Masters' Degree in Informatics Engineering

Dissertation

Final Report

September 2014

# Natural User Interfaces

Sara João Cardoso Ferreira

sjoao@student.dei.uc.pt

Supervisors:

Professor Daniel Castro Silva

Professor Fernando Penousal Machado

FCTUC DEPARTAMENTO
DE ENGENHARIA INFORMÁTICA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

*"Any darn fool can make something complex;*
*it takes a genius to make something simple."*
Albert Einstein

# Abstract

This project's main subject are Natural User Interfaces. These interfaces' main purpose is to allow the user to interact with computer systems in a more direct and natural way. The popularization of touch and gesture devices in the last few years has allowed for them to become increasingly common and today we are experiencing a transition of interface paradigms from graphical user interfaces to natural user interfaces. However, these interfaces are still being explored, as well as the possibilities that come with them. Whether or not they succeed as a new interface paradigm will depend on whether they are able to provide the user with a truly natural way of interaction. This project's main goal was to study new possibilities for these interfaces. This was achieved through the development of two video games (based on common and / or traditional games) that should reveal a new way to play them, reinventing these games for these new type of interfaces. The diversity of the types of game that were chosen incite the study of new possibilities for natural user interfaces. These video games were developed for *Leap Motion*, a device that enables the creation of natural user interfaces since it allows users to interact with the computer through gesture. The development was made using the *Unity 3D* game engine and, afterwards, evaluations were made in to determine if the goal to create games that explore natural user interfaces was correctly achieved. The results of these evaluations were, overall, positive in all of the age groups included in the evaluation, but were better in the younger ones. The games were later submitted on Airspace, the *Leap Motion*'s app store.

Keywords: Natural User Interfaces, User Interfaces, Leap Motion, Video Games, Gesture.

# Contents

# List of Figures

# List of Tables

# 1. Introduction

The emergence of computer systems brought along the need for a way to allow users to interact with them. This interaction started with command line interfaces, a type of interfaces that allowed for the user to type in commands in order to instruct the machine on what to do. Soon, another kind of interfaces emerged: the graphical user interfaces that allowed the user to interact with the computer in a much more visual way. These interfaces led to the popularization of computers and were the main interface paradigm for many years. Today we are experiencing a new transition of interface paradigms from graphical to natural user interfaces.

Natural User Interfaces intend to allow the user to interact in a more direct and natural way with computer systems. The popularization of touch and gesture devices has allowed for these interfaces to become increasingly common in the last few years.

Natural User Interfaces are this project's main subject. These interfaces are still being explored as well as the possibilities that come with them. For these interfaces to succeed as a new interface paradigm they must find their place in people's lives as did graphical user interfaces when they first appeared.

The project's main goal was to study new possibilities for natural user interfaces. This was achieved through the development of two video games that were based on common games and should reveal a new way to play them, therefore reinventing these games for these new type of interfaces.

This project was created in the context of Vizualyzart, a project of the CDV Lab[1] a research facility at DEI/FCTUC[2] at the University of Coimbra[3].

The video games were developed for *Leap Motion*[4], a recent device that incites the creation of natural user interfaces by allowing users to interact with their computer through hand gestures.

[1]More information available online at `http://cdv.dei.uc.pt`
[2]More information available online at `http://dei.uc.pt`
[3]More information available online at `http://uc.pt`
[4]More information available online at `www.leapmotion.com`

The development was made using the *Unity 3D* game engine which allows for cross-platform development and facilitates the development for *Leap Motion*. After the games were developed and tested, they were submitted to be placed on Airspace[5], the *Leap Motion*'s app store.

This report includes the state of the art, the system description and specification, the planning of the project, the development specifications, evaluations and tests that were made on the games and the conclusions that were drawn from the project as well as an insight on the future work that can be made to continue it.

In the State of the Art (chapter 2), the main subjects approached during the project are exposed and some related work is presented. The main topics that are discussed are User Interfaces (with a bigger focus on Natural User Interfaces) and the History of Games.

The System Description (chapter 3) includes the definition of the main requirements (through user stories and use cases) as well as other artifacts that represent different parts of the system.

The Project Planning (chapter 4) includes the project's main goals and the task definition and how it changed throughout the project. It also explains some of the choices that were made regarding the technology that was used and the games that were chosen. Finally it presents the main challenges that were dealt with during this project.

In the Development, (chapter 5) the system description is further explained for each game, as well as the usability choices that had to be made along the way. All the main aspects of the development phase of the two games are also explained in this chapter.

The Usability Evaluation (chapter 6) begins with an explanation of some of the different methods for Evaluation that are most commonly used and states which of those were chosen to evaluate these games and why. It also presents the tests that were made for each game and the results that were drawn from them.

Finally, the Conclusions (chapter 7) presents a summary of the work that was developed throughout the year and some improvements that could be made to each game and what else could be made if this project was to be further explored.

[5]More information available online at `airspace.leapmotion.com`

# 2. State of the Art

In this chapter some of the key concepts for this project are introduced and some of the work that has been produced in the area is explored. This chapter is divided into three sections: first, an introduction to user interfaces is presented, second, the concept of "natural user interfaces" is further developed, and third, the last section is dedicated to games and its history.

## 2.1 User Interfaces

As long as humans continue to interact with computer systems user interfaces will be needed. But as computer systems evolve, user interfaces evolve with them and new opportunities become possible.

With the variety of choices that users have nowadays, the quality of an interface becomes of critical importance in order for products to be successful. Therefore, the development of good user interfaces is essential and is becoming a major part of software construction [2].

The next sections provide a definition for user interfaces, present a brief history of user interfaces paradigms and elaborate on the user experience concept.

### 2.1.1 Definition

For humans and computer systems to communicate efficiently, the system must have a component that allows interaction. This component is called a User Interface (or UI) [2].

This interface is what creates the connection between a user and a system's underlying technology, and represents everything that the user sees and feels when he is using the system [3]. Figure 2.1 shows Windows XP's user interface which allows the user to communicate with the operating system.

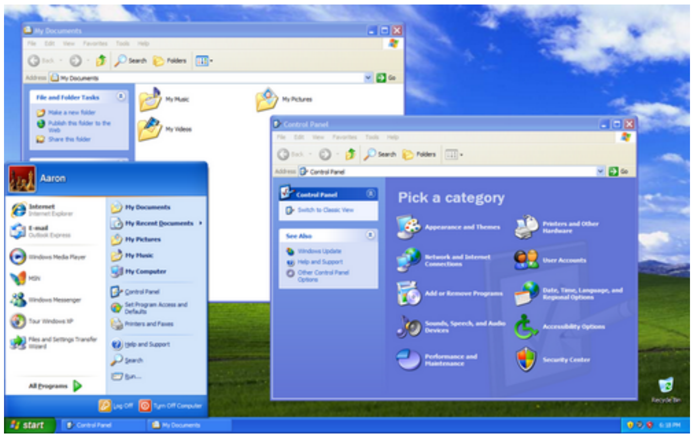


Figure 2.1: Windows XP's user interface.

Designing a user interface includes the design of every aspect of the computing system that is visible to the user and implies both hardware and software components. To work properly, the user interface must be a part of the design process of the computer system from the very beginning, that is, from its conception. Applying a user interface after the system has already been built will result in a poor interface system that is likely to be very inefficient [4].

The main goal of a user interface should be to provide interaction that allows the user to effectively operate and control a computer system. To achieve this, the interface should provide the user with means of input and output.

### 2.1.2 Evolution of interfaces paradigms

The creation of computers brought along the need to find a way humans could interact with them. The first type of interaction that solved this problem was the command line. Through command line interfaces (CLI), users could instruct the computer on what to do by typing in commands. Afterwards came the graphical user interfaces (GUI). These interfaces were much simpler to use, which allowed the use of computers to spread. Nowadays we are witnessing the transition from graphical user interfaces to natural user interfaces (NUI), which was made possible by the appearance of new technologies such as touchscreens and motion-sensing controllers. Even though natural user interfaces still have a long way to go, some researchers are already discussing what will come next: organic user interfaces (OUI) [5]. These four interface paradigms are further explained in the next sections.

***Command Line Interfaces (CLI)***

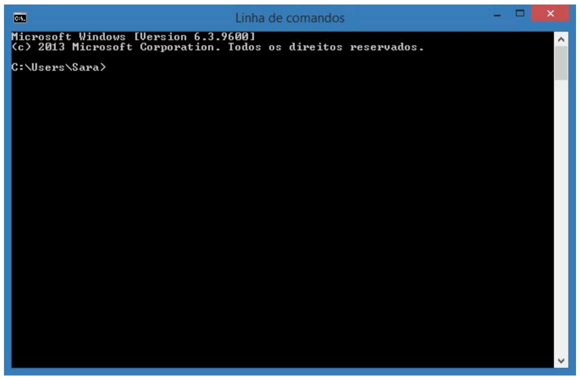
These types of interfaces were the first that allowed users to interact with computers. With command line interfaces, the user enters commands into the computer via keyboard. These commands are textual and must belong to a big list of instructions that the computer is able to process. These interfaces are based on the psychological function of recall: users must remember commands in order to be able to efficiently interact with the computer.

The concept for command line interfaces originated in the 1950s when teletypewriters machines (figure 2.2) were connected to computers which offered results on demand, unlike the other methods used at the time [6].

Figure 2.2: A teletypewriter (Model 33 ASR).

Figure 2.3: The VT100 video terminal used a command line interface.

The command line interface (figure 2.3) presents the user with a lot of commands that can be entered but has very little ways to interact with the computer. Therefore, the user's experience feels disconnected and abstract. Interaction with a computer with this kind of interfaces requires skills from the users and uses a lot of cognitive load. This, unfortunately, means that these interfaces are not for everyone. However, these interfaces are still used in a smaller niche by experienced users. Even though personal computers now come with a graphical user interface, the command line (figure 2.4) is still available for those who want to use it.



```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. Todos os direitos reservados.

C:\Users\Sara>
```

Figure 2.4: Windows 8 command line.

***Graphical User Interfaces (GUI)***

Graphical user interfaces represented a turning point in computing history. The way content is represented, using graphical icons, makes interaction much simpler and easier to use by the general public. GUIs are based on the psychological function of recognition: users recognize graphical objects as representative of functions they want the computer to do. This allowed computers to spread widely and enabled personal computers to appear.

Although these interfaces only became popular in the 1980s, their story begins in 1945 when Vannevar Bush published the article "As we may think" [7] where he stated his ideas about a computing device that would use (what is now called) hyperlink technology to provide information to the users [8] [9] [10]. This article inspired Douglas Englebart to try and build the described computing devices. It was Englebart who invented the computer mouse, to which he called a "X-Y Position Indicator" (figure 2.5) [8] [11]. The mouse was intended to be an integral part of a "graphical windowed interface" [8]. In 1968 Englebart presented a public demonstration of the first GUI: the *NLS* system (figure 2.6).

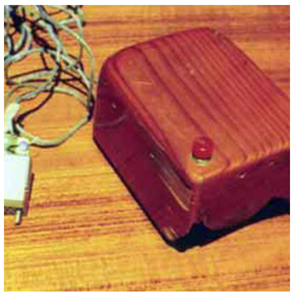


Figure 2.5: The first mouse.

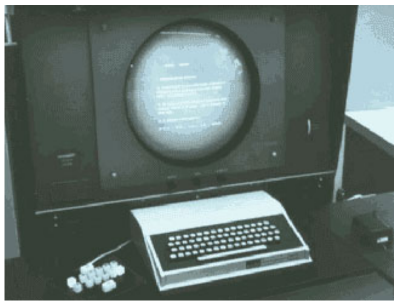


Figure 2.6: The *NLS* system.

Around the same time, another visionary, Ivan Sutherland (figure 2.7), was also creating a graphical user interface. His project was called "*Sketchpad*" and allowed users to manipulate objects on a CRT screen using a light pen [8].

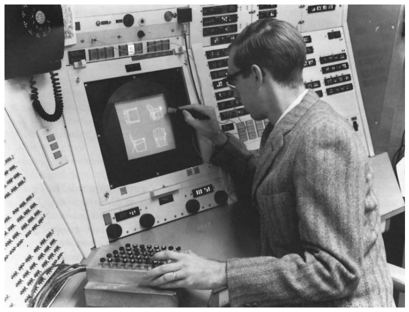


Figure 2.7: Ivan Sutherland and his project: the *Sketchpad*.

It was Xerox PARC[1] that created the first real-life usable GUI. It was called the *Alto* computer (figures 2.8 and 2.9) and was presented in 1974 [8] [9] [10].

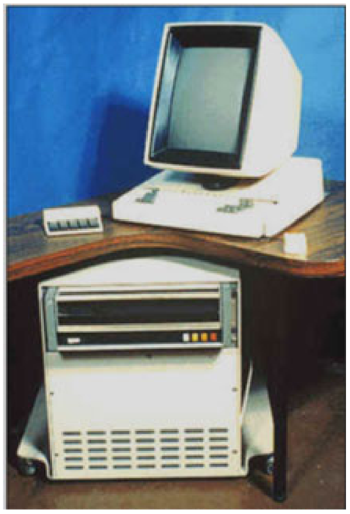
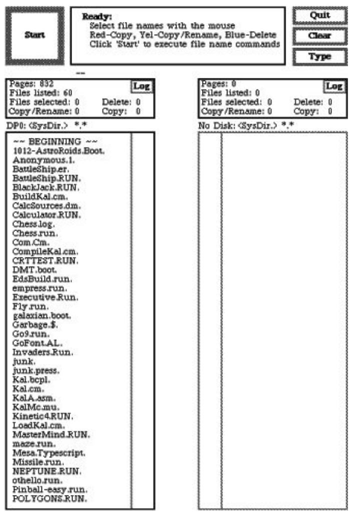


Figure 2.8: The Xerox *Alto*.


```
Start    Ready:
           Select file names with the mouse
           Red-Copy, Yel-Copy/Rename, Blue-Delete
           Click 'Start' to execute file name commands      Quit  Clear  Type
--
Pages: 832                        Log    Pages: 0                          Log
Files listed: 60                         Files listed: 0
Files selected: 0     Delete: 0          Files selected: 0      Delete: 0
Copy/Rename: 0        Copy:   0          Copy/Rename: 0         Copy:   0
DP0: <SysDir.>  *.*                      No Disk: <SysDir.>  *.*
 ~~ BEGINNING ~~
 1012-AstroRoids.Boot.
 Anonymous.1.
 BattleShip.er.
 BattleShip.RUN.
 BlackJack.RUN.
 BuildKal.cm.
 CalcSources.dm.
 Calculator.RUN.
 Chess.log.
 Chess.run.
 Com.Cm.
 CompileKal.cm.
 CRTTEST.RUN.
 DMT.boot.
 EdsBuild.run.
 empress.run.
 Executive.Run.
 Fly.run.
 galaxian.boot.
 Garbage.$.
 Go9.run.
 GoFont.AL.
 Invaders.Run.
 junk.
 junk.press.
 Kal.bcpl.
 Kal.cm.
 KalA.asm.
 KalMc.mu.
 Kinetic4.RUN.
 LoadKal.cm.
 MasterMind.RUN.
 maze.run.
 Mesa.Typescript.
 Missile.run.
 NEPTUNE.RUN.
 othello.run.
 Pinball-easy.run.
 POLYGONS.RUN.
```
Figure 2.9: The Xerox *Alto*'s file system.

In 1979 Apple[2] started working on the computer that would make graphical user interfaces popular and computer systems accessible to the general public. This computer was called *Lisa* and was released in 1983 (figures 2.10 and 2.11).

Figure 2.10: Apple's *Lisa*.

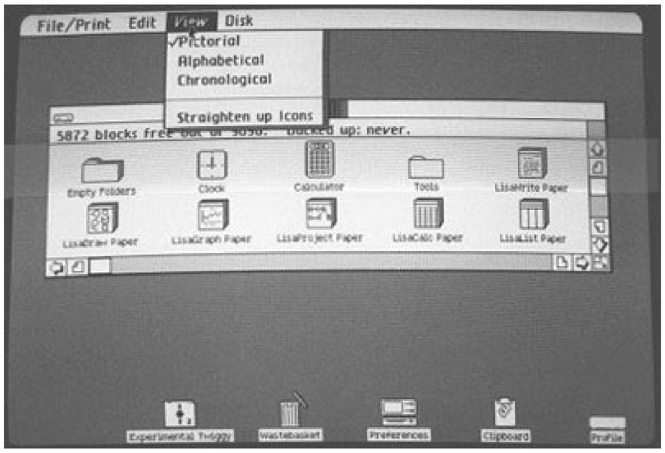


Figure 2.11: *Lisa*'s interface.

The main design principle behind graphical user interfaces is WYSIWYG (What You See Is What You Get). This principle implies that what the user sees on the computer is what he will get as an end result.

[1] Palo Alto Research Center
[2] More information available online at `www.apple.com/`

Another important concept with graphical user interfaces is the concept of WIMP (figure 2.12). WIMP stands for Windows, Icons, Menus and Pointer which are the four main elements that constitute a GUI.

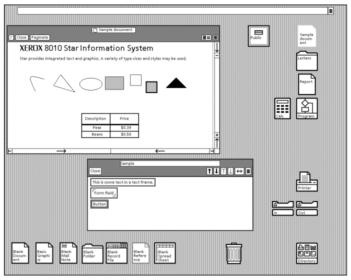


Figure 2.12: Example of an interface that uses the WIMP concept.

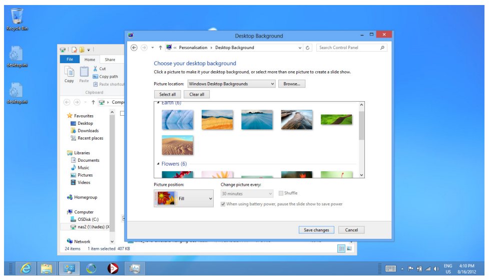


Figure 2.13: Windows 8 graphical user interface.

The interaction is responsive but indirect: the user does not interact directly with the content, therefore an intermediary between them is necessary (usually this intermediary is presented in the form of a mouse and / or a keyboard).

Furthermore, users have a lot of possibilities for interacting with the computer and can easily explore all the options the system has to offer.

These interfaces are still the most common nowadays and widely used for office work. Most desktop and laptop personal computers still use graphical user interfaces and are based on the WIMP concept (figure 2.13).

### ***Natural User Interfaces (NUI)***

Natural user interfaces are becoming more and more common every day. Touchscreens (figure 2.14) and motion-sensing controllers (figure 2.15) were facilitators in spreading this interaction paradigm around the world. These interfaces try to avail of human intuition. While the design principle behind GUI is "What You **See** Is What You Get", NUI try to use our perception of the world, where "What You **Do** Is What You Get".

The content is treated as objects in space, and the interactions are responsive to the environment in which the system is located. The interface should take in the context where it is inserted to suggest what the next action should be.

In NUI, the interaction is direct: the user manipulates the content with their own body (or parts of it) without needing intermediaries such as a mouse or a keyboard.

Examples of these interfaces are the ones present in motion-sensing controlled games and applications (such as the ones created for Nintendo *Wii*[3], Microsoft *Kinect*[4], *Leap Motion*[5] among others.) and also in systems with a touchscreen.

Figure 2.14: Touch surface.

Figure 2.15: Motion Control Technology.

***Organic User Interfaces (OUI)***

User interface experts such as Dennis Wixon predict that OUI will supersede NUI. In organic user interfaces everything is connected and fluid, like in an organic system. These interfaces will use everyday objects as both input and output devices. The interactions between the users and the objects will be completely fluid since the form of the objects will clearly hint the user on how they should be used.

Organic user interfaces should make its user forget that he is using a machine and allow him to experience media and applications as part of his physical environment [12].

Carsten Schwesig, a member of the Interaction Laboratory at Sony Computer Science Laboratories[6] explored the possibility of using analog sensors to develop the Gummi interface concept. This interface was based on the belief that at some point in the future it will be possible to build credit-card sized, flexible computers. The created prototype allows users to browse digital media by bending the device in different ways (figures 2.16 and 2.17) [12].

Organic user interfaces are user interfaces with non-planar displays. These displays may actively or passively change shapes and eventually evolve [13].

[3]More information available online at `www.nintendo.com/wii`
[4]More information available online at `www.microsoft.com/en-us/kinectforwindows`
[5]More information available online at `www.leapmotion.com`
[6]More information available online at `www.sonycsl.co.jp`

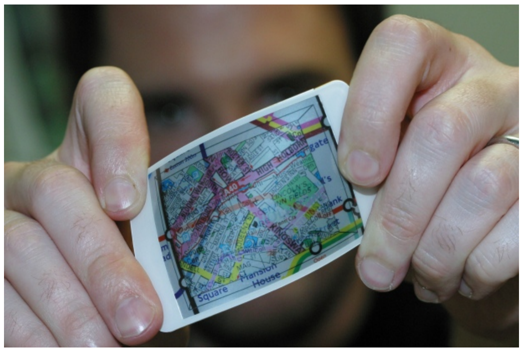


Figure 2.16: The Gummi interface.

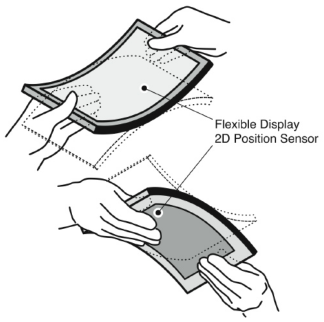


Figure 2.17: The Gummi interface scheme.

***Paradigm transition***

The transition between paradigms is not a complete one. In other words, once a new interface paradigm appears, it is not necessarily the end for the present paradigm. When graphical user interfaces appeared, command line interfaces did not seize to exist, they simply found a smaller *niche* where they were still useful (mainly for programming) [14] [5].

Graphical user interfaces have dominated the technology landscape since 1984. These interfaces provided something that was new to the users of computer systems: the ability to interact with the machine in a new way, other than simply typing commands. Because of this, graphical user interfaces were of great importance to the success of the desktop paradigm. Its users became so familiar with these interfaces that it is not just second nature anymore. It has become a part of the way people think about digital experiences [5].

But the graphical user interfaces paradigm has reached its' peak and is now losing dominance in some environments. A new paradigm is emerging: that of natural user interfaces [5].

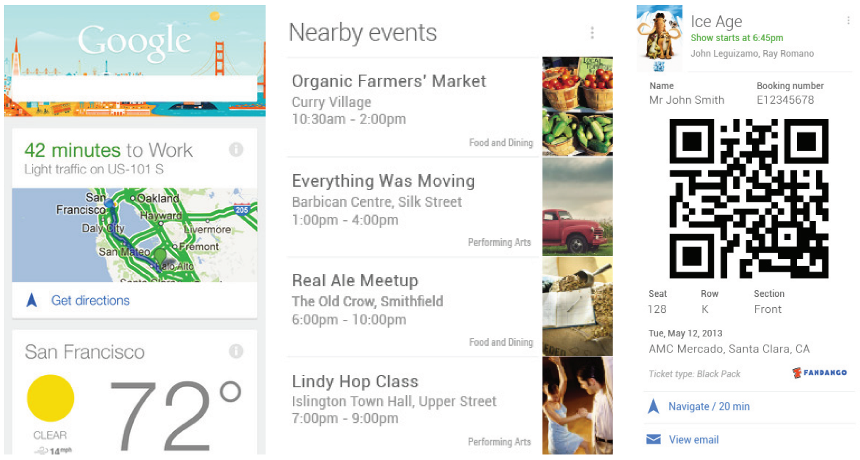
Natural user interfaces do not necessarily replace the existing way of interaction but allow computer systems to expand into new *niches* that could grow to be of tremendous size and importance [14]. As society becomes highly social and mobile, the desktop computer no longer serves its' needs [15].

This new interface paradigm is here to stay. However, it is yet unclear how much it will grow. These interfaces can either find a small *niche* in which to succeed (as did command line interfaces) or come to dominate the computer landscape. Either way, graphical user interfaces will likely coexist with natural user interfaces since they are well adapted to a very important *niche*: office work. [14].

### 2.1.3 User Experience (UX)

The concept of user experience (commonly referred to as UX) is related to the concept of user interface. User experience includes user interfaces but also transcends them since it includes other aspects such as the product's branding, the purchasing process, costumer support, etc [3]. While designing a good user experience is much more complex than designing a good user interface, the first still highly depends on the second. In fact, when observing the projects that were awarded with a UX Award[7], one notices that on the projects' descriptions, the quality of the interaction is constantly being referred to.

Google Now[8] won 2013's UX Awards Grand Prize receiving the title of "Best everyday utility". This application (figure 2.18) is described as "a proactive and contextually aware assistant for smartphones". The interaction between the user and this application is said to be very effective and very quick when compared to other applications of the same sort, which implies a good user interface within a great user experience.


Google

42 minutes to Work
Light traffic on US-101 S

Get directions

San Francisco
72°
CLEAR

Nearby events

Organic Farmers' Market
Curry Village
10:30am - 2:00pm
Food and Dining

Everything Was Moving
Barbican Centre, Silk Street
1:00pm - 4:00pm
Performing Arts

Real Ale Meetup
The Old Crow, Smithfield
6:00pm - 10:00pm
Food and Dining

Lindy Hop Class
Islington Town Hall, Upper Street
7:00pm - 9:00pm
Performing Arts

Ice Age
Show starts at 6:45pm
John Leguizamo, Ray Romano

| Name | Booking number |
|---|---|
| Mr John Smith | E12345678 |

| Seat | Row | Section |
|---|---|---|
| 128 | K | Front |

Tue, May 12, 2013
AMC Mercado, Santa Clara, CA
Ticket type: Black Pack
FANDANGO

Navigate / 20 min

View email

Figure 2.18: Google Now.

Don Normal and Jakob Nielsen [16] state that for a user experience to be exemplar it must meet the users' exact needs. They also state that it is important to distinguish UX from UI: even though the user interface is an extremely important part of the design, a good user interface does not make a good user experience by itself. It is also important to distinguish between UX and usability. The term usability refers to an attribute of the user interface that guarantees that the system is easy to learn, efficient to use, pleasant to the user, etc. User experience depends on both these concepts.

[7]More information available online at `userexperienceawards.com`
[8]More information available online at `www.google.com/landing/now/`

## 2.2 Natural User Interfaces

Natural user interfaces are a new way for humans to interact with computers. Because they are new, they are still being explored and a common vision of NUI possibilities is not yet developed [17].

People are usually drawn to interfaces with which they can interact in the same manner they do with real life objects. Even though this kind of interfaces have been appearing in futuristic movies for a long time, only recently has technology been able to support these interfaces in real life. Software should augment human abilities helping users to surpass their weaknesses and improve their strengths [18].

Natural user interfaces are an opportunity to better mirror how human life works: skills are learned gradually and actions are a consequence of human needs and desires, and depend on the environment we are in [14].

Even though the concept of natural user interfaces has only recently appeared, some attempts to create interfaces that feel natural to its users have been made before, especially related to video gaming. An example of these was Atari's[9] arcade game: *Gran Trak 10*, that was released in 1974 and some of SEGA's[10] arcade racing game machines that appeared in the 1980's (figure 2.19) [19]. These machines provided a more natural experience to the player while they raced virtually.
But the search for more natural user interfaces predates even the graphical user interfaces. In the 1950s, when the communication between computer systems and users was still achieved using a command line interface, some research was being made to attempt the creation of devices that could read handwritten characters. The interest for this research was to develop a way to reduce the cost of getting information into forms that computers can understand [20].

Before the term *natural user interfaces* was coined, some authors were already referring to these interfaces using different terms. An example are *perceptual user interfaces* that are the subject of an article by Matthew Turk and George Robertson [21], dated back to 2000. It states that perceptual user interfaces "seek to make the user interface more natural and compelling by taking advantage of the ways in which people naturally interact with each other and with the world".

The reason the concept of natural user interfaces has emerged and grown recently has to do with some new technologies that have become very popular such as touchscreens and motion-sensing controllers. These new technologies have deeply facilitated user interfaces to become

[9]More information available online at `www.atari.com`
[10]More information available online at `www.sega.com/Home`

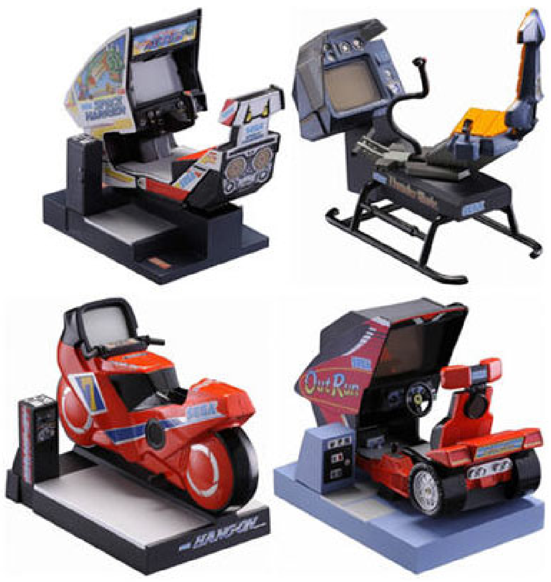


Figure 2.19: Sega's Arcade Games from the 1980s: Space Harrier, Super Hang-On, Out Run and Thunderblade.

natural.

In the next sections some definitions for the term *natural user interface* are discussed according to different authors. Also, a collection of principles and guidelines that should be followed when developing a NUI is presented as well as some of the technologies that facilitate natural user interfaces' existence. Finally some of the challenges that developers and designers of natural user interfaces must address are discussed.

### 2.2.1 Definition

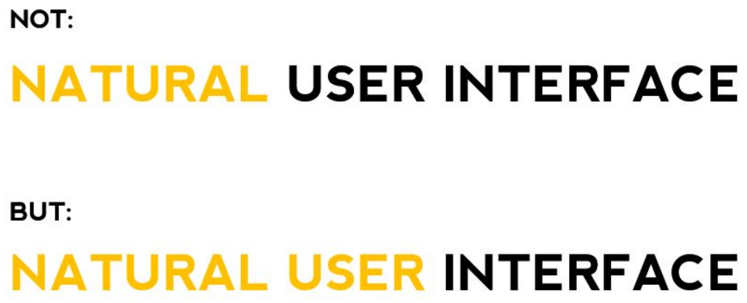
The term *Natural User Interface* was introduced for the first time by Steve Mann in 2001 [22]. It was defined as "the use of wearable computing or of physical matter (solid, liquids and gases) as direct user interfaces for metaphor-free computing". Mann considered human beings as cyborgs in the sense that the way we experience nature is not direct, but rather through objects such as shoes, clothing, smartphones, etc. He expected natural user interfaces to challenge this wall that comes between users and nature.

Since then, the term has evolved and while there is yet to be a complete consensus on a single definition for the term, experts in the field seem to have similar views.

In 2011, a group of experts incited a discussion to stimulate the exchange of knowledge about the subject. In their paper [17] they describe natural user interfaces as those that allow their users to interact with computer systems in the same way they interact with objects in the real world. They advocate that these interfaces are based in combinations of inputs and outputs that are perceived as natural by the users. This includes interfaces such as gesture, body language, proximity, position, audio and visual inputs, eye direction, expression, smell, object location and touch. For an experience to feel natural to the user it should be multi-modal since this is a characteristic of real world experiences. A multi-modal experience is one that uses a combination of inputs and outputs in which there is more than one input and / or more than one output.

In the book "A Brave NUI World" [14] a natural user interface is defined as "one that provides a clear and enjoyable path to unreflective expertise in its use". This definition's main focus is on the process of use of the interface. A natural user interface should make the learning process easy and enjoyable. It should also make sure that practice (for those who already mastered the skill) is still pleasant so that expert users continue to enjoy the process of using the interface.

In this definition, the meaning of the term *natural* in *natural user interfaces* is the one captured in the expression "that person is a natural". It means that a person's performance of a certain task is graceful and effortless. The natural property is not referring to the interface but rather to the way users interact with it and to what they feel while they use it (figure 2.20). A natural user interface should make the user act and feel like a natural while performing a task.



NOT:

NATURAL USER INTERFACE

BUT:

NATURAL USER INTERFACE

Figure 2.20: The natural property is not referring to the interface but rather to the way users interact with it and to what they feel while they use it.

The authors also claim that for a user interface to be a natural user interface it must contain three qualities:

- Enjoyable: The performance of a task by a user must always be enjoyed whether they are performing it for the first time or for the thousandth time.
- Leading to skilled practice: The interface must allow a novice user to evolve and become an expert rapidly.
- Appropriate to context: The interface must be appropriate to the context where it is inserted

and act accordingly with it.

In his book "Natural User Interfaces in .NET" [23], Joshua Blake defines the natural user interface as "a user interface designed to reuse existing skills for interacting appropriately with content". This definition tells us three things about natural user interfaces:

- Natural user interfaces are designed: These interfaces require forethought and deep planning prior to their development. The designer must take precautions to ensure that all the interactions are appropriate to the user, the task in hand, the device where the interface is built and to the context where the device is inserted.
- Natural user interfaces reuse existing skills: This refers to the word "natural". Users are experts in many everyday skills that they have come to master simply by living. By reusing these skills the developer is helping the user to rapidly understand how to use the interface.
- Natural user interfaces have appropriate interaction with content: The focus of the application should be on the content and not on the controllers. The interface should allow for the interaction between the user and the system to be as appropriate as possible according to the context and the situation.

The NUI Group focus on furthering NUI research. In their website [11] they also present a definition for these interfaces. They advocate that a natural user interface is a "computer interaction methodology which focus on human abilities such as expression, perception and recall". These interfaces should tackle the power of a wide variety of communication modalities that are based on skills that people have learned through interaction with the world.

### 2.2.2 Principles and Guidelines

Since NUI is still a relatively new concept a well-established set of principles that should be followed when designing an interface of this kind does not yet exist. Still, some experts have shared their thoughts on what they feel should be considered for a good natural user interface to be created. Some of these principles and guidelines differ slightly, but most of them seem to point in the same directions. This subsection present the main principles and guidelines that have been stated in books and articles on the subject and discusses where these different opinions intersect and what the main conclusions to be drawn are.

[11] More information available online at `wiki.nuigroup.com/Frequently_asked_questions`

***Design Ethos of NUI (by Wigdor and Wixon) [14]*** Wigdor and Wixon define this set of rules that should be followed to build a natural user interface.

- Less is more: Natural user interfaces should allow the user to rapidly evolve from novice to a skilled practitioner. Also, the process of developing skills and the interaction itself should be fun and enjoyable for the user. The system should introduce new challenges in a gradual way so that users can progress easily.
- Contextual Environments: A natural user interface should consider what actions are elicited in the environment where the interface is installed. Another aspect that should be deeply considered is content. The main focus of the users is content visualization and manipulation. Therefore, the main focus of the interface should be on the content and not on interaction objects that have the sole function of allowing users to interact.
- The Spatial NUI: Most natural user interfaces go beyond simple plane views and provide depth, make content appear to have volume or have 3D behaviors. This *design ethos* does not state that one should always use 3D environments when developing a natural user interface, but it does state that one should always consider the z-axis.
- The Social NUI: NUI experiences should contribute to the involvement of several users. Natural user interfaces should be designed for multi-person input. The experience should be more efficient and fun if multiple users are working simultaneously, but should also work with fewer users, so that a person is able to leave without disrupting the others' experience. The experience is not limited to the interaction that occurs between the interface and the user, but also to the interaction between users. The less communication between the user and the interface the better.
- Seamlessness: Natural user interfaces should create experiences in which users are so immersed (cognitively and emotionally) that they embrace these experiences and quickly learn to use the interface. The interface should respond immediately to every contact that the user makes and every transition should feel fluid and natural to the user. The best way to accomplish this is to mimic the real world in the transitions. An interface where things do not feel continued or where content appears and disappears out of nowhere break the user's sense of connection to the objects and stops feeling natural.
- Super Real: NUI experiences can be more fluid and natural if the designer mimics real-world physical interactions and augments them. Super realism allows user interaction to go further than what is physically possible in the real world. The interactions should be extensions of the real world and should make the user feel surreal and grounded at the same time.
- Scaffolding: The design of the interface should promote autonomous learning. The actions should encourage users to develop their own skills (cognitive, affective and psychomotor). The user should be able to easily understand how to interact with the interface while enjoying

the experience. To achieve this goal, a natural user interface's designer should use scaffolding. Scaffolding breaks bigger challenges into smaller problems. These smaller problems should then be addressed through hints, prompts and questions.

- User Differentiation: Users are not separated from reality, they are part of a context. Therefore, to design a natural user interface one must take into account the following elements: the users, their contexts, theirs responsibilities and their goals. The interface should act according to its user.

***NUI Principles (by Rachel Hinman) [5]***

Rachel Hinman advocates eight principles that every NUI should follow.

- Performance Aesthetics: NUI experiences should focus on generating a satisfaction feeling on the user. The user should feel a "joy of doing" while they interact with the system.
- Direct Manipulation: Users should be able to interact directly with the content. This principle is facilitated by using touchscreens or motion-sensing controllers that allow the user to feel like they are physically touching and manipulating the content with their fingertips.
- Scaffolding: Natural user interfaces should be intuitive and easy to use. The objects should behave in the way users expect them to behave. A NUI can contain very little options but should present clues and guides that hint the user on how the interaction will occur.
- Contextual Environments: A NUI should be dynamic and located in time and space. It should be responsive to the environment it is inserted in and suggest the next interaction to the user, based on that environment.
- Super Real: With natural user interfaces objects should be extended in a logical way that makes them appear to be surreal. Through gestures like pitch (figure 2.21), that allows the user to zoom in and out, a NUI can not only seem real, but rather super real, since the content can be modified in a way that extends the real world.

Figure 2.21: Gestures such as pinch are a logical extension of the real world.

- Social Interaction: Natural user interfaces should be simple to use and require little cognitive investment from its users. They should create opportunities for users to interact with one another instead of only interacting with the system.
- Spatial Relationships: In natural user interfaces the content is represented as objects. These objects are intelligent and have auras that help the users understand them.
- Seamlessness: The use of a natural user interface should feel seamlessness to its users because the interaction between the user and the content is direct. There are no intermediaries (such as a mouse or a keyboard); the user manipulates the content directly with their fingers.

***Guidelines for NUI developers (by Joshua Blake) [23]***

Joshua Blake put together a set of four guidelines that he believes that any NUI designer should have in mind.

- Instant expertise: The hardest part of using skills is learning them. When the user has previous knowledge about a skill it usually takes them little time to master it. This guideline states that when designing natural user interfaces the designer should reuse existing skills. By doing this, the users will not need to learn something new, they simply need to apply the skill they already know to the new situation. By reusing existing skills the designer is allowing the user to master the interaction very quickly.
- Cognitive load: This guideline states that when designing a natural user interface the developer should make sure that the most common interactions in the application use innate abilities of the user and that the skills users will need to learn will be easy to comprehend and apply.
- Progressive learning: The learning path the user must go through should be as smooth as possible. The user should be able to move from basic tasks to advanced tasks continuously, without finding major obstacles. The advanced tasks should be broken down in smaller tasks so that the user can continue to use simple skills to perform them and does not need to learn new and complicated skills.
- Direct interaction: All the interactions should be direct and appropriate to the context where the user is inserted. This is how we interact with the real world so by applying this to an interface, the interaction will feel fluid and natural to the users and will also allow them to use several features of the application at the same time without feeling overwhelmed.

***Other Guidelines for NUI development***

In his article, "Natural Search User Interfaces" [24], Hearst defends that for a natural user interface to be usable it should have as few distractions as possible. Content should be the main focus and only objects that are strictly necessary should appear.

In "Design Considerations for a Natural User Interface" [25] Murphy advocates that the first step before designing a natural user interface is to analyze the application's purpose and determine what user experience is intended since nowadays, in most market segments, user experience is a critical factor for success.

The authors of "The Future of Natural User Interfaces" [17] state that the designers should aim for the interface to provide a seamless user experience in such a way that the technology goes unnoticed to the user.

***Comparison of principles and guidelines***

Even though principles and guidelines to obtain a good natural user interface are yet being explored and there is not a well-established set of them, these previously stated principles have much in common.

The main statements that appear to be common amongst authors are:

- The interface should consider its context.
- The interaction between the user and the machine should be direct (there should be no intermediaries).
- The experiences should mimic real world interactions.
- The interactions should be enjoyed by the user.
- The skills must be easy to learn and should use pre-existing knowledge.

The main characteristic that a natural user interface should have is the ability to make the user interact in such a natural and direct way that they forgets they are interacting with a computer system.

### 2.2.3 Technologies

When we think about natural user interfaces the most common input / output devices that come to mind are touchscreens, motion-sensing controllers and voice systems. Although these technologies enable the construction of a natural user interface they do not, by themselves, define or guarantee one [14] [26].

A project by Koert van Mensvoort presented an approach to design a natural user interface without resorting to these devices that are most commonly used. Instead, he designed a natural user interface with a mouse as the input device. The concept is based on the fact that in the real world objects have a kinectic behavior that informs us of their physical properties. The author advocates that by applying tiny displacements upon the cursors' movement one can evoke in the user tactile sensations such as stickiness, weight and touch [27].

Still, modern devices are changing the way we interact with technology and although they do not guarantee it, they certainly facilitate the creation of interfaces that are more natural for users to use [14].

***Input and output devices***

An input device is one that allows a user to enter information into a computer system. These devices convert the incoming data and instructions into a pattern of electric signals in binary code so they can be interpreted by the system. Output devices do the opposite. They allow the computer system to deliver information to its user by translating the signals into a language that is understood by the users [28] [29]. Some input / output devices are part of the computer system while others are optional and intent to improve the experience [29].

These devices usually serve a specific purpose: to insert / receive a specific type of data. As computer systems evolved so did the needs for new types of input and output devices. In the earliest computers three devices were especially common: the keyboard, the monitor and the printer. These were associated with the command line interface paradigm. The first computer mouse was introduced in 1968 by Douglas Engelbart and was of the most importance to the growth of graphical user interfaces [29] [8].

Not only did the number of different devices grow significantly since the earliest computers but they also evolved to deliver more features and increased performance [29]. Today there is an impressive number of input and output devices. Most of these devices are related to graphical user interfaces, such as the case of the mouse, keyboard, joystick, among others. However, there are also a number of interesting devices that can facilitate the creation of natural user interfaces and improve the user experience. The most commonly used devices to develop a natural user interface are microphones, touchscreens, cameras and motion-sensing controllers [30].

Touchscreens and motion-sensing control technologies are becoming increasingly common and are very adequate to the development of a natural user interface. These devices are the focus of the next sections.

***Touchscreens***

The history of touch devices begins in 1948 with a touch-sensitive music synthesizer (figure 2.22) built by Hugh Le Caine [31]. It was only in 1965 however, that E. A. Johnson built the world's first touchscreen. The technology that was used in its construction was similar to today's technology but this first touchscreen was not multi-touch, it allowed only one touch at a time. These devices were used in air traffic control (figure 2.23) until the 1990s [32] [31].

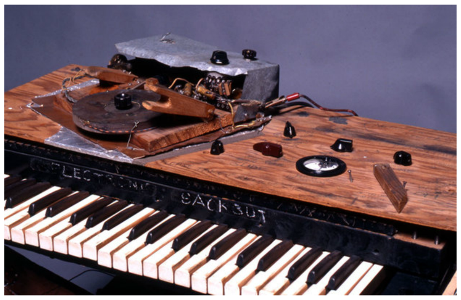


Figure 2.22: Touch-sensitive music synthesizer built by Hugh Le Caine.

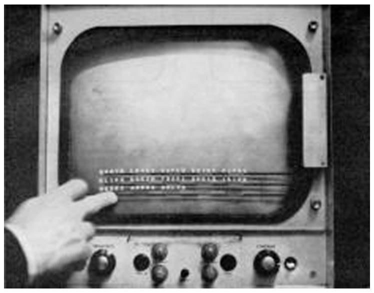


Figure 2.23: Air traffic control touch device.

Although they were created almost half a century ago, touchscreens evolved at a very slow rate and on the background. It was Apple that made the touchscreen widely popular by introducing the iPhone[12] in 2007 [5] [33]. Around the same time, Microsoft[13] also introduced a new product that made use of these technology: the surfaces[14].

Today touchscreens are widely spread in our everyday life. They are used in mobile phones, tablets, surfaces, laptop computers, monitors, televisions, ATM machines, GPS systems, game consoles among others. [33]. Most touchscreens today are multi-touch which means they allow for more than one input at a time [9]. This brings new possibilities of interaction and allows for the designers of natural user interfaces to follow a principle referred in section 2.2.2 that states that a natural user interface should explore social interactions.

***Motion-sensing controllers***

Motion-sensing controllers are devices that allow users to interact with a computer system through movement. These devices usually use sensors such as accelerometers to perceive the users' motion. The most common use for these devices are video games.

Although they have only recently become popular, the first attempt of using motion control in

[12]More information available online at `www.apple.com/iphone`
[13]More information available online at `www.microsoft.com`
[14]More information available online at `www.microsoft.com/surface/pt-pt`

games dates back to 1976, when Sega launched an arcade boxing game where the player had to physically move a boxing glove in order to punch. In 1986 an arcade motorbike racing game called *Hang-On* in which the user had to physically sit and move on a motorbike in order to control the virtual motorbike in the game was released. In 1993, the *Sega Activator* that could read the user's physical movements, was released for Mega Drive (figure 2.24). However this device was considered a failure since it was not very accurate.

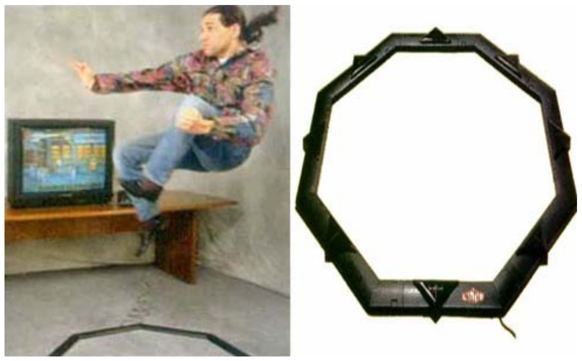


Figure 2.24: Sega Activator for Megadrive.

In 2006 the Nintendo *Wii*'s (figure 2.25) remote popularized the use of accelerometers as video games' controllers. It was followed by a similar device: the *PlayStation*'s Move[15] [34].

Figure 2.25: Nintendo *Wii*.

Accelerometers are not, however, the only mean for input in motion-sensing controller devices. Microsoft's *Kinect* uses a combination of infrared structured light and computer vision to perceive motion [34]. A more recent device, the *Leap Motion* controller, creates a 3D interaction space above the device and senses hands and fingers which allows the user to interact with the computer using precise motions as can be seen in figure 2.26 [35]. Leap Motion is further explained in section 2.3.3.

[15]us.playstation.com/ps3/playstation-move/

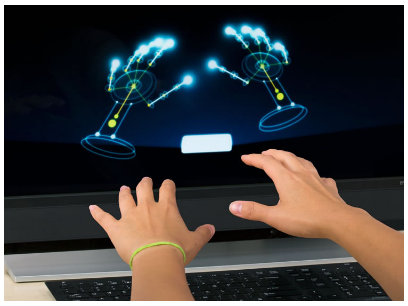


Figure 2.26: The Leap Motion device.

Even more recently a new device has appeared: the Myo[16] controller. Myo is a gesture control armband (figure 2.27) that uses sensors to detect muscle activity and motion in order to perceive commands.

Figure 2.27: Myo armband.

These devices are great facilitators of natural user interfaces since they allow the user to interact with a computer system using their body as one does in everyday life.

***Speech recognition***

Speech recognition systems allow its users to interact with a computer through voice commands. Their history starts in 1952 when the Bell Laboratories[17] designed the *Audrey* system which recognized only digits when spoken by a single voice. In the 1970s, speech recognition made some major strides with the *Harpy* system created in Carnegie Mellon[18]. Over the 1980s the vocabulary in speech recognition systems went from a few hundred words to several thousand. This leap was due to the use of new approaches such as the hidden Markov model[19]. It was during the 1990s that speech recognition became available for home usage. By 2001, there were speech recognition systems with an 80 percent accuracy rate [36] [37].

[16]More information available online at `www.thalmic.com/en/myo`
[17]More information available online at `www.alcatel-lucent.com/bell-labs`
[18]More information available online at `www.cmu.edu/index.shtml`
[19]More information available online at `www.cse.unsw.edu.au/~waleed/phd/html/node34.html`

*Google voice search*[20] and *Siri*[21] are two systems that allow the user to interact with their mobile devices through voice commands and that have revolutionized the way users interact with these devices [36] [37].

Studies have been made regarding the benefits of speech recognition for students with learning disabilities and for people with physical disabilities, poor or limited motor skills or vision impairments.

For students with learning disabilities speech recognition systems can encourage a more thoughtful and deliberate type of writing since it is easier for these students to "write" through dictation than it is to actually write words.

By removing the physical barriers imposed by graphical user interfaces such as the keyboard and the mouse, it is possible to increase the access to technology for people with disabilities and consequently increase their independence [38] [39] [40].

Despite its possibilities, speech recognition still needs some improvements in order to be a fully reliable method for interacting with computers [41].

***Technologies are a means, not an end***

These input devices will continue to evolve, but it is important to understand that no matter how further they evolve, they are merely a technology, a means to an end. They do not define the user interface by themselves. The conceptual model of the user interface is more important than the technology [26]. Touchscreens and motion-sensing controllers can add great value to a natural user interface, but a natural user interface involves a lot more than just a technology.

### 2.2.4 Challenges

When creating natural user interfaces many challenges arise. This subsection describes the most common issues that are found with natural user interfaces.

[20]More information available online at `www.google.com/insidesearch/features/voicesearch/index-chrome.html`

[21]More information available online at `www.apple.com/ios/siri`

***Creating a NUI does not occur naturally [14] [42]***

Developing a natural user interface is a goal that takes a lot of effort and thinking. These interfaces cannot simply mimic other experiences or rely on familiar metaphors. The developers of natural user interfaces have to forget their preconceived ideas of what a user interface should be. Since most of these developers have been living in a world dominated by graphical user interfaces for most (or all) of their lives, they tend to be stuck with GUI concepts. When developers rely on these concepts and approaches that worked for GUI and try to apply them to NUI, their interface is not really a natural user interface but rather a graphical user interface with touch or gesture.

Wigdor and Wixon state that to create a natural user interface takes "a clear viewpoint, a lot of hard work, careful design, rigorous testing and some luck". As was referred in section 2.2.2, natural user interfaces should allow the users to evolve rapidly and become experts in no time. To achieve this, the developer must elicit behaviors that are likely to be successful and subsequently create a trajectory of learning that leads to expertise. Wigdor and Wixon advocate that to accomplish this one should consider the original definition of affordance [43]: "an affordance is a property of whatever the person interacts with, but to be in the category of affordances it has to be a property that interacts with a property of an agent in such a way that an activity can be supported". This definition starts by stating that an **affordance is a property of the environment or context**. It goes on by stating that an **affordance elicits an action**. Finally it says that **the elicited action is supported by the environment**. Simplifying, the user is likely to do the "right thing" without training. The "right thing" refers to an action that is successful in the near term and that increases the likelihood that the next action will also be successful. The naturalness of a NUI begins with a symbiotic relationship between the environment and the user. This symbiosis is the starting point for designing the user interface. A system that has a natural user interface reacts in such a way as to show the user what their next step should be. For example, if a person tries to retrieve a liquid by cupping one's hand the lack of success will lead the person to try it with both hands. The way the liquid reacts shapes the way in which the person shapes their hands.

***The artificiality of natural user interfaces***

An issue that has been referred to by some authors is related to the artificiality of natural user interfaces. The statement that natural user interfaces are not natural, but rather artificial is based on the fact that the gestures that are used are not known by the user and must be learned.

Donald Norman [44] states that most gestures are neither natural nor easy to learn and remember. Even though some gestures are intrinsic in our everyday lives, they still highly depend on our culture so it would be impossible to use them to create a universal interface.

Belluci and Malizia [45] also refer to the artificiality of gestures. They state that the idea that these new interfaces are natural is due to the contrast between these and the graphical user interfaces. Using the mouse is not natural at all. The users must learn how to work with it and for someone who never interacted with a graphical user interface, it may not be easy to understand how it works. Consequently, natural user interfaces are more natural than the previous interfaces, since they can be simpler to learn, but that does not make them natural. The authors also claim that for natural user interfaces to be truly natural they should allow users to interact with them using the same gestures that they use to interact with everyday objects.

Even if natural user interfaces are not yet natural to most people, studies [46] [47] show that to the younger generations they usually are. Most children in developed countries use today's technologies such as touchscreens and motion-sensing controllers easily and often without having to be told how to use them (figure 2.28). An interesting and funny video[22] has filmed the reaction of small children to a command line interface and shows some interesting reactions when the children try to use it as they would use nowadays technology (touching the screen or trying to give the computer voice commands).

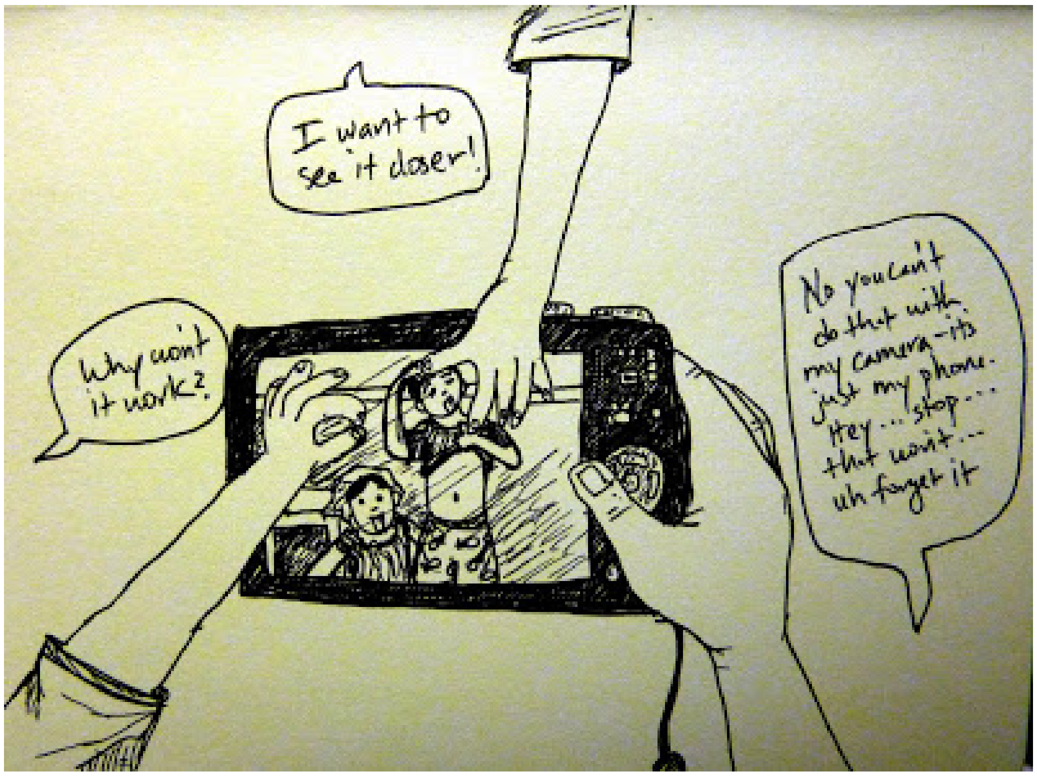

I want to see it closer!
Why won't it work?
No you can't do that with my camera - its just my phone. Hey... stop... that won't... uh forget it

Figure 2.28: Gestures such as pinch come naturally to younger generations. [23]

[22]More information available online at `goo.gl/j8IyCG`

***The Need for Standards [48]***
As natural user interfaces evolve the amount of gestures that are used to interact with them grows. This is aggravated when an interface uses multiple sensors at once. The user must remember a lot of different actions and when to execute them.

Another problem is that different companies, when developing their interfaces, create their own language of gestures. This results in a number of different interaction paradigms which causes great confusion to users when they need to use products from different brands.

These problems lead to a need for standards. It is necessary to establish a language of gestures that can be used freely, independently of the brand.

***Research [17] [49]***
Natural user interfaces concepts are not well defined. Researchers of this subject have presented different opinions about guidelines for natural user interfaces and the definition in itself but there is no consensus on the subject. For natural user interfaces to evolve some concepts and definitions need to be studied and established.

***Technologies [49]***
Natural user interfaces potential is also dependent of the evolution of technologies. As new technologies, such as touch and motion-sensing controllers become more ubiquitous and new sensors appear, the potential for natural user interfaces grows.

[23]source of the image: `www.bahhumpug.com/2011/08/generation-tech.html` available on January 16th of 2014

## 2.3 Games

Playing games is a common activity amongst humans and is one that has been around since the beginning of human life. Games are a big part of the human life: we play games for entertainment and fun and rely on them to pass the time. The emergence of computer systems rapidly brought the interest of visionaries that saw in it huge potential for games.

In the following sections a brief introduction to the history of games (especially video games) is made, the connection between video games and natural user interfaces is discussed, a set of games that will be the basis for this project is presented and the concept of procedural generation in games is addressed.

### 2.3.1 The history of games

There is no way of knowing when the first game appeared. Games have been a part of human life since early civilizations. The most common games that existed in these early civilizations were games of sports. Sports were usually involved with the preparation and training for war or hunting. The most common sport games involved throwing objects further than the other player and fighting [50]. Egyptian wall paintings (figure 2.29) show sport games of throwing, catching, running, jumping and fighting. The first Olympic Games date back to 776 BC and were introduced in Ancient Greece. The Olympic Games included sports such as races, wrestling or disc throwing [50] [51].

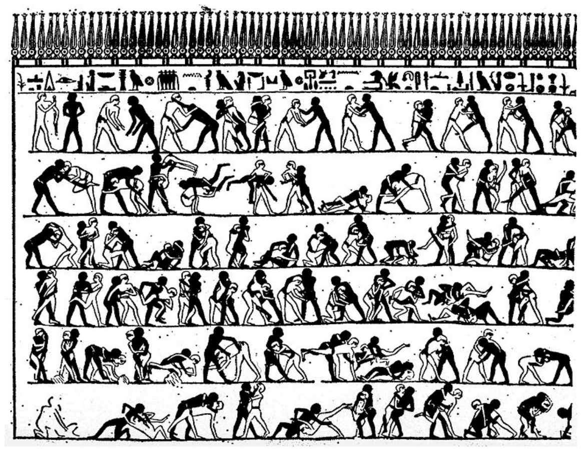


Figure 2.29: Egyptian representation of sports.

Board games were also quite common in early civilizations. One of the first documented games, Senet, dates back to 3000 BC and was popular in Egypt [52] [51].

In Mesopotamia, 2500 years BC, Backgammon (figure 2.30) was a usual game to play. This game remains very popular today, over 4500 years later [51].

Figure 2.30: Old backgammon game recovered from the ship Vasa, sunk in 1628.

As civilizations evolved, board games evolved with them and in the 6th century in India, a game appeared that would come to be a tremendous success: chess (figure 2.31) [52] [51].

Figure 2.31: Chess set from the 18th century.

Today, games continue to be a huge part of human life. Playing games is a way of entertainment and amusement. Games are present in our everyday life: from placing a bet with a friend, to watching a sports game or playing a game on our smartphones while we wait for something. New games appear every day and will likely always be a part of human life.

### 2.3.2 Video games origins

The industry of video games did not appear suddenly. It took some inspiration, technological advances and a lot of experimentation.

***Pinball [19]***

The history of video games starts with the pinball game. The beginnings of pinball can be traced back to the game bagatelle. This game consisted in the players using a cue to shoot balls up a sloped table. There are no records of how or when the cue sticks in bagatelle were replaced by a "plunger" device and the game started being called pinball except that it happened with the turn of the 19th to the 20th century.

A game that helped pave the way for today's video game industry was *Baffle Ball*, a game created by David Gottlieb in 1931. *Baffle Ball* did not use electricity and was of little resemblance to modern pinball. This was the game that made pinball popular. The *Baffle Ball* game was rapidly imitated and in 1933 Harry Williams built *Contact*, the first electric pinball machine.

***Novelty Games [19]***

Back in those days, these coin-operated amusement machines were commonly known as novelty games. By the 1940's, companies had already invented mechanical baseball games; games that tested the player's strength and games that simulated horse racing, hunting and Western gunfights. One of the most popular themes for these games were shooting games. In time, novelty games become quite sophisticated. By the 1960's, black lights were built into cabinets to make objects glow. These games were the direct ancestors of video games.

***The early computers [19] [53]***

In the 1940's, most computers were large enough to fill entire rooms. The biggest transformation in computer systems that prompted the gaming world was the possibility to display information through monitors. The technology that allowed for this transformation was the Cathode Ray Tube (figure 2.32).

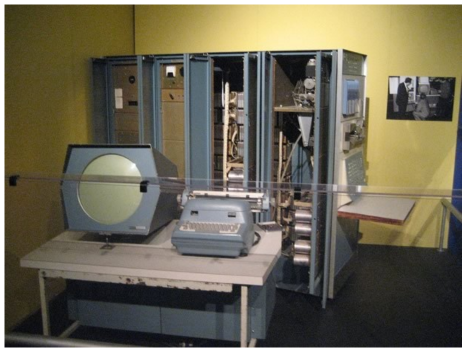


Figure 2.32: Cathode Ray Tube: CRT.

***The ancestors [53] [54]***

The *Cathode Ray Tube Amusement Device* was the first electronic game. This game was developed at the DuMont Laboratories (a television manufacturer). The authors of the project were Thomas Goldsmith Jr. and Estle Ray Man who, in 1947, devised a system for manipulating the electron beam by controlling a set of variable resistors to simulate a missile shooting game.

In 1952, Alexander Douglas made the first actual game programmed on a computer. This game was a tic-tac-toe game and was named *OXO*. The *OXO* game ran on the Electronic Delay Storage Automatic Calculator, the first machine programmable by assembly instructions and not by turning switches and connecting cables as happened with earlier computers. However, this machine was one-of-a-kind and therefore the game never has the chance to be played outside of Cambridge University.

William Higinbotham, in 1958, designed the first two-player sport-inspired video game: *Tennis for Two* (figure 2.33). The game could be played by using two controllers featuring a button that allowed the user to change the ball direction and a knob to affect the rebounding angle. This game, like *OXO*, was restricted to the laboratory in which it was created which prevented it from becoming known by the outside world.

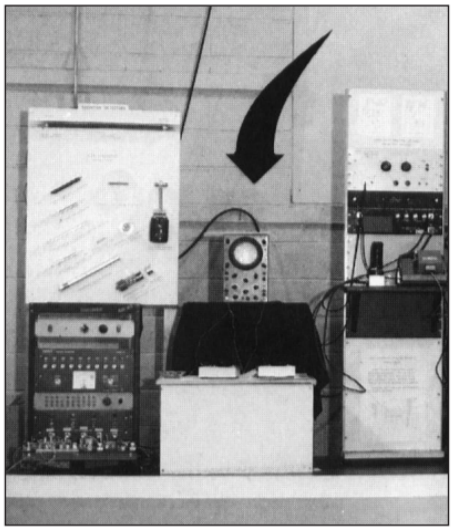


Figure 2.33: *Tennis for Two.*

### 2.3.3 Video games industry

The video games industry evolved quite rapidly. Soon, games stopped being solely developed for the existing computer systems and instead, new machines were being built with the single purpose of running games. This section presents some of the most important moments in the video game industry.

***Spacewar***

The first game that managed to attract people's attention outside a laboratory was *Spacewar* (figure 2.34). This video game was created in 1962 by Stephen Russel for the most recent computer at the time: the Programmed Data Processor 1: PDP-1 (figure 2.35).

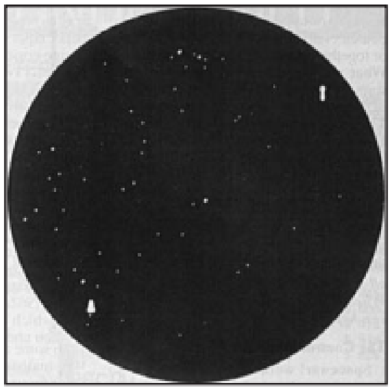


Figure 2.34: *Spacewar* screen capture.

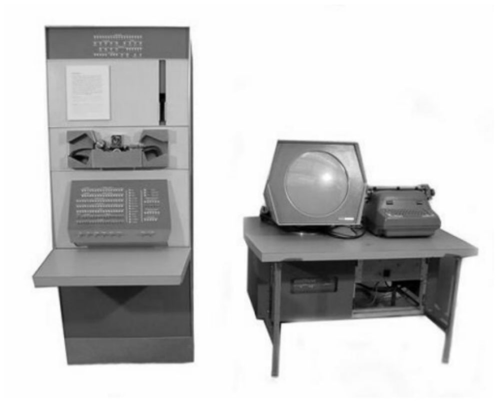


Figure 2.35: The PDP-1.

*Spacewar* was a combat-style game between to players in which they had to face each other in a shootout while avoiding the gravity well of a star that was placed in the middle of the screen. The game had four switches that allowed the user to control the spaceship. *Spacewar* soon became so well known that PDP-1 computers started being shipped with a demo of the game [19] [53] [55].

***The first commercial game: Computer Space [19] [53]***

In 1969, Nolan Bushnell and Ted Dabney, two visionaries that shared a passion for games and saw the potential that video games had from a commercial perspective, decided to start turning their vision into reality.

Their goal was to use a microcomputer (the DEC PDP-8) to design a coin-operated system that allowed the user to play a few different games when they inserted a coin. The first prototype they developed was a single-player version of *Spacewar*, to which they called *Computer Space* (figure 2.36).

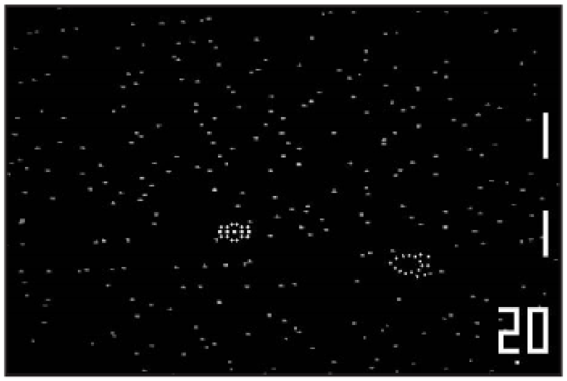


Figure 2.36: *Computer Space*, the first commercial coin-operated game.

*Computer Space* was the first commercial coin-operated video game. In this game, the player controlled a rocket ship and had to destroy a computer-controlled flying saucer. The game had a fixed length of 100 seconds and once the time was up, the player would get the results of how many times they successfully hit the saucer and whether or not they had scored more hits than the computer did.

Bushnell had worked in amusement parks during his youth, so he felt strongly that the game's cabinet was an important factor in attracting players. For this reason he designed a shiny and futuristic case in fiberglass (figure 2.37).

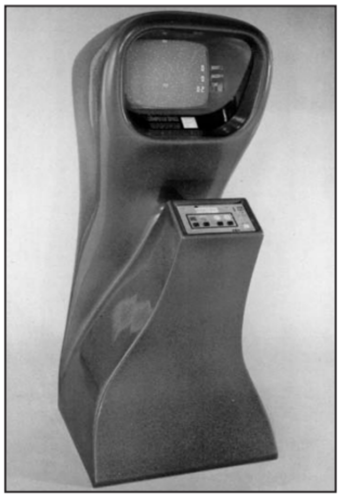


Figure 2.37: The *Computer Space* cabinet.

The machine was rapidly placed in pubs across the country but unfortunately the game did not fare as well as expected.

***The Brown Box [53]***

In 1967, Ralph Baer started working on a project to bring video games into people's homes. He built a prototype for a two-player ping-pong game and named it the *Brown Box* (figure 2.38). The prototype featured a light gun device that allowed the player to shoot white dots on the screen. Once the prototype was ready, a manufacturer was needed. The manufacturer that sealed the deal was Magnavox and by 1972 the product was being sold as the *Odyssey Home Entertainment System* (figure 2.39).

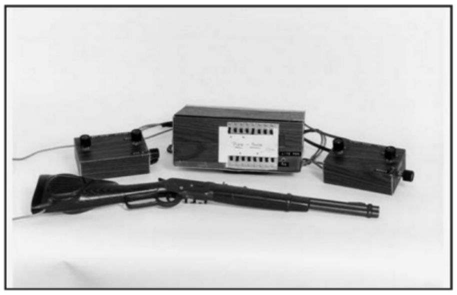


Figure 2.38: The *Brown Box* prototype.

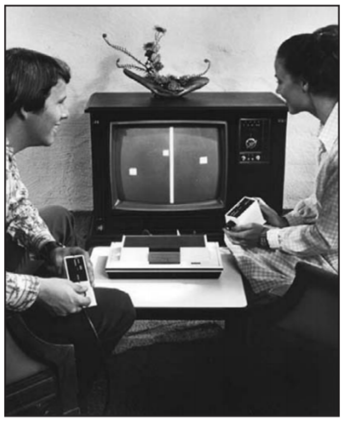


Figure 2.39: Playing table tennis on the *Odyssey*.

The machine came with a set of cartridges that when inserted altered the behavior of the game. Beside the default game (table tennis), there were also other games that could be played by inserting specific cartridges.

The *Odyssey* was a truly revolutionary system that successfully brought several board and table games onto the TV screen of its users.

***Atari and Pong [53]***

Despite the debut of *Computer Space* not being as spectacular as expected, Nolan Bushnell and Ted Dabney did not get discouraged. They had some ideas up their sleeves and so they decided it was time to create their own company. This was the beginning of Atari.

Since the lack of success of *Computer Space* was mainly due to the fact that it was too complex for the average person, they decided that their next game would be a simple sport-based game. The chosen game was ping-pong and the prototype for this game was named *Pong* (figures 2.40 and figure 2.41) and was released in 1972.

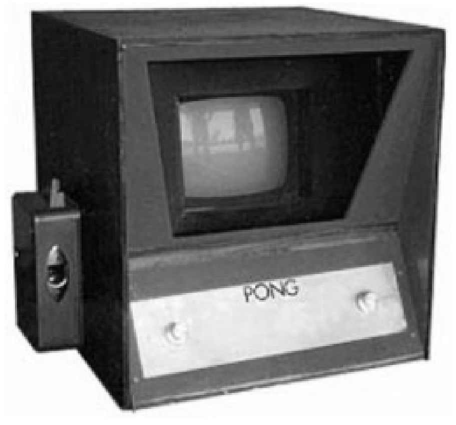


Figure 2.40: The *Pong* machine.

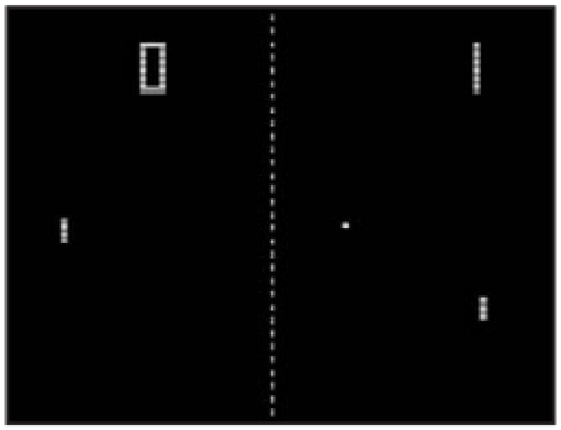


Figure 2.41: *Pong* screen capture.

The prototype was installed in a bar for testing. Two weeks after the installation, the owner of the bar called Atari because the game had stopped working. Once they arrived to fix it they realized with great surprise that the failure was due to coins overflowing out of the coin box. The game was so popular that customers were lining up to play it!

Soon imitators started to appear. Instead of a legal fight, Atari decided to fight in a different way: by creating its own *Pong* clones. At first Atari produced new games that were very similar to *Pong*, but soon they started to produce different games such as *Gotcha* (figure 2.42) and *Gran Trak 10* (figure 2.43).

In 1974, Atari decided that it was time to move in a different direction. They began building a prototype for a home console to bring their most famous game (*Pong*) into people's homes. The product was called *Home Pong* (figure 2.44) and in the Christmas season of 1975, 150 000 units of these product were sold.

Figure 2.42: *Gotcha*.

Figure 2.43: *Gran Trak 10*.

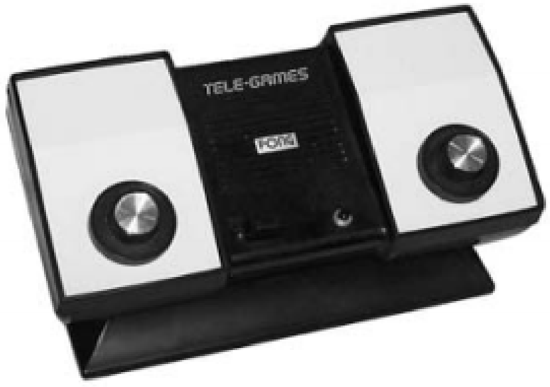


Figure 2.44: *Home Pong*.

***Space Invaders [56] [53]***

As soon as the first video games from Atari arrived at Japan, different Japanese companies started developing their own products. In 1978, Tomohiro Nishikado designed a game that was released by Taito, a Japanese company. The name of the game was *Space Invaders* (figure 2.45) and the player was in charge of defending a planet from slowly descending aliens. What differentiated this game from the other video games of that time was that it was a ferocious human-versus-machine battle. It was more exhilarating, stressful and adrenaline-pumping to play than any other game so far. This game was an instant hit and was a landing mark in the popularization of video games.

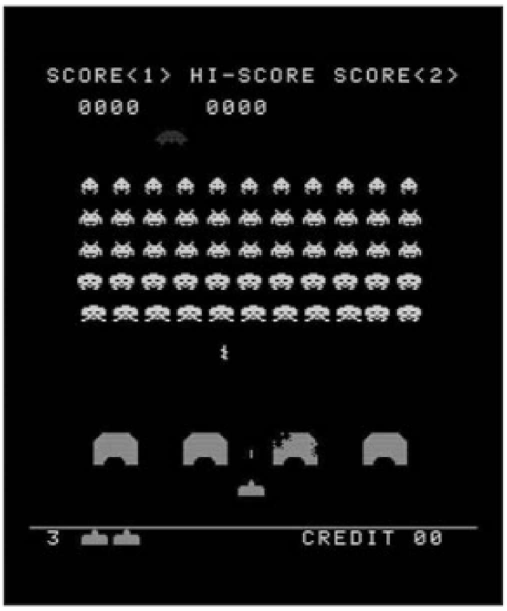


Figure 2.45: The original *Space Invaders* game.

***Pac-man [56] [53]***

In 1980 another hit was released: *Pac-Man* (figure 2.46 and 2.47). Originally named *Puck-Man*, this game, designed by Toru Iwatani, differed a lot from the existing games. It had colorful graphics and the attention to the detail was extreme. It also had short cut scenes used to advance the plot in the game and introduce the characters. At the time, most video games were violent and focused on male players. This game appealed to everyone regardless of gender, and that is what turned it into a worldwide phenomenon. The phenomenon was not only restricted to the game on itself but

A labyrinth of fun & amusement!
PAC-MAN
MIDWAY'S AMAZING NEW 1 or 2 PLAYER FULL COLOR VIDEO GAME!
MIDWAY

Figure 2.46: *Pac-Man* flyer.

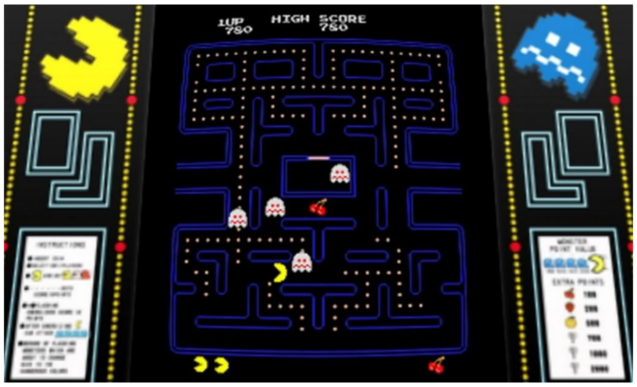


Figure 2.47: *Pac-Man* screen capture.

also to all the merchandising that was created over its design. Even a song, *Pac-Man Fever*, was successfully released and climbed up the American charts.

***Nintendo [53]***

Nintendo entered the video game industry in 1981 with its game *Donkey Kong* designed by Shigeru Miyamoto. *Donkey Kong* (figure 2.48) was the first platform game featuring mechanic which started a completely new genre. It was also the first game that told a simple yet entertaining story through short cut scenes that immersed players into the game.

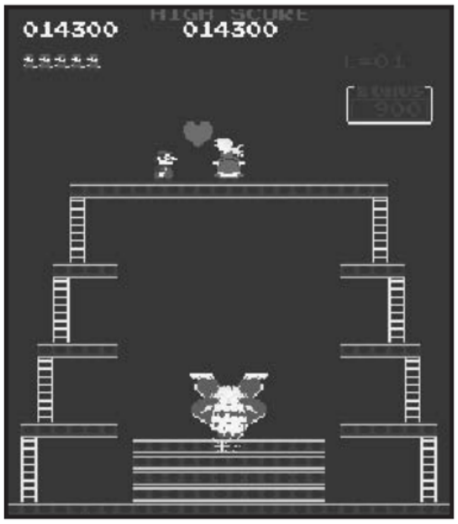


Figure 2.48: *Donkey Kong* screen capture.

In 1983 Nintendo released its first console: the *Famicom* (figure 2.49). The name was short for "Family Computer" and was launch with the *Donkey Kong* game and *Popeye*.

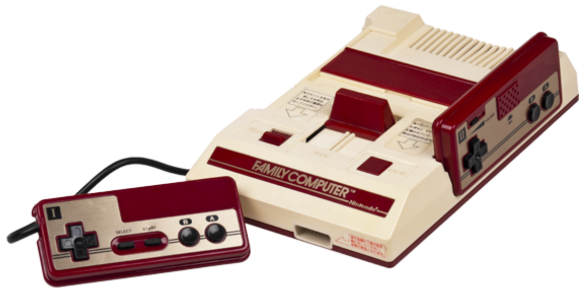


Figure 2.49: Nintendo's *Famicom*.

Even though it was released in Japan in 1983, it took two years for it to start being sold in the United States of America. The name of the console changed for this overseas release and it started being called *Nintendo Entertainment System* and it was marketed to be much more than just a new gaming console. The NES featured a light gun and a small robot that could perform simple tasks like running and carrying small objects. Besides developing a strong marketing campaign Nintendo was also concerned about the quality of its games and soon *Super Mario Bros.*

(figure 2.50) was launched. The main character of this game, Mario, also became iconic, such as *Pac-Man* had a few years before. *Super Mario* represented a new kind of video games not only due to better graphics and sound but mainly because it featured a much more complex adventures set than any game so far.

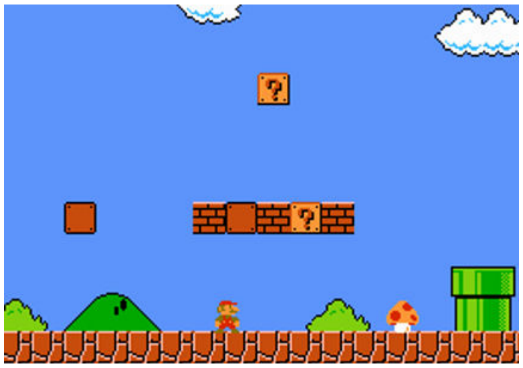


Figure 2.50: *Super Mario Bros.* screen capture.

***Tetris [56]***

It was Alexey Pajitnov who invented *Tetris* (figure 2.51) in 1984. The game was based on a puzzle called *Pentomines*. The computers available to him at the time were not state-of-the-art. His work computer was an Elektronika 60 an old computer from 1970. It had no graphics, so Pajitnov had to construct his digital pieces using punctuation marks.

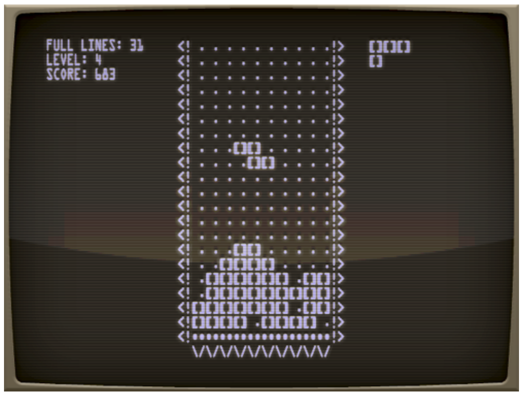


Figure 2.51: The original *Tetris* game was built using punctuation marks.

This game was highly addictive and quickly spread across Moscow and the rest of the world. Innumerous kinds of little portable machines that imitated the game were created. These machines allowed the users to play (a variant of) *Tetris* (figure 2.52) anywhere and millions of people used it for entertainment.

Figure 2.52: An example of a portable machine with the *Tetris* game.

***Sega [53]***

Sega was founded in 1940 and provided coin-operated amusement machines such as jukeboxes. In the mid-sixties, the company started working on electro-mechanical games and then on video games in the seventies.

In 1991, Sega released a game that featured a character that was able to compete with Nintendo's most iconic character: Mario. The game was *Sonic the Hedgehog* and Sonic was able to run at high speed across the levels which added new excitement to platform action games.

***Sony's PlayStation [53]***

Sony's aim when designing *PlayStation* (figure 2.53) was to design and develop the most powerful gaming console ever with 3D graphics. What differentiated Sony from the other companies at the time was that it did not try to sell its console as a family friendly console, but rather decided to target tech-thirsty teenagers right away.

Figure 2.53: The first *PlayStation*.

The *PlayStation* was a turning point for the video game industry because it run a new kind of games. Games that were more realistic and that appealed to a specific and highly influential type of public: teenagers.

***Nintendo's Game Boy [56]***

The *Game Boy* was an invention of Gunpei Yokoi, a Nintendo engineer. Handheld consoles were not new when the *Game Boy* appeared. Atari had already designed *Lynx* (figure 2.54) and Sega had the *Game Gear* (figure 2.55). Both were portable consoles that allowed the user to play anywhere.

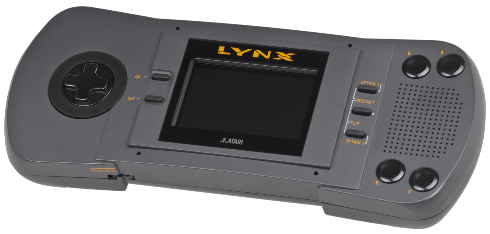


Figure 2.54: Atari's *Lynx*.

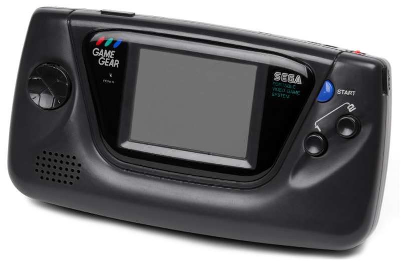


Figure 2.55: Sega's *Game Gear*.

But while its competitors were focused on engineering flashy consoles with color graphics and imposing sound capabilities, Nintendo opted for a console with a monochrome screen and a tiny speaker. While it could not compete in hi-tech, it was better than its rivals when it came to battery life and retail price.

At first, Nintendo intended to launch the *Game Boy* with a new spin-off of their popular game *Super Mario*. However, at the time *Tetris* was becoming very popular. In an attempt that the console would be popular not only amongst children and teenagers but also amongst adults, Nintendo decided to buy the rights of *Tetris*.

When the console was launched, in 1989, it was a huge success. More than 40 million *Game Boys* with copies of the *Tetris* game were sold worldwide.

***Nintendo's Wii [56] [57] [58]***

Nintendo launched its new console *Wii* (figure 2.56) in November 2006. The console suffered from its lack of high-definition graphics when compared to its direct rivals (Microsoft's *Xbox 360* and Sony's *PlayStation 3*). But this console also had something that its rivals had not: a more natural way of playing. *Wii*'s launch game was *Wii Sports*, a game where players could play virtual sports such as tennis, baseball, golf, boxing, etc., and the *Wiimote* controllers were perfectly adapted to this type of games.

Figure 2.56: *Wii*.

The console was a huge success and more importantly reached a new kind of public. People who had never displayed any interest in owning and playing video games were buying *Wii*'s. This success was mainly due to the fact that the controllers allowed for a much more natural type of gaming that players craved for and also for the type of games that this console supported that were mainly physical games. Another relevant aspect was the fact that *Wii*'s games were very adequate to play as a family, and since it had a lot of games that allowed up to four players, it ended up being a console for everyone in the family to play together. The name itself emphasizes that it is a console for everyone. When it was being announced Nintendo stated that the fact that *Wii* was pronounced "we" was not a coincidence but rather to emphasize that it intended to bring everyone together. In the marketing campaign it was also stressed that it was a console that would provide "family fun".

***Microsoft's Kinect [59]***

*Kinect* followed *Wii*'s lead on trying to create a more natural way to play, but in a different way. While with a *Wii* the player needs a controller to play the games, with *Kinect* the player's body is the controller. *Kinect* (figure 2.57) uses three depth-sensing cameras for image recognition and to determine where the player is in three-dimensional space.

Figure 2.57: The Microsoft's *Kinect*

This was a new kind of motion control, one where the user can play using nothing but themselves, which is a much more natural way to play.

***Leap Motion* [60][61][62][63][64]**

Like *Kinect*, *Leap Motion* also allows the users to use their body to control the computer. In this case the control is made only by using one's hands.

The Leap Motion device promises a unique way to control a computer using hands. It can even, more specifically, track single fingers to individual phalanges.

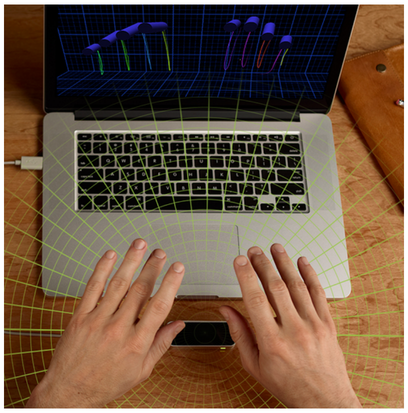


Figure 2.58: The *Leap Motion*.

*Leap Motion* is a small device and it aims for a different market than *Kinect*. While *Kinect* focus on applications that involve physical movement of the user's body, *Leap Motion* (figure 2.58) is dedicated to allow users to pursue their everyday activities in the computer in a more natural way. Until now, playing games in personal computers usually involved a mouse, a keyboard or a gaming devices such as the joystick or a gaming steering wheel. This device came to change that.

The device is composed of two hidden cameras that are placed under a protective sheet of dark plexiglass. From a technical point of view, this is a very similar implementation to *Microsoft's Kinect*. However, unlike *Kinect*, *Leap Motion* focuses only on the space above the hardware, since its sensors need to cover a smaller field. This translates into accuracy, and even smaller fingers are detected.

Inputs up to 40 cm around the device are recognized with reasonable accuracy, although it drops significantly around the edges of the field of view. This is one of the limitations of the device, specially when compared to *Kinect*, that has a much wider field.

Another limitation has to do with the fact that the device is not able to see through fingers (for example when a finger is covering another). When two or more fingers are very close to each other, they also might not be recognized individually but rather as an only finger. Even thought the device appears to try to interpolate missing data, the result is usually not successful. When using both hands, the device usually does not detect every finger in each hand and sometimes does not even perceive that both hands are in use.

Another problem that occurs frequently is erratic arrow movement, a gesture that allows the user to point to the screen and move a pointer around.

Overall, the critics [62] [63] [61] seem to agree that the *Leap Motion* device is still a little ahead of its time, but offering a look into the future of this kind of technology.

### 2.3.4 Games and NUI

The previous section made it quite obvious that the search for natural user interfaces has been a recurrent theme in gaming. Since the first arcade games that there has been a search for a more natural way to play.

Nintendo's *Wii* was a turning point on the subject [65]. Not only did it allow for a more natural way to play, but it also reinvented gaming making it appealing to a different kind of public [56] [57] [58].

With the appearance of this revolutionary device, the other players in the industry had to step up. Microsoft created *Kinect* which was even more natural to play than a *Wii* since the player does not need anything but his body to play. Sony created *Move* for *PlayStation* that is similar to the *Wii* since the player has to use a controller. Lately, other devices (not so focused on gaming) started to appear such as the *Leap Motion* and *Myo*.

The games that are created for these devices tend to be, as expected, focused on physical activity [65]. This fact makes the games appealing, not only to the usual video game public but also to a new type of public.

### 2.3.5 Procedural Generation

The concept of "procedural generation" refers to the programmatic generation of content using a random (or pseudo-random) algorithm [66] [67]. The main goal is that the data structures are created and then populated with data that comes from the code itself, rather than being loaded from files [68]. The type of content that is created this way depends on the purpose of the program.

Procedural generation of content is often used on video games because it adds value to them: each game turns into a different and unpredictable experience [69].

This technique also has the advantage of allowing games to have a big set of sceneries overcoming constraints of memory and storage [66].

Some examples of games that use a procedural generator are *Borderlands* (in the creation of weapons), *Just Cause* (in the creation of the game world) and *Civilization* (in the creation of world maps as can be seen in figure 2.59).

Figure 2.59: *Civilization* screen capture.

Some methods for procedural content generation are:

- **Random Number Generator [70]**: Allows the creation of a sequence of numbers and symbols without any pattern and that appear to be random. Generating a large set of random numbers takes a lot of time and work. These kind of algorithms usually use some physical phenomenon (that is expected to be random, such as atmospheric or thermal noise) and then uses that to generate the set of random numbers.

- **Pseudorandom Number Generator [71]**: An algorithm for creating a sequence of numbers whose properties approximate the ones of sequences of truly random numbers. The generated sequence of numbers is not truly random since it is determined by a relatively small set of initial numbers called the seed. Pseudomrandom number generators are quite fast in the generation of the set of output numbers and are often used in Procedural Content Generation.

- **Linear Congruential Generator [72]**: Is a Pseudorandom Number Generator that uses the following formula $X_{n+1} = (aX_n + c) \mod m$ where $X_0$ is the seed, and $a$, $c$ and $m$ are predefined numbers.

- **Fractal [73]**: The term fractal describes a broad set of shapes. It is a natural phenomenon that exhibits a repeating pattern. Fractals are often used in Procedural Content Generation because they seem to mimic natural processes such as erosion and plant growth.
- **L-systems [74]**: An L-system (or Lindenmayer System) consists of an alphabet of symbols. These systems can generate fractals and are commonly used to describe the behaviour of plant cells and to model the growth of plant development and the morphology of some organisms. It is usually used to generate all kinds of plants: from weeds to trees.
- **Perlin Noise [75]**: Perlin Noice is a fractal algorithm. The basic idea behind Perlin Noise is to add layers of noises.
- **Genetic Algorithm [76]**: A Genetic Algorithm mimics the process of natural selection and it is usually used to generate solutions for optimization and search problems.

# 3. System Description

This chapter provides an overview of the systems that were developed. The requirements are described in the form of user stories and use cases, the flow of the games are represented in a generic navigation diagram, the main screens are represented in a set of mockups and the system architecture is presented.

## 3.1 User Stories

The basic requirements for the games are described in this section using user stories. User stories are short descriptions of a feature of the product told from the perspective of the person that needs or wants that feature [77] [78]. User stories are usually created using the following structure:

**As a** <type of user>
**I want** <some goal>
**So that** <some objective>

In this case the user is always the same: the player. Six user stories, common to all games, were defined.

**User Story 1**: **As a** player **I want** to play the game **so that** I am amused.

**User Story 2**: **As a** player **I want** to check the credits of the game **so that** I learn about who developed the game.

**User Story 3**: **As a** player **I want** to check the game's instructions **so that** I learn how to play the game.

**User Story 4**: **As a** player **I want** to change the game's settings **so that** I set the volume the way I like.

**User Story 5**: **As a** player **I want** to change the game's settings **so that** I change my nickname.

**User Story 6**: **As a** player **I want** to check the high scores **so that** I can remember my achievements.

## 3.2 Use Cases

To define the general requirements of the games, use cases were created. The following table (3.1) presents these use cases, their goals, pre and post conditions, main flow of events and secondary flow of events (when necessary).

The use cases were built on the following assumptions:

- The game is installed on the player's computer.
- The player has a leap motion device active on the computer.
- The game is running.

Table 3.1: Use Cases

| Name | Goal | Pre-conditions | Post-conditions | Main Flow of Events | Secundary Flow of Events |
|---|---|---|---|---|---|
| 1. Visualize high scores | Allow the player to check their high scores. | There are none. | There are none. | The **player** indicates their intention to see the high scores.<br><br>The **game** presents the high scores.<br><br>The **player** checks the high scores. | If there are no high scores saved:<br><br>The **game** indicates that there are no high scores yet. |
| 2. Change sound settings | Allow the player to change their sound settings. | There are none. | The sound settings are saved. | The **player** indicates their intention to change the sound settings.<br><br>The **game** presents an item that allows the player to change the sound.<br><br>The **player** adapts the sound.<br><br>The **game** saves the changes. | |
| 3. Change player nickname | Allow the player to change its player nickname. | There are none. | The nickname is saved. | The **player** indicates their intention to change their nickname.<br><br>The **game** presents a text box that allows the player to change their nickname.<br><br>The **player** types in the new nickname.<br><br>The **game** saves the changes. | If any of the characters of the new nickname is invalid:<br><br>The **game** indicates to the user that some of the characters cannot be used and asks for a valid nickname.<br><br>The **player** inserts a valid nickname.<br><br>The **game** saves the new nickname if it is valid. Otherwise repeats. |
| 4. Visualize credits | Allow the players to see the game's credits. | There are none. | There are none. | The **player** indicates their intention to see the credits.<br><br>The **game** presents the credits.<br><br>The **player** sees the credits. | |
| 5. Visualize instructions | Allow the player to learn how to play the game. | There are none. | There are none. | The **player** indicates their intention to visualize the instructions.<br><br>The **game** presents the instructions.<br><br>The **player** sees the instructions. | |
| 6. Play | Allow the player to play the game. | There are none. | The game is playing. | The **player** indicates their intention to start playing.<br><br>The **game** starts.<br><br>The **player** uses gestures to play.<br><br>The **game** alters its state according to those gestures. | If the game is lost:<br><br>The **game** saves the player's score and resumes to the "game over" state. |

## 3.3 Navigation Diagram

In order to represent the flow between the different parts of the game, a navigation diagram was designed (figure 3.1). This diagram represents the possible flow of the games but was adjusted to the different game's needs as can be seen in chapter 5.

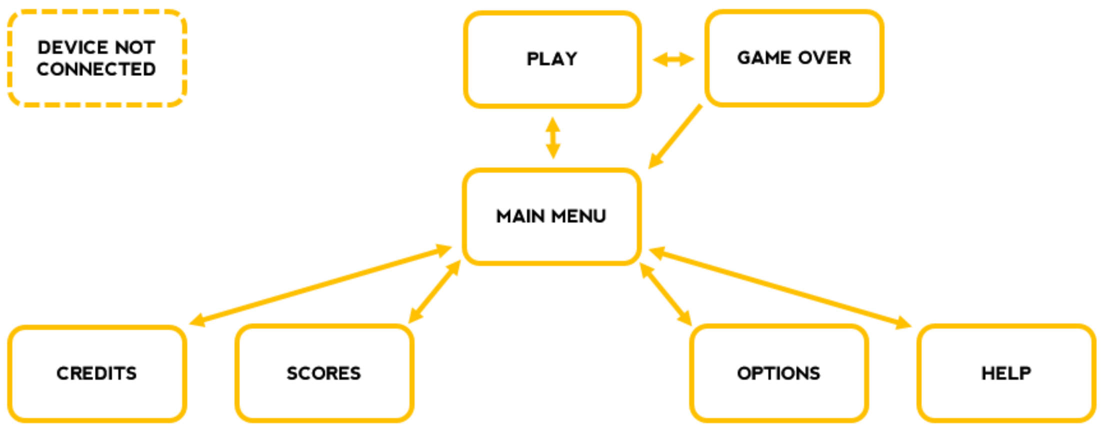


Figure 3.1: Generic navigation diagram of the games.

The **Main Menu** is the state that appears when the player runs the game. This state is the connection between all the other parts of the game mentioned in the use cases: the **Credits**, **High Scores**, **Options**, **Instructions** and **Game Play**. From the Main Menu the player can move back and forth between this different states of the game. When playing, the game might come to an end, which will lead to the **Game Over** state. There is yet another state in the game: the **Device not Connected** state. This state is accessible through any of the others and is activated when the *Leap Motion* device is disconnected.

## 3.4 Mockups

The mockups for each screen are presented in this section (figure 3.2). These mockups are generic and were later adapted to each particular game as can be seen in chapter 5.

The Main Menu should allow the player to either: start the game, see the game's instructions, check their scores, change the game's settings or see the credits.

The buttons that are represented in the mockups are not typical (GUI) buttons but rather objects that the player should point to for a short period of time, in order to chose the action.

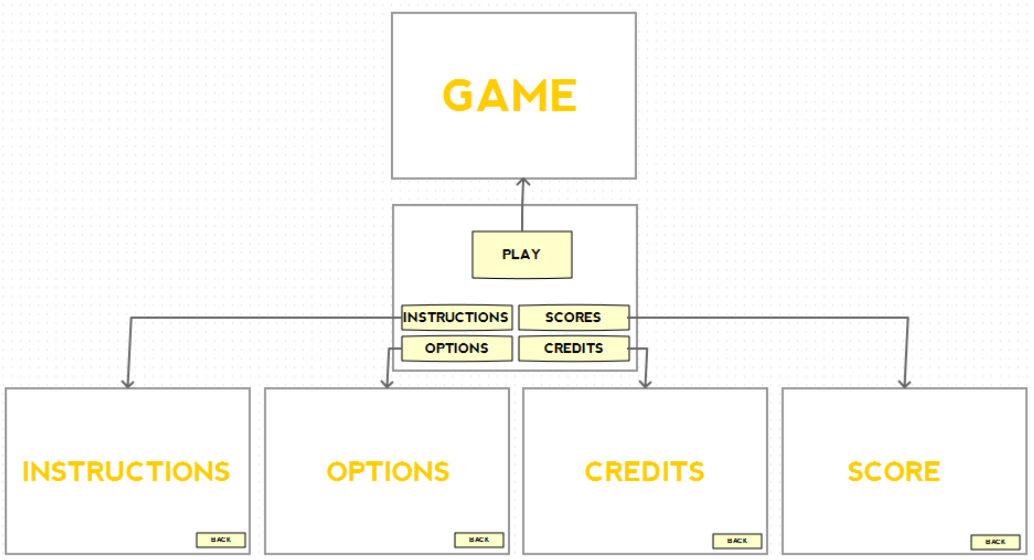


Figure 3.2: Generic mockups for the games.

## 3.5 System Architecture

This section describes the system architecture for this project through a components diagram.

The *Leap Motion* software runs on *Windows* as a service (or as a daemon on *Mac* and *Linux*). This service makes the connection between the operating system and the *Leap Motion* controller device over the USB bus. The *Leap Motion* applications must access the *Leap Motion* service to receive the motion tracking data.

The *Leap Motion* SDK provides two different APIs for getting motion tracking data: a native interface and a WebSocket interface. In this project the native interface was used since the WebSocket interface is meant for web applications. The *Leap Motion* service is also connected to the *Leap Motion* settings that allow the user to configure their *Leap Motion* installation.

The *Unity 3D* C# engine uses the *Leap Motion* native library to connect to the *Leap Motion* service and retrieve the motion tracking data. The games were built in the *Unity 3D* platform which is composed of three main components: graphics, audio and physics. It also uses the motion tracking data that comes from the *Leap Motion* service.

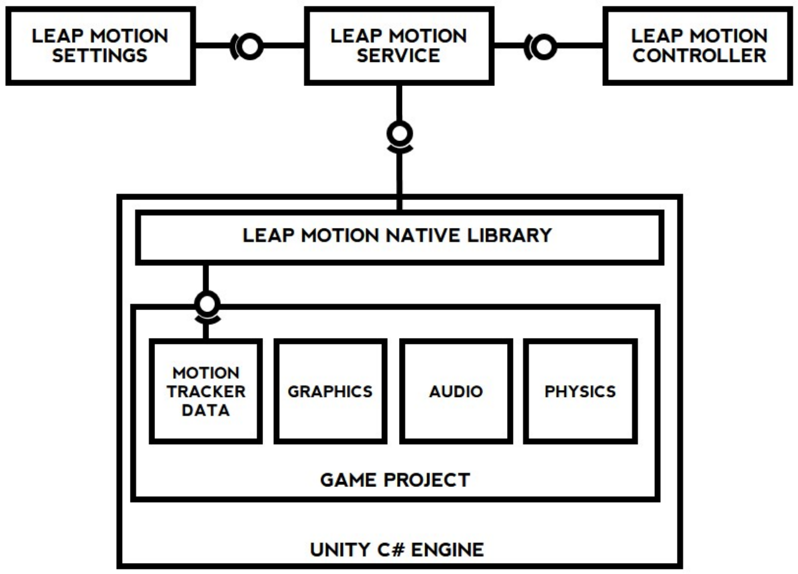


Figure 3.3: Components diagram that represents the system's architecture.

Figure 3.3 presents the components diagram for the previously described system.

# 4. Project Planning

This chapter describes the planning of the project. This plan includes the definition of the project's main goals and challenges, the definition of the tasks and subtasks, the technological choices that were made and the games that were chosen for implementation.

## 4.1 Goals

The main goal of this project was to study new possibilities for natural user interfaces. This was achieved through the development of two video games with natural user interfaces.

The developed video games were based on common computer games or traditional physical games and should reveal a new way to play them. The main idea was the reinvention of these games for a new type of interface: natural user interfaces.

The video games were developed for *Leap Motion*, a device that enables the creation of natural user interfaces since it allows users to interact with the computer through gestures (more information on the *Leap Motion* device in section 2.3.3). The project should explore the possibilities that come with this device and provide a great user experience for the player. It is important for this project that the games have a truly natural user interface.

A secondary goal for this project was for the two developed games to be viable for commercialization in the *Leap Motion* store (*Airspace*).

## 4.2 Challenges

This section presents the main challenges that were expected in this project. The main challenges were three: the definition of the gestures that were used in the games, to follow the guidelines of natural user interfaces in order to provide a truly natural experience and the use of procedural generation in the development of the games.

**Gesture Definition**
An important challenge in this project was to define what gestures to use in a way that complies the usual gestures in *Leap Motion* applications. It was important to keep the gestures as simple and intuitive as possible in order to provide the users with a natural user interface.

**NUI Guidelines**
To guarantee that the games provided a truly natural experience to its users, an effort was made to assure that all of the main guidelines (established in subsection 2.2.2) were followed. This was the most important of all challenges because if the games did not provide a natural interface the project would fail.

**Procedural Generation**
Another challenge in this project was to develop the games using Procedural Generation in order to keep the games different every time the user played them.

## 4.3 Tasks

This section describes all the tasks that were initially defined for this project and their planning throughout the schedule. It also explains a tool (*Trello*[1]) that was used for task control during the project.

### 4.3.1 Task definition and planning

In the beginning of the second semester, after the interim presentation, a reassessment to what had been planned was made since the planning appear to be overly optimistic. In order to make it more realistic it was decides to produce only two games, but have the second game be a little more complex and original and include aspects from more than one of the four games that had been acknowledged as interesting to reinvent in the first semester. These games and the games that were developed are further explained in the section 4.5.

Besides these alterations to the subtasks, another task was included: the placement of the games on the Airspace store. This task also depended of an external entity: DITS[2], a division of University

[1]More information available online at `trello.com`
[2]More information available online at `www.uc.pt/gats`

of Coimbra that deals with the products that are developed by the researchers of the University of Coimbra. The interaction had to be made since the games were developed by a researcher os the University. This interaction is further explained in subsection 5.4.2 of the Development chapter.

The scheduling became very tight with the changes and this, added to some delays due to problems with the *Leap Motion* API and the DITS division, quickly led to extending this project to the September deadline.

The project can be divided into eight main tasks:

1. Study of the State of the Art
2. Planning
3. System Description
4. Interim project delivery
5. Development
6. Usability Evaluation
7. Games Publishing
8. Final project delivery

The first four tasks were assigned to the first semester while the last four were meant to be accomplished in the second semester. These main eight tasks remained intact throughout the project. However, the due dates on the three final tasks was changed due to delays that appeared in the development of the games.

The following tables divide each task into a group of subtasks and figures 4.1 and 4.2 show the Gantt diagrams that were built for each semester. These tables and diagrams represent the planning that was made for this project. The following tables divide each task into a group of subtasks. Figure 4.3 shows the final Gantt diagram for the second semester.

***Study of the State of the Art***

This task involves researching about the subjects that will be addressed in the project. The main three subjects are: user interfaces, natural user interfaces and games.

| 1. State of the Art | Description | Begin Date | End Date |
|---|---|---|---|
| 1.1. User Interfaces | Research about the concept of user interfaces, their definition and different paradigms. | 16.09.2013 | 13.10.2013 |
| 1.2. Natural User Interfaces | Research about the concept of natural user interfaces, its definition, principles, guidelines, technologies and related work. | 16.09.2013 | 03.11.2013 |
| 1.3. Games | Research about the history of games, especially video games, and their relation with natural user interfaces. | 04.11.2013 | 01.12.2013 |

Table 4.1: Study of the State of the Art

***Planning***

This task involves all of the planning of the project. It was divided into four subtasks: goal definition, challenges that would be dealt with during the project, tasks definition, technological choices and game choices.

| 2. Planning | Description | Begin Date | End Date |
|---|---|---|---|
| 2.1. Goals definition | Definition of the project's main goals. | 30.09.2013 | 06.10.2013 |
| 2.2. Tasks definition | Definition of the project's main tasks and subtasks. | 07.10.2013 | 20.10.2013 |
| 2.3. Technological choices | Study of the different technological devices and frameworks available and decision of which ones to use. | 02.12.2013 | 15.12.2013 |
| 2.4. Game choices | Selection of a set of games to be developed in the project. | 06.01.2014 | 19.01.2014 |
| 2.5. Challenges | Definition of the challenges that would be dealt with during the project | 10.02.2014 | 17.02.2014 |

Table 4.2: Planning

***System Description***

This task involves the description of the system through user stories, use cases, navigation diagram, mockups and system architecture.

| 3. System Description | Description | Begin Date | End Date |
|---|---|---|---|
| 3.1. User stories | Creation of user stories for the requirements of the games. | 02.12.2013 | 08.12.2013 |
| 3.2. Use cases definition | Definition of use cases for the requirements of the games. | 02.12.2013 | 08.12.2013 |
| 3.3. Navigation Diagram | Creation of a navigation diagram to represent the flow between the different parts of the game. | 09.12.2013 | 15.12.2013 |
| 3.4. Mockups | Creation of mockups of the games. | 09.12.2013 | 15.12.2013 |
| 3.5. System Architecture | Definition of the system's architecture. | 20.01.2014 | 26.01.2014 |

Table 4.3: System Description

***Interim Delivery***

This task includes the elaboration of the interim report and the preparation of the interim presentation.

| **4. Interim Delivery** | **Description** | **Begin Date** | **End Date** |
|---|---|---|---|
| 4.1. Interim Report | Elaboration of the interim report. | 02.12.2013 | 26.01.2014 |
| 4.2. Interim Presentation | Preparation of the interim presentation. | 27.01.2014 | 02.02.2014 |

Table 4.4: Interim Delivery

***Development***

This task included the development of the games and was divided into three subtasks: the study of *Leap Motion* development and the development of the first and the second game.

| **5. Development** | **Description** | **Begin Date** | **End Date** |
|---|---|---|---|
| 5.1. Leap Motion development with Unity | Study of the development for *Leap Motion* with *Unity 3D* | 03.02.2014 | 02.03.2014 |
| 5.2. First game | Includes the development of the game engine, game menus, options, instructions, scores and credits. | 03.03.2014 | 13.04.2014 |
| 5.3. Second game | Includes the development of the game engine, game menus, options, instructions, scores and credits. | 14.04.2014 | 23.06.2014 |

Table 4.5: Development

***Evaluation and Improvements***

This task includes the evaluation of the developed games and improvements to the games based on the results of the evaluation.

| **6. Evaluation and Improvements** | **Description** | **Begin Date** | **End Date** |
|---|---|---|---|
| 6.1. Evaluation: First Game | Evaluation for the first game. | 14.04.2014 | 27.04.2014 |
| 6.2. Evaluation: Second Game | Evaluation for the second game. | 24.06.2014 | 13.07.2014 |
| 6.3. Improvements: First Game | Improvements to the game based on the problems detected during the evaluation. | 28.04.2014 | 11.05.2014 |
| 6.4. Improvements: Second Game | Improvements to the game based on the problems detected during the evaluation. | 14.07.2014 | 03.08.2014 |

Table 4.6: Evaluation and Improvements

***Games Publishing***

This task was divided into two subtasks that had to be made in order to publish the games: pass the Airspace tests and reach an agreement with DITS on how the games would be published.

| **7. Games Publishing** | **Description** | **Begin Date** | **End Date** |
|---|---|---|---|
| 7.1. Leap Motion and the Airspace | Have the games accepted by the store | 23.06.2014 | 30.06.2014 |
| 7.2. DITS | Reach an agreement on how to publish the games | 24.02.2014 | 30.06.2014 |

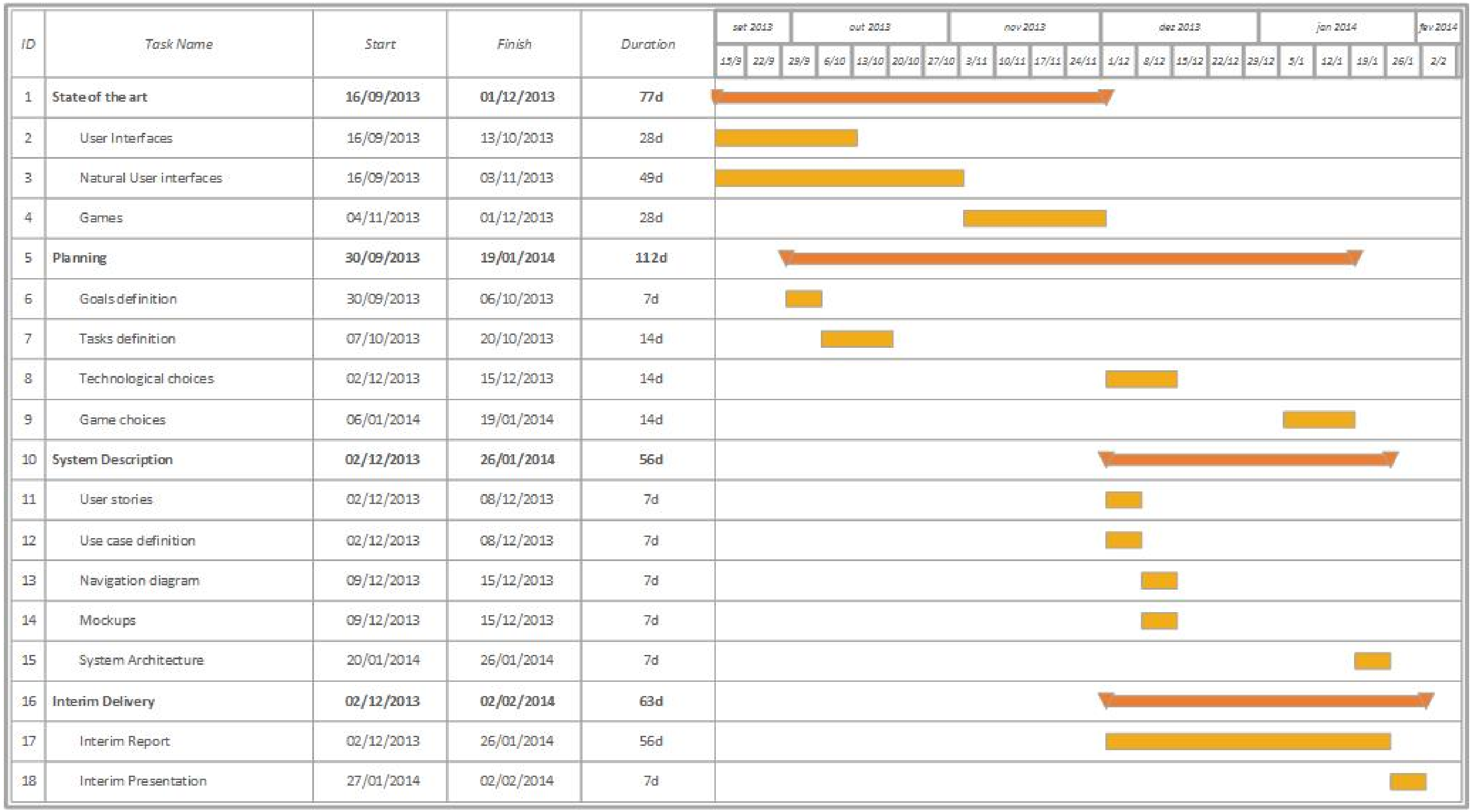
Table 4.7: Games Publishing

***Final Delivery***

This task is divided into two subtasks: the elaboration of the final report and the preparation for the final presentation.

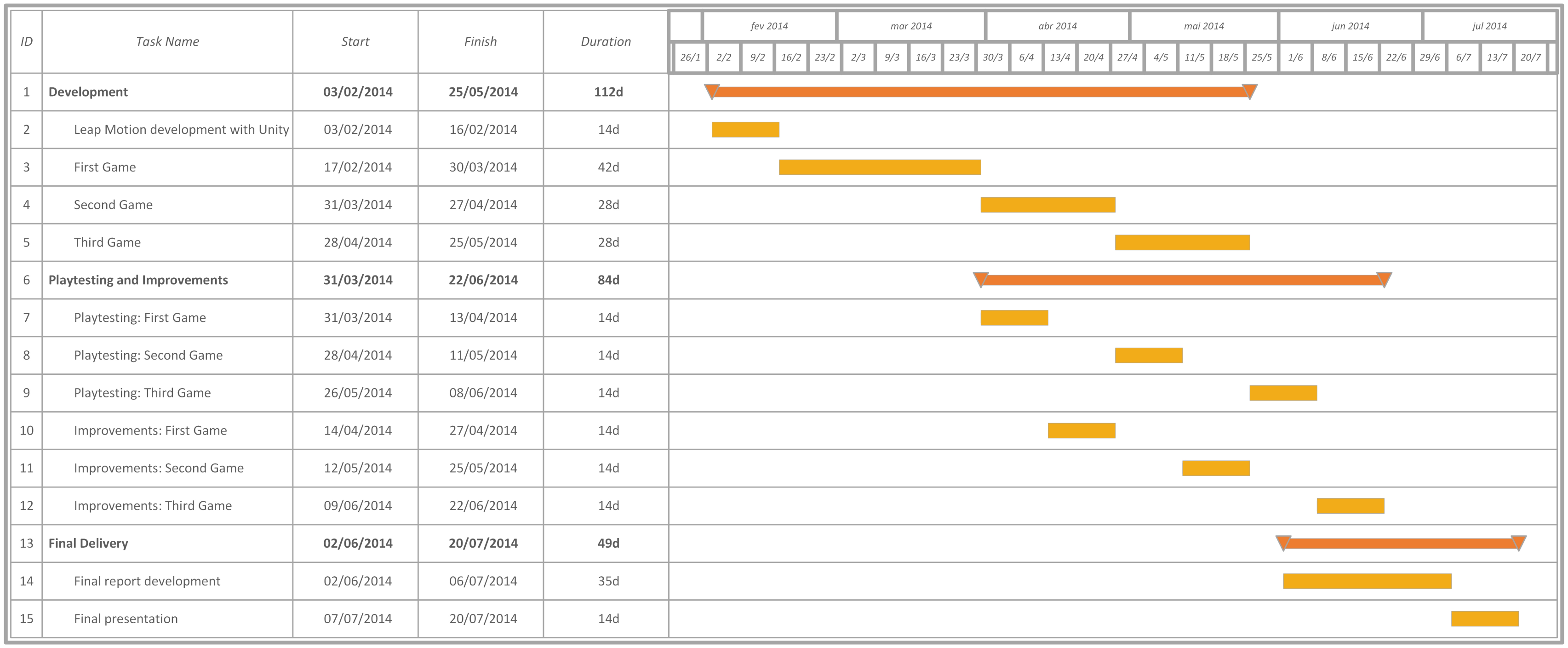
| **7. Final Delivery** | **Description** | **Begin Date** | **End Date** |
|---|---|---|---|
| 7.1. Final report development | Elaboration of the project's final report. | 02.06.2014 | 06.07.2014 |
| 7.2. Final presentation | Preparation of the project's final presentation. | 07.07.2014 | 20.07.2014 |

Table 4.8: Final Delivery



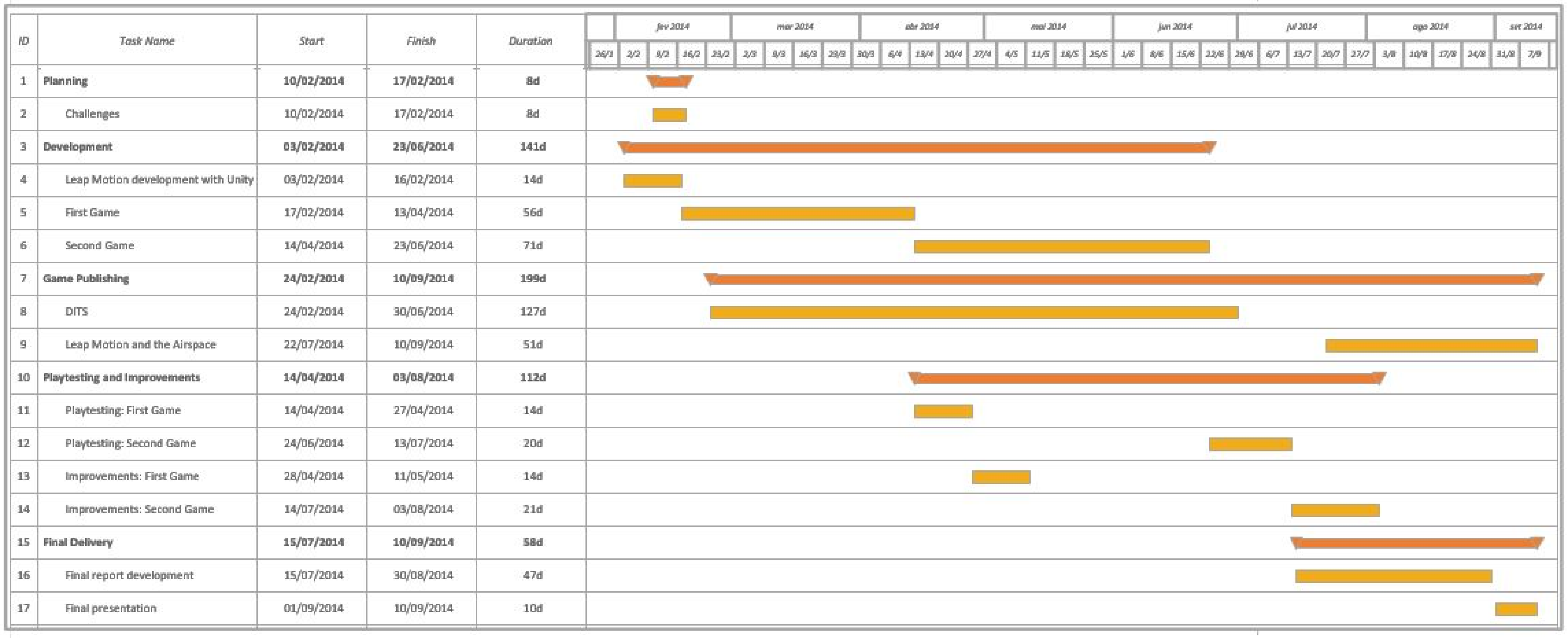
| ID | Task Name | Start | Finish | Duration |
|---|---|---|---|---|
| 1 | **State of the art** | **16/09/2013** | **01/12/2013** | **77d** |
| 2 | User Interfaces | 16/09/2013 | 13/10/2013 | 28d |
| 3 | Natural User interfaces | 16/09/2013 | 03/11/2013 | 49d |
| 4 | Games | 04/11/2013 | 01/12/2013 | 28d |
| 5 | **Planning** | **30/09/2013** | **19/01/2014** | **112d** |
| 6 | Goals definition | 30/09/2013 | 06/10/2013 | 7d |
| 7 | Tasks definition | 07/10/2013 | 20/10/2013 | 14d |
| 8 | Technological choices | 02/12/2013 | 15/12/2013 | 14d |
| 9 | Game choices | 06/01/2014 | 19/01/2014 | 14d |
| 10 | **System Description** | **02/12/2013** | **26/01/2014** | **56d** |
| 11 | User stories | 02/12/2013 | 08/12/2013 | 7d |
| 12 | Use case definition | 02/12/2013 | 08/12/2013 | 7d |
| 13 | Navigation diagram | 09/12/2013 | 15/12/2013 | 7d |
| 14 | Mockups | 09/12/2013 | 15/12/2013 | 7d |
| 15 | System Architecture | 20/01/2014 | 26/01/2014 | 7d |
| 16 | **Interim Delivery** | **02/12/2013** | **02/02/2014** | **63d** |
| 17 | Interim Report | 02/12/2013 | 26/01/2014 | 56d |
| 18 | Interim Presentation | 27/01/2014 | 02/02/2014 | 7d |

Figure 4.1: Gantt Diagram for the first semester.



| ID | Task Name | Start | Finish | Duration |
|---|---|---|---|---|
| 1 | **Development** | **03/02/2014** | **25/05/2014** | **112d** |
| 2 | Leap Motion development with Unity | 03/02/2014 | 16/02/2014 | 14d |
| 3 | First Game | 17/02/2014 | 30/03/2014 | 42d |
| 4 | Second Game | 31/03/2014 | 27/04/2014 | 28d |
| 5 | Third Game | 28/04/2014 | 25/05/2014 | 28d |
| 6 | **Playtesting and Improvements** | **31/03/2014** | **22/06/2014** | **84d** |
| 7 | Playtesting: First Game | 31/03/2014 | 13/04/2014 | 14d |
| 8 | Playtesting: Second Game | 28/04/2014 | 11/05/2014 | 14d |
| 9 | Playtesting: Third Game | 26/05/2014 | 08/06/2014 | 14d |
| 10 | Improvements: First Game | 14/04/2014 | 27/04/2014 | 14d |
| 11 | Improvements: Second Game | 12/05/2014 | 25/05/2014 | 14d |
| 12 | Improvements: Third Game | 09/06/2014 | 22/06/2014 | 14d |
| 13 | **Final Delivery** | **02/06/2014** | **20/07/2014** | **49d** |
| 14 | Final report development | 02/06/2014 | 06/07/2014 | 35d |
| 15 | Final presentation | 07/07/2014 | 20/07/2014 | 14d |

Figure 4.2: Initial Gantt Diagram for the second semester.



| ID | Task Name | Start | Finish | Duration |
|---|---|---|---|---|
| 1 | **Planning** | **10/02/2014** | **17/02/2014** | **8d** |
| 2 | Challenges | 10/02/2014 | 17/02/2014 | 8d |
| 3 | **Development** | **03/02/2014** | **23/06/2014** | **141d** |
| 4 | Leap Motion development with Unity | 03/02/2014 | 16/02/2014 | 14d |
| 5 | First Game | 17/02/2014 | 13/04/2014 | 56d |
| 6 | Second Game | 14/04/2014 | 23/06/2014 | 71d |
| 7 | **Game Publishing** | **24/02/2014** | **10/09/2014** | **199d** |
| 8 | DITS | 24/02/2014 | 30/06/2014 | 127d |
| 9 | Leap Motion and the Airspace | 22/07/2014 | 10/09/2014 | 51d |
| 10 | **Playtesting and Improvements** | **14/04/2014** | **03/08/2014** | **112d** |
| 11 | Playtesting: First Game | 14/04/2014 | 27/04/2014 | 14d |
| 12 | Playtesting: Second Game | 24/06/2014 | 13/07/2014 | 20d |
| 13 | Improvements: First Game | 28/04/2014 | 11/05/2014 | 14d |
| 14 | Improvements: Second Game | 14/07/2014 | 03/08/2014 | 21d |
| 15 | **Final Delivery** | **15/07/2014** | **10/09/2014** | **58d** |
| 16 | Final report development | 15/07/2014 | 30/08/2014 | 47d |
| 17 | Final presentation | 01/09/2014 | 10/09/2014 | 10d |

Figure 4.3: Re-assessed Gantt Diagram for the second semester.

### 4.3.2 *Trello*

*Trello* is a tool that allows individuals or groups of people to organize their projects into boards.

A board is a representation of a project and is composed of lists and cards. Within a board (as can be seen in figure 4.4), one can create their own lists and cards and organize them in the way that works best for them. A list can have multiple cards and a card may also be subdivided with inner checklists.

Besides the lists with the cards that are yet to be completed, there's also a "Done" list, to which the user can drag the cards whenever they are finished. Figure 4.4 shows a board with lists while figure 4.5 shows an example of a card.

*Trello* was used in this project to define tasks and subtasks, define delivery dates for each task and add new tasks whenever such tasks appeared.

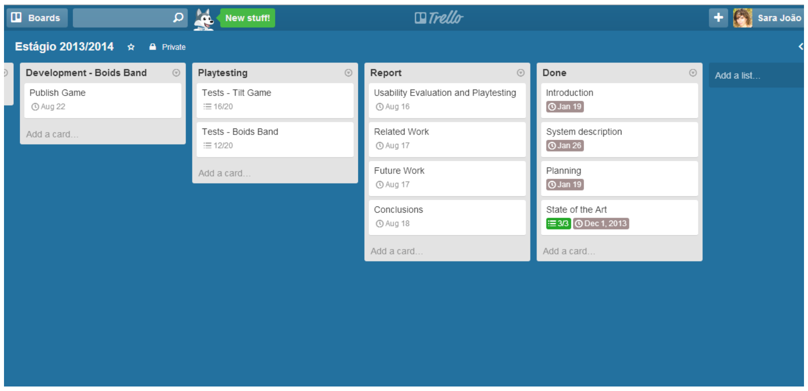


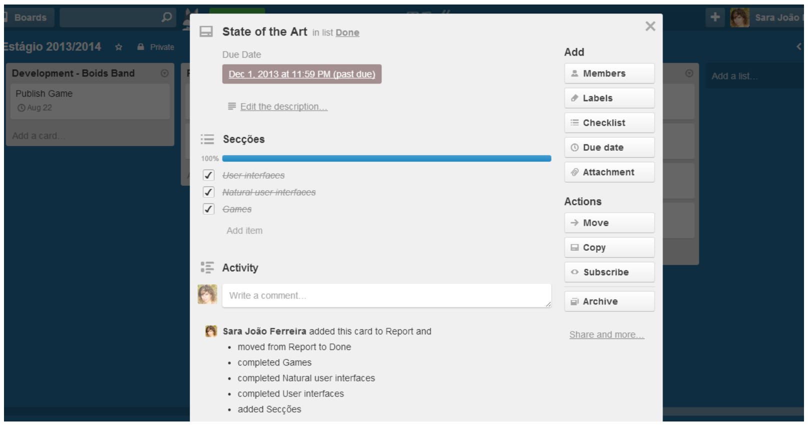
Figure 4.4: A *Trello* board can be divided in multiple lists, each composed of several cards.

## 4.4 Technological choices

This subsection describes the choices that were made about the technological tools that were used in this project.

#### *Leap Motion*

The games were developed for *Leap Motion*. As has already been referred in subsection 2.2.3, this device is very promising for the creation of natural user interfaces since it allows the user



State of the Art in list Done

Dec 1, 2013 at 11:59 PM (past due)

Secções

- ✓ ~~User interfaces~~
- ✓ ~~Natural user interfaces~~
- ✓ ~~Games~~

**Sara João Ferreira** added this card to Report and
- moved from Report to Done
- completed Games
- completed Natural user interfaces
- completed User interfaces
- added Secções

Figure 4.5: A *Trello* card represents a task and can contain a list of subtasks.

to interact with the computer using hand gestures instead of the traditional mouse. Even though *Leap Motion* is not the only device for creating these interfaces, it is a great choice since it is less expensive than the alternatives.

Another reason for choosing *Leap Motion* has to do with the ease of putting created apps in the market. The games that were made for this project are to be put on *Airspace*[3]: the *Leap Motion* store. *Leap Motion* allows developers to put their apps in their store for free as long as it follows their set of guidelines[4]. The developer can choose at what price the app should be sold (from *free* to *999,99$*) and receives 70% of every app sold.

At the time of writing this report[5] there are only 123 games in the Airspace store, of which 38 are free. This means it is a market where there is still plenty of space for new games to appear.

However, as was discussed in section 2.3.3, the *Leap Motion* device is still not free of limitations, and some of these limitations lead to project delays and some changes in planning as was referred in subsection 4.3.1.

#### ***Development***

*Leap Motion* apps can be written in *C++*, *Objective-C*, *C#*, *Java*, *Python* and *JavaScript*. There are two game engines that enable *Leap Motion* development: *Unity 3D* and *Cocos 3D*[6]. Developing with a game engine facilitates the process of creating a cross-platform game [79][80].

[3]More information available online at `airspace.leapmotion.com`
[4]More information available online at `developer.leapmotion.com/apps/guidelines`
[5]15th of August of 2014
[6]More information available online at `brenwill.com/cocos3d/`

A comparison was made between the two game engines, as can be seen in table 4.9.

| **Game Engines** | ***Unity 3D*** | ***Cocos 3D*** |
|---|---|---|
| Cost | Free version | Free version |
| Supported Platforms | Windows, Mac OS | Windows, Mac OS |
| Supported Targets | Windows, Wii, Web, Flash, Mac OS, iOS, Android, PS3, Xbox360, Others | iOS, Android, OSX |
| Supported Languages | C#, JavaScript, Boo | Objective-C |
| Documentation | Plenty | Fairly |

Table 4.9: Comparison between Game Engines

The choice was made in favor of *Unity 3D* for several reasons: the languages allowed were better known to the developer, it supports a larger amount of target operating systems and the documentation that exists for developing games in *Unity 3D* and more specifically for *Leap Motion* is very extensive.

*Unity 3D* allows the developer to use either *Javascript*, *C#* or *Boo*. In terms of performance the languages seem to be similar. [81][82] Therefore, the choice that was made to use *C#* was based solely on the developer's background and experience.

## 4.5 Game choices

The main goal of this project was the reinvention of traditional and common games using natural user interfaces. With that in mind, some games were chosen that seem adequate for this reinvention.

In this section the chosen games are described through their main characteristics: number of players, goal and rules description. Some examples of variants for each game are also presented.

This section also elaborates on the games that were decided to pursuit in this project.

### 4.5.1 Chosen Games

This subsection presents some games that would be interesting to reinvent for the *Leap Motion* device.

#### ***The Tilt Game***

The game: The Tilt Game is a single player game and its goal is for the player to drive a ball through a maze. This game usually includes a maze and one or more balls. There is a hole in the maze for each ball. There are two main variants of this game.

In the first variant, the player has to move the board in order to drive the ball(s) through the maze and to the hole(s). In the other variant of the game, the maze has a beginning and an end point and the path between them is filled with holes. The player has to move the ball from the beginning to the end without the ball falling in any hole.

Examples: This game exists both physically and for the computer.

The physical games usually include handles that allow the user to turn the maze. An example of these games can be seen in figure 4.6.

To simulate this physical game, some video games were created. In this games the player uses the mouse to move the maze and drive the ball. An example of these games is TILT[7], a free video game (4.7). With this game the player can also see how much time it takes them to win each level and improve their high scores.

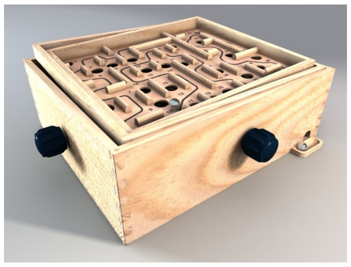


Figure 4.6: A Tilt physical game.

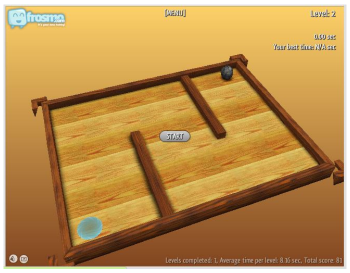

frasmo.com
[MENU]
Level: 2
0.00 sec
Your best time: N/A sec
START
Levels completed: 1, Average time per level: 8.16 sec, Total score: 81

Figure 4.7: A Tilt video game.

Reasons for the choice: This game is very interesting in terms of gesture possibilities. Although some versions exist with graphical user interfaces, they are usually difficult to play. The *Leap Motion* device could allow for a different and more compelling version of the game.

[7]More information available online at `www.silvergames.com/tilt`

### ***Shooting Game***

The game: This is a game than can either be single or multiplayer. Its goal is to shoot as many targets as possible.

The player has a (play) gun and has to shoot against the targets. The targets can be moving or standing still depending on the game. When a target is hit, the player receives points. The number of points given by hitting a target can vary in some games, depending on the difficulty of that specific target. In some games there are also fake targets that the player must avoid hitting (see figure 4.8).

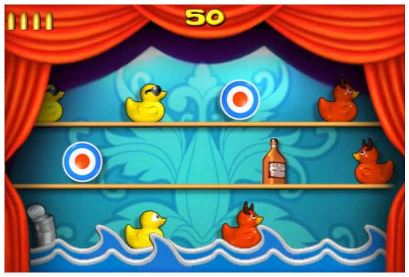


Figure 4.8: In this shooting game for iPhone, if the players shoots a red duck he will lose points.[8]

Examples: Games of shooting are very common in carnivals but there are also many video games that use this concept. A lot of times, the subject of the shooting are ducks (figure 4.9).

Figure 4.9: The *Carnival Duck Shooting* game is a physical game that comes with a wireless revolver that shoots infrared beams.

Reasons for the choice: Shooting games usually involve the player holding an object. With *Leap*

[8]More information available online at `itunes.apple.com/app/carnival-shooting-gallery/id312666738?mt=8`

*Motion*, a version can be created where the player's hand is the weapon.

***Driving Game***

The game: This is a game for one or more players (depending on the game). Its goal is to drive a vehicle avoiding obstacles.

These games work as driving simulators. The user is responsible for driving a virtual car and has to avoid going into obstacles.

Examples: There are a big variety of driving games: in some of them the goal is simply to simulate driving (figures 4.10 and 4.11) while others are racing games where the player also has to be the first driver to cross the finish line in order to win. Need for Speed (figure 4.12) is one of these games. This has been a recurring theme for arcade games, electronic games, computer games, mobile games among others.

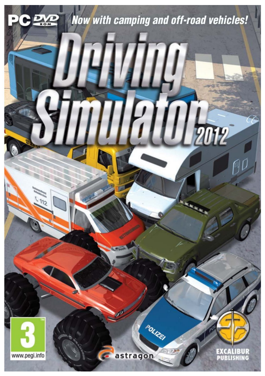


Figure 4.10: *Driving Simulator*: a PC game.

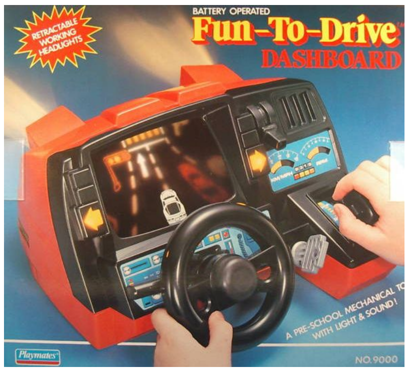

RETRACTABLE WORKING HEADLIGHTS
BATTERY OPERATED
Fun-To-Drive
DASHBOARD
A PRE-SCHOOL MECHANICAL TOY WITH LIGHT & SOUND!
Playmates
NO. 9000

Figure 4.11: *Fun-to-drive Dashboard*: a mechanical toy for toddlers that simulates driving.

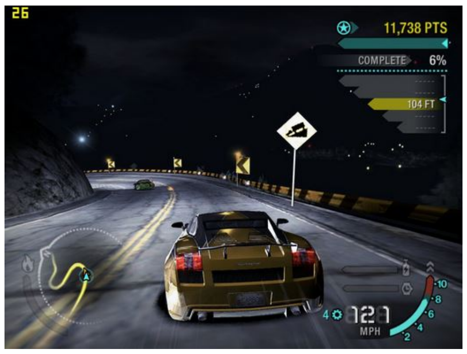


Figure 4.12: *Need for Speed*: a PC racing game.

Reasons for the choice: Driving games are very popular and a driving game that does not involve holding any object could be very interesting.

### *Throw a Ball Game*

The game: This is usually a single player game and its goal is to throw a ball against objects in an attempt to make them fall.

The player is given a set of balls that he must throw (one of a time) to a set of objects (usually cans) in order to try and make the all fall down. The player wins if he is able to make all the objects fall down with the set of balls he is given.

Examples: This game is very common in carnivals (figure 4.13) and fairs where the player usually gets a prize if he wins. Usually the objects that the player has to make fall are cans (figure 4.14).

Figure 4.13: Some carnival games such as *throw a ball* and *ring toss*.

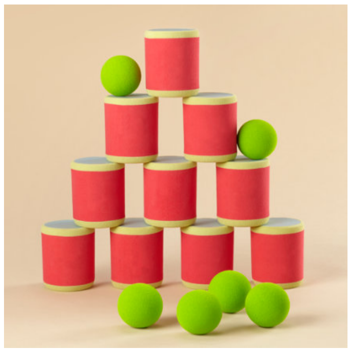


Figure 4.14: Throw a Ball game.

Reasons for the choice: This is a simple game but is also very popular and fun. The *Leap Motion* can bring new possibilities for this game.

### 4.5.2 Existing Games

Before making any decisions on what games to chose, the Airspace store was examined in order to find out what had already been developed. No games similar to the Tilt Game were available. However, some games that used the other three concepts (shooting, driving and throw a ball games) were found.

**Shooting Game**
Two games that use the concept of a Shooting Game were found. Both these games are rated age 16+.

The first one was Duck-n-Kill[9], a single player game whose main goal is to shoot the ducks that appear on screen. The player points at the screen with the index finger and the thumb up (as can be seen in figure 4.15) as to make a gun with their hand. To shoot, the player lifts is hand quickly. This is a 2D game as can be seen in figure 4.16.

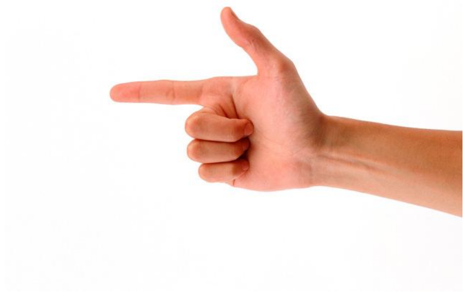


Figure 4.15: Gun pointing gesture with hand.

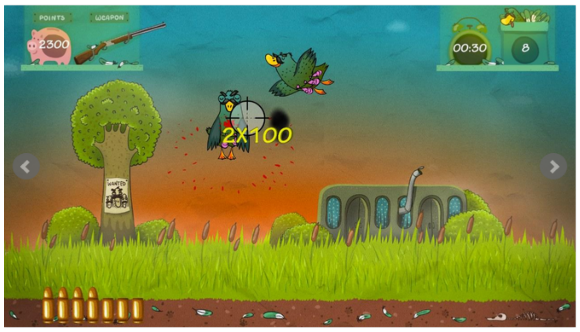


Figure 4.16: Duck-n-Kill screen capture.

[9]More information available online at: `airspace.leapmotion.com/apps/duck-n-kill/windows`

Blue Estate[10] is also a shooting game but in 3D. The gesture to shoot is the same as in Duck-n-Kill (4.15).

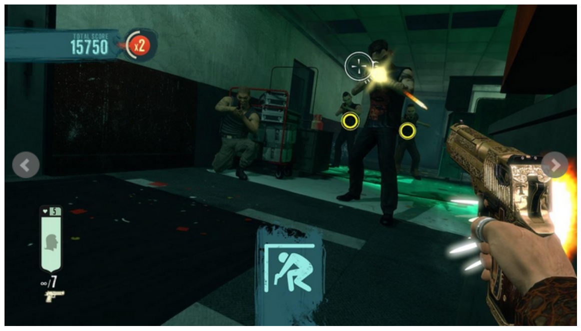


Figure 4.17: Blue Estate screen capture.

### Driving Game

There were two driving games in the Airspace store, and in both of them the player is responsible for driving a spaceship. Both these games are 3D and are rated age 7+. The first game is Catch Up Calu[11] and its screenshot can be seen in figure 4.18. The second game is Escape Velocity[12] and can be seen in figure 4.19. In both these games, the player uses their hand straight as can be seen in figure 4.18 and tilts the hand to the right or the left according to which way they want the spaceship to move.

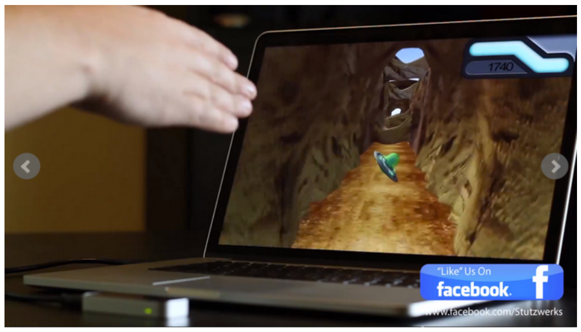


Figure 4.18: Catch Up Calu screen capture.

[10]More information available online at: `airspace.leapmotion.com/apps/blue-estate/windows`
[11]More information available online at: `airspace.leapmotion.com/apps/catch-up-calu/windows`
[12]More information available online at: `airspace.leapmotion.com/apps/escape-velocity/windows`

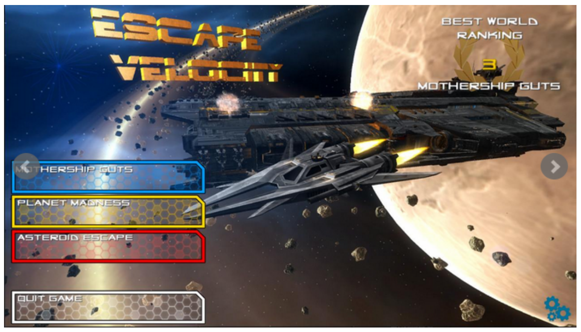


Figure 4.19: Escape Velocity screen capture.

**Throw a ball Game**
Two game were also found in which the player had to throw some sort of ball to a pile of objects in order to make them fall down. Both these games are rated age 3+.

The first game is Boom Ball[13] (figure 4.20) and there is already a sequel of this game: Boom Ball Adventures[14] (figure 4.21). In both these games the player uses their hand to make a ball go into a pile of boxes. The player points with their index finger at the screen and a disc surrounds the point to where the player is pointing. When the ball goes against that disc it ricochets and moves in the opposite direction.

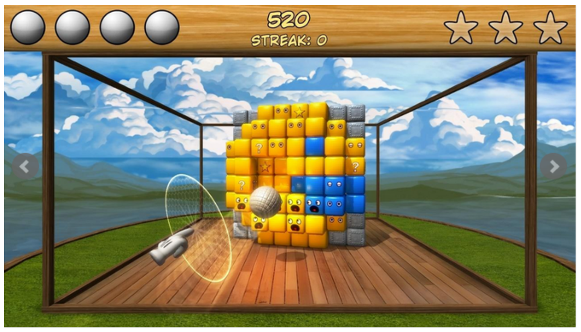


Figure 4.20: Boom Ball screen capture.

The second game is a bowling game and it is called Super Punch Bowl[15] (figure 4.22). The player starts by moving their hand right or left to chose the direction in which to throw the ball and then let go of the ball. The ball is then thrown against the bowling pins.

[13]More information available online at: `airspace.leapmotion.com/apps/boom-ball/windows`
[14]More information available online at: `airspace.leapmotion.com/apps/boom-ball-adventures/windows`
[15]More information available online at: `airspace.leapmotion.com/apps/super-punch-bowl/windows`

Figure 4.21: Boom Ball Adventures screen capture.

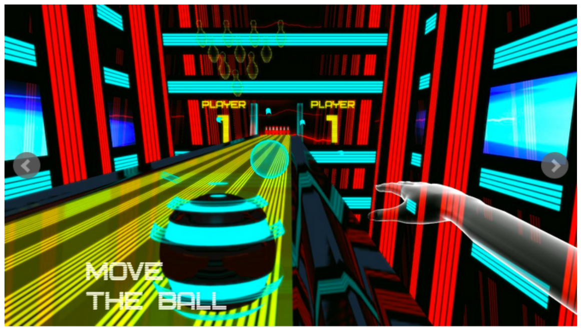


Figure 4.22: Super Punch Bowl screen capture.

### 4.5.3 Developed Games

For the first game it was decided to reinvent the Tilt Game. The goal of the developed game is to lead a ball through a maze into a goal without letting it fall into any holes. The player has to move their hand in order to move the board. The technicalities of this game are further explained in section 5.1.

Since the concepts for the other games already existed, it was decided to merge these concepts into a new kind of game. Therefore, in a attempt to create something original, a different game was thought of.

The developed game follows and leads a group of birds that fly through a scenery and have to attack a different species of birds whenever they get in the groups way. If the group finds another bird of the same species that bird can join them. The group also has to find food in order to survive. The technicalities of this game are further explained in section 5.2.

# 5. Development

One of the main goals of this project was to reinvent games for the Leap Motion device, by using Natural User Interfaces and following the established principles and guidelines (subsection 2.2.2) that these types of interface should follow.

This chapter explains the production of the two games that were built, since the planning phase until the deployment phase. There were also some tests and evaluations made to the games, using real players, but those are further discussed in chapter 6. The chapter is divided into four sections: the first two discuss for each game, the definition of the game, the planning phase that preceded the development and the main development issues that were addressed; the third section explains the usability choices that were made in order to ensure that the games followed the NUI guidelines and principles; and finally, the four section explains the process that was made in order to publish the games into the *Leap Motion* store (Airspace).

## 5.1 Tilt Game

The first game to be developed was a reinvention of a board game: the Tilt Game (subsection 4.5.1).

This section is divided into four subsections:

- **Game Description** explains how the game works.
- The **System Description** explains the process prior to the development that included adapting the general use cases, navigation diagram and mockups to this specific game.
- In **Development** some details about the development of the game in *Unity 3D* are further explained.

### 5.1.1 Game Description

To clearly define the game, some major points had to be decided. These points were:

- The **components** of the game: the objects needed to create the game.
- The **rules of the game** that make sense of the game and determine how the player wins or loses.
- How the **difficulty** of the game changes through different levels
- The **rules of the board generation**
- How the player **scores** in the game
- The **gestures** that were used in order to rotate the board

In this subsection, these points are further explained.

#### ***Components***

The components of this game are:

- **The ball**: The object that the player as to move in order to play the game.
- **Holes**: The places where, if the ball falls, the player loses.
- **The goal**: The place where, if the ball falls, the player wins.
- **Initial point**: The place where the ball starts.
- **Board's floor**: The places where the ball can go through without the player losing.
- **Board's walls**: The walls that form the maze the ball must go through.

For development purposes, all the components were designed the same size: a square of 1 per 1 unit. The board measures 30 per 20 units, which allows it to contain 600 components.

#### ***Rules of the game***

This game has a very simple set of rules:

1. If you fall into a hole, you lose.
2. If you manage to get from the initial point to the goal without falling into a hole, you win.

#### ***Difficulty***

It was decided to keep with the usual three levels that many games use: **Easy**, **Medium** and **Hard**.

However, since this is a game that can become very difficult (nearly impossible!) if the number of holes is high enough, it was decided to also create an extra level: **Impossible**.

What changes from level to level is the board. The changes are based on the insertion of more holes and less walls (since holes difficult the game and walls facilitate it).

***Rules of the board generation***

A different board is created each time the player starts a new game. The generation of the board is procedural, more precisely pseudo random. The technicalities on this subject are further explained in subsection 5.1.3. To generate each board in a way that serves the purpose of the game a set of rules on how it must be created were defined:

- The board is composed of 30 per 20 units (each unit is a component as mentioned earlier)
- There is always one (and one only) initial point
- There is always one (and one only) goal
- There is always one (and one only) ball
- There is always a path (composed of floor) from the initial point to the goal
- There are a number of holes based on the chosen level
- There are a number of walls based on the chosen level

***Scores***

This is a game that incites for the player to be as fast as possible. The idea is to lead the ball into its goal as fast as you can. With this in mind, it only made sense for the highest score to be the lowest time. This could not be applied to the game as a whole, since the time it takes to win a game in the Easy level is different from the time it takes to win a game in the Hard level. This led to the creation of a score board for each of the four levels, in which the best 10 results (the lowest times the player as accomplished) for each level are shown.

***Gestures***

It was decided to have the board mimicking the player's hand, as if the hand was glued to the board. The usability choices behind this decision are further explained in section 5.3.

### 5.1.2 System Description

Prior to the development of the Tilt Game, some of the methods used to describe the general system that would become each game were further specified. Therefore, a specific use case table for this game was built, such as specific mockups and a navigation diagram.

These documents supported the consequent development of the game. In this subsection the use cases table, the mockups and the navigation diagram are further explained.

***Use Cases***

To define the specific requirements for the Tilt Game, specific use cases were created. The following table (5.1) presents these use cases, their goals, pre and post conditions, main flow of events and secondary flow of events (when necessary).

Table 5.1: Tilt Game Use Cases

| Name | Goal | Pre-conditions | Post-conditions | Main Flow of Events | Secundary Flow of Events |
|---|---|---|---|---|---|
| 1. Visualize high scores | Allow the player to check their high scores for a chosen level. | There are none. | There are none. | The **player** indicates their intention to see the high scores.<br><br>The **player** chooses the level for which they want to check the high scores.<br><br>The **game** presents the high scores.<br><br>The **player** checks the high scores. | If there are no high scores yet:<br><br>The **game** presents an empty high scores table. |
| 2. Change sound effects settings | Allow the player to increase or decrease the volume for the game's sound effects. | There are none. | The sound effects settings are saved. | The **player** indicates their intention to change the sound effects settings.<br><br>The **game** presents an item that allows the player to change the volume of the sound effects.<br><br>The **player** adapts the volume.<br><br>The **game** saves the changes. | |
| 3. Change music settings | Allow the player to increase or decrease the volume for the game's music. | There are none. | The music settings are saved. | The **player** indicates their intention to change the music settings.<br><br>The **game** presents an item that allows the player to change the volume of the music.<br><br>The **player** adapts the volume.<br><br>The **game** saves the changes. | |
| 4. Change camera settings | Allow the player to change the angle of the camera. | There are none. | The camera settings are saved. | The **player** indicates their intention to change the camera settings.<br><br>The **game** presents an item that allows the player to change the angle of the camera.<br><br>The **player** choses an angle.<br><br>The **game** saves the changes. | |
| 5. Change player nickname | Allow the player to change its player nickname. | There are none. | The nickname is saved. | The **player** indicates their intention to change their nickname.<br><br>The **game** presents a text box that allows the player to change their nickname.<br><br>The **player** types in the new nickname.<br><br>The **game** saves the changes. | If any of the characters of the new nickname is invalid:<br><br>The **game** indicates to the user that some of the characters cannot be used and asks for a valid nickname.<br><br>The **player** inserts a valid nickname.<br><br>The **game** saves the new nickname if it is valid. Otherwise repeats. |

| | | | | | |
|---|---|---|---|---|---|
| 6. Visualize credits | Allow the players to see the game's credits. | There are none. | There are none. | The **player** indicates their intention to see the credits.<br><br>The **game** presents the credits.<br><br>The **player** sees the credits. | |
| 7. Visualize instructions | Allow the player to learn how to play the game. | There are none. | There are none. | The **player** indicates their intention to visualize the instructions.<br><br>The **game** presents the instructions.<br><br>The **player** sees the instructions. | |
| 8. Change level settings | Allow the player to learn how to change the level settings. | There are none. | The chosen level is saved. | The **player** indicates their intention to change the level settings.<br><br>The **game** presents an item that allows the user to chose between different difficulty levels.<br><br>The **player** selects the desired level.<br><br>The **game** saves the changes. | |
| 9. Play | Allow the player to play the game. | There are none. | The game is playing. | The **player** indicates their intention to start playing.<br><br>The **game** starts.<br><br>The **player** uses gestures to play.<br><br>The **game** alters its state according to those gestures. | If the game is lost:<br><br>The **game** saves the player's score, if it is a new high score, and resumes to the "win" state. |

***Navigation Diagram***

In order to represent the flow between the different parts of the game, a specific navigation diagram was designed (figure 5.1). This diagram represents the flow that occurs between the different parts of the game.

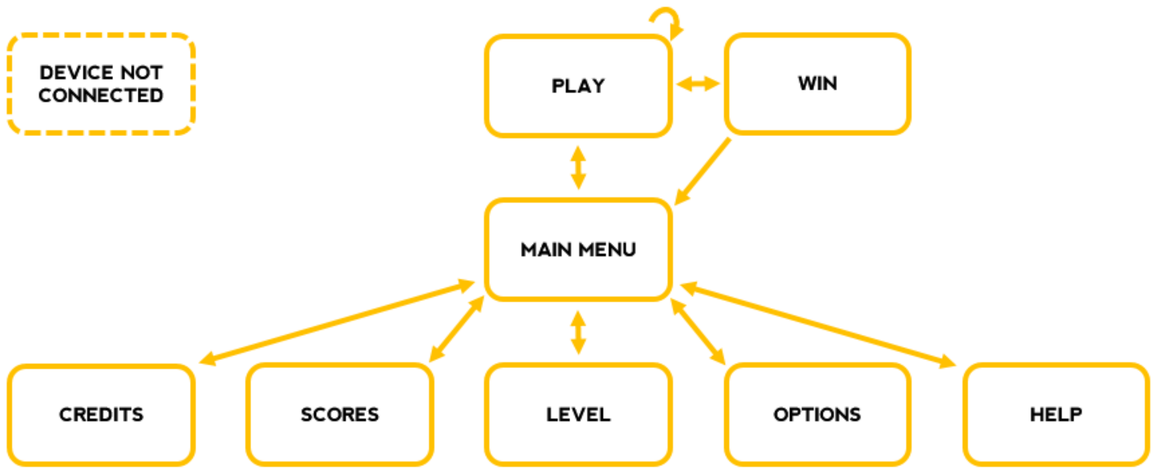


Figure 5.1: Navigation diagram for the Tilt Game.

The main menu allows the user to chose from a number of different actions in the game that are described by the previously defined use cases 5.1.

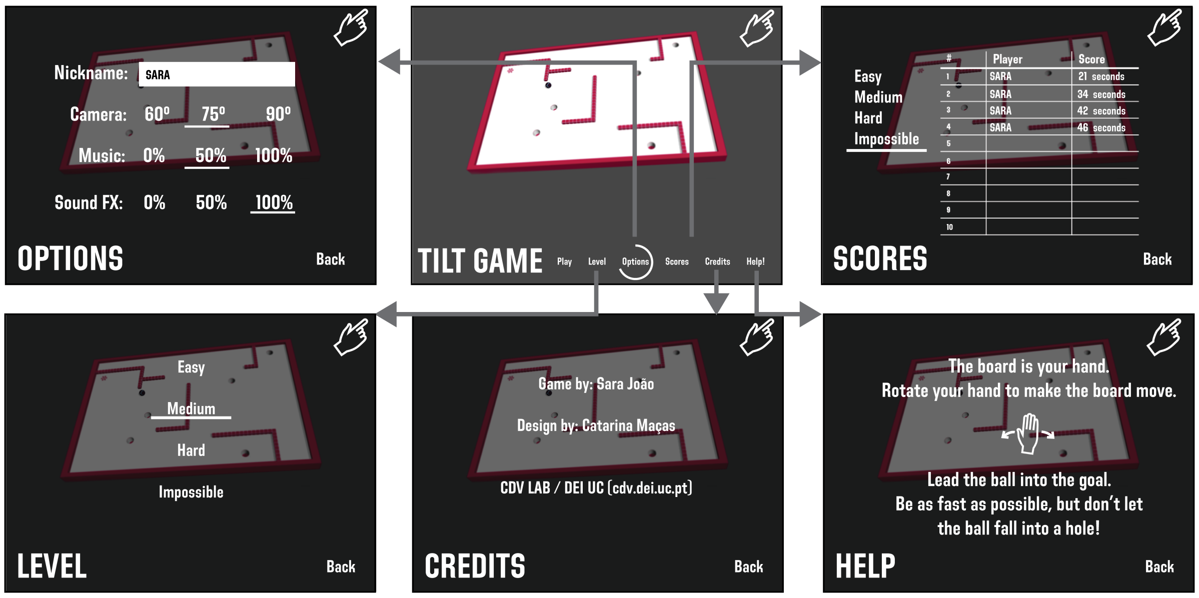
The diagram is very similar to the previously designed generic navigation diagram 3.1 with a few differences.

Since the game includes different difficulty levels it was necessary to include a section that allows the user to change the level according to the desired difficulty. Also, this is a very quick game, and it would be boring for the player to be redirected to a "Game Over" window every time they lost. With this in mind, the game was designed so that when the player loses, the game automatically restarts with a new labyrinth. This is reflected in the diagram by the arrow that leaves the "Play" window and goes right back to it. If the player wins, however, they will be redirected to the "Win" window, that will show the player their score and allow them to chose between going back to the main menu or playing again.

***Mockups***

In order to define the disposition of the windows of the Tilt Game, several mockups were designed.

The design of the objects that were used in the game were produced by Catarina Maçãs[1] a researcher of the CDV Lab at DEI/FCTUC. This design included the choice of the typography and colors, and the creation of the 3D and 2D objects that were used in the game. The disposition of the elements on the screen also included the designer's inputs.



Nickname: SARA
Camera: 60º 75º 90º
Music: 0% 50% 100%
Sound FX: 0% 50% 100%

| # | Player | Score |
|---|---|---|
| 1 | SARA | 21 seconds |
| 2 | SARA | 34 seconds |
| 3 | SARA | 42 seconds |
| 4 | SARA | 46 seconds |
| 5 | | |
| 6 | | |
| 7 | | |
| 8 | | |
| 9 | | |
| 10 | | |

Game by: Sara João
Design by: Catarina Maçãs
CDV LAB / DEI UC (cdv.dei.uc.pt)

The board is your hand.
Rotate your hand to make the board move.
Lead the ball into the goal.
Be as fast as possible, but don't let the ball fall into a hole!

Figure 5.2: Mockups for the Tilt Game: Main Menu, Options, Scores, Help, Level and Credits.

Figure 5.2 represents the first set of mockups. These exemplify how the Main Menu will look like as well as some of the states that can be accessed through that menu.

- The **Main Menu** presents all the possible actions for the player to take: Play, change level of game play, change other settings, check high scores, visualize the credits and visualize the instructions.
- The **Options** state allows the player to choose a nickname and change the camera angle and the volume.
- The **Level** state allows the player to change the difficulty of the game from between four levels: easy, medium, hard and impossible.
- The **Credits** state allows the player to visualize details on the creators of the game.

[1]More information available online at `http://cdv.dei.uc.pt/authors/catarina-macas/`

- The **Scores** state allows the player to check their high scores on the different difficulty levels.
- The **Help** state provides information to the player on how to play the game.

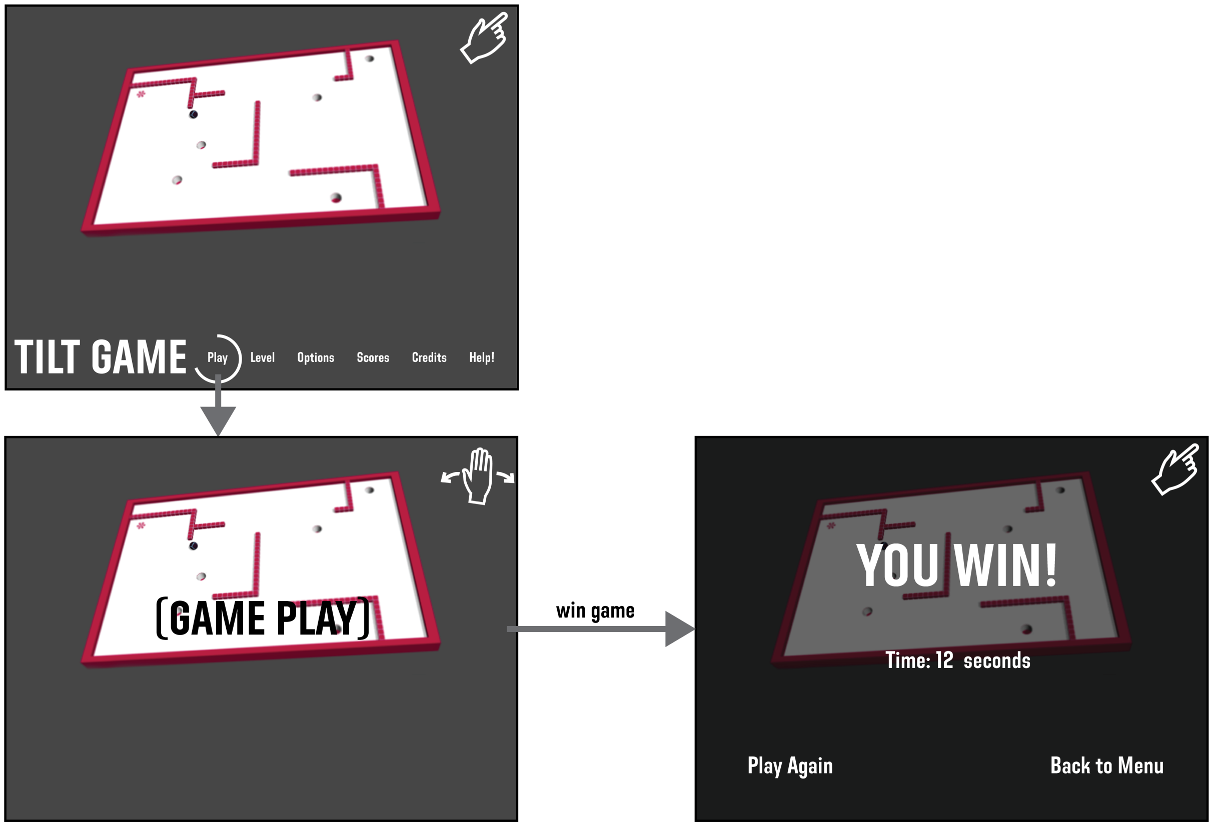


Figure 5.3: Mockups for the Tilt Game: Game Play and Win state.

Figure 5.3 represents the mockups for the **Play** action. The player starts by choosing to play in the Main Menu and is redirected to the game. When the player loses they are redirected to the **Win** state that indicates their score.

Finally, figure5.4 represents the mockup for the **Device not Connected** state, that appears when the *Leap Motion* device is not connected.

The main reasons for the choices that were made, related to the disposition of the objects on screen and the way the user interacts with them are further explained on the next subsection (5.3).

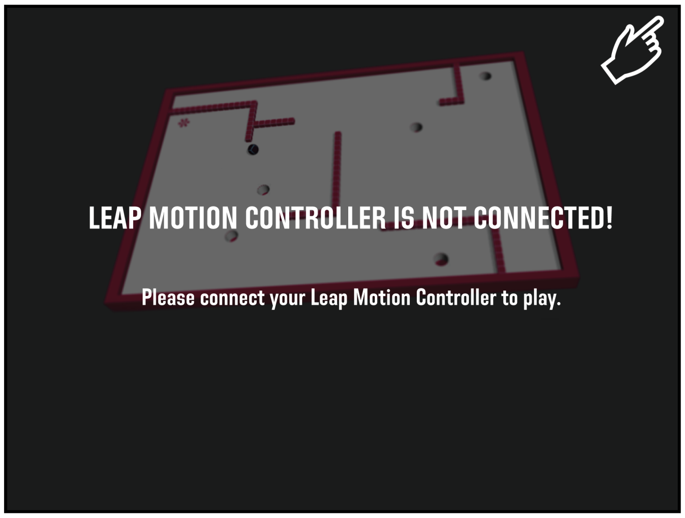

LEAP MOTION CONTROLLER IS NOT CONNECTED!
Please connect your Leap Motion Controller to play.

Figure 5.4: Mockups for the Tilt Game: Device disconnected state.

### 5.1.3 Development

This subsection explains the main challenges that were dealt with during the development phase of the Tilt Game. Procedural Generation was used to create a different board every time the game is initialized. For the rotation of the board the *Leap Motion* API is used to perceive in which way and in which angle the player intends to rotate the board.

***Board Generation***

As previously stated, the board is generated pseudo randomly. This is due to the fact that it uses *Unity 3D*'s pseudo random number generator to decide the initial point and the goal's position as well as the path. The algorithm that is used to define the board is explained next.

The board is initially divided into 24 quadrants, as is shown in figure 5.5.

In order to have at least one possible path in every game, passage points are marked between each consecutive quadrant. This passage points are generated randomly. At the end of this process, the board looks like the one in figure 5.6.

To chose an initial point and a goal's position is the next step. The initial point's position is always in one of the first six quadrants since it is more intuitive for the player that tha ball starts in the upper part of the screen. To chose it, one of these six quadrants is firstly chosen as the home of the initial point. Secondly, a point within that quadrant is chosen.

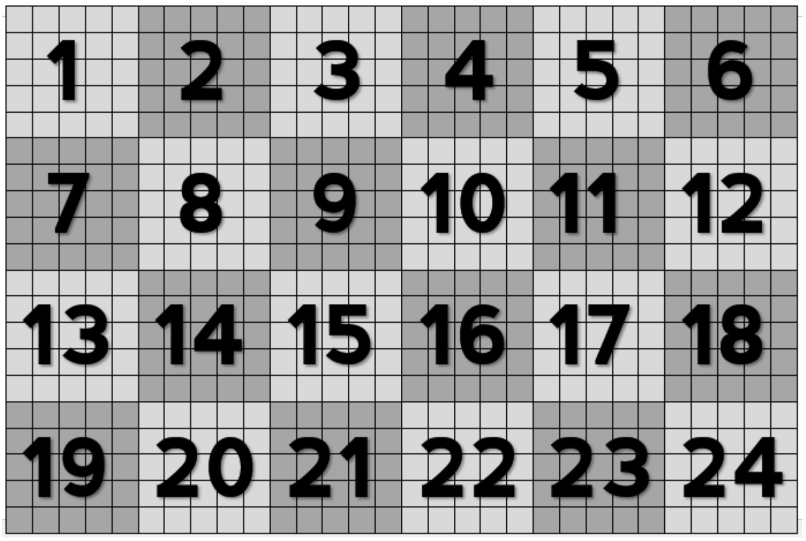


Figure 5.5: Division of the board into quadrants.

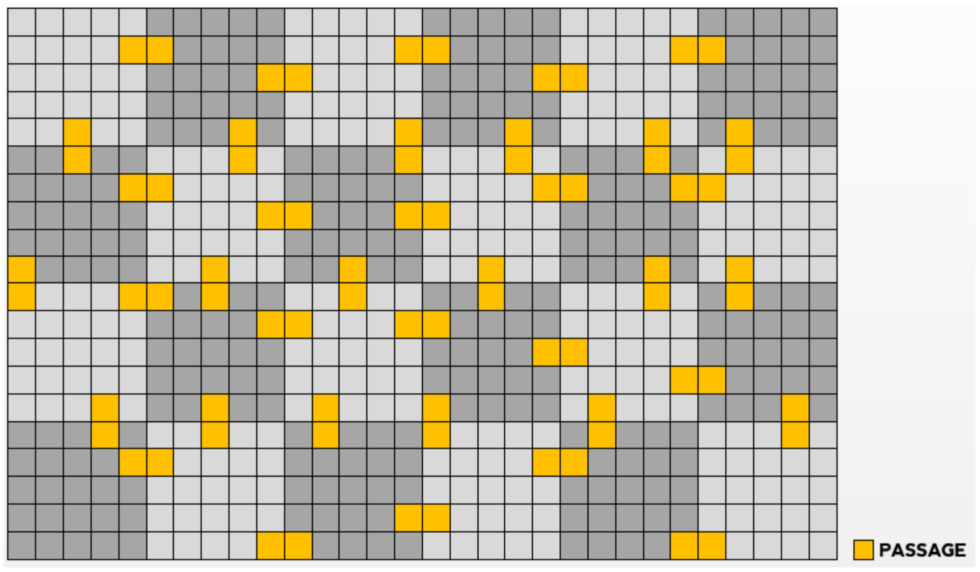


Figure 5.6: Board after marking the passage points.

The goal's position depends on which quadrant holds the initial point to ensure that it is far enough from the initial point. For example, if the initial point was in quadrant 1, it would probably be a very easy game if the goal was in quadrant 2, 7 or 8 (that are right next to quadrant 1). To prevent that from happening, the goal's position is assigned after the search for the initial point has ended. Figure 5.7 shows which quadrants would be available for goal's position if the initial point were in quadrant 1. A pseudo random number is generated for the goal's position in one of those quadrants.

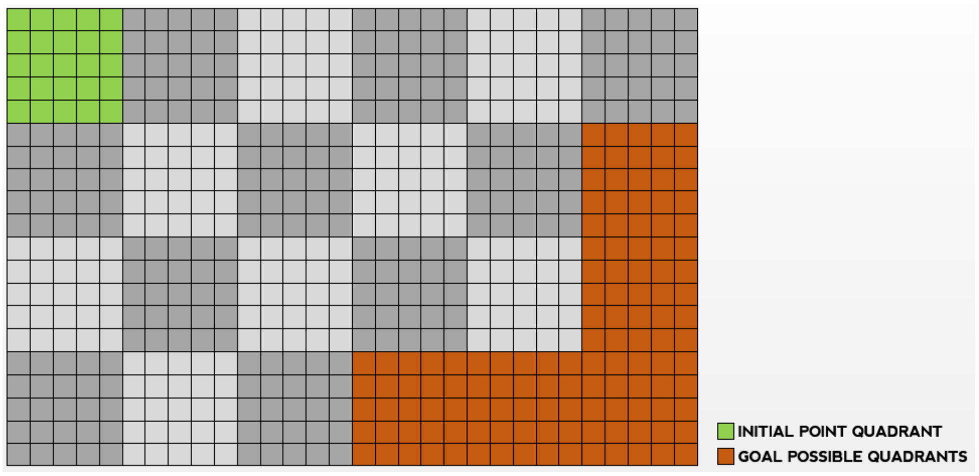


Figure 5.7: Possible quadrants to host the goal's position.

At the end of these step, both the initial point and the goal's positions have been found, as can be seen in figure 5.8.

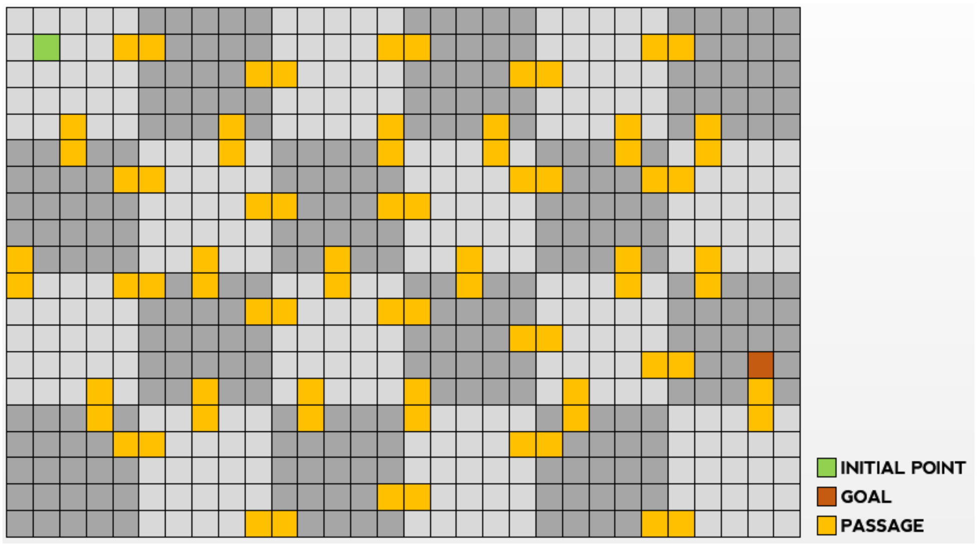


Figure 5.8: Board after marking the initial point and the goal.

The next step is to define a path (in which there are no holes or walls) from the initial point to the goal. This is made by assuring that there is a path between every passage point previously defined. There also has to be a path between the initial point and the passages in that quadrant. The same goes for the goal. At the end of this step, the board will look like the one in figure 5.9.

After the path has been marked, it is time to mark the walls. The number of walls that will be marked depend on the type of level that is being built.

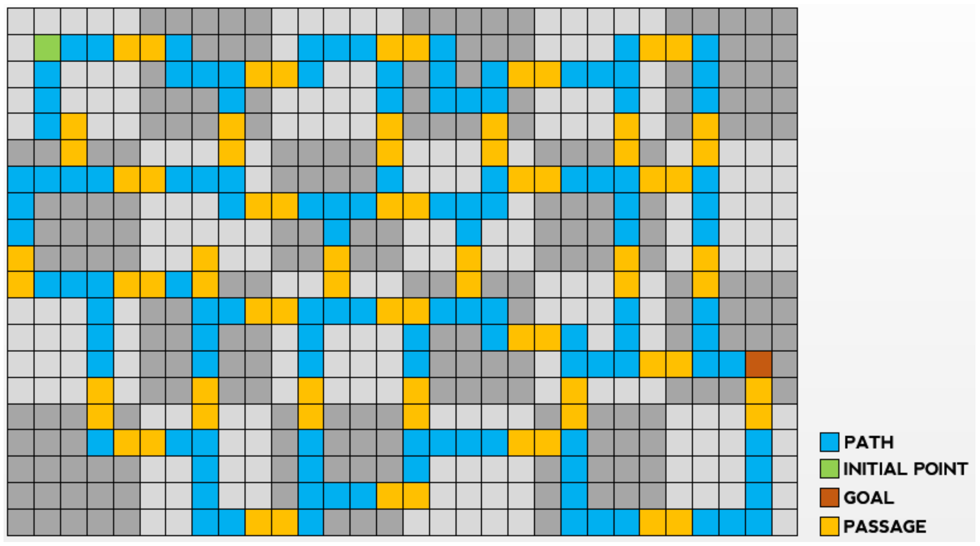


Figure 5.9: Board after marking the path.

Pseudo random positions are determined and if they have not yet been marked (as passage, initial point, goal or path) they will be marked as walls. The result is something similar to what is shown in figure 5.10.

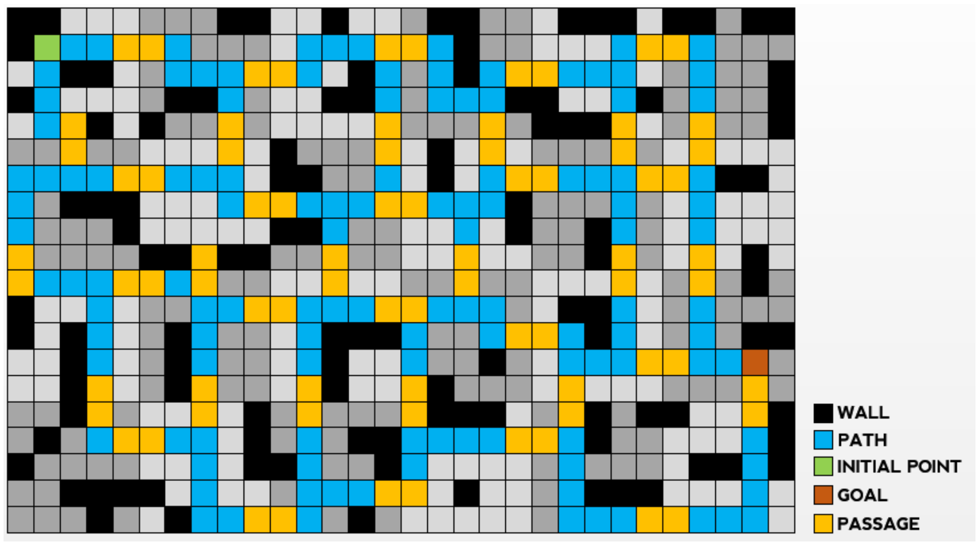


Figure 5.10: Board after marking the walls.

Finally, the holes are also included in the same way that the walls were. Pseudo random spaces are marked as holes as long as they have not been marked before as passage, initial point, goal, path or wall. The result is shown in figure 5.11.

At the end of all of these process, the board will be built as shown in figure 5.12.

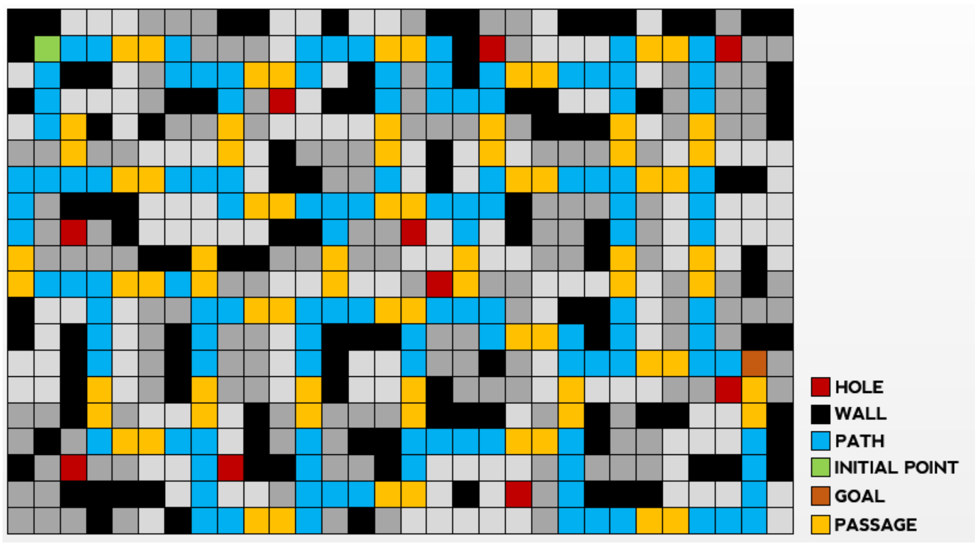


Figure 5.11: Board after marking the holes.

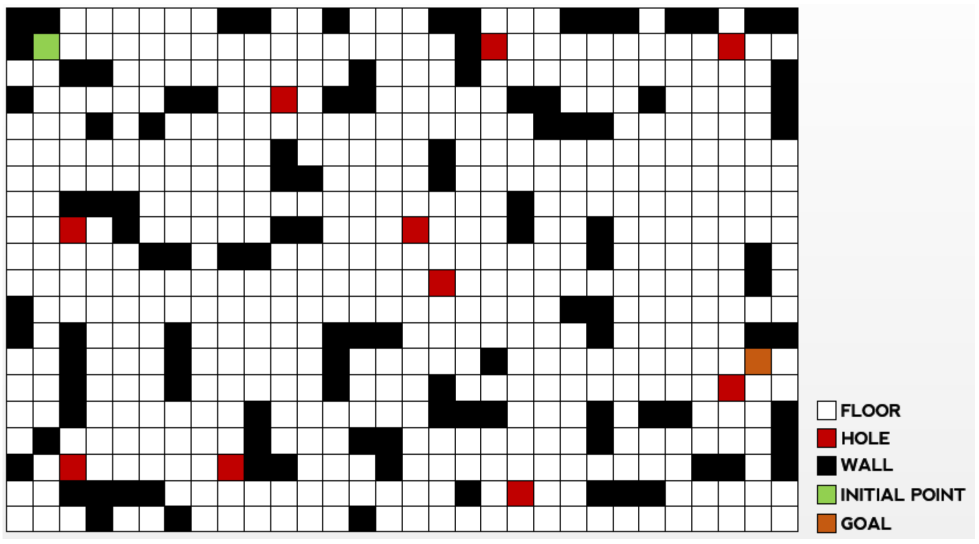


Figure 5.12: Final board

***Board Rotation***

To make the board rotate in a such a way that mimics the user's hand, the *Leap Motion* API was used. This API provides two vectors: the normal vector that points perpendicularly out of the hand and the direction vector that points forward. This is exemplified in figure 5.13.

It also provides the values for the pitch (angle around the x-axis), yaw (angle around the y-axis), and roll (angle around the z-axis). These values are used to calculate the angle that the board has to move in order to comply with the movement of the hand.

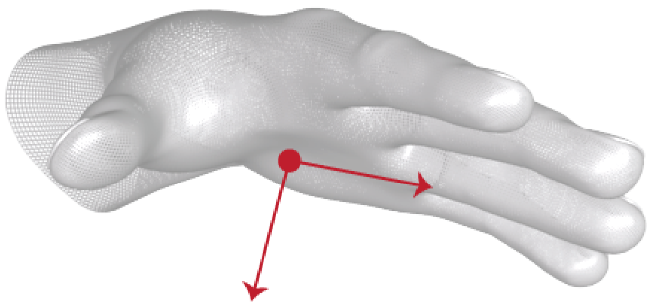


Figure 5.13: Vectors provided by the *Leap Motion* API

The board can not, however, mimic the player's hand completely if the player places their hand too vertically because this would cause the board to be in a vertical position too and that does not abide by the purpose of the game. To keep this from happening, the maximum angle that the board can move is 5 degrees in each direction.

## 5.2 Boids Band

The second game to be developed was a more complex game in which the player is resposible for guiding a flock of birds throughout their flight (subsection 4.5.1).

This section is divided into four subsections:

- **Game Description** explains how the game works.
- The **System Description** explains the process prior to the development of this game, that included adapting the general use cases, navigation diagram and mockups to this specific game.
- In **Development** some details about the development of the game in *Unity 3D* are further explained.

### 5.2.1 Game description

To clearly define the game, some major points had to be decided. These points were:

- The **components** of the game that are composed of objects needed to create the game.
- The **rules of the game** that make sense of the game and determine how the player wins or loses.
- How the **difficulty** of the game changes through time
- How the player **scores** in the game
- The **gestures** that were used in the game

In this subsection, these points are further explained.

#### ***Components***

The components of this game are:

- **The boids**: The player controls a set of boids that fly around.
- **The evil birds**: There is an evil species of birds that attack or fight the boids.
- **The food**: The food floats around waiting for the boids to eat it.
- **The buldings**: The obstacles that the boids must avoid.
- **The flock life bar**: The life bar that represents the flock's health.

#### ***Rules of the game***

This game has the following set of rules:

1. Your flock has a life bar. If it runs out you lose.
2. If you run out of boids in your flock, you also lose.
3. You can increase your life bar by eating food.
4. If you fly near enough to a boid that is not part of your flock, it will join your flock.
5. You lose boids when they go against a building.
6. If you find a bad bird you have two options: do nothing or challenge them in a fight.
7. If you fight a bird you have a 50% chance of winning the battle. If you win, you keep your boid and the bad bird disappears. If you lose, you lose the boid that fought it.
8. If you do not fight the bird and one of your boids goes against it, your flock's life bar decreases.

***Difficulty***

This is a one level only game, but the difficulty increases with time. The scenery is generated randomly around the boids.

Each time the player starts a new game, there is a counter that keeps count of how much time has elapsed. The more time it passes, the most difficult the game becomes.

In the beginning, a small amount of buildings and evil birds are generated in the space around the flock. There is also a lot of food available. Gradually, as the time passes, more and more buildings start appearing as well as evil birds. In contrast, the food starts to be scarcer.

***Scores***

In this game the player tries to keep their flock surviving as long as possible. Therefore, the best score had to be the highest time that the player achieved, since this represents the longest they could keep their flock alive.

***Gestures***

To guide the flock, the user must rotate their hand to the left or to the right, depending on which way they want the flock to move.

To attack an evil bird, the player must perform one of two pre-defined[2] *Leap Motion* gestures, the *ScreenTap* (figure 5.14) or the *KeyTap* (figure 5.15). The choice to have both gestures perform the same task was due to their high similarity. When testing the gestures, many times, the *Leap Motion* device detected the KeyTap gesture whenever the user was trying to do the other. Therefore, in the instructions, only one gesture (the ScreenTap) is taught to the player. However, when one of the two gestures is detected and there is an evil bird in the proximities, the flock is given the instruction to attack.

Initially, it was decided to have a gesture to make the boids eat. As it was observed that the gestures (even the *Leap Motion*'s pre-defined ones) are many times not recognized, and in order to keep the game simpler, it was decided to have the flock eat whenever it goes into food. Therefore, all the player has to do is drive the flock through food in order to eat and increase the life bar.

[2]More information available online at `developer.leapmotion.com/documentation/csharp/devguide/Leap_Overview.html#gestures`

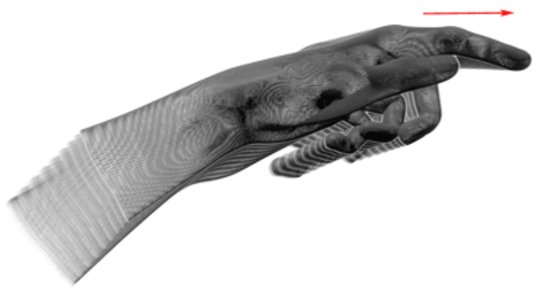


Figure 5.14: The *Leap Motion*'s ScreenTap gesture.

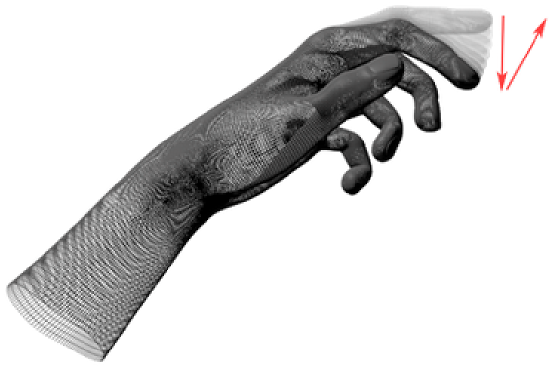


Figure 5.15: The *Leap Motion*'s KeyTap gesture.

### 5.2.2 System Description

Prior to the development of the Boids Band game, some of the methods used to describe the general system that would become each game were further specified. Therefore, a specific use case table for this game was built, such as specific mockups and a navigation diagram.

These documents supported the consequent development of the game. In this subsection the use cases table, the mockups and the navigation diagram are further explained.

***Use Cases***

To define the specific requirements for the Boids Band game, specific use cases were created. The following table (5.2) presents these use cases, their goals, pre and post conditions, main flow of events and secondary flow of events (when necessary).

Table 5.2: Boids Band Use Cases

| Name | Goal | Pre-conditions | Post-conditions | Main Flow of Events | Secundary Flow of Events |
|---|---|---|---|---|---|
| 1. Visualize high scores | Allow the player to check their high scores for a chosen level. | There are none. | There are none. | The **player** indicates their intention to see the high scores.<br><br>The **player** chooses the level for which they want to check the high scores.<br><br>The **game** presents the high scores.<br><br>The **player** checks the high scores. | If there are no high scores yet:<br><br>The **game** presents an empty high scores table. |
| 2. Change sound effects settings | Allow the player to increase or decrease the volume for the game's sound effects. | There are none. | The sound effects settings are saved. | The **player** indicates their intention to change the sound effects settings.<br><br>The **game** presents an item that allows the player to change the volume of the sound effects.<br><br>The **player** adapts the volume.<br><br>The **game** saves the changes. | |
| 3. Change music settings | Allow the player to increase or decrease the volume for the game's music. | There are none. | The music settings are saved. | The **player** indicates their intention to change the music settings.<br><br>The **game** presents an item that allows the player to change the volume of the music.<br><br>The **player** adapts the volume.<br><br>The **game** saves the changes. | |
| 4. Change player nickname | Allow the player to change its player nickname. | There are none. | The nickname is saved. | The **player** indicates their intention to change their nickname.<br><br>The **game** presents a text box that allows the player to change their nickname.<br><br>The **player** types in the new nickname.<br><br>The **game** saves the changes. | If any of the characters of the new nickname is invalid:<br><br>The **game** indicates to the user that some of the characters cannot be used and asks for a valid nickname.<br><br>The **player** inserts a valid nickname.<br><br>The **game** saves the new nickname if it is valid. Otherwise repeats. |
| 5. Visualize credits | Allow the players to see the game's credits. | There are none. | There are none. | The **player** indicates their intention to see the credits.<br><br>The **game** presents the credits.<br><br>The **player** sees the credits. | |
| 6. Visualize instructions | Allow the player to learn how to play the game. | There are none. | There are none. | The **player** indicates their intention to visualize the instructions.<br><br>The **game** presents the instructions.<br><br>The **player** sees the instructions. | |
| 7. Play | Allow the player to play the game. | There are none. | The game is playing. | The **player** indicates their intention to start playing.<br><br>The **game** starts.<br><br>The **player** uses gestures to play.<br><br>The **game** alters its state according to those gestures. | If the game is lost:<br><br>The **game** saves the player's score, if it is a new high score, and resumes to the "win" state. |

***Navigation Diagram***

In order to represent the flow between the different parts of the game, a specific navigation diagram was designed (figure 5.16). This diagram represents the flow that occurs between the different parts of the game.

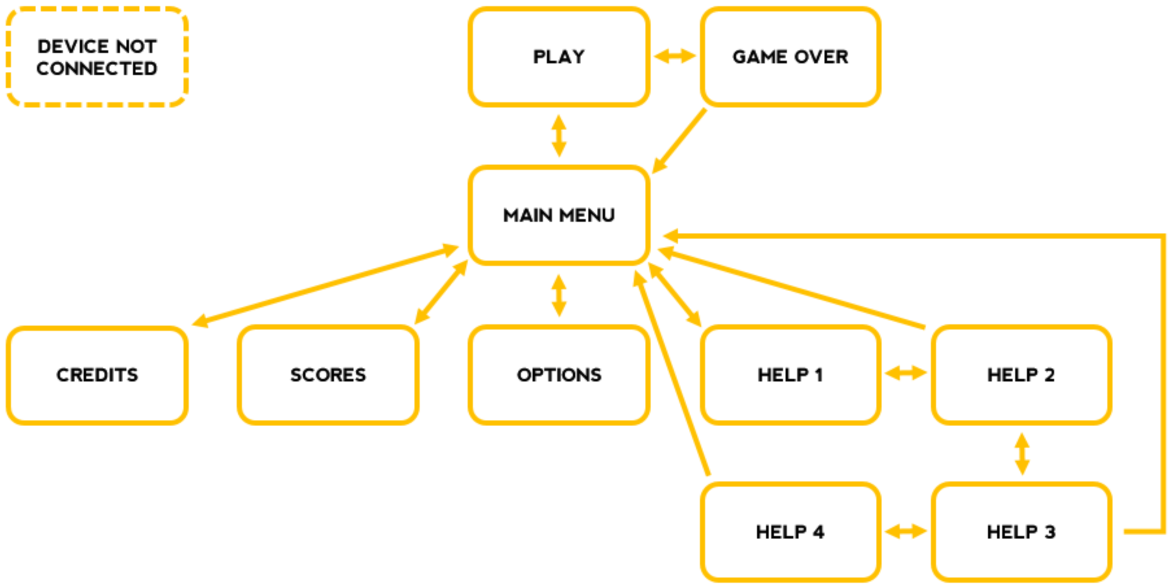


Figure 5.16: Navigation diagram for the Boids Band game

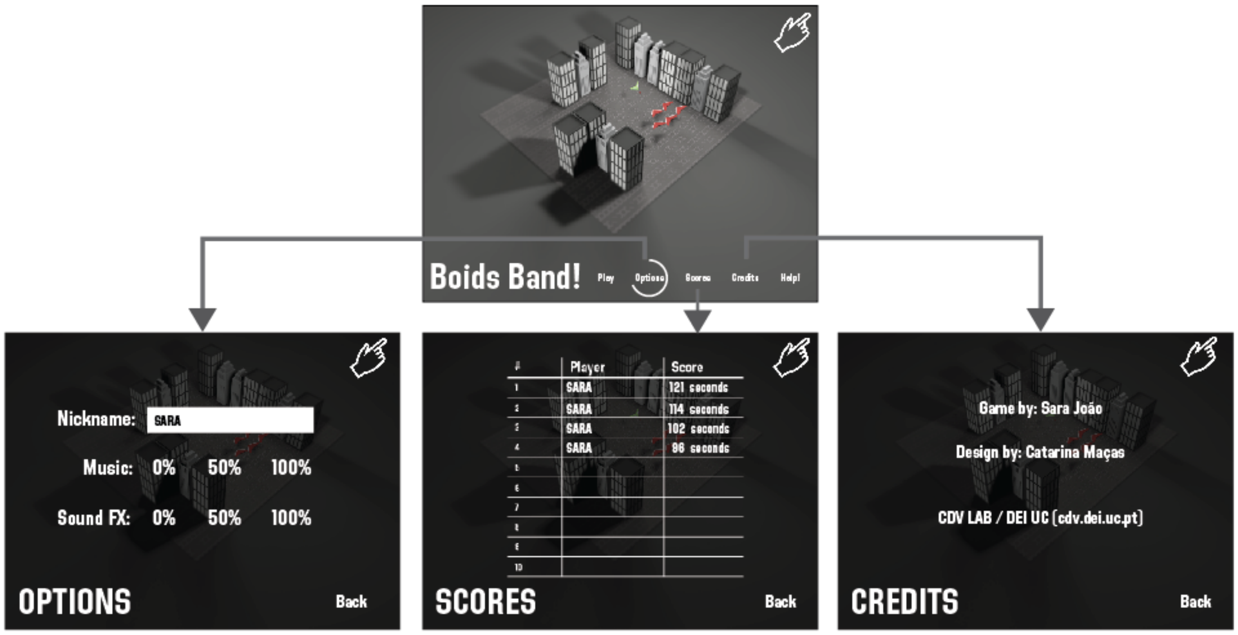
The main menu allows the user to chose from a number of different actions in the game that are described by the previously defined use cases (section 5.2).

The diagram is very similar to the previously designed generic navigation diagram (section 3.1) with a single difference: the Help state is divided into four different states. This happened due to the complexity of the game, that demanded for more complex instructions. When choosing the Help state the user can navigate through the different parts of the instructions.

***Mockups***

In order to define the disposition of the windows of the Boids Band Game, several mockups were designed.

As in the previous game, the design of the objects that were used in the game were produced by Catarina Maçãs, a researcher of the CDV Lab. This design included the choice of the typography and colors, and the creation of the 3D and 2D objects that were used in the game. The disposition of the elements on the screen also included the designer's inputs.



Nickname: SARA

Music: 0% 50% 100%

Sound FX: 0% 50% 100%

| # | Player | Score |
|---|---|---|
| 1 | SARA | 121 seconds |
| 2 | SARA | 114 seconds |
| 3 | SARA | 102 seconds |
| 4 | SARA | 96 seconds |
| 5 | | |
| 6 | | |
| 7 | | |
| 8 | | |
| 9 | | |
| 10 | | |

Game by: Sara João

Design by: Catarina Maças

CDV LAB / DEI UC (cdv.dei.uc.pt)

Figure 5.17: Mockups for the Boids Band: Main Menu, Options, Scores and Credits.

Figure 5.17 represents the first set of mockups. These exemplify how the Main Menu will look like as well as some of the states that can be accessed through that menu.

- The **Main Menu** presents all the possible actions for the player to take: Play, change settings, check high scores, visualize the credits and visualize the instructions.
- The **Options** state allows the player to choose a nickname and change the volume of the sound effects and the game's music.
- The **Credits** state allows the player to visualize details on the creators of the game.
- The **Scores** state allows the player to check their high scores.
- The **Help** state provides information to the player on how to play the game.

Figure 5.18 represents the mockups for the **Play** action. The player starts by choosing to play in the Main Menu and is redirected to the game. When the player loses they are redirected to the **Game Over** state that indicates their score.

Figure 5.19 represents the mockups for the instructions screens. Being more complex than the last game, the instructions had to be extended to four different screens in order to explain all the rules to the player.

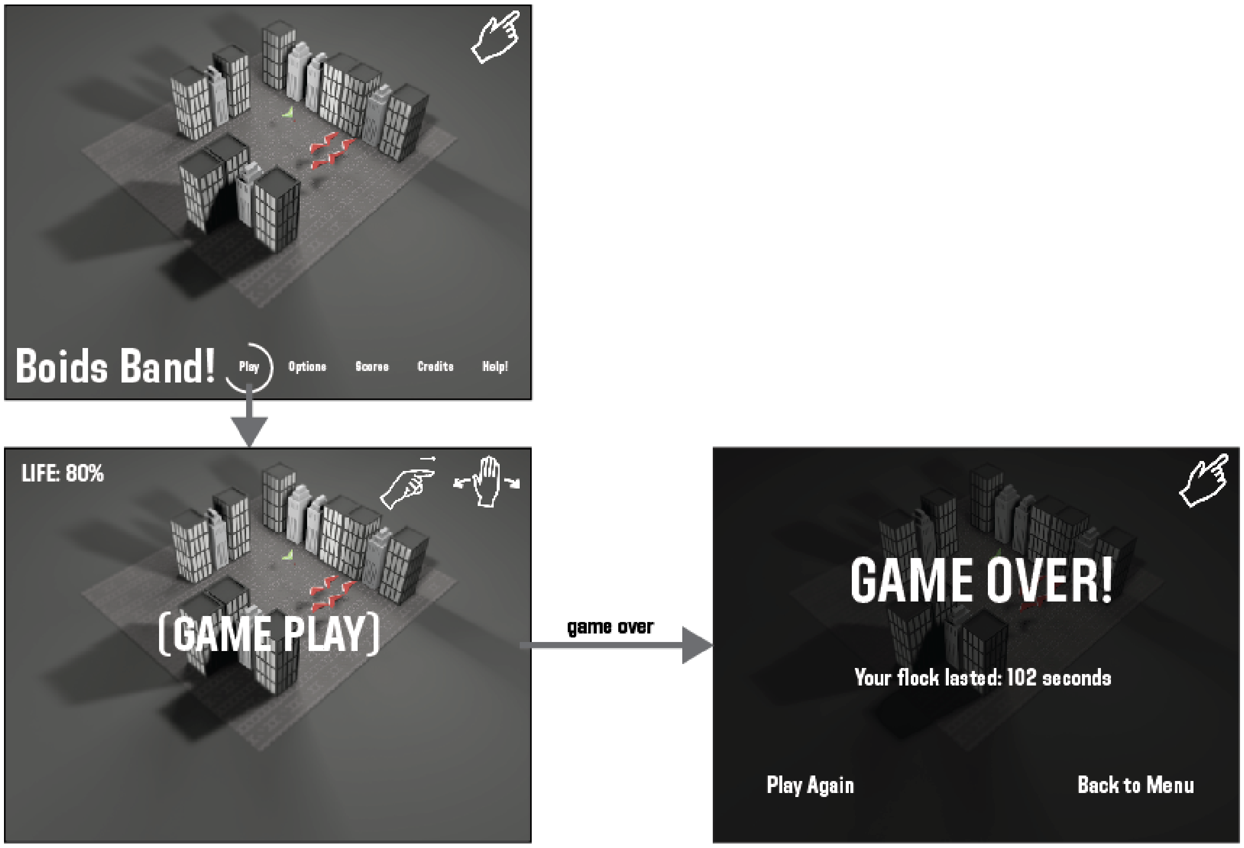
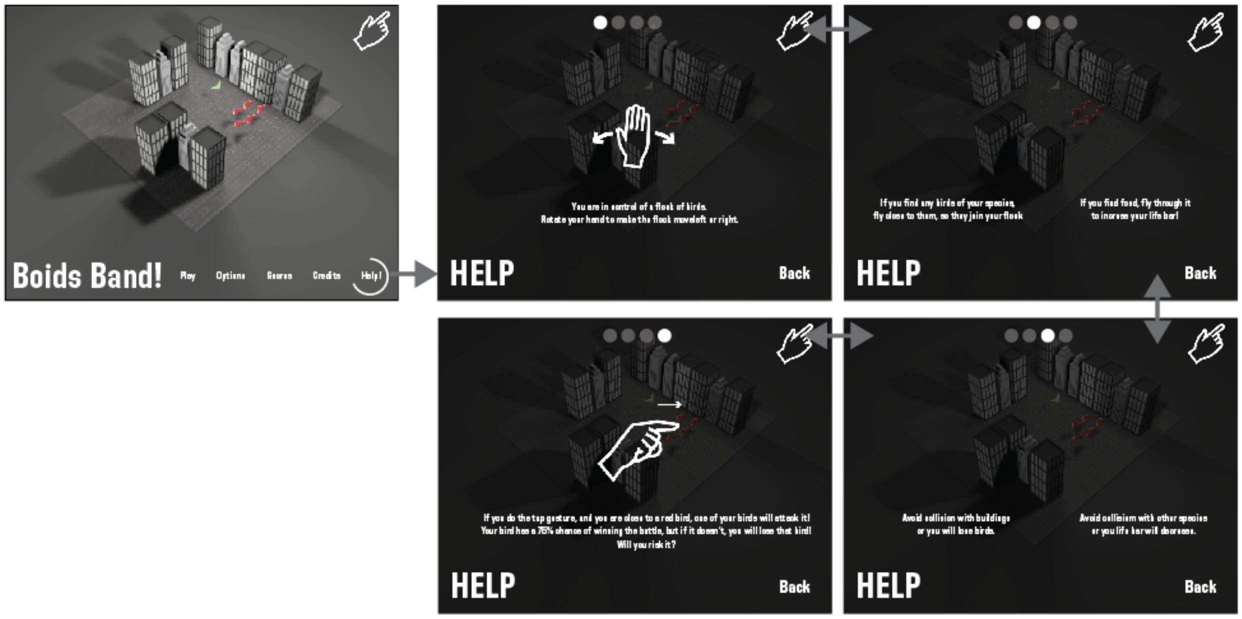


Figure 5.18: Mockups for the Boids Band: Game Play and Game Over states.



You are in control of a flock of birds.
Rotate your hand to make the flock move left or right.

If you find any birds of your species, fly close to them, so they join your flock

If you find food, fly through it to increase your life bar!

Avoid collision with buildings or you will lose birds.

Avoid collision with other species or you life bar will decrease.

If you do the tap gesture, and you are close to a red bird, one of your birds will attack it! Your bird has a 75% chance of winning the battle, but if it doesn't, you will lose that bird! Will you risk it?

Figure 5.19: Mockups for the Boids Band: Instructions.

Finally, figure 5.20 represents the mockup for the **Device not Connected** state, that appears when the Leap Motion device is not connected.

The main reasons for the choices that were made, related to the disposition of the objects on screen and the way the user interacts with them are further explained on section (5.3).

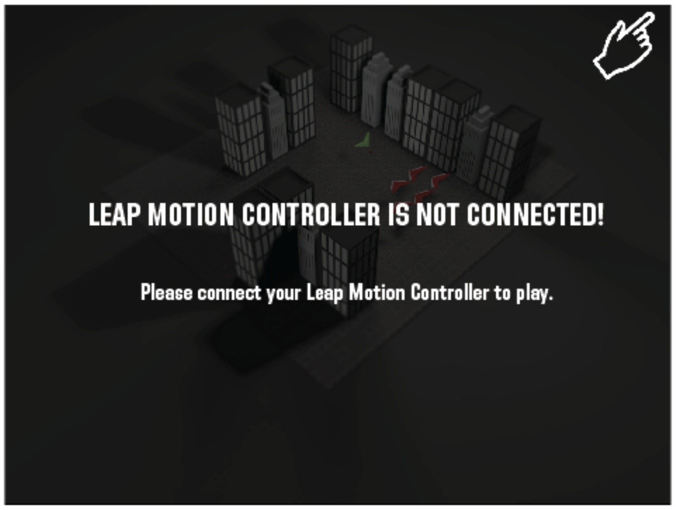

LEAP MOTION CONTROLLER IS NOT CONNECTED!
Please connect your Leap Motion Controller to play.

Figure 5.20: Mockups for the Boids Band: Device disconnected state.

### 5.2.3 Development

***Flocking behaviour: Boids***

The birds on this game simulate the flocking behaviour of real birds. Flocking behaviour is the name given to the behaviour exhibited by a group of birds in flight. Flocking behaviour was simulated for the first time on a computer in 1986 by Craig Reynolds[83]. His simulation program was called Boids and simulated a set of agents (called boids) that move according to a set of basic rules. [84][85][86]

The basic set of rules that were used were as followed:

- **Separation**: Steer to avoid crowding local flock-mates
- **Cohesion**: Steer to move toward the average position of local flock-mates
- **Alignment**: Steer towards the average heading of local flock-mates

More complex rules can be added to this set to make the boids more complex, such as rules to avoid obstacles and seek goals. [86][87]

This game's boids are based on the three basic rules: separation, to make sure the boids do not crowd each other; cohesion, to ensure the boids move as a group and not as separate individuals; and alignment, to ensure the boids move in the direction in which most of the band is heading. [88][89]

Initially four boids are instantiated in the middle of the screen. This is a reasonable number of boids for the user to start with, since they do not crowd the game too much and yet, are enough for the player to make further choices on how to evolve the game.

In order to ensure that the boids act as a flock, some parameters had to be included and some experimentation was made with different values to define which were the best for such parameters. Some of these parameters are:

- **Flock Radius**: Represents the radius of the flock. The boids are supposed to stay inside this radius.
- **Speed**: Controls the speed of the boids.
- **Neighbor Radius**: If a boid is within the neighbor radius of another boid, then it is part of the flock.
- **Separation**: The minimum distance between boids.

It was also necessary to define the importance that each rule has compared with one another:

- **Separation Weight**: The weight of the importance to keep the agents from colliding into each other.
- **Cohesion Weight**: The weight of the importance to keep the agents from drifting apart from each other.
- **Alignment Weight**: The weight of the importance to keep the agents moving towards the same direction.

After experimenting different values for each weight, it became clear that the cohesion weight had to be heavier than the other two, otherwise the boids start drifting apart and it becomes difficult to keep the flock together. Therefore, this parameter was defined as 1.5 times heavier than the other two.

***Attack!***
Whenever the flock passes nearby an evil bird, the player can either try to deviate from it or choose to fight it.

If the player chooses to fight, all they have to do is command it by doing the screentap gesture. At that moment the bird in the flock that is nearest to the evil bird goes to fight it. The user can either lose the battle and consequently lose the bird that went on the fight, or win the battle, in which case, the evil bird disappears.

This is decided through a pseudo random number that is calculated every time the player decides to fight. And each bird has a 50% chance of winning the battle. This number was chosen so that the player feels like they have a fair chance of winning the battle and is willing to risk a bird for the chance of an evil bird disappearing.

## 5.3 Usability Choices

Since Natural User Interfaces are the main subject of this project, the produced games had to follow the Natural User Interfaces guidelines established in subsection 2.2.2. Therefore, an attempt was made in order to produce games that follow the main guidelines:

- The interface should consider its context.
- The interaction between the user and the machine should be direct (there should be no intermediaries).
- The experiences should mimic real world interactions.
- The interactions should be enjoyed by the user.
- The skills must be easy to learn and should use pre-existing knowledge.

Some decisions had to be made in order to follow these guidelines, and whilst some of them were planned from the beginning (since the planning phase), others were made merely during the development and testing phases. In this subsection some of this decisions are explained.

During the testing phase, some questions were asked the participants to find out if the guidelines had been successfully followed. This is further explained in chapter 6.

***The interface should consider its context***
In this case, the context will mainly be the player's personal computer at home with an attached *Leap Motion* device. The games should, therefore, use similar symbolism to the one used in other *Leap Motion* applications.

An example of a decision that had to be made regarding context was how to create menus and respective transitions. How would the player indicate that they intended to proceed with a certain action? After examining many other applications for the *Leap Motion*, and some opinions in the *Leap Motion* developer forum[3], it became clear that there were a few main possibilities [90].

The first possibility was to use the concept of the Z axis to create a "touch" button. This allows the user to "click" on a button. The button is being pressed when the user's finger has passed a given threshold in the Z axis. Despite being very easy to implement, it is the least popular solution since it tries to mimic GUI interfaces and people seem to end up confused on how to interact [90].

Another solution would be to use gestures. The gestures could be created specifically for the game or the pre-defined gestures that the *Leap Motion* API provides could be used. Figure 5.14 shows a gesture that would serve the purpose: the screentap gesture. This solution is easy to implement but it involves giving prior knowledge to the user. A user that never used an application with the defined gesture will wander around the screen pointing not knowing what to do.

A third option was to use a timer. This was the most user-friendly option since it does not involve the user learning new skills. When the user sees a word or icon that interests them on screen they will logically point to them. At that time, a timer (in the form of a bar) will appear and most users will assume that if it reaches completion they will enter a new state of the game.

***The interaction between the user and the machine should be direct***
These guideline is already accomplished simply by using the *Leap Motion* device, since it allows the user to manipulate the objects on screen by using their bare hands.

***The experiences should mimic real world interactions.***
In the Tilt Game, the interaction of the user with the board of the game is meant to mimic a real world interaction. To understand how this interaction would be, a question was asked: "If a person is given a board with a ball, how would they move their hands in order to keep the balance of the board and keep the ball from falling?" The most common result of this experiment would be the

[3]More information available on `developer.leapmotion.com/forums`

person using both hands for this task and tilt the board in the opposite direction than the one the ball is currently falling towards, to keep it from falling from the board.

However, to use both hands in the game would bring complications since the *Leap Motion* device has more difficulty perceiving the gestures from two hands simultaneously than just one, as was previously explained in subsection 2.3.3. The use of both hands in the game was discarded and a solution for using only one hand was sought.

The same question was asked but this time with a limitation: "What if the person could only use one hand?". The most common answer to this question would probably be that the person should place their hand below the center of the board and slightly tilt it in the opposite direction than the one the ball is currently falling towards. This would mean that the person's hand and the board would act as a single object and always move in parallel.

This idea was transposed to the game by having the board mimic the player's hand position. If the player tilts their hand to the left, the board will also tilt to the left, if the player tilts their hand forward the board will also tilt forward and so on. This way the real world interaction was mimicked into the game.

***The interactions should be enjoyed by the user***

In order to have the player really enjoying the game, an attempt was made in both games to avoid boring the user with unnecessary actions. The games were created as simple as could be and some expected actions were deleted.

For example, in the Tilt Game, it would be expectable for a "Game Over" screen to appear when the player loses the game. However, this game is very quick and the player can either win or lose in a matter of seconds (especially lose). It would be boring to have the player go to a "Game Over" state whenever he lost and have to do an action to start playing again. To avoid this, the state was not included and the game simply restarts with a new generated map, whenever the player loses.

The object disposition in the menus also try to simplify to a maximum the game experience. For example in the main menu, all the possible actions that the player can take, are set side by side on a list of objects as can be seen in figure 5.21.

***The skills must be easy to learn and should use pre-existing knowledge***

As mentioned before in one of the previous points, the menus were built in a similar way to other

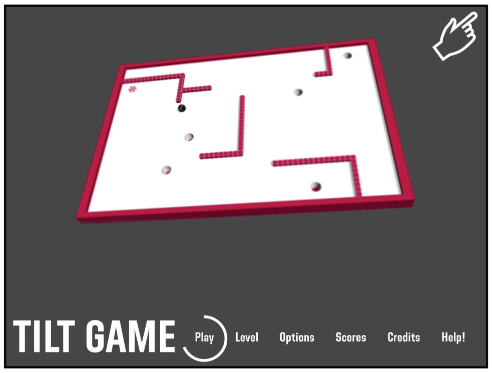


Figure 5.21: Main Menu for the Tilt Game

menus from the *Leap Motion*'s store's other applications and games. Not only that, they were built in a way that tries to avoid the need for the user to learn any new skills to interact with the device. This is thought to help reduce the amount of learning the player must do in order to play the game.

Another example of how this guideline was followed was by the insertion in every screen, of an image with the gesture that the user is supposed to be using on that part os the game in order to interact correctly with the game. This image was inserted in the right upper corner of the screen, as can be seen in image 5.21.

This helps the user understand what gesture must be done at that time and place, reducing the amount of experimentation that the user would have to do, to understand which gesture to use by themselves.

***Other usability issues***

The Tilt Game has four difficulty levels that the player can chose from: Easy, Medium, Hard and Impossible. Initially, the objects that allowed the user to chose between levels were planned to be inside the options menu. However, changing levels seemed to need more relevance since it was something far more important than changing the game's volume or the player's nickname. It was therefore decided that there should be a new menu dedicated solely to the choice of the difficulty level.

Also in the Tilt Game, an option that allows the user to choose the angle in which they want to visualize the board was included to the Options menu. This was inserted after some tests were made with different people. Some people complained when the board was seen from straight up (90 degress) whilst others complained when the viewing angle was lower (60, or 75 degrees). To allow the player to chose what they consider to be the best way to play, that option was included in the game.

In the Boids Band game some usability choices also had to be made regarding what gestures to use to attack. As was mentioned earlier, it was decided to keep two of Leap Motion's most similar gestures, since many time the users were trying to use one and having the other one detected.

To represent the weakness in the birds, as the life bar decreases, the color of the birds starts to vanish. This is more intuitive to the user than checking the state of the life bar.

Also in Boids Band it was decided to remove the gesture for catching food since it added unneeded complexity to the game.

## 5.4 Games Publishing

Since the games were meant to be developed in order to be viable to be uploaded to the Airspace store, precautions had to be made to ensure that the games would follow the store's rules and could be accepted in it.

Another issue lied with the need for the games to be published by the University of Coimbra, since they where developed by researchers of the University.

Both these issues are explained in the next subsections. At the time of writing this report, both games are waiting to be accepted on the Airspace store.

### 5.4.1 *Leap Motion* and the Airspace

The Airspace has a very strict set of rules[4] to accept applications and games. For the games to be accepted in the store, this set of rules were taken into consideration since the beginning of the development.

The most challenging of these rules was "Section 2: Functionality and Compliance". This section presents many rules about performance, stability, functionality and compliance standards and

[4]More information available online at `developer.leapmotion.com/apps/guidelines/review-guidelines`

user experience.

The User Experience subsection has very specific details on how the games should behave given a certain input. Some of the most important rules that were taken into account when developing the games were:

- Create a finished product. No missing audio or other assets, no placeholders, no debug tools/text or Prototype/Alpha/Beta phases.
- Welcome and engage the user with a splash screen and menus.
- Educate and enlighten the user with clear instructions, tutorials or walkthroughs.
- Smoothly transition users into the core experience.
- Music and sound effects from the app should mute when the app is minimized or not in focus, and unmute when focus is regained.
- Gracefully allow the user to pause, resume and/or exit. Users should be instructed how to exit the App either by keyboard press (i.e. Esc), gesture and/or a selectable Exit button within a menu.
- During loading or screen transitions, any transitions that exceeds 5 seconds should display a loading/progress bar and/or other indicator.
- Create straightforward and intuitive experiences. Controls and navigation should feel organic and logical. Providing a tutorial option on first boot or from a Menu selection will satisfy this guideline. Menu icons should be intuitive or have accompanying text.
- Grab the user's attention with polished and visually appealing design.
- Leverage the precision, accuracy and speed of the *Leap Motion* controller.
- Take full advantage of all 3 axes (X, Y and Z) where relevant.
- Utilize the most recently released APIs provided by *Leap Motion* to ensure the most up-to-date functionality.
- Account for natural and comfortable human interaction when designing the experience. Consider extended play, repeated actions and effort required.

These, and many other rules were taken into account to create games that would later be accepted by the Airspace store.

### 5.4.2 Publishing through the University

To publish the games in the Airspace store for sale, it was necessary to follow the procedures of an external entity: DITS[5], a division of the University of Coimbra. This division is responsible for handling the work originated from the University's labs.

Any project that is created with commercial purposes by researchers of the University, must be published by the University and not by the author. The University than keeps 30% of the gains and distributes the rest by the project's inventors.

In the beginning of the second semester, contact was made with this division to find how to proceed in order to publish the games.

After many interactions with the division, it was decided that to avoid delays and complications, the only option was to publish both games as free, since this would prevent many problems that would have to be solved. However, this solution was reached to too late (in late July) which led to a consequent delay on the publishing of the games. At the time of writing this report neither of the games have yet been accepted by the Airspace store.

[5]More information available online at `www.uc.pt/gats`

# 6. Usability Evaluation

Natural User Interfaces are the main subject of this project, and it was very important that the produced games follow the NUI guidelines that were established in subsection 2.2.2.

To ensure that the guidelines had been successfully followed, an usability evaluation was conducted.

This chapter begins by presenting some of the existing evaluation methods and which of these methods were chosen. It then introduces the results of the evaluations performed to the developed games as well as some important remarks made by the testers.

## 6.1 Evaluation Methods [1]

An evaluation's main goal is to ensure that the final product is as convergent as possible with the goals that were defined for that product. Evaluations allow a deeper understanding of the context in which the application will be used.

To evaluate usability, it is necessary to sometimes contemplate factors as subjective as satisfaction that may be relevant for the case in study.

An usability evaluation implies:

- Projecting the experience.
- Selecting the participants (it is important to have a variety of user groups that represent the end user, using factors like age, genre, etc.).
- Preparing the tasks that the participants will be asked to do (not only the tasks must be listed, but also a script must be created for the monitor of the experience to behave in the exact same way with every participant).
- Preparing all the needed materials.

- Conducting the experience (it is recommended to do at least five tests for each group of users, which will allow the identification of around 75% of the usability problems [91]).
- Collecting and analyzing data.
- Reporting the results.

If the evaluation involves asking the users to give a score to pre-defined questions, it is important that the given scale ranges from one to an even number. This keeps the user from choosing the safe number (the one in the middle) and having to decide to go for the "I agree completely" or the "I disagree completely" side. It is also important that the range of numbers is not too wide to keep the user more objective. A scale from one to six is usually adequate.

A lot of techniques are available to evaluate applications and games. This section presents some of those techniques.

***Quick and Dirty***

This technique consists on seizing opportunities to evaluate design options with users or experts.

The application is given to the users to experiment. The monitor of the study simply observes while the user interacts, taking notes of the main comments and actions.

This method dos not require a detailed planning phase which makes the evaluations quick and opportune. However its results can vary a lot from user to user since they are not focused on particular issues os tasks.

***Heuristic Evaluation***

A number of heuristics are defined to evaluate the application. Some authors have established a set of heuristics as is the case of the ten heuristics presented by Nielsen in is book *Usability Engineering* [92].

Heuristics are usually presented to the user in the form of a sentence which the user is asked to give a score to, in a given scale, after they have used the application to perform certain tasks.

This technique allows for obtaining indexes that may be relevant to tune the performance of the application.

Overall, this is a simple and relatively quick evaluation technique.

***Design Walkthrough***

Design Walkthroughs are simulations of a process. These simulations are performed by a user and are usually done before the application is developed.

In order to use this technique it is necessary to prepare a prototype of the application and select a set of tasks for validation. Then, the user is asked to perform the tasks step-by-step preferably while thinking-aloud. During this performance, the monitor takes notes for later analysis.

This is a more complex evaluation since it involves a lot of previous work. It is also done before development instead of after, which causes it not to evaluate a final application but a prototype.

***Formal Lab Test***

The Formal Lab Tests intend to be a more rigorous way to evaluate an application. They try to isolate the user from distractions or interruptions, but by doing so, it also loses the context behind the application that is essential to correctly evaluate it (specially in a NUI application that must be aware of context, as is referred in the established guidelines (2.2.2).

In this evaluation, besides the notes that are taken by the monitor, the user is recorded for posterior analysis.

***Field Evaluation***

The application is experienced by end users in its end environment. The users are asked to perform certain tasks and later answer some questions.

This technique is complex since it involves many previous work. It is also difficult to manage to create a final environment. However, if done correctly it can be a very useful technique to understand if the application is working properly.

## 6.2 Chosen Methods

To evaluate the developed games, two evaluation techniques were chosen: Heuristic Evaluation and Field Evaluation. The reason for choosing these two was their simplicity and relevance.

An **Heuristic Evaluation** is very simple to prepare and is useful to understand which aspects of the games should be improved. The set of heuristics that were used in this project, were the ones defined by Nielsen, in his book Usability Engineering [92]. These heuristics cover most aspects common to a game and are as follows:

- Simple and natural dialogue
- Speak the users's language
- Minimize user memory load
- Consistency
- Feedback
- Clearly marked exits
- Shortcuts
- Good error messages
- Prevent errors
- Help and documentation

The **Field Evaluation** is one of the best evaluation techniques since it allows the evaluation of the application in its real environment.

A script was created with tasks for each game, as well as a set of sentences for the user to evaluate. The evaluation monitor must always follow the script completely so that every user has the same information. There were 17 sentences for the user to give a score to (seven that refer to the NUI guidelines, plus ten that correspond to the Nielsen heuristics. The scripts (appendix A), the questionnaire (appendix B) and a table with the answers of the participants (appendix C) are available in the appendix of this report.

The users were asked to give each sentence of the questionnaire a score from 1 to 6 which is, as mentioned in the previous section, an adequate scale since it is even and not very wide ranged.

It was decided to use four age groups and have five tests in each group, which, as mentioned in the previous section, is enough to discover up to 75% of the errors [91]. After some study on the subject, it was decided to use the following age groups, that appear to be the most common with smaller samples [93][94]:

- 12 - 14
- 15 - 24
- 25 - 44
- 45 - 65

For this evaluation 20 testers were selected, 5 for each age group. All the testers participated in the evaluation of both games.

## 6.3 Tilt Game Evaluation

The next subsections present the results of the two evaluation techniques that were used to evaluate the game.

The testers were given a Laptop Computer with the Leap Motion device and were asked to perform several tasks. The appendix A shows the script that was used by the monitor of the conducted evaluations. Figures 6.1 and 6.2 shows one of the testers playing the game.

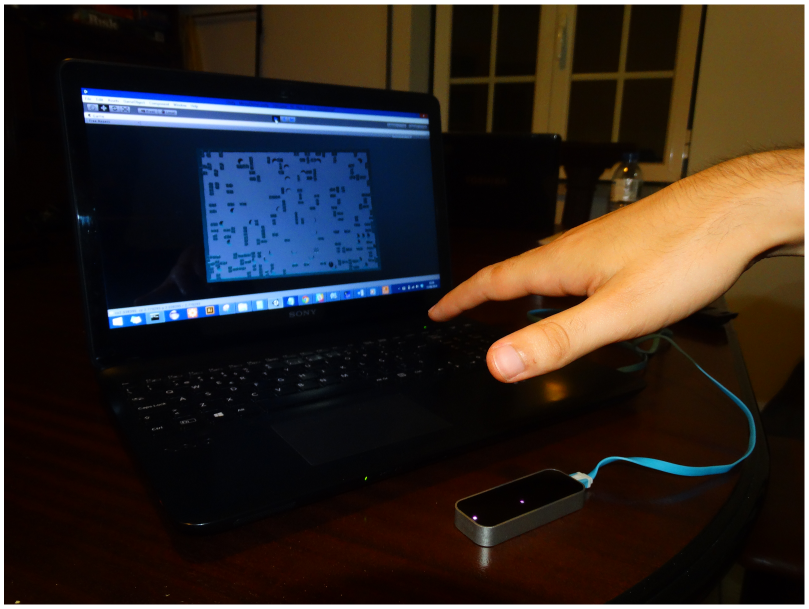


Figure 6.1: Evaluation of the Tilt Game: Playing

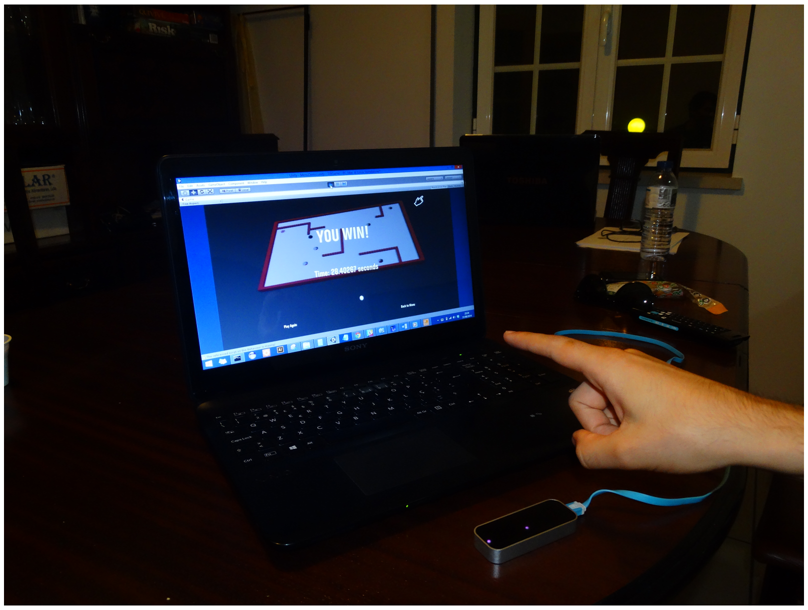


Figure 6.2: Evaluation of the Tilt Game: Game Over

### 6.3.1 Field Evaluation

The testers were given a computer with the *Leap Motion* device connected and were asked to follow a set of tasks. In the end, they were asked to answer some questions related to what they had experienced. The first seven questions that were asked had to do with the established natural user interfaces guidelines 2.2.2. On all sentences, the testers were asked to evaluate the sentence with a number from 1 to 6, 1 meaning "I completely disagree" and 6 meaning "I completely agree".

***The interface should consider its context***

For this guideline, two sentences were presented to the testers:

1. "I find the menus to be well constructed."
2. "The game is adequate to the *Leap Motion* device."

The first sentence intends to perceive if the testers find the menus appropriate while the second sentence intends to find out if the testers found the game adequate to the device.

*"I find the menus to be well constructed"*

In this sentence the global mean of the answers was 5.0. The testers that belong to the youngest age groups seem to find that the menus were better constructed than the oldest testers. Table 6.1 shows the various means for each age group, while figure 6.3 shows how many testers chose each score.

| Age Group | Mean | Deviation |
|---|---|---|
| 12 - 14 | 5.60 | 0.55 |
| 15 - 24 | 5.60 | 0.55 |
| 25 - 44 | 4.20 | 0.45 |
| 45 - 65 | 4.60 | 1.14 |
| Total | 5.00 | 0.92 |

Table 6.1: "I find the menus to be well constructed": Tilt Game Means

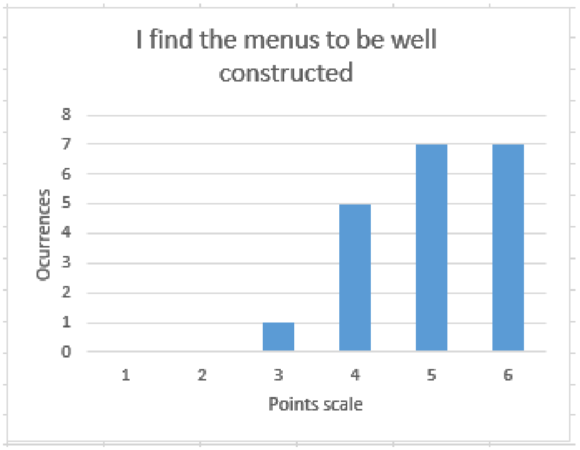


Figure 6.3: "I find the menus to be well constructed": Tilt Game scores

*"The game is adequate to the Leap Motion device"*

In this sentence the global mean of the answers was better: 5.3. Table 6.2 shows the various means for each age group, while figure 6.4 shows how many testers chose each score.

| Age Group | Mean | Deviation |
|---|---|---|
| 12 - 14 | 5.40 | 0.89 |
| 15 - 24 | 5.60 | 0.55 |
| 25 - 44 | 5.00 | 0.71 |
| 45 - 65 | 5.20 | 0.45 |
| Total | 5.30 | 0.66 |

Table 6.2: "The game is adequate to the *Leap Motion* device": Tilt Game Means

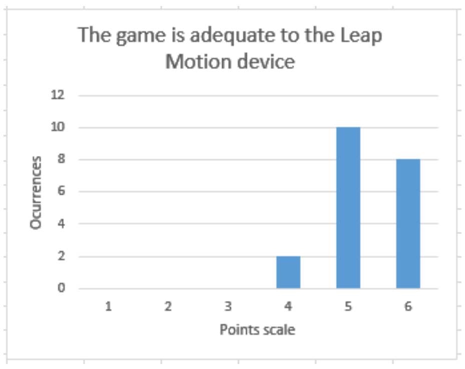


Figure 6.4: "The game is adequate to the *Leap Motion* device": Tilt Game scores

***The interaction between the user and the machine should be direct***

The sentence that was used used to evaluate this guideline was: *"I consider the interaction to be similar to the one I would have when using a physical board"*. In this sentence the global mean of the answers was 4.55. The means show that the youngest age groups consider the interaction to be more direct than the oldest age groups. Table 6.3 shows the various means for each age group, while figure 6.5 shows how many testers chose each score.

| Age Group | Mean | Deviation |
|---|---|---|
| 12 - 14 | 4.80 | 0.84 |
| 15 - 24 | 4.60 | 1.14 |
| 25 - 44 | 4.60 | 0.55 |
| 45 - 65 | 4.20 | 0.84 |
| Total | 4.55 | 0.83 |

Table 6.3: "I consider the interaction to be similar to the one I would have when using a physical board": Tilt Game Means

***The experiences should mimic real world interactions.***

The sentence that was used used to evaluate this guideline was: *"I consider that I could interact directly with the game objects"*. In this sentence the global mean of the answers was 4.8. Table 6.4 shows the various means for each age group, while figure 6.6 shows how many testers chose each score.

***The interactions should be enjoyed by the user***

The sentence that was used to evaluate this guideline was: *"I enjoyed playing this game"*.

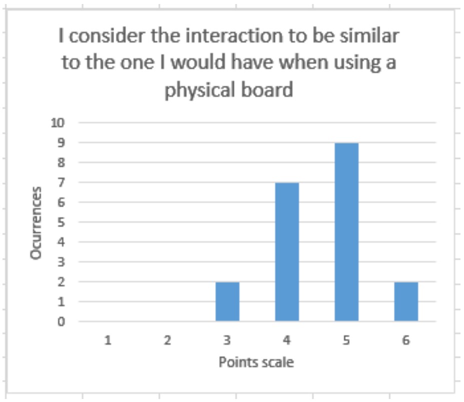


Figure 6.5: "I consider the interaction to be similar to the one I would have when using a physical board": Tilt Game scores

| Age Group | Mean | Deviation |
|---|---|---|
| 12 - 14 | 5.00 | 1.52 |
| 15 - 24 | 4.80 | 1.44 |
| 25 - 44 | 4.80 | 0.84 |
| 45 - 65 | 4.60 | 1.30 |
| Total | 4.80 | 1.15 |

Table 6.4: "I consider that I could interact directly with the game objects": Tilt Game Means

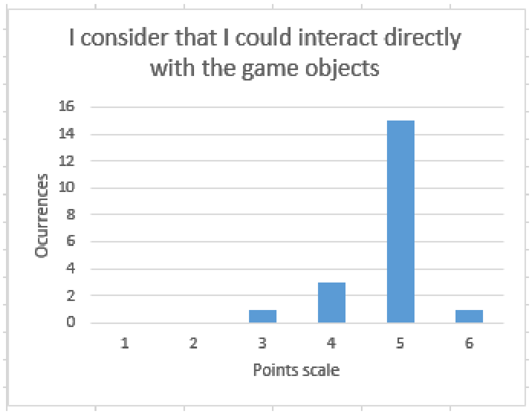


Figure 6.6: "I consider that I could interact directly with the game objects": Tilt Game Scores

In this sentence the global mean of the answers was one of the best: 5.15. Table 6.5 shows the various means for each age group, while figure 6.7 shows how many testers chose each score.

| Age Group | Mean | Deviation |
|---|---|---|
| 12 - 14 | 5.80 | 0.45 |
| 15 - 24 | 5.60 | 0.55 |
| 25 - 44 | 4.80 | 0.45 |
| 45 - 65 | 4.40 | 0.55 |
| Total | 5.15 | 0.72 |

Table 6.5: "I enjoyed playing this game": Tilt Game Means

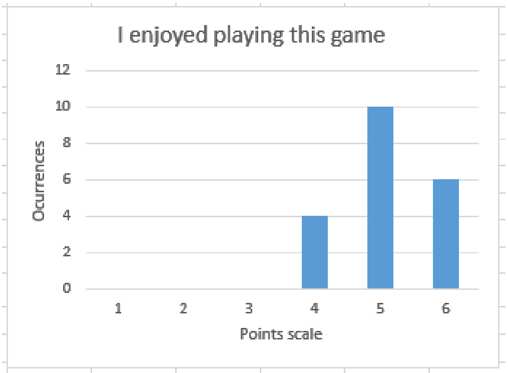


Figure 6.7: "I enjoyed playing this game": Tilt Game scores

***The skills must be easy to learn and should use pre-existing knowledge***

For this guideline, two sentences were presented to the testers:

1. "I did not need to learn new skills to play this game."

2. "I managed to easily interact with the game."

The first sentence intends to perceive if the testers found the need to learn new skills to be able to play the game while the second sentence intends to find out if the testers managed to interact easily with the game.

*"I did not need to learn new skills to play this game"*

In this sentence the global mean of the answers was 4.5. Despite being a positive ranking, it clearly shows that some users felt the need to learn new skills to be able to play. This is even more evident in the oldest age group that has a mean of 3.8. Table 6.6 shows the various means for each age group, while figure 6.8 shows how many testers chose each score.

| Age Group | Mean | Deviation |
|---|---|---|
| 12 - 14 | 4.20 | 1.92 |
| 15 - 24 | 5.60 | 0.55 |
| 25 - 44 | 4.40 | 1.14 |
| 45 - 65 | 3.80 | 1.64 |
| Total | 4.50 | 1.47 |

Table 6.6: "I did not need to learn new skills to play this game": Tilt Game Means

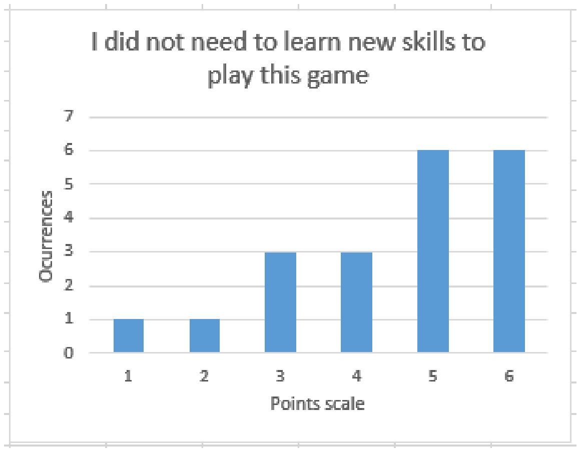


Figure 6.8: "I did not need to learn new skills to play this game": Tilt Game scores

*"I managed to easily interact with the game"*

In this sentence the global mean of the answers was similar: 4.2, which shows that the users had some difficulty interacting with the game. Table 6.7 shows the various means for each age group, while figure 6.9 shows how many testers chose each score.

| Age Group | Mean | Deviation |
|---|---|---|
| 12 - 14 | 4.40 | 0.00 |
| 15 - 24 | 4.40 | 0.84 |
| 25 - 44 | 4.20 | 0.45 |
| 45 - 65 | 3.80 | 0.89 |
| Total | 4.20 | 0.62 |

Table 6.7: "I managed to easily interact with the game": Tilt Game Means

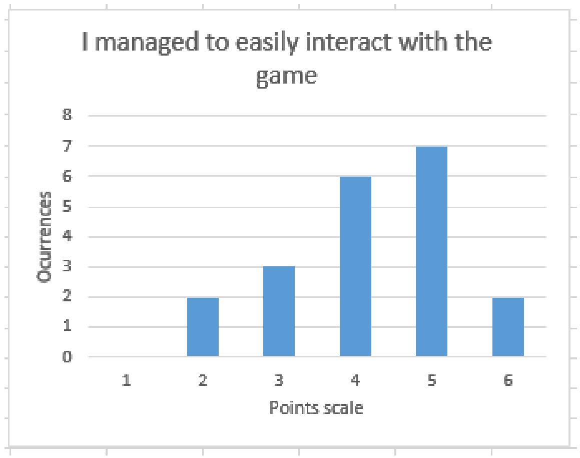


Figure 6.9: "I managed to easily interact with the game": Tilt Game scores

***Results***

Overall, all the guidelines had satisfactory results, all on the positive side of the scale. Young users seem to be able to interact better with the game in almost every aspect.

The guidelines that were most successfully followed were:

- The interface should consider its context
- The interactions should be enjoyed by the user

The one guideline that should be improved is "The skills must be easy to learn and should use pre-existing knowledge".

### 6.3.2 Heuristic Evaluation

In the same questionnaire, the testers were also asked to score 10 sentences matching the ten heuristics defined by Nielsen [92]. The full sentences that were presented to the user can be seen in the appendix B. The testers were asked to evaluate the sentence with a number from 1 to 6, 1 meaning "I completely disagree" and 6 meaning "I completely agree". These evaluation resulted in a set of indexes of which aspects are more prominent to improve in the game.

Overall the results were positive. One of the worst results goes to the heuristic "Prevent Errors"

| Heuristic | 12 - 14 Mean | 15 - 24 Mean | 25 - 44 Mean | 45 - 65 Mean | Total Mean |
|---|---|---|---|---|---|
| I. Simple and Natural Dialogue | 5.6 | 5.8 | 5.2 | 5.2 | **5.45** |
| II. Speak the Users' Language | 4.6 | 4.8 | 4.8 | 4.8 | **4.75** |
| III. Minimize User Memory Load | 5.6 | 5.8 | 5.4 | 5.2 | **5.5** |
| IV. Consistency | 5.6 | 5.6 | 5.4 | 5.4 | **5.5** |
| V. Feedback | 5.6 | 4.6 | 5.2 | 4.8 | **5.05** |
| VI. Clearly Marked Exits | 5.4 | 5.4 | 5.6 | 4.8 | **5.3** |
| VII. Shortcuts | 5.4 | 5.6 | 5.2 | 5.0 | **5.3** |
| VIII. Good Error Messages | 6.0 | 6.0 | 5.4 | 5.4 | **5.7** |
| IX. Prevent Errors | 5.2 | 4.8 | 5.0 | 4.0 | **4.75** |
| X. Help and Documentation | 5.0 | 5.6 | 5.4 | 5.2 | **5.3** |

Table 6.8: Heuristic Evaluation for the Tilt Game

that had the sentence "The game prevents me from making actions that are not the most indicated". Some of the testers sometimes ended up choosing the wrong option in the menu because the timer that surrounds the option was moving to fast. To solve this, the timer was extended and now the user has to point longer in order for the option to be chosen.

Another heuristic that scored below a 5.0 mean was "Speak the Users' Language" that had the associated sentence "I can easily understand how to interact with the application". This is easily explained since it was the first time for most testers with the *Leap Motion* device and it is not something that they are used to. It took some time for them to understand how to interact, especially in the oldest age groups.

## 6.4 Boids Band Evaluation

The next subsections present the results of the two evaluation techniques that were used to evaluate the Boids Band: a Field and an Heuristic Evaluation.

The testers were given a Laptop Computer with the Leap Motion device and were asked to perform several tasks. The appendix A shows the script that was used by the monitor of the conducted evaluations. Figures 6.10 and 6.11 shows one of the testers playing the game.

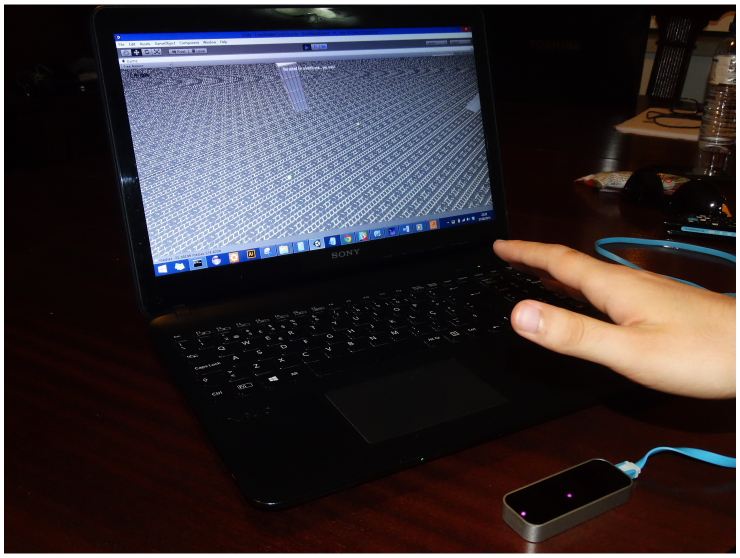


Figure 6.10: Evaluation of the Boids Band: Playing

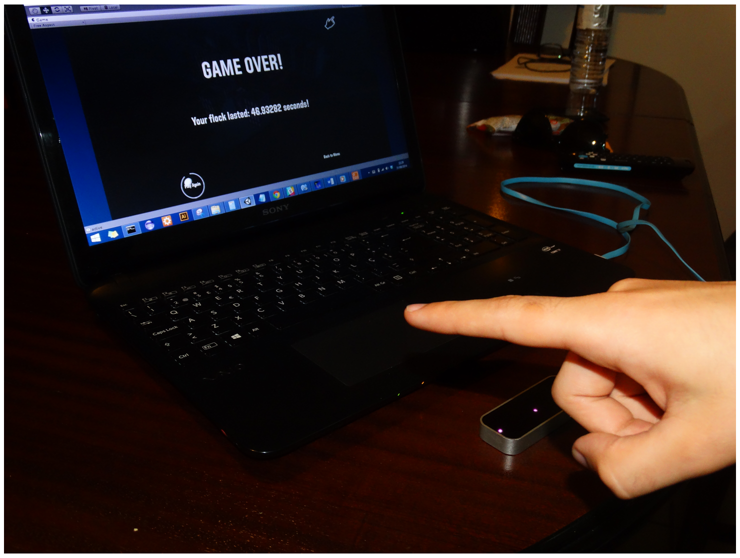


Figure 6.11: Evaluation of the Boids Band: Game Over

### 6.4.1 Field Evaluation

As in the previous game, the testers were given a computer with a connected *Leap Motion* device and were asked to follow a set of tasks. In the end, they were asked to evaluate some sentences related to what they had experienced. The first seven sentences that were asked had to do with the established natural user interfaces guidelines 2.2.2. On all sentences, the testers were asked to evaluate the sentence with a number from 1 to 6, 1 meaning "I completely disagree" and 6 meaning "I completely agree".

***The interface should consider its context***

For this guideline, two sentences were presented to the testers:

1. "I find the menus to be well constructed."
2. "The game is adequate to the *Leap Motion* device."

The first sentence intends to perceive if the testers find the menus appropriate while the second sentence intends to find out if the testers found the game adequate to the device.

*"I find the menus to be well constructed"*

In this sentence the global mean of the answers was 5.1, a very similar score to the one that the Tilt Game had received (5.0). As happened in the Tilt Game, the testers that belong to the youngest age groups seem to find that the menus were better constructed than the oldest testers. Table 6.9 shows the various means for each age group, while figure 6.12 shows how many testers chose each score.

| Age Group | Mean | Deviation |
|---|---|---|
| 12 - 14 | 5.60 | 0.55 |
| 15 - 24 | 5.60 | 0.55 |
| 25 - 44 | 4.40 | 0.55 |
| 45 - 65 | 4.80 | 0.84 |
| Total | 5.10 | 0.79 |

Table 6.9: "I find the menus to be well constructed": Boids Band Means

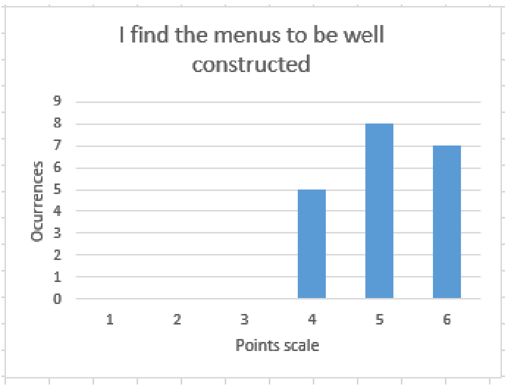


Figure 6.12: "I find the menus to be well constructed": Boids Band scores

*"The game is adequate to the Leap Motion device"*
In this sentence the global mean of the answers was better: 5.4, but very similar to what the Tilt Game had obtained (5.3). Table 6.10 shows the various means for each age group, while figure 6.13 shows how many testers chose each score.

| Age Group | Mean | Deviation |
|---|---|---|
| 12 - 14 | 5.60 | 0.55 |
| 15 - 24 | 5.60 | 0.55 |
| 25 - 44 | 5.20 | 0.45 |
| 45 - 65 | 5.20 | 0.45 |
| Total | 5.40 | 0.50 |

Table 6.10: "The game is adequate to the *Leap Motion* device": Boids Band Means

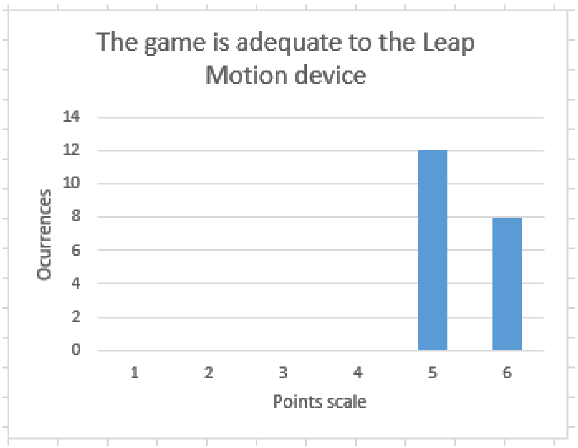


Figure 6.13: "The game is adequate to the *Leap Motion* device": Boids Band score

***The interaction between the user and the machine should be direct***
The sentence that was used to evaluate this guideline was: *"I consider the interaction to be similar to the one I would have when guiding physical birds"*.

In this sentence the global mean of the answers was 4.3. The means show that the youngest age groups consider the interaction to be more direct than the oldest age groups. This is one of the only means that was lower than in the Tilt Game which may be because the interaction that one has guiding birds would usually not happen in the physical world. Table 6.11 shows the various means for each age group, while figure 6.14 shows how many testers chose each score.

| Age Group | Mean | Deviation |
|---|---|---|
| 12 - 14 | 4.60 | 0.55 |
| 15 - 24 | 5.20 | 0.45 |
| 25 - 44 | 4.00 | 0.71 |
| 45 - 65 | 3.40 | 0.55 |
| Total | 4.30 | 0.86 |

Table 6.11: "I consider the interaction to be similar to the one I would have when using a physical board": Boids Band Mean

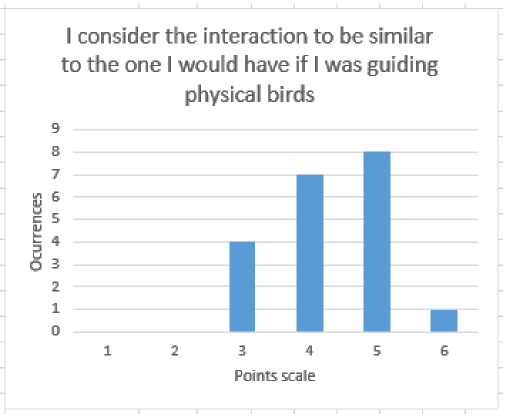


Figure 6.14: "I consider the interaction to be similar to the one I would have when using a physical board": Boids Band score

***The experiences should mimic real world interactions.***
The sentence that was used to evaluate this guideline was: *"I consider that I could interact directly with the game objects"*.

In this sentence the global mean of the answers was 4.7, very similar tho the one obtained for the Tilt Game (4.8). Table 6.12 shows the various means for each age group, while figure 6.15 shows how many testers choose each score.

| Age Group | Mean | Deviation |
|---|---|---|
| 12 - 14 | 5.0 | 1.14 |
| 15 - 24 | 5.0 | 0.55 |
| 25 - 44 | 4.4 | 0.45 |
| 45 - 65 | 4.4 | 1.14 |
| Total | 4.7 | 1.15 |

Table 6.12: "I consider that I could interact directly with the game objects": Boids Band Means

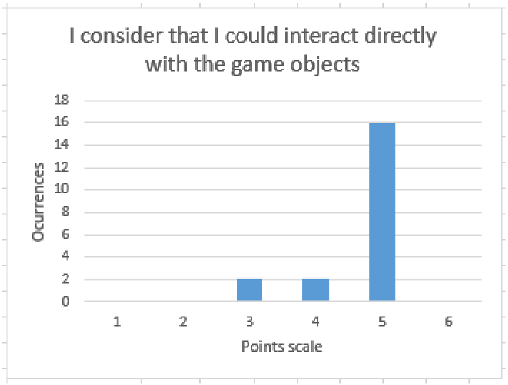


Figure 6.15: "I consider that I could interact directly with the game objects": Boids Band score

***The interactions should be enjoyed by the user***

The sentence that was used to evaluate this guideline was: *"I enjoyed playing this game"*. In this sentence the global mean of the answers was one of the best: 5.65, a huge improvement from the results obtained for the Tilt Game (5.15). This shows that the testers were better impressed with this game and enjoyed it the most. Table 6.13 shows the various means for each age group, while figure 6.16 shows how many testers chose each score.

| **Age Group** | **Mean** | **Deviation** |
|---|---|---|
| 12 - 14 | 5.80 | 0.45 |
| 15 - 24 | 6.00 | 0.00 |
| 25 - 44 | 5.40 | 0.55 |
| 45 - 65 | 5.40 | 0.55 |
| Total | 5.65 | 0.49 |

Table 6.13: "I enjoyed playing this game": Boids Band Means

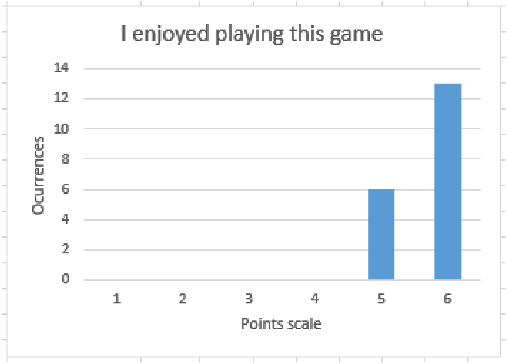


Figure 6.16: "I enjoyed playing this game": Boids Band scores

***The skills must be easy to learn and should use pre-existing knowledge***

For this guideline, two sentences were presented to the testers:

1. "I did not need to learn new skills to play this game."
2. "I managed to easily interact with the game."

The first sentence intends to perceive if the testers found the need to learn new skills to be able to play the game while the second sentence intends to find out if the testers managed to interact easily with the game.

*"I did not need to learn new skills to play this game"*

In this sentence the global mean of the answers was 5.0, a lot better than the score obtained for the Tilt Game (4.5), which shows that users may have learn things in the evaluation for the Tilt Game that could reuse in this game. Table 6.14 shows the various means for each age group, while figure 6.17 shows how many testers chose each score.

| Age Group | Mean | Deviation |
|---|---|---|
| 12 - 14 | 4.60 | 1.14 |
| 15 - 24 | 5.80 | 0.45 |
| 25 - 44 | 5.20 | 0.87 |
| 45 - 65 | 4.40 | 1.14 |
| Total | 5.00 | 1.03 |

Table 6.14: "I did not need to learn new skills to play this game": Boids Band Means

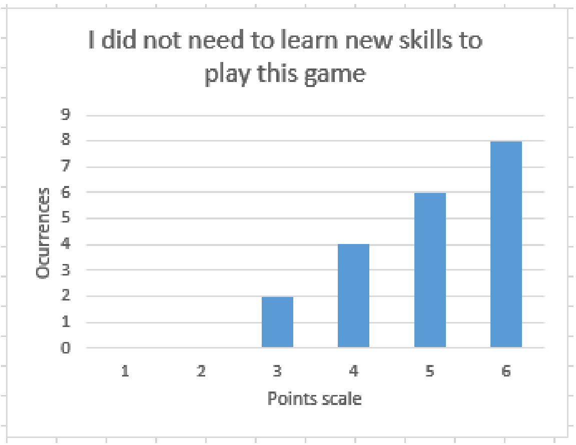


Figure 6.17: "I did not need to learn new skills to play this game": Boids Band scores

*"I managed to easily interact with the game"*

In this sentence the global mean of the answers was 4.85, much better than the one obtain for the Tilt Game (4.2). Table 6.15 shows the various means for each age group, while figure 6.18 shows how many testers chose each score.

| Age Group | Mean | Deviation |
|---|---|---|
| 12 - 14 | 4.60 | 0.00 |
| 15 - 24 | 5.60 | 0.00 |
| 25 - 44 | 4.80 | 0.89 |
| 45 - 65 | 4.40 | 0.89 |
| Total | 4.85 | 0.62 |

Table 6.15: "I managed to easily interact with the game": Boids Band Means

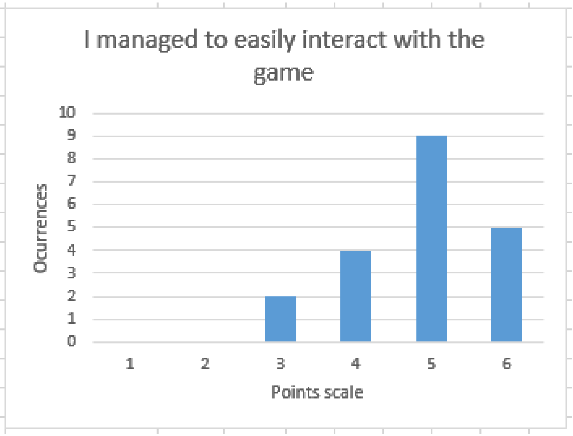


Figure 6.18: "I managed to easily interact with the game": Boids Bans score

***Results***

Overall, has had happened with the Tilt Game, all the guidelines had satisfactory results, all on the positive side of the scale. Also young users seem to be able to interact better with the game in almost every aspect, but in this game, the oldest users, in general, gave better scores to the sentences.

The guidelines that were most successfully followed were "The interface should consider its context" and "The interactions should be enjoyed by the user". These guidelines were even more successful in the Boids Band than in the Tilt Game.

### 6.4.2 Heuristic Evaluation

In the same questionnaire, the testers were also asked to score ten sentences matching the ten heuristics defined by Nielsen [92]. The full sentences that were presented to the user can be seen in the appendix B. The testers were asked to evaluate the sentence with a number from 1 to 6, 1 meaning "I completely disagree" and 6 meaning "I completely agree". These evaluation resulted in a set of indexes of which aspects are more prominent to improve in the game. Table 6.16 shows the results of this evaluation.

| Heuristic | 12 - 14 Mean | 15 - 24 Mean | 25 - 44 Mean | 45 - 65 Mean | Total Mean |
|---|---|---|---|---|---|
| I. Simple and Natural Dialogue | 5.6 | 5.4 | 5.2 | 5.4 | **5.4** |
| II. Speak the Users' Language | 4.8 | 5.2 | 5.0 | 5.0 | **5.0** |
| III. Minimize User Memory Load | 5.6 | 5.8 | 5.6 | 5.2 | **5.55** |
| IV. Consistency | 5.6 | 5.6 | 5.6 | 5.6 | **5.6** |
| V. Feedback | 5.6 | 4.6 | 5.4 | 5.4 | **5.25** |
| VI. Clearly Marked Exits | 5.6 | 5.6 | 5.6 | 4.8 | **5.4** |
| VII. Shortcuts | 5.4 | 5.6 | 5.2 | 5.0 | **5.3** |
| VIII. Good Error Messages | 6.0 | 6.0 | 5.6 | 5.4 | **5.75** |
| IX. Prevent Errors | 5.2 | 5.2 | 4.8 | 4.6 | **4.95** |
| X. Help and Documentation | 5.6 | 5.8 | 5.8 | 5.6 | **5.7** |

Table 6.16: Comparison between Game Engines

Overall the results were better than in the Tilt Game. The same two heuristics as in the Tilt Game continue to be the less scored ("Prevent Errors" and "Speak the Users' Language"). However, there was a slight increase in both of them.

## 6.5 Remarks

When performing the evaluations, some important remarks, made by the testers, were recorded. These remarks were not about the games themselves and often occurred while the player was playing other games, already published on the Airspace store.

The most common remark was the complaint that their arms were getting tired. This presents a problem for the Leap Motion device, since people seem to feel tired when using the device, which causes them to want to stop using it.

It was also mentioned many times that the device "could not sense" the hands or was "sensing it wrong". Sometimes the device is not able to find all fingers and does not respond correctly.

The fact that these remarks were common between testers, and not specific to a certain age group or genre, is indicative that, as has been mentioned earlier in section 2.3.3, the device still has its limitations, and a long way ahead of it, before it can be a truly useful tool.

# 7. Conclusions

In the last few years, natural user interfaces have been gaining ground in relation to graphical user interfaces. The emergence of touchscreens and motion control technologies has boosted the popularization of these interfaces. This project intended to explore this new concept of interaction, more specifically through the *Leap Motion* device.

This project started with the study of subjects such as user interfaces (especially natural user interfaces) and the history of games. The main subjects of this study were definitions of relevant concepts, projects that have been made in the area, and the gathering of information on principles and guidelines for development. This study that was conducted during the first semester, resulted in a set of information that was the backbone for the games that were developed during the second semester.

Another important phase during the first semester was the planning of what should be achieved by the end of the project. Goals were defined and consequently so were tasks to achieve these goals. Some choices were made related with the technologies that would be used and also related to the type of games that would be developed.

About the technologies it was decided to use *Unity 3D* since it facilitates the development for *Leap Motion* and the documentation that is available for developing with *Unity 3D* is considerably larger than the documentation that is available to work with its competitor *Cocos 3D* (section 4.4).

About the type of games that were chosen an attempt was made to have a wide variety of games. Four types of games were considered the most interesting to develop for *Leap Motion* (section 4.5). The main goal for the second semester was to take some of the concepts in those games and apply them to new games in such a way that reinvented them while exploring the possibilities that the *Leap Motion* device brings.

The main requirements for the games were defined through user stories and use cases. To describe the main flow of the games a navigation diagram was designed as well as some generic mockups for the game screens. Finally the architecture for the games was also defined.

In the second semester the development of the two games began. First, a simpler game was developed: the Tilt Game, based on a physical game of the same name. The second game to be developed was a little more complex and was based on the concepts of a driving and a shooting game. The games were developed with the goal of being published on the Airspace store. Therefore, some rules had to be followed for them to be accepted.

After the games were developed, usability evaluations were done to understand if the games were working as expected and if they followed the main natural user interfaces guidelines that had been established during the study of the state of the art. The results of these evaluation were, overall, positive and showed that the games are promising.

About the *Leap Motion* device, it became clear, during the development and evaluation phases, that it has not yet achieved its' full potential. During Summer University[1] in the Department of Informatics Engineering of University of Coimbra, an opportunity arose for the author to prepare and give a workshop about *Leap Motion* for the future students of the University. The students were given plenty of time to interact with the Leap Motion through a wide range of Airspace's games available to them. In all the games, the complaints seem to be similar to the ones that were recorded during the evaluation to the games produced in this project:

- "The sensor has difficulty tracking my hand."
- "It [the device] does not sense my hand."
- "It [the device] does not respond well to my commands."
- "My arms are getting tired."

The *Leap Motion* is a promising device, but some evolution is still to be done for it to achieve its' full potential.

## 7.1 Future Work

In this project, new possibilities for Natural User Interfaces, more precisely for the Leap Motion device, were explored. However, the project had limited time and some ideas could not be implemented.

[1]More information available online at `http://www.uc.pt/UV`

On June 10th, 2014, a new SDK[2] (beta version) for the *Leap Motion* API was released. The games were built using the previous release which presented some imperfections in hand tracking and gesture recognition. This API is said to include significant improvements to hand and finger tracking reliability, latency improvements, and better tool tracking. New possibilities could be studied with this API to try and improve the games in terms of usability.

The developed games were meant to be good enough so they could be accepted in the Airspace, and for that, everything that was planned had to be done correctly. Therefore, it was necessary to keep the games simple, given the limited time for the project. However, some improvements to the games were thought and are hereby presented.

The Tilt Game could be added more features such as: customizable skins for the board and objects of the game; insertion of new obstacles (besides the holes) such as, for example, special holes that instead of making the player lose, make the ball appear in the opposite side of the screen; insertion of power-ups that when caught make the ball go faster or slower; etc.

In the Boids Band game, lots of improvements could be added as well, such as a better representation for the fight between birds; the insertion of new bird species; power-ups that can make the birds fly faster or slower; new obstacles besides the buildings; etc.

For both games, the evaluation that was made during this project could be further studied and an attempt could be made to improve the aspects that scored lower for each heuristic defined in the Evaluation chapter (6).

[2]More information available online at `developer.leapmotion.com/features`

# A. Evaluation script

## A.1 Tilt Game

Hello. This testing session will be divided into three phases. First, you will be introduced to the Leap Motion device environment. Second, I will give you some tasks for you to try to perform. In the third phase, you will be asked to answer a small questionnaire.

**Phase 1** (The Airspace window is opened with two apps: Orientation and Duck-n-kill.)

1. Open the Orientation Application from the menu.
2. Try to use you hands to understand how the *Leap Motion* works.
3. Exit Orientation by pressing the Escape button.
4. Open the Duck-n-kill game.
5. Try to play it by using you hand to shoot the screen.

**Phase 2** (The Tilt Game is running and presents the Main Menu.)

1. Go to the instructions, read them, and then go back to the main menu.
2. Change the difficulty level to "Easy", and then go back to the main menu.
3. Check the High Scores for the "Hard" level, and then go back to the main menu.
4. Play the game.
5. Exit the game.
6. Change the difficulty level to "Hard", and then go back to the main menu.
7. Play the game.
8. Exit the game.

**Phase 3** (A questionnaire and a pen is given to the tester.)

1. Please score the given sentences, from 1 to 6, truthfully.

## A.2 Boids Band

Hello. This testing session will be divided into two phases[1]. First, I will give you some tasks for you to try to perform. In the second phase, you will be asked to answer a small questionnaire.

**Phase 1** (The Boids Band is running and presents the Main Menu.)

1. Go to the instructions, read them, and then go back to the main menu.
2. Check the Credits, and then go back to the main menu.
3. Check the High Scores, and then go back to the main menu.
4. Play the game.
5. Play again.
6. Exit the game.

**Phase 2** (A questionnaire and a pen is given to the tester.)

1. Please score the given sentences, from 1 to 6, truthfully.

[1]Since the testers were the same, and had already been introduced to the Leap Motion in the Tilt Game Evaluation, it was unnecessary to repeat the first phase of the previous evaluation.

# B. Evaluation Questionnaire

***Personal data***

Name:
Occupation:
Age:
Genre:

***Questionnaire***

Please give a score from 1 to 6 to the given sentences according to what you have experienced when playing the game.
(1 - I completely disagree | 6 - I completely agree)

1. I find the menus to be well constructed.

2. The game is adequate to the Leap Motion device.

3. I consider that I could interact directly with the game objects.

4. I consider the interaction to be similar to the one I would have when using a physical board. / I consider the interaction to be similar to the one I would have if I was guiding physical birds.

5. I enjoyed playing this game.

6. I did not need to learn new skills to play this game.

7. I managed to easily interact with the game.

I. Simple and Natural Dialogue

It is easy to understand how the application works.

II. Speak the Users' Language

I can easily understand how to interact with the application.

III. Minimize User Memory Load

I do not have to memorize things to be able to play.

IV. Consistency

The design and interaction with the application is consistent.

V. Feedback

The game gives me feedback when I take actions.

VI. Clearly Marked Exits

I understand how to leave the game or go back to the previous menu.

VII. Shortcuts

The shortcuts that are used to navigate the game are adequate.

VIII. Good Error Messages

I understand the error messages.

IX. Prevent Errors

The game prevents me from making actions that are not the most indicated.

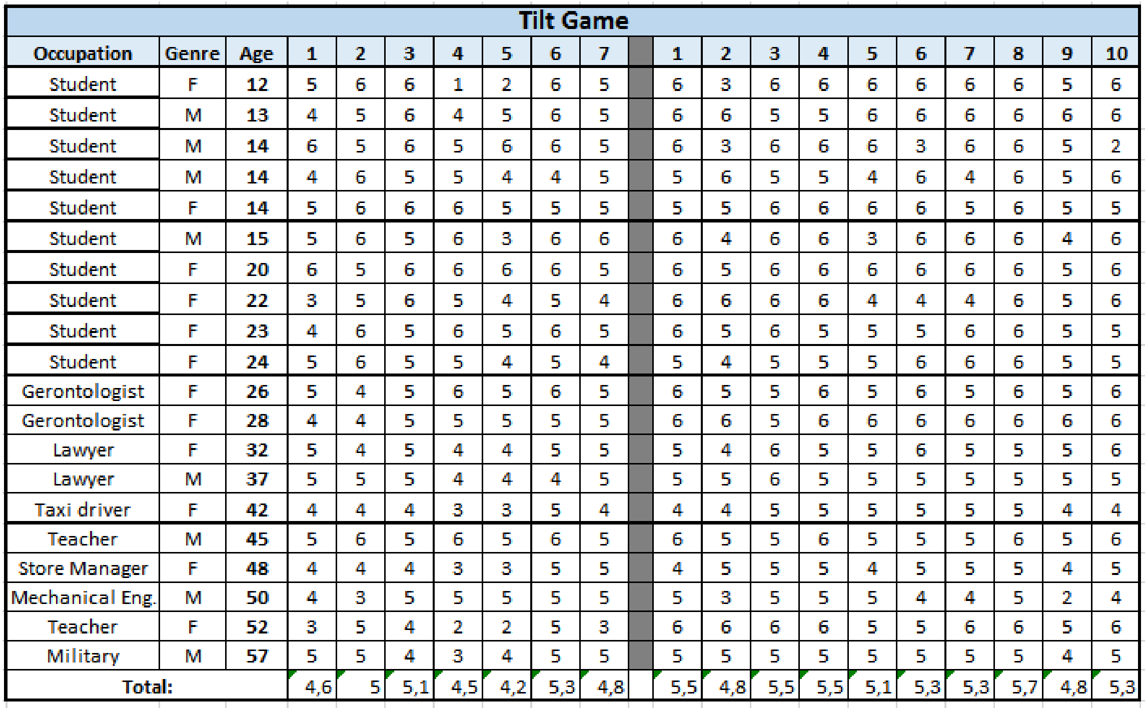
X. Help and Documentation

The game instructions are simple and clear.

# C. Evaluation Results

## C.1 Tilt Game

Figure C.1 shows the results of the scores given by the testers during the evaluations to the Tilt Game.

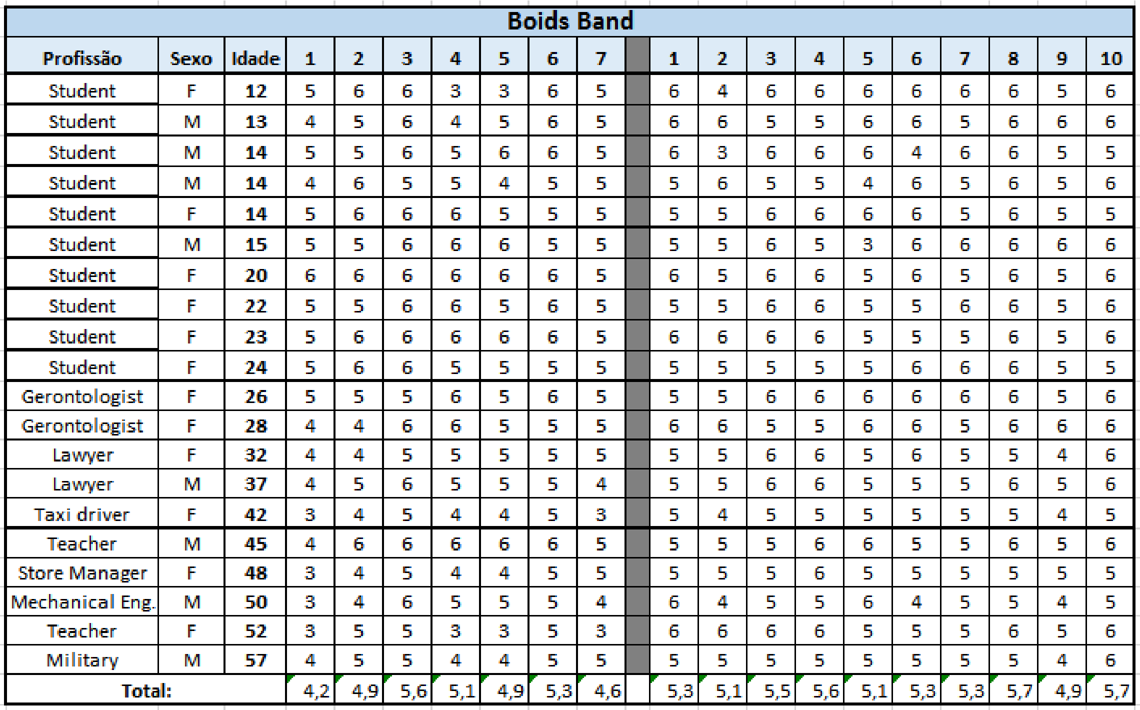


| Tilt Game | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Occupation | Genre | Age | 1 | 2 | 3 | 4 | 5 | 6 | 7 | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Student | F | 12 | 5 | 6 | 6 | 1 | 2 | 6 | 5 | | 6 | 3 | 6 | 6 | 6 | 6 | 6 | 6 | 5 | 6 |
| Student | M | 13 | 4 | 5 | 6 | 4 | 5 | 6 | 5 | | 6 | 6 | 5 | 5 | 6 | 6 | 6 | 6 | 6 | 6 |
| Student | M | 14 | 6 | 5 | 6 | 5 | 6 | 6 | 5 | | 6 | 3 | 6 | 6 | 6 | 3 | 6 | 6 | 5 | 2 |
| Student | M | 14 | 4 | 6 | 5 | 5 | 4 | 4 | 5 | | 5 | 6 | 5 | 5 | 4 | 6 | 4 | 6 | 5 | 6 |
| Student | F | 14 | 5 | 6 | 6 | 6 | 5 | 5 | 5 | | 5 | 5 | 6 | 6 | 6 | 6 | 5 | 6 | 5 | 5 |
| Student | M | 15 | 5 | 6 | 5 | 6 | 3 | 6 | 6 | | 6 | 4 | 6 | 6 | 3 | 6 | 6 | 6 | 4 | 6 |
| Student | F | 20 | 6 | 5 | 6 | 6 | 6 | 6 | 5 | | 6 | 5 | 6 | 6 | 6 | 6 | 6 | 6 | 5 | 6 |
| Student | F | 22 | 3 | 5 | 6 | 5 | 4 | 5 | 4 | | 6 | 6 | 6 | 6 | 4 | 4 | 4 | 6 | 5 | 6 |
| Student | F | 23 | 4 | 6 | 5 | 6 | 5 | 6 | 5 | | 6 | 5 | 6 | 5 | 5 | 5 | 6 | 6 | 5 | 5 |
| Student | F | 24 | 5 | 6 | 5 | 5 | 4 | 5 | 4 | | 5 | 4 | 5 | 5 | 5 | 6 | 6 | 6 | 5 | 5 |
| Gerontologist | F | 26 | 5 | 4 | 5 | 6 | 5 | 6 | 5 | | 6 | 5 | 5 | 6 | 5 | 6 | 5 | 6 | 5 | 6 |
| Gerontologist | F | 28 | 4 | 4 | 5 | 5 | 5 | 5 | 5 | | 6 | 6 | 5 | 6 | 6 | 6 | 6 | 6 | 6 | 6 |
| Lawyer | F | 32 | 5 | 4 | 5 | 4 | 4 | 5 | 5 | | 5 | 4 | 6 | 5 | 5 | 6 | 5 | 5 | 5 | 6 |
| Lawyer | M | 37 | 5 | 5 | 5 | 4 | 4 | 4 | 5 | | 5 | 5 | 6 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| Taxi driver | F | 42 | 4 | 4 | 4 | 3 | 3 | 5 | 4 | | 4 | 4 | 5 | 5 | 5 | 5 | 5 | 5 | 4 | 4 |
| Teacher | M | 45 | 5 | 6 | 5 | 6 | 5 | 6 | 5 | | 6 | 5 | 5 | 6 | 5 | 5 | 5 | 6 | 5 | 6 |
| Store Manager | F | 48 | 4 | 4 | 4 | 3 | 3 | 5 | 5 | | 4 | 5 | 5 | 5 | 4 | 5 | 5 | 5 | 4 | 5 |
| Mechanical Eng. | M | 50 | 4 | 3 | 5 | 5 | 5 | 5 | 5 | | 5 | 3 | 5 | 5 | 5 | 4 | 4 | 5 | 2 | 4 |
| Teacher | F | 52 | 3 | 5 | 4 | 2 | 2 | 5 | 3 | | 6 | 6 | 6 | 6 | 5 | 5 | 6 | 6 | 5 | 6 |
| Military | M | 57 | 5 | 5 | 4 | 3 | 4 | 5 | 5 | | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 4 | 5 |
| Total: | | | 4,6 | 5 | 5,1 | 4,5 | 4,2 | 5,3 | 4,8 | | 5,5 | 4,8 | 5,5 | 5,5 | 5,1 | 5,3 | 5,3 | 5,7 | 4,8 | 5,3 |

Figure C.1: Tilt Game Evaluation results table.

## C.2 Boids Band

Figure C.2 shows the results of the scores given by the testers during the evaluations to the Boids Band.



| Boids Band | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Profissão** | **Sexo** | **Idade** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
| Student | F | **12** | 5 | 6 | 6 | 3 | 3 | 6 | 5 | | 6 | 4 | 6 | 6 | 6 | 6 | 6 | 6 | 5 | 6 |
| Student | M | **13** | 4 | 5 | 6 | 4 | 5 | 6 | 5 | | 6 | 6 | 5 | 5 | 6 | 6 | 5 | 6 | 6 | 6 |
| Student | M | **14** | 5 | 5 | 6 | 5 | 6 | 6 | 5 | | 6 | 3 | 6 | 6 | 6 | 4 | 6 | 6 | 5 | 5 |
| Student | M | **14** | 4 | 6 | 5 | 5 | 4 | 5 | 5 | | 5 | 6 | 5 | 5 | 4 | 6 | 5 | 6 | 5 | 6 |
| Student | F | **14** | 5 | 6 | 6 | 6 | 5 | 5 | 5 | | 5 | 5 | 6 | 6 | 6 | 6 | 5 | 6 | 5 | 5 |
| Student | M | **15** | 5 | 5 | 6 | 6 | 6 | 5 | 5 | | 5 | 5 | 6 | 5 | 3 | 6 | 6 | 6 | 6 | 6 |
| Student | F | **20** | 6 | 6 | 6 | 6 | 6 | 6 | 5 | | 6 | 5 | 6 | 6 | 5 | 6 | 5 | 6 | 5 | 6 |
| Student | F | **22** | 5 | 5 | 6 | 6 | 5 | 6 | 5 | | 5 | 5 | 6 | 6 | 5 | 5 | 6 | 6 | 5 | 6 |
| Student | F | **23** | 5 | 6 | 6 | 6 | 6 | 6 | 5 | | 6 | 6 | 6 | 6 | 5 | 5 | 5 | 6 | 5 | 6 |
| Student | F | **24** | 5 | 6 | 6 | 5 | 5 | 5 | 5 | | 5 | 5 | 5 | 5 | 5 | 6 | 6 | 6 | 5 | 5 |
| Gerontologist | F | **26** | 5 | 5 | 5 | 6 | 5 | 6 | 5 | | 5 | 5 | 6 | 6 | 6 | 6 | 6 | 6 | 5 | 6 |
| Gerontologist | F | **28** | 4 | 4 | 6 | 6 | 5 | 5 | 5 | | 6 | 6 | 5 | 5 | 6 | 6 | 5 | 6 | 6 | 6 |
| Lawyer | F | **32** | 4 | 4 | 5 | 5 | 5 | 5 | 5 | | 5 | 5 | 6 | 6 | 5 | 6 | 5 | 5 | 4 | 6 |
| Lawyer | M | **37** | 4 | 5 | 6 | 5 | 5 | 5 | 4 | | 5 | 5 | 6 | 6 | 5 | 5 | 5 | 6 | 5 | 6 |
| Taxi driver | F | **42** | 3 | 4 | 5 | 4 | 4 | 5 | 3 | | 5 | 4 | 5 | 5 | 5 | 5 | 5 | 5 | 4 | 5 |
| Teacher | M | **45** | 4 | 6 | 6 | 6 | 6 | 6 | 5 | | 5 | 5 | 5 | 6 | 6 | 5 | 5 | 6 | 5 | 6 |
| Store Manager | F | **48** | 3 | 4 | 5 | 4 | 4 | 5 | 5 | | 5 | 5 | 5 | 6 | 5 | 5 | 5 | 5 | 5 | 5 |
| Mechanical Eng. | M | **50** | 3 | 4 | 6 | 5 | 5 | 5 | 4 | | 6 | 4 | 5 | 5 | 6 | 4 | 5 | 5 | 4 | 5 |
| Teacher | F | **52** | 3 | 5 | 5 | 3 | 3 | 5 | 3 | | 6 | 6 | 6 | 6 | 5 | 5 | 5 | 6 | 5 | 6 |
| Military | M | **57** | 4 | 5 | 5 | 4 | 4 | 5 | 5 | | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 4 | 6 |
| **Total:** | | | 4,2 | 4,9 | 5,6 | 5,1 | 4,9 | 5,3 | 4,6 | | 5,3 | 5,1 | 5,5 | 5,6 | 5,1 | 5,3 | 5,3 | 5,7 | 4,9 | 5,7 |

Figure C.2: Boids Band Evaluation results table.

# Bibliography

[1] Licínio Roque. Iu/usabilidade e avaliação in human-computer interaction course; dei; uc, 2013.

[2] Ian H. Witten and Saul Greenberg. User interfaces. Available online at `citeseerx.ist.psu.edu/viewdoc/download?rep=rep1&type=pdf&doi=10.1.1.217.4117` on January 25th of 2014.

[3] Everett N. McKay. *UI is Communication: How to Design Intuitive, User Centered Interfaces by Focusing on Effective Communication*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1st edition, 2013.

[4] Robert Jacob. User interfaces. Available online at `web.media.mit.edu/~anjchang/ti01/rjp.html` on January 25th of 2014.

[5] Rachel Hinman. *The Mobile Frontier: A Guide for Designing Mobile Experiences*. 1st edition, 2012.

[6] Command-line interface. Available online at `www.princeton.edu/~achaney/tmve/wiki100k/docs/Command-line_interface.html` on January 25th of 2014.

[7] Vannevar Bush and Jingtao Wang. As we may think. *Atlantic Monthly*, 176:101–108, 1945.

[8] The real history of the gui. Available online at `www.sitepoint.com/real-history-gui` on January 25th of 2014.

[9] A history of the gui. Available online at `arstechnica.com/features/2005/05/gui` on January 25th of 2014.

[10] History of writing technologies. Available online at `imrl.usu.edu/oslo/technology_writing/004_003.htm` on January 25th of 2014.

[11] Larry Greenemeier. The origin of the computer mouse, 2009. Available online at `www.scientificamerican.com/article.cfm?id=origins-computer-mouse` on January 25th of 2014.

[12] Carsten Schwesig. What makes an interface feel organic?, 2008. Available online at `www.organicui.org/?page_id=71` on January 25th of 2014.

[13] Roel Vertegaal and Ivan Poupyrev.

[14] Daniel Wigdor and Dennis Wixon. *Brave NUI World: Designing Natural User Interfaces for Touch and Gesture*. San Francisco, CA, USA, 1st edition, 2011.

[15] Marcus Ghaly. A short history of natural user interfaces, 2013. Available online at `www.identitymine.com/forward/marcus-ghaly` on January 25th of 2014.

[16] Jakob Nielson and Don Norman. The definition of user experience. Available online at `www.nngroup.com/articles/definition-user-experience/` on January 25th of 2014.

[17] Jhilmil Jain, Arnold Lund, and Dennis Wixon. The future of natural user interfaces. In *CHI '11 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '11, pages 211–214, New York, NY, USA, 2011. ACM.

[18] Jakob Nielson. Best application designs. 2012. Available online at `www.nngroup.com/articles/best-application-designs` on January 25th of 2014.

[19] Steve L. Kent. *The ultimate history of video games: from Pong to Pokémon and beyond*. Prima, 2001.

[20] T. L. Diamond. Devices for reading handwritten characters. 1957.

[21] Matthew Turk. Perceptual user interfaces. *Communications of the ACM*, 43:33–34, 2000.

[22] Steve Mann. *Intelligent Image Processing*. John Wiley and Sons, November 2 2001.

[23] Joshua Blake. *Natural User Interfaces in .Net*. Manning Publications Company, 1st edition, 2012.

[24] Marti A. Hearst. *Search User Interfaces*. Cambridge University Press, 1 edition, 2009.

[25] Sean Murphy. Design considerations for a natural user interface (nui). Available online at `http://www.ti.com/lit/wp/spry181/spry181.pdf` on January 25th of 2014.

[26] Bill Buxton. A touching story: A personal perspective on the history of touch interfaces past and future. Available online at `www.billbuxton.com/SID10%2031-1.pdf` on January 25th of 2014.

[27] Koert van Mensvoort. What you see is what you feel: exploiting the dominance of the visual over the haptic domain to simulate force-feedback with cursor displacements. In *Symposium on Designing Interactive Systems*, pages 345–348, 2002.

[28] Input/output device, 2012. Available online at www.britannica.com/EBchecked/topic/288883/inputoutput-device on January 25th of 2014.

[29] Nadia Jorney. About input and output devices. Available online at yourbusiness.azcentral.com/input-output-devices-20031.html on January 25th of 2014.

[30] Input device. Available online at www.computerhope.com/jargon/i/inputdev.htm on January 25th of 2014.

[31] Nicole Cohen. Timeline: A history of touch-screen technology, 2011. Available online at www.npr.org/2011/12/23/144185699/timeline-a-history-of-touch-screen-technology on January 25th of 2014.

[32] Bill Buxton. Multi-touch systems that i have known and loved, 2007. Available online at billbuxton.com/multitouchOverview.html on January 25th of 2014.

[33] Christine Erickson. The touching history of touchscreen tech, 2012. Available online at mashable.com/2012/11/09/touchscreen-history/ on January 25th of 2014.

[34] Motion control, 2013. Available online at www.giantbomb.com/motion-control/3015-474/ on January 25th of 2014.

[35] Leap motion: How does it work?, 2013. Available online at support.leapmotion.com/entries/23837547-How-does-it-work on January 25th of 2014.

[36] Melanie Pinola, 2011.

[37] Xuedong Huang, James Baker, and Raj Reddy. A historical perspective of speech recognition. *Commun. ACM*, 57(1):94–103, January 2014. Available online at cacm.acm.org/magazines/2014/1/170863-a-historical-perspective-of-speech-recognition/abstract on January 25th of 2014.

[38] MarshallH. Raskind and EleanorL. Higgins. Speaking to read: The effects of speech recognition technology on the reading and spelling performance of children with learning disabilities. *Annals of Dyslexia*, 49(1):251–281, 1999.

[39] National Center for Technology Innovation. Speech recognition for learning, 2010. Available online at www.ldonline.org/article/38655 on January 25th of 2014.

[40] Robert DeRosier and Ruth S. Farber. Speech recognition software as an assistive device: A pilot study of user satisfaction and psychosocial impact. *Work: A Journal of Prevention, Assessment and Rehabilitation*, 25(2):125–134, 2005. Available online at iospress.metapress.com/content/F7U1MUA2TEGUXGXV on January 25th of 2014.

[41] Anson Denis, Daveski Luann, Chavannes Patrick, and Karen Shaughnessy. Does speech recognition deserve recognition? a study comparing the efficacy of speech recognition vs. the standard keyboard. Available online at `atri.misericordia.edu/Papers/Speech.php` on January 25th of 2014.

[42] Alex M. Chong and Jacky Li. The future of ux / ui: The natural user interface.

[43] James J. Gibson. Theory of affordances. 1977.

[44] Donald A. Norman. Natural user interfaces are not natural. *interactions*, 17(3):6–10, May 2010.

[45] Alessio Malizia and Andrea Bellucci. The artificiality of natural user interfaces. *Commun. ACM*, 55(3):36–38, March 2012.

[46] Kim Gaskins. Study: Robots inspire new learning and creativity possibilities for kids, 2012. Available online at `latd.com/2012/01/16/robots-at-school-findings/` on January 25th of 2014.

[47] Philippa Roxby. Does technology hinder or help toddlers' learning?, 2013. Available online at `www.bbc.co.uk/news/health-22219881` on January 25th of 2014.

[48] Don Norman and Bahar Wadia. Opportunities and challenges for touch and gesture-based systems. 2013. Available online at `www.jnd.org/dn.mss/opportunities_and_ch.html` on January 25th of 2014.

[49] Steven C. Seow, Dennis R. Wixon, Ann Morrison, and Giulio Jacucci. Natural user interfaces: the prospect and challenge of touch and gestural computing. In Elizabeth D. Mynatt, Don Schoner, Geraldine Fitzpatrick, Scott E. Hudson, W. Keith Edwards, and Tom Rodden, editors, *CHI Extended Abstracts*, pages 4453–4456. ACM, 2010.

[50] Mary Bellis. History of sports. Available online at `inventors.about.com/od/sstartinventions/tp/History-Of-Sports.htm` on January 25th of 2014.

[51] History of games and sports. Available online at `www.historyworld.net/wrldhis/PlainTextHistories.asp?historyid=ac02` on January 25th of 2014.

[52] History of games timeline. Available online at `historicgames.com/gamestimeline.html` on January 25th of 2014.

[53] R. Dillon. *The Golden Age of Video Games: The Birth of a Multibillion Dollar Industry*. An A K Peters book. Taylor & Francis, 2011.

[54] Video game timeline. Available online at `www.onlineeducation.net/videogame_timeline/video-game-timeline.jpg` on January 25th of 2014.

[55] Pdp-1. Available online at pdp-1.computerhistory.org/ on January 25th of 2014.

[56] Tristan Donovan. *Replay: The History of Video Games*. Yellow Ant Media Limited, 2010.

[57] Wii: The total story. Available online at http://wayback.archive.org/web/20061218073811/http://wii.ign.com/launchguide/hardware1.html on January 25th of 2014.

[58] Introducing nintendo wii: Revolution gets an official name. Available online at wayback.archive.org/web/20061215105249/http://wii.ign.com/articles/703/703502p1.html on January 25th of 2014.

[59] Kinect vs. playstation move vs. wii: Motion-control showdown. Available online at www.pcmag.com/article2/0,2817,2372244,00.asp on January 25th of 2014.

[60] Darrell Etherington. Leap motion controller, 2013. Available online at techcrunch.com/2013/02/27/leap-motion-controller-pre-orders-to-ship-may-13-hits-best-buy-store-shelves-may-19-for-79-99 on January 25th of 2014.

[61] Dennis Ziesecke. Review leap motion motion control technology, 2013. Available online at http://www.notebookcheck.net/Review-Leap-Motion-Motion-Control-Technology.98821.0.html on August 25th of 2014.

[62] Limitations of the leap motion, 2012. Available online at https://forums.leapmotion.com/forum/general-discussion/general-discussion-forum/122-limitations-of-the-leap-motion on August 25th of 2014.

[63] Ashley Esqueda. Leap motion controller hands-on: Future technology not ready for daily use, 2013. Available online at http://www.technobuffalo.com/videos/leap-motion-controller-hands-on-video/ on August 25th of 2014.

[64] Alex Colgan. How does the leap motion controller work?, 2014. Available online at http://blog.leapmotion.com/hardware-to-software-how-does-the-leap-motion-controller-work/ on August 25th of 2014.

[65] Brian Williamson, Chadwick A. Wingrave, and Joseph J. LaViola Jr. Realnav: Exploring natural user interfaces for locomotion in video games. In *3DUI*, pages 3–10. IEEE, 2010.

[66] Procedural content generation wiki. Available online at pcg.wikidot.com on January 25th of 2014.

[67] Procedural generation. Available online at www.giantbomb.com/procedural-generation/3015-328/ on January 25th of 2014.

[68] Procedural graphics. Available online at in4k.northerndragons.ca/index.php?title=Procedural_Graphics_-_an_introduction on January 25th of 2014.

[69] Procedural generation. Available online at diablo.gamepedia.com/Procedural_Generation on January 25th of 2014.

[70] Random number generator. Available online at http://pcg.wikidot.com/pcg-algorithm:random-number-generation on August 25th of 2014.

[71] Pseudorandom number generator. Available online at http://pcg.wikidot.com/pcg-algorithm:pseudorandom-number-generator on August 25th of 2014.

[72] Linear congruential generator. Available online at http://pcg.wikidot.com/pcg-algorithm:linear-congruential-generator on August 25th of 2014.

[73] Fractal. Available online at http://pcg.wikidot.com/pcg-algorithm:fractal on August 25th of 2014.

[74] L-system. Available online at http://pcg.wikidot.com/pcg-algorithm:l-system on August 25th of 2014.

[75] Perlin noise. Available online at http://pcg.wikidot.com/pcg-algorithm:perlin-noise on August 25th of 2014.

[76] Genetic algorithm. Available online at http://pcg.wikidot.com/pcg-algorithm:genetic-algorithm on August 25th of 2014.

[77] User stories. Available online at www.mountaingoatsoftware.com/agile/user-stories on January 25th of 2014.

[78] User stories. Available online at www.extremeprogramming.org/rules/userstories.html on January 25th of 2014.

[79] Ray Wenderlich. Cocos2d vs sprite kit vs unity 2d tech talk video, 2014. Available online at http://www.raywenderlich.com/67585/cocos2d-vs-sprite-kit-vs-unity-2d-tech-talk-video on August 25th of 2014.

[80] Cocos3d versus unity for simple ios 3d games? Available online at http://gamedev.stackexchange.com/questions/21044/cocos3d-versus-unity-for-simple-ios-3d-games on August 25th of 2014.

[81] Unity answers, 2009. Available online at http://answers.unity3d.com/questions/7567/is-there-a-performance-difference-between-unitys-j.html on August 25th of 2014.

[82] Unity answers, 2009. Available online at http://answers.unity3d.com/questions/7528/how-should-i-decide-if-i-should-use-c-javascript-u.html on August 25th of 2014.

[83] Craig W. Reynolds. Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics*, 21(4):25–34, 1987. Available online at http://www.cs.toronto.edu/~dt/siggraph97-course/cwr87/ on August 25th of 2014.

[84] Reza Olfati Saber and Richard M. Murray. Flocking with obstacle avoidance: Cooperation with limited communication in mobile networks. 2003.

[85] Reza Olfati Saber. Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Transactions on Automatic Control*, 51:401–420, 2006.

[86] Craig Reynolds. Boids: Background and update, 2001. Available online at http://www.red3d.com/cwr/boids/ on August 25th of 2014.

[87] John Wakefield. Flocking boids (c#), 2008. Available online at http://dynamicnotions.blogspot.pt/2008/12/flocking-boids-c.html on August 25th of 2014.

[88] Flocking, 2013. Available online at http://wiki.unity3d.com/index.php?title=Flocking on August 25th of 2014.

[89] Boids pseudocode. Available online at http://www.kfish.org/boids/pseudocode.html on August 25th of 2014.

[90] Pierre Semaan. Menu gui options for leap motion and unity3d, 2013. Available online at http://pierresemaan.com/menu-gui-options-for-leap-motion-and-unity3/ on August 25th of 2014.

[91] Jakob Nielsen. Usability inspection methods. chapter Heuristic Evaluation, pages 25–62. John Wiley & Sons, Inc., New York, NY, USA, 1994.

[92] Jakob Nielsen. *Usability Engineering*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[93] Standardized survey classifications - individuals. Available online at http://www.pgagroup.com/standardized-survey-classifications.html on August 25th of 2014.

[94] Susan E. Wyse. 5 examples of survey demographic questions, 2012. Available online at http://www.snapsurveys.com/blog/5-survey-demographic-question-examples/ on August 25th of 2014.