DM

Augmented Reality Selection through Smart glasses

MASTER DISSERTATION

Jéssica Spínola Franco

MASTER IN COMPUTER ENGINEERING

UNIVERSIDADE da MADEIRA

A Nossa Universidade

www.uma.pt

December | 2019

Augmented Reality Selection through Smart glasses

MASTER DISSERTATION

Jéssica Spínola Franco
MASTER IN COMPUTER ENGINEERING

SUPERVISOR
Diogo Nuno Crespo Ribeiro Cabral

UNIVERSIDADE da MADEIRA

Augmented Reality Selection Through Smart Glasses

Jéssica Spínola Franco

Constituição do júri de provas públicas:

Karolina Baras, (Professora Auxiliar da Universidade da Madeira), Presidente

Pedro Filipe Pereira Campos, (Professor Associado com Agregação da Universidade da Madeira), Vogal

Diogo Nuno Crespo Ribeiro Cabral, (Investigador Auxiliar do Instituto Superior Técnico), Vogal

Novembro 2019

Funchal – Portugal

Acknowledgment

I would like to acknowledge my parents for making all this possible. For all their support and comprehension. For listening to all my stories and supporting my decisions.

To my sister and my cheerleader, thank you for believing in me when I struggle to believe in myself.

To Tiago, thank you for supporting me. For being a good friend and for always helping when I struggled with something. For all the coffee breaks and for every time you said you believe in me.

I want to acknowledge my dear friend Marta for all the talks we had over the past years. For all the laughs when things were going wrong and for being the same friend you always were.

I want to acknowledge my friend Delia for hearing out my dreams, for doing volunteering work with me and for the walks that helped clear my mind. For your time and for helping me in some illustrations of this work.

To my supervisor, Diogo Cabral, thank you for being always available, for the patience, for the weekly meetings, for pointing me in the right direction when I was feeling lost and for your time in revising everything.

I want to thank all participants in my study for their time and patience.

This work was partially funded by ITI, FCT/MCTES LARSyS (UID/EEA/50009/2013 (2015-2017)) and by FCT/MCTES LARSyS (UID/EEA/50009/2019).

Resumo

O mercado de óculos inteligentes está em crescimento. Este crescimento abre a possibilidade de um dia os óculos inteligentes assumirem um papel mais ativo tal como os smartphones já têm na vida quotidiana das pessoas.

Vários métodos de interação com esta tecnologia têm sido estudados, mas ainda não é claro qual o método que poderá ser o melhor para interagir com objetos virtuais. Neste trabalho são mencionados diversos estudos que se focam nos diferentes métodos de interação para aplicações de realidade aumentada. É dado destaque às técnicas de interação para óculos inteligentes tal como às suas vantagens e desvantagens.

No contexto deste trabalho foi desenvolvido um protótipo de Realidade Aumentada para locais fechados, implementando três métodos de interação diferentes. Foram também estudadas as preferências do utilizador e sua vontade de executar o método de interação em público. Além disso, é extraído o tempo de reação que é o tempo entre a deteção de uma marca e o utilizador interagir com ela. Um protótipo de Realidade Aumentada ao ar livre foi desenvolvido a fim compreender os desafios diferentes entre uma aplicação de Realidade Aumentada para ambientes interiores e exteriores.

Na discussão é possível entender que os utilizadores se sentem mais confortáveis usando um método de interação semelhante ao que eles já usam. No entanto, a solução com dois métodos de interação, função de toque nos óculos inteligentes e movimento da cabeça, permitem obter resultados próximos aos resultados do controlador. É importante destacar que os utilizadores não passaram por uma fase de aprendizagem os resultados apresentados nos testes referem-se sempre à primeira e única vez com o método de interação. O que leva a crer que o futuro de interação com óculos inteligentes possa ser uma fusão de diferentes técnicas de interação.

Palavras-Chave: Realidade Aumentada, Óculos inteligentes, Técnicas de interacção, *Touch inputs*, *Touchless inputs*, *Marker-based*

Abstract

The smart glasses' market continues growing. It enables the possibility of someday smart glasses to have a presence as smartphones have already nowadays in people's daily life.

Several interaction methods for smart glasses have been studied, but it is not clear which method could be the best to interact with virtual objects. In this research, it is covered studies that focus on the different interaction methods for reality augmented applications. It is highlighted the interaction methods for smart glasses and the advantages and disadvantages of each interaction method.

In this work, an Augmented Reality prototype for indoor was developed, implementing three different interaction methods. It was studied the users' preferences and their willingness to perform the interaction method in public. Besides that, it is extracted the reaction time which is the time between the detection of a marker and the user interact with it. An outdoor Augmented Reality application was developed to understand the different challenges between indoor and outdoor Augmented Reality applications.

In the discussion, it is possible to understand that users feel more comfortable using an interaction method similar to what they already use. However, the solution with two interaction methods, smart glass's tap function, and head movement allows getting results close to the results of the controller. It is important to highlight that was always the first time of the users, so there was no learning before testing. This leads to believe that the future of smart glasses interaction can be the merge of different interaction methods.

Keywords: Augmented Reality, Smart glasses, Interaction techniques, Touch inputs, Touchless inputs, Marker-based

Table of content

List of figures

List of tables

List of Abbreviations

API – Application programming interface

AR –Augmented Reality

FOV – Field of View

GPS – Global Positioning System

HMD – Head-Mounted display

POI – Point of Interest

SDK – Software Development Kit

URL – Uniform Resource Locator

VR – Virtual Reality

1. Introduction

1.1. Problem and Motivation

The market of augmented and virtual reality has been growing in the last few years and, according to Digi-Capital's report ('After Mixed Year, Mobile AR to Drive $108 Billion VR/AR Market by 2021 | Digi Capital' 2017), mobile augmented reality could become the primary of a $108 billion VR/AR market by 2021 with AR taking the lion's share of $83 billion. Augmented reality complements reality rather than replace it (R. Azuma et al. 2001). One of the main features of augmented reality is to provide complemented information about the user's surroundings. The technology should not be too intrusive to the point of distracting the user from the real world. It is possible that in a near future smart glasses will be very present in our life like smartphones already are**.** The use of smartphones forces the use of hands for looking at a screen; these actions sometimes can cause some discomfort, fatigue and loss of environmental awareness. Unlike smartphones, smart glasses allow the user to do two different things while keeping the hands free: reading information but simultaneously be aware of what is happening around (Rzayev et al. 2018).

There are several smart glasses on the market, with different interactions. Lee et al. (Lee and Hui 2018) performed a study about the interaction method for smart glasses. As a conclusion of their work, it is possible to understand that the smart glasses interaction methods are a little uncertain. The authors have selected several interaction methods such as on-device touch input, on-body touch input, hands-free input, and freehand input. All these interaction methods were split into two categories: touch input and touchless input. So far does not exist a "better" interaction method because some can cause discomfort others can be too difficult.

Usually, augmented reality applications overlay virtual 3D objects in interesting scenarios. However, these applications are limited, missing to provide additional information, like text or images, associated with physical elements, e.g., objects or places. Such information implies alternative user interactions that go beyond the traditional inputs like mouse and keyboard. However, it is important to explore alternatives are not too intrusive or awkward. Such alternatives can be either used in an indoor or outdoor context. Therefore, it is crucial to explore different interaction methods and understand which one is more efficient and which one causes less public embarrassment.

1.2. Objectives and Application Scenario

The goal of this work is the creation of possible scenarios to use smart glasses in a daily context. Two different types of interaction will be explored and some benefits and drawbacks of each will be presented. Besides, these interactions will be compared with the default interaction provided by the smart glasses.

In an indoor context, the focus will be a marker-based augmented reality system, using the smart glass's camera for detection. Three interaction methods will be implemented and evaluated.

In an outdoor context, a marker-less augmented reality system will be developed. In this scenario, the user will have access to all places in her surroundings and more information about a place will be available after selection. To achieve this goal, the sensors from smart glasses like GPS, accelerometer and geomagnetic sensor will be used.

1.3. Contribution

The contributions in this project are:

1. Study different types of smart glasses and understand the interaction techniques of each one
2. Explore interaction methods for selecting augmented reality information
3. Understand users' preferences and their willingness to perform different types of interaction methods in a public environment
4. Explore the difficulties of an outdoor solution and the interaction methods
5. Study of tools that allow the creation of markers

1.4. Publications

Part of this work is included in the following publication:

- (Franco and Cabral 2019) Jéssica Franco and Diogo Cabral. 2019. Augmented object selection through smart glasses. In Proceedings of the 18th International Conference on Mobile and Ubiquitous Multimedia (MUM '19). ACM, New York, NY, USA, Article 47, 5 pages. DOI: https://doi.org/10.1145/3365610.3368416

1.5. Thesis Outline

Chapter 2 highlights why smartphones can be replaced by smart glasses and how augmented reality can assume an important and convenient role in day-to-day life. Introduces the augmented reality concept, presents a set of AR applications that help the user to perform some daily tasks, and discusses the possible paths to build an augmented reality application, including a marker and markerless implementations. Chapter 2 also compares different Augmented Reality frameworks, the smart glasses available on the market and the different types of interaction methods using smart glasses.

Chapter 3 describes the implementation of the developed prototypes as well as the different interaction methods. It includes the specifications of the chosen smart glasses and the process of choosing fiducial markers. This chapter covers the two different scenarios: the first is an augmented reality application for an indoor environment using fiducial markers and the second is an augmented reality application for an outdoor environment using sensors. In both, it is discussed the interaction methods for these two different environments.

Chapter 4 covers the evaluation of the indoor application. It is detailed the process of evaluation as well as the results.

In Chapter 5, it is coved some limitations and it is defined as the goals for future work.

2. Background and Related Work

This chapter describes the background of augmented reality. It explains the process of building an augmented reality application and the differences between a marker-based and markerless based application. In addition, discusses how the sensors can complement the solutions. This chapter presents a comparison between the different SDKs that can be used to build an AR application, the comparison between three different smart glasses and the different interaction methods in AR using smart glasses.

2.1. Augmented Reality

Augmented reality combines real-world environments and digital data. The digital data can be either virtual objects or text and it is overlaid onto an image of the real world (R. Azuma et al. 2001; Poelman 2010).

The smartphones are already being used for augmented reality applications. However, the constant use of these devices can be uncomfortable. Having the device always positioned in parallel can be tiring. Besides, using the smartphone while walking can be dangerous because the user is not aware of the surrounding. The augmented reality in smartphones can also bring a not good experience to the user in an outdoor environment. Because of the light and the smartphone's screen can sometimes be difficult for the user to understand what is happening. The smart glasses can be the solution for most of the problems. The user does not have to carry the smart glasses in their hands all the time. And with it can always see the information on the display and be aware of what is happening around. Though, it is important not to overwhelm the display with information.

The computer-generated information is the core of augmented reality and although it was not created by only one man, Ivan Sutherland is considerate as the father of computer graphics. Sutherland (Sutherland 1965) described what it was a guideline of augmented reality when the author highlighted a visual display that can easily make solid objects transparent – the user can "see through matter". In 1968 (Sutherland 1968), Sutherland built the first virtual reality system, a display with head tracking and see-through optics, called "Sword of Damocles".

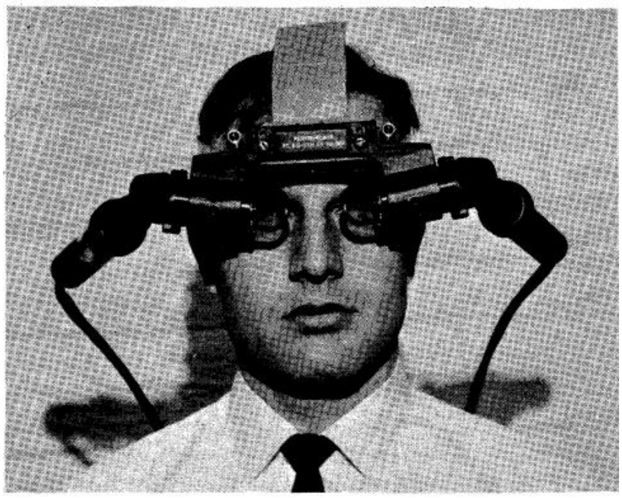


Figure 1 The head-mounted display optics (Sutherland 1968)

The term Augmented Reality only was invented by Caudell & Mizell in 1992 (Caudell and Mizell 1992). The authors used this term to describe the project they were building intending to help workers to perform manufacturing activities. To attain this, the workers would use a “heads-up display headset incorporating with head position sensing and real-world registration systems” (Caudell and Mizell 1992). In 1997, Azuma published a paper (R. T. Azuma 1997) about this field and explored the different applications that were being explored like medical, military, manufacturing and others. A few years later, the author wrote a new paper where the recent advances in AR were explored (R. Azuma et al. 2001). In this paper, the characteristics of an AR system were defined. For the authors, it is necessary to have an AR system that has properties that combine real and virtual objects in a real environment, runs interactively and aligns the two typed of objects, virtual and real, with each other. In the same paper, the authors defended that the definition of AR does not restrict to display technologies.

In the early years of AR, the term was more frequently used in literature even so there was not a consistent definition (Milgram et al. 1994). Milgram et al. suggested a clarification of the concept presented, as shown in Figure 2. According to the authors, Augmented Reality (AR) and Augmented Virtuality (AV) belong to the Mixed Reality (MR). The difference between both is that AR has a real environment with virtual elements and the AV offers an environment where the user is immersed in a virtual world that could resemble the real world.

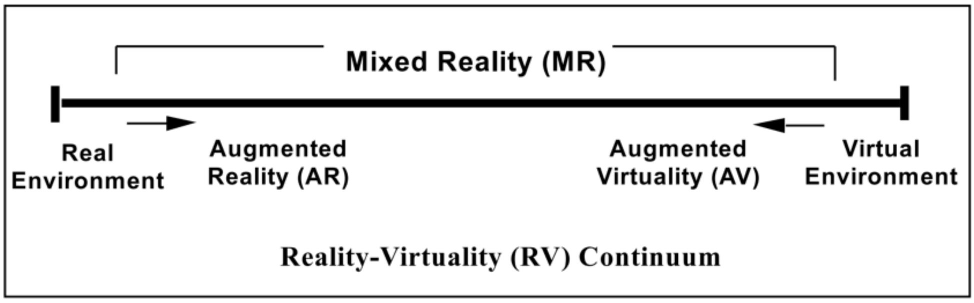


Figure 2 Milgram's Reality-Virtuality Continuum (Milgram et al. 1994)

To have an AR environment some technologies are required such as hardware to help to capture the scene, a display to allow the user to have the visualization of additional information and tracking the user movements or his location to display relevant content. The last one is an important requirement since its purpose is to attempt to calculate the trajectory of an object in the image plane while it moves around a scene.

As mentioned before, one of the properties of augmented reality is real and virtual objects combined in a real environment. One of the challenges is how to show the information to the user and how the device knows where the user is looking at. In this work two categories of augmented reality, Marker-based and Markerless, will be covered and how they overcome this challenge.

2.1.1. Marker-Based

Marker Based Augmented Reality is one of the ways to present virtual objects in a real environment, specifically on the top of a marker. This vision-based tracking method consists of place fiducial (or artificial markers). Siltanen (Siltanen 2012) details the requirements of marker-based AR, particularly: use of a camera, a computational unit, and a display. The author explains through a diagram, Figure 3, the simple AR system has three modules. The Capturing module is responsible for image capturing. In this module, the camera is activated, and the camera will be capturing the image. The tracking module is responsible for the calculation of the pose of the camera in real-time and to calculate the correct location and orientation of a virtual object. The rendering module is responsible for combining the real image and the virtual content using the calculated pose and then showing it in the display in a correct position and orientation. Figure 3 Simple AR system (Siltanen 2012a)

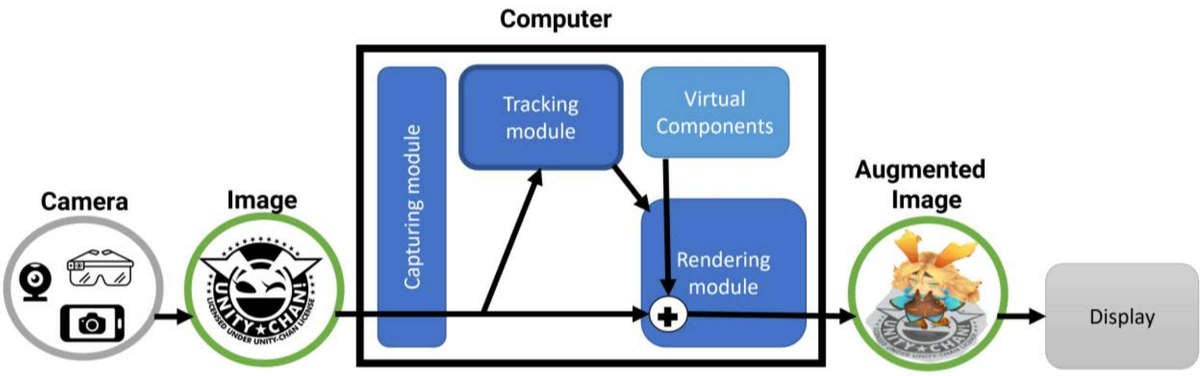


Figure 3 Simple AR system (Siltanen 2012a)

In 1999, Hirokazu Kato developed the ARToolkit which is a fiducial marker system. This tool is a software library for building applications that allows the overlay of virtual objects on the markers. The markers must have a square black border with a pattern inside, as shown in Figure 4.

Figure 4 ARToolkit markers

In the work of Martin Hirzer (Hirzer 2008) it is explained in detail how the process of markers' recognition works**.** The first step in the recognition process is to detect the marker's border. The second step is extracting the pattern inside of the borders. A vector is created, and it contains the grey values of the pattern. Then by correlation, this vector is compared to other vectors that belong to the ARToolkit library. For better understanding, the mechanism is illustrated in Figure 5.

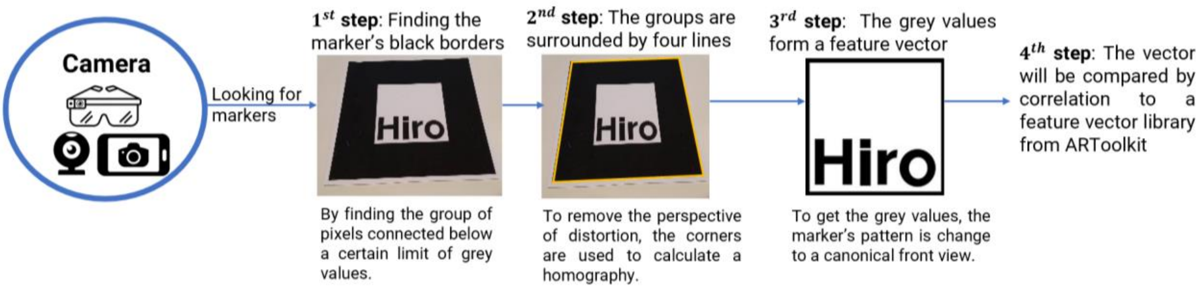


Figure 5 ARToolkit markers detection mechanism (Based on (Hirzer 2008))

After, the process of recognition is presented as a virtual object on the top of the marker. To present the virtual object in a correct position and orientation, this planar marker system uses a video tracking library that calculates the real camera position and orientation relative to physical markers in real-time.

The ARToolkit was very popular in the AR community at the beginning because it was simple to use and relatively robust but had some drawbacks. As mentioned previously, ARToolkit uses a correlation mechanism. Yet this method causes high false positives which means that a marker is identified even when it is not present. Another drawback of this framework is that only square black borders are identified, which means that also the singularity of the markers can decline as far as the library size increase and the confusion rate will increase, (Hirzer 2008; Fiala 2005).

ARTag (Fiala 2005) is alike the ARToolkit, both allow video tracking capabilities. Though, this tool changed the approach and tried different ways of solving the drawbacks of ARToolkit. Instead of a correlation mechanism like ARToolkit, ARTag uses a digital approach. The ARTag maintained the square black borders of the markers, as shown in Figure 6, but improved their detection by being more resistant to light changes. The size of the library was increased and this planar marker system discards the pattern files (Fiala 2005).

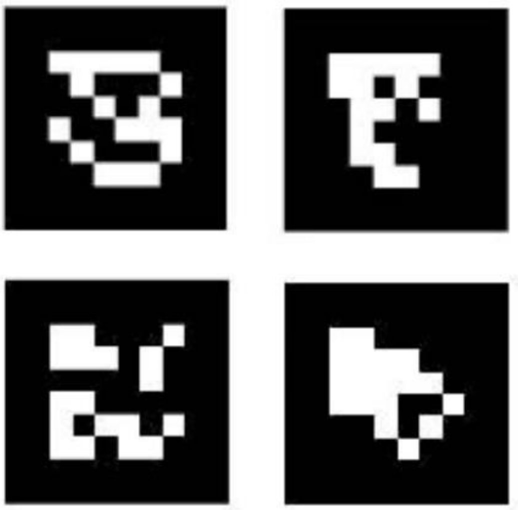


Figure 6 ARTag Markers

Currently, the most popular framework is Vuforia[1]. Vuforia is an SDK of augmented reality for mobile devices that allows the creation of new augmented reality

[1] https://library.vuforia.com/content/vuforia-library/en/getting-started/overview.html

applications. Through computer vision technology it is possible to recognize and tracking image in real-time. This SDK has several features that allow attaching virtual content to a specific object. It is possible to have model targets, image targets, multi targets, cylinder targets, object targets and VuMarks. With model targets through pre-existing 3D models, it is possible to recognize objects by shape. This SDK can also put virtual content in flat images by having image targets. One advantage of Vuforia is the capacity of identifying multiple targets like objects or multiple images. Besides, it is possible to create new image targets or even a VuMarks which is like a bar code. These VuMarks can be created by using the Vuforia Model Target Generator. Another feature is the possibility of cylinder targets like objects with cylindrical and conical shapes. At least the object target is created by scanning the object. With Vuforia it is possible to attach content to plane surfaces like tables, floors, and others. One of the main reasons for using Vuforia was the possibility of creating virtual buttons. It is possible to add more information, objects or images by simply adding a button and when the user clicks on it more relevant information is shown.

Even with the development of new and better tools, it is completely unthinkable to place markers everywhere. It would be impossible to cover the entire world with different markers, and for that reason, there is a Markerless category in the Augmented Reality field.

An augmented reality technology must have a detecting/ tracking method like an optical tracking method (marker-based and markerless) and this can be complemented by sensors (See chapter 2.1.2. Markerless).

2.1.2. Markerless

In the previous section, it was described how the marker-based systems work and how their recognition is made. Although marker-based systems are easier to implement, mostly in terms of recognition and registration, it would be impossible to fill the world with markers. However, without such markers, tracking objects is quite challenging. To address such challenge, augmented reality applications rely on three main approaches of coordinate systems: a 3D virtual model of the real world; a real-world position system; or a hybrid approach that combines both virtual and real-world coordinates. Although these three systems present and combine different tracking methods, they share the same problem: to know the relative position between the user and the augmented object.

The 3D virtual model helps by providing a virtual map of the environment to the system. This map is invisible to the user and the system only renders the augmented virtual elements. The usage of a 3D model facilitates object tracking by only requiring the position, direction, and field of view of the user in the virtual world. This information can be used by the system to compute the virtual elements that are visible to the user in the 3D model and consequently should be rendered. This approach is quite useful in small spaces that correspond to a small 3D model, e.g., a building floor, but quite time consuming if for large areas.

The relative position between the user and the augmented object using real-world coordinates is usually obtained through location and inertial sensors, e.g., GPS, or through image processing and classification techniques. While sensors can lose signal and lack accuracy, pure image processing and classification techniques can be computationally demanding and sensible to real-world changes, e.g., light changes or changes in the background.

An hybrid approach combines a 3D model with real-world coordinates, allowing to mitigate the loss of sensor signal and its lack of accuracy with image processing by matching real and virtual visual features. At the same, it can reduce the quantity of image processing with sensor data, by matching the real-world coordinates of the user with the corresponding 3D coordinates. Although this approach can improve object tracking and reduce the computational power needed, still presents scaling issues due to the need for a 3D virtual model.

Object tracking, i.e., the correct calculation of the relative position (virtual or real) between the user and the augmented object, in augmented reality systems can be divided into three main categories: computer vision approaches; sensor-based approaches and hybrid approaches that combine both computer vision and sensors.

Computer vision approaches aim to extract the relative position from the visual content and motion obtained through the video camera, e.g., using natural features. The natural features that are more frequently used are interest points or key points. The target object has salient points, these interest points must be easily found and must stay stable even in different points of view or different conditions. For this to happen it is important that the object has an irregular texture and is sufficiently dense. However, not all objects have a texture that satisfies the needs to track the interest point and, for that reason the alternative is tracking the object by edge feature, in case that the object's outline is observable. Besides interest points and edge features, there are other ways of tracking a target object without using the fiducial markers. It is possible to capture the camera's image and use the whole-image alignment to keyframes.

Unlike the marker-based systems, there is no need to prepare the target objects before. Although this is a big advantage of natural features this brings other challenges. In a system like this, it is fundamental to know the position and orientation of the user/camera. The registration of virtual and real objects in a three-dimensional world needs to establish the object's pose with six degrees of freedom (DOF). Three degrees of freedom for position to track the location and three degrees of orientation to track the head movement to understand in which direction the user is looking.

As said above, natural features are one of the markerless approaches. In a way to identify the natural features in the object, a model of the object it is previously created (Schmalstieg e Höllerer 2016). However, two other approaches are tracking by detection and the homography. The tracking detection is a simple computer vision approach, by computing the camera pose from the matching interest points in every frame. The interest points are represented by descriptors. The most popular algorithms of descriptors, of sparse interest points that will be explained below, are scale-invariant feature transform (SIFT), SURF and Brief (Schmalstieg and Höllerer 2016; Siltanen 2012a).

The homography-based visual tracking is an incremental tracking and motion tracking. In contrast to tracking detection, in this approach, two frames of the same surface can be related. The points from the image or a frame in the plane are related to the correspondent points of a second image or frame (Hartley e Zisserman 2004). The incremental tracking requires two components. The first component is the incremental search. The other component is the direct matching, and it can be simply done by comparing the patch image around an interest point with the image patch from the image that is being studied.

The tracking of natural features can be done with one or multiple cameras. However, with multi-cameras, the hardware cost will increase as well as the computational demands. While with one camera there is a need to track the points and make the match between the 2D points from the camera's image and the 3D points from the world this match can be made in two different ways densely matching or sparsely matching.

As mentioned before, the tracking model from reference is available before the tracking begins however there is the model-free tracking. The model-free tracking, also called visual odometry, is the simplest way and can be seen as a precursor to simultaneous location and mapping (SLAM). Visual odometry means continuous tracking of a camera concerning an arbitrary starting point. Moreover, it helps to calculate a 3D reconstruction of the environment to use it only in an incremental tracking. The SLAM can provide model-free tracking, however, for an outdoor solution it might not be the most suitable approach. The SLAM trusts in the actual visual environment, because of that the outdoor environment can involve several weather phenomena.

Furthermore, there are areas with unsuitable or poor texture and repetitive structures and that cannot be visually discriminated. The outdoors also brings difficulties in terms of location of the database, the database for a large area could grow markedly and would be time-consuming searching in this type of database. In addition to these problems, computer vision methods are computationally heavy and sometimes may not be accurate.

Tracking devices use several inertial and location sensors, such as GPS, linear accelerometers, magnetometers, and others. These devices allow to know user position and direction with low computational power. GPS is the most popular source of information, particularly in outdoor applications. GPS coordinates can be used to get data about the device's surroundings. However, GPS is usually not enough accurate for small object augmentation. Smartphone GPS accuracy is usually within 4.9m (National Coordination Office for Space-Based Positioning, Navigation, and Timing).

As mentioned before, the 6 DOF is extremely important for an AR application. It would be ideal that the tracking systems would use 6 DOF however some sensors only deliver 3 DOF orientation or only 3 DOF positions. Rabbi and Ullah (Rabbi and Ullah

2013) highlighted the degrees of freedom of each kind of sensor as well as their advantages and disadvantages through Table 1.

Sensor Tracking	Accuracy	Sensitivity	Cost	DOF	Advantages	Disadvantages
Optical	Accurate	Light	Cheaper	3/6 DOF	High update rate, better resolution	Effect with optical noise, occlusion
Magnetic	Less Accurate	Electronic Devices, electromagnetic noises	Cheaper	6 DOF	No occlusion problema, high update rate	Small working volume, distance affect accuracy
Acoustic	Less Accurate	Temperature, Humidity, Pressure	Cheaper	3/6 DOF	Slow, Small, light, no distortion	Occulusion and ultrasonic noise
Inertial	Accurate	Friction	Cheaper	1/3 DOF	No reference needed, No prepared environment needed	Due to small friction conservation error
Hybrid Techniques	Accurate	Depend on the sensors used	Costly	6 DOF	Compact, accurate, stable	Depend on sensors used

Table 1 Sensors based Tracking (based on (Rabbi and Ullah 2013))

As can be observed all sensors have their limitations, therefore the hybrid technique (fusion between the sensor-based tracking and markerless based tracking) tries to minimize the disadvantages of each technique by using multiple measurements. As a result, the hybrid technique produces robust and optimized results, e.g., only the feature points sufficiently close to the GPS coordinates will be considered to matching with the feature's points retrieved from the captured image and if it is certain that the most features will be observed at limited distance this can help to organize the database. In the previous section, it was possible to understand the kind of drawbacks that marker tracking brings. In this section, it was possible to understand that computer vision techniques can be more complex and computational heavy comparing with markers tracking. For these reasons, in the project, it would be used as a hybrid solution using sensors and markers.

2.1.3. Comparison between AR SDKs

Different AR SDKs have been made available, in order to facilitate the development of AR applications. They include image and fiducial marker tracking algorithms as well as methods for virtual object rendering on camera images. The ArCore is an SDK developed by Google that allows the creation of augmented reality applications. This SDK allows many AR features like motion tracking, light estimation, and environmental understanding. The motion tracking feature allows the user to walk around and interact with virtual content that is displayed in the world. The light estimation allows a more realistic environment because the objects have their light and it changes dynamically. The environmental understanding is a feature that allows place virtual objects that connect with the world. ArCore can be set up in both the development environment (Unity and Android Studio). The problem with this SDK is that the API target minimum has to be 24 (‘Choose Your Development Environment | ARCore’ n.d.) and the API target of the chosen smart glasses (Moverio BT-300) is 22.

As mentioned in section 2.1.1, the ARToolkit was the first tool developed that allowed the creation of AR applications and because of that several other tools were developed based on ARToolkit. It was convenient to have a comparison between the oldest tool and the recent ones. However, all the packages and plugins found to lead to errors and obsolete warnings. With these, it was impossible to compile the solution.

The ARreverie brings all the ARtoolkit functionalities. This SDK adds a new tracking method that will detect an ImageTarget without creating an external Marker database. When it was compiling a demo project and a project made from scratch by the tutorial available in the ARreverie page (‘Getting Started with ARToolKit+ Unity Plugin (Open Source AR) – ARreverie Technology’ n.d.), gave errors and wasn’t possible to deploy in Moverio BT-300.

Nyartoolkit («Welcome to NyARToolkit.EN | NyARToolkit project») has a library of augmented reality-based on ARToolkit. The Nyartoolkit project has a tool that allows adding new image targets.

Vuforia can be integrated into both android studio as in unity. As mentioned above, with Vuforia it is possible to do different things like recognition of objects, text or markers. Vuforia has its markers, although have a limitation of the number of VuMarks and the Cloud recognition. The free functionalities include a Vuforia watermark.

SDK	Disadvantage	Advantage	Development environment
ARCore	API target minimum is 24 while the Moverio's API target is 22.	Motion Tracking, Light estimation, and environmental understanding	Unity, Android Studio
ARToolkit	All plugins founded have errors and have many obsoletes functions	First system	Unity, Android Studio
ARreverie	Paid	Bring all ARToolkit functionalities	Unity
NyARToolkit	Have some false positives	Based on ARToolkit	Unity
Vuforia	Not all markers are detected as easily	Lots of information on how to use and how to do	Unity, Android Studio

Table 2 Comparison of SDK

Due to the smart glasses' limitations, like the Android version, the ARCore could not be used. Because of the disadvantages of ARToolkit and ARreverie, it was chosen the Vuforia and NyARToolkit in Unity due to the vast documentation and the ease to compile both projects. The NyARToolkit allowed to detect a double tap on the glasses' frame, while the Vuforia allowed to create virtual buttons for mid-air interactions.

2.2. Smart glasses

Smart Glasses are a wearable technology similar to common glasses but with it the possibility to combine virtual and real content and is displayed to the user in his field of view. The smart glasses are provided by a variety of sensors and internet connection.

Several visual displays have been employed in smart glasses to implement augmented reality. Two methods enable the combination of real and virtual content: optical see-through display and video see-through display. The main difference between the two is how the real world is captured by the device (Medeiros et al. 2016). On the optical see-through system, the real world is seen through a semi-transparent mirror and the computer generates virtual images that are reflected in the mirror (Rolland, Holloway, and Fuchs 1995; Medeiros et al. 2016). On the video see-through display systems, the real-world image is captured by one or multiple cameras and the computer generates virtual images that are electronically combined with the video (Rolland, Holloway, and Fuchs 1995; Medeiros et al. 2016).

The augmented reality smart glasses are useful in fields like medical, education, sports, entertainment and others. In the medical and healthcare field, for example, smart glasses can be used for broadcasting surgeries, facilitating resident teaching (Whitaker and Kuku 2014), reducing the time spent in document patient visits («How Google Glass Automates Patient Documentation For Dignity Health - CIO Journal. - WSJ»), or even to get the relevant information about the patient when he is in the operating room («Does Google Glass Have a Place in the Operating Room?»).

Smart glasses can also be used for educational reasons in the healthcare field, but this is not the only field that is using smart glasses for educational purposes. Researchers at Carnegie Mellon University found a solution to help students learning by informing teachers about how each student is learning, this real-time information is displayed in smart glasses (Holstein, McLaren, and Aleven 2018; Locklear 2018). Another example of its application in education is distance learning and virtual assistant using smart glasses (Spitzer, Nanic, and Ebner 2018). Also, improves communication between student and teacher, by giving feedback to the teacher through smart glasses if a student understood what the teacher said (Trassard 2013).

Sörös et al. (Sörös, Daiber, and Weller 2013) created a prototype for cyclists, this prototype will help the training but also be, like the previous example, a virtual assistant. The cyclist could receive messages through the smart glasses but also could be informed about how well they are doing compared with previous training sessions.

In the work of Ruy and Park (Ryu and Park 2016), it was implemented an AR system able to detect text documents in real scenes. Initially was explained that attach markers to each page from the document or book are inconvenient and not good for users. The system created does not recognize words or characters but allows partial occlusions. This work can be relevant as although it is inconvenient to have markers to identify each page, it is possible in a specific context with a marker to be able to access a document or image. For example, in a meeting, the leader shares a marker with all the attendees so that they have access to a specific document.

Several fields have found smart glasses useful to achieve certain purposes. As a result, the market is growing and in the latest years, several companies have invested and released smart glasses. According to Vera Romeiro from wearable-technologies (Romeiro n.d.), 2018 could be the year of smart glasses.

In augmented reality, the interaction of the user with virtual and real content is very important. As mentioned previously, when there is knowledge of where the user is and where he is looking at, related content would overlay the real image and the user could interact with it. However, it was not mentioned how this interaction would happen. For smartphones, touchscreens are the primary interaction method. Nowadays, gestures like tapping and swiping are very familiar to the users (Hsieh et al. 2016). Yet the content displayed on smart glasses is not touchable, so this direct and familiar interaction is not an option as input. As outlined in the beginning, the smart glasses market has been growing in recent years. As a result of that research on interaction with smart glasses has increased. The aim of having a better experience interacting with smart glasses can lead to daily usage of it as seen today with smartphones (Lee and Hui 2018). So, if the users feel comfortable and have an intuitive and efficient experience with smart glasses this will increase the willingness to wear it. In this section, several approaches to interaction in augmented reality will be discussed. Highlighting articles that explored interaction methods for smart glasses.

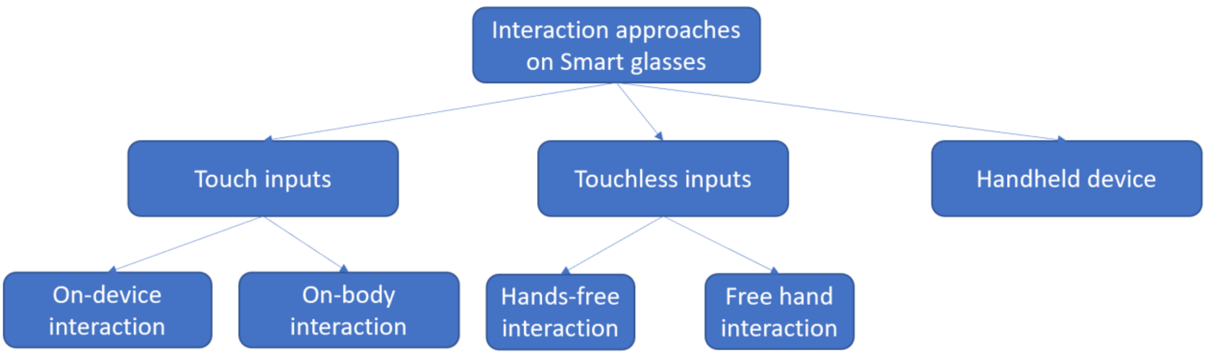


Figure 7 Classification of interaction approaches

As illustrated in Figure 7, the interaction approaches on smart glasses can be divided into three classes: touch, touchless and handheld device.

The touch inputs denote to a non-handheld touch input like gestures or touch in body surfaces and this also includes touching in wearables devices. This class is characterized by the presence of tactile feedback (Lee and Hui 2018).

The touchless inputs refer to a non-handheld and non-touch input. It includes mid-air gestures, head and body movements, gaze interaction and voice recognition. Unlike the touch input class, touchless inputs do not involve much tactile feedback, but this could be increased by devices (Lee and Hui 2018). The tactile feedback or tactile cue is the output that the user receives after executing an input. This output helps the user to know if the system received that input.

The handheld class refers to the inputs executed through a wired portable controller connected to the smart glasses. The biggest drawback of this interaction approach is that it requires users to carry the device on their hands and does not allow them to perform simultaneous tasks.

2.2.1. Interaction based on touch inputs

As schematized in Figure 8, the touch input can be on-device interaction or on-body interaction. In on-device interaction, the user can execute the input by touching or tapping devices; these devices can be incorporated in smart glasses or they can be wearable devices, therefore, they are considered physical forms of external devices.

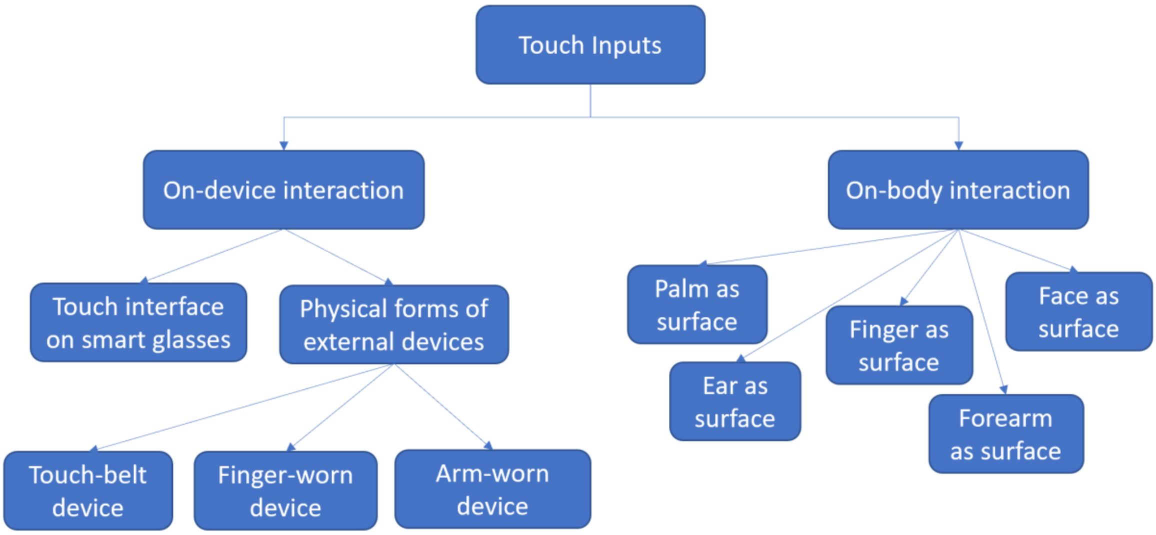


Figure 8 Touch Inputs Categories

One way of on-device interaction is by touching interface on smart glasses, this means that the user will touch the smart glass to interact. Some smart glasses as Google Glass already have a touchable surface on his frame. Yu et al. (Yu et al. 2016) and Grossman et al. (Grossman, Chen, and Fitzmaurice 2015) presented two different methods to text entry by using a side touchpad as input. While Yu et al. proposed a method of one-dimensional text input when making uni-stroke gestures (Figure 9), Grossman et al. proposed SwipeZone, Figure 10, a method that has a wider dimension and the touchpad is divided into three zones and use vertical gestures in the zone. Islam et al. (Islam et al. 2018) concerned with the user personal content, suggest an interaction to help in the authentication in smart glasses.

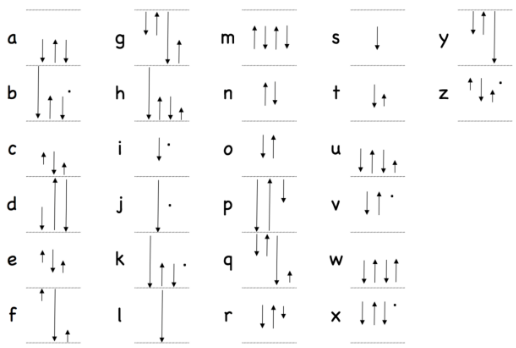


Figure 9 Uni-stroke method

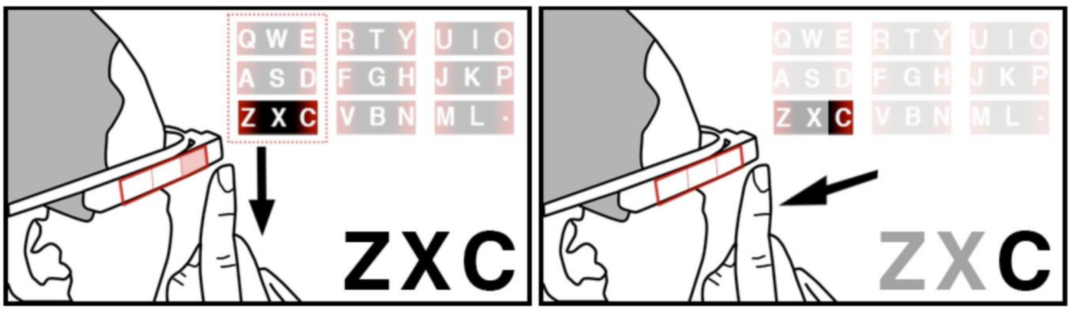


Figure 10 SwipeZone

Research (Lee and Hui 2018) argued that smart glasses have a small size and are lightweight so this is not an impediment for exploring complementary devices that can offer more methods to interact with smart glasses. These complementary devices known as wearables can be presented in many ways such as rings, wristbands, gloves, belts, and others.

The finger-worn devices have drawn attention because with it the user can make slight and discrete movements with only one hand. The LightRing (Kienzle and

Hinckley 2014) and TypingRing (Nirjon et al. 2015) are two input wearable rings. The LightRing tracks the 2D position of a finger on any surface through two sensors: the gyroscope and a proximity sensor. The input method used by Kienzle et al. consisted of two basic movements: rotate the hand around the wrist which meant left and right and flex the finger with the device which meant up and down. On the other hand, TypingRing has the aim to enable text input. This wearable ring is used on the user's middle finger and he can type on a surface. This is composed of three different sensors: the accelerometer, proximity sensor, and displacement sensor. Both the presented wearable rings need a surface to the input to be well executed. Although this could be a good thing to reduce fatigue it can be a drawback if the user can use it while he is moving. Ens et al. (Ens et al. 2016) remind that wearable technologies are being used to support small tasks or short-duration tasks but there are natural methods available to support everyday tasks. However, these techniques can cause some fatigue and have limited precision. With this motivation, Ens et al. proposed a ring with hand tracking by a head-worn depth camera. The author argues that the ring input provides precision and causes low fatigue, but the drawbacks are the need for the wearable device as well as the depth camera which not all devices are provided with.

In comparison, the arm-worn devices have more surface than the finger-worn devices and are less likely to make people feel uncomfortable because the wearing sensation is like wearing a watch. Ham et al. (Streitz and Markopoulos 2014) presented a wristband-type 3D input system. With this smart wristband, the user can easily interact with the smart glasses. This interaction can be made in various ways by using the finger to point and click, by rotating the wrist when the user intends to scroll the augmented information. To make a program switchover by quickly rotating the wrist and others.

It has become more common to use items of fashion as a wearable device for example belts or shoes. Dobbelstein et al. (Dobbelstein, Hock, and Rukzio 2015) presented an unobtrusive belt as an input device. This belt in addition to providing a large horizontal surface area allows the user to interact with both hands. Users can easily interact with it through familiar touch gestures like swipe and tap on both sides and allow to instantly stop the interaction leaving the user's hands-free.

These types of interaction methods bring some drawbacks like the necessity of having another device besides the smart glasses, having the time to put the device and if using if it fits the user's culture or environment. These reasons where the motivation to start searching for methods where the body is used as a surface. These body parts must be stable. The on-body input method has become a research target because it has a distinguishing characteristic which is the additional feedback mechanism. This is, the user can feel the tactile cue when the interaction is made on the skin surface. In a study performed by Weigel et al. (Weigel, Mehta, and Steimle 2014) it was possible to conclude that the preferred body part to interact with is the forearm as can be seen in Figure 11. Though, the author highlights that clothes can be a limitation since might lower accessibility.

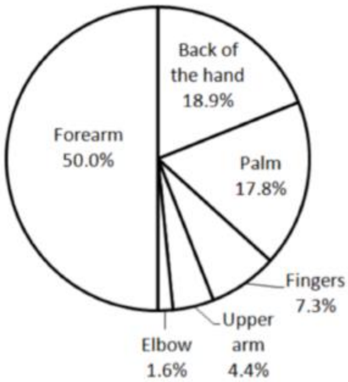


Figure 11 Location of user-defined gestures (Weigel, Mehta, and Steimle 2014)

Azai et al. (Azai et al. 2017) presented in a study a method to display a widget menu in the user's forearm. As already mentioned, the user has feedback when touching his forearm. Using his forearm as a surface, the user can interact with it by using touch and drag gestures. When the user executes a rotation gesture the camera mode changes or the volume of a music player is adjusted. To achieve this the real image from the forearm is acquired by a stereo camera and the information of a menu, a virtual content, is displayed by HMD.

The palm was preferred specifically for private interactions and tends to be associated with positive actions (Weigel, Mehta, and Steimle 2014). Due to this clear preference, some research has been made in a way to provide the best interaction method input using the palm as surface. PalmType (Wang, Chu, et al. 2015) and PalmGesture (Wang, Hsiu, et al. 2015) are two gesture interactions on palms that highlight the fact that the user can interact without looking at his hand palm. PalmType uses the user's dominant hand as an intuitive keyboard. To know the finger position, the user wears a wristband with sensors. PalmGesture, on the other hand, uses the palm as a surface to enable the user to draw strokes without having to look while drawing. With this kind of interaction, the user can open applications. This work aimed to present an alternative to all previous works where palm-interaction was only the standard gesture.

As seen in the previous examples, the finger is used as part of the palm as surface interaction. However, it is also possible to see a finger as a surface manipulated by the thumb. DigitSpace (Huang et al. 2016) is one of the examples that use a thumb-to-finger interface. With this method, users can easily interact in a precisely and discrete way. Another example of a finger as the surface is TIMMi (Yoon et al. 2015). Yoon et al. introduced a finger-worn device to help users to interact with smart glasses. This device senses the finger bending as well as the pressure made by the thumb. The authors remind that with this approach the gestures are accurate and subtle.

Other stable body parts have been investigated such as face as surface and ear as surface (Lissermann et al. 2013). The face can be considered as suitable for a natural interaction due to the facial area frequently touching so it makes it a discrete area to interact (Serrano, Ens, and Irani 2014). Still, for long-duration tasks, this kind of interaction method can cause some fatigue (Lee and Hui 2018). Serrano et al. (Serrano, Ens, and Irani 2014) declare that some gestures can be considered inappropriate but the author also argues that the face has a larger surface area and it is no less clothed than other areas.

2.2.2. Interaction based on touchless inputs

The interaction based on touchless inputs refers to a non-handheld and non-touch input. Examples of this type of interaction are gestures mid-air, head and body movements, voice recognition and interaction with the gaze. This category does not require much tactile cue feedback but this can be increased by the devices (Lee and Hui 2018).

As schematized in Figure 12 Touchless inputs categories, the interaction based on touchless inputs is mainly divided into two categories: hands-free and freehand.

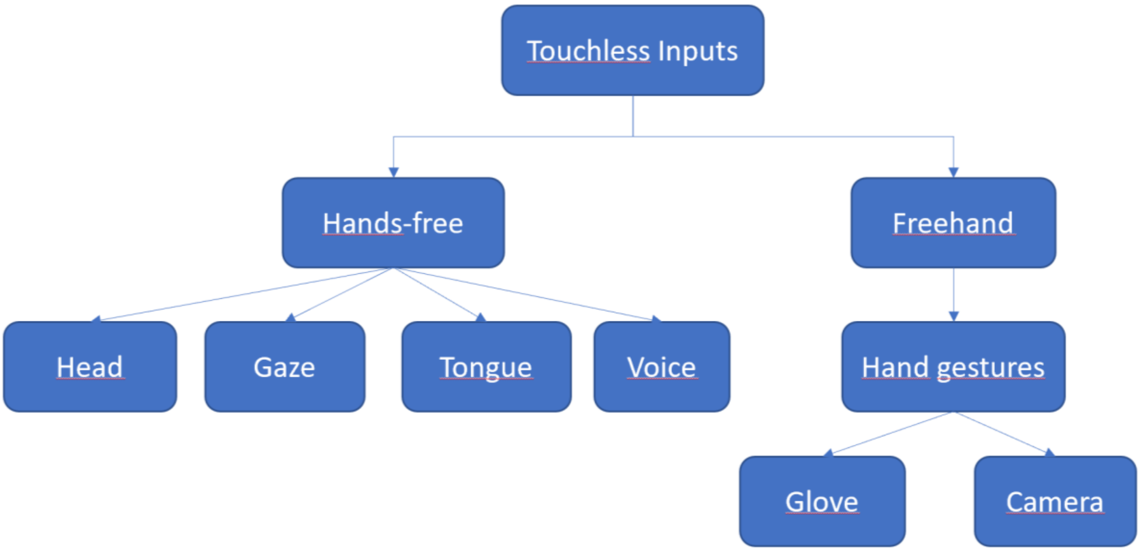


Figure 12 Touchless inputs categories

In the first type of interaction, hands-free, it enables the user to interact with the system without using their hands, which is an advantage. The most popular technology inside this category is voice recognition. Although some smart glasses companies start providing this kind of interaction, it brings some drawbacks. Yi et al. (Yi et al. 2016) remind some disadvantages, for example, the possibility of being accidentally activated in public, inconvenient in some scenarios like in a conference or meeting and an impossibility for disable users. In addition to that, can be very uncomfortable or difficult to interact in public spaces.

Besides voice recognition, inside the hands-free category, it is possible to interact with the device through head movement. One example of this interaction is GlassGesture (Yi et al. 2016). In the authors, opinion head gestures are intuitive and easy-to-use. This method is possible through the sensors, accelerometer, and gyroscope, integrated into smart glasses. Thereby is possible to measure all kinds of movements made by the user. There is research that combines more than one technique. For example, Ishimaru et al. (Ishimaru et al. 2014), proposed combining head motion and eye blink frequency. As reminded previously, disabilities can establish if the user can or cannot benefit from interaction techniques. To overcome this problem, systems based on tongue gesture detection have been proposed. However, Goel et al. explain that methods can include intrusive instrumentation in the user’s mouth. To overcome this big drawback, the authors proposed Tongue-in-Cheek. It is a wireless non-intrusive and non-contact facial gesture detection.

Tung et al. (Tung et al. 2015) conducted a study in which the smart glasses input was defined by the user. Through this study was possible to conclude that users preferred non-touch and non-handheld interactions. In this category, user's preferred to use the mid-air gestures rather than voice control, gaze interaction and head tilting.

Unlike voice control, gaze interaction and head tilting, mid-air gestures are a freehand interaction technique. Aigner et al. (Aigner et al. 2012) presented a gesture type classification scheme and classified the hand gestures into eight categories as shown in Figure 13.

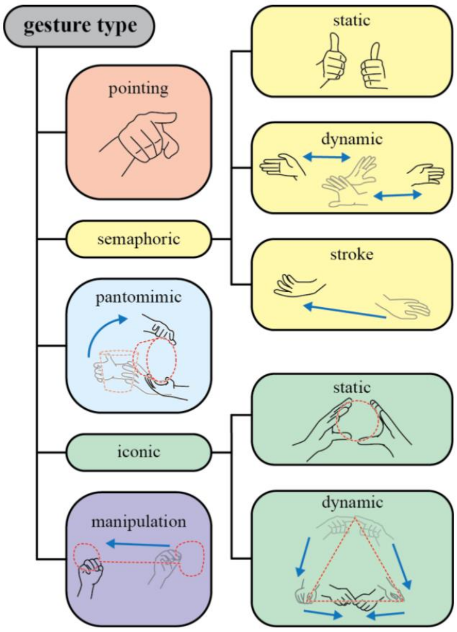


Figure 13 Gestures Classification (Aigner et al. 2012)

To recognize these gestures, sensors can be used to capture the hand's movement. In freehand interaction, the camera and the glove are the most common. Depth cameras or RGB cameras can be used which can track and recognize gestures. In this scenario besides hands, forearms can be also tracked, and it is the example of ShoeSense. It was created by Bailly et al. (Bailly et al. 2012) and has the purpose of recognizing relaxed and discreet gestures based on a depth camera. These gestures are a triangle, 3D Radial and finger count. The triangle gesture is a set of gestures formed by the arms. The finger count gesture consists of expressing a number by extending that number of fingers (Bailly et al. 2012). Minagawa et al. (Minagawa et al. 2015), also proposed a system with hand gesture in which the gestures such as "scissors", "paper", and "rock" were recognized by the camera using the Intel perceptual computing SDK. For the gestures to be captured by the camera, the gestures have to be made in front of the user's face. The ShoeSense allows the user to make discrete gestures because the camera is placed in the shoe.

Research has concluded that users usually prefer not to perform gestures in front of their faces in a public area due to the social acceptance issues and fatigue and they prefer in-air gestures in front of the torso (Tung et al. 2015).

Hsieh et al. (Hsieh et al. 2016) presented a haptic glove as an interaction technique and highlighted why a glove is a good form for hand tracking. The glove is equipped with multiple sensors and these can capture subtle movements. Besides that, the actuators can be placed for example in the finger and enable the user to receive tactile feedback which allows a richer experience. However, this interaction method also brings some challenges like the others previously explored. The more notable drawbacks are climate change and false positives. The hands are used to perform the most varieties of activities and tasks and the system cannot differentiate between the daily movements that are made and the movements to interact with the system (Hsieh et al. 2016).

2.2.3. Handheld Device

As mentioned above, the handheld interaction class refers to the inputs executed through a wired portable controller. Two examples of these approaches are the Epson Moverio BT-300 and Sony's SmartEyeglass, as shown in Figure 14 both have a wired portable controller.

Figure 14 (Left) Sony's SmartEyeglass (Right) Epson Moverio BT-300

The most evident drawback of this interaction approach is that it requires users to carry the device on their hands. This can prevent the user to perform another task while using smart glasses. Due to the separation between the controller and visual feedback, this may hinder the interaction or delay it.

2.2.4. Smart glasses Hardware

A few years ago, Google and Epson invested in smart glasses. Google glasses allow the user to do micro-interactions such as map navigation, photo or video capturing, and receiving notifications/messages (Lee and Hui 2017). Epson arrived at the smart glasses' world in 2011. Throughout the years Epson has made their devices less heavy, improved their process capability, extended battery life and enhanced camera capabilities (Toal 2018). Another company that recently launched its smart glasses was Microsoft Hololens.

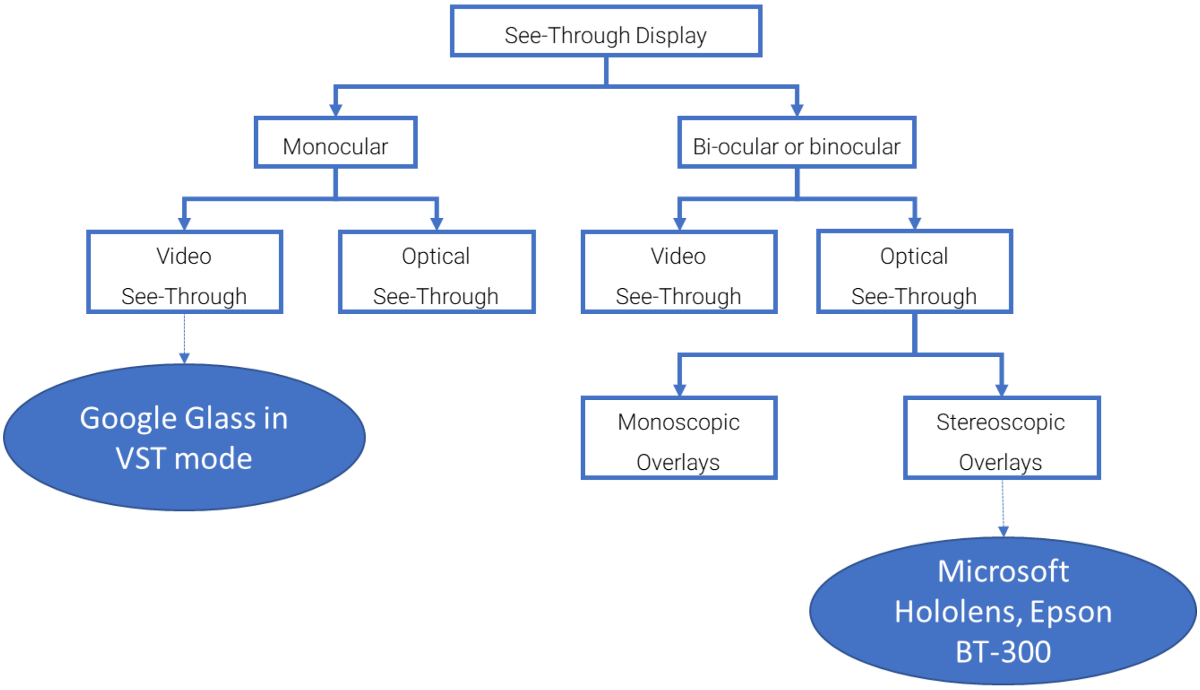


Figure 15 Categorization of see-through displays (Schmalstieg and Hollerer, n.d.)

As illustrated in Figure 15 the see-through display can be monocular or binocular. Google Glasses are a monocular optical see-through and, unlike Epson's smart glass, the see-through display is positioned in front of one eye as represented in

Figure 16. Epson's smart glasses, as shown in Figure 17, and they are a binocular see-through display the Epson Moverio BT-300 display has a high resolution.

Figure 16 Google Glass

Figure 17 Epson Moverio BT300

Figure 18 Microsoft Hololens

In terms of sensors, all three smart glasses are supplied with a camera, microphone, GPS, accelerometer, gyroscope, magnetometer, and light sensor. The biggest difference between these three smart glasses is interaction. Google Glasses has a trackpad as an input method, and this is on one side of the smart glasses. It also has the voice as an input method. Whilst Moverio BT-300 has an external controller to interact with the system. The interaction can be challenging because of the separation between the controller and the visual feedback. The Hololens from Microsoft has a deep camera which allows the detection of gestures. These smart glasses have also other two key forms of input which are voice and gaze. The gaze is one of the primary forms of targeting because with gaze the glasses can know where the user is looking at in the world.

Table 3 shows the comparison between the three different smart glasses. After comparing the three options, the choice for this project was the Moverio BT-300 from Epson. Although the Hololens was an interesting option, including all the same features as the other two models as well as a depth camera, the price was much higher and considered an important factor to make the decision. Between the Moverio BT-300 and the Google Glasses, there were a few differences, but the fact that Moverio has a binocular display, a more recent Android and a lower price was decided to make this choice.

	Epson Moverio BT-300	Google Glass	Microsoft Hololens
Optical System			
Display	Si-OLED (Silicon – Organic Light-Emitting Diode)	Prism projector	Stereoscopic head-mounted display
Display Size	0.43 inch wide panel (16:9)	640x360 pixels	2.3 megapixel widescreen
Screen Size	80 inches at 5 m 320 inches at 20m	25 inch at 2.4 m	
Operating System			
	Android 5.1	Android 4.4	Windows Mixed Reality
Sensors			
Camera	✓	✓	✓
Microphone	✓	✓	✓
GPS	✓	✓	✓
Accelerometer	✓	✓	✓
Gyroscope	✓	✓	✓
Magnetometer	✓	✓	✓
Light Sensor	✓	✓	✓
External Controller	✓		
Trackpad		✓	
Price	\$699 (~ 617€)[2]	\$1,800 (~1.599€)	\$3,000 (~2.648€)

Table 3 Comparison of Smart Glasses

[2] Prices based on the website www.aniwaa.com

3. Prototype

As explained in the previous chapter, the Moverio BT-300 was chosen due to its binocular display, the operative system, and the price. In the process of exploring these smart glasses, it was found several developer paths. The Epson developer website suggested working with Android Studio, Unity and HTML5.

In the Indoor scenario, two applications were developed, implementing distinct interaction techniques, each one using a different framework. Both frameworks have a module that allows the tracking of markers and the rendering. In the Indoor application, the smart glasses' camera is always capturing the image of the real world. If the framework identifies a marker, it starts to track the camera movement. Each marker has a different virtual object and when the correspondent marker is identified the virtual object is positioned in the scene.

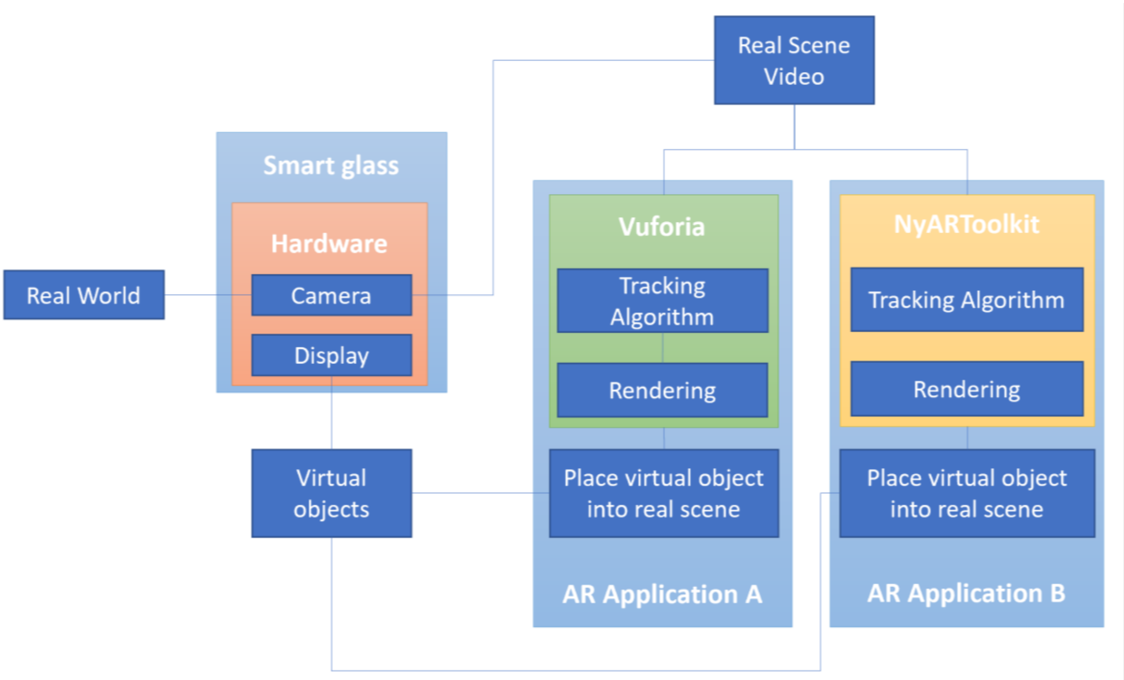


Figure 19 Marker-based system architecture for Indoor Application

After, the object is shown in the smart glasses display overlapped with the image of the real environment. The camera live image is not shown in the display otherwise the user would see the real world twice. In Figure 19, it is illustrated the prototype architecture.

In the outdoor application, no fiducial marker framework was needed. The application uses the sensors from the smart glasses to get the user location. After getting the position, FourSquare can be used to get all the Points-of-Interest (POIs) near the position. To get these POIs it is necessary to have a connection to the internet. For each POI it is calculated the distance and then it is placed a virtual object in the scene. If a POI is too far the virtual object will be smaller however if the POI is close the virtual object will be bigger.

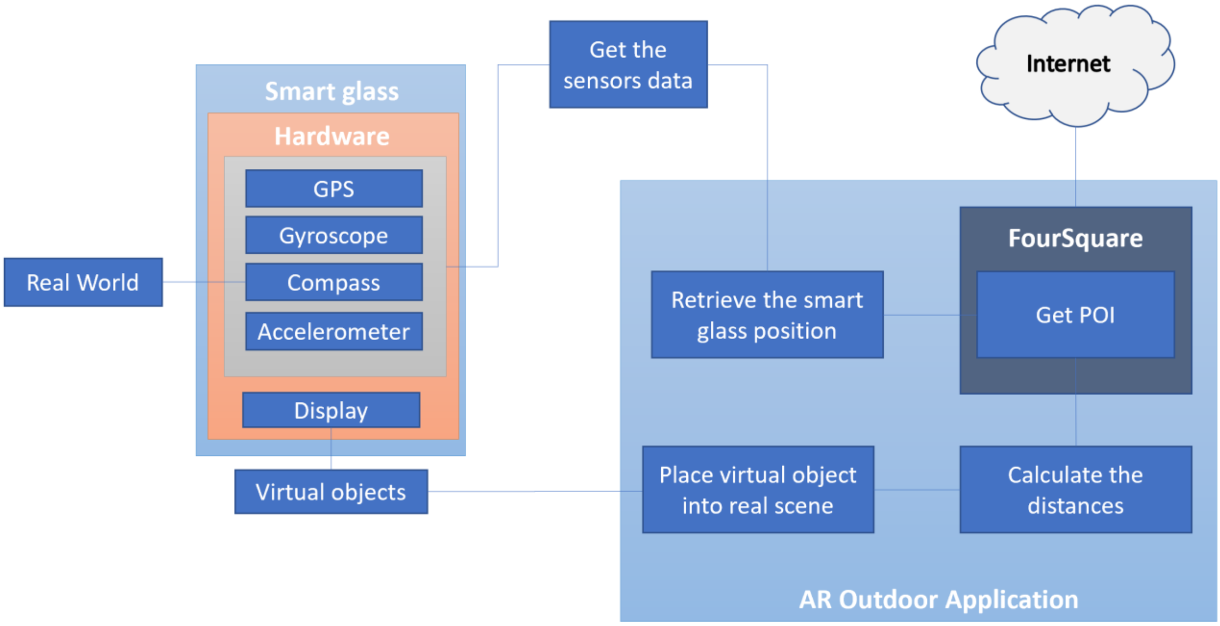


Figure 20 Markerless system architecture for Outdoor Application

The virtual objects are placed in the direction of the POI in the real scenario. So, these virtual identifiers will appear on top of the real POI. The architecture for the Outdoor application is illustrated in Figure 20.

In the scheme illustrated in Figure 21, it is explained how the two different interactions were implemented. It was created two different applications with a different interaction and each interaction used a different tool. In the application A, it was used Vuforia which allows creating virtual buttons overlapping the markers. In the application, B was used NyARToolkit, which also required the data from the sensors. In this application after placing the virtual objects, it is necessary to know the head position and for that reason, the data from the gyroscope is important to determine which marker the user is turning the head. After the marker choice, the marker is highlighted, and the user can open it by double-tap the tap detector sensor.

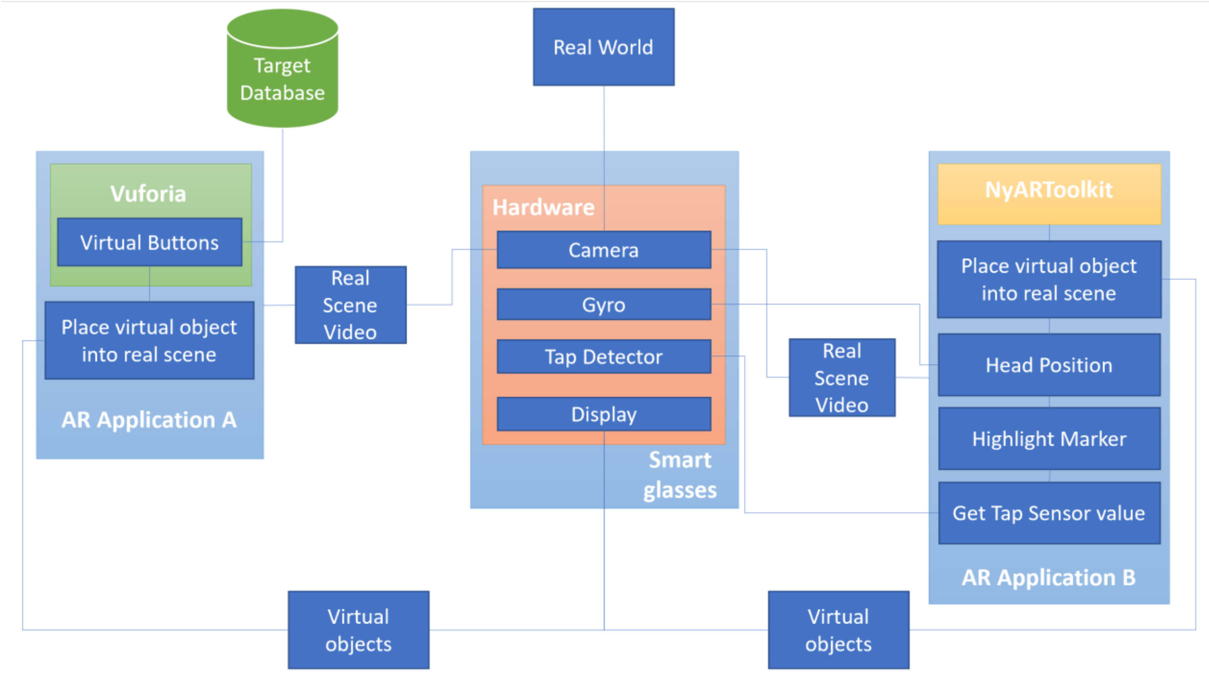


Figure 21 Architecture of the two applications with different interaction methods

3.1. Moverio BT-300

The EPSON Moverio BT-300 operating system is Android 5.1 Lollipop, API level is 22. This model includes specific features, like displaying content in 3D or switching between 3D and 2D, as well as the detection of tap interaction, i.e., the headset has a sensor that can detect if a user touches the headset. This latest feature allows the user to turn off or turn on the screen without using the controller. However, when using BT-300 special features or when building an application on these glasses, it is important to consider that it is necessary to change some definitions in the smart glasses to effectively implement and run those features, e.g., if the tap is implemented to perform other actions and not to turn on and off the screen. Before running the application, it is indispensable to turn off the special feature in the device's definitions.

This model of smart glasses is equipped with several sensors, both on the headset and on the controller that will be explored further ahead. Although it runs the Android OS system like smartphones, it is not compatible with Google store or Google applications like google maps. Epson's smart glasses have their own application store and it is not possible to use Google applications neither Google's services in customized applications.

In terms of hardware, it is provided with a CPU Intel Cherry Trail Atom x5 1.44GHz Quad Core. These glasses are binoculars and the display is composed of the Si-OLED (organic light-emitting diode) technology. This enables the transparency of the screen, delivering a good quality image and a high contrast with the real world. The transparency of the screen and the good quality image delivers a good augmented reality experience. The resolution of the display is 1280RGB x720 and the screen orientation is fixed in the landscape. It can project images on 80 inches floating screen, 5 meters in front of the user's eyes. The Moverio-BT300 also has a frontal camera with 5 Megapixels. When the camera is being used it has a green flashing led light. The battery has a duration of approximately 6 hours.

The see-through over the glass Moverio BT-300 smart glasses are composed with a headset and a controller. Both components are separated, but wired, and include individual sensors. In the controller, it is possible to find the accelerometer, geomagnetic sensor, gyroscope sensor, and the rotation vector sensor. The headset, besides the sensors mentioned above, has also the azimuth detection, illumination sensor, temperature sensor, gravity sensor, a linear acceleration sensor, and the tap detector. For each sensor, in the headset as well in the controller, the output value is delivered according to an X, Y and Z axis. The coordinate axis in the controller and the headset is as shown in Figure 22 and Figure 23.

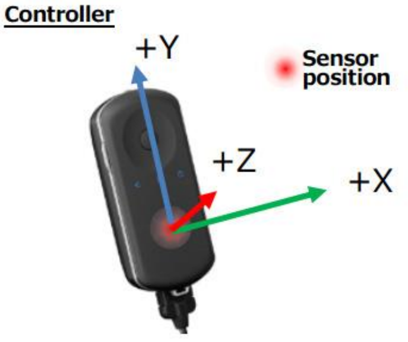


Figure 22 Coordinate axis in Controller

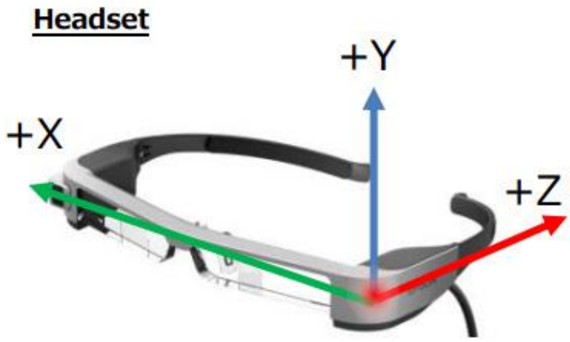


Figure 23 Coordinate Axis in the headset

3.2. Indoors

Fiducial markers have a good purpose to fill the needs of indoor environments. Although markers are popular for adding 3D objects in the world, they can be used to other types of information. In a future scenario, there is a possibility to see more people using smart glasses as happens now with smartphones. It is important to not restrict the use of augmented reality and smart glasses to 3D objects but extend to information. Similar to what is proposed by Ryu and Park (Ryu and Park 2016), they use text documents or books to augment them with virtual content.

In chapter 2, it was highlighted that Hirokazu Kato developed the ARToolkit and the markers like Hiro and Kanji were examples of markers with a square black border and a pattern inside. Besides using these two popular markers and an extra one was added. Using the same black border but instead having a symbol or letters it has black and white squares as shown in Figure 24 (on the right).

Figure 24 Markers with black square border

All these markers follow the same type of pattern. The black square border has half of the inner marker. The white region was already mentioned, is used to identify. However, it was interesting to understand how the system could behave with different types of markers. For that reason, besides the ones chosen above, it was also used one that is a logo, one handmade and other with a difference as shown in Figure 25.

Figure 25 Logo, on the left, handmade in the middle and different one on the right

3.2.1. Solution with Vuforia

As mentioned before, the Vuforia is a very popular toolkit that allows creating augmented reality applications for smartphones, tablets, and eyewear. With this toolkit, it is possible to add the computer vision functionality and give the user a more realistic AR experience. Also, it is provided with documentation for different platforms like Android and/or Unity. This tool allows recognizing model targets, image targets, object targets, multi-targets, and cylinder-targets. Although this tool has its markers, known as VuMarks, it was the intention of this study to have different toolkits with the same markers to better understand how well the tracking works. The Vuforia allows the developer to create his markers, and all it must do is go to the target manager on the developer website and add a database. For this database five target images as shown in Figure 26.

☐	Target Name	Type	Rating	Status
☐	8	Single Image	☆☆☆☆☆	Active
☐	1	Single Image	★★☆☆☆	Active
☐	jsf	Single Image	★★★★☆	Active
☐	7	Single Image	★★☆☆☆	Active
☐	Hiro	Single Image	★★★★☆	Active

Figure 26 Target Images in Vuforia's Database

One advantage of this feature is to see how well the Vuforia will track the marker. For example, as shown in Figure 26, the first marker does not have any rating star while the last image target has four rating stars which means that the Vuforia will be able to detect easily the last one and probably would not detect the first one. When Marker is selected it is possible to see the rating points from the image, as shown in Figure 27. This means that these are the points that Vuforia needs to find to detect and track the image target. These points are very important especially if a virtual point is associated with an image target.

Figure 27 Feature points from Hiro Image Target

After adding all the markers, it is possible to download the database. When the project was started in unity one thing that was always important in all projects was to make sure that the camera was working, and the virtual object lays on the real object. However, the camera image should not appear to the user because in this case, the user would have the image of the world twice. Before adding the marker's database, previously downloaded from the Vuforia's website, to unity, it was searched about the best way to create an application for the smart glasses and using Vuforia as a tracking system. It was found in the unity asset store a Vuforia Digital Eyewear Sample that could help to achieve what was set. To do this and all projects with Vuforia it was important to create a License Key on Vuforia's website and add it into the Vuforia Configuration in Unity (Appendix 1 - Figure 58 A).

To use the marker's database, previously created on Vuforia's website, it is necessary to add it to the Unity through the Vuforia Configuration (Appendix 1 Vuforia configuration). In the digital eyewear, it was specified that the device type was digital eyewear. However, when the image captured by the smart glasses' camera was shown in the display, the user was seeing the real world duplicated. So, this asset from the unity asset store was not achieving the goal. Initially one of the potential solutions was to replace the Custom/VideoBackground with a different file that allows the capture of the image and that image was not displayed to the user. But this solution leads only to a black display and the camera did not work; at least did not detect the markers that were desired. The solution for this problem was changing the Digital Eyewear and set the properties as present in Appendix 1 Vuforia configuration to achieve transparency of the display and only display the virtual objects.

In this project three different markers will be used: one of the markers will have an object, while the other two will have two virtual buttons. These virtual buttons allow the user to use their hands and pass in front of the camera as if they would press a real button. To detect the markers from the database created in Vuforia's website it is important to add a new image in the hierarchy, as shown in Figure 28.

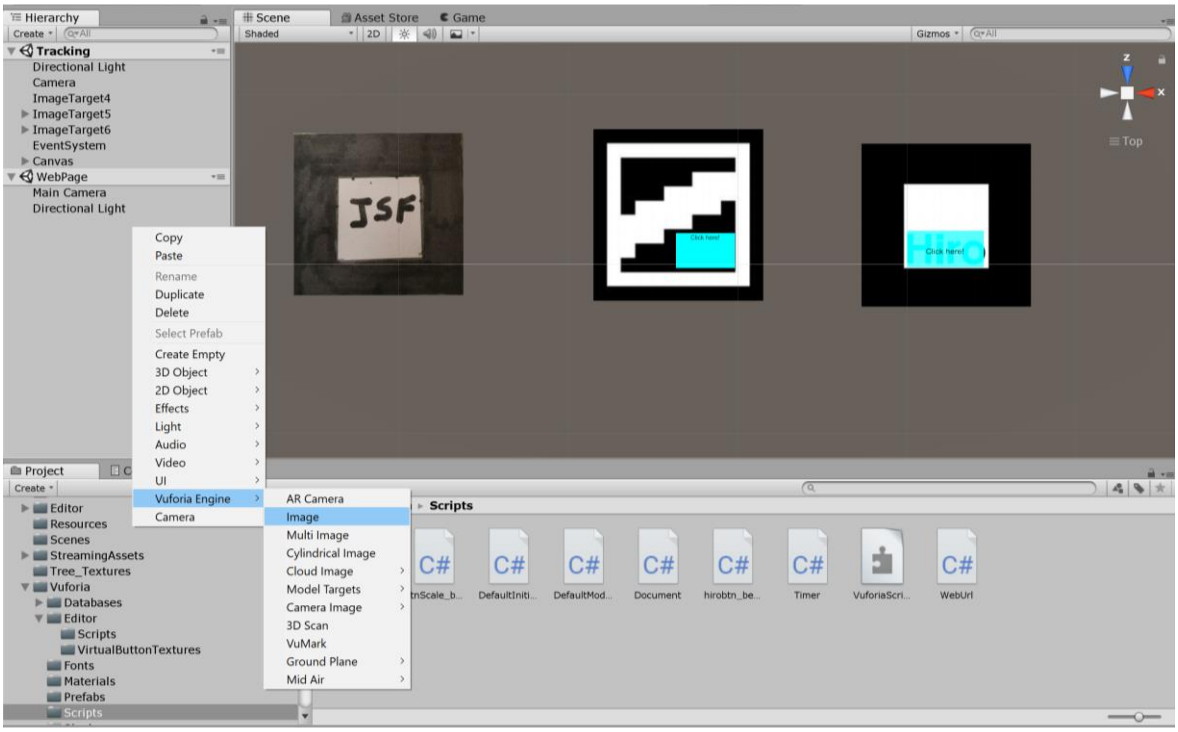


Figure 28 Add a new Image target

After creating a new image target it is necessary to specify in the unity inspector area the name of the image target from the database. In the properties of that image target, it is possible to switch between images that are in the database.

The three chosen markers have different ratings, and this influences the time that takes to identify the marker, as shown in Figure 26. To better understand the effect of ratting on track, a small test was created to understand how much time it takes for a marker to be detected. The test was divided into three parts. In the first one, all markers have the size of approximately an A4 page. In the second one, the size was reduced to ½ A4 page and in the third to $^{1}/_{3}$ A4 page. Apart from the different tracking times, this would help to understand if the smart glasses would detect better with large, medium or small markers. All the markers were placed with the same environmental conditions and each condition was tested six times and then the average was made. However, it is important to highlight that these times can differ because of the light conditions, markers conditions and position.

As expected, the test results showed that it is easier for the smart glasses camera to detected and track when the markers are larger (A4 page size). However, when it was used small markers the smart glasses can detect and track the Hiro marker more easily than the rest. Comparing the times between the three markers it is possible to understand that JSF and Third have tracking times closer than the Hiro.

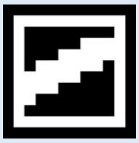
Markers	1st (size: A4 page) N=6	2^{nd} (size ½ A4 page) N=6	3^{rd} (size $^{1}/_{3}$ A4 page) N=6
Hiro 	M = 3.83 sec. (SD = 2.48)	M = 13.83 sec. (SD = 4.81)	M = 18.33 sec. (SD = 4.15)
JSF 	M = 8.50 sec. (SD = 3.35)	M = 19.17 sec. (SD = 1.77)	M: 26.83 sec. (SD = 8.73)
Third 	M = 11.67 sec. (SD = 6.37)	M = 22.33 sec. (SD = 5.15)	M = 28.67 sec. (SD = 11.90)

Table 4 Comparison of the average of tracking times (in seconds) and the standard deviations

After verified this difference, it would make more sense to set the virtual buttons in the markers that the tracking times are fastest. Otherwise, the user would be able to see a virtual button in one marker and wait for the other show.

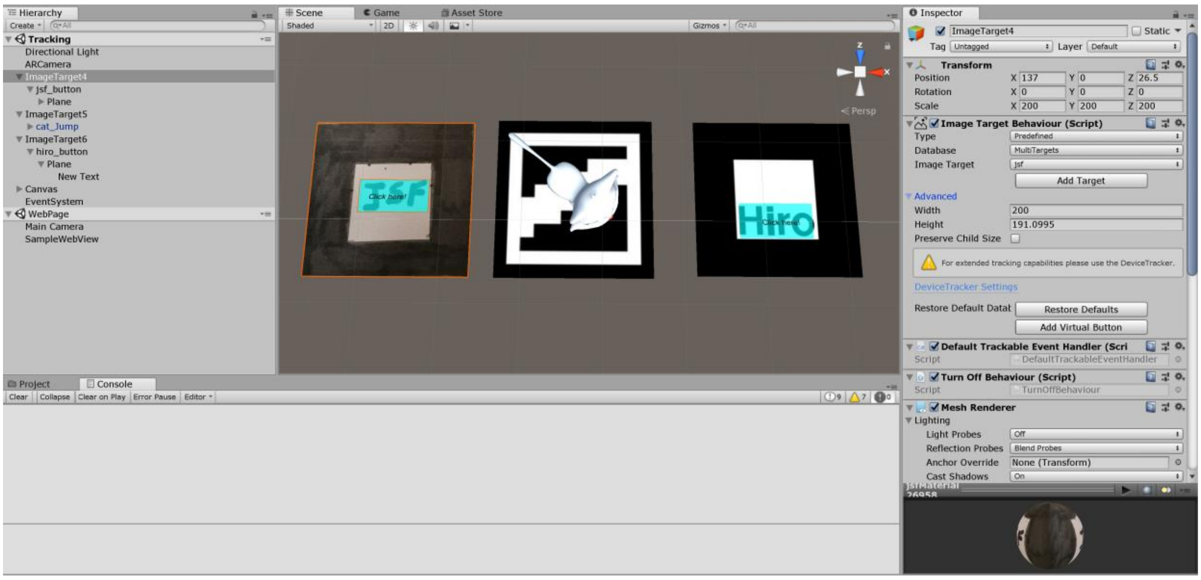


Figure 29 Placing virtual buttons

To create a virtual button the first step is to create a new image target. The second step is to select this game object and, in the inspector, choose the image from the database. After choosing one of the images from the database to place a virtual button the Image Target Behaviour must select the advanced properties and select the "add virtual button" button, as shown in Figure 29. After these steps, it is necessary to determine the behaviour of this virtual button. So, it is created a script with the behaviour of this virtual button. This script is attached to the virtual button created. Each marker would have a different page associated. And if the button was pressed the correspondent page would open.

3.2.2. Solution with NyARToolkit

The NyARToolkit project is provided with several scenarios and following the tutorial provided on the website it becomes clearer the way this tool works. Like in Vuforia, NyARToolkit allows creating new markers by using the NyAR NftFileGenerator as shown in Figure 30. After import an image into the NyAR NftFileGenerator program it is possible to make the feature set and then export it into a NyARTK NFT dataset file.

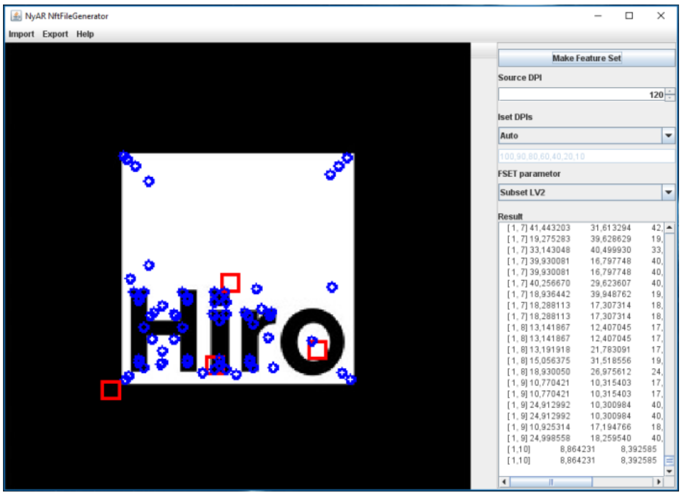


Figure 30 Generate a nft marker

In the different scenarios provided by NyARToolkit project, there are two different ways of detecting the markers. It is possible to track markers with the NyARUnityMarkerSystem and with the NyARUnityNftSystem, the former uses a byte file from the marker to loads a marker from the texture of the camera, while the latter uses files exported by NyAR NftFileGenerator. However, the behavior between both can be slightly different.

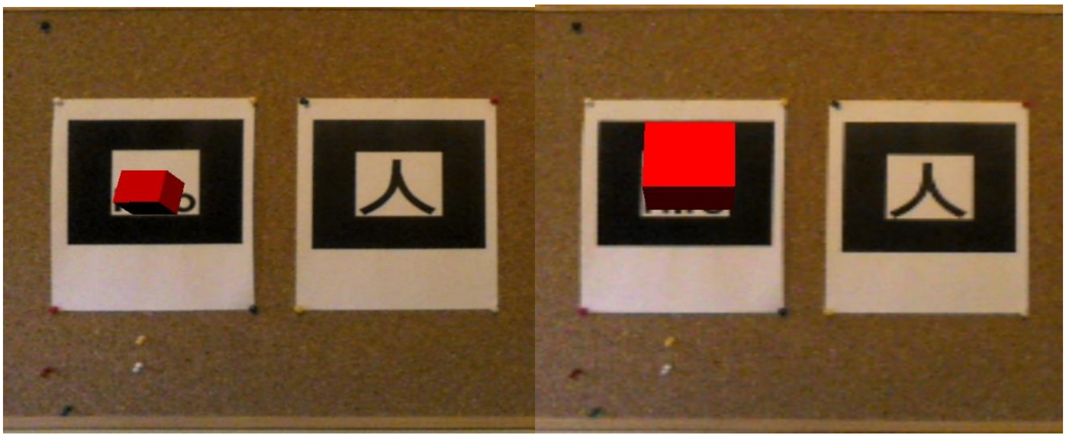


Figure 31 Tracking with NyARUnityNftSystem (left) Tracking with NyARUnityMarkerSystem (right)

As shown in Figure 31, the *NyARUnityMarkerSystem* can be more stable than *NyARUnityNftSystem*. Even with the camera standing still, the tracking system *NyARUnityNft* displays the object always in movement with a slight rotation. While the system on right keep the cube still and only with a quick movement of the smart glasses the cube stands in the initial position and then appears on the top of the marker. All the markers used with the NyARUnityNftSystem suffer the same result included the marker suggested in the tutorial, shown in Figure 32.

Figure 32 Unity Chan Marker

The constant movement of one object can be tolerated by the user. However, if there is more than one object can be very disturbing. For this reason, it was better to use the *NyARUnityMarkerSystem* to detect and track the markers. Although this system works better because it can be more stable it also brings some disadvantages. To add a new marker to this system is necessary to add a byte file into the resources folder and add this resource to the memoryStream through the script. But there is no information on how these files can be generated.

The bytes files have an appearance like the file present in Appendix 2 Bytes File(Figure 60). In the corner, it is possible to visualize the entire file. This type of file helps the marker system to track the position of the marker. However, as mentioned, there is no information about the creation and all the attempts lead to a blank space or a file with special characters. The ARToolkit package has a system that generates markers, and these files were called patt files. These patt files are similar to the bytes files, but it is not easy to create them without the generated system particularly markers that have letters as a pattern. If the pattern is white and black squares it is simpler as shown in Appendix 2 (Figure 61 - the patt file refers to the third marker with black square border in Figure 24).

Occasionally, this system shows the object when the marker is not showing up. This leads to believe that can cause some false positives. For that reason, to minimize this effect when the marker stops to being detected the size of the object would be resized to zero. If the false, positives happen the object would not disturb the user.

The goal of this application would have something like the Vuforia application so that both can be compared. In this case, instead of having a button the marker would be selected. It would be important to add to this solution more than one marker. The application would not detect the marker and open the page. The application would detect all the markers captured by the camera and wait for the user interaction to open the selected marker. In a way to accomplish a similar scenario, the solution would be to add a frame to both markers, tracked by the *NyARUnityMarkerSystem,* and add an object to another marker tracked by the system *NyARUnityNftSystem.*

In the beginning, the impossibility of tracking the two types of markers was detected but several potential solutions for this problem were explored. The simplest solution would be to use exclusively the NFT markers, markers created with the *NftFileGenerator*, but as shown above, these markers are more unstable and work better with drawings or another type of markers like the one provided by *nyartoolkit* the unity chan. So, when this solution was executed what happened was that the frames on the markers were always moving and shaking. It would be best using a more stable solution.

The second solution for this problem was the creation of two different scripts attached to the camera to track the two different types of markers. The result was the overlapping of the memory stream. Another solution was using threads to solve this problem. However, the result was similar to the previous one. The only solution that could support this would be to first allow the system to track one type of marker and after detecting the marker this system would switch the tracking system to the other. The system would work as shown in Figure 33.

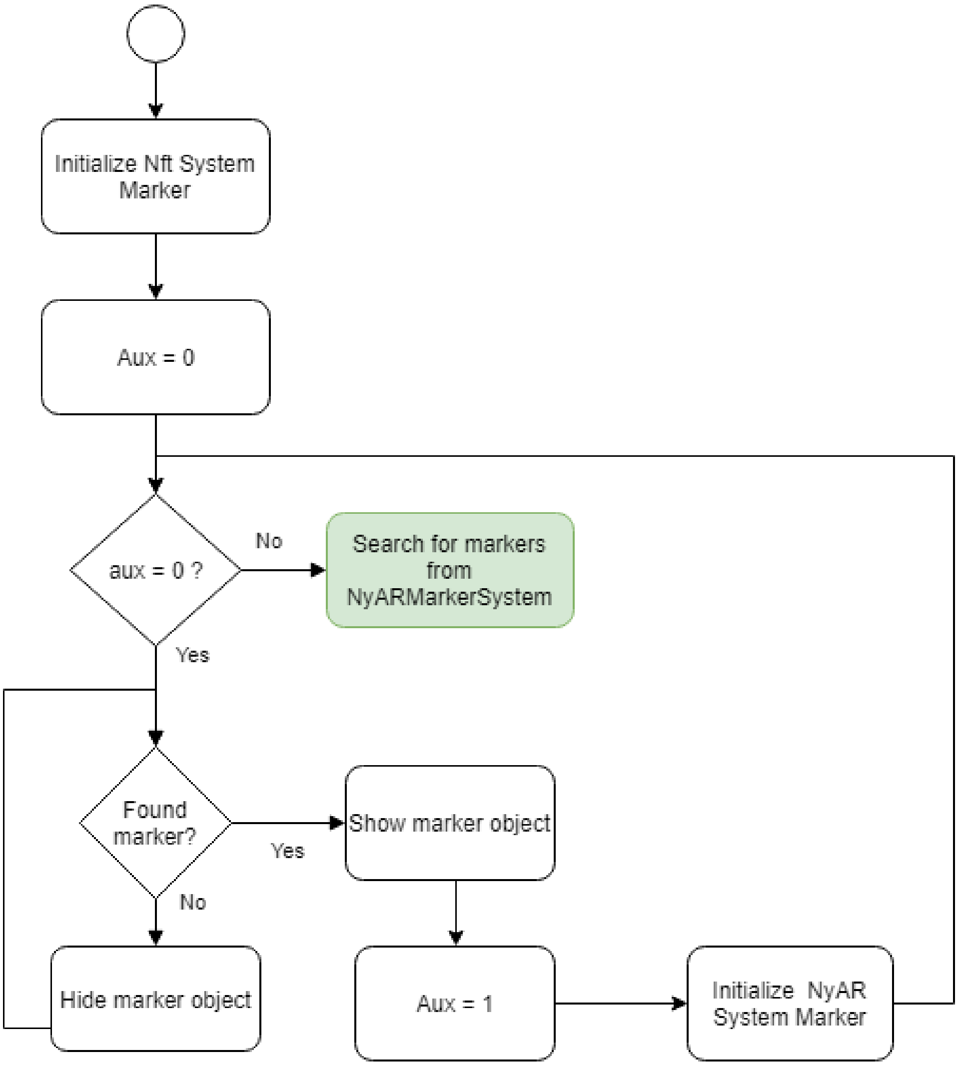


Figure 33 Flowchart of both tracking systems working

The first marker that would be detected by the system would be an object placed in the world so that the user could identify it, as shown in

Figure 34, and then the other two markers would be placed with frames as shown in Figure 35.

Figure 34 Nft image tracked by the smartglasses

Both images are taken from the smart glasses with the projection of the active camera. However, the projection is removed from the display so that the user does not see the physical world as well as the image captured by the camera.

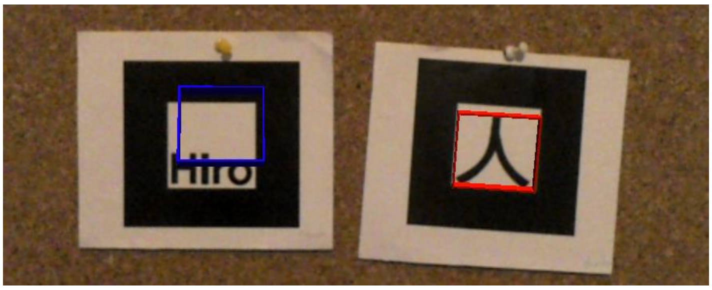


Figure 35 Two markers tracked simultaneously

The common drawback in all the solutions described above is having the camera constantly capturing, which can raise privacy issues. However, some research has been done in this field which is the case of the work from Koelle et al (Koelle et al. 2018).

3.2.3. Webpage inside the Application

In both previous applications, the goal is to present relevant information. That information can be a photo, a webpage or something else. For example, the tourists' guides instead of having too much information, could have suggestions of restaurants and select could open the restaurant’s page and with the point interest could opening a gallery of images of that specific place.

In early phases, by selecting each marker it would open a webpage outside of the application. In unity, there is a function that easily allows the application to open a webpage outside of the application. However, it was only implemented in the early stages of the process in all the applications. The drawback of this was the difficulty of return to the app. This was not satisfying because it would force always the user to close the browser and re-open the application. This re-open would require more than one interaction and it was a goal to keep the interaction simple as well quick.

So, for that reason in all the applications, when the webpage is open it happened inside the application allowing the user to easily return. To achieve this, it was necessary to open the webpage inside of the application through WebView. Thanks to a plugin that overlays WebView on Unity[3] it was possible, with some changes, to accomplish what was aimed, as seen in Figure 36. One of the things that were changed were some obsolete properties that were not needed. Also, it was essential to making a slight change in the code such that it could receive a different URL depending on the selected marker.

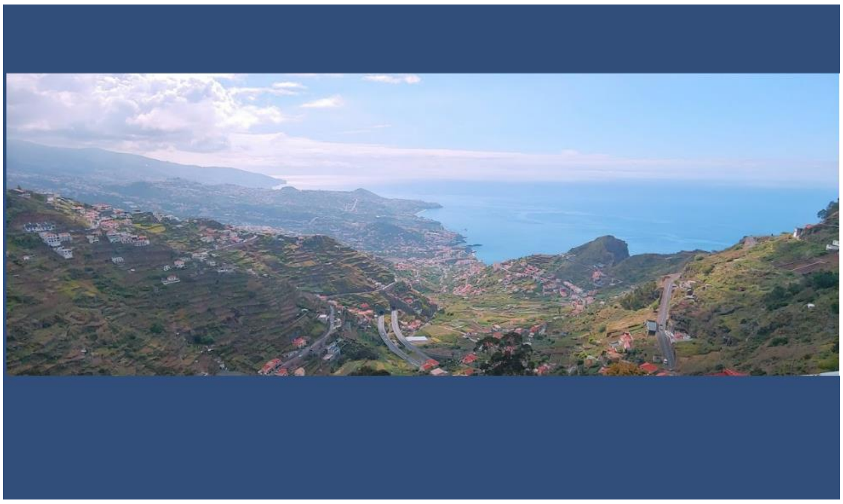


Figure 36 WebView with Image

3.3. Outdoor

The solution Markers can help in displaying information indoor, but as already mentioned, it would be difficult to fill the world with markers. All the devices are equipped with several sensors that can help to know the user position in the world. The goal of this solution outdoor would be giving information about the user surroundings like usually it is done with smartphones. The use of smartphones can distract the user from his surrounding and the use of smart glasses could avoid that.

[3] https://github.com/gree/unity-webview/blob/master/README.md

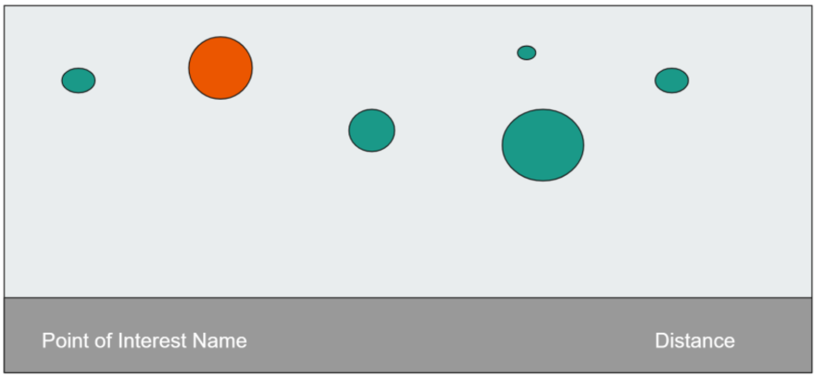


Figure 37 Outdoor Solution Scheme

The aim of this application would be giving the user the possibility to know more about the Point-of-Interest (POI) within a specific radius. To achieve this would be necessary to know the user position and which direction he is looking for. In a way to obtain this kind of data, it was used sensors like GPS, accelerometer, gyroscope and magnetic sensor also known as a compass.

With the user location, all POIs would appear like a sphere in the FOV of the user. As shown in Figure 37, the size of the sphere would be smaller when the POI was more distance and bigger when the POI was closer. The process of this application would be receiving the user's GPS points and then with this information get all the POI. Due to incompatible between Moverio BT-300 and Google Play, would be necessary to get this information from another source.

In the process of understanding all the different important aspects of building an AR outdoor application, the article of Rosa and Guitiérrez (Rodríguez-Rosa and Martín-Gutiérrez 2013) helped in the process of building the project.

The first step was to look for a free API to retrieve all the points of interest near the user. In the end, the FourSquare was the API chosen because all the others either were paid or did not cover the intent area or did not have many POIs.

The second step was understanding how the smart glasses would know the user's location and where he was looking. To retrieve the GPS coordinates from the smart glasses all it was necessary was to access the sensor. However, the smart glasses know where the user was looking at the environment was important to use the gyroscope. In this process, the article by Rosa and Guitiérrez helps to create a scenario where the camera is in the center of the world as seen in Figure 38.

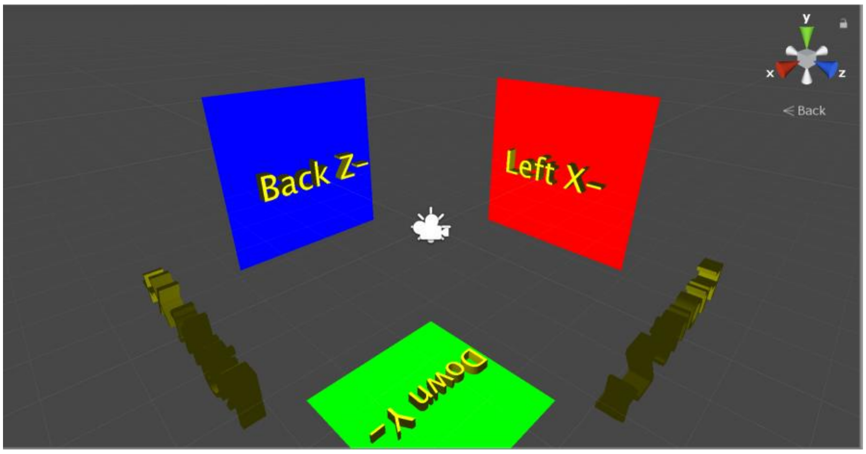


Figure 38 Setting the scenario

After this setup, it would be used the location of GPS to filter the POI near the user. For every POI it was created a new object in the world. The position of this new object is determined by the distance and position relative to the user's coordinates. The objects have a fixed position and only show in the user's FOV if he is looking in the direction of the POI.

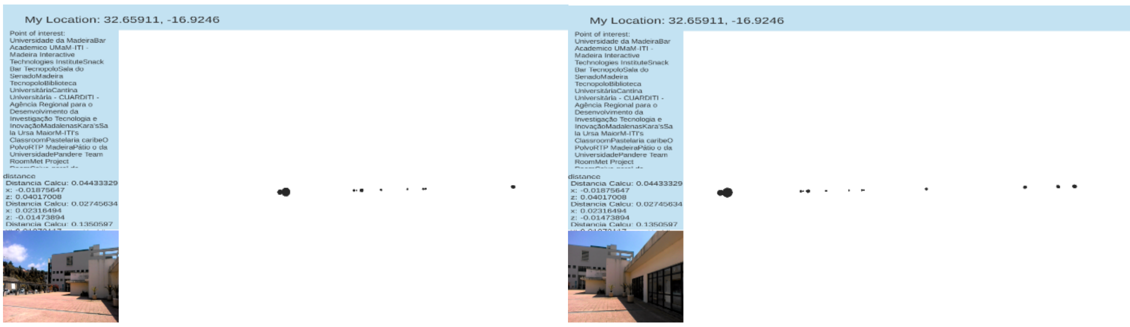


Figure 39 Same place looking in different directions

The user is in the same place but looking in a different direction, as shown in Figure 39. The POIs are the same, the distance of them is the same but the points (objects) that show in the display are different. The size of the object can help the user to know if the POI is closer or further.

3.4. Interaction

As mentioned in section 3.2, two solutions for indoor were implemented using two different toolkits and both with different interaction methods: one method allows the direct selection of a virtual button with hand gestures and uses Vuforia, the other uses the double-tap and head movement for selection and uses the NyARtoolkit. However, it was important for this project to understand if the solutions implemented were more intuitive or convenient than the interaction method default offered by the smart glasses, a handheld controller. For this reason, a third scenario was built where the markers are detected by the NyARtoolkit but selected through the handheld device.

3.4.1. Gestures

In the previous section, it was explained how the system of tracking would work. After changing the tracking system, the system would check the different possibilities. In the early stages of the project, several options were taken into account. One of the possibilities of interaction was gestures. However, most of the literature points to depth cameras and Moverio BT-300 is not equipped with one.

The first solution uses Vuforia capabilities to track the markers and put virtual buttons on top of that images shown in Figure 40. Through movements, as if the user would click a real button[4]. This kind of interaction brings the challenge to go back. After a marker is being clicked there is no way of going back with the same kind of interaction. The first reason is when a web view is open the camera stops to track. The second reason is if using the button was the only method of interaction the gesture of pressing it could open but the user would need to stand with the same position to keep the page open. So, when the user stops pressing the button would close the page.

Figure 40 Three markers and the virtual objects

3.4.2. On-device Interaction and Head Movement

As mentioned in the 3.1 section, these smart glasses had a headset sensor that can detect a double tap in the smart glasses headset's frame. The GlassPass (Islam

[4] https://www.youtube.com/watch?v=iOkyynlMi6M&feature=youtu.be

et al. 2018) inspired the use of double-tap function in this solution In the work of Islam et al. is possible to see how discrete this method can be to unlock the smart glasses when these are locked with a pin. So, this double-tap interaction could be used in the application to open or close the correspondent webpage. This type of interaction belongs to touch inputs on the device.

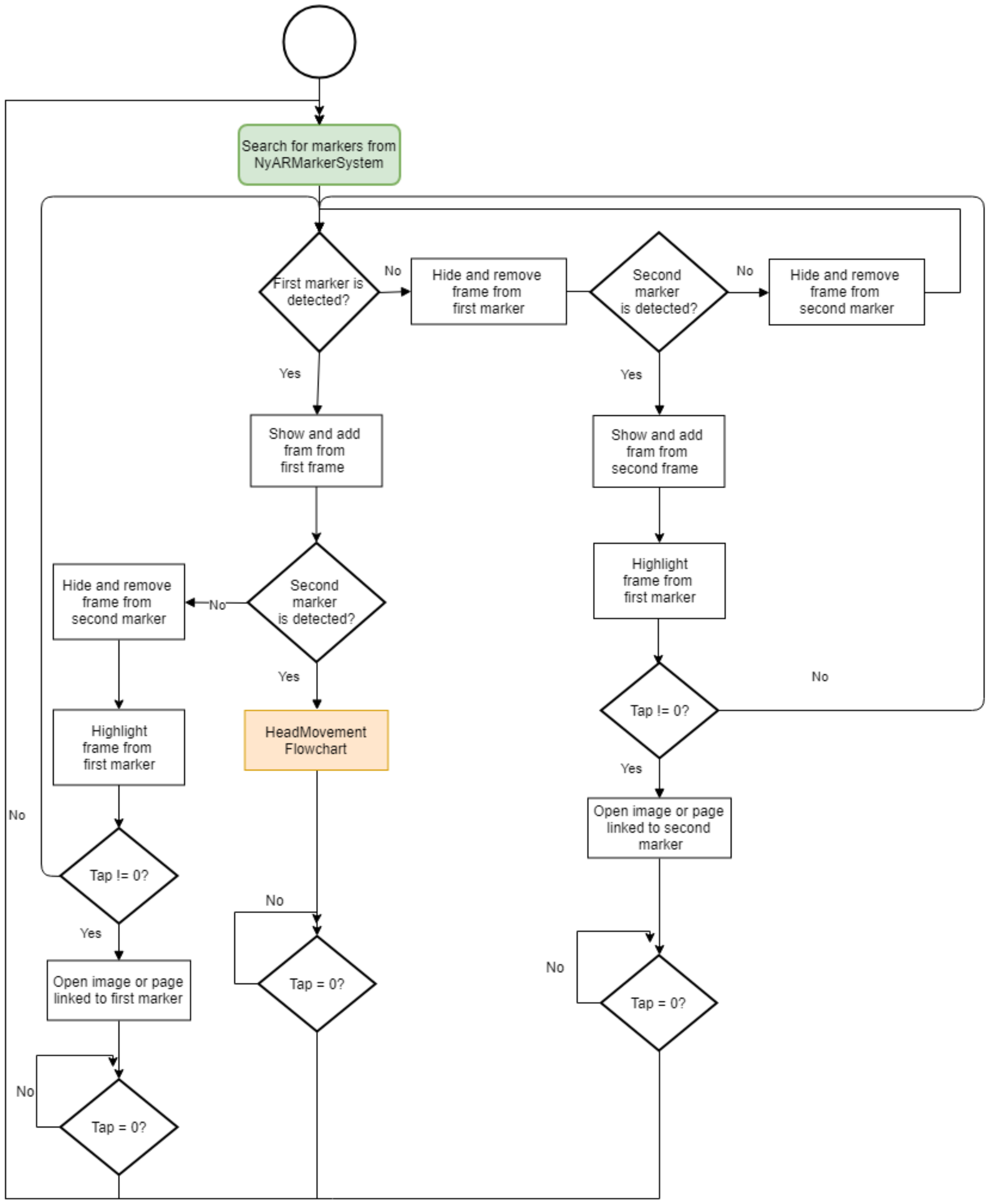


Figure 41 Interaction in NyARToolkit application

A scenario of one single marker could lead the user to believe that the system would open the page by itself. Though the aim of these markers was a trigger of action around the user but does not take the user's decision off the equation. So, the system can track more than one marker and the user, with small interactions, can interact with it. With this method, the system is kept simple and less intrusive. As shown in Figure 41, the system starts by looking for the first marker and then starts looking for the second.

In this application, there are two interaction methods. The first one is the double function of the Moverio BT-300 smart glasses as already mentioned. This method allows the user to open the webpage related to a specific marker. However, when the system detects and tracks two markers simultaneously, by tapping the headset's frame the system would not know which marker the user would like to open.

The initial approach was to select the marker that was closer to the camera, but it could withdraw the user's decision. Another solution that was considered, it would be to select the marker that would be more centered in the image captured by the camera. Yet this solution could be not intuitive for the user and in some circumstances when one marker is placed in the middle the other one sometimes is not tracked very well.

Another possibility to solve this would use the controller, by shaking it, it would select the desired marker. However, it was intended to keep this solution with a touchless input. Ideally, if the double-tap was made on the left side of the headset the user would want the left marker and if the tap was made on the right the user would want the marker on the right. But there is no way to know from each side the double-tap is made.

The solution that was found was by using the head movement to track the selection of the user[5]. In Figure 35, it is possible to see the first moment when the markers are both detected. Each marker has a different color frame to better identify it and the frame would change color to yellow, as shown in Figure 42 and Figure 43. To make the change between these so that the user can detect the difference, it was used coroutines to allow some waiting time. At the begging, it was considered the use of threads however some of the operations made to make changes in the position of objects and changing colors are not allowed inside of a thread, only in the main.

[5] https://www.youtube.com/watch?v=ZoZBn5gfpy4&feature=youtu.be

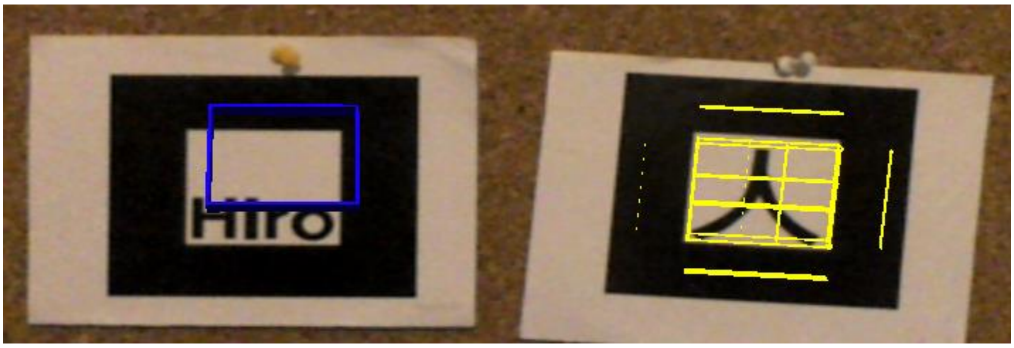


Figure 42 Head movement to the right

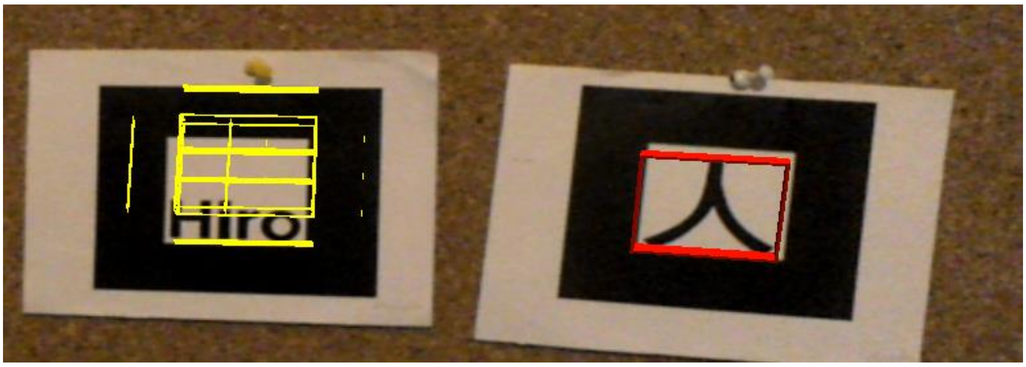


Figure 43 Head movement to the left

At first, it was used the Moverio BT-300 headset's accelerometer sensor to track the movement of the head. After some tests, it was noticed that the values took a little time and the values were inaccurate. The measure of acceleration is influenced by the force of gravity. In a way to overcome this, it was used the gyroscope to track the head movement.

The gyroscope is more accurate and has a short response time. To get the 3D orientation of the device it was used the Gyro.attitude. And then with this and using, Euler angles were possible to represent the spatial orientation of any frame.

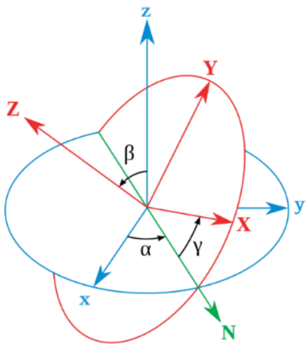


Figure 44 Proper Euler angles geometrical definition ('Euler Angles' 2019)

It was important to look for the values from the coordinate x. This would tell if the movement that was made was to the left or the right. But to make this comparison it is important to have the initial value. It was noticed through the tests that sometimes the first value tends to take a little time to update so every time the first value was kept in a variable the value would be (0,0,0). For this reason, as shown in the flowchart Figure 45, it is important to verify if the gyroscope is available and if a variable (first) that keep the first value is still zero. In the case of this value is zero the value of gyroscope will continue to be updated each frame. And the value obtains through Euler angles will be assigned to the first variable until the value is no longer zero.

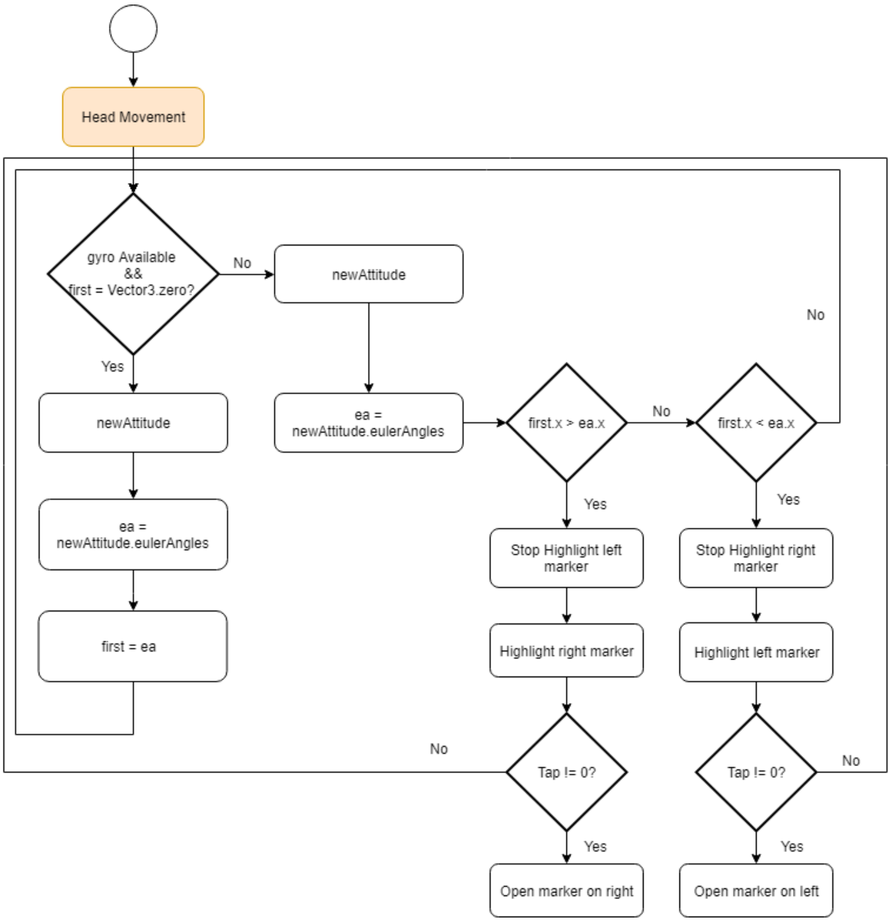


Figure 45 Flowchart of Head Movement

When finally, the variable first is no longer zero, this value will be compared with the real-time value of Euler angles. The advantage of using the Euler angles is using values that mean something, like degrees. For example, in Figure 46 it is possible to understand how it should work. If the x component of the variable first vector is near to the value of zero, if the real-time vector is minor then the user is looking to the left otherwise is looking to the right. The Euler angles do not have negative values because of that before this is necessary to transforming all the Euler Angles to Quaternion.

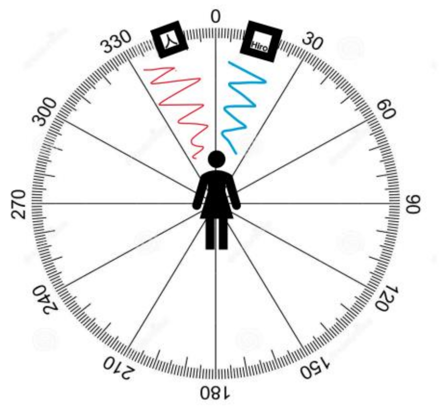


Figure 46 Head movement for fiducial marker selection

In the process of development, the limitation of angles that the user could do to select a marker was considerate. However, and as schematized in Figure 46, after some degrees the marker would be off the FOV. So, the function of head movement stops to being evocated. In the case of the minimum of degrees to select a marker was also not important because the slight movement of user heads only cause the unselect but would decide nothing.

Another problem detected was that when the value of the gyroscope was updated in each frame the tracking of marker would stop. For this reason, was necessary to have a thread for the gyroscope. These two should work concurrently.

3.4.3. Handheld Interaction

It was important to understand if the two interaction methods that were implemented were better than the default of the device, the handheld controller[6]. For that reason, a third interaction method using the Moverio BT-300 handheld controller was tested. In this method, the NyARToolkit was used to detect and track the markers. Initially, it was intended to use the UI buttons from unity since this element already has onClick function. However, when it was joined to a marker the button always shows up in the middle of the scenario. This was a problem while using two markers, due to the overlap of both buttons, and it was hard to tell which one belonged to each marker. A solution was to give the marker's position to the button but when this was done the button

[6] https://www.youtube.com/watch?v=c5qY1XwxTDU&feature=youtu.be

always shows up below and towards the left of the marker. Sometimes it was difficult to see it in the display because it was outside of the field of view. As a solution, and as verified with another scenario, it would be best to use a 3D element and build a button from it, as shown in Figure 47. Yet would maintain the interaction method by double-tapping the button with the controller's help.

Figure 47 Buttons to interact with the controller

3.4.4. Outdoor interaction

The goal of the outdoor interaction was extending the indoor solution. Create an application that could continually keep tracking of the environment. However, in the development phase of the outdoor scene, most of the interaction's methods would not be able to work. In the case of Vuforia, for example, it would only work if there are some markers to track and put some virtual buttons. The case of double-tap and head movement could work however would involve a lot of head movements and in some cases would not be so slight how to show in beginning. The user would have several points around him and would need to turn and move the head until the POI that he wanted to be selected. The only interaction method that could work in an outdoor scenario would be using the controller from the smart glasses. However, using the controller would bring some drawbacks as mentioned above like hinder the user to perform another task simultaneously.

4. Evaluation

It was conducted a brief study of these three interaction methods. Condition A was the application in which was used the Vuforia and the user would have virtual buttons to interact with the markers. Condition B uses the NyArtoolkit to track system and in this system, the interaction methods are the head movement and the double-tap in the smart glasses frame. The last condition, C, uses the NyArtoolkit to track the markers and when it tracked, display a button over the marker that can be interacted with the smart glasses' controller. In Figure 48, it is illustrated the three interaction methods.

Figure 48 Interaction Methods with a) gestures b) head movement and on-device interaction c) and with the smart glasses controller

In this study, it was collected the time between the marker detection and the interaction with it in a way to understand which one can be easiest to interact. To complement this information will be taken note of how many times the user tries to interact but without success.

To compare the different interaction methods, a within-subjects study design was followed, where the same participant tests all the conditions. The within-subject study's experimental design can bring some advantages and some disadvantages. The within-subject study can maximize the learning and transfer across the different tests and have longer sessions, unlike the between-subject tests. However, it requires fewer participants and is cheaper to run. The most important advantage is minimizing the random noise because external events can influence the behavior of the user when he is testing the system. But if he tests all conditions it all affects the three conditions minimizing the random noise between them (Budiu 2018).

As explained above, this kind of method of testing user interfaces can increase user learning. For that reason, it was used the Latin square method for rotation of the different conditions. In Table 5, it is possible to see in summary form how is it intended to rotate the different conditions between the participants.

Participant	A –Gestures; B – Head movement and on-device interaction (double-tap); C – Controller
1	ABC
2	BCA
3	CAB
4	ABC
5	BCA
6	CAB
7	ABC
8	BCA
9	CAB
10	ABC
11	BCA
12	CAB

Table 5 Participants and Conditions

The study had a total of 12 participants and 7 were women. Participants' ages ranged from twenty to sixty. The mean of ages was M = 37.2 years (SD = 15.5). Five of the participants were students while the remaining were workers. One of the participants had already used virtual reality devices as VR PlayStation and VR Samsung. The remaining never used augmented or virtual reality systems.

The experimental set up (Figure 49) was designed considering the selection of information in a multiple marker scenario. All the markers have the same distance between each other however not all of them are detected in all the projects. In all scenarios will be asked to the participant to look around to recognize the environment and ask him to identify the virtual 3D object that appears. In the scenario A, where Vuforia was used, it was asked to find a 3D cat. After identifying it, the user should look for the markers that will have the virtual buttons. All the scenarios go by this approach, having one isolated marker which will have a 3D object like a cat, tree and a doll and then the other two markers will be side by side to give the user the option to choose. It was important also to show that the image or page simply did not just pop up.

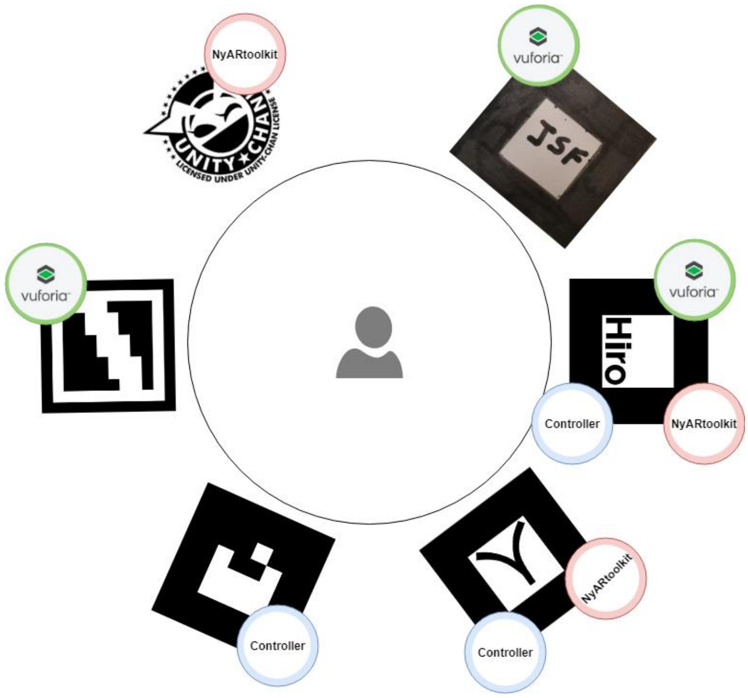


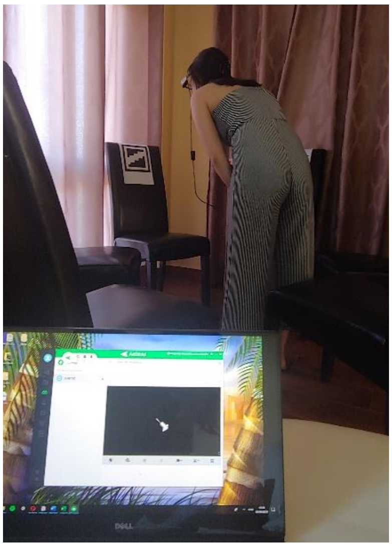
Figure 49 Evaluation Scheme

After testing each scenario, the user answered two questions about that interaction method. The first question was from 0 to 10 how comfortable the user felt to do that gesture to interact with the system. Where 0 was nothing comfortable and 10 was completely comfortable. The second question was from 0 to 10 how available would feel to perform the gesture publicly. Where 0 was nothing available and 10 was completely available.

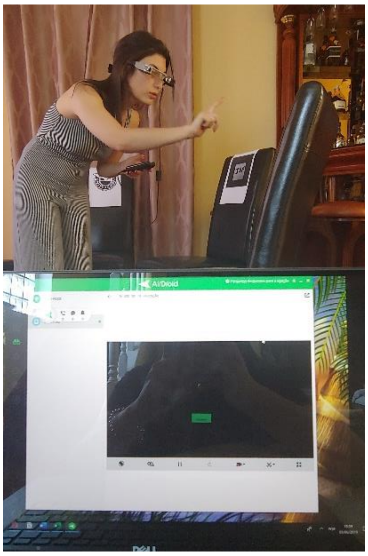
All the activities with the smart glasses inside the applications were logged in a file. A timer started, hidden from the user, with the application and every time a marker was detected, or the user select one the activity, a timestamp was logged in the file.

Figure 50 Participant in the middle of all markers

Thanks to the application AirDroid [7], it was possible to see what the user was seeing through mire cast in another device. It was registered every time the user tried to interact with the markers but without success.



a)



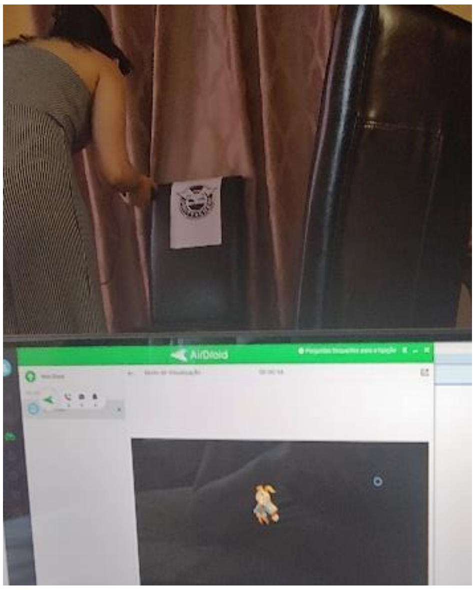
b)

Figure 51 Experimental task with hand gestures: a) Identify the virtual cat, b) Identify and interact with the Virtual Button (Vuforia)

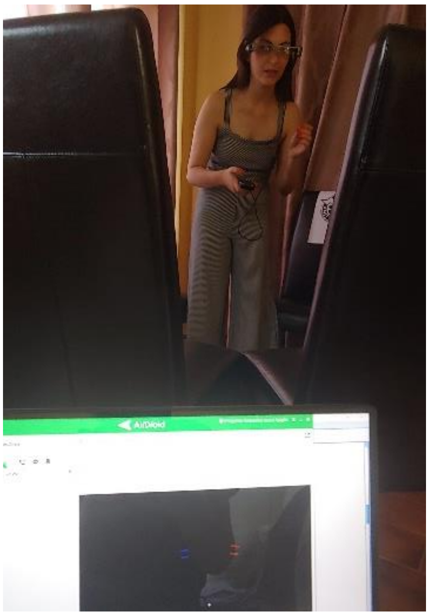
In the application A, which the interaction method was the hand gestures, the user would be identifying the virtual cat as illustrated in Figure 51 a). After that it was asked to the user looking for the other two markers that have a virtual button and using the hand to click as illustrated in Figure 51 b).

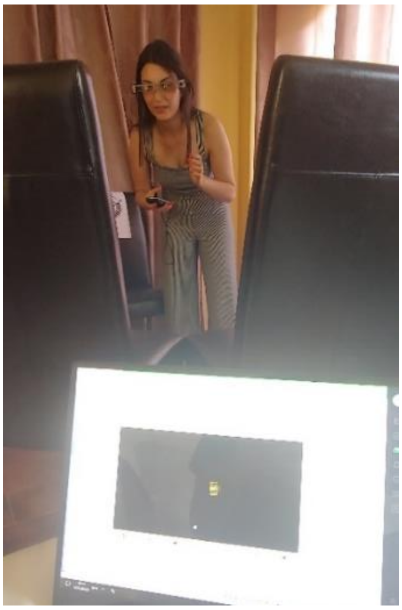
In the application B, as mentioned above the interaction method is the movement of the head and tap the smart glasses frame, the user would look for a virtual object that in this application would be a doll as illustrated in Figure 52 a). After that, the user would look for two other markers that would have a frame and choose one to interact. In Figure 52 b), it is illustrated the frame in the two markers, after this the system would highlight the frame depending on the head movement. After highlighting the marker's frame all the user needs to do is double click the smart glasses frame, as shown in Figure 52 c).

7 https://web.airdroid.com for windows; https://moverio.epson.com/jsp/pc/pc_application_detail.jsp?pack=com.sand.airdroid&page=0&key=airdroid&cat=&tab=category&device=3 for Moverio BT-300



a)

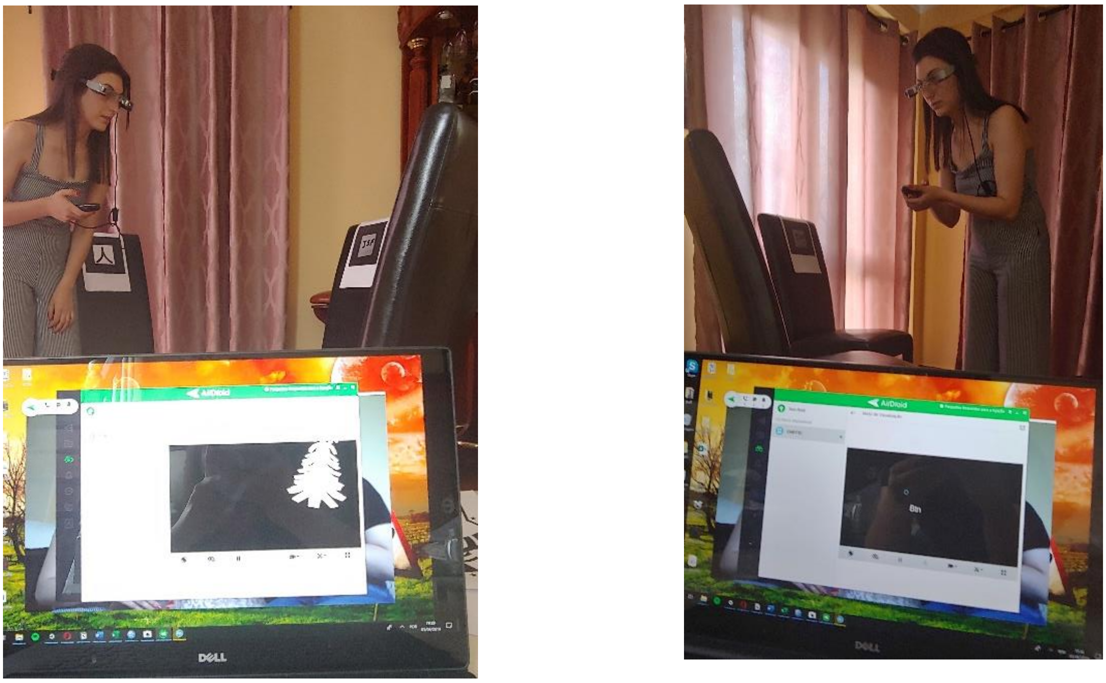




b) c)

Figure 52 Experimental task: a) Identifying the doll object, b) Two markers identified simultaneously, c) Marker's frame highlighted, and user lift the arm to interact

In the last scenario, the user would use the smart glasses controller to move and click the UI button. Initially, it was asked to the user to look for a virtual object, a tree, as shown in Figure 53 a). After this it was asked to the user to look for the other two markers and choose one to interact with by moving the cursor with the smart glass controller as illustrated in Figure 53 b).



a) b)

Figure 53 The experimental task with controller: a) User looking for the tree, b) User using the controller to click in the UI button

As mentioned above, every time the user starts the application, it a log file is created, and a timing system starts. In this file, the time is logged when the camera detects the marker. It is also logged the time the user interacts with any of the markers. Aside it was registered the number of times the user tried to interact with a marker without success (missed targets). It was made a small survey to understand how comfortable the participants felt doing the three different movements and if she/he would be available to do it in a public environment.

To compare the results of the three different applications with each other, it was started an analysis of each parameter of the study. For the reaction times, which were logged in a file, it was needed to know if the time had a normal distribution.

The Shapiro-Wilk test indicates that reaction times using the hand as an interaction method follow a normal distribution, $D(12)=0.92$, $p=0.34$. The Shapiro-Wilk test indicates that reaction times using the tap function and head movement also follow a normal distribution, $D(12)=0.93$, $p=0.37$. However the same test indicates that reaction times using the smart glasses controller do not follow a normal distribution, $D(12)=0.79$, $p=0.01$. Therefore, nonparametric statistical tests were used.

The reaction times are calculated by the difference between the time that the system identifies the marker and the time that the user interacts with the system. Thus, the reaction times are the time that the user takes to interact with the system after the system identifies the marker.

Since the reaction times using the smart glasses controller do not follow a normal distribution it was performed the Friedman test for all three reaction times. A non-parametric Friedman test of differences among repeated measures was conducted and rendered a Chi-square value of 5.17 and a p-value = 0.08 which was not significant ($p > 0.05$).

Post hoc analysis with Wilcoxon signed-rank tests was conducted with a Bonferroni correction applied, resulting in a significance level set at $p < 0.017$ ($0.05 \div 3 = 0.017$).

A Wilcoxon Signed-Ranks Test indicated that the “Hand Gestures” times scores (Mdn = 48.35) were not statistically significantly higher than the “Head Movement and Tap Function” times scores (Mdn=27.80), Z=12.00, p = 0.03.

A Wilcoxon Signed-Ranks Test indicated that the “Head Movement and Tap Function” times scores (Mdn = 27.80) were not statistically significant than the “UI Button” times scores (Mdn=17.45), Z=30.00, p= 0.48.

A Wilcoxon Signed-Ranks Test indicated that the “Hand Gestures” times scores (Mdn = 48.35) were not statistically significantly higher than the “UI Button” times scores (Mdn=17.45) Z=65.00, p = 0.04.

The Posthoc analysis confirms the Wilcoxon Signed-Ranks test showing no significant differences in reaction times.

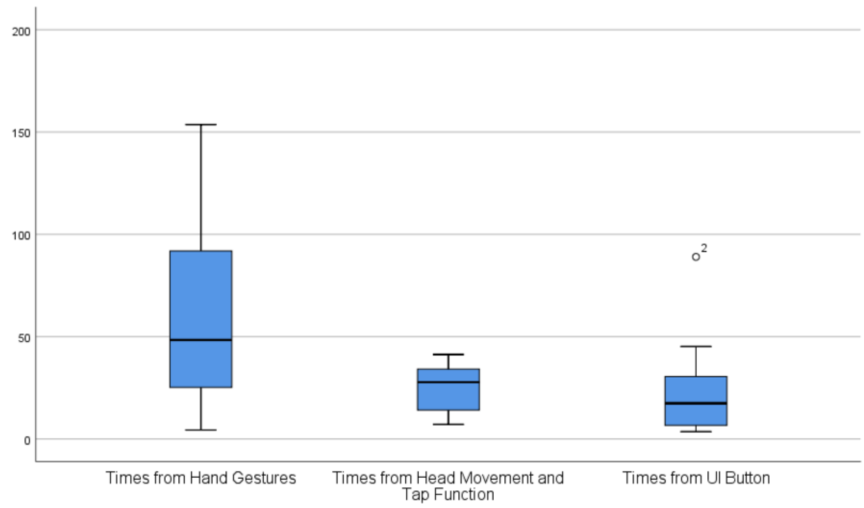


Figure 54 Boxplot of the reaction times in the three scenarios

Since the number of failed attempts is nonparametric data, the first test performed was the Friedman Test. A non-parametric Friedman test of differences among repeated measures was conducted and rendered a Chi-square value of 10.23 and a p-value = 0.01 which was significant ($p < 0.05$).

Post hoc analysis with Wilcoxon signed-rank tests was conducted with a Bonferroni correction applied, resulting in a significance level set at $p < 0.02$ ($0.05 \div 3 = 0.017$).

A Wilcoxon Signed-Ranks Test indicated that the "Hand Gestures" attempts rank (Mdn = 4.00) were not statistically significantly higher than the "Head Movement and Tap Function" attempts rank (Mdn=2.00), Z=56.50, p = 0.04.

A Wilcoxon Signed-Ranks Test indicated that the "Head Movement and Tap Function" attempts rank (Mdn=2.00) were not statistically significantly higher than the "UI Button" attempts rank (Mdn=0.00), Z=10.50, p = 0.04.

A Wilcoxon Signed-Ranks Test indicated that the "Hand Gestures" attempts rank (Mdn =4.00) were statistically significantly higher than the "UI Button" attempts rank (Mdn=0.00), Z=55.00, p = 0.01.

The Posthoc analysis confirms the Wilcoxon Signed-Ranks test, showing a significant difference in the number of failed attempts with Hand Gestures when compared with a UI button, being the UI Button significantly lower.

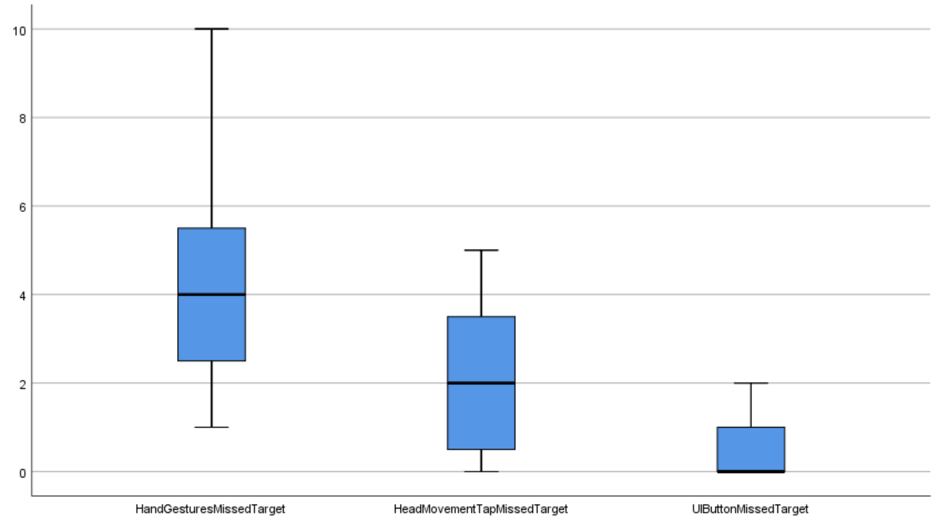


Figure 55 Boxplot of the Missed Targets

The comfort rank was also nonparametric data. A non-parametric Friedman test of differences among repeated measures was conducted and rendered a Chi-square value of 10.39 and a p-value = 0.01 which was significant ($p < 0.05$). Post hoc analysis with Wilcoxon signed-rank tests was conducted with a Bonferroni correction applied, resulting in a significance level set at $p < 0.02$ ($0.05 \div 3 = 0.017$).

A Wilcoxon Signed-Ranks Test indicated that the “Head Movement and Tap Function” comfort rank (Mdn = 9.00) was statistically significantly higher than the “Hand Gestures” comfort rank (Mdn = 8.00), Z=53.00, p = 0.01.

A Wilcoxon Signed-Ranks Test indicated that the “Head Movement and Tap Function” attempts rank (Mdn = 9.00) were not statistically significantly higher than the “UI Button” attempts rank (Mdn=9.00), Z=55.00, p=0.76.

A Wilcoxon Signed-Ranks Test indicated that the “UI Button” comfort rank, (Mdn=9.00) was statistically significantly higher than the “Hand gestures” comfort rank (Mdn = 8.00), Z=0.00, p=0.01.

The Posthoc analysis confirms the Wilcoxon Signed-Ranks test showing a significant difference in the comfort with the Head Movement and Tap Function when compared with Hand Gesture, being the Head Movement and Tap Function significantly higher. The Posthoc analysis confirms the Wilcoxon Signed-Ranks test shows also a significant difference in the comfort with UI Button when compared with Hand Gestures, being UI Button significantly higher.

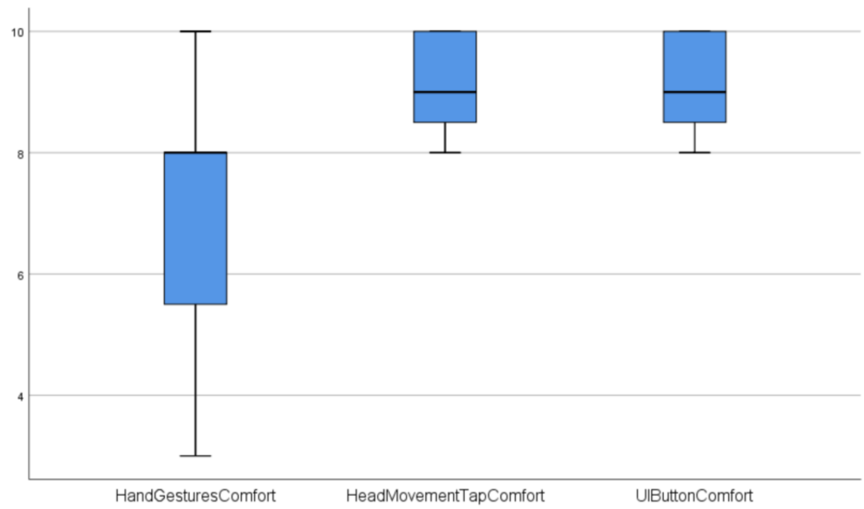


Figure 56 Boxplot of Comfort felt performing the three different interaction methods

The last nonparametric data retrieved from this evaluation was the user availability to perform the movement in public. A non-parametric Friedman test of differences among repeated measures was conducted and rendered a Chi-square value of 19.64 and a p-value = 0.00 (0.000054) which was significant ($p < 0.05$).

Post hoc analysis with Wilcoxon signed-rank tests was conducted with a Bonferroni correction applied, resulting in a significance level set at $p < 0.02$ ($0.05 \div 3 = 0.017$).

A Wilcoxon Signed-Ranks Test indicated that the “Head Movement and Tap Function” availability rank (Mdn = 9.00) was statistically significantly higher than the “Hand Gestures” availability rank, Mdn = 5.50, Z=78.00, p=0.00 (0,002).

A Wilcoxon Signed-Ranks Test indicated that the “UI Button” availability rank (Mdn = 10.00) was not statistically significantly higher, p-value = 0.76 than the “Head Movement and Tap Function” availability rank (Mdn=9.00), Z=20.00, p=0.76.

A Wilcoxon Signed-Ranks Test indicated that the “UI Button” availability rank (Mdn = 10.00) was statistically significantly higher than the “Hand gestures” availability rank (Mdn = 5.50), Z=0.00, p=0.00 (0.002).

The Posthoc analysis confirms the Wilcoxon Signed-Ranks test showing a significant difference in the user willingness with Head Movement and Tap Function when compared with Hand Gesture, being the Head Movement and Tap Function significantly higher. The Posthoc analysis confirms the Wilcoxon Signed-Ranks test shows also a significant difference in the user willingness with UI Button when compared with Hand Gestures, being UI Button significantly higher.

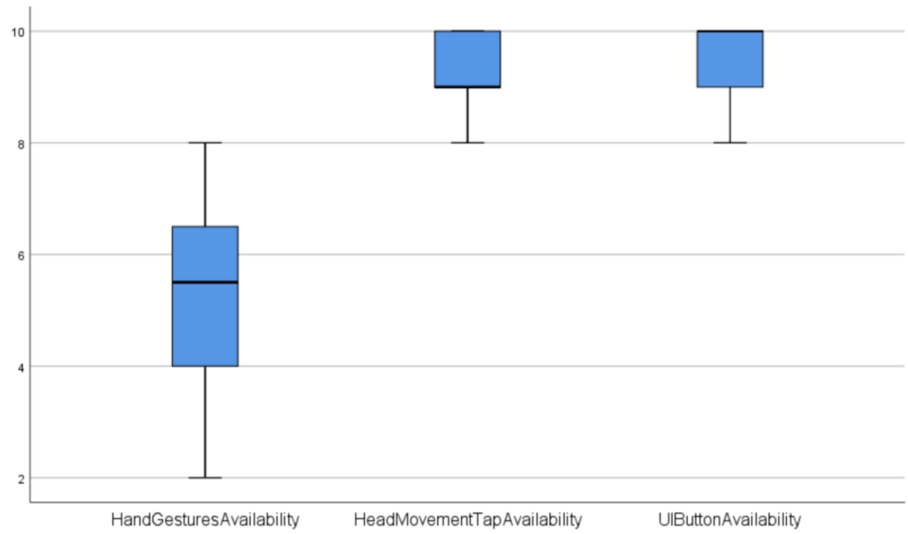


Figure 57 Boxplot of User's Availability of performing the three different gestures in the public environment

4.1. Discussion

The results founded in the present study suggest that in general the interaction with an external controller to select a UI button was better in comparison to the other two interaction methods. Although some tests show no significance, there were some differences between the three interaction methods.

Having in mind the median reaction of times, the UI Button interaction was the fastest but the head movement and tap interaction were very close. However, it is necessary to have into consideration that, as mention above, the users considered the movement on external control similar to what they already do in a smartphone. The other two interaction methods would require learning or at least more time to interact. It would be expected that by doing a similar movement of what users already do the missed targets would be lower. Yet, if we put hand gestures and head movement & tap interactions side by side it is possible to conclude that the second one had a better performance.

In terms of comfort, the results founded show that users felt more comfortable using the Head Movement & Tap interaction and the external control of the smart glasses. However, further and longer studies could provide more feedback about comfortability. As mention above, although the UI Button, the interaction method that uses the external control, is faster to interact and causes fewer missed targets this in a long run could cause more discomfort. Because it required the user to always have the controller in his hands.

Though the studies conducted were not conducted in a public environment, the results indicated that usually, the user would not be available in performing the gestures in public. However, if the studies were conducted in a public environment could change the results obtained because the users would fell in some cases the embarrassment.

5. Conclusion

Through this work, it was possible to know three different smart glasses in the market. It was possible to understand how augmented reality applications are built and how it can be simple to create an augmented reality application for smart glasses for an indoor solution. And it was possible to know different interaction methods for augmented reality applications with special attention the ones suggested for smart glasses. As highlighted in this work the interaction methods for smart glasses are uncertain. There are several ways but no certain in which one could be more reliable.

To smart glasses have a more relevant role in our daily lives, it is essential to understand how users will be willing to use them. Because of that three different interaction methods were compared. The results suggested that the user of the controller was better in comparison with the other two. However, the head movement and tap function presented a good result for a new interaction method for the participants.

In conclusion, it is very important to study and compare the several interaction methods. In a way to understand which interaction methods can be better and provide a better user experience and which ones can be combined to offer that.

5.1. Limitations

At the begging one, the first struggles founded were the fact that smart glasses were not compatible with Google Play Services. Most of the ideas could evolve google maps because are more completed while others have many gaps in Madeira. For example, when it was used the foursquare in the outdoor app there were many places misplaced and others did not exist there all. Another limitation smart glass-related was the fact that it took too much time to the GPS coordinates to update, in several times this led me to believe that the code was not working and not the problem was with the smart glasses.

In terms of the tools that were used, the Nyartoolkit had a big problem with false positives. Several times in testing there was no marker in the front of the camera and yet it was detected as if existed. Like mentioned through this work, it was made several versions to make both tracking systems to work side by side however none of the efforts lead a success.

5.2. Future Work

In this work, it was explored one kind of smart glasses and three interaction methods. However, as explained above there are other types of smart glasses like Hololens that are equipped with depth cameras that allow better recognition of gestures. With the depth cameras, it is possible to recognize more discrete gestures and creates the opportunity of exploring and improving other types of interactions. In the market already exists the new model of EPSON BT-35E that could improve some of the difficulties founded in this project.

In recent times, several studies have been made and allow us to enrich the options to interact with an AR system. For example, the work of Müller et al (Müller et al. 2019) that purpose a foot-based user interface. Other technologies have been studied to improve the user experience and allow interaction methods to grow and improve. Kim et al. (Kim et al. 2019) presented a near-eye AR display with resolution and focal depth dynamically driven by gaze tracking.

All these new studies open the path to the opportunity to merge several explored types of interaction methods. In this way, one type of interaction will rectify another type of interaction.

References

‘After Mixed Year, Mobile AR to Drive $108 Billion VR/AR Market by 2021 | Digi Capital’. 2017. 2017. https://www.digi-capital.com/news/2017/01/after-mixed-year-mobile-ar-to-drive-108-billion-vrar-market-by-2021/.

Aigner, Roland, Daniel Wigdor, Hrvoje Benko, Michael Haller, David Lindlbauer, Alexandra Ion, and Shengdong Zhao. 2012. ‘Understanding Mid-Air Hand Gestures: A Study of Human Preferences in Usage of Gesture Types for HCI’, 10.

Azai, Takumi, Shuhei Ogawa, Mai Otsuki, Fumihisa Shibata, and Asako Kimura. 2017. ‘Selection and Manipulation Methods for a Menu Widget on the Human Forearm’. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems - CHI EA ’17*, 357–60. Denver, Colorado, USA: ACM Press. https://doi.org/10.1145/3027063.3052959.

Azuma, R., Y. Baillot, R. Behringer, S. Feiner, S. Julier, and B. MacIntyre. 2001. ‘Recent Advances in Augmented Reality’. *IEEE Computer Graphics and Applications* 21 (6): 34–47. https://doi.org/10.1109/38.963459.

Azuma, Ronald T. 1997. ‘A Survey of Augmented Reality’, August, 48.

Bailly, Gilles, Jörg Müller, Michael Rohs, Daniel Wigdor, and Sven Kratz. 2012. ‘ShoeSense: A New Perspective on Gestural Interaction and Wearable Applications’. In *Proceedings of the 2012 ACM Annual Conference on Human Factors in Computing Systems - CHI ’12*, 1239. Austin, Texas, USA: ACM Press. https://doi.org/10.1145/2207676.2208576.

Budiu, Raluca. 2018. ‘Between-Subjects vs. Within-Subjects Study Design’. Nielsen Norman Group. 13 May 2018. https://www.nngroup.com/articles/between-within-subjects/.

Caudell, T.P., and D.W. Mizell. 1992. ‘Augmented Reality: An Application of Heads-up Display Technology to Manual Manufacturing Processes’. *Proceedings of the Twenty-Fifth Hawaii International Conference on System Sciences*, no. February: 659–69 vol.2. https://doi.org/10.1109/HICSS.1992.183317.

‘Choose Your Development Environment | ARCore’. n.d. Google Developers. Accessed 4 October 2018. https://developers.google.com/ar/develop/.

Dobbelstein, David, Philipp Hock, and Enrico Rukzio. 2015. ‘Belt: An Unobtrusive Touch Input Device for Head-Worn Displays’. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI ’15*, 2135–38. Seoul, Republic of Korea: ACM Press. https://doi.org/10.1145/2702123.2702450.

‘Does Google Glass Have a Place in the Operating Room?’ n.d. US News & World Report. Accessed 27 November 2018. https://health.usnews.com/health-news/hospital-of-tomorrow/articles/2014/06/03/does-google-glass-have-a-place-in-the-operating-room.

Ens, Barrett, Ahmad Byagowi, Teng Han, Juan David Hincapié-Ramos, and Pourang Irani. 2016. ‘Combining Ring Input with Hand Tracking for Precise, Natural Interaction with Spatial Analytic Interfaces’. In *Proceedings of the 2016 Symposium on Spatial User Interaction - SUI ’16*, 99–102. Tokyo, Japan: ACM Press. https://doi.org/10.1145/2983310.2985757.

‘Euler Angles’. 2019. In *Wikipedia*. https://en.wikipedia.org/w/index.php?title=Euler_angles&oldid=891315801.

Fiala, M. 2005. ‘ARTag, a Fiducial Marker System Using Digital Techniques’. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern*

Recognition (CVPR'05), 2:590–96. San Diego, CA, USA: IEEE. https://doi.org/10.1109/CVPR.2005.74.

Franco, Jéssica, and Diogo Cabral. 2019. 'Augmented Object Selection through Smart Glasses'. In *Proceedings of the 18th International Conference on Mobile and Ubiquitous Multimedia - MUM '19*, 1–5. Pisa, Italy: ACM Press. https://doi.org/10.1145/3365610.3368416.

'Getting Started with ARToolKit+ Unity Plugin (Open Source AR) – ARreverie Technology'. n.d. Accessed 5 October 2018. http://www.arreverie.com/blogs/getting-started-with-artoolkit-unity-plugin/.

Grossman, Tovi, Xiang Anthony Chen, and George Fitzmaurice. 2015. 'Typing on Glasses: Adapting Text Entry to Smart Eyewear'. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI '15*, 144–52. Copenhagen, Denmark: ACM Press. https://doi.org/10.1145/2785830.2785867.

Hirzer, Martin. 2008. 'Marker Detection for Augmented Reality Applications'. *Inst. For Computer Graphics and Vision, Graz …*, 25.

Holstein, Kenneth, Bruce M. McLaren, and Vincent Aleven. 2018. 'Student Learning Benefits of a Mixed-Reality Teacher Awareness Tool in AI-Enhanced Classrooms'. In *Artificial Intelligence in Education*, edited by Carolyn Penstein Rosé, Roberto Martínez-Maldonado, H. Ulrich Hoppe, Rose Luckin, Manolis Mavrikis, Kaska Porayska-Pomsta, Bruce McLaren, and Benedict du Boulay, 10947:154–68. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-93843-1_12.

'How Google Glass Automates Patient Documentation For Dignity Health - CIO Journal. - WSJ'. n.d. Accessed 27 November 2018. https://blogs.wsj.com/cio/2014/06/16/how-google-glass-automates-patient-documentation-for-dignity-health/.

Hsieh, Yi-Ta, Antti Jylhä, Valeria Orso, Luciano Gamberini, and Giulio Jacucci. 2016. 'Designing a Willing-to-Use-in-Public Hand Gestural Interaction Technique for Smart Glasses'. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*, 4203–15. Santa Clara, California, USA: ACM Press. https://doi.org/10.1145/2858036.2858436.

Huang, Da-Yuan, Liwei Chan, Shuo Yang, Fan Wang, Rong-Hao Liang, De-Nian Yang, Yi-Ping Hung, and Bing-Yu Chen. 2016. 'DigitSpace: Designing Thumb-to-Fingers Touch Interfaces for One-Handed and Eyes-Free Interactions'. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*, 1526–37. Santa Clara, California, USA: ACM Press. https://doi.org/10.1145/2858036.2858483.

Ishimaru, Shoya, Kai Kunze, Koichi Kise, Jens Weppner, Andreas Dengel, Paul Lukowicz, and Andreas Bulling. 2014. 'In the Blink of an Eye: Combining Head Motion and Eye Blink Frequency for Activity Recognition with Google Glass'. In *Proceedings of the 5th Augmented Human International Conference on - AH '14*, 1–4. Kobe, Japan: ACM Press. https://doi.org/10.1145/2582051.2582066.

Islam, MD. Rasel, Doyoung Lee, Liza Suraiya Jahan, and Ian Oakley. 2018. 'GlassPass: Tapping Gestures to Unlock Smart Glasses'. In *Proceedings of the 9th Augmented Human International Conference on - AH '18*, 1–8. Seoul, Republic of Korea: ACM Press. https://doi.org/10.1145/3174910.3174936.

Kienzle, Wolf, and Ken Hinckley. 2014. 'LightRing: Always-Available 2D Input on Any Surface'. In *Proceedings of the 27th Annual ACM Symposium on User Interface*

Software and Technology - UIST '14, 157–60. Honolulu, Hawaii, USA: ACM Press. https://doi.org/10.1145/2642918.2647376.

Kim, Jonghyun, Zander Majercik, Peter Shirley, Josef Spjut, Morgan McGuire, David Luebke, Youngmo Jeong, et al. 2019. 'Foveated AR: Dynamically-Foveated Augmented Reality Display'. *ACM Transactions on Graphics* 38 (4): 1–15. https://doi.org/10.1145/3306346.3322987.

Koelle, Marion, Swamy Ananthanarayan, Simon Czupalla, Wilko Heuten, and Susanne Boll. 2018. 'Your Smart Glasses' Camera Bothers Me!: Exploring Opt-in and Opt-out Gestures for Privacy Mediation'. In *Proceedings of the 10th Nordic Conference on Human-Computer Interaction - NordiCHI '18*, 473–81. Oslo, Norway: ACM Press. https://doi.org/10.1145/3240167.3240174.

Lee, Lik-Hang, and Pan Hui. 2017. 'Interaction Methods for Smart Glasses'. *IEEE Access* 6: 28712–32. https://doi.org/10.1109/ACCESS.2018.2831081.

———. 2018. 'Interaction Methods for Smart Glasses: A Survey'. *IEEE Access* 6: 28712–32. https://doi.org/10.1109/ACCESS.2018.2831081.

Lissermann, Roman, Jochen Huber, Aristotelis Hadjakos, and Max Mühlhäuser. 2013. 'EarPut: Augmenting behind-the-Ear Devices for Ear-Based Interaction'. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems on - CHI EA '13*, 1323. Paris, France: ACM Press. https://doi.org/10.1145/2468356.2468592.

Locklear, Mallory. 2018. 'Augmented Reality Smart Glasses to Boost Learning in the Classroom'. BOLD. 10 September 2018. https://bold.expert/augmented-reality-smart-glasses-to-boost-learning-in-the-classroom/.

Medeiros, Daniel, Maurício Sousa, Daniel Mendes, Alberto Raposo, and Joaquim Jorge. 2016. 'Perceiving Depth: Optical versus Video See-Through'. In *Proceedings of the 22nd ACM Conference on Virtual Reality Software and Technology - VRST '16*, 237–40. Munich, Germany: ACM Press. https://doi.org/10.1145/2993369.2993388.

Milgram, Paul, Haruo Takemura, Akira Utsumi, and Fumio Kishino. 1994. 'Augmented Reality: A Class of Displays on the Reality-Virtuality Continuum'. In , edited by Hari Das, 282–92. Boston, MA. https://doi.org/10.1117/12.197321.

Minagawa, Jun, Woong Choi, Satoshi Tsurumi, Nobuto Hirakoso, Nobuo Sasaki, Liang Li, and Kozaburo Hachimura. 2015. 'Supporting System with Hand Gesture for Location-Based Augmented Reality'. In *2015 IEEE 4th Global Conference on Consumer Electronics (GCCE)*, 479–80. Osaka, Japan: IEEE. https://doi.org/10.1109/GCCE.2015.7398533.

Müller, Florian, Joshua McManus, Sebastian Günther, Martin Schmitz, Max Mühlhäuser, and Markus Funk. 2019. 'Mind the Tap: Assessing Foot-Taps for Interacting with Head-Mounted Displays'. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems - CHI '19*, 1–13. Glasgow, Scotland Uk: ACM Press. https://doi.org/10.1145/3290605.3300707.

National Coordination Office for Space-Based Positioning, Navigation, and Timing. n.d. 'GPS.Gov: GPS Accuracy'. GPS Accuracy. Accessed 8 September 2019. https://www.gps.gov/systems/gps/performance/accuracy/.

Nirjon, Shahriar, Jeremy Gummeson, Dan Gelb, and Kyu-Han Kim. 2015. 'TypingRing: A Wearable Ring Platform for Text Input'. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services - MobiSys '15*, 227–39. Florence, Italy: ACM Press. https://doi.org/10.1145/2742647.2742665.

Poelman, R. 2010. ‘Systems Engineering Section, Delft University of Technology Jaffalaan 5, 2628BX Delft, The Netherlands’, 22.
Rabbi, Ihsan, and Sehat Ullah. 2013. ‘A Survey on Augmented Reality Challenges and Tracking Authors’, 19.
Rodríguez-Rosa, Jose, and Jorge Martín-Gutiérrez. 2013. ‘Considerations on Designing a Geo-Targeted AR Application’. *Procedia Computer Science* 25: 436–42. https://doi.org/10.1016/j.procs.2013.11.055.
Rolland, Jannick P., Richard L. Holloway, and Henry Fuchs. 1995. ‘Comparison of Optical and Video See-through, Head-Mounted Displays’. In , edited by Hari Das, 293–307. Boston, MA. https://doi.org/10.1117/12.197322.
Romeiro, Vera Digner. n.d. ‘2018 Promises to Be the Year for Smart Glasses! | Wearable Technologies’. Accessed 30 August 2018. https://www.wearable-technologies.com/2018/02/2018-promises-to-be-the-year-for-smart-glasses/.
Ryu, Ho-Sub, and Hanhoon Park. 2016. ‘A System for Supporting Paper-Based Augmented Reality’. *Multimedia Tools and Applications* 75 (6): 3375–90. https://doi.org/10.1007/s11042-014-2439-7.
Rzayev, Rufat, Paweł W. Woźniak, Tilman Dingler, and Niels Henze. 2018. ‘Reading on Smart Glasses: The Effect of Text Position, Presentation Type and Walking’. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems - CHI ’18*, 1–9. Montreal QC, Canada: ACM Press. https://doi.org/10.1145/3173574.3173619.
Schmalstieg, D., and Tobias Höllerer. 2016. *Augmented Reality: Principles and Practice*. Addison-Wesley Usability and HCI Series. Boston: Addison-Wesley.
Serrano, Marcos, Barrett M. Ens, and Pourang P. Irani. 2014. ‘Exploring the Use of Hand-to-Face Input for Interacting with Head-Worn Displays’. In *Proceedings of the 32nd Annual ACM Conference on Human Factors in Computing Systems - CHI ’14*, 3181–90. Toronto, Ontario, Canada: ACM Press. https://doi.org/10.1145/2556288.2556984.
Siltanen, Sanni. 2012a. *Theory and Applications of Marker-Based Augmented Reality*. *Espoo 2012. VTT Science Series 3*.
———. 2012b. ‘Theory and Applications of Marker-Based Augmented Reality [Markkeriperustaisen Lisätyn Todellisuuden Teoria Ja Sovellukset]’, 250.
Sörös, Gábor, Florian Daiber, and Tomer Weller. 2013. ‘Cyclo: A Personal Bike Coach through the Glass’. In *SIGGRAPH Asia 2013 Symposium on Mobile Graphics and Interactive Applications on - SA ’13*, 1–4. Hong Kong, Hong Kong: ACM Press. https://doi.org/10.1145/2543651.2543660.
Spitzer, Michael, Ibrahim Nanic, and Martin Ebner. 2018. ‘Distance Learning and Assistance Using Smart Glasses’. *Education Sciences* 8 (1): 21. https://doi.org/10.3390/educsci8010021.
Streitz, Nobert, and Panos Markopoulos. 2014. *Distributed, Ambient, and Pervasive Interactions: Second International Conference, DAPI 2014, Held as Part of HCI Interational 2014, Heraklion, Crete, Greece, June 22-27, 2014. Proceedings*. 1st edition. Lecture Notes in Computer Science 8530. New York: Springer.
Sutherland, Ivan. 1965. ‘The Ultimate Display’, 2.
———. 1968. ‘A Head-Mounted Three Dimensional Display’, 5.
Toal, Leonie. 2018. ‘Epson Smart Glasses Reveal New Augmented Horizons with Optimised Wikitude AR SDK - Epson’. 2018. https://www.epson.ie/insights/article/epson-smart-glasses-reveal-new-augmented-horizons-with-optimised-wikitude-ar-sdk.

Trassard, Pauline. 2013. ‘Education: “Smart” Glasses to Help Improve Teacher-Student Communication’. L’Atelier BNP Paribas. 2013. https://atelier.bnpparibas/en/smart-city/article/education-smart-glasses-improve-teacher-student-communication.

Tung, Ying-Chao, Chun-Yen Hsu, Han-Yu Wang, Silvia Chyou, Jhe-Wei Lin, Pei-Jung Wu, Andries Valstar, and Mike Y. Chen. 2015. ‘User-Defined Game Input for Smart Glasses in Public Space’. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI ’15*, 3327–36. Seoul, Republic of Korea: ACM Press. https://doi.org/10.1145/2702123.2702214.

Wang, Cheng-Yao, Wei-Chen Chu, Po-Tsung Chiu, Min-Chieh Hsiu, Yih-Harn Chiang, and Mike Y. Chen. 2015. ‘PalmType: Using Palms as Keyboards for Smart Glasses’. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI ’15*, 153–60. Copenhagen, Denmark: ACM Press. https://doi.org/10.1145/2785830.2785886.

Wang, Cheng-Yao, Min-Chieh Hsiu, Po-Tsung Chiu, Chiao-Hui Chang, Liwei Chan, Bing-Yu Chen, and Mike Y. Chen. 2015. ‘PalmGesture: Using Palms as Gesture Interfaces for Eyes-Free Input’. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI ’15*, 217–26. Copenhagen, Denmark: ACM Press. https://doi.org/10.1145/2785830.2785885.

Weigel, Martin, Vikram Mehta, and Jürgen Steimle. 2014. ‘More than Touch: Understanding How People Use Skin as an Input Surface for Mobile Computing’. In *Proceedings of the 32nd Annual ACM Conference on Human Factors in Computing Systems - CHI ’14*, 179–88. Toronto, Ontario, Canada: ACM Press. https://doi.org/10.1145/2556288.2557239.

‘Welcome to NyARToolkit.EN | NyARToolkit project’. n.d. Accessed 10 April 2019. https://nyatla.jp/nyartoolkit/wp/?page_id=198.

Whitaker, Matthew, and Esther Kuku. 2014. ‘Google Glass: The Future for Surgical Training?’ *The Bulletin of the Royal College of Surgeons of England* 96 (7): 223–223. https://doi.org/10.1308/147363514X13990346756445.

Yi, Shanhe, Zhengrui Qin, Ed Novak, Yafeng Yin, and Qun Li. 2016. ‘GlassGesture: Exploring Head Gesture Interface of Smart Glasses’. In *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, 1–9. San Francisco, CA, USA: IEEE. https://doi.org/10.1109/INFOCOM.2016.7524542.

Yoon, Sang Ho, Ke Huo, Vinh P. Nguyen, and Karthik Ramani. 2015. ‘TIMMi: Finger-Worn Textile Input Device with Multimodal Sensing in Mobile Interaction’. In *Proceedings of the Ninth International Conference on Tangible, Embedded, and Embodied Interaction - TEI ’14*, 269–72. Stanford, California, USA: ACM Press. https://doi.org/10.1145/2677199.2680560.

Yu, Chun, Ke Sun, Mingyuan Zhong, Xincheng Li, Peijun Zhao, and Yuanchun Shi. 2016. ‘One-Dimensional Handwriting: Inputting Letters and Words on Smart Glasses’. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI ’16*, 71–82. Santa Clara, California, USA: ACM Press. https://doi.org/10.1145/2858036.2858542.

Appendix 1 Vuforia configuration

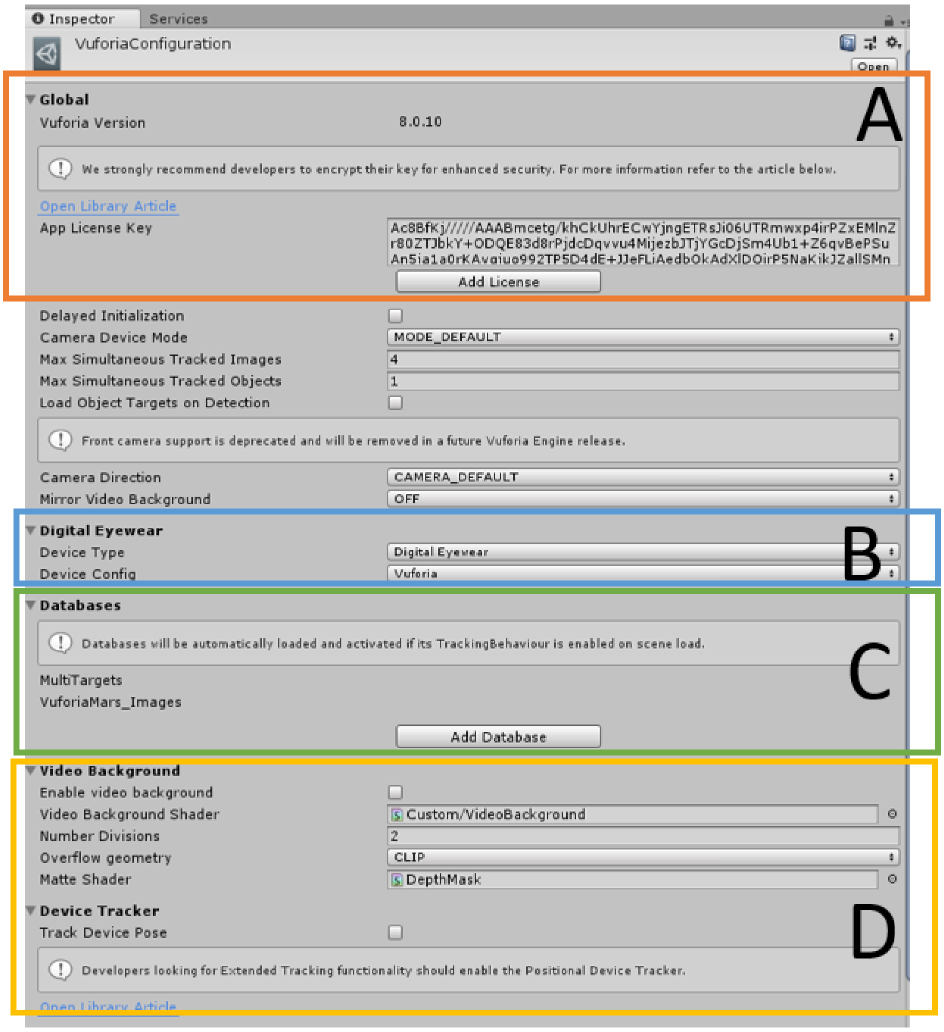


Figure 58 Vuforia Configuration – A- App License Key; B- Digital EyeWear; C – Databases; D -Video Background

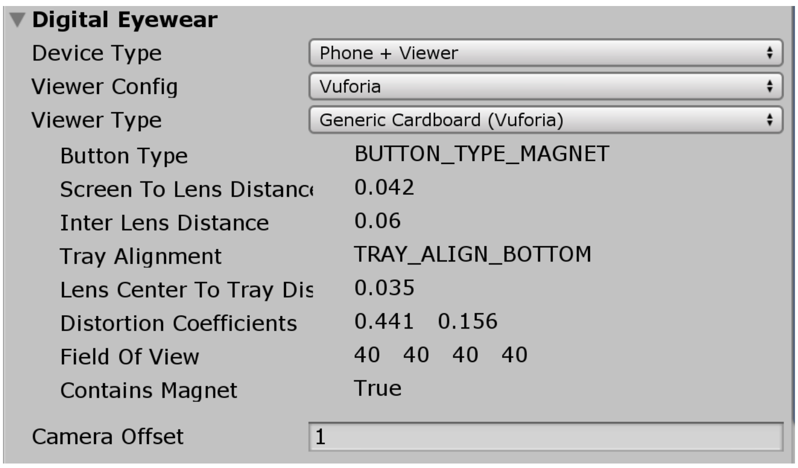


Figure 59 Digital Eyewear properties

Appendix 2 Bytes File

```
214 225 240 225 214 240 216 204 214 227 181 192 198 192 181 192
240 240 240 240 240 240 225 232 240 240 240 240 240 240 240 236
240 240 240 240 240 240  75 128 220 240 240 240 240 240 240 240
240 240 240 240 240 240 106  53 240 240 240 240 240 240 240 237
240 240 240 240 240 238 118  31 240 240 240 240 240 240 240 234
240 240 240 240 240 240  74  49 207 240 240 240 240 240 240 240
240 240 240 240 240 240  53  54 177 240 240 240 240 240 240 240
240 240 240 240 240 240  64  31 130 240 240 240 240 240 240 219
240 240 240 240 240 180  37  57  78 228 240 240 240 240 240 240
240 240 240 240 240 118  62 157  36 185 240 240 240 240 240 231
240 240 240 240 240  82  65 225  67  80 230 240 240 240 240 217
240 240 240 225  53  76 225 240 156  62 158 240 240 240 240 226
240 240 199  61   9 111 235 240 240 104  58 174 228 240 240 240
240 142  64  26  92 227 240 240 240 229  93  64 170 226 238 216
 90  26  12 156 240 240 240 240 240 240 204  95  30 117 192 200
156  16 195 233 235 240 236 240 238 239 240 186  93  53 120 237
214 226 240 225 212 240 216 204 212 226 181 192 198 192 185 194
240 240 240 240 240 240 227 232 240 240 240 240 240 240 240 238
240 240 240 240 240 240  95 138 225 240 240 240 240 240 240 240
240 240 240 240 240 240 108  59 240 240 240 240 240 240 240 237
240 240 240 240 240 238 118  31 240 240 240 240 240 240 240 234
240 240 240 240 240 240  83  47 207 240 240 240 240 240 240 240
240 240 240 240 240 240  56  49 177 240 240 240 240 240 240 240
240 240 240 240 240 240  73  41 130 240 240 240 240 240 240 222
240 240 240 240 240 185  46  49  86 230 240 240 240 240 240 240
240 240 240 240 240 118  58 165  45 192 240 240 240 240 240 234
240 240 240 240 240  91  63 222  74  82 240 240 240 240 240 222
240 240 240 226  66  86 225 240 158  63 162 240 240 240 240 228
240 240 202  76  11 103 235 240 234  91  49 174 228 240 240 240
240 142  68  16  91 226 240 240 240 228  96  74 178 233 239 222
 90  26   4 150 240 240 240 240 240 240 213 109  46 133 204 213
```

Figure 60 Excerpt from Kenji's bytes file

```
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 0   0   0   0   0   0   0   0   255 255 255 255
255 255 255 255 0   0   0   0   0   0   0   0   255 255 255 255
255 255 255 255 0   0   0   0   0   0   0   0   255 255 255 255
255 255 255 255 0   0   0   0   0   0   0   0   255 255 255 255
0   0   0   0   255 255 255 255 0   0   0   0   0   0   0   0
0   0   0   0   255 255 255 255 0   0   0   0   0   0   0   0
0   0   0   0   255 255 255 255 0   0   0   0   0   0   0   0
0   0   0   0   255 255 255 255 0   0   0   0   0   0   0   0
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 0   0   0   0   0   0   0   0   255 255 255 255
255 255 255 255 0   0   0   0   0   0   0   0   255 255 255 255
255 255 255 255 0   0   0   0   0   0   0   0   255 255 255 255
255 255 255 255 0   0   0   0   0   0   0   0   255 255 255 255
0   0   0   0   255 255 255 255 0   0   0   0   0   0   0   0
0   0   0   0   255 255 255 255 0   0   0   0   0   0   0   0
0   0   0   0   255 255 255 255 0   0   0   0   0   0   0   0
0   0   0   0   255 255 255 255 0   0   0   0   0   0   0   0
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
```

Figure 61 Patt file appearance