MINISTRY OF NATIONAL EDUCATION

TECHNICAL UNIVERSITY
OF CLUJ-NAPOCA

FACULTY OF AUTOMATION AND COMPUTER SCIENCE
COMPUTER SCIENCE DEPARTMENT

GESTURE DETECTION IN VIRTUAL REALITY USING LEAPMOTION

LICENSE THESIS

Graduate: Radu PETRIŞEL
Supervisor: Assist. Prof. Dr. Eng. Adrian SABOU

2019

MINISTRY OF NATIONAL EDUCATION
TECHNICAL UNIVERSITY
OF CLUJ-NAPOCA

FACULTY OF AUTOMATION AND COMPUTER SCIENCE
COMPUTER SCIENCE DEPARTMENT

DEAN,
Prof. dr. eng. Liviu MICLEA

HEAD OF DEPARTMENT,
Prof. dr. eng. Rodica POTOLEA

Graduate: **Radu PETRIŞEL**

GESTURE DETECTION IN VIRTUAL REALITY USING LEAPMOTION

1. **Project proposal:** *A Reactive Programming oriented Unity asset for gesture detection using the LeapMotion controller*
2. **Project contents:** *(enumerate the main component parts) Presentation page, advisor's evaluation, title of chapter 1, title of chapter 2, ..., title of chapter n, bibliography, appendices.*
3. **Place of documentation:** *Technical University of Cluj-Napoca, Computer Science Department*
4. **Consultants:** Assist. Prof. Dr. Eng. Adrian SABOU
5. **Date of issue of the proposal:** November 1, 2018
6. **Date of delivery:** June 14, 2019

Graduate: ______________________________

Supervisor: ______________________________

MINISTRY OF NATIONAL EDUCATION

TECHNICAL UNIVERSITY
OF CLUJ-NAPOCA

FACULTY OF AUTOMATION AND COMPUTER SCIENCE
COMPUTER SCIENCE DEPARTMENT

Declaraţie pe proprie răspundere privind autenticitatea lucrării de licenţă

Subsemnatul(a) ____, legitimat(ă) cu ____________________ seria _______________ nr. ____________________
CNP ____, autorul lucrării ______________
____
____
elaborată în vederea susţinerii examenului de finalizare a studiilor de licenţă la Facultatea de Automatică şi Calculatoare, Specializarea ___________________________________ din cadrul Universităţii Tehnice din Cluj-Napoca, sesiunea ____________________ a anului universitar _______________, declar pe proprie răspundere, că această lucrare este rezultatul propriei activităţi intelectuale, pe baza cercetărilor mele şi pe baza informaţiilor obţinute din surse care au fost citate, în textul lucrării şi în bibliografie.

Declar, că această lucrare nu conţine porţiuni plagiate, iar sursele bibliografice au fost folosite cu respectarea legislaţiei române şi a convenţiilor internaţionale privind drepturile de autor.

Declar, de asemenea, că această lucrare nu a mai fost prezentată în faţa unei alte comisii de examen de licenţă.

În cazul constatării ulterioare a unor declaraţii false, voi suporta sancţiunile administrative, respectiv, *anularea examenului de licenţă.*

Data   Nume, Prenume

_______________   _________________________

Semnătura

Contents

Chapter 1

Introduction - Project Context

Virtual Reality is an experience that has gained huge popularity in the recent years. Because of this new means of interaction with this virtual world are needed and they should feel as natural as possible. Ergo, hand tracking and gesture detection is a ”must have” for modern VR applications.

1.1 Virtual reality

The term ”virtual” began its life in the late 1400s, meaning ”being something in essence or effect, though not actually or in fact” [1], but, in the IT context, the word has the meaning ”not physically existing but made to appear by software” [1]. The original use of the phrase ”virtual reality” is found in French playwright' Antonin Artaud collection of essays *Le Théâtre et son double*, first published in 1938 [2].

1.1.1 History

The precise roots of virtual reality are challenged, partially because of how hard it was to formulate a definition of an alternate reality notion. In 1968, Ivan Sutherland created what was widely regarded as the first head-mounted display system for use in immersive simulation applications, with the help of his students. In the next two decades, VR devices were mainly used for medical, automobile industry design, militry training and flight simulation purposes.

The 1990s saw the first commercially extensive release of consumer headsets, notably *Sega VR* (1991) and *Sega VR-1* (1994) launched by Sega, and *Nintendo's Virtual Boy* (1995). The 2000s were a period of comparative indifference from the public and investment towards VR techniques available on the market. Google launched *Street View* in 2007, a service that offers panoramic views of a growing amount of global locations such as highways, indoor houses and rural regions, which also integrates a stereoscopic 3D mode as of 2010.

The modern, consumer version of headsets started developing in the early 2010s. In 2013, Valve Corporation found and freely shared the breakthrough of low-persistence screens that make it possible today to show VR content lag-free and smear-free.

This discovery was quickly adopted by the other companies on the market, with Sony announcing *Project Morpheus* in 2014 and Google announcing *Cardboard* in 2015. In 2016, HTC released and shipped the first units of *Vive SteamVR*, the first major commercial headset for average users.

Figure 1.1: Project Morpheus (PlayStation VR) at gamescom in 2015

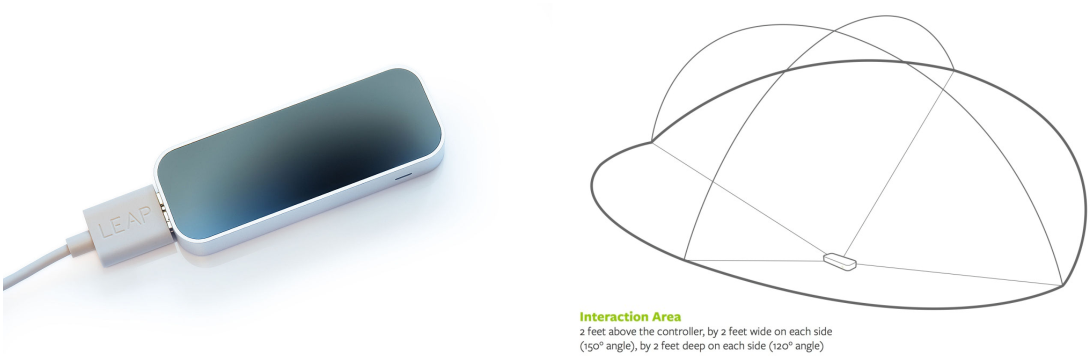
1.1.2 Modern technology

Present virtual reality headset displays rely on smartphone technologies including: gyroscopes and motion sensors for head, hand and body position monitoring, tiny high definition stereoscopic displays and small, lightweight and powerful computer processors.

Special input devices are required for interaction with the virtual world, such as hand controllers, haptic gloves, 3D mouse and optical tracking sensors. Both haptic gloves and hand controllers provide force feedback (in the form of vibration), with haptic gloves providing also feedback in the form of response force (like when picking a rubber duck).

1.2 Leap Motion

The *Leap Motion Controller* is a tiny peripheral USB device intended to be facing upwards on a physical desktop, but can also be mounted on a VR headset.



(a) The LeapMotion controller

(b) The interaction area

Figure 1.2: The LeapMotion system

1.2.1 Hardware

The *Leap Motion Controller* is really quite straightforward from a hardware view. Two cameras and three infrared LEDs are the core of the device. These track infrared light with a wavelength of 850 nanometers, which is outside the visible light spectrum. [3]

The unit has a big interaction room of 0.22 m^3 thanks to its wide-angle glasses, which takes the form of an inverted pyramid – the intersection of the areas of perspective of the binocular cameras (see figure 1.2b). The viewing range of the device is 60cm to 80cm, depending on the version of the firmware used.

This raw data is then stored in the device's local memory and then sent via USB to the *Leap Motion tracking software*. As the cameras work with near-infrared light, the data is in the form of grayscale stereo images, as shown in figure 1.3.

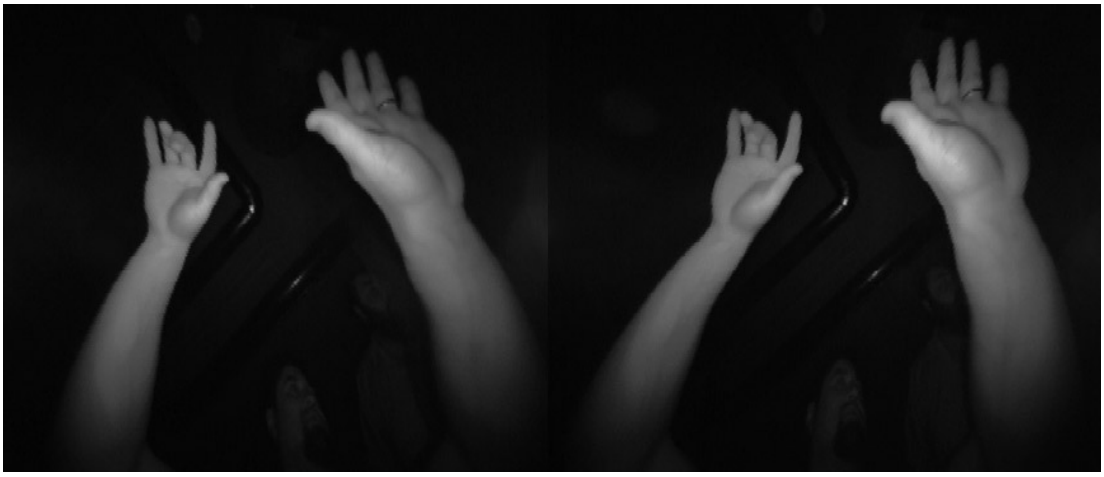


Figure 1.3: Leap Motion raw data

1.2.2 Sofware

It's time for some heavy mathematical lifting once the picture information is streamed to the computer. The *Leap Motion Controller* does not produce depth maps despite common misconceptions - instead it applies sophisticated algorithms to the raw sensor information.

The Leap Motion Service first compensates background objects (e.g. head) and lightning, and then extracts from the data the relevant information - arms, hands and fingers.

Though a transport layer, the results(frames) are fed to the *Leap Motion Control Panel* or to native and web clients. These orgaize the data into an object-orietented structure.

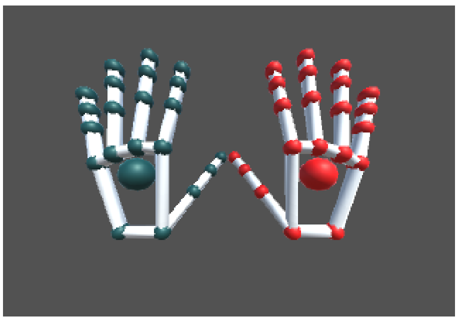


Figure 1.4: Capsule hands

1.3 Reactive Extensions

ReactiveX is a library for composing asynchronous and event-based programs by using observable sequences. [4]

It extends the observer pattern to support data and/or event sequences and provides operators that help you compose sequences declaratively while abstracting issues such as low-level threading, synchronization, thread security, concurrent data structures, and non-blocking I/O.

Table 1.1 shows how observables integrate in the programming world.

	single items	multiple items
synchronous	T GetData	IEnumerable<T> GetData
asynchronous	Awaitable<T> GetData	Observable<T> GetData

Table 1.1: Observable position in multiple items and asynchronous world

1.3.1 Why use observables?

ReactiveX observables are intended to be ***composable, flexible***and ***less opitionated***. These provide a huge advantage over structures like Java *Futures* or C# *Awaitables*, because it removes the need for ambiguous nesting of callbacks.

RX Observables also offer three methods for of flow control - *OnNext*, *OnError* and *OnCompleted* - which give the programmer very much liberty.

Iterable

```
getDataFromLocalMemory()
  .skip(10)
  .take(5)
  .map({ s -> return s + " transformed"
})
  .forEach({ println "next => " + it })
```

Observable

```
getDataFromNetwork()
  .skip(10)
  .take(5)
  .map({ s -> return s + " transformed"
})
  .subscribe({ println "onNext => " + it
})
```

Figure 1.5: Iterable vs Observable

Chapter 2

Project Objectives and Specifications

Describe the proper theme (as a research/design proposal, clearly formulated, with clear objectives, and some explanatory figures).

Stretches over about 10% of the paper.

2.1 Title

2.2 Other title

Chapter 3

Bibliographic research

Bibliographic research has as an objective the establishment of the references for the project, within the project domain/thematic. While writing this chapter (in general the whole document), the author will consider the knowledge accumulated from several dedicated disciplines in the second semester, 4^{th} year (Project Elaboration Methodology, etc.), and other disciplines that are relevant to the project theme.

Represents about 15% of the paper.

Each reference must be cited within the document text, see example below (depending on the project theme, the presentation of a method/application can vary).

This section includes citations for conferences or workshop [**?**], journals [**?**], and books [**?**].

In paper [**?**] the authors present a detection system for moving obstacles based on stereovision and ego motion estimation. The method is ... *discus the algorithms, data structures, functionality, specific aspects related to the project theme, etc.*... Discussion: *pros and cons.*

In chapter 4 of [**?**], the *similar-to-my-project-theme algorithm* is presented, with the following features ...

3.1 Title

3.2 Other title

Chapter 4

Analysis and Theoretical Foundation

Together with the next chapter takes about 60% of the whole paper

The purpose of this chapter is to explain the operating principles of the implemented application. Here you write about your solution from a theory standpoint - i.e. you explain it and you demonstrate its theoretical properties/value, e.g.:

- used or proposed algorithms
- used protocols
- abstract models
- logic explanations/arguments concerning the chosen solution
- logic and functional structure of the application, etc.

YOU DO NOT write about implementation.

YOU DO NOT copy/paste info on technologies from various sources and others alike, which do not pertain to your project.

4.1 Title

4.2 Other title

Chapter 5

Detailed Design and Implementation

Together with the previous chapter takes about 60% of the paper.

The purpose of this chapter is to document the developed application such a way that it can be maintained and developed later. A reader should be able (from what you have written here) to identify the main functions of the application.

The chapter should contain (but not limited to):

- a general application sketch/scheme,
- a description of every component implemented, at module level,
- class diagrams, important classes and methods from key classes.

Chapter 6

Testing and Validation

About 5% of the paper

6.1 Title

6.2 Other title

Chapter 7

User's manual

In the installation description section your should detail the hardware and software resources needed for installing and running the application, and a step by step description of how your application can be deployed/installed. An administrator should be able to perform the installation/deployment based on your instructions.

In the user manual section you describe how to use the application from the point of view of a user with no inside technical information; this should be done with screen shots and a stepwize explanation of the interaction. Based on user's manual, a person should be able to use your product.

7.1 Title

7.2 Other title

Chapter 8

Conclusions

About. 5% of the whole
Here your write:

- a summary of your contributions/achievements,
- a critical analysis of the achieved results,
- a description of the possibilities of improving/further development.

8.1 Title

8.2 Other title

Bibliography

[1] 2019. [Online]. Available: https://www.etymonline.com/search?q=virtual

[2] A. Artaud, *The Threatre and its Double*, 1938.

[3] A. Colgan, "How does the leap motion controller work?" 2014. [Online]. Available: http://blog.leapmotion.com/hardware-to-software-how-does-the-leap-motion-controller-work/

[4] 2019. [Online]. Available: http://reactivex.io/intro.html

Appendix A

Relevant code

```
 /** Maps are easy to use in Scala. */
object Maps {
  val colors = Map("red" -> 0xFF0000,
                   "turquoise" -> 0x00FFFF,
                   "black" -> 0x000000,
                   "orange" -> 0xFF8040,
                   "brown" -> 0x804000)
  def main(args: Array[String]) {
    for (name <- args) println(
      colors.get(name) match {
        case Some(code) =>
          name + " has code: " + code
        case None =>
          "Unknown color: " + name
      }
    )
  }
}
```

Appendix B

Other relevant information (demonstrations, etc.)

Appendix C

Published papers