Universidade de Aveiro
Departamento de Electrónica, Telecomunicações e Informática,
2019

Pedro Miguel
André Coelho

Generic Modalities for Silent Interaction
Modalidades Genéricas para Interação Silenciosa

Universidade de Aveiro
Departamento de Electrónica, Telecomunicações e Informática,
2019

Pedro Miguel
André Coelho

Generic Modalities for Silent Interaction
Modalidades Genéricas para Interação Silenciosa

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica do Doutor Samuel de Sousa Silva, Investigador do Instituto de Engenharia Electrónica e Informática de Aveiro e do Doutor António Joaquim da Silva Teixeira, Professor Associado com Agregação do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro.

o júri / the jury

presidente / president	**Doutora Beatriz Sousa Santos** Professora Associada com Agregação da Universidade de Aveiro
vogais / examiners committee	**Doutor José Casimiro Pereira** Professor Adjunto do Instituto Politécnico de Tomar
	Doutor Samuel de Sousa Silva (Orientador) Investigador da Universidade de Aveiro

agradecimentos

Em primeiro lugar, quero agradecer aos meus orientadores, Doutor Samuel Silva e Professor Doutor António Teixeira, pela confiança que em mim depositaram, por toda a dedicação e pela motivação incondicional.

Agradeço, de igual forma, ao Doutor Nuno Almeida, pelo incentivo e apoio na elaboração da presente dissertação.

Aos meus amigos e colegas, por todo o encorajamento e amizade.

À Mariana, pela sua paciência, compreensão e ajuda prestada, contribuindo para chegar ao fim deste percurso.

Por último, o meu profundo e sentido agradecimento à minha família, por toda a força e por estarem sempre presentes ao longo da minha vida académica.

palavras-chave

Interação, interação multimodal, modalidades genéricas, gestos, olhar, fala silenciosa

resumo

A comunicação humana não se reduz ao verbal. Existem vários aspetos que podem complementar ou substituir a fala audível, como gestos, olhar e fala silenciosa.

A diversidade e os contextos de uso de sistemas interativos exigem a proposta de formas mais naturais (e intuitivas) de interação. Neste contexto, a multimodalidade foi introduzida, oferecendo diferentes métodos de interação, permitindo encurtar a distância existente entre o humano e o computador.

No entanto, o desenvolvimento de sistemas interativos onde são integradas múltiplas modalidades de interação é uma tarefa complexa, particularmente devido à ampla variedade de tecnologias que os programadores necessitam de dominar. Adicionalmente, o suporte para uma tecnologia de interação particular é frequentemente desenvolvida para aplicações específicas, o que dificulta a sua reutilização para outros contextos.

Neste trabalho, é proposto e implementado um modelo para a integração de modalidades de interação silenciosas em sistemas interativos. É desenhado e desenvolvido um conjunto de modalidades genéricas para suportar uma interação silenciosa alinhada com a visão proposta.

Este trabalho contribuiu para o desenvolvimento de três modalidades genéricas, nomeadamente a modalidade de gestos, a modalidade de $gaze$ e a modalidade de fala silenciosa. As modalidades genéricas propostas já incluem módulos básicos para que seja possível fazer o uso das mesmas. Foram também realizados testes às capacidades das modalidades, foi demonstrado o seu uso numa aplicação de prova de conceito, com o intuito de controlar o VLC, um reprodutor de conteúdos multimédia, e uma avaliação feita por programadores de modo a avaliar a facilidade de uso da modalidade de gestos.

keywords Interaction, multimodal interaction, generic modalities, gestures, gaze, silent speech

abstract Human communication is not reduced to verbal. There are several aspects that can complement or replace audible speech, such as gestures, gaze and silent speech.

The diversity and contexts of use of interactive systems demand the proposal of more natural (and intuitive) forms of interaction. In this context, multimodality was introduced, providing different methods of interaction, allowing to shorten the gap between the human and the computer.

However, the development of interactive systems where there are integrated multiple interaction modalities is a complex task, particularly due to the wide range of technologies that developers need to master to do it. Additionally, the support for a particular interaction technology is often developed for specific applications, which hinders reusing it for other contexts.

In this research work, a model is proposed and developed for the integration of silent interaction modalities in interactive systems. A set of generic modalities is designed and developed to support a silent interaction aligned with the proposed vision.

This research work contributed to the development of three generic modalities, namely the gestures modality, the gaze modality and the silent speech modality. The generic modalities proposed already include basic modules so that they can be used. Tests were also carried out on the capabilities of the modalities, it was demonstrated their use in a proof-of-concept application, so as to control the VLC, a multimedia content player, and an evaluation done by developers in order to evaluate the facility of use of the gestures modality.

Contents

List of Figures

List of Tables

Acronyms

AAL Ambient Assisted Living

ABNF Augmented Backus-Naur Form

API Application Programming Interface

ASR Automatic Speech Recognition

AU Action Unit

CCXML Call Control eXtensible Markup Language

EEG Electroencephalography

EMMA Extensible MultiModal Annotation markup language

EOG Electro-Oculogram

FACS Facial Action Coding System

GUI Graphical User Interface

HCI Human-Computer interaction

HD High Definition

HTML Hypertext Markup Language

HTTP Hypertext Transfer Protocol

IM Interaction Manager

IR Infrared

JSGF Java Speech Grammar Format

LAN Local Area Network

LSTM Long Short-Term Memory

MLP Multilayer Perceptron

MMI Multimodal Interaction

MS Microsoft

MVC Model-View-Controller

NAM Non-Audible Murmur

NIR Near Infrared

RGB Red-Green-Blue

SCXML State Chart XML

SDK Software Development Kit

SEMG Surface Electromyography

SMIL Synchronized Multimedia Integration Language

SRGS Speech Recognition Grammar Specification

SSI Silent Speech Interface

SSML Speech Synthesis Markup Language

SVM Support Vector Machine

TTS Text to Speech

URI Uniform Resource Identifier

URL Uniform Resource Locator

VGB Visual Gesture Builder

VLC Video LAN Client

VOG Video Oculography

W3C World Wide Web Consortium

XHTML Extensible Hypertext Markup Language

XML Extensible Markup Language

Chapter 1

Introduction

1.1 Context

Human-human communication is not always about audible speech. There are several aspects that can complement or replace it, such as how we move our hands, the way and to where we look, and how our lips and jaw move, when we speak.

To improve the way we communicate with interactive systems, we need to move towards the most natural forms of communication, for each context, getting closer to what happens in human-human communication. The proposal of multimodal interactive systems aims to bring forward and implement systems that enable the use of multiple forms of interaction, and the integration of those forms that bring an increased naturalness to the interaction, such as gestures, gaze and silent speech should deserve particular attention, because these non-verbal forms of interaction are the most used in communications between humans.

1.2 Problems and Challenges

The development of interactive systems that include a variety of interaction modalities is a complex task, particularly due to the wide range of technologies that developers need to master to do it. Additionally, the support for a particular interaction technology is often developed for specific applications, which hinders reusing it for other contexts.

In many cases, the complexity of the required interaction requires mastering aspects, such as the training of gestures, which place an extra burden on the developer and preclude an off-the-shelf use for common requirements. Thus, this type of interaction must serve or complement a human-human interaction, which developing from square one could be a hard challenge for developers.

One challenge is to support the development of multimodal applications, by programmers, by using the required modalities. With a practical and simple integration of the gestures, gaze and silent speech modalities into application systems, a human-computer silent interaction is achieved.

1.3 Motivation Scenarios and Requirements

One of the greatest advantages of the generic modalities is the fact that they can be reused as often as the user wants and applied in several scenarios. In the context of silent interactions,

there is a wide range of silent interaction scenarios.

In this section, there are presented some illustrative scenarios that could be used aligned with the generic modalities developed. There are also presented the requirements of this research work.

1.3.1 Illustrative Scenarios

Nowadays, more recent technologies are increasingly being used in the context of a Smart Home, as technology evolves and the user is always searching for new needs and more efficient ways of performing daily tasks.

In this type of context, the interaction needs to be more focused to the user. Thus, one considers some interactions between a human and a computer, using this type of technologies, the interactions with users that needs more privacy or when the user needs a quiet interaction. Furthermore, scenarios where the user may have physical limitations from the upper body are also applicable scenarios using a verbal and a non-verbal interaction.

1.3.1.1 Scenario 1: Nursing home

In a nursing home, there are several types of patients. There are patients that have no physical and mental disabilities, some of them have one or another and others have a more severe physical but no mental disabilities and they are bedridden. This last type of patients, due to their inability to move, must have a much more support and a more complex health care than the other patients.

In this scenario, all rooms in a nursing home have a command button above each bed of the patients that, if it is pressed, a nurse is called to the specific room and respond to the patient's request. One problem that is driven to this type of patients is that they can not press the command button because they are bedridden and have motor disabilities.

Nowadays, every thirty minutes, a nurse have to check each room of the bedridden patients and, in addition to consume more time to the nurse that could be useful for other tasks, something could happen to the patient in the thirty minutes period of time and have no response.

1.3.1.2 Scenario 2: Making a cake

In a home, a user is making a cake and, at that moment, he or she is beating egg whites and the noise from it is clearly high. The user can move his or her head to the software equipment and call other family member, uttering silently "call" while looking at the picture of the person that he or she wants to call, that is inside the house, but in another compartment.

1.3.1.3 Scenario 3: Home actions in a silent interaction

The user has a baby sleeping on his or her lap and does not want to make any noise or, in another situation, the user is listening to music or there is noise around from the television. The user can perform home actions interacting with the system in a silent communication.

For example, the user wants to turn on / off the heating, the air conditioning, the electric fireplace, the television and even changing the volume up / down of the television. The user can move his or her head to the software equipment and utters silently "turn on" / "turn off" or "more volume" / "less volume" while looking, by a gaze interaction, at the home equipment.

When the user utters silently the action of turning on an equipment, it responds back with its actual value and asks if he or she is satisfied with that value. If the user is pleased with that value, he or she can make a happy face to the software equipment and the action is performed. If not, the user can make an angry face and, after that, utters silently "more volume" / "less volume", corresponding to the final value that he or she wants to introduce in the system.

1.3.1.4 Scenario 4: Making a mockup in an electronic laboratory

A user is making a mockup model for a project and he or she has both hands busy building or welding the mockup. The user wants to visualize the next steps of the mockup design, so he or she gazes to the upper or to the lower area of the software equipment display in order to go to the previous or to the next page of the design.

The user can also zoom in / zoom out by uttering silently "more zoom" / "less zoom" to the software equipment.

1.3.1.5 Scenario 5: Navigating on the internet

A user is navigating on the internet and, in a certain moment, he or she starts to approach to the display of the computer and closing his or her eyes with the purpose of reading more closely to the computer. The computer recognizes the user's approach and it decreases the luminosity of the computer.

1.3.1.6 Scenario 6: Turning off the television

A user watching TV at home closes his or her eyes and falls asleep due to the tiring day he or she had. The computer recognizes that the user fell asleep and in order to save power and energy consumption from the TV, the computer turns it off after 5 minutes of the gesture recognition.

1.3.1.7 Scenario 7: Opening blinds

A user woke up and he or she wants to open the blinds. The user opens the lamp light and moves his head to the software equipment and utters to the system to perform the action required.

1.3.1.8 Scenario 8: Responding to daily basic tasks

The user has voice problems in that week due to a cold and he or she does not know what time is it at that moment, so he or she asks to the software system, uttering, the actual time.

The user also wants to make an arithmetic calculation, so he or she asks the system to do it and then the system responds by voice feedback or, if the system recognizes that there is noise around, it responds by visual feedback on the software equipment.

1.3.2 Requirements

The main requirement of this research work is to make simple the development of multimodal applications by generic multimodal modalities that are flexible and practical to use.

These generic modalities should be manageable and easy to integrate into the visual applications that the user creates.

With this approach, the developer does not need to know precise details about the technology that he or she is going to use. Thus, the generic modalities facilitate the development process.

There are various types of scenarios that could be developed by the gestures, gaze and silent speech generic modalities. The great advantage of using these generic modalities is that a modality used in an application in a specific scenario can be reused as many times as the user wants in another applications in different scenarios.

Additionally, the user can use the same gestures previously created in the new application systems or create new ones.

1.4 Objectives

Given its relevance, because it can reach a wide audience, the general goal for this research is to enable an easier exploration of silent aspects of interaction in multimodal contexts. This silent human-computer interaction should integrate established and novel modalities, such as gestures, gaze and silent speech.

Aligned with the general goal, the main objectives for this research work are:

- Propose a conceptual high level model for integration of silent interaction modalities in interactive systems;
- Design and develop a set of generic modalities to support silent interaction, aligned with the proposed vision, paying particular attention to gesture, gaze and silent speech interaction;
- Harness the proposed modalities with basic features enabling their use off-the-shelf;
- Develop a proof-of-concept application demonstrating the developed modalities.

1.5 Document structure

The remainder of this document is organized as follows. Chapter 2 presents an overview of background and related work regarding the topics covered by the work carried out, namely verbal and non-verbal communication, the W3C Multimodal Interaction Architecture, and interaction using gestures, gaze and silent speech.

Chapter 3 presents the input generic modalities for a silent communication, supporting gestures, gaze and silent speech.

Chapter 4 includes the results in this research work, such as some tests for the modalities capabilities, the proof-of-concept application developed using the three generic modalities and an evaluation by other developers.

Finally, chapter 5 presents the conclusions about this research work, summarizing and discussing the main contributions. It is concluded with some ideas deemed relevant for future work.

Chapter 2

Background and Related Work

This chapter introduces the background of a non-verbal human communication and how it can be transposed to human-computer interactions in digital devices.

Additionally, and considering the goals of this work, this chapter also presents some related work about gestures, gaze and silent speech interactions, technologies and a summary of recent research, deemed representative for each technology.

2.1 Verbal and non-verbal communication

Human communication is both social and cognitive because it is a process by which humans exchange information and influence one another through a common system of symbols and signals (Higgins and Semin, 2001).

Conn (2016) refers that humans often display other social communication skills before starting to talk. He also added that "two-day-old newborns are sensitive to visual cues with purposeful movements suggesting that a primitive form of sensory–motor association is already present. Then, during the first year of life, infants begin to obtain a social sense." A communication between two people can occur, linguistically, with arbitrary acoustic conventions or, non-linguistically, with iconic gaze or gestures (Zuberbühler and Gomez, 2018).

Verbal communication refers not only to the spoken or written transmitted messages but also to the behaviorally or visually ones (Hassan and Abdulhussain, 2018). Knapp et al. (2013) explain that during human interactions, the messages carried by verbal forms account for, approximately, **35%** where the rest is all conveyed through non-verbal means. In Figure 2.1 it is illustrated the approximated values, in percentage, of verbal and non-verbal messages during a human-human communication.

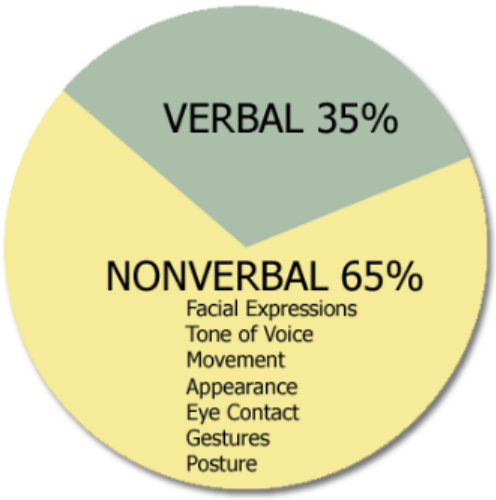


Figure 2.1: Verbal and non-verbal aspects involved in a human-human communication. Non-verbal aspects have a strong predominance. From *Non-Verbal Communication* (2014).

It is possible to infer that all verbal communication involves in some way some non-verbal cues, such as body gestures, facial gestures and eye movements.

Non-verbal communication is also an important part of how people communicate and interact with each other. It can be performed by several ways, such as hands and arms gestures, touch and eye contact/gaze (Hamdan, 2016).

A non-verbal communication accounts for **two thirds** of all communication. The main expressions and feelings are transmitted to others by non-verbal signals. They have such a relevant meaning and a great impact on a conversation because they represent our ways of expressing ourselves, our ways of interacting and the way we speak to others. A non-verbal communication can convey a message with the correct body signals (Eaves, D. Leathers, and D. G. Leathers, 2017).

It is illustrated, in Figure 2.2, some examples of non-verbal communications using body gestures.

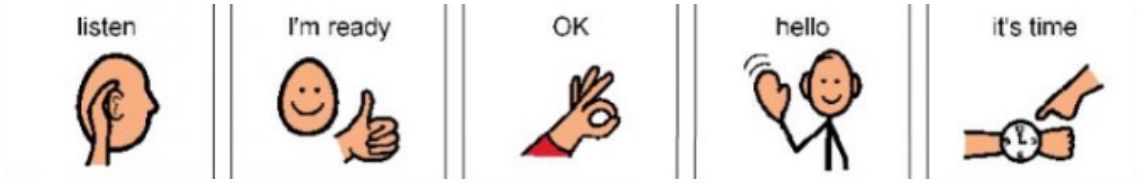


Figure 2.2: Some examples of non-verbal communications using body gestures. Adapted from *Sign Language for Children with Autism* (2015).

Dorai and Webster (2015) states that a non-verbal communication enhances a first impression in, for example, a job interview: the main impressions are formed within the first 4 seconds of contact. Also, they refer that the five senses used in interactions for one person to another or to a group is, approximately, 83% sight, 11% hearing, 3% smell, 2% touch and 1% taste.

2.2 Gestures

This section starts by analyzing a human-human communication by gestures and it shows how this form of non-verbal communication is important to a communication between humans.

It is also presented how gesture communications can be integrated in interactive application systems. To do so, a literature research was made.

2.2.1 Background

Often called "body language", gestures are a type of a non-verbal communication between humans. Non-verbal communication plays an important role in our everyday interactions and the bodily gestures are one of the several ways that people communicate with each other.

During a communication between humans, one person can send thousands of non-verbal messages to another one (Angraini, 2017). Most of the body gestures that a person uses to interact with another one are most likely to occur when a receiver is visibly present. The most common gestures performed in a human-human interaction are the hand movements, gestures that represent the train of thought, the pointing and shrugging. This form of non-verbal communication emphasizes the messages that are being communicated.

Integrate this type of interaction to a human-computer interaction involves some aspects that must be taken into account. In order to a computer recognize that a gesture was performed by a user, a process of gesture detection and recognition must be performed with machine learning algorithms. Detection errors can happen, such as for example, if the user performs a specific gesture, but the computer recognizes some other gesture with a higher predicted percentage, the output will be different than the desired by the user.

Figure 2.3 shows an example of a gestures interaction, where a person is interacting with a display in front of a MS Kinect One camera. The person is pointing his right hand to the intended option from the available options in the screen.

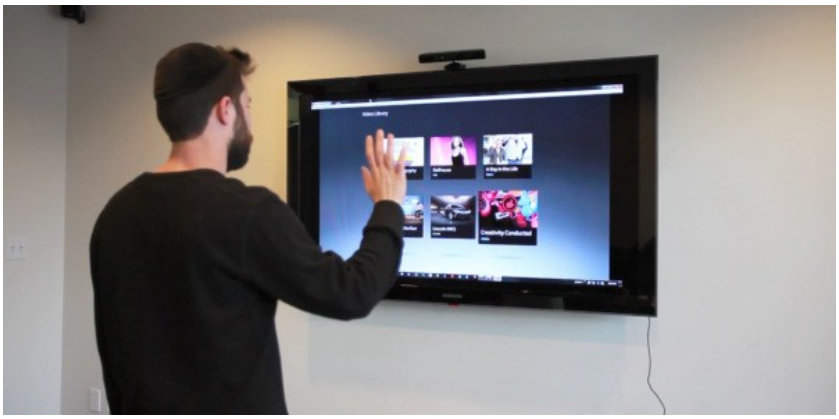


Figure 2.3: An example of a gestures interaction, where a person is interacting with a display in front of a MS Kinect One camera. The person is pointing his right hand to the intended option from the available options in the screen. From *Kinect socket server hack brings gesture recognition to Flash, Silverlight, and more* (2011).

2.2.1.1 Kinesics

Kinesics is the study of the human movement, gestures and posture. It refers to the study of the hands, arms, body and face movements. This section will outline the use of body gestures, arm and hand movements as a non-verbal communication.

It is shown in Figure 2.4 some examples of non-verbal gestures that can be used in a human-human communication.

Figure 2.4: Examples of non-verbal gestures that can be used in a human-human communication. From *Oxytocin Improves Social Cognition In Schizophrenia: Bipolar Network News* (2013).

The main difference between hands and arms gestures and other body languages is that the rest of the body can indicate more general emotional states and the hands and arms gestures can have more specific linguistic content, i.e., they have a greater association with speech and language. This type of gestures is the most used when talking about human-computer interactions, because they are easy to perform and intend to transmit a more natural and intuitive non-verbal communication. This way, Ende et al. (2011) divided this type of hands and arms gestures into five categories:

- Symbolic gestures: are gestures that have firm meanings and that are used in our everyday experiences;
- Interactional gestures: are gestures that are used to interact with another person. They can be performed in order to initiate an interaction, maintain, invite, organize or terminate a specific interaction behavior;
- Pointing gestures: are gestures that refer to objects or humans that are in the environment;
- Side effects of expressive behavior: are gestures that do not have any specific meaning in the interaction but they are performed while communication with other people, like motions of hands, arms and face;

- Body/manipulator motions: are gestures that are not employed to communicate or interact with someone but they are simply instances of effects of human motion.

Figure 2.5 shows an example of a human-human communication using hand gestures.

Figure 2.5: Example of a human-human communication using hand gestures. From *The Fascinating Science Behind 'Talking' With Your Hands* (2016).

2.2.1.2 Structure of gestures and its properties

When performing a gesture, one must consider its structure and its motion from the beginning to the end phases of the gesture. In the research work of Ende et al. (2011) it is presented the structure of gestures that includes four phases: one basic position phase and three temporal phases:

- Basic position: represents the person's initial position without starting performing the gesture;
- Preparation: consists of moving the arms to the start of the stroke and to the rest position. They are composed by smooth transitions;
- Stroke: represents the non-verbal message that a person wants to transmit with the gesture. It is the most meaningful and effortful part of the gesture;

  Indicators for the stroke phase (from electromagnetic trackers) are strong acceleration of hands, rapid changes in movement directions and strong hand tensions (Wachsmuth and Kopp, 2001);
- Retraction: consists of moving the arms to the end of the stroke and from the rest position. They are composed by smooth transitions.

It is illustrated in Figure 2.6 the structure of gestures with the four phases detailed above.

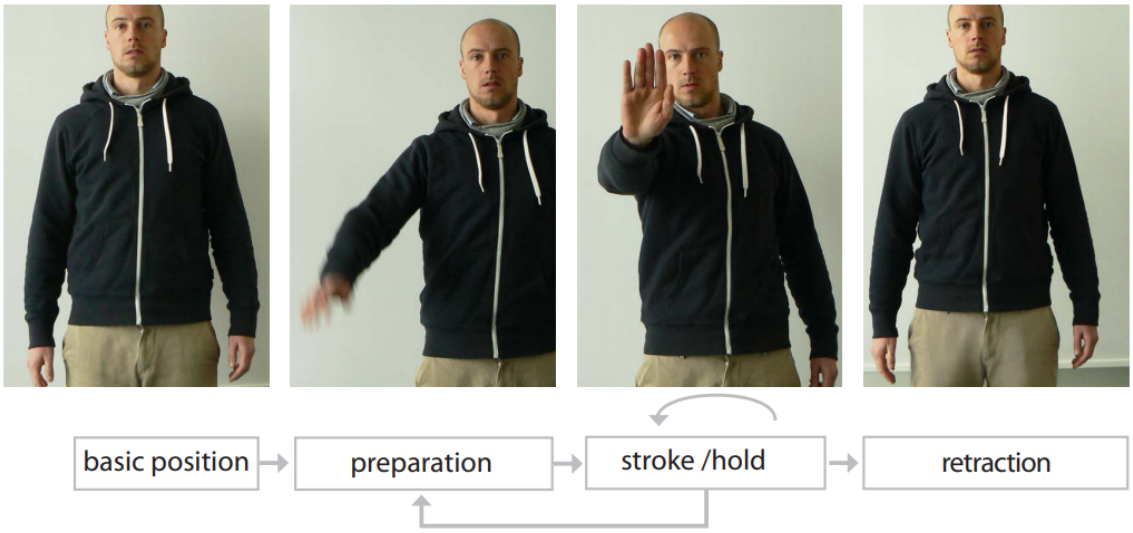


Figure 2.6: Structure of gestures with four phases: basic position, preparation, stroke/hold and retraction. It is observed that a repetitive gesture behavior can occur either from the stroke phase to the preparation phase or only within the stroke phase. From Ende et al. (2011).

The gestures performed by a person also follow some specific properties, such as the body parts involved in the gesture, the form of the movement (sinusoidal, elliptical, angular or straight) and the plane of the movement (sideways, transverse, towards or away from the auditor). The speed of the gesture (if it has abrupt transitions), the size of the radius of the movement and from which body part the movement started are other gestures' properties that must be considered into account.

2.2.2 Gestures interaction

Gestures are a type of an intuitive non-verbal human communication and can be used in human-computer interaction applications. Gestures are characterized as intended movements with a manipulative or a communicative character (Lambrecht, 2019).

Manipulative gestures involve a direct interaction with real or virtual objects through user movements.

A command gesture with a **communicative** character consists of a single pose, e.g. pointing towards a certain direction.

Figure 2.7 illustrates an example of two people interacting with an interactive system in front of a MS Kinect One camera using gestures.

Figure 2.7: Two people interacting with an interactive system in front of a MS Kinect One camera using gestures. From *First impressions: Microsoft's Kinect gaming system - CNN.com* (2010).

Gestures can be divided into two types:

- Online gestures: are direct manipulation gestures. They are used, for example, to scale or rotate a tangible object.
- Offline gestures: are gestures that are processed after the user interaction with an object. For example, the gesture to activate a menu.

Online gestures are real time gestures and they are processed at the same time of the interaction. There could be more challenges in these types of gestures due to the higher difficult recognition of gestures in image processing applications.

2.2.2.1 Advantages vs Disadvantages

There are presented some advantages of using gestures interaction in application systems (Attwenger, 2019):

Immediate and powerful interaction: Gestures do not interrupt the activity by forcing the user to move his or her hand to the location of a command. They can be performed directly from the current cursor position.

Intuitiveness and enjoyability: Gestures mirror the user experiences in the real world and they feel very natural to perform.

There are presented some disadvantages of using gestures interaction in application systems (Attwenger, 2019):

Memorability: Gestures need to be known and remembered before executing them.

One possible solution to create memorable gestures is to make them as intuitive as possible.

Fatigue: Gestures normally require muscle tension and complex movements over a long period of time and that can be exhausting.

One approach is to create gestures that are quick and comfortable to execute.

Distraction: It can occur distractions by the user, due to the higher comfort in these interactions and their naturalness. For example, performing a gesture with a larger angle from the original.

2.2.2.2 Limitations

There are some limitations when talking about gestures interaction systems, some of them already presented in the disadvantages subsection 2.2.2.1.

One must deal with limitations associated with the equipment used and lighting conditions. If there are no sufficient lighting conditions in the environment, the accuracy and the usefulness of the gestures recognition software would not be the same as expected. Additionally, items in the background or distinct features of the users may make recognition more difficult.

The external factors during a gestures interaction are another significant limitation. The presence of another person, for example, can interfere with the interaction and make the recognition more difficult. The same situation can occur in public installations, with interactive public displays.

It is illustrated in Figure 2.8 an example of a human-computer gestures interaction in a public large display.

Figure 2.8: A human-computer gestures interaction in a public large display. From *GestureTek Combines Gesture Control with Immersive Virtual Reality to Enhance Rehabilitation* (2018).

2.2.2.3 Evaluation

Gestures interactive systems can be seen as a way for computers to begin to understand human body language and build a richer communication between the human and the computer.

It is increasingly being used for several interaction scenarios, in home or in public installations, and for all types of users.

There are presented two research works that use gestures interaction in their systems:

A Web-based Real-Time Kinect Application for Gestural Interaction with Virtual Musical Instruments:

Zlatintsi et al. (2018) presents a web-based real-time application system that enables gestures interaction with virtual musical instruments. The user performs gestures in front of a MS Kinect One camera in order to virtually play a musical instrument.

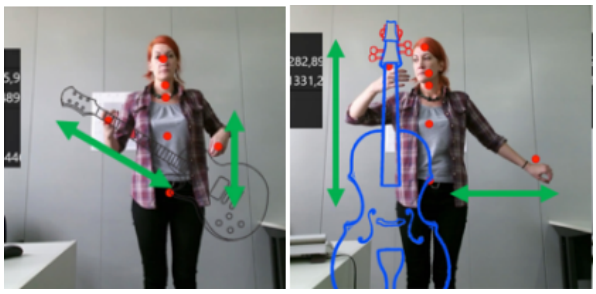
The feedback results were positive, as most users said that the system was enjoyable, the gestures were intuitive and the recognition had good accuracy levels.

A 3D playful framework for learning the components of a wind turbine using Kinect:

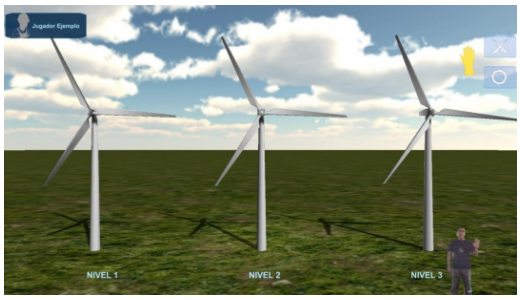
In the research work of Reyna et al. (2018), it is presented a playful 3D platform for students to learn the interior and exterior components of a wind turbine. It contains a set of games that students can interact with gestures in front of a MS Kinect One camera.

It was concluded that the application system was functional and intuitive due to the Kinect natural interaction. The students were able to learn more about wind turbines, their component parts and coupling.

It is presented in Figure 2.9a and Figure 2.9b two virtual musical instruments used in the first research work's system and the scene when a player starts a session from the second research work, respectively.



(a) Two virtual musical instruments.



(b) The scene when a player starts a session.

Figure 2.9: Figure (a), from Zlatintsi et al. (2018), shows two virtual musical instruments used in the first research work's system and Figure (b), from Reyna et al. (2018), illustrates the scene when a player starts a session from the second research work.

2.2.2.4 Technologies

The gesture recognition technology has been considered to be one of the highly successful technologies as it is possible to recognize any type of gestures in a very few time. Gestures recognition can be conducted with techniques from computer vision and image processing.

With the MS Kinect cameras, it is possible to capture body and hand motions in real-time, freeing all types of users from keyboards and computer mouses.

Another approach goes to the use of Leap Motion, which consists of a sensor that detects hand and finger motions as input. It allows control computer devices and also allows for hand tracking in virtual reality.

The areas where gestures technologies are applied the most are the educational and research areas, psychological, clinical and also marketing fields.

Figure 2.10 illustrates the different sensors that the MS Kinect One camera provides.

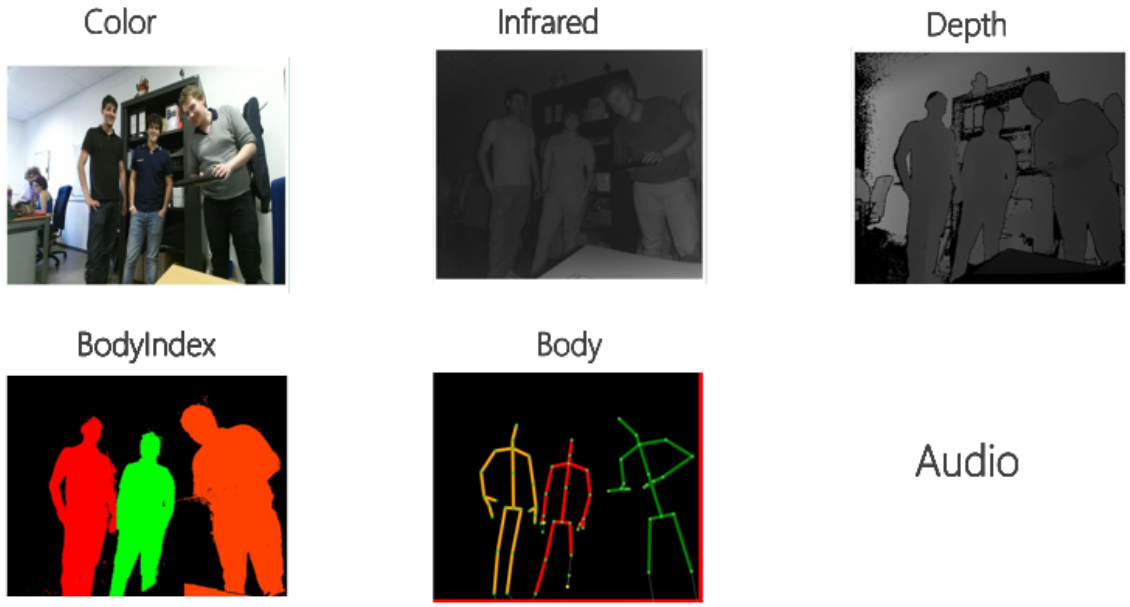


Figure 2.10: The different sensors that the MS Kinect One camera provides. Adapted from Teixeira, M. Silva, and S. Silva (2019).

2.2.2.5 Literature research

In the next two pages there are presented some research works about gestures interaction systems and it is detailed in Table 2.1 and in Table 2.2 the description of the interaction systems.

The tables are organized by year, title of the research work, the target audience / scenarios, types of operations of the system, the hardware devices used and its features.

Table 2.1: Gestures interaction systems ordered by year, title of research work, target audience / scenarios, hardware devices and types of operations of the system.

Year	**Title**	**Target audience / scenarios**	**Hardware devices**	**Types of operations**
2018	A Web-based Real-Time Kinect Application for Gestural Interaction with Virtual Musical Instruments.	People musically educated.	MS Kinect camera.	The user performs body gestures in front of a MS Kinect camera in order to virtually play a musical instrument.
2018	A 3D playful framework for learning the components of a wind turbine using Kinect.	Students that want to learn the components of a wind turbine.	MS Kinect camera.	The user interacts with the system in front of a MS Kinect camera playing a set of games in order to learn the components of a wind turbine.
2018	Improving Kinect-2 based User-Interface Interaction.	Physically impaired people. Smart homes.	MS Kinect camera.	The user performs simple daily tasks at home in front of a MS Kinect camera.
2017	Social Development for Children with Autism Using Kinect Gesture Games: A Case Study in Suzhou Industrial Park Renai School.	Children with autism.	MS Kinect camera.	The user plays games in front of a MS Kinect camera using body gestures. Examples of games are "Fruit Ninja" and "Motion Sports".

Table 2.2: Gestures interaction systems ordered by year, title of research work, hardware devices and its features.

Year	**Title**	**Hardware devices**	**Features**
2018	A Web-based Real-Time Kinect Application for Gestural Interaction with Virtual Musical Instruments.	MS Kinect camera.	Full HD RGB video: 30 fps. 25 joint coordinates extracted.
2018	A 3D playful framework for learning the components of a wind turbine using Kinect.	MS Kinect camera.	Distance from Kinect sensor: between 1.2 and 3.5 meters. Height of Kinect sensor: between 0.8 and 1.6 meters.
2018	Improving Kinect-2 based User-Interface Interaction.	MS Kinect camera.	Distance from Kinect sensor: 1.4 meters. Height of Kinect sensor: 1.45 meters. Range: 10° downwards.
2017	Social Development for Children with Autism Using Kinect Gesture Games: A Case Study in Suzhou Industrial Park Renai School.	MS Kinect camera.	RGB video: 30 fps.

2.3 Gaze

This section starts by analyzing a human-human communication by gaze and it shows how this form of non-verbal communication is important to a communication between humans.

It is also presented how gaze communications can be integrated in interactive application systems. To do so, a literature research was made.

2.3.1 Background

A communication between two or more people includes verbal and non-verbal features in order to have a natural interaction with each other. The eyes give information to humans about the other's feelings, thoughts and intentions-to-act, referred by Jording et al. (2018). The eyes represent the main coordination point of communication, as its gaze provides information not only from the environment but also of other people and our own attention and following intentions-to-act.

When talking about non-verbal gaze communication, the first point to consider is the eye contact. The eye contact can provide us information about some kinds of qualities or social norms that are often associated with gaze. Korkiakangas (2018) states that "much credence has been given to the idea that we can simply 'read' someone from their eyes – as the familiar proverb, 'the eyes are the window of the soul'." This can lead to a better notion of a body language in a simple communication. The body language can be misunderstanding as only body movements, postures and proxemics[1] but the gaze behaviours also have a very important role in it (Korkiakangas, 2018).

It is shown in Figure 2.11 an example of a human-human communication using eye contact.

Figure 2.11: Example of a human-human communication using eye contact. From *Support groups vital for mental health* (2016).

The non-verbal gaze communication expresses various types of feelings, such as friendliness, honesty, sincerity, openness, among others. On the other hand, our eyes can also express

[1]The study of how humans use space during a communication. From *Proxemics - Wikipedia* (2019).

feelings like wonderment, anxiety and fear (Capriola-Hall et al., 2018). This type of communication is also known as a two-way communication as we are constantly "sending" and "receiving" information without a conscious state of mind of it (Ganea et al., 2018).

2.3.1.1 Oculesics

Oculesics, a subdivision of kinesics[2], is the study of the role of eyes in a non-verbal communication. It comprises the eye movement, eye behavior, gaze and eye-related non-verbal communication. Oculesics include 4 dimensions: the levels of the eye contact, the eye movement, the pupil dilation and the gaze direction.

The **eye contact** can be direct or indirect. The direct eye contact happens when the eyes of one person are fixated on the eyes of the other person who is interacting with. This eye contact can show interest, attention and involvement of the discussion to the other person (Angraini, 2017). It is also important in maintaining the flow of the conversation and for gauging the other person's response (Hamdan, 2016). On the other hand, the indirect eye contact happens when two people are interacting with each other but not locking both of their eyes.

The **eye movements** occur by changing the eyes direction, changing the focus (the luminosity affects the eyes greatly when focusing on an object) or following a moving object (Van Der Stigchel, Meeter, and Theeuwes, 2006).

The **pupil dilation** is responsible by the changing in size of the pupil, voluntarily or involuntarily. This change occurs in situations of alertness, focus and mental effort (Tummeltshammer, Feldman, and Amso, 2018).

The **gaze direction** is defined by the actions of looking while talking, looking while listening, frequency of glances, patterns of fixation and blink rate (Angraini, 2017).

It is demonstrated an example of an oculesic (eye behavior) from Leontovich and Gulyaeva (2018) where there is an eye aversion from an obtrusive interlocutor, demonstratively closed eyes when the communicator is not inclined to engage in the conversation or avoiding the eye contact:

"Maybe we can catch up properly later? With the others?"

"Uh-yeah." Emily nods without looking me in the eye.

"Why is she being so off? What's wrong?"

2.3.1.2 Types of eye movements

Humans have the ability to point their eyes at target locations of interest, either consciously or unconsciously. This leads us to 5 types of eye movements: fixations, saccades, smooth pursuits, vergences and vestibulo-ocular movements.

Fixations represents points of attention, where the gaze is held within 1^{o} of the visual field for a duration of at least 100-300 ms (Toh, Rossell, and Castle, 2011). This type of eye movements corresponds to the most of viewing time (about 90%) and it is an important feature of looking for researchers to analyze and to make inferences about cognitive processes. Duchowski (2018) characterizes fixations by "tremor, drift, and microsaccades

[2]The study of non-verbal communication related to the facial expressions and gestures. From *Kinesics - Wikipedia* (2019).

which are used to stabilize gaze on the point of interest on the one hand, but keep the eyes in constant motion on the other, so as to prevent adaptation".

The aim in a human-computer interaction is to reconstruct these meaningful eye movements as faithfully as possible through a fixation filter where fixations generate proper events by our visual system ("Eye tracking technology for research - Tobii Pro" 2015).

During an interaction, this type of eye movement can reveal information about the person's attention, visibility, mental processing and understanding.

Saccades are rapid eye movements between fixations, shifting the focus from one point to another. This occurs when the fovea[3] is repositioned by large jumps of the eyes and saccade amplitudes generally range between 1^{o} and 45^{o} of the visual angle, with the head starting to rotate at about 30^{o} (Duchowski, 2018).

By measuring the latency of the first saccade to a given stimulus, it is possible to infer the person's attentional engagement (early processing) and by measuring the saccade latency away from a stimulus can provide information about the person's attentional disengagement (late processing), referred by Kerr-Gaffney, Harrison, and Tchanturia (2018).

It is illustrated in Figure 2.12 an example of an alternating sequence of fixations and a saccade.

Figure 2.12: The point 1 is a fixation point (the focus point where a person is looking at) and a saccade is performed towards the point 2 (the second fixation performed). From "Eye tracking technology for research - Tobii Pro" (2015).

[3]It is located in the center of the *macula lutea* of the retina and it is responsible for the sharp central vision. From *Fovea - Wikipedia* (2019).

Smooth pursuits are voluntary gaze movements that trace the eyes following moving objects. The eye velocity is most often less than 30 deg/sec.

In this type of eye movements it is difficult to trigger smooth pursuit events voluntarily in absence of a moving target. Vidal, Bulling, and Gellersen (2013) used the Pearson's product-moment correlation[4] to detect a synchronization between a person's visual pursuit of a moving object and its trajectory.

In Figure 2.13 it is illustrated an example of a pursuit eye movement.

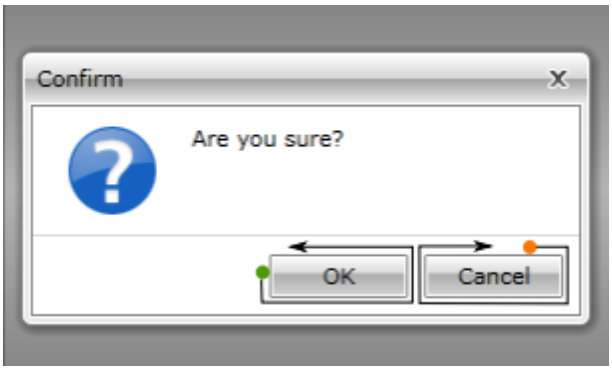


Figure 2.13: The hospital screen control. This is an example of how smooth pursuit movements can be used to control static desktop interfaces in touch-sensitive environments (Vidal, Bulling, and Gellersen, 2013).

Vergence movements are defined by the ability of the eyes to focus on different objects, i.e., they align the fovea of each eye with targets located at different distances from the observer, referred by *Types of Eye Movements [Saccades and Beyond] - iMotions* (2019). It involves the eyes moving in synchronously opposite directions to allow them both to point at the same position in a visual scene.

These eye movements can be classified into two types: (1) far-to-near focus triggers convergent movements and (2) near-to-far focus triggers divergent movements. This type of eye movements are generally slower than saccades.

In Figure 2.14 it is illustrated the model of a vergence eye movement.

[4]It attempts to draw a line of best fit through the data of two variables. From *Pearson Product-Moment Correlation - Laerd Statistics* (2018).

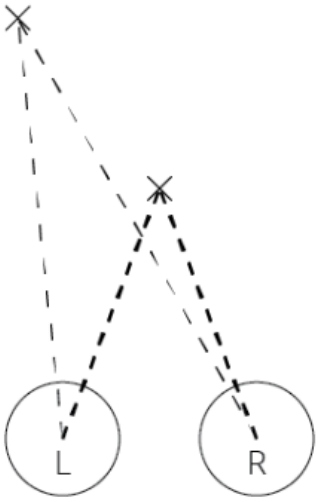


Figure 2.14: Model of a vergence eye movement (*Types of Eye Movements [Saccades and Beyond] - iMotions* 2019).

Vestibulo-ocular movements are what allows people to see clearly even when moving their heads. They can be defined by fixating an object with the eyes and moving the head from side to side.

The eyes automatically compensate for the head movement by moving the same distance but in the opposite direction. This occurs because the vestibular system detects brief and transient changes in head position, sends signals to the various parts of the brain and then produces rapid corrective eye movements (*Types of Eye Movements [Saccades and Beyond] - iMotions* 2019).

In Figure 2.15 it is illustrated the model of a vestibulo-ocular eye movement.

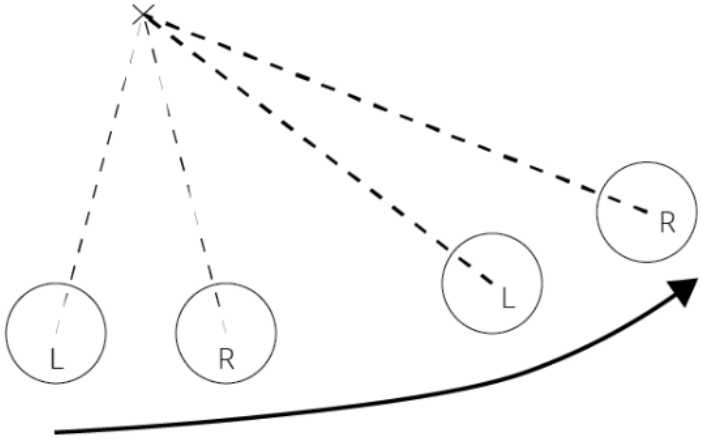


Figure 2.15: Model of a vestibulo-ocular eye movement (*Types of Eye Movements [Saccades and Beyond] - iMotions* 2019).

2.3.2 Gaze interaction

A gaze interaction is categorized into four forms: diagnostic (off-line measurement), active (selection, look to shoot), passive (gaze-contingent displays) and expressive (gaze synthesis).

Due to its offline requirements, the **diagnostic** interaction is the main category of eye-tracked applications, including the training or assessment of expertise, for example, in applications of flights or surgery training.

Active interaction is responsible for using the eyes to point and click, i.e., to gaze at a predefined location and generate a proper event for it.

Passive interaction corresponds to the manipulation of elements in a scene in response to a gaze direction.

The **expressive** interaction makes use of the model of the eye to draw its synthesis in tasks like reading (Duchowski, 2018).

This type of non-verbal communication determined by the direction of the eyes has positive influences on collaboration, as people can apply this communication modality to manage their interaction and to enrich their communication (Lankes et al., 2018). A gaze-based interaction should be intuitive, efficient and as most natural as possible.

A gaze interaction also has a great significance when feeling of a person's presence. The eye contact and gaze are considered to be connected to a mutual attention, i.e., when a person interacts with a computer, it is possible to infer that he or she is only interacting with it when there is a direct eye contact and his or her gaze is headed to the computer's display. On the contrary, if the person gazes to another direction other than the computer's display, we can say that the person lost his or her gaze (Krahmer, Swerts, and Shahid, 2012).

Figure 2.16 shows a surgery training example using augmented reality glasses with eye tracking technology.

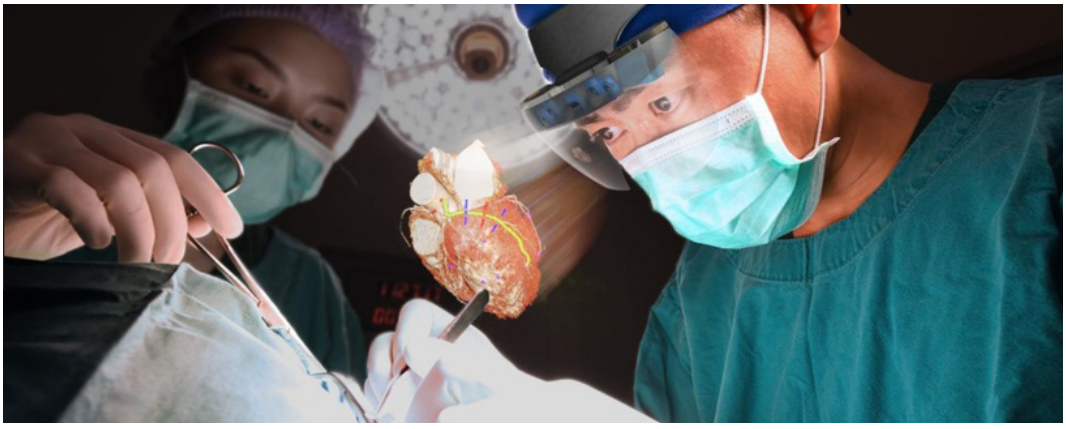


Figure 2.16: A surgery training example using augmented reality glasses with eye tracking technology. From *Ocutrx Vision Technologies Unveils Groundbreaking Oculenz AR Cellular with Eye-Tracking Glasses Design* (2019).

2.3.2.1 Gaze communication vs Gaze interaction

Gaze communication is, as it was previously defined, a non-verbal communication using only the eyes and the person's gaze. A person A can provide information to a person B or

to a group of people and convey his thoughts and feelings only using his or her gaze during a conversation or a public speaking (Korkiakangas, 2018).

Gaze interaction, in addition to gaze communication, also refers to the interaction itself in a particular situation, i.e., it highlights the feelings associated to a person.

The main difference between a gaze communication and a gaze interaction is that in the last one the actions to be performed during an interaction have a dependency on the person's gaze and on the expressions transmitted by non-verbal signals (Sibert and Jacob, 2000).

For example, in the research of X. Zhang et al. (2017), it is used an eye tracker device for interactions between the person and the computer replacing the normal mouse and keyboard. The users can use the single left-click, single right-click, double left-click, drag and scroll functions and even a virtual keyboard only by using their gaze on the computer screen and interacting with the computer.

Figure 2.17: A user interacting with a computer using only his gaze with a Tobii Eye Tracker device. From "Tobii and Microsoft Collaborate to bring Eye Tracking Support in Windows 10" (2017).

2.3.2.2 Advantages vs Disadvantages

It is presented some advantages of using gaze interaction in application systems (Vigo, 2013):

Ease of use: A gaze interaction is a handsfree interaction and there is no need of using the hands or speech to interact. This facilitates the communication not only for the people who have physical disabilities but also for people that do not want to overstress the hand muscles or to interact by voice at that moment.

Faster interaction: The interaction is faster due to the faster eye movements. Some interactions with only eye gaze as input could slower the interaction, such as virtual keyboards. The solution to speed up the interaction is to combine gaze with other modalities.

Hygienic interaction: It is possible to infer that gaze interactions are hygienic interactions because there is no need of touching in the device systems in order to perform an interaction, which that is not possible with a mouse or a keyboard, for example.

Over distance interaction: It is possible to have a remote control interaction with eye tracking systems. In Ferhat and Vilariño (2016), there are presented eye tracking devices that are able to detect the eyes within one meter of distance with high resolution camera lenses.

User's activity information: Tracking where a user is looking in a human-computer interaction is a great advantage to a further data analysis.

It is presented some disadvantages of using gaze interaction in application systems (Vigo, 2013):

Calibration and head control: A calibration is an important step to have an efficient human-computer interaction. Most of the eye tracking systems can not support a full range motion of the user's head and this limits the head and body movements.

A solution for it would be the system being able to auto-calibrate the user's eyes during the interaction regardless the position of his or her head.

Midas Touch: The Midas Touch problem happens when the user finds that everywhere he or she looks, voluntarily or involuntarily, a new function in the interaction is activated. The system must be capable of distinguishing between (1) gaze intended to gather only visual information and (2) gaze intended to invoke an action for interaction.

To combat this problem, dwell time and blinks are used as clicking modalities in many gaze-controlled systems, but it is still a topic for researchers.

Eyes control and environment: Even though we are able to control our eyes and our intended eye movements, as humans, we can not control the unconscious eye movements from external factors in the environment.

For example, if we are interacting with an eye tracking system and someone appears in the same environment and calls us, our eyes will instantly look at that person, unconsciously.

2.3.2.3 Limitations

Although gaze interaction systems contribute to efficient and natural interactions, there are some limitations when using them.

One of the limitations is the calibration process that every new user in the system must perform, because every person has his own eyes features. In addition, the head control is also important, because a system without auto-calibration does not support a full motion of the user's head during the interaction.

Another limitation is the one associated with the ambient lighting. The user's gaze prediction would be slightly different if there are not sufficient lighting conditions in the interaction environment.

2.3.2.4 Evaluation

As already talked about in the subsection 2.3.2.2, the main advantages in gaze interactions are the fact that it is easy to use (hands free interaction), it is a faster interaction due to the faster eye movements and it is an over distance interaction, allowing the user to interact with a system within one meter of distance.

Even though it is not a common method that people use to interact with digital devices, it is increasingly being used and the efficient rate has also very good values when comparing with other types of modalities.

There are presented two research works that use gaze tracking in their systems:

GazeDrone - Mobile Eye-Based Interaction in Public Space Without Augmenting the User:

Khamis et al. (2018) presents a system that a drone and a display are positioned in front of the user. The user interacts with the display's interface using his or her gaze and the gaze drone detects and estimates the user's gaze in stationary and movement modes.

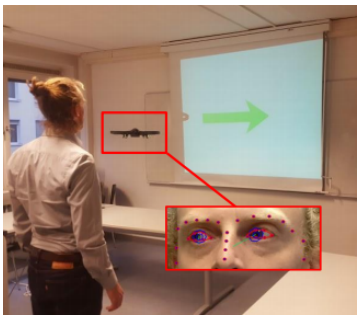
The best results were for the stationary mode for the selection experience where the success rate was 75,13% and the worst results were for the parallel and orthogonal movement modes in which the success rates were 46,92% and 61,27%, respectively. These last results occurred due to the motion blur and the incorrect input of the participants when they were moving in parallel from the display.

Efficient Eye Typing with 9-direction Gaze Estimation:

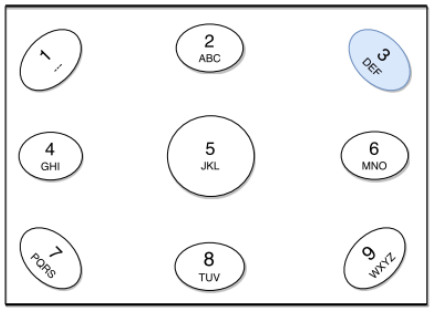
In the research work of C. Zhang, Yao, and Cai (2017) it is presented a system that has an input text method based only on the user's gaze. The user looks and interacts with a display, which is divided in 9 different regions, of a virtual keyboard and selects the letter that he or she pretends with the voluntary blinking of the eyes.

The best results using both eyes were for the models using their data augmentation methods, i.e., people already calibrated in the eye tracking system. The success rate for this experiment was 99,80%. The worst results were the ones that did not use their data augmentation methods and the success rate was 78,34%.

It is presented in Figure 2.18a and in Figure 2.18b the gaze drone used in the first research work's system and the system interface of the second research work, respectively.



(a) Gaze interaction with the drone's camera.



(b) Interface when users look at number 3.

Figure 2.18: Figure (a), from Khamis et al. (2018), shows a gaze interaction with the drone's camera in the first research work's system and Figure (b), from C. Zhang, Yao, and Cai (2017), presents the system interface of the second research work.

It is possible to conclude that gaze tracking systems can have good accuracy results and they can provide efficient and natural human-computer interactions. However, there may be some problems that can affect the success rate of a system, such as ambient lighting, calibration and head control and the Midas Touch problem.

2.3.2.5 Technologies

Eye tracking is a technology that measures a person's eye movement to identify both where a person is looking at and the sequence in which the person's eyes are shifting from one location to another (Othman et al., 2017). This tracking can be done by a calibration process that varies from person to person (it depends on the person's eyes characteristics or whether the person uses glasses or not, for example). Eye tracking can provide information about points of interest in which the person's eyes are relatively stable (fixation) and rapid eye movements (saccade) from one fixation to another.

Eye tracking techniques vary from video-oculography VOG, video-based infrared IR to electrooculography EOG. In this research work it is used the video-based infrared IR eye tracking technique with the Tobii Eye Tracker 4C device. In order to estimate gaze positions of a user interacting with an eye tracker device, the pupil positions of the user are detected by the infrared camera of the eye tracker hardware device.

There are several interaction scenarios for this type of technology. The research, clinical, psychological and commercial applications are the most used. Some examples using eye tracking technologies are: (1) a research for tracking reading processes (Vigo, 2013), (2) an eye tracking system that enables an interaction while maintaining sterility and freeing the hands to manipulate surgical instruments in an operating room (Unger et al., 2019), (3) a system with an eye tracker mounted in a drone capable of detecting the user's eye movements in stationary and movement modes (Khamis et al., 2018) and (4) an eye tracking system that perceives the user's visual attention when booking a hotel room (Chiao Wang, Tsai, and Wei Tang, 2018).

It is illustrated in Figure 2.19 a Tobii Eye Tracker 4C device connected to a tablet during an execution of a gaze application.

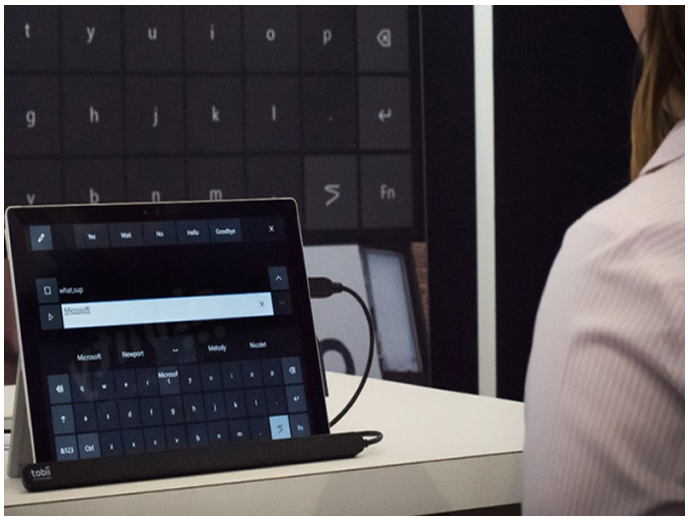


Figure 2.19: A Tobii Eye Tracker 4C device connected to a tablet during an execution of a gaze application. From *Windows 10 now supports eye tracking for greater accessibility* (2017).

2.3.2.6 Literature research

In the next two pages there are presented some research works about gaze interaction systems and it is detailed in Table 2.3 and in Table 2.4 the description of the interaction systems.

The tables are organized by year, title of the research work, the target audience / scenarios, types of operations of the system, the hardware devices used and its features.

Table 2.3: Gaze interaction systems ordered by year, title of research work, target audience / scenarios, hardware devices and types of operations of the system.

Year	**Title**	**Target audience / scenarios**	**Hardware devices**	**Types of operations**
2018	GazeDrone: Mobile Eye-Based Interaction in Public Space Without Augmenting the User.	People with motor disabilities. Elderly people.	Desktop computer. CCD camera. Infrared leds.	The user looks at a display and the drone is positioned in front of him/her. The user interacts with the display in stationary and motion modes and the drone detects the user's gaze in real time.
2018	Designing Gaze-based Interaction for Pervasive Public Displays.	Scenarios with large and public displays: shopping centers, train stations, airports, urban areas, universities and libraries.	MS Kinect camera mounted in a rail system. Tobii Rex Eye Tracker.	The user walks ahead of the camera and have to follow a red circle with his/her gaze. It is made in stationary and in motion modes and the eye tracker follows the user in real time.
2018	Non-invasive Gaze Direction Estimation from Head Orientation for Human-Machine Interaction.	People with motor disabilities. Elderly people.	MS Kinect camera.	The user follows a red dot projected on the wall up to 1 minute.
2018	Exploring Advertising Effectiveness of Tourist Hotels' Marketing Images Containing Nature and Performing Arts: An Eye-Tracking Analysis.	Tourists. Marketing scenarios.	Eye tracker.	The user looks for a display with an eye tracker detecting his/her gaze when navigating in a hotel website and simulating a booking.
2016	Low Cost Eye Tracking: The Current Panorama.	People with motor disabilities. People that suffer from amyotrophic lateral sclerosis. Elderly people.	Eye tracker.	The user interacts with a laptop or desktop computer with an eye tracker mounted in front of him/her in order to perform the calibration and the extraction of the results.

Table 2.4: Gaze interaction systems ordered by year, title of research work, hardware devices and its features.

Year	**Title**	**Hardware devices**	**Features**
2018	GazeDrone: Mobile Eye-Based Interaction in Public Space Without Augmenting the User.	Drone. Mobile Eye Tracker.	Distance: 60-90 cm. Video: 640×360 px at 7–9 frames per second. Aerius quad-copter: 3 cm × 3 cm × 2 cm.
2018	Designing Gaze-based Interaction for Pervasive Public Displays.	MS Kinect camera mounted in a rail system. Tobii Rex Eye Tracker.	Distance: 50 cm. Eye tracker range angle: 35^{o}-50^{o}.
2018	Non-invasive Gaze Direction Estimation from Head Orientation for Human-Machine Interaction.	MS Kinect camera.	Distance: 160 cm. Gaze margin: -48:37^{o} to 48:37^{o} in horizontal direction and -18:81^{o} to 21:34^{o} in vertical direction. Video: 30 frames per second.
2018	Exploring Advertising Effectiveness of Tourist Hotels' Marketing Images Containing Nature and Performing Arts: An Eye-Tracking Analysis.	Eye tracker.	Resolution: 1920 × 1080 pixels. Calibration of 9 points.
2016	Low Cost Eye Tracking: The Current Panorama.	Eye trackers.	Uses several estimation and calibration techniques (appearance-based, feature-based and model-based). Realization of various experiments.

2.4 Silent Speech

This section starts by analyzing a human-human communication by silent speech and it shows how this form of non-verbal communication is important to a communication between humans.

It is also presented how silent speech communications can be integrated in interactive application systems. To do so, a literature research was made.

2.4.1 Background

Human speech is a natural and efficient form of communication, but people with speech disorders, scenarios where there is a need for a silent and quiet interaction, or even in noisy environments could be a strong challenge for communicating with acoustic messages. Another advantage of using a human-computer silent speech interaction is that it provides more privacy when used as a computer interface (Freitas, Teixeira, Dias, et al., 2011).

One SSI (Silent Speech Interface) system can go through the implementation of a non-invasively silent speech interaction where a person simply "mouths" his or her speech only by lip movements (also called subvocal speech recognition where no voice is produced). Silent speech interfaces are insensitive to the ambient background noise due to the non-acoustically acquired speech cues.

It is illustrated in Figure 2.20 a diagram depicting the overview of the speech production process model and its phases described by Freitas, Teixeira, S. Silva, et al. (2016). The first phase converts the communication intentions into messages, the next phase corresponds to the sequence of phones (electrical impulses) that are fed to the articulators and the third phase represents the actual process of articulation. Finally, the last phase is composed by the effects that resulted from the previous phases, such as acoustic speech signal and alterations in the face, for example.

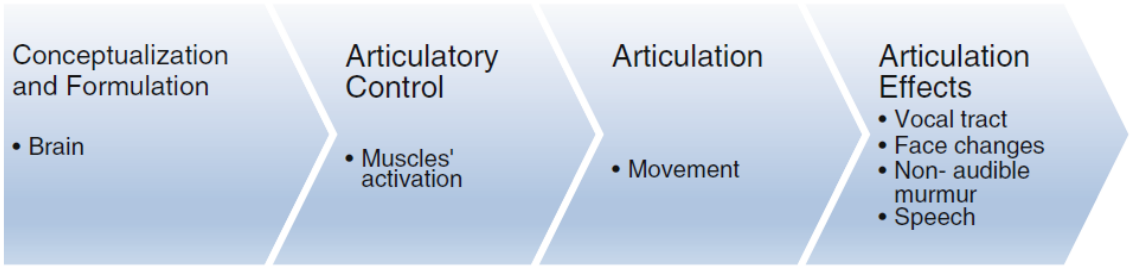


Figure 2.20: Overview of the speech production process model (Freitas, Teixeira, S. Silva, et al., 2016).

A wide range of biosignal modalities have been studied in order to develop non-acoustic speech communication systems. Denby et al. (2010) identified seven different approaches to SSI systems:

- Capture of the movement of fixed points on the articulators using Electromagnetic Articulography (EMA) sensors;

- Using ultrasound and optical imaging of the tongue and lips in a real-time characterization of the vocal tract[5];
- Digital transformation of signals from a Non-Audible Murmur (NAM) microphone[6];
- Analysis of the glottal activity[7] using electromagnetic or vibration sensors;
- Surface electromyography (sEMG) of the articulator muscles or the larynx;
- Interpretation of signals from electro-encephalographic (EEG) sensors;
- Interpretation of signals from implants in the speech-motor cortex.

Figure 2.21 illustrates sEMG setups for four different speakers.

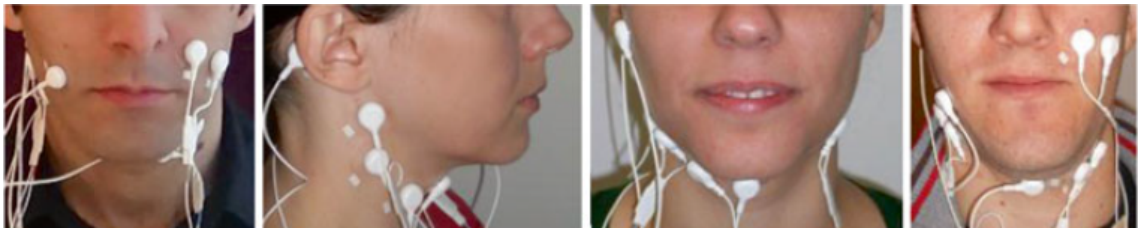


Figure 2.21: Different possibilities of sensor positioning in sEMG setups for four different speakers and highlighting the challenge of acquiring signals independent of speakers' anatomy. From Freitas, Teixeira, S. Silva, et al. (2016).

2.4.1.1 Techniques for extraction of face and lip features

The research work of Freitas, Teixeira, Dias, et al. (2011) presents three techniques for the extraction of the features of face and lips in silent speech systems: appearance-based, shape-based and motion features, which is the fusion of both.

The **appearance-based** method is based on the information extracted from the pixel intensity values around the mouth area of the image (Magre and Ghodke, 2019). It is assumed that all pixels extracted contain important information about the spoken utterance. This method is the most popular because it is efficient and simple to use.

Shape-based methods use predefined points on the lips contours, the distance between two points, the angle between three points on the lip and also parts of the face, such as cheek and jaw, as features. In order to extract attributes like the height, width and the mouth area, this method uses geometrical and topological aspects of the face.

One disadvantage of this method is that it depends greatly on the imaging conditions like illumination and view angle of the camera (Liew and Wang, 2008).

The **motion features** method is a combination of the appearance-based and shape-based methods and the features are extracted from the static frames (Fan et al., 2018). It combines features from the two previous methods to detect visual speech dynamics.

[5]The vocal tract is the area from the nose and the nasal cavity down to the vocal cords deep in the throat. From *The Vocal Tract - Pronuncian* (2016).

[6]Stethoscopic microphone that can capture very quietly uttered murmur. From Denby et al. (2010).

[7]The glottis is the opening between the vocal folds. As the vocal folds vibrate, the resulting vibration produces a "buzzing" quality to the speech, called voice. From Hayes (2009).

One advantage of using this method is that the features extracted are independent of the static image and background and they directly represent the mouth movement.

It is illustrated in Figure 2.22 a representation of 18 lips feature points and their assigned ID values (Yargic and Doğan, 2013).

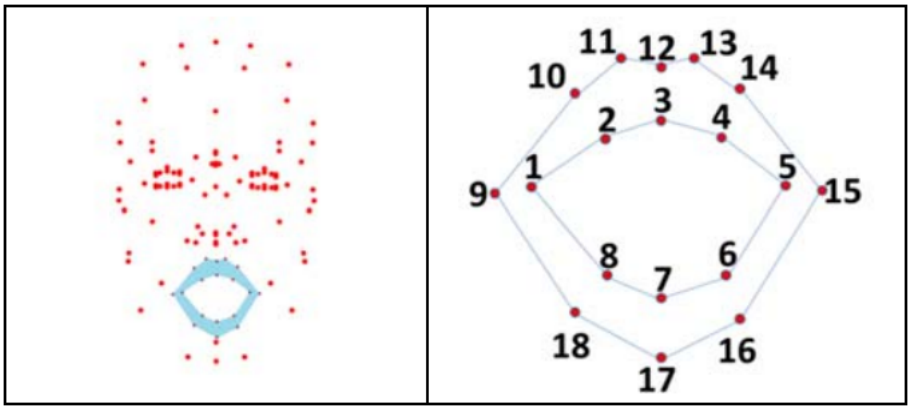


Figure 2.22: A representation of 18 lips feature points and their assigned ID values (Yargic and Doğan, 2013).

2.4.1.2 Classification processes

The process of classification in silent speech systems is important in order to classify the features extracted from the face and especially the lips. To do so, several machine learning algorithms (also called classifiers) are used to analyze the data received and output the matched results.

In this subsection it is presented some of the most used machine learning algorithms in the context of SSIs:

AdaBoost: is one of the most successful algorithms in the field of machine learning in recognition systems. It consists of building a strong classifier based on an amount of weak classifiers.

The AdaBoost algorithm combines a number of MLP (Multilayer Perceptron)[8] neural network based predictors to output final high-precision wind speed predictions (Liu et al., 2015).

Hidden Markov Model: is a model in which the system being modeled is assumed to be a Markov process with hidden states.

Each state has a probability distribution over the possible output tokens and the sequence of tokens that are generated gives information about the sequence of states (Zucchini, MacDonald, and Langrock, 2016).

Support-Vector Machine is a supervised learning model with associated learning algorithms that analyze data used for classification.

This model represents points in space, mapped so that the examples of the different categories are divided by a clear gap.

[8]MLP is a class of a feedforward artificial neural network. From *Multilayer perceptron - Wikipedia* (2019).

Sequential Minimal Optimization: is an iterative algorithm used for training support-vector machines. Its training is simpler than the SVM training.

It optimizes the quadratic programming problem through two steps: (1) identify and solve analytically the two Lagrange multipliers and (2) choose adequate Lagrange multipliers using heuristics (Pham et al., 2017).

***k*-Nearest Neighbor:** is a type of instance-based learning and it is the simplest of all machine learning algorithms.

The input consists of the k closest training examples in the feature space and the output is a class membership. An object is classified by its neighbors and it is assigned to the most common class among its k nearest neighbors.

Naive Bayes: is a highly scalable classifier that requires a number of parameters linear with the number of variables (features).

In this classifier, each variable can be independently estimated as a one dimensional variable and it often performs surprisingly well (H. Zhang et al., 2017).

Dynamic Time Warping: is a technique used to find an optimal match between temporal sequences, which may vary in speed.

One advantage of this method is that it provides temporal alignment to timevarying signals that have different durations (Freitas, Teixeira, Dias, et al., 2011).

Random Forest: is an ensemble learning method for classification that is based on the construction of decision trees at training time and then outputs the class that is the mode of the classes.

Random forest algorithms also correct for the overfitting problem to their training set.

It is illustrated in Figure 2.23 a k-Nearest Neighbor classification model example with two classes.

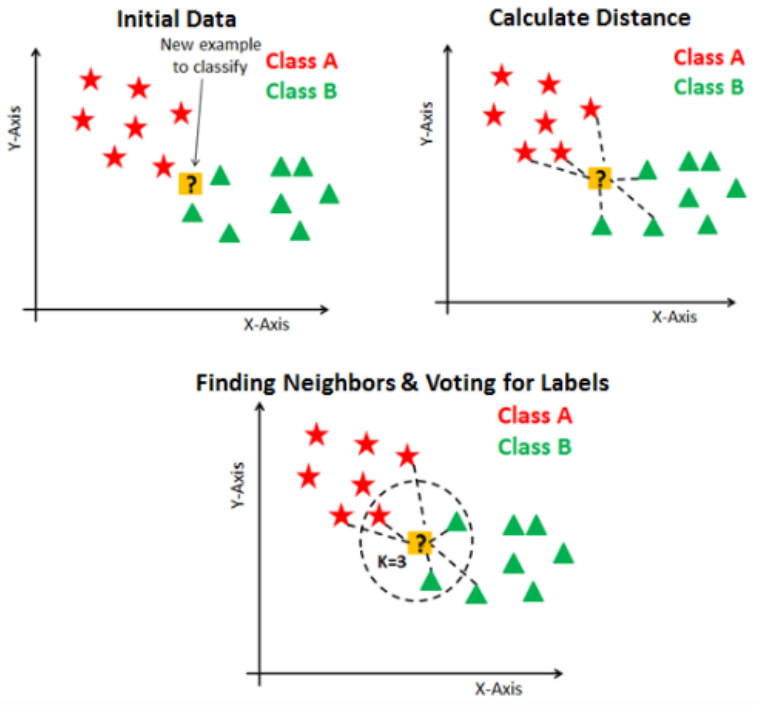


Figure 2.23: A k-Nearest Neighbor classification model example with two classes. From *Day 3 - K-Nearest Neighbors and Bias–Variance Tradeoff* (2018).

2.4.2 SSI interaction

Silent speech interactions have been increasingly used in quiet environments or environments where there is noise around. Even though humans are capable of producing and understanding uttered speech in both environments, the one that is more efficient is the quiet environment because the concentration levels of the person that is trying to lip reading are higher.

Thus, most people can also understand a few uttered words by lip reading (Yargic and Doğan, 2013) and many non-hearing people are more proficient at this skill. The goal of this research work is to convey this non-acoustic verbal communication to a human-computer interaction.

The introduction of silent speech interaction systems started in 1968 with the science-fiction movie "2001 - A Space Odyssey". In that movie, the conversations of the astronauts were lip-read by a "HAL 9000" computer (Denby et al., 2010). From there, silent speech systems have been able to recognize words and phrases with higher accuracy rates.

It is illustrated in Figure 2.24 an example of data acquisition for the tongue and lips (Freitas, Teixeira, S. Silva, et al., 2016).

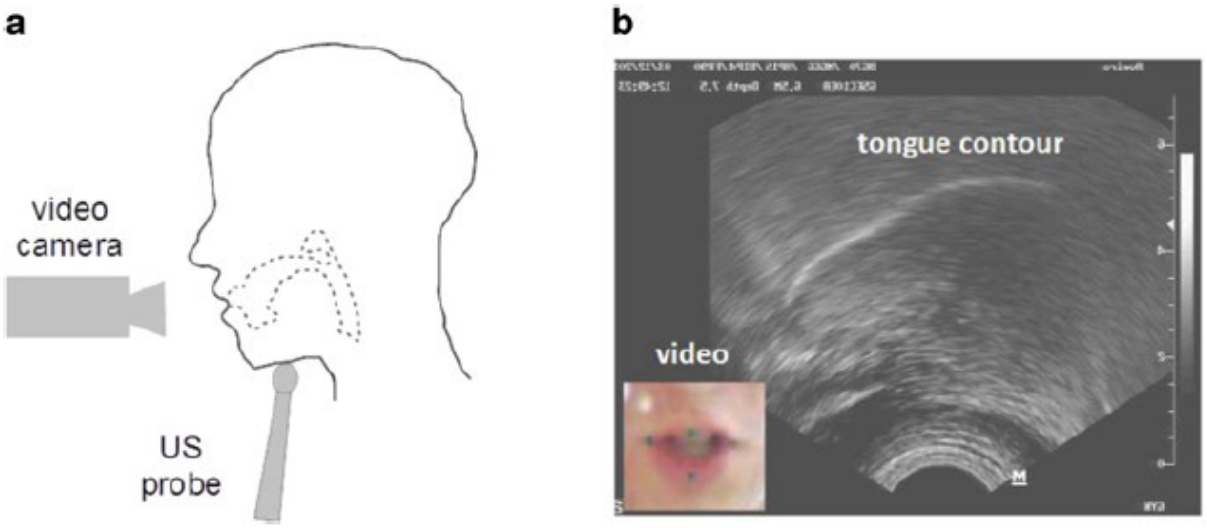


Figure 2.24: An example of data acquisition for the tongue and lips, from Freitas, Teixeira, S. Silva, et al. (2016). Figure a) illustrates an ultrasonic probe placed under the speaker's chin and a video camera capturing the lips. Figure b) illustrates an ultrasonic vocal tract image with embedded frontal lip view.

2.4.2.1 Advantages vs Disadvantages

It is presented some advantages of using silent speech interaction in application systems:

Quiet interaction: A silent speech interaction is a quiet interaction and there is no presence of acoustic signals.

Immediate and powerful interaction: Silent speech interaction systems do not interrupt the activity by forcing the user to move his or her hand to the location of a command. They can be performed directly from the current position. It is a handsfree silent interaction.

Over distance interaction: It is possible to have a remote control interaction with silent speech systems.

It is presented some disadvantages of using silent speech interaction in application systems:

Speaker independence: Silent speech systems rely on the speaker's lips anatomy. If a user tests a silent speech system with the trained words of another user, the recognition results will be lower.

Head control: Some systems can not support a full range motion of the user's head and this limits the head movements.

One solution is the normalization by distance of the features extracted from the user's lips for the words training process.

It is illustrated in Figure 2.25 an example of a silent speech interaction with a mobile phone device.

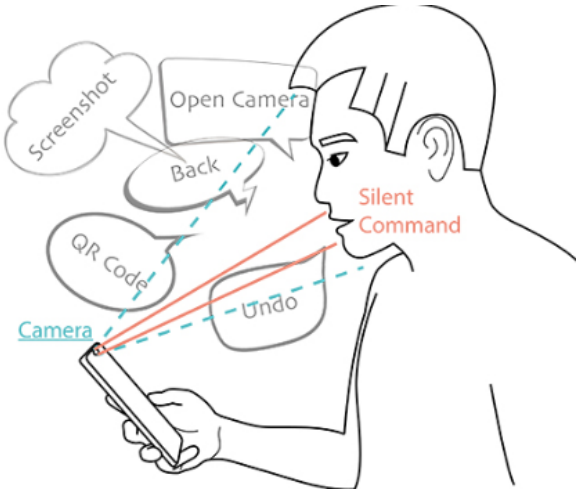


Figure 2.25: An example of a silent speech interaction with a mobile phone device. From Sun et al. (2018).

2.4.2.2 Limitations

Even though silent speech interface systems have great advantages of using them in human-computer interactions, one must consider some limitations that developers may face.

It is presented the common limitations for silent speech interaction systems (Denby et al., 2010):

Sensor positioning and robustness: In most of the silent speech technologies, the sensors used must be carefully positioned with the user. On the contrary, the accuracy levels could not be as high as expected.

Speaker independence: The silent speech systems depend on the speaker's anatomy and on the movements of his or her articulatory muscles.

Lombard effect: The lombard effect occurs when speakers are prone to articulate differently when they do not have auditory feedback of their own speech. Quiet or high-noise environments are examples of situations that this effect may occur.

Prosody and nasality: The silent speech systems must present a realistic output synthesis and it can be more difficult to achieve that because the glottal signal that is necessary for the pitch estimation is absent or substantially modified due to the no production of acoustic signals.

Dictionaries: Automatic speech recognition (ASR) systems are capable of recognizing words and phrases made by the developer with good accuracy levels. Silent speech interface (SSI) systems, due to of its complexity, focus on a limited vocabulary silent speech recognition.

2.4.2.3 Evaluation

Silent Speech Interfaces (SSI) are used for several interaction scenarios. Some examples are for users with speech impairments and for scenarios with environmental noise. Situations in which privacy, confidentiality or non-disturbance are also important.

There are presented two research works that use silent speech interaction in their systems:

Deep Lip Reading: a comparison of models and an online application:

Afouras, Chung, and Zisserman (2018) presents a system that includes three architectures: (1) a current model using LSTMs[9], (2) a fully convolutional model and (3) a proposed transformer model.

The user reads a set of texts in front of a camera during a silent speech interaction in the three architecture models proposed.

The results were better for the proposed transformer model which obtained a 50% word error rate, lower by 20% than the worst result (recurrent model using LSTMs).

AlterEgo: A Personalized Wearable Silent Speech Interface:

In the research work of A. Kapur, S. Kapur, and Maes (2018) it is presented a wearable interface that allows a user to silently interact with a computing device and the communication is bidirectional. The user's data are extracted by neuromuscular sensors and the device's response is heard by the headphones.

To test and evaluate the system, 10 participants were recruited to read a set of texts, as if they were reading a book without moving their lips, and the results averaged 92.01%.

It is presented in Figure 2.26a and in Figure 2.26b the lip reading model used in the first research work and the AlterEgo wearable of the second research work, respectively.

(a) The lip reading model.

(b) The AlterEgo wearable.

Figure 2.26: Figure (a), from Afouras, Chung, and Zisserman (2018), shows the lip reading model used in the first research work's system and Figure (b), from A. Kapur, S. Kapur, and Maes (2018), shows the AlterEgo wearable.

[9]Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture used in the field of deep learning. From *Long short-term memory - Wikipedia* (2019).

2.4.2.4 Technologies

Silent speech interface systems have been created using ultrasound devices and optical camera input of tongue and lip movements.

Some techniques for the extraction of the tongue, lips and other facial features in order to develop a silent speech interface interaction system are: (1) the use of electromagnetic devices, (2) the use of the electromyography of speech articulator muscles and the larynx and (3) the use of non-audible murmurs.

The technology that will be used in this research work is the silent speech interface based on the detection and recognition by a MS Kinect One camera (it features a RGB camera and a depth infrared sensor).

Figure 2.27 illustrates an example of surface electromyography electrodes positioning and the respective channels (1–5) plus the reference electrode (R) (Freitas, Teixeira, S. Silva, et al., 2016).

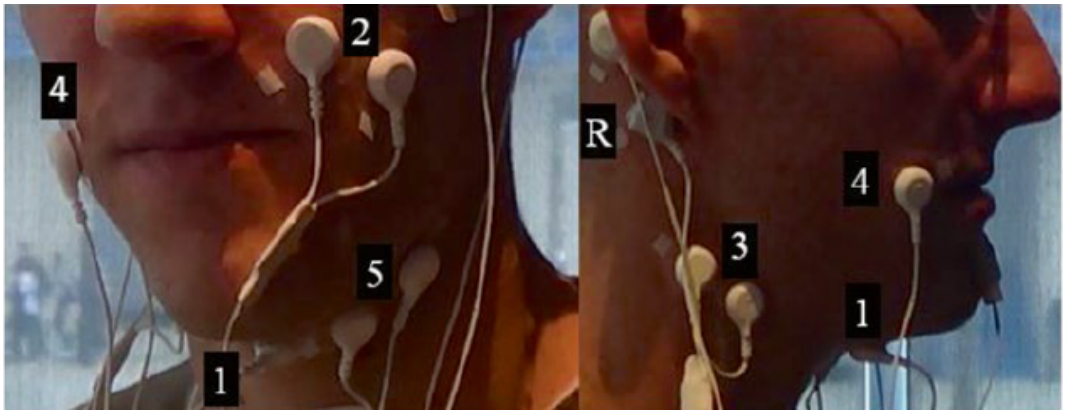


Figure 2.27: Surface electromyography electrodes positioning and the respective channels (1–5) plus the reference electrode (R) (Freitas, Teixeira, S. Silva, et al., 2016).

2.4.2.5 Literature research

In the next two pages there are presented some research works about silent speech interaction systems developed and it is detailed in Table 2.5 and in Table 2.6 the description of the interaction systems.

The tables are organized by year, title of the research work, the target audience / scenarios, types of operations of the system, the hardware devices used and its features.

Table 2.5: Silent speech interaction systems ordered by year, title of research work, target audience / scenarios, hardware devices and types of operations of the system.

Year	Title	Target audience / scenarios	Hardware devices	Types of operations
2018	Deep Lip Reading: a comparison of models and an online application.	People with voice disabilities.	Camera. Computer.	The user reads a set of texts in front of a camera in a silent speech mode, only with the movement of lips and no acoustic signal.
2018	AlterEgo: A Personalized Wearable Silent Speech Interface.	People with voice disabilities. Scenarios where there is a need of more privacy.	Device of silent speech communication.	The user can ask the system to make arithmetic calculations, turn on/off lights or fan, actual time, calendar, play chess and call and answer phone calls with certain parameters.
2018	Large-Scale Visual Speech Recognition.	People with voice disabilities.	Computer.	The data is collected from "raw videos" of YouTube and other channels of video streams and audio segments.
2014	Multimodal Corpora for Silent Speech Interaction.	People with voice disabilities. Elderly people. Scenarios with noisy environments.	MS Kinect camera. Surface electromyographic sensor. Dedicated circuit with sound meter of Ultrasonic Doppler effect.	The user reads a script in two different ways: one in silent speech mode and another in acoustic speech, with several sensors extracting information data.
2013	A Lip Reading Application on MS Kinect Camera.	Hearing impaired children.	MS Kinect camera.	The user reads a set of 15 Turkish words of color names in front of a MS Kinect camera.

Table 2.6: Silent speech interaction systems ordered by year, title of research work, hardware devices and its features.

Year	Title	Hardware devices	Features
2018	Deep Lip Reading: a comparison of models and an online application.	Camera. Computer.	489.000 samples of unique words from a dictionary of 500 words. Video: 29 frames per second.
2018	AlterEgo: A Personalized Wearable Silent Speech Interface.	Device of silent speech communication.	Signals sampled at 250 Hz. Notch filter at 60 Hz.
2018	Large-Scale Visual Speech Recognition.	Computer.	127,055 word samples. 3886 hours of YouTube videos. Video: 30 frames per second.
2014	Multimodal Corpora for Silent Speech Interaction.	MS Kinect camera. Surface electromyographic sensor. Dedicated circuit with sound meter of Ultrasonic Doppler effect.	Kinect distance: 70 cm. Resolution: 640x480 pixel, 24bit RGB at 30 frames per second. Transducers at 40kHz. Sound meter distance: 40 cm.
2013	A Lip Reading Application on MS Kinect Camera.	MS Kinect camera.	Kinect distance: 40 cm. Video: 12 frames per second. Resolution: 1280x960 pixel. 750 word samples.

2.5 Multimodal Interaction

Jaimes and Sebe (2007) state that "a multimodal HCI system is simply one that responds to inputs in more than one modality or communication channel". Thus, multimodal interfaces are interfaces that process more than one user input mode that can be combined with each other, such as speech, touch, gaze, head and body movements.

The application named "Put-That-There", developed by Bolt et al. (1980), introduces a system composed of two input modalities: gestures and speech. It is one of the first multimodal interfaces developed and it offers a new functionality with the combination of gestures and speech. The processing data is made by semantics via speech and by deictic terms corresponding to the coordinates of pointing gestures with the hands. It can be represented by "[PUT] (24, 57) (86, 28)", where "[PUT]" is the user's intention-to-act, "(24, 57)" is the source coordinate of the user's pointing gesture and "(86, 28)" is the destination coordinate of the user's pointing gesture.

Although multimodal systems are a great way of communication between a human and a computer, recognizers make errors and there are problems with real world scenarios. This happens due to indefinite signals (even humans can not interpret everything correctly) and environmental influences. Some examples of errors that may occur are the human miscommunication, ambient noise, social interchange, multitasking and interruption of tasks.

2.5.1 W3C Multimodal Interaction architecture

In order to develop multimodal applications easily and more efficiently, the World Wide Web Consortium (W3C) has proposed a standard for multimodal applications - the Multimodal Architecture and Interfaces specification, based on the work of the "W3C Multimodal Interaction Working Group".

The goal of this open architecture for multimodal development is to provide a way to coordinate multiple modalities in a standard way, with standards methods of communication (Dahl, 2013). This architecture enables developers to create multimodal applications without necessarily being experts in the technologies they want to consider for interaction in their applications.

One advantage of using the W3C multimodal interaction architecture is that it is flexible, to aggregate easily new modalities and to accept multi-device applications and multi-user applications. Another advantage of the MMI architecture is that it is independent of specific application domains and the overall system is maintainable and extensible. Thus, this independent architecture can lead to the reusing of modalities components and their integration into different applications.

The *Multimodal Interaction* (2012) states that "an implementation is conformant with the MMI architecture if it consists of one or more software constituents that are conformant with the MMI Lifecycle Event specification". In this way, in order to support the lifecycle event interface, a constituent must be able to handle all lifecycle events.

It is presented the MMI Runtime Framework:

Interaction Manager: It coordinates the multiple modalities and it is the "Controller" in the MVC[10] paradigm.

[10]Model–View–Controller is an architectural pattern commonly used for developing user interfaces that divides an application into three interconnected parts. From *Model–view–controller - Wikipedia* (2019).

All lifecycle events that the modality components generate must be delivered to the IM and all lifecycle events that are delivered to the modality components must be sent by the IM.

Normally the markup languages associated with the IM are SMIL, CCXML, SCXML (state machines can be composed) and ECMAScript.

Data Component: Provides the common data model and it is the "Model" in the MVC paradigm.

The data component is responsible for storing application-level data. The interaction manager can access and update it as part of its control flow logic, but the modality components do not have direct access to it (the only way that modality components can share data among themselves is via the interaction manager).

Modality Components: Provide the interaction capabilities for each modality. They are the "Views" in the MVC paradigm.

They are responsible for controlling the input and output modalities and handling all the interaction with the user.

Runtime Framework: Provides the infrastructure and enables the communication between the interaction manager, the data component and the modality components.

The event transport layer is responsible for delivering events among the IM and the modality components and it includes starting the components, handling the communication, logging, etc.

2.5.2 Standard Communication

In multimodal interaction, multiple devices can be used and each device can open multiple communication endpoints at the same time. This standard communication among system components includes lifecycle event messages (*Multimodal Interaction* 2012).

The lifecycle event messages allow the Interaction Manager to invoke Modality Components and receive results from them. The Modality Components communicate with the IM via asynchronous lifecycle events. This communication supports the sending and handling events, generating an asynchronous response to them.

The responses generated by these events are transmitted from the Modality Components to the IM and the majority of the events are defined by request/response pairs. All lifecycle event messages are presented in the "W3C Lifecycle Events" appendix section A.

2.5.3 Generic Modalities

2.5.3.1 Requirements

Non-verbal communication is always present in humans as we are constantly interacting with each other and using it in a way of interaction. Thus, in the context of multimodality there could be great advantages in conveying gestures, gaze and silent speech human-human interactions to human-computer interactions.

The multimodal framework used in each one of the generic modalities and developed in the Department of Electronics, Telecommunications and Informatics in Aveiro, related to the W3C pattern, is capable of generating events and responses from all kinds of modalities assigned

(Almeida, 2017). Thus, defining previously the types of input modalities, the types of actions, scenarios the user may be involved and the feedback responses reproduced from the computer, the framework can couple all features and design use cases scenarios.

In order to facilitate the development of a multimodal application system, the developer does not need to master each one of the technologies. Each modality can be decoupled and be independent from the rest of the framework components. Thus, sending asynchronous lifecycle events from each modality to the Interaction Manager and to the Application, and vice-versa, gives to the developer the support and maintenance of an independent and efficient system.

In addition, the developers are able to reuse each modality for several interaction scenarios, such as for research, educational, clinical, psychological and also for marketing purposes. They also have the possibility of creating new features (multimodal events) or modify and remove the ones that were previously created.

2.5.3.2 Multimodal architecture

Each one of the generic modalities is composed by the same multimodal architecture. Thus, it smooths the way for developers to integrate each modality into application systems. It is also easier for developers to fusion all modalities later into a single application system. Examples of multimodal architectures are the W3C Multimodal Interaction architecture and the "Cue-me Multimodal Platform", which is going to be detailed later in subsection 2.5.4.2.

It is presented in Figure 2.28 a diagram of the runtime architecture of the system overview.

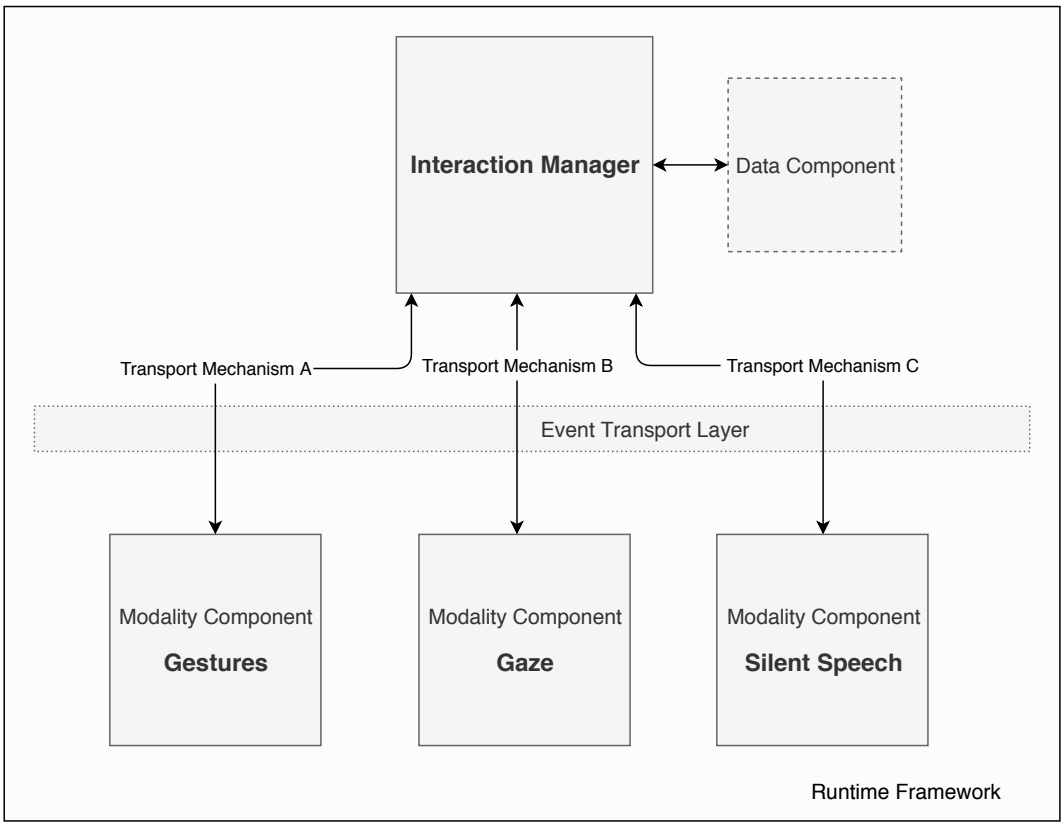


Figure 2.28: The runtime architecture diagram of the system overview, modified from the original in *Multimodal Interaction* (2012).

2.5.4 Examples

2.5.4.1 AALFred: The Personal Assistant of Project PaeLIFE

Teixeira, S. Silva, et al. (2016) developed a Personal Life Assistant of the Project PaeLIFE[11] for AAL (Ambient Assisted Living) scenarios. It was developed for elderly users with some experience in using computers for inside and outside house scenarios.

AALFred offers a variety of services such as messaging services, calendar, social networking, weather information, news, and others. AALFred also supports several languages for interaction like Portuguese, English, French, Polish and Hungarian, for input and output. The ways the user can interact with the system are: a MS Kinect One camera for gestures recognition, a keyboard and a mouse.

It is characterized by a W3C-based architecture for MMI, so several W3C recommendations were used:

- Extensible MultiModal Annotation markup language (EMMA)[12]: represents the data transferred among the entities of the system;
- Lifecycle events: are responsible for the communication between the modalities and the Interaction Manager.
- Modality definition: using external services.

The Personal Life Assistant was developed using a multimodal framework based on an AAL MMI architecture. The goal of this multimodal architecture is to enable a simpler and faster MMI with the decoupling of the modalities, which leads to a simpler development of multimodal applications without the need to master the modality complexity. The complex modalities developed can then be easily integrated in any MMI application.

In Figure 2.29 it is illustrated a diagram of the integration of the multimodal application AALFred with the Multimodal Framework.

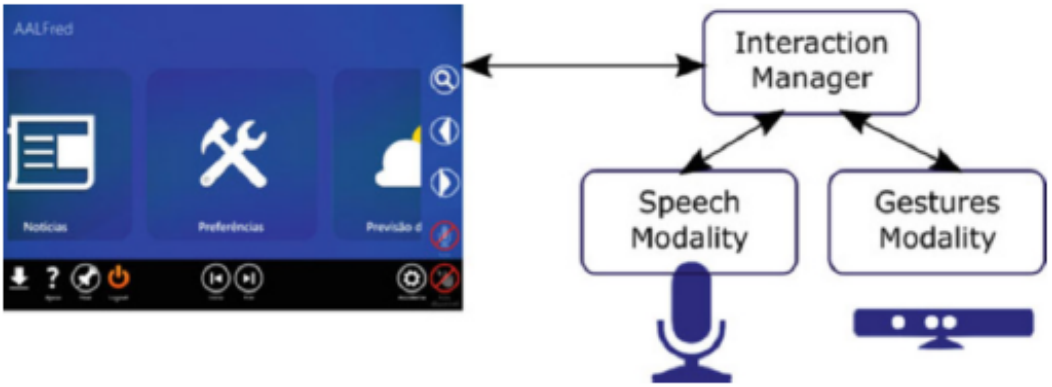


Figure 2.29: Integration of AALFred with the Multimodal Framework. All interaction related events are managed by the Interaction Manager (Teixeira, S. Silva, et al., 2016).

[11] Project PaeLIFE — Personal Assistant to Enhance the Social Life of Seniors. From Teixeira, S. Silva, et al. (2016).

[12] A XML markup language for containing and annotating the interpretation of the user input. From *EMMA: Extensible MultiModal Annotation markup language Version 1.1* (2013).

Figure 2.30 shows how the user can interact with the system in order to create a new appointment in the agenda, by touch or speech input modalities.

Figure 2.30: "Interactions flow to create a new appointment in the agenda, the user starts by choosing the preferred way to interact with the application (touch or speech) to select the day and then select the option to add a new appointment, then fill the subject with the virtual keyboard, and again select to save the appointment by touch or speech" (Teixeira, S. Silva, et al., 2016).

2.5.4.2 Cue-me Multimodal Platform

Tumuluri and Kharidi (2017) developed a multiplatform framework to facilitate the programmer's applications development, named "Cue-me Multimodal Platform". The applications developed using the Cue-me architecture may include multimodal elements such as Gestures, Face Recognition, Speech Recognition (ASR), Synthesis (TTS), Motion Sensing and EEG-headset based interactions and they can run on desktops, laptops, tablets and smartphones. All of the applications developed use the W3C's Multimodal Architecture (*Multimodal Interaction* 2012) and Markup Languages.

The framework is based on the MVC Design Pattern, where the *Data Model* represents the logical structure of the data and the associated integrity constraints, the *Views* are the objects that the user directly interacts with and the *Controller* coordinates the multiple modalities.

The Cue-me runtime framework is composed by the Interaction Manager and the Modality Components. The Core components are HTML-GUI and Voice. Other modalities can be added by the user like the Digital Ink Modality illustrated in Figure 2.31, but the core runtime framework includes:

- GUI Modality (HTML): is the visual application and delivers HTML or XHTML and the images and files supported to the web browser.
- Voice Modality (Speech Recognition and Synthesis): consists of a number of grammar (grammar file formats can be JSGF, ABNF and SRGS) and TTS files (TTS file formats can be raw text or SSML) that are delivered to a speech engine for processing.
- Interaction Manager: integrates the interaction between the GUI Modality and the Voice Modality (or other modalities). It runs a variant of State Chart XML (SCXML).

It is illustrated in Figure 2.31 a diagram of the Cue-me MMI architectural overview.

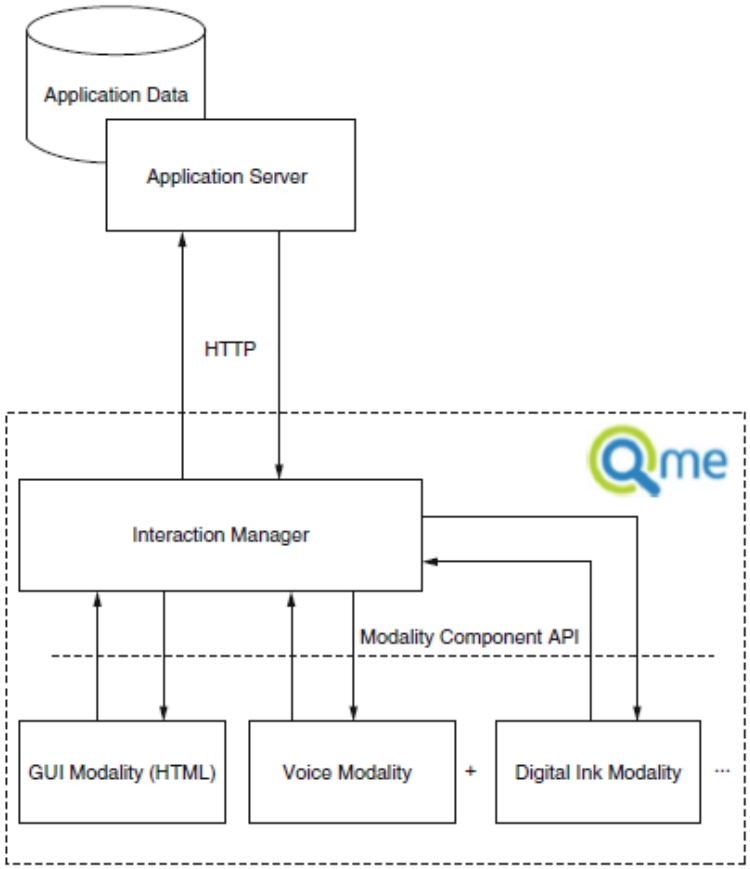


Figure 2.31: Cue-me MMI architectural overview.

The multimodal application is developed according to the norms of the W3C Standards (*Multimodal Interaction* 2012), so it provides Markup Languages, such as XML, VoiceXML, CCXML and State Chart XML (SCXML). From Tumuluri and Kharidi (2017).

Chapter 3

Generic Input Modalities for Silent Interaction

A multimodal interaction enables a more free and natural communication, using automated systems in both input and output. For an intuitive and efficient communication occurs, one must understand the functionalities of each one of the modalities used in the system architecture.

There are some interactive systems where a person can interact with a machine using only gestures as an input modality (operating with a MS Kinect camera, for example), using only gaze (operating with an eye tracker or a web camera, for example) or even using only lip movements as a silent communication (operating with a MS Kinect camera, for example).

However, in order to such systems interpret the user's intentions-to-act more accurately, it is possible to leverage multiple input modalities and assign them to a single system. Also, as the development from the scratch is complex, these modalities should be generic and decoupled from the visual applications, in order to simplify their integration into those applications developed by programmers. Thus, programmers do not need to master the technology used in each modality.

In this research work, three generic modalities are developed: the Generic Gestures Modality, the Generic Gaze Modality and the Generic Silent Speech Modality.

Each generic modality must follow the requirements: (1) the user develops a visual application ("Application"); (2) the user creates and trains gestures, gaze or silent speech commands; (3) a configuration file is created with the multimodal events; (4) the generic modality reads the configuration file and, if a multimodal event is recognized in real time performed by the user, the respective event is sent via Interaction Manager to the "Application" in order to generate visual feedback responses in the visual application; (5) there are already some basic modules, supporting features (multimodal events) so that the user can test the system and (6) the user can create, modify or remove basic features.

In Figure 3.1 it is presented the diagram of the system design overview of the combination of the three modalities developed: gestures, gaze and silent speech.

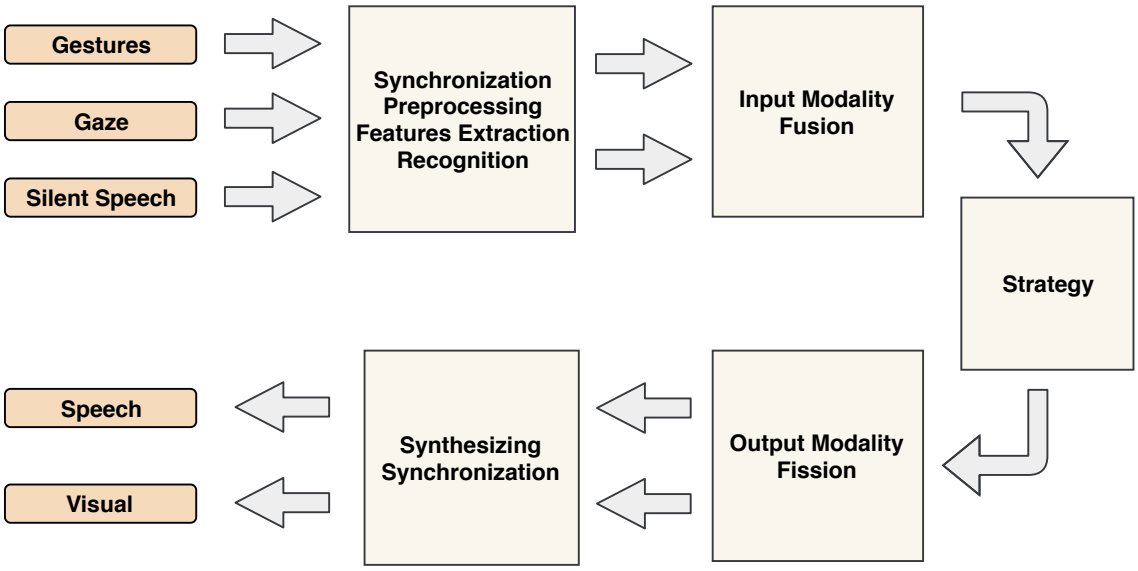


Figure 3.1: The diagram of the system design overview of the combination of the three modalities developed: gestures, gaze and silent speech.

It is illustrated in Figure 3.2 a diagram of the system architecture of the three generic modalities developed in this research work.

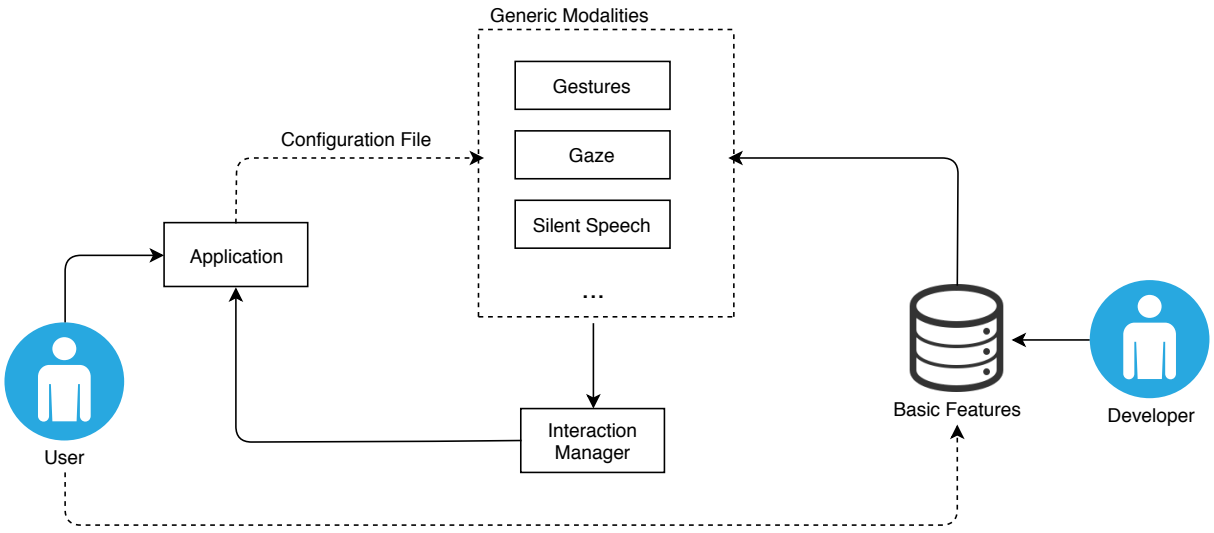


Figure 3.2: The system architecture of the three generic modalities developed.

Each generic modality accepts a configuration file that includes some basic features wherein the user can test the system. The technologies used in each one of the generic modalities are different, so each generic modality must be able to read its respective configuration file. Also, the user can create new features (multimodal events) or even modify and remove the ones that are in the database (configuration file).

In conclusion, each one of the generic modalities follow this approach:

- Configuration file that contains some features (multimodal events);
- Functionalities of the respective generic modality;
- The possibility of using some features (multimodal events) already created in the database;
- The possibility of creating new multimodal events or modifying and removing those that are in the database.

3.1 Generic Gestures Modality

In order to simplify the integration of the Generic Gestures Modality into the visual application, the developer only needs to create and train the gestures in front of a MS Kinect One camera device with the software program *Visual Gesture Builder (VGB)*.

It is illustrated in Figure 3.3 a MS Kinect One camera device.

Figure 3.3: A MS Kinect One camera device. From *Xbox One Standalone Kinect Sensor Available in October* (2014).

The *Visual Gesture Builder* is a tool included with the Kinect for Windows v2 SDK that uses machine-learning algorithms to create gesture databases for Kinect applications. These databases allows the detection of the gestures at runtime using the VGB.

The algorithms used by the VGB software program are the *AdaBoost* (*Adaptive Boosting*) algorithm that determines when a user performs a certain gesture and the *RFRProgress* (*Random Forest Regression Progress*) algorithm that determines the progress of a gesture. These gestures are recorded with the *Kinect Studio* and fed into the VGB.

The user can create two types of gestures: (1) discrete gestures and (2) continuous gestures. They will be detailed in the subsections 3.1.2 and 3.1.3.

3.1.1 System architecture

The system architecture of the "Gestures Modality" is composed by the "Gestures Modality" Component, the Interaction Manager and the "Application" Modality Component.

The "Gestures Modality" is a generic modality that is responsible for the runtime detection and recognition of the gestures that the user is performing at that moment. The user performs a gesture from all the available gestures in the database ("Gestures.gbd" configuration file) in front of a MS Kinect One camera. If the modality recognizes a specific gesture, it is possible to know the name of that gesture and its confidence level.

If the confidence level of the recognized gesture is above 60%, the "Gestures Modality" sends to the Interaction Manager the "ID" of the gesture, its name and its confidence level value. The IM resends the same lifecycle event message to the "Application".

The "Application" is a visual interface application developed by the user. Additionally, it can also be a website using its API[1]. Accordingly to the gesture detected, the "Application" performs a response action in its interface.

It is presented in Figure 3.4 a diagram of the system architecture of the Generic Gestures Modality.

[1]An Application Programming Interface is a set of subroutine definitions, communication protocols and tools for building software. From *Application programming interface - Wikipedia* (2019).

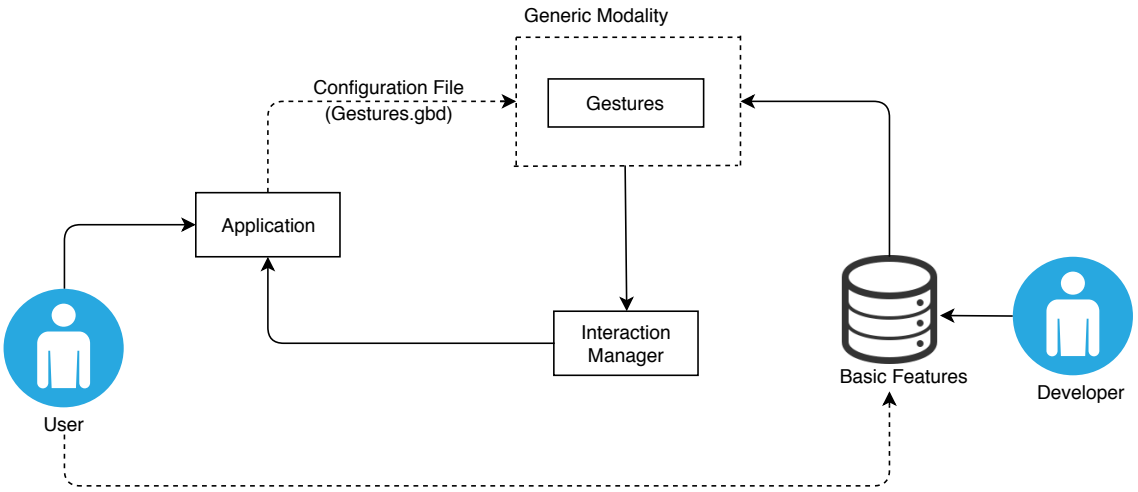


Figure 3.4: The system architecture of the Generic Gestures Modality.

It is also presented in Figure 3.5 a diagram of the gestures interaction between the user and the computer using the multimodal framework Interaction Manager based on the W3C Multimodal Architecture (*Multimodal Interaction* 2012).

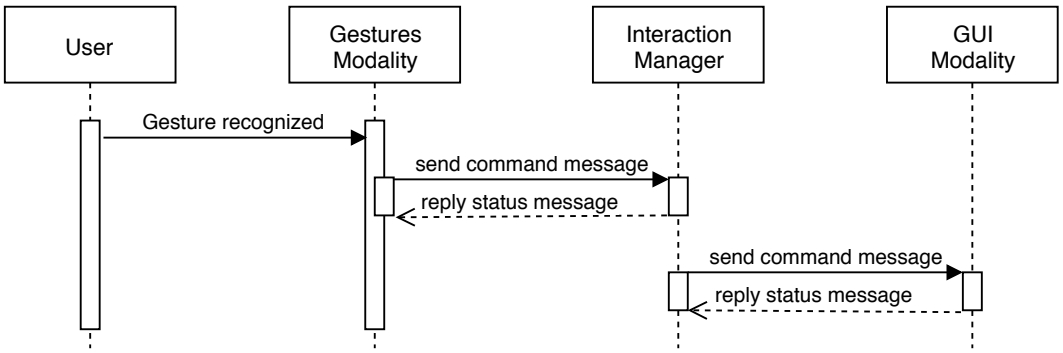


Figure 3.5: Gestures interaction between the user and the computer using the multimodal framework Interaction Manager based on the W3C Multimodal Architecture (*Multimodal Interaction* 2012).

3.1.2 Discrete gestures

Discrete gestures are, typically, single-pose gestures. These type of gestures are detected by the *AdaBoost* (*Adaptive Boosting*) algorithm, which is a binary classifier that returns "true" if a gesture in the database is detected, or "false" if it is not detected.

This type of gestures involves singular (non-repetitive) and large movements, such as sitting down or moving a hand to the left or to the right. Gestures that involve repetitive motions are more difficult for VGB to learn.

It is illustrated in Figure 3.6 the tagging process of a "Swipe Right" discrete gesture using the Visual Gesture Builder software program.

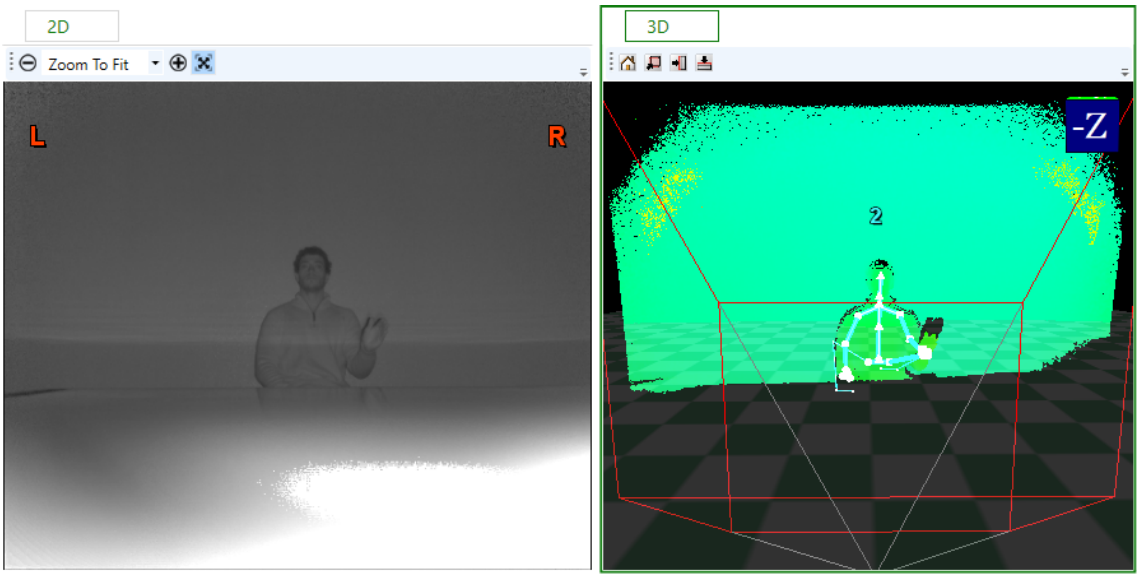


Figure 3.6: The tagging process of a "Swipe Right" discrete gesture using the VGB software program, in 2D on the left image and in 3D on the right image.

3.1.3 Continuous gestures

Continuous gestures are, typically, moving gestures. Differently from the discrete gestures, they are detected by the *RFRProgress* (*Random Forest Regression Progress*) algorithm which takes advantage of regression analysis and detects the moving gesture as it is being performed. During the execution of the gesture, a progression value of 0 means that the gesture did not started yet and a progression value of 1 means that the gesture is at the end stage.

It is illustrated in Figure 3.7 the tagging process of a "Steering Right" continuous gesture using the VGB software program, imitating the changing of the car direction to the right with a steering wheel.

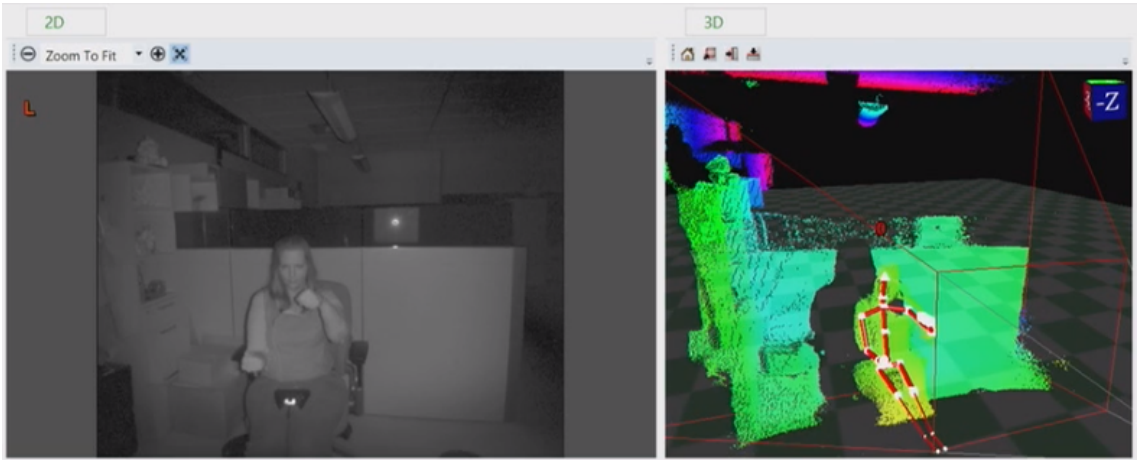


Figure 3.7: The tagging process of a "Steering Right" continuous gesture using the Visual Gesture Builder software program, in 2D on the left image and in 3D on the right image. From *Custom Gestures End to End with Kinect and Visual Gesture Builder (part 2)* (2014).

3.1.4 Gestures creation and training

The gestures detection and recognition is made by machine learning algorithms that recognize complex patterns in data automatically. The *Visual Gesture Builder* software program learns from empirical data examples performed by the user in front a MS Kinect One camera. A classification process is followed to classify data which has not yet been observed.

Firstly, in order to create and train the gestures, the user has to record himself or herself performing the specific gestures in front of a MS Kinect One camera with the *Kinect Studio* software program. The raw recordings are then converted to processed clips using *KSConvert* and saved into a user's directory.

It is illustrated in Figure 3.8 the creation of a "Push" gesture using the *Kinect Studio* software program.

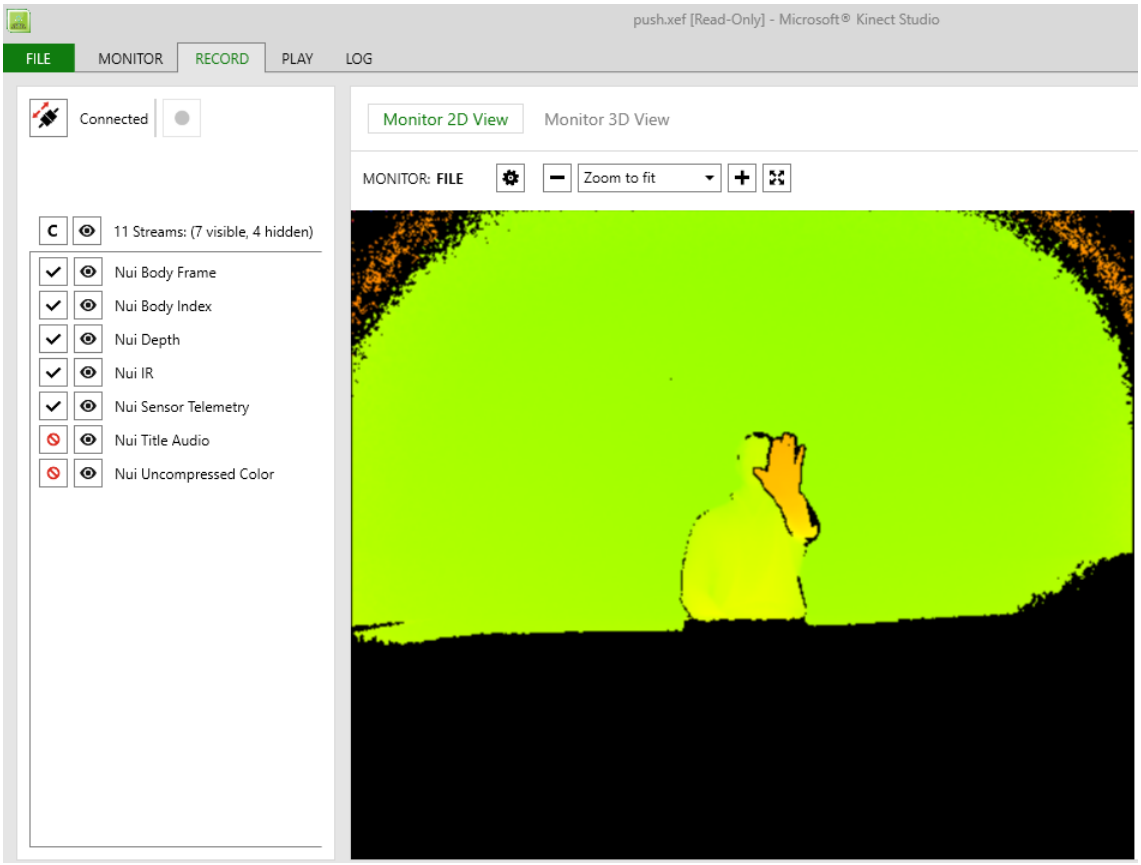


Figure 3.8: Creation of a "Push" gesture using the *Kinect Studio* software program.

The next step is to label the recording frames for each gesture, using the VGB software program. The user can create a "New Solution" and then, for each gesture created and trained, create a "New Project" with the gesture name, body side, gesture type and some training settings. Afterwards, the user can right click on each project and add the recorded clip of the respective gesture by clicking on "Add clip". The recorded clips are generated by *Kinect Studio* and they have ".xef" extension.

To select the frames on each recorded clip, the user can press the *shift* key and then, the *left* and *right* keys, accordingly to its time sequence. To save each frame, the user only needs

to press the *enter* key, after each frame selection. Finally, after all sequences of frames are labeled, the user can save the project.

It is illustrated in Figure 3.9 the tagging process of a "Push" gesture in the VGB software program.

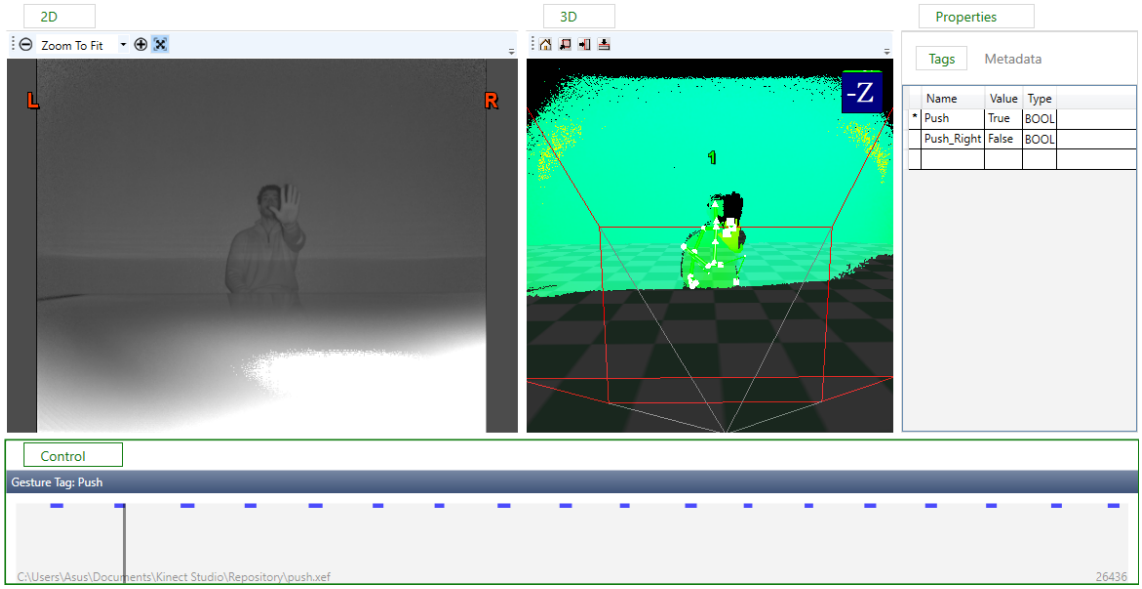


Figure 3.9: The labelling process of a "Push" gesture in the VGB software program, in 2D on the left image and in 3D on the right image.

It is possible to see in Figure 3.9 that were labelled 18 clips from the "Push" gesture and they are represented by the time sequences in blue color. The beginning of these time sequences defines the beginning of the gesture execution and the end of the time sequences defines the conclusion of the gesture execution. These sequences are the part of the whole recording that allow the detection and recognition of the gesture execution.

In the graph 3.10 it is possible to see that the error rate decreases, for false positives, with the higher number of training examples (in frames, not gestures).

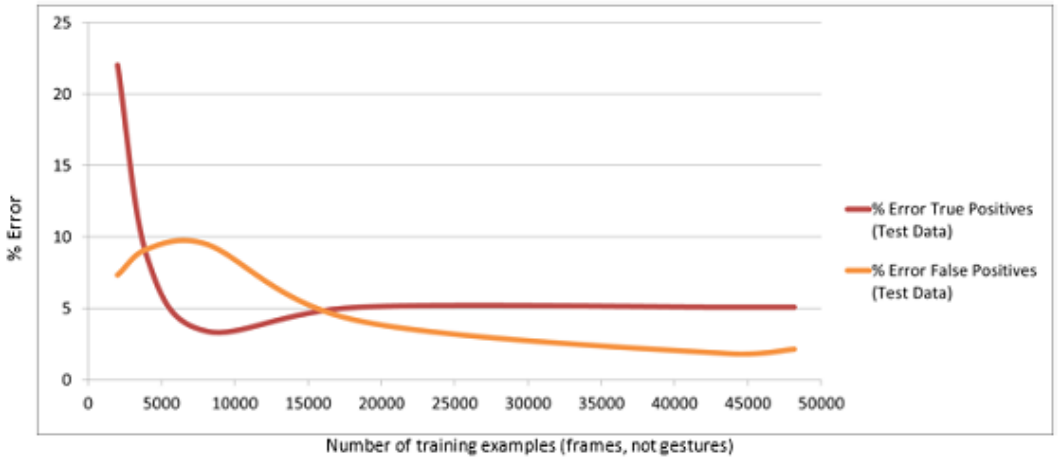


Figure 3.10: Variations of the error rate with the number of training examples. From Ballester Ripoll (2017).

The user can now right right on the solution and then, click on "Build" to generate the database file. All gestures trained with VGB are built into a single configuration file "Gestures.gbd". This file will be read by the "Gestures" Modality Component.

It is illustrated in Figure 3.11 a representation of the gestures training process.

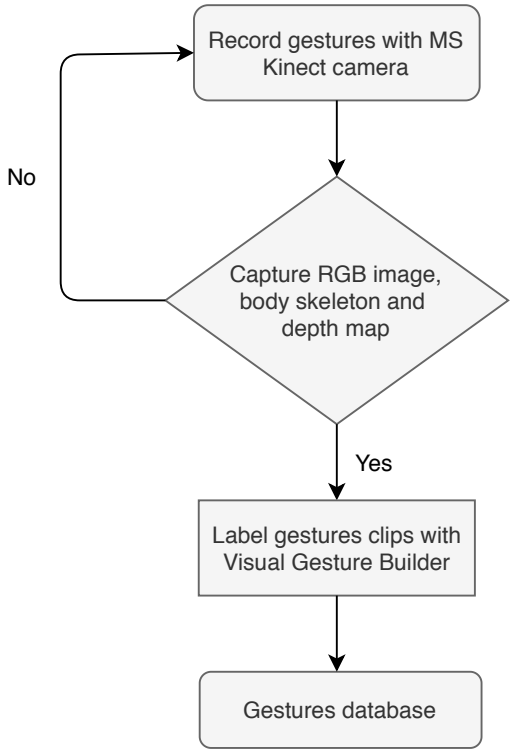


Figure 3.11: A representation of the gestures training process.

3.1.5 Implementation

The configuration file generated by the *Visual Gesture Builder* software program is loaded to the Visual Studio project:

```
private readonly string gestureDatabase = "Gestures.gbd";
```

The user only needs to save the gestures files generated by *Kinect Studio* in a folder called "Repository" within the "GenericGesturesModality" folder. It is now possible to compile and execute the project and the "Gestures Modality" is able to detect and recognize in runtime each gesture created and trained by the user.

If a gesture is detected, the "Gestures Modality" sends to the Interaction Manager a lifecycle event message with the gesture ID, the gesture name and its confidence level or progress (if it is a discrete or a continuous gesture):

```
if (sendGestureName != null)
{
    if (flag)
    {
        int id;
        id = SendGestureID(sendGestureName);

        SendMessage(id, sendGestureName, sendConfidence);
        flag = false;
        anyGestureDetected = false;
    }
}

if (gesture.Name.Equals(gestName[i]))
{
    int id;
    id = SendGestureID(gestName[i]);

    SendMessage(id, gestName[i], progress);
    anyGestureDetected = true;
}
```

The "Generic Gestures Modality" also provides the visualization of the user's body skeleton in real time. It is illustrated in Figure 3.12 the execution of a "Push" gesture in front of a MS Kinect One camera and the skeleton representation of the user's body movements.

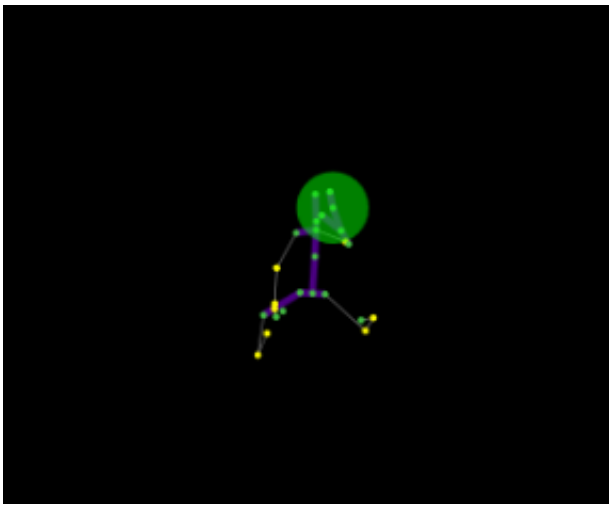


Figure 3.12: The execution of a “Push” gesture in front of a MS Kinect One camera and the skeleton representation of the user’s body movements.

3.2 Generic Gaze Modality

Nowadays, the number of eye trackers' users are increasing day after day. The eye tracker that is globally the most used and the one that is used in this research work is the Tobii Eye Tracker 4C. The eye trackers developed by the *Tobii Technology* company work for any kind of user in any kind of environment. They can be used for research, diagnostics, process control, somnolence detection, clinic facilities and disabilities purposes (Vigo, 2013).

The Tobii Pro SDK offers multi-platform support (Windows, Linux and Mac), API bindings for several programming languages (.NET, Python, Matlab/Octave and C) and prefabs for 3D engines. In this research work, it is used the C# programming language.

These eye trackers are based on the technique of the corneal reflection, which uses multiple infrared diodes to create a reflection on the cornea[2] of the eyes. The image sensor gets the reflections and other visual data information about the user and processes the images. The eyes are detected at first, then the exact position of the pupil and finally the right reflections from the diodes and their exact positions are found. The point of the user's gaze is calculated at the end by a complex mathematical model used to represent the eyes in 3-dimensions.

In Figure 3.13 it is presented the Tobii Eye Tracker 4C device.

Figure 3.13: Tobii Eye Tracker 4C device. From *Tobii Focuses On Foveated Rendering, More Games Get Eye-Tracking Support* (2017).

The operating distance of the Tobii Eye Tracker 4C device is from 50 to 95 cm, the illuminators are Near Infrared (NIR 850nm) Only and the data rate is 90 *Hz*. The tracking distance represents the distance from the user to the sensor where the eyes can be detected and tracked. The data rate is the number of gaze data points recorded per second. This means that this device collects 90 gaze data points per second for each one of the eyes.

The Tobii Core SDK provides streams and functions that enable the user to use inputs in their gaze interaction applications, such as the gaze position, eyes position, head position, calibration, settings and the user profiles, for example.

The "Gaze Modality" is based on the creation of figures and tagging a gaze event to each one of the figures created. The eye movements that are possible to perform are gaze fixations and gaze saccades, which the last one can be divided into "swipe" gaze gestures and "sequence" gaze gestures. These eye movements will be described in the subsections 3.2.2 and 3.2.3, respectively.

[2]It is the transparent layer forming the front of the eye. From *Cornea - Wikipedia* (2019).

3.2.1 System architecture

The system architecture of the "Gaze Modality" is composed by the "GazeML" Modality Component, the Interaction Manager explained in the beginning of this chapter 3, the "State Machine" Modality Component and the "Application" Modality Component.

The "Application" is an interface application where the user can create figures (rectangles or ellipses), have the possibility to associate an image to each figure and define gaze gestures to each one of them. In the "Application" Modality Component, the figures created are written to a "Figures.xml" configuration file and the gaze gestures created are written to a "GazeEvents.xml" configuration file.

The "GazeML" Modality Component is responsible of creating the same display as in the "Application" Modality Component, such as the figures and the associated images. This is made by reading the "Figures.xml" configuration file with the figures' features created previously by the "Application" Modality Component. It is also responsible of creating the panels for each figure that assign an interaction gaze behavior of the gaze gestures added by the user in the "Application" Modality Component.

The "State Machine" Modality Component receives the POST requests from a HTTP Client from the "GazeML" when the user directs his or her gaze to a figure or a sequence of two (swipe gaze gesture) or more figures (sequence gaze gesture). If it recognizes the gaze gesture, it sends a lifecycle event message to the Interaction Manager and then to the "Application" the corresponding gaze gesture event and region(s) associated for proper visual feedback responses.

It is presented in Figure 3.14 a diagram of the system architecture of the Generic Gaze Modality.

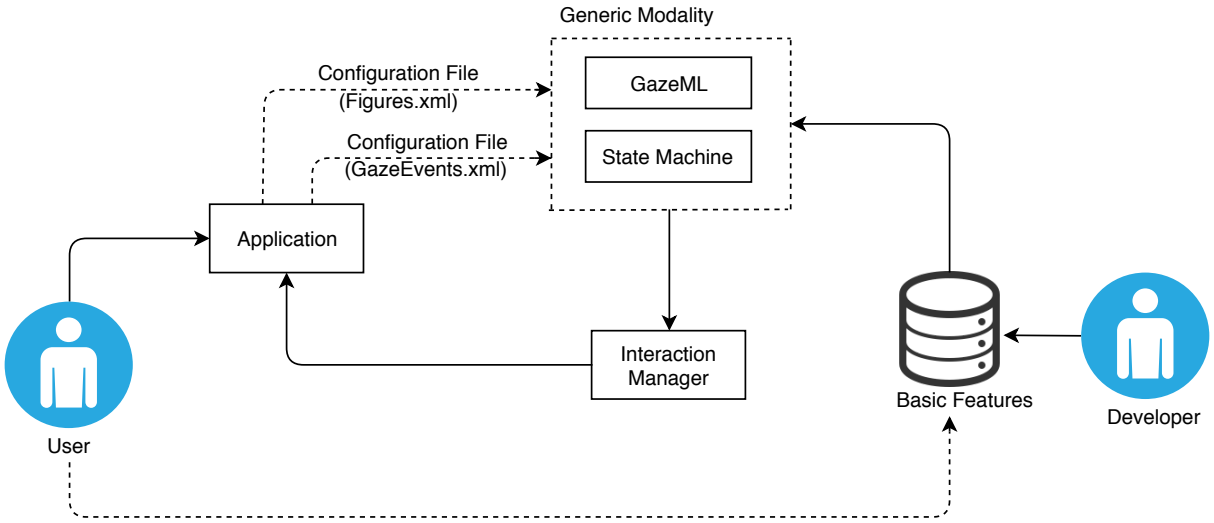


Figure 3.14: The system architecture of the Generic Gaze Modality.

It is presented in Figure 3.15 a diagram of the gaze interaction between the user and the computer using the multimodal framework Interaction Manager based on the W3C Multimodal Architecture (*Multimodal Interaction* 2012).

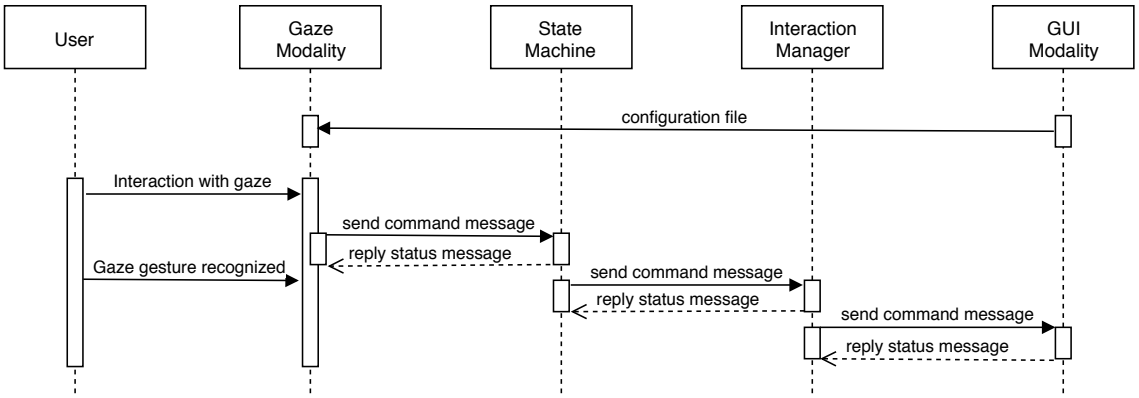


Figure 3.15: Gaze interaction between the user and the computer using the multimodal framework Interaction Manager based on the W3C Multimodal Architecture (*Multimodal Interaction* 2012).

3.2.2 Gaze fixations

Toh, Rossell, and Castle (2011) defined gaze fixations as the representation of points of attention, where the gaze is held for a duration of at least 100-300 ms.

In this modality, the user can perform a gaze fixation by holding his or her gaze to a fixation gaze region of the computer display. This gaze fixation must be held for a duration of 500 ms (0,5s):

```
behaviorMap1.Add(panel1, new EyeXFramework.GazeAwareBehavior(OnGaze1)
{ DelayMilliseconds = 500 });
```

This time duration enables the user to interact with the system with a natural intention-to-act and also avoiding the Midas Touch Problem, where the user generate unconsciously gaze events just by looking to other regions of the display.

3.2.3 Gaze saccades

As it was already described in the subsection 2.3.1.2, gaze saccades are rapid eye movements between fixations, shifting the focus from one point to another.

In this modality, the user can perform a gaze saccade event by gazing to a specific gaze region to a different gaze region ("swipe" gaze gesture) or to different consecutive gaze regions ("sequence" gaze gesture) of the computer display.

For example, a "swipe right" gaze gesture event occurs when the user gazes to a figure located in the left area of the computer display and then gazes to a figure located in the right area, both labelled with the same "swipe" gaze gesture.

Another example is when the user wants to perform a "sequence" gaze gesture with 4 gaze points of interest. This gaze gesture event occurs when the user gazes, consecutively, to the first gaze point, followed the second, the third and ending with the fourth. All of these gaze points are labelled with the same "sequence" gaze gesture.

This type of eyes movements enables the user to perform several gaze gestures, such as the "swipe" gaze gestures and the "sequence" gaze gestures, with infinite consecutive gaze points, depending on what the user pretends to do.

3.2.4 Implementation

3.2.4.1 Application interface

In the "Application" Modality Component, the figures created are written to a "Figures.xml" configuration file and the gaze gestures created are written to a "GazeEvents.xml" configuration file.

The only two requirements for the creation of the figures are: (1) if there is an image associated with a figure, the image name must end with "_pic" and the file extension must be ".png" and (2) if there is an image associated with a figure, the figure name must be the same as the image. For example, if an image has the name "preto_pic", the figure name associated to this image must also be "preto_pic".

The only requirement for the creation of the gaze gestures' events is: (1) add the gaze gesture event to the "List<string> gazeEvents" in the method "GazeEvents()" in the file "MainWindow.xaml.cs" by typing the semantic of the gaze gesture (fixation, swipe or sequence) followed by a "_" and ending with the figure(s) name(s) associated to the respective gaze gesture event. For example, if the user wants to add a fixation gaze gesture to the "azul" image, a swipe gaze gesture from the "preto" to the "vermelho" images and a sequence gaze gesture with the "ponto1", "ponto2" and "ponto3" images with consecutive gaze points, he or she can create the gaze gestures' events by:

```
gazeEvents.Add("fixation_azul");
gazeEvents.Add("swipe_preto_vermelho");
gazeEvents.Add("sequence_ponto1_ponto2_ponto3");
```

The features of the figures created are then written to a XML file which it is going to be read later by the "GazeML" Modality Component. This configuration file has the purpose of passing the configuration features of the figures to "GazeML" and create the same figures with the same images (if they have association with them) and also the same positions from each figure initially defined by the user.

This initial configuration file contains the following attributes of the figures: ID, type (rectangle or ellipse), name, text, the "X" and "Y" positions on the canvas, the width and the height values.

It is presented an example of a created figure by the user written to the "Figures.xml" configuration file:

```
<Figures>
  <Figure>
    <ID>0</ID>
    <TYPE>Rectangle</TYPE>
    <NAME>amarelo_pic</NAME>
    <TEXT>amarelo</TEXT>
    <X>150</X>
    <Y>265</Y>
    <WIDTH>170</WIDTH>
    <HEIGHT>160</HEIGHT>
  </Figure>
<Figures>
```

3.2.4.2 GazeML

The "GazeML" Modality Component receives the features of the figures created by the user from the "Figures.xml" configuration file and creates them in its "Form1.cs" file. For each figure created in "Application", it is created a panel that is going to be associated later a gaze event. If the user directs his or her gaze to a figure in the "GazeML" display, it will generate the gaze event defined by the user to that figure.

In order to get a precise lightly filtered gaze data points from the Tobii Eye Tracker 4C device, it was created a "GazeDataPoints.cs" class.

Firstly, it is needed to instantiate the Host which serves as an entry point into the Tobii Core SDK:

```
private readonly Host _host = new Host();
```

Next, the host is used to instantiate the "GazePointDataStream", as "_lightlyFilteredGaze-PointDataStream". It contains the "X" and "Y" coordinates of the user's gaze location and the corresponding "Timestamps":

```
private readonly GazePointDataStream _lightlyFilteredGazePointDataStream;

_lightlyFilteredGazePointDataStream = _host.Streams.CreateGazePointData
Stream();
```

Finally, when data is pushed to the stream in the "sendGazePoints()" method, it gets the associated coordinates and timestamps of the user's gaze:

```
_lightlyFilteredGazePointDataStream.GazePoint((x, y, timestamp)
=> {
    X = x;
    Y = y;
    Timestamp = timestamp;
});
```

The figures are now passed to "GazeML" and the same display representation is created in the "GazeML" Modality Component.

If the user directs his or her gaze to a figure, a HTTP Post request message from a HTTP Client is sent to the "State Machine" with the "text" property of that figure. For example, if the user interacts with his or her gaze to the "azul_pic" figure, it will be sent a HTTP Post request event message with "azul". Then, the "State Machine", which will be explained later in the subsection 3.2.4.3, processes the HTTP Post request message received and checks what is the corresponding gaze event associated to that figure. For example, if it is a fixation gaze gesture, the "State Machine" recognizes it and sends a new lifecycle event message to the Interaction Manager with the name of the gaze gesture ("fixation") and the region associated ("azul"). The Interaction Manager resends the message to the "Application", which takes a visual action in response to the gaze gesture event.

It is illustrated in Figure 3.16 a sequential representation of the Generic Gaze Modality.

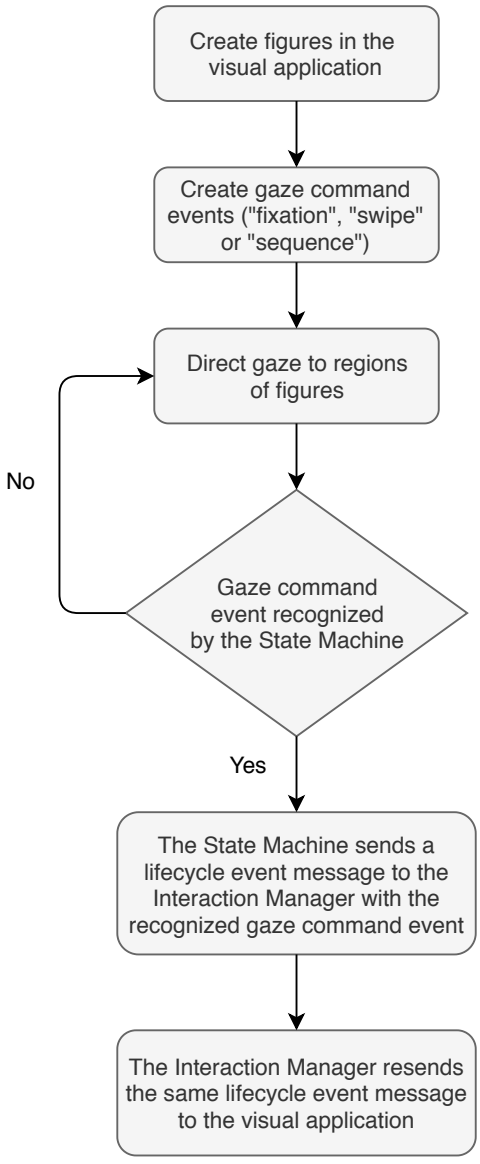


Figure 3.16: A sequential representation of the Generic Gaze Modality.

- Fixation gaze gesture:

  It is shown in Figure 3.17 that when the user fixates his or her gaze during 500 ms to the middle rectangle (figure "azul"), it is performed a fixation gaze gesture. This event is known by the transmission of a HTTP Post from the "GazeML" to the "State Machine", then the "State Machine" sends a lifecycle event message to the Interaction Manager which is resent to the "Application".

Figure 3.17: The IM sends the lifecycle message to the "Application": `['COMMAND', 'Fixation', 'ORIGIN', 'azul']`.

- Swipe gaze gesture:

  It is shown in Figure 3.18 that when the user gazes from the right rectangle to the left one, it is performed a "Swipe Left" gaze gesture. This event is known by the transmission of a HTTP Post from the "GazeML" of the two figures to the "State Machine", then the "State Machine" sends a lifecycle event message to the Interaction Manager which is resent to the "Application".

Figure 3.18: The IM sends the lifecycle message to the "Application": `['COMMAND', 'Swipe', 'DIRECTION', 'left', 'ORIGIN', 'verde', 'END', 'amarelo']`.

- Sequence gaze gesture:

  It is shown in Figure 3.19 that when the user gazes to the first ellipse point, then to the third ellipse point and finally to the fourth ellipse point consecutively, it is performed a "Sequence" gaze gesture. This event is known by the transmission of a HTTP Post from the "GazeML" of the three ellipse figures to the "State Machine", then the "State Machine" sends a lifecycle event message to the Interaction Manager which is resent to the "Application".

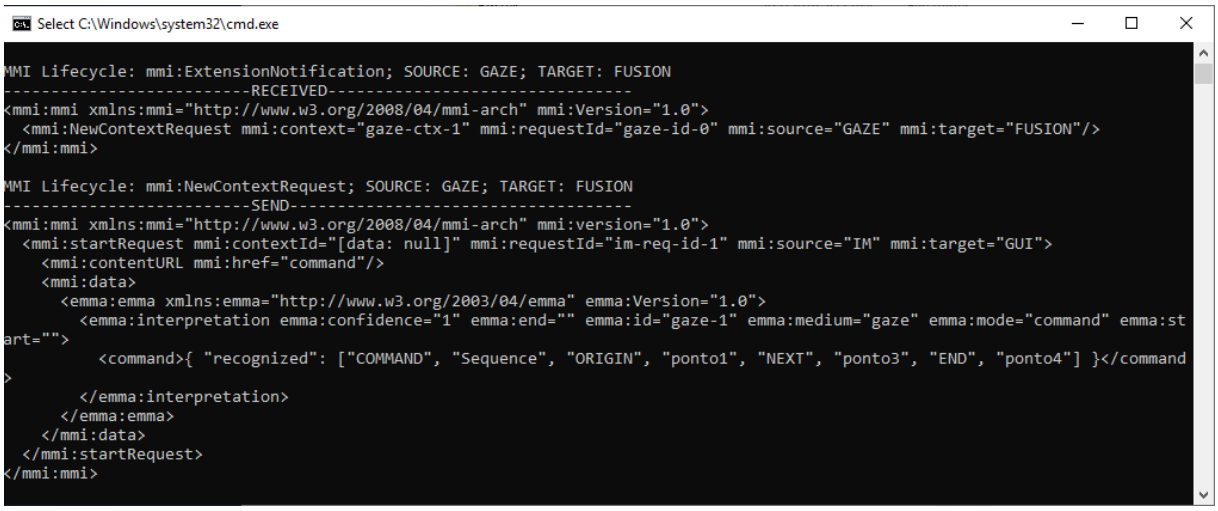


Figure 3.19: The IM sends the lifecycle message to the "Application": ["COMMAND", "Sequence", "ORIGIN", "ponto1", "NEXT", "ponto3", "END", "ponto4"].

3.2.4.3 State machine

The gaze gestures' events created in the "Application" Modality Component are written to the "GazeEvents.xml" configuration file which it is going to be read later by the "State Machine" Modality Component. This configuration file has the purpose of passing the gaze gestures to the "State Machine" in order to recognize the events for each gaze gesture added by the user.

This configuration file contains the following attributes of the gaze gestures' events: ID, semantic and region(s) associated to each of the gaze gesture. It is presented an example of two created gaze gestures by the user written to the "GazeEvents.xml" configuration file:

```
<Gaze>
    <Event>
    <ID>0</ID>
    <SEMANTIC>swipe</SEMANTIC>
    <REGION1>preto</REGION1>
    <REGION2>vermelho</REGION2>
  </Event>
  <Event>
    <ID>1</ID>
    <SEMANTIC>sequence</SEMANTIC>
```

```
    <REGION1>ponto1</REGION1>
    <REGION2>ponto3</REGION2>
    <REGION3>ponto4</REGION3>
  </Event>
<Gaze>
```

In order to receive the figures that the user is directing his or her gaze to, it is created a HTTP Listener that "listens" the messages received by the "GazeML" (the messages containing the text property of the figures where the user directs his or her gaze to).

It is necessary to load the "State Machine" to have a proper communication between the HTTP Client (from "GazeML") and HTTP Listener (from "State Machine"). To make this initial action, the user must open the Postman software program[3] and send a new post to the url "http://localhost:9876/" with the "Body" value ""load":"","sessionToken":"0"". The "KEY" of the "Headers" section is "Content-Type" and the "VALUE" is "application/json".

It is presented in Figure 3.20 the representation of the parameters values in the Postman software program for the loading of the "State Machine".

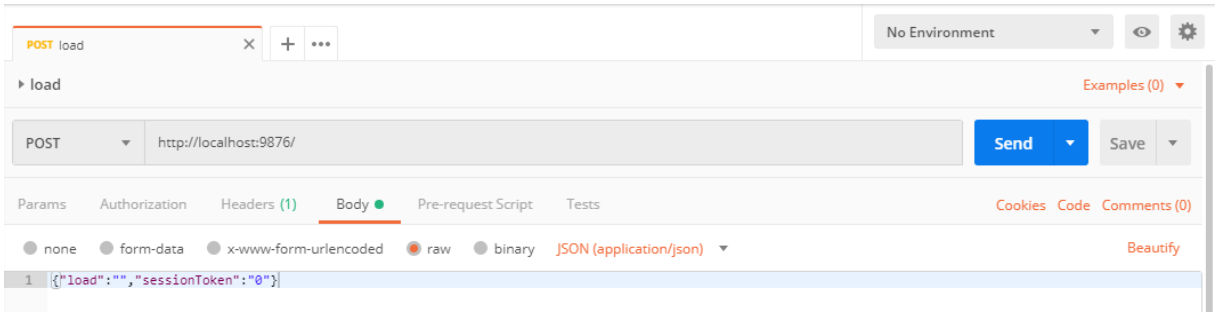


Figure 3.20: The representation of the parameters values in the Postman software program for the loading of the "State Machine".

The "State Machine" Modality Component reads all of the gaze gestures' events from the "GazeEvents.xml" configuration file and creates a dynamic state machine file ("stateMachine.scxml") in SCXML language based on the gaze gestures' events.

In the "stateMachine.scxml" it is created an initial state with all the first text properties of the figures (the first regions of each one of the gaze gestures' events). It is presented an example of an initial state of a state machine created dynamically based on the gaze gestures added by the user in the "Application":

```
<state id="initial">
  <transition target="azul" event="azul"/>
  <transition target="preto" event="preto"/>
  <transition target="vermelho" event="vermelho"/>
  <transition target="amarelo" event="amarelo"/>
  <transition target="verde" event="verde"/>
  <transition target="ponto1" event="ponto1"/>
</state>
```

[3] https://www.getpostman.com/

In this example, it was created the gaze gesture "fixation_azul", so it must generate a fixation gaze gesture event in the figure called "azul_pic". In the "stateMachine.scxml" file of this example, it is created a new state called "azul" and if the user directs his or her gaze to that figure during 2 seconds, a new state called "fus_azul" sends a lifecycle event message to the Interaction Manager with the corresponding gaze gesture event:

```
<state id="azul">
  <onentry>
    <send sendid="state0-timer" event="timeout" delay="2s"/>
  </onentry>
  <transition target="fus_azul" event="timeout"/>
  <onexit>
    <cancel sendid="state0-timer"/>
  </onexit>
</state>

<state id="fus_azul">
  <onentry>
    <script>
      sendMessage("Fixation_azul");
    </script>
  </onentry>
  <transition target="initial"/>
</state>
```

The same method occurs for the swipe and sequence gaze gestures. If the "State Machine" receives the first region of a swipe gaze gesture and then the second region, it is sent the respective swipe gaze gesture to the Interaction Manager. However, if it receives the first region of a swipe gaze gesture and it does not receive the second region within 2 seconds, the state machine goes to the initial state. The same happens for sequence gaze gestures but the time delay is now 5 seconds.

The lifecycle event messages are sent to the Interaction Manager based on the fixation, swipe and sequence gaze gestures added by the user. These messages are composed by the semantic of the gaze gesture (fixation, swipe or sequence), if it is a swipe gaze gesture, it sends its direction (left, right, up or down) and the region(s) associated to that gaze gesture.

3.3 Generic Silent Speech Modality

In order to simplify the integration of the "Silent Speech Modality" into the application, the user only needs to train the words that he or she wants to use in the application system in front of a MS Kinect One camera device and creating them in the "Application" Modality Component. This process is performed to firstly create a database of the words' features extracted from the MS Kinect One camera.

Then, after the words training, a process of classification is made in order to create a model for the database of the words. The classification process consists of reading the database of the words trained and a machine learning classifier trains and builds a model.

Finally, the user utters the word in front of a MS Kinect One camera and the features extracted are classified in real time and compared to the model created previously. This gives to the user a prediction of the word that he or she uttered and the "Silent Speech Modality" sends the predicted word to the Interaction Manager, which is resent to the "Application" or to the application software that the user wants to use.

3.3.1 System architecture

The system architecture of the "Silent Speech Modality" is composed by the "Application" Modality Component, the "Features Extraction" Modality Component, the "Classification Models" Modality Component, the "Silent Speech" Modality Component and the Interaction Manager explained in the beginning of this chapter 4.

The "Application" Modality Component is an interface application where the user can create the words to train, writing them to a "Words.xml" configuration file, and use them later for a real time recognition.

The "Features Extraction" Modality Component is an application where the user trains the words from the configuration file. The features from the lips and chin are extracted, normalized and fixed size during the words recordings in front of a MS Kinect One camera.

The "Classification Models" Modality Component reads the training and test files that contain the lips and chin features. Some classifiers are evaluated and the one that gets the highest number of correctly instances predicted is the one that trains and builds a model.

The "Silent Speech" Modality Component extracts the same features of the lips and chin in order to make a real time comparison with the model trained and built by the "Classification Models". At the same time, it is performed the process of the word prediction and the word that gets the highest predicted percentage is the one that is sent to the Interaction Manager. The Interaction Manager resends this lifecycle event message to the "Application".

It is presented in Figure 3.21 a diagram of the system architecture of the Generic Silent Speech modality.

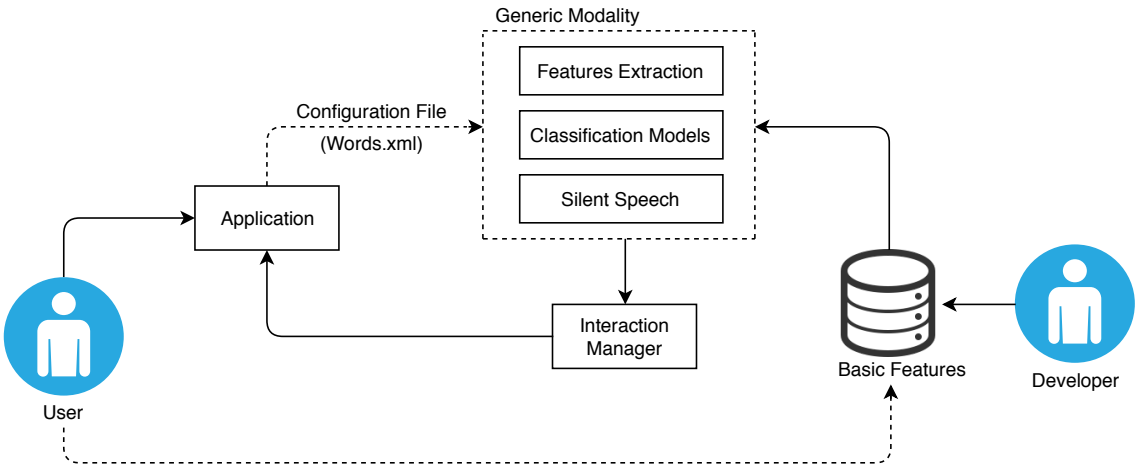


Figure 3.21: The system architecture of the Generic Silent Speech modality.

It is presented in Figure 3.22 a diagram of the silent speech interaction between the user and the computer using the multimodal framework Interaction Manager based on the W3C Multimodal Architecture (*Multimodal Interaction* 2012).

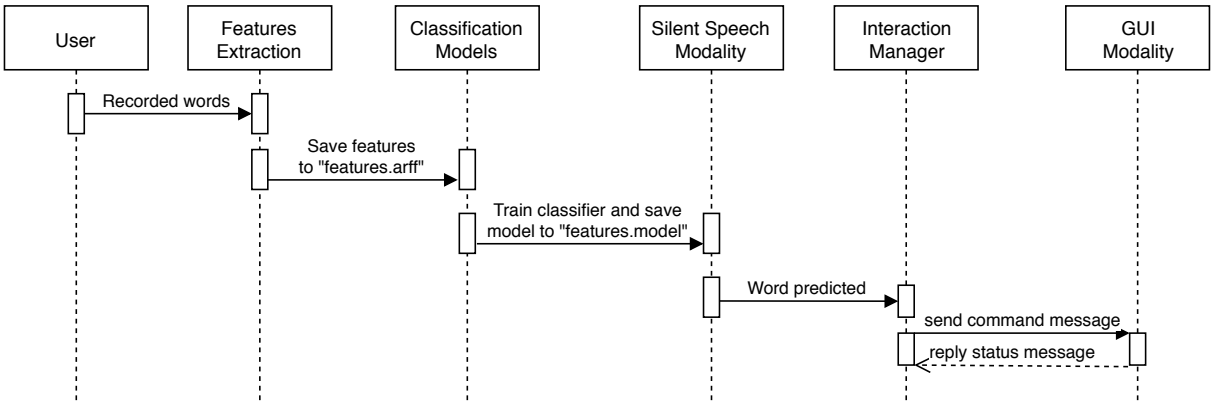


Figure 3.22: Silent speech interaction between the user and the computer using the multimodal framework Interaction Manager based on the W3C Multimodal Architecture (*Multimodal Interaction* 2012).

3.3.2 Implementation

3.3.2.1 Application interface

The "Application" is a visual application interface where that are generated proper feedback responses from the words predicted performed in real time by the user in front of a MS Kinect One camera.

As a generic modality, the user can create the words that he or she wants to use in the system by adding them to a List<string> in the "Words()" method in the "MainWindow.xaml.cs" file:

```
words.Add("play");
```

```
words.Add("pause");
words.Add("stop");
words.Add("more_volume");
words.Add("less_volume");
words.Add("forward");
words.Add("backward");
```

The words created in the "Application" are then written to a "Words.xml" configuration file that is going to be read later by the "Features Extraction" Modality Component to train each one of those words.

The words in the "Words.xml" configuration file can also be modified, removed and created new ones by the user at any time.

3.3.2.2 Features extraction

The first step of the "Generic Silent Speech Modality" is to extract the lips and chin features of the face. In total, there are extracted 9 features from a person's face: the lip left corner (x coordinate), the lip right corner (x coordinate), the lip width, the lip top corner (y coordinate), the lip bottom corner (y coordinate), the lip height, the lip protrusion (z coordinate) and the chin center (x and y coordinates).

The lip width is the distance from the lip left corner to the lip right corner and the lip height is the distance between the lip top corner and the lip bottom corner. The lip protrusion is the distance from the lip top corner to the MS Kinect One sensor. All of these features are extracted using a Microsoft Kinect package, such as the *HighDetailFacePoints*.

To obtain better recognition results, a normalization process of the extracted features is performed. In this research work, the normalization process used is the z-score normalization (see Equation 3.1):

$$z = \frac{X - \mu}{\sigma} \tag{3.1}$$

where z is the result of the z-score normalization, X is the current feature value, μ is the mean obtained in one features database and σ is the variance. The features must be normalized due to the major changes of the lips and chin movements.

The features, already normalized, must also have a fixed size. In the training process, the size of the features extracted from the recording of a word can be different from the size of the features extracted of the next word. So, it is needed a length normalization for each frame of features. In this research work, the fixed length of each frame of features extracted is 20, taking into account that the features are extracted during a time period of 1,5 seconds.

After the recording of the words, the features are saved into two directories: training and test. The features for each word are saved in both directories with a split proportion of 80% for the training directory and 20% for the test directory. The training for a word consists of 10 consecutive recordings. So, for each word, 8 recordings are saved in the training directory and 2 recordings are saved in the test directory.

In this research work, there are created the words "play", "pause", "stop", "more_volume", "less_volume", "forward" and "backward". The total number of features extracted are: 7 words * 10 repetitions * 20 frames * 9 different features = 12.600 features.

The normalized and fixed size features extracted are written to a "features.arff" file in both directories. These two files represent the database of the lips and chin features extracted during the recordings of the words.

It is illustrated in Figure 3.23 the face detection by a MS Kinect One camera and the recording of the “play” word.

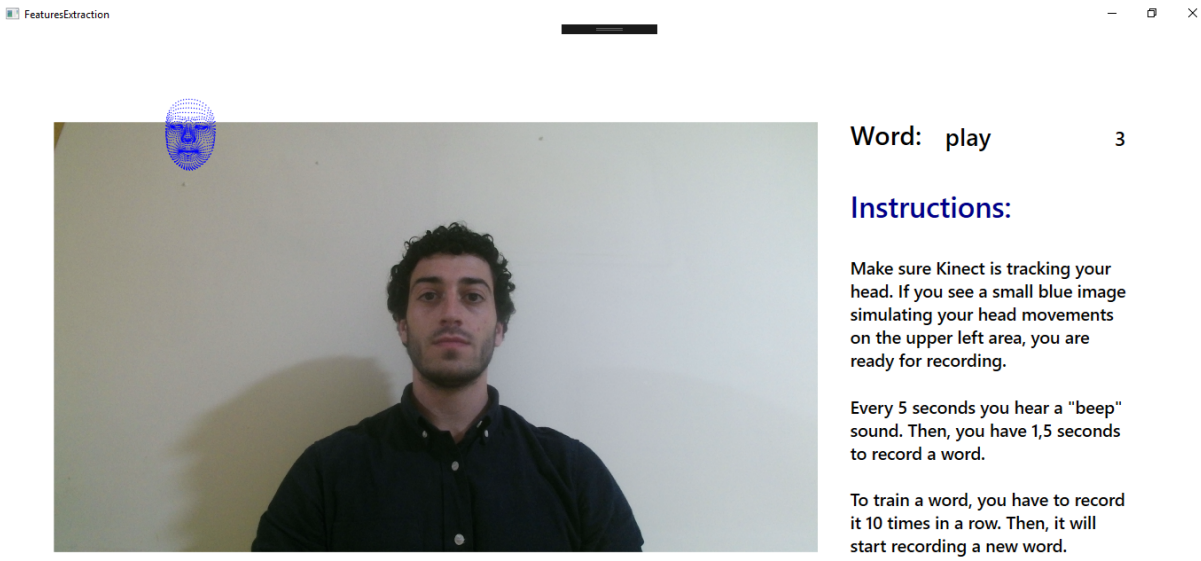


Figure 3.23: Face detection by a MS Kinect One camera and the recording of the “play” word in the “Features Extraction” Modality Component.

It is possible to observe in Figure 3.23 above the “Instructions” section with the indication of the instructions to record a word. It is shown in the figure in the upper left area a small blue image simulating the user’s head movements. That indicates that the MS Kinect One camera is tracking correctly the user’s head. If the small blue image is not appearing, maybe the ambient light conditions are not good or the user is too far or too near from the Kinect sensor.

Every 5 seconds a “beep” sound is produced by the program, so the sound must be turned on, indicating that the program is ready to record a new word. Each word must be recorded 10 times in a row to be fully trained. This number of times was chosen as it is sufficient to have good word recognition results. However, each word could be trained more times and thus increase the percentage of the recognition results.

3.3.2.3 Classification models

The classification process is made, firstly, by evaluating the results of the words classification by some machine learning algorithms: Support Vector Machine, Decision Tree, Naive Bayes and K-Nearest Neighbor. To classify the words, the Weka Software Program[4] was used for the training, validation and classification processes.

[4]https://www.cs.waikato.ac.nz/ml/index.html

In each of the classifiers above, the "features.arff" files from the training and test directory are read and the respective classifier is trained. It is performed a randomization of the order of the instances in the training dataset and the classifier is built. Finally, the classifier is evaluated with the test dataset and it is possible to know the number of correctly classified instances (classes).

In this research work, it is used a "Winner Takes All" perspective, so the best classifier, i.e., the classifier that has the highest number of correctly classified instances, is chosen for serializing the model to a "features.model" file and saving it into a directory.

3.3.2.4 Real time predictions

After the classification process and the serialized model, the user can now use the "Silent Speech" Modality Component to make real time predictions of the words that he or she is uttering in front of a MS Kinect One camera.

As the features are normalized and fixed size in the "Features Extraction" Modality Component, the features that are extracted in real time in the "Silent Speech" Modality Component must also be normalized and fixed with the same size as in the first one.

After a "beep" sound produced by the program, the user can utter a word from the database in front of the MS Kinect One camera. At the same time the user is uttering the word, the normalized and fixed size features are written to a "featuresToPredict.arff" file. The difference from this file to the "features.arff" file from the "Features Extraction" Modality Component is that in the last one, each line (word recorded) terminates with the name of the class (word name) and in the first one, each line (word recorded) terminates with a question mark symbol "?".

After the uttering of the word to predict and the features written to the "featuresToPredict.arff" file, a process of the word prediction is made. Firstly, it is necessary to create a new class of *Instances* of the new extracted features:

```
Instances unlabeledData = new Instances(new java.io.FileReader(pathFeatures
+ "featuresToPredict.arff"));
```

Then, a classifier reads the model created previously by the "Classification Models" Modality Component:

```
Classifier cl = (Classifier)SerializationHelper.read(pathModel +
"features.model");
```

At the same time, the classifier classifies each instance in the "unlabeledData" and predicts the respective class of the instance with the name of the class (name of the word predicted) and its predicted percentage. It is also possible to know the predicted percentages of all of the other classes, as it can be seen in Figure 3.24.

```
Word 0 ->  Predicted: stop
[play : 0.0158730158730159]
[pause : 0.0158730158730159]
[stop : 0.904761904761905]
[more_volume : 0.0158730158730159]
[less_volume : 0.0158730158730159]
[forward : 0.0158730158730159]
[backward : 0.0158730158730159]
```

Figure 3.24: The name of the word predicted and its percentage in the "Silent Speech" Modality Component and the predicted percentages of all of the other classes.

The predicted word is now sent by a lifecycle event message to the Interaction Manager and then it is resent to the "Application" or to an application interface that the user wants to use in response to these event messages. In Figure 3.25 it is presented an example of a lifecycle event message sent by the "Silent Speech" Modality Component to the Interaction Manager with the word predicted by the classifier.

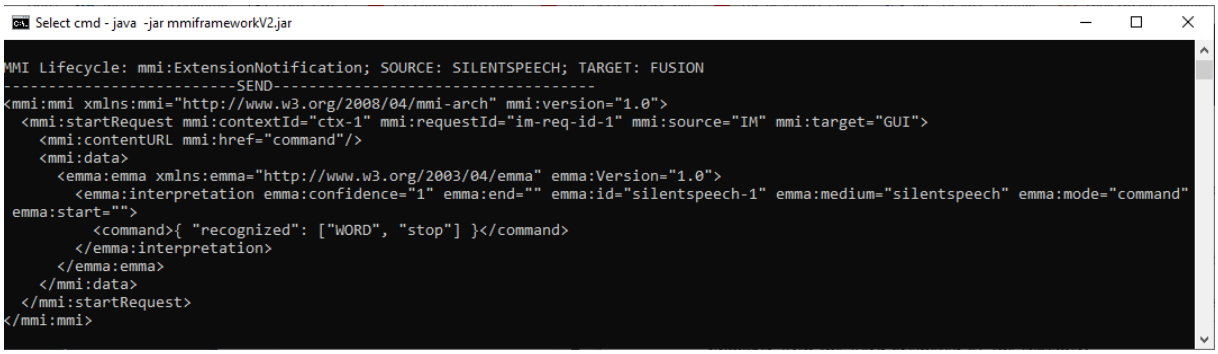


Figure 3.25: An example of a lifecycle event message sent by the "Silent Speech" Modality Component to the Interaction Manager with the "stop" word predicted by the classifier: `["WORD", "stop"]`.

If the predicted word is different from the one that the user uttered, that predicted word is still sent by a lifecycle event message to the Interaction Manager and then it is also resent to the "Application". According to the predicted word, it will generate a different feedback response from the one that the user expected.

Chapter 4

Results

This chapter includes some tests performed about the modalities capabilities, namely for the Generic Gestures Modality, for the Generic Gaze Modality and for the Generic Silent Speech Modality. It is demonstrated a proof-of-concept application using the 3 modalities that controls the VLC, a Video LAN (Local Area Network) Client. The VLC is a multimedia content player.

It is also presented an evaluation done by developers and its results in order to evaluate the facility of use of the Generic Gestures Modality.

4.1 Tests of Modalities Capabilities

Two subjects participated in the evaluation of the system: Tester1 (author of the research work, finalist of the Integrated Masters Degree in Computer and Telematics Engineering, 25 year old male) and Tester2 (student of the Integrated Masters Degree in Computer and Telematics Engineering, 22 year old male).

The Tester2 had never used a MS Kinect before nor an eye tracker. Also, regarding the Generic Gaze Modality, both testers do not use glasses. If any of the testers wore glasses, he or she would have to be more cautious when calibrating the eyes with the Tobii Eye Tracker 4C device.

4.1.1 Generic Gestures Modality

A set of 5 gestures were created by Tester1 and added to the modality for evaluation purposes. The gestures were created with the *Kinect Studio* software program and there were performed 10 repetitions for each gesture. The time spent for the creation of the 5 gestures with the *Kinect Studio* was, approximately, 4 minutes.

Afterwards, it was needed to label each one of the 5 gestures and build the whole solution with the VGB program, in order to generate the "Gestures.gbd" configuration file. This task lasted, approximately, 15 minutes.

Firstly, the gestures recognition accuracy was evaluated for the Tester1, varying the distance from the MS Kinect One camera (100 cm, 150 cm and 200 cm). 10 repetitions for each gesture were performed.

The same test was repeated for the Tester2, but now with the distance from the MS Kinect One camera of 150 cm and a different gestures database (from the Tester1, previously created).

It was chosen this distance because the best results for the Tester1, varying three different distances, was the 150 cm one. Results are shown in Table 4.1.

Table 4.1: The gestures recognition accuracy results, in percentage, for the Tester1 and the Tester2, varying the distance from the MS Kinect One camera.

Gesture	**Tester1 Hit (%)**			**Tester2 Hit (%)**
	Distance (cm)			**Distance (cm)**
	100	**150**	**200**	**150**
Push	75,7%	76,9%	74,9%	72,7%
Swipe Left	76,5%	78,4%	76,7%	76,1%
Swipe Right	72,8%	76,1%	75,9%	74,8%
Swipe Up	**79,6%**	**81,9%**	**83,8%**	**78,1%**
Swipe Down	74,9%	78,7%	77,4%	74,6%
Average	75,9%	**78,4%**	77,7%	75,3%

It is possible to visualize in Table 4.1 that the best recognition results for the Tester1 were obtained for the 150 cm distance, as it was previously referred. Additionally, in respect to the gestures, the ones that got the best results were the "Swipe Up" and the "Swipe Down", possibly because they are the ones that differentiate the most from the other ones.

It is also possible to observe that the recognition results for the Tester2 are slightly lower than the ones for the Tester1. Certainly the reason why it happened was because the gestures were trained by a different user (Tester1) and the execution of the gestures was rather differently. Even so, the accuracy results showed that the system is capable of being performed by a different user from the one that trained the database. The maximum difference results between Tester1 and Tester2 was 4,2% and the best results were also for the "Swipe Up" gesture.

4.1.2 Generic Gaze Modality

In order to evaluate the modality, it was created a visual application with 10 zones / figures. These figures correspond to points of interest in the gaze gestures. The gaze gestures created in the "Application" Modality Component, in the "GazeEvents()" method in the "MainWindow.xaml.cs" file were as follows:

```
gazeEvents.Add("fixation_azul");
gazeEvents.Add("swipe_amarelo_verde");
gazeEvents.Add("sequence_ponto1_ponto2_ponto3_ponto4");
```

The time spent for the creation of the fixation, the swipe and the sequence gaze gestures was, approximately, 1 minute. It is presented in Figure 4.1 the Application interface developed with the display of the figures and the images associated to each of them.

Figure 4.1: The Application interface developed with the display of the figures and the images associated to each of them.

In the Generic Gaze Modality, it was evaluated the system by comparing the modality with 2 exclusive methods: (1) interacting with only eye gaze and (2) interacting with only a computer mouse.

With a stopwatch, it was marked the time duration for the performance of the fixation, the swipe and the sequence gaze gestures with an eye gaze interaction and with a mouse. For the fixation gaze gesture, it was used the figure where there is written “azul”, for the swipe gaze gesture, the two figures that were used were the ones that have written “amarelo” and “verde”, respectively, and for the sequence gaze gesture, there were used the four circle figures in the four corners of the visual application above in Figure 4.1.

Regarding the eye gaze interaction, firstly, the tester directs his gaze to the upper left corner of the visual application, then it is initialized the time with the stopwatch and it is only stopped when the tester performs the respective gaze gesture, by his gaze.

It is executed the same experiment, but now with a mouse. Firstly, the tester points the mouse to the upper left corner of the visual application, then it is initialized the time with the stopwatch and it is only stopped when the tester performs the respective gaze gesture, by pointing the mouse to its respective regions. Both experiments were evaluated for Tester1 and Tester2.

It is shown in Table 4.2 the duration time results, in milliseconds, of the Tester1 and Tester2, for an eye gaze and a mouse interaction.

Table 4.2: The duration time results, in milliseconds, of the Tester1 and Tester2, for an eye gaze and a mouse interaction.

Gaze gesture	**Time duration (ms)**			
	Tester1		**Tester2**	
	Eye gaze	**Mouse**	**Eye gaze**	**Mouse**
Fixation	**33**	89	39	98
Swipe	**92**	248	101	259
Sequence	**166**	472	189	480

It can be observed that the time taken to perform the fixation, the swipe and the sequence gaze gestures are lower for an eye gaze interaction. The fact that the user does not need to move his or her hand to the location points with the computer mouse, but only by gazing with his or her eyes, turns the interaction faster.

Comparing the results of the Tester1 with the Tester2, the time durations of the Tester1 are slightly lower, probably because of the more experience from Tester1 in using the Tobii Eye Tracker 4C device.

This modality allowed to quickly develop an alternative faster than the use of the mouse. In a short time, it was possible to develop an interactive application using only the eye gaze and more efficient than the mouse.

4.1.3 Generic Silent Speech Modality

This modality was applied in the control scenario of the VLC Media Player in a silent interaction, having been selected 7 words or sequences of 2 words for evaluation purposes. The words created and evaluated in the application system were as follows: “play”, “pause”, “stop”, “more volume”, “less volume”, “forward” and “backward”.

To train the words above in the “FeaturesExtraction” project, it was, approximately, 7 minutes and 30 seconds. This was only the recording of the words examples.

4.1.3.1 Evaluation with Testers

The Tester1 trained each one of the 7 words, creating a database configuration file, for each one of the distances chosen from the Kinect camera (60 cm, 100 cm and 200 cm). Then, the words training models are evaluated by 4 different classifiers: Support-Vector Machine, Decision Tree, Naive Bayes and K-Nearest Neighbor.

The Tester2 performed the same procedure, but now with the distance from the Kinect camera of 60 cm. It was chosen this distance because the best results for the Tester1, varying three different distances, was the 60 cm one. Results are shown in Table 4.3.

Table 4.3: The evaluation of the modality for Tester1 and for Tester2.

	Distance (cm)	**SVM (%)**	**Decision Tree (%)**	**Naive Bayes (%)**	**KNN (%)**
Tester1	**60**	50%	50%	50%	**64,3%**
	100	50%	21,4%	**57,1%**	42,9%
	200	35,7%	35,7%	28,6%	**57,1%**
Tester2	**60**	28,6%	21,4%	**50%**	35,7%

By analyzing Table 4.3, it is possible to conclude that the evaluation results for the Tester1, in percentage, were higher for the database of the words that were recorded from a 60 cm distance from the MS Kinect One camera.

Comparing the evaluation results for both testers, it is possible to observe that, for a 60 cm distance from the Kinect camera, the predicted instances of the Tester1 are around twice as much as the predicted instances of the Tester2.

Additionally, the classifiers that got the highest predicted results for both Tester1 and Tester2 were the Naive Bayes and the K-Nearest Neighbor.

It is presented in Figure 4.2 the confusion matrix of the system accuracy from the *Weka* software program. It was chosen the confusion matrix of the best accuracy result of the Tester1 from Table 4.3 (words database from the 60 cm distance from the MS Kinect camera).

Figure 4.2: The confusion matrix of the system accuracy from the *Weka* software program.

```
a b c d e f g   <-- classified as
2 0 0 0 0 0 0 | a = play
0 1 0 0 0 0 1 | b = pause
0 0 2 0 0 0 0 | c = stop
0 1 0 1 0 0 0 | d = more_volume
0 0 0 0 2 0 0 | e = less_volume
0 0 1 0 0 1 0 | f = forward
0 2 0 0 0 0 0 | g = backward
```

From the confusion matrix analysis, it is possible to conclude that the words “play”, “stop” and “less volume” were 100% well classified. This could have happened because when articulating these words, they can differentiate more than the other words. On the contrary, the word “backward” is 100% incorrectly classified as the word “pause” and the word “pause” is 50% correctly classified as “pause” and 50% incorrectly classified as “backward”, most probably due to the articulation similarities they both have.

4.1.3.2 Live evaluation

Some scenarios were chosen to evaluate this modality. It was evaluated the live words recognition for the Tester1 for different distances from the Kinect camera (60 cm, 100 cm and 200 cm) and his own words database to be compared. Then, the same test was performed but now with a 60 cm distance from the Kinect camera and a different database to be compared (words database from the Tester2).

It was also evaluated the live words recognition for the Tester2 for a 60 cm distance from the Kinect camera in 2 different scenarios. The first scenario was the test with his own words database and the second one was the test with a different database to be compared (words database from the Tester1).

The live evaluation recognition results are presented in Table 4.4.

Table 4.4: The live evaluation recognition results.

Speaker	Database to be compared	Distance (cm)	Hit	Miss	Hit (%)
Tester1	**Tester1**	**60**	21	14	**60%**
		100	18	17	51,4%
		200	16	19	45,7%
	Tester2	**60**	9	26	25,7%
Tester2	**Tester2**	**60**	14	21	**40%**
	Tester1	**60**	10	25	28,6%

By analyzing Table 4.4, it is possible to observe that the best hit ratio recognition results for the Tester1 and for the Tester2 scenarios occurred when the distance from the Kinect camera was 60 cm and the database to be compared was his own. When the distance increased to 100 cm and 200 cm for the Tester1 scenarios, the hit ratio started to decrease.

In relation to the scenarios where the database to be compared is different from the respective tester, the hit predicted percentage decreased around 50% from the best hit result, for both Tester1 and Tester2. This occurred due to of the speaker dependency in silent speech interactions. Silent speech systems depend on the speaker's anatomy and on the movements of his or her articulatory muscles, so that is why the hit ratio was so low.

4.2 Proof-of-concept application using the 3 modalities

The application chosen to be controlled by the 3 generic modalities is the VLC Media Player[1] and it can be used in a silent interaction scenario.

This application, using the gestures, gaze and silent speech modalities, has also some advantages when comparing to a speech recognition modality because the absence of sound in the interaction can achieve better and more efficient results. An application using the VLC Media Player controlled by a speech recognition modality is more prone to recognition errors.

4.2.1 System architecture

The system architecture is the same for all the generic modalities. It is composed by the "VLC Control Application" Modality Component, the Interaction Manager and the respective generic modality. The system architecture diagram of the application with the VLC Media Player is illustrated in Figure 4.3.

[1]The VLC Media Player is a free and open-source, cross-platform media player and streaming media server developed by the VideoLAN project. From *VLC media player - Wikipedia* (2019).

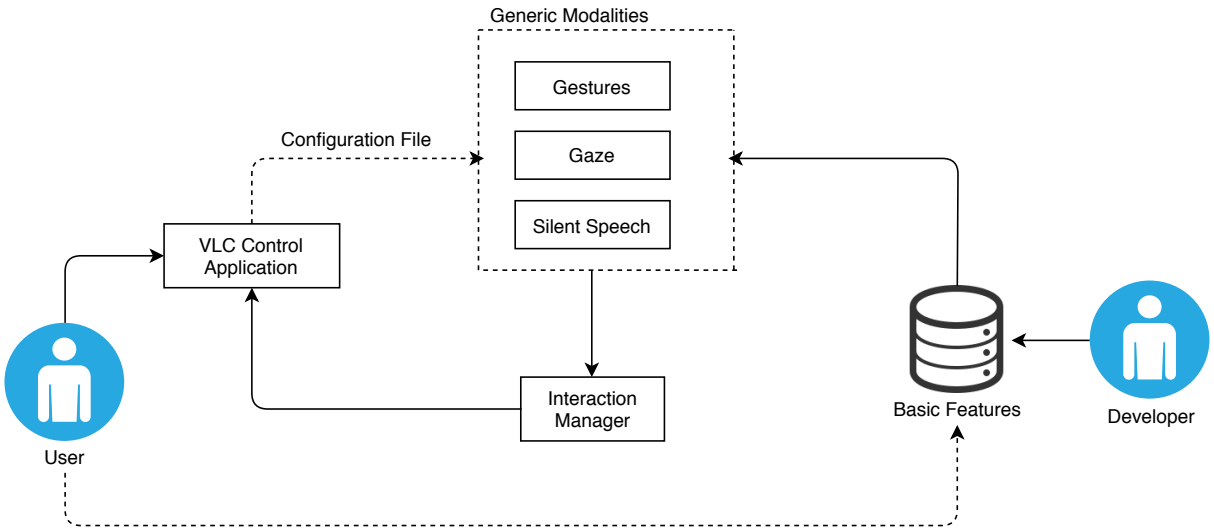


Figure 4.3: The system architecture diagram of the application with the VLC Media Player.

Each modality is responsible for its interaction in real time and, if it recognizes a gesture, gaze or silent speech command, the modality sends the command name and its semantic to the Interaction Manager. The Interaction Manager resends this lifecycle event message to the "VLC Control Application".

The "VLC Control Application" receives the lifecycle event message with the modality command name and its semantic and it generates a proper feedback response in the video that is being displayed by the VLC Media Player.

This application was only created once and there were created the same gestures with the purpose of integrating the Generic Gestures Modality, the Generic Gaze Modality and the Generic Silent Speech Modality.

The gestures created were as follows:

- Play;
- Pause;
- More volume;
- Less volume;
- Backward;
- Forward;
- Stop;
- Help.

It is shown in Figure 4.4 a representation of the visual application interface of the developed VLC Control Application.

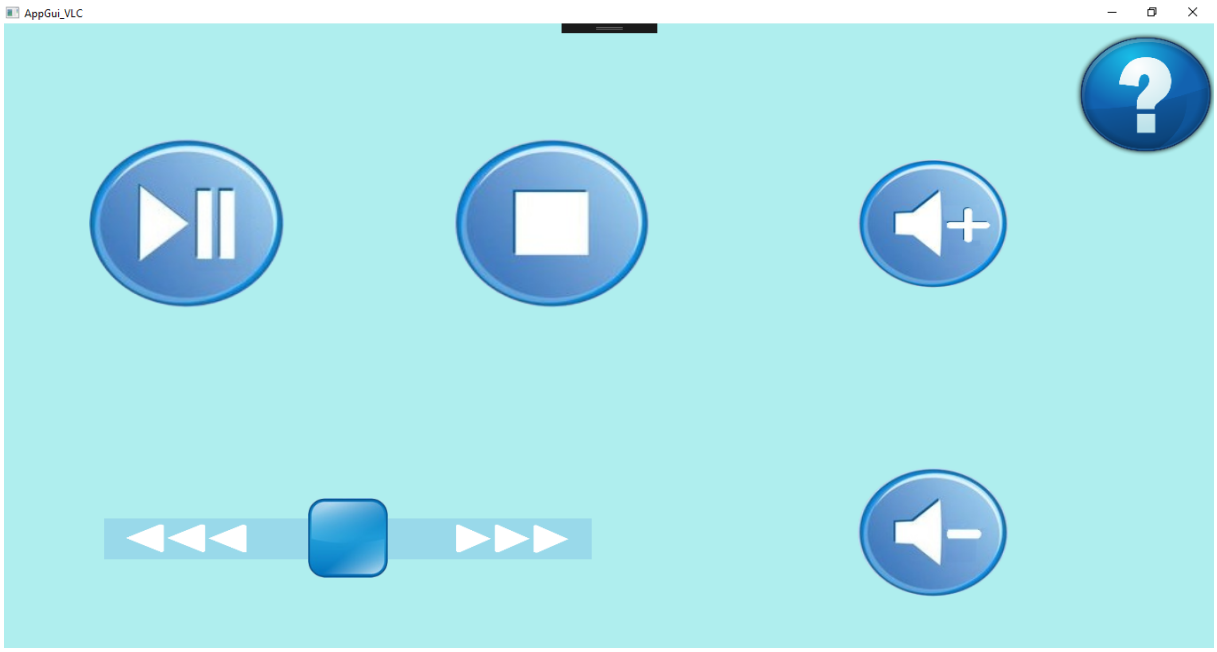


Figure 4.4: Representation of the visual application interface of the developed VLC Control Application.

4.2.2 Functionalities

The functionalities of the VLC Control Application that can be used by each one of the 3 modalities are as follows:

- Play: the video starts displaying;
- Pause: the video is paused;
- Stop: the video is stopped and it returns to the beginning;
- Increase volume: the volume of the video increases by 15 units;
- Decrease volume: the volume of the video decreases by 15 units;
- Backward: the time of the video goes backward for 15 seconds;
- Forward: the time of the video goes forward for 15 seconds;
- Help: a text box appears in order to give a brief explanation to the user about the functionalities of the application.

The inputs available for the 3 modalities are presented in Table 4.5.

Table 4.5: The "commands" that are possible to perform in each one of the 3 modalities.

	Gestures Modality	**Gaze Modality**	**Silent Speech Modality**
Play	"Push"	"Fixation"	Uttered word "play"
Pause	"Push"	"Fixation"	Uttered word "pause"
Stop	"Crossed arms"	"Fixation"	Uttered word "stop"
Increase volume	"Swipe up"	"Fixation"	Uttered words "more volume"
Decrease volume	"Swipe down"	"Fixation"	Uttered words "less volume"
Backward	"Swipe left"	"Swipe"	Uttered word "backward"
Forward	"Swipe right"	"Swipe"	Uttered word "forward"
Help	"Both arms to front"	"Fixation"	Uttered word "help"

4.3 Evaluation by Developers

It was performed an evaluation of the Generic Gestures Modality to evaluate its facility of use by developers. This modality was chosen because it was the one that was more advanced.

The main objective was to test the concept of the ease of use and integration of this generic modality into applications. This evaluation consisted in using the modality in a project followed by a survey, with the questions presented in Table 4.6.

Table 4.6: Questions included in the survey for the evaluation of the Generic Gestures Modality.

Questions included in the survey
What is your age?
Have you ever used the MS Kinect camera before Multimodal Interaction course?
Have you ever developed a gestures interface prior to the Multimodal Interaction course?
How would you rate the facility of integrating the new modality with your application?
How would you rate the facility of adding the gestures (already having the models) to the modality?
How would you rate the training process / creation of new gestures with Visual Gesture Builder?
How many gestures did you add?
Did you add discrete and / or continuous gestures?
What percentage of the total time for the development of the gestures interaction that was spent developing code?
How would you rate the facility of use, as a developer, of the gestures modality?
If you had to include gestures interaction in a new application, would you consider using this generic modality?
What was the point you liked the most about using the modality?
What was the point you disliked the most about using the modality?

It was requested to the students from the Multimodal Interaction course (second semester of 2018/2019 academic year) from University of Aveiro to be the survey population. 10 students, 9 of them were male and 1 was female, with ages between 21 and 26 years participated in the study.

The participants added to the modality between 4 and 10 gestures, essentially discrete ones. Two of the participants had already experienced the MS Kinect camera before the Multimodal Interaction course and for 8 participants it was the first time. Concerning if the participants had developed a gestures interface prior to Multimodal Interaction course, it was the first time for all of them.

4.3.1 Results

This subsection presents the results of the survey. The average responses from the survey participants to some of the questions, in a scale of 1 to 5, are presented in the graph, illustrated in Figure 4.5.

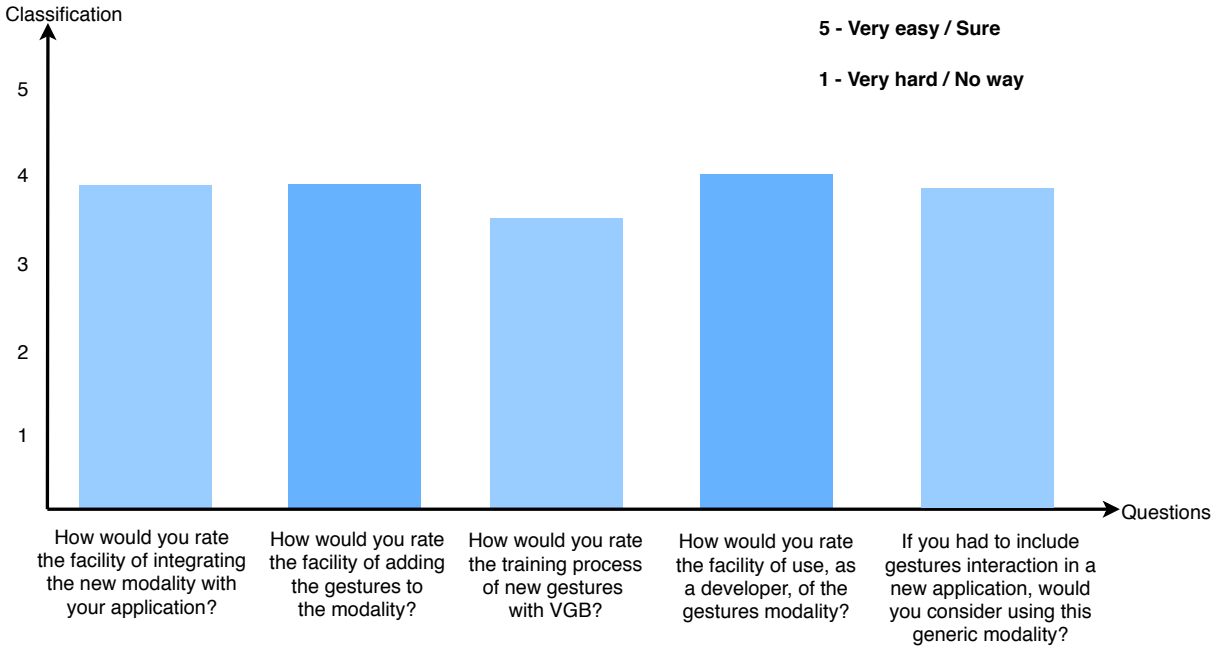


Figure 4.5: The average responses from the survey participants to some of the questions.

The results illustrated in Figure 4.5, regarding the facility of use of the Generic Gestures Modality and its integration into applications, presented a median of 4, in a scale of 1 to 5. As 1 represents "very hard / no way" and the 5 represents "very easy / sure", this means that the participants, although this was the first time they have developed a gestures interface, classified the Generic Gestures Modality as easy to use.

In relation to the question about the training process of new gestures with the VGB program, the results obtained were 40% in the scales 4 and 5 and 60% in the scales 2 and 3, of which 40% were 3. The creation of new gestures involves performing the gestures several times in front of a MS Kinect camera, so having basic gestures in the generic modality is a great feature to the user.

Regarding the question about the percentage of the total time for the development of the gestures interaction that was spent developing code, no one reported the use of more than 75% and more than half of the participants used less than 25%.

The question "How would you rate the facility of use, as a developer, of the gestures modality?" obtained a result of 90% in the scales 4 and 5, of which 70% were 4.

Concerning the question if the participants had to include gestures interaction in a new application, if they consider using this generic modality, the results obtained were 70% in the scales 4 and 5. In a scale of 1 to 5, being 1 of "no way" and 5 of "sure", it was also a good result because this was the first time all participants used gestures interaction in an application and this modality can give them more motivation to develop new multimodal applications.

There were also obtained some responses about the points that the participants liked the most about using the modality. Part of the responses were related to the gestures interaction in general or to the training process and choice of gestures and were not relevant to the objectives of the intended evaluation. The responses that were related to the generic modality are presented in Table 4.7.

Table 4.7: Points that the participants liked the most, related to the generic modality.

Points that the participants liked the most, related to the generic modality
Ease of implementation and use
Simplicity in adding the gestures that are created
Simple and direct
Integration with existing code / application

Lastly, the points that most participants disliked the most were about limitations from the Kinect camera, i.e., the fact that it did not recognize the way as expected some specific trained gestures. In some way, the Kinect camera drove them away from the implementation.

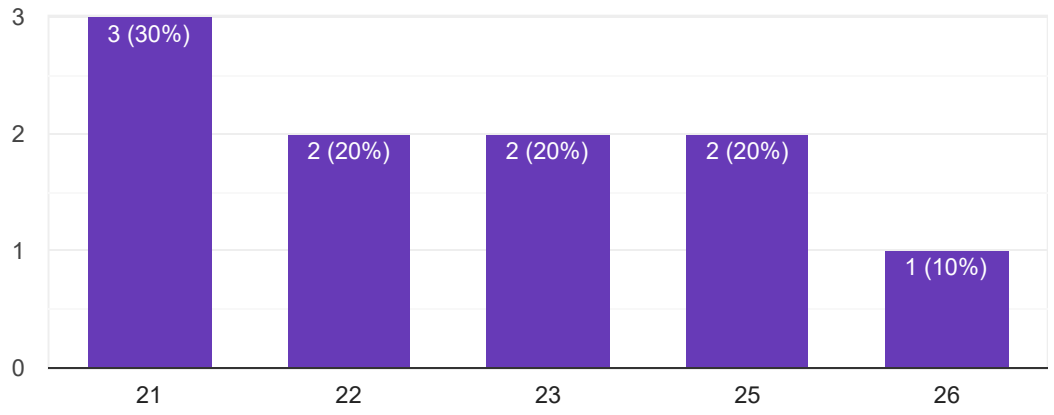
The detailed results of the survey are illustrated in the next pages.

IM - Gestos

10 respostas

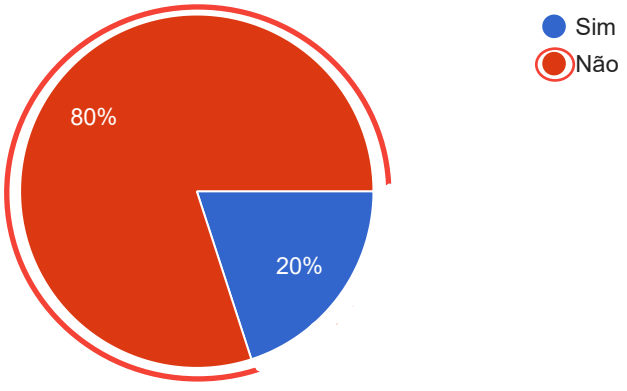
Qual a sua idade?

10 respostas



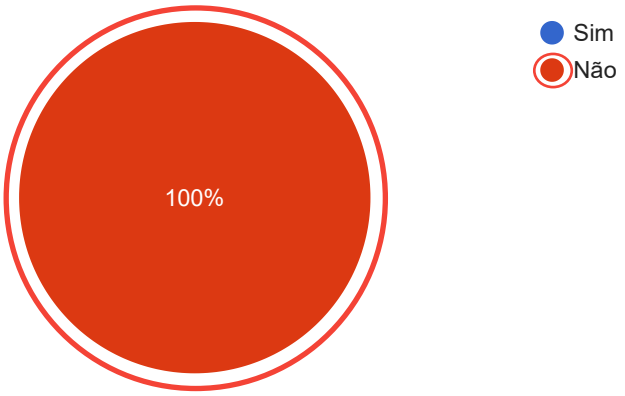
Já alguma vez tinha usado a Kinect antes de Interação Multimodal?

10 respostas



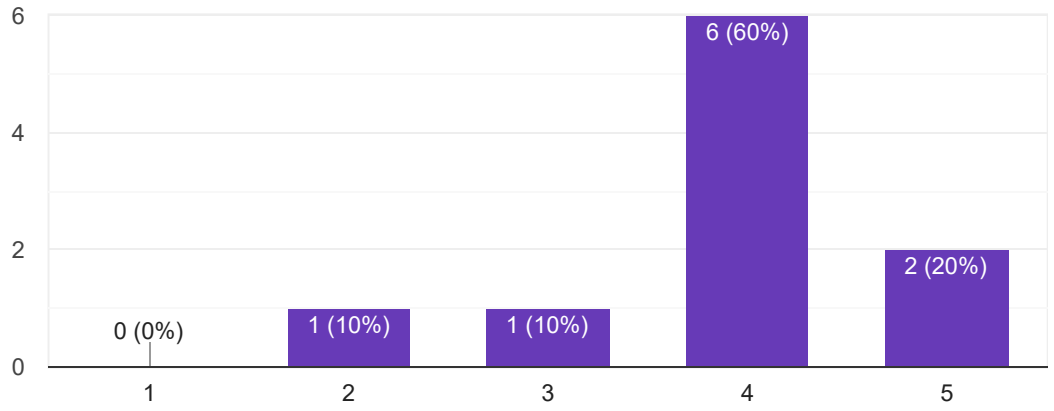
Já alguma vez tinha desenvolvido uma interface por gestos antes de Interação Multimodal?

10 respostas



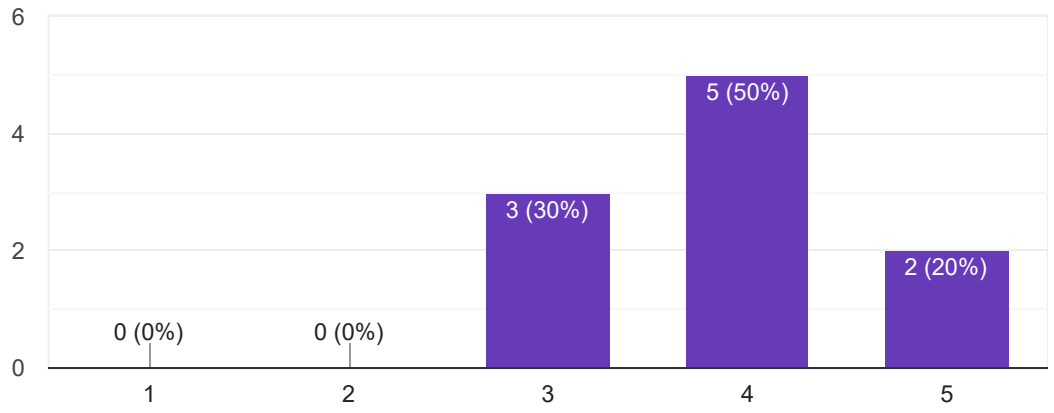
Como classificaria a facilidade de integrar a nova modalidade com a sua aplicação?

10 respostas



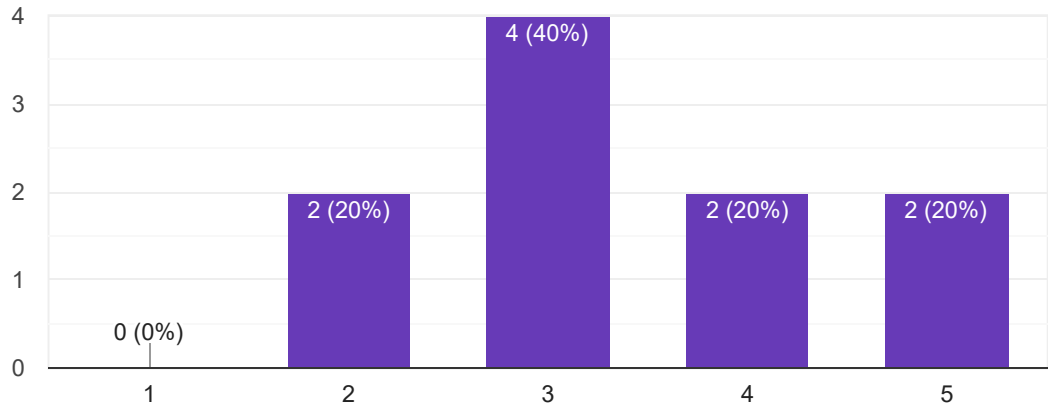
Como classificaria a facilidade de adicionar à modalidade os gestos (tendo já os modelos) que considerou necessários?

10 respostas



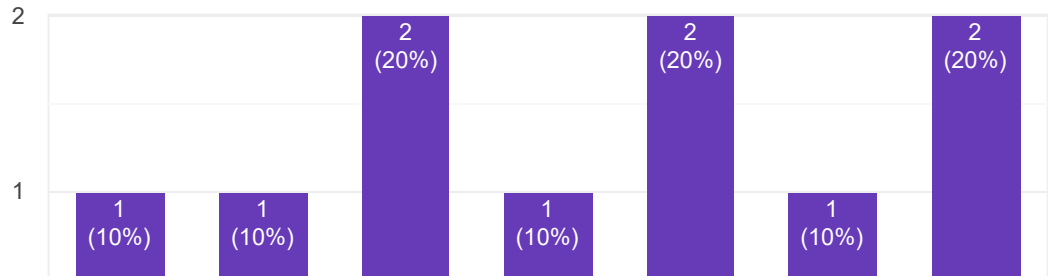
Como classificaria o processo de treino / criação de novos gestos com o Visual Gesture Builder?

10 respostas



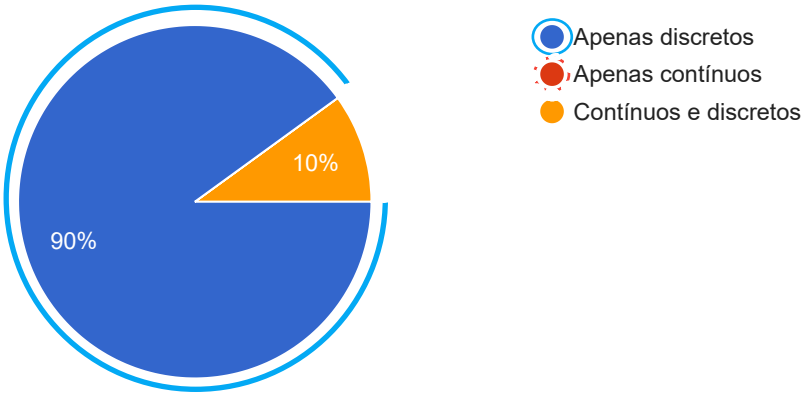
Quantos gestos adicionou?

10 respostas



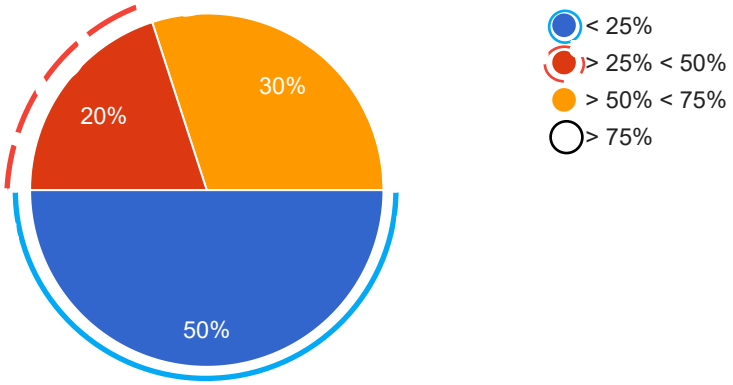
Adicionou os gestos discretos e/ou contínuos?

10 respostas



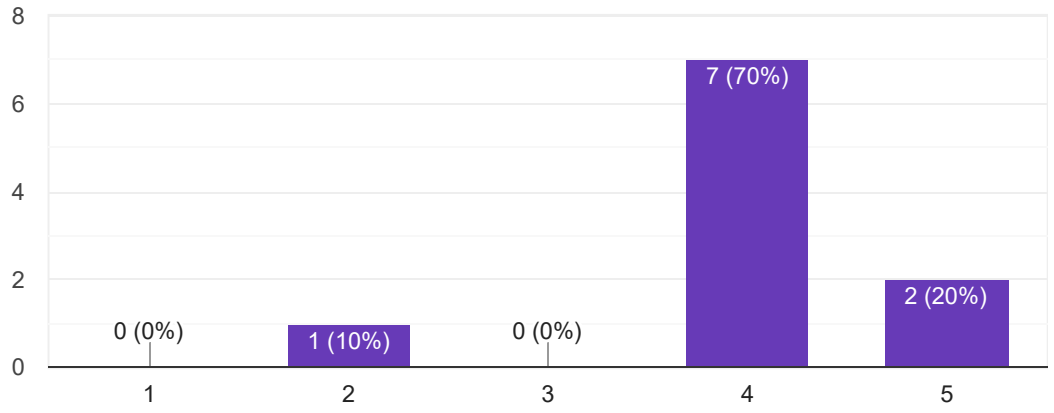
Qual a percentagem do tempo total para o desenvolvimento da interação dos gestos que foi gasto a desenvolver código?

10 respostas



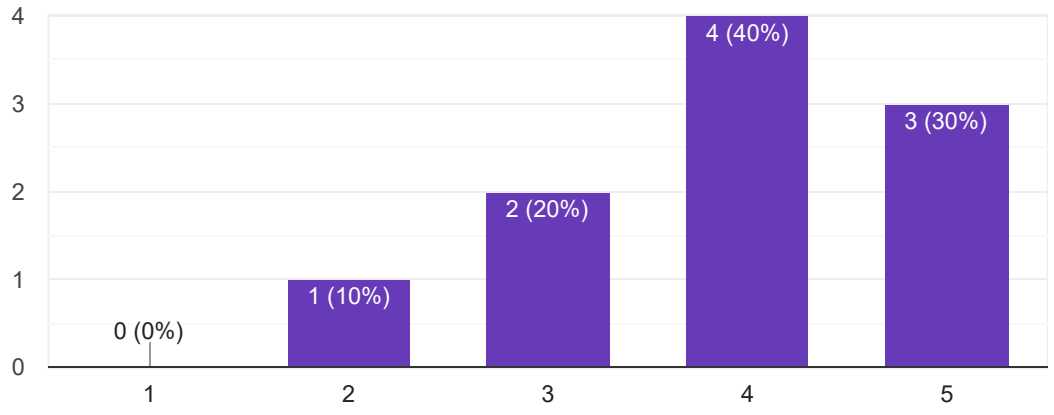
Em termos gerais, como classificaria a facilidade de utilização, como developer, da modalidade de gestos?

10 respostas



Se tivesse de incluir interação por gestos numa nova aplicação, consideraria a utilização desta modalidade genérica?

10 respostas



Qual foi o ponto que mais gostou em usar a modalidade?

10 respostas

Facilidade de implementação.

Possibilidade de poder utilizar gestos do dia a dia

Simplicidade em adicionar os gestos que criamos

Simples e direto

Pode facilitar-nos a vida em certas situações

Intuitividade

Ser um método alternativo não convencional

Reconhecimento de gestos usados em cenários reais para, por exemplo, lançar dados, ou fazer outros gestos que são usados no quotidiano para interagir com pessoas

A integração com o código / aplicação existente

Fácil de utilizar.

Qual foi o ponto que menos gostou em usar a modalidade?

10 respostas

O Kinect por vezes não era reconhecido pelo pc e a única solução era reiniciar o computador.

Dificuldade em arranjar gestos para certas ações

Falta de documentação, apesar de ser simples de se usar

Ao adicionar gestos o ficheiro xml cria novos id's para gestos existentes dependendo da ordem alfabética

O ter de treinar incansavelmente e mesmo assim falhar alguns gestos

Baixa confiança

...

Não reconhecimento das diferentes posições da mão/dedos, devendo-se às limitações do kinect

O facto de após os modelos terem sido gravados o Kinect não detetar muito bem os gestos gravados, tinham de estar numa certa posição para um reconhecimento exato de acordo com o gesto definido. Não gostei de usar o Kinect e a grande parte do tempo passado foi a tentar arranjar uma solução que ao menos nos permitisse fazer a modalidade.

Não era possível alterar a confiança (corrigido) e devia enviar mensagem sempre que deteta um utilizador.

Chapter 5

Conclusions

This chapter presents the summary of work and the several tasks that were performed to develop this research work. The main results are presented and they are analyzed with a brief discussion about them in light of the objectives set at the beginning.

Lastly, it is concluded with some comments about future work and some improvements that could be performed to the current stage of the work for the generic modalities.

5.1 Summary of work

This work started by defining possible silent interaction scenarios and studying the forms of interaction involved, such as gestures, gaze and silent speech interfaces.

A set of requirements were derived and defined as objective the development of three generic modalities, namely the Generic Gestures Modality, the Generic Gaze Modality and the Generic Silent Speech Modality.

The development began with the gestures modality due to the existence of work and knowledge acquired in the curricular unit Multimodal Interaction, following gaze - where more work was done - and ending in a first embryonic version of a silent speech interface.

5.2 Main results

The main results of this research work are:

- the concept of multimodal interactive systems that enable the use of multiple forms of interaction, such as interactions by gestures, gaze and silent speech;
- the three generic modalities and the respective architectures that allow developers to easily integrate them into multimodal applications;
- the survey evaluation of the Generic Gestures Modality and its results showing that it is simple and easy to integrate and use.

5.3 Discussion

In general, the goals of this research work (three generic modalities with a multimodal architecture that provides an easy integration into applications and a proof-of-concept appli-

cation) have been accomplished. The development of the three generic modalities followed the same overall design and development approach: to be independent and decoupled from the multimodal applications and to have some basic modules, supporting a set of features considered important to enable that they can be easily tested, at any time.

The expectations for the use of the generic modalities have been reached: to simplify their integration into applications for developers. The results obtained in the survey evaluation and in each one of the generic modalities confirmed that the method to integrate each modality into applications facilitate the work of developers to adopt multimodal interactions. The positive results, even taking into account the limitations of each modality, also showed that the gestures, gaze and silent speech modalities are intuitive and, at the same time, efficient interactions, and they can be used in several silent application scenarios.

However, some tasks were not performed due to the lack of time, such as an auto calibration method regarding the Generic Gaze Modality and the features normalization by distance in the Generic Silent Speech Modality.

5.4 Future work

The future work in this research work can go to different applications and scenarios related to social networks, such as Facebook, Twitter and Instagram and also to audio and video applications like Spotify and Netflix.

As well as gestures, gaze and silent speech are forms of non-verbal communication, this research work could also be extended to facial expressions. Facial expressions and emotions are another forms of non-verbal communications and they are part of the human social interactions.

One improvement that could be performed in the system is the reduction of speaker dependency. In the three generic modalities the user can test the system with already available basic modules, supporting features created in the database but, especially in the Generic Silent Speech Modality, the recognition results could be lower due to the lack of initial training features for a different user.

In respect to the Gaze Modality, the changing of the head position while using the system can affect the "gaze trace" of the Tobii Eye Tracker 4C device, and consequently, the direction of the predicted user's gaze and the recognition results. An auto calibration method should be performed in order to provide more freedom to the user concerning the head movements.

Other improvements that could be made in the Generic Silent Speech Modality are normalization of features by distance and the freedom to the user of uttering the words in front of the Kinect camera. The features normalization by distance should be integrated in the system so that the user can test the words but with the possibility of changing his or her head position by depth from the Kinect camera.

It is hard for a Visual Speech Recognition System distinguish between the situation when a person is uttering a word, in order to interact with the system, and when a person is talking to another one, for example. This kind of ability to better serve the user needs to be improved, so that the user can utter a word whenever he or she wants and the system can recognize it.

Bibliography

Afouras, Triantafyllos, Joon Son Chung, and Andrew Zisserman (2018). *Deep Lip Reading: a comparison of models and an online application.*

Almeida, Nuno (2017). "Multimodal interaction: contributions to simplify application development". PhD thesis. University of Aveiro.

Angraini, Yuli (2017). "Cross-Culture Analysis: Cultural Adaptation and Nonverbal Communication in Non-Native English-Speaking Countries". In: 8.2, pp. 239–250. ISSN: 2598-9995. URL: https://www.journal.unrika.ac.id/index.php/jurnalanglo-saxon/article/viewFile/1224/954.

Application programming interface - Wikipedia (2019). URL: https://en.wikipedia.org/wiki/Application_programming_interface (visited on 2019-04-15).

Attwenger, Andrea (2019). *Advantages and Drawbacks of Gesture-based Interaction.* URL: https://www.grin.com/document/369514 (visited on 2019-06-09).

Ballester Ripoll, Marina (2017). "Gesture recognition using a depth sensor and machine learning techniques". In: URL: https://riunet.upv.es/handle/10251/77950.

Bolt, Richard A. et al. (1980). ""Put-that-there"". In: *Proceedings of the 7th annual conference on Computer graphics and interactive techniques - SIGGRAPH '80.* Vol. 14. 3. New York, New York, USA: ACM Press, pp. 262–270. ISBN: 0897910214. DOI: 10.1145/800250.807503. URL: http://portal.acm.org/citation.cfm?doid=800250.807503.

Capriola-Hall, Nicole N. et al. (2018). "The Influence of Social Communication Impairments on Gaze in Adolescents with Social Anxiety Disorder". In: *Child Psychiatry & Human Development* 49.4, pp. 672–679. ISSN: 0009-398X. DOI: 10.1007/s10578-018-0782-z. URL: http://link.springer.com/10.1007/s10578-018-0782-z.

Chiao Wang, Tsai, Chia-Liang Tsai, and Ta Wei Tang (2018). "Exploring Advertising Effectiveness of Tourist Hotels' Marketing Images Containing Nature and Performing Arts: An Eye-Tracking Analysis". In: *Sustainability* 10, p. 3038. DOI: 10.3390/su10093038.

Conn, P. Michael (2016). *Conn's translational neuroscience.* ISBN: 9780128025963.

Cornea - Wikipedia (2019). URL: https://en.wikipedia.org/wiki/Cornea (visited on 2019-03-12).

Custom Gestures End to End with Kinect and Visual Gesture Builder (part 2) (2014). URL: https://channel9.msdn.com/Blogs/k4wdev/Custom-Gestures-End-to-End-with-Kinect-and-Visual-Gesture-Builder-part-2-%7B%5C#%7Dtime=00h05m07s (visited on 2019-05-09).

Dahl, Deborah A. (2013). "The W3C multimodal architecture and interfaces standard". In: *Journal on Multimodal User Interfaces* 7.3, pp. 171–182. ISSN: 1783-7677. DOI: 10.1007/s12193-013-0120-5. URL: http://link.springer.com/10.1007/s12193-013-0120-5.

Day 3 - K-Nearest Neighbors and Bias–Variance Tradeoff (2018). URL: https://medium.com/30-days-of-machine-learning/day-3-k-nearest-neighbors-and-bias-variance-tradeoff-75f84d515bdb (visited on 2019-06-05).

Denby, B et al. (2010). "Silent Speech Interfaces". In: *Speech Communication* 52.4, p. 270. DOI: 10.1016/j.specom.2009.08.002. URL: https://hal.archives-ouvertes.fr/hal-00616227.

Dorai, Sundaram and Cynthia Webster (2015). "The Role of Nonverbal Communication in Service Encounters". In: *A Multidisciplinary Approach to Service Encounters*. Leiden, The Netherlands: Brill. ISBN: 9789004260160. URL: https://brill.com/view/book/edcoll/9789004260160/B9789004260160_011.xml.

Duchowski, Andrew T (2018). *Gaze-based Interaction: A 30 Year Retrospective*. Tech. rep. URL: www.elsevier.com/locate/cag.

Eaves, Michael H., Dale Leathers, and Dale G. Leathers (2017). *Successful Nonverbal Communication*. Routledge. ISBN: 9781315542317. DOI: 10.4324/9781315542317. URL: https://www.taylorfrancis.com/books/9781134881185.

EMMA: Extensible MultiModal Annotation markup language Version 1.1 (2013). URL: https://www.w3.org/TR/emma11/ (visited on 2019-02-20).

Ende, Tobias et al. (2011). "A Human-Centered Approach to Robot Gesture Based Communication within Collaborative Working Processes". In: pp. 3367–3374. DOI: 10.1109/IROS.2011.6094592.

"Eye tracking technology for research - Tobii Pro" (2015). In: URL: https://www.tobiipro.com/.

Fan, Lijie et al. (2018). "End-to-end learning of motion representation for video understanding". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6016–6025.

Ferhat, Onur and Fernando Vilariño (2016). "Low Cost Eye Tracking: The Current Panorama". In: *Computational Intelligence and Neuroscience* 2016, pp. 1–14. DOI: 10.1155/2016/8680541.

First impressions: Microsoft's Kinect gaming system - CNN.com (2010). URL: http://edition.cnn.com/2010/TECH/gaming.gadgets/11/03/kinect.video.game/index.html (visited on 2019-06-17).

Fovea - Wikipedia (2019). URL: https://en.wikipedia.org/wiki/Fovea_centralis (visited on 2019-02-11).

Freitas, João, António Teixeira, Miguel Sales Dias, et al. (2011). *Towards a Multimodal Silent Speech Interface for European Portuguese*. Tech. rep. URL: https://www.intechopen.com/books/speech-technologies/towards-a-multimodal-silent-speech-interface-for-european-portuguese.

Freitas, João, António Teixeira, Samuel Silva, et al. (2016). *An Introduction to Silent Speech Interfaces*, p. 14.

Ganea, Nataşa et al. (2018). "Development of adaptive communication skills in infants of blind parents". In: *Developmental Psychology* 54.12, pp. 2265–2273. ISSN: 1939-0599. DOI: 10.1037/dev0000564. URL: http://doi.apa.org/getdoi.cfm?doi=10.1037/dev0000564.

GestureTek Combines Gesture Control with Immersive Virtual Reality to Enhance Rehabilitation (2018). URL: https://www.fitness-gaming.com/news/health-and-rehab/gesturetek-virtual-reality-rehabilitation.html (visited on 2019-06-12).

Hamdan, Mahmood Ibrahim (2016). "Oculesics as part of body language in the glorious qur'an with reference to translation". In: *Journal Of Al-Frahids Arts* 1.27, pp. 13–31. ISSN: 20749554. URL: https://www.iasj.net/iasj?func=article%7B%5C&%7DaId=117260.

Hassan, Azhar and Maali Abdulhussain (2018). "A Study of Non-Verbal Codes in August Strindberg's The Stronger Stylistic Study". In: *journal of kerbala university* 16.3, pp. 10–15.

Hayes, Bruce (2009). *Introductory phonology*. Wiley-Blackwell, p. 323. ISBN: 9781405184113.

Higgins, E.T. and G.R. Semin (2001). "Communication and Social Psychology". In: *International Encyclopedia of the Social & Behavioral Sciences*. Ed. by Neil J. Smelser and Paul B. Baltes. Oxford: Pergamon, pp. 2296–2299. ISBN: 978-0-08-043076-8. DOI: https://doi.org/10.1016/B0-08-043076-7/01816-7. URL: http://www.sciencedirect.com/science/article/pii/B0080430767018167.

Jaimes, Alejandro and Nicu Sebe (2007). "Multimodal human–computer interaction: A survey". In: *Computer Vision and Image Understanding* 108.1. Special Issue on Vision for Human-Computer Interaction, pp. 116–134. ISSN: 1077-3142. DOI: https://doi.org/10.1016/j.cviu.2006.10.019. URL: http://www.sciencedirect.com/science/article/pii/S1077314206002335.

Jording, Mathis et al. (2018). "The "Social Gaze Space": A Taxonomy for Gaze-Based Communication in Triadic Interactions". In: *Frontiers in Psychology* 9, p. 226. ISSN: 1664-1078. DOI: 10.3389/fpsyg.2018.00226. URL: http://journal.frontiersin.org/article/10.3389/fpsyg.2018.00226/full.

Kapur, Arnav, Shreyas Kapur, and Pattie Maes (2018). "AlterEgo: A Personalized Wearable Silent Speech Interface Silent Speech Interface; Intelligence Augmentation; Peripheral Nerve Interface; Human-Machine Symbiosis". In: DOI: 10.1145/3172944.3172977. URL: https://doi.org/10.1145/3172944.3172977.

Kerr-Gaffney, Jess, Amy Harrison, and Kate Tchanturia (2018). "Eye-tracking research in eating disorders: A systematic review". In: *International Journal of Eating Disorders* 52. DOI: 10.1002/eat.22998.

Khamis, Mohamed et al. (2018). "GazeDrone: Mobile Eye-Based Interaction in Public Space Without Augmenting the User". In: pp. 66–71. DOI: 10.1145/3213526.3213539.

Kinect socket server hack brings gesture recognition to Flash, Silverlight, and more (2011). URL: http://jasenrosse.blogspot.com/2011/01/kinect-socket-server-hack-brings_20.html (visited on 2019-06-19).

Kinesics - Wikipedia (2019). URL: https://en.wikipedia.org/wiki/Kinesics (visited on 2019-04-28).

Knapp, Mark L. et al. (2013). *Nonverbal communication in human interaction*, p. 510. ISBN: 1133311598. URL: https://books.google.pt/books/about/Nonverbal_Communication_in_Human_Interac.html?id=-g7hkSR_mLoC&redir_esc=y.

Korkiakangas, Terhi (2018). *Communication, Gaze and Autism: A Multimodal Interaction Perspective*. ISBN: 978-1-138-65655-0. DOI: 10.4324/9781315621852.

Krahmer, Emiel, Marc Swerts, and Suleman Shahid (2012). "Video-mediated and co-present gameplay: Effects of mutual gaze on game experience, expressiveness and perceived social presence". In: *Interacting with Computers* 24.4, pp. 292–305. ISSN: 0953-5438. DOI: 10.1016/j.intcom.2012.04.006. eprint: http://oup.prod.sis.lan/iwc/article-pdf/24/4/292/2015009/iwc24-0292.pdf. URL: https://dx.doi.org/10.1016/j.intcom.2012.04.006.

Lambrecht, Jens (2019). *Gesture-based Interaction - Gestalt Robotics*. URL: https://www.gestalt-robotics.com/gesture-based-interaction.html (visited on 2019-06-09).

Lankes, Michael et al. (2018). "Socialeyes: social gaze in collaborative 3D games". In: pp. 1–10. DOI: 10.1145/3235765.3235766.

Leontovich, Olga and Marianna Gulyaeva (2018). "Refusal to communicate as a positive and negative communication strategy". In: *DISCOURSE and INTER ACTION* 11. DOI: 10.5817/DI2018-1-52. URL: https://doi.org/10.5817/DI2018-1-52.

Liew, Alan Wee-Chung and Shilin Wang (2008). "Visual Speech Recognition: Lip Segmentation and Mapping". In: p. 390. DOI: 10.4018/978-1-60566-186-5.

Liu, Hui et al. (2015). "Comparison of four Adaboost algorithm based artificial neural networks in wind speed predictions". In: *Energy Conversion and Management* 92, pp. 67–81. ISSN: 0196-8904. DOI: https://doi.org/10.1016/j.enconman.2014.12.053. URL: http://www.sciencedirect.com/science/article/pii/S0196890414010929.

Long short-term memory - Wikipedia (2019). URL: https://en.wikipedia.org/wiki/Long_short-term_memory (visited on 2019-03-26).

Magre, Ritesh A and Ajit S Ghodke (2019). "Robust feature extraction for visual speech and speaker recognition". In:

Model–view–controller - Wikipedia (2019). URL: https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller (visited on 2019-06-12).

Multilayer perceptron - Wikipedia (2019). URL: https://en.wikipedia.org/wiki/Multilayer_perceptron (visited on 2019-05-18).

Multimodal Interaction (2012). URL: https://www.w3.org/TR/mmi-arch/ (visited on 2019-03-09).

Non-Verbal Communication (2014). URL: https://www.slideshare.net/nijazn/nonverbal-communication-37341504 (visited on 2019-05-10).

Ocutrx Vision Technologies Unveils Groundbreaking Oculenz AR Cellular with Eye-Tracking Glasses Design (2019). URL: http://vrdevelopernet.digitalmedianet.com/2019/02/26/ocutrx-vision-technologies-unveils-groundbreaking-oculenz-ar-cellular-with-eye-tracking-glasses-design/ (visited on 2019-05-24).

Othman, Mohammad et al. (2017). "CrowdEyes: Crowdsourcing for Robust Real-World Mobile Eye Tracking". In: DOI: 10.1145/3098279.3098559.

Oxytocin Improves Social Cognition In Schizophrenia: Bipolar Network News (2013). URL: http://bipolarnews.org/?p=1735 (visited on 2019-06-21).

Pearson Product-Moment Correlation - Laerd Statistics (2018). URL: https://statistics.laerd.com/statistical-guides/pearson-correlation-coefficient-statistical-guide.php (visited on 2019-04-15).

Pham, Binh Thai et al. (2017). "A comparative study of sequential minimal optimization-based support vector machines, vote feature intervals, and logistic regression in landslide susceptibility assessment using GIS". In: *Environmental Earth Sciences* 76.10, p. 371. ISSN: 1866-6299. DOI: 10.1007/s12665-017-6689-3. URL: https://doi.org/10.1007/s12665-017-6689-3.

Proxemics - Wikipedia (2019). URL: https://en.wikipedia.org/wiki/Proxemics (visited on 2019-04-28).

Reyna, Martín Hernández et al. (2018). "A 3D playful framework for learning the components of a wind turbine, using Kinect". In: *International Journal of Education and Development using ICT* 14.2. ISSN: 1814-0556. URL: https://www.learntechlib.org/p/184686.

Sibert, Linda E and Robert J K Jacob (2000). *Evaluation of Eye Gaze Interaction*. Tech. rep. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.40.5534%7B%5C&%7Drep=rep1%7B%5C&%7Dtype=pdf.

Sign Language for Children with Autism (2015). URL: https://nurturingoliver.wordpress.com/2015/10/07/sign-language-for-children-with-autism/ (visited on 2019-06-04).

Sun, Ke et al. (2018). "Lip-Interact: Improving Mobile Device Interaction with Silent Speech Commands". In: *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. UIST '18. Berlin, Germany: ACM, pp. 581–593. ISBN: 978-1-4503-5948-1. DOI: 10.1145/3242587.3242599. URL: http://doi.acm.org/10.1145/3242587.3242599.

Support groups vital for mental health (2016). URL: https://www.mailtimes.com.au/story/4217161/support-groups-vital-for-mental-health/?cs=19 (visited on 2019-04-20).

Teixeira, António, Miguel Silva, and Samuel Silva (2019). *Multimodal Interaction curricular unit slides*. University Lecture.

Teixeira, António, Samuel Silva, et al. (2016). *Multimodal Interaction with W3C Standards: Toward Natural User Interfaces to Everything*. 1st. Springer Publishing Company, Incorporated, pp. 271–291. ISBN: 9783319428147.

The Fascinating Science Behind 'Talking' With Your Hands (2016). URL: https://www.huffpost.com/entry/talking-with-hands-gestures_n_56afcfaae4b0b8d7c230414e (visited on 2019-06-20).

The Vocal Tract - Pronuncian (2016). URL: https://pronuncian.com/the-vocal-tract (visited on 2019-03-29).

"Tobii and Microsoft Collaborate to bring Eye Tracking Support in Windows 10" (2017). In: URL: https://www.tobii.com/group/news-media/press-releases/2017/8/tobii-and-microsoft-collaborate-to-bring-eye-tracking-support-in-windows-10/.

Tobii Focuses On Foveated Rendering, More Games Get Eye-Tracking Support (2017). URL: https://www.tomshardware.co.uk/tobii-foveated-rendering-eye-tracking,news-54491.html (visited on 2019-05-02).

Toh, Wei Lin, Susan Rossell, and David Castle (2011). "Current visual scanpath research: A review of investigations into the psychotic, anxiety, and mood disorders". In: *Comprehensive psychiatry* 52, pp. 567–79. DOI: 10.1016/j.comppsych.2010.12.005.

Tummeltshammer, Kristen, Estée C.H. Feldman, and Dima Amso (2018). "Using pupil dilation, eye-blink rate, and the value of mother to investigate reward learning mechanisms in infancy". In: *Developmental Cognitive Neuroscience*. ISSN: 18789293. DOI: 10.1016/j.dcn.2018.12.006. URL: https://linkinghub.elsevier.com/retrieve/pii/S1878929318301002.

Tumuluri, Raj and Nagesh Kharidi (2017). "Developing Portable Context-Aware Multimodal Applications for Connected Devices Using the W3C Multimodal Architecture". In: pp. 173–211. ISBN: 978-3-319-42814-7. DOI: 10.1007/978-3-319-42816-1_9.

Types of Eye Movements [Saccades and Beyond] - iMotions (2019). URL: https://imotions.com/blog/types-of-eye-movements/ (visited on 2019-01-10).

Unger, Michael et al. (2019). "Design and evaluation of an eye tracking support system for the scrub nurse". In: *The International Journal of Medical Robotics and Computer Assisted Surgery* 15.1, e1954. ISSN: 14785951. DOI: 10.1002/rcs.1954. URL: http://doi.wiley.com/10.1002/rcs.1954.

URI - Uniform Resource Identifier - Webopedia (2019). URL: https://www.webopedia.com/TERM/U/URI.html (visited on 2019-05-10).

Van Der Stigchel, Stefan, Martijn Meeter, and Jan Theeuwes (2006). "Eye movement trajectories and what they tell us". In: DOI: 10.1016/j.neubiorev.2005.12.001. URL: www.elsevier.com/locate/neubiorev.

Vidal, Mélodie, Andreas Bulling, and Hans Gellersen (2013). "Pursuits: Spontaneous Interaction with Displays based on Smooth Pursuit Eye Movement and Moving Targets". In: DOI: 10.1145/2493432.2493477. URL: http://dx.doi.org/10.1145/2493432.2493477.

Vigo, Julia Alexandra (2013). *Eye Gaze Tracking for Tracking Reading Progress Spring 2013*. Tech. rep. URL: http://es.aau.dk.

VLC media player - Wikipedia (2019). URL: https://en.wikipedia.org/wiki/VLC_media_player (visited on 2019-06-20).

Wachsmuth, Ipke and Stefan Kopp (2001). "Lifelike Gesture Synthesis and Timing for Conversational Agents". In: vol. 2298, pp. 120–133. DOI: 10.1007/3-540-47873-6_13.

Windows 10 now supports eye tracking for greater accessibility (2017). URL: https://www.techradar.com/news/windows-10-now-supports-eye-tracking-for-greater-accessibility (visited on 2019-05-20).

Xbox One Standalone Kinect Sensor Available in October (2014). URL: https://news.xbox.com/en-us/2014/08/27/xbox-one-standalone-kinect/ (visited on 2019-06-02).

Yargic, Alper and Muzaffer Doğan (2013). "A lip reading application on MS Kinect camera". In: pp. 1–5. ISBN: 978-1-4799-0659-8. DOI: 10.1109/INISTA.2013.6577656.

Zhang, Chi, Rui Yao, and Jinpeng Cai (2017). *Efficient Eye Typing with 9-direction Gaze Estimation*. Tech. rep. arXiv: 1707.00548v1. URL: https://arxiv.org/pdf/1707.00548.pdf.

Zhang, Hui et al. (2017). "Development of novel in silico model for developmental toxicity assessment by using naïve Bayes classifier method". In: *Reproductive Toxicology* 71, pp. 8–15. ISSN: 0890-6238. DOI: https://doi.org/10.1016/j.reprotox.2017.04.005. URL: http://www.sciencedirect.com/science/article/pii/S0890623817301843.

Zhang, Xuebai et al. (2017). "Eye Tracking Based Control System for Natural Human-Computer Interaction". In: *Computational Intelligence and Neuroscience* 2017, pp. 1–9. ISSN: 1687-5265. DOI: 10.1155/2017/5739301. URL: https://www.hindawi.com/journals/cin/2017/5739301/.

Zlatintsi, Athanasia et al. (2018). "A Web-based Real-Time Kinect Application for Gestural Interaction with Virtual Musical Instruments". In: *AM* 18. DOI: 10.1145/3243274.3243297. URL: https://doi.org/10.1145/3243274.3243297.

Zuberbühler, Klaus and Juan-Carlos Gomez (2018). "Primate Intentional Communication". In: *The International Encyclopedia of Anthropology*. Oxford, UK: John Wiley & Sons, Ltd, pp. 1–10. DOI: 10.1002/9781118924396.wbiea2211. URL: http://doi.wiley.com/10.1002/9781118924396.wbiea2211.

Zucchini, Walter, Iain L MacDonald, and Roland Langrock (2016). *Hidden Markov Models for Time Series: An Introduction Using R*.

Appendix A

W3C Lifecycle Events

Table A.1 presents the meaning of all the possible properties that may be included in each one of the lifecycle events.

Table A.1: Possible properties of the lifecycle events (*Multimodal Interaction* 2012).

Lifecycle Events Properties	
Context	A URI[1] that is used to identify the interaction. It must be unique for the lifetime of the system.
Source	A URI that represents the address of the sender of the event.
Target	A URI that represents the address to which the event will be delivered.
RequestID	A unique identifier for a Request/Response pair.
Status	It represents the "Success" or "Failure" of the event.
StatusInfo	Provides additional status information in a Response event of a Request/Response pair.
Data	Provides arbitrary data to any event (optional).

There are detailed the lifecycle events that are included in the W3C Multimodal Interaction architecture. All of these asynchronous events must be supported by the Interaction Manager and the Modality Components.

NewContextRequest / NewContextResponse: A modality component sends a *NewContextRequest* to the IM to request that a new context be created. Then, the IM must respond to this event with a *NewContextResponse* event.

The IM can also create a new context by sending to the modality component a *PrepareRequest* or a *StartRequest* that contains a new context ID to the modality component. The IM is also able to respond to *NewContextRequests* from multiple modality components. In Table A.2 there are shown the properties of the *NewContextRequest* and the *NewContextResponse* lifecycle events.

[1] A Uniform Resource Identifier is a generic term for all types of names and addresses that refer to objects on the World Wide Web. From *URI - Uniform Resource Identifier - Webopedia* (2019).

Table A.2: Properties of the *NewContextRequest* and the *NewContextResponse* lifecycle events (*Multimodal Interaction* 2012).

Lifecycle Events Properties	
NewContextRequest	**NewContextResponse**
RequestID	RequestID
Source	Status
Target	Context
Data (Optional)	StatusInfo (Optional)
	Source
	Target
	Data (Optional)

PrepareRequest / PrepareResponse: The interaction manager sends a *PrepareRequest* that allows the modality components to pre-load markup and prepare to run the applications. The modality components must respond with a *PrepareResponse event* and if they return the event with a "Success" status, they should be ready to run.

In the table A.3 there are shown the properties of the *PrepareRequest* and the *PrepareResponse* lifecycle events.

Table A.3: Properties of the *PrepareRequest* and the *PrepareResponse* lifecycle events (*Multimodal Interaction* 2012).

Lifecycle Events Properties	
PrepareRequest	**PrepareResponse**
RequestID	RequestID
Context	Context
ContentURL (Optional)	Status
Content (Optional)	StatusInfo (Optional)
Source	Source
Target	Target
Data (Optional)	Data (Optional)

StartRequest / StartResponse: The IM sends a *StartRequest* in order to invoke a modality component. The modality component must respond with a *StartResponse* event.

If a modality component receives a new *StartRequest* while it is executing a previous one, it must finish the execution of the previous *StartRequest* and begin the execution of the most recent *StartRequest* event, or reject the new *StartRequest* event, with a *StartResponse* with status value equal to "Failure".

In the table A.4 there are shown the properties of the *StartRequest* and the *StartResponse* lifecycle events.

Table A.4: Properties of the *StartRequest* and the *StartResponse* lifecycle events (*Multimodal Interaction* 2012).

Lifecycle Events Properties	
StartRequest	**StartResponse**
RequestID	RequestID
Context	Context
ContentURL (Optional)	Status
Content (Optional)	StatusInfo (Optional)
Source	Source
Target	Target
Data (Optional)	Data (Optional)

DoneNotification: The *DoneNotification* event indicates the completion of the processing that was initiated by the interaction manager with a *StartRequest* event.

The modality component returns a *DoneNotification* lifecycle event to the IM corresponding to the *StartRequest* event when the first reaches the end of its processing.

In the table A.5 there are shown the properties of the *DoneNotification* lifecycle event.

Table A.5: Properties of the *DoneNotification* lifecycle event (*Multimodal Interaction* 2012).

Lifecycle Event Properties
DoneNotification
RequestID
Context
Status
StatusInfo (Optional)
Source
Target
Data (Optional)

CancelRequest / CancelResponse: The interaction manager sends a *CancelRequest* event to the modality component to stop its processing. The modality component must stop its processing and it must respond with a *CancelResponse* event.

In the table A.6 there are shown the properties of the *CancelRequest* and the *CancelResponse* lifecycle events.

Table A.6: Properties of the *CancelRequest* and the *CancelResponse* lifecycle events (*Multimodal Interaction* 2012).

Lifecycle Events Properties	
CancelRequest	**CancelResponse**
RequestID	RequestID
Context	Context
Source	Status
Target	StatusInfo (Optional)
Data (Optional)	Source
	Target
	Data (Optional)

PauseRequest / PauseResponse: The interaction manager sends a *PauseRequest* event to the modality component to suspend its processing. The modality component must pause its processings or once it determines that it will be unable to pause, it must respond with a *PauseResponse* event.

In the table A.7 there are shown the properties of the *PauseRequest* and the *PauseResponse* lifecycle events.

Table A.7: Properties of the *PauseRequest* and the *PauseResponse* lifecycle events (*Multimodal Interaction* 2012).

Lifecycle Events Properties	
PauseRequest	**PauseResponse**
RequestID	RequestID
Context	Context
Source	Status
Target	StatusInfo (Optional)
Data (Optional)	Source
	Target
	Data (Optional)

ResumeRequest / ResumeResponse: The interaction manager sends a *ResumeRequest* event to the modality component to resume its processing that was paused by a previous *PauseRequest* event.

When there was a *PauseRequest* previously sent by the IM and then a *ResumeRequest*, the component modalities that were paused must continue their processings and respond with a *ResumeResponse* event. If the implementation succeeded is the resuming processing, the "status" value must be "Success" and "Failure" on the contrary.

In the table A.8 there are shown the properties of the *ResumeRequest* and the *ResumeResponse* lifecycle events.

Table A.8: Properties of the *ResumeRequest* and the *ResumeResponse* lifecycle events (*Multimodal Interaction* 2012).

Lifecycle Events Properties	
ResumeRequest	**ResumeResponse**
RequestID	RequestID
Context	Context
Source	Status
Target	StatusInfo (Optional)
Data (Optional)	Source
	Target
	Data (Optional)

ExtensionNotification: The *ExtensionNotification* event can be generated by the interaction manager and by the modality component. It refers to the events that are extensions to the W3C Multimodal Framework.

For example, if a gesture recognition modality application wants to notify the IM that when a gesture was detected, it would generate an *ExtensionNotification* event by the modality component informing the IM that a gesture was recognized.

In the table A.9 there are shown the properties of the *ExtensionNotification* lifecycle event.

Table A.9: Properties of the *ExtensionNotification* lifecycle event (*Multimodal Interaction* 2012).

Lifecycle Events Properties
ExtensionNotification
RequestID
Name
Context
Source
Target
Data (Optional)

ClearContextRequest / ClearContextResponse: The interaction manager sends a *ClearContextRequest* to free all resources associated with the respective context. The modality components must respond with a *ClearContextResponse*, but do not take any particular action in it.

In the table A.10 there are shown the properties of the *ClearContextRequest* and the *ClearContextResponse* lifecycle events.

Table A.10: Properties of the *ClearContextRequest* and the *ClearContextResponse* lifecycle events (*Multimodal Interaction* 2012).

Lifecycle Events Properties	
ClearContextRequest	**ClearContextResponse**
RequestID	RequestID
Context	Context
Source	Status
Target	StatusInfo (Optional)
Data (Optional)	Source
	Target
	Data (Optional)

StatusRequest / StatusResponse: The *StatusRequest* event and the *StatusResponse* event are used to provide keep-alive functionality. The interaction manager or the modality component can send a *StatusRequest* event and the recipient must respond with a *StatusResponse* event, unless the request has specified a context which is unknown to it.

In the table A.11 there are shown the properties of the *StatusRequest* and the *StatusResponse* lifecycle events.

Table A.11: Properties of the *StatusRequest* and the *StatusResponse* lifecycle events (*Multimodal Interaction* 2012).

Lifecycle Events Properties	
StatusRequest	**StatusResponse**
RequestID	RequestID
Context	Context
Source	Status
Target	Source
Data (Optional)	Target
	Data (Optional)