Universidad de Valladolid

ESCUELA DE INGENIERÍAS INDUSTRIALES

UNIVERSIDAD DE VALLADOLID

ESCUELA DE INGENIERIAS INDUSTRIALES

Grado en Ingeniería Mecánica

Comparing the Perception and Performance of Selected VR Systems

Autor:
Izquierdo Santos, Álvaro

Francisco Javier Rey Martínez

Technische Universität Dresden

Valladolid, Febrero 2020.

TFG REALIZADO EN PROGRAMA DE INTERCAMBIO

TÍTULO:	Comparing the Perception and Performance of Selected VR Systems.
ALUMNO:	Álvaro Izquierdo Santos
FECHA:	05/02/2020
CENTRO:	Faculty/Department of Mechanical Science and Engineering
TUTOR:	Prof. Dr.-Ing. habil. Ralph Stelzer

Resumen:

La realidad virtual (VR) tiene un amplio uso en ingeniería mecánica y es necesario determinar la validez de los dispositivos que la utilizan para saber el grado de precisión a la hora de percibir el entorno virtual al usar este tipo de sistemas.

El objetivo de esta tesis es la evaluación de la implementación técnica y de la percepción de usuario de una escena de realidad virtual usando el sistema de realidad virtual "Oculus Rift S".

Se realizará una introducción al tema y se determinarán aspectos relevantes en relación con la percepción en los sistemas de realidad virtual y, específicamente, de las gafas de realidad virtual (HMD). Se expondrán distintas escenas posibles para el testeo de su validez a la hora de medir dichos aspectos relevantes y se realizarán varias de ellas utilizando el programa "Unity", con su posterior prueba de validez y análisis mediante el ensayo en personas.

Palabras clave: Realidad, Virtual, Percepción, Test, Escena.

Abstract

Virtual Reality (VR) has a wide use in mechanical engineering, and it is necessary to determine the validity of devices which use it in order to know the level of precision when perceiving the virtual environment using this kind of systems.

The goal of this thesis is to evaluate the technical implementation and the user perception in a virtual reality scene using the "Oculus Rift S" VR system.

It contains an introduction to the topic, the relevant aspects concerning perception of virtual reality systems will be determined and, specifically, of the Head Mounted Display (HMD). Several possible scenes for testing their validity when measuring those relevant aspects will be exposed. A few selected scenes will be developed using "Unity", with their subsequent validity tests and evaluation of the Oculus Rift with help of a user study.

Keywords: Reality, Virtual, Perception, Test, Scene.

TECHNISCHE
UNIVERSITÄT
DRESDEN

COMPARING THE PERCEPTION AND PERFORMANCE OF SELECTED VR SYSTEMS

ÁLVARO IZQUIERDO SANTOS
TECHNISCHE UNIVERSITÄT DRESDEN
Fakultät Maschinenwesen-Maschinenbau

COMPARING THE PERCEPTION AND PERFORMANCE OF SELECTED VR SYSTEMS

Index

1. Abstract

1.1. English Version

Virtual Reality (VR) has a wide use in mechanical engineering, and it is necessary to determine the validity of devices which use it in order to know the level of precision when perceiving the virtual environment using this kind of systems.

The goal of this thesis is to evaluate the technical implementation and the user perception in a virtual reality scene using the “Oculus Rift S” VR system.

It contains an introduction to the topic, the relevant aspects concerning perception of virtual reality systems will be determined and, specifically, of the Head Mounted Display (HMD). Several possible scenes for testing their validity when measuring those relevant aspects will be exposed. A few selected scenes will be developed using “Unity”, with their subsequent validity tests and evaluation of the Oculus Rift with help of a user study.

Keywords: Reality, Virtual, Perception, Test, Scene.

1.2. Spanish Version

La realidad virtual (VR) tiene un amplio uso en ingeniería mecánica y es necesario determinar la validez de los dispositivos que la utilizan para saber el grado de precisión a la hora de percibir el entorno virtual al usar este tipo de sistemas.

El objetivo de esta tesis es la evaluación de la implementación técnica y de la percepción de usuario de una escena de realidad virtual usando el sistema de realidad virtual “Oculus Rift S”.

Se realizará una introducción al tema y se determinarán aspectos relevantes en relación con la percepción en los sistemas de realidad virtual y, específicamente, de las gafas de realidad virtual (HMD). Se expondrán distintas escenas posibles para el testeo de su validez a la hora de medir dichos aspectos relevantes y se realizarán varias de ellas utilizando el programa “Unity”, con su posterior prueba de validez y análisis mediante el ensayo en personas.

Palabras clave: Realidad, Virtual, Percepción, Test, Escena.

1.3. German Version

Virtuelle Realität (VR) hat eine breite Verwendung im Maschinenbau. Es ist notwendig, die Tauglichkeit der dabei verwendeten Geräte zu bestimmen. Zu diesem Zweck muss die von den Geräten erreichte Präzision bei der Wahrnehmung der virtuellen Umgebung bestimmt werden.

Das Ziel dieser Arbeit ist es, die technische Umsetzung und die Wahrnehmung einer Virtual-Reality-Szene mit dem VR-System "Oculus Rift S" zu bewerten.

Es wird eine Einführung in das Thema gegeben und relevante Aspekte der Wahrnehmung in VR-Systemen und speziell in Head-Mounted-Displays (HMD) ermittelt. Es werden verschiedene mögliche Szenen vorgestellt, um ihre Validität bei der Messung dieser relevanten Aspekte zu testen. Ausgewählte Szenen werden mit dem Programm "Unity" umgesetzt. Diese werden in einer Nutzerstudievalidiert und zur Bewertung der Oculus Rift S.

Stichworte: Realität, Virtuelle, Wahrnehmung, Test, Szenen.

2. Thesis goals

The goal of this thesis is to evaluate the technical implementation and the user perception in a virtual reality scene using the “Oculus Rift S” VR system.

For doing so, it is needed to analyze which performance aspects of VR in mechanical engineering influence the most and, once done this, what is the best possible way to test them.

In order to test these aspects, it is so relevant that tests can be developed so that another possible user can replicate them using another or the same VR system.

Several possible scenes for testing their validity when measuring those relevant aspects will be exposed, pointing the focus on “Head Mounted Displays” (HMDs) and, specifically, on “Oculus Rift S”.

The way the tests are going to be tested is through “Unity” scenes, using C# programming language and developing a scene, with their subsequent evaluation of the Oculus Rift with help of a user study.

In this thesis, only four tests of all possible tests proposed where conducted. The reason is the time had and the pitfalls while trying to carry them through.

In next pages, an introduction into VR will be exposed and, afterwards, all the performance aspects and tests will be worded. Next, the user study’s tests will be stated with their results.

To sum up, it will be exposed here a little introduction into VR and some of its important aspects when applied to mechanical engineering, some of them with help of a user study.

3. Approach to Virtual Reality

Virtual Reality (VR) is the utilization of technology (mostly using a computer) to create and simulate an environment. The most used component used in virtual reality and the most recognized one is the HMD (head mounted display). There are so much HMD in the market, such as Oculus Rift, HTC Vive and PlayStation VR.

Virtual reality is so famous because of videogames. Nowadays, there are a lot of videogames that use virtual reality. Nevertheless, virtual reality can be used in very different applications. For example, it is used in medical applications, in military training and in industrial design.

Even though HMD is the most famous item of virtual reality, there are more items in the virtual reality world with advantages and disadvantages to the others.

Illustration 1. Oculus Rift S HMD (Source: Oculus)

3.1. History of Virtual Reality

❖ **<u>1838</u>**

The first to describe stereopsis (a term that is referred to the perception of depth and tridimensional structure obtained from the two eyes by individuals with normally developed binocular vision) was Sir Charles Wheatstone. He constructed the stereoscope (*Illustration 2*).

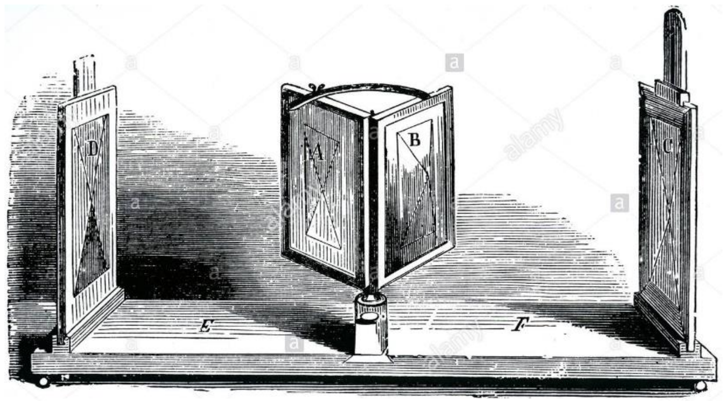


Illustration 2. Stereoscope (Source: researchgate.net)

The stereoscope creates the illusion of tridimensional images using two mirrors at 45º degrees to the eyes of the user, so that each of them reflects an image located off to the side.

❖ <u>**1935**</u>

In 1935, Stanley Weinbaum presented a fictional model for virtual reality in one of his stories, Pygmalion´s Spectacles.

In the story, a professor created a pair of goggles which enabled “a movie that gives one sight and sound, taste, smell and touch” (*Illustration 3*). Stanley wrote about a pair of goggles very similar to currently ones, and pretended them to allow the user to enter in a “new dimension” where everything could be real.

PYGMALION'S SPECTACLES
By STANLEY G. WEINBAUM
Author of "The Black Flame," "A Martian Odyssey," etc.
© 1935 by Continental Publications, Inc.
Unbelieving, still gripping the arms of that unseen chair, Dan was staring at a forest

Illustration 3. Pygmalion´s Spectacles Googgles (Source: medium.com)

❖ <u>**1956**</u>

In 1956, Morton Heilig (a cinematographer) created “Sensorama” (*Illustration 4*), the very first machine of virtual reality (it was patented in 1962).

“Sensorama” consisted in a booth that could be used to up to four people and combined multiple technologies to stimulate senses with the use of a 3D video, audio, smell, vibrations and several effects, such as wind. It used a vibrating chair, scent producers, a stereoscopic 3D screen and stereo speakers. The goal was to fully immerse people in his films.

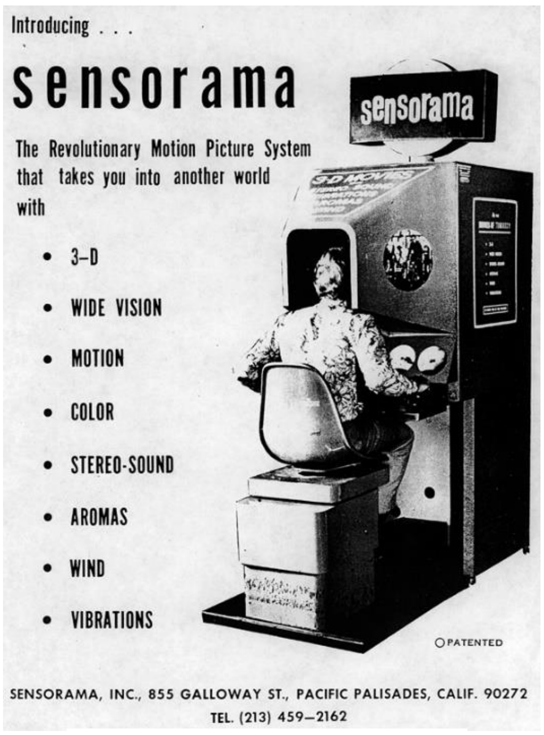

Introducing . . .
sensorama
sensorama
The Revolutionary Motion Picture System that takes you into another world with
3-D
WIDE VISION
MOTION
COLOR
STEREO-SOUND
AROMAS
WIND
VIBRATIONS
PATENTED
SENSORAMA, INC., 855 GALLOWAY ST., PACIFIC PALISADES, CALIF. 90272
TEL. (213) 459—2162

Illustration 4. Sensorama machine (Source: researchgate.net)

❖ <u>**1960**</u>

In 1960, Heilig (Sensorama´s creator) patented the “Telesphere Mask” (*Illustration 5*), the first HMD (head mounted display). This HMD had not motion tracking in the headset but provided stereoscopic images in 3D with stereo sound and wide vision.

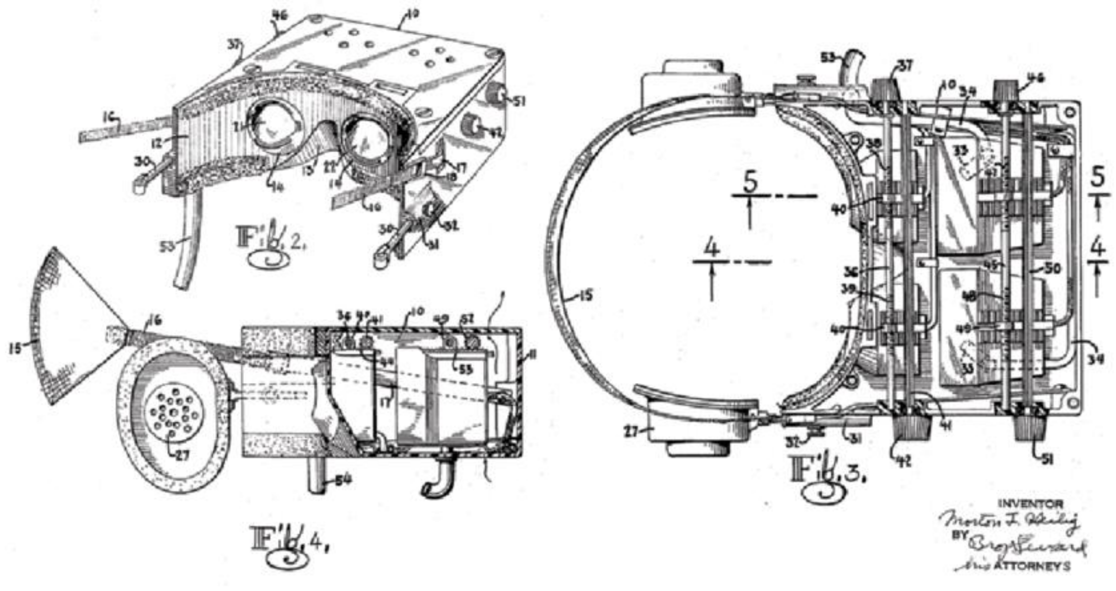


Illustration 5. Telesphere Mask (Source: Wikimedia Commons)

❖ **<u>1961</u>**

In 1961, two Philco Corporation engineers created the first HMD with motion tracking. His priority use was for military uses.

❖ **<u>1968</u>**

In 1968, after some inventions related to virtual reality, the first virtual reality HMD was created (named “The Sword of Damocles”, *Illustration 6*). The creators were Sutherland and Bob Sproull. It was connected to a computer and could only show simple virtual wireframes shapes. Nevertheless, it had a very heavy tracking system, which made it non-viable outside the lab.

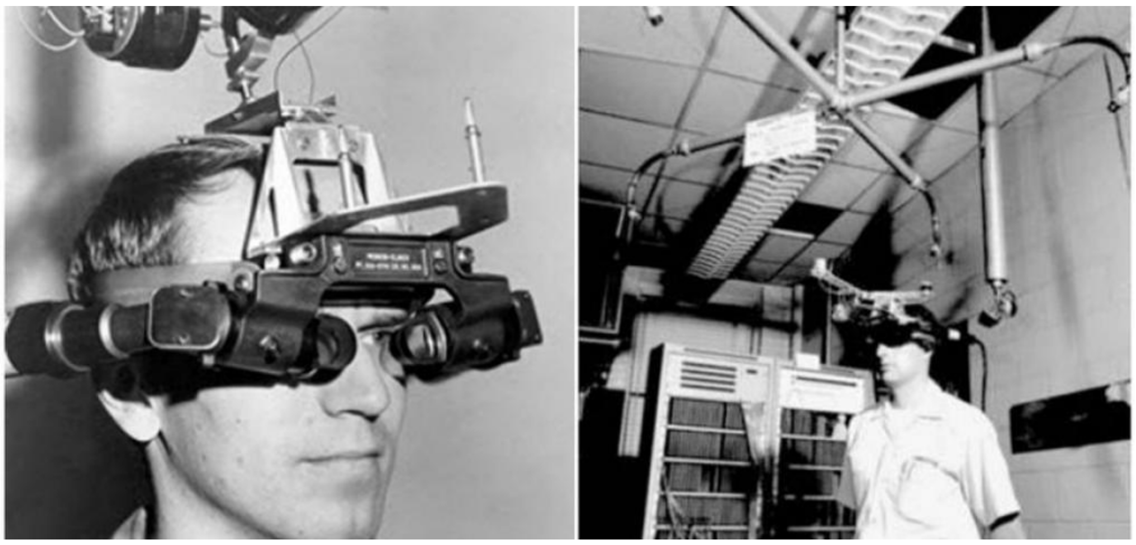


Illustration 6. The Sword of Damocles HMD
(Source: vroom.buzz)

❖ **1975**

In 1975, Krueger's VIDEOPLACE was developed (*Illustration 7*). It was the first interactive VR platform and used projectors, video displays, computer graphics, video cameras and position and sensor technology. One important aspect was that it did not use gloves or goggles, it consisted of dark rooms surrounded by video displays. It was like CAVE systems.

One person saw himself in a screen and could interact with virtual items.

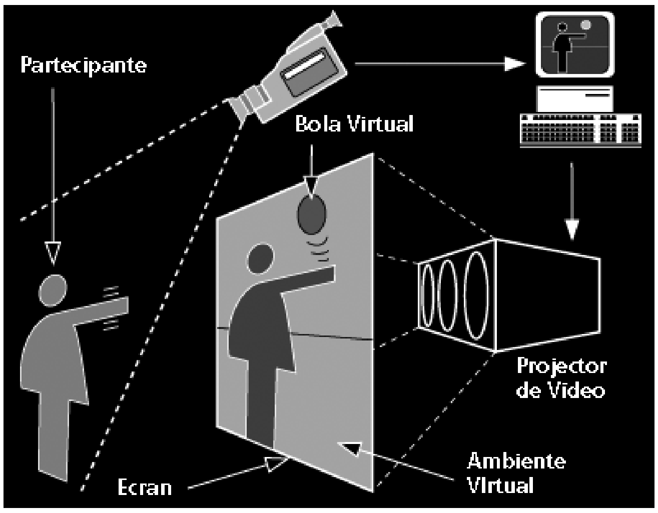


Illustration 7. Krueger´s VIDEOPLACE (Source: proyectoidis.org)

❖ **1989**

Crystal River Engineering Inc. was founded in 1989 by Scott Foster after being hired by NASA to develop the audio element of VIEW (Virtual Environment Workstation Project, *Illustration 8*). It was a VR simulator for NASA's astronauts. This project combined head mounted device with gloves to enable the haptic interaction.

Illustration 8. Virtual Environment Workstation Project HMD (Source: nasa.gov)

❖ **1991**

In 1991 there were some relevant advances in VR.

Antonio Medina designed a virtual reality system to drive the Mars Robot from Earth in real time (taking in account all signal delays between planets).

The Virtuality Group produced VR entertainment systems in mass, "Virtuality" (*Illustration 9*) was an arcade machine with 3D games. The Virtuality machine used HMD and immersive stereoscopic 3D images in real time.

SEGA announced the SEGA VR HMD. It used LCD displays, stereo headphones and tracking sensors. However, it was never released.

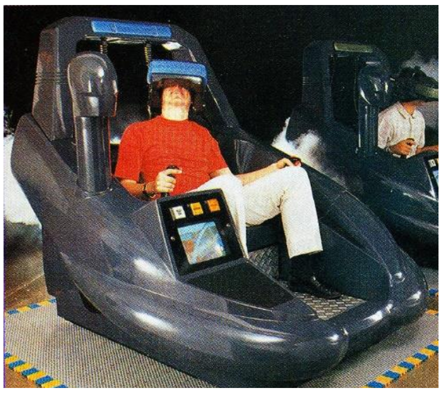


Illustration 9. Virtuality arcade machine (Source: vrs.org.uk)

❖ <u>2007</u>

In 2007, Street View was introduced by Google. They used a dodecahedral camera array on a moving car.

❖ <u>2010</u>

In 2010, Google created a stereoscopic 3D mode for Street view.

In 2010 too, Palmer Luckey created the first prototype of Oculus Rift HMD. It had 90-degree field of view and used a computer to process images. Thanks to Oculus Rift, interest on VR was renewed.

❖ <u>2014</u>

In 2014 Facebook bought Oculus VR Companion for 2.000 million dollars, Sony announced “Project Morpheus” (a VR headset for PS4), Google released “Cardboard” (a low cost stereoscopic viewer for smartphones) and Samsung announced “Samsung Galaxy Gear VR” (a VR viewer that uses Samsung smartphones).All this made VR more and more popular, increasing money that is dedicated to this type of research.

❖ <u>Recent years</u>

From 2014 on, VR started becoming available to the general public. Companies started developing VR products, mostly headsets. Headsets were mostly button operated, they didn’t have haptic interface (haptic interface is an interface that allows the interaction with a computer using touch and movements, such as gloves).

HTC released HTC VIVE SteamVR HMD, which was the first HMD with a tracking system.

Nowadays, virtual reality is being used in a wide range of applications, from videogames to engineering or medical purposes.

Illustration 10. HTC VIVE HMD (Source: HTC)

3.2. Characteristics of Virtual Reality

SABIA, one section of the La Coruña University (UDC) dedicated to Artificial Intelligence, proposes the characteristics of VR as the 3 "*i*" (*Immersion*, *Interaction* and *Imagination*). Between these three characteristics can be constructed the VR Triangle (*Illustration 11*).

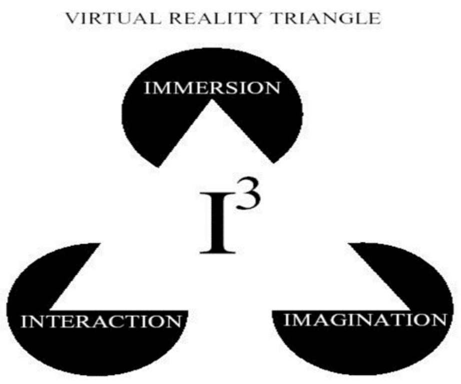


Illustration 11. Virtual Reality Triangle (Source: SABIA)

- ❖ **Interaction:** The user interacts with the virtual world by using several devices and receives the response in real time through his/her senses.
- ❖ **Immersion:** The user loses all contact with reality and does only get impulses from virtual world. The degree of immersion depends on real world contacts that the users has in this moment.
- ❖ **Imagination:** Through virtual world, the user percepts and envisages non existing realities.

3.3. Types of Virtual Reality

According to "Open Future", a Telefónica's blog, an enterprise dedicated to development and innovation in immersive technology, there are 3 types of Virtual Reality: *Immersive Systems*, *Semi-Immersive Systems* and *Non-Immersive Systems*.

- ❖ **Immersive Systems:** They are defined as systems that permit the user feel part of the virtual world without any contact with reality. For doing this, the user must harness some devices, such as a VR headset (HMD) and gloves. In that way, the user could immerse himself/herself in a virtual world (as accurate as the technology limitations are).

  In terms of utility, these immersive systems are being used to training/formations of employees in several professions. Furthermore, they are mostly being used in entertainment applications, like videogames or roller coasters (*Illustration 12*), or for commercial purposes.

Illustration 12. Roller Coaster with VR (México) (Source: Samsung)

It can be seen in such a great way in the film *Ready Player One* (*Illustration 13*), directed by Steven Spielberg (2018). A big virtual world was developed, and you can sense everything that happens there by only putting on you a headset, a pair of gloves and a suit.
In short, immersive virtual reality pretends to make you feel that you are in another place in the best possible way, depending highly on current technology to get to the closest sensation of being in that virtual world.

Illustration 13. Ready Player One cover

❖ **Semi-Immersive Systems**: This type of systems consists of a cube with, at least, 4 screens. One on the floor and the other 3 on the walls. These screens surround the user and allow him/her to interact with the virtual environment and with real world elements.
To interact with the virtual world, it is needed a tracking system, normally situated on the glasses that the user must wear in order to see the virtual world. Thanks to these glasses, when the user moves the head, the tracking system detects it and changes the images on the screens to reflect that change.
The unique similarity to immersive systems is that devices are needed to interact with virtual world, while the difference lays on user experience (in semi-immersive systems the user is not completely dived into the virtual world) and technologies (brain and ears are not totally coordinated and can produce dizziness).
An example of semi-immersive system is CAVE (Cave Assisted Virtual Environment, *Illustrations 14 & 15*), a system with cube-shape with screens in its walls, floor and ceiling which allow the user to feel a virtual environment with glasses and a tracking system on them.

Illustrations 14 & 15. CAVE in TU Dresden
(Source: tu-dresden.de)

- **Non-Immersive Systems**: These systems do only need a screen to enter the virtual world. The user can interact through a mouse, a keyboard or a device (gadgets) that allows him/her to control something in this virtual world.
  This type of systems are the most common ones, but the less immersive of them all. In fact, in videogame's world, mostly all development is being now pointed to this type of systems because of their cheaper prizes and their higher popularity between gamers (PS4, XBOX ONE, Nintendo Switch, gaming computers…).

3.4. Augmented Reality

Augmented Reality (AR) is a very recent technology, it does not have so many definitions thus. The definition that gives "ThinkMobile" (a mobile virtual network operator [MVNOs]) is:

Augmented reality is the technology that expands our physical world, adding layers of digital information onto it. Unlike Virtual Reality (VR), AR does not create the whole artificial environments to replace real with a virtual one. AR appears in direct view of an existing environment and adds sounds, videos, graphics to it.

A view of the physical real-world environment with superimposed computer-generated images, thus changing the perception of reality, is the AR.

Augmented reality is a mix between real world and virtual world in real time and with stimuli which replace the reality with "a new reality".

A great example of Augmented Reality can be seen in "Black Mirror", a Netflix series. It has a lot of chapters where treats the topic of VR and AR. Nevertheless, in my opinion, there is a chapter that shows it in a really good way. In "Playtest" (season 3, episode 2) a man is subdued to a virtual environment using a chip on his neck that can influence in all his senses, making this environment a real one for him.

Illustration 16. Playtest (Black Mirror episode). The mole is a virtual one and can be seen only through the eyes

3.5. Virtual Reality & Augmented Reality

Each technology has his pros and his cons. Hereby, it will be exposed some of them.

- **Sensations**: VR is fully computer-based. The user cannot be in a virtual scene feeling himself a part of it as if he/she were part of the virtual world. Conversely, AR dives the user in another reality, a reality which feels further real than VR one, because it gets components of reality in the scene.
- **Senses:** In VR the information comes completely from the device (the headset normally), so it is a different reality and his/her senses get distorted. Notwithstanding, the AR reality includes the user's real one and the user senses do not get too much distorted (not in the VR level at least).

- **Devices:** In VR there are some devices needed, such as a headset, gloves… The user must be isolated from the real world in a little way leastwise. In AR, conversely, it is only needed a screen, glasses are optional. Briefing, less devices are needed in AR than in VR (in general).
- **Prizes:** VR is, in principle, a cheaper technology than AR's.

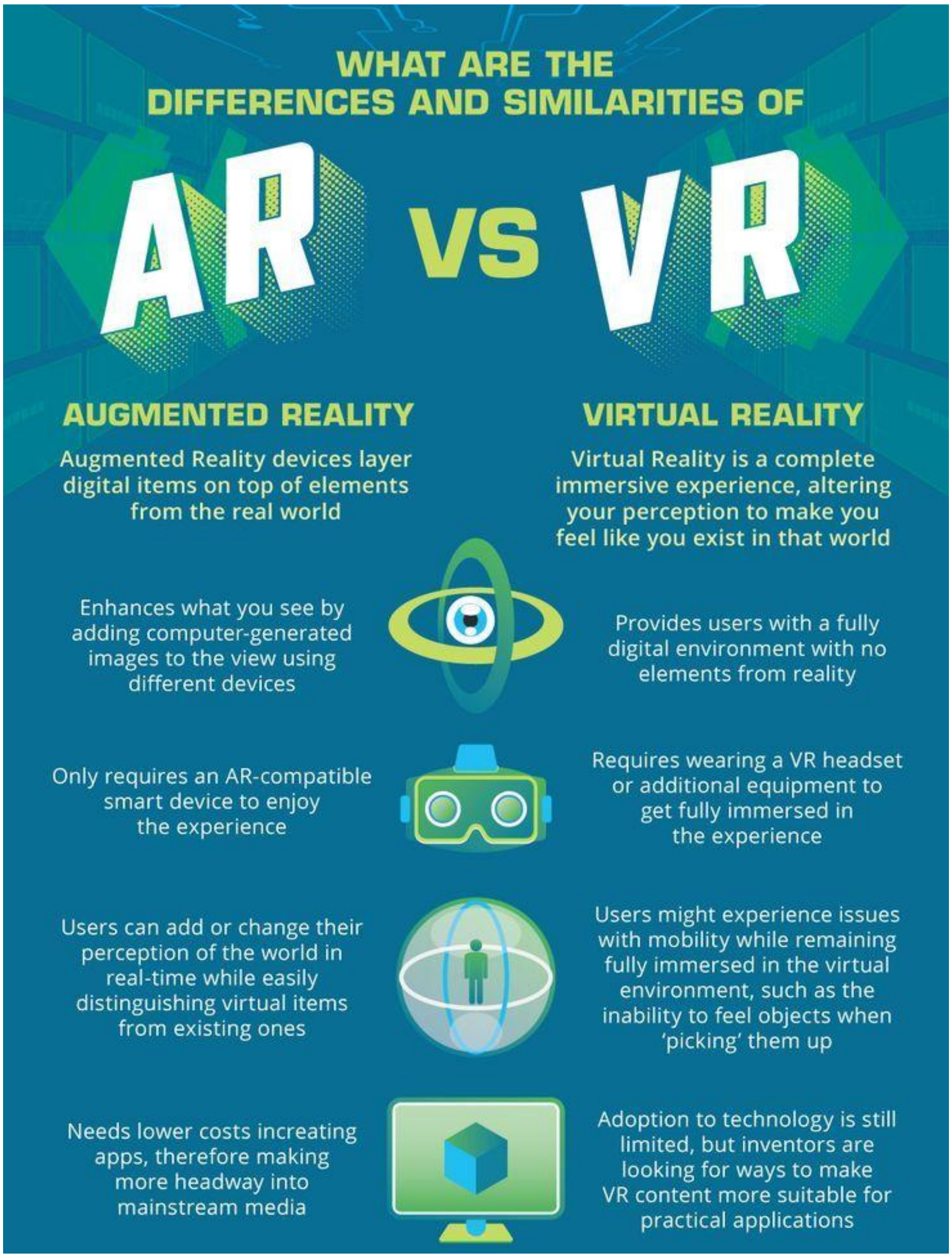

WHAT ARE THE DIFFERENCES AND SIMILARITIES OF
AR VS VR
AUGMENTED REALITY
Augmented Reality devices layer digital items on top of elements from the real world
VIRTUAL REALITY
Virtual Reality is a complete immersive experience, altering your perception to make you feel like you exist in that world
Enhances what you see by adding computer-generated images to the view using different devices
Provides users with a fully digital environment with no elements from reality
Only requires an AR-compatible smart device to enjoy the experience
Requires wearing a VR headset or additional equipment to get fully immersed in the experience
Users can add or change their perception of the world in real-time while easily distinguishing virtual items from existing ones
Users might experience issues with mobility while remaining fully immersed in the virtual environment, such as the inability to feel objects when 'picking' them up
Needs lower costs increating apps, therefore making more headway into mainstream media
Adoption to technology is still limited, but inventors are looking for ways to make VR content more suitable for practical applications

Illustration 17. Differences and similarities of VR & AR (Source: Noobie |Technology and Gadget Information)

3.6. Mixed Reality

Mixed reality (MR) is a concept that involves Virtual Reality and Augmented Reality at the same time. Both technologies mixed in a unique one.

The main idea of Mixed Reality is to carry real world to virtual one, to generate a 3D model of reality and superpose virtual information on it, like holograms.

According to a "Forbes Magazine" definition:

Mixed reality is a significant advancement of augmented reality (AR) – the technology behind 2016's Pokémon GO phenomenon. In a "hybrid" environment, interactive virtual objects can be mapped to the physical environment, blending the real and the virtual.

The most highlighted idea in Mixed Reality is "Project Tango" (*Illustration 18*), a Google project that scans space in real time to mix 3D models, the real one and the virtual one.

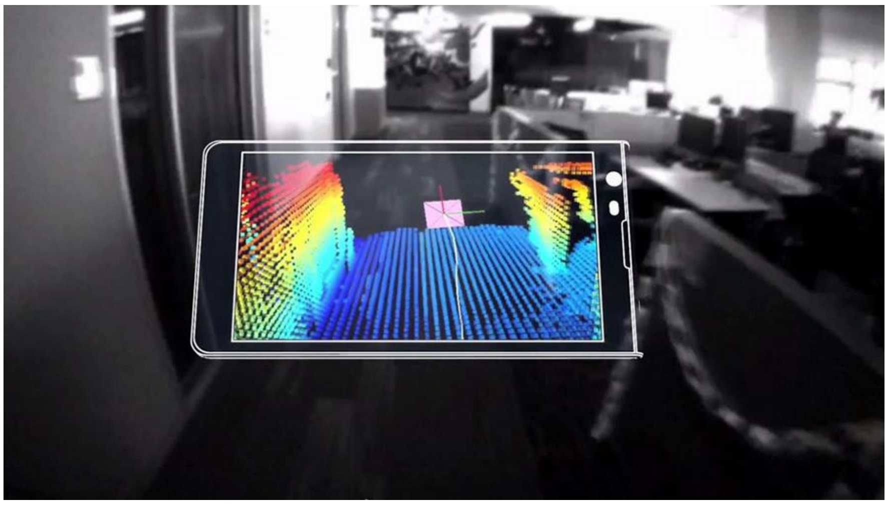


Illustration 18. Project Tango (Source: Google)

- **Potential applications:** Mixed Reality can be used in a wide variety of applications. For example, in the disappearance of size restricted TVs, in prototyping, technical formation on-site…
- **Devices:** Nowadays, there are some Mixed reality devices, such as "Microsoft Hololens" (*Ilustration 19*), "Meta 2" and "Daqri Smart Helmet".

Illustration 19. Microsoft Hololens (Source: Microsoft)

3.7. CAVE System

CAVE (Cave Assisted Virtual Environment) is an installation which allows to visualize tridimensional objects in a semi-immersive form. Tridimensional objects are projected like holograms on the walls, ceilings and floor and the user can move between them with a tracking system and a pair of glasses.

The image is generated through projectors (at least, three), which project stereoscopic images on screens situated forming part of the cube.

As it is seen in the *illustration 20,* CAVE is built by projectors, screens and mirrors. All together can make a tridimensional view if the user is correctly situated.

For being correctly situated on the scene, the user has to wear a pair of glasses with a tracking system on them. This way, it is possible to control the projectors and to change the images every time in real time, so that the user could see in each moment a tridimensional object. If the user does not wear the glasses and the tracking system, the images do not move, and the tridimensional object can only be seen being positioned in a certain point (the point where the user eyes can build the tridimensional object using the image projections).

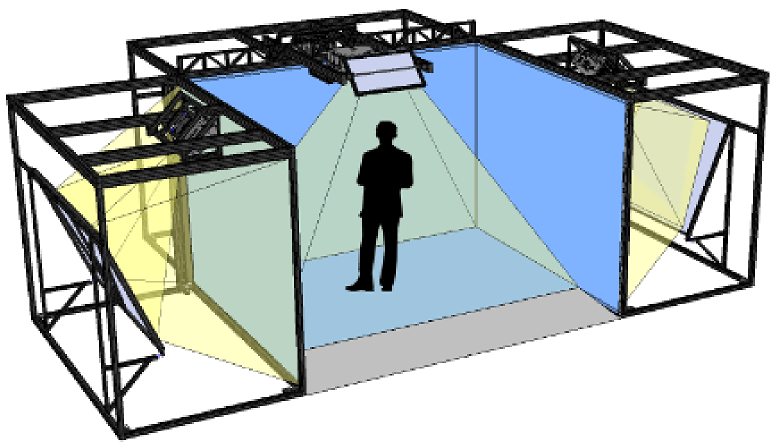


Illustration 20. CAVE with 3 walls (Source: visbox.com)

In my opinion, the following CAVE explanation is so accurate (Source*: howstuffworks.com*):

A CAVE is a small room or cubicle where at least three walls (and sometimes the floor and ceiling) act as giant monitors. The display gives the user a very wide field of view -- something that most head-mounted displays can't do. Users can also move around in a CAVE system without being tethered to a computer, though they still must wear a pair of funky goggles that are similar to 3-D glasses.

The active walls are rear-projection screens. A computer provides the images projected on each screen, creating a cohesive virtual environment. The projected images are in a stereoscopic format and are projected in a fast alternating pattern. The lenses in the user's goggles have shutters that open and shut in synchronization with the alternating images, providing the user with the illusion of depth.

Tracking devices attached to the glasses tell the computer how to adjust the projected images as you walk around the environment. Users normally carry a controller wand in order to interact with virtual objects or navigate through parts of the environment. More than one user can be in a CAVE at the same time, though only the user wearing the tracking device will be able to adjust the point of view -- all other users will be passive observers.

Illustration 21. CAVE System. The user sees in 3D, but we can only see in 3D because we are in a different position (Source: antycipsimulation.com)

5. HMD, Oculus Rift S and Unity

HMD (Head Mounted Display) is a headset, normally stereoscopic, used to create depth effects and which emits the image output near the eyes. It uses little screens, projectors, lenses or mirrors to make the signal get the eyes.

HMDs have subjection elements to stay fixed in a position on the head. They can also have headphones to send sonorous stimulus.

Some of them, like *Samsung Gear VR*, are only an item where a phone can be put, so that the phone could send the image stimulus.

Some others, like *Oculus Rift* or *HTC Vive*, own a vision system and have to be connected to a computer to work. Some HMDs don't need to be connected to a computer, they are completely autonomous (*Oculus Go* and *HTC Vive Focus*).

Finally, it is noteworthy that some of them can show simultaneously real world and virtual world (Mixed Reality, like *Hololens* and *Magic Leap One*). Most of them can only show a virtual environment, they don't have cameras/holes on the eyes.

Illustration 22. HMD (Oculus Rift on the left side and HTC Vive on the right side)
(Source: gamestar.de)

5.1. Oculus Rift S

Oculus Rift S is the headset (HMD) used to develop this thesis. The main goal is to compare the technical implementation and perception/experience of a VR scene using selected VR systems, in this case, Oculus Rift S.

Illustration 23. Oculus Rift S HMD (Source: Oculus)

Oculus Rift S has got these specifications:

- LCD Screen, 2560x1440 pixels. Refresh rate of 80HZ.
- Weight: 470g.
- Producer: Lenovo.
- Tracking: Up to 6 degrees of freedom. Oculus Insight tracking system.
- Recommended specifications: PC with Windows 10, NVIDIA GTX 1060, 8GM RAM, USB 3.0
- Cable: 5m
- Audio: Stereo Passtrough+ System

They need to be connected to a computer. This model uses artificial vision algorithms and tracks the physical space in real-time.

They have Oculus Insight, a new tracking technology that increases the movement detection for making the movement sensation more immersive.

They allow the user to see the environment without putting the headset out (Passthrough+).

It will be exposed theoretically how to test this HMD, which relevant aspects have to be tested and how to test them.

5.2. Unity and C#

Unity is a cross-platform game engine developed by Unity Technologies. Unity can be used to create three-dimensional, two-dimensional, virtual reality and augmented reality games. It can be used to create simulators too.

Illustration 24. Unity logo (Source: Unity)

Outside videogames, Unity is used in engineering, architecture, construction, automation or in films.

Unity has been used to program pointed to Virtual Reality to use with Oculus Rift S. The programming language used for it is C#.

C# is a multi-paradigm programming language which derivates from C/C++ and which is very similar to Java.

For developing the thesis, Unity and C# were used for creating scenes that could be used on Oculus Rift S and that could help testing some performance aspects of mechanical engineering in Virtual Reality. Furthermore, the goal is that these tests could be implemented in any other Virtual Reality devices to test the same things (and not only to test Oculus Rift S).

6. Performance Aspects of Virtual Reality in Mechanical Engineering

Here are the performance aspects of virtual reality in mechanical engineering considered in this thesis and discussed afterwards.

1. **Precision of perception.**
2. **Precision of input devices.**
3. **Latency of display and refresh rate.**
4. **Degree of immersion.**
5. **Field of View.**
6. **Interactivity**.
7. **Image duplication (JUDDER)**, formed by two phenomena:
   a. **Smearing**: Image smudge/ Movement blur.
   b. **Strobing**: Stroboscopy/ Perception of multiple image copies at the same time, making it look like that it’s no movement between them.
8. **Sensation and Comfort.**

7. Performance Tests

Subsequently, there will be exposed each performance aspects and the tests that could fit to them.

Each aspect could be in more than a test, so this will be reflected properly.

Only a few tests were programmed, and other tests were proposed but they were not as good as thought at first (they did not fit at all, or they were not appropriated to each aspect), so these tests were discarded. However, all of them will be written and discussed so that a possible future reader will be informed of everything taken into account at the time this thesis was being developed.

7.1. Precision of Perception

This aspect is referred to the precision with which a person receives visual stimulus from the VR device and how accurate could it be compared to the same stimulus in real life.

In order to do a test of precision of perception, it is necessary to think first how to reproduce visual stimulus from reality in virtual reality.

For example, a car prototype could be tested in virtual reality, and compared to the prototype in real life. This example is not so accurate because the results depend on the resolution and the prototype quality, and the tester opinion cannot be measured, so tests must be something that could be measured and that can be reproduced with each VR device used.

It is remarkable that every test will rely on tester perception and on tester vision, because the tester can have some visual problem (like myopia, astigmatism or daltonism). This visual problem can be solved if the user uses glasses while using the VR device, in order to see properly and like in real life.

If the visual problem is daltonism, there will be a problem. When testing in an aleatory number of people, each one should see as good as possible and should see the same colors because if every person uses glasses and all images are projected in the good point of the retina, daltonic people will see different colors and tests could not be meaningful (as it is portrayed in *Illustration 25*). For this reason, each "precision of perception" test has to be tested in people with a correct view and, if someone has a vision problem, he/she must correct this problem before doing the test (with glasses correctly graduated, for example). Furthermore, each virtual reality device could show colors differently, so the problem increases.

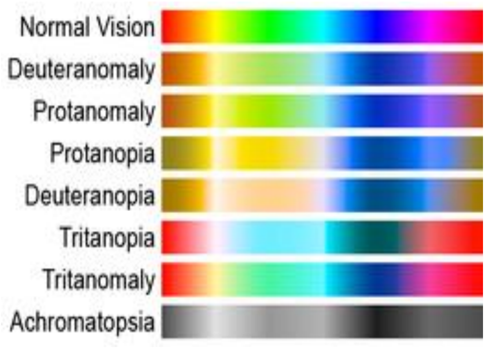


Illustration 25. Color vision deficiencies (Source: Wikipedia)

Known this, we can start proposing situations to test precision of perception.

7.1.1. Readability Test

The readability test consists in reading a text in Virtual Reality. This user should be able to move the text through space, not only further and closer to him/her but also in the better possible position in order to see the text as good as possible (the text should be able to be moved in all directions).

Furthermore, the user must be in the same point during the experiment.

The test consists in varying the text offset from the user until the user cannot read it properly (from a far distance until it is read). Once the user cannot read it, the offset is written down. The font size must be the same in all experiments to have meaningful results.

When one user has done the test properly, the same test should be done in real life, so that results could be compared to virtual ones. For doing this, the font size must be the same and the offset where the user cannot read the text too. This way, a comparison between virtual reality and reality can be done.

This text can be improved in so many different ways, and there are some relevant aspects to be considered:

❖ The user should not know the text he/she is going to read. Thereby, it is sure when the user starts reading the text. Different tests could be made with different texts each time, and, when enough measures are taken, a mean could be a really significant parameter (one with virtual reality experiments and one in reality).

❖ Colors are important, even when there are no visual problems. The same text can be read better in one color than in another one. If a background is put, this fact is accentuated.

Karl Borggrafe created a table (*Illustration 26*) that shows different numbers in various colors with a background. Each background has a color and, depending on the colors mixed between the background and the number, the readability is better or worse. Basing on this fact, it is better to use pairs of color that are better for readability.

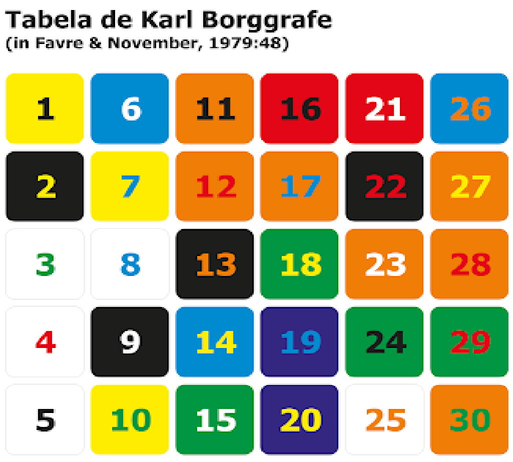


Tabela de Karl Borggrafe
(in Favre & November, 1979:48)

1	6	11	16	21	26
2	7	12	17	22	27
3	8	13	18	23	28
4	9	14	19	24	29
5	10	15	20	25	30

Illustration 26. Karl Borggrafe table (Source: mafecolor.com)

❖ Comparison between virtual reality and reality in terms of distances is really important. For this reason, if we want this test to have precision it is needed that, before doing this, comparison between measures is done. Meters in virtual reality could be accurate or not. For example, one meter in real life could be 0,998 meters in virtual reality. Differences are allowed depending on which precision is required, but it is a good tip to take this into account. A test about this will be done in later pages, so, if wanted, its results can be got and translate to this test.

- As said before, font size must be the same every time (in the reality and in the virtual reality). Otherwise, results would not be valid results.

- As stated before, the text should be in front of the tester eyes. This way, minimal distance is between them, and the offset is more accurate.

This test is developed in 8.1..

7.1.2. Image Discernment Test

The image discernment test consists in a test aimed to compare images between virtual reality and reality. Therefore, it is similar to the "Readability Test".

In this test the difference is that an image will be used to be compared, not a text.

While with a text it is easy to discern when the tester starts reading the text, with an image it is not that easy. For this reason, a medical image is the best option in my opinion.

For doing this test, the image should be able to be moved through space (like with the text). With the image in front of the tester, offset should be modified to a properly value.

Which is this value? It should be one standardized value. In my opinion, the same value as used in medical purposed.

Visual acuity test uses 6 meters as the distance between the image and the person, and it is used to determine the smaller letters the person can read. While in the "Readability Test" (see 7.1.1.) it is possible to obtain a number when the tester starts reading the text, with an image it is very different because there is nothing to read, it is more abstract. For this reason, this test is not all about discerning an image, but about discerning shapes, lines…

My main idea is to put an image with simple images that can be identified no matter who is the tester (see *Illustration 27*). With an image this type, the tester will need to discern shapes and lines. It should be done in virtual reality and then in reality in order to compare results.

The best way to check if the tester has seen the image properly, he/she could draw it on a paper (that's why the images should be simple ones).

As in the "Readability Test", some things must be taken into account in order to do this test properly:

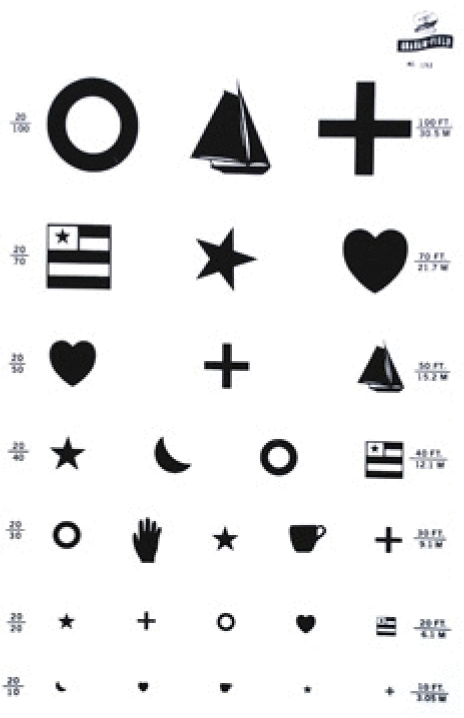


Illustration 27. Image Discern Test Example (Source: mdsupplies.com)

❖ The user must not know the image he/she is going to see. Thereby, it is sure when the user doesn't see the image properly. This image should be drawn in order to check if it was seen well.

❖ Colors are important (as pointed in 7.1.1.), so it is good to use colors that fit well with each other and that are not a problem while reading/seeing.

❖ Comparison between virtual reality and reality in terms of distances is really important (as pointed in 7.1.1.). For this reason, if we want this test to have precision it is needed that, before doing this, comparison between measures is done. If 6 meters are used as the offset, for example, it is good to assure that each 6 meters measure the same. A test about this will be done in later pages, so, if wanted, its results can be got and translate to this test.

❖ As said before, image dimensions must be the same (in the reality and in the virtual reality, likewise with the 6 meters). Otherwise, results would not be valid results.

❖ As stated in 7.1.1., the image should be in front of the tester eyes. This way, minimal distance is between them, and the offset is more accurate.

<u>This test is developed in 8.2., but in a little different way.</u>

7.1.3. Close Distance Test

The close distance test consists in a test aimed to compare how good close distances can be seen in virtual reality.

As 7.1.1. and 7.1.2., this text will use texts/images to see when the tester starts reading/seeing this text/image, modifying the offset from close to far. It is mostly the same test as “Readability Test” and “Image Discernment Test” depending if a text of an image is used, but from a close distance and increasing the offset until the tester can read/see it.

The conditions of the text/image must be the same, and the relevant tips to follow too (see 7.1.1 for a text test and 7.1.2. for an image test).

Different relevant tips are the following:

- Images/texts can be the same, always knowing that the tester should not know its content.

- Images/texts that are so close to the eyes start to be blurred. The test should end at the distance when the tester sees the text/image completely sharp.

Illustration 28. Blurred image and sharp image
(Source: imo.es)

7.1.4. Laser Test (DISCARDED)

At first, it had been thought about a test in which a laser could be used to measure precision in Virtual Reality. The first complication was that there is no way to measure something like that. A dartboard could be situated on the scene, and the tester could point at it to measure how accurate can he/she point at. The problem is that the tester sees the virtual scene, and precision of perception cannot be measured this way, what can be measured this way is the user precision (and not the VR device aspects).

For this reason, this test has been discarded and it has been thought about other tests to measure precision of perception in Virtual Reality that allow to get a measurement or a comparison between real world perception and virtual world.

In 7.1.5., a similar test is developed besides this was discarded. The difference is that what will be measured is the precision of the position on the scene, not the user’s ability to position. For this, the user will close the eyes.

7.1.5. Touching Test

The main idea of the “Touching Test” is to test the precision of perception in terms of position.

Some figures are situated on the scene, and the tester has to touch them in order to measure how accurate is the virtual scene positioning things on the user’s view.

It is not about distances, that is another test. It is about perception of distances. For doing so, it is important to consider that there is the same problem as in 7.1.4., the test relays on the tester in an important quantity and the precision when touching and feeling that the object is on the right position depends on the trajectory, so the user will adapt himself and will feel like everything is good and he/she is touching where the object was at first, even if the object was 10 meters away and he/she has touched it like there were only 2 meters.

The idea is that, once the tester has seen the object he/she has to touch, he/she closes the eyes, so that there is not any adjustment or correction while moving and the tester can touch exactly where he/she had seen the object at first. This way, the previous problem does not involve a problem anymore.

Once done the experiment in virtual reality, the same experiment must be recreated in real world with the same positions. Measure of distances could be a problem like explained in previous tests, so, when the test where real and virtual distances is done, results can be extrapolated. Even so, there is no way in precising so much because results depend on the tester perception, so measurements are not really valid.

This test has no manner to measure with numbers something, but, in my opinion, is a good test to test precision of perception.

This test is mostly the same as the 7.1.6. test and must be done in the same way (it is important to close the eyes when translating). In my opinion, if testing a VR system is ongoing, it is a good idea to do one of these tests, even if no numerical results can be obtained.

7.1.6. Moving Objects Test

The main idea of this test is mostly the same as the previous one (7.1.5.), to test the precision of perception in terms of position.

An object or several objects are situated on the scene, and the user has to move them from a point to another of his/her election. The start coordinates of the object are stored, and the final coordinates too. Like the 7.1.5. test, the experiment should be repeated in real life with the same tester, and the results/opinion, compared.

This test has the same problems as the last two tests, it is all about perception and the tester will adapt his/her senses while moving (better explained in 7.1.5.: *the test relays on the tester in an important quantity and the precision when touching and feeling that the object is on the right position depends on the trajectory, so the user will adapt himself and will feel like everything is good and he/she is touching where the object was at first, even if the object was 10 meters away and he/she has touched it like there were only 2 meters*).

For this reason, this test is mostly the same as the 7.1.5. test and must be done in the same way (it is important to close the eyes when translating). In my opinion, if testing a VR system is ongoing, it is a good idea to do one of these tests, even if no numerical results can be obtained.

7.1.7. Comparing Measures Test

This test is the test it has been talked about before. The test consists in measuring an established distance both in virtual world and in real world.

The idea (one of so much ideas applicable to this test) is that an object is situated on virtual scene with determined dimensions, for example a cube of 1 meter per edge. The tester has to put his/her hand measuring this object and a helper has to measure his/her hands separation in order to compare this measurement with the real one.

For example, if the cube has 1 meter per edge and the tester hands separation is 0,94 meters, they can be written down and do the same experiment repeatedly, so that the mean of all these values is valid enough.

This experiment should be done in all the three axes. There could be a reduction on perception depending on the direction chosen.

This test is developed in 8.3..

7.2. Precision of Input Devices

This aspect is referred to the precision in terms of position of the input devices, and not only of a simple position, but also with translations.

In order to measure this, it is necessary to think first what are the input devices.

Input devices are all hardware that is used on the scene and which the user can interact with. A piece of computer hardware equipment used to provide control signals and data to a computer. For example: mouse, keyboard, joysticks, HMDs, cameras…

Mouse, keyboard and all this type of hardware does not need to be tested. The hardware that need to be tested are VR hardware, like joysticks and HMDs.

Precision can be associated to the VR devices tracking systems, because button only detect if someone has pressed it and joysticks detect the pressing state. About the cameras, it is more related to precision of perception because they show the virtual scene. The last thing to be tested is the tracking systems, and it does make sense that the relative position (and the precision of the position) between devices is so relevant. Virtual position in the scene must be the correspondent to the real one (the same relative positions).

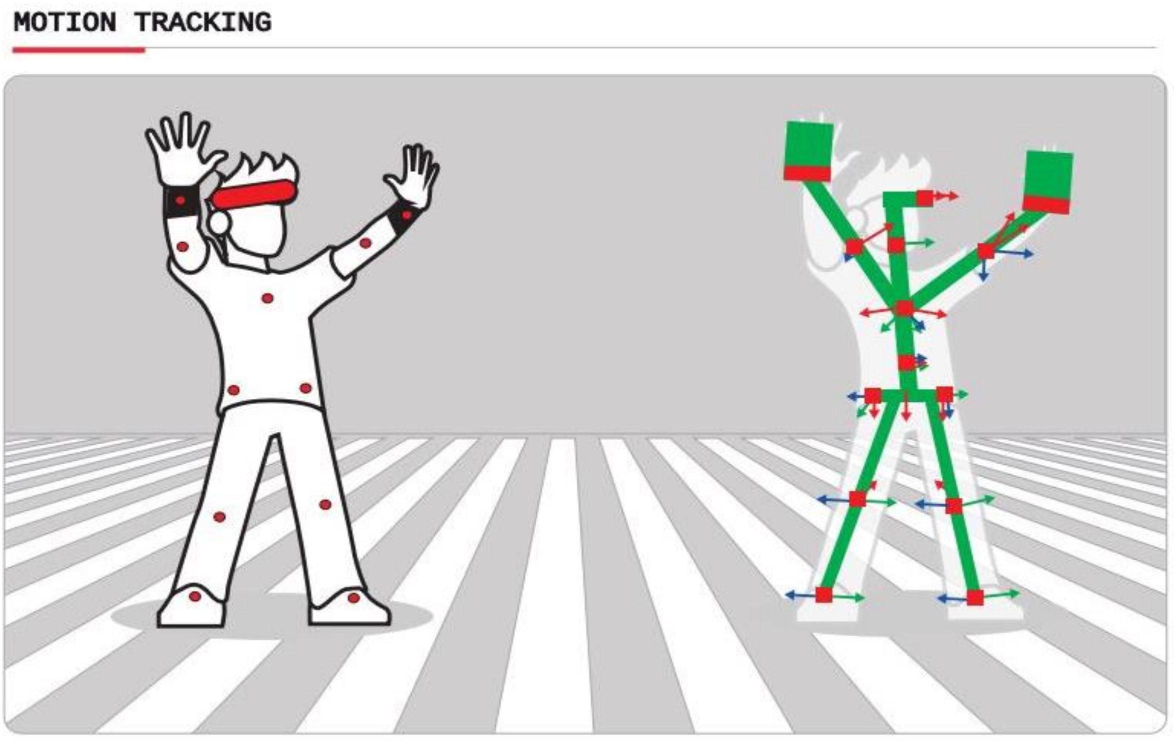


Illustration 29. Motion tracking. Each point must be well tracked in order to reproduce movement in the best way in VR (Source: 3dcoil.grupopremo.com)

7.2.1. Precision of the Tracking System Test

This test is aimed to test the precision of input devices.

The headset/camera is hardly testable, but it is not necessary if all the other devices are well positioned every time. The goal, therefore, is to know how good the other devices are being tracked.

When using Oculus Rift S, there is only two joysticks further than the headset, so this test is aimed to these two joysticks.

The test is the following: The two joysticks must be tracked. To track a joystick it is needed that, in the virtual scene, the 3D model of the joystick appears. This way and supposing that it is modeled properly and faithfully, the joystick will appear exactly in the same position it is situated in the reality. To know if it is well located, my idea is to mix the 3D model of the joystick and the real scene seen by the camera on the headset (in Oculus Rift S, it is called Passthrough+). The camera sees the real scene precisely, and the 3D model will appear in the place the tracking system says, so the superposed images will give us a good approach of how good the tracking system is (as the *Illustration 30*).

However, some things must be written down and highlighted:

- This test only works when the VR device has a camera and can superpose the images as said before and, as stated previously, when it is assumed that the 3D model is faithful.

- The same way, it can be used while moving the devices and not only when they are quiescent.

- The tester person can position where the real devices are and where the virtual ones are (let's not forget that he/she can see both), so measurement can be taken between one point in the real device and the same point in the virtual one. Furthermore, distances between devices can be measure too, so, in my opinion, it is a good test to do if the VR system allows to use it.[1]

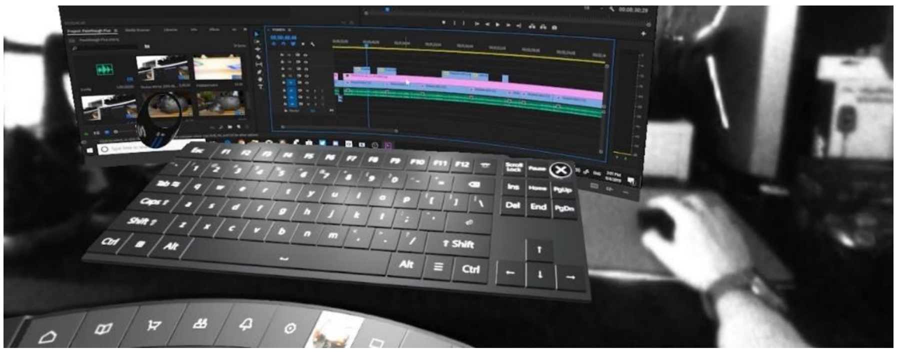


Illustration 30. Image seen while using Oculus Rift S Passthrough+ (Source: YouTube, Tyriel Wood)- See note [1] below

[1]*Note: The keyboard and the screen are virtual, but the hand and everything else is real and it is viewed through the camera.

7.3. Latency of Display and Refresh Rate

Before talking about these two topics, “Latency” and “Refresh Rate”, it is important to introduce which problem they are involved with.

Using an HMD, some images leave a kind of trail when doing fast spins with the head. After some time using these devices, some people start experimenting dizziness and sickness. With technology development, this have been decreasing.

This effect, according to Javier Salinas (audiovisual coordinator in the postgraduate area at CEU Educational Group), is due to three factors: *Persistence*, *Latency* and *Refresh Rate*.

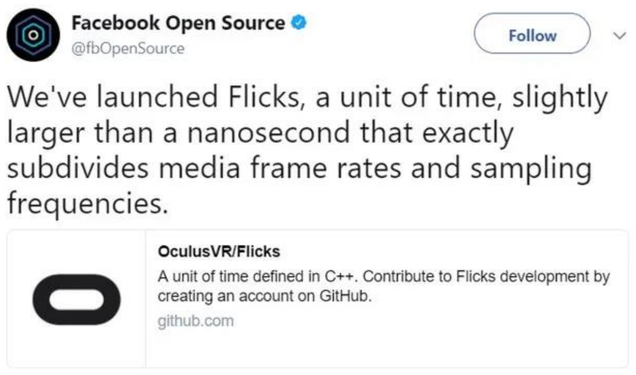
- **Persistence**: In VR, the persistence is the time that a pixel remains switched on. Higher the persistence, higher is the blur effect or the image wink. For having realistic sensations, low persistence is required (lower than 3 milliseconds).
- **Latency**: In VR, latency is the time elapsed since movement is produced (like a head spin) until the display reflects these changes. If the images late too much in going along with this movement, the latency is high, and the brain suffers a contradiction between what it is being seen and reality (although it a little difference).
- **Refresh Rate**: In VR, the refresh rate is the frequency or the number of times per second that a screen can update an image for giving a movement sensation. Its unity is Hz. Increasing the refresh rate of a screen, latency can be decreased. For example, a screen with a refresh rate of 60Hz has half the latency of a 30Hz one.

FLICK

“Flick” was developed by the Oculus engineers (Facebook VR team). What is “Flick”? “Flick” is a new time unit used to measure highly accurate the frame rate and its relationship with refresh rate.

“Flick” is the abbreviation of “Frame-Tick”. It is a very small-time unit, concretely 1/705600000 seconds. It is based on a work (2004) of La Sorbona University and the Institut National de l’Audiovisuel (INA). They stablished a time unit denominated “TimeRef”. One TimeRef equals to 50 Flicks.

According to its creators, “Flick” is the smallest time unit which is bigger than a nanosecond and can be expressed with whole numbers for frequencies of: 24 Hz, 25 Hz, 30 Hz, 48 Hz, 50 Hz, 60 Hz, 90 Hz, 100 Hz y 120 Hz and, in audio frequencies, 8 kHz, 16 kHz, 22.05 kHz, 24 kHz, 32 kHz, 44.1 kHz, 48 kHz, 88.2 kHz, 96 kHz y 192 kHz.


Facebook Open Source
@fbOpenSource
Follow

We've launched Flicks, a unit of time, slightly larger than a nanosecond that exactly subdivides media frame rates and sampling frequencies.

OculusVR/Flicks
A unit of time defined in C++. Contribute to Flicks development by creating an account on GitHub.
github.com

Illustration 31. Facebook Open Source Tweet about Flick

“Flick” was created because, when measuring the precise time that a photogram appears on the screen, the usual units gave fractions.

When it is necessary that everything is so accurate and precise (like in programming, VR…), “Flick” is a great option.

Comparing the Flick with FPS:

1 photogram, 24 fps = 29.400.000 Flicks

1 photogram, 25 fps = 28.224.000 Flicks

1 photogram, 30 fps = 23.520.000 Flicks

1 photogram, 48 fps = 14.700.000 Flicks

1 photogram, 50 fps = 14.112.000 Flicks

1 photogram, 60 fps = 11.760.000 Flicks

1 photogram, 90 fps = 7.840.000 Flicks

1 photogram, 100 fps = 7.056.000 Flicks

1 photogram, 120 fps = 5.880.000 Flicks

1 photogram, 8.000 fps = 88.200 Flicks

1 photogram, 16.000 fps = 44.100 Flicks

1 photogram, 22.050 fps = 32.000 Flicks

1 photogram, 24.000 fps = 29.400 Flicks

1 photogram, 32.000 fps = 22.050 Flicks

1 photogram, 44.100 fps = 16.000 Flicks

1 photogram, 48.000 fps = 14.700 Flicks

1 photogram, 88.200 fps = 8.000 Flicks

1 photogram, 96.000 fps = 7.350 Flicks

1 photogram, 192.000 fps = 3.675 Flicks

Programs, like Unity, have not implanted yet “Flick”. In my opinion, “Flick” is a very interesting concept that could help increasing resolution in VR (and not only in VR) because there will be no more accumulated error and the time responses will be more accurate.

7.3.1. Latency Test

The main idea of this test is to measure the screen latency in a good way, with numbers. It was really difficult to come up with the idea of something with which measure latency. It can be measured spinning fast the head, and different comparison could be made between different devices. This form is far away from being accurate, so it is not a good way anything that involves sensations.

This test needs a high-speed camera to be done, and it is about the following: A scene is programmed. In this scene, something happens (like the appearance of a sphere) when the tester touches a button. When the tester sees this event, he/she has to press another button. This way, latency can be measured well. However, there are some problems:

- ❖ The exact moment when the tester presses each button is not known. This problem can be easily solved because, with the high-speed camera, we can set the time when the finger starts moving or ends moving as the time when the tester sees the event appearance.

- ❖ The big problem here is that the user has a response time. When first pressing, this problem does not exist because it is the exact moment when we start measuring. The problem is when the tester sees the event and wants to press the button. Each person has a response time, so this time must be subtracted to the time used if we want to know the exact latency (or as accurate as possible). This time can be measured with the high-speed camera in another test.

7.4. Degree of Immersion

Immersion can be defined as the senses disconnection of the real world and the connection to the virtual world. Consequently, the user does not perceive the environment and starts to be immerse in the real world.

Eyesight is the main sense in immersion, but audition is relevant too because it helps with the immersion.

The key in immersion is stereoscopic vision, as long as it provides two images to the eyesight in order to produce a tridimensional sensation. This way deep can be seen.

The main goal in this thesis is to test the degree of immersion in different VR devices, but this is so difficult to test because the degree of immersion cannot be measured properly, it is all about the tester opinion. For this reason, only sensations can be measured while testing immersion, there is no other way as the tester impressions.

In next page is shown an example of this.

7.4.1. Immersion Test

The main idea was to program a scene where, suddenly, a big hole appears. Because of this, the tester gets shocked and, depending how shocked he/she is, so it will be the immersion level. The problem is that it depends on the realism, so it is not a good idea because we want to measure only immersion, not realism or sensations.

For the reason above, it has been searched about tests that could help to do an immersion test. In my opinion, the following test is a great test to test immersion, so I have decided to write it down here.

Immersion test (*Source: https://imotions.com/blog/measuring-virtual-reality-immersion-case-study/ -> See the* [2]*note at the end of the explanation*):

<u>*Test approach*</u>:

How immersive is the experience? With such unfathomable sums of money, and so many fields of use involved, there is a great need to quantify and substantiate the claims made by manufacturers and producers who promise the absorbing experience of VR.

This is something that biosensors are fully equipped to take on, both in terms of ease of use, and depth of information. Non-invasive sensors can unobtrusively record physiological information from a participant without causing distraction or discomfort. Such sensors also don't place any cognitive load on participants, helping them have an effortless experience.

But which biosensors should we use? For the following experiment we chose EEG (electroencephalography, a measure of brain activity), and GSR (galvanic skin response, a measure of electrical activity across the skin). We can see one of our participants in the image above, wearing a VR headset, EEG headset, and a GSR device.

With both of these sensors combined, synchronized and used within iMotions, we can get a robust measurement of the level of physiological arousal that someone is experiencing, and understand how their brain responds too (more about these measures below).

Facial expression analysis would be to put into the mix, but is of course rather difficult to perform with half the face obscured by the VR device, a way around this would be to include fEMG that measure muscle activity through electrodes.

<u>*The test*</u>:

To test how someone responds to the VR experience, we threw our willing participants (and me) onto a virtual roller coaster, with both a screen-based and VR setup (and thankfully for me, not a real-life setup, although that certainly would have had an impact).

Setting up the experiment is simply a matter of starting up iMotions, importing the video file, and getting strapped into the sensors – all in all it's about as straightforward as getting onto a roller coaster, just without the queuing (although there is of course some benchmarking to be done before you can begin).

<u>*Methods and Measures*</u>:

The GSR device measures the electrical activity that occurs across the skin. When we think of our palms sweating when we're nervous – they are. But it's not just when we're nervous, but when we are

[2] Note: iMotions is high tech software made to execute human behavior research with high validity. iMotions seamlessly integrates multiple biosensors that provide different insight; such as Eye Tracking, EDA/GSR, EEG, ECG and Facial Expression Analysis. The combination of different sensors and data sources allows you to make a clearer and more incisive understanding of human behavior. Through real-time measurements of nonconcious responses, iMotions provides the bigger picture on human actions, thoughts and feelings for you to tap into new innovation – getting you the results faster.

physiologically aroused by anything – if something creates intense feelings of stress or happiness, sadness or surprise, then our sweat glands increase in their activity too. As the sweat glands become more active, this also influences the level of electrical activity across the skin.

The intensity of emotions can be related to the level of GSR activity, although it can't tell us which emotion is being experienced. While there is always a certain level of activity – we don't switch on or off – there are peaks and troughs as the activity changes. It's the peaks in particular are interesting, that pass a certain threshold, as they can simply be counted and give a number to quantify the level of GSR activity (an example of how the peaks look is shown below).

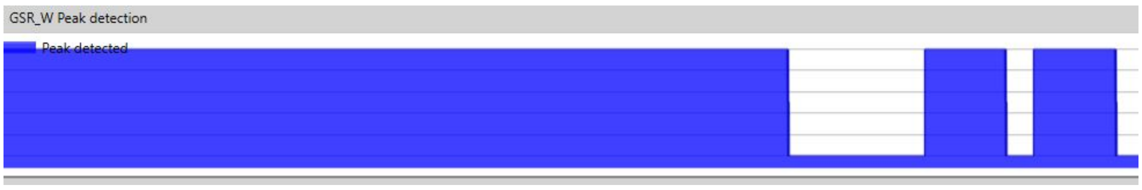


Illustration 32. GSR_W Peak Detection (Source: https://imotions.com/blog/measuring-virtual-reality-immersion-case-study/)

EEG devices also measure electrical activity, but of the kind that occurs in the brain. Whenever we think, or do anything, and even when we don't do anything, bursts of electrical activity are fired across the brain. This electrical activity isn't just part of the brain – it essentially is the brain. EEG offers an insight into this by measuring the voltage changes that occur at the surface.

Through some clever analytics and algorithms, these voltage changes can be translated into more immediately understandable data, such as "motivation / avoidance", and also "engagement / distraction", the latter of which we'll be looking at here.

With these measurements combined, we can make a judgement about how engaging, or distracted, the participant feels when on the virtual roller coaster according to the EEG recordings – and also support the intensity of the feeling through the GSR recordings.

***Results of an example experiment can be seen in**: https://imotions.com/blog/measuring-virtual-reality-immersion-case-study/*

The explanation above needs a program called "iMotion". This software is a high-tech software made to execute human behavior research with high validity. As said at the beginning of 7.4., to measure immersion it is needed high tech software because the measurements must be done to the tester sensations (cerebral signals, heartbeat, etc.). If it is wanted to measure immersion, there is no easy way to do that.

7.5. Field of View

Field of View (FoV) is the total area in which objects can be seen in the lateral view and in the vertical view (peripherical view)while focusing in a central point. This Field of View can be well seen in the next illustration (*Illustration 33*).

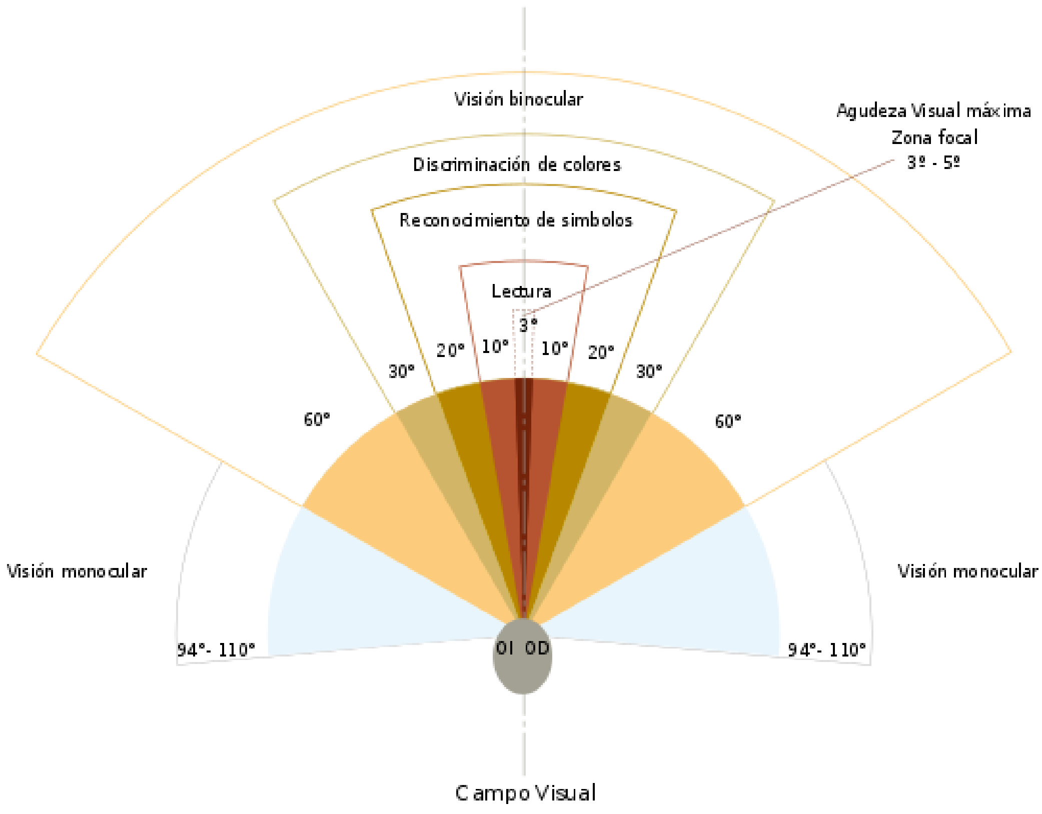


Illustration 33. Field of View (Source: Wikipedia)

In the previous illustration several angles can be appreciated. The lower angles are where a person has the maximum visual acuity. The higher the angle (measured from the center edge), the lower the visual acuity, until a point where the person cannot see anything.

The visual acuity can vary depending on distance to the eyes (as can be seen in *Illustration 33*). Depending on how far the object is, this object will be seen better or worse.

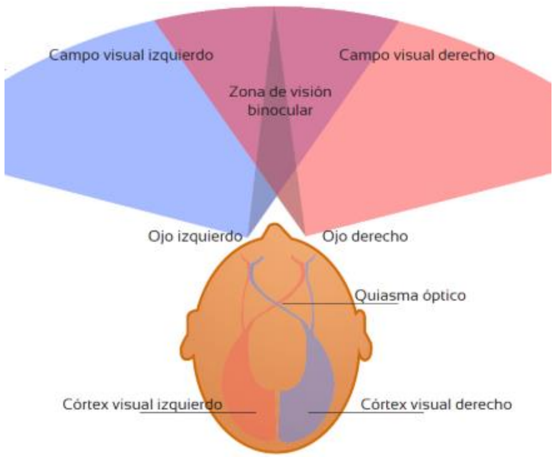


Illustration 34. Field of View (Source: tuoptometrista.com)

Basing on information got in *tuoptometrista.com* (a page from the Official Institute of Opticals and Optometrists in Andalucía, Spain), field of view reaches the following values:

- 91,5º in temporal direction (from the eyes to the ears), 75º down, 55º up and 64º in the nasal direction.

As seen in the *Illustration 34*, the field of view depends on both eyes. They are not necessarily equal eyes, so the field of view could be better in one direction than in another.

In VR, the important thing is that two projections must be projected, one to each eye. It is important to know that a VR device cannot overpass the field of view of a human because it is impossible for the user to see what it is being projected further from his/her maximum visual angle.

In next pages, a test about how to measure Field of View in VR is exposed.

7.5.1. Field of View Measurement Test

To measure the field of view, I have based on medical measurements. Three tests are the most relevant ones:

- **Confrontation visual field testing**: Confrontation visual field testing involves having the patient looking directly at your eye or nose and testing each quadrant in the patient's visual field by having them count the number of fingers that you are showing. This is a test of one eye at a time. It is useful for the examiner to close one eye so that one can determine if the patient is seeing appropriately in their visual field (according to *webeye.ophth.uiowa.edu, Ophthalmology and Visual Sciences, University of Iowa Health Care*).
- **Tangent screen exam**: The tangent screen exam (Goldmann field exam) can be conducted in your eye doctor's office. You will be seated about 3 feet away from a computer screen. This screen will have a target in the center for you to focus on throughout the test. The computer will generate images on different areas of the screen. Without moving your eyes, you will tell your doctor when you are able to see objects in your side vision. Your doctor will be able to use the information collected to form a map of your visual field. This will help them determine if there are certain areas in your visual field that you are not able to see. The location of these areas can help your doctor diagnose the cause of the visual field problems (according to *healthline.com/health/visual-field#tangent-screen-exam).*
- **Goldmann perimetry test**: A method of testing the complete visual field was developed by Hans Goldmann. His bowl-shaped perimeter uses bright light as targets superimposed on a white background. Targets may vary in size, luminance, and color. Goldmann perimetry requires trained perimetrists to measure and draw the visual field. Challenges include cost and inter-perimetrist variability. In practice, Goldmann perimetry is a form of kinetic perimetry: a stimulus is moved from beyond the edge of the visual field into the field. The location at which the stimulus is first seen marks the outer perimeter of the visual field for the size of the stimulus tested (according to *eyerounds.org/tutorials/VF-testing/, Ophthalmology and Visual Sciences, University of Iowa Health Care*).

Basing on the tests above, the idea is to measure the field of view is to rotate an object around the scene (in horizontal, vertical and in the directions wanted). When the tester starts seeing the object, he/she has to press a button and the angle of the object (related to the point where the user is) is stored.

However, one thing must be considered. The tester must not move the head, so it is good to deactivate the headset tracking system. The test can be done in all directions wanted and should be repeated sometimes to have a mean value, which is so much valuable than a unique value.

7.6. Interactivity

Interactivity is so related to immersion. The higher the immersion, the higher the interactivity with the VR device and with another person with a VR device that is on the scene too.

What is interactivity? Interactivity is every interaction that a person makes with another person and that involves a reaction from the other person. In terms of VR, interactivity mostly the same, but with the big difference that the action/interaction is made through a computer or a computerized environment. That is a “virtual interaction”, the other person is not there physically, it is needed a technological way to send and receive information.

If a technological way is needed to interact with the other (either if the other is in the scene or is seeing you in real life), the better realism sensation the technological way can produce, the better interaction between the users.

If the tester and the companion are together in real life, the virtual interaction can only be done through the scene, putting elements on it, making sounds or writing texts that appear in the scene and which the VR user could see.

If both are in the VR scene, the interaction can be made in all the same ways as before, but adding every other interaction that the VR systems allow to.

Knowing which is stated above, to interact better with any other person in VR it is needed that the immersion is the closer to reality. Because of this, the higher immersion, the higher interactivity in my opinion.

If the headset could send electrical signals to the user that produce every sensation that happens in VR as it were real, the interactivity will be the best possible (as TESLASUIT pretends to, *Illustration 35).*

Once defined the interactivity, it is the moment to test it.

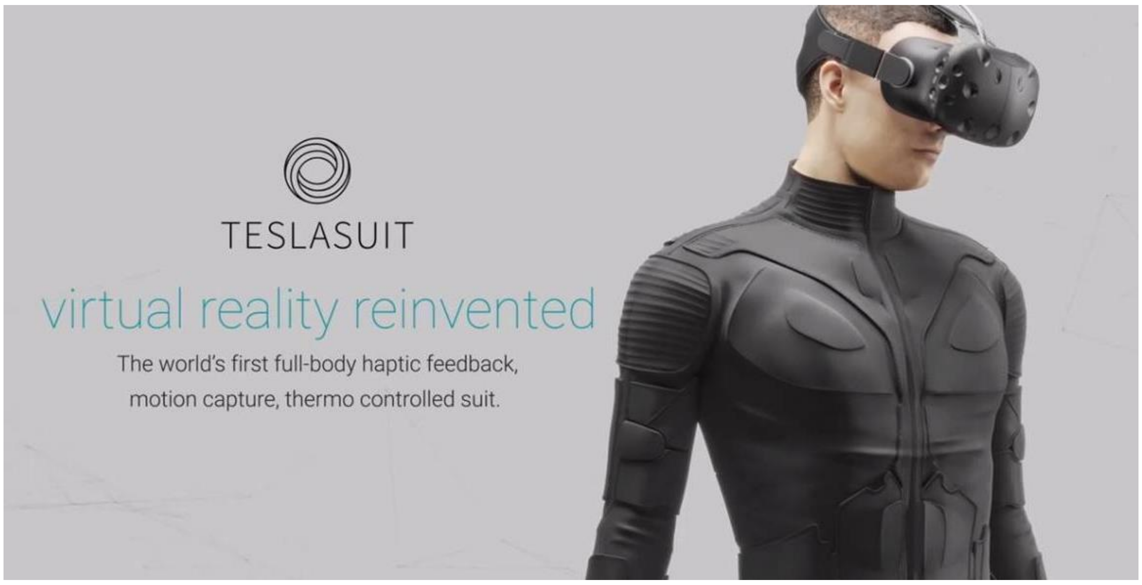

TESLASUIT
virtual reality reinvented
The world's first full-body haptic feedback,
motion capture, thermo controlled suit.

Illustration 35. TESLASUIT, a suit that allows the user to interact better with the virtual environment (Source: https://vandal.elespanol.com/)

7.6.1. Interactivity Test

The unique way I have found to test interactivity is the following:

If virtual interactivity consists on interacting in the best way possible with someone through a technological way, the best way to know if interactivity is good is to know which interactions are allowed and which are not allowed. Everything with which or through with the user wants to interact to can be written down. If it is possible to interact, we can put a tick ✓ and, if it is not possible, we can put a cross X.

To sum up, I am saying that the unique way (in my opinion) to test interaction is with comparisons between VR systems. We can only order them from least to greatest and, when developing or improving a VR system/device, check if it fulfills as much points as possible in order to make this device more interactable.

7.7. Image Duplication (Judder)

Image duplication (Judder) is a mix of smearing and strobing. Judder becomes apparent when the display is moved quickly, for example when moving the head fast. This problem can reduce in a really significant way the visual quality of a display and can produce dizziness. Its cause is a low refresh rate or a high persistence of the display (persistence is the length of time that each pixel remains lit, and can cause blurring effects and image smearing).

One easy example of judder is when seeing a film played with 24 frames per second (fps) (all films are typically played in 24fps). If a 60Hz screen is used to play this film, 60 gaps have to be filled with 24 images, and 60 is not divisible by 24. To solve this problem, 2:3 pull down was created.

2:3 pull down is an assignment system where each frame in one second (in a 24 frame per second) is assigned to a determined quantity of images, 2 images to the first frame in the second, 3 to the second, 2 to the third… (as shown in *Illustration 36*).

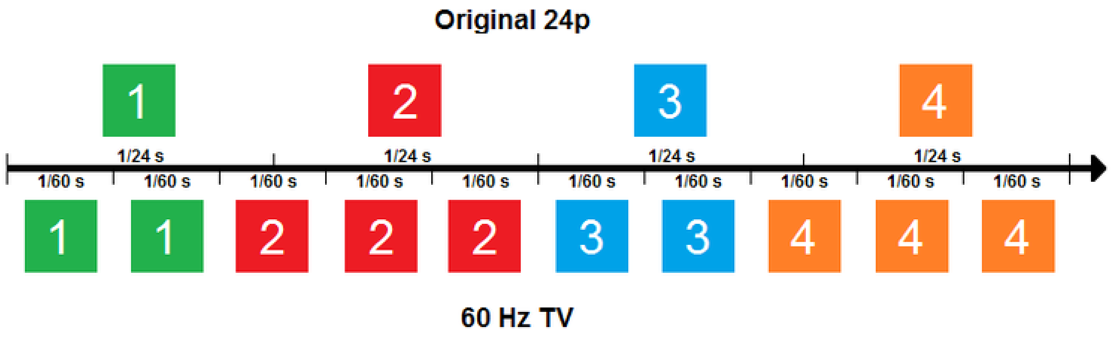


Illustration 36. 2:3 pull down assignment (Source: wikiversus.com)

The first image will appear on the screen 2 times (0,03 seconds in total), the second one 3 times (0,05 seconds in total)…

All gaps will be filled, but little skips will be appreciated due to the time each image is on the screen.

Known this, it is time to know what means judder in VR and which are its cons.

In judder, the illuminated area of each pixel in a screen sweeps a constant color across the eye's retina for the time it is lit (the persistence time stated in the first paragraph). It ends up in a smear, followed by a jump that causes strobing. All this results in a detail lost, probably in eye fatigue and maybe increased motion sickness.

Before continuing, strobing and smearing will be defined properly:

- **Smearing**: According to *xinreality.com*, smearing is the perception of motion blur that reduces the sharpness and detail of the image in VR. Smearing occurs when each pixel moves across the retina while it is lit. The longer the pixels are lit (full persistence) and the more movement of the HMD (quickly turning your head), the more smearing occurs. Smearing can be eliminated by either having really high refresh rate, about 1000 Hz is needed, or using a low persistence display, a more practical method employed by most HMDs.

- **Strobing**: According to *xinreality.com*, strobing is the perception of multiple copies of an image at the same time in VR. Strobing along with smearing is part of judder. These effects reduce the visual quality of a virtual reality display and cause simulator sickness. Strobing becomes more apparent when smearing is eliminated through the use of a low persistence display (smearing no longer hides strobing).
  Strobing can be eliminated by increasing the refresh rate of the display.

CLOSER APPROACH TO JUDDER

First, space-time diagrams have to be introduced:

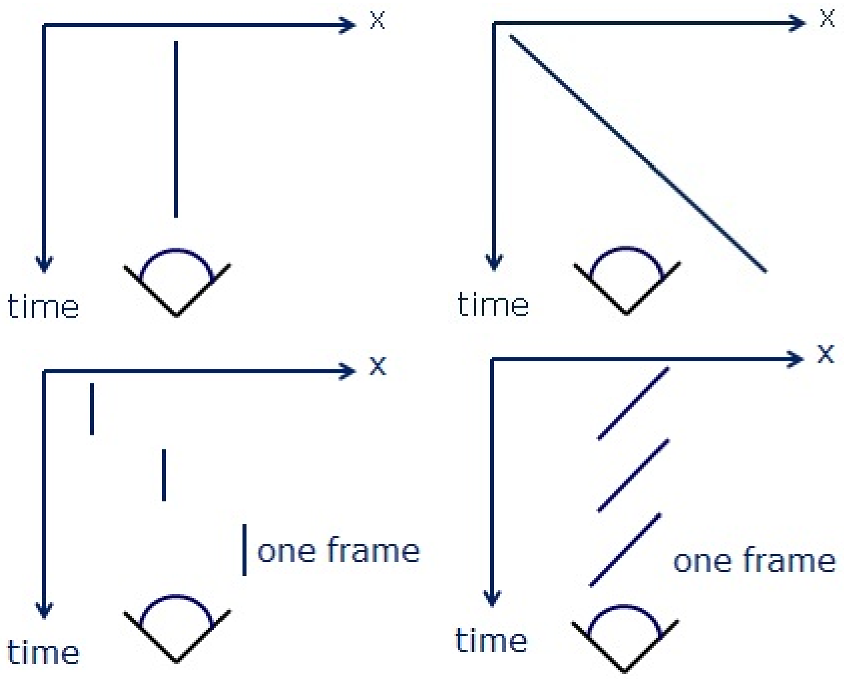


Illustration 37. Space-time diagrams. Diagrams 1 (up-left), 2 (up-right), 3 (down-left) and 4 (down-right) (Source: blogs.valvesoftware.com)

X is the relative position to the eye on the horizontal axis and time axis is just time.

The Diagram 1 shows a real-world object staying in the same position (relative to the eyes). It could also be a moving real-world object at the same speed the eye is moving, so, relatively, there is no difference to the quiet one.

The Diagram 2 shows a real-world object moving relative to the eye, for example, when staying in one position and seeing a bus going from left to right.

The Diagram 3 shows the same situation as in Diagram 2, but with a virtual-world object. With an infinite refresh rate, it would be like the Diagram 2.

The Diagram 4 shows the Diagram 1 situation with a virtual-world object. The object is fixed in a position relative to the eye at first, and the eye starts moving (moving the head, for example). The object has to be in its place, fixed in a position relative to the eye, but it cannot change position that easy. It depends on refresh rate, and does only change to its new position when the display refresh. If the head continues moving, the same happens again. If the display had and infinite refresh rate, it would be a vertical line. The explanation above is the reason why the pixel slides over the retina for a frame duration.

At this point, seems that judder is only smearing, there is no strobing. Images would only appear as stable smeared images. The following explanation is, in my opinion, a good one to see why virtual objects strobe and not only smear (Source: *blogs.valvesoftware.com*):

You might wonder why juddering virtual objects would strobe, rather than appearing as stable smeared images. One key factor is that any variation in latency, error in prediction, or inaccuracy in tracking will result in edges landing at slightly varying locations on the retina, which can produce strobing. Another reason may be that the eye's temporal summation period doesn't exactly match the persistence time. For illustrative purposes only, suppose that the persistence time is 10ms, and the eye's temporal integration period is 5ms (a number I just made up for this example). Then the eye will detect a virtual edge not once but twice per frame, and if the eye is moving rapidly relative to the display, those two detections will be far enough apart so that two images will be perceived; in other words, the edge will strobe. (In actuality, the eye's integration window depends on a number of factors and does not take a discrete snapshot.) Note, however, that this is only a theory at this point. In any case, the fact is that the eye does perceive strobing as part of judder.

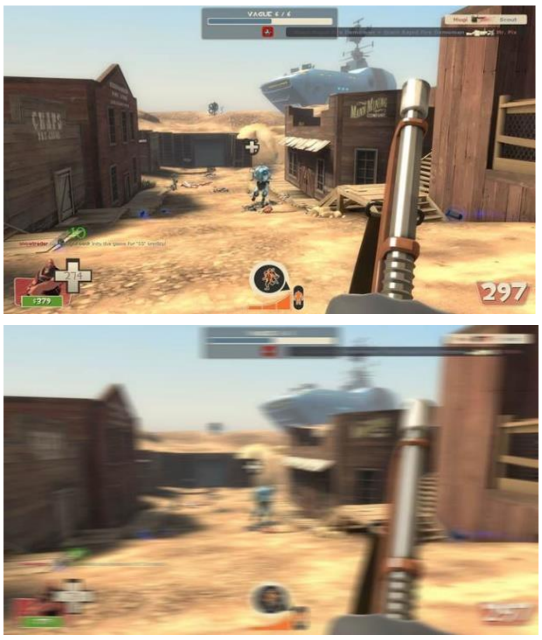


Illustration 38. Smearing example (Source: blogs.valvesoftware.com)

In the *Illustration 38*, a smearing example can be seen. Strobing cannot be appreciated, but it is difficult to appreciate it as stated before.

I will mention one more thing from the previous source:

The net effect of smearing and strobing combined is much like a choppy motion blur. At a minimum, image quality is reduced due to the loss of detail from smearing. Strobing tends not to be very visible on full-persistence displays – smearing mostly hides it, and it's less prominent for images that don't have high spatial frequencies – but it's possible that both strobing and smearing contribute to eye fatigue and/or motion sickness, because both seem likely to interfere with the eye's motion detection mechanisms. The latter point is speculative at this juncture, and involves deep perceptual mechanisms.

Judder can produce eye fatigue, dizziness and other health problems while using a display. For this reason, it is important to control it in every display and, of course, more in HMDs because of head movements (a person can move his/her head faster than he/she can track any object), because of the field of view (the field of view is bigger than in another display, so objects have to be tracked for longer) and because of virtual images (virtual images in an HMD appear to be directly in the world, not like on a usual display, where they appear to be on a surface in the world, so deviations are easily detected).

In the following pages it will be explained a judder test.

7.7.1. Judder Test

Judder cannot be measured properly, no number can be obtained while measuring it. Because it is all about human impressions, the unique way is to test it comparatively.

Real life has no judder, so it is the best circunstance that can happen while visualizing something on a display. The more clear-cut a transition is, the less judder is present.

Several videos can be displayed in order to test judder and, depending on which video is displayed and in which display it is played, the judder will be more or less appreciated.

My idea is to play a certain video made to test judder in every VR system we want to test. This video should be done in different frames per second. This way, differences between each of them can be appreciated better (the quality should be the better as possible).

More frames per second means that the video will be played with less judder (as explained in 7.7.).

The video chosen is as the following:

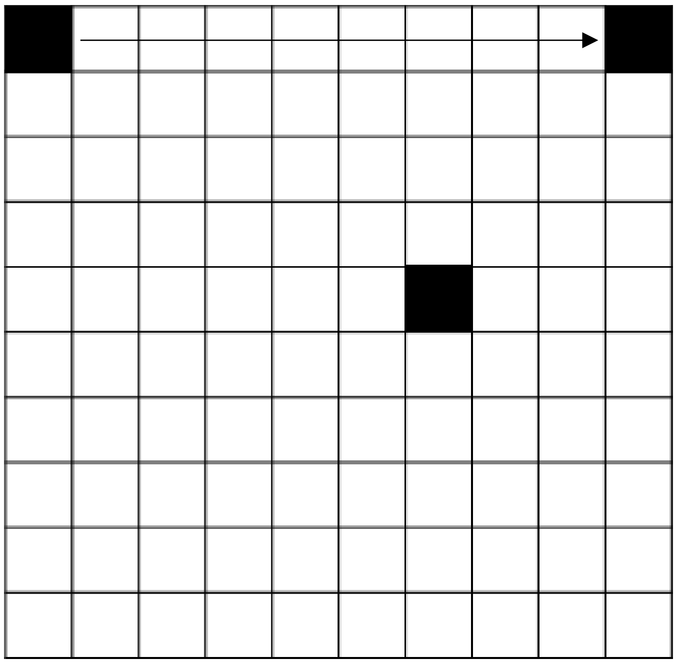


Illustration 39. Judder Test example

As the *Illustration 39* shows, there is a square "moving" through the grid from one point to the adjacent at a determined speed (determined by the frames per second of the vídeo). There is not a real movement, each square of the grid will blink at a determined moment and, in the next frame, this square will be switched off and the Depending on the display it is played, it will have more judder or less (the squares will leave a trail in their way).

In my opinion, this is the simplest way to test judder. Another way could be to play more complex videos and to see judder there.

Because VR is being tested, instead of squared could be cubes. Also tridimensional sensation will be felt then.

7.8. Sensation and Comfort

Sensation and Comfort involves all aspects written before, and a combination of every of the is needed to feel comfortable while using a determined VR system. However, in this aspect I will go deeper and I will talk about which bad sensations a VR system can produce if every previous aspect is not taken care.

Eye fatigue, dizziness and headaches are the main diseases that a VR system can produce. To avoid this the best way is to design the VR system in order to minimize judder. Furthermore, the higher resolution, the better sensations too.

According to Martin Banks (optometry professor at the California University), the eye growth could be affected by VR because of seeing a display so close. There are evident clues that involve displays as the main cause of myopia in the world.

Therefore, it is also important to moderate the use of VR HMDs because the display is so close to the eyes, specially on children, who are still growing.

Walter Greenlaf, a behavioral neuroscientist who has studied VR in medical environments for more than 30 years, has said that, in a virtual environment, the way we see and interact changes because we can project something on the eyes which is so far away but, actually, it is only some centimeters away from the eye. In his opinion, that's one of the main causes of dizziness, headaches and eye fatigue, our brain is not used to that sensation.

Some manufacturers like Oculus recommend to have a break of 10/15 minutes for each 30 minutes using VR HMDs.

Anyway, the goal of this thesis is to try to test all these diseases present in VR systems. In the next pages there is an idea of a test.

7.8.1. Diseases Test

To test diseases there is no other way than using a person who test different situations and writing down the results, it all subjective.

Apparently, it is a bit creepy to make someone to feel dizzines or headaches but, in my opinion, it is the unique way.

These things have to be tested:

- **Headache.**
- **Dizziness.**
- **Eye fatigue.**

Eye fatigue can be tested by playing a certain videogame and measuring time until the tester feels the fatigue. Doing this several times a mean value can be obtained and, compared to the mean values of another VR systems (it must be the same tester everytime), we could determine which one produces more eye fatigue.

Headache can be tested in the same way.

About dizziness I have thought about a test that could be faster than playing several hours. In a determined prefab scene, a rotation of the scene is programmed (with the headset motionless, deactivation the tracking system). We measure the time when the tester starts getting dizzy and we repeat the test some times to get a mean value. This test could be fast, but could be a little creepy too.

8. Tests Programmed with Unity & Instructions

The tests programmed with Unity and tested on people were the following:

- **8.1.** Readability Test: Based on the "Readability Test" stated above (7.1.1.).
- **8.2.** Image Discernment Test: Based on the "Image Discernment Test" stated above (7.1.2.).
- **8.3.** Precision Test: Based on the "Comparing Measures Test" stated above (7.1.7.).
- **8.4.** Field of View Test: Based on the "Field of View Measurement Test" stated above (7.5.1.).

In the following pages, these tests will be exposed, as well as their results.
First, the instructions to these tests can be found below, as well as testers data:

Tests Instructions

Readability Test

You will see a text on the scene. You can move the text through the plane by using the right controller joystick (it must be situated in front of your eyes). Next step is to augment the offset until you cannot read the text. At this point, another person will change the text and you have to get the text closer in little steps until the point you can read it. Press "B" button when you can read it.

When you have read it, augment the offset until the same point and another person will change the text to repeat the test.

This test will be done changing the text colors too.

Image Discernment Test

You will see two images on the scene. Both of them are the same image. The purpose of the test is to make the image in front of you to go further, until the point where you cannot discern any of the shapes which appear in the image. For doing this, use the right controller triggers (the bigger one to amplify the offset and the smaller one to decrease it). The image must be in front of your eyes, so use the right controller joystick to move it through the plane.

At this point, the image will change by pressing the space button on the keyboard (another person will do it).

The purpose is to get the image closer in little steps until the point you can discern an asked image. For doing so, you will be asked about one shape (which appear in the other image you have next to you) and you will have to say its position (using the same coordinates as exposed on the closest image). Press "B" button every time you stop.

Precision Test

You will see three cubes (1mx1mx1m each), two red cubes and one green cube. They have “lines” across them (each line measures 1m). Their colors are not relevant at all.

The purpose of this test is to measure one meter in every axis of the plane. You can select every line or every edge of the cubes to measure it in the most comfortable way for you. For doing so, position the controllers on the scene as they were a meter, using the joysticks upper position as the start and final point to measure (as it is shown in the joint image). You have to keep the position as good as possible until a person measures it with a meter.

Repeat this with every axis.

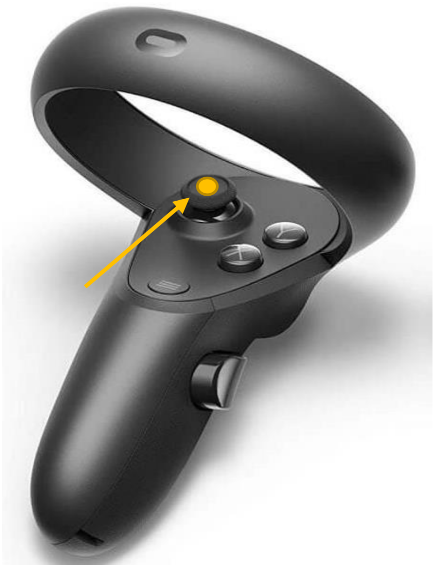


Illustration 40. Oculus Rift S controller (Source: Oculus)

Field of View Test

You will see a scene with a sphere in front of you and two cubes, one on your right side and one on your left side. You have to focus on the sphere with the reticle and move the right controller joystick horizontally to the left in little steps until you can see that one of these cubes appear. At this point, press “B” button. If you cannot see both cubes, continue doing the same until you can see the other.

Once finished this test, another person will change the scene by pressing the space on the keyboard. This scene is the same scene as the previous one, with the difference that vertical field of view will be measured instead of horizontal. Use the right controller joystick vertically (up) to move the cubes this time.

***Note: To recenter at any time in any test, press “Y” on the keyboard or on the controller**

Testers data:

Tester	Age	Glasses
1	51	Long distance glasses (worn during tests)
2	27	No glasses or lentils
3	35	Contact lenses (worn during tests)
4	26	No glasses or lentils
5	58	Close distance glasses (not worn during tests)
6	54	Long distance glasses (worn during tests)

Table 1. Information about the testers.

8.0. Relevant Note Concerning All Tests

All tests have had the same issue. The positioning of the tracking system was displaced to the point it was assigned. To correct this, a C# script was set to the Tacking System in which it was corrected. The code is the following:

```csharp
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class CenterCamera : MonoBehaviour
{
   // Start is called before the first frame update
   public GameObject PointZero;
   void Start()
   {
       transform.position = new Vector3(-PointZero.transform.position.x, -PointZero.transform.position.y, -PointZero.transform.position.z);
   }

   // Update is called once per frame
   void Update()
   {
       if (Input.GetKeyDown(KeyCode.Y) || OVRInput.Get(OVRInput.Button.Two, OVRInput.Controller.LTouch))
       {
           transform.position = new Vector3(-PointZero.transform.position.x, -PointZero.transform.position.y, -PointZero.transform.position.z);
       }
   }
}
```

The position is set to the “minus” coordinates where the CenterEyeAnchor appears, so that the position of the Tracking System and of the controllers is good. Thus, the PointZero GameObject is the CenterEyeAnchor, and the assigned GameObject is the Tracking System GameObject.

The position is set at the start of the program and at every update when the user presses the “Y” button (either on the keyboard or on the controller).

8.1. Readability Test

This test consists in measuring how far can a certain person read a text using Oculus Rift S. For doing so, a text is set with a white background. The text will be in a certain color, which can be changed as well as the text. From a large distance and getting the text closer in little steps, the goal is to measure at wich distance (offset) the tested person can read it. The text will be changed everytime so that the tested person does not know its content (could be easy to read if he/she does know it). Once done this with one text color, it will be changed (the background will remain white). **Text Font:** Arial**. Character Size:** 0'3. **Font Size:** 60.

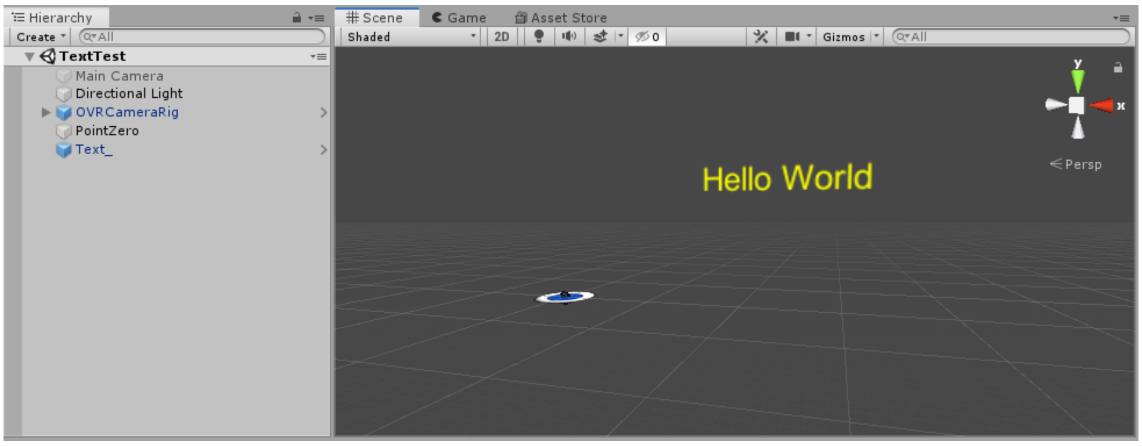


Illustration 41. Unity Test: Text to be read and point (0, 0, 0) where it has to be read

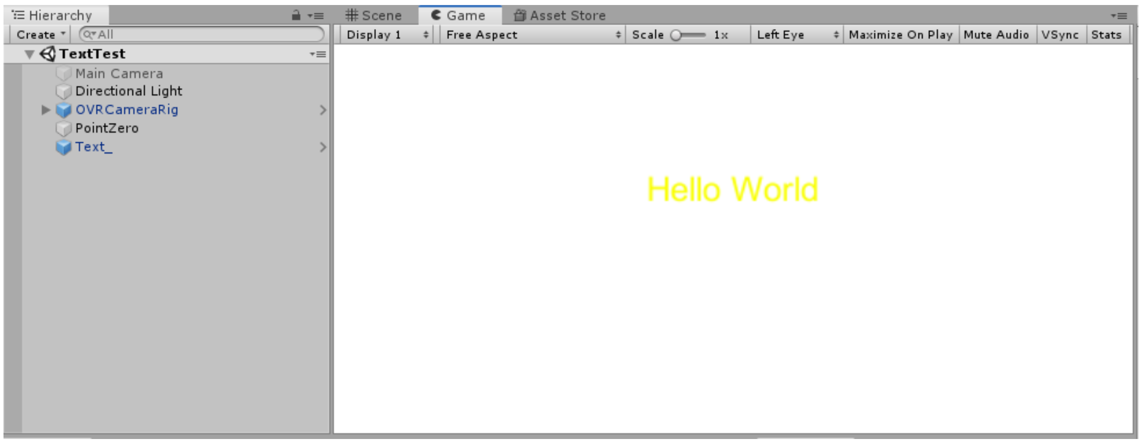


Illustration 42. Unity Test: Same as above, but here the white background can be appreciated

8.1.1. C# Code

Then, the "Text Controller" can be seen. This is a little example about programming in C# and how the controller was programmed. Below, some tips will be seen:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class TextController : MonoBehaviour
 {

   public float offset = 10.0f;
   public int textSize = 60;
   public float speed = 0.2f;
   public string text = "Hello World!";
```

```
    private float X_Component = 0.0f;
    private float Y_Component = 0.0f;

    private int i = 0;

    private int j = 0;

    // Start is called before the first frame update
    void Start()
    {
        // Instructions about which button to press
        Debug.Log("Press Y to see font size and B to see the distance to the eyes");

        Debug.Log("Press A to see the coordinates X/Y of the text");

        Debug.Log("Press right hand triggers to modify offset");

        Debug.Log("Use right joystick to move the text through plane XY");

        Debug.Log("Press space to change color once you start seeing it bad to
  compare between colors");

        GetComponent<TextMesh>().color = Color.yellow;
    }

    // Update is called once per frame
    void Update()
    {

        if (i == 0)
        {
            // Message for seeing if controllers are active or not
            if (OVRInput.IsControllerConnected(OVRInput.Controller.RTouch))
            {
                Debug.Log("Right controller is active");
            }
            else
            {
                Debug.Log("Right controller is not active");
            }

            if (OVRInput.IsControllerConnected(OVRInput.Controller.LTouch))
            {
                Debug.Log("Left controller is active");
            }
           else
            {
                Debug.Log("Left controller is not active");
            }
        }
        i=1; //It will be only processed once

        UpdateFontOffset();

        GetComponent<TextMesh>().fontSize = textSize;
        GetComponent<TextMesh>().offsetZ = offset;

        MovementText();

        ChangeColor();

        ChangeText();



    }

    void ChangeText()
    {
        GetComponent<TextMesh>().text = text;
    }
    void UpdateFontOffset()
    {
        if (OVRInput.IsControllerConnected(OVRInput.Controller.LTouch))
```

```
      {
      float fl = OVRInput.Get(OVRInput.Axis1D.PrimaryIndexTrigger),
OVRInput.Controller.LTouch);
      if (fl != 0.0f)
      {
          textSize += 1;
      }
      else if (OVRInput.Get(OVRInput.Axis1D.PrimaryHandTrigger,
OVRInput.Controller.LTouch) != 0.0f)
      {
          textSize -= 1;

          if (textSize < 1)
          {
              textSize = 1;
              Debug.Log("You can´t reduce font size more");
          }
      }
      }
      float fl2 = OVRInput.Get(OVRInput.Axis1D.SecondaryIndexTrigger,
OVRInput.Controller.LTouch);
      if (fl2 != 0.0f)
      {
          Debug.Log("False.");
      }

      if (OVRInput.IsControllerConnected(OVRInput.Controller.RTouch))
      {
        if (OVRInput.Get(OVRInput.Axis1D.PrimaryIndexTrigger,
OVRInput.Controller.RTouch) != 0.0f)
          {
              offset += speed * OVRInput.Get(OVRInput.Axis1D.PrimaryIndexTrigger,
OVRInput.Controller.RTouch);
          }
          else if (OVRInput.Get(OVRInput.Axis1D.PrimaryHandTrigger,
OVRInput.Controller.RTouch) != 0.0f)
          {
              offset -= speed * OVRInput.Get(OVRInput.Axis1D.PrimaryHandTrigger,
OVRInput.Controller.RTouch);

              if (offset < 0.0f)
              {
                  offset = 0.0f;
                  Debug.Log("You can´t reduce offset more");
              }
          }
      }

      // Press Y to see font size  and B to see the distance to the eyes
      if (OVRInput.Get(OVRInput.Button.Two, OVRInput.Controller.LTouch))
      {
          Debug.Log("Font size is " + textSize);
      }

      if (OVRInput.Get(OVRInput.Button.Two, OVRInput.Controller.RTouch))
      {
          Debug.Log("The distance to the eyes is " + offset);
      }
  }
  void MovementText()
  {
      X_Component = OVRInput.Get(OVRInput.Axis2D.PrimaryThumbstick,
OVRInput.Controller.RTouch).x;
      Y_Component = OVRInput.Get(OVRInput.Axis2D.PrimaryThumbstick,
OVRInput.Controller.RTouch).y;

      //Update the position
      transform.position = transform.position + new
Vector3(speed*X_Component*Time.deltaTime, speed*Y_Component*Time.deltaTime, 0);

      // Press A to see X/Y coordinates
      if (OVRInput.Get(OVRInput.Button.One, OVRInput.Controller.RTouch))
      {
```

```
158             Debug.Log("Coordinates X/Y are: " + transform.position.x + " / " +
159 transform.position.y);
160         }
161     }
162 
163     void ChangeColor()
164     {
165         if ((OVRInput.Get(OVRInput.Button.PrimaryThumbstickDown,
166 OVRInput.Controller.RTouch)))
167         {
168             if (j == 0)
169             {
170                 GetComponent<TextMesh>().color = Color.black;
171             }
172             else if (j == 1)
173             {
174                 GetComponent<TextMesh>().color = Color.blue;
175             }
176             else if (j == 2)
177             {
178                 GetComponent<TextMesh>().color = Color.red;
179             }
180             else
181             {
182                 j = -1;
183                 GetComponent<TextMesh>().color = Color.yellow;
184             }
185 
186             j++;
187         }
188     }
189 }
```

- **Lines 1-3:** Unity libraries.
- **Line 4:** Main class.
- **Lines 7-16:** Variable declaration. “Speed” is to reduce the velocity of the movement if necessary (that is why it is a “public” variable, can be modified in Unity).
- **Line 19:** Start function. Only executed once at the beginning.
- **Lines 22-30:** Debug.Log(“…”) shows its content while executing.
- **Line 37:** Update function: Executed every frame.
- **Lines 43-59:** Shows a message at the beginning if the controllers are active. OVRInput is how to call the Oculus controllers.
- **Lines 63-72:** Call to the functions below and assignment of the current text size and offset to the variable used.
- **Line 78:** One of the functions is defined (ChangeText( )).
- **Line 84-88:** A float number is created if the controller is active and, if the user presses the OVRInput.Get(OVRInput.Axis1D.PrimaryIndexTriggerOVRInput.Controller.LTouch) the float number assigned to this trigger takes a value (if it is not pressed will be 0). OVRInput can be found in the Oculus scripting manual for Unity, where ways to call the controller buttons can be found.
- **Lines 88-132:** If the user presses the triggers, offset and font size value will be modified depending how much it is pressed (multiplied by the speed to change its velocity if needed).
- **Lines 133-142:** If these buttons are pressed, the messages will appear on the screen with the current values of offset and font size.

- **Lines 144-162:** MovementText function. It takes the vector the joystick returns and associates it to a coordinate. Next, it updates the text position using transform.position = … and shows the coordinates if wanted.
- **Line 153:** Time.deltaTime adapts the speed to the user because, using each frame to increase speed, would not be adapted to the reality.
- **Lines 163-187:** ChangeColor function. It changes the color when pressing a certain button.

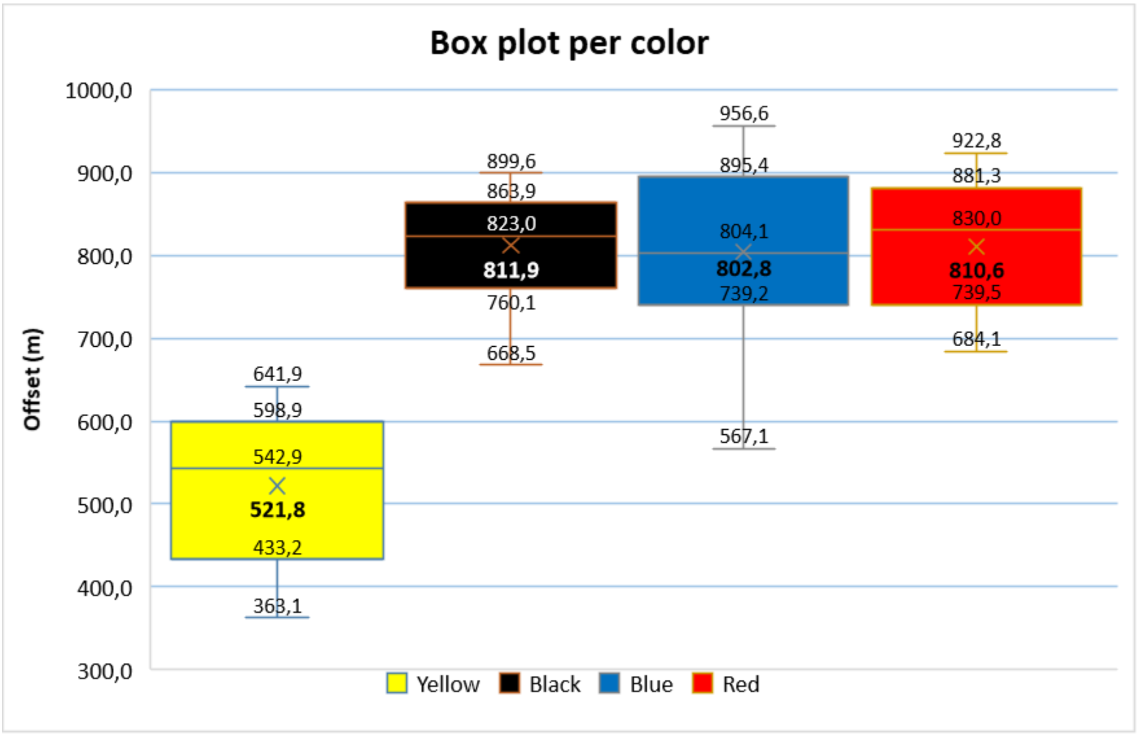
8.1.2. Results and conclusions

Readability Test								
Testers	Angebot	Aufgabe	Bahnhof	Tschüss	Entschuldigung	Katze	Fuβball	Gemüse
1	427,7045	440,6289	863,6541	808,5024	775,6668	899,375	812,7875	736,2098
2	547,2335	641,9143	847,6528	899,5808	782,3745	661,2624	885,1077	850,82
3	592,4265	554,2515	736,1675	752,0576	785,7169	819,8624	870,0258	847,1375
4	430,7324	363,1214	837,5157	668,4928	567,1075	727,1035	684,1404	701,5819
5	621,4269	601,0624	784,4088	863,9638	938,8264	956,6269	922,7884	892,9126
6	538,499	503,0606	808,4631	872,0198	883,3608	852,1479	774,4116	749,3804
Average per word	526,3371333	517,33985	812,977	810,7695333	788,84215	819,39635	824,8769	796,3403667
Variance per word	6572,619729	10752,50521	2221,7888	7632,713223	16164,0986	11965,20117	7550,49936	5934,933195

Table 2. Results obtained, averages and variances. ***Unit:*** *Meters*

The words used are: Angebot (yellow), Aufgabe (yellow), Bahnhof (black), Tschüss (black), Entschuldigung (blue), Katze (blue), Fuβball (red) and Gemüse (red).

If all colors are analyzed, the graphic obtained is as the next one:



Graphic 1. Box plot per color.

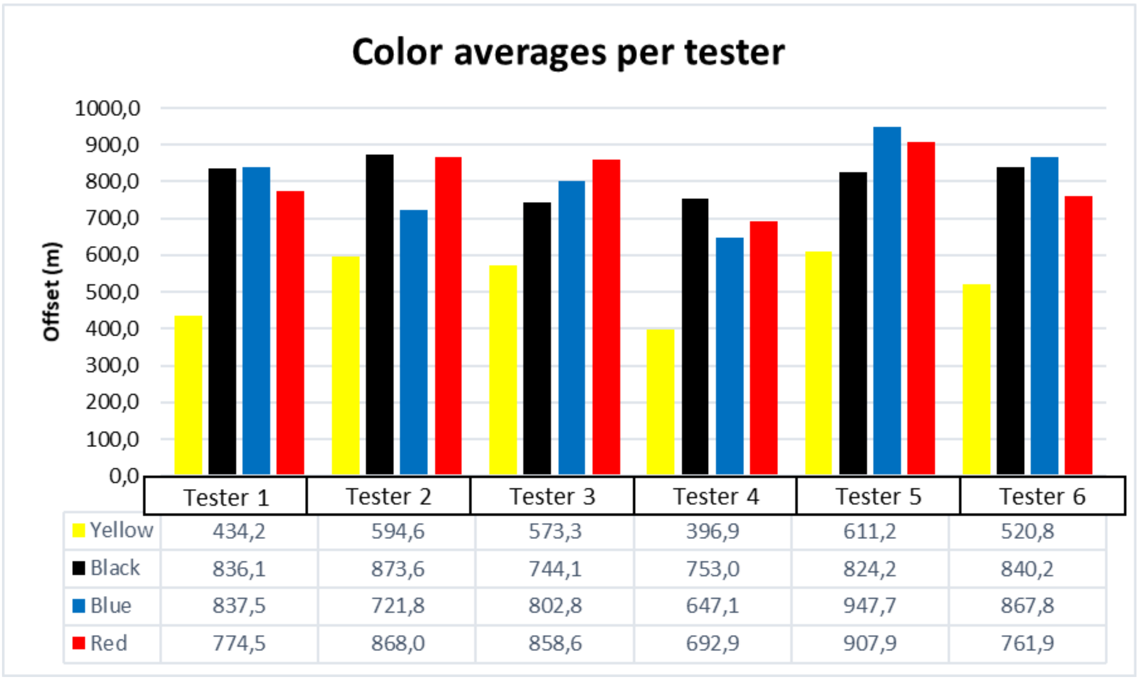
Easily, some important aspects can be seen in "*Graphic 1*". Yellow is the worst color in order to bee readen (521,8m). Furthermore, the other three colors are, on average, mostly equally readable. If this user study is taken as a good sampling, black would be the most readable color on average (811,9m), followed by red (810,6) and, then, by blue (802,8).

It can be seen too that blue and yellow have more variance than the other two because their values are more scattered.

Nevertheless, it can only be talked about averages to come to a conclusion, because almost the 25% of the blue values are over the top black value but, on average, black average is bigger because its concentration next to the average value. Blue has a very scattered first quartile, having some values with low numbers.

Red is more concentrated than blue and, in a first view, could seem to be more readable than black. In my opinion, following this graphic, it is not clear which one is the most readable one, but it is very clear that yellow is the less readable one by far.

Next graphic is the color average per tester (*Graphic 2*):

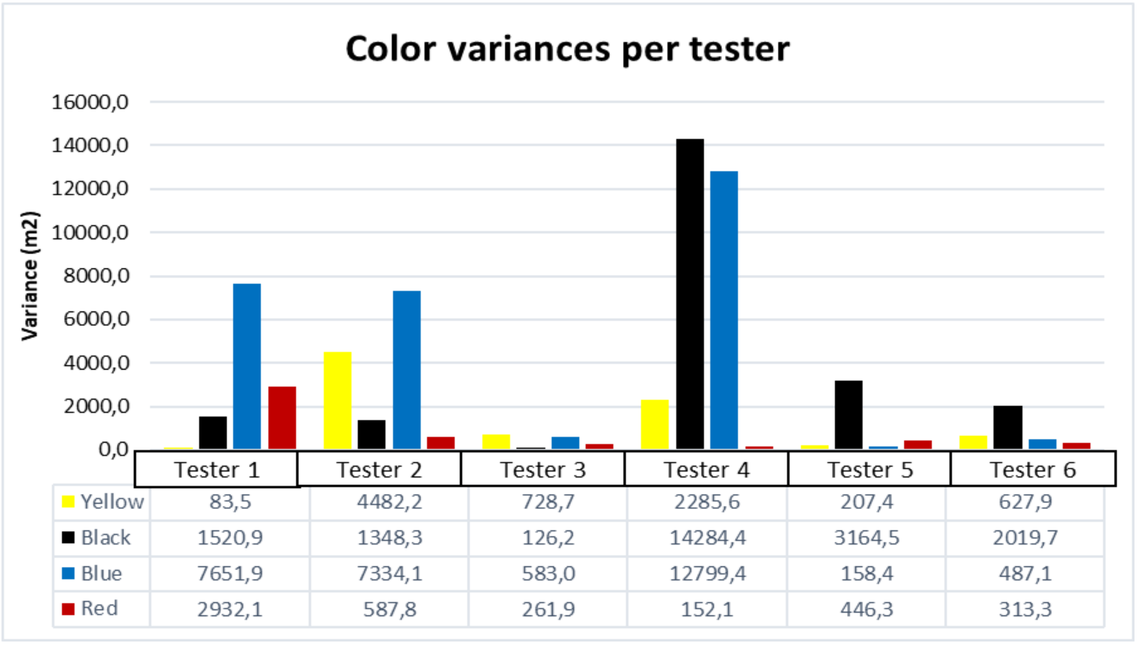


	Tester 1	Tester 2	Tester 3	Tester 4	Tester 5	Tester 6
🟨 Yellow	434,2	594,6	573,3	396,9	611,2	520,8
⬛ Black	836,1	873,6	744,1	753,0	824,2	840,2
🟦 Blue	837,5	721,8	802,8	647,1	947,7	867,8
🟥 Red	774,5	868,0	858,6	692,9	907,9	761,9

Graphic 2. Color average per tester.

As shown in "*Graphic 2*", same as said before can be appreciated. All tester seem to have the same readability ability, except number 4, who seem to discern worse.

If the next graphic is looked (*Graphic 3*), the results of all tester seem to be precious little scattered, except number 4 again. Number 3 has the best variances, the values of each color are very close between them, which should be the best situation because once someone can discern a word, this person should be able to discern any other one without varying too much the distance. Even so, higher variances are present mostly in black and blue colors, so "Entschuldigung" and "Tschüss" could be harder to read compared to the other words. Number 5 and 6 have very good variances too (low variances) but, because of having only two values of each color, variances are not really representative.

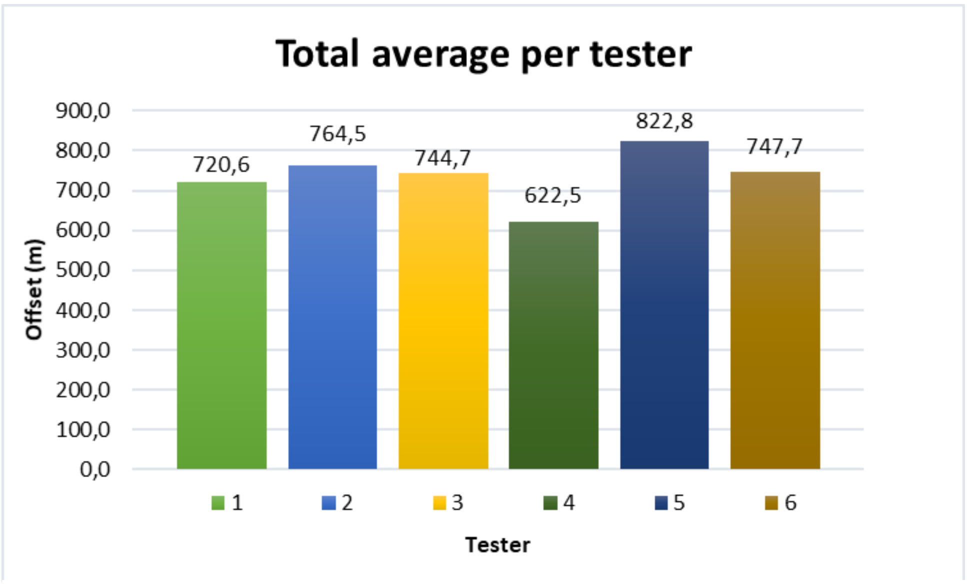


	Tester 1	Tester 2	Tester 3	Tester 4	Tester 5	Tester 6
🟨 Yellow	83,5	4482,2	728,7	2285,6	207,4	627,9
⬛ Black	1520,9	1348,3	126,2	14284,4	3164,5	2019,7
🟦 Blue	7651,9	7334,1	583,0	12799,4	158,4	487,1
🟥 Red	2932,1	587,8	261,9	152,1	446,3	313,3

Graphic 3. Color variances per tester.

Considering the data as representative data, tester 3 seem to be the one who has done better the test in general. The averages obtained with low variances are more aproximated to reality than the others, becaused one of the values with high variances surely is not really valid because of being so far from the other, and values with low variances should be better attending to this aspect.

Next graphic (*Graphic 4*) is the total average per tester:



Graphic 4. Total average per tester

In *Graphic 4* how good each tester can discern word can be seen. These values are not valid for anything else than this, because values used to do these averages have nothing in common. However, if a tester sees better than other it is expected his/her total averageto be higher. This really makes sense looking at *Graphic 2*, tester 4 is, on average, who discern words worse and tester 5 who discern better than the others. Looking at page 52 (*Table 1*), it seems that age is not relevant at all. However, it could seem that the 4th tester sees worse than the others, so this tester could need to wear glasses because his/her vision is the unique which is a bit different. Therefore, this test can also help to see if a tester sees bad compared to the others and if that person could need to wear glasse or to have his/her sight calibrated.

In conclusion, the test seems to be valid for being repeated in other VR devices because results seem to be good enough.

Firstly, differences between colors can be found. Yellow is the less readable color and this can be easily spotted. Probably, with a really representative sample results would leave clear the color order in terms of readability (or if they are equally readable). Anyway, comparing color averages using the same colors and words (and font size, font and character size [*page 55*]) is possible between VR systems, so the goal of this test is completed.

Secondly, as said before, the sample is not really representative so more words for each color should be read in order to have a better average and in order to avoid possible errors that could ruin the color average because of only having two values.

Finally, in my opinion, is good to use similar words. As stated before, some words as "Entschuldigung" and "Tschüss" seem to be hard readable than the others, so this could help in order to make the tests better.

8.2. Image Discernment Test

The Image Discernment Test consists in picking out an the correct image that the tester says to the tested person. The tested person will decrease its offset to the image by using the controllers and, at the moment he/she can discern the image, results of the offset will be taken.

At first, offset will be increased and, when the person does not see well the image, it is changed to the test one (to position the image another image will be used and, then, substituted in its position). Next, the tested person will be asked for a certain image, and offset has to be decreased until he/she can guess the image coordinates. Finally, the test has to be repeated with another image to guess (when guessed, the test table will be changed rapidly in order to avoid the tested person knowing its image position, which could influence).

The images are the followings:

<u>Table</u>	1	2	3	4	5
A	✯	💧	✿	■	➳
B	⚛	﹏	♠	☢	♪
C	?	◍	🧩	↪	◎
D	△	⬣	✉	▬	○

<u>Test Table</u>	1	2	3	4	5
A	△	?	↪	○	💧
B	☢	✿	⬣	◎	⚛
C	﹏	■	✯	◍	✉
D	♠	♪	🧩	➳	▬

Illustration 43. Images to discern (down) and images to be compared (up)

Illustration 43 shows the images that the tested person will see at any time (up) and the images that have to be discerned (down) by decreasing the offset.

This test is a little different from the test shown in 7.1.2. In the other test the image is a medical image and the offset was constant, so a distance cannot be measured properly. For this reason, this test was developed in another way, similar to the previous one but, definitely, mostly the same way.

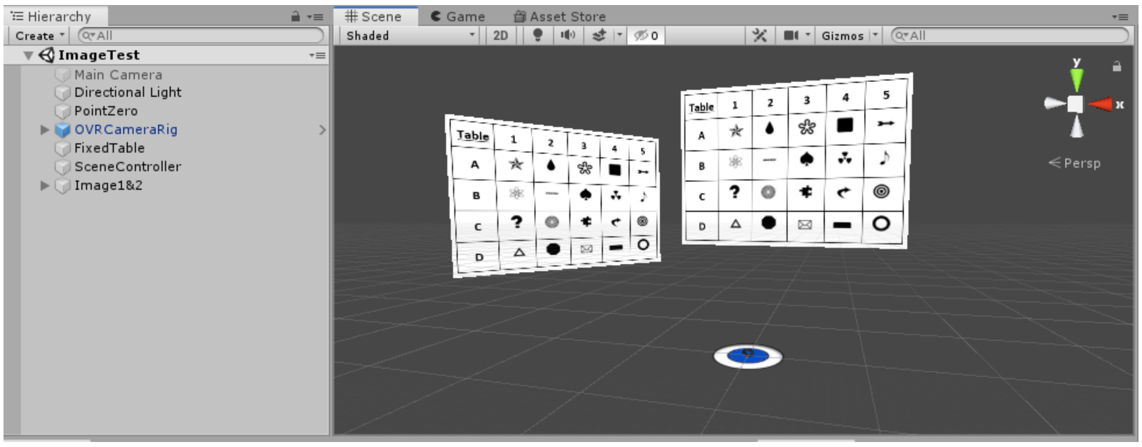


Illustration 44. Unity Test: Images and point (0, 0, 0) where they have to be discerned

8.2.1. C# Code

The code to move the image is the same as in 8.1.1.. Below, it will be written the code used to change between images:

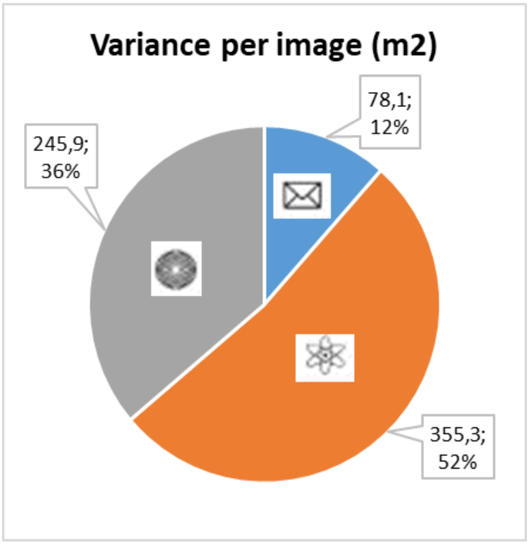
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Scene_Controller : MonoBehaviour
 {
    public GameObject Image1;
    public GameObject Image2;

    private int i = 1;

    // Start is called before the first frame update
    void Start()
    {
        Debug.Log("Press the space to change between images");
        Image1.SetActive(true);
        Image2.SetActive(false);
    }

    // Update is called once per frame
    void Update()
    {
        //When pressing space button...
        if(Input.GetKeyDown(KeyCode.Space))
        {
            if (i == 0)
            {
                Image1.SetActive(true);
                Image2.SetActive(false);
            }
            if (i == 1)
            {
                Image1.SetActive(false);
                Image2.SetActive(true);
            }

            i++;

            if (i > 1)
            {
                i = 0;
            }
        }
    }
 }
```

- **Lines 7-8:** GameObjects definition. They have to be set in Unity to pick which GameObjects are the correct ones to use (the user drags a GameObject into the proper cell).
- **Lines 16-17:** GameObject.SetActive(false) hides the Game Object assigned to it. GameObject.SetActive(active) makes the GameObject visible again.
- **Line 24:** Input.GetKeyDown(KeyCode.Space) returns "true" when space button is pressed (keyboard).

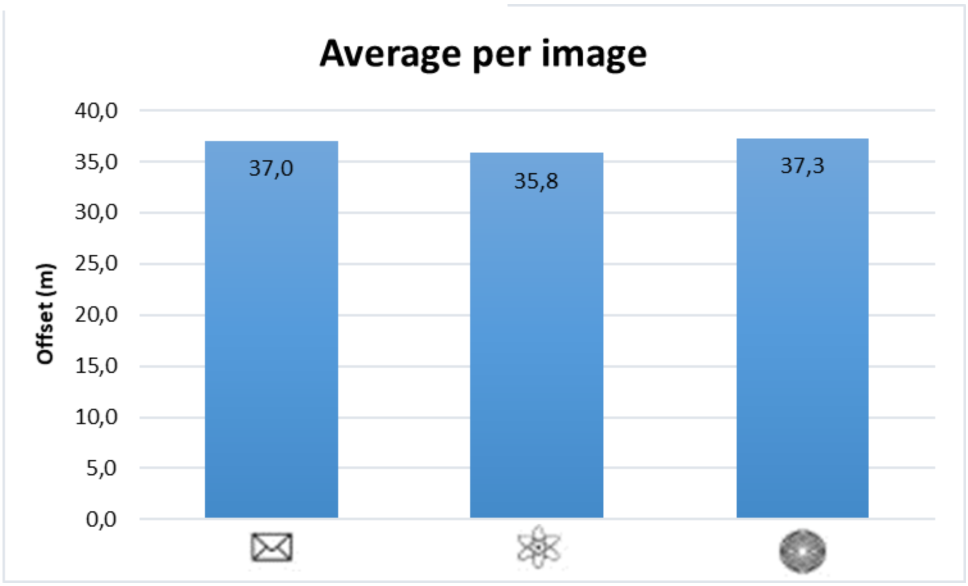
8.2.2. Results and conclusions

Image Test			
Testers	**1** ✉	**2** ⚛	**3** ●
1	37,57393	30,87724	48,12783
2	22,05367	29,65596	38,34255
3	41,09803	21,74779	22,72295
4	Failed Test	16,61129	14,33635
5	39,19561	49,10459	45,32814
6	45,18763	66,97417	54,81482
Average per image	37,021774	35,8285067	37,2787733
Variance per image	78,09291094	355,298021	245,873134

*Table 3. Results obtained, averages and variances. **Unit**: Meters.*



Graphic 5. Variance per image

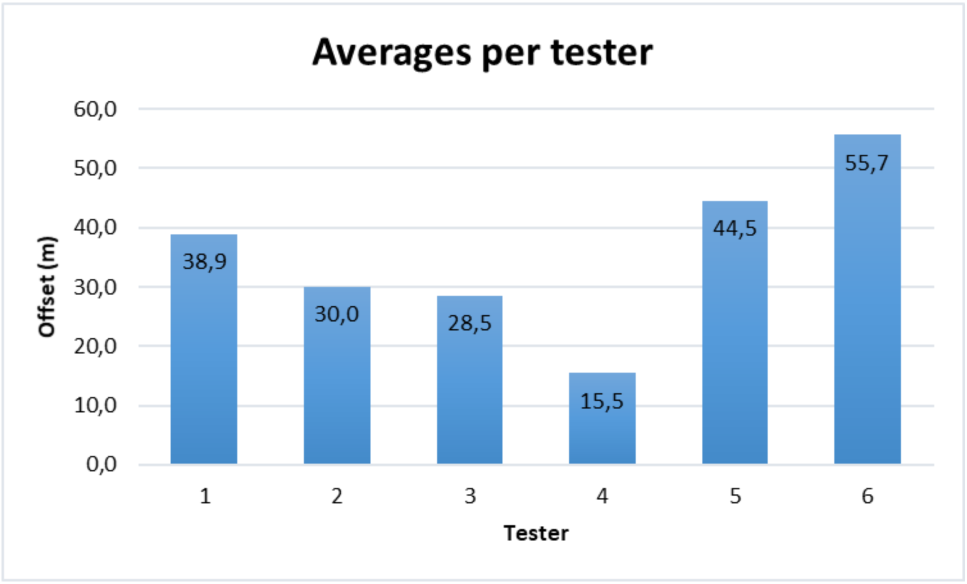


Graphic 6. Average per image.

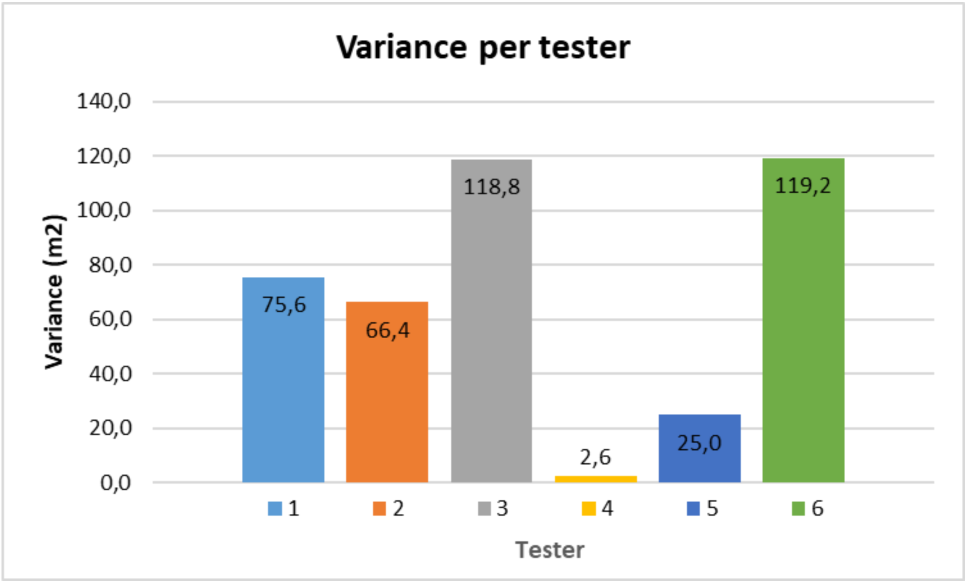
Looking at *Graphic 6*, all images seem to be equally discernable. They only differ in a few units and, if speed when reducing offset is considered, this fact can be attached to this reason. Even so, the second image could be a bit harder to identify because it differs in more than one unit. If, when having a more representative sample, all three images have the same average, a distance can be defined where images start being discernable.

Looking now at *Graphic 5*, the second image has a bigger variance compared to the other 2 and the first image has a very little variance compared to the other two images. This could be because the variance for the first image do not consider 6 values, it does consider only 5. If we had this value, the email variance would be greater attending to the results that the 4th tester has shown.

Anyway, it would only change a few the graphic and the second image is clearly the one with the most scattered values, followed by the third image and, next, by the first one. In my opinion, it is expected that the differences will be reduced with more testers, but it is a remarkable fact.



Graphic 7. Average per tester.



Graphic 8. Variance per tester.

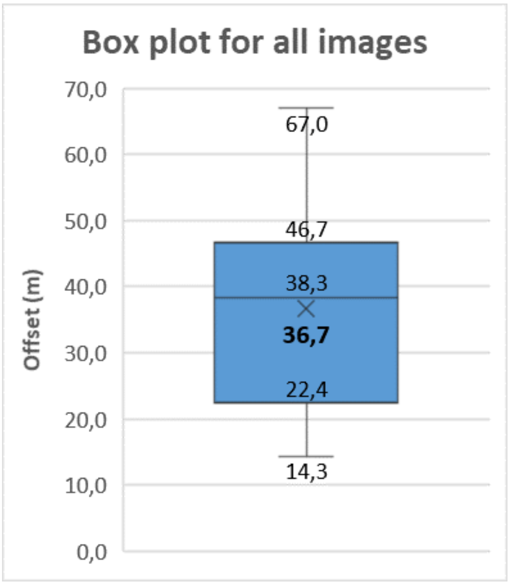
Looking at *Graphic 7* and *Graphic 8*, it is clear that each tester differ discerning the images. The 4th tester sees worse than the others, but has the less scattered values (it is remarkable that he/she only has two values). The 6th and 5th testers have the bigger offset average, but the 5th have the lower variance, so his/her values could be more accurated. This fact is the same fact as the "Readability Test". The 4th tester sees the worst and the 5th tester sees very well with the less scattered values. This can sustain that the 4th tester could need to calibrate his/her sight in order to see better. Furthermore, it also sustains the fact that the results are very similar, which supports the tests' results.

In conclusion, the test seems to be valid for being repeated in other VR devices because results seem to be good enough.

First, the averages are so close between them (as shown in *Graphic 7*), which could be because there is a distance where image starts to be discernable. Looking at the *Graphic 9*, values seem to be very scattered, so more samples are needed to determine wheter they have the same average or not.

Secondly, the variance is so difference between images (as shown in *Graphic 8*). This could be because of having only a few testers. Futhermore, that image could be confused with some others present on the table, so an improvement could be changing the test increasing the number of images, having some images similar to the other two. This way, this can be discarded if that is the case.

Finally, it would be good to lower the speed with which the image reduces its offset. The variances seem to be pretty high, so this could help.



Graphic 7. Box plot for all images

8.3. Precision Test

This test is aimed to compare how accurate are the measurements in the VR scene compared to reality. In this test, a person measures 1 meter in every of the three axis and another person measures his/her hand separation with an instrument (like a meter or a rope). The controller separation will be measured too in the VR to see the error.

In 8., the instructions of this test can be found. In them, there is set an specific point for the controllers to measure between.

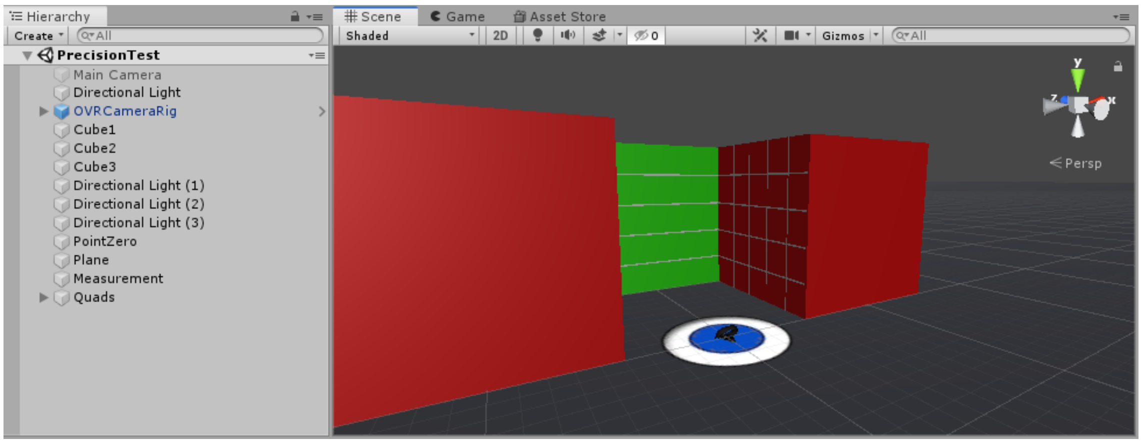


Illustration 45. Test scene with three cubes, (0, 0, 0) point and some given lines to pick and measure

In the previous illustration (*Illustration 45*), some lines are shown. The tested person has to select one line or axis (the cube axis) to measure, the most comfortable ones.

8.3.1. C# Code

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Measurement : MonoBehaviour
 {
    public GameObject PointMeasureR;
    public GameObject PointMeasureL;

    private float X = 0.0f;
    private float Y = 0.0f;
    private float Z = 0.0f;

    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
   {
        if (Input.GetKeyDown(KeyCode.Space))
        {
            Debug.Log("Overall distance is: " +
 Vector3.Distance(PointMeasureR.transform.position, PointMeasureL.transform.position));
        }
    }
 }
```

- **Line 26:** The points used to be measured (see Test Instructions on 8.) was set on an empty body and its coordinates were taken in order to measure the error committed measuring the VR 1 meter (positioning on the space). Vector3.Distance(Vector1, Vector2) returns the distance between two points.

8.3.2. Results and conclusions

Precision Test						
Tester 1	X axis		Y axis		Z axis	
Tries	Real	VR error	Real	VR error	Real	VR error
1st try	101,2	0,9979503	102,4	0,9943787	103,9	0,996963
2nd try	100,2	0,987415	101,4	1,000378	104,9	0,9958694

Table 4. Precision test values

	X axis	Y axis	Z axis
Average	100,7	101,9	104,4
Variance	0,5	0,5	0,5

Table 5. Averages and variances per axis.

The goal of this test was to measure 100 VR centimeters in reality. The same tester did the test two times positionating the controllers as stated in 8.3. because it wasted too much time and because this test does not depend on the tester, it does only depend on how accurate is the controller position when starting measuring (the more close is the VR error to 1, the best the result). Once done, the distance between controllers were measured in real life (in centimeters) and the error measuring in VR (positionating the controllers in the precise point to measure 1 VR meter).

Looking at *Table 4*, the VR errors are very small (1,000 means that there was no error). This value is only to see that the controllers were well positionated, because that does no influence too much the results, knowing that the real distance was measured using a cable and it is not very accurate. However, some things can be taken into account:

Firstly, the averages show that the X axis measures less than the Y axis, and the Y axis measures less than the Z axis. The 3D virtual space is perfect, so there is no reduction depending on the axis. Nevertheless, here something weird can be observed, The axis seem to be different when measuring in real world. Dued to measure with a cable, this could be only because of human error. In my opinion, more tests are needed in order to check this.

Secondly, this test does only depend on how accurate the controller position is when starting measuring (the more close is the VR error to 1, the best the result) and how accurate is the measurement in reality (a more accurate method to measure will be really helpful. It does not depend on how much tester there were. That is the reason why this was not tested in every person. Furthermore, this test can be done in every VR system with two tracked devices.

Finally, the variance (*Table 5*) is 0,5 in each axis because all values differ in 0,5 from the average and there are only two values (it supports the fact that it is needed a better measurement method in order to reduce the differences).

8.4. Field of View Test

This test is based on the tests explained in 7.5.1.. The main idea is to measure the field of view of the Oculus Rift S. The test uses four cubes and a sphere. The sphere is in front of the tested person, so that he/she points the vision on it using a reticle. Once done this, the cubes have to be moved by using the controllers in order to move them to the point where the user can see them.

First, two cubes will be situated horizontally, so the horizontal field of view will be measured first. Next, the vertical one will be measured. A white background will be used in order to keep the points more visible and in order not to distract the tested person.

Because of the two cubes may not appear at the same time in the field of view, two angles will be measured. It is important that the tested person stays as quiet as possible, in order to maintain the position and have the camera centered at every time using the reticle.

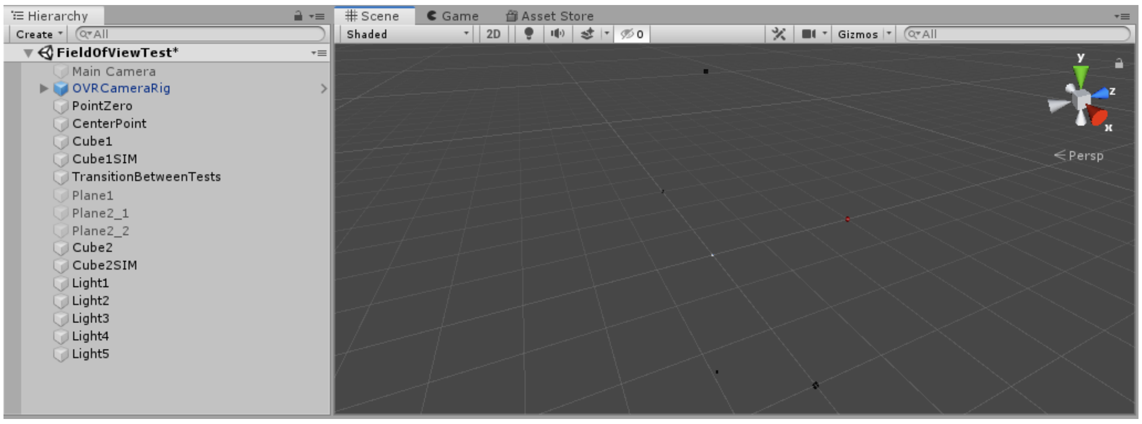


Illustration 46. Unity Test: The focus sphere (red point) and the cubes (4 black points) and the (0, 0, 0) point (white point)

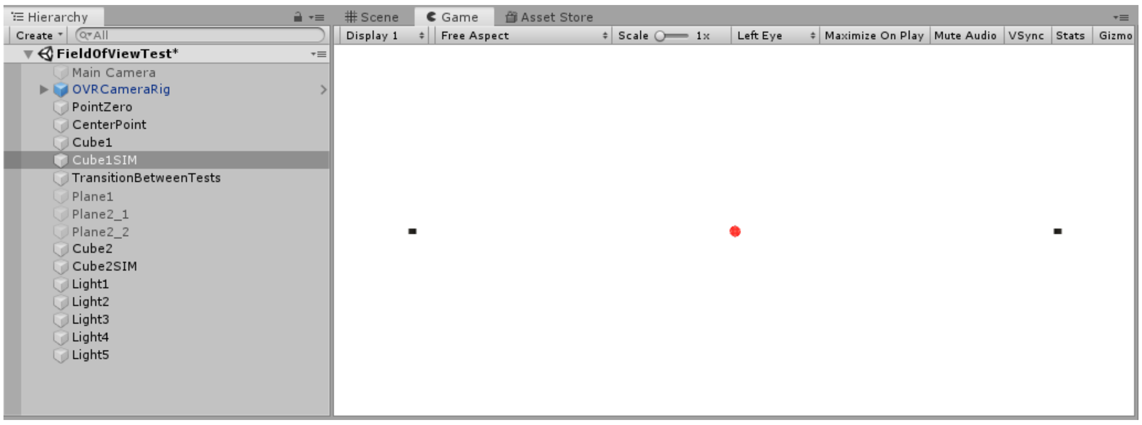


Illustration 47. Unity Test: Image with the white background. Red sphere (middle) and two black cubes in a certain angle (rotated from the original position)

The horizontal field of view will be measured and the cubes situated horizontally will rotate. Next, pressing the space button, the test will change to the vertical field of view measurement.

The cubes will rotate around the center point. For doing so, two of them will be programmed to rotate (one from the horizontal test and one from the vertical one) and the others will be moved symmetrically.

8.4.1. C# Code

The rotating code is the following, applied to two cubes (one from the horizontal test and one from the vertical one):

```csharp
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class VisualFieldController : MonoBehaviour
 {

    private float controller_variable;
    public float speed_multiplier = 0.05f;
    private float vision_angle;
    private int i = 0;
    private float a;
    private float b;

    // Start is called before the first frame update
    void Start()
    {
        Debug.Log("Press B when you start seeing the cube, and move the right joystick to rotate it around you");

        Debug.Log("Start rotating to the left");
    }

    // Update is called once per frame
    void Update()
    {
        if (i == 0)
        {
            // Message for seeing if controllers are active or not
            if (OVRInput.IsControllerConnected(OVRInput.Controller.RTouch))
            {
                Debug.Log("Right controller is active");
            }
            else
            {
                Debug.Log("Right controller is not active");
            }

            if (OVRInput.IsControllerConnected(OVRInput.Controller.LTouch))
            {
                Debug.Log("Left controller is active");
            }
            else
            {
                Debug.Log("Left controller is not active");
            }
        }
        i=1; //It will be only processed once

        controller_variable = speed_multiplier * OVRInput.Get(OVRInput.Axis2D.PrimaryThumbstick, OVRInput.Controller.RTouch).x;

        transform.RotateAround(Vector3.zero, Vector3.up, controller_variable * Time.deltaTime);

        if (OVRInput.Get(OVRInput.Button.Two, OVRInput.Controller.RTouch))
        {
            a = transform.position.y;
            b = transform.position.x;


            vision_angle = 90 - Mathf.Atan(a / b);
            Debug.Log("The vision angle is: " + vision_angle);

        }
    }
}
```

- **Lines 53-54:** Transform.RotateAround rotates around the first argument (a point), using the second argument (a vector) and at a certain speed (third argument). Time.deltaTime is a command that adapts the speed to the user because, using each frame to increase speed, would not be adapted to the reality.

The code attached to the symmetrical cube is the following:

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class VisualFieldControllerSIM : MonoBehaviour
6  {
7     public GameObject Cube_SIM;
8     // Start is called before the first frame update
9     void Start()
10    {
11
12    }
13
14    // Update is called once per frame
15    void Update()
16    {
17        transform.position = new Vector3(-Cube_SIM.transform.position.x,
18 Cube_SIM.transform.position.y, Cube_SIM.transform.position.z);
19        transform.rotation = new Quaternion(Cube_SIM.transform.rotation.x, -
20 Cube_SIM.transform.rotation.y, Cube_SIM.transform.rotation.z, 1);
21    }
21 }
```

- **Lines 17-20:** The symmetrical cube coordinates and rotation transformation.

8.4.2. Results and conclusions

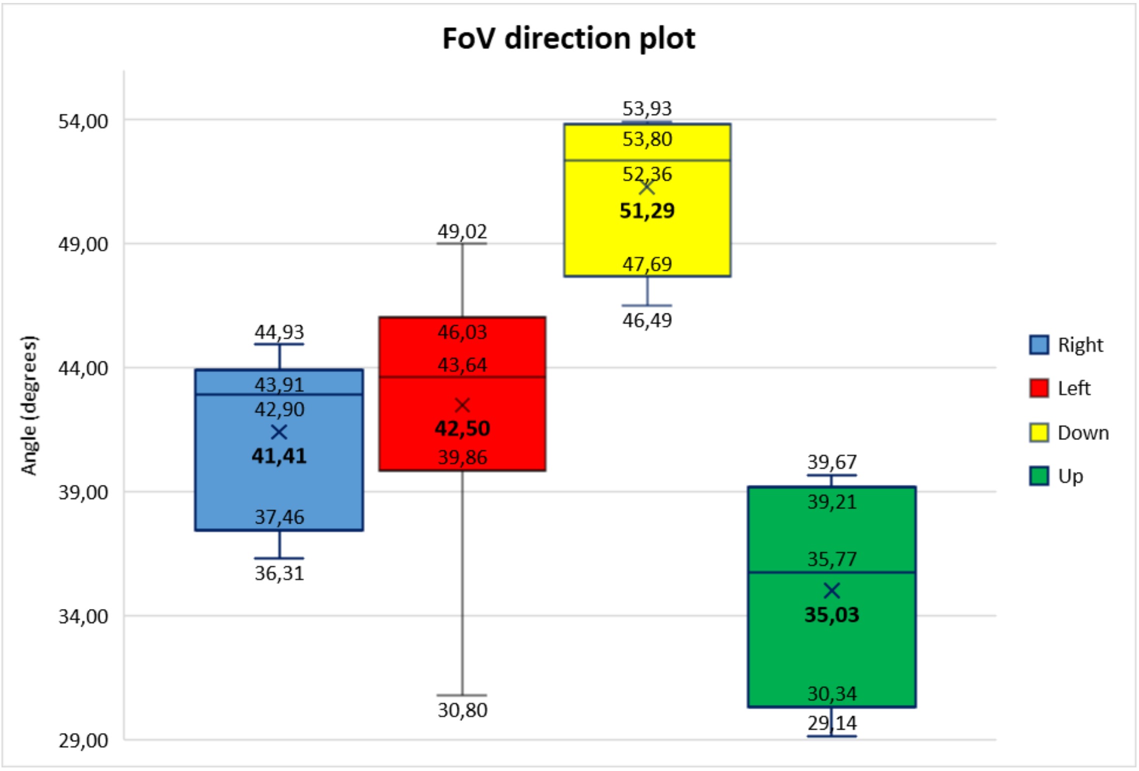
Field of View Test							
Testers	Tries →	1st try		2nd try		3rd try	
	Plane\Direction	→↓	←↑	→↓	←↑	→↓	←↑
1	Horizontal	37,85	49,02	38,06	49,41	37,05	49,00
	Vertical	51,31	36,64	47,37	37,97	51,90	37,79
2	Horizontal	42,99	45,03	42,90	44,69	43,71	44,95
	Vertical	53,42	39,06	51,88	38,73	52,33	40,24
3	Horizontal	42,82	42,88	43,45	43,33	42,45	44,19
	Vertical	53,93	34,89	53,22	33,53	52,57	35,13
4	Horizontal	43,57	43,57	43,57	42,20	43,93	42,24
	Vertical	46,49	39,67	42,52	39,38	48,06	39,77
5	Horizontal	44,93	43,72	44,94	43,98	44,49	44,49
	Vertical	52,11	29,14	50,34	29,08	51,16	31,53
6	Horizontal	36,31	30,80	41,55	41,55	42,33	42,33
	Vertical	43,28	30,75	48,55	31,92	48,12	31,34
	Plane/Direction	→↓	←↑	→↓	←↑	→↓	←↑

*Table 4. Results obtained. **Unit**: Degrees.*

Table 6 shows results obtained. <u>The angles are measured from the central point until the point where the user starts seeing the black cube</u>. For searching for a results it is only needed to search by tester. Then, the plane wanted (horizontal or vertical) and, finally, the direction of study (right or down for some columns and left and up for the others). The intersection between

"Horizontal" and "Right and Down" will give the "Right Direction" value. The intersection between "Vertical" and "Right and Down" will return the "Down Direction" value.

Example: Tester 1, value obtained on the 1st try. Upper angle → 36,64º.



Graphic 8. Field of View direction box plot

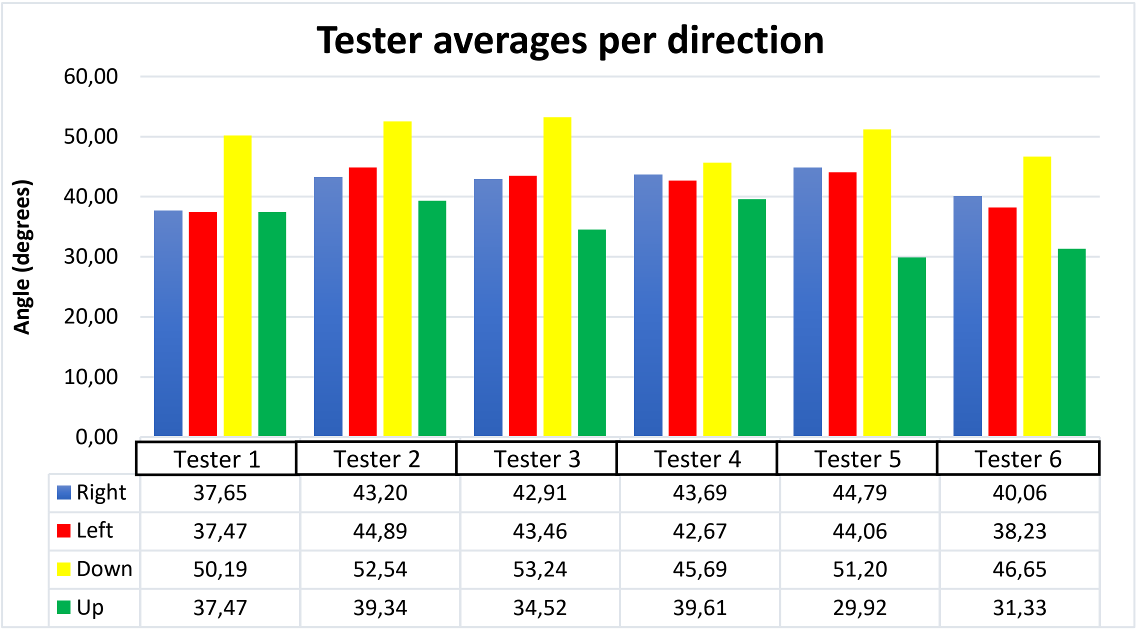
Graphic 8 show the results per direction. All of them have a small variance except red, but this is only an impression because the lowest value (30,80) is the unique below 39. Knowing this, it could be attached to a puntual case.

On one side, right field of view and left field of view have mostly the same average angle. The horizontal field of view is close to be symmetrical, so this makes sense. However, it seems that the left one has a bigger angle of view.

On the other side, the lower field of view is clearly bigger than the upper one. This fact agrees with the fact that the human upper field of view is smaller than the lower one, as stated on *page 40*.

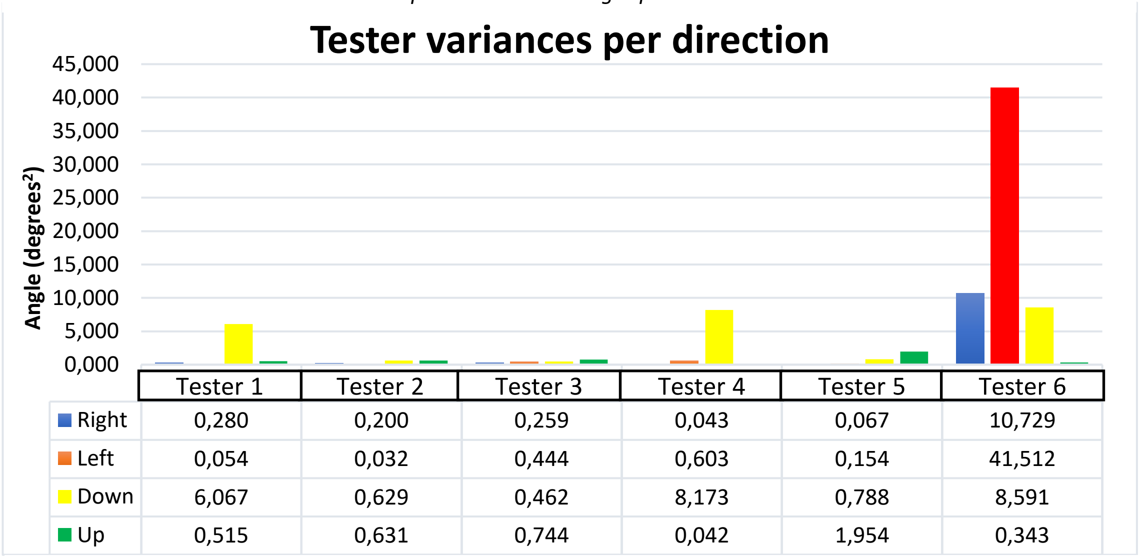
Analyzing now individually and pointing the focus in *Graphic 9* and *Graphic 10*, it is clearly visible that most of the averages are so similar between testers, only the 6th tester is differing a little. As shown in *Graphic 10*, some of the 6th tester variances are pretty high compared to the other ones, so the fact that the averages differ could be dued to this. Nevertheless, in my opinion, the 6th tester should have repeated the test in order to lower these high variances.

The other variances seem to be really good. With three tests per direction, having these low variances could mean that the test was very good designed in order to measure the field of view. Furthermore, the averages are pretty similar, so this could mean that this test could really help finding the VR systems fields of view. The human field of view too big compared to the results obtained with the VR system, so this test could have found the real VR system field of view because all people's field of view will be reduced to this point.



	Tester 1	Tester 2	Tester 3	Tester 4	Tester 5	Tester 6
🟦 Right	37,65	43,20	42,91	43,69	44,79	40,06
🟥 Left	37,47	44,89	43,46	42,67	44,06	38,23
🟨 Down	50,19	52,54	53,24	45,69	51,20	46,65
🟩 Up	37,47	39,34	34,52	39,61	29,92	31,33

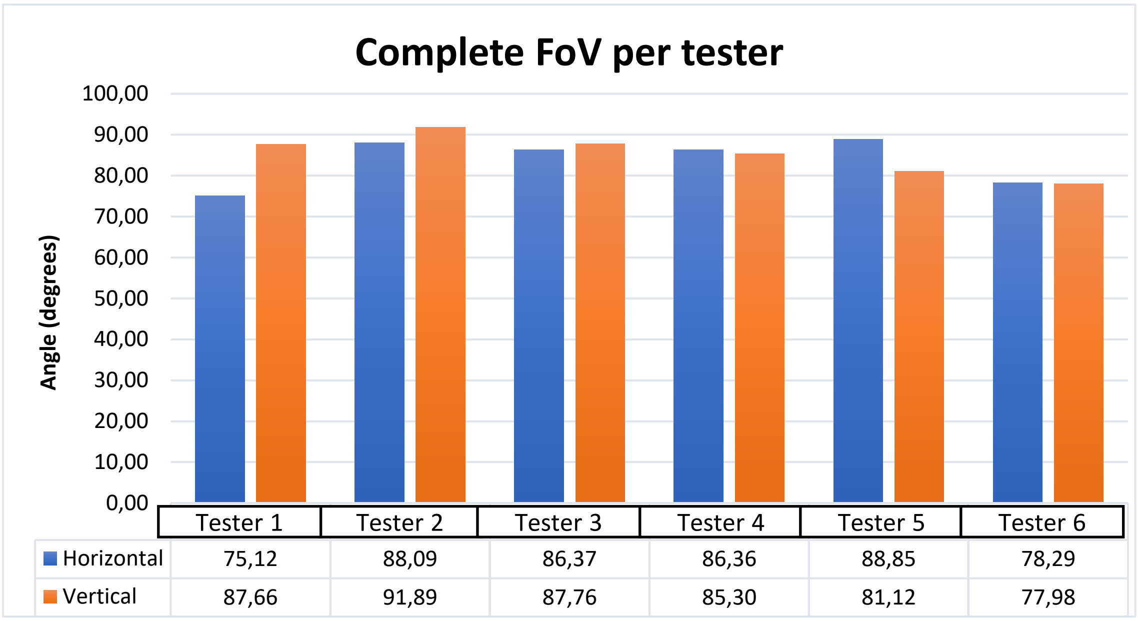
Graphic 9. Tester averages per direction



	Tester 1	Tester 2	Tester 3	Tester 4	Tester 5	Tester 6
🟦 Right	0,280	0,200	0,259	0,043	0,067	10,729
🟧 Left	0,054	0,032	0,444	0,603	0,154	41,512
🟨 Down	6,067	0,629	0,462	8,173	0,788	8,591
🟩 Up	0,515	0,631	0,744	0,042	1,954	0,343

Graphic 10. Tester variances per direction.

Looking now at the *Graphic 11*, the average field of views, the averages of the 6th tester are lower compared to the other averages. It is the same fact as stated before, probably dued to his/her high variances. However, all averages are pretty similar, so this intensifies the idea said above of a common field of view for all testers (the VR system field of view).



	Tester 1	Tester 2	Tester 3	Tester 4	Tester 5	Tester 6
🟦 Horizontal	75,12	88,09	86,37	86,36	88,85	78,29
🟧 Vertical	87,66	91,89	87,76	85,30	81,12	77,98

Graphic 11. Complete Field of View per tester.

Table 7 and *Table 8* show the total averages and the total variances:

	→	←	↓	↑
Total average	42,0506239	43,7441072	49,9198567	35,3647506
Total variance	7,6318592	16,0216320	11,4146395	15,4840671

Table 7. Total averages and variances per direction.

Average FoV	**Horizontal**	83,8478728
	Vertical	85,2846072

Table 8. Total Field of View average.

The variances are pretty low, as stated above. Right angle and Left angle are so similar, with the left one a bit bigger. Lower angle is bigger than Upper angle with a substantial difference. Finally, the complete horizontal and verical fields of view are really similar (the vertical is only two degrees bigger than the horizontal).

To sum up, there are some relevant aspects to consider:

Firstly, the test seems to be really good in order to measure a VR system field of view. It can find an angle per direction which does not really differ from one tester to another, so, in my opinion, this could mean that it can find the real VR system field of view.

Secondly, the number of tests could be really ready-witted. Results are pretty similar between testers and the test can find a similar angle for each tester. In my opinion, if comparing between VR systems is wanted, this number of sample could give significant differences between them.

Finally, the test is easily compatible with every VR system with a tracking system.

9. Conclusion

The goal of this thesis was to develop some scenes to test the selected performance aspects of VR in mechanical engineering and to evaluate the technical implementation and the user perception in a virtual reality scene using the “Oculus Rift S” VR system.

Furthermore, it was also necessary to write a little introduction an to select all possible aspects and how they were selected. Some tests were developed and tested by using a user study.

There were a lot of possible tests to develop, but only four of them were selected. However, all the rest tests were explained in order to help everyone to test them if wanted and to be used for a different VR system, and all performance aspects of VR in mechanical enginnering have a test to be compared between VR systems.

The four tests developed were tested on a six-person group. Their conclusion are below their explanations.

Every test was improved while being programmed compared to the first idea. Therefore, all tests stated above which were not developed will probably need a little improvement to adapt them for being tested. Nevertheless, the main ideas were written down.

Concerning the four tests programmed, everything seem to have gone well. Seeing their results, all of them can be used to compare results between different VR systems. Some need more samples to see if every impression the results seem to give is correct, but their results are very encouraging.

To sum up, all tests programmed offer hopeful results in order to be compared to other VR systems. Moreover, the other tests can be used if testing anything stated there is wanted, they were developed for being tested in every possible VR system. Finally, all the performance aspects of VR in mechanical engineering considered have a test to test them.

10. Bibliography

1. Abrash, M. (July 26th, 2013). *Down the VR rabbit hole: Fixing judder*. Retrieved from http://blogs.valvesoftware.com/abrash/down-the-vr-rabbit-hole-fixing-judder/
2. Arrioja Landa Cosio, N., (2010). *Guía Total del Programador*. Buenos Aires: Manual Users.
3. Ávila Veldes, N. (2003). *Interactividad y arte interactivo. La Realidad Virtual Inmersiva*. Arte, individuo y sociedad, 2003, 15 163-168 ISSN: 1131-5598.
4. BBC (n.d.). *Facebook invents new unit of time called a flick*. Retrieved from https://www.bbc.com/news/technology-42787529
5. Carrol, J. N. and Johnson, C. A. (August 22nd, 2013). *Visual Field Testing: From One Medical Student to Another*. Ophthalmology and Visual Sciences, University of Iowa Health Care. Retrieved from http://eyerounds.org/tutorials/VF-testing/
6. Cohen, M. A., Dennett, D. C. and Kanwisher, N. (May 1st, 2017). *What is the Bandwidth of Perceptual Experience?*. US National Library of Medicine National Institutes of Health. Retrieved from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4898652/
7. Colegio Oficial de Ópticos-Optometristas de Andalucía (n.d.) *Alteraciones del campo visual*. Retrieved from https://www.tuoptometrista.com/deteccion/alteraciones-del-campo-visual/
8. Computerhoy (n.d.). *¿Qué es latencia?* Retrieved from https://computerhoy.com/noticias/que-es-latencia-76407
9. Díaz Estrella, A. (2011). *Inmersión mental y Realidad Virtual*. Uciencia: Revista de divulgación científica de la Universidad de Málaga, ISSN-e 1889-7568, Nº. 6 (Abril), 2011 , págs. 30-33
10. Díaz Pier, M. (January, 2007). *Realidad Virtual basada en percepción* (Doctoral thesis). Computational Sciences PhD. Instituto Tecnológico y de Estudios Superiores de Monterrey. Campus Estado de México
11. Editeca (n.d.). *Realidad mixta – ¿Qué es y qué oportunidades nos ofrecerá?*. Retrieved from https://editeca.com/realidad-mixta/
12. Espinoza, M. et al (2012). *Descripción de un sistema de realidad virtual dirigido a la promoción del bienestar emocional en pacientes oncológicos hospitalizados: Ensayo Fase II OncoHelp*. REGIO. Revista Internacional de Grupos en Investigación en Oncología, Vol. 1. Num. 2., pag. 40-46. Retrieved from https://www.elsevier.es/es-revista-regio-revista-internacional-grupos-investigacion-339-articulo-descripcion-un-sistema-realidad-virtual-X2253645012578938
13. Facultat d'Informàtica de Barcelona (n.d.). *La Realidad Virtual*. Retrieved from https://www.fib.upc.edu/retro-informatica/avui/realitatvirtual.html

14. Farnsworth, B. (n.d.). *Measuring the Power of Virtual Reality Immersion [A Case Study].* Retrieved from https://imotions.com/blog/measuring-virtual-reality-immersion-case-study/
15. Ferrer, J. (2018). *HMD (head-mounted display).* Retrieved from http://multimedia.uoc.edu/blogs/rx/es/2018/03111/hmd-head-up-display/
16. Hektor Profe (2019). *Tutorial Unity 5.* Retrieved from https://www.youtube.com/channel/UCtjAOyZmqDXO-Oz87cZnWgw
17. Heroasset (n.d.). CAVE Virtual Environment. Retrieved from https://www.heroasset.com/vmweb/cave-cave-virtual-environment/
18. Intel (n.d.). *Virtual Reality Vs. Augmented Reality Vs. Mixed Reality.* Retrieved from https://www.intel.com/content/www/us/en/tech-tips-and-tricks/virtual-reality-vs-augmented-reality.html
19. Levis, D. (2006). *¿Qué es la realidad virtual?*
20. Marxent (n.d.). What is Virtual Reality? [Definition and Examples]. Retrieved from https://www.marxentlabs.com/what-is-virtual-reality/
21. Mundo Virtual (n.d.). *¿Qué es la realidad virtual?* Retrieved from http://mundo-virtual.com/que-es-la-realidad-virtual/
22. Oculus (2019). *Scripting Manual for Oculus.* Retrieved from https://developer.oculus.com/documentation/unity/unity-reference-scripting/
23. Oculus (n.d.). *Productos de Realidad Virtual.* Retrieved from https://www.oculus.com/?locale=es_ES
24. Otegi, J. (2017). *La Realidad Virtual y la Realidad Aumentada en el proceso de marketing* (Trabajo de Fin de Grado). Universidad del País Vasco/Euskal Herriko Unibertsitatea UPV/EHU, Guiuzkoa. *Revista de Dirección y Administración de Empresas.* Número 24, diciembre 2017, págs. 155-229.
25. Pérez Martínez, F. J. (2011). *Presente y Futuro de la Tecnología de la Realidad Virtual.* Creatividad y sociedad: revista de la Asociación para la Creatividad, ISSN 1578-214X, ISSN-e 1887-7370, Nº. 16, 2011, 39 págs.
26. Realidad Virtual GBI (n.d.). *Historia de la Realidad Virtual.* Retrieved from https://realidadvirtualgbi.wordpress.com/2012/10/12/historia-de-la-realidad-virtual/
27. Revistas de Robots (n.d.). Definición de qué es la Realidad Virtual. Retrieved from https://revistaderobots.com/rv/definicion-de-que-es-la-realidad-virtual/
28. Rogers, S. (2019). 2019: The Year Virtual Reality Gets Real. *Forbes Magazine.* Retrieved from https://www.forbes.com/sites/solrogers/2019/06/21/2019-the-year-virtual-reality-gets-real/#41c15c86ba99
29. Salinas, J. (2018). ¿Qué es el Flick? Retrieved from https://javiersalinas.es/que-es-el-flick/

30. Shaun W. Jerdan et al. (July 6th, 2018). *Head-Mounted Virtual Reality and Mental Health: Critical Review of Current Research*. US National Library of Medicine / National Intitutes of Health. Retrieved from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6054705/
31. Spielberg, S. (2018). *Ready Player One*. USA Film.
32. Steven Kim, MD. (January 6th, 2016). *Visual Field Exam*. Retrieved from https://www.healthline.com/health/visual-field
33. The Franklin Institute (n.d.). *History of Virtual Reality*. Retrieved from https://www.fi.edu/virtual-reality/history-of-virtual-reality
34. Thinkmobiles (n.d.). *What is Augmented Reality (AR) and How does it work*. Retrieved from https://thinkmobiles.com/blog/what-is-augmented-reality/
35. Trachtenberg, D. (2016). *Playtest*. Black Mirror. 3rd season, episode 2.
36. Troncy, R., Carrive, J., Lalande, S., Poli, J.-P. (October, 2004) *A motivating scenario for designing an extensible audio-visual description language* (Paper). Sorbonne University and the French National Institute of Audiovisual.
37. TuDosisDigital (n.d.). *Realidad Virtual, ¿qué es y qué tipos existen?*. Retrieved from https://www.tudosisdigital.com/noticias/realidad-virtual-que-es-y-que-tipos-existen/
38. Unity (2019). *Manual: Scripting*. Retrieved from https://docs.unity3d.com/Manual/ScriptingSection.html
39. US National Library of Medicine / National Intitutes of Health (n.d.). *Campo visual*. Retrieved from https://medlineplus.gov/spanish/ency/article/003879.htm
40. VirtualSpeech (n.d.). *History of VR - Timeline of Events and Tech Development*. Retrieved from https://virtualspeech.com/blog/history-of-vr
41. Visbox (n.d.). *CAVE Automatic Virtual Environment*. Retrieved from http://www.visbox.com/products/cave/
42. Wikiuniversity (n.d.). *Realidad virtual*. Retrieved from https://es.wikipedia.org/wiki/Realidad_virtual
43. Wikiversus (January 11th, 2019*). ¿Qué es el judder? ¿Y qué tiene que ver la industria del cine?* Retrieved from https://www.wikiversus.com/tv/judder/
44. Xinreality (n.d.). *Judder*. Retrieved from https://xinreality.com/wiki/Judder

11. Appendix

11.1. Illustration Index

11.2. Table Index

11.3. Graphic Index

Erklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Werken wörtlich oder sinngemäß übernommenen Gedanken sind unter Angabe der Quellen gekennzeichnet.

Ich versichere, dass ich bisher keine Prüfungsarbeit mit gleichem oder ähnlichen Thema bei einer Prüfungsbehörde oder anderen Hochschule vorgelegt habe.

....

Ort, Datum   Unterschrift