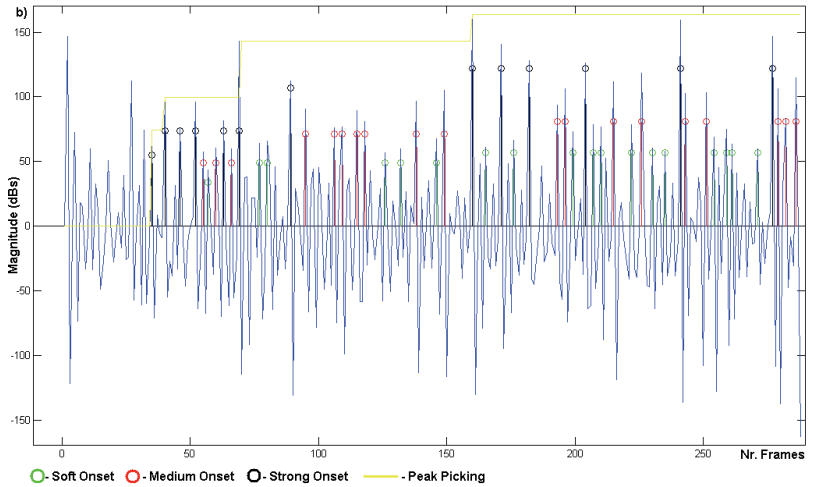
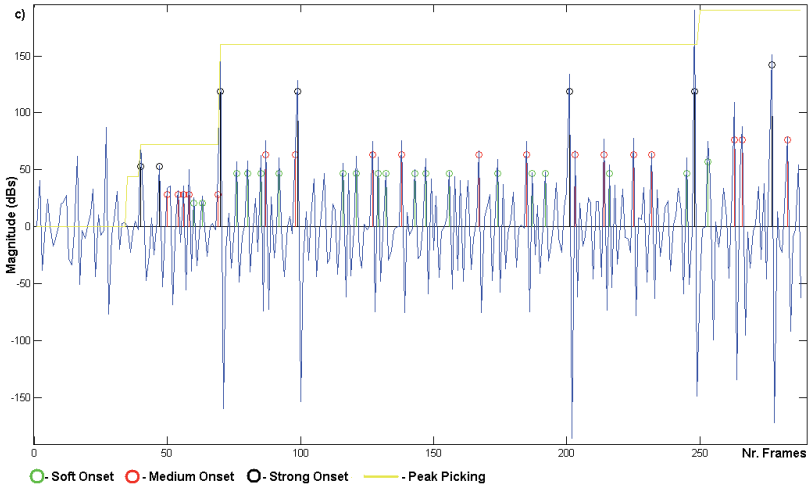


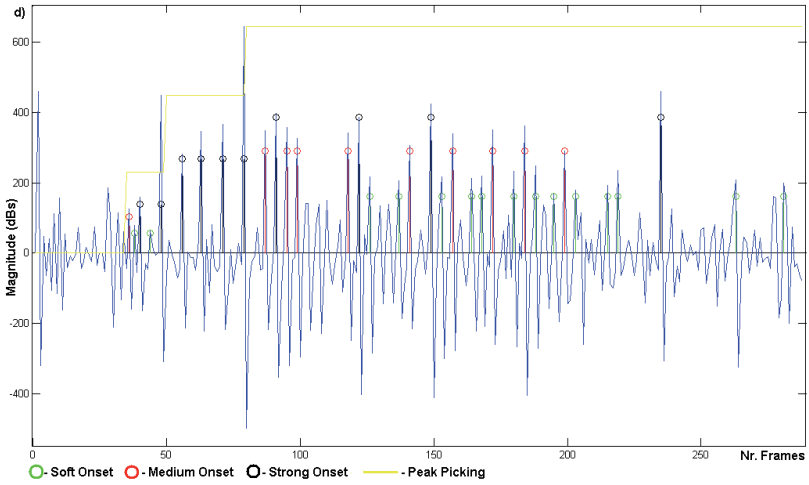
Figure 38 - Output of the implemented onset detection set with different thresholding parameters, in response to excerpts of different instrument classes: **a)** PN excerpt using $\texttt{thres}_1 = 0.15$; $\texttt{thres}_2 = 0.40$; $\texttt{thres}_3 = 0.70$.



b) PP excerpt using $\texttt{thres}_1 = 0.25$; $\texttt{thres}_2 = 0.50$; $\texttt{thres}_3 = 0.75$.



c) NP excerpt using $\texttt{thres}_1 = 0.25$; $\texttt{thres}_2 = 0.50$; $\texttt{thres}_3 = 0.75$.



d) CM excerpt using $\texttt{thres}_1 = 0.35$; $\texttt{thres}_2 = 0.65$; $\texttt{thres}_3 = 0.80$.

Table 3 – Optimal onset detection parameters.

Music Style	$\texttt{thres}_1$	$\texttt{thres}_2$	$\texttt{thres}_3$
PN	0.15	0.40	0.70
PP	0.25	0.50	0.75
NP	0.25	0.50	0.75
CM	0.35	0.65	0.80

4.2 Assessment of the Robot Dance Performance

For evaluating the resulting robot dance performance we based its assessment on live empiric observation (Oliveira, Reis, Faria, & Gouyon, 2012). For this purpose, we considered a student population constituted by 254 individuals, 118 girls and 136 boys, with ages comprising 6 to 17 years old. The focus on a young audience composed of children and teenagers was demanded by the educational and entertaining applications of our dancing robotic system. For such evaluation, we performed several demonstrations run during the Engineer Open-Week at FEUP, and at College Dom Diogo de Sousa, in Braga, during an open-session to aware students of the power of mathematics and its applications. This system was also exhibit in Portugal Tecnológico, a major technological event, where a variety of people, from all ages, also gave their feedback. In order to better demonstrate the adaption of the robot's dancing to the music, while enforcing the symbiosis with the public, different mainstream musical excerpts were chosen, and distinct dance compositions were defined *a priori* for each.

Figure 25 illustrates the real-world dance environment where the demonstrations took place. A video demonstration of the robot dance performance can be observed in (Oliveira, Reis, & Gouyon, 2008d).

For evaluating the quality of the robot's dancing and the system's overall performance each student fulfilled a Likert-scaled questionnaire (Likert, 1932) after observing one live demo of the robot dance performance. This questionnaire assessed the system in respect to the robot's *musical-synchrony*, its variety of movements (crucial to the performance's *variability*), its human characterization (crucial to the robot dancing's *animacy*), and about the flexibility of the user control over the system. Besides, the audience was also inquired about the potential application of such robotic system in educational settings, and about its degree of amuse. Namely, this questionnaire approached a set of three qualitative aspects of the robot dance performance. It objected the evaluation of *technical issues*, through the questions:

a) Was the robot dancing tuned to the music?

b) Does the robot show a good variety of movements?

c) Does the application supply a flexible control over the robot?

d) Does the robot resemble human behavior?

It objected the evaluation of the *system's potentiality in educational and entertainment applications*, through questions:

e) Was the robot dancing performance amusing?

f) This robot may have applications in education?

Finally, this questionnaire assessed the *student's appreciation of the dancing robot and performed demonstration*:

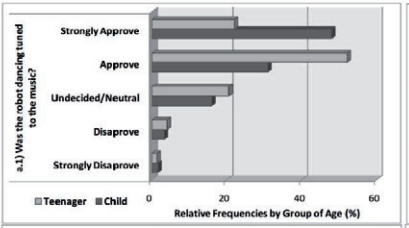
g) Do you like to own a Lego robot dancer?

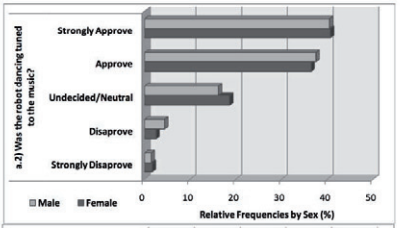
h) Did you like the robot dancing demo?

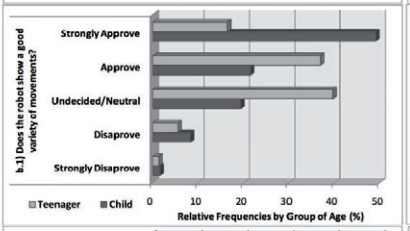
The first step of our analysis was to determine the correlations between the variables, measured through Spearman's Correlation. Next, we evaluated the association between them by recurring to the Chi-square test. Ultimately, we investigated if the distribution of the variables were significantly different by sex and age, using the Mann-Whitney test. Given the high statistical difference between the variables distribution, Figure 39 presents the relative frequency graphs for each question of the questionnaire, distributed by *Group of Age* (1 – left charts) and *Sex* (2 – right charts). A discussion of these results is presented in the following section.

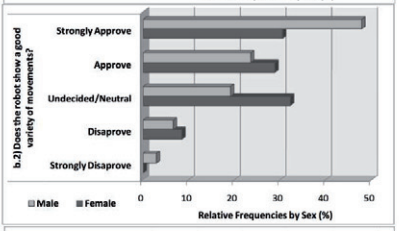
4.2.1 Discussion

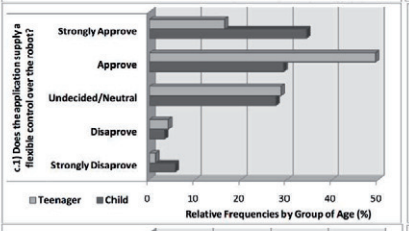
Using the Spearman's coefficient correlation we identified strong correlations between every pair of questions, getting the highest correlation, $r = 0.449$, between questions *e)* and *h)*, and the lowest, $r = 0.105$, with the pair *e)* and *d)*. To reveal statistical evidences on the differences of the answers' distribution by *Sex* and *Group of Age* we applied the Mann-Whitney test. The variable age was recoded into *child* ($\leq$ 11 years old) and *teenagers* ($\geq$ 12 years old) to establish a frontier between the youngest and the ones with higher maturity. The mean and standard deviation were, respectively, $x = 8.68$ and $s = 2.34$ for the child group, and $x = 14.49$ and $s = 2.37$ for the teenagers.

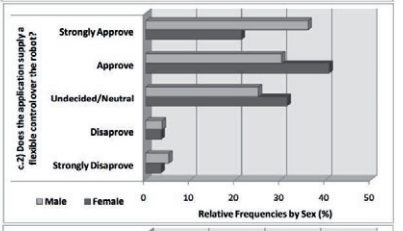

a.1) Was the robot dancing tuned to the music?
Strongly Approve
Approve
Undecided/Neutral
Disaprove
Strongly Disaprove
Teenager
Child
0
20
40
60
Relative Frequencies by Group of Age (%)

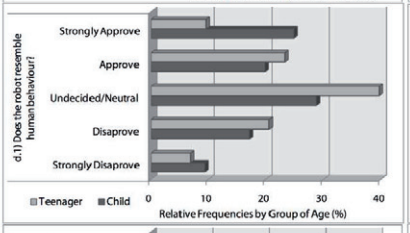

a.2) Was the robot dancing tuned to the music?
Strongly Approve
Approve
Undecided/Neutral
Disaprove
Strongly Disaprove
Male
Female
0
10
20
30
40
50
Relative Frequencies by Sex (%)

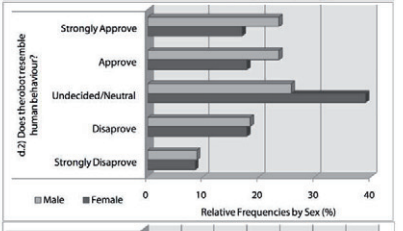

b.1) Does the robot show a good variety of movements?
Strongly Approve
Approve
Undecided/Neutral
Disaprove
Strongly Disaprove
Teenager
Child
0
10
20
30
40
50
Relative Frequencies by Group of Age (%)

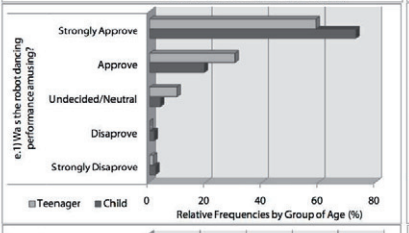

b.2) Does the robot show a good variety of movements?
Strongly Approve
Approve
Undecided/Neutral
Disaprove
Strongly Disaprove
Male
Female
0
10
20
30
40
50
Relative Frequencies by Sex (%)

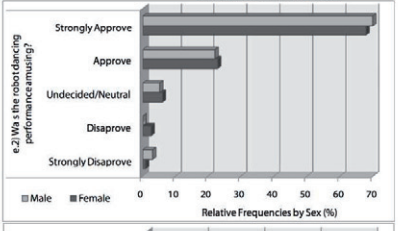

c.1) Does the application supply a flexible control over the robot?
Strongly Approve
Approve
Undecided/Neutral
Disaprove
Strongly Disaprove
Teenager
Child
0
10
20
30
40
50
Relative Frequencies by Group of Age (%)

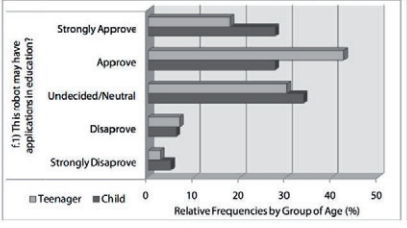

c.2) Does the applications supply a flexible control over the robot?
Strongly Approve
Approve
Undecided/Neutral
Disaprove
Strongly Disaprove
Male
Female
0
10
20
30
40
50
Relative Frequencies by Sex (%)

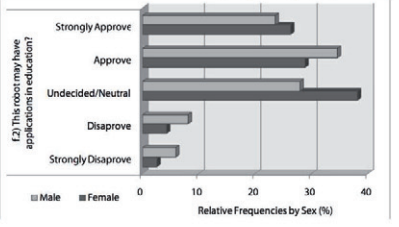

d.1) Does the robot resemble human behaviour?
Strongly Approve
Approve
Undecided/Neutral
Disaprove
Strongly Disaprove
Teenager
Child
0
10
20
30
40
Relative Frequencies by Group of Age (%)


d.2) Does the robot resemble human behaviour?
Strongly Approve
Approve
Undecided/Neutral
Disaprove
Strongly Disaprove
Male
Female
0
10
20
30
40
Relative Frequencies by Sex (%)


e.1) Was the robot dancing performance amusing?
Strongly Approve
Approve
Undecided/Neutral
Disaprove
Strongly Disaprove
Teenager
Child
0
20
40
60
80
Relative Frequencies by Group of Age (%)


e.2) Was the robot dancing performance amusing?
Strongly Approve
Approve
Undecided/Neutral
Disaprove
Strongly Disaprove
Male
Female
0
10
20
30
40
50
60
70
Relative Frequencies by Sex (%)


f.1) This robot may have applications in education?
Strongly Approve
Approve
Undecided/Neutral
Disaprove
Strongly Disaprove
Teenager
Child
0
10
20
30
40
50
Relative Frequencies by Group of Age (%)


f.2) This robot may have applications in education?
Strongly Approve
Approve
Undecided/Neutral
Disaprove
Strongly Disaprove
Male
Female
0
10
20
30
40
Relative Frequencies by Sex (%)

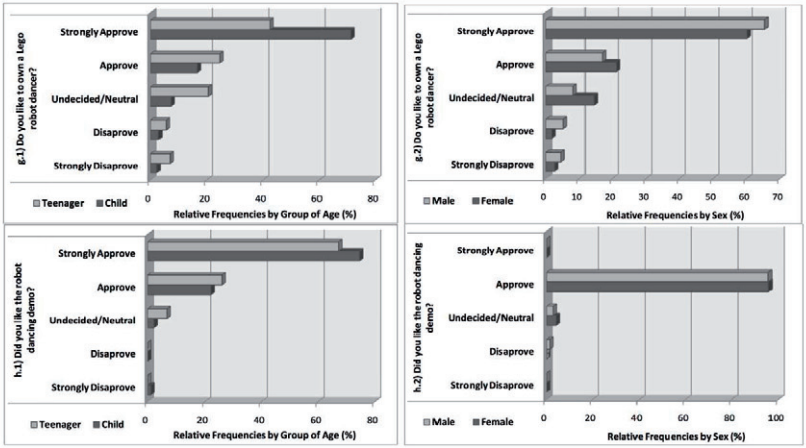


Figure 39 - Relative frequency graphs of the questionnaire's responses by *Group of Age* (1 – left charts) or *Sex* (2 – right charts).

In fact, the variability of the age is very similar in the two groups and the mean of ages are really apart. When analyzed by *Sex*, the resultant p-value was lower that the level of significance ($p = 0.05$) in question *b)*, with $p \approx 0.014$. This evinces a different distribution of the variable for males and females. This is supported by a graphical analysis of Figure 39b).2, which points for a more positively asymmetric distribution of males in comparison to females. This may be interpreted as a more positive attitude of males towards the variety of movements. When analyzed by *Group of Age*, the p-values were also lower than the level of significance for questions *a)*: $p \approx 0.002$; *b)*: $p \approx 0.000$; *e)*: $p \approx 0.038$; and *g)*: $p \approx 0.000$, separating them into two independent samples. In these cases there are statistical evidences to affirm that the distributions of the variables are different within the groups of children and teenagers. This points for the higher expectations of teenagers in comparison to young children, revealing that we need to enhance the variation of the robot moves for captivating older kids. Besides, the answers to these questions reveal that despite some system flaws and inconsistencies most of the subjects did not realize it, which suggests that strict musical-synchrony may not be so relevant for keeping an interesting dancing behavior.

On a global descriptive analysis, we may still infer some relevant conclusions. Question *d)* denotes higher frequency on undecided/neutral answers, revealing a relative frequency of 41.4% of answers with negative connotation. This may be implied by the aesthetics of the robot and its 360-degrees rotating movements, suggesting the need of replacing it with a different, more humanly shaped robot design. On the other hand, question *e)* reveals the great amusing potentiality of this framework, where 66.1% of the subjects strongly approved it, uniformly across all ages (within a total of 91.3% approvals). It is also interesting to notice that 81.9% approve or strongly approve the intention of acquiring a Lego robot dancer (see Figure 39g)), with a greater adherence of the male group.

Ultimately, we strongly believe that the robot aesthetics, dressed with a proper outfit (see Figure 23b)), along with its reactive strong moves was determinant for keeping an entertaining atmosphere. The artisanal aspect of the dance environment (see Figure 25) and the chosen music were also fundamental to keep the spectators' attention during the demonstrated dance performance. Ultimately, we may refer the ambiguity of question *c)*'s responses since the inquired did not have the opportunity to configure and control the system by themselves.

By corroborating the subjects' opinions with our personal overall assessment, we finalize our discussion by focusing on three requisites that we consider of most relevance for a meaningful and interesting robot dance performance:

- **Musical-Synchrony:** essentially due to processing and Bluetooth communication delays we verified some flaws in terms of musical-synchrony. The use of a multi-threading architecture granted the required simultaneity between the modules' processing but caused some musical-synchrony flaws due to race conditions in the processing of the dance movement decision mechanism. In terms of hardware-software communication, the Bluetooth interface had to constantly deal with communication overflows, as it can only receive/send data in time-intervals of approximately 50-100 ms while taking around 30 ms to transit from transmit mode (i.e., send motor data), to receive mode (i.e.,

receive sensor data). In addition to such limitations, the high number of detected onsets in many occasions surpassed the refresh rate of the robot's sensorimotor processing which also induced flaws by the fault of not executing the requested movement. Although all these flaws represent detachments with the music it enforces autonomy to the robot dancing behaviors that ultimately enhance the performance's variety, and consequently the interest to the spectators. These issues might be solved through the use of some kind of multithreading synchronization objects (e.g., critical sections, events, semaphores, or mutexes), which are used to protect memory from being modified by multiple threads at the same time. These would assure that each thread waits for the others when facing data dependence among the threads. Yet, this solution is impracticable due to the real-time requirements, which imposes that every action shall occur in a reactive manner, through a cause effect behavior (so music cannot wait for a dance decision, which on its hand cannot wait for communicating with the robot). A proper solution might then imply the use of a multi-processing architecture instead, or the use of more robust and advanced humanoid robot with an embedded CPU (Central Processing Unit), capable of higher clock rates for accompanying the music with sequenced dancing behaviors.

- **Variability:** the variability of our framework's architecture was granted by the great variety of possible dance style compositions (in a total of 15^{12}-1), formed by 14 distinct individual dance movements (plus *None*) distributed through 12 conjunctions of events (3 rhythmic events x 4 color events); and enforced by the robot's perambulation around the dance environment while avoiding its obstacles. This variable behavior is ultimately transposed to the human decision, which has the versatility to adapt the robot performance *a priori* through a flexible definition of the robot dancing behavior. Although, in theory, these characteristics enable a more varied behavior, the lack of individual dance moves, imposed by the robot's limited degrees-of-freedom, restricted the performance to repetitive dancing sequences, which only differ in

velocity of execution or orientation. Again, more variety of movements demands the use of a more articulated humanoid robot.

- **Animacy:** As the public suggested, although the robot's dancing was greatly inspired on human dancing, its performance is still far from being human representative, by presenting mainly 360-degree spinning moves. However, our robotic system was inspired on human behavior by interacting with the real-world in a reactive behavioral-basis that connects perception to action. Not unexpectedly, a robot dance performance comparable to human behavior is greatly dependent on the former two requisites (i.e., *musical-synchrony* and *variability*) and therefore also requires a more advanced humanoid robot. Yet, despite the undeniable improvements, such robot might bring to artificial dancing its rigid, strict mechanical moves, greatly attached to music, which may decrease the robot's animacy and consequently the spectator's interest.

In conclusion, despite some musical-synchrony issues, referred above, the robot seems to react in real-time to the external music and other external events while demonstrating reasonable variability and animacy. This suggests that a varied and flexible robot dancing behavior in compromise with a reasonable extent of musical-synchrony assures the required interest and entertaining relationship between the artificial agent and a human audience.

Chapter 5

Conclusions and Future Work

Designing entertainment systems that exhibit a dynamic compromise between short-term synchronization and long-term autonomous behavior is the key to maintain an interesting relationship between a human and an artificial agent, while sustaining long-term interest (Aucouturier & Ogai, 2007).

Based on this claim, we focused our efforts in the development of a user-customizable dancing robotic system which essentially rely on *musical-synchrony*, *variability* and *animacy* criteria, as described in Section 4.2.

In this chapter, in Section 5.1 – *Work Revision and Summary of Contributions*, we summarize our approach and its main results, presenting our contribution and a list of possible applications. Following, in Section 5.2 – *Future Work* we present our proposal to enhance this framework through further research towards human-interactive robot dancing.

5.1 Work Revision and Summary of Contributions

It seemed a reasonable requirement that intelligence be reactive to dynamic aspects of the environment, that a mobile robot operate on time scales similar to those of animals and humans, and that intelligence be able

to generate robust behavior in the face of uncertain sensors, an unpredictable environment, and a changing world (Brooks, 1991b).

In this research project, we developed a flexible framework for autonomous robot dancing applications. The proposed architecture was applied to a Lego NXT mobile humanoid robot that bodily reacts, in real-time, to multi-modal events by performing autonomous dance movements alternated through a variety of motion styles. We report on an empirical evaluation made by a group of students over the overall robot dance performance after a set of live demonstrations. The empiric evaluation validated our approach suggesting that, despite its limitations, the resulting dance shown a reasonable level of musical-synchrony with variable dancing sequences while interacting with the surrounding environment; besides providing flexible human control over the system's behavior. The discrepancies in the overall opinions between the younger and the older subjects indicate clear differences in their expectations, typically younger children being less critical about the robot's dancing behavior than the older subjects. Besides, giving the age and/or the unawareness of the audience about the technical issues underneath the system, their technical opinions may be quite optimistic. This also suggests that the robot's variety of movements and its physical aesthetics and outfit, interleaving from musically attached movements to others more freely executed (many due to system flaws), may have increased the audience's interest. By being more enthusiastic about the dance performance, the audience might have consequently ignored eventual flaws or unpredictable behaviors. Concerning the inquired genders, we could not realize relevant differences in opinion, except when inquired about the robot variety of movements (question *b)*), which pointed for greater approval by males.

In conclusion, the multidisciplinary concepts and the Lego foundations of the implemented robotic system, allied to an amusing aesthetics through dance performances interchanging musically-synchronous with variable dance movements validated the edutainment purposes of the proposed framework. It enforced the idea that designing robotic entertainment systems exhibiting such dynamic compromise between short-term

synchronization and long-term autonomous behavior might be the key to maintain the interest of the general audience.

5.1.1 Summary of Contributions

The proposed research and work can be used in different fields of application, which can be decomposed into six main areas:

- **Entertainment:** it is undeniable the increasing role of robotics in multidisciplinary entertainment areas and the great investment that is constantly made to improve the robots‘ entertaining capabilities, due to their potential help in people‘s life, being further enjoyable. In this context we intended to enhance robot dancing to a new level of expressiveness and captivation by focusing on high-level human control and variable dancing by reacting to multi-modal events.

  Specifically, we must refer some robot dancing events, namely RoboCup-Junior’s Dance (RoboCupJunior, 2008) (see Section 2.2), where the proposed architecture can be applied as a plausible framework.

- **Education:** from an educational point of view this framework may provide an intuitive environment for learners and children to experiment the creation of their own dancing behaviors, by generating robotic motion patterns in response to multi-modal external events. The approached multi-disciplinary project addressed multi-domain aspects such as *rhythm*, *dance*, *movement*, *robotics*, *computation*, and *human-robot interaction*, among others. We hope that through the medium of music and dance we can attract students from diverse backgrounds, who are not regularly drawn to fields such as mathematics, physics, computation, and robotics, allowing them to move across modalities and media, and between action and embodied expression.

- **Therapy:** art, dance, and music therapy are a significant part of complementary medicine in the twenty-first century. These creative art therapies contribute to all areas of healthcare and are present in

treatments for most psychological and physiological illnesses. The art therapies also contribute significantly to the humanization and comfort of modern healthcare institutions by relieving stress, anxiety and pain of patients and caregivers.

- **Research:** to the research community, our robotic system may provide a controlled environment to study the coordination and relations between body, brain, and musical environment. In such way, it may foster a playground for professional dancers and choreographers to test their dance movements' composition, and for researchers to experiment embodied perception theories.

5.2 Future Work

After analyzing and evaluating the lacks and limitations of the proposed robot dancing framework, this chapter ends this book by proposing and describing an improved robot dancing framework envisioning the interaction with humans in a non-verbal dance-based relationship founded on multi-modal environmental rhythms.

The proposed system may account for a proper architecture where a physical agent shall have multi-modal rhythmic knowledge for conveying proper dance motions towards interacting with human subjects. In order to achieve this kind of rhythmic interactional intelligence the conceived framework may be composed of a set of interconnected layers (depicted in Figure 40), built incrementally and maintained within a behavioral-based topology, concerning reaction and anticipation (Brooks, 1991a).

Each layer should consist of an activity-producing subsystem incorporating its own perceptual modeling, and planning requirements, individually connecting sensing to action, and parallelly processed, in a multi-tasking (ideally multi-processing) architecture, in order to process all modules with the required simultaneity.

This architecture must be conceived for generic robot dancing applications, being applicable to different humanoid robotic platforms. For

such, and following Figure 40, every constituent *Layer* may be described as follows:

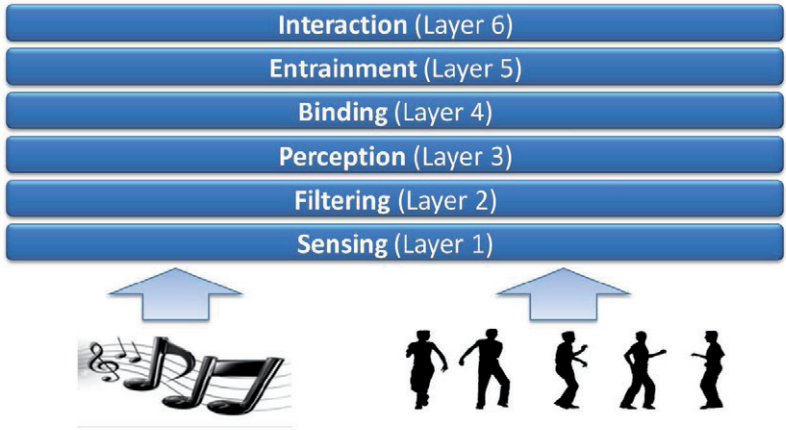


Figure 40 - Layered decomposition of the future work proposal.

- **Sensing – Real-Time Observation of Multi-Modal Rhythmic Stimuli (Layer 1):**

This layer is responsible for the acquirement of multi-modal environmental stimuli in the form of auditory and visual/spatial signals. Besides, it shall apply proper techniques for real-time tracking and pre-processing of these incoming data in frame sequences, within a rhythmical perspective.

In respect to vision, these techniques may comprise a series of possible real-time motion tracking methodologies such as the optical flow, based on the Lucas-Kanade algorithm (Lucas & Kanade, 1981); temporal templates methods, as the described by (Bobick & Davis, 2001); background subtraction algorithms, such as the proposed by (Stauffer & Grimson, 1999), for silhouette extraction and its division in sub-regions or definition of Silhouette Motion Images (SMIs); shape-based segmentation models, mostly founded on the human silhouette shape, as proposed by (Zhao & Thorpe, 2000); or even color-based or edge-based image registration techniques, such as the ones applied by (Jáuregui & Horain, 2009). For a

comprehensive review on vision-based human motion tracking and analysis see (Moeslund, Hilton, & Krüger, 2006).

As an alternative or complement to vision-based motion acquisition, we may use accelerometers embedded in the body of the interactors for retrieving motion onsets compiled in body rhythmical patterns (see e.g., (Enke, 2006)).

This layer shall then output images and body trajectories along with pre-processed audio signals, for the proper extraction of relevant rhythmic features.

- **Filtering – Extraction of Relevant Low-Level Features (Layer 2):**

This layer comprises the extraction of the most relevant cues for conceiving a coherent rhythmic perception. Respecting to vision, these cues may describe the energy of each movement (Quantity of Motion (QoM)), as an overall measure of the amount of detected motion, involving velocity and force; or related to the body contraction/expansion within the surrounding environment, both proposed by (Camurri, Lagerlof, & Volpe, 2003). Additional cues account for the 2D centre of mass, retrieved from the body silhouette with central moments. The variation of these cues may depend on motion trajectories in respect to their length, direction, and dynamic models.

In the presence of accelerometers, a rhythmic spatial analysis may account for the extraction of relevant low-level features such as the alternation of bulges, for detecting singular movements; accentuation variations, by analyzing maximum magnitudes and covered areas; and by denoting singular acceleration, counter-acceleration and deceleration peaks.

Regarding audio signals, the extraction of rhythmically relevant cues account essentially for accentuation filter-banks (Scheirer, 1998) or on methodologies for detecting note onsets, through temporal, spectral, or combined features. For a comprehensive review on note onset detection see Section 2.1.

Additional rhythmical cues may be given by simple statistical measures directly retrieved from the visual/spatial and auditory signals.

- **Perception – Appliance of High-Level Rhythmic Computational Models (Layer 3):**

This layer comprises a series of higher level computational models for inferring the desired multi-modal rhythmical information.

In respect to the auditory analysis, this layer may apply a noise-robust causal and real-time computational audio beat tracker, founded on the previously onset detection function, for beat prediction, and as a tempo descriptor for sonic interactive applications. For future references on this subject see (Oliveira, Davies, Gouyon, & Reis, 2012b), (Oliveira, Ince, Nakamura, & Nakadai, 2012c), and (Oliveira, et al., 2012d).

Considering the visual analysis, some motion signal processing techniques (Bruderlin & Williams, 1995) may be used in order to find distinctive directional changes periodically occurring in real motions, as proposed by (Kim, Park, & Shin, 2003). In a visual perspective, this layer is responsible for segmenting the individual motion computed from the formerly retrieved cues and applying statistical and Bayesian methods for measuring dance periodic cycles defined by repeated transitions of prominent stop-motions. Such phase and periodicity analysis may be also processed from the retrieved motion features by recurring to the real-time beat tracker described above. This kind of rationale may even be applied to motion onsets retrieved from accelerometer data.

- **Binding – Correlating Auditory and Visual Information (Layer 4):**

Integration and synchronization between audio and video signals is essential to multi-modal audio-visual applications, due to their strict timing constraints.

This layer is so responsible for fusing the musical beats with the motion metrical structure. For this purpose, non-linear signal matching procedures, denominated “dynamic time-wrapping” methods, can be applied in order to identify a combination of expansion and compression which can best “wrap” the two discrete signals together (Bruderlin & Williams, 1995). This problem may be solved by combining the optimal sample correspondences between the two signals, and by applying the wrap which forms the discrete, point-sampled correspondence that minimizes the

"difference" of the two signals. An exemplar method is constituted by the Dynamic Programming (DP) matching, which matches features points (e.g., beats) extracted from the auditory and the motion signal by time-warping and synchronizing both signals (Lee & Lee, 2005).

The processing of multi-modal features may be also contemplated by the integrated beat tracking system (in *Layer 3*) by considering both signals and finding the most salient periodicities among them, or by correlating each other in order to disambiguate metrical ambiguity.

The different modalities may even generate motion reactions from distinct body parts.

- **Entrainment – Sensorimotor Synchronization and Musical Expression (Layer 5):**

This layer must treat the rhythmic coordination between perception and action while embodying human-like dancing gestural patterns, previously synthesized from motion captured data. This musically-synchronous behavior must consider and deal with psychological/cultural issues, such as intension, variability, and disambiguation; and mechanical constraints, such as motor rate limits, balancing, and limited degrees-of-freedom (Repp, 2005).

The generation of human-inspired rhythmically meaningful gestural patterns may depend of two interdependent tasks: automatic dance key-poses' mapping from metrically segmented human motion data; and joint trajectories generation within key-poses, in rhythmic metrical cycled transitions (Oliveira, et al., 2012a). The former might be achieved by directly mapping the human body extremities (hands and feet) positions, of metrically segmented key-poses, onto the humanoid (this way mapping the spatial intentionality) and applying Inverse Kinematics (IK) to infer the remaining robot's joint angles within coupled body segments (arms and legs); intrinsically contemplating the robot's morphological constraints. The latter shall organize those key-poses at specific spatiotemporal beat-scaled points and must threat the interpolation (using simple spline functions, e.g., sine-interpolation) between them, in metrical closed-loops. In order to increase stability and overcome biped balancing issues,

stabilization criteria, such as the Center of Pressure (CoP) (Goswami, 1999), or the Zero-Moment Point (Vukobratovic, Borovac, Surla, & Stokic, 1990), may also be applied to reduce the amount of moments generated by movements of the robot's upper body.

The envisioned methodology should follow the Topological Gesture Analysis (TGA), proposed by (Naveda & Leman, 2010). Their method analyses the dancing spatial reasoning of music and its immanent temporal organization (i.e., rhythm and meter), by projecting a sequence of musical features onto the three-dimensional spatial trajectories of the corresponding dancing patterns. By considering accumulations of topological points in delimited regions of space they decompose each dancing gestural pattern in discrete stop-motions segmented at ¼ beat subdivisions, from a 2-beat length scale (i.e., measure). Future work approaching this method have already been implemented and tested in (Oliveira, et al., 2012a) and (Sousa, Oliveira, Reis, & Gouyon, 2011).

Additionally, in order to reproduce the reciprocal and dynamical coupling between body and brain, perception and action, the multi-modal rhythmic model shall support online feedback control through which the robot may adjust the metrical level to its morphological naturalness, following the Dynamic Attending Theory (DAT) (Drake, Baruch, & Jones, 2000). This methodology shall insure a more natural dancing performance, overcoming the robot's limited motor rates, while disambiguating the variety of perceptive rhythmic patterns, by providing a rhythmic resonance model around the robot's "preferred" tempo (Santiago, Oliveira, Reis, Sousa, & Gouyon, 2012).

To sustain the variability of the performance, within repeated gestural patterns, the robot joint target angles may be injected with "controlled noise" correlated with the musical rhythmic salience, given by the energy of the occurring beat-onset.

Some combined techniques for robotic sensorimotor rhythmical synchronization have already been proposed. In order to overcome the robot joint limitations and keep the motion rhythmicity, (Shiratori, Kudoh, Nakaoka, & Ikeuchi, 2007) used temporal scaling techniques for attenuating high frequency components of captured human motion

components, as the musical tempo becomes faster; while preserving the extracted key-poses. This technique consisted of the use of a hierarchical B-spline interpolation for control frequency resolution, by only setting control points at desired temporal intervals, denoted by the musical tempo. This method also preserves motion acceleration continuity by iteratively applying constraints for optimizing the inter-joint kinematics of the robot's body and the robot balance.

Other techniques, more focused on legged motion and balance issues, used the captured motion data to specify a trajectory for the ZMP in the control system rather than using it explicitly as the desired joint angles (Nakaoka, 2003). This method modified the original motion in a way that the upper body is horizontally translated so that a pseudo ZMP follows the desired one. Since translating the upper body is an approximation of translating the whole body, applying this method must be iterated until translation converges. For interpolating the joints within each posture the authors additionally applied inverse kinematics techniques.

Alternatively, for keeping the sensorimotor synchronization, Yoshii *et al.* used a feedback step control method for adjusting the robot's step intervals tuned to the predicted beat times, while accounting for musical tempo deviations (Yoshii, et al., 2007).

Concerned with keeping the dance kinematic continuity, Ellenberg *et al.* considered the full dance as a state machine transiting from a sequence of motion states (Ellenberg, Grunberg, Oh, & Kim, 2008). This transitions consisted of the point-to-point interpolation between successive poses with recur to intermediate states to maintain stability, such as switching supporting legs, or walking forward. In order to keep synchrony with the musical beats, their beat tracker was optimized to predict beat times in the needed anticipatory fashion for generating the selected key-poses on beat-time.

- **Interaction – Interactivity and Improvisation (Layer 6):**

This layer might apply specific methodologies for improving the interactive and improvisational sense of the proposed architecture. This task may be implicitly comprised by the entrainment model itself

(described in *Layer 5*), which may grant the robot-human interactional synchrony while keeping an enduring dynamic relationship with "humanized" and varied robot dancing movements.

A real-time method, applied to dance interaction, was proposed by (Tanaka, Fortenberry, Aisaka, & Movellan, 2005), which advocated that for keeping a long-term interesting relationship between a robot and a human one must consider *sympathy* and *variation* factors within an imitational process. For this effect, the authors proposed an Entrainment Ensemble Model based on multiple entrainment factors (multiple ensemble rhythmic motions), accomplished by a naive rule-based structure and a recurrent neural network with parametric bias (RNNPB), both trained with observed rhythmic moving regions (exclusive visual observation). The level of imitation could be controlled by changing the order of strengths of the multiple entrainment factors.

Other robotic imitational methodologies keep the entrainment between interactors by recurring to similar models like non-linear oscillators (Ijspeert, Nakanishi, & Schaal, 2002) or other recurrent neural oscillators (Williamson, 1999).

In order to synthesize new motions based on existing motion capture data, some methods have been proposed for splicing collections of motion sequences in a directed graph (Arikan & Forsyth, 2002). In this kind of representation, each motion sequence becomes a node in the graph with an edge between nodes for every frame of one sequence, which can then be spliced to a frame in another sequence or itself. In order to constrain the motion sequences, a randomized search method can be applied for searching appropriate paths. By following *Layer 5*, some additional methods must then be used for keeping the kinematic continuity between motion segments. This can be achieved through transition graphs, representing a collection of rhythmic motions of an identical type, by traversing the movement transition graph from node to node (i.e., the conjunction of key-poses to be interpolated), guided by the transition probabilities, while synthesizing a basic movement at each node (Kim, Park, & Shin, 2003).

Above all the layers we shall integrate a proper user interface through which a human user can set the main definable parameters and control the overall behavior of the system, akin to the implemented in the robot dancing control system as described in this book.

The following Figure 41 illustrates the whole general idea, as a clarification of the proposed framework. It is observable the incremental capabilities of this system, achieved through the subsequent incorporation of each layer and their interconnection, while keeping the parallel processing in an ideal multi-processor architecture.

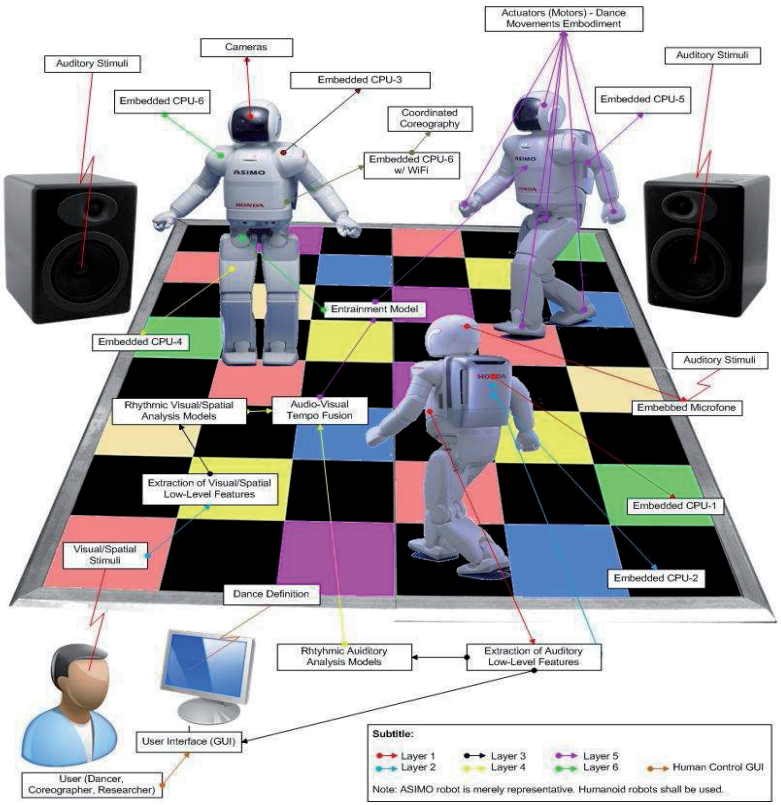


Figure 41 – Future work proposal, incorporating all the proposed layers and their interconnection.

Appendix A

Color Sensor

The NXT color sensor, designed by HiTechnic[18], operates by using three different color light emitting diodes to illuminate the target surface and measure the intensity of each color reflected by the surface. Using the relative intensity of each color reflection, the color sensor calculates a color number that is returned to the NXT program. The color sensor connects to an NXT sensor port using a standard NXT wire and uses the digital I^2C communications protocol. The color number calculated by the sensor is refreshed approximately 100 times per second.

Figure 42 presents the color number chart which shows the relationship between the target color and the color number returned by the color sensor.

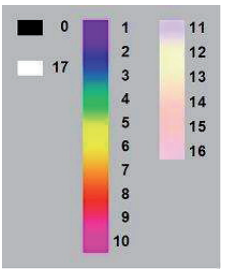


Figure 42 – HiTechnic's color sensor number chart.

[18] For more information visit http://www.hitechnic.com/.

Appendix B

XML Dance File Structure

In this appendix, we present the structure of the system's XML dance file, on which the dance choreographies created by the user can be saved, through the *Human Control Module*:

```
<?xml version="1.0" encoding="utf-8" ?>
<Dance>
    <Rhythmic_Event colour="Color">
        <movement>dance_movement</movement>
        <speed>speed</speed>
    </Rhythmic_Event>
</Dance>
```

References

Abdallah, S., & Plumbley, M. (2003). *Probability as Metadata: Event Detection in Music Using ICA as a Conditional Density Model.* International Symposium on Independent Component Analysis and Signal Separation (ICA), Nara, Japan, pp. 233-238.

Apostolos, M. K. (1988). *A Comparison of the Artistic Aspects of Various Industrial Robots.* International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems (IEA/AIE), Tullahoma, USA, Vol. 1, pp 548-552.

Apostolos, M. K., Littman, M., Lane, S., Handelman, D., & Gelfand, J. (1996). *Robot Choreography: An Artistic-Scientific Connection.* International Journal on Computers & Mathematics with Applications, Elsevier, Vol. 32, No. 1, pp. 1-4.

Arikan, O., & Forsyth, D. A. (2002). *Interactive Motion Generation from Examples.* SIGGRAPH ACM Transactions on Graphics, Vol. 21, No. 3, pp. 483-490.

Arsenio, A., & Fitzpatrick, P. (2003). *Exploiting Cross-Modal Rhythm for Robot Perception of Objects.* International Conference on Computational Intelligence, Robotics, and Autonomous Systems (CIRAS), Singapore, pp. 1-6.

Arsenio, A., & Fitzpatrick, P. (2005). *Exploiting Amodal Cues for Robot Perception.* International Journal of Humanoid Robotics, Springer, Vol. 2, No. 2, pp. 125-143.

Aucouturier, J.-J. (2008). *Cheek to Chip: Dancing Robots and AI's Future.* IEEE Intelligent Systems, Vol. 23, No. 2, pp. 74-84.

Aucouturier, J.-J., & Ogai, Y. (2007). *Making a Robot Dance to Music Using Chaotic Itinerancy in a Network of FitzHugh-Nagumo Neurons.* International Conference on Neural Information Processing (ICONIP), Kitakyushu, Japan, pp. 647-656.

b2. (2008). *miToys.* http://www.b2stuf.com.

Bello, J. P., Daudet, L., Abdallah, S., Duxbury, C., Davies, M., & Sandler, M. (2005). *A Tutorial On Onset Detection in Musical Signals.* IEEE Transactions on Speech and Audio Processing (TASP), Vol. 13, No. 5, pp. 1035-1047.

Bello, J. P., Duxbury, C., Davies, M., & Sandler, M. (2004). *On The Use of Phase and Energy for Musical Onset Detection In The Complex Domain.* IEEE Signal Processing Letters, Vol. 11, No. 6, pp. 553-556.

Bello, J., & Sandler, M. (2003). *Phase-Based Note Onset Detection for Music Signals.* IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Hong Kong, pp. 441-444.

Bobick, A. F., & Davis, J. (2001). *The Recognition of Human Movement Using Temporal Templates.* IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 23, No. 3, pp. 257-267.

Brooks, R. (1991a). *Intelligence Without Representation.* Artificial Intelligence, Elsevier, Vol. 47, No. 1-3, pp. 139-159.

Brooks, R. (1991b). *New Approaches to Robotics.* Science, AAAS, Vol. 253, No. 5025, pp. 1227-1232.

Bruderlin, A., & Williams, L. (1995). *Motion Signal Processing.* Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH), Los Angeles, USA, pp. 97-104.

Burger, B. (2007). *Communication of Musical Expression from Mobile Robots to Humans: Recognition of Music Emotions by Means of Robot Gestures.* MSc. Thesis, KTH Stockholm, pp. 1-137.

Burger, B., & Bresin, R. (2007). *Displaying Expression in Musical Performance by Means of a Mobile Robot.* In Paiva, A., Prada, R., & Picard, R. W. (Eds.), Affective Computing and Intelligent Interaction, Berlin / Heidelberg: Springer, pp. 753-754.

Camurri, A., & Coglio, A. (1998). *An Architecture for Emotional Agents.* IEEE Multimedia, Vol. 5, No. 4, pp. 24-33.

Camurri, A., Lagerlof, I., & Volpe, G. (2003). *Emotions and Cue Extraction from Dance Movements.* International Journal of Human Computer Studies, Elsevier, Vol. 59, No. 1-2. pp. 213-225.

Collins, N. (2005). *A Comparison of Sound Onset Detection Algorithms with Emphasis on Psychoacoustically Motivated Detection Functions.* Convention of the Audio Engineering Society (AES), Barcelona, Spain, pp. 6363-6374.

Daudet, L. (2001). *Transients Modeling by Pruned Wavelet Trees.* in Proc. International Computer Music Conference (ICMC), Havana, Cuba, pp. 1-4.

Davy, M., & Godsill, S. (2002). *Detection of Abrupt Spectral Changes Using Support Vector Machines: An Application to Audio Signal Segmentation.* IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Orlando, USA, Vol. 2, pp. 1313–1316.

Desain, P., & Honing, H. (1992). *Music, Mind, and Machine: Studies in Computer Music, Music Cognition, and Artificial Intelligence.* Amsterdam: Thesis Publishers, pp. 1-330.

Dixon, S. (2001). *Automatic Extraction of Tempo and Beat from Expressive Performances.* Journal of New Music Research, Taylor & Francis, Vol. 30, pp. 39-58.

Dixon, S. (2006). *Onset Detection Revisited.* International Conference on Digital Audio Effects (DAFx), Montreal, Canada, pp. 133–137.

Downie, J. S., West, K., Ehmann, A., & Vincent, E. (2005). *The 2005 Music Information Retrieval Evaluation Exchange (MIREX 2005):*

Preliminary Overview. International Conference on Music Information Retrieval (ISMIR), London, UK, pp. 320-323.

Drake, C., Baruch, C., & Jones, M. (2000). *The Development of Rhythmic Attending in Auditory Sequence: Attunement, Reference Period, Focal Attending.* Cognition, Elsevier, Vol. 77, pp. 251-288.

Duxbury, C., Bello, J. P., Davies, M., & Sandler, M. (2003a). *A Combined Phase and Amplitude Based Approach to Onset Detection for Audio Segmentation.* European Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS), London, UK, pp. 275-280.

Duxbury, C., Bello, J. P., Davies, M., & Sandler, M. (2003b). *Complex Domain Onset Detection for Musical Signals.* London, UK: International Conference on Digital Audio Effects (DAFx), London, UK, pp. 6-9.

Duxbury, C., Sandler, M., & Davies, M. (2002). *A Hybrid Approach to Musical Note Onset Detection.* International Conference on Digital Audio Effects (DAFx), Hamburg, Germany, pp. 33-38.

Ellenberg, R., Grunberg, D., Oh, P., & Kim, Y. (2008). *Exploring Creativity through Humanoids and Dance.* International Conference on Ubiquitous Robotics and Ambient Intelligence (URAI), Seoul, Korea, pp. 1-6.

Enke, U. (2006). *DanSense: Rhythmic Analysis of Dance Movements Using Acceleration-Onset Times.* MSc. Thesis, RWTH Aachen University, Germany, pp. 1-152.

Gizmodo. (2008). *USB Dancing Robot.* http://gizmodo.com/278100/usb-dancing-robot.

Goswami, A. (1999). *Postural Stability of Biped Robots and The Foot-Rotation Indicator (FRI) Point.* International Journal of Robotics Research (IJRR), SAGE Journals, Vol. 18, No. 6, pp. 523–533.

Goto, M., & Muraoka, Y. (1999). *Real-Time Beat Tracking for Drumless Audio Signals.* Speech Communication, Elsevier, Vol. 27, No. 3-4, pp. 331-335.

Gouaillier, D., Hugel, V., Blazevic, P., Kilner, C., Monceaux, J., Lafourcade, P., Marnier, B., Serre, J., Maisonnier, B. (2008). *The NAO Humanoid: A Combination of Performance and Affordability.* CoRR, abs/0807.3223, pp. 1-10.

HitecRobotics. (2008). *RoboNova-I.* http://www.hitecrobotics.com.

Ijspeert, A. J., Nakanishi, J., & Schaal, S. (2002). *Learning Rhythmic Movements by Demonstration using Nonlinear Oscillators.* IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Lausanne, Switzerland, pp. 958-963.

Jáuregui, D. A., & Horain, P. (2009). *Region-Based vs. Edge-Based Registration for 3D Motion Capture by Real Time Monoscopic Vision.* MIRAGE International Conference on Computer Vision / Computer Graphics Collaboration Techniques, Rocquencourt, France, pp. 344-355.

Jehan, T. (1997). *Musical Signal Parameter Estimation.* MSc. Thesis, University of California, Berkeley, USA, pp. 1-82.

Kapanci, E., & Pfeffer, A. (2004). *A Hierarchical Approach to Onset Detection.* International Computer Music Conference (ICMC), Miami, USA, pp. 438-441.

Kauppinen, I. (2002). *Methods for Detecting Impulsive Noise in Speech and Audio Signals.* International Conference on Digital Signal Processing (DSP), Santorini, Greece, Vol. 2, pp. 967-970.

Kim, T. H., Park, S. I., & Shin, S. Y. (2003). *Rhythmic Motion Synthesis based on Motion Beat Analysis.* ACM Transactions on Graphics, Vol. 22, No. 3, pp. 392-401.

Klapuri, A. (1999). *Sound Onset Detection by Applying Psychoacoustic Knowledge.* IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Phoenix, USA, pp. 3089-3092.

Klapuri, A., Eronen, A., & Astola, J. (2006). *Analysis of the Meter of Acoustic Musical Signals.* IEEE Transactions on Audio, Speech and Language Processing (TASLP), Vol. 14, No. 1, pp. 342-355.

Kozima, H., Michalowski, M. P., & Nakagawa, C. (2008). *Keepon: A Playful Robot for Research, Therapy, and Entertainment.* International Journal of Social Robotics, Springer, Vol. 1, No. 1, pp. 3-18.

Lacoste, A., & Eck, D. (2005). *Onset Detection With Artificial Neural Networks.* Music Information Retrieval Evaluation Exchange (MIREX) Note Onset Detection Contest, International Symposium on Music Information Retrieval (ISMIR), London, UK, pp. 1-4.

Lacoste, A., & Eck, D. (2007). *A Supervised Classification Algorithm for Note Onset Detection.* EURASIP Journal on Applied Signal Processing, SpringerOpen, 2007(ID 43745), pp. 1-13.

Lee, E., Enke, U., Borchers, J., & de Jong, L. (2007). *Towards Rhythmic Analysis of Human Motion using Acceleration-Onset Times.* International Conference on New Interfaces for Musical expression (NIME), New York, USA, pp. 136-141.

Lee, H. C., & Lee, I. K. (2005). *Automatic Synchronization of Background Music and Motion in Computer Animation.* European Association for Computer Graphics (EUROGRAPHICS), Dublin, Ireland, pp. 353-362.

Leveau, P., Daudet, L., & Richard, G. (2004). *Methodology and Tools for The Evaluation of Automatic Onset Detection Algorithms in Music.* International Symposium on Music Information Retrieval (ISMIR), Barcelona, Spain, pp. 72-75.

Likert, R. (1932). *A Technique for the Measurement of Attitudes.* Archives of Psychology, New York, No. 140, pp. 1-55.

Lucas, B., & Kanade, T. (1981). *An Iterative Image Registration Technique with an Application to Stereo Vision.* International Joint Conference on Artificial Intelligence (IJCAI), Vancouver, Canada, pp. 674-679.

Marolt, M., Kavcic, A., & Privosnik, M. (2002). *Neural Networks for Note Onset Detection in Piano Music.* International Computer Music Conference (ICMC), Gotenborg, Sweden, pp. 2-5.

Masri, P. (1996). *Computer Modeling of Sound for Transformation and Synthesis of Musical Signal.* PhD Thesis, University of Bristol, Bristol, UK, pp. 1-240.

Michalowski, M. P., & Kozima, H. (2007). *Methodological Issues in Facilitating Rhythmic Play with Robots.* ACM/IEEE International Conference on Human-Robot Interaction (HRI), Washington DC, USA, pp. 89-96.

Michalowski, M. P., Kozima, H., & Sabanovic, H. (2007). *A Dancing Robot for Rhythmic Social Interaction.* IEEE International Conference on Robot and Human Interactive Communication (Ro-Man), Jeju, Korea, pp. 95-100.

Michalowski, M. P., Sabanovic, S., & Michel, P. (2006). *Roillo: Creating a Social Robot for Playrooms.* IEEE International Symposium on Robot and Human Interactive Communication (Ro-Man), Hatfield, UK, pp. 587–592.

Mizumoto, T., Takeda, R., Yoshii, K., Komatani, K., Ogata, T., & Okuno, H. G. (2008). *A Robot Listens to Music and Counts Its Beats Aloud by Separating Music from Counting Voice.* IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Nice, France, pp. 1538-1543.

Moeslund, T. B., Hilton, A., & Krüger, V. (2006). *A Survey of Advances in Vision-Based Human Motion Capture and Analysis.* International Journal of Computer Vision and Image Understanding, Elsevier, Vol. 104, No. 2-3, pp. 90-126.

Murata, K., Nakadai, K., Takeda, R., Okuno, H. G., Torii, T., Hasegawa, Y., & Tsujino, H. (2008). *A Beat-Tracking Robot for Human-Robot Interaction and Its Evaluation.* IEEE-RAS International Conference on Humanoid Robots (Humanoids), Daejeon, Korea, pp. 79-84.

Murata, K., Nakadai, K., Yoshii, K., Takeda, R., Torii, T., Okuno, H. G., Hasegawa, Y., Tsujino, H. (2008). *A Robot Singer with Music Recognition Based on Real-Time Beat Tracking.* International

Symposium on Musical Information Retrieval (ISMIR), Philadelphia, USA, pp. 199-204.

Nakadai, K., Okuno, H. G., & Kitano, H. (2002). *Real-Time Sound Source Localization and Separation for Robot Audition.* IEEE International Conference on Spoken Language Processing (INTERSPEECH), Denver, USA, pp. 193-196.

Nakahara, N., Miyazaki, K., Sakamoto, H., Fujisawa, T. X., Nagata, N., & Nakatsu, R. (2009). *Dance Motion Control of a Humanoid Robot Based on Real-Time Tempo Tracking from Musical Audio Signals.* International Conference on Entertainment Computing (ICEC), Paris, France, Lecture Notes in Computer Science (LNCS), Springer-Verlag, Vol. 5709, pp. 36-47.

Nakaoka, S. (2003). *Generating Whole Body Motions for a Biped Humanoid Robot.* MSc. Thesis of Information Science and Technology in Computer Science, at Graduate School of Information Science and Technology, University of Tokyo, pp. 1-78.

Nakaoka, S., Nakazawa, A., Kanahiro, F., Kaneko, K., Morisawa, M., & Ikeuchi, K. (2005). *Task Model of Lower Body Motion for a Biped Humanoid Robot to Imitate Human Dances.* IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Edmonton, Canada, pp. 3157-3162.

Nakaoka, S., Nakazawa, A., Kanehiro, F., Kaneko, K., Morisawa, M., Hirukawa, H., & Ikeuchi, K. (2007). *Learning from Observation Paradigm: Leg Task Models for Enabling a Biped Humanoid Robot to Imitate Human Dance.* International Journal on Robotics Research (IJRR), SAGE Journals, Vol. 26, No. 8, pp. 829-844.

Nakazawa, A., Nakaoka, S., Ikeuchi, K., & Yokoi, K. (2002). *Imitating Human Dance Motions through Motion Structure Analysis.* IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Lausanne, Switzerland, pp. 2539-2544.

Naveda, L., & Leman, M. (2010). *The Spatiotemporal Representation of Dance and Music Gestures using Topological Gesture Analysis (TGA).* Music Perception, University of California Press, Vol. 28, No. 1, pp. 93-111.

Oliveira, J. L. (2008b). *Towards an Interactive Framework for Robot Dancing Applications.* MSc. Thesis in Electrical and Computers Engineering, Faculty of Engineering of the University of Porto, pp. 1-100.

Oliveira, J. L., Davies, M. E., Gouyon, F., & Reis, L. P. (2012b). *Beat tracking for multiple applications: A Multi-Agent System Architecture with State Recovery.* IEEE Transactions on Audio Speech and Language Processing (TASLP), Vol. 20, No. 10, pp. 1-11.

Oliveira, J. L., Gouyon, F., & Reis, L. P. (2008a). *Towards an Interactive Framework for Robot Dancing Applications.* International Conference on Digital Arts (ARTECH), Porto, Portugal, pp. 52-59.

Oliveira, J. L., Ince, G., Nakamura, K., & Nakadai, K. (2012c). *Online Audio Beat Tracking for a Dancing Robot in the Presence of Ego-Motion Noise in a Real Environment.* IEEE International Conference on Robotics and Automation (ICRA), Minnesota, USA, pp. 403-408.

Oliveira, J. L., Ince, G., Nakamura, K., Nakadai, K., Okuno, H. G., Reis, L. P., & Gouyon, F. (2012d). *Live Assessment of Beat Tracking for Robot Audition.* To appear in IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vilamoura, Portugal, pp. 1-6.

Oliveira, J. L., Naveda, L., Gouyon, F., Reis, L. P., Sousa, P., & Leman, M. (2012a). *A Parameterizable Spatiotemporal Representation of Popular Dance Styles for Humanoid Dancing Characters.* EURASIP Journal on Audio, Speech, and Music Processing (ASMP), SpringerOpen, Vol. 2012:18, No. 19, pp. 1-20.

Oliveira, J. L., Reis, L. P., & Gouyon, F. (2008c). *Lego-NXT Robot Dancing Framework.* http://www.youtube.com/watch?v=NtonkjhlvbY.

Oliveira, J. L., Reis, L. P., & Gouyon, F. (2008d). *Lego-NXT Robot Dancing Demo.* http://www.youtube.com/watch?v=4Ezywrc0ioA.

Oliveira, J. L., Reis, L. P., Faria, B. M., & Gouyon, F. (2012). *An Empiric Evaluation of a Real-Time Robot Dancing Framework based on Multi-Modal Events.* Submitted to TELKOMNIKA Indonesian Journal of Electrical Engineering, pp. 1-12.

Or, J. (2006). *A Control System for a Flexible Spine Belly Dancing Humanoid Robot.* Artificial Life Journal, MIT Press, Vol. 12, No. 1, pp. 63-87.

Or, J. (2009). *Towards the Development of Emotional Dancing Humanoid Robots.* International Journal of Social Robotics, Springer, Vol. 1, No. 4, pp. 367-382.

Park, I. W., Kim, Y. D., Lee, B. J., Yoo, J. K., & Kim, J. H. (2007). *Generating Performance Motions of Humanoid Robot for Entertainment.* IEEE International Symposium on Robot and Human Interactive Communication (Ro-Man), Jeju, Korea, pp. 950-955.

Repp, B. H. (2005). *Sensorimotor Synchronization: A Review of the Tapping Literature.* Psychonomic Bulletin & Review, Springer, Vol. 12, pp. 969–992.

RoboCupJunior. (2008). *Dance Contest.* http://rcj.robocup.org/dance.html.

ROBO-ONEEntertainment. (2008). *GATE Robodance Competition.* http://www.roboenta.com.

Rodet, X., & Jaillet, F. (2001). *Detection and Modeling of Fast Attack Transients.* International Computer Music Conference (ICMC), Havana, Cuba, pp. 30-33.

Santiago, C. B., Oliveira, J. L., Reis, L. P., Sousa, A., & Gouyon, F. (2012). *Overcoming Motor-Rate Limitations in Online Synchronized Robot Dancing.* International Journal of Computational Intelligence Systems (IJCIS), Taylor & Francis, Vol. 5, No. 4, pp. 700-713.

Scheirer, E. (1998). *Tempo and Beat Analysis of Acoustic Musical Signals.* Journal of the Acoustical Society of America, ASA Publications, Vol. 103, No. 1, pp. 588-601.

Scheirer, E. (2000). *Music-Listening Systems.* PhD Thesis, MIT, Cambridge, pp. 1-248.

Schöllig, A., Augugliaro, F., Lupashin, S., & D'Andrea, R. (2010). *Synchronizing the Motion of a Quadrocopter to Music.* IEEE International Conference on Robotics and Automation (ICRA), Anchorage, USA, pp. 3355-3360.

SegaToys. (2008). *iPets.* http://www.idog-segatoys.com/.

Shinozaki, K., Iwatani, A., & Nakatsu, R. (2007). *Concept and Construction of a Robot Dance System.* International Journal of Virtual Reality (IJVR), Vol. 6, No. 3, pp. 29-34.

Shinozaki, K., Oda, Y., Tsuda, S., Nakatsu, R., & Iwatani, A. (2006). *Study of Dance Entertainment Using Robots.* International Conference on Technologies for E-Learning and Digital Entertainment (Edutainment), Hangzhou, China, pp. 473-483.

Shiratori, T., Kudoh, S., Nakaoka, S., & Ikeuchi, K. (2007). *Temporal Scaling of Upper Body Motion for Sound Feedback System of a Dancing Humanoid Robot.* IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), San Diego, USA, pp. 3251-3257.

Sony. (2008). *Rolly.* (Manual) http://pdf.crse.com/manuals/3870834321.pdf.

Sousa, P., Oliveira, J. L., Reis, L. P., & Gouyon, F. (2011). *Humanized Robot Dancing: Humanoid Motion Retargeting based in a Metrical Representation of Human Dance Styles.* Workshop on Intelligent Robotics (iRobot), Portuguese Conference on Artificial Intelligence (EPIA), Lisbon, Portugal, pp. 392-406.

Stauffer, C., & Grimson, W. E. (1999). *Adaptive Background Mixture Models for Real-Time Tracking.* IEEE Computer Society Conference

on Computer Vision and Pattern Recognition (CVPR), City of Fort Collins, USA, Vol. 2, pp. 2-8.

Suzuki, K., & Hashimoto, S. (2004). *Robotic Interface for Embodied Interaction via Dance And Musical Performance.* IEEE Special Issue on Johannsen, G. (Guest Editor) Engineering and Music, Vol. 92, No. 4, pp. 656-671.

Takeda, T., Hirata, Y., & Kosuge, K. (2007). *Dance Step Estimation Method Based on HMM for Dance Partner Robot.* IEEE Transactions on Industrial Electronics, Vol. 54, No. 2, pp. 699-706.

Tanaka, F., & Suzuki, H. (2004). *Dance Interaction with QRIO: A Case Study for Non-boring Interaction by using an Entertainment Ensemble Model.* IEEE International Workshop on Robot and Human Interactive Communication (Ro-Man), Kurashiki, Japan, pp. 419-424.

Tanaka, F., Fortenberry, B., Aisaka, K., & Movellan, J. (2005). *Plans for Developing Real-time Dance Interaction between QRIO and Toddlers in a Classroom Environment.* IEEE International Conference on Development and Learning (ICDL), Osaka, Japan, pp. 142-147.

Tidemann, A., & Öztürk, P. (2007). *Self-Organizing Multiple Models for Imitation: Teaching a Robot to Dance the YMCA.* IEA/AIE 2007, Vol. 4570 of Lecture Notes in Computer Science, Springer, pp. 291-302.

Tzanetakis, G., & Cook, P. (2000). *MARSYAS: A Framework for Audio Analysis.* Organized Sound, Cambridge Journals, Vol. 4, No. 3, pp. 169-175.

UAS. (2008). *Hexapodmeisterschaft.* University of Applied Sciences, Austria. http://www.hexapod.at.

Vukobratovic, M., Borovac, B., Surla, D., & Stokic, D. (1990). *Biped Locomotion: Dynamics, Stability, Control and Application (Scientific Fundamentals of Robotics).* Springer-Verlag, pp. 1-349.

Weinberg, G. (2007b). *Robotic Musicianship Musical Interactions between Humans and Machines.* Chapter in Robotic Musicianship, I-Tech Education and Publishing, pp. 1-22.

Weinberg, G., & Driscoll, S. (2007a). *The Perceptual Robotic Percussionist – New Developments in Form, Mechanics, Perception and Interaction Design.* ACM/IEEE International Conference on Human-Robot Interaction (HRI), Washington DC, USA, pp. 97-104.

Weinberg, G., Aimi, R., & Jennings, K. (2002). *The Beatbug Network: A Rhythmic System for Interdependent Group Collaboration.* International Conference on New Interfaces for Musical Expression (NIME), Dublin, Ireland, pp. 106-111.

Weinberg, G., Driscoll, S., & Parry, M. (2005). *Musical Interactions with a Perceptual Robotic Percussionist.* IEEE International Workshop on Robot and Human Interactive Communication (Ro-Man), Nashville, USA, pp. 456-461.

Wikipedia. (2008). *Butterworth Filter.* http://en.wikipedia.org/wiki/Butterworth_filter.

Williamson, M. (1999). *Designing Rhythmic Motions Using Neural Oscillators.* IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Kyongju, Korea, pp. 494-500.

WooWeeRobotics. (2008). *RoboSapiens.* http://www.wowwee.com.

Yoshii, K., Nakadai, K., Torii, T., Hasegawa, Y., Tsujino, H., Komatani, K., Ogata, T., Okuno, H. (2007). *A Biped Robot that Keeps Steps in Time with Musical Beats while Listening to Music with Its Own Ears.* IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), San Diego, USA, pp. 1743-1750.

Zhao, L., & Thorpe, C. E. (2000). *Stereo- and Neural Network-Based Pedestrian Detection.* IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 1, No. 3, pp. 148-154.