UNIVERSIDADE CATÓLICA PORTUGUESA

ZATLAB: RECOGNIZING GESTURES FOR ARTISTIC PERFORMANCE INTERACTION

Dissertation submitted to the Portuguese Catholic University in partial fulfillment of requirements of the Doctoral Degree in Science and Technologies of the Arts – Computer Music

by

André Miguel Passos Baltazar

ESCOLA DAS ARTES
April 2014

UNIVERSIDADE CATÓLICA PORTUGUESA

ZATLAB: RECOGNIZING GESTURES FOR ARTISTIC PERFORMANCE INTERACTION

Dissertation submitted to the Portuguese Catholic University in partial fulfillment of requirements of the Doctoral Degree in Science and Technologies of the Arts – Computer Music

By André Miguel Passos Baltazar

Supervised by Professor Luís Gustavo Pereira Marques Martins
Co- Supervised by Professor Jaime S. Cardoso

ESCOLA DAS ARTES
April 2014

This Doctorate Research was partially supported by the Portuguese Science and Technology Foundation (FCT), trough the award of the Doctorate Grant reference SFRH / BD / 61662 / 2009.

FCT
Fundação para a Ciência e a Tecnologia
MINISTÉRIO DA EDUCAÇÃO E CIÊNCIA

The author is affiliated with:

UNIVERSITAS CATHOLICA LUSITANA
VERITATI
CATÓLICA
PORTO
ARTES

CITAR
Centro de Investigação em Ciência e Tecnologia das Artes

Abstract

Most artistic performances rely on human gestures, ultimately resulting in an elaborate interaction between the performer and the audience.

Humans, even without any kind of formal analysis background in music, dance or gesture are typically able to extract, almost unconsciously, a great amount of relevant information from a gesture. In fact, a gesture contains so much information, why not use it to further enhance a performance?

Gestures and expressive communication are intrinsically connected, and being intimately attached to our own daily existence, both have a central position in our (nowadays) technological society. However, the use of technology to understand gestures is still somehow vaguely explored, it has moved beyond its first steps but the way towards systems fully capable of analyzing gestures is still long and difficult (Volpe, 2005). Probably because, if on one hand, the recognition of gestures is somehow a trivial task for humans, on the other hand, the endeavor of translating gestures to the virtual world, with a digital encoding is a difficult and ill-defined task. It is necessary to somehow bridge this gap, stimulating a constructive interaction between gestures and technology, culture and science, performance and communication. Opening thus, new and unexplored frontiers in the design of a novel generation of multimodal interactive systems.

This work proposes an interactive, real time, gesture recognition framework called the Zatlab System (ZtS). This framework is flexible and extensible. Thus, it is in permanent evolution, keeping up with the different technologies and algorithms that emerge at a fast pace nowadays. The basis of the proposed approach is to partition a temporal stream of captured movement into perceptually motivated descriptive features and transmit them for further processing in Machine Learning algorithms. The framework described will take the view that perception primarily depends on the previous knowledge or learning. Just like humans do, the framework will have to learn gestures and their main features so that later it can identify them. It is however planned to be flexible enough to allow learning gestures on the fly.

This dissertation also presents a qualitative and quantitative experimental validation of the framework. The qualitative analysis provides the results concerning the users acceptability of the framework. The quantitative validation provides the results about the gesture recognizing algorithms. The use of Machine Learning algorithms in these tasks allows the achievement of final results that compare or outperform typical and state-of-the-art systems.

In addition, there are also presented two artistic implementations of the framework, thus assessing its usability amongst the artistic performance domain.

Although a specific implementation of the proposed framework is presented in this dissertation and made available as open source software, the proposed approach is flexible enough to be used in other case scenarios, paving the way to applications that can benefit not only the performative arts domain, but also, probably in the near future, helping other types of communication, such as the gestural sign language for the hearing impaired.

Resumo

Grande parte das apresentações artísticas são baseadas em gestos humanos, ultimamente resultando numa intricada interação entre o performer e o público.

Os seres humanos, mesmo sem qualquer tipo de formação em música, dança ou gesto são capazes de extrair, quase inconscientemente, uma grande quantidade de informações relevantes a partir de um gesto. Na verdade, um gesto contém imensa informação, porque não usá-la para enriquecer ainda mais uma performance?

Os gestos e a comunicação expressiva estão intrinsecamente ligados e estando ambos intimamente ligados à nossa própria existência quotidiana, têm uma posição central nesta sociedade tecnológica actual. No entanto, o uso da tecnologia para entender o gesto está ainda, de alguma forma, vagamente explorado. Existem já alguns desenvolvimentos, mas o objetivo de sistemas totalmente capazes de analisar os gestos ainda está longe (Volpe, 2005). Provavelmente porque, se por um lado, o reconhecimento de gestos é de certo modo uma tarefa trivial para os seres humanos, por outro lado, o esforço de traduzir os gestos para o mundo virtual, com uma codificação digital é uma tarefa difícil e ainda mal definida. É necessário preencher esta lacuna de alguma forma, estimulando uma interação construtiva entre gestos e tecnologia, cultura e ciência, desempenho e comunicação. Abrindo assim, novas e inexploradas fronteiras na concepção de uma nova geração de sistemas interativos multimodais .

Este trabalho propõe uma *framework* interativa de reconhecimento de gestos, em tempo real, chamada Sistema Zatlab (ZtS). Esta *framework* é flexível e extensível. Assim, está em permanente evolução, mantendo-se a par das diferentes tecnologias e algoritmos que surgem num ritmo acelerado hoje em dia. A abordagem proposta baseia-se em dividir a sequência temporal do movimento humano nas suas características descritivas e transmiti-las para posterior processamento, em algoritmos de *Machine Learning*. A *framework* descrita baseia-se no facto de que a percepção depende, principalmente, do conhecimento ou aprendizagem prévia. Assim, tal como os humanos, a *framework* terá que aprender os gestos e as suas principais características para que depois possa identificá-los. No entanto, esta está prevista para ser flexível o suficiente de forma a permitir a aprendizagem de gestos de forma dinâmica.

Esta dissertação apresenta também uma validação experimental qualitativa e quantitativa da *framework*. A análise qualitativa fornece os resultados referentes à aceitabilidade da *framework*. A validação quantitativa fornece os resultados sobre os algoritmos de reconhecimento de gestos. O uso de algoritmos de *Machine Learning* no reconhecimento de gestos, permite a obtenção de resultados finais que são comparaveis ou superam outras implementações do mesmo género.

Além disso, são também apresentadas duas implementações artísticas da *framework*, avaliando assim a sua usabilidade no domínio da performance artística.

Apesar duma implementação específica da *framework* ser apresentada nesta dissertação e disponibilizada como software *open-source*, a abordagem proposta é suficientemente flexível para que esta seja usada noutros cenários. Abrindo assim, o caminho para aplicações que poderão beneficiar não só o domínio das artes performativas, mas também, provavelmente num futuro próximo, outros tipos de comunicação, como por exemplo, a linguagem gestual usada em casos de deficiência auditiva.

To my loving grandmother Alice, who was an avid reader, crosswords master and an inspirational autodidact.

Acknowledgements

First of all, I would like to thank my mentor and adviser Prof. Luis Gustavo Martins for his support. Throughout all these years he has been a tremendous help for me by sharing his knowledge, his patience, his availability and, most important, his friendship and guidance.

I would like also to acknowledge the participation of my co-adviser Prof. Jaime dos Santos Cardoso, in particular for his wisdom and capability of explaining, in small words, cumbersome algorithms and equations.

Another person I would like to standout is Prof. Ricardo Queirós, that recently obtained his Phd degree. Learning from his experience, he has become, for me, an academic beacon and, most of all, a great friend. I also would like to thank him for giving my first opportunity to teach at the Academy.

Carrying on with the academic side, my recognition goes to all senior researchers and Professors, with whom I shared my obstacles and victories along these years, in particular Professor Paulo Ferreira-Lopes (the PhD Program Coordinator), Prof. Álvaro Barbosa and Prof. Sofia Lourenço. Moreover i would like to thank Prof. Chris Chafe for receiving me as an invited researcher at the Center for Computer Research in Music and Acoustics - CCRMA in the Stanford University, California, EUA.

I also would like to address my best acknowledgements to the researchers with whom I worked closely, some due to sharing of the office in UCP and others due to

common interests in research. These are: André Perrotta, Francisco Bernardo, Helena Figueiredo, Joana Gomes, João Cordeiro, Jorge Cardoso, Mailis Rodrigues, Nicolas Makelberge, Samuel Van Ransbeeck, Vasco Carvalho and Vitor Joaquim. Thanks everyone for the support, we make a great research group.

A word of gratitude goes to all the participants of the experiments described in this thesis and also to MisoMusic and UCP for commission the artistic applications of the framework here presented.

On the personal side, I would like to thank all my family and friends for the love and support given. In particular, to my parents Etelvina and Valdemar and sister Gisela, for making me the person I am proud to be today. Furthermore, I want to acknowledge my longtime friends (almost brothers) and be-dom band members: André, Marco, Raul, Rui and Tiago.

Last but not least, I want to thank Ana, my best-friend and lifetime partner, for being so wonderful, caring and simply fantastic throughout all these years we have been together. Most important, for having the patience of hearing me grumbling about the Phd and the thesis, almost 24-7, and always having a positive word that brought me to the finish line.

Contents

List of Tables

List of Figures

List of Acronyms

AI Artificial Intelligence

CITAR Centro de Investigação em Ciência e Tecnologia das Artes

CMOS Complementary Metal-Oxide Semiconductor

CV Computer Vision

DTW Dynamic Time Warping

FestivalIN Festival Innovation and Creativity

FOSS Free and open-source software

fps frames per second

GUI Graphical User Interface

HCI Human Computer Interaction

HMM Hidden Markov Models

IP Internet Protocol

IR infrared

LAN Local Area Network

MIDI Musical Instrument Digital Interface

ML Machine Learning

MoCap Motion Capture

NIME New Instruments for Musical Expression

OSC Open Sound Control

TDNN Time Delay Neural Network

UCP Universidade Católica Portuguesa

Chapter 1

Introduction

"Small gestures can have a big impact."

Julianna Margulies

1.1 Context and Motivation

There is so much information in a simple gesture. Why not use it to enhance a performance? We use our hands constantly to interact with things. Pick them up, move them, transform their shape, or activate them in some way. In the same unconscious way we gesticulate in communicating fundamental ideas: stop; come closer; go there; no; yes; and so on. Gestures are thus a natural and intuitive form of both interaction and communication (Watson, 1993). Children start to communicate by gestures (around 10 months age) even before they start speaking. There is also an ample evidence that by the age of 12 months children are able to understand the gestures other people produce (Rowe and Goldin-meadow, 2009). For the most part gestures are considered an auxiliary way of communication to speech, tough there are also studies that focus on the role of gestures in making interactions work (Roth, 2001).

Gestures have been studied for a long time. The research on gesture analysis, processing and synthesis has seen a growing interest on the part of the scientific community in recent years, and demonstrated its paramount importance for the Human Computer Interaction (HCI) (Volpe, 2005). One of the main characteristics in gesture research is its cross-disciplinary nature. The philosophical research on gesture allows a deep investigation into the mechanisms of human-human communication (e.g. in the fields of psychology, social science, art and humanities) and this knowledge can be successfully exploited by the rather more technological research, for example, in interaction design. This cross-disciplinary nature can highly benefit the research and open new perspectives in both fields. If, on one hand, scientific and technological research can grow from models and theories borrowed from psychology, social science, art and humanities, on the other hand, these disciplines can start using, with increasing confidence, the tools technology can provide for their own research (i.e. examine, at a depth never before reached, the remaining mysteries and hidden complexities of human beings).

It is also important to understand that whereas all gestures derive from a chain of movements, not all movements can be considered gestures (Kendon, 1994). Gestures are the principal non-verbal, cross-modal communication channel, and they rely on movements for different domains of communication. Looking at the Merriam-Webster dictionary [1], one will find the word “gesture” means a movement usually of the body or limbs that expresses or emphasizes an idea, intention, sentiment, or attitude, as well as the use of motions of the limbs or body as a means of expression.

There are unconscious gestures, those used on the common day to day routine. And there are conscious gestures, rehearsed over and over, and carried out in a stage during a performance. In this later sense, gestures are truly used as conveyors of information, the performer has the ability to change the meaning

[1] `http://www.merriam-webster.com/dictionary/gesture`

and feeling of what is transmitting just with a simple nod of the head, positioning of the shoulders or raise of an arm. This focus on the affective and emotional information gesture conveys leads to the concept of "expressive" gesture (Volpe, 2005), as carrier of a set of temporal/spatial features responsible for conveying expressiveness. While for many years research was devoted to the investigation of more cognitive, intellective aspects, in the last decade a lot of studies have focused on emotional processes and social interaction (e.g. the Kansei research project (Inokuchi, 2010)). There is also a growing effort in the research areas of movement and gesture, in particular the expressive gesture. This (expressive gesture) can be considered a broad concept that includes music, human movement and visual gesture. And thus, it assumes an important role for research in music, computer music and performing arts. Actually, the performing arts have become a key research and application field, since they are an ideal test-bed for works concerning mechanisms for non-verbal communication of affective, emotional, expressive content. Volpe (Volpe, 2005) even refers that *"a main topic for current and future research consists of using music and dance performances to study expressive gestures and their ability to convey emotional states (e.g., the well-known and consolidated basic emotions) and engage spectators"*".

Gestures and expressive communication are therefore intrinsically connected, and being intimately attached to our own daily existence, both have a central position in our (nowadays) technological society. However, the use of technology to understand gestures is still somehow vaguely explored, it has moved beyond its first steps but the way towards systems fully capable of analyzing gestures is still long and difficult. Probably because if on one hand, the recognition of gestures is somehow a trivial task for humans, on the other hand, the endeavor of translating gestures to the virtual world, with a digital encoding is a difficult and ill-defined task. It is necessary to somehow bridge this gap, stimulating a constructive interaction between gestures and technology, culture and science, performance and communication. Opening thus, new and unexplored frontiers in the design of a novel generation of multimodal interactive systems.

1.2 Thesis Statement

The main problem this work addresses is the real-time identification and recognition of gestures, particularly in the complex domain of artistic performance.

The overall goal of this research is to foster the use of gestures, in an artistic context, to the creation of new ways of expression. By recognizing the performed gestures, one is able to map them to several controls, from lightning control to the creation of visuals, sound control or even music creation, thus allowing performers the real-time manipulation of creative events.

However, the objective is not, at least at this stage, to provide a complete gesture classification system, neither a model for expressing and communicating the linguistic or psychological meaning of gesture. Instead, the work presented and discussed in this dissertation proposes a modular gesture recognition framework along with the individual building blocks (modules) involved. These building blocks range from movement analysis and gesture recognition, to human skeleton representation or data transmission. This modular framework provides a solid basis of development, already paving the way for the future inclusion of higher level processes, such as the ones involved, for example, in the recognition and automatic translation from gesture to speech (in the case of sign language).

1.3 Current State

Generally speaking, there are two lines of thought running through the gesture research field (Zhao and Badler, 2001). In one line, there is work by linguists, psychologists, neurologists, choreographers and physical therapists. Their concern is largely related with a conceptual understanding of gesture and its function. Although their work often involves some deep analysis, most of their models are qualitative and theoretical, making it very difficult to verify their correctness, generality, and appropriateness. They are not committed to building a computational gesture model to verify their theories, and are rarely concerned with any computer implementation implications of their work.

The other line of research on gesture operates in areas such as Computer Vision (CV), HCI, human motor control, and computer graphics and animation. Most of these approaches are in a system-oriented context that enables the experimentation and empirical analysis.

However, while these approaches explore different areas of research, some fundamental questions remain unanswered. On the Chapter 2, is presented a discussion on the main approaches taken in each line of research. It will give a complete overview about the state of the art, carefully map the challenges and sustain the path taken on this work to overcome them.

The gesture recognition systems (including the one proposed in this thesis) face several demanding difficulties, making their performance somehow limited when compared to the human recognition capabilities. Nevertheless, some of the current results already provide improved alternatives to the common gesture analysis and recognition applications.

1.4 The Main Challenges

The gesture recognition is rather simple for the average person. Humans can process multiple factors (such as muscle volume and tension or facial expressions) from multiple senses simultaneously to analyze an action, while a computerized system often limits available data to one or two channels (i.e. sensors) (Zhao and Badler, 2001). Automatically recognizing gestures is a complex task which involves many aspects such as motion modeling, motion analysis, pattern recognition and machine learning, and even psycholinguistic studies (Wu and Huang, 1999).

Nevertheless, there have been many systems implemented (Wu and Huang, 1999) in domains such as virtual environments, smart surveillance, teleconferencing, sign language translation. And some solutions for the performance domain have been already provided (Bevilacqua and Muller, 2005), (Camurri et al., 2000). But, there is still a big gap between what the systems are able to do when compared to humans capabilities.

This work takes the challenge of shortening that gap, doing gesture recognition in real-time, using a multidisciplinary approach to the problem, based in some of the known principles of how humans recognize gestures, together with the computer science methods to successfully complete the task.

1.5 Related Research Areas

The work proposed in this thesis is multidisciplinary, ranging from sciences such as HCI and gesture related research to more mathematical, objective sciences such as CV and Machine Learning (ML). The following paragraphs provide short descriptions of these research areas and their connection to the project.

Human Computer Interaction (Chairman-Hewett, 1992) is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them. Because HCI studies a human and a machine in communication, it draws from supporting knowledge on both the machine and the human side. On the machine side, techniques in computer graphics, operating systems, programming languages, and development environments are relevant. On the human side, communication theory, graphic and industrial design disciplines, linguistics, social sciences, cognitive psychology, and human performance are relevant. All this knowledge from the different areas of the HCI field contribute in some degree to this work.

Machine Learning (Grosan and Abraham, 2011) derives from the artificial intelligence field. It is concerned with the study of building computer programs that automatically improve and/or adapt their performance through experience. ML can be thought of as "programming by example" and has many common aspects with other domains such as statistics and probability theory (understanding the phenomena that have generated the data), data mining (finding patterns in the data that are understandable by people) and cognitive sciences. Instead of the human programming a computer to solve a task directly, the goal of ML is to devise methods by which a computer program is able of come up with is own solution to the task, based only on examples provided. In the particular case of the work presented in this thesis, machine-learning techniques are used to implement a computer system, which is able to identify specific gestures in a complex human movement.

Computer Vision (Aggarwal, 2011) consists in the estimation of several properties of physical objects, based on their two dimensional (projection) images through the use of computers and cameras. With its beginnings in the early 1960s, it was thought to be an easy problem with a solution probably possible over a short time period. However, it revealed to be a task far more difficult. Since those early days CV has matured from a small research topic to a complete field of research and application. Some CV techniques will be described on this thesis, including some algorithms developed and published by the author (e.g. (Baltazar et al., 2010)).

Gesture Research is of upmost importance for this thesis. Before programming a computer to recognize gestures it is important to know as much as possible about their characteristics. How they are defined, their main features, the notations used, what features are important to extract and how to extract them.

As an overview one can already state there are various domains of research in gestures. These domains can range from the psychological-linguistic, to the cognitive science or the performative arts. And regarding each of them one will find different definitions and conventions on gestures. With respect to references in the field, one investigated the work of leading authors in their respective research domain. These include: Kendon (Kendon, 1970, 8, 9); McNeill & Levy (McNeill, 1985, 9; McNeill and Levy, 1982); and Rimé & Schiaratura (Feldman and Rimé, 1991; Rimé, 1982)); Godoy (Godoy and Leman, 2009); Camurri (Camurri et al., 2000); amongst other important research works in the field.

A thorough discussion about gestures, their understanding, observation, capture and recognition, will be presented in Chapter 2.

1.6 Applications

Real-time gesture recognition is a challenging problem by itself. Adding the performance context to it, where the technology should pass unnoticed by the audience, and without disturbing the performers natural abilities, increases even further the difficulty. This work presents an opportunity for the development of gesture recognition solutions for a very specific set of conditions. Nevertheless, when developing for such a specific case scenario, one will reach solutions that can be used in a broader sense, not only for the performing arts field, but also in others related researches. Some examples of possible applications for the solutions proposed in this thesis are listed next:

- **Automatic sign language recognition** - If a gesture itself conveys information, the sign language conventions allow for the deaf to communicate using hand gestures and body language to express meaning, as opposed to acoustically transmitted sound patterns. This can involve simultaneously combining hand shapes, orientation and movement of the hands, arms or body, and facial expressions to fluidly express a speaker is thoughts (Zafrulla et al., 2011). Even though this thesis does not present the solution for automatically translating sign language into speech, or written text, it can contribute for some of the tools to do it in the future.

- **Physiotherapy** - is a health care profession primarily concerned with the recovery of impairments and disabilities and the promotion of mobility, functional ability, quality of life and movement potential through examination, evaluation, diagnosis and physical intervention. Most of the physical work of recovery consists in repeating certain movements or gestures over and over. With a gesture recognition program working with the patient, he can easily get automatic feedback if the gesture he is performing is correct or not, without the need of a physiotherapist presence (Ravi, 2013).

- **Automatic gesture notation** - Just like in music there is the music score, in the performing arts there are two main currents that allow to keep a written score of a choreography. They are Labanotation (Loke et al., 2005) or Benesh Movement Notation (Harrison et al., 1992). Usually the notation in done by hand by the choreographer or performer himself. This could be done automatically by developing further the framework presented here, therefore enabling the performer to record and annotate his practice for later offline revision or share it with others (Kahol et al., 2004).

Further applications are possible, extending the spectrum of research, these may include:

- enabling very young children to interact with computers;

- monitoring medical patients emotional states or stress levels;

- navigating and/or manipulating in virtual environments;

1.7 Main Contributions

This dissertation includes several contributions that can be divided in two facets: conceptual and implementation. Conceptual contributions are related with abstract concepts such as ideas, algorithms, studies and theoretical frameworks. Implementation contributions are related with the development of tools and specifications.

One conceptual contribution is the proposal and experimental validation of an efficient and modular, real-time gesture recognition framework for the performance context (one is tempted to consider the study and literature review concerning gestures also a conceptual contribution).

As implementation contributions there are the various modular tools developed in the scope of this thesis and implemented as open-source software, in the form of *addons* for openFrameworks[2], namely:

- a skeleton joint representation module - allows the visual feedback of the subject being captured;
- an Open Sound Control (OSC) transmission module which is able to read and transmit data in real-time from the Vicon Blade Motion Capture[3] proprietary program;
- a gesture recognition module based in Dynamic Time Warping (DTW);
- a gesture recognition module based in Hidden Markov Models (HMM);
- the entire framework (named ZatLab System) consisting on the tools listed previously, working together in a single operational framework for the recognition of real-time gestures and event triggering.

[2] `http://www.openframeworks.cc/` - openFrameworks is a powerfull C++ toolkit designed to develop real-time projects. Nowadays, is a popular platform for experiments in generative sound art, creating interactive installations and audiovisual performances.

[3] `http://www.vicon.com/software/blade`

A detailed description about the software implementation of these contributions will be provided in Chapter 5.

Additionally, the gestures recorded on the course of this study are also available[4] for further scientific research. These constitute a data bank of 5 different gestures with 290 samples each, resulting in 1450 gesture samples.

1.8 Publications Related to the Thesis

The research work presented in this thesis has resulted in the collaborative publications[5] listed below:

- André Baltazar, Carlos Guedes, Fabien Gouyon, and Bruce Pennycook. A Real-time Human Body Skeletonization Algorithm for MAX/MSP/JITTER. In Proceedings of the International Computer Music Conference, 2010 (Baltazar et al., 2010);

- Andre Baltazar, Luis Gustavo Martins, and Jaime S. Cardoso. ZATLAB: A Gesture Analysis System to Music Interaction. In Proceedings of the 6th International Conference on Digital Arts (ARTECH 2012) (Baltazar et al., 2012).

- Andre Baltazar, Luis Gustavo Martins. Zatlab: A Framework for Gesture Recognition and Performance Interaction. Book chapter in “Innovative Teaching Strategies and New Learning Paradigms in Computer Programming”, IGI Global 2014 (in print process).

Upon this thesis completion there were also papers submitted to other conferences, but the results are still pending evaluation.

[4] The gesture data bank is available at `http://andrebaltazar.wordpress.com`

[5] The articles are available in PDF format at `http://ucp.academia.edu/AndreBaltazar`.

1.9 Outline of the Dissertation

Six chapters compose this dissertation. The remainder of the thesis is organized as follows.

Chapter 2 focuses on the main aspects involved in gesture analysis. It discusses the various definitions of gesture, its understanding, observation and recognition. An overview of the importance gestures have in human life is also presented, followed by a discussion about how they are perceived and evaluated in different research fields. The remainder of the chapter presents some of the technological approaches to capture human movements and the chapter concludes with a review of some of the most important and relevant work previously conducted in the field of human movement analysis and gesture recognition.

Chapter 3 introduces the Zatlab System, a computational framework for the real-time recognition of gestures. Following an overview of the framework, a description of its main modules is presented. These include the Data Acquisition, the Data Processing, the Gesture Recognition and the Triggers Output Modules. The description of each Module will include the theory that supports the respective implementation, explained in Chapter 5. Finishing this chapter is a description of the available operation modes of the framework.

Chapter 4 presents a set of evaluation experiments and application scenarios where several aspects of the gesture recognition framework proposed in Chapter 3 are tested and validated experimentally. The research methodology is described, followed by the evaluation model and the respective experiment design. These will allow the understanding of evaluations performed. The questionnaire results will allow to assess the framework acceptability. Experimental results are presented for the main recognition algorithms and the latency of the Motion Capture (MoCap) technologies. The Chapter closes with the description of the artistic applications of the framework.

Chapter 5 discusses the most relevant aspects of the software implementation of the gesture recognition framework proposed in this thesis. The design requirements, implementation strategies and the major contributions towards the development of an open source software framework for gesture recognition are put into perspective and ultimately justify the adoption of the openFrameworks framework as the base software platform. The software implementation of the different processing algorithms that comprise the methods proposed in this thesis are detailed.

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

This thesis also comprises an appendix. This includes additional and detailed information about the ML algorithms described in Chapter 3 and also the questionnaire done for the experimental validation of the framework, described in Chapter 4.

Chapter 2

Gestures

"For an action to be treated as a gesture it must have features which make it stand out as such."

(Kendon, 1980)

2.1 Introduction

As Godoy (Godoy and Leman, 2009) refers, there is no clear definition of what a gesture is: *"Given the different contexts in which gestures appear, and their close relationship to movement and meaning, one may be tempted to say that the notion of gesture is too broad, ill-defined, and perhaps too vague."* This work is focused on gesture recognition, so there is intrinsically a demand for the explanation and definition of terms that are not well clarified.

This chapter is dedicated to the understanding and definition of a gesture and how it can be captured and recognized. It will also discuss the previous works published on the research field of this thesis and present a review and technical comparison of the different MoCap systems available nowadays. This section will provide valuable input for the development of the proposed framework.

2.2 Understanding Gestures

The human movement (Zhao and Badler, 2001) can be involuntary, subconscious, that occurs for biological or physiological purposes (e.g. blinking, breathing, balancing), or voluntary, conscious like those task-driven actions such as speaking or running to get somewhere. There is also a wide class of movements that fall in between these two, having both the voluntary and involuntary qualities. Such movements are the ones that occur in an artistic performance or music concert and perhaps unconsciously with other activities. These can range from leg and foot coordination enabling walking, till the communicative gestures, such as facial expressions, expressive limb gestures and postural attitude. The communicative gestures are the focus of this work and thus, their definition is of central importance. The word gesture on the remainder of this thesis will always refer to this notion of communicative gesture.

A good perspective on how to distinguish movement from gesture is given by Kurtenbach and Hulteen (Wachsmuth and Fröhlich, 1998), they state that *"A gesture is a motion of the body that contains information. Waving goodbye is a gesture. Pressing a key on keyboard is not a gesture because the motion of a finger on its way to hitting a key is neither observed nor significant. All that matters is which key was pressed. Pressing the key is highlighted as the meaning-bearing component, while the rest of the movement of the person is considered irrelevant."*.

Actually, there is no single universally accepted definition of what a gesture actually is. Depending on the domain of research one will find different meanings (Zhao and Badler, 2001). These domains can range from the psychological-linguistic, to the cognitive science or the performative arts. In the following subsections the different approaches will be explained.

2.2.1 Gestures in the psychological-linguistic domain

In psychological-linguistic domain, there are three authors that have made significant contributions, following the seminal work David Efron started in the 40s (re-issued later (Efron, 1972)). They are Kendon (Kendon, 1970, 8, 9), McNeill & Levy (McNeill, 1985, 9; McNeill and Levy, 1982), and Rimé & Schiaratura (Feldman and Rimé, 1991; Rimé, 1982)).

Kendon, presented the following definition: *"...for an action to be treated as a gesture it must have features which make it stand out as such."* Although this is clearly not a definition, it suggests the analysis of features as classification characteristics. Kendon started his research by attempting to determine if people recognize gestures when they watched videos of subjects talking in a foreign language (unknown by the viewers) (Kendon, 1970). He reported that the viewers had no trouble finding out gestures. Observing the relations between speech and gesture, he proposed his gesticulation theory. A gesture is the "*nucleus of movement with definite form and enhanced dynamic qualities (...) preceded by a preparatory movement and succeeded by a movement which either moves the limb back to is rest position or repositions it for the beginning of a new gesture phrase.*" ((Kendon, 1980) pp.34). Kendon also noticed different modes of expression, depending on the context gestures are used. For example, gestures are used more often when the conditions of speech reception occur in a noisy environment or by limited knowledge of a foreign language. Or, when is difficult to express something through speech, this may be conveyed by gesture, specially regarding spatial information such as distance, orientation and trajectories. Kendon concluded stating that speech and gestures are one integrated system.

Mcneill & Levy made the same discovery as Kendon: speech and gesture are part of a coherent whole (McNeill and Levy, 1982). Through their experiments they have found gestures can present meaning in a form fundamentally different from that of speech. First, gestures are non-combinatoric (two gestures done together do not combine to form a larger, more complex gesture), second there is no hierarchical structure of gestures made out of other gestures, which contrasts with the hierarchical structure of language, and third, gestures do not share linguistic properties such as standard forms and duality of patterning.

Rimé & Schiaratura conduct experiments that involved a speaker talking to a listener with and without visual contact. They found that the frequency of gestures was not particularly affected by the presence or absence of mutual visibility of partners (Rimé, 1982). Thus, they concluded the gesture had some function or purpose for the speaker, besides the communicative aspect to the listener. Actually, they found that when a speaker is restricted of his gestures during his speech, he tends to give poorer descriptions and induce more compensatory motor activity of eyebrows and fingers. Furthermore, careful analysis of the semantic content of the speech showed that the speakers used more words but the speech was less clear and less fluid (Feldman and Rimé, 1991). Again, this empirical evidence can be interpreted to support theories like McNeill, that gesture and speech are elements of a single integrated system.

2.2.2 Gestures in the cognitive science domain

The cognitive science domain is a research area also related to psychology but with a strong branch on Artificial Intelligence (AI). The research consists in building cognitive models in order to understand human behavior. If the model can reproduce human behavior under certain assumptions, it will also provide answer about human behavior in different assumptions. By changing these assumptions one can achieve different explorations and thus, different results. The speech and gesture relation has been broadly studied in the cognitive science context (Feyereisen and de Lannoy, 1991), but yielded contradictory hypotheses. Here are explained two of the more prominent but contradictory hypotheses.

In one hand, there is the competitive model. This model generated empirical evidence that the gestural stroke phases alternate with rest phases and the gesture is sometimes prevented or delayed instead of being done simultaneously with the process of thought expression (Feyereisen and de Lannoy, 1991). According to this, the researchers hypothesize that gesture and speech are two rival tasks, this is, assuming a perspective where resources are limited and both of them compete for it, the attention load required for one task reduces the amount that can be allocated to the other task. This means that speaking and gesturing implies to divide our attention between both, and if the attention load reaches its maximum, hesitation pauses may occur.

On the other hand, there is the coactivation model. In this model, the researchers assumed there is an inevitable activation of the gestural system during speech production. Thus, the gesture is a visible manifestation of the speaker ongoing thinking process. The hypothesis presented is that gesture and speech share the same origin and are triggered simultaneously, and then separated in different output channels. However, this model presents some problems because implies a direct relationship between speech and gesture. Accordingly to that the more one speaks, the more gestures are performed. But in some circumstances this may be false, (i.e. for sign language interpreters) the gestural rate and speech fluency can be inversely related (Feyereisen and de Lannoy, 1991).

In short, the cognitive perspective does not provide a consistent answer. Some defend the coactivation, others the competition. These different hypotheses still need to be further investigated and reviewed. Maybe with different approaches from psychology, neurophysiology and even pathology, some day one will be able to delineate the functioning of communicative gesture.

2.2.3 Gestures in the performing arts domain

Gestures are seen as the most appropriate mean of expression for theater and dance. Performers use gestures to communicate to an audience, either if it is a comedy or a tragedy, either if a character is good or evil. Thus, through gestures, actors enhance the emotional content of their stories and characters. For the contemporary dance and avant-garde theater the gesture is not simply a complement or a decoration. It is yes, the source, the cause and the conductor thread (Royce, 1984).

In this performative domain, gestures can have different interpretations due to culture specifications. In ballet, the gesture is based in greco-roman ideals of posture and and movement. Standing straight, with slow, expansive and gracious movements will portray an elegant and graceful ballerina, while narrow, clumsy and rough movements will be seen as ugly and poor. Also in a play, the director must plan the combined movement of the cast, treating the movement as an extension of the line, mass and form. The actors themselves must be aware the quantity of movement used in a gesture, and how much space they are occupying in a stage, in order to transmit energy or weakness. The length of a gesture, either short or long, its intensity, either strong or soft, everything will add and convey emotional content. One wrong gesture can ruin a character or all the stage dynamics. Thus, adequately planned, chosen and executed, gestures can create a mood, or a state of mind and arouse an emotional response from the audience (Dietrich, 1983).

Also in the music research field, body movement has been often related to the notion of gesture. The reason is that many musical activities (performance, conducting, dancing) involve body movements that evoke meanings, and therefore these movements are called gestures (Godoy and Leman, 2009). Reading about musical gestures, there is a curious research conducted by Leante (Leante, 2007) in which she used available footage of the rock band Genesis and investigated the gestures used by the singer, Peter Gabriel. Using the categories defined by

McNeill, Rimé and Schiaratura and Kendon, seen previously in this section, Leante investigated how the singer used gestures to enrich the song “The Musical Box”. For instance, to highlight parts of the song, such as “wanting”, “feeling”, “knowing” and “touching”, Peter used a single pantomimic gesture (moving the hands as though “grabbing” something). Leante argues this gesture conveys a stronger sense of physicality, or tactility than what is expressed in the text, and adds to the pathos and emotion of the lyrics.

To summarize, the study of gesture is a broad research field, with long branches extending from the rather philosophical, theoretical approaches, till the more technological, experimental areas. This gives a cross-disciplinary nature to the research (what is good) but also adds to the difficulty on defining precisely what is a gesture. What is common with the different approaches is that a gesture implies expression, communication and a purpose. It is the voluntary act of synthesizing movements to achieve a goal, fulfill an intention.

2.3 Notating Gestures

There is also a parallel research area devoted to the human movement and gesture notation (Craine and MacKrell, 2004). This departed from the importance of registering and maintaining records of traditional dance, in particular, ballet. The first fully comprehensive system of notation was established on the 20th century, which means that many ballets prior to this date were either lost or handed down in partial form.

The fact that gestures requires both spatial and temporal notation makes it hard to record accurately on paper, although attempts to do so date back to the 15th century. These are proven by surviving manuscripts of that era, e.g. *Margherita d’Austria’s Livre des Basses Danses* (in 1460). Since then, the notation has evolved in its accuracy and elaboration, with the first sophisticated attempt at a system published by Feuillet in his *Chorégraphie ou L'Art de décrire la danse par caractères, figures et signes demonstratifs* (Paris, 1700), which was based on ideas originated by Beauchamps (Pierce, 1998). This became popular all around Europe as a means of recording and teaching dances. It depicted the floor patterns of the dances, adding signs for the direction of each step as well as for turns, beats, and other details of footwork.

Already on the 19th century, the idea of writing down dance in a manner similar to music was first developed by B. Klemm in 1855 and further developed by the Russian dancer Stepanov (Craine and MacKrell, 2004). In his *Alphabet des mouvements du corps humain* (Paris, 1892) he placed movement symbols on a special stave while recording the floor patterns above it.

During the 20th century there were attempts at more rigorous and complete notation based on abstract symbols, in order to record styles of movement other than ballet. The most famous of these was originated by Laban and first published in 1926 in his *Choreographie* (Maletic, 1987). Now widely referred to as Labanotation (Loke et al., 2005), this system uses a vertical staff to represent the body and has symbols that indicate not only the position but also the direction, duration, and the quality of any movement.

Another widely used system is that developed by Rudolf and Joan Benesh. This began as a shorthand for notating ballets and was first published as *An Introduction to Benesh Dance Notation* (London, 1956) (Harrison et al., 1992). Now termed *Choreology*, the Benesh system (see Figure 2.1) uses a five-line musical stave running horizontally across the page with abstract stick figures indicating the position of the body and special symbols indicating timing, direction, etc. Though most widely used in ballet companies, such as the Royal Academy of Dance[1], Choreology has subsequently evolved to deal with non-classical movement also, and together with Labanotation (presented in Figure 2.2) is the most internationally

[1] https://www.rad.org.uk/study/Benesh/how-benesh-movement-notation-works

used system. Other systems have been less widely adopted, for example N. Eshkol and A. Wachman's, published in Movement Notation (London, 1958), which is based on a mathematical record of the degree of rotation made by each of the moving parts of the body.

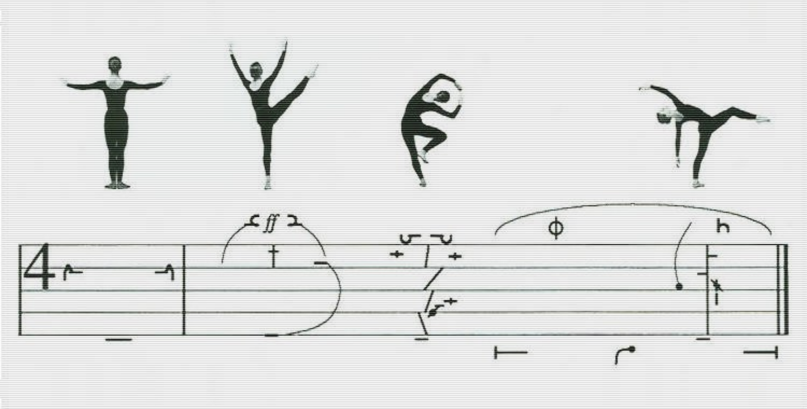


Figure 2.1: An example how movement is described using Benesh Notation, from the Royal Academy of Dance.

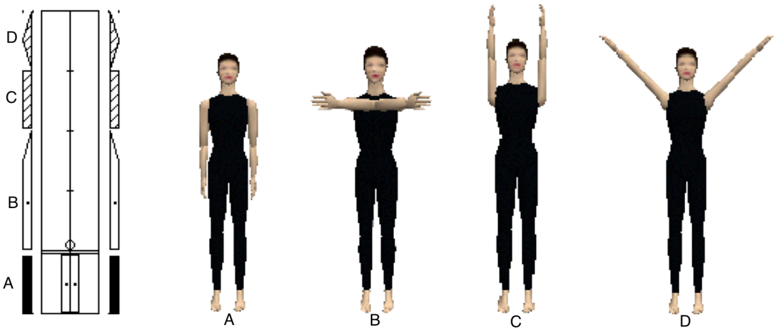


Figure 2.2: An example how movement is described using Labanotation, adapted from (Ryman, 2001) .

Recently the availability of simple and inexpensive video equipment has provided another means of recording dances. Nevertheless, the automatic notation of dance still requires further research until satisfactory results can be achieved.

Although the notation and transcription of human movements and gestures is not the main focus of this thesis, the work developed can be used for future application in this field.

In the next sections it will be described the state of art of technology that allows capturing human movements for future gesture recognition.

2.4 Motion and Gesture Capture Technology

Although a few years ago MoCap systems were only available for the movie and animation industries, due to the high costs of the technology, nowadays, thanks to the decrease in the hardware prices, there are systems somehow affordable for research purpose or even entertainment.

There are four main types of motion capture systems, they can be: Magnetic, Mechanical, Optical and Inertial (Wong, 2007). Now, there is also development and use of a combination of two or more of these techniques, creating thus the Hybrid Motion Capture Systems.

The next sub sections present a technological review and comparison of the several MoCap systems available nowadays. This will provide valuable input for the decisions made on the path taken to develop the framework.

2.4.1 Magnetic Motion Capture Systems

In these systems, magnetic sensors are placed on the object being tracked (Su et al., 2003), (Mitobe et al., 2006). These are able to measure the magnetic field generated by a magnetic transmitter. Based on the measurements, one can calculate the position and orientation of the sensors in relation to the transmitter. The main advantages of the magnetic motion capture systems are they are not affected by occlusion and can measure absolute positioning of an object in three dimensional (3D) space. However there are also disadvantages. The strength of magnetic fields decreases greatly with the distance between transmitter and sensor. The data acquired is noisier than the one obtained using optical systems. And besides the movement constraint, the magnetic motion capture is very sensitive to magnetic interference, which can occur often due to all the wiring and technology surrounding us everywhere. This makes the magnetic motion capture systems to be used only in very specific projects and most of the times depreciated against the other systems.

Yusu (Su et al., 2003) used a model like this to accurately model and capture the motion of the human in order to detect the tremor evident in subjects with Parkinson's disease. He used 11 three-dimensional electromagnetic sensors to model the human hand including all the phalanges. A capture rate of 10 measurements per second was achieved. A discrete Fourier analysis has been applied to extract the tremor frequency from the sensor data time series. The technique described provides an objective and quantitative method for the analysis of clinic conditions, such as Parkinson disease and essential tremor, as a way to assess the effect of therapeutic interventions.

Mitobe (Mitobe et al., 2006), described a Magnetic MoCap system for the human hands. The magnetic tracker that is composed from one transmitter and sixteen receiver, can calculate the distance (x, y, z) from a transmitter to a receiver. Each receiver was attached to each finger using Kinesiotex tape and liquid type plastic in order to prevent the receiver from sliding. The hand MoCap can measure the data (six degree of freedom) of 32 receivers at the rate of 240Hz simultaneously. That resulted in the cumbersome setup demonstrated in the Figure 2.3.

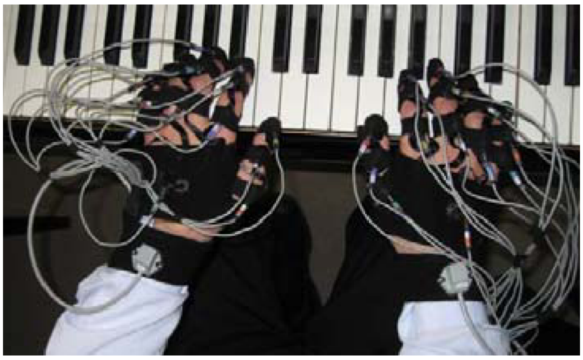


Figure 2.3: Electromagnetic hand MoCap system, from (Mitobe et al., 2006).

2.4.2 Mechanical Motion Capture Systems

In the mechanical motion capture systems, an exoskeleton is attached to the person (or object) being tracked. This exoskeleton is equipped with various sensors at each joint of the subject being tracked. The mechanical segments accompany the movements of the subject, which allow the sensors at the joints to determine the relative motion of the subject. An example of such system is the "IGS 180 Range" motion capture from Animazoo[2]. The main advantages of such systems are their in-susceptibility to occlusion, the high sampling rate of data (since it does not require a lot of processing to extract the motion information) and the low cost of development, due to their construction made of primarily plastic or metal rods and potentiometers that act as sensors. As disadvantages, they are unable to directly measure the absolute positioning of the object, their use limits or constraints some of the movements and they are only suitable to objects with movable joints such as human performers.

In the 2010 New Instruments for Musical Expression (NIME) Conference there was a paper presented by Collins (Collins and Kiefer, 2010) addressing the use of exoskeletons. Their paper describes the initial experiments in mapping the suit control data to sonic attributes for musical purposes. As the suit provides up to 66 channels of information, they confronted a challenging mapping problem, and described techniques for automatic calibration, and the use of echo state networks for dimensionality reduction. The Figure 2.4 presents the exoskeleton used in Collins research.

2.4.3 Optical Motion Capture Systems

The optical motion capture systems usually imply the use of markers on the object being tracked. Optical cameras then track the individual markers on the object. The

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

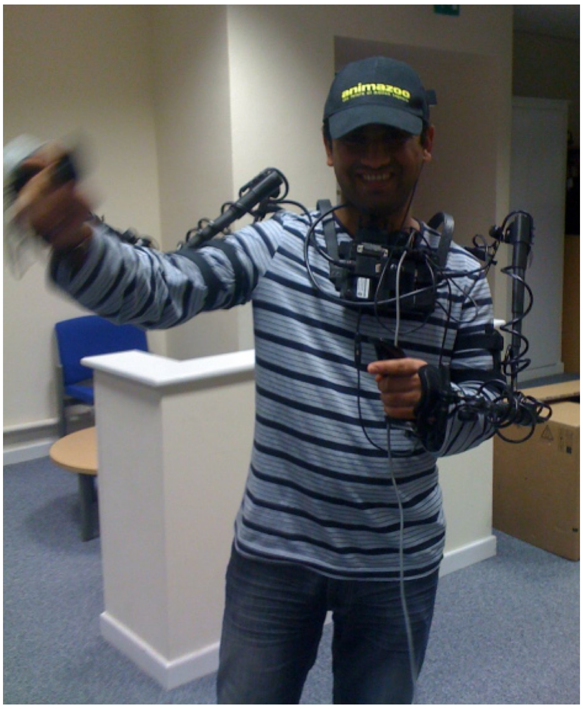


Figure 2.4: Animazoo exoskeleton, from (Collins and Kiefer, 2010).

setup consists in various cameras surrounding the object so software can process the images captured and triangulate the 3D position of each marker. These can be grouped in two categories: Active or Passive. In the active optical systems, such as the Optotrak from NDigital[3], each marker contains an embedded emitter with a unique identification. This makes the system more reliable when overlaps occur and decreases the processing time required to track and distinguish individual markers. The passive, such as the commercially produced by Vicon[4] consist in placing retro-reflective markers on the object and then emitting infra-red light to illuminate the markers so they can be tracked by the cameras (see example on Figure 2.6)

Besides these two techniques, there are also markerless options available. With the launch of 3D/depth cameras such as the Microsoft Kinect and the development of software that can easily track and map the human skeleton, such as the OpenNi

[3] http://www.ndigital.com/

[4] http://vicon.com/

library (Villaroman et al., 2011) nowadays one can do a MoCap laboratory in almost any room by just placing one, two or several of these cameras, without the need of markers or any special conditions.

There are a number of advantages in the use of optical systems over other methods. First, they can measure the absolute positioning of an object in the 3D space. Second, the data obtained is less noisy than the above-mentioned techniques. Third, the cameras are flexible in terms of positioning and therefore allow for greater freedom of movements and to track more then one object at the same time. Off course there are also some disadvantages in these systems. The most relevant is the susceptibility to occlusion, this can be reduced by increment the number of cameras, but that will also increase the computational and economical costs. Another disadvantage is that everything implies a lot more of computational processing, the software has the complex task of extracting information from visual data, then determining the absolute positioning of each marker and finally the 3D orientation has to be calculated based on the relative positioning between neighboring markers.

In this Section one will present the Kinect and Vicon MoCap technologies with more detail, since this will support the choice made in Section 3.3.

2.4.3.1 The Kinect

The advances made in 3D depth cameras, in the recent years, such as Microsoft Kinect sensors have open new possibilities to multimedia computing (Zhang, 2012a). Kinect was developed to revolutionize the way people play games and how they experience entertainment. The key feature is on the third dimension information, the depth. The foreground - background separation and tracking of an object (or human) in a scene has always been an active research field in CV, but always considered a formidably difficult task for video cameras. The Kinect sensor allows the computer to directly capture the third dimension (depth) of the scene, enabling thus to reduce the task of foreground - background separation to a simple threshold measure.

The impact of the Kinect has extended far beyond the gaming industry. Being available anywhere gaming consoles are sold (almost everywhere in the world) and having a reasonable low price, enables it to be used among the communities of researchers and practitioners in computer science, electronic engineering and robotics, allowing them to develop creative new ways to interact with machines (Stowers et al., 2011) and to perform other tasks, like helping in physical rehabilitation (Chang et al., 2011) for instance.

The Kinect sensor incorporates several advanced sensing hardware. Not only contains a depth sensor and a color camera, but also includes a four-microphone array (although one will not go into the sound specifications of the device). Figure 2.5 shows the infrared (IR) projector, the color camera, and the IR camera. The depth sensor consists of an IR projector combined with an IR camera (a Complementary Metal-Oxide Semiconductor (CMOS) sensor). The IR projector consists in an IR laser diffracted into a set of IR dots. Knowing the relative geometry between the IR projector and the IR camera, as well as the projected IR dots, makes it possible to triangulate each dot position, thus reconstructing a 3D map of the scene and matching it to the color camera capture.

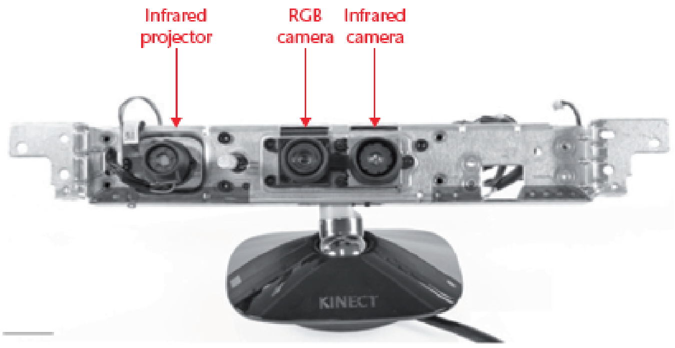


Figure 2.5: The infrared projector, infrared camera and the RGB camera inside a Kinect, from (Zhang, 2012a).

The depth values are encoded with gray values; the darker a pixel, the closer the point is to the camera in space. The black pixels indicate that no depth values are available for those pixels. This might happen if the points are too far (and the depth values cannot be computed accurately), are too close (there is a blind region due to limited fields of view for the projector and the camera), are in the cast shadow of the projector (there are no IR dots), or reflect poor IR lights (such as hairs or specular surfaces).

Besides the advances in hardware, also the software that accompanies the Kinect brought innovation and advances in skeletal tracking. The fact it was developed primarily for commercial purposes pressured the Microsoft developers to full-proof the algorithms and make it robust enough to detect almost every person on the planet, in every household set, without any calibration (Zhang, 2012a).

2.4.3.2 The Vicon MoCap

The Vicon MoCap is a system commercially developed to track human or other movement in a room-size space. Spheres covered with reflective tape, known as markers, are placed on visual reference points on different parts of the human body. The different cameras surrounding the scene project IR light and capture the reflection of the markers. Being the same marker captured by three or more different cameras, its absolute position in the space is calculated by triangulation. Due to the fact this is a proprietary system, there are no details published about software implementations.

The position of the markers is calculated in the Vicon Blade program (a commercially closed program). The data derived from the captured motion are most commonly saved to disk, as a Vicon-standardized .C3D file. Captured data files are then used as offline input to an animation program such as Maya[5] for realistic generation of lifelike animated characters, or used for bio mechanical studies of

[5] http://www.autodesk.com/products/autodesk-maya/overview

body motion (sports, physical therapy, ergonomics, etc.). In the .C3D format, each frame of information is represented as a list consisting of Cartesian x, y, z coordinates in 3D space for each marker. The Vicon system at Universidade Católica Portuguesa (UCP) reports up to 120 frames per second (fps). The user determines the ordering of the markers in the list when recording the data.

One good example of the use of a Vicon MoCap system is the work in progress in the MAPP project, developed at CITAR and UCP, being the author of this thesis directly involved in the technological aspects of the study. This project is developed under the supervision of Dra. Sofia Lourenço (Lourenço, 2010) and has the goal of detecting different pianists schools according to the gestures performed by professional pianists playing.

Summarizing the project, there are three main traditional schools currents of piano: the Russian, the French and the German. Each school can be characterized by subtle differences in the expressive movements when in live performance context. For instance, if one school constricts the movement, other encourages it, or if in one school one can identify a curl of the wrists to reach the piano keyboard, in other are the elbow and forearm that change. The use of the MoCap system in this project allows to gather data to do a statistical analysis of the pianists movements. The Figure 2.6 presents the setup of the MoCap capture sessions (in this case Dra. Sofia Lourenço during the setup tests).

2.4.4 Sensor Motion Capture Systems

This kind of systems employs inertial sensors, such as gyroscopes and accelerometers to measure the relative motion of the object being tracked. Gyroscopes are used to determine orientation while accelerometers are used to determine accelerations. By placing the sensors normal to each other, inertial motion capture systems can determine the relative 3D orientation and position at a particular joint. Usually this systems are used for vehicle navigation and tracking (e.g the work of

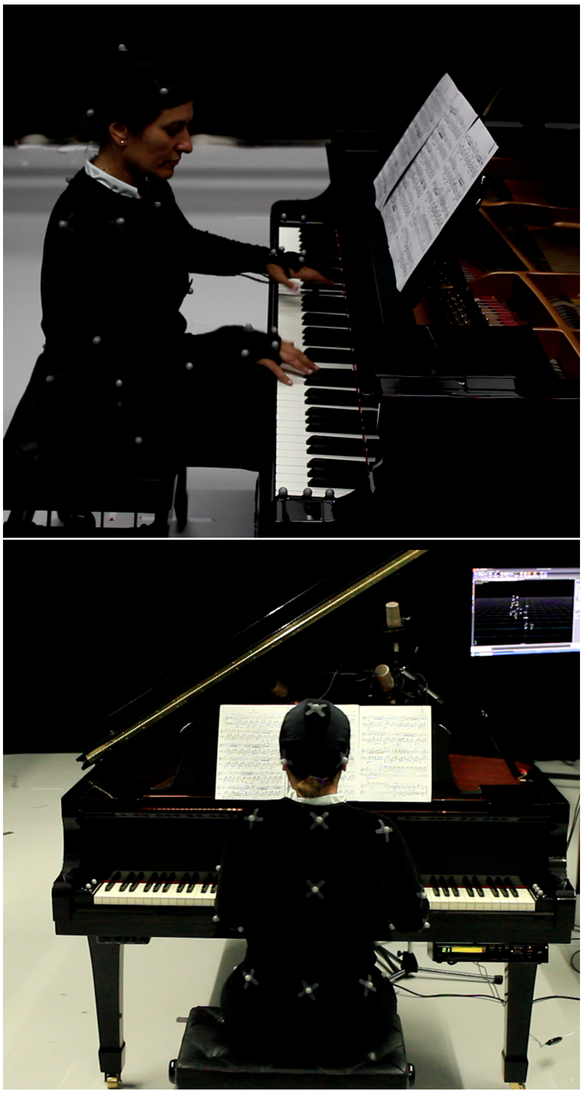


Figure 2.6: Pianist performing on one of the MAPP project capture sessions.

the portuguese researcher Jorge Lobo (Lobo et al., 1995)) and human motion tracking (Luinge, 2002), (Roetenberg, 2006).

The main advantages are the ability to do a direct measure in six degrees of freedom, which cannot be achieved using an optical system. Also, it only needs one sensor for tracking both 3D position and orientation, which is great for wireless 3D controllers, such as the Wii-Remote by Nintendo (Shih et al., 2010). Another advantage is that it can acquire data at high sampling rates, without the processing requirements of the optical systems, neither the occlusion problems of the former.

As disadvantages, it has three that are relevant. First, like what happens in the mechanical MoCap, it can not directly measure the absolute positioning of the object. Second, the inertial sensors do not measure explicitly the position and orientation, they derive it from the measured acceleration and angular velocity. This leads to rapid error accumulation over time and therefore these are stable only in short periods of time. And third, it also implies the use of sensors attached to the body being measured, which can constrict movements.

One can see on Figure 2.7 the artist Tom Tlalim (featured in an article from Popular Science[6]), creator of a full-body, eight-piece "suit" of Wiimotes interfaces with custom software to turn his entire body into an electronic instrument.

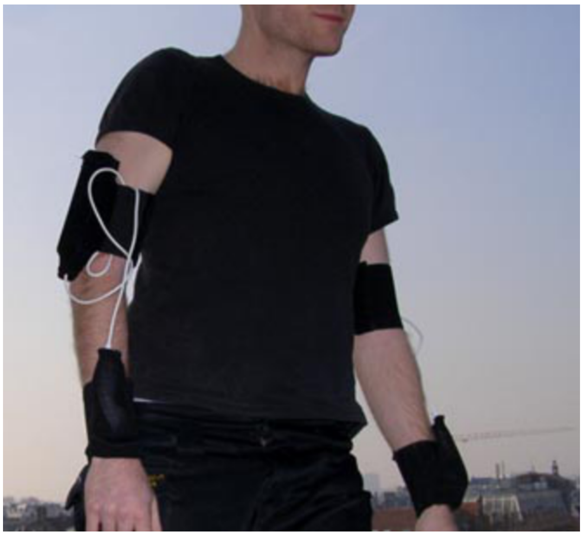


Figure 2.7: Tom Tlalim wearing his Wii-Remotes suit (image extracted from Popular Science)

There is also the example of the BioMuse (Tanaka, 2000), a biosignal musical interface. In this case instead of gyroscopes and accelerometers, the system takes bioelectrical signals in the form of electroencephalogram, electromyogram and electrooculogram and translates them into serial digital data and MIDI. The placement of electrodes in different locations of the arms allows to detect not only

[6] http://www.popsci.com/entertainment-gaming/article/2008-02/dancing-song-full-body-wiimote-music-controller-suit

the arm movements, but also the muscles contraction.

2.4.5 Hybrid Motion Capture Systems

Other works on motion capture systems center around hybrid systems, which make use of different sensors to provide more robust and accurate 3D position and orientation measurements. These include systems which combines GPS and inertial sensors to localize an autonomous land vehicle (Caron et al., 2006), magnetic and inertial (Roetenberg et al., 2007) (the magnetic tracker is able to calculate relative distances and orientations between body segments while the inertial tracker registers accelerations and angular rates), magnetic and optical (Joguet et al., 2003) and inertial and optical (Blomster, 2006), (Foxlin and Naimark, 2003).

For instance, Foxlin (Foxlin and Naimark, 2003) presented a self-tracker, using robust software that fuses data from inertial and vision sensors, as an approach to use in the Augmented Reality context. Self-trackers have the advantage that objects can be tracked over an extremely wide area, when compared to infrastructure-based trackers, without the prohibitive cost of an extensive network of sensors or emitters to track them. Thus, Foxlin develop a self-tracker which is small enough to wear on a belt, low cost, easy to install and self-calibrate. The Figure 2.8 shows the self-tracker being tested.

The Table 2.1 presents a comparative summary of various characteristics of the different MoCap systems.

Analyzing this table to choose one of the technologies to work with to develop a framework without a set of specifications can be very ambiguous. Each method has its pros and cons along several dimensions, such as: accuracy, resolution, latency, range, user comfort, and cost amongst others. If on the one hand a gestural interface based in a glove or exoskeleton allows for great accuracy for joints rotation, on the other hand this typically require the user to wear a cumber-

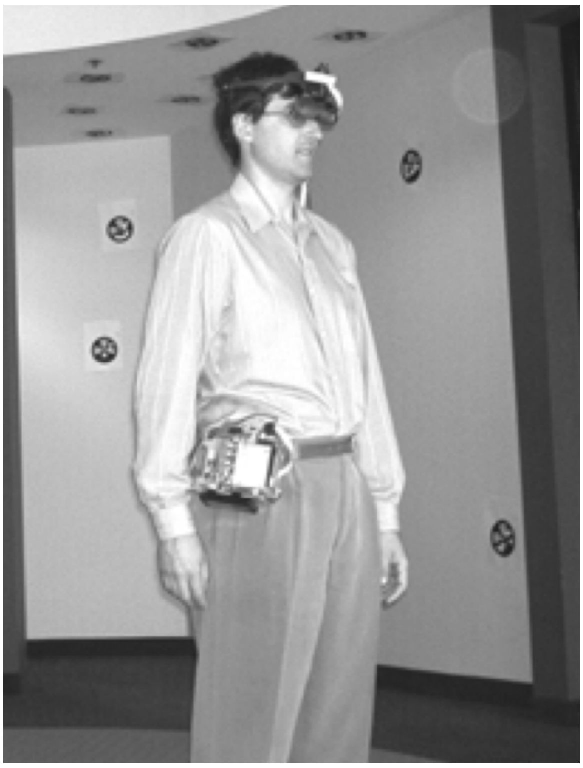
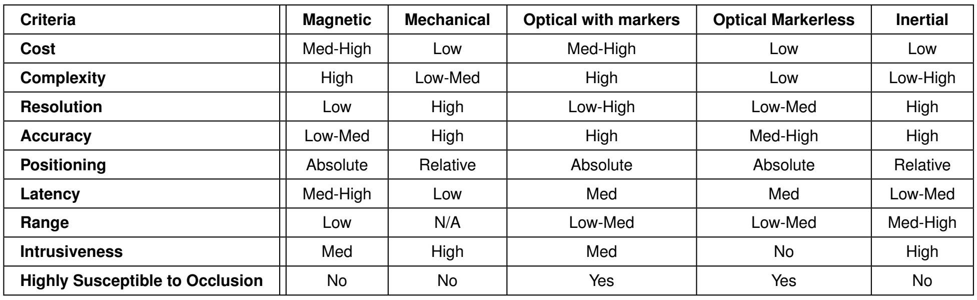


Figure 2.8: The self-tracker device in tests, from (Foxlin and Naimark, 2003)

some device and carry a load of cables to connect to a computer. This constricts the user movements and consequently his gestures. Vision based techniques overcome this restrictions, but have to deal with other problems related to occlusion of parts of the user body. Other techniques have also some advantages but some disadvantages as well. That is why gesture recognition is not trivial, there is not one proven and foolproof method to do it. It is always dependent on the goals one wants to achieve and the limiting conditions to achieve them.

The system chosen has to detect and capture human movements, but also recognize the gestures in a continuous stream of movement. While humans have few problems separating a hand gesture (e.g. waving goodbye), this is much more problematic for computers. This is not only due to the remarkable capacity of visual scene analysis in humans, but is also due to the fact that we understand the intended meaning of the gesture based on its context and on our life-long experience of multimodal communication. The next section will present the difficulties

Table 2.1: Comparison of existing motion capture techniques, based on (Wong, 2007).



Criteria	Magnetic	Mechanical	Optical with markers	Optical Markerless	Inertial
Cost	Med-High	Low	Med-High	Low	Low
Complexity	High	Low-Med	High	Low	Low-High
Resolution	Low	High	Low-High	Low-Med	High
Accuracy	Low-Med	High	High	Med-High	High
Positioning	Absolute	Relative	Absolute	Absolute	Relative
Latency	Med-High	Low	Med	Med	Low-Med
Range	Low	N/A	Low-Med	Low-Med	Med-High
Intrusiveness	Med	High	Med	No	High
Highly Susceptible to Occlusion	No	No	Yes	Yes	No

and some of the possible solutions for recognizing gestures.

2.5 Recognizing Gestures

Gesture recognition consists in recognizing meaningful expressions of motion by a human, either to communicate or to interact with the environment. Typically, the meaning of a gesture can be dependent on:

- the spatial information: where it occurs;
- the temporal information: when and how fast it occurs;
- pathic information: the path it takes;
- symbolic information: the sign it makes;
- affective information: its emotional quality.

Learning from the psycholinguistic research field, gestures can be physically distinguished from other movements mainly by four characteristics (Kendon, 1994):

1. gestures begin on a position of rest, move away from that position, and then return to rest;

2. gestures have what is commonly referred as a stroke, generally recognized as a moment of accented movement to denote the function of meaning of a movement;

3. one can identify a preparation phase before the stroke, and a recovery phase after it in which the hand and arm return to their rest position.

4. gestures are often symmetrical.

Furthermore, gestures are usually language and culture specific, nevertheless there are some commons to almost every society, such as:

- hand and arm gestures: recognition of hand poses, sign languages, and entertainment applications (allowing children to play and interact in virtual environments);

- head and face gestures (e.g. nodding or shaking of head; direction of eye gaze; raising the eyebrows; winking; flaring the nostrils; looks of surprise, happiness, disgust, fear, anger, sadness, contempt, etc.);

- body gestures: involvement of full body motion, as in tracking movements of two people interacting outdoors; analyzing movements of a dancer for generating matching music and graphics; recognizing human gaits for medical rehabilitation and athletic training.

Indeed, gestures can involve the hands, arms, face or even the entire body. They can be static, where the user assumes a certain pose, or dynamic, where the user treads a set of poses through time. Some gestures can also have both static and dynamic elements. To detect and recognize all this range of gestures one needs to specify where it begins and where it ends in terms of frames of movement, both in time and space. So the automatic recognition of gestures implies the temporal or spatial segmentation of the movement.

Besides, in order to determine the relevant aspects of a gesture, the human body position, the angles and rotations of its joints as well as their kinetic information (velocities, accelerations) need to be determined. This can be done (as seen in the previous section), either by using sensing devices attached to the user, or using cameras and CV techniques.

The next sections present the literature review on movement segmentation, analysis and feature extraction.

2.5.1 Movement Segmentation

In order to recognize gestures automatically, the computer must be able to segment a continuous movement at its temporal and spatial level. The spatial segmentation is the problem of identifying where the gesture starts and ends. Likewise the temporal segmentation is the problem of identifying when the gesture begins and when it ends.

The literature proposes various methods for recognizing gestures in a continuous stream of movement when the temporal segmentation is unknown. The suggested methods can be grouped into two basic approaches (Alon et al., 2009).

The first is the direct approach, where the temporal segmentation precedes gesture recognition. That is, first one calculates low level motion parameters, such as speed, acceleration, trajectory, and curvature (Kang et al., 2004) or mid-level motion parameters such as the activity of the human body (Kahol et al., 2004), and then looks to abrupt changes (e.g. zero-crossings) in parameters to identify candidate gesture boundaries. A major limitation of such methods is that they require every gesture to be preceded and followed by intervals of rest, a requirement that may not be satisfied in continuous gestures.

The second consists in the indirect approach, that intertwines the temporal segmentation with gesture recognition. This is, the limits of the gesture are detected by finding the sequence of unknown input that gives good recognition results, when tested against the models of previous trained gestures. Most indirect methods are based on extensions of Dynamic Programming, for example, DTW (Corradini, 2001a; Gillian et al., 2011) or various forms of HMM (Bevilacqua et al., 2005; Elmezain and Al-Hamadi, 2009; Rabiner, 1989). In those methods, the gesture endpoints are detected by comparing the recognition likelihood score to a threshold. The threshold can be fixed or adaptively computed (Yang et al., 2006).

Besides the movement segmentation, there are other crucial aspects for the recognition of gestures. These are the analysis of the movement and what features best describe it, presented next.

2.5.2 Movement Analysis and Feature Extraction

There are unlimited possibilities and some more research is needed to determine which features are best for gesture recognition. In the literature there are examples of a variety of motion features employed. Rubine (Rubine, 1991), used mainly geometrically based features to recognize simple pen gestures. Segen et al (Segen and Kumar, 1998) rely on local features such as “peaks” and “valleys” on the contour of the hand shape to help classify gestures. Zhao (Zhao and Badler, 2001), in other hand, defends the use of case specific categories accordingly to the following criteria:

- efficiently computable: each feature should be geometrically, algebraically, or incrementally computable, using only data available from the motion capture process;
- meaningful: features should be correlated to the motion qualities;
- minimum coverage: there should be sufficient features to capture and differentiate the motion qualities, but the feature set should not be redundant.

Al-Hamadi et al (Al-Hamadi et al., 2010), also addressed this problem when developing their hand gesture recognizing system. One of their main contributions were the tests performed to examine the influence of the various features of coordinates position, orientation angle and velocity in their gesture recognition system.

In essence, they concluded the best results were obtained when using a combination of the X and Y coordinates, the orientation angles and the respective velocity of the analyzed joint (these provided 98.33% accuracy). However, when using isolated features they realize the main contributor for the recognition was the orientation angle of the gesture path, with an accuracy of 96.94%. Furthermore, the velocity feature shows a lower discrimination power (57.22%) and the coordinates position feature result has the lowest recognition rate of 32.78%.

Also, Yoon et al (Yoon et al., 2001) in their approach to hand gesture recognition (using a 2D webcam) realize that when using separated movement features, the angle features had recognition rates of 87%, and that they are better than the recognition rate using the location or velocity features. Their conclusion was the most effective feature among the three basic features was the angle feature.

To complete this description about movement analysis and feature extraction, is important to mention gesture recognition can be divided in two sub problems (Zhao and Badler, 2001): feature representation and classification. Consequently, a complete gesture recognition framework consists in two assets: a representer and a classifier. The representer uses the raw data, captured through optical, magnetic, mechanical or hybrid sensors and outputs or creates an internal representation of the data. Often is a set of parameters and features extracted from the raw data in a convenient form to pass to the classifier (described in the following Chapter 3). The classifier, taking the features passed by the representer as input, outputs an appropriate classification (if one exists). Usually the classifier consists in various methods based in various fields of research (Mitra and Acharya, 2007), they range from statistical modeling, CV and pattern recognition, image processing and machine learning, making this topic a good example of multidisciplinary research.

The approaches can go from rather typical template matching, pattern recognition or neural networks in case of a static gesture (i.e. a pose) to the more complicated techniques, such as DTW, HMM in the case of dynamic gesture recognition. These algorithms will be further described in Chapter 3.

Once the features are extracted and the gestures recognized, these can be mapped to events using several techniques, next section presents a short review concerning the mapping of motion to computer music.

2.6 Mapping Motion to Computer Music

The mapping of human gestures or movements to music has been discussed for several years. It can be said the invention of the Theremin, in 1919, was a major driver of the development of music produced by electrical means and is more remarkable by being the first musical instrument without the need of physical contact to play. The results obtained by the instrument was somewhat more intricate than a simple oscillator, since the sound produced reflected the expressive quality of human movement (Winkler, 1995a). Already in the 60s, many composers have explored the human movements as a way of creating electronic music. From these stands out the collaborative work *"Variations V"*, in 1965, with music by John Cage and choreography by Merce Cunningham, in which the sounds were derived from movements of dancers (depending on their proximity to various sensors placed around the stage). With the technology constantly evolving, there has been a refinement and improvement of the various techniques of motion capture and the creation of several systems which show the feasibility of using computers to interpret and use human motion data creatively (as presented in the next Section 2.7). More recently the research questions are also related with the mapping[7] of these

[7] the word "mapping" refers to the correspondence between control parameters (derived from the human movements) and consequent events (e.g. sound synthesis parameters, visual effects, etc).

movements for sound synthesis and their performance in musical compositions. Specifically, how can composers create music based on motion data? How can movement be mapped to form and structure musical material?

Interactive music systems can be used to answer these questions by interpreting the data and extending the power of expression of the performer beyond simple relationship of one-to-one sound trigger, to include the control of multiple processes of composition, musical structure, signal processing and sound synthesis (Winkler, 1995a).

In reality there are two main points of view regarding the mapping in interactive systems (Hunt et al., 2000). The first states the mapping is a specific feature of a composition. The second states mapping is an integral part of the instrument. Besides these two currents, there are also three categories in which the mapping can be classified (Rovan et al., 1997):

- One-to-One Mapping - in this case each control parameter (i.e. gesture recognized) is assigned to one musical parameter. This is the simplest mapping scheme, but usually is also the least expressive approach.
- One-to-Many Mapping - Also known as divergent mapping, usually implies that one control parameter is used to control multiple musical parameters. Although it provides a more expressive experience when compared to the one-to-one mapping, it is nevertheless a macro approach not allowing the access and manipulation of micro features of the sound object.
- Many-to-One Mapping - i.e. the convergent mapping - This approach consists in coupling many control parameters to produce one musical parameter. This scheme is the most complex of them all and usually implies the performer to train and rehearse with the system in order to achieve effective control. Nevertheless this is far more expressive than the simpler mapping strategies.

In summary, it is important to mention that in terms of expressivity, the manner in which the mapping of gestural data onto the consequent events or sound parameters is done is just as important as the capture of the gesture itself. Nevertheless, in the case of the framework proposed, the mapping strategies are left to the decision of the final user. This is, the framework provides the control parameters, but the subsequent events and mapping strategies employed are dependent on the user choice.

Section 2.7 will provide a review of previous works developed in the area of gesture recognition, with particular emphasis for the performative arts.

2.7 Previous Works

The field of human movements and gesture analysis has, for a long time now, attracted the interest of many researchers, choreographers and dancers. Thus, since the end of the last century, a significant corpus of work has been conducted relating movement perception with music (Fraisse, 1982). The important role of the human body in complex processes such as action and perception, and the interaction of mind and physical environment has been acknowledged originating new concepts such as embodiment (the argument that the motor system influences our cognition, just as the mind influences body actions) and enactive (the human mind organizes itself through interaction with the environment) (Varela et al., 1993). Along with these relatively new concepts, many approaches have been proposed to translate the human physical movement and gesture into digital signals for further observation, study or plainly so that one can use them to control musical parameters in algorithmic music composition systems.

Already in the 90s, Axel Mulder (Mulder et al., 1994) characterized three techniques for tracking/capturing human movements, that still remains an important reference. Accordingly to him, the human movement tracking systems can be classified as inside-in, inside-out and outside-in systems.

Inside-in systems are defined as those which employ sensors and sources that are both on the body (e.g. a glove with piezo-resistive flex sensors). The sensors generally have small form-factors and are therefore especially suitable for tracking small body parts. Whilst these systems allow for capture of any body movement and allow for an unlimited workspace, they are also considered obtrusive and generally do not provide 3D world based information.

Inside-out systems employ sensors on the body that sense artificial external sources (e.g. a coil moving in a externally generated electromagnetic field), or natural external sources (e.g. a mechanical head tracker using a wall or ceiling as a reference or an accelerometer moving in the earth gravitational field). Although these systems provide 3D world-based information, their workspace and accuracy is generally limited due to use of the external source and their form factor restricts use to medium and larger sized bodyparts.

Outside-in systems employ an external sensor that senses artificial sources or markers on the body, e.g. an electro-optical system that tracks reflective markers, or natural sources on the body (e.g. a videocamera based system that tracks the pupil and cornea). These systems may suffer from occlusion, and a limited workspace, but they are considered the least obtrusive. Due to the occlusion it is hard or impossible to track small bodyparts unless the workspace is severely restricted (e.g. eye movement tracking systems). The optical or image based systems require sophisticated hardware and software and may be therefore expensive.

Following this least obtrusive Outside-In technique, several projects with the purpose of creating and controlling electronic music have been developed since the mid 1990s. Early works of composers Todd Winkler (Winkler, 1995b) and Richard Povall (Povall, 1998), or the choreographer Robert Weschler work with Palindrome[8]. Also, Mark Coniglio continued development of his Isadora programming

[8] http://www.palindrome.de

environment[9], plus the groundbreaking work Troika Ranch[10] has done in interactive dance, stand out as important references on how video analysis technologies have provided interesting ways of movement-music interaction.

Other example of research in this field is the seminal work of Camurri, with several studies published, including:

- an approach for the recognition of acted emotional states based on the analysis of body movement and gesture expressivity (Castellano et al., 2007). By using non-propositional movement qualities (e.g. amplitude, speed and fluidity of movement) to infer emotions, rather than trying to recognise different gesture shapes expressing specific emotions, they proposed a method for the analysis of emotional behaviour based on both direct classification of time series and a model that provides indicators describing the dynamics of expressive motion cues;
- the Multisensory Integrated Expressive Environments (Camurri et al., 2005), a framework for mixed reality applications in the performing arts such as interactive dance, music, or video installations, addressing the expressive aspects of nonverbal human communication;
- the research on the modelling of expressive gesture in multimodal interaction and on the development of multimodal interactive systems, explicitly taking into account the role of non-verbal expressive gesture in the communication process (Camurri et al., 2004). In this perspective, a particular focus is on dance and music as first-class conveyors of expressive and emotional content;
- the Eyesweb software (Camurri et al., 2000), one of the most remarkable and recognised works, used toward gestures and affect recognition in interactive dance and music systems.

[9] http://www.troikatronix.com/isadora.html

[10] http://www.troikaranch.org/

In 2005, Guedes (Guedes, 2005a) realized that analysing the number of pixels in video sequences whose luminance levels changed, due to repetitive movements, allowed to detect the periodicity in the video signal. Using the Goertzel Algorithm, a variation of the common spectral analysis algorithms (such as the Fast Fourier Transform), permitted the efficient computation of the fundamental frequency from the video signal, and subsequently estimates the rhythm of the physical movements on the video stream. Thus, Guedes developed the m-objects, a series of external objects for Max/MSP[11], for musical rhythm generation and musical tempo control from dance movement in real-time (Guedes, 2005b).

In 2008, Naveda (Naveda and Leman, 2008) proposed an approach for the representation of dance gestures in Samba dance. The representation was based on a video analysis of body movements, carried out from the viewpoint of the musical meter. His method provided the periods, a measure of energy and a visual representation of periodic movement in dance. He developed tools to relate the music and the dance on a metrical level, proposing relevant heuristic methods to connect music and dance.

Also Bevilacqua, at IRCAM-France worked on projects that used unfettered gestural motion for expressive musical purposes (Dobrian and Bevilacqua, 2003) (Bevilacqua et al., 2005) (Bevilacqua and Muller, 2005). The first involved the development of software to receive data from a Vicon motion capture system and to translate and map it into music controls and other media controls such as lighting (Dobrian and Bevilacqua, 2003). The second (Bevilacqua et al., 2005) consisted in the development of the toolbox “Mapping is not Music” (MnM) for Max/MSP, dedicated to mapping between gesture and sound. And the third (Bevilacqua and Muller, 2005) presents the work of the a gesture follower for performing arts, which indicates in real-time the time correspondences between an observed gesture sequence and a fixed reference gesture sequence.

Likewise, Nort and Wanderley (Nort et al., 2006) presented the LoM toolbox. This

[11] `http://cycling74.com/products/max/`

allowed artists and researchers access to tools for experimenting with different complex mappings that would be difficult to build from scratch (or from within Max/MSP) and which can be combined to create many different control possibilities. This includes rapid experimentation of mapping in the dual sense of choosing what parameters to associate between control and sound space as well as the mapping of entire regions of these spaces through interpolation.

Schacher (Schacher, 2010) searched answers for questions related to the perception and expression of gestures in contrast to pure motion-detection and analysis. Presented a discussion about a specific interactive dance project, in which two complementary sensing modes were integrated to obtain higher-level expressive gestures. Polloti (Polotti and Goina, 2011) studied both sound as a means for gesture representation and gesture as embodiment of sound and Bokowiec (Bokowiec, 2011) proposed a new term, “Kinaesonics”, to describe the coding of real-time one-to-one mapping of movement to sound and its expression in terms of hardware and software design.

Another important work, also published in 2011, is the one of Gillian (Gillian et al., 2011). He presented a machine learning toolbox that has been specifically developed for musician-computer interaction. His toolbox features a large number of machine learning algorithms that can be used in real-time to recognize static postures, perform regression and classify multivariate temporal gestures.

Also in 2009, the author made part of the project “Kinetic controller driven adaptive and dynamic music composition systems”[12]. One of the aims of the project was to utilize video cameras as gestural controllers for real-time music generation. The project included the development of new techniques and strategies for computer-assisted composition in the context of real-time user control with non-standard human interface devices. The research team designed and implemented real-time software that provided tools and resources for music, dance, theatre, installation artists, interactive kiosks, computer games, and internet/web information systems.

[12] `http://smc.inescporto.pt/kinetic/`

The accurate segmentation of the human body was an important issue for increased gestural control using video cameras. In the International Computer Music Conference (ICMC) of 2010 the author published a paper (Baltazar et al., 2010), presenting an algorithm for real-time human body skeletonization for Max/MSP. This external object for Max/MSP was developed to be used with the technology available at that time, a computer webcam capturing video in two dimensions. The algorithm was inspired by existing approaches and added some important improvements, such as means to acquire a better representation of the human skeleton in real-time. The output of the algorithm could be used to analyze in real-time the variation of the angles of the arms and legs of the skeleton, as well as the variation of the mass center position. This information could be used to enable humans to generate rhythms using different body parts for applications involving interactive music systems and automatic music generation. Nevertheless, the common CV problems of image segmentation using a two dimensional webcam, reduced the applications of the algorithm.

By the end of 2010 a new sensor was launched, with three dimensions video capture technology, that changed the way the human body could be tracked, the Microsoft Kinect camera (Zhang, 2012b), introduced in the previous Section 2.4. The Kinect impact has extended far beyond the gaming industry. Being a relatively cheap technology, many researchers and practitioners in computer science, electronic engineering, robotics, and even artists are leveraging the sensing technology to develop creative new ways to interact with machines. Being for health, security or just entertainment purposes. For instance, Yoo (Yoo et al., 2011) described the use of a Microsoft Kinect to directly map human joint movement information to MIDI.

Also, using a Kinect, already in the scope of this thesis, the author published a first version of the framework in ARTECH 2012 conference (Baltazar et al., 2012). The paper described a modular system that allows the capture and analysis of human movements in an unintrusive manner (using a custom application for video feature extraction and analysis developed using openFrameworks). The extracted gesture features are subsequently interpreted in a machine learning environment (provided by Wekinator (Fiebrink et al., 2009)) that continuously modifies several input parameters in a computer music algorithm (implemented in ChucK (Wang et al., 2003). The paper published at ARTECH was one of the steps for the framework presented in this thesis.

Despite all these relevant works made in this sub theme of the Human-Computer Interaction field, there are always new technologies emerging and new algorithms to apply to somehow improve and go further. This is the main objective of this thesis, to push through existing technology and contribute with a new framework to analyze gestures and use them to interact/manipulate/create events in a live performance setup.

2.8 Summary

This chapter was focused on gestures. It started by giving a definition and explanation about gestures and how they are defined in the psychological-linguistic domain, cognitive science domain and in the performing arts domain. Then, it described the different technologies available for capturing gestures.

After understanding gestures and learning the technological resources to capture them, one discussed how they can be recognized, the main applications of the recognition and the main approaches to do it successfully.

To finish the section, a review of the previous works done on the scope of this thesis was presented.

The corpus of research developed in this area in recent years has made available a new basis for the creation of computational models inspired on the human expression and perception of movement and gesture. These models have been used to test and refine existing theories, and to create interactive systems that are able to perform perceptually and artistically relevant tasks in real-time. Nevertheless, we are still looking for more satisfactory approaches and solutions to understand, interpret and creatively use human gestures in interactive art contexts. In this sense the framework proposed intends to fill some of the gaps still vaguely explored, such as the development and assembly of an entire system, from the capture of human movements till the training, recognition of gestures and consequent artistic event triggering, allowing thus a more straightforward utilization by the end users.

Chapter 3

Zatlab: A Framework for Gesture Recognition

"Reduce your plan to writing. The moment you complete this, you will have definitely given concrete form to the intangible desire."

Napoleon Hill

3.1 Introduction

As explained in Chapter 2, humans have excellent capabilities to learn, recognize and perceive gestures (Rowe and Goldin-meadow, 2009). Computers already present some qualities for the same task, however, many of the computational issues of recognizing gestures are still unsolved.

This chapter proposes an interactive gesture recognition framework called the Zatlab System (ZtS). This framework is flexible and extensible. Thus, it is in permanent evolution, keeping up with the different technologies and algorithms that emerge at a fast pace nowadays. The basis of the proposed approach is to partition a temporal stream of captured movement into perceptually motivated descriptive features. The analysis of the features will then reveal (or not) the presence of a gesture (similarly to the way a human "unconsciously" perceives a gesture (Kendon, 1980)).

The framework described in this chapter will take the view that perception primarily depends on the previous knowledge or learning. Just like humans do, the framework will have to learn gestures and their main features so that later it can identify them. It is however planned to be flexible enough to allow learning gestures on the fly.

In this particular case, while developing a framework to be used on a stage, by a dancer or performer, one wanted to allow as much freedom of movements as possible without being intrusive on the scene. The less the performer had to change is routine (by wearing sensors, markers or specific clothes) the better. That, together with the low cost of the technology (that allows the framework to reach to a broader number of performers), lead to the decision of using the optical MoCap option instead of others. The challenge of choosing this path resides on the development of sensor and CV solutions, and their respective computational algorithms.

Designed to be efficient, the resulting system can be used to recognize gestures in the complex environment of a performance, as well as in "real-world" situations, paving the way to applications that can benefit not only the performative arts domain, but also, helping the hearing impaired to communicate, in a near future.

First, one presents the architecture of the framework, along with a description of the different modules and the main algorithms involved in its development. Then, the various modes of functioning of the ZtS are discussed.

3.2 System Overview

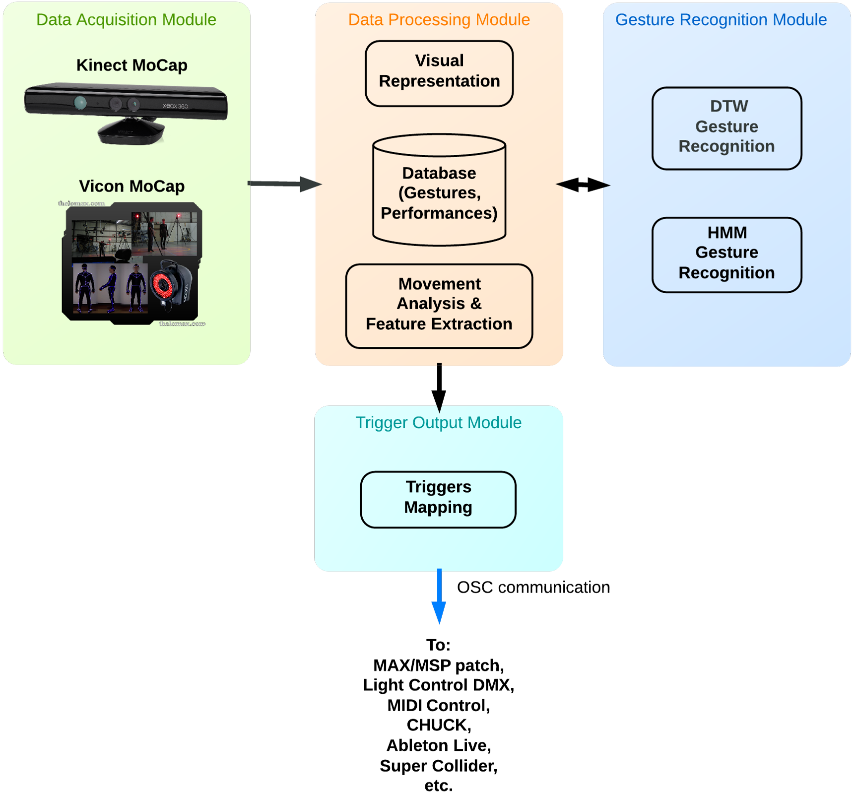


Figure 3.1: ZatLab system architecture diagram.

An overview of the proposed gesture recognition framework is presented in Figure 3.1. Summarized descriptions of the main blocks that constitute the proposed system are presented in this section. More detailed discussions about each of the processing stages will appear in the subsequent sections of this chapter.

The ZtS is a modular framework that allows the capture and analysis of human movements and the further recognition of gestures present in those movements. Thus, using the optical approach, the Data Acquisition Module will process data from a Microsoft Kinect or a Vicon Blade MoCap (presented in detail in Section 2.4.3). However it can be easily modified to have input from any type of data acquisition hardware.

The data acquired will go through the Data Processing Module. Here, it is processed in terms of movement analysis and feature extraction. This will allow providing a visual representation of the skeleton captured and its respective movements features. This module has also access to the database where it can record or load files. These can include: gestures, an entire captured performance, or features extracted from the movements.

Once the features are extracted, these are processed by the Gesture Recognition Module using two types of ML algorithms. The DTW and HMM (explained in the following sections). If a gesture is detected, it is passed to the Processing Module and this will store it, represent it or pass it to the Trigger Output Module.

In the Trigger Output Module the selected movement features or the detected gestures are mapped into triggers. These triggers can be continuous or discrete and can be sent to any program that supports the OSC communication protocol (Wright et al., 2001) (this protocol will be further explained in the following Section 3.6). In the next sections the different modules are presented in detail.

3.3 Data Acquisition Module

The human body tracking is one of the key elements of this system. The acquisition of human movements should be as accurate as possible, to ensure a proper analysis of their features and a correct gesture recognition. But the technology chosen must also be available and affordable to a broad range of performers. Also it should be the least intrusive possible. This arises some issues to solve and decisions to make. In a previous research, the author developed a similar module using a 2D webcam, whose output was then analyzed using image segmentation algorithms (as described in Section 2.7 (Baltazar et al., 2010)). Not being as accurate as one intended, another solution had to be taken.

More recently, with the Microsoft Kinect, it became possible to obtain a full-body detection using the depth information combined with the video signal. When compared with the previous webcam version, it can be said that it becomes simpler to detect and track a foreground object/person. The “traditional” CV tracking problems, such as light constraints or background/foreground separation can be solved using this new hardware.

Another advantage, that is very important in the scope of this framework, it is its portability. Not only it can be used in almost every environment imaginable (indoors, outdoors, good or bad light conditions, crowded places) but also, this sensor can be considered (almost) a Plug & Play technology. After some drivers and software installations, and computer teaks to make it work native, one just needs to plug it to the USB port and start working with it. To users/performers that are not keen to informatics, there is also the alternative to download applications that already have the drivers and software packages embedded, that will work instantly, such as the Synapse[1].

Altogether the Kinect provides a good solution for the framework: it is portable, reasonable cheap and has high performance tracking capabilities. In the Chapter 4, it will be possible to review the tests made with it and draw the conclusions of its suitability to this framework.

There is also a higher end method for detection, the Vicon MoCap system. With the advantages of remarkable tracking and low latency. It has, nevertheless, explicit disadvantages, such as: the cost, the rather complex and somewhat fixed setup for several infrared cameras and the necessity of wearing a special suit equipped with reflective markers.

Another disadvantage is that Vicon Blade only allows the real-time transmission of data to other commercially developed programs of their company or with companies that have established sharing protocols. Also, the transmission is made in a proprietary protocol. Consequently, in the case of this work, the real-time OSC

[1] `http://synapsekinect.tumblr.com`

transmission between the Vicon Blade and the ZtS (or any other external program) had to be developed (its implementation will be explained in Chapter 5).

This application, named ofxViconOSC, developed within the scope of this thesis, that can stream, in real-time the data from a Vicon system to any computer, is now available to the scientific community at the CITAR website[2].

Having these two technologies available at UCP, the framework developed should allow working with both. This way one can compare the Vicon MoCap against the Kinect in terms of the specific purpose of the gesture recognition latency (in Chapter 4, one can analyze the latency results).

In summary, this module consists on the acquisition of the real-world data to the virtual-world. It is independent of the hardware chosen to acquire the human movements, but is preset to work with a Microsoft Kinect and a Vicon Blade. In this module the hardware messages are decoded into human body joints to feed the Data Processing Module, presented next.

3.4 Data Processing Module

This module is the core of the framework, it will process and redirect the data to other modules keeping the framework functioning properly and effectively. This receives the skeleton joints data from the aforementioned Data Acquisition Module and processes it for three different purposes:

1. **Visual Representation**.
2. **Database Management**.
3. **Movement Analysis and Feature Extraction**.

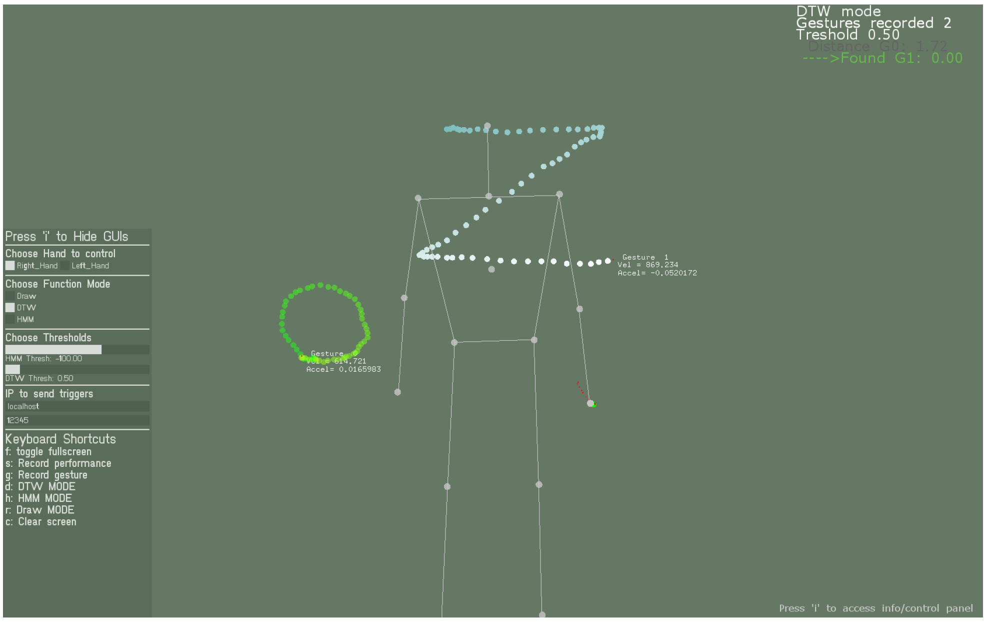
The following sections will discuss these three different purposes in detail.

[2] `http://artes.ucp.pt/citar/`

3.4.1 Visual Representation

The Graphical User Interface (GUI) provides a real-time, intricate but intuitive visual feedback to the user. Not only displays the skeleton of the user as if he was in front of a mirror (a virtual mirror in this case), but it can also display different panels of information. These range from the gestures previously recorded (with velocity and acceleration information attached), the gesture that was recognized, what triggers are setup and if a movement trigger was activated or not.

The Figures 3.3 and 3.2 present different views of the ZtS GUI.



DTW mode
Gestures recorded 2
Treshold 0.50
Distance G0: 1.72
---->Found G1: 0.00

Press 'i' to Hide GUIs

Choose Hand to control
☑ Right_Hand □ Left_Hand

Choose Function Mode
□ Draw
☑ DTW
□ HMM

Choose Thresholds
HMM Thresh: -100.00
DTW Thresh: 0.50

IP to send triggers
localhost
12345

Keyboard Shortcuts
f: toggle fullscreen
s: Record performance
g: Record gesture
d: DTW MODE
h: HMM MODE
r: Draw MODE
c: Clear screen

Gesture 1
Vel = 869.234
Accel= -0.0520172

Gesture 0
Vel = 614.721
Accel= 0.0165983

Figure 3.2: The GUI in development mode and the respective control panel. On the control panel one can see the DTW Mode is activated and the triggers are being sent to "localhost" and port 12345. Next to the gesture is presented its index and some statistics about it, in this case its average speed and acceleration. On the top right corner one can see the algorithm just recognized gesture "1".

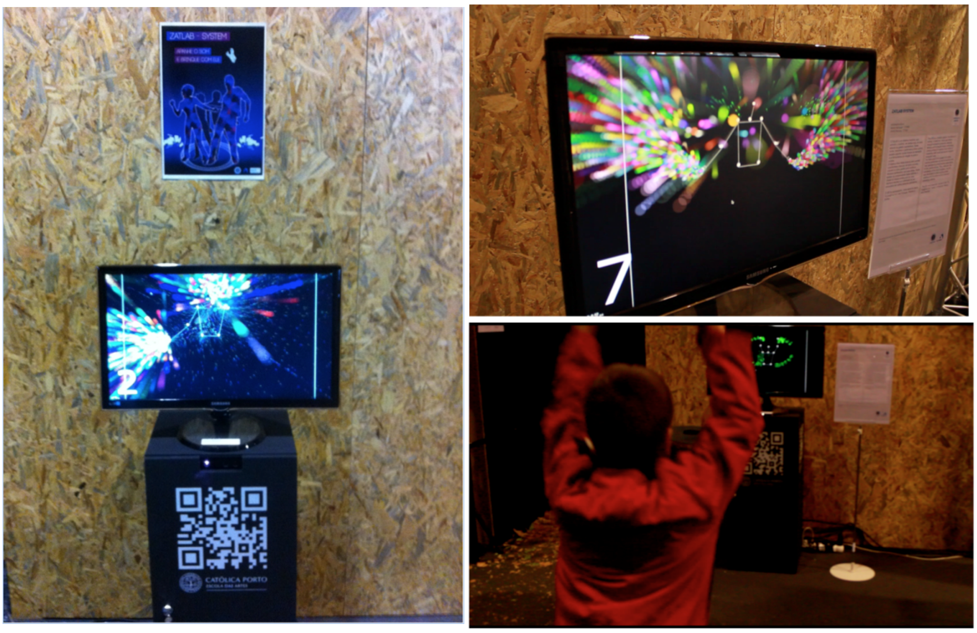


Figure 3.3: An application of the framework. The setup for FestivalIN (further described in Chapter 4) and a young boy interacting with the framework. The different color particles indicate triggers have been activated (in this case sound triggers).

3.4.2 Database

The database allows the user to record and load several types of files. It is organized in the following folders:

- **Performances** - The user can record an entire performance (e.g. a dance, a presentation, etc). It records the several skeleton joints data sequence in a text file. It allows to reproduce exactly what was done by the user, thus enabling the review, setup and adjustment of triggers in offline mode (for instance, can be used to record a dance rehearsal, review it and setup some gesture triggers to use on the next rehearsal or in the presentation of the dance performance).

- **Gestures** - The user can record a set of gestures for training the recognition algorithms or for gesture notation purposes. Different from the performance recording, hence will be recording only the segment of data that represent the gesture and its main features (for instance, the circles presented in Figure 3.3 can be recorded in the database for future use).

- **Gesture Models** - When the user trains the gesture recognition algorithms, he is creating a gesture model. This model contains the features of the recognition algorithms, necessary for the recognition of a similar gesture. This folder stores the model files.

- **Drawings** - The user can use the framework in a more lateral purpose for free drawing (like for instance, a virtual board). In this folder the user can store the drawings.

The files are stored with a single identifier name consisting on the data and time of the start of recording.

3.4.3 Movement Analysis and Feature Extraction

Having in mind the results of previous researches, presented in Chapter 2 - Section 2.5.2, the features chosen to compute are the ones provided by the Physics kinematic equations[3] to describe movement along with the orientation angle of the gesture path, described next.

From the data acquired one already has the information of the coordinates and the respective timestamp t for each joint of the human body. Therefore, the following features can be computed.

[3] `http://www.physicsclassroom.com/class/1dkin/u1l6a.cfm`

Time

For a given movement segment, its total time can be easily computed by subtracting the first sample time-stamp t_1 from the last sample time-stamp t_n:

$$T = t_n - t_1 \tag{3.1}$$

Displacement

Knowing all the coordinates of the movement segment, the total displacement D can be calculated by summing the relative difference among $coord_i$ and the previous $coord_{i-1}$, from the first sample ($i = 1$) till the last (n).

$$D = \sum_{i=1}^{n} \|coord_i - coord_{i-1}\| \tag{3.2}$$

Velocity

Also, the velocity and acceleration can be computed. The average velocity will be defined as the quotient of the displacement Δd and the interval time Δt. In the case of consecutive frames (where the Δt is very small) we can assume this is the instantaneous velocity v_i.

$$v_i = \frac{coord_i - coord_{i-1}}{t_i - t_{i-1}} \tag{3.3}$$

And the average velocity can be computed as the sum off all the v_i divided by the number of samples n:

$$v_{avg} = \frac{\sum_{i=1}^{n} \|v_i\|}{n} \tag{3.4}$$

Acceleration

Similarly, the instantaneous acceleration can be approximated by the average acceleration over a small interval Δt.

$$a_i = \frac{v_i - v_{i-1}}{t_i - t_{i-1}} \tag{3.5}$$

And the average acceleration can be computed as:

$$a_{avg} = \frac{\sum_{i=1}^{n} \|a_i\|}{n} \tag{3.6}$$

All the features are extracted within a motion segment. These features are very important to describe the joint movements. Although with these features one is already able to visualize and extract relevant information from the data, the direction of movement the joint takes at each frame is also a key feature for the ML algorithms (explained in the next section). This feature will allow not only to detect immediately if the movement is done from left to right, but also if it is a simple line or something more complex like a circle or a square.

Direction of Movement

The angle or direction of movement can be calculated using the known coordinates at consecutive frames and applying the arc-tangent function. This is given by Equation (3.7) and the result is given in degrees (in this case computed only in two dimensions: Δx is the displacement along the X axis and ΔY is the displacement along the Y axis).

$$\theta = atan\frac{\Delta Y}{\Delta X} \tag{3.7}$$

θ ranges from 0° till 360°. This would create a tremendous range of data to be analyzed, in real-time, by the ML algorithms (Al-Hamadi et al., 2010). Also, measuring the direction of the movement in single unit degrees could lead to additional noise in the data. Therefore, it is necessary to normalize the data to an observable "codeword". This can be done by dividing the total range of the angles in 12 equally separated spaces (12 spaces allow to understand differences in increments of 30°). So, the direction of movements is classified accordingly to the degrees belonging to a determined interval. The framework is setup to work with these 12 symbols, but it can be easily adapted to work with more or less (however these achieved good results, as presented in Chapter 4). See Table 3.4.3 and Figure 3.4 for better understanding this angle based "codeword".

Table 3.1: Angles codeword table

Angle	**Codeword Value**	**Angle**	**Codeword Value**
[0°,30°]	0	**[181°,210°]**	6
[31°,60°]	1	**[211°, 240°]**	7
[61°,90°]	2	**[241°, 270°]**	8
[91°,120°]	3	**[271°, 300°]**	9
[121°,150°]	4	**[301°, 330°]**	10
[151°,180°]	5	**[331°,359°]**	11

3.5 Gesture Recognition Module

The gesture recognition in HCI has many similarities with other areas of research. Being encompassed in a more general area of pattern recognition, stand out, in particular, the similarities with speech or handwriting recognition. Being these areas already more developed in scientific terms, it is natural to try to mirror the various techniques applied in these areas to gesture recognition (Corradini, 2001a).

Considering a gesture G can be described as a sequence of feature vectors, it can be assumed that the best way to describe it is to gather N sequences (prototypes)

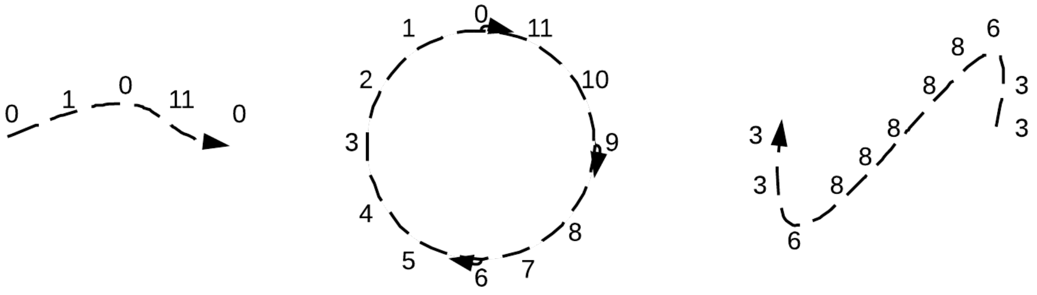


Figure 3.4: Examples of gestures recorded and their associated angle orientation codeword. In the case of the circle all the orientation values are present, but the timestamped sequence will reveal if it was executed in clockwise or counter-clockwise motion.

of that gesture (performed in different ways). Therefore, when in recognition mode, an unknown input can be compared against each one of these N prototypes and, taking into account the measures and criteria chosen, a degree of similarity can be assigned.

Although it has a high computational cost, a large set of reference patterns N should be used for this comparison, representing each gesture G. The biggest problem with this approach is the choice of a suitable distance measure. The simplest way to define it is by calculating the distances between the corresponding samples of the reference and the unknown input sequences and accumulate the result. Unfortunately, gestures have a variable spatio-temporal structure. They vary when performed by different people and even the same user is not able to perform a gesture exactly the same way several times in a row. This means that, depending on both the speed of the movement performance and the user, the recorded gesture signals can be stretched or compressed.

Therefore, to compare two signals permitting them to have different lengths requires dynamic programming. Learning from speech recognition, since speech shares the varying temporal structure of gestures, an algorithm often used in that field is the DTW (Rabiner and Juang, 1993). The DTW algorithm, performs a time alignment and normalization by computing a temporal transformation allowing two signals of different lengths to be matched.

Another alternative of dynamic programming is the statistical and probabilistic approach, such as Hidden Markov Model (HMM). It is a rich tool used for gesture recognition in diverse application domains. Probably, the first publication addressing the problem of hand gesture recognition is the seminal paper by Yamato et al (Yamato et al., 1992). In his approach, a discrete HMM and a sequence of vector-quantized (VQ)-labels have been used to recognize six different types of tennis strokes.

In this section, one will discuss the principles and background of both the algorithms working on the Gesture Recognition Module, the DTW and the HMM.

In this particular instance, the module is being trained and fed with the human movement features (sequence of coordinates, velocities and orientation of the movement). But it can be used also for recognition of any other signals sequence (writing, speech, image), the user just needs to use the right features to feed the module at each case.

3.5.1 The Dynamic Time Warping

When two signals with temporal variance must be compared (e.g. a reference time sequence of features that represent a gesture against an unknown time sequence of features), or when looking for a pattern in a data stream, the signals may be stretched or shrunk along its time axis in order to fit into each other. A comparison made after these operations can give false results because we may be comparing different relative parts of the signals. The DTW is one of the methods to solve this problem (Ten Holt et al., 2007). The algorithm calculates the distances between each possible pair of the two signals taking into account their associated feature values. With these measured distances it builds a matrix of accumulated distances and finds the path that guarantees the minimum distance between the reference and tested signals. This path represents the best synchronization of both signals and thus, the minimum feature distance between their synchronized points.

Consequently, the DTW has become popular by being extremely efficient as the time-series similarity measure which minimizes the effects of shifting and distortion in time, allowing "elastic" transformation of time series in order to detect similar shapes with different phases. It has been used in various fields, such as speech recognition (Rabiner and Juang, 1993), data mining (Keogh and Ratanamahatana, 2005), and movement recognition (Corradini, 2001b; Gillian et al., 2011).

Formally explaining, given two time series $X = (x_1, x_2, ...x_N), N \in \mathbb{N}$ and $Y = (y_1, y_2, ...y_M), M \in \mathbb{N}$ represented by the sequences of values, the DTW will permit the synchronization of the two signals, see Figure 3.5 .

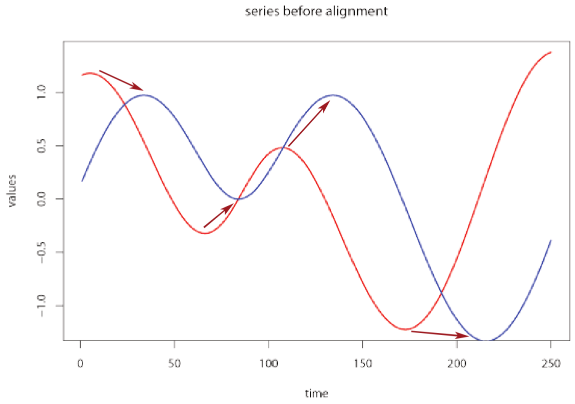


Figure 3.5: Time alignment of two time-dependent sequences. Aligned points are indicated by the arrows, adapted from (Senin, 2008a).

If sequences are taking values from some feature space ϕ then in order to compare two different sequences $X, Y \in \phi$ one needs to use a local distance measure d (usually the Euclidean distance). Intuitively d has a large value when sequences are different and a small value if they are similar. This distance function is usually called the "cost function" and the task of optimal alignment of the sequence becoming the task of arranging all sequence points by minimizing the cost function (or distance). The algorithm starts by building the local cost matrix $C \in \mathbb{R}^{N \times M}$ representing all pairwise distances between X and Y as it is described by Equation 3.8:

$$C \in \mathbb{R}^{N \times M} : c_{ij} = \|x_i - y_j\|, i \in [1 : N], j \in [1 : M] \tag{3.8}$$

Once the local cost matrix is computed, the algorithm finds the alignment path that runs through the low cost areas of the matrix. This alignment path (or warping path) defines the correspondence of an element $x_i \in X$ to $y_j \in Y$ following the boundary condition which assign first and last elements of X and Y to each other (see Figure 3.6).

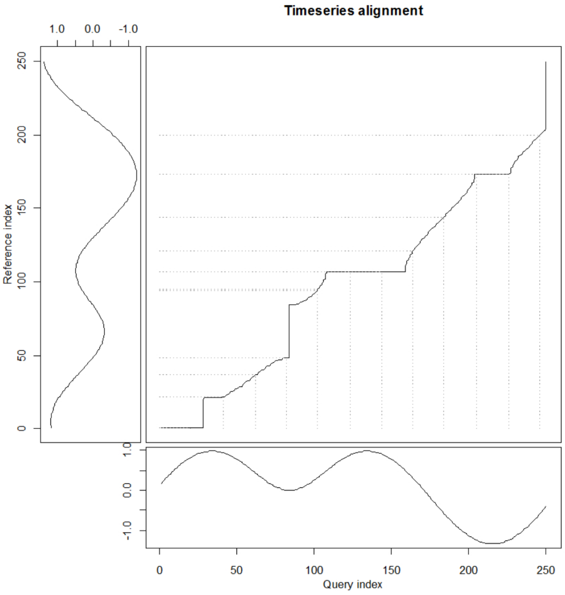


Figure 3.6: The optimal warping path for the signals alignment. The vertical axis presents the reference signal and the horizontal axis presents the query input, adapted from (Senin, 2008b).

This accumulated cost matrix results to be the minimum distance possible among the reference gesture and the unknown input. Having thus, an overall measure to state if the signals are similar or not. In this case, if the gesture being tested has an accumulated cost matrix value low enough (in regard to the original recorded gesture), then it is safe to say we are in the presence of the same gesture.

When testing the unknown input against different gestures one can end up with the respective cost matrix values similar among themselves. This can create doubt in which is the correct gesture. In this case there are two options: to consider only the nearest-neighbor (the one that presents the lower cost value) as the correct one or to consider evaluating K-nearest-neighbors to achieve weighted results (e.g. consider the 5 nearest-neighbors and choose the gesture more represented in that group as the correct one).

A more detailed explanation on the DTW and how the cost matrix is computed is presented in *Annex A*.

The DTW works good enough when the gesture is simple, due to simplicity of the training (further analysis is explained in Chapter 4). But if the gesture is somehow more complex, it needs to be trained with more samples and in a more statistical and probabilistic driven algorithm. This lead to the implementation of the HMM, explained next.

3.5.2 The Hidden Markov Model

HMM (Rabiner, 1989), (Yamato et al., 1992), are powerful statistical models for representing sequential or time-series data, and have been successfully used in many tasks such as speech recognition, protein/DNA sequence analysis, robot control, and information extraction from text data. HMM have also been applied to hand and face recognition (Nefian and Hayes III, 1998). The HMM is rich in mathematical structures and has been found to efficiently model spatio-temporal information in a natural way. The model is termed “hidden” because all that can be seen is only a sequence of observations (symbols). It also involves elegant and efficient algorithms, such as Baum-Welch and Viterbi (Viterbi, 1967), for evaluation, learning and decoding.

Formally, an HMM is defined as a quintuple (S, V, Π, A, B) (Rabiner, 1989) where $S = \{s_1, ..., s_N\}$ is a finite set of N hidden states (that model a gesture); $V = \{v_1, ..., v_M\}$ is a set of M possible symbols (e.g. features of the gesture) in a vocabulary; $\Pi = \{\pi_i\}$ are the initial state probabilities; $A = \{a_{ij}\}$ are the state transition probabilities; $B = \{b_i(v_k)\}$ are the output or emission probabilities.

Therefore, each HMM is modeled and expressed as $\lambda = (\Pi, A, B)$ where the parameters are:

- π_i - the probability that the system starts at state i at the beginning;
- a_{ij} - the probability of going from state i to state j;
- $b_i(v_k)$ - the probability of generating symbol v_k at state i.

The generalized topology of an HMM is a fully connected structure, know as an $ergodic$ model, where any state can be reached from any other state. When employed in dynamic gesture recognition, the state index transits only from left to right with time (left to right HMM). The global structure of the HMM recognition is constructed by training of each HMM $(\lambda_1, \lambda_2, ..., \lambda_M)$, whereby insertion (or deletion) of a new (or existing) HMM is easily accomplished. λ corresponds to a constructed HMM model for each gesture and M is the total number of gestures being recognized.

When working with HMM there are three basic problems to solve:

1. **Evaluation**: Given a model and a sequence of observations, how do we compute the probability that the observed sequence was produced by the model? Namely, one has to evaluate the probability of an observed sequence of symbols $O = o1, o2, ..., ot$ (where $o_i \, \epsilon \, V$) given a particular HMM (λ), e.g.

$p(O|\lambda)$. This is extremely useful, in this case having several competing "models" of gestures, this will allow to find which gesture "model" best matches the observations (of the gesture being performed live).

2. **Decoding**: This is to uncover the hidden part of the model, i.e. to find the state sequence that illustrates best the model. In other words, to find the most likely state transition path associated with an observed sequence. Having a sequence of states $q = q1, q2, ..., qt$ we will want to find the $q* = argmax_q p(q \wedge O|\lambda)$.

3. **Training**: Is the crucial part of HMM, since it will allow to adapt the model parameters to the observed training sequence, hence creating the best models for the gestures performed. In other words, is to adjust all the parameters of our model λ to maximize the probability of generating an observed set of sequences O, this is, to find the $\lambda* = argmax_\lambda p(O|\lambda)$.

These three problems already have solutions. The first is solved by implementing part of the Forward-Backward iterative algorithm. The second by using the Viterbi algorithm, and the third by using the Baum-Welch algorithm, which uses the the Forward and Backward probabilities calculated previously to update the parameters iteratively. Next, one will explain briefly these three algorithms. A more detailed description of these algorithms is presented in *Annex A*.

Forward-Backward Algorithm

The forward–backward algorithm can be used to find the most likely state for any point in time. In HMM, each symbol emission and each state transition depend only on the current state. There is no memory of what happened before, no lingering effects of the past. This means that having a sequence of observation symbols $O(1...T)$, it can be broken into two parts, a "past" sequence $O(1...t)$ and a "future" sequence $O(t+1...T)$. Thus, one can work on each half separately. The motive for splitting the sequence into two parts is to use induction on t. The inductive calculation where t advances from 1 towards T is called the forward calculation, while the calculation where t is decremented down from T towards 1 is called the backward calculation.

The Forward probabilities will allow solving the problem 1 and finding the probability of a sequence of observations to belong to a determined HMM model. The Backward probabilities will allow solving problem 3 along with the Baum-Welch algorithm, explained ahead.

Viterbi Algorithm

Although being, nowadays, one of the most used algorithms in the field of Pattern Recognition and ML (Jr, 2005), the Viterbi algorithm was created and published by Andrew Viterbi in 1967 (Viterbi, 1967) as simply an explaining support tool for his Information Theory classes. Curiously, at that time he had no idea that the algorithm was actually an optimum (maximum likelihood) decoder, nor that it was potentially practical.

The Viterbi algorithm allows computing the most likely state transition path given an observed sequence of symbols. It is similar to the Forward algorithm, but in this case, instead of doing the sum over all the possible ways to arrive at the current state being considered, it will keep only the path segments with maximum likelihood. Thus, having a sequence of states $q = q1, q2, ..., q_T$ we want to find the $q^* = argmax_q p(q \wedge O|\lambda)$.

The algorithm will return the Viterbi Probability - VP, i.e. the best score (highest probability) along a single path, at time t, which accounts for the first t observations and ends in state S_i.

Baum-Welch Algorithm

The Baum-Welch algorithm allows to solve the fundamental problem of an HMM. This is, to adjust the model (λ) parameters in order to maximize the probability of the observation sequence. This is again a maximum likelihood problem. Actually there is no optimal analytically method of estimating the model parameters, given any finite observation sequence as training data. Neither there is a known way to analytically solve for the model that maximizes the probability of the observation sequence. It is possible, however, to use an iterative procedure (such as Baum-Welch method) to choose $\lambda = (A, B, \pi)$ such that $P(O|\lambda)$ is locally maximized.

The training procedure is done by iteratively computing the expected probability of all possible hidden state transition paths, and then re-estimates all the parameters based on the expected counts of the corresponding events. The process is repeated until the likelihood converges.

The training of the model can go through a fixed number of iterations or, as in the case of this framework, it stops if there is no substantial difference between consecutive iterations values, what is assumed to be the best model possible to the observed sequence.

The results of the HMM algorithm recognition rate are presented in Chapter 4.

The Gesture Recognition Module is of paramount importance for this framework. The recognition algorithms (DTW and HMM) can be used in simultaneous or individually, providing different modes of training and recognition (presented in Chapter 4). Their computer implementation will be presented in Chapter 5.

When a gesture is recognized, this is communicated to the Processing Module that will redirect the information to the Triggers Output Module. Next is the description of this module.

3.6 Triggers Output Module

Paraphrasing Newton third law of movement, "*For every action, there is an equal and opposite reaction*". This module is responsible for the reaction. It may not be opposing neither equal, but it is definitely a reaction, in this case to a gesture performed.

This module has the setup of what will be the framework reaction to a gesture recognized. This can be internal or external. Internally it can react by generating visual contents on the GUI such as images, information or drawings. And externally it can control anything that directly assumes OSC communication protocol, what nowadays is pretty common.

OSC (Wright et al., 2001) was originally developed to facilitate the distribution of control structure computations to small arrays of loosely coupled heterogeneous computer systems. A common application of OSC is to communicate control structure computations from one client machine to an array of synthesis servers. OSC is a 'transport-independent' network protocol (Wright, 2005), meaning that OSC data can be carried by any general-purpose network technology. Today most implementations use the main Internet protocols (UDP and TCP/IP) via Ethernet or wireless network connections.

Part of what makes OSC to be used so often is that it comes with no standard set of messages, no preconceptions of what parameters should be available or how they should be organized. Each implementer of OSC can and must decide which parameters to make accessible, what to name them, and how to organize them in a tree structure. This form of openness has led to great creativity among OSC implementations, supporting idiosyncratic, creative software and hardware. Thus, most of the programs used in the performative arts domain (and other domains) allow communication through OSC, these range from sound and music control programs, video or light setup and display tables, till computers and robotic hardware.

Therefore, is possible to control a vast amount of events with a gesture. One just have to decide on the trigger mapping and respective OSC syntax.

For each gesture trained in the framework a trigger is assigned. It can be discrete (triggering only events each time gesture is recognized) or continuous (controlling events such as sound pitch or modulation accordingly to a velocity or coordinate value). The triggers can be further customized by the user, but are preset to work in the following fashion:

1. **Discrete triggering** - Each gesture trained for recognition is associated with a single identifier trigger, matching the gesture index (e.g. Gesture 1, Gesture 2, etc.). When a gesture is recognized a trigger message is sent through OSC, using the following syntax:

   $\backslash Gesture\ index\ joint\ coord_X\ coord_Y\ coord_Z\ Avg.Velocity\ Avg.Acceleration$

2. **Continuous triggering** - The default configuration for continuous triggering consists on maintaining a constant communication of the joints kinematic features. For instance, the left hand OSC message will be:

   $\backslash HandL\ coord_X\ coord_Y\ coord_Z\ Inst.Velocity\ Inst.Acceleration$

In order to create an interesting result one needs to map the triggers to the respective events. As reviewed in Chapter 2, there are several strategies to do the mapping of the triggers to expressive events. The choice of which to apply is done by the users of the framework. This is, the framework allows the association of triggers to gestures, therefore when the gesture is performed and recognized the trigger is sent. What the user does with that trigger is depends on his creativity or purpose. For instance, on the applications described on the following Chapter 4, the triggers were mapped internally to the emission of visual particles and externally to the control of sound events.

3.7 Framework Operation Modes

During the course of developing the framework, it was tested in different scenarios. It was used in public spaces for people interaction, in laboratory context to test the recognition algorithms and in the performance context in an internationally commissioned Opera (these applications will be further explained in the Chapter 4). Consequently, the framework was customized to work in different setups:

- **Drawing mode** - this mode operates manly for public installations or class teaching. No triggers are associated, the user can draw freely in a canvas, as if working with a virtual board.
- **Simple movement trigger** - the user can set and activate triggers based on body joints position, velocities and accelerations.
- **Gesture mode** - the user can record gestures, choose the algorithm to use for recognition (DTW or HMM) and use the recognized gestures as triggers.
- **Gesture and Movement mode** - probably the most interesting of the setups. A mixture of the previous modes, where not only the gesture is important, but also the speed and where it is performed are relevant.

The main concern when dividing the framework in these main operation modes was to allow the end user an easy customization for his own purpose or application. These modes can be chosen using the control panel of the framework.

3.8 Summary

This chapter presents the ZtS framework as a conceptual system to recognize gestures in real-time, using algorithms and specifications already tested in other research domains. These specifications were selected based on the analysis presented in Chapters 2 on the current standards and systems in the gesture recognition realm. The framework presents an abstraction of a modular system that should base the implementation of the gesture recognition tool for the complex domain of performative arts. This framework is developed to be a highly modular system, where any module can be reused for a different purpose or in a different scenario.

Actually, the framework presented was abstracted from the concrete application presented in the following chapters. It is expected that this abstract framework may be applied to other domains, unrelated to the domain that motivated this research. In a strict sense the ZtS cannot be called as a framework since it was only ”applied” to a single instance. It will be part of the future work, resulting from this dissertation to apply the ZtS to other domains and validate it as a framework.

The implementation of this framework for the computer-programming domain is presented in Chapter 5. In the following Chapter 4 will be conducted an objective and comprehensive evaluation and validation of the ZatLab System.

Chapter 4

Experimental Validation and Applications

"The value of an idea lies in the using of it."

Thomas A. Edison

4.1 Introduction

This chapter describes the overall research approach used throughout this study and presents, in finer detail, the evaluations and artistic applications of the framework proposed in the previous chapter.

The research methodology used will be explained, followed by the brief description of Nielsen evaluation model for acceptability. Knowing the methods and items proposed for evaluation, the experiment design is laid out and also both the qualitative and quantitative evaluation results of the framework are exposed.

The chapter finishes with the artistic applications of the framework. Namely its use in an Interactive Opera, in collaboration with Miso Music Portugal, and the use of the framework as a public interactive installation in the Festival of Creativity and Innovation, in Lisbon, 2013.

4.2 Research Methodology

To perform a validation, first one needs to establish the research methodology. This is generally defined as a set of procedures one has to systematically complete in order to find the solution for a scientific problem. These are standardized techniques known and shared by the scientific community, ensuring the reproducibility of results when all the original conditions are replicated. Furthermore, they allow for the identification and quantification of the systematic errors existing in each procedure (for example, when measuring something). A consolidated methodology and a stable terminology are necessary conditions for the production of cumulative knowledge.

The methodology should be compliant with the phenomenon being studied and the scientific field where this is inserted. In this case, the framework proposed is in the scientific field of HCI but also has points in common other research areas, such as Cognitive Sciences, or Psychological-Linguistic Sciences and Performating Arts, amongst others. Therefore, the choice of research method is not trivial, this should comply with the various practices of these research fields, but also merge and adapt to this particular project demands.

HCI is still a young scientific discipline; established methodologies started during the 70s in the Silicone Valley but new assessment methods are still being introduced. HCI has emerged from the combination of core computer science with cognitive psychology, which *“encompasses many sub disciplines with different research questions and methods”* (Boring, 2002), such as sociology, anthropology or communication. HCI is, therefore, related to human factors. In this field, most of the research methods rely on empirical studies, since the phenomena assessed are always somehow related with the use of technology.

From the Social Sciences (e.g. Cognitive, Psychological-Linguistic) there are three main methods for research: experiments, surveys and ethnography.

- An experiment is usually a laboratorial test where most of the variables are kept under control. This is able to provide great internal validity since all the systematic errors are taken into consideration while designing the study. However, the possibility of inferring universal laws are constrained to the laboratorial settings, and real-world applications are sometimes hindered by the lack of real and more complex settings during the study. The outcome of an experiment is usually based on quantitative data.
- Surveys are one of the most common tools used in social sciences research. They can range from closed questionnaires to open interviews. In the former, respondents are asked to answer a fixed number of questions, each one with a fixed number of answers; in the latter, interviews may have a structural script, which adapts to the answers being given. Closed questionnaires produce numeric quantitative data and are more easily compared with each other, while interviews require a linguistic analysis and tend to produce more qualitative data. Nowadays, using web technologies, surveys can have an instantly wide spread across a population at a very low cost.
- Ethnography is the scientific description of the traditions and habits of a population. It is, therefore, a methodology based on the observation of individuals in their natural habitat. The scientist has no control over independent variables, thus it is difficult to assess a case-effect dynamic. On the other hand, these studies are more prone to having external validity, since the experimental setting corresponds to a real-world scenario.

Taking all this into account, the approach chosen puts an emphasis on HCI and combines several methodologies in an interdisciplinary fashion, where borders of distinct knowledge are defied, aiming at the integration of different methods into a science of its own. Consequently, the following evaluation studies were conducted:

1. a qualitative measurement of the acceptability of the framework amongst the users, using the Nielsen Acceptability model (Nielsen, 1993) (using surveys);

2. a quantitative laboratorial experiment for the analysis on several aspects of the framework performance;

3. two, real-world context, artistic experiments, being one of them a professional application in a performance, that allowed also a qualitative evaluation done by a professional performer.

The next sections describe these studies, starting by a brief explanation of the Nielsen model to measure acceptability.

4.3 Evaluation Model

Hence the framework presented was developed with the main purpose of being used broadly by different people one must evaluate its acceptance.

According to Nielsen (Nielsen, 1993) the acceptability of a system is defined as the combination of social and practical acceptability. The former determines the success/failure of the system, since the more the system is socially acceptable the greater is the number of people using it. The latter relates factors such as usefulness, cost, reliability and interoperability with existing systems. An adaptation to the Nielsen model is depicted in Figure 4.1.

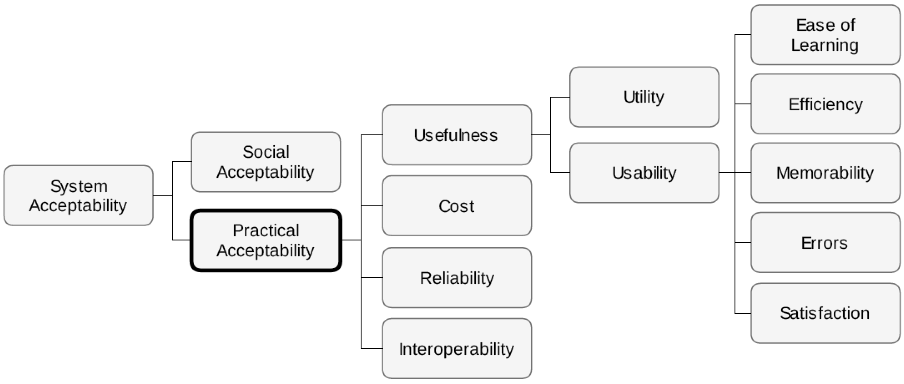


Figure 4.1: System acceptability, adapted from (Nielsen, 1993).

The **usefulness** factor relates the **utility** and **usability** offered by the system. **Utility** is the capacity of the system to achieve a desired goal. As the system performs more tasks, the more utility it has. **Usability** is defined by Nielsen as a qualitative attribute that estimates how easy is to use an user interface. He mentions five characteristics involved in the concept of usability:

1. **ease of learning** - the system should be easy to learn so that the user can start doing some work with the system;

2. **efficiency** - the system should be efficient to use, so after the user learns the system, a high level of productivity is possible;

3. **memorability** - the system should be easy to remember so that the casual user is able to return to the system after a period without using it, without requiring to learn it all over again;

4. **errors** - the system should prevent the user from committing errors as should deal with them gracefully and minimizing the chance of occurring catastrophic errors;

5. **satisfaction** - the system should be pleasant to use so that users become subjectively satisfied when using.

The **reliability** is the ability of a system or component to perform its required functions under stated conditions. **Interoperability** is the ability of two or more systems or components to exchange information and to use the information that has been exchanged.

In the experiments explained next the overall system acceptability was evaluated. In regard to the practical acceptability, the **usefulness** evaluation was done applying users questionnaires. The **reliability** evaluation is provided by a quantitative analysis and the **interoperability** is shown in artistic performance applications of the framework. The **cost** factor was not considered since ZtS is a free (and open source) software. The social acceptability was also done recurring to users questionnaires.

4.3.1 Experiment Design

The experiment took place at UCP, MoCap Laboratory. This allowed to use two MoCap technologies to capture gestures, the Vicon system and the Kinect. In terms of setup, the framework was working in a Macintosh Apple Computer (MacBook Pro model with processor Intel Core i7 at 2,7 GHz QuadCore, 16GB of RAM and a NVIDIA GeForce GT 650M 1024 MB graphic board), the GUI of the system was being displayed in a large (48 inches) LCD screen (see Figure 4.2).

Overall there were 29 participants. These ranged from musicians, artistic performers, scientists and even sportsmen of different age, gender and education background. Further details are provided in the following sections of results.

The experiment itself consisted in providing, to each participant independently, an explanation (15min duration) of the framework main functions. There was a script of the briefing to guaranty the consistency through out all the participants. First, to explain the possibilities of the framework, one presented the fully working application of the system that was installed at FestivalIn (described in section 4.6.2) and let the participants interact with it. Second, the participants were taught to work with the framework in order to achieve a result similar to the application presented. This included teaching them:

- how to capture a gesture for future recognition;
- how to record an entire performance for working offline with it;
- how to load a previously recorded performance or gesture;
- how to train both the recognition algorithms;
- how to setup the recognition algorithms thresholds;
- how to setup the Internet Protocol (IP) address and Port to where the recognized gesture triggers should be sent.

Next, the participants were asked to interact again with the system. Figure 4.2 depicts the system setup.

They were placed in front of a LCD screen displaying the framework GUI and were asked to use the free drawing mode first. This allowed them to understand the basics of the framework, more specifically, to get a grasp of their real-time human body representation and learn how to "draw" their gestures. After this, they were asked to record various predetermined gestures, granting thus, the gathering of a good amount of gesture data for the quantitative evaluation explained ahead.

The interaction with the system comprised the use of a wireless mouse to trigger the capture of the gestures. While keeping the left mouse key pressed, the system was capturing the gesture made. To delete the gestures the user just had to press the right mouse key (to clarify, the use of the mouse is just for capturing the gestures to train the algorithms, in performance context there is no need to use it). The choice of the mouse standout from other approaches tested, mainly for two reasons. First, by its practicability, since it is a common object and familiar

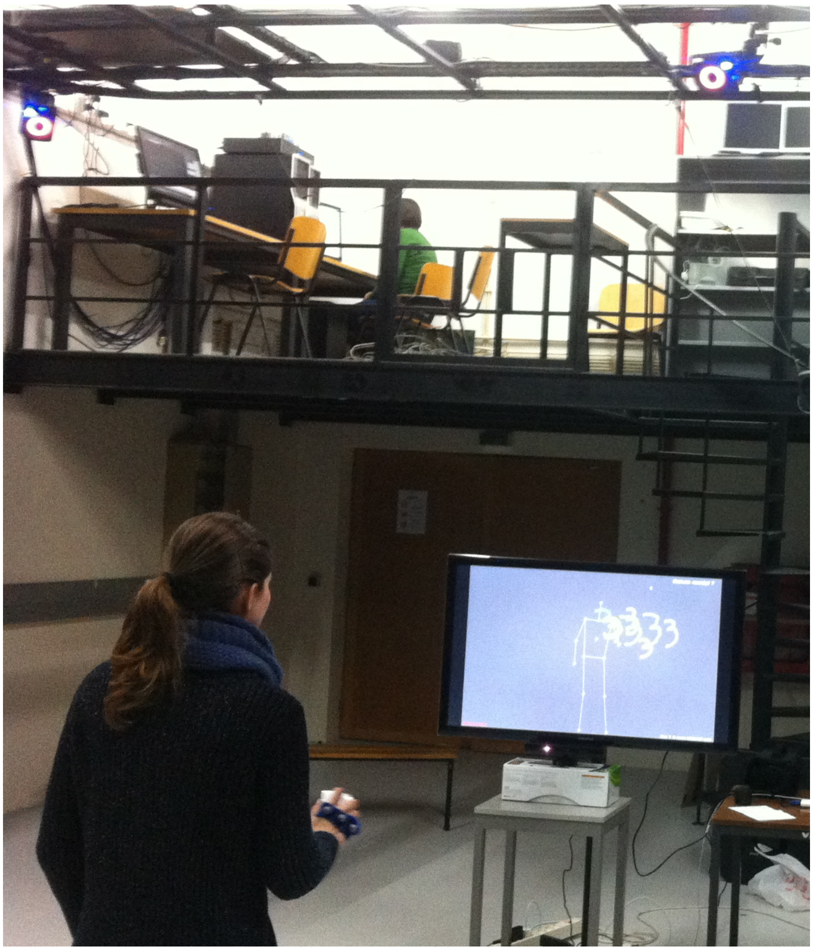


Figure 4.2: A participant of the experiment. One is able to see the Microsoft Kinect just bellow the LCD screen and two of the 10-camera array of the Vicon system (up corners of the photo). Looking closely you may notice, in her right hand, the 3 markers used for the capture with the Vicon and also the wireless mouse to trigger the gesture capture. In this case the participant was asked to draw the Gesture “3” several times.

to computer users. Second, for its accuracy, since the use of its buttons allows to identify precisely where the gesture begins and where it ends. The other approaches tested consisted on having a capture trigger either by making a specific pose or either by placing one hand in a virtual button. However, these methods revealed to be confusing and did not allow to specify accurately the start and end of the gesture captured.

In the end the participants filled out a survey, in order to analyze (qualitatively) the system acceptability. The surveys were fulfilled and collected on-line using Google Forms and include questions on the ZtS usability (the entire questionnaire is presented in Appendix A).

The participants also contributed for the quantitative evaluation, by recording (each), ten samples of five gestures of different complexity. This data was recorded by the framework in order to process gesture training and evaluate the recognition algorithms with real life recordings. To build up an uniform data bank, all the participants were asked to execute the alphanumerical gestures showed in Figure 4.3. The choice of these specific gestures was done taking into account mainly their shapes features. The intention was to have a set of gestures with rather similar shapes in their composition (e.g. the curves of Gesture “0”, “3” and “8”, or the angles of Gesture “1” and “Z”) to test the recognition algorithms. The fact that the gestures are alphanumeric is to benefit their execution, since they are familiar to the participants. This allows getting a somehow homogeneous, gestures, data bank.

The use of both capturing technologies, Kinect and Vicon MoCap also allowed to test the latency of both systems in regard to the framework response.

The results are described in the next sections.

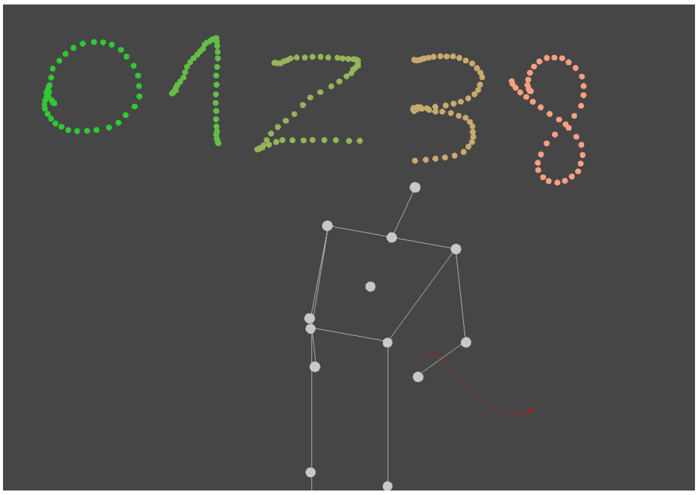


Figure 4.3: The 5 alphanumerical gestures the participants were asked to execute (10 samples of each one).

4.4 Qualitative Evaluation

This section presents the evaluation of the system from the perspective of its usability amongst the participants of the experience. The analysis presented here is grounded in questionnaires. Therefore, first one will deconstruct and explain the different sections of the questionnaire, clarifying what information was aimed to acquire and why. And after, the results are exposed. This section could also fit in the methodology, but presenting it at this stage can provide the reader with a better understanding of both, the questionnaire and the results.

4.4.1 The Questionnaire

Questionnaires are widely used in several scientific areas, with predominance in those related with human factors. Designing a questionnaire is a process that should not be overlooked, since poorly formulated questions may lead to useless data. One of the key points of the questionnaire is its validity, therefore it is wise to produce it iteratively, testing the questionnaire with small groups before consider it ready for assessment with the final audience. This approach was followed, testing and refining the document with a small test group till the questions achieved a satisfactory degree of objectiveness and purpose.

The majority of the questions offer a four-point Likert scale (Likert, 1932), “forcing” the respondents towards a positive or negative answer by removing the neutral central option (although there was also a “No Answer (N.A.)” option). At the end of each section there was a commentary box, in case the participants felt there was something left unsaid on their multiple choice answers.

The full version of the questionnaire can be found in Appendix A.

1. **Sample Characterization** - The goal of this section is to collect basic identification elements from users, such as: age, genre, degree of formation (elementary, high, graduate education or other) and main background area (Science, Arts, Humanity, Sports, Economy, or other). This will allow profiling the test group giving further insight in terms of demographic analysis.

2. **Ease of Learning** - This section is related with the comprehension of the system and respective ease of learning. The questions address if the system is intuitive, easy to learn and asks for an estimate on the time the user would need to work with the main functions of the system.

3. **Visibility** - Here, the goal is to assess if the system provides enough visual information to the user, therefore addressing its efficiency and memorability. For instance, questioning if the buttons, sliders and toggles present in the GUI control panel are correctly identified, as well as if their state (activated/deactivated) is clearly indicated.

4. **Usability** - This section aims to analyze the overall interaction with the system, and the simplicity of the displayed information. The questions concern the control the user has on the information being displayed, if this information is enough for the different operation modes and if stands out from the background. Also, it has one question concerning the aesthetic and the methods

used for the record of gestures.

5. **Complexity of Recorded Gestures** - The goal of this section is to infer on the users perception about the complexity of the gestures they were asked to record.

6. **Acceptability** - Finally, this section consists in questions to assess the acceptability of the framework. In particular, one asks the users opinion regarding the framework suitability for live artistic performances or interactive installations. There is also a more general question about the user qualification of the ZtS framework accordingly to all the parameters he previous analyzed.

4.4.2 Questionnaire Results

The following sections present the results of the data collected. On the end of each section, the main commentaries done by the participants are shown and the main conclusions are draw.

4.4.2.1 Sample Characterization

There were 29 participants on the experiment. These formed what can be classified as a heterogeneous group, with ages from 18 to around 60 years old, education backgrounds from secondary till post-docs and from different areas of formation. Next, are the graphical results that allow characterizing the experiment group.

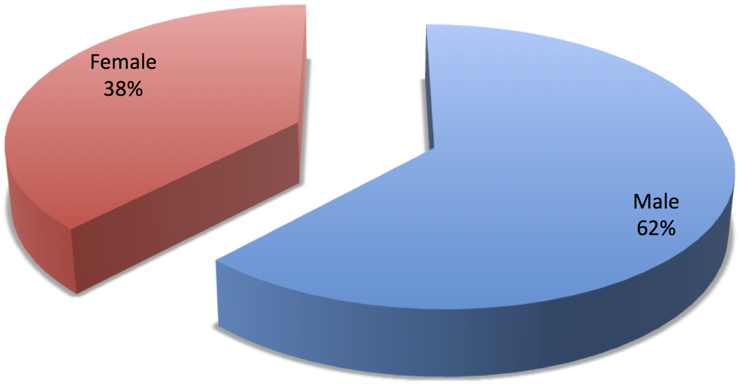


Figure 4.4: The gender distribution of the experiment sample.

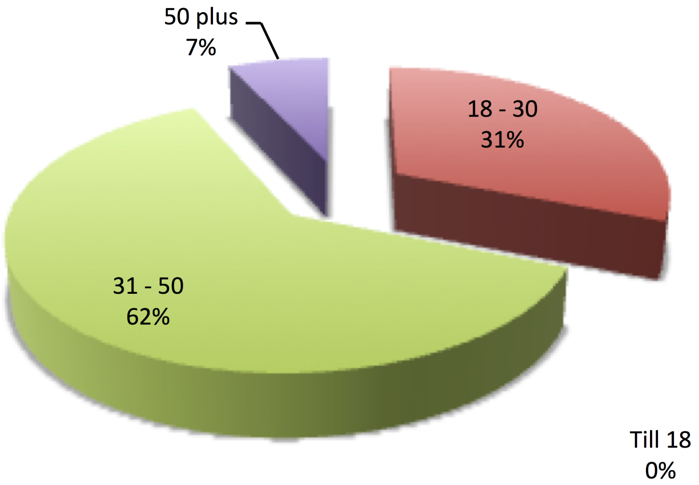


Figure 4.5: The age distribution of the experiment sample.

1. **Gender** - The group was comprised of 18 male and 11 female, what in percentage is the equivalent to 62% male and 38% female. As depicted on the chart in Figure 4.4.

2. **Age** - the ages ranged from 18 till over 50 years old. Most of them were mid

30s and curiously there were two elements in their 60s (and not very familiar with computer technology) Figure 4.5.

3. **Scholar Degrees and Main Area of Formation** - The group was mainly constituted by elements with University degrees and had one person with only secondary school. The elements were from different areas of formation. The results are depicted in Figure **??**.

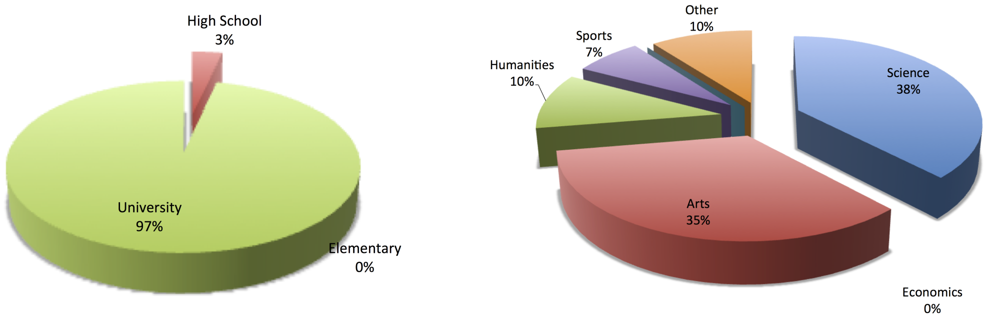


Figure 4.6: The left graphic displays the scholar degrees of the experiment sample. The right graphic displays the area of formation of the experiment sample. The field “other” refers to one element from Marketing, one with secondary school and another that specified Engineering.

4. **Artistic Performance Related** - There were 13 participants (45%) that were related with artistic performances, either by being performers themselves (e.g. in music, dance or theatre) or being involved in the development for artistic performances (e.g. sound and light producers, technical computer developers, artistic performance direction, etc.). Presented next in Figure **??**.

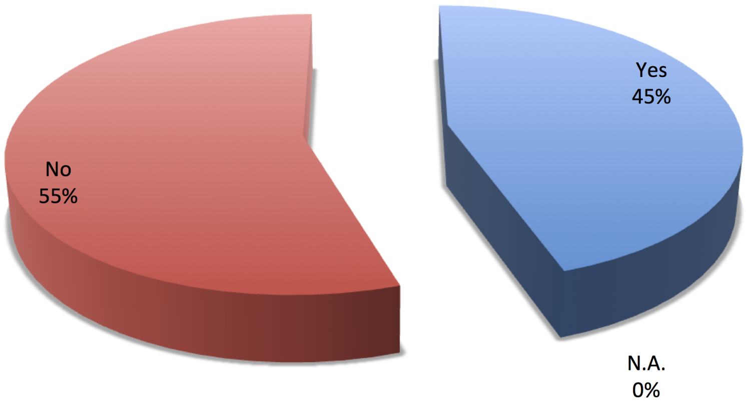


Figure 4.7: The distribution of answers regarding the relation of the participants with artistic performance.

Concerning the commentaries made in this section, some elements felt the need to specify further their formation background. From two who chose Arts, one stated particularly Dance Formation and the other Music, Musicology and Piano Performance. One who chose Science stated in particular Social Sciences and Psychology and finally one element from Humanities stated Theatre formation.

Provided that the group had so different elements it was interesting to analyze their responses about the framework, in particular relating to the subjects education degrees and formation background.

4.4.2.2 Ease of Learning

This section is aimed to analyze if the system was intuitive enough to operate and if so, how much time would the participants assume they would need to learn its main functions. The following list presents the results of this section:

- **Intuitiveness** - Figure **??** presents the answer to the question if the system is intuitive.

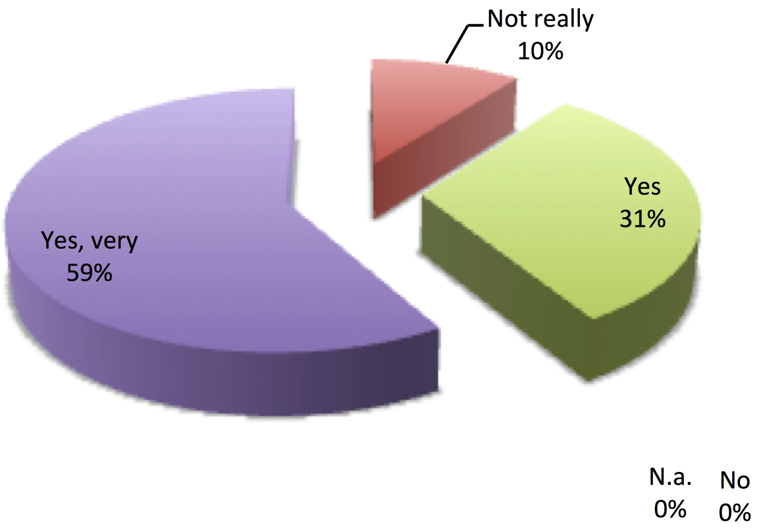


Figure 4.8: The distribution of answers regarding the system intuitiveness.

- **Learning** - The questions were: *-Is it easy to learn how to use the system?* and *-Give an estimate on how much time would you need to work with the main functions of the system*. Results in Figure **??** and 4.6 respectively.

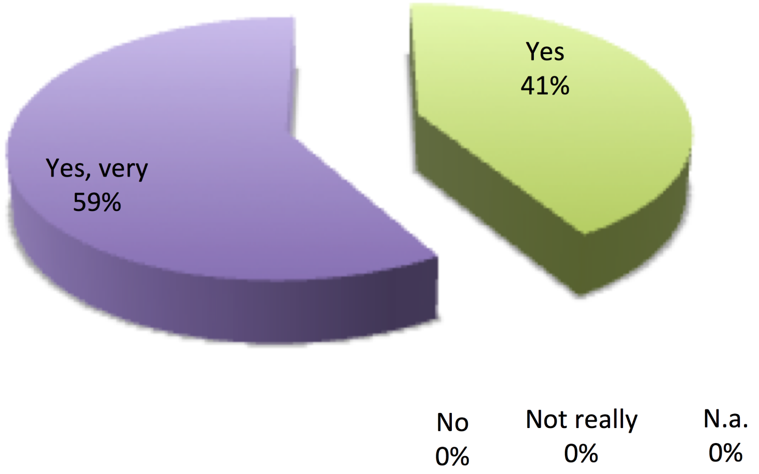


Figure 4.9: The distribution of answers regarding the system ease of learning.

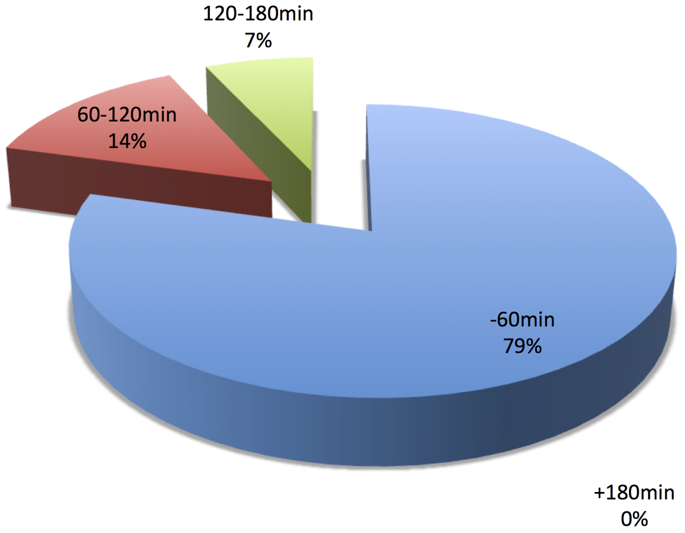


Figure 4.10: The distribution of answers regarding the expectation of how much time a user need to learn to work with the system main functions.

Is interesting to review these results and frame them with the previous characterization of the sample. Regarding the intuitiveness, the main answer, with 59% was “Yes, very” followed by the “Yes” with 31% and 10% “Not really”. The “Not really” was answered by three users that are used to develop computer applications, and their commentaries address the interface should be further developed, they considered the functions as clear, but the interface could be more user friendly (e.g. presenting a checklist of tasks for the user to follow, depending on the objective to accomplish).

Having a look at Figure 4.7, that represents the distribution of the answers per background formation, concerning the intuitiveness, one can see the answers are balanced amongst the several areas, thus demonstrating the system is rather intuitive independently of the background formation.

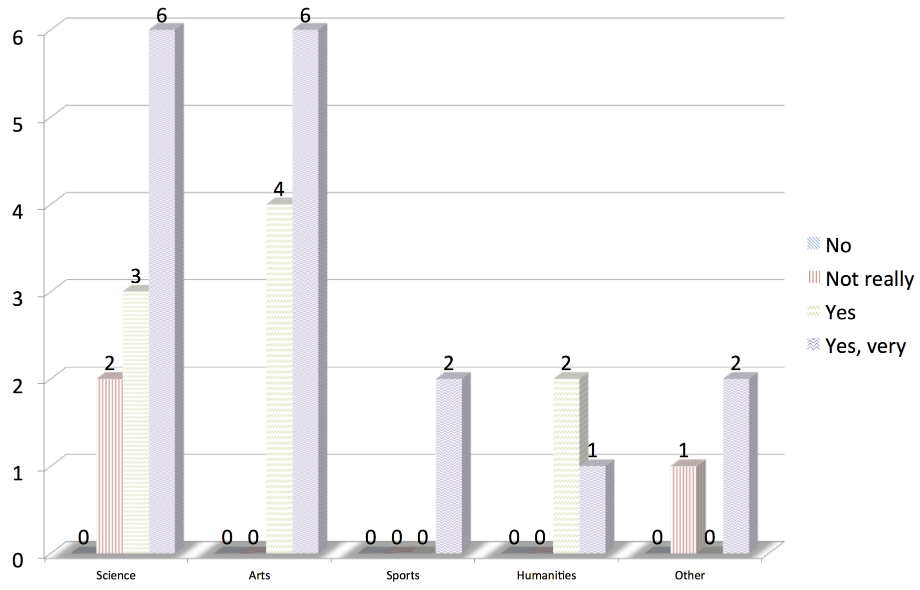


Figure 4.11: The distribution of answers per background formation.

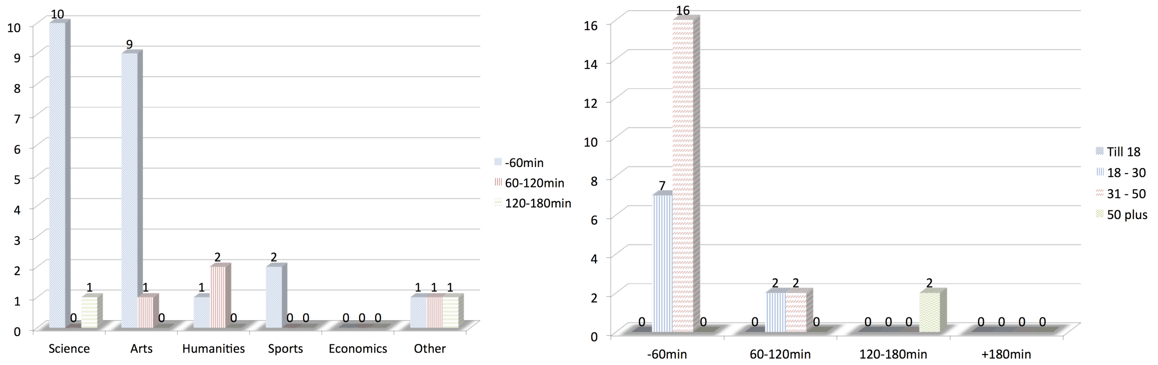


Figure 4.12: The distribution of answers regarding the time to learn estimate per formation background and per age respectively.

Another interesting fact when addressing the time to learn estimate on how to work with the framework (see Figure 4.8), is that the majority has chosen the under 60min option. The 60-120min slot was chosen mainly by the Humanities formation background elements (along with one from Arts and another from Marketing) and the 120-180min slot was chosen by the two elements aged above 50 years old. Also interesting is that despite the fact of considering the framework “Not really” intuitive, the same subject choose the under 60min option for learning, and the subject over 50 years with the High-school degree found the framework very intuitive, but admitted he needed 120-180min to learn to interact with.

This section provides already some conclusions. The majority of the subjects find the framework intuitive and expect to learn how to interact with it in under 60 minutes. Nevertheless, this should be further developed to become more user friendly, taking into account the aforementioned commentaries. In general the age and background formation influences the time users need for learning to work with new technologies, in this case, with this framework. Still, in the future, the framework should become straightforward enough in order to decrease this time window. Ideally to the under 60 minutes threshold, no matter age of formation.

4.4.2.3 Visibility

Hence the framework is to be used by individuals with different computer expertise, the visibility addresses the concern with the GUI control panel. In detail, if its buttons are correctly identified, and if there is enough visual feedback when these are activated, or modified (in the case of sliders).

The results of the main questions -“The buttons used for the main tasks are correctly identified?” and “The state of the buttons (selected/not selected, slider positions) is clearly indicated?” - are depicted in Figures 4.9 and 4.10.

In respect to the first question most of the answers were positive, the users either choose the ”Always” (45%) or the ”Almost always” (31%) options. The N.A. is relative to one user that commented he could not see the buttons during the test due to the distance of the screen. The participant that chosen ”Almost never” (is used to computer application development) commented there should be more buttons and not so many keyboard shortcuts.

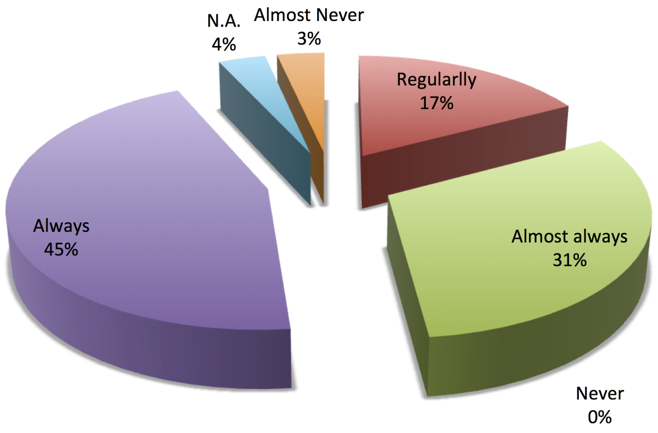


Figure 4.13: The answers concerning the buttons visibility.

In the case of the second question, if the state of the buttons is clearly identified, there was a 62% choice of the ”Always” option and an equal 17% for the ”Almost always” and ”Regularly” options. Again the N.A. was chosen by the user that could not see properly at that distance.

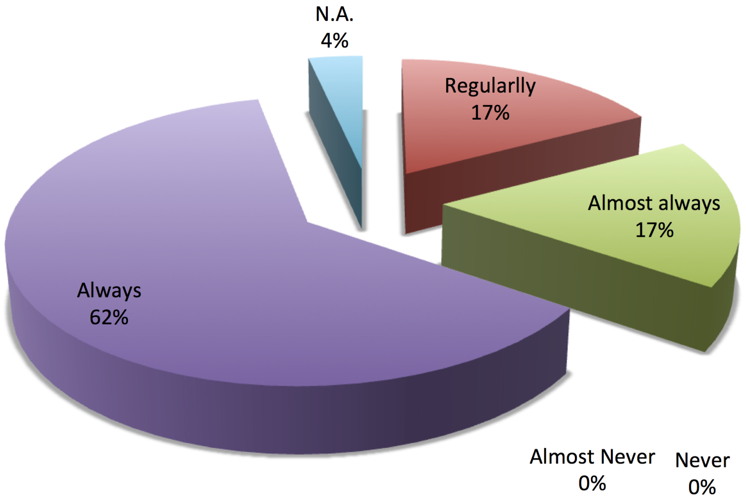


Figure 4.14: The answers concerning if the buttons state is clearly identified.

Regarding this section the main commentaries were to use more colors to better differentiate the several buttons, also to use a simpler nomenclature (not so technical). Another good input was to add the ability of language choice (is working only in English presently).

4.4.2.4 Usability

This section was the longest, comprising questions about the use of the framework. For instance, the control the user has on the information displayed, if this information stands out from the rest, or if the system keeps the user informed of every task being performed. There was also a question addressing the method and accessory used to record/delete gestures usability. Again, the main results are limned in the following Figures 4.11, 4.12 and 4.13.

Regarding the first question about the control the user feels he has on the information being displayed, the principal choice was ”Yes, very” control, with 59%.

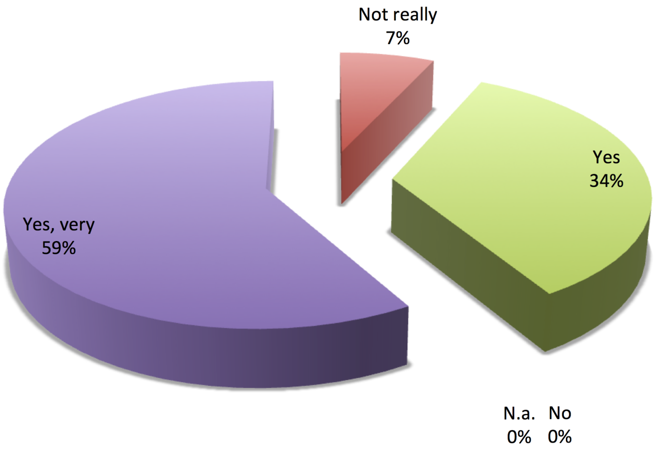


Figure 4.15: The distribution of answers about the control of information displayed.

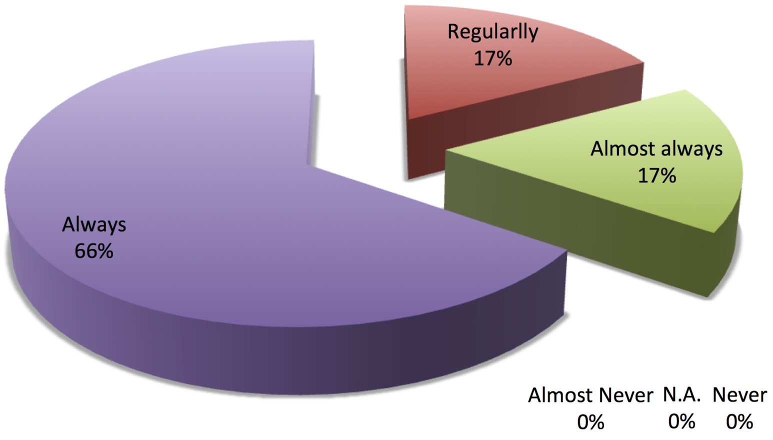


Figure 4.16: The answers concerning the system feedback when performing tasks such as recording performance, gesture training, recognizing, etc.

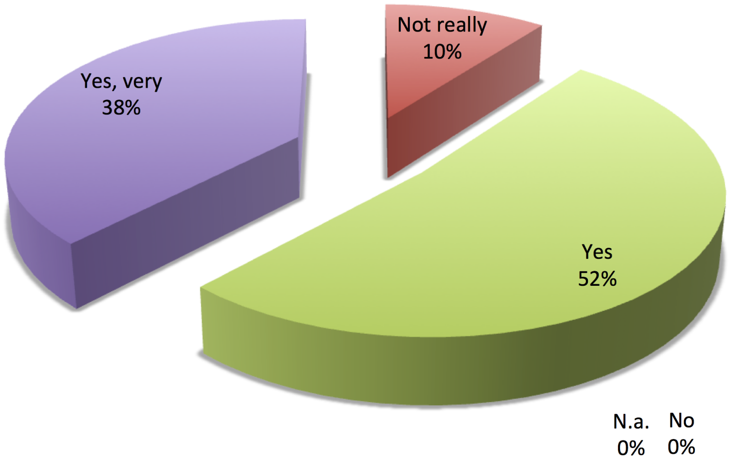


Figure 4.17: The distribution of answers about the method and accessory for recording gestures .

Bearing on the question about the information feedback of the system when performing the tasks, in particular those that did not involve directly the GUI - tasks such as recording a file or training a gesture recognition algorithm, among others - the answers were positive. The ”Always” had the higher percentage with 66% followed by the ”Almost always” with 17%.

The question regarding the suitability of the accessory for recording gestures, in this case a wireless mouse, the distribution of answers had a predominance on the ”Yes” with 52% followed by the ”Yes, very” with 38% and with a 10% choice of the ”Not really”.

The commentaries made on the end of this section may explain some of the obtained results. Some refer the color of information should stand out more from the background. Two users stated the mouse was easy to use, but in the case of complex gestures, it would be better to have a customized accessory that would fit better in the hand. Another good input from one user was that the representation of the gestures could be done in perspective (in 3D), since presently the representation is done in 2D.

4.4.2.5 Complexity of Recorded Gestures

The users were also asked to sort the gestures according to their complexity of execution (from 1 to 5 being 1 the less complex and 5 the most complex). Although the questionnaire explicitly asked for a scale with only one choice of complexity value for each gesture, some of the participants (seven of them) classified the gestures without building the required scale. Those were not included from the following graphic (Figure 4.14).

Although the classifications are disperse along the several complexity values there are some facts that are clear. The gesture considered most complex to execute is the “8” with 72,7% of choices, followed by the “3” equally classified as complexity 4 or 3 with 31,8%. Then the Gesture “Z” was mostly classified as complexity 3 (36,4%), the circle “O” was considered as complexity 2 (31,8%) and the Gesture “1” of complexity 1 with 36,4%.

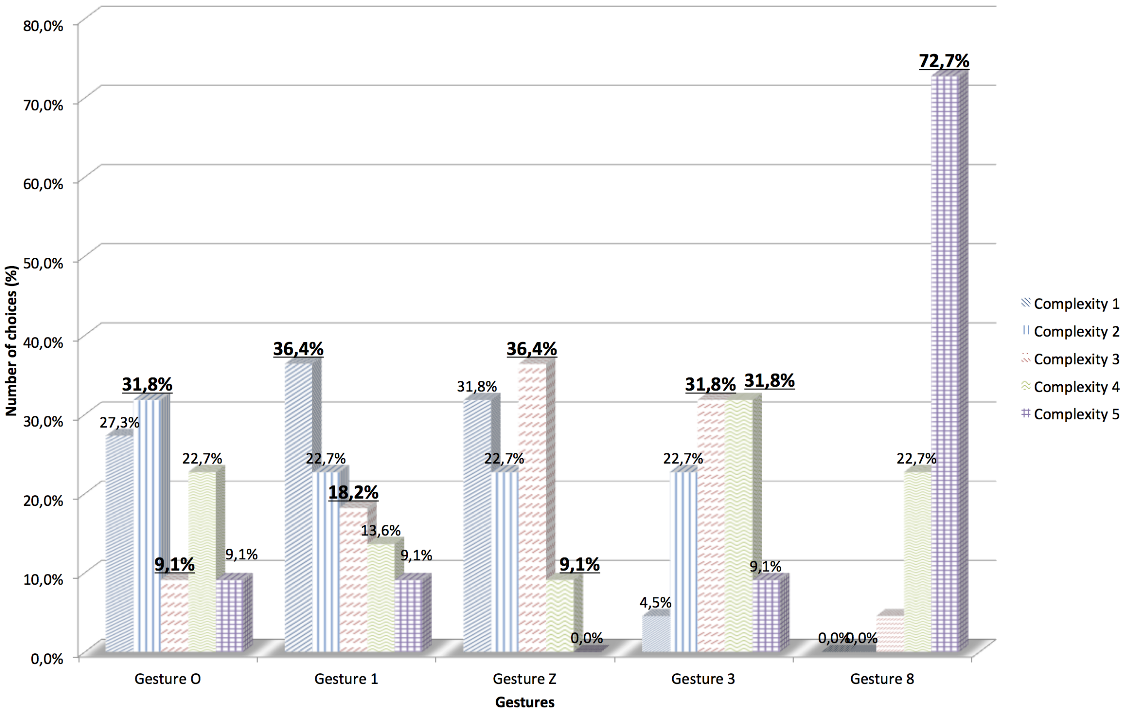


Figure 4.18: The complexity scale in regard to the recorded gestures. The higher percentage of each complexity classification value is shown underlining the value. Seven participants were left out, because did not build correctly the scale.

Interesting enough, some users found the number "1" to be more complex than what was expected when designing the experiment. Some commentaries point out the difficulty in executing straight vertical lines, opposing to the facility in doing horizontal lines that lead to the "Z" balancing amongst classification values 1 (31,8%) and 3 (36,4%).

Knowing the participants classification concerning the complexity of the gestures performed, is interesting to compare it to the system performance when recognizing it, in Section 4.5.1. The expectation that the recognition rate is inversely proportional to the complexity classification is not observed in the following results. More details are provided at the following sections.

4.4.2.6 Social Acceptability

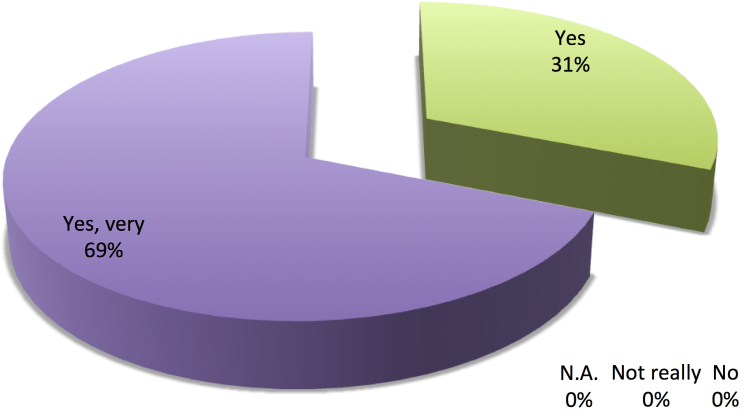


Figure 4.19: The participants answers on the acceptance of the framework regarding its use in live performance or installations (e.g. public display, interactive).

The last section of the questionnaire addressed the overall opinion of the participants regarding the framework social acceptability. The questions made to the participants of the experiment were:

- Do you think the framework can be used in live performance scenarios or as other type of installations (public display, interactive)?

- Considering all the parameters you have analyzed how do you classify the ZtS Framework?

The answers are graphically displayed in the Figures 4.15, 4.16 and 4.17.

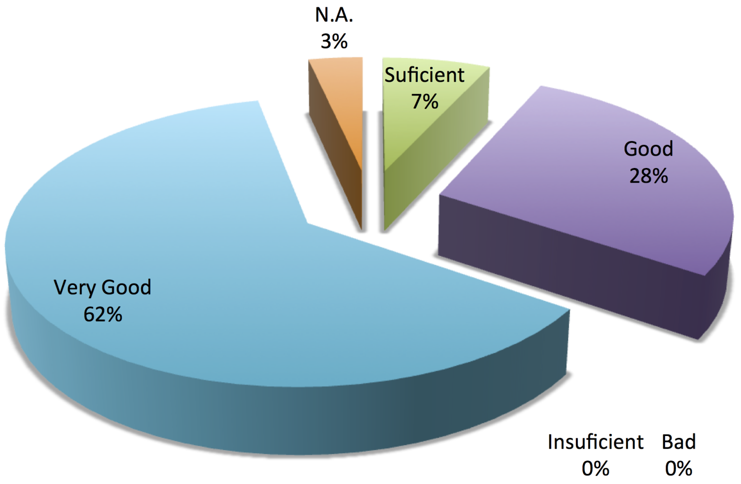


Figure 4.20: The overall classification of the ZtS framework considering all the aspects observed by the participants.

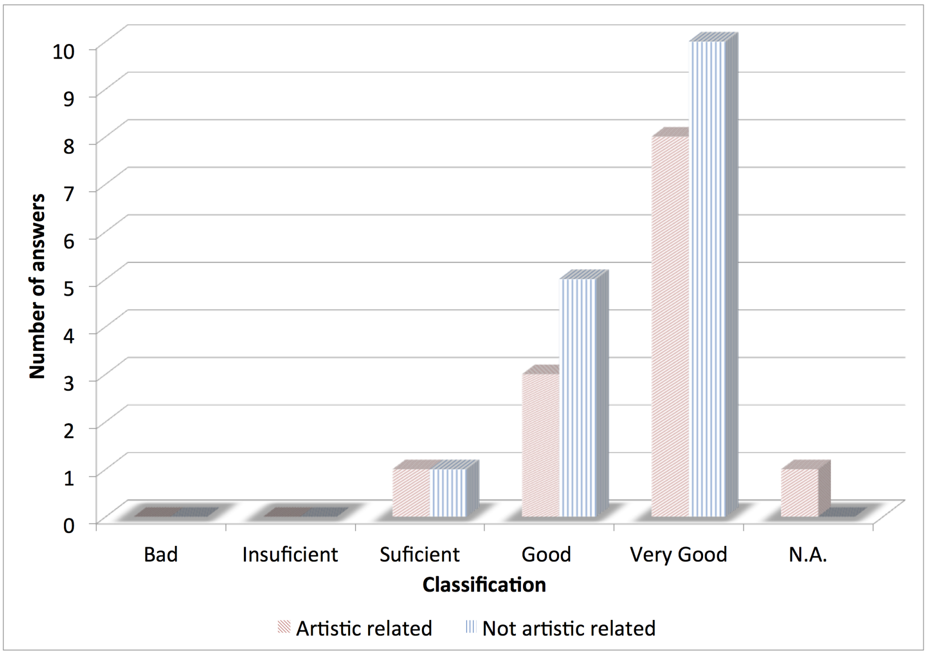


Figure 4.21: The overall classification of the ZtS framework concerning the users that are related with artistic performances.

These results about the acceptability are encouraging. First, the answers were generally positive and all the users viewed the potential of the framework for being used in live performance or other types of interactive installations. Second, knowing the overall classification of the participants that are usually involved in performative aspects, reveals the system can have a future in the artistic performance domain.

On the end of the questionnaires there was a commentary box where the participants were encouraged to write something, for instance: a critic, a compliment, suggestions of implementations, etc.

The commentaries were in general positive, regarding the overall working of the framework and its respective functions. Nevertheless, also point out some improvements such as, to make it more simple and intuitive, specially regarding the use of it by people with different levels of computer or technological skills. Another interesting input was to add a help button linking to a manual document or video tutorials.

Some participants mentioned it was easier to perform the gestures asked, without looking at the screen and that actually the outcome seamed better. These encourage to further work with this “No feedback” possibility having in mind future applications in medical surgeries, music creation or domestic use (e.g. *intelligent houses*). On the other side, some wanted more feedback besides the visuals, something like task confirmation sounds or pop-up windows.

In a more lateral note, another positive result, that one was not expecting, was the use of the framework just in the draw mode. In the end of one session that had six participants attending, they started playing Pictionary[1] with each others and would not leave the Laboratory. This lead to the consideration of using of the framework also for simple entertainment purposes.

[1] Pictionary is a guessing word game, with players trying to identify specific words from their teammates drawings.

In summary, the participants found the Framework useful, very interactive and responsive.

4.5 Quantitative Evaluation

Besides the qualitative evaluation provided by the participants of the previous experiment, the framework was also evaluated quantitatively. In this case the main technical aspects were considered for evaluation, especially the performance of the different MoCap systems, and recognition algorithms.

The following subsections describe the methods and evaluations done.

4.5.1 Evaluation on the Recognition Algorithms

A challenge when working with ML algorithms is collecting enough real-world data to partition into three substantial training, validation, and testing sets. The training set is used to fit the models. For model selection, the validation set is used in tuning the model parameters to yield the best results. For model assessment, the chosen model prediction recognition rate is estimated using the previously unseen testing set (Hastie et al., 2003).

Regarding the evaluation of both recognition algorithms, one of the main differences, already known, between HMM and DTW is on their training.

In one hand, the DTW is an algorithm that is able to do pairwise comparison of signals, therefore it only needs one example of a ground-truth signal to instantly start looking for a similar. This brings the benefit of simple, immediate training and recognition, that in the purpose of an artistic performance can be a key feature (e.g. for live improvisation). Nevertheless this simplicity of training has some disadvantages, in particular depending on the signal complexity, as will be reviewed in the next sections.

In the other hand, the HMM relies on statistics and probabilities to do a correct recognition. Thus, the more data is used to train it, the better results will be achieved. This implies the gathering of more data until one can start the recognition process. Therefore, one will need more time until accomplish the training (and probably this training should be done before a performance), but the achieved recognition results can compensate the work.

Another key aspect of the evaluation one have to consider, is if the training is done by the same person that is going to use the system or by someone else. This will influence the results of the recognition. Consequently there are two approaches for testing: the “first-person” and the “third-person” methods. The former consists on using different samples of the same data set (same user) for training, refining and testing (one data set divided for the three functions). The latter comprises the use of different data sets (different users) to train, refine and then test (one different data set for each function).

On the experiment explained previously, the 29 users were asked to draw 10 samples of each of the gestures (5 gestures) depicted in Figure 4.3. This generated a database of around 290 samples for each gesture and a total of 1450 gesture samples. Having in mind the enormous variety of gestures a person can perform, this database can be classified as small, nevertheless this can be considered a base test, knowing that if the recognition algorithms perform well enough with this database, then one is on the right track and can expand it in the future. The features previously explained in Chapter 3 (Section 3.4.3) were computed for each gesture sample in order to use them on the recognition algorithms.

The next sections describe the evaluations and results made for each recognition algorithm using the recorded gesture database.

4.5.1.1 DTW Evaluation

For the DTW evaluation, since it only requires one gesture sample to start the recognition process, it is very difficult to determine one gesture sample that will represent the majority of that gesture database.

Therefore, the first approach taken, using the “third-person” method, was on choosing a random example of each gesture, train the 5 gesture DTW and then testing the models against 90 unknown samples of each gesture (450 test samples total).

The second approach, using the “first-person” method, was to go to each data set (50 samples - 10 for each gesture), use one of the samples of each gesture to train the 5 gesture DTW and then testing this against the other 45 samples. Repeat this 10 times to get the 90 sample test for comparison.

The results are depicted in the confusion matrices Tables 4.1 and 4.2.

Table 4.1: The confusion matrix of the 90 samples DTW test of each gesture with the “third-person” method.

		Prediction					**Recognition**
	Gesture	**0**	**1**	**Z**	**3**	**8**	**Rate**
Actual	**0**	44	12	5	7	22	49%
	1	1	86	0	0	3	96%
	Z	2	6	45	14	23	50%
	3	17	0	2	66	5	73%
	8	28	5	2	7	48	53%

Average Recognition: 64%

It can be perceived an improvement of the results when using the “first-person” method. This may be explained by the fact of each subject executes the gestures in different ways, so when using one of their own gestures as a training set, obviously this will provide better results. Nevertheless, the results of the “third-person” method are good, in particular for the Gesture “1”, which did not change the recognition rate (96%). This may serve as an indicator on the simpler kind of

Table 4.2: The confusion matrix of the 90 samples DTW test of each gesture with the "first-person" method.

		Prediction					Recognition
	Gesture	**0**	**1**	**Z**	**3**	**8**	**Rate**
Actual	**0**	51	2	8	12	16	57%
	1	2	86	0	0	2	96%
	Z	0	6	71	12	1	79%
	3	1	5	4	80	0	89%
	8	3	6	13	5	63	70%

Average Recognition: 78%

gestures that work best with this recognition method.

Another interesting point, Gesture "1" was considered the simplest to execute regarding the previous complexity classification done (in Section 4.4.2.5) and is the one with better recognition rate. However, Gesture "0" was the second in the complexity scale, but has the lowest recognition rate with 57%. This may be explained by this gesture being codified by all the "codewords" on the feature normalization process (referring to Section 3.4.3), or can even be explained by the discrepancy on the recording of the several examples.

The overall recognition rate of the DTW for the five distinct gestures was of 78%. Therefore, one is led to conclude its advantage for real-time improvisation, has the downsize of limiting its recognition rate to simple gestures, such as the Gesture "1" (96% recognition rate). One assumes it will perform well distinguishing simple one-direction gestures (e.g. vertical top to bottom gesture, or horizontal left to right gesture).

4.5.1.2 HMM Evaluation

For this evaluation, the methodology consisted in training one HMM for each gesture (using 130 samples), and refine it using the 70 samples of validation. Once, the best description models are found, these are tested against the 90 remaining samples (“third person” method).

The main refinements made were in respect to the number of hidden states that best suit each gesture description. Therefore, each gesture HMM was trained with 3, 5, 7, 9 and 11 hidden states, then they were analyzed in terms of accuracy rate against the 70 samples. Besides the accuracy, also the training time was measured, since the number of hidden states has direct relation with the calculations need to build the HMM transition matrices. The respective results are depicted in Tables 4.3 and 4.4.

Table 4.3: The time it takes to train each of the gestures (using 130 samples) regarding the number of hidden states. The last column presents the average time of training per number of hidden states (in milliseconds).

HMM Train Times (ms)						
Hidden States	**Gesture 0**	**Gesture 1**	**Gesture Z**	**Gesture 3**	**Gesture 8**	**Avg Time**
3	930	140	398	361	627	491
5	2130	4015	2247	1611	2535	2508
7	5953	6978	5869	4338	12507	7129
9	10416	5925	12364	15208	10942	10971
11	14053	31299	17541	11270	18160	18465

Analyzing Table 4.3 one is able to notice the more hidden states are used, the more time it takes to train each HMM. It is also interesting to note that the Gesture “1” is the one that shows larger increment in time, in respect to the hidden states used (particularly when using 11 hidden states, the training took 31,3 seconds). This may be due to the simplicity of the gesture only requiring few states, and when it is creating a model using so many hidden states, this makes the training process diverge and go through more iterations until being complete.

Important to realize, although the training times can take up to a few seconds, the recognition time is much more efficient, practically instantaneous.

Table 4.4: The recognition rate of the HMM using different number of hidden states (testing against the 70 refinement samples).

	Gestures Recognition Rate (%)					
Hidden States	**0**	**1**	**Z**	**3**	**8**	**Avg. Recog.**
3	76%	94%	91%	90%	97%	90%
5	73%	**100%**	91%	**96%**	**100%**	92%
7	79%	100%	91%	96%	100%	93%
9	**86%**	100%	**96%**	91%	100%	**95%**
11	84%	96%	94%	91%	96%	92%

Regarding Table 4.4, one can realize that when using 11 hidden states to describe the gestures, their respective performance of recognition decreases. This again is probably due to the elevated number of hidden states used, when compared to the number of observable symbols (one is using 12 different “codewords” in this case, as described in Chapter 3, Section 3.4.3). The Table 4.4 presents results only up to 11 hidden states, since from there on the results were even worse.

One can also perceive two distinct results, regarding the recognition rates on the refinement samples:

1. The best results, for each gesture, are achieved using the following combination of hidden states:

   - Gesture “0” - 9 hidden states - 86%.
   - Gesture “1” - 5 hidden states - 100%.
   - Gesture “Z” - 9 hidden states - 96%.
   - Gesture “3” - 5 hidden states - 96%.
   - Gesture “8” - 5 hidden states - 100%.

2. The maximum overall average recognition is obtained when using 9 hidden states for every gesture (95%).

Following these enumerated results, the HMM were trained again and tested against the last 90 samples, using two approaches:

1. This approach consisted in training the HMM with the number of hidden states that achieved better results (in less time to train when there is a tie in performance) in the refinement test, for each gesture. Thus, considering the former result enumerated, the Gesture “0” was trained with 9 hidden states, Gesture “1” with 5, Gesture “Z” with 9, Gesture “3” with 5 and Gesture “8” with 5. The results achieved with this setup are displayed in the confusion matrix present in Table 4.5.

Table 4.5: The confusion matrix using different number of hidden states in the training of each HMM.

		Prediction					
	Gesture	**0**	**1**	**Z**	**3**	**8**	**Recognition Rate**
Actual	**0**	**83**	0	1	1	5	92%
	1	1	**89**	0	0	0	99%
	Z	0	0	**85**	3	2	94%
	3	26	0	0	**63**	1	70%
	8	1	0	0	0	**89**	99%
	Hidden States	9	5	9	5	5	

Average Recognition: 91%

2. This approach consisted in training every HMM with 9 hidden states, to see if accordingly to the former evaluation, the overall recognition would improve. The confusion matrix is represented in Table 4.6

Analyzing both Tables one can observe that the main difference is on the Gesture “3” recognition. When using 5 hidden states (the suitable number of states accordingly to Table 4.4) this has a 70% recognition rate and when using 9 hidden states, for all gestures, this achieved 90% recognition rate. One is able to realize also that the main confusion made on the Gesture “3” recognition was against Gesture “0”.

Table 4.6: The recognition rate using 9 hidden states for each gesture HMM.

		Prediction					
	Gesture	**0**	**1**	**Z**	**3**	**8**	**Recognition Rate**
Actual	**0**	**77**	1	0	5	7	86%
	1	0	**89**	0	0	1	99%
	Z	1	0	**87**	2	0	97%
	3	8	0	0	**81**	1	90%
	8	0	0	0	0	**90**	100%
	Hidden States	9	9	9	9	9	

Average Recognition: 94%

The performance of the remainder gestures stood almost the same between both tests. Gesture “0” decreased shortly, Gestures “Z” and “8” increased and Gesture “1” remained equal. Overall the recognition rate improved from 91% on the former test to 94% on this, mainly because the aforementioned confusion amongst Gesture “3” and “0” achieved better results.

This leads to the conclusion that the fact of using different number of hidden states to describe different gestures influences the recognition rates, and that in the particular case of this set of gestures, defining a single good number of hidden states for all, provides the best results.

Although there are not implementations exactly like this in the literature, when comparing this recognition rates to similar HMM experiments (Kim, 1999), (Elmezain and Al-Hamadi, 2009) one is able to conclude the results match the former, and are in some cases better.

4.5.2 Evaluation on the Acquisition Module Latency

Concerning the Acquisition Module, since the experiment took place in a controlled environment (with no occlusions that could harm the capture of gestures), and the capture was synchronized in the ZtS framework (thus recording the same gesture data for both systems) the main evaluation performed was in regard to the latency of both capturing technologies.

By simple observation, the Vicon seems faster and more fluid than the Kinect outcome, but hence the Kinect was directly plugged into the computer running the framework and the Vicon had to transmit its data, first internally to the ofxOSCVICON and second through a router establishing a Local Area Network (LAN), this can provide different results in regard to the overall framework response time.

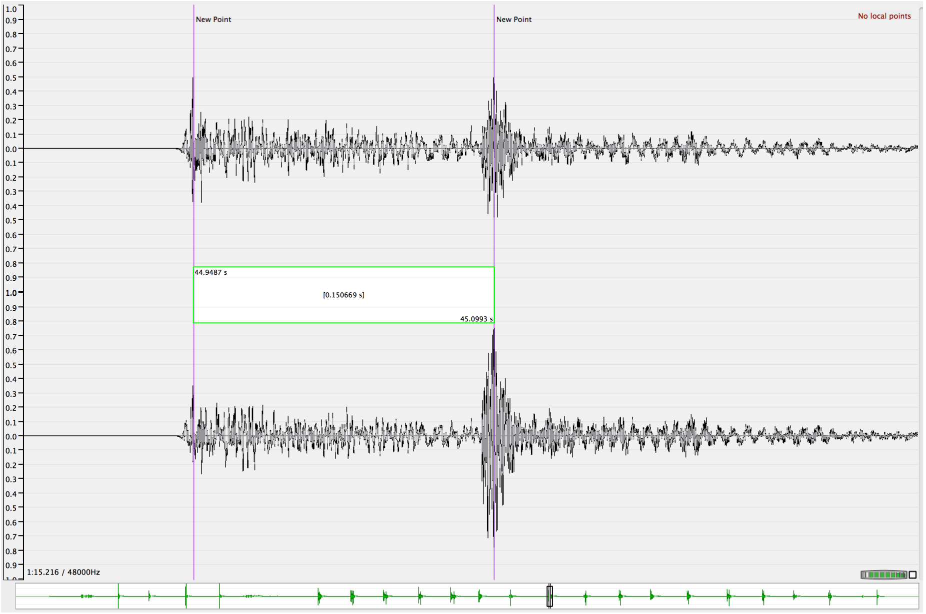


Figure 4.22: The latency measurement. Top and bottom waves are from the same audio signal, since it was captured in stereo. The wave peaks are identified by the vertical lines. The first sound wave peak is the clapping, the second is the sound response of the system.

Therefore, the experiment engineered for measuring the latency consisted in the following: when a detected subject claps his hands the framework responds immediately triggering a distinct sound if detecting the *“hands together pose”* from Kinect or from Vicon. Both clapping sound and framework response sounds are captured externally by a microphone and then the difference between sound waves is measured (if there is latency in the sound capture setup, this is constant for every sound captured, thus not influencing the overall latency measure of the system). After first experiments, the systems were responding very close and analyzing the sound waves was very difficult due to the overlapping sounds. Therefore the measures had to be made again and testing one system at the time. Same setup for both, just volume down for one and up on the other on the first test and then switch. Figure 4.18 shows one of the set of sound waves captured, in this case for the Kinect test.

The test was done 10 times for each setup and the Table 4.7 depicts the latency results.

Table 4.7: The measurement of the MoCap system latency times in miliseconds. First column depicts the Vicon, second the Kinect and the third is the difference (Kinect-Vicon). It is also presented the average measurement and the standard deviation measure.

	Time Delay (ms)		
	VICON	**Kinect**	**Difference**
	136	158	22
	150	164	14
	159	170	11
	128	156	28
	145	150	5
	137	141	4
	156	153	-3
	144	158	14
	123	147	24
Average:	**142,00**	**155,22**	**13,22**
St. Deviation	**12,14**	**8,76**	**10,21**

These results are not what one expected. In theory, Vicon (since working at 120 fps) should provide latency times inferior to the ones observed. To derive further conclusions one decided to measure the time difference between consecutive OSC messages arriving for the same human body joint (e.g right hand). Surprisingly the Vicon MoCap has, in average, a measured time of 33 milliseconds between consecutive frames what signifies that it transmits the data at only 30fps, even if working at 120fps. Investigating the reason of such, one discovered that in the transmission process it drops 3 frames and sends each fourth one. This can be explained by the Server-Client transmission buffers being filled up with data. Nevertheless, is important to mention, this might be a problem of the Vicon Blade DataStream Server-Client transmission and not a problem in the ofxViconOSC module (further explained in Chapter 5, Section 5.5.4), since this was tested and debugged with other OSC transmission program working at 120fps and it maintained the transmission rate. Further tests are needed to resolve this problem.

The overall system latency measured with the Kinect was, in average, of 155 milliseconds. There are two perspectives to discuss this result:

1. Regarding the clapping, it is a very sudden movement. Moreover, if one considers the clapping sound and the respective audio system response, the latency between both sounds is noticeable. Therefore, in this particular case, the latency is not satisfactory. In future work, one will implement anticipation methods to overcome this problem (Rett and Dias, 2007; Vamplew and Adams, 1995).

2. If one take into account the remainder gestures performed by the participants of the aforementioned laboratorial experience (Figure 4.3). These have a duration ranging from 1 second until 5 seconds. Consequently, for those kind of gestures, this latency measure can be considered residual (Licsár and Szirányi, 2005; Wolf et al., 2002).

Important to mention, although the latency is of 155 milliseconds, the jitter (variation between latency measures) is very small, having a value of 8,76 milliseconds. Consequently, one can consider the latency measure as constant, therefore enabling the users to predict the delay and anticipate themselves the movements and gesture performing.

4.6 Artistic Applications

Besides the experimental setup described previously, the system was also tested in real-life scenarios. The following sections will describe the applications of the ZtS framework in the performative arts and interactive installation domains. Allowing thus, analyzing its performance out of the laboratorial controlled environment.

4.6.1 Using ZtS in an Artistic Performance

MisoMusic Portugal[2] was commissioned to create an interactive multimedia Opera (to debut in September 2013), by the renown Polish Festival *'Warsaw Autumn' (Warszawska Jesień)* [3].

Knowing the work developed in the scope of this thesis, MisoMusic proposed the use of the ZtS framework in the Opera to control real-time audio samples and the direct sound input of the voice of one performer.

But before entering on further details about the developments made, the following section will describe briefly the Opera, named "A Laugh to Cry". This will set the benchmark for the work developed in the ZtS framework.

4.6.1.1 About the Opera "A Laugh to Cry"

A *Laugh to Cry* explores some primary concerns, which have always haunted human beings, and reveals them from the perspective of our contemporary globalized world. The opera is shaped like a meditation on the hegemonic power of

[2] Music Portugal Cultural Association, which has the status of Portuguese Public Utility Institution, was born as an extension of Miso Ensemble, to develop and promote contemporary musical creation in Portugal and Worldwide. Its founders are Paula and Miguel Azguime, composers, performers and directors that since the foundation of the Miso Ensemble in 1985, develop their work tirelessly in the field of new music, contributing actively to expand the contemporary way.

[3] `http://warszawska-jesien.art.pl/en/wj2013/home`

the destruction of memory, the devastation of the Earth and even the collapse of humanity. It evolves in the fringes between dream and reality, between the visible and invisible, being divided in several acts where five characters, two sopranos, one bass and two narrators (a female and a male voice), live and dwell constantly between these two parallels. The opera also involves seven acoustic instruments: flute, clarinet, percussion, piano, violin, viola, cello, as well as live electronics and extended video scenography.

A *Laugh to Cry* is a metaphysical theatre embodying eternal archetypes with music and multilingual libretto by Miguel Azguime.

A *Laugh to Cry* pursues Miguel Azguime goal, as poet and composer, to grasp an ideal balance between language and music, to merge the language semantic and metaphorical components with its sonic values, in order to achieve his concept of "speech as music and music as speech". A Laugh to Cry extends Miguel Azguime research on voice analysis, re-synthesis and processing, aiming at creating a dynamic continuum between timbre, harmony, rhythm and voice spectra.

4.6.1.2 System Requirements

The framework had to be *tailored* to the composer/performer (Miguel Azguime) needs. Specifically, he wanted to control sound samples and live voice input with his movements and gestures. In this case, the framework was adapted with several triggers that controlled sounds in a MAX/MSP patch (this patch was developed by a fellow researcher, André Perrotta).

The framework went through a series of tests and refinements, in particular to respond to the composer choices and performer abilities.

In the end the ZtS framework enabled several types of sound control:

- the trigger of sound samples with the movement velocity of the hands of the performer;

- the cycle through eight banks of sound samples by performing a gesture;
- the trigger of *capturing* a sound action (sound sample or live voice input). The performer was able to *freeze* a sound when he performed a *holding hands* pose. This enabled the performer to control the captured sound in terms of pitch, reverb, feedback and loudness. When he wanted he just needed to do a more sudden movement with both hands (exceeding a pre-determined hand movement velocity threshold) to release the sound.

In Figure **??** one can see the hardware setup.

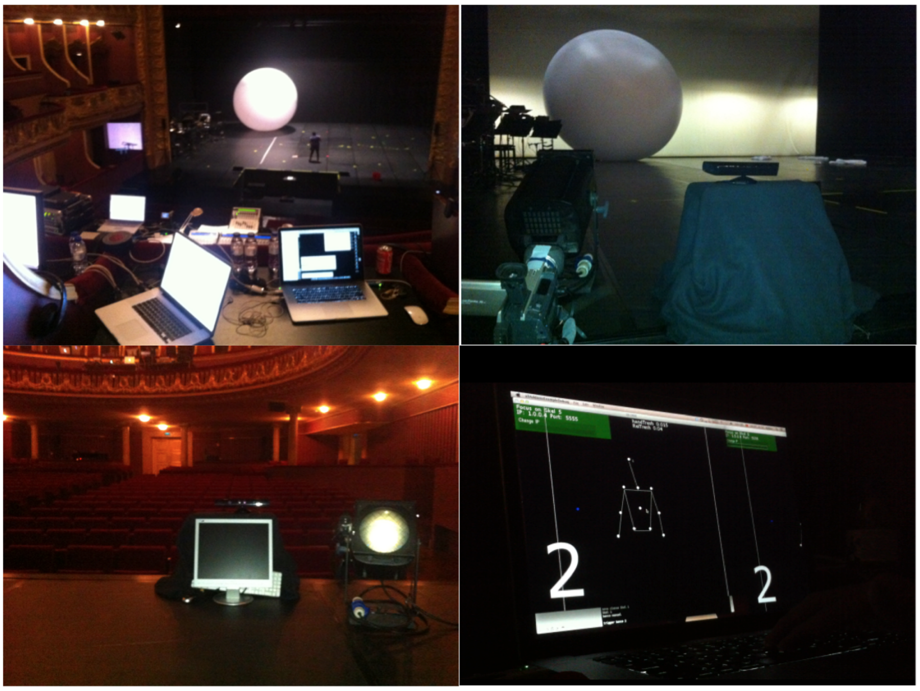


Figure 4.23: The setup used for the Opera "A Laugh to Cry". On the top left image is the view from the technical sound area. The top right and left bottom images present the view of the ZtS setup. The last photo illustrates the view Miguel had when using the system.

A Microsoft Kinect was used to capture the human body and an Apple MacMini running the ZtS was hidden under a black cloth. The framework was sending the control triggers to the sound computer on the technical regie at 25 meters of distance. One setup a LAN to enable the trigger transmission. Also, in this case, the performer wanted the visual feedback to make sure he was in the right position, so there was a 15 inch LCD on stage (also hidden from the audience).

The framework ended up being used for the solo of one of the main Opera characters, performed by Miguel Azguime himself. The ZtS framework travelled with the Opera throughout the entire tour. In the first performances the setup was done by the author of this thesis, which also supervised its function during the Opera. Since everything ran smoothly on the first three performances of the Opera (two in Lisbon and one in Poland), for the Sweden leg of the tour (four more presentations) one of the Opera technicians received a brief formation on how to do the setup and execute the ZtS. Important to realize that he did the setup alone and operated the framework on those four shows without any problem, thus revealing the usability of the framework.

In sum, the result of the developments made specially for the Opera use was very interesting. The relation between human movement/gestures and sound manipulation was immediately perceived by the audience, therefore creating a particular arouse during that part of the piece. Of course the principal credit goes to the performer, in this case Miguel, which learned very quickly to interact and get exactly what he wanted from the framework, when he wanted, thus enabling him to add extra layers of emotion and enhancement to the solo he performed.

In the following section is the statement Miguel gave regarding the use of the framework.

4.6.1.3 Evaluation

Once the Opera presentations were finished, one asked Miguel Azguime, the author/performer and main user of the ZtS framework, to answer a few questions about the system and to transmit his opinion about it. Here is a literal quote of the text he sent.

“Since the beginning , in the design of the opera “The Laugh to Cry”, were implicit certain technological aspects and modes of interaction, which had not been possible to research, develop and use in previous works. In particular the relationship sound - gesture took this project a clear role that was intended to develop and the Zatlab System developed by André Baltazar came precisely to meet this desire, having been adapted to respond to musical, performative and expressive purpose I intended for a crucial moment of the opera and true climax of the symbolic and narrative discourse thereof.

Playwright and musical composition itself for this decisive moment in the opera were designed to take advantage of the interaction with the system and conditioned by the type of gestural control offered by the same.

A clear perception to the public that the gesture is that of inducing sound, responsiveness of the system to allow clarification of musical and expressive speech, effectively ensuring the alternation between sudden, rapid, violent gestures, sounds on the one hand and modular suspensions by gesture in total control of the sound processing parameters on the other, constituted a clear enrichment both in terms of communication (a rare cause and effect approach in the context of electronic music and it certainly is one of its shortcomings compared with music acoustic instruments) and in terms of expression by the ability of the system to translate the language and plastic body expression.

Clearly, as efficient as the system may be, the results thereof and eventual artistic validation, are always dependent on composite music and the way these same gestures are translated into sound (or other interaction parameters) and therefore is in crossing gesture with the sound and the intersection of performance with the musical composition (in this case) that is the crux of the appreciation of Zatlab. However, regardless of the quality of the final result, the system has enormous potential as a tool sufficiently open and malleable in order to be suitable for different aesthetic, modes of operation and different uses.”

4.6.2 Using ZtS in a Public Interactive Installation

Another application of the system consisted in making it as an interactive installation at FestivalIN [4], Lisbon.

The FestivalIN was announced as the biggest innovation and creativity aggregating event being held in Portugal, precisely in Lisbon at the International Fair of Lisbon. It is described as an unique event that integrates, in a practical, dynamic and consistent way, the core concepts associated to Creativity and Innovation. It presents itself as an absolutely innovative event, anchoring sensorial experiences (physical and virtual interactions), crossing different areas of the Creative Industries. It is a space which involves people, ideas and experiences and promotes, both nationally and internationally, Portugal most creative possessions, boosting its authors, creators and entrepreneurs in a worldwide scale.

4.6.2.1 System Requirements

Departing from the developments made to the Opera, the framework was adapted to be more responsive and easy to interact with. The users were able to trigger and control sound samples, much like Miguel did on the Opera, however they did not had the same level of control.

Since the purpose was to install the application at a kiosk and leave it there for people to interact with, the visuals were further developed to create some curiosity and attract users. The human body detection algorithm was also customized in order to filtrate the control, amongst the crowd, to only the person closer and centered to the system.

In Figure 4.19 you can see the setup and some interactions with the system. The closet was provided by CITAR. This stored inside a MacMini running the ZtS and had a custom fit opening for a Microsoft Kinect. Outside the visuals were displayed

[4] http://www.festivalin.pt/

in a 32 inch LCD and the sound was provided by a stereo setup mounted by the FestivalIn organization.

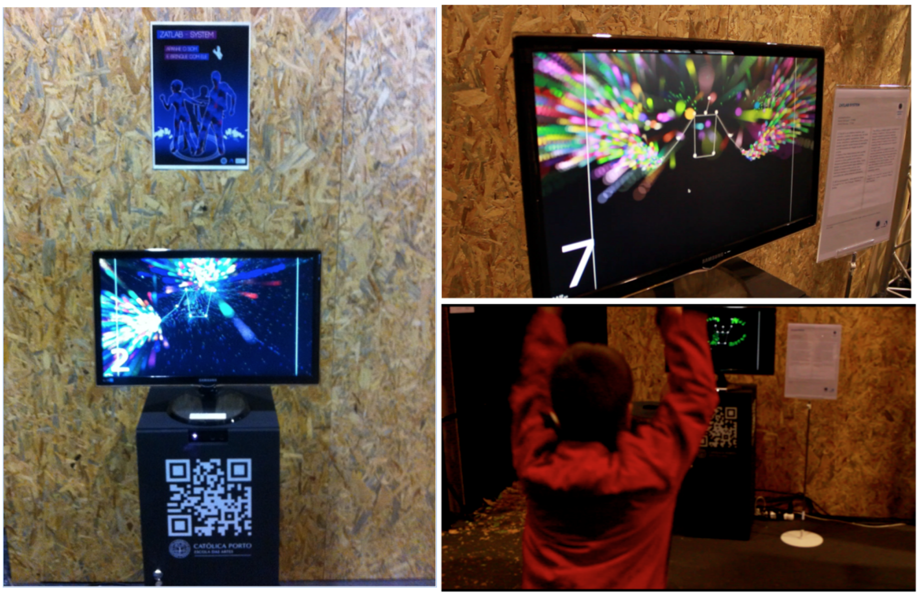


Figure 4.24: The setup used for FestivalIn. On the left, the cabinet provided by CITAR, you can notice the Kinect bellow the LCD TV. On the right top the visuals when someone interacted and left bottom a kid playing with the system.

4.6.2.2 Evaluation

The response to the system was very good, in particular amongst the children. All day long there was someone playing with it. The fact that the people were detected immediately either if they were just passing by or really wanted to interact was a key factor to the system popularity. The persons saw their skeleton mirrored on the screen and wave at it, therefore triggering sounds and building up the users curiosity. Soon enough they understand the system response to their gestures and were engaged, interacting and creating musical expressions.

4.7 Summary

This chapter presented the experimental validation of the framework proposed in Chapter 3 and its main applications.

First it was explained the research methodology, followed by Nielsen heuristics to determine the framework acceptability. Next the validation approaches were described.

The qualitative validation was done by means of a public experiment and questionnaires. These revealed the framework usability and acceptability results. The conclusions achieved are that the system presents a solid basis and already is considered a good tool for artistic applications. Nevertheless, the interface should be further developed to make the framework more straightforward and easy to interact with.

The quantitative validation used the data gathered on the public experiment to test the recognition algorithms performance. The DTW recognition method achieved an average recognition rate of 78% in general and, depending on the gesture, it can present recognition rates till 96%. The HMM recognition presented better results, ranging from 86% till 100% depending on gesture and with an overall average recognition rate of 94%.

The chapter finished with a description of the ZtS framework in artistic applications. Its use on the Miso Music Opera “A Laugh to Cry” and its installation as an interactive application on the FestivalIn, in Lisbon.

Chapter 5 will provide some details regarding the software implementation of the framework.

Chapter 5

Software Implementation

"I think, fundamentally, open source does tend to be more stable software. It is the right way to do things."

Linus Torvalds

5.1 Introduction

The research on the topic of gesture recognition poses challenging demands on the development of software modules and tools so that any proposed hypothesis and algorithms can be objectively implemented and evaluated. The prototypes developed are also important to establish a starting point for future research, therefore enabling the further improvement and validation of the algorithms implemented.

Inevitably, during the development of this thesis a great deal of work has been invested into software development, mainly focused on the gesture recognition framework proposed and discussed in the previous chapters.

Therefore, this chapter presents the main design requirements and the implementation strategies taken towards the development of a software framework for the analysis of gestures. It also describes in detail the major software contributions. Despite these software descriptions, a thorough discussion about Software Engineering is out of the scope of this thesis, and the reader will be redirected to the specialized literature whenever appropriate.

Having the mindset of a computer programmer, the tasks are distributed among different blocks and then coordinated by a core function to form an effective and coherent application.

5.2 Design Requirements

The designing and implementation of a multimedia framework, efficient enough to recognize gestures in real-time, poses challenging demands.

First, considering the huge amounts of multimedia data available today, the application should be able to process content in a timely manner, otherwise it may compromise its usefulness.

Second, when creating a flexible software framework, one should be concerned with attributes such as code reusability and modularity. This will allow the following developers the ability to perform rapid-prototyping of complex algorithmic processes or even develop fully fledged applications without the need to repeat from scratch the cycle of coding, testing, debugging and validating, being able instead to focus on the development of new and highly specialized software modules.

Third, if the framework is to remain useful with the passing of time, an effort should be put into designing it so that it makes the best use of the increasingly available computational power, allowing it to scale as well as possible with increasing amounts of data to process.

Furthermore, an important feature is it should be portable. This means that it should be possible to use the same software code in the nowadays, increasingly distinct and fast growing developing computational systems. Hence, the framework should allow execution in different:

- modes, depending not only on the user is expertise, but also the goal he wants to achieve with its use (e.g. as a console service running in an artistic performance, or as a GUI application providing easy interaction in a public display);
- architectures (e.g. x86, Power-PC, SPARC);
- platforms (e.g. UNIX/Linux, MacOSX, Microsoft Windows).

However, this requires the use of well-established coding standards (e.g. ANSI C, portable C++, JAVA) and discourages the use of architecture and platform specific features and optimisations (e.g. assembly coding, platform specific libraries or technologies). Although there are external libraries that are multi-platform and could be used as processing algorithms (e.g. HMMToolkit[1]), GUI (e.g. QT[2], GTK[3]), or even numeric routines (e.g. BLAS[4]), among other. These would require dependencies and installations that would create portability issues. Thus, its important that they are kept as limited as possible. In the case of this framework, no external libraries (to the platform chosen) were used. This made the implementation of the algorithms more difficult, but it payed of in terms of knowledge. Furthermore, the framework can be executed as a standalone application without any external dependencies, besides the chosen MoCap hardware drivers.

Interoperability with existing software packages or applications (e.g. MATLAB[5],

[1] http://htk.eng.cam.ac.uk

[2] http://qt-project.org/

[3] http://www.gtk.org

[4] http://www.netlib.org/blas/index.html

[5] http://www.mathworks.com/products/matlab/

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

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

5.3 Implementation Strategies

Different strategies may be adopted when facing the task of implementing software modules that are just a small part of a larger system. Usually there are two approaches to consider:

1. **Commercially available platforms** - Have the downside of, as the majority

[6] http://www.cs.waikato.ac.nz/ml/index.html

[7] http://www.python.org

of the available commercial software packages, being released as closed source. Hence, it may not be possible to develop a tailored framework that will fulfill entirely the needs and specificities of the project. And certainly, at the light of the discussion in the previous Section, it wont be easy to find a commercial software package that perfectly fits all the requirements posed by a specific research task. Equally important, the cost of purchasing and maintaining a commercial software license is often high, turning such solutions unsustainable for low-budget or unfunded projects.

On the other hand, traditionally, these software packages already provide some ready-to-use building blocks and routines for commonly known and used tasks (which may range from basic mathematical or numerical routines to advanced signal processing algorithms), and allow in most of the cases coding new user-defined ones. (e.g. MATLAB, Simulink[8], LabView[9], MAX/MSP).

2. **Free and open-source software (FOSS) projects** - This approach can be potentially complex and challenging (both in time and in expertise). Usually requires writing all the software from scratch and finding some way to integrate the different software modules into a working system. It demands more time and effort to implement, test and evaluate the system and all the processing modules. Nevertheless, it has the advantage of allowing the definition and fine-tuning of the software architecture taking into consideration any specific requirements, staying in complete control of the way the software is designed and subsequently implemented. The learning experience gained from undertaking such an endeavor would also be a valuable added bonus.

In fact, the second approach may turn out to be productive far beyond the scope of its development. FOSS projects, fruit of individual or team efforts have been released to the community and were successfully embraced by the scientific research

[8] http://www.mathworks.com/products/simulink/

[9] http://www.ni.com/labview

community. Examples of such, include Processing [10], Cinder [11], MARSYAS[12], PureData[13], and openFrameworks.

These projects are an interesting opportunity for carrying on existing work and contributing back with any original or relevant achievements (be it software, algorithms or improved results). This usually ends up originating a positive feedback loop of contributions of software modules, tools, collections of routines and even complete frameworks around a field of study. An entire community may end up using the software and putting it to the test, eventually reporting any found deficiencies or limitations, posting new feature requests, or even becoming an active collaborator of the project. Specially relevant when used for research, FOSS allows using code as a means of communication, where publications can not possibly describe all the nuances and details of how an algorithm is implemented. At the same time, replication of experiments is essential for progress in research especially with new emerging technologies and controllers in HCI. For complex systems and algorithms it is almost impossible to know if a reimplementation is correct and therefore the ability to run the original code is crucial. Finally, FOSS solutions have a lower cost when compared to proprietary or commercial software, a particularly important point given that traditionally researchers have limited financial resources.

5.4 Developing with openFrameworks

Taking into account the personal past experience of working both commercial software packages (i.e. MATLAB and MAX/MSP) (Baltazar, 2009) and FOSS frameworks (i.e. openFrameworks) (Baltazar et al., 2010) and considering the previous discussions about design requirements and implementation strategies, the decision was made to look for a suitable FOSS framework.

[10] http://www.processing.org/

[11] http://libcinder.org/

[12] http://marsyas.sourceforge.net

[13] http://crca.ucsd.edu/˜msp/software.html

Among the available options described in the previous section, openFrameworks meets most of the discussed requirements for a research software framework and the previous acquaintance with programming in C++ implied a smoother learning curve and a good prospect of easily reusing some of the functions provided by the platform. All things considered, openFrameworks was the software framework of choice.

A key point of openFrameworks is its division in core functions (i.e. math, text and other generic functions) and “addons”. The addons are software code modules developed by the community to tackle some specific problem. These addons can be integrated in any openFramewroks application, hence allowing to develop some rather complex applications, such as the ZtS framework. The addons are published in the openFrameworks website, thus becoming available for download and reuse by anyone.

Given the free and open source nature of openFrameworks, their openness and availability for integrating new ideas into the framework gave space for most of the software development contributions that will be described in the following Sections.

5.4.1 Contributions to the openFrameworks Platform

openFrameworks provides a general, extensible and flexible framework that enables the easy and efficient combination of a variety of existing addons as components which ultimately allow to implement efficient, robust algorithms and also create complex applications.

As a result, openFrameworks provided a solid software base upon the gesture recognition framework proposed in this thesis was implemented. However, given the challenging and specific requirements posed by the proposed ZtS framework, its implementation in openFrameworks asked for the substantial development of new processing and composite modules (addons), such as the HMM module or the DTW module. As explained previously, these addons can be used by anyone in other projects.

The following list summarizes the main software implementations done in the scope of this thesis. Thus, resulting in contributions to the openFrameworks software framework, in the form of addons:

- **ofxVisualKinect** - This addon receives the data information from the Microsoft Kinect sensor (explained in Section 2.4.3.1). It transposes the data of each skeletal joint to 3D coordinates in the openFrameworks canvas and displays it. Can be very useful as a point-start to developing applications with this hardware. It can easily be adapted to use with other hardware, such as Vicon. It is preset to work with Kinect since it is one of the most popular MoCap solutions.

- **ofxDTW** - This addon implements the DTW algorithm, as explained in Section 3.5.1 and the practical implementation explained next, in Section 5.5.2. It allows to measure the similarity of two signals (feature vectors) by executing the DTW “cost” matrix.

- **ofxHMM** - This addon implements the HMM algorithms, thus enabling to perform all the general HMM associated functions. These include: train several HMM, output the most likely sequences of hidden states for each model, test if a sequence of observations belongs to a trained model. The practical implementation is described next, in Section 5.5.3.

- **ofxViconOSC** - This addon implements the communication of the Vicon Blade software through OSC to any IP in a known network. The implementation and final application a described in Section 5.5.4

- **ofxZtS** - This addon is the entire ZtS framework. It is a composite using the previous addons with some more code modules to integrate everything and extract the correct features to feed the machine learning addons. Also explained in detail in the next Section.

This complex endeavor demanded a long time of studying and programming. At the end, this effort allowed the implementation of the gesture recognition framework proposed in Chapter 3 and to subsequently conduct the evaluation experiments and concrete applications presented in Chapter 4.

Furthermore, because it is designed as a flexible, modular and efficient framework, this allows the current framework to be easily expanded to include future ideas and algorithms, such as the ones anticipated during the course of this work, and compiled in Chapter 6.

The following sections will present and describe the main aspects of the implementation of some of the main algorithms of the framework proposed in this thesis. It is assumed that the reader is sufficiently familiarized with computer programming (in particular Object Oriented Programming).

5.5 Implementing the ofxZtS Framework

Figure 5.1 shows the overall architecture of the implemented system proposed in this thesis (see Chapter 3). Next is a brief explanation on how the system works and detailed descriptions of each block will be made in the following sections.

The human movements are acquired in real-time, either by the Kinect or using the Vicon MoCap system. The data gathered by these systems is then transmitted to the ***ofxVisualKinect Module*** through OSC. In the case of the Vicon, the ***ofxViconOSC Module*** had to be implemented to stream the data in real-time from the proprietary Vicon Software through OSC as well.

The ***ofxVisualKinect Module*** interprets the data stream as a set of human joint coordinates (e.g. $JointLeftHand x, y, z$), sends them to the ***Visual Representation Module*** and also transmits the data to the ***ZatLab System Core*** block. In the

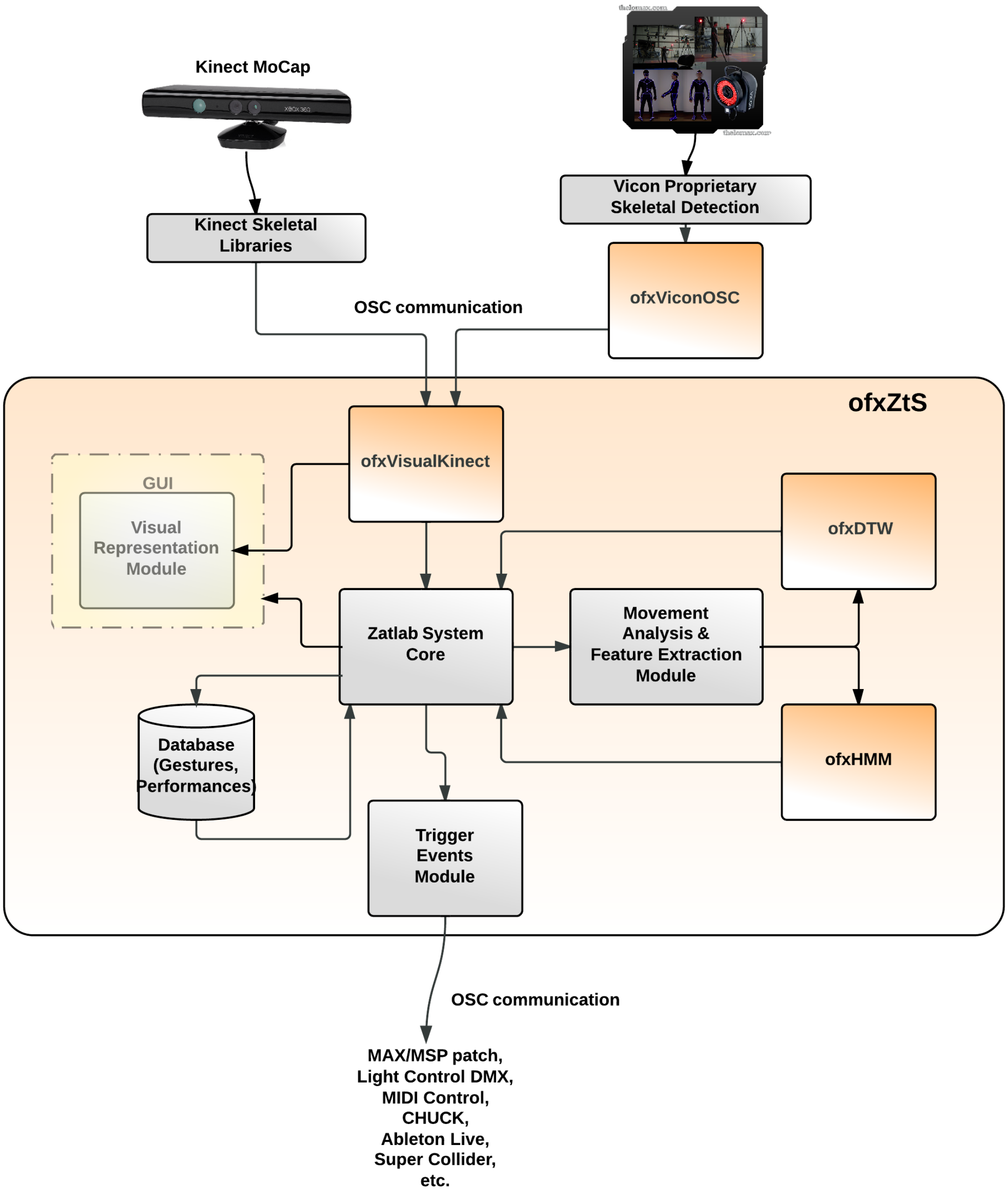

Kinect MoCap
Kinect Skeletal Libraries
Vicon Proprietary Skeletal Detection
ofxViconOSC
OSC communication
ofxZtS
ofxVisualKinect
GUI
Visual Representation Module
ofxDTW
Zatlab System Core
Movement Analysis & Feature Extraction Module
Database (Gestures, Performances)
ofxHMM
Trigger Events Module
OSC communication
MAX/MSP patch, Light Control DMX, MIDI Control, CHUCK, Ableton Live, Super Collider, etc.

Figure 5.1: The ZtS architectural diagram. The blocks starting by *"ofx"* are the modules published to openFrameworks as addons. Besides being integrated in the ***ofxZtS***, they can be reused independently in other openFrameworks projects.

Visual Representation Module these are displayed (as a virtual human skeleton) integrated in the GUI.

In the ***ZatLab System Core*** this data is processed and sent to the ***Movement Analysis and Feature Extraction Module***. This module retrieves the essential features about the movements being performed (accelerations, velocities, movement orientation, etc). If the system is already operating in Gesture Recognition Mode (explained in Section 3.7), the features extracted will be transmitted for further processing to the ***ofxDTW***, the ***ofxHMM*** or both. The ***ZatLab System Core*** is also responsible for the GUI, and the respective toggles and commands the user might change. The ***ZatLab System Core*** processes all the user choices including the communication to the ***Database*** for recording or loading various types of files (described in Section 3.4.2).

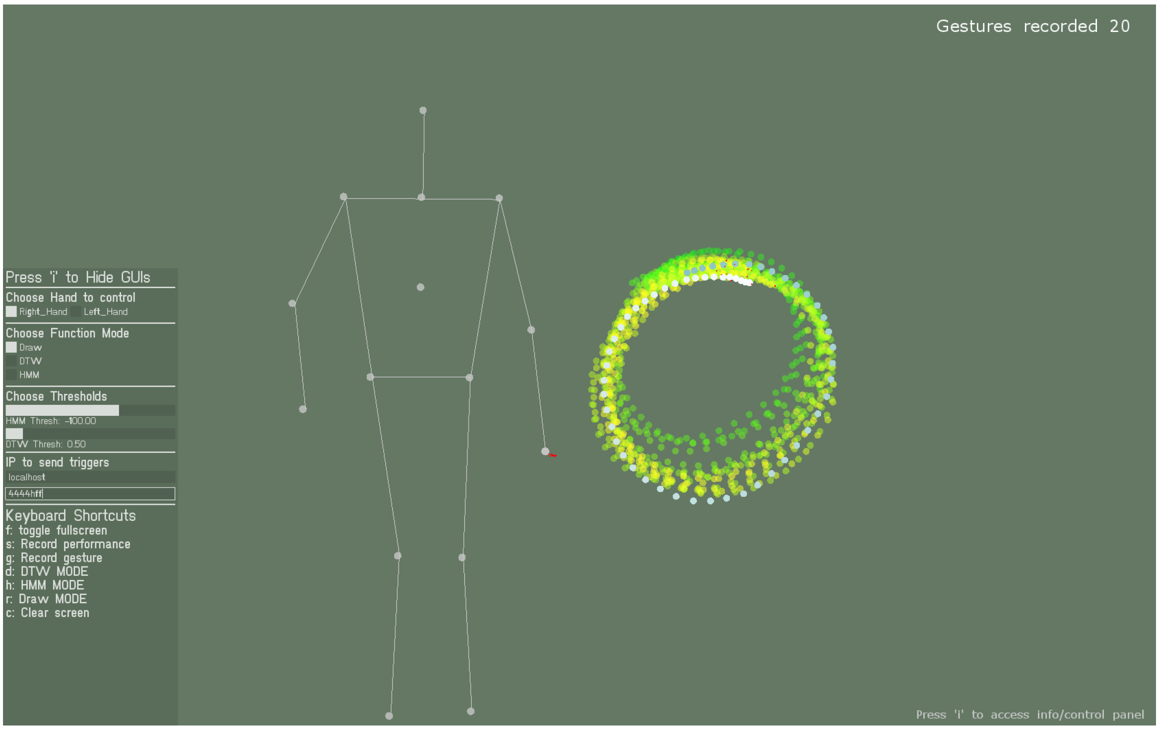
The ***ofxDTW*** and the ***ofxHMM*** use the movement analysis resultant features for the train, analysis and recognition of gestures. If a gesture is recognized, its index is returned to the ***ZatLab System Core***, that again, accordingly to the user choices will act on it. These actions include recording it, display it, or signal it to ***Trigger Output Module***.

The ***Trigger Output Module*** activates a discrete or continuous trigger, once a gesture or special feature is received. This trigger can be used to control an event in any program compliant with OSC, such as Chuck, MAX/MSP, PD, or any other exemplified in the Figure 5.1.

With this global perspective on how the system works, the next sections will detail the main contributions and implementations of the ZtS framework modules.

5.5.1 The Graphical User Interface

The GUI is one of the key points of the ZtS. This section details the main functions the users can control using it.



Gestures recorded 20

Press 'i' to Hide GUIs

Choose Hand to control

■ Right_Hand □ Left_Hand

Choose Function Mode

■ Draw

□ DTW

□ HMM

Choose Thresholds

HMM Thresh: -100.00

DTW Thresh: 0.50

IP to send triggers

localhost

4444hff

Keyboard Shortcuts
f: toggle fullscreen
s: Record performance
g: Record gesture
d: DTW MODE
h: HMM MODE
r: Draw MODE
c: Clear screen

Press 'i' to access info/control panel

Figure 5.2: The ZtS graphical user interface.

Figure 5.2 displays the interface. In this particular example the user was operating in “Draw” mode and just finished drawing 20 circles. On the top right corner is the indication of the number of gestures recorded and on the left of the screen is the control panel. With it, the user can control:

- the hand to use in the gesture training or recognition. Important to realize the decision was to include here only both hands, since it is the more common to use, but the framework is ready to work with any of the body joints;
- the operation mode - Draw, DTW Recognition, HMM Recognition (the recognition algorithms can be used simultaneously);
- the thresholds of the recognition algorithms;
- the IP and Port addresses to where the user wants to send the triggers.

Besides that, the control panel also provides information concerning the keyboard shortcuts.

The work of the interface is straightforward. Once the user approaches the application, he is detected and his skeleton represented on the canvas. He can interact with the panel and choose its options using the computer mouse or the keyboard shortcuts. Once he draws a gesture he can activate the recognition methods. Having at least one recognition method activated, once the training of the respective algorithm is complete, this immediately starts the recognition process (for the DTW is only required one sample and for the HMM at least 10 samples).

Next are described the recognition addons.

5.5.2 The *ofxDTW*

The DTW algorithm allows the comparison of two signals or the detection of a pattern in a larger stream of data (Ten Holt et al., 2007). The algorithm calculates the distance between each possible pair of points out of two signals in terms of their associated feature values. In this case, this is calculated using the Euclidean distance. It builds a cumulative distance matrix with the distances measured and finds the least expensive path through this matrix, the optimal warping path. Specifically the path represents the best synchronization of the two signals, this is, the minimum feature distance between their synchronized points. Therefore, the DTW is useful to compare pairs of data vectors, in this particular case, vectors of movement features data. This makes it very immediate and simple to use. The advantage of this against the HMM (explained in next section) is this immediate using without the need for several samples of training to each gesture.

To explain this implementation, first it is important to realize how to proceed in order to recognize a gesture. Regarding a case-study example of an user using his right hand to record and test gesture recognition. This relies in two main procedures:

1. **Recording Gestures** - When recording a gesture, a **`vector_of_features`** is incremented, at each frame, with several feature values, for instance (x, y, z, ϕ) where x, y, z are the coordinates and ϕ is the orientation angle of the hand movement. So when the user decides to record a gesture he will really be recording the sequence of movement features he is performing. The user can record as many gestures he wants, thus creating a database of several of these **`vector_of_features`** stored in a **`vector_of_gestures`**. This database will be the reference to which the forthcoming "test" gestures will be compared. Refer to Figure 5.3 to a graphical explanation of the recording procedure.

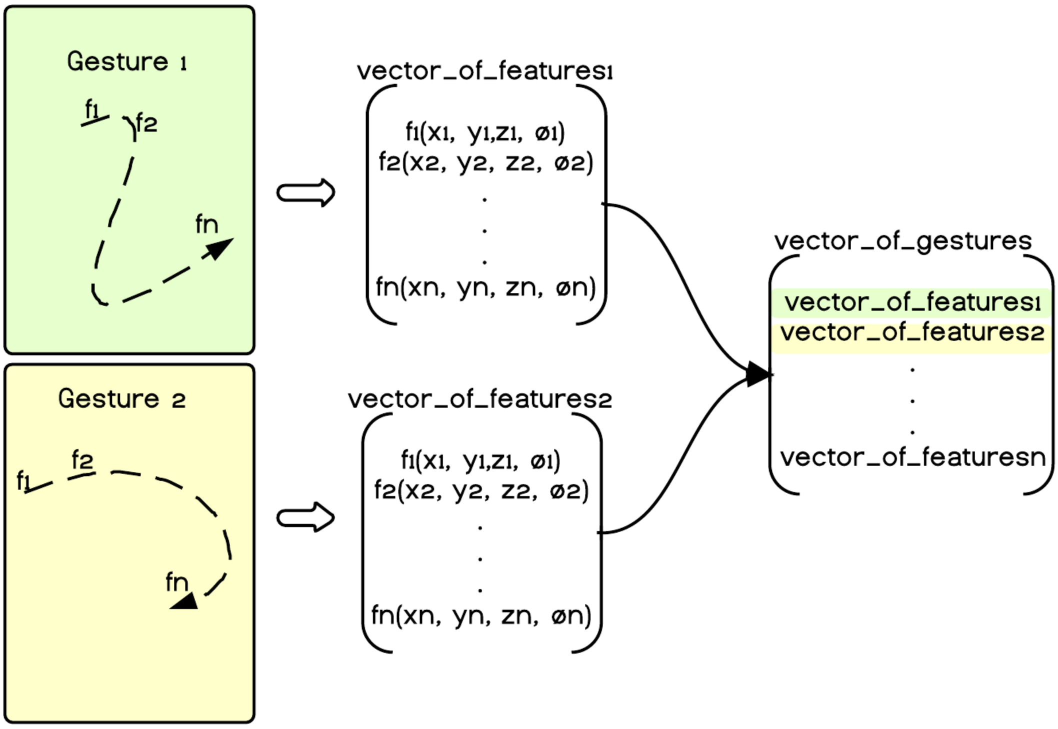


Figure 5.3: The sequence of gesture features are accumulated in a vector. When the user records the gesture, this sequence will be stored as a new gesture in the *vector_of_gestures*.

2. **Recognizing** - Having at least one gesture recorded on the database, the system enters in recognition mode. At each frame the **vector_of_test** will be fed with the same features the previous **vector_of_features**. This vector stores the data, keeping thus a real-time array of features (with size N - the double of space the biggest gesture recorded).

   Once it gets N feature samples, the system will cyclically divide the movement input at regular intervals creating several (**vec_of_test**) that will increment (**vector_to_dtw**). The system performs the DTW distance of each one of this **vector_of_test** against each **vector_of_features** stored in the **vector_of_Gestures**. When the DTW distance to one of the gestures recorded is lower then a determined threshold, the input sequence is recognized as a gesture. Refer to Figure 5.4 to a graphical explanation of the procedure.

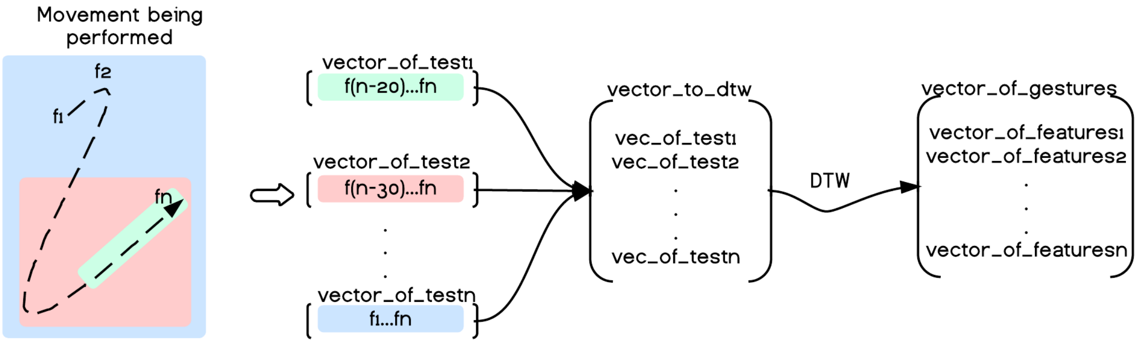


Figure 5.4: A movement is tested through the DTW distance in order to find if it is present in the Gestures Database. Relating to the previous Figure 5.3 when testing the entire movement (in blue) it would result in finding the stored Gesture 1 (**vector_of_features1**). In this case, you can realize the signal being tested in slightly bigger than *Gesture 1*, nevertheless, is the same gesture in shape. Therefore, despite some distance between both signals, the DTW algorithm will detect it as being similar to Gesture 1 (as intended).

The key point of this algorithm is the construction of the DTW cost matrix. This is built by iteratively finding the minimum Euclidean Distance amongst the components of both vector signals, hence finding the optimal warping path, also named minimum warping distance.

Breaking down into a detailed description (refer to Figure 3.6), each component of **`vector_of_test`** will be tested against each component of **`vector_of_features`**. Once the minimum pair-wise distance is found, this distance will be stored in the cost matrix, and proceed to the next component. This cycle repeats until all the components have been analyzed and the cost matrix built. By summing all these minimum distance values along the cost matrix, one will have the shortest warping path, or the minimum distance of the signals. The implementation code was done based on Lemire (Lemire, 2009) approach to DTW algorithm, but with some modifications to work with the openFrameworks methods.

5.5.3 The *ofxHMM*

As explained in Section 3.5.2, HMM allows the modeling of sequential or time-series data through powerful statistical methods (Rabiner, 1989). In fact HMMs have been successfully used in many tasks, such as, speech recognition, protein/DNA sequence analysis and face recognition (Nefian and Hayes III, 1998). It involves elegant and efficient algorithms, such as Baum-Welch, Viterbi and Forward-Backward, for learning, evaluation and decoding.

Although the algorithms are elegant and sophisticated, their implementation is not very straightforward. Consequently, the next paragraphs will explain how these work together in gesture recognition. Specifically the **`HMM class`** was developed with 3 modes of operation: Train, Evaluate, Test. These are called by using the pointer to the class and choosing the operation mode wanted (1-for testing, 2 - for evaluating, 3- for training). This implementation was based in (Liu, 2009) and (Rabiner, 1989).

Again, for a gesture to be recognized, first one will have to “teach” the algorithm how the gesture look like and how it is executed. In the previous DTW approach, one is able to do direct and immediate comparison of signals. In the case of HMM, being a probabilistic model build upon statistics, the “teaching” is not so forthcoming. It will involve the creation of a training set of gestures for each one we wish to detect. Recalling the same case-study proposed before, imagine an user using his right hand to record and test gesture recognition. In order to do so, this module operates in the following fashion.

1. **Record gesture samples** - To train a HMM of a gesture first one needs to create several instances of the same gesture. Thus, using a similar method to the one explained before (Section 5.5.2) one will be recording, at each frame, several feature values of the user movement (kept in **`vector_of_features`**). The user will record several identical samples of the same gesture being each one stored in a **`vector_of_gestures`**.

2. **Create a new HMM** - Having a reasonable amount of examples of the same gesture (defined by the user), when the order to train a new HMM is made, this has to be created and initialized. For each new HMM the user can dynamically choose the number of hidden states (N_{states}). For instance to create a new HMM with a *vector_of_gestures* and N_{states} one would do:

   - `vec_hmm_models.push_back(new HMM(vector_of_gestures,` N_{states}`));`

   This creates a new instance of **`HMM class`** with a new position in the pointer **`vec_hmm_models`** to it. The matrices of this new HMM are initiated following the next rules:

   (a) The initial states probability (matrix N_{states} x 1) is initiated as $1/N_{states}$ to give an equal probability distribution amongst the states.

   (b) Considering the gesture is done in one continuous, fluid movement, the transition probability between states should have more weight between the adjacent ones, thus the state transition probability matrix (a_{ij} of size N_{states} X N_{states}) is initiated as exemplified on the following Table 5.1.

Table 5.1: The state transition probability matrix initialization example. The probability is divided amongst adjacent states. The N-ish state is connected to the first, closing thus the probabilities loop.

State	**0**	**1**	**2**	**N**
0	0,5	0,5	0	0
1	0	0,5	0,5	0
2	0	0	0,5	0,5
N	0,5	0	0	0,5

(a) At last, the state output matrix ($N_{observations}$ X N_{states}), that allows to relate the observed output data ($N_{observations}$) to the state transition, is initiated by distributing equally the probabilities of the output: $1/N_{observations}$.

3. **Train a HMM** - Having the new HMM created, the system will train it using the samples provided. To do so, the **`vector_of_gestures`** will be passed to the Baum-Welch algorithm (formally explained in Section 3.5.2 and computer implementation explained next) by calling the HMM class with the respective operation mode (mode 3, for training):

   - `vec_hmm_models[last]->RunHMM(3, vector_of_gestures);`

   The train routine will breakdown the **`vector_of_gestures`** in its constituents (**`vector_of_features`**). These features are the observed data and with it the algorithm performs a statistical evaluation of the data sequence that will lead to the update of the emission and transition probabilities matrices, modeling thus the hidden states for the gesture performed.

Computing the Baum-Welch

The algorithm takes sequences of observations as input and estimates the new values of transition matrix (a_{ij}) and emission matrix ($b_i(v_k)$) that maximize the probability for the given observations. It runs iterations over the input data and terminate until convergence or certain threshold condition is met, for instance: number of iterations, difference in parameter changes. The algorithm takes two passes over the data. In the first pass, it uses forward algorithm to construct α probabilities (the pseudo-code for this algorithm is explained in the following section Computing the Likelihood). Besides the α probabilities, in the second pass the algorithm runs a similar backward algorithm to construct β probabilities. The backward probability $\beta(t,i)$ is the probability of seeing observation from o_{t+1} to the end, given that we are in state j at time t. Based on the α and β probabilities, one can compute the expected number (counts) of transitions ($\xi(i,j)$) from state i to state j at a given observation t ($\gamma(t,i)$) as described by Equations A.11 and A.12 in Appendix A. Part of the pseudo-code for Baum-Welch algorithm is presented in Listing 5.1. The α probabilities are updated after calling the forward function at line 2. The remaining code computes $\xi(i,j)$ and $\gamma_(t,i)$ counts.

With $\xi(i,j)$ and $\gamma_(t,i)$ computed, the a_{ij} and $b_i(v_k)$ matrices are updated using the Equations A.14 and A.15, described in Appendix A.

```
1 initialize all cells of $\alpha$, $\beta$, $\gamma$, $\xi$ to $0$
2 calculate $likelihood \leftarrow Forward(o)$
3 $\beta(o_T,1) = 1$ // base case $t = T$, end of sequence
4 for $t = o_T$ to $o_1$ //cycle to compute the Backward algorithm
5   for $i = 1$ to $N$
6     do $\gamma(t,i) = \gamma(t,i) + (\alpha((t,i) \cdot \beta(t,i)/likelihood)))$
7          for $j = 1$ to $N$
8              do $\beta(t,i) = \beta(t,i) + \beta(t+1,i)\alpha_{ji}b_{it}$
9                 $\xi(j,i) = \xi(j,i) + (\alpha(t,j)\beta(t+1,i)\alpha_{ji}b_{it}/likelihood)$
```

Listing 5.1: The pseudo-code for the Baum-Welch algorithm.

4. **Verify the Model** - Having the model constructed with its respective emission and transition matrices one can verify if the training was done properly. This is accomplished using the Viterbi algorithm (formally described in Section 3.5.2 and computer implementation explained next). This algorithm will provide the sequence of hidden states in respect to the HMM built:

   - `vec_hmm_models[last]->RunHMM(2, 0);`

   Computing the Viterbi

   The Viterbi algorithm finds the most likely path of states that generate the observations. Instead of summing over all α probabilities (like Baum-Welch algorithm does), Viterbi algorithm finds the maximum one and keeps a pointer to trace the state that leads to the maximum probability. The pseudo-code for Viterbi algorithm is given in Listing 5.2. The input to the algorithm is a sequence of observations and output is a sequence of the most likely states that generate the observation.

```
initialize all cells of $\alpha$ to $0$
$\alpha(o_1, s) = 1$ //base case t=1, there are no preceding states
for $t = o_2$ to $o_T$ //cycle to compute the Viterbi algorithm
  for $i = 1$ to $N$
    for $j = 1$ to $N$
      if $\alpha(t-1, j)a_{ij}b_{it} > \alpha Max(t, i)$
        then $\alpha Max(t, i) = \alpha(t-1, j)a_{ij}b_{it}$
             $MaxPointer(t, i)$ = $j$
Seq_of_states= sequence(MaxPointer)
return Seq_of_states
```

Listing 5.2: The pseudo-code for the Viterbi algorithm.

5. **Recognizing** - Once having a trained HMM the system can enter in test mode. Again, like in the DTW case (Section 5.5.2) the **`vector_of_test`** will be fed with the same features of the previous samples used to train the

model. In this case the vector will be continuously tested against the trained HMM:

- `vec_hmm_models[last]->RunHMM(1, vector_of_test);`

If there are more than one HMM trained, the **vector_of_test** is iteratively tested against all the models (M) of the **vec_hmm_models[**M**]**. The highest likelihood HMM is returned by the Forward Algorithm (formally explained in Annex A.2 and computer implementation next).

This test is done using equation A.3 in regard to each trained model emission and transition probabilities matrices. If the observed test sequence matches the probabilities previously calculated for the model matrices, the likelihood of that sequence will be maximized. Therefore, if that returned likelihood is high enough to surpass a user-defined threshold, the gesture is recognized as belonging to that respective model.

Computing the Likelihood

To compute the likelihood, the Forward algorithm computes the α for the sequence of O observations and N hidden states. This can be viewed as a matrix, where each cell $\alpha(o_t, i)$ is the probability of being in state i while seeing the observations until t. An overview of Forward algorithm is shown in the pseudo-code below (Listing 5.3). The input to the algorithm is a sequence of observations O. The output is the likelihood probability for the observation. The algorithm makes the assumption the first observation in sequence is the start state, and the last observation is the end state.

```
initialize all cells of $\alpha$ to $0$
$\alpha(o_1, s) = 1$ //base case t=1, there are no preceding states
for $t = o_2$ to $o_T$ //cycle to compute the Forward algorithm
  for $i = 1$ to $N$
    for $j = 1$ to $N$
      do $\alpha(t,i) = \alpha(t,i) + \alpha(t-1,j)a_{ij}b_{it}$
```

```
$likelihood = \alpha(o_T, N)$
return $likelihood$
```

Listing 5.3: The pseudo-code for the Forward algorithm.

5.5.4 The *ofxViconOSC*

Although this module is not really integrated as part of the *ofxZtS*, since was developed as an additional feature, it is *per se* a fundamental contribution, not only for the openFrameworks community, but to the researchers and laboratories operating with the Vicon MoCap. Its implementation took a long time for developing due to the software restrictions of the Vicon software, nevertheless, was worth it.

With it, it is possible to do real-time transmission of any scene being recorded in a MOCAP laboratory to other programs besides those officially compliant with Vicon. The Vicon data is encapsulated in an OSC message and can be transmitted to any IP address determined by the user. The module is developed to work as a standalone application and has a GUI where the user can specify the IP and communication port number he wishes to send the data to.

The only software that was found to work like ofxViconOSC was QVICON2OSC[14], but this is already obsolete since Vicon introduced the new software capture Blade 1.7 in 2010 (substituting the previous Vicon Targus).

Nowadays Vicon Blade is on version 2.1, the ofxViconOSC works with every version of it. The implementation of this software required the learning and use of the Vicon DataStream Server drivers and their integration as an openFrameworks addon.

The Vicon DataStream Server can operate in three different modes. Each mode has a different impact on the Client, Server, and network resources used:

[14] http://sonenvir.at/downloads/qvicon2osc/

1. In **ServerPush** mode, the Server pushes every new frame of data over the network to the Client. The Server will try not to drop any frames. This results in the lowest latency one can achieve. If the Client is unable to read data at the rate it is being sent, then it is buffered, firstly in the Client, then on the TCP/IP connection, and then at the Server. Once all buffers have filled up then frames may be dropped at the Server and the performance of the Server may be affected.

2. In **ClientPull** mode, the Client waits for a call to `GetFrame()`, and then request the latest frame of data from the Server. This increases latency, because one needs to send a request over the network to the Server, the Server has to prepare the frame of data for the Client, and then it needs to send the data back over the network. Network bandwidth is kept to a minimum, because the Server only sends what you need. It is very unlikely to fill up our buffers, and Server performance is unlikely to be affected.

3. **ClientPullPreFetch** is an enhancement to **ClientPull** mode. A thread in the SDK continuously and preemptively does a **ClientPull** on our behalf, storing the latest requested frame in memory. When next calling `GetFrame()`, the SDK returns the last requested frame which had cached in memory. As with normal **ClientPull**, buffers are unlikely to fill up, Server performance is unlikely to be affected. Latency is slightly reduced, but network traffic may increase if one request frames on behalf of the Client, which are never used.

Since one wants the least latency possible, the **ServerPush** mode is the one chosen to be implemented.

Another characteristic of the Vicon DataStream Server is that one can request what data to transmit by implementing the following functions:

- **EnableSegmentData** - Enable kinematic segment (bones connecting markers) data transmission.

- **`EnableMarkerData`** - Enable labeled reconstructed marker data transmission.
- **`EnableUnlabeledMarkerData`** - Enable unlabeled reconstructed marker data transmission.
- **`EnableDeviceData`** - Enable ForcePlate, Electromyography (EMG), and other devices complaint with Vicon MoCap data transmission.

Currently the ofxViconOSC only transmits the labeled Marker Data, this was the most relevant for the present work being developed. Nevertheless, the implementations of the other modes of transmission can be easily accomplished in future works.

5.6 Summary

Following the proposal of a gesture recognition framework proposed in Chapter 3, this chapter discussed some of the requirements, choices and the major contributions towards the development of an open source software platform for the computational analysis of gestures. Some implementation details about the main building blocks of the framework proposed in this work were described, where the efficiency, flexibility and code reusability aspects taken into consideration during the software development, were highlighted.

The next chapter presents the conclusions and discusses future research and developments.

Chapter 6

Conclusions

”A conclusion is the place where you got tired of thinking.”

Martin H. Fischer

6.1 Results and Contributions

The goal of this research is to foster the use of gestures, in an artistic context, for the creation of new ways of expression. Consequently, the approach taken envisioned the study of the gesture: its understanding, how to capture it (in a non intrusive way) and how to recognize it (in real-time).

Following this study, one concluded the gesture recognition is a rather simple task for the average person, but its automatically recognition, by a machine, is a much more complex task. Therefore, this dissertation proposes a flexible and extensible computer framework for recognition of gestures in real-time.

Designed to be causal and efficient, the resulting system can be used to capture and recognize human body gestures, in real-time, paving the way to applications such as interactive installations, computer music interaction, performance events controlling, amongst others.

The main advantage of this framework against other works developed in this area is to have a fully functional pipeline of integrated modules, allowing the human movement capture, movement feature extraction, gesture training and its recognition, all in a single application. Consequently, enabling a more straightforward use (specially by the artistic community).

In this dissertation a specific implementation of the framework is also presented, where several assumptions had to be considered due to practical constraints. Nevertheless, the proposed framework was designed to be modular, efficient and flexible enough to be able to utilize different analysis front-ends and to incorporate further methods in a straightforward manner.

The proposed system is based in a relatively cheap MoCap system (Microsoft Kinect) and is developed to work without any third party installations besides the respective capture device drivers. The recognition process is then based in ML algorithms, namely DTW and HMM. The use of both methods is justified by the different training processes and recognition rates achieved.

Although, there is not a system working like this, described in the state of art, the experimental validation shown the methods presented in this dissertation (in particular, the ML algorithms) provide results that compare satisfactorily to other state of the art implementations.

The gestures used for the quantitative evaluation are only a small representative sample of the enormous variety possible of human gestures, nevertheless this experiment can be considered a successful test case, showing the framework is on the right track for the recognition of a broader range of gestures.

The qualitative evaluation of the framework, based in Nielsen heuristics, allowed classifying the framework in respect to its practical and social acceptability. The results obtained suggest it has good overall acceptability and it is intuitive enough for being used amongst the performative arts community.

This thesis also described two artistic applications of the framework. One was an interactive artistic installation and the other was its use in an interactive Opera. These applications sustain the artistic relevance of the framework.

In particular regarding its application in the Opera, one can conclude the framework was successfully applied in performance context, recognizing the performer gestures, in real-time, and triggering events. Being the performers the ultimate users of the framework, one reckons their opinion is very important. Therefore the fact that Miguel Azguime (the Opera performer) considers the use of the framework "*constituted a clear enrichment (to the performance) both in terms of communication and in terms of expression*" leads to the conclusion the main goal one proposed to achieve (using gestures, in an artistic context, for the creation of new ways of expression) was accomplished.

A software implementation of the system described in this thesis was also made available as free and open source software. Together with the belief that this work showed the potential of gesture recognition, it is expected that the software implementation may stimulate further research in this area as it can have significant impact in many HCI applications such as interactive installations, performances and Human-Computer Interaction *per se*.

6.2 Future Work

After a great deal of investment in the area of algorithm development, which has given rise to the framework proposed in Chapter 3 and to the results presented in Chapter 4, there are nevertheless several lines of future work that are now possible to anticipate.

In regard to the present software implementation one of the main improvements that can be accomplished is the further development of the GUI in order to make the framework even more intuitive and easy to work with.

Also, the current version still requires the prior specification of the number states to train each new HMM. This is a limitation of the current implementation, but the framework is flexible enough to include new approaches to an automatic estimation of the number of hidden states for each HMM.

Moreover, the latency measured by either capture systems should be further studied and research methods to overcome it, either by using anticipatory methods (Rett and Dias, 2007) or new MoCap approaches (e.g. Kinect 2[1]).

The proposed framework is able to easily accommodate other movement and gesture related researches, such as Choreology or Labanotation (described in Section 2.3). It would be also interesting to integrate human movement feature analysis methods previously developed (when there were no 3D cameras available). Works like the human movement rhythm determination done by Guedes (Guedes, 2005b) or explore further the emotion contained in the gesture has Camurri intended (Camurri et al., 2004).

The motivation for this research was drawn from the performative art domain. However, it was always kept in mind that the proposed concepts and methods could be used in other domains. Thus, interesting opportunities for future research comes from extending this framework to other domains and requirements. For instance, one has the future goal of applying these methods in benefit of the earing impaired community.

[1] There are not yet papers published on this emerging MoCap technology, nevertheless one can find more information on: `http://www.wired.com/2013/05/xbox-one#kinect` or `http://www.youtube.com/watch?v=Hi5kMNfgDS4`

Annex A

Recognition Algorithms Detailed Description

A.1 Dynamic Time Warping

The alignment path (or warping path, or warping function) of the DTW defines the correspondence of an element $x_i \in X$ to$y_j \in Y$ following the boundary condition which assigns first and last elements of X and Y to each other (Senin, 2008a).

Formally speaking, the alignment path built by DTW is a sequence of points $p = (p_1, p_2, ..., p_k)$ with $p_l = (p_i, p_j) \in [1 : N] \times [1 : M]$ for $l \in [1 : K]$ that must satisfy to the following criteria:

1. Boundary condition: $p_1 = (1, 1)$ and $p_K = (N, M)$. The first and last points of the warping path must be the first and the last points of aligned sequences.
2. Monotonic condition: $n1 \leq n2 \leq ... \leq nK$ and $m1 \leq m2 \leq ... \leq mK$. This condition preserves the order of the points.
3. Step size condition: this criteria limits the warping path from long jumps (shifts in time) while aligning sequences. One can set this condition to allow only

jumps of one unity in time or multiple.

So, the cost function C_p associated with a warping path (of length L) that represents all the pairwise distances of the aforementioned sequence of points p, will be given by equation A.1:

$$C_p(X,Y) = \sum_{L}^{l=1} c(x_{n1}, y_{m1}) \tag{A.1}$$

The warping path has a minimal cost associated with alignment called the optimal warping path. In order to find it, one has to test every possible warping path between X and Y which could be computationally challenging due to the exponential growth of the number of optimal paths as the lengths of X and Y grow linearly. To overcome this challenge, DTW employs the following distance function (equation **??**):

$$DTW(X,Y) = c_{p^*}(X,Y) = min\left\{c_p(X,Y), p \ \in \ P^{N\times M}\right\} \tag{A.2}$$

where $P^{N\times M}$ is the set of all possible warping paths, and then builds the accumulated cost matrix or global cost matrix D defined as follows:

1. First row: $D(1,j) = \sum_{k=1}^{j} \ c(x_1, y_k), j \ \in [1, M]$
2. First column: $D(i,1) = \sum_{k=1}^{i} \ c(x_k, y_1), i \ \in [1, N]$
3. All other elements:

   $D(i,j) = min\left\{D(i-1,j-1), D(i-1,j), D(i,j-1))\right\} + c(x_i, y_j),$

   $i \,\epsilon \left[1, N\right], j \,\epsilon \left[1, M\right]$

A.2 Hidden Markov Model

As described in Chapter 3, an HMM is defined as a quintuple (S, V, Π, A, B) where $S = \{s_1, ..., s_N\}$ is a finite set of N states (Rabiner, 1989); $V = \{v_1, ..., v_M\}$ is a set of M possible symbols in a vocabulary; $\Pi = \{\pi_i\}$ are the initial state probabilities; $A = \{a_{ij}\}$ are the state transition probabilities; $B = \{b_i(v_k)\}$ are the output or emission probabilities.

Therefore, each HMM is modeled and expressed as $\lambda = (\Pi, A, B)$ where the parameters are:

- π_i - the probability that the system starts at state i at the beginning;
- a_{ij} - the probability of going from state i to state j;
- $b_i(v_k)$ - the probability of generating symbol v_k at state i;

So, the probabilities constraints apply:

- $\sum_{i=1}^{N} \pi_i = 1$
- $\sum_{j=1}^{N} a_{ij} = 1$ for $i = 1, 2, ..., N$
- $\sum_{k=1}^{M} b_i(v_k) = 1$ for $i = 1, 2, ..., N$

When working with HMM there are three basic problems to solve:

1. Evaluation: one has to evaluate the probability of an observed sequence of symbols $O = o1, o2, ..., ot$ (where $o_i \, \epsilon \, V$) given a particular HMM, this is $p(O|\lambda)$.
2. Decoding: to find the most likely state transition path associated with an observed sequence. Having a sequence of states $q = q1, q2, ..., qt$ we will want to find the $q* = arg_{max} p(q \wedge O|\lambda)$
3. Training: to adjust all the parameters of our model λ to maximize the probability of generating an observed set of sequences O, this is, to find the $\lambda* = arg_{max} \lambda p(O|\lambda)$

These three problems already have solutions. The first is solved by implementing the Forward-Backward iterative algorithms. The second by using the Viterbi algorithm, and the third by using the Baum-Welch algorithm, which uses the Forward and Backward probabilities calculated previously to update the parameters iteratively.

Forward-Backward Algorithm

The Forward probabilities will allow solving the problem 1 and finding the probability of a sequence of observations to belong to a determined HMM model. The Backward probabilities will allow solving problem 3 along with the Baum-Welch algorithm.

Calculating Forward Probabilities

Having $\alpha_t(i) = p(o_1, ..., o_t \wedge q_t = s_i|\lambda)$ as the probability of observing the symbols $o_1, ..., o_t$ and the system is at a state s_i at time t, given our current HMM λ. The α can be calculated starting with the base case and following the recursive procedure:

1. The base case is when $t = 1$. Thus, as seen previously, the probability that the system start at state i is π_i and the probability of generating a symbol o_k at state i as also been explained as being $bi(o_k)$. Therefore, numerically, $\alpha_1(i) = \pi_i b_i(o_1)$, for any i state.

2. For $1 \leqslant t \leqslant T$, we want to generate the symbol o_{t+1} and arrive to state s_i from any previous s_j with a probability (already known) a_{ij}. Thus we will have to multiply the the probability $bi(o_{t+1})$ by the sum of all the possible intermediate states j. This probability is given by equation A.2:

$$\alpha_{t+1}(i) = b_i(o_{t+1}) \sum_{j=1}^{N} \alpha_t(j) a_{ij} \tag{A.3}$$

Knowing that $\alpha_1(i), ..., \alpha_T(i)$ corresponds to the T observed symbols and that one may end at any of the N states. To determine which of the λ models ascribes the highest probability to a sequence we will need to do equation A.3.

$$p(O|\lambda) = \sum_{i=0}^{N} \alpha_T(i) \tag{A.4}$$

Calculating Backward Probabilities

Defining $\beta_t(i) = p(o_{t+1}, ..., o_T \wedge q_t = s_i|\lambda)$ as the probability of observing the symbols $o_{t+1}, ..., o_T$, given that the state is s_i at time t and knowing the parameters of our model λ. Note how this complements the definition of α. In this case we are going down from T, hence the name backward algorithm. Again the procedure is recursive starting from the base case when $t = T$.

1. When $t = T$ There is no symbol to generate, we reach the end of the sequence and any state s can be a possible ending state. Thus, $\beta_T(i) = 1$.
2. For $1 \leqslant t \leqslant T$, as with the forward calculation, we have to multiply an emission probability, a transition probability, and a rest-of-sequence probability. Hence, obtaining the following equation A.4.

$$\beta_t(i) = \sum_{j=1}^{N} \beta_{t+1}(j) a_{ij} b_j(o_{t+1}) \tag{A.5}$$

Viterbi Algorithm

The algorithm will return the Viterbi Probability - VP, i.e., the best score (highest probability) along a single path, at time t, which accounts for the first t observations and ends in state S_i.

This can be done by induction:

1. At $t = 1$ there are no preceding states, so as what happened in the forward probabilities calculation, $VP_1(i) = \pi_i b_i(o_1)$.

2. For $1 \leqslant t \leqslant T$, the VP will be similar (again) to the forward algorithm, but instead of doing the sum, one will do the "max" probability of being in state i at time t over all state sequences that account for the first t observed symbols, resulting in equation A.5

$$VP_{t+1}(i) = b_i(o_{t+1}) max_{j=1}^{N}[a_{ij} VP_t(j)] \tag{A.6}$$

3. To retrieve the final sequence of maximum likelihood states one will need to keep track of the argument that maximized equation A.5, for each t and j. Thus we will need an auxiliary array $\phi(j)$ that will be actualized using the following equation A.6:

$$\phi(j) = argmax_{j=1}^{N}[a_{ij} VP_t(j)] \tag{A.7}$$

In the end one will get the states probability maximum for VP and the respective argument that maximized the probability. Given by equations A.7 A.8.

$$P = max_{i=1}^{N}[VP_T(i)] \tag{A.8}$$

$$Q_T = argmax_{i=1}^{N}[VP_T(i)] \tag{A.9}$$

Therefore, using both equations A.7 and A.8 only thing left to do is backtracking and constructing the best state sequence transition path, using equation A.9.

$$Q_t^* = \phi_{t+1} Q_{t+1}^* \ , \quad where \; t = T-1, T-2, ...1 \tag{A.10}$$

Baum-Welch Algorithm

The Baum-Welch algorithm allows to solve the fundamental problem of an HMM. This is, to adjust the model parameters in order to maximize the probability of the observation sequence. This is again a maximum likelihood problem. Actually there is no optimal way of estimating the model parameters, given any finite observation sequence as training data. Neither there is a known way to analytically solve for the model, which maximizes the probability of the observation sequence. It is possible, however, to use an iterative procedure (such as Baum-Welch method) to choose $\lambda = (A, B, \pi)$ such that $P(O|\lambda)$ is locally maximized.

The formulas for updating can be expressed in terms of the equations A.2 and A.4 together with the current parameter values. So, defining $\xi_t(i,j)$ as the probability of being in state S_i at time t, and state S_j at time $t+1$, given the model and the observation sequence, one will get equation **??**.

$$\xi_t(i,j) = P(q_t = S_i \wedge q_{t+1} = S_j | O \wedge \lambda). \tag{A.11}$$

This can be decomposed in the probability of:

- $\alpha_t(i)$ - observing the sequence $o_1...o_t$ and ending in state i and
- a_{ij} - the transition from state i to state j and
- $b_i(v_k)$ - the emission of symbol o_t while in state i and
- $\beta_t(i)$ - observing $o_{t+1...T}$, given that $s_t = j$

what leads to the following equation A.10:

$$\xi(i,j) = \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{\sum_{j=0}^{N}\alpha_t(j)\beta_t(j)} \tag{A.12}$$

And defining $\gamma_t(i)$ as in equation A.11:

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=0}^{N}\alpha_t(j)\beta_t(j)} \tag{A.13}$$

will allow to simplify $\xi(i,j)$ to equation A.12:

$$\xi(i,j) = \frac{\gamma_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{\beta_t(j)} \tag{A.14}$$

Therefore, allowing to finally get the updating formulas for all the parameters at each iteration (equations A.13,A.14, A.15:

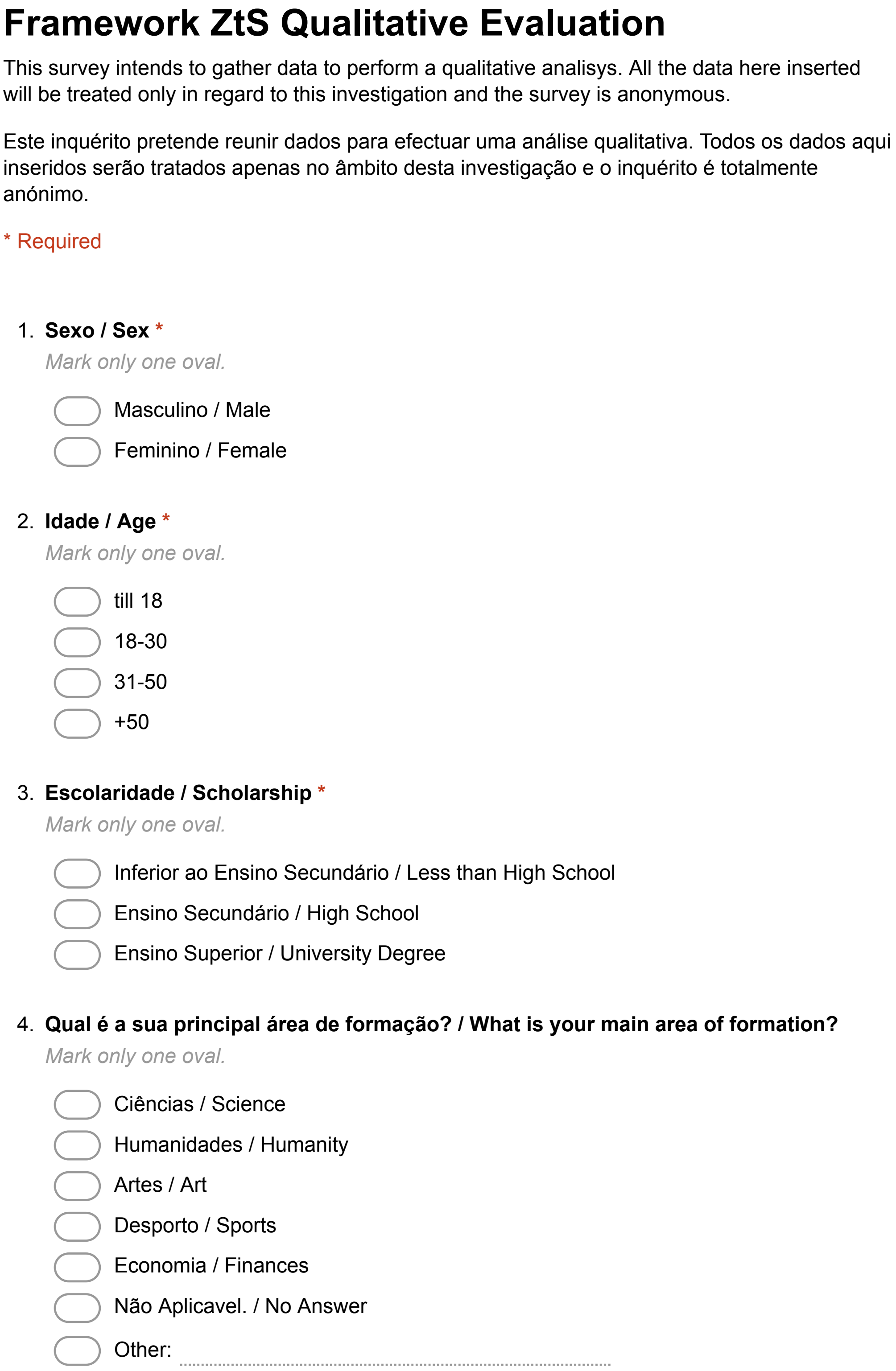
$$\pi_i^{'} = \gamma_1(i) \tag{A.15}$$

$$a_{ij}^{'} = \frac{\sum_{t=1}^{T-1}\xi_t(i,j)}{\sum_{j=1}^{N}\sum_{t=1}^{T-1}\xi_t(i,j)} \tag{A.16}$$

$$b_i(v_k)^{'} = \frac{\sum_{t=1,o_t=v_k}^{T}\gamma_t(i)}{\sum_{t=1}^{T}\gamma_t(i)} \tag{A.17}$$

Appendix A

Questionnaire



Framework ZtS Qualitative Evaluation

This survey intends to gather data to perform a qualitative analisys. All the data here inserted will be treated only in regard to this investigation and the survey is anonymous.

Este inquérito pretende reunir dados para efectuar uma análise qualitativa. Todos os dados aqui inseridos serão tratados apenas no âmbito desta investigação e o inquérito é totalmente anónimo.

* Required

1. **Sexo / Sex** *

   Mark only one oval.

   ○ Masculino / Male
   ○ Feminino / Female

2. **Idade / Age** *

   Mark only one oval.

   ○ till 18
   ○ 18-30
   ○ 31-50
   ○ +50

3. **Escolaridade / Scholarship** *

   Mark only one oval.

   ○ Inferior ao Ensino Secundário / Less than High School
   ○ Ensino Secundário / High School
   ○ Ensino Superior / University Degree

4. **Qual é a sua principal área de formação? / What is your main area of formation?**

   Mark only one oval.

   ○ Ciências / Science
   ○ Humanidades / Humanity
   ○ Artes / Art
   ○ Desporto / Sports
   ○ Economia / Finances
   ○ Não Aplicavel. / No Answer
   ○ Other: ....

5. **Comentários / Comments**

....

....

....

....

....

Facilidade / Ease of Learning

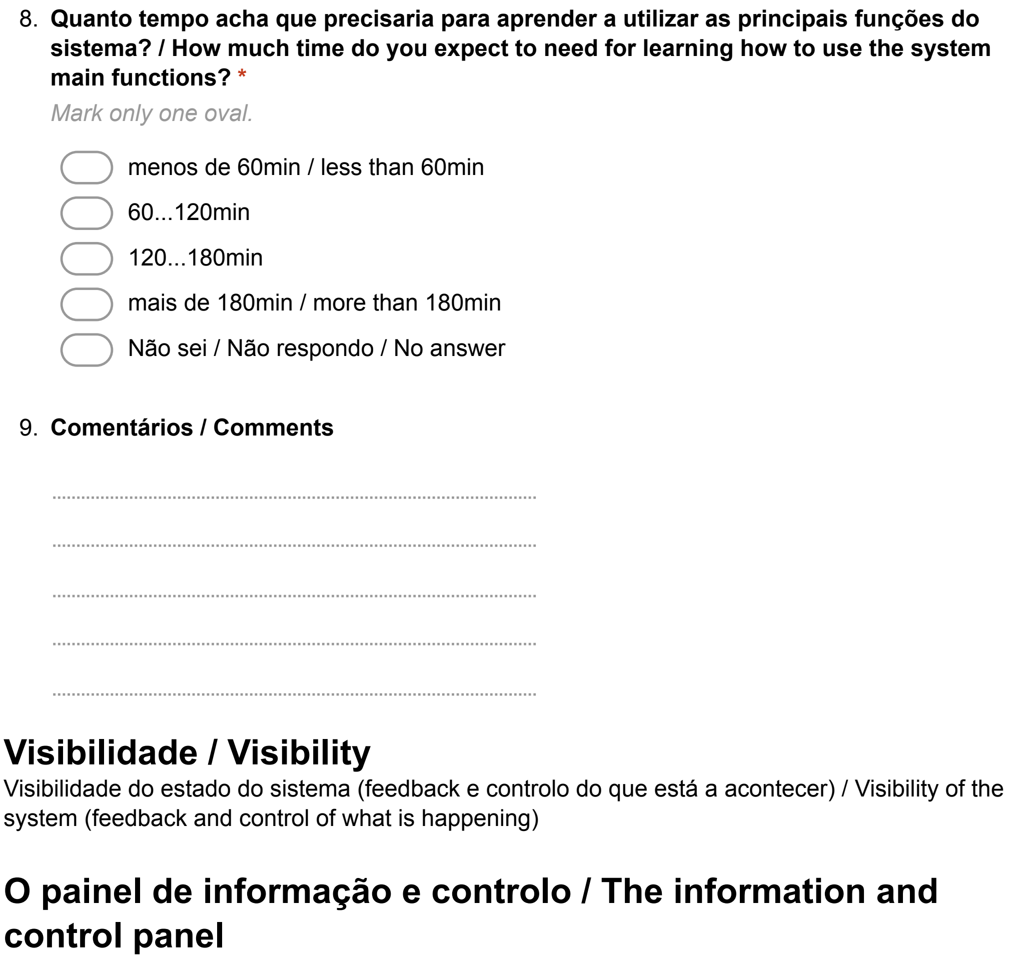
Facilidade na aprendizagem e compreensão do sistema / Learning and understanding the system

6. **O sistema é intuitivo? / Is the system intuitive?** *
(percebo facilmente como funciona à medida que experimento) / (i can understand easily how it works as i try it)
Mark only one oval.

○ Não / No
○ Nem por isso / Not really
○ Sim / Yes
○ Sim, bastante / Yes, very
○ Não sei / Não respondo / No answer

7. **Tenho facilidade em aprender a utilizar o sistema? / Is it easy to learn how to use the system?** *
(embora apenas tenha experimentado brevemente, considera que é facil aprender a utilizar o sistema?) / (althoug this brief experiment do you consider is easy to learn how to work with the system?)
Mark only one oval.

○ Não / No
○ Nem por isso / Not really
○ Sim / Yes
○ Sim, bastante / Yes, very
○ Não sei / Não respondo / No answer



8. **Quanto tempo acha que precisaria para aprender a utilizar as principais funções do sistema? / How much time do you expect to need for learning how to use the system main functions?** *

Mark only one oval.

○ menos de 60min / less than 60min

○ 60...120min

○ 120...180min

○ mais de 180min / more than 180min

○ Não sei / Não respondo / No answer

9. **Comentários / Comments**

....

....

....

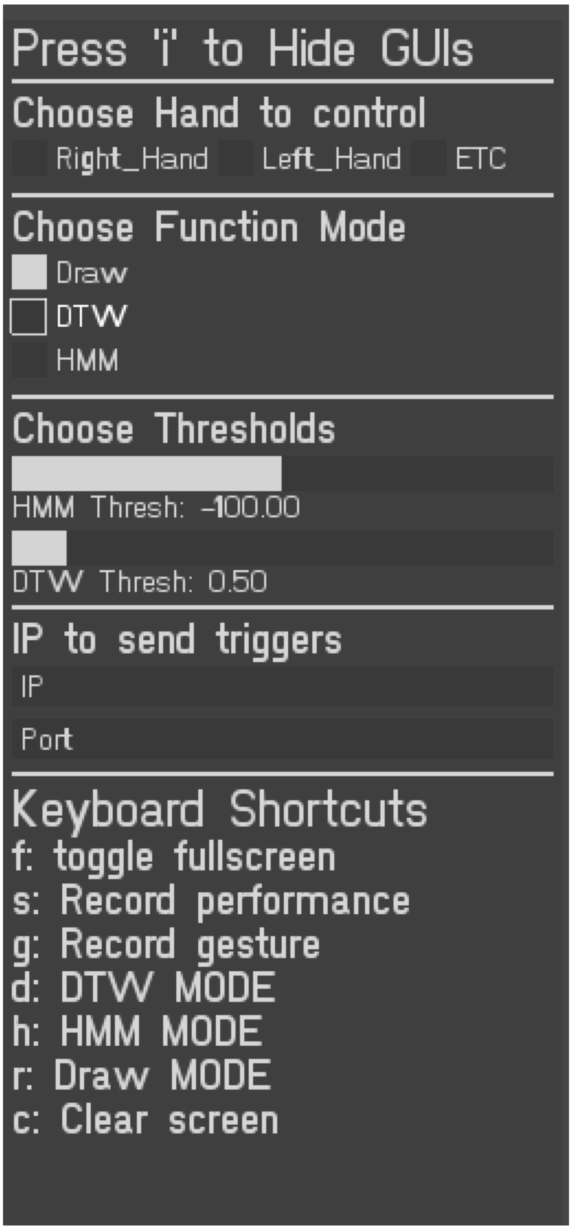
....

....

Visibilidade / Visibility

Visibilidade do estado do sistema (feedback e controlo do que está a acontecer) / Visibility of the system (feedback and control of what is happening)

O painel de informação e controlo / The information and control panel



Press 'i' to Hide GUIs

Choose Hand to control

□ Right_Hand □ Left_Hand □ ETC

Choose Function Mode

■ Draw

□ DTW

□ HMM

Choose Thresholds

HMM Thresh: -100.00

DTW Thresh: 0.50

IP to send triggers

IP

Port

Keyboard Shortcuts

f: toggle fullscreen

s: Record performance

g: Record gesture

d: DTW MODE

h: HMM MODE

r: Draw MODE

c: Clear screen

10. **Os botões usados para realizar as tarefas mais importantes estão claramente identificados? / Are the buttons for the main functions clearly identified?** *

Refere-se ao painel de botões que lhe permite escolher os modos de funcionamento e thresholds

Mark only one oval.

○ Nunca / Never
○ Quase nunca / Almost Never
○ Regularmente / Regularly
○ Quase sempre / Almost always
○ Sempre / Always
○ Não sei / Não respondo / No answer

11. **O estado dos botões (seleccionado / não seleccionado, posição dos sliders) é indicado com clareza? The state of the buttons (activated/ deactivated, position of the sliders) clearly identified?** *

Mark only one oval.

○ Nunca / Never
○ Quase nunca / Almost Never
○ Regularmente / Regularly
○ Quase sempre / Almost Always
○ Sempre / Always
○ Não sei / Não respondo / No answer

12. **Comentários / Comments**

....

....

....

....

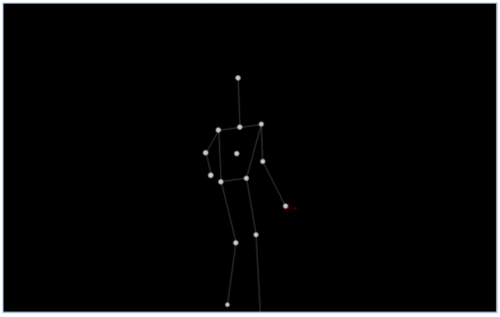
....

Usabilidade / Usability

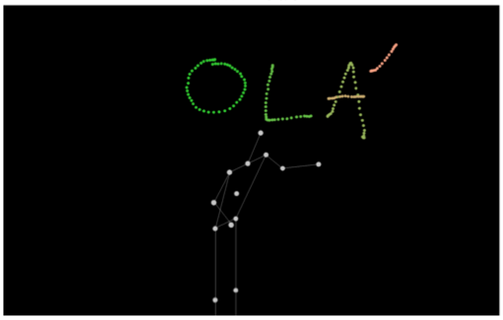
Interacção com o sistema e simplicidade de apresentação da informação. / Interaction with the system and simplicity when presenting the information.

Várias vistas possiveis do sistema / The several views possible.

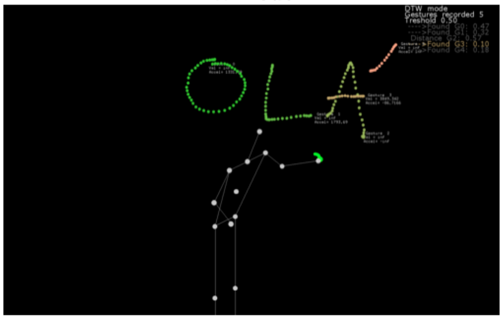
Clean view



Draw Mode

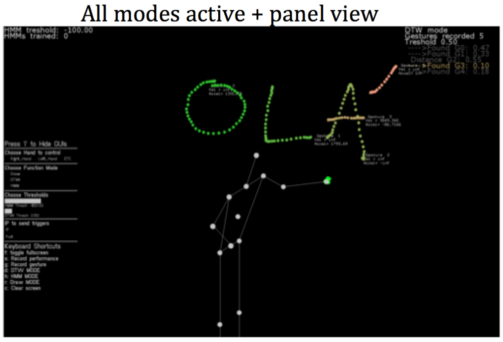


DTW Mode



```
DTW mode
Gestures recorded 5
Treshold 0.50
---->Found G0: 0.47
---->Found G1: 0.32
Distance G2: 0.57
---->Found G3: 0.10
---->Found G4: 0.18
```

All modes active + panel view



```
HMM treshold: -100.00
HMMs trained: 0
```

```
DTW mode
Gestures recorded 5
Treshold 0.50
---->Found G0: 0.47
---->Found G1: 0.33
Distance G2: 0.55
---->Found G3: 0.10
---->Found G4: 0.18
```

```
Keyboard Shortcuts
f: toggle fullscreen
s: Record performance
g: Record gesture
d: DTW MODE
h: HMM MODE
r: Draw MODE
c: Clear screen
```

13. **Consigue controlar a informação que é apresentada no ecrã? / Can you control the information that is presented on the screen?** *

Por exemplo, se consigo activar/desactivar as várias vistas possiveis (modo desenho, modo HMM, modo DTW). / For instance, can you control (activate / deactivate) the several possible views (draw mode, DTW mode, HMM mode)).

Mark only one oval.

○ Não / no

○ Nem por isso / Not Really

○ Sim / Yes

○ Sim, bastante / Yes, very

○ Não sei / Não respondo / No answer

14. **A informação contida no ecrã destaca-se do fundo? / Does the information on the screen stand out from the background?** *

Mark only one oval.

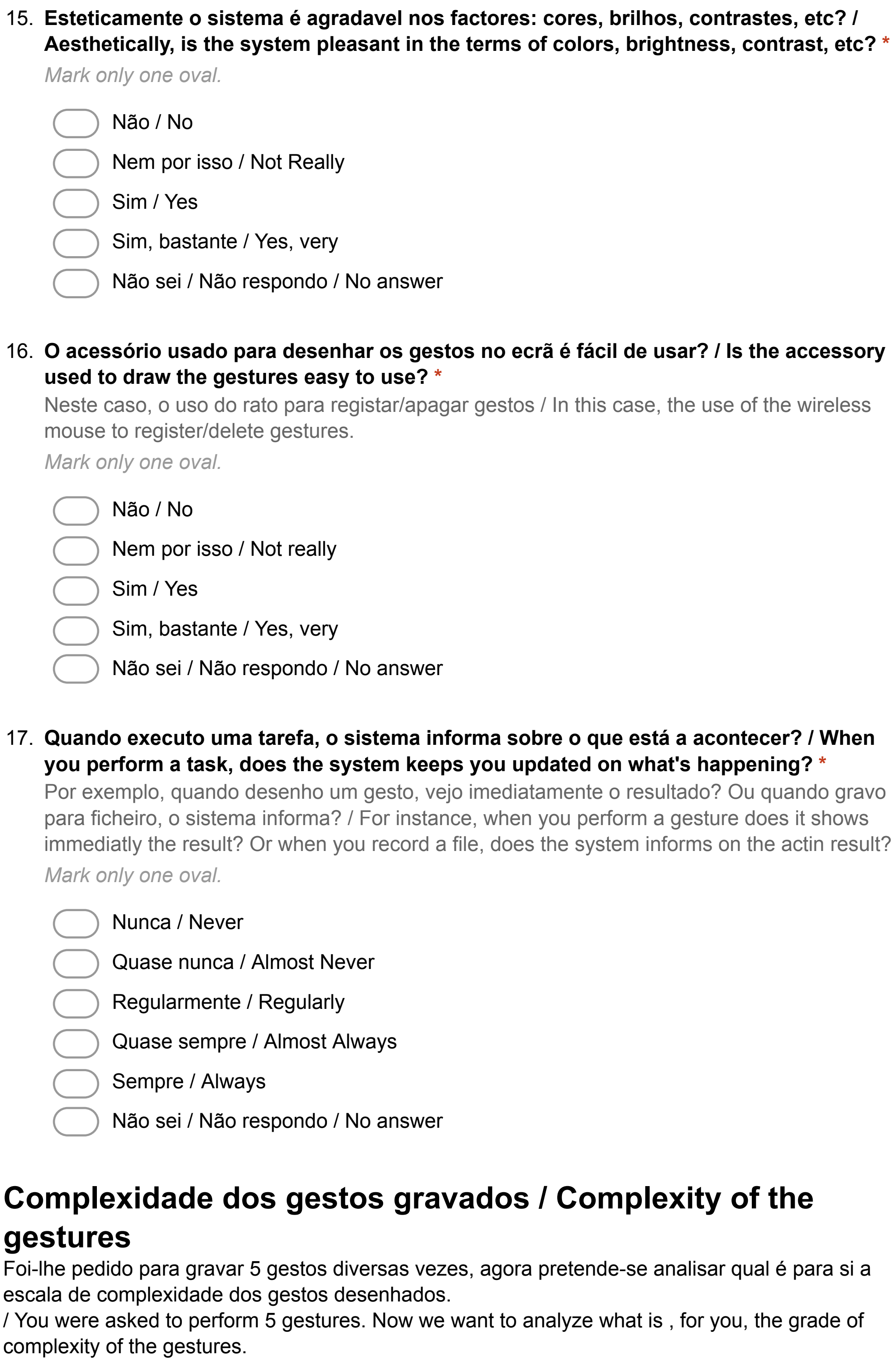
○ Não / No

○ Nem por isso / Not really

○ Sim / Yes

○ Sim, bastante / Yes, very

○ Não sei / Não respondo / No answer



15. **Esteticamente o sistema é agradavel nos factores: cores, brilhos, contrastes, etc? / Aesthetically, is the system pleasant in the terms of colors, brightness, contrast, etc?** *

Mark only one oval.

○ Não / No
○ Nem por isso / Not Really
○ Sim / Yes
○ Sim, bastante / Yes, very
○ Não sei / Não respondo / No answer

16. **O acessório usado para desenhar os gestos no ecrã é fácil de usar? / Is the accessory used to draw the gestures easy to use?** *

Neste caso, o uso do rato para registar/apagar gestos / In this case, the use of the wireless mouse to register/delete gestures.

Mark only one oval.

○ Não / No
○ Nem por isso / Not really
○ Sim / Yes
○ Sim, bastante / Yes, very
○ Não sei / Não respondo / No answer

17. **Quando executo uma tarefa, o sistema informa sobre o que está a acontecer? / When you perform a task, does the system keeps you updated on what's happening?** *

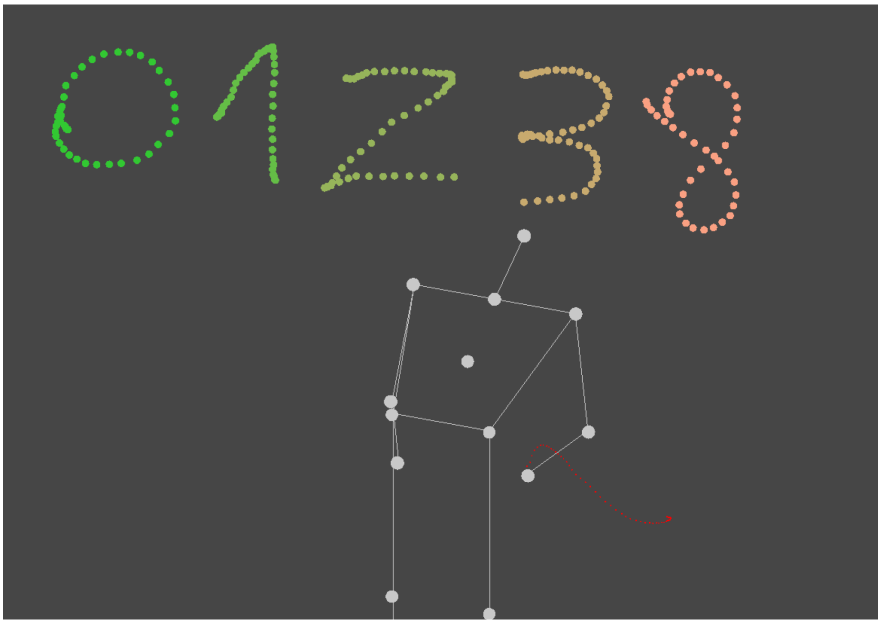
Por exemplo, quando desenho um gesto, vejo imediatamente o resultado? Ou quando gravo para ficheiro, o sistema informa? / For instance, when you perform a gesture does it shows immediatly the result? Or when you record a file, does the system informs on the actin result?

Mark only one oval.

○ Nunca / Never
○ Quase nunca / Almost Never
○ Regularmente / Regularly
○ Quase sempre / Almost Always
○ Sempre / Always
○ Não sei / Não respondo / No answer

Complexidade dos gestos gravados / Complexity of the gestures

Foi-lhe pedido para gravar 5 gestos diversas vezes, agora pretende-se analisar qual é para si a escala de complexidade dos gestos desenhados.
/ You were asked to perform 5 gestures. Now we want to analyze what is , for you, the grade of complexity of the gestures.

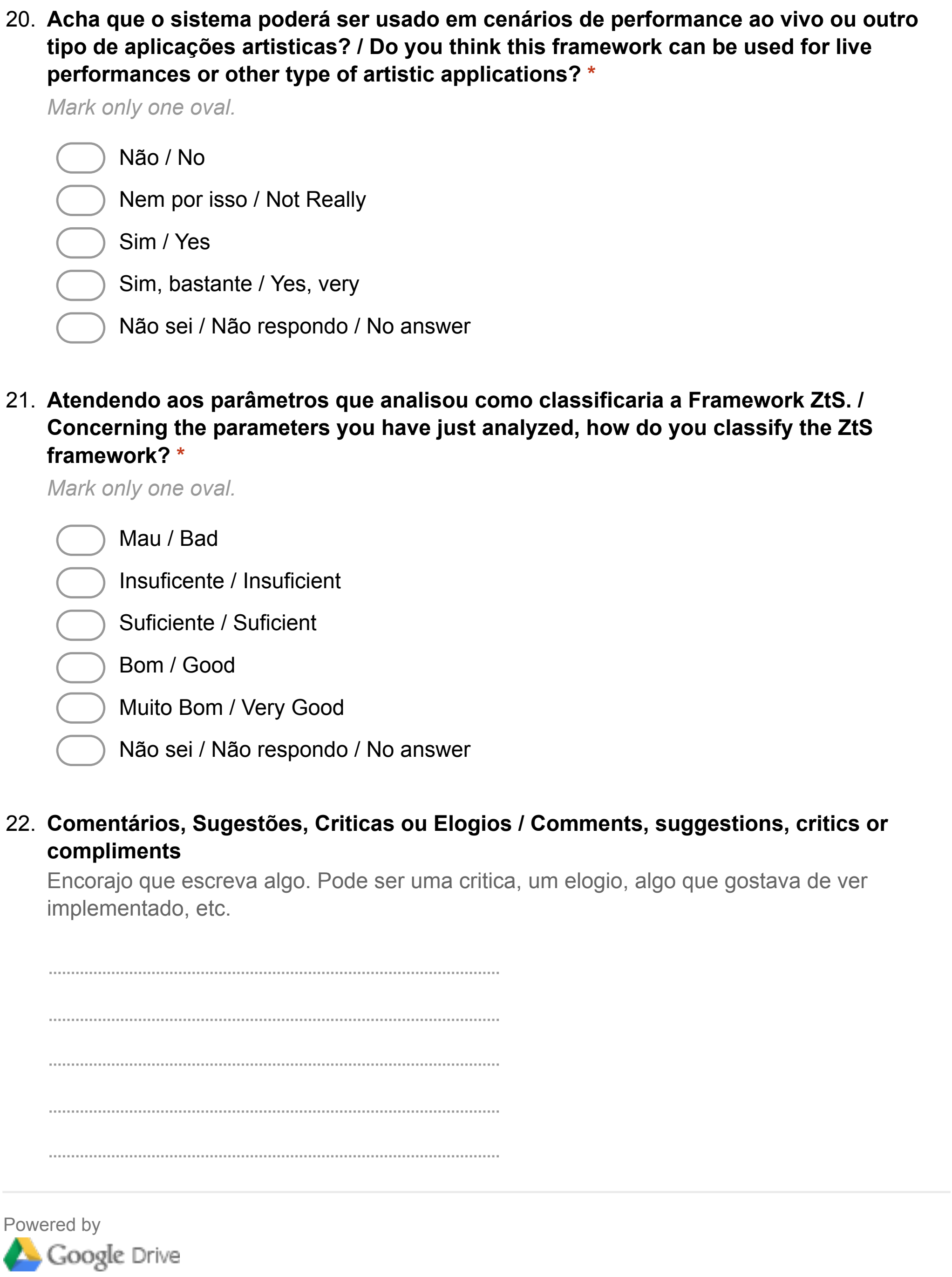


18. **Ordene os gestos que desenhou por complexidade crescente / Order the gestures you performed by its increasing complexity.**
(sendo o valor 1 o menos complexo e 5 o mais complexo. Não repita o grau de complexidade.) / Being the value 1 the less the complex and 5 the most complex. Do not repeat the classification for different gestures.
Mark only one oval per row.

	1	2	3	4	5
Gesto "0"	○	○	○	○	○
Gesto "1"	○	○	○	○	○
Gesto "Z"	○	○	○	○	○
Gesto "3"	○	○	○	○	○
Gesto "8"	○	○	○	○	○

19. **Comentários / Comments**

Aceitação da framework ZtS / Framework ZtS acceptability



20. **Acha que o sistema poderá ser usado em cenários de performance ao vivo ou outro tipo de aplicações artisticas? / Do you think this framework can be used for live performances or other type of artistic applications? ***

Mark only one oval.

○ Não / No
○ Nem por isso / Not Really
○ Sim / Yes
○ Sim, bastante / Yes, very
○ Não sei / Não respondo / No answer

21. **Atendendo aos parâmetros que analisou como classificaria a Framework ZtS. / Concerning the parameters you have just analyzed, how do you classify the ZtS framework? ***

Mark only one oval.

○ Mau / Bad
○ Insuficente / Insuficient
○ Suficiente / Suficient
○ Bom / Good
○ Muito Bom / Very Good
○ Não sei / Não respondo / No answer

22. **Comentários, Sugestões, Criticas ou Elogios / Comments, suggestions, critics or compliments**

Encorajo que escreva algo. Pode ser uma critica, um elogio, algo que gostava de ver implementado, etc.

....
....
....
....
....

Powered by
Google Drive

References

Aggarwal, J. (2011). Recognition of human activities. In Aggarwal, J., Barneva, R., Brimkov, V., Koroutchev, K., and Korutcheva, E., editors, *Combinatorial Image Analysis*, volume 6636 of *Lecture Notes in Computer Science*, pages 1–4. Springer Berlin Heidelberg.

Al-Hamadi, A., Elmezain, M., and Michaelis, B. (2010). Hand Gesture Recognition Based on Combined Features Extraction. *International Journal of . . .* , pages 1–6.

Alon, J., Athitsos, V., Yuan, Q., and Sclaroff, S. (2009). A unified framework for gesture recognition and spatiotemporal gesture segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(9):1685–1699.

Baltazar, A. (2009). *Extracção de Informações Rítmicas de Movimentos de Dança Através de um Sinal de Vídeo*. PhD thesis, FEUP.

Baltazar, A., Guedes, C., Gouyon, F., and Pennycook, B. (2010). A Real-time Human Body Skeletonization Algorithm for MAX/MSP/JITTER. In *Proceedings of the International Computer Music Conference*. Ann Arbor, MI: MPublishing, University of Michigan Library.

Baltazar, A., Martins, L., and Cardoso, J. (2012). ZATLAB: A Gesture Analysis System to Music Interaction. In *6th International Conference on Digital Arts (ARTECH 2012)*.

Bevilacqua, F. and Muller, R. (2005). A gesture follower for performing arts. *Proceedings of the International Gesture . . .* , pages 3–4.

Bevilacqua, F., Müller, R., and Schnell, N. (2005). Mnm: a max/msp mapping toolbox. In *Proceedings of the 2005 conference on New interfaces for musical expression*, pages 85–88. National University of Singapore.

Blomster, J. (2006). Orientation estimation combining vision and gyro measurements. *KTH Electrical Engineering, Master's Degree Project, Stockholm, Sweden*, pages 1–44.

Bokowiec, M. A. (2011). V ! OCT (Ritual): An Interactive Vocal Work for Bodycoder System and 8 Channel Spatialization. In *NIME 2011 Proceedings*, number June, pages 40–43.

Boring, R. L. (2002). Human-computer interaction as cognitive science. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 46, pages 1767–1771. SAGE Publications.

Camurri, A., Hashimoto, S., Ricchetti, M., Ricci, A., Suzuki, K., Trocca, R., and Volpe, G. (2000). Eyesweb: Toward gesture and affect recognition in interactive dance and music systems. *Comput. Music J.*, 24(1):57–69.

Camurri, A., Mazzarino, B., and Ricchetti, M. (2004). Multimodal analysis of expressive gesture in music and dance performances. *Gesture-based ...*, (1990).

Camurri, a., Volpe, G., Poli, G. D., and Leman, M. (2005). Communicating expressiveness and affect in multimodal interactive systems. *Multimedia, IEEE*, 12(1):43–53.

Caron, F., Duflos, E., Pomorski, D., and Vanheeghe, P. (2006). Gps/imu data fusion using multisensor kalman filtering: introduction of contextual aspects. *Information Fusion*, 7(2):221–230.

Castellano, G., Villalba, S., and Camurri, A. (2007). Recognising human emotions from body movement and gesture dynamics. *Affective computing and intelligent ...*, pages 71–82.

Chairman-Hewett, T. T. (1992). *ACM SIGCHI curricula for human-computer interaction*. ACM.

Chang, Y.-J., Chen, S.-F., and Huang, J.-D. (2011). A kinect-based system for physical rehabilitation: A pilot study for young adults with motor disabilities.

Research in Developmental Disabilities, 32(6):2566 – 2570.

Collins, N. and Kiefer, C. (2010). Musical exoskeletons: Experiments with a motion capture suit. In *New Interfaces for Musical Expression*, number June, pages 15–18.

Corradini, A. (2001a). Dynamic time warping for off-line recognition of a small gesture vocabulary. In *Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, 2001. Proceedings. IEEE ICCV Workshop on*, pages 82–89. IEEE.

Corradini, A. (2001b). Dynamic time warping for off-line recognition of a small gesture vocabulary. In *In Proceedings of the IEEE ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in RealTime Systems (RATFG-RTS'01*, page 82. IEEE Computer Society.

Craine, D. and MacKrell, J. (2004). *The Oxford Dictionary Of Dance*. Oxford Paperback Reference. Oxford University Press, Incorporated.

Dietrich, J. E. (1983). *Play Direction*. Prentice Hall, 2nd edition.

Dobrian, C. and Bevilacqua, F. (2003). Gestural control of music: using the vicon 8 motion capture system. In *Proceedings of the 2003 conference on New interfaces for musical expression*, pages 161–163. National University of Singapore.

Efron, D. (1972). *Gesture, Race and Culture*. Mouton and Co.

Elmezain, M. and Al-Hamadi, A. (2009). A Hidden Markov Model-Based Isolated and Meaningful Hand Gesture Recognition. *International Journal of ...*, pages 156–163.

Feldman, R. S. and Rimé, B., editors (1991). *Fundamentals of Nonverbal Behavior (Studies in Emotion and Social Interaction)*. Cambridge University Press.

Feyereisen, P. and de Lannoy, J.-D. (1991). *Gestures and Speech: Psychological Investigations (Studies in Emotion and Social Interaction)*. Cambridge University Press.

Fiebrink, R., Trueman, D., and Cook, P. (2009). A metainstrument for interactive, on-the-fly machine learning. In *Proc. NIME*, volume 2, page 3.

Foxlin, E. and Naimark, L. (2003). Vis-tracker: A wearable vision-inertial self-tracker. In *Proceedings of the IEEE Virtual Reality 2003*, VR '03, pages 199–, Washington, DC, USA. IEEE Computer Society.

Fraisse, P. (1982). Rhythm and Tempo. In Deutsch, D., editor, *The Psychology of Music*, Springer Handbook of Auditory Research, pages 149–180. Academic Press.

Gillian, N., Knapp, R. B., and O'Modhrain, S. (2011). A machine learning toolbox for musician computer interaction. *Proceedings of the 2011 International Coference on New Interfaces for Musical Expression (NIME11)*.

Godoy, R. I. and Leman, M. (2009). *Musical Gestures: Sound, Movement, and Meaning*. Routledge.

Grosan, C. and Abraham, A. (2011). *Machine Learning*, volume 17 of *Intelligent Systems Reference Library*. Springer Berlin Heidelberg.

Guedes, C. (2005a). *Mapping Movement to Musical Rhythm: A Study in Interactive Dance*. PhD thesis, New York University.

Guedes, C. (2005b). THE M-OBJECTS : A SMALL LIBRARY FOR MUSICAL RHYTHM GENERATION AND MUSICAL TEMPO CONTROL FROM DANCE MOVEMENT IN REAL TIME. In *Proceedings of the International Computer Music . . . .*

Harrison, M. A., Atkinson, H., and De Weerdt, W. (1992). Benesh movement notation: A tool to record observational assessment. *International Journal of Technology Assessment in Health Care*, 8:44–54.

Hastie, T., Tibshirani, R., and Friedman, J. (2003). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction (Springer Series in Statistics)*. Springer, 0 edition.

Hunt, A., Wanderley, M., and Kirk, R. (2000). Towards a model for instrumental mapping in expert musical interaction. In *Proceedings of the 2000 International Computer Music Conference*, pages 209–212.

Inokuchi, S. (2010). Review of kansei research in japan. *IJSE*, 1(1):18–29.

Joguet, C., Caritu, Y., and David, D. (2003). Pen-like'natural graphic gesture capture disposal, based on a micro-system. In *Proc. of Smart Objects Conference SOC'03, Grenoble, France*. Citeseer.

Jr, G. F. (2005). The viterbi algorithm: A personal history. *arXiv preprint cs/0504020*.

Kahol, K., Tripathi, P., and Panchanathan, S. (2004). Automated gesture segmentation from dance sequences. In *Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on*, pages 883–888. IEEE.

Kang, H., Woo Lee, C., and Jung, K. (2004). Recognition-based gesture spotting in video games. *Pattern Recognition Letters*, 25(15):1701–1714.

Kendon, A. (1970). Movement coordination in social interaction: Some examples described. *Acta Psychologica*, 32(0):101 – 125.

Kendon, A. (1980). Gesticulation and speech: two aspects of the process of utterance. In Key, M. R., editor, *The Relationship of Verbal and Nonverbal Communication*, pages 207–227. Mouton, The Hague.

Kendon, A. (1994). Do Gestures Communicate? A Review. *Research on Language & Social Interaction*, 27(3):175–200.

Keogh, E. and Ratanamahatana, C. A. (2005). Exact indexing of dynamic time warping. *Knowledge and Information Systems*, 7(3):358–386.

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

Leante, L. (2007). Multimedia Aspects of Progressive Rock Shows: Analysis of the Performance of ”The Musical Box”. In *Proceedings of the International Conference “Composition and Experimentation in British Rock 1966-1976*.

Lemire, D. (2009). Faster Retrieval with a Two-Pass Dynamic-time-warping lower bound. *Pattern recognition*, (June 2009):1–26.

Licsár, A. and Szirányi, T. (2005). User-adaptive hand gesture recognition system with interactive training. *Image and Vision Computing*, 23(12):1102–1114.

Likert, R. (1932). A technique for the measurement of attitudes. *Archives of* *psychology*.

Liu, C. (2009). cuhmm: a cuda implementation of hidden markov model training and classification. *The Chronicle of Higher Education*.

Lobo, J., Lucas, P., Dias, J., and Traca de Almeida, A. (1995). Inertial navigation system for mobile land vehicles. In *Industrial Electronics, 1995. ISIE '95., Proceedings of the IEEE International Symposium on*, volume 2, pages 843–848 vol.2.

Loke, L., Larssen, A., and Robertson, T. (2005). Labanotation for design of movement-based interaction. *Proceedings of the second ...*, pages 113–120.

Lourenço, S. (2010). European piano schools : Russian , german and french classical piano interpretation and technique. *Journal of Science and Technology of the Arts*, 2(1).

Luinge, H. (2002). *Inertial Sensing of Human Movement*.

Maletic, V. (1987). *Body, Space, Expression: The Development of Rudolf Laban's Movement and Dance Concepts*. Approaches to semiotics. Mouton de Gruyter.

McNeill, D. (1985). So you think gestures are nonverbal? *Psychological review*, 92(3):350–371.

McNeill, D. (1992). *Hand and Mind: What Gestures Reveal about Thought*. University of Chicago Press, Chicago.

McNeill, D. and Levy, E. (1982). Conceptual representations in language activity and gesture. In Jarvella, R. J. and Klein, W., editors, *Speech, Place, and Action*, pages 271–295. Wiley, Chichester.

Mitobe, K., Kaiga, T., Yukawa, T., Miura, T., Tamamoto, H., Rodgers, A., and Yoshimura, N. (2006). Development of a motion capture system for a hand using a magnetic three dimensional position sensor. *ACM SIGGRAPH 2006 Research posters on - SIGGRAPH '06*, page 102.

Mitra, S. and Acharya, T. (2007). Gesture recognition: A survey. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 37(3):311–324.

Mulder, A. et al. (1994). Human movement tracking technology. *Simon Fraser University School of Kinesiology Technical Report*, pages 94–1.

Naveda, L. and Leman, M. (2008). Representation of Samba dance gestures, using a multi-modal analysis approach. In *5th International Conference on Enactive Interfaces*, pages 68–74. Edizione ETS.

Nefian, A. V. and Hayes III, M. H. (1998). Hidden markov models for face recognition. In *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, volume 5, pages 2721–2724. IEEE.

Nielsen, J. (1993). *Usability Engineering*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Nort, D. V., Wanderley, M. M., and Van Nort, D. (2006). The LoM Mapping Toolbox for Max/MSP/Jitter. In *Proceedings of the International Computer Music Conference, New Orleans, USA*.

Pierce, K. (1998). Dance notation systems in late 17th-century france. *Early Music*, 26(2):286–299.

Polotti, P. and Goina, M. (2011). EGGS in Action. In *NIME*, number June, pages 64–67.

Povall, R. (1998). Technology is with us. *Dance Research Journal*, 30(1):1–4.

Rabiner, L. (1989). Tutorial on Hidden Markov Models and Selected Applications in speech Recognition. *Proceedings of the IEEE*.

Rabiner, L. and Juang, B.-H. (1993). *Fundamentals of Speech Recognition*. Prentice Hall, 1 edition.

Ravi, A. (2013). Automatic gesture recognition and tracking system for physiotherapy. *Electrical Engineering and Computer Sciences University of California at Berkeley*.

Rett, J. and Dias, J. (2007). Human-robot interface with anticipatory characteristics based on laban movement analysis and bayesian models. In *Rehabilitation Robotics, 2007. ICORR 2007. IEEE 10th International Conference on*, pages

257–268. IEEE.

Rimé, B. (1982). The elimination of visible behaviour from social interactions: Effects on verbal, nonverbal and interpersonal variables. *European Journal of Social Psychology*, 12(2):113–129.

Roetenberg, D. (2006). *Inertial and Magnetic Sensing of Human Motion*, volume PhD. University of Twente.

Roetenberg, D., Slycke, P. J., and Veltink, P. H. (2007). Ambulatory position and orientation tracking fusing magnetic and inertial sensing. *Biomedical Engineering, IEEE Transactions on*, 54(5):883–890.

Roth, W.-M. (2001). Gestures: Their Role in Teaching and Learning. *Review of Educational Research*, 71(3):365–392.

Rovan, J. B., Wanderley, M. M., Dubnov, S., and Depalle, P. (1997). Instrumental gestural mapping strategies as expressivity determinants in computer music performance. In *Proceedings of Kansei-The Technology of Emotion Workshop*, pages 3–4. Citeseer.

Rowe, M. L. and Goldin-meadow, S. (2009). development. *First Language*, 28(2):182–199.

Royce, A. P. (1984). *Movement and Meaning: Creativity and Interpretation in Ballet and Mime*. Indiana Univ Pr.

Rubine, D. H. (1991). *The automatic recognition of gestures*. PhD thesis, University of Toronto.

Ryman, R. (2001). Labanotation and life forms®: Computer animation as a complement to dance notation. In *Proceedings of the 22nd Biennial Conference of International Council of Kinetography Laban/Labanotation, Columbus, Ohio, USA*, volume 26. Citeseer.

Schacher, J. C. (2010). Motion To Gesture To Sound : Mapping For Interactive Dance. Number Nime, pages 250–254.

Segen, J. and Kumar, S. (1998). Gesture vr: Vision-based 3d hand interace for spatial interaction. In *Proceedings of the Sixth ACM International Conference on Multimedia*, MULTIMEDIA '98, pages 455–464, New York, NY, USA. ACM.

Senin, P. (2008a). Dynamic time warping algorithm review. *Honolulu, USA*, (December):1–23.

Senin, P. (2008b). Dynamic time warping algorithm review. *Honolulu, USA*, (December):1–23.

Shih, C.-H., Chang, M.-L., and Shih, C.-T. (2010). A limb action detector enabling people with multiple disabilities to control environmental stimulation through limb action with a nintendo wii remote controller. *Research in Developmental Disabilities*, 31(5):1047 – 1053.

Stowers, J., Hayes, M., and Bainbridge-Smith, A. (2011). Altitude control of a quadrotor helicopter using depth map from microsoft kinect sensor. In *Mechatronics (ICM), 2011 IEEE International Conference on*, pages 358–362.

Su, Y., Allen, C., Geng, D., Burn, D., Brechany, U., Bell, G., and Rowland, R. (2003). 3-d motion system (”data-gloves”): application for parkinson’s disease. *Instrumentation and Measurement, IEEE Transactions on*, 52(3):662–674.

Tanaka, A. (2000). Musical performance practice on sensor-based instruments. *Trends in Gestural Control of Music*, 13:389–405.

Ten Holt, G., Reinders, M., and Hendriks, E. (2007). Multi-dimensional dynamic time warping for gesture recognition. In *Thirteenth annual conference of the Advanced School for Computing and Imaging*, volume 119.

Vamplew, P. and Adams, A. (1995). Recognition and anticipation of hand motions using a recurrent neural network.

Varela, F., Thompson, E., and Rosch, E. (1993). *The Embodied Mind: Cognitive Science and Human Experience*. Cognitive science: Philosophy, psychology. MIT Press.

Villaroman, N., Rowe, D., and Swan, B. (2011). Teaching natural user interaction using openni and the microsoft kinect sensor. In *Proceedings of the 2011 conference on Information technology education*, SIGITE ’11, pages 227–232, New York, NY, USA. ACM.

Viterbi, A. (1967). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *Information Theory, IEEE Transactions on*, 13(2):260–269.

Volpe, G. (2005). Expressive Gesture in Performing Arts and New Media: The Present and the Future. *Journal of New Music Research*, 34(1):1–3.

Wachsmuth, I. and Fröhlich, M., editors (1998). *Gesture and Sign Language in Human-Computer Interaction, International Gesture Workshop, Bielefeld, Germany, September 17-19, 1997, Proceedings*, volume 1371 of *Lecture Notes in Computer Science*. Springer.

Wang, G., Cook, P., and Others (2003). ChucK: A concurrent, on-the-fly audio programming language. In *Proceedings of International Computer Music Conference*, pages 219–226.

Watson, R. (1993). A survey of gesture recognition techniques technical report tcd-cs-93-11. *Department of Computer Science, Trinity College . . .*, (July).

Winkler, T. (1995a). Making motion musical: Gesture mapping strategies for interactive computer music. In *ICMC Proceedings*, pages 261–264.

Winkler, T. (1995b). Making motion musical: Gesture mapping strategies for interactive computer music. In *ICMC Proceedings*, pages 261–264.

Wolf, W., Ozer, B., and Lv, T. (2002). Smart cameras as embedded systems. *Computer*, 35(9):48–53.

Wong, A. (2007). *Low-Cost Visual/Inertial Hybrid Motion Capture System for Wireless 3D Controllers*. PhD thesis.

Wright, M. (2005). Open sound control: an enabling technology for musical networking. *Organised Sound*, 10(3):193.

Wright, M., Freed, A., Lee, A., Madden, T., and Momeni, A. (2001). Managing complexity with explicit mapping of gestures to sound control with osc. In *International Computer Music Conference*, pages 314–317. Citeseer.

Wu, Y. and Huang, T. (1999). Vision-based gesture recognition: A review. *Urbana*, pages 103–115.

Yamato, J., Ohya, J., and Ishii, K. (1992). Recognizing Human action in Time-sequential Images using Hidden Markov Model. *Computer Vision and Pattern*

. . . .

Yang, H.-D., Park, A.-Y., and Lee, S.-W. (2006). Robust spotting of key gestures from whole body motion sequence. In *Automatic Face and Gesture Recognition, 2006. FGR 2006. 7th International Conference on*, pages 231–236. IEEE.

Yoo, M., Beak, J., and Lee, I. (2011). Creating Musical Expression using Kinect. *visualcomputing.yonsei.ac.kr*, (June):324–325.

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

Zafrulla, Z., Brashear, H., Starner, T., Hamilton, H., and Presti, P. (2011). American sign language recognition with the kinect. In *Proceedings of the 13th international conference on multimodal interfaces*, pages 279–286. ACM.

Zhang, Z. (2012a). Microsoft kinect sensor and its effect. *MultiMedia, IEEE*, 19(2):4–10.

Zhang, Z. (2012b). Microsoft kinect sensor and its effect. *MultiMedia, IEEE*, 19(2):4–10.

Zhao, L. and Badler, N. I. (2001). Synthesis and acquisition of laban movement analysis qualitative parameters for communicative gestures.