FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

AR Interaction and Collaboration in Interchangeable Reality

Diogo Serra Duque

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Rui Nóbrega

Co-Supervisor: Teresa Matos

Co-Supervisor: João Jacob

July 23, 2019

AR Interaction and Collaboration in Interchangeable Reality

Diogo Serra Duque

Mestrado Integrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

July 23, 2019

Abstract

Augmented Reality is a technology which enables a user to view virtual content overlaid to its own vision of the world. Such a technology paves the way for a new type of interfaces, allowing to take advantage of real world features and create interactions where real and virtual blend in a natural way. Such a natural interface may even allow natural interactions from daily life to be used in this mixed reality.

It brings many new possibilities for many application fields, one of them being collaboration. AR can be used to augment a physical workspace so as to provide better access to information and its visualization, efficiently improving task execution.

AR is also part of a much broader Virtuality-Continuum, which defines a spectrum of environment immersion, ranging from reality "as is" to fully immersive environments like VR, where each different immersion level brings different advantages. However, not many projects take advantage of multiple levels of the Virtuality-Continuum at the same time. This dissertation aims to explore the use of different places from the Virtuality-Continuum, namely AR and VR, in a collaborative context, thus enabling both environments to coexist simultaneously in an Interchangeable Reality.

The proposed solution should therefore consist of a framework prepared to handle multiple AR-capable devices simultaneously in a collaborative context, while also enabling the possibility of integration with VR-capable devices. Every user using the system should have access to the same shared space, and interactions from a user should be synchronized with every collaborator with the help of a server.

With this framework, a proof of concept has been developed for performing tests with real users. The retrieved information from the user tests helps understand, given this Interchangeable Reality context, the impact of AR in collaboration and to what extent does AR makes interactions easier.

Keywords **- Augmented Reality, Collaboration, Interaction Interfaces**

Resumo

A Realidade Aumentada é uma tecnologia que permite ao utilizador visualizar conteúdo virtual sobreposto à sua própria visão de mundo. Esta tecnologia abre as portas para um novo tipo de interfaces, permitindo tirar partido de características do mundo real para criar interações onde o real e o virtual se misturam de forma natural. Uma interface tão natural pode ainda permitir que interações igualmente naturais, típicas do dia-a-dia, sejam utilizadas nesta realidade mista.

Esta tecnologia traz várias novas possibilidades com aplicação em várias áreas, sendo um deles a colaboração. A Realidade Aumentada pode ser usado para "aumentar" um espaço de trabalho físico de modo a fornecer melhor acesso a informações e à sua visualização, melhorando de forma eficiente a execução de tarefas.

A Realidade Aumentada também faz parte de um *Virtuality-Continuum* muito mais amplo, que define um espectro de imersão no ambiente, variando da realidade "tal como é" a ambientes totalmente imersivos como Realidade Virtual, onde cada nível de imersão diferente traz vantagens diferentes. No entanto, poucos projetos tiram proveito dos vários níveis presentes no Virtuality-Continuum ao mesmo tempo. Esta dissertação visa explorar o uso de diferentes ambientes, nomeadamente AR e VR, num contexto colaborativo, permitindo assim que ambos os ambientes coexistam simultaneamente numa *Interchangeable Reality*.

A solução proposta deve, portanto, consistir numa framework preparada para lidar simultaneamente com vários dispositivos AR num contexto colaborativo, permitindo ainda a possibilidade de intergração com dispositivos VR. Todos os utilizadores devem ter acesso ao mesmo espaço partilhado e as interações de um utilizador devem ser sincronizadas com os outros colaboradores com a ajuda de um servidor.

Com esta framework, uma prova de conceito será desenvolvida para testar posteriormente com utilizadores reais. As informações obtidas destes testes devem ajudar a entender, neste contexto de *Interchangeable Reality*, o impacto da Realidade Aumentada na colaboração e em que medida esta facilita as interações.

Keywords **- Realidade Aumentada, Colaboração, Interfaces de Interação**

Acknowledgements

To my supervisors, Rui Nóbrega, Teresa Matos and João Jacob, thank you for all the guidance, patience and criticism.

To my girlfriend, Sofia, thank you for being the ultimate "rubber ducky" real-life debugger and for always telling me to go work.

To all the friends that accompanied me during this journey, specially those that took the time to participate in the user tests, a big thank you, as I would not be here if it were not for you.

To my family, thank you for always hearing my never-ceasing rambling and for the constant encouragement.

Diogo Serra Duque

Contents

List of Figures

LIST OF FIGURES

List of Tables

Abbreviations

AR	Augmented Reality
AV	Augmented Virtuality
CSCW	Computer-Supported Cooperative Work
HMD	Head-Mounted Display
MAR	Mobile Augmented Reality
MR	Mixed Reality
SDK	Software Development Kit
UWP	Universal Windows Platform
UX	User Experience
VR	Virtual Reality

Chapter 1

Introduction

Augmented Reality (AR), a technology which enables a user to see virtual objects overlaid to its own view of the world, has become very popular in the recent years. Its rising popularity among the general public comes essentially from smartphone applications. With applications such as Pokémon Go [1] and IKEA Place [2] featuring AR content and labelling it as AR, this technology has become much more familiar to the world.

However, the general public is not the only interested in such technology. In fact, AR also has a great synergy with less visible application fields, such as maintenance support [ZCGTS+17] and collaboration [BK02]. In a world where teamwork is getting more and more important, tools that support this type of work are of high value. But even teamwork is not always at its best. Productivity tools are increasingly fashionable, and traditional strategies for collaborative work are no longer enough by today's standards, when compared to teams using these modern tools.

One way to enhance collaboration is by using AR as a means to share information with other collaborators. AR can make this information both accessible and integrated in the users' view of the world. Such a technology expands the limits of each user's perceptions by creating a collaborative environment where users can see each other's position, as well as the results of their interactions. But this concept can be taken yet a step further, by adding Virtual Reality to the equation. Even though both technologies are different in its essence, both support a virtual world, despite doing this to different levels. Since both support a virtual world, it can be beneficial for collaboration purposes to use the advantages of the two in a single collaborative environment. The usage of the same unified environment in AR, VR and in the real world is what will be called Interchangeable Reality throughout this thesis.

1.1 Context and Motivation

Computer-Supported Cooperative Work (CSCW) is a research area that attempts to better understand the role and impact of technology in the work environment [Gru94], where collaborative

[1] Pokémon GO, https://pokemongolive.com/en
[2] IKEA Place, https://highlights.ikea.com/2017/ikea-place

software seeks to boost the efficiency of collaborating users. One of CSCW's areas that seek to provide more intuitive and broad ways to see the world is Augmented Reality, which brings many new ways of interacting with the said software. These interactions are of special utility in collaborative environments, where metrics like productivity are taken into high account. AR collaboration in face-to-face environments provide access to a series of cues related to the surrounding environment and non-verbal cues of other collaborators, allowing more natural interactions. But whenever the collaboration is remote, everything becomes more complicated, as these cues are not visible anymore. Even though remote AR collaboration is an active research subject, it still does not provide many easy answers.

However, there are other research topics that do not get so much focus, thus also not providing many solutions, one of them being the interactions between AR and VR. It is from the gap of knowledge from these two subjects that this thesis appears, as part of a much larger project: PAINTER[3] (Procedurally Assisted INTErchangeable Reality). PAINTER is a project that consists on a framework with many interdisciplinary goals. It encompasses: (1) the creation of content using Virtual Reality (VR); (2) procedural environment generation; (3) AR and VR collaborative interaction with an application developed with the framework.

This thesis is part of the larger scope of PAINTER, taking only into attention the third goal, which relates to AR and VR collaborative interaction. As such, this work pretends to explore this combination of AR and VR, with a heavy focus on AR collaboration mechanisms.

1.2 Research Questions

In the interest of focusing the goals of this research and later development, two research questions have been introduced. An explanation is also associated to each one of them, so as to better clarify the underlying goals. These questions are the following:

- **Q1. What is the impact of using AR on collaborative teamwork, given an Interchangeable Reality context?**
  AR has been proved to enhance task performance on its own, either in individual or cooperative terms. The goal here is to understand if this ground still holds in a collaborative context of Interchangeable Reality, where AR and VR devices may be present simultaneously.

- **Q2. To what extent does AR make the interactions easier in an Interchangeable Reality context?**
  This question intends to further understand the implications of the previous one. AR should make tasks easier, and the objective is to understand in what ways it makes the tasks at hand easier, given this context of Interchangeable Reality.

[3]PAINTER, https://dei.fe.up.pt/gig/projects/painter/

1.3 Objectives

In this section, the thesis' objectives are defined as complement of the research questions:

1. Ascertain the usability degree of AR, given an Interchangeable Reality context.
2. Analyze the distinction between real interaction mechanics and AR-powered interaction mechanics in an Interchangeable Reality context.
3. Understand which AR interaction mechanics are more intuitive for users to use, in an Interchangeable Reality context.

The first objective relates solely to the first research question (Q1), as a usability study is necessary to better understand if this thesis has any impact on collaborative teamwork. Regarding Q2, both the second and third goal relate to it. The second one seeks to verify whether AR-related mechanics actually enhance collaboration when compared to those unrelated to AR, while the last objective aims at understanding which mechanics makes the interactions easier and which do not.

1.4 Implemented Solution

Given the defined goals, an environment to achieve them and find answers to the research questions must be created. With this in mind, a solution is proposed as an environment for exploration, where the research questions shall be put to the test.

The implemented solution consists of an AR component for an Interchangeable Reality framework, with a heavy focus on AR interactions. This solution contains many manipulation options regarding virtual objects such as translation, rotation and scale. It also provides heavy support for communication, mostly under the form of non-verbal cues like arrows, which can be spawned by users. Other cues are also present in this tool, though with the goal to guide the user, such as each selected object having an overlay around it signaling selection and messages warning the user about what to do when the phone looses track of the environment. AR markers are used to link a real world position to a virtual world one, allowing users to navigate the virtual environment.

A Proof of Concept (PoC) was also developed, as a way to put the objectives and research questions to the test. This PoC was designed as a three-room residence with several objects to tidy up. These objects were mostly placed around the AR markers' virtual location to facilitate user interaction without the need for much movement, with the goal of making it easy to experience every aspect of the platform in the user tests.

1.5 Contributions

This thesis' contributions start with the research in topics such as Collaboration, Augmented Reality and Remote and Face-to-Face Collaborative Augmented Reality, providing a comprehensive overview of these topics. This research work also paved the way for the implemented solution described in the previous section, consisting of a set of mechanics and tools useful for implementing AR interactions.

However, the foremost feature of the implemented solution is also a highly relevant contribution: integration with a Mixed Reality environment. Such an environment allows VR users to share the same virtual space as AR users, giving each of the user types space for their own mechanics for interaction, while also enabling them to support common features.

The results obtained are also a valuable contribution, providing some insight into the users' perceptions when using AR and how they collaborate in a Mixed Reality context.

It should also be noted that this thesis' implemented solution fits as part of the PAINTER project, thus also counting as a contribution to the project's goals on a broader scope.

1.6 Document Structure

This introduction serves as a starting point for this thesis, briefly explaining its context and the problem it attempts to tackle, as well as how it can contribute to the scientific community. Following this chapter, five more chapters exist.

In Chapter 2, the State of The Art related to Collaborative Interface, Augmented Reality and Collaborative Augmented Reality is described and some related works are presented. This includes types of collaborative interfaces, tracking and other AR-related topics, as well as an analysis on remote and face-to-face AR collaboration.

Afterwards, Chapter 3 proposes an AR component for an Interchangeable Reality framework, including its architecture and the system requirements properly divided according to different goals of the tool.

Chapter 4 seeks to show exactly what was developed as a solution for the problem presented. Implementation details are presented, along with decisions made during development which steered the development itself.

The next chapter (Chapter 5) puts the developed solution to the test with the goal of answering the research questions. User tests results are thus analyzed in order to try and answer these questions.

Finally, in Chapter 6, some general conclusions are drawn from the results discussion. Future developments for this tool and other research opportunities are also suggested at the end of this document.

Chapter 2

State of The Art Review

In this chapter, the state of the art for the topics surrounding this dissertation are presented in order to provide an up-to-date vision of what are its most common problems, their solutions and also the latest research topics. This starts with an overview of collaborative interfaces for a wider context on Computer Supported Cooperative Work (CSCW), soon followed by AR-specific topics, including up-to-date tracking methods, as well as many interaction approaches, such as Mobile AR, Tangible AR and Interchangeable Reality. Subsequently, Collaborative Augmented Reality is also reviewed, so as to better understand the latest research topics, as well as some common problems and their solutions, similarly to the Augmented Reality review. This section will have its focus turned to the collaborative aspect of Augmented Reality, specially on Face-to-Face and Remote Collaboration. Last but not least, an overview of the most used technologies related to Augmented Reality Collaborative AR is also presented, showing either software and hardware commonly used for these tasks.

2.1 Collaborative Interfaces

CSCW is a popular research field among others in Human-Computer Interaction. It focus highly on the interactions of several users with a computer, rather than a single one. This adds another myriad of challenges, some of them closely related to the interface through which the users are allowed to interact with the computer. This includes issues such as workspace awareness, which relates to the preservation of a mutual and present understanding of every other collaborating user's actions in the workspace, or the separation between collaborative and individual tasks, which consists on separating tasks that should be done by a single user or a restricted group from tasks where everyone should be able to cooperate, possibly even preventing users from doing tasks not associated with their roles [TPP18]. Solving these problems can lead to an improvement in the collaborative work's efficiency.

However, solving the general problems of such interfaces is not always easy, and different interfaces may prove to be more useful for different tasks. As such, in this section several interfaces are presented. Some of these interfaces may also be used in combination with each other.

The most traditional and also by far the most common interface with the computer is the keyboard-mouse duo accompanied by a screen, with the keyboard and mouse serving as input and the screen as output. Most computer applications are made with this interface in mind, with a very diverse range of goals. In games such as League of Legends [1] and Dota 2 [2], the mouse is mostly used as a movement and attack tool, with the keyboard being left with more specific actions. In the Counter Strike [3] game series, the mouse is used for weapon actions (like shooting or see through the weapon's scope), while the keyboard deals with movement and more specific actions. When shifting to the productivity and work subjects, we can see Google Docs' [4] use is very similar to that of non-collaborative text processors, with the keyboard serving for writing and shortcuts, while the mouse is used mostly for selection and menu navigation. A common characteristic among all the previous cases (and most others) is that, should the context be non-collaborative, the interface would be exactly the same. On one hand, this can be a benefit since the user's familiarity with the application in an individual context is taken advantage of when in a collaborative context. On the other hand, this may prevent more collaboration-related actions.

2.1.1 Distributed User Interface

Also known as cross-device interfaces, these interfaces are shared across many, possibly different, devices. There are already commercial solutions that are capable of mirroring, coordinating and stitching multiple displays, but research highlights several challenges when adapting these interfaces for real-time collaboration (*e.g.* functional UI coordination) [PGR$^+$18]. However, as would be expected, these challenges are already being tackled. This work further adds that research already allows collaboration around a single digital whiteboard with the help of mobile devices.

These authors address the assignment of UI elements to devices in a context where different users may have personal devices, like smartphones, but also shared devices across everyone or just some users. It does so by allowing users to set a priority indicator for each of the UI elements. In a personal device (belonging to just one user), the device will reflect the top elements, according to the user's priority indications, that fit the screen. When the device is shared, the interface will show UI elements according to each user's priority indicators, trying to show the relevant information for both, assuming that some elements may remain "hidden" in the personal device, as well that some personal elements may also be showed in the shared display, all according to the priorities indicated.

[1]League of Legends, https://leagueoflegends.com
[2]Dota 2, http://www.dota2.com
[3]Counter Strike, https://counter-strike.net
[4]Google Docs, https://www.google.com/docs/about

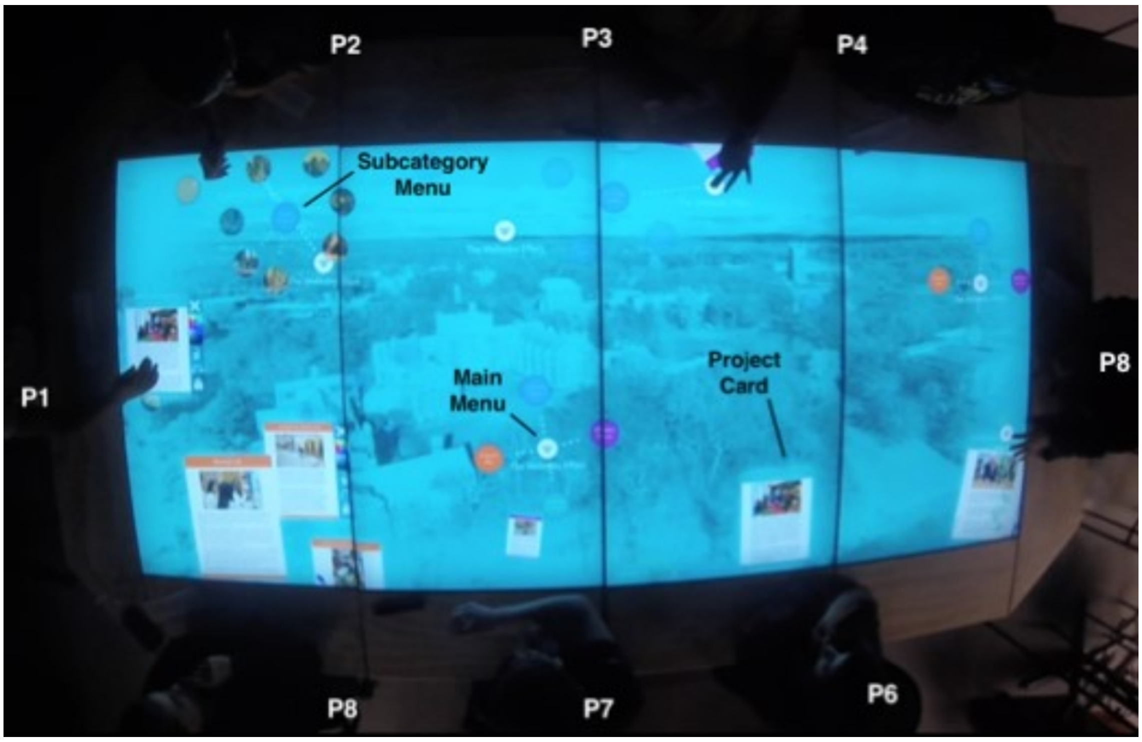


Figure 2.1: Tabletop interface proposed in [WSV+17].

2.1.2 Tabletop Interface

Interactive surfaces are gaining traction and becoming increasingly available [WSV+17], and as such have been studied extensively. Suited for a multi-user environment and also used in multi-device environments, it is a very flexible interface, where interactions are usually made by touch, as these interfaces typically have a touchscreen.

An example of a tabletop interface was developed with the goal of aiding funding decisions for projects [WSV+17]. Users can use menus to browse for projects and open several of them in the form of cards, as can be seen in Figure 2.1. These cards can be scaled, moved or reoriented with the help of standard multi-touch gestures. Cards can also be annotated with infrared pens, and when closed, the annotations are shared with all the other users. Drawings and annotations can also be made in the general workspace, shared with every user.

2.1.3 Augmented Reality Interface

Augmented Reality is usually enabled through a Head-Mounted Display (HMD) or a smartphone, though not necessarily. Capable of integration with hand tracking [HKBA19] and gaze tracking [PDE+17], Augmented Reality provides a very diversified interface with a great potential for many natural interactions, besides its already immense capabilities for mixing virtual objects with the environment for a more believable interaction.

Even though most setups need a device for each user, this is not mandatory. HoloDesk [HKI+12] is such an example, where interactions are all made inside its physical structure, with users looking through a mirror that shows the augmented content. HoloDesk is a very specific AR interface, as it only works in a constrained space, but one that allows the users to use their own hands as inputs with the absence of HMDs, enabling a more natural experience, similar to that of the real world.

2.1.4 Virtual Reality Interface

Virtual Reality allows full immersion in a digital world, usually enabled by an HMD, with interactions being only possible in that very same world. Some common problems include limited Field-of-View (imposed by the hardware), lack of haptic feedback and network delays [FGV+00], as well as VR sickness [FF16]. In a collaborative context, solving these problems is very important to achieve a more intuitive interface to use. But despite these challenges, Virtual Reality still has a great potential since the interface is the world itself, providing an endless world of possibilities.

VR has one particular advantage over other interfaces: immersion. This particular characteristic can abstract users from the distractions of the outside world, increasing their focus on the tasks at hand in the virtual world. Some works [PHPK19] study interaction techniques in collaborative environments, where results show that, when accompanied, users feel higher levels of enjoyment while also feeling more present in the virtual world. But even with this unique advantage, some tasks are still more efficient when done with a desktop [SSDP+08].

2.2 Augmented Reality

Augmented Reality (AR) is commonly defined as a technology that adds digital information over the user's view of the world, almost as if the digital information belongs there [SvdH16, NTLY17]. This digital information can range from simple annotations and informations to fully computer-generated objects overlayed as if they were really part of the real world. And even though there are so many ways of receiving new information with this technology, it is not purely visual, providing a high degree of interaction as well and making AR a two-way connection between users and the environment.

Besides enabling an augmented vision of one's surroundings, in a world where our traditional inputs and outputs are falling shorter of our expectations, AR also shows itself as a multimodal interface with great potential for exploration [SvdH16].

This approach on interaction with the surrounding world gives rise to the user of AR in various other fields. Applied in industrial maintenance, it helps workers by showing them virtual red lines that illustrate the positions of many types of infrastructures which are usually hidden from view, such as telecommunication or electricity lines [KBB+18]. Medical doctors can also benefit from the overlay of x-ray images onto their patients, allowing a better correlation between the image and reality [BSEN12]. In the case of mechanics, these type of tools allow faster identification

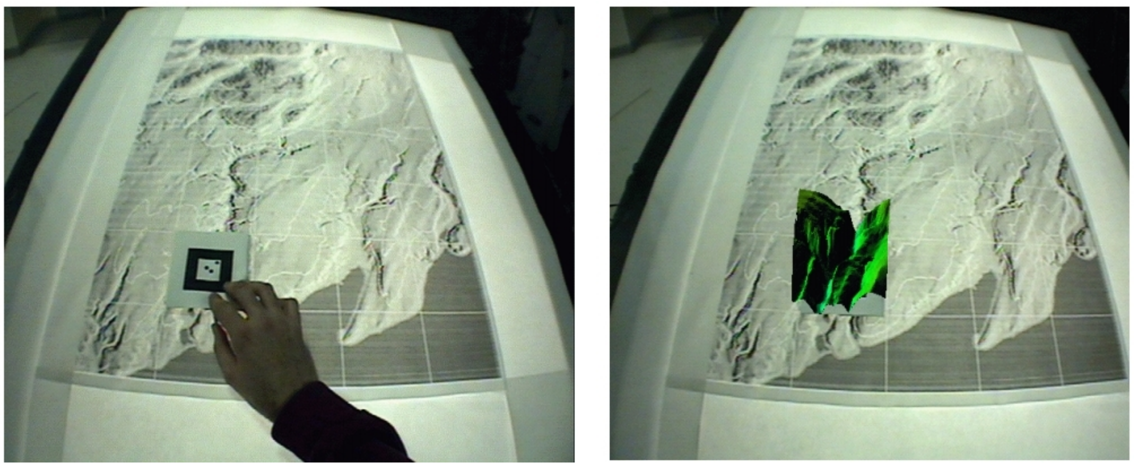


Figure 2.2: Left: map with a marker; right: map with an augmented marker [BKP02].

of tasks that need to be done and their order, resulting in improvements over task performance [KBB$^+$18].

2.2.1 Tracking

This subsection introduces the current options for tracking the environment and position of virtual and physical objects. These techniques were split into two groups: marker-based and markerless. Marker-based comprise all approaches that take advantage of fiducial markers, and markerless presents tracking alternatives to the use of fiducial markers.

2.2.1.1 Marker-based Tracking

A very common form of tracking is carried out using fiducial markers. These markers, as can be seen in Figure 2.2 are most often matrix-like black and white patterns [KAS10], also commonly referred to as QR Codes, with features easy to spot through Computer Vision algorithms like SIFT [Low04]. They can also appear under the form of template images or even circular markers, a round variation of the original ones. Systems using this technique are easier to implement than markerless ones and also use less resources [NTLY17].

One of the problems with these markers is that they are visually obtrusive [KAS10], making them an unwanted view in the scene. Another problem is that they also oblige a user to keep the markers in its field of view if he wants to view/interact with the virtual object associated with the marker. Several approaches attempted to solve or mitigate these drawbacks. One approach is the use of multimarkers, where the AR system knows the relative position of several markers to each other, and so even if it only sees one marker, it is able to know the positions of the others, even without never seeing them, slightly relieving the user's field of view and facilitating the occlusion management of other markers [ZCGTS$^+$17]. It also enables a greater robustness regarding illumination. As computers are very sensitive to illumination changes, if using only one marker, it may cause some flickering on the virtual objects. However, if using multiple markers, this flickering is reduced.

Another experiment optically hides a QR code inside a picture [NTLY17]. By turning these pictures into sets of tiles, it is possible to later encode it with a QR code that will remain hidden to everyone except a knowing algorithm. By changing the gap sizes between individual tiles, it is possible to hide such information in plain sight. However, even though this approach brings the best aspects of using an image and a marker together, the results showed that the decryption of this encoding does not always works correctly, specially at a distance higher than 1.5m.

Yet another endeavour consisted on trying to hide the marker altogether from the view presented to the user [KAS10]. This is achieved by determining the geometry of the marker and then computing a high resolution hiding texture for it with the help of image inpainting methods. The texture is later updated for every frame using a low resolution texture, accounting for lighting changes and adapting the original hiding texture, ensuring the hiding texture does not draw the attention of users.

2.2.1.2 Markerless Tracking

Tracking can also be made without any markers at all. Instead, there is a more exhaustive effort to map the environment itself and use it as a reference for augmented objects.

A common choice for the main algorithm is SLAM [DB06], an algorithm originally from the field of robotics that creates and updates a map of the surrounding physical space. It became so widely used that it gave birth to many variations and improvements of itself over the last years, as well as being used in many tools and devices such as ARCore, ARKit, Wikitude and HoloLens [KBB$^+$18].

A more recent approach is the usage of RGB-D cameras. These cameras are similar to ordinary ones regarding the capture of a RGB image, but they add an extra 'D' for detecting depth. This type of camera started to become famous due to Microsoft's Kinect [5]. In fact, according to [KBB$^+$18], the most highly cited paper between 2008 and 2017 was one about a system, KinectFusion, which could accurately map and track, in real-time, complex indoor spaces with a low-cost RGB-D camera, like the Kinect. This work further increased the popularity of RGB-D, as KinectFusion was also robust enough to work with different lighting conditions, paving the way for further development with this technology.

Hybrid tracking is also a promising alternative, given its ability to combine multiple sources from different sensors, in order to improve tracking quality, *e.g.* images and depth from RGB-D camera and sensor inputs from Inertial Measurement Unit (IMU) and Global Positioning System (GPS) for enhanced tracking. This is especially useful for mobile platforms, where several sensors are already embedded. Besides, this approach can also be combined with software algorithms like SLAM, providing access to yet another class of benefits.

[5]Kinect, https://developer.microsoft.com/en-us/windows/kinect

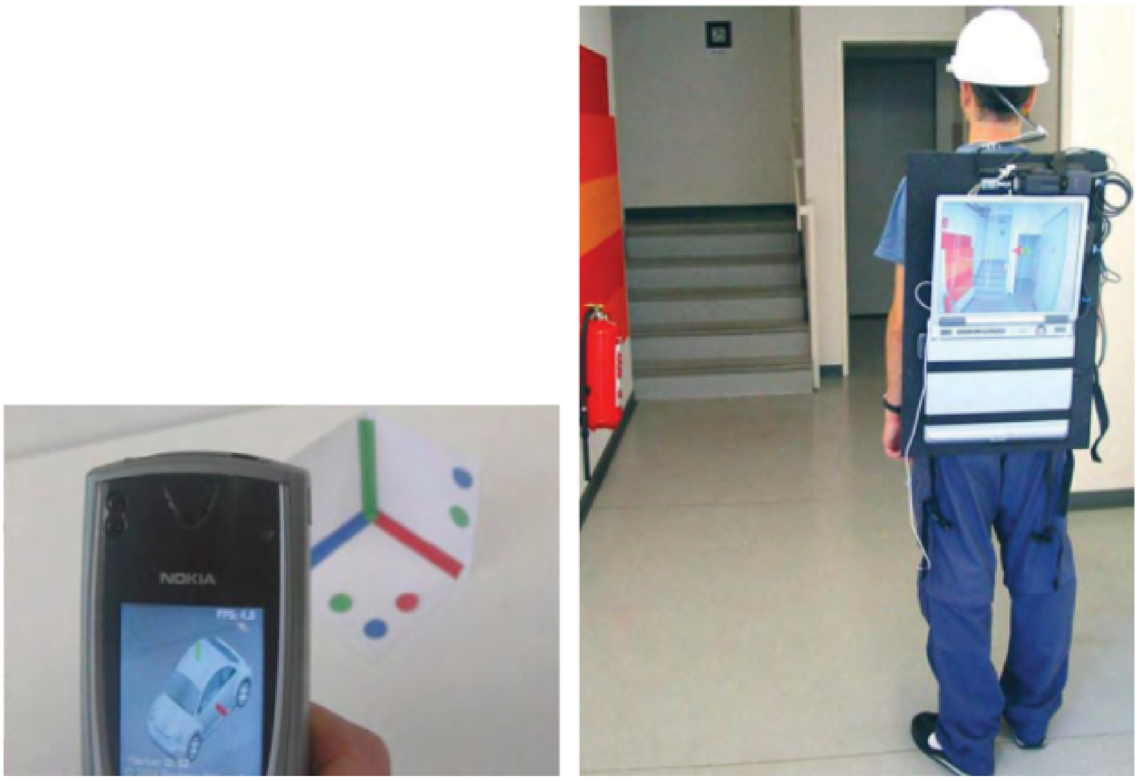


Figure 2.3: A phone (left) and a notebook (right) [HHPC13].

2.2.2 Mobile AR

Mobile AR relates to the usage of this technology in a mobile environment. It consists in using mobile devices, such as smartphones, as the medium through which we perceive the superimposed reality. The high mobility inherent to Mobile AR (MAR) itself enforces higher flexibility regarding the user's movement and spatial surroundings, so they do not feel constrained, but truly mobile. Another important factor that weighs on this flexibility is the physical part of the device itself, as the smaller and lighter it is, the easier it is to carry it around [HHPC13] (see Figure 2.3).

Albeit smartphones are among the most popular devices in this area, they are not alone. Keeping in mind that the purpose of Mobile AR is all about a mobile environment, this does not restrain the concept to just smartphones. Some suitable devices, according to [HHPC13], are shown below.

- *Notebook computers*. Present in early MAR prototypes, though they had major drawbacks related to the fact that they were used in a backpack. Besides the backpack, another display (*e.g.* HMD) was needed as the the notebook's screen was not of easy access (and thus was only used for debugging), contributing in size and weight to the hindered mobility of this setup.

- *Personal Digital Assistants (PDAs)*. A simple device which is easy to both transport and use, but due to its small size, the screen is commonly also very small. Besides, most PDAs do not support floating point operations and have poor computational power. To minimize the impact of the last problem, some approaches outsource CPU-intensive tasks to servers, leaving the PDA as simple client mainly for interaction and display.

- *Ultra mobile PCs*. A computationally powerful portable computer commonly focused on the commercial business market, making it expensive and thus less appealing. Also, similar to PDAs, the limited screen size is a negative factor.
- *Tablets*. Provide a large and multitouch screen, allowing for more intuitive interactions and more visualization capabilities. However, tablets tend to be rather expensive, as well as being tiring for long-time single-handed use due to its weight.
- *Mobile phones*. These are devices that brings measurable process in many aspects. Mobile phones may provide embedded cameras (act as see-through display), built-in sensors (used for pose tracking), powerful processors given their size and dedicated graphics hardware, making it the predominant device for Mobile AR. But even though it has many capabilities, it also has many weak spots. The processing power may be limited and comparable to that of an old computer, also worsened by slow memory access and tiny caches. The sensors are not flawless, cameras have a narrow field of view and the obtained image may contain noise and the accelerometer may be too noisy to accurately detect the user's position. However, as devices with minimal intrusion, high portability and social acceptance, it still makes it the most popular device in MAR.
- *AR Glasses*. Provide a hands-free experience with the advantage of not requiring users to look down at the screen of a device, making them feel more natural. Though there is some controversy regarding its classification as a MAR device, in this case it is considered as such since common MAR goals can be achieved through these glasses, such as facial recognition and path finding.

As a very versatile type of interaction, it has multiple applications in the real world. Some of these are mentioned in another work [HHPC13], such as tourism, navigation, games, training, education, modelling and scene construction. There are, however, some downsides, such as poor-looking user interface, lack of feedback from the user interface and a tiring hold [KBB$^+$18]. Nevertheless, there are also efforts to surpass these problems, such as ergonomic improvements (joystick handles or making the device wrist-worn) under study, paving the way for an even better interaction with the users.

2.2.3 Tangible AR

Tangible Augmented Reality refers to when overlaid AR content and real objects have a deeper connection with each other. Put simply, this interaction technique combines the physical manipulation of everyday objects working as Tangible User Interfaces and the display capabilities of AR. This way of interacting with AR makes it easier for the user to understand the interface and

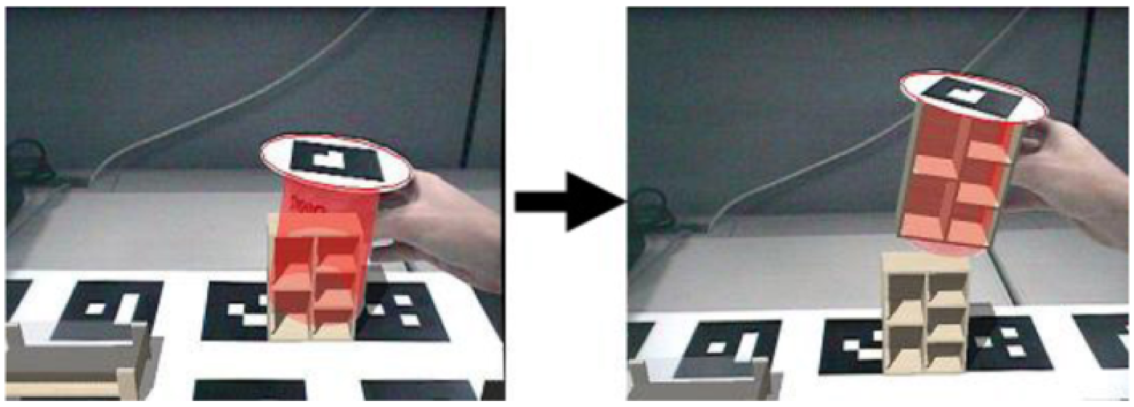


Figure 2.4: Virtual manipulation using a physical cup (Magic Cup) [BKP02].

its available interactions, as every tangible element of the interface serves a single purpose in the virtual world [BKP02]. Such an interface can be seen in Figure 2.4

In a general way, an interface is the means through which a user interacts with a system. When talking about a Tangible AR interface, a virtual object must be mapped one-to-one to a physical object (*i.e.* a virtual object is associated to one real object and vice-versa) and interactions with a virtual object should be done by manipulating its respective physical object [BGSD09, BKM09b].

Previous works have provided a set of design principles, intending to help create powerful Tangible AR applications [BKM09b, BGSD09]. The principles from both papers are stated in the following list:

- "The use of physical controllers for manipulating virtual content"
- "Support for spatial 3D interaction techniques (such as using object proximity)"
- "Support for both time- and space-multiplexed interaction"
- "Support for multi-handed interaction"
- "Matching the physical constraints of the object to the task requirements"
- "The ability to support parallel activity with multiple objects"
- "Collaboration between multiple participants"

[BGSD09] also suggests some key elements to take into consideration when developing an application with Tangible AR interfaces:

- "The physical elements in the system"
- "The visual and audio display elements"
- "The interaction metaphor that maps interaction with the real world to virtual object manipulation"

These interactions can be further enhanced when the result has direct consequences back on the physical environment, also called Ambient AR. Such an example would be a system described in [BGSD09], where tangible cubes influence a light's color or intensity in a certain room, just by shifting and rotating these cubes. An example in the same system also exemplifies that Tangible AR can modify the distribution of air flow with real fans being controlled by a Tangible Interface, such as a cube's movement over a 2D plane.

2.2.4 Interchangeable Reality

When explaining the concept of Augmented Reality, it is also relevant to address the concept of virtual objects and environment. Besides the fact that Augmented Reality does not exist without the virtual objects, there is also another reason that makes it appealing to mention a virtual environment, and that is the popularity of Virtual Reality (VR) and its similarities to AR.

Virtual Reality wields a fully virtual world for the user to explore. Unlike AR, where the user sees the real world and digital information is overlaid on it, in VR the perceived world is the virtual world, with all its virtual objects, providing full immersion. A fully virtual world may mimic the same constraints that rule our world, such as physics principles, or may also exceed the bounds of the world we know and create a world where the "normal rules" apply no more, enabling different mechanics, material properties and space and time notions [MK94]. These advantages over Augmented Reality seem very appealing, however, there are also negative issues. As an example, VR applications should be careful with performance, as a slow interface (or one that even moves at all without the user willing to) can cause discomfort and nausea in the user, as the user's senses transmit different informations to the brain [FF16].

Augmented Reality and Virtual Reality appear as technologies very closely related, but also with some disparities. This became a problem when the *VR* label became used in environments that were not entirely immersive, but also seem related to this family of technologies. With this problem in mind, a virtuality-continuum [MK94] was proposed in order to define the boundaries of these terms and how they relate to each other, also paving the way for terms like Augmented Virtuality (AV) and Mixed Reality (MR). As seen in Figure 2.5, this continuum provides a spectrum of interaction environments, ranging from Real Environments, where there is no virtual immersion whatsoever, to Virtual Environments, where immersion is total, passing through Mixed Reality, where real and virtual environment are mixed.

However, the Virtuality Continuum does not make any reference about many of these environments being used simultaneously in the same system, in a collaborative fashion. Even though many systems are developed with the goal of interoperability with many environments from the Virtuality Continuum, this collaborative aspect is usually a minor feature with little spotlight [SD06, Mai17]. The use of this collaborative interaction technique that takes advantage of the different environments for executing different tasks is not easy to find, although it is not completely non-existent [PDE^{+}17, AZDA15]. The term Interchangeable Reality is a concept defined inside the context of PAINTER, which refers to an environment supporting a blend between many simultaneous realities from the virtuality-continuum in the same system.

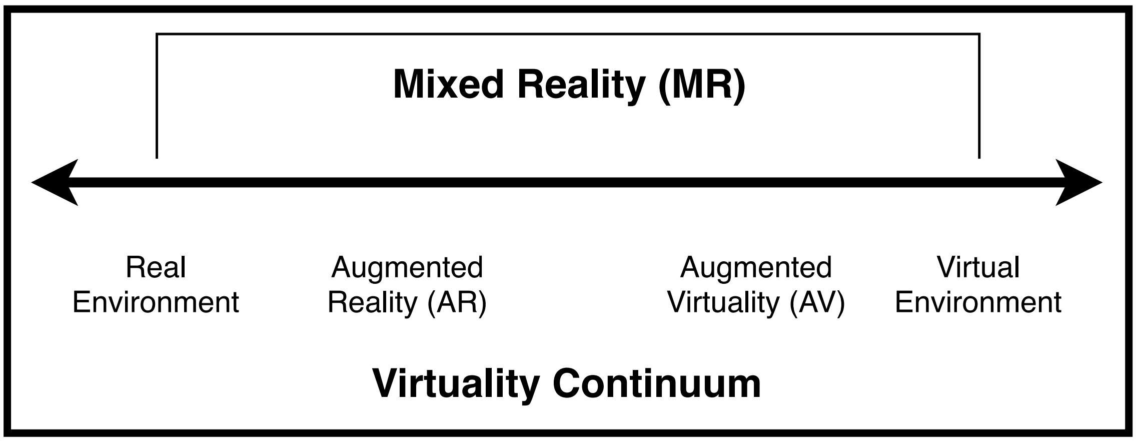


Figure 2.5: Virtuality Continuum, as presented by Milgram and Kishino [MK94].

But even if the literature does not have much to show regarding Interchangeable Reality, it does not mean that we cannot learn from those tools that make this cooperative aspect of secondary importance. The system architecture suggested by Seibert and Dähne brings many interesting ideas, such as the usage of VRML (Virtual Reality Markup Language) or its successor, X3D, since these formats allow sharing information about the 3D scene (even including the behavior of virtual objects) in an accepted standard by ISO and IEC, which is also independent of the device used [SD06]. Another idea from the same project, also found in generic software development is, the separation of the main logic from the code for handling the hardware devices, so as to allow better interoperability.

The work of Arenas, Zarraonandia, Díaz and Aedo consists of a set of tools and applications designed with the development and execution of AR and VR Education Games in mind [AZDA15]. The VR and AR 3D scenes are created separately and some of their objects are later matched to each other in a *MR scene*, therefore ending up with 3 type of scenes: with content shared between AR and VR, VR-only content and AR-only content. The game rules are created using GREM (Game Rules scEnario Model), which is XML-based, providing a device-independent format for these behaviors. The presented proof of concept used a firefighting scenario, where an AR user had an observer role, while a VR user played the firefighter role.

Yet another work with Interchangeable Reality at play is [PDE+17], presenting the CoVAR system, capable of Augmented Reality and Augmented Virtuality simultaneously. Unlike the system from [AZDA15], which centered around high-level development, this work is more focused around interactions, providing a series of mechanisms for greater feeling of presence with two remote users, one in AR and another in AV. It reinforces the use of visual and non-visual cues, as well as environment reconstruction, allowing the use of objects in the communication.

2.3 Collaborative Augmented Reality

As was seen in Section 2.2, Augmented Reality has a great potential for improving user's interactions with their surroundings, whether from a professional and performance perspective, or from an entertainment perspective. If AR has such an impact in interaction, it is necessary to ponder over its application with multiple users. Some usages for this sub-field of AR have already been explored, for example, in police and military personnel [KBB$^+$18]. In this particular case, a collaborative tool was developed to be used in joint planning tasks when dealing with catastrophic situations. The results showed some benefits regarding task performance when comparing the tool's usage with traditional approaches.

However, similarly to general AR, there are certain key attributes that are recommended to be paid attention to, so as to achieve a better collaborative AR experience. The subject of User Experience (UX) is not always an easy issue to tackle within AR, as it can be challenging to allow interaction and modifications of the scene for multiple users simultaneously. Previously works have identified the following 5 key attributes as characteristic of these environments [SSFG98, BK02]:

- "*Virtuality*. Objects that don't exist in the real world can be viewed and examined"
- "*Augmentation*. Real objects can be augmented with virtual annotations"
- "*Cooperation*. Multiple users can see each other and cooperate in natural ways"
- "*Independence*. Individual users control their own independent viewpoints"
- "*Individuality*. Displayed data can appear in different form for individual viewers depending on their personal needs and interests"

Additionally, one of the reasons indicated for why users prefer AR to immersive virtual environments and do perform better on some collaborative tasks is because they can interpret each other's non-verbal cues, as they are able to see each other. This makes the visualization of other collaborating users, though not mandatory, extremely useful and something worth taking into consideration.

2.3.1 Face-to-Face Collaboration

Face-to-face collaboration is the most natural form of collaboration. Unwittingly, every human being transmits a series of cues when communicating. These cues include speech, gestures, gaze and other non-verbal cues. The way we interact with our surrounding environment and its objects also has an important role in communication. These surrounding objects can be used as reference frames for communication, as semantic representations, or even be used for comparisons through their appearance or their physical affordances, such as size or weigh [BK02]. This makes it highly desirable to attempt to take advantage of these cues when using AR for collaboration.

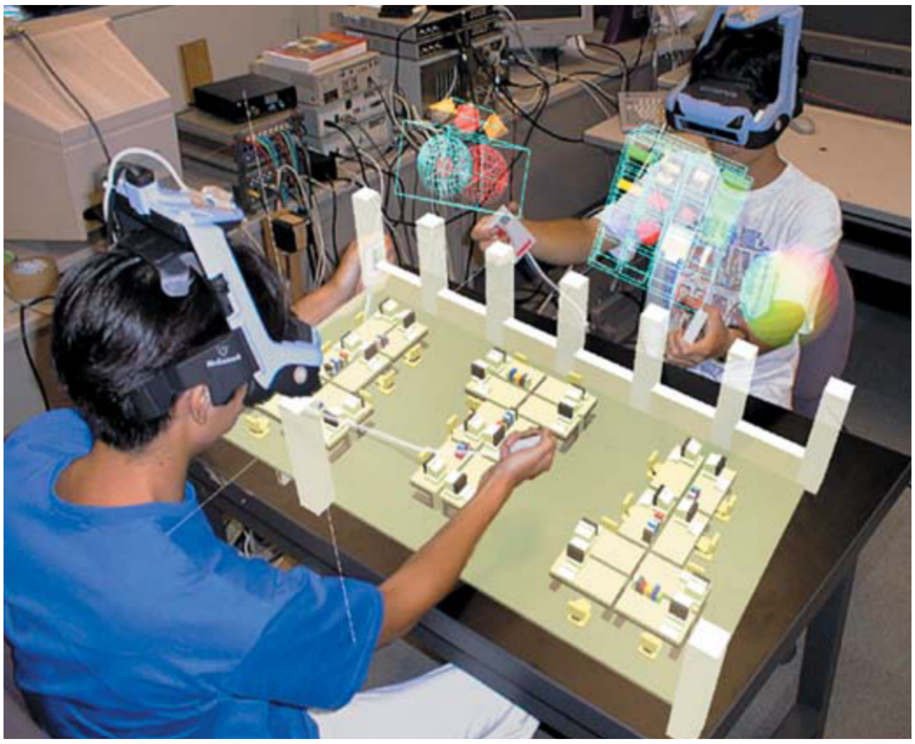


Figure 2.6: A collaborative interface [BK02].

Computer-supported collaborative work is not a recent topic. Experiments with screens for co-present collaboration, though of some utility, cannot take advantage of several principles of natural human communication, commonly creating an artificial separation between the shared collaboration space and the physical world. Also, when a group of people is crowded around a screen or looking at a projection, the group is usually inhibited to use natural communication behaviors or refer to real objects. Observations also found that, even when using large shared displays, these natural interaction behaviors almost never happen due to the lack of support for these interactions, also related to the lack of input devices for this co-located collaboration [BK02].

Face-to-face AR collaboration tries to get the most out of these communication cues, while enabling several users to experience a shared augmented physical space, hopefully intuitive, in order to enhance collaboration efficacy [BK02]. An example of such an interface is presented in Figure 2.6. However, this is not so easy to achieve, and even though AR definitely has the potential to serve as an intuitive interface, it can also undermine its own efforts. An example of such a problem exists with HMDs, which given the importance of eye contact in natural communication, makes it very challenging for users to establish effortless eye-contact [KBB$^+$18].

UMI3D is an example that brings all these principles to life, as a toolbox developed with complex cooperative situations in mind. Its feature set was designed with the purpose of addressing some of the main concerns of CSCW [CPB18]. The feature set is presented below:

- *Shared Context*. A basic building block for CSCW. It means that many users share a context and knowledge. For this feature, shared feedback upon user interaction is needed.

- *Coordination Mechanisms*. Also a building block of CSCW. It infers that the system should improve coordination of collaborative tasks and be prepared for cooperative and parallel interactions.
- *Malleability*. Assumes that users should have the ability to add or modify coordination mechanisms in runtime.
- *User roles*. Assuming that complex interactions exist in the system, then it is also assumed that there are different roles with different rights. However, the existence of these different roles should not prevent users from cooperating.
- *Individual activities*. Users should also be able to perform individual activities and receive individual feedback in case the other users do not need to receive it, which should be properly evaluated.
- *Awareness of others*. Consists of enabling users to understand each other's activities.

One of several experiments performed with collaborative environments was related to matching contextually related augmented objects, with many users present in the same physical space executing the same task [BKP02]. Even though they were not told they should cooperate, cooperation naturally emerged, even among complete strangers. This work concluded that this combination between Tangible Interfaces and Augmented Reality enables the developer to take advantage of this natural capability for cooperation and go along with it.

2.3.2 Remote Collaboration

When talking about remote collaborative work, it usually involves a medium through which communication is transmitted. Most people will usually think about Skype [6] or Hangouts [7], when asked about these mediums. These two platforms enable text, audio and voice-chat (among other features), thus being a common and easy solution for people wanting to work together or hold a meeting while being at a distance. However, while using this type of service, users do not usually feel that much closer to each other.

In fact, one of the main challenges in collaborative AR systems is how to handle the physical absence of a user in the local collaboration space [KBB$^+$18]. In 2002, it was considered of high difficulty for the available technology at that time to provide remote participants with the exact same experience as if they were physically present in the collaborative space [BK02]. This is especially true when we take into account that a great number of attempts were made with screens as the medium of communication. Even when multiple cameras were used to capture and reconstruct avatars for all users involved, the use of the screen, which was not even portable, deeply hindered the efforts at enhancing collaboration. Although the field of remote AR collaboration has advanced greatly since 2002, this statement still holds somewhat true. Despite still not being

[6] Skype, https://www.skype.com
[7] Google Hangouts, https://hangouts.google.com

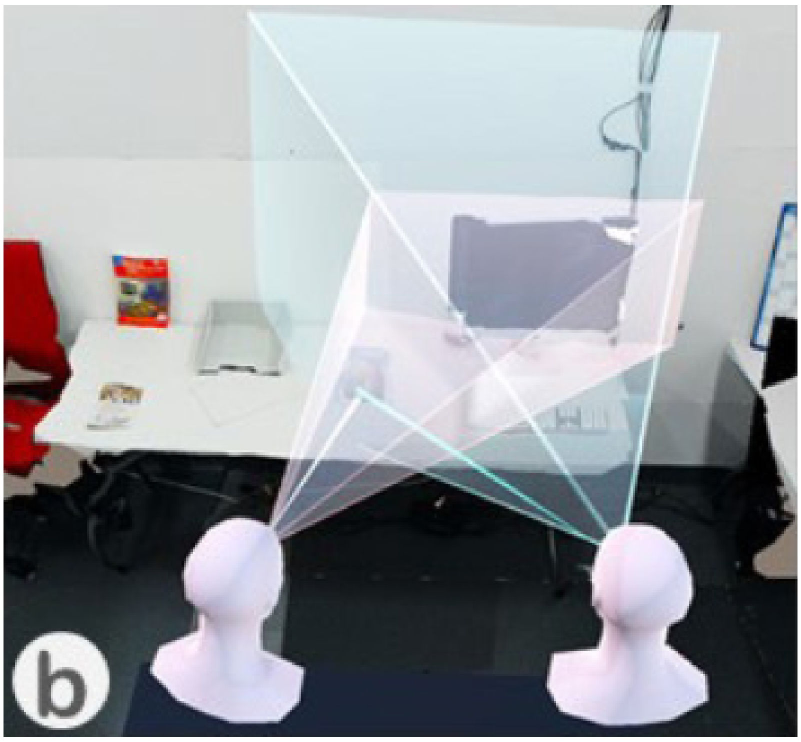


Figure 2.7: Users viewing each other's Field of View and Gaze Direction [PDE^{+}17].

good enough for real usage with the general public, remote AR collaboration is presented as having no technical limitations and appearing very feasible (save for needs of high bandwidth related to the synchronization of remote users' avatars across workspaces) [CPB18]. However, the matter of transmitting the unconscious and yet fundamental non-verbal cues is still a great problem.

Taking into account the challenge presented in the previous paragraph, it is easier to understand why one the main goals of remote collaborations systems is to allow users at distant physical locations to feel as if they are in the same space, and study different methods for achieving this [PDE^{+}17]. Ideally, far apart users should be able to perceive each other's cues and share a virtual environment so as to take the most advantage out of these cues (Figure 2.7).

CoVAR is a MR system with a deep focus on this remote problem [PDE^{+}17]. It consists on the 3D reconstruction of an AR user's environment in an AV user's environment, thus making this environment shared. Augmented objects placed in the AR's physical space are also shared with the AV user. With the goal to assess the importance of cues in communication, the system enables the sharing of the AR's Field of View and Gaze Direction, as well as providing an "AV-Snap-to-AR" feature, allowing the AV user to snap to the AR's head position and also orientation if wanted. The results found that the usage of awareness cues in a remote environment was crucial for improving performance.

2.4 Technologies

The AR community has reached a time when there are several different libraries, frameworks and diversity of hardware that can be used for development, all with different advantages and disadvantages. In this section, a brief overview about the most used software and hardware is given.

2.4.1 Software

In this section, the most prominent AR development tools are shown and their suitability regarding this dissertation. An interesting feature which exists in all of below tools is integration with Unity[8] game engine, thus making Unity a very appealing choice for development. Another important characteristic which is important to note is that most of these tools have at least a free tier, specially suited for non-commercial purposes.

Vuforia [9]

As one of the most popular tools, it is also one with the highest number of features. It can track both planar images and simple 3D objects in real-time, also supporting fiducial markers with great flexibility. This flexibility with markers is one of its key features, called *VuMarks*, which is a combination between a fiducial marker and an image. Simple 3D objects can also be used in replacement of these markers. It can play videos on supported surfaces, as well as recognize text (with possibility of custom vocabulary expansion). The interfaces can also show virtual buttons and background effects, as well as manage occlusion for finding partially hidden objects. It offers support for smart glasses and integration with ARCore, ARKit, Android, iOS and UWP (Universal Windows Platform) while still having a free tier. Some known apps made with this framework are BMW Individual 7, MANGO MNG and Nikola Tesla.

Wikitude [10]

Given the general user's preference for free software, it is interesting to find this as one of the most popular tools despite not having a free tier (only a trial), which can be interpreted as a sign of quality from this SDK (Software Development Kit). With tracking capabilities ranging from (SLAM-based) surface tracking to geolocated markers, it still finds place for a very unique feature: extended recording and tracking of objects. This enables the user to scan a marker and not having to keep to marker in sight so as to see the virtual object that emerged from it. It also provides integration with smart glasses, Android, iOS and Windows. Some apps made with this tool include Time Magazine Special, ROOMLE and Wikitude Navigation.

[8]Unity, https://unity3d.com/
[9]Vuforia, https://www.vuforia.com
[10]Wikitude, https://www.wikitude.com

Kudan [11]

Kudan shows itself as very versatile tool in terms of tracking. It is capable of marker-less and marker-based tracking, with the latter having the possibility to be applied to posters or stickers besides the usual QR-codes. This tool is also capable of 2D and 3D objects if their models have been provided beforehand. Every detected object is recognized by initializing its local coordinates and adding a layer up on it. Likewise, it supports camera sensors for other image enhancements such as higher accuracy on objects' locations, 3D graphics and even real-time texture morphing. Compatible with iOS and Android, it has given birth to apps like Virtual Mustang for Ford and DHL Formula E Xperience.

ARKit [12]

Apple's solution for AR is available for iPhones and iPads with iOS 11 or higher. It is able to detect user's facial features and further apply effects in real-time as well as understand its surroundings (with special emphasis on planar surfaces) and the lights present. Visual Inertial Odometry is also implemented, mainly for motion tracking. However, Apple did not take long to release a second version of the tool, ARKit 2. It places a high focus on collaborative interactions, scene persistence between app restarts and also detection and tracking of 2D and 3D objects. The latter feature even gets as far as to detect more complex 3D objects such as furniture and toys, for example.

ARCore [13]

This is Google's response to ARKit, with minimum OS requirements being 7.0 for Android and 11 for iOS. Its core features are somewhat similar to ARKit, however this platform still manages to pull some new tricks. It brings a big emphasis on tracking the phone's position relative to its surroundings, as well as a proper understanding of the said surroundings through detection of the location and dimensions of the surfaces as well as estimation of real-life lighting conditions. With these features, others can be better achieved, such as placing objects or the physical space or even text. Many popular apps use this tool, such as Just A Line, ARuler and Ikea Place.

2.5 Summary

This chapter presents AR, a powerful technology capable of blending real and virtual environments, with a high range of applicabilities. It enables a plethora of different ways to track the environment, such as the use of markers when only a constrained environment is concerned, or algorithms such as SLAM in order to take full advantage of the environment.

Likewise important is the device and type of interface chosen to be used as medium, since different interfaces provide different capabilities. A Mobile AR interaction assumes the usage of

[11] Kudan, https://www.kudan.eu
[12] ARKit, https://developer.apple.com/arkit
[13] ARCore, https://developers.google.com/ar

mobile devices, capable of moving around the environment. When combined with such interfaces, applications should present much more flexibility to the environment, as they can be used in many more places and contexts. Yet another technique is Tangible AR, which enables a more "hands-on" approach, where a user interacts with physical objects in order to affect the virtual ones. This interaction proves to be very intuitive and helpful.

Even though the Virtuality-Continuum is a known reference, specially in the Mixed Reality community, the exploration of many of its levels simultaneously is not as common. Neverthelesss, projects with these characteristics exist, such as the MR system CoVAR.

This chapter also shows the high applicability of Collaborative Augmented Reality, with a heavy focus on enhancing natural channels of communication. These channels include verbal and non-verbal cues, of which the latter is a more difficult matter. In face-to-face collaboration, these are easier to take advantage of and incorporate, also benefiting from the use of Tangible interfaces. However, in remote collaboration this becomes much more complicated, and so a great part of the research focuses on environment reconstruction and mechanisms that allow non-verbal cues to overcome the distance.

Taking into account all the different interaction techniques in AR and their unique benefits, a solution will be proposed in the next chapter, based on all the knowledge contained in this chapter. This solution will take advantage of several of these contributions in order to create a prototype, which will then be explored and tested in order to answer the proposed research questions.

Chapter 3

AR Interaction and Collaboration Solution

In this chapter, the proposed solution and several details will be explained. The solution consists of an AR component for an Interchangeable Reality framework that enables the possibility of collaboration in real-time for editing and interacting with a Virtual Environment, between multiple simultaneous users. After a thorough analysis of the current state of the art with a high focus on AR interactions and, specifically, Collaborative AR interactions, the system's detailed requirements, technologies, evaluation methodology and expectations are defined.

3.1 General Description

This tool should, in its essence, provide the base mechanics for intuitive interactions with an AR collaborative interface. In the context of PAINTER, the research project that motivated this dissertation, the environment is one where either Virtual Reality and Augmented Reality can be used, together or separately, in a collaborative manner. As such, the interface should take this highly specific context into account, making sure that the interface is ready for many users, no matter the device they are using. But most of all, the interactions between different devices should be unified, as even though they are intrinsically different, the result of their interactions should be the same, while still allowing for each device to have customized means of interaction, adapted to its advantages and limitations.

As a framework, the main goal is not to provide many specific features, but instead lay the foundations for these features to be easily implemented or extended. With this in mind, there are three main areas on which this framework should focus: interaction with the virtual environment, interaction with other users and visualization techniques. A more in-depth list of what mechanics are expected from each of these three areas is presented in Table 3.1.

Interaction with the Virtual Environment	Object Manipulation Operations
	Object Finding Mechanics
Interactions with Other Users	Call Other Users' Attention to the Calling User
	Call Other Users' Attention to an Object
	Show Helpful Annotations
Visual Aids	Runtime Changes on what should or should not be seen (such as non-controllable objects, like walls)
	Change of Perspective to Other Users
	Bird's-eye View

Table 3.1: Main areas on which this framework should focus.

3.2 Requirements

Given the general description of the system, more specific system requirements must also be defined in order to better guide the development of the tool. The system requirements hereby presented do not refer simply to interactions, but rather the AR user's perspective of these interactions, divided according to the three areas presented in Section 3.1, containing as well the networking requirements.

These requirements are also shared with PAINTER's own requirements, and as such were proposed and defined with the help of many researchers from this project. A minimum viable product should be developed in order to obtain relevant information to allow the execution of user tests. As the virtual environment, a small house or apartment was chosen. The context for the user tests should be one related to decoration of house interiors so that the minimum set of features can be tested. These minimum features include:

- Manipulate objects.
- Find objects.
- Find users.
- Call a user's attention.
- Place annotations.

In Table 3.1, general areas on which this solution should focus were presented. As a further complement, Table 3.2 turns these more general requirements into specific features, able to be implemented. These features are explained in further detail in the next subsections, divided according to the areas they belong to. A conceptual diagram for the interactions is also presented in Figure 3.1, in order to provide a better overview of how most of these features might be connected to each other.

3.2.1 Interaction with the virtual environment

As one of the three core areas of this tool, interactions with the virtual environment will, in all likelihood, be the area users dedicate the most time to. This is especially true when considering that

Feature	**Area**
F.1 Select an object	Interaction with the Virtual Environment
F.2 Translate an object	
F.3 Scale an object	
F.4 Rotate an object	
F.5 Create a preview of a manipulation	
F.6.Accept/Reject a manipulation preview	
F.7 Show all present users	Interactions with Other Users
F.8 Call other users' attention to the user in question or to an object	
F.9 Place and view annotations	
F.10 Toggle walls visibility	Visual Aids
F.11 View other users' field of view	
F.12 Bird's-eye view	
F.13 Creation of rooms	Network
F.14 Set maximum number of users per room	
F.15 Restrict type of reality	

Table 3.2: List of the features and the areas they belong to.

user tests will be about decorating a house, thus requiring a large amount of object manipulation. As such, these features need to be particularly well thought for giving feedback to user, so they do not think of the system as cumbersome. The features are presented below:

- *F.1 Select an object.* The user should be able to select an object, providing access to the manipulation of said object. A selected object should have a different visual representation that makes it clear for every user that it is selected by a specific user. With this in mind, the selected object could have a semitransparent overlay all around, whose coloring should be different from user to user, implicating that each user has its own unique color when selecting an object. Upon deselection, the colored overlay should disappear and the object's material should remain the same as the original. Concurrent interaction should also be prevented so as to prevent confusion in the would-be concurrent users. This latter issue could be solved by implementing a list of ownership requests per object, and users would pass the ownership down the list when deselecting an object. Also, when a user looks in another direction causing the selected object to disappear from the screen, some sort of indication should point the direction of the selected object, so that the user always knows where it is.

- *F.2 Translate an object.* The user should be able to move an object to a different position. This operation should only be available after an object is successfully selected.

- *F.3 Scale an object.* The user should be able to scale an object, changing its dimensions. This operation should only be available after an object is successfully selected.

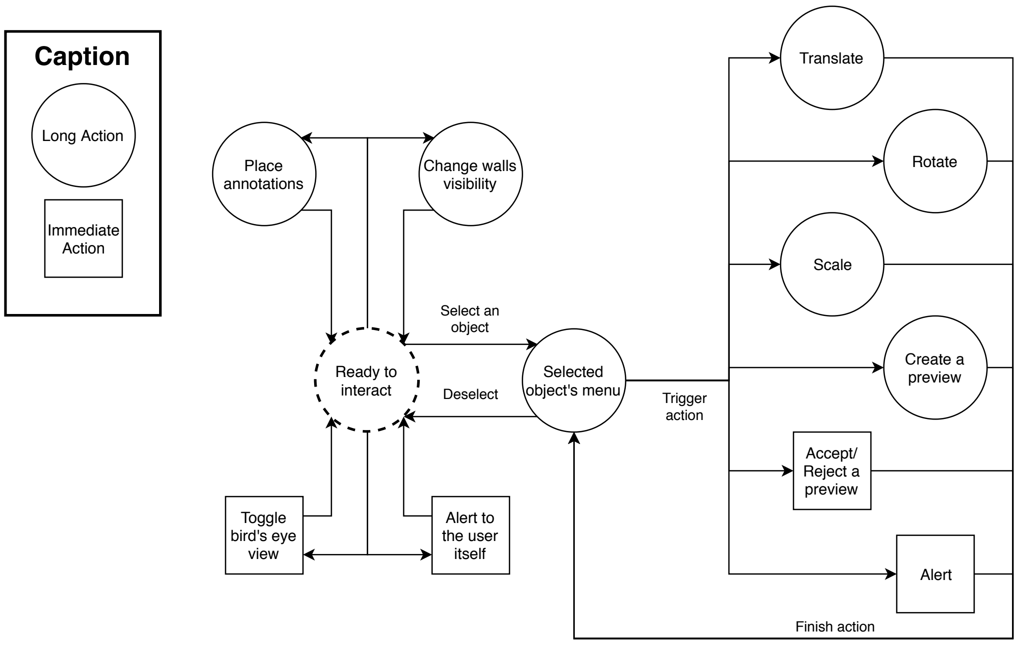


Figure 3.1: Conceptual diagram displaying interactions available.

- *F.4 Rotate an object.* The user should be able to rotate an object, changing its orientation. This operation should only be available after an object is successfully selected.
- *F.5 Create a preview of a manipulation.* The user should be able to make a preview of an object. This preview's main usage should be to make suggestions without really changing the original object. It should be a clone of the original object, though with a different material to make it appear as if it is not real (*e.g.* make it translucent or appear like wireframe). This clone could be subjected to all the other object manipulation operations, enabling users to see both the original object and a modified clone with a new position, rotation and/or scale.
- *F.6.Accept/Reject a manipulation preview.* The user should be able to accept or reject an object's preview. This action should be available either by interacting with the preview or the original object.

3.2.2 Interaction with other users

Though interactions with other users is not very heavy on user feedback, it does not mean this core area is not conveying relevant information in other ways. In fact, this area aims precisely at conveying information in a seamless way, so the user receives information about who surrounds them without much effort. These features for interacting with other users are presented below:

- *F.7 Show all present users.* The user should be able too see a virtual representation of all other users present on the same Virtual Environment. In the case of a mobile AR user, such a representation could be a smartphone-shaped box, while a headset or a head-themed box could suffice for a VR user. These representations should follow the position and orientation of the view by which each user has access to the virtual world, thus allowing real perception of each user's situation.

- *F.8 Call other users' attention to the user in question or to an object.* The user should be able to make calls of attention which are easy and quick to track down. These attention calls could point either to the user the user in question or to a designated object, with the goal of making it easier to refer to a specific object. This attention call could include spatial sound to better help users find the source of the alert, as well as some animation that more effectively calls users' attentions.

- *F.9 Place and view annotations.* The user should be able to place annotations virtually anywhere on the Virtual Environment. These annotations could come under the form of arrows, text or sprays. They should be easy to spot in the environment, and at the same time not cluttering, as their role is to help understand what other users are doing or viewing. Arrows could be more used as means of calling attention to a certain part of an object or to a place, which could help, for example, to better choose the location where to move an object. Text would not have such an active role, but rather a more passive and explanatory one, as it could be used to leave notes regarding certain objects or areas. Sprays would be a more all-round tool that could be used for drawing or for other more imaginative purposes.

3.2.3 Visual aids

Comparing to the two previous core areas, this one is the most passive of all. Though being rather focused on users' perceptions, similarly to the one about interacting with other users, it focus on mechanics that provide general cues to the user about what surrounds them. Not just that, but it also has the capability to change the user's perspective regarding the whole system. The features that would make this possible are shown below:

- *F.10 Toggle walls visibility.* The user should be able to change some non-selectable objects' transparency. Since this is Augmented Reality, it may be desirable for some objects (*e.g.* virtual ground or some virtual walls) to be invisible or show some transparency. The goal here is to allow some distance from the VR end of the Mixed Reality spectrum, as to keep every object's opacity will make the environment almost fully virtual. With this in mind, and by turning some key objects invisible or semitransparent, we can allow the real and virtual to blend in true AR.

- *F.11 View other users' field of view.* The user should be able to view other users' frustums, in order to better understand what the other users are seeing. These frustums should be rendered with high transparency, so as to not get in the way of viewing the rest of the environment.

- *F.12 Bird's-eye view.* The user should be able to get an elevated view of the scene, so as to better view the scene as a whole. This could be implemented with the help of an AR marker, which would contain, encompassed within its bounds, the full scene, thus allowing a this view to be easily used, as it does not require extra interactions through the interface, just showing the marker.

3.2.4 Networking

To understand the requirements presented in this particular subsection, it is important to first understand the concept of rooms in online multiplayer games. A room is usually used to create a match for a certain number of players. It is a mechanic used to group players together in small groups. This means that, even though there is a great amount of players online at the same time, they do not need to be (nor should be, in most cases) playing all together. Applied to this dissertation's context, this means that different scenes may be available for choice, and users may join them as long as they do not surpass the maximum number of users permitted in that 3D scene's room.

- *F.13 Creation of rooms.* The user should be able to create multiple rooms for users to gather and customize certain aspects of the each room, in order to provide the desired experience.
- *F.14 Set maximum number of users per room.* The user should be allowed to set the maximum number of users each room can have, as a high number of users will also put more stress on the system, which could lead to a decrease of performance as well as unpredicted events with the experiment at hand.
- *F.15 Restrict type of reality.* The user should have the ability to create/join AR-only rooms, VR-only rooms or IR rooms (where none of the previous restrictions apply). Once again, this may be a decisive factor for a good experience for every user involved, as some 3D scenes may have been specially crafted for AR devices, for example.

3.3 System Architecture

The proposed solution will be executed with a client-server architecture, as shown in Figure 3.2. The system will have 2 types of user roles corresponding to the 2 types of devices: AR and VR. Also, even though the intended purpose is to use AR and VR simultaneously in a context of Mixed Reality, the focus of this document is on the AR role of such a context. The system may also need a server, for synchronization purposes. So, when viewing the full architecture, there are 3 different components at play:

- *AR Client.* Represents an AR device connected to the system. In case of many AR Clients, they should share the same physical space, in order to enhance cooperation. AR Clients may be able to see the scene's virtual objects in their physical shared space, while also being able to interact with these same objects. Though it is not mandatory for AR users to share the same physical space, it should be noted that by not sharing this same space, collaboration

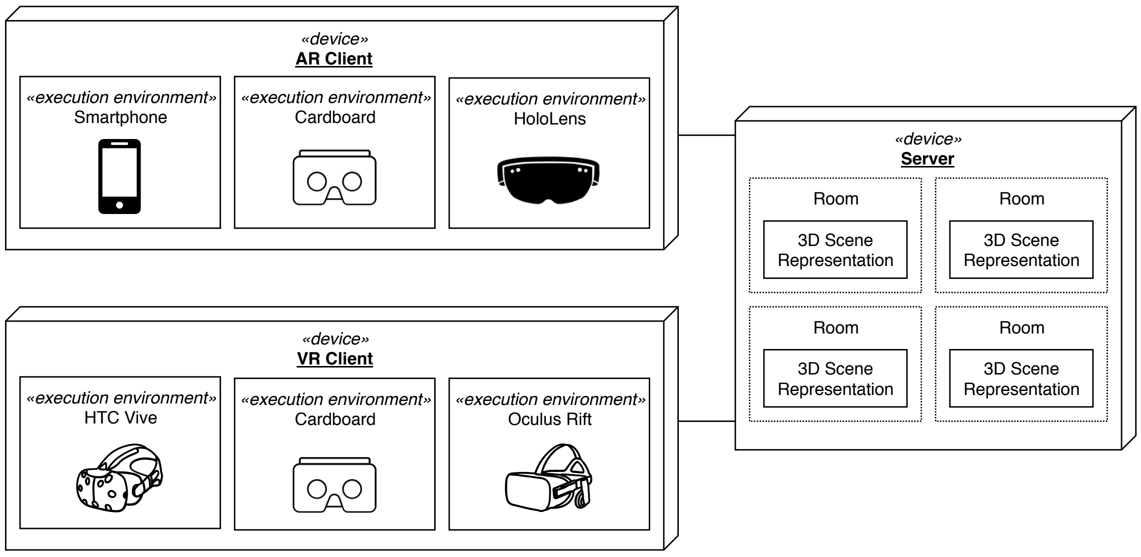

«device»
AR Client
«execution environment» Smartphone
«execution environment» Cardboard
«execution environment» HoloLens
«device»
VR Client
«execution environment» HTC Vive
«execution environment» Cardboard
«execution environment» Oculus Rift
«device»
Server
Room
3D Scene Representation
Room
3D Scene Representation
Room
3D Scene Representation
Room
3D Scene Representation

Figure 3.2: System architecture

efficiency may suffer. These interactions should be sent to the Backend Server as they happen, so as to maintain a synchronized 3D scene.

- *VR Client*. Represents a VR device connected to the system. The physical location of this Client is irrelevant, as its capabilities are not affected by such a factor, contrary to AR Clients.

- *Backend Server*. A server holding the state of the 3D scene. It is also responsible for keeping every Client's own version of the scene as synchronized as possible. It should also handle all the interactions coming from the AR Clients, update the scene accordingly, and send it to every device.

3.3.1 Interface

The interface is what bridges the user and the system, and should thus provide intuitive interaction so the user can confidently use the system without spending too much time learning how to use it. The main interface used will be a smartphone due to its high mobility and versatility. Nowadays most smartphones come packed with a myriad of sensors that can help keep track of the environment. These sensors range from gyroscopes and accelerometers capable of tracking the position and orientation of the smartphone, to the cameras commonly found in these devices, with the latter one being rather important since Augmented Reality depends heavily on Computer Vision algorithms applied to camera images.

Given such a powerful device, the interface should also make use of its capabilities to shine. As an example, it is understandable that, when holding the phone sideways (also known as landscape mode), users commonly feel more comfortable holding it with both hands instead of only one, as it

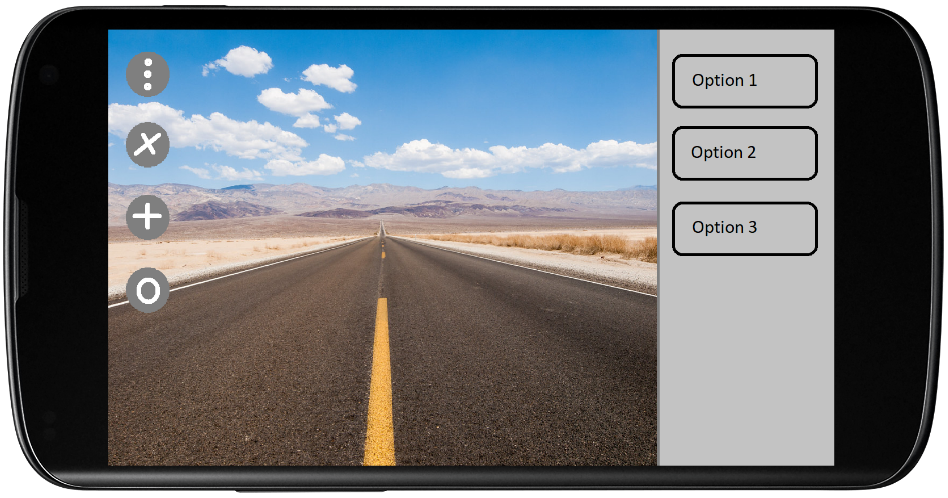


Figure 3.3: A mockup that represents the general appearance of the system interface.

provides more stability. However, if an application needs the user to change the hand posture quite often, it implicates that the phone will shake, which can accumulate errors regarding positional tracking. With this example, one can better understand how apparently inconsequential factors such as hand postures should also be handled carefully, and in this case the posture should remain two-handed as much as possible. In order for the stance to be kept, touch interactions should therefore be moved to the edges of the screen, as touching in the center area of the screen may cause the user to change stance.

An interface mockup that attempts to comply with these restrictions is shown in Figure 3.3. General interactions may be kept to the left side, such as leaving the app, changing some options regarding interactions or trigger an action that is connected to the user and not an object. On the right side, a menu with object-related options may appear when an object is selected. Such options should include manipulation operation such as translate and rotate, but also calling other users' attention to it or create/accept/reject a preview. However, these options should only appear when an object is selected, which means that when such an option has been chosen or no object is selected, the menu should not appear in order to free screen space, both to allow for a wider view and to enable the user to touch the screen while maintaining the two-handed stance.

3.4 Summary

The solution hereby proposed consists on part of a collaborative framework for Interchangeable Reality, with a stronger focus on the AR side of the tool. Developed with ease of use and effective interaction support in mind, it should enable a collaborative environment supporting many AR interactions, as well as the integration of a remote VR user. Such a cooperative environment should be supported by a server enabling consistency and synchronization across all devices, although users may restrict which devices can be together in which 3D scenes, along with other network customizations.

User tests should also be made. These tests should consist on a series of tasks, performed in a prototype scene developed using the framework. This prototype should allow the users to experiment different means of interaction and object-related operations, with a heavy focus on how the user really interacts with the system. Users should be evaluated with quantitative and qualitative tests, as both exact and subjective data are required to tackle the research questions proposed in Section 1.2.

It is expected that user tests help find, after a prototype is developed, how easy it would be to use such a system, as well as understand the system's impact on collaborative work. Even though the environment is one of Mixed Reality, users are expected to still feel confident and comfortable throughout the experiment. It is also expected that the test results should help better understand what helps users' interactions and what hinders.

Chapter 4

Implementation

In this chapter, a detailed explanation is presented on how the chosen tools were applied and used to create the solution proposed in Chapter 3. The reasoning behind most implementation decisions is also explained, as well as feedback given during informal tests realized along the development. A list of all features, as previously seen in Table 3.2, is presented in Table 4.1 including the implementation status of each one.

Feature	**Status**
F.1 Select an object	Completed
F.2 Translate an object	Completed
F.3 Scale an object	Completed
F.4 Rotate an object	Completed
F.5 Create a preview of a manipulation	Partially completed
F.6.Accept/Reject a manipulation preview	Completed
F.7 Show all present users	Completed
F.8 Call other users' attention to the user in question or to an object	Completed
F.9 Place and view annotations	Partially completed
F.10 Toggle walls visibility	Completed[a]
F.11 View other users' field of view	Completed
F.12 Bird's-eye view	Undone
F.13 Creation of rooms	Completed[a]
F.14 Set maximum number of users per room	Completed[a]
F.15 Restrict type of reality	Undone

Table 4.1: List of features and their status.

[a]Feature is completed, but not customizable by users.

This chapter starts by presenting the technological choices for the development in Section 4.1, followed closely by some features used in this this dissertation but not developed in its context, in Section 4.2. Sections 4.3, 4.4 and 4.5 are more closely related to interactions *per se* and how the user perceives the world with which they interact. Finally, Section 4.6 has more to do with system mechanics that attempt to help the user in better perceiving the world and other users' intentions.

4.1 Tools and Technologies

This sections intends to present every major tool used in the development of the framework proposed in Chapter 3. For each technology, a summary of each will be provided, containing a brief description of why it was chosen or what are its advantages.

- *Unity*. A highly versatile game engine, providing as well integration with all the technologies below. It should be used as the core engine of the framework, and thereby all the logic code should use this engine.
- *Vuforia*. A well-established SDK with plenty of features that fit the requirements, regarding AR interactions. It supports marker's extended tracking, which allows for users to delve into the environment even when the marker is not visible, thus making the exploration process more flexible.
- *ARCore*. An SDK for Augmented Reality specially tailored for Android, with features such as Motion Tracking (understands where the device is relative to the world, tracking both position and orientation) and Environmental Understanding (understands the location of planes in the real world, mapping the real world's boundaries). By allowing Vuforia's capability *Vuforia Fusion* to use ARCore, tracking can be highly enhanced, as well as the motion tracking that enables users to really walk inside the scene. *Vuforia Fusion* knows the device's hardware and software capabilities, using the best available, which for Android's case is ARCore.
- *LeanTouch*. A library that abstracts the touch inputs' logic provided by Unity. It offers many supporting utilities (*e.g.* counting each finger's touches, keep dragging finger's start position, etc) that greatly simplify the logic of this tool.
- *Photon Unity Networking (PUN)*. Photon is a networking engine, usually used for online multiplayer games. PUN is Photon's integration with Unity, which allows for players to synchronize object's position, orientation and scale, among other properties, while also supporting multiple useful remote operations. Developers may host their servers on a location of their choice, or instead may also use Photon's cloud service for this purpose.

4.2 VR and AR Interaction Features

As defined in Section 3.1, this solution consists of an AR component for a framework where both AR and VR-enabled devices exist. As such, it is important to create and describe the mechanics that make this Mixed Reality environment work, using the technologies from the previous section.

One of the basis for a collaborative environment in Photon is the assumption that all users have access to the same 3D scene. However, this is a problem since interaction mechanics are very different between AR and VR. The solution for this was achieved through Unity's additive scene loading. The 3D scene is first loaded without AR or VR-specific GameObjects, and then a device-specific scene is loaded into the already existing scene, adding its content to the original 3D scene. As long as the GameObjects of this device-specific scene are not created as network objects, they will remain local, allowing each device to deploy whatever it needs right to the scene. From an AR perspective, this device-specific scene includes components mostly related to the screen interface and to the markers used.

Though the main 3D scene is mostly filled with network objects, intended to be used by all users present in the room, some objects in the scene may not need to be tracked in the network. This includes walls and static objects, as well as other types of components that do not need to be tracked in the network to play their role.

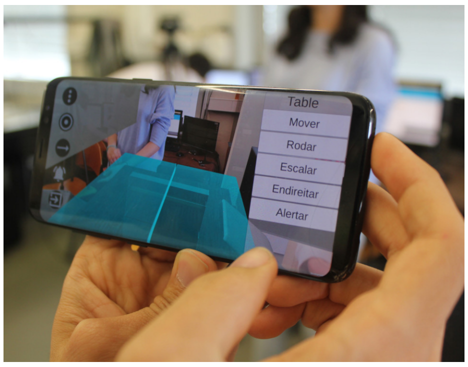


Figure 4.1: User operating an AR device.

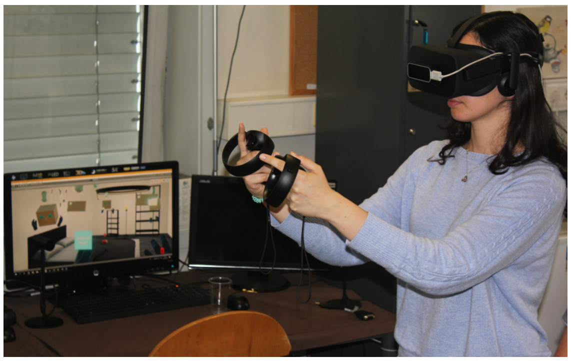


Figure 4.2: User operating a VR device.

Some features developed in the context of PAINTER and outside of this dissertation were also used. These include mostly network-related features and are presented in the following list:

- *Ownership Transfer*. A user needs to own an object to be able to change it, but only one user can own it at a time, which makes this feature truly necessary when multiple users exist. This mechanic allows a list of users' ownership requests to be created for each object. This is useful when one user requests ownership of an object whenever it is already owned and being used by a different user. With this mechanic, whenever the owner stops using the object, it is automatically passed down to the next user on the list.

- *User Numbering*. A user numbering system was developed so that, when a user exits the room, other users' numbers are updated in order to keep them sequential (*e.g.* when User 1 leaves the room, User 2 updates to User 1). These numbers act as IDs that are easier to view by users.

- *Other Network Utilities*. Several other minor network utilities were implemented. One of them served as a wrapper to see informations about the connections and its status. Another one also worth mentioning extends Photon's network instantiation feature, to allow naming objects and attributing other characteristics, such as a layer or a tag.

- *Alerts.* The alert mechanic is used to call users' attention to a user or an object. This is done first by triggering a one-time spatial sound and a repetitive yellow wave pattern, both originating from the object or user being alerted. Also, when this origin is out of view, a purple arrow appears in the screen pointing at it.
- *Creation of a preview.* A preview is a copy of an object, used to suggest an alternative to the original object. It is presented in grayscale coloring with a certain degree of transparency.
- *Acceptance/Rejection of a preview.* The act of accepting or rejecting a preview can happen either in the original object or its preview, resulting in the rejected object being deleted and the accepted one remaining in the scene. If the accepted object was the preview, it inherits the material and other graphical qualities present in the original one.

To test whether this interactions with VR worked and features could be used interchangeably, it was necessary to use the prototype developed by Vasco Pereira in his thesis, as it consisted on a VR component for the PAINTER project. This shared context made it easier test the system integration and experience this mixed reality experience.

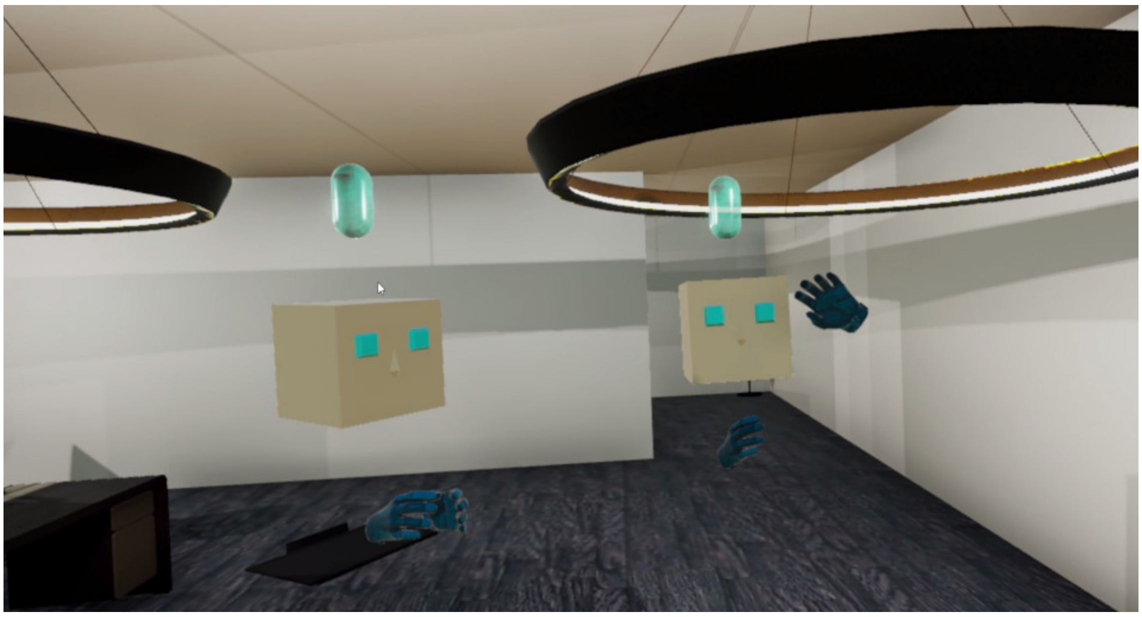


Figure 4.3: 2 VR users in the living room.

Informal tests were conducted with up to 7 simultaneous users, using the devices specific for each technology (see Figure 4.1 for AR and Figure 4.2 for VR). Several settings were experimented with different quantities of VR and AR users in the scene. Specifically for the 7-user experiment, 3 AR users and 4 VR were present in the scene at the same time. These informal testers found the system to be sturdy enough to handle seven users at a time, and noticed different visual representations for VR (a VR representation can be seen in Figure 4.3) and AR users (further explored in Section 4.6.3), as the AR ones resembled phones and the VR ones resembled heads (see Figures 4.4 and 4.5). Tests were conducted during approximately 30min, also placing stress in the devices. With seven users, sometimes a slight delay was experienced, though some users did not even notice it. Smartphones were also warm by the end of the tests due to the high screen lighting and the running application, though users did not feel too uncomfortable with the experienced temperatures.

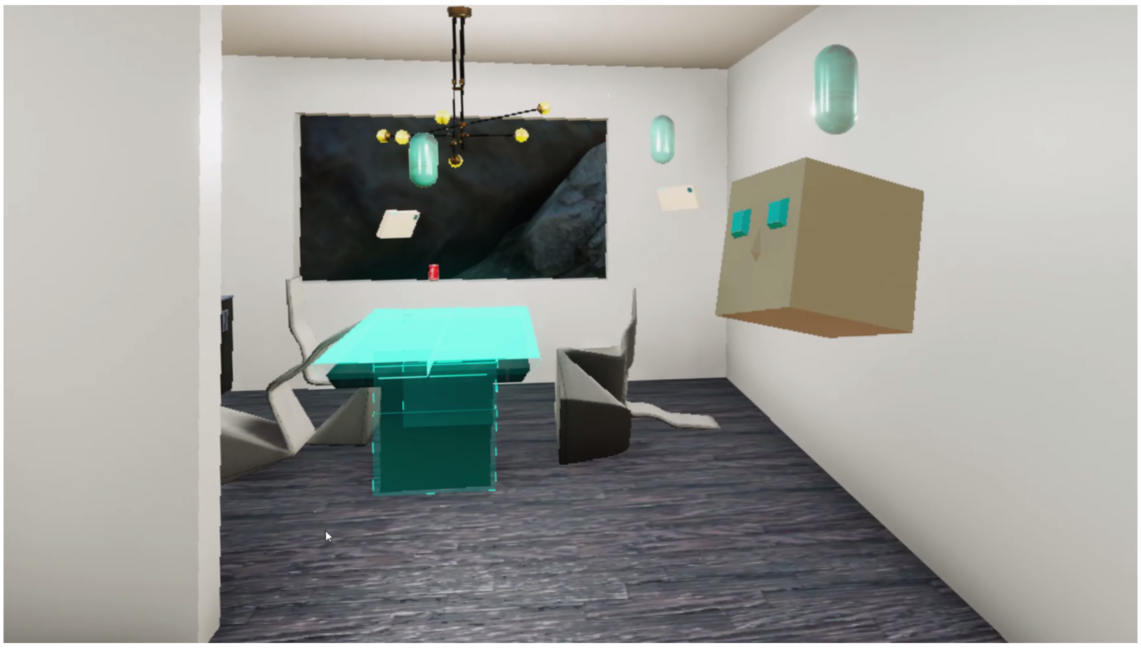


Figure 4.4: 2 AR users and 1 VR user in the kitchen.

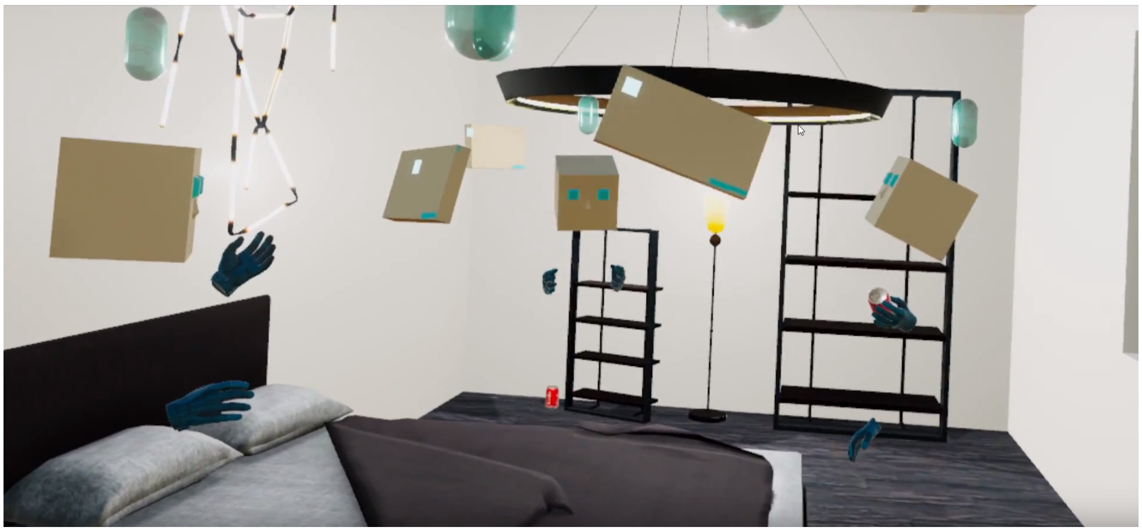


Figure 4.5: 3 AR users and 3 VR users in the bedroom.

4.3 User Interface

The interface is what enables the user to communicate and interact with and through the system, thus being one of the core parts of this tool. Besides receiving user inputs and using them to trigger actions in the system, an interface also needs to be able to convey information back to the user in a way easy to understand. However, one must not forget that the device chosen for this implementation has limited screen space, and so cautions need to be taken so that this condition hinders the user's experience as little as possible.

4.3.1 Screen space

Starting with the topic of screen space saving, even though a screen free of clutter is desired most of the times, the said clutter is still needed at some specific times. More complex interactions still need to present menus, options or buttons, which tend to occupy screen space.

The real challenge is to understand what should be displayed on the screen and when. An uncluttered interface might look better, but may lack functionality. This way, the screen should provide options relevant for the context of the application in a given moment, and hide such options when they are not useful. This provides a certain balance between an uncluttered and a helpful interface.

Buttons

On the left side of the screen, as can be seen in Figure 4.6, rest some buttons. Even though only four of these buttons were essential, it was the addition of a fifth button that actually solved the screen space problem.

However, first it must be understood why these buttons are essential:

- The second button allows the user to change between *Interaction Modes* (explored in Section 4.3.2);
- the third button is used to enter or leave *Arrow Mode* (explored in Section 4.6.1);
- the fourth button enables a user to alert all other users to itself (explored in Section 4.2);
- the fifth and final button simply exits the Photon room and then the application.

All the actions triggered by these buttons are not related to any object which already exists in the scene and/or can be selected, but instead are related to the user itself. As such, they would have to be placed somewhere the user could easily reach. Also, as it was previously referred in Section 3.3.1, in order to maintain the two-handed stance as much as possible so the user feels comfortable, inputs should be drawn to the sides so the user may reach for them with the thumb while maintaining the hands' position.

Finally, after understanding that these four buttons are necessary and cannot be hidden somewhere else, the addition of another button (the first one, from top to bottom) allows to hide or show

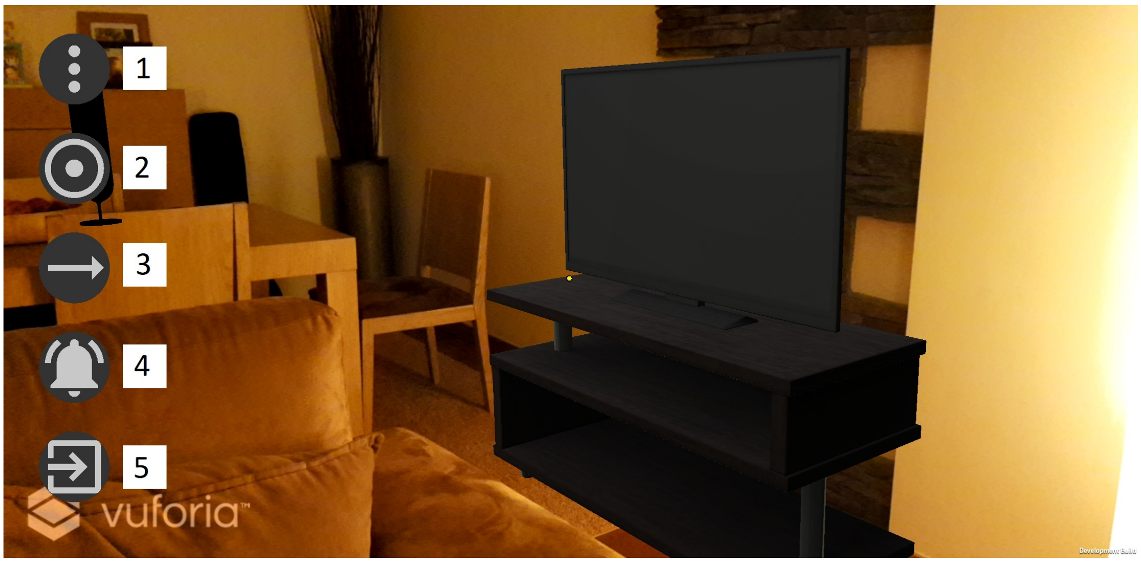


Figure 4.6: Looking at the TV stand with an open buttons' menu. Buttons' functionalities are: (1) collapse other buttons, (2) toggle interaction mode, (3) toggle arrow mode, (4) alert other users to this user, (5) exit the application.

the other buttons. This is not a novelty, being already used in some applications. In Figure 4.7, we can see that this results in a cleaner screen with less buttons, allowing the user more free space to the left side of the screen whenever they want it.

Selected-object's menu

When selecting an object, the user should have access to a menu with the manipulation actions displayed for that object, available on the right side of the screen. This menu was put on the right edge of the screen to be of easy access by the right hand, but without occupying too much screen space when it is visible. Even though the time this menu spends on screen is close to nothing when compared to the buttons on the left side of the screen, the screen space problem still exists and the menu's width was tinkered carefully so as not be too wide or remain on screen more time than it needs. As such, it only appears on the screen when a user successfully selects an object, disappearing as soon the user selects one of its options. An example of what this menu can hold is shown in Figure 4.8. This menu was made to hold both short and long actions (see Section 4.4). In the example provided, the options shown are translated as: "Move", "Rotate", "Scale", "Straighten" and "Alert". Although the options presented in the menu on Figure 4.8 match the use cases considered for this framework, the menu was designed to be dynamic and adapt to new options that may be required for future work.

4.3.2 Inputs

The interface is not only about one sees, but also about what type of inputs can be given back to the system and how it handles such feedback or inputs. This section shows different ways to

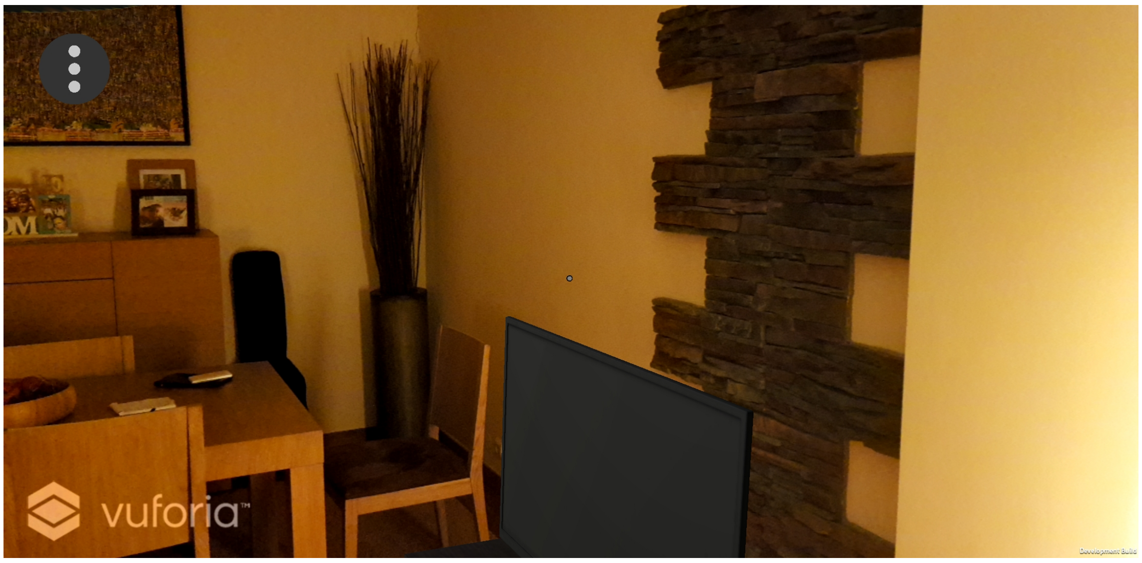


Figure 4.7: Looking away from the TV stand with an hidden buttons' menu.

interact with the application, as well as explaining how interactions were implemented and how object's behaviors to inputs can be changed.

As stated in Section 4.1, LeanTouch was used to abstract some input logic. This Unity package, amongst many other capabilities, provides C# Delegates for every input event needed in this tool, from the more simple ones such as when a finger begins/stops touching the screen, to more advanced such as detection of a gesture.

Interaction Modes

Touch can be used to click buttons and use the selected object's menu, whose details and uses were already explored in Section 4.3.1. And even though this application reads inputs for several other purposes (such as object manipulation, further explored in Section 4.4), here only object selection will be explored. Object selection is the base interaction that paves the way for object manipulation, and as such shall be one of the more used actions. With the importance of this action in mind, two different methods for selecting an object were developed, each with their own pros and cons:

- *Crosshair Mode*. This is the default mode, easily identified by a crosshair (in this case, a small dot) present in the middle of the screen. In order to select an object, users merely need to point the crosshair at the desired object and click anywhere on the screen. The result of this interaction should select the object. This mode has yet another feature, common to find in applications and games, that acts as visual feedback to the user's action: the crosshair's color changes if it is over an object that can be selectable, indicating to the user they can interact with it. If over a selectable object, the crosshair's color turns yellow (see Figure 4.6), otherwise it will be gray (see Figure 4.7). This feature is quite helpful when first exploring a

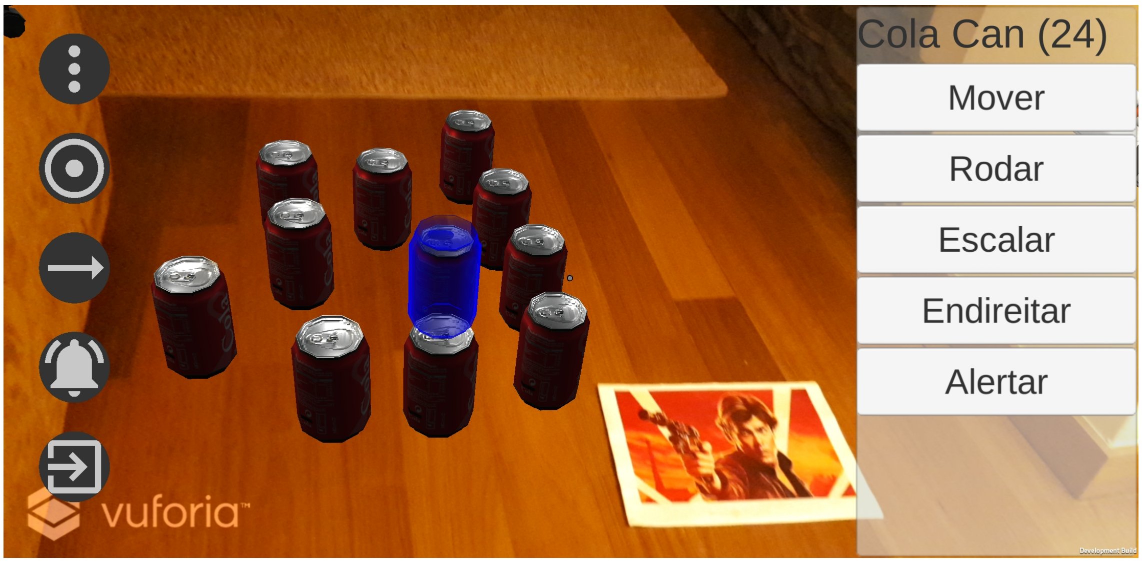


Figure 4.8: A selected Cola can, with the menu for manipulation options on the right.

3D scene, as it allows users to better understand which objects are interactive and which are not. Besides, this mode should be quite comfortable as it does not force the user to change from the two-handed stance.

- *Touch Mode*. By opposition, this mode is easily identified by the absence of the crosshair. To select an object in this mode, the user just needs to tap the object's location on the screen. This mode has the advantage of a seemingly more natural interaction. However, as shown during informal tests along the development. It also may not provide an interaction as accurate as the crosshair, since the user will not be able to see so clearly where they are clicking and since they will often need to change the hand-stance, causing some imbalance.

Interaction Mechanics

Custom functions for this framework were developed to handle button actions, as Unity's default behavior regarding buttons had to be deactivated so UI GameObjects could be triggered from this tool's logic. The reason for such a change is because UI GameObjects will, at some point, exist in the 3D scene and outside the 2D screen UI. Such an example is a text annotation. Even though it was not implemented for the purpose of this dissertation, future developments might include placing UI objects within the 3D scene, with which users may interact differently from screen UI objects it might be implemented someday, and the system is already ready for it. The reason why this is a problem is better explained with an example: Consider the example of a user who is using Crosshair Mode. The user point the crosshair at Object A and clicks on a screen location that coincides with that of Object B. On a normal situation, the touch location would not matter and the only observable behavior would be that of Object A's selection. However, if Object B is a UI object (such as a button or a text), than the touch will also trigger it, since it is the default behavior. This is a problem because the user wanted to trigger only Object A, but Object B was also triggered without the user wanting so. For this reason, the default behavior could not be allowed. The solution found for this problem was to handle UI touches alongside object touches.

Given this restriction, whenever a touch is detected, the first verification to happen is whether the Selection Point, which is the crosshair when in Crosshair Mode and the finger's position when in Touch Mode, is over a UI GameObject belonging to the Screen UI (not the 3D scene). If such a condition verifies as true, the UI GameObject is then triggered; otherwise, if the Selection Point is over a selectable object, then the object should be set as selected (and all that it implies).

All the input detection logic is centered around an *InteractionManager* Singleton script, which afterwards redirects it to objects or screen UI's elements, thus not triggering any changes by itself. This *InteractionManager* triggers (by using Unity's *GameObject.SendMessage()*) actions in the clicked object, in case handlers for such actions exist. The default handler created in this dissertation, *ARDefaultInteractionHandler*, can be added to objects for a default behavior, but other handlers can also be created for customized behavior. Available actions transmitted by the *InteractionManager* can include putting/lifting a finger on/from the screen, short tap, double short tap, etc. Other gesture or click patterns can also added to the system by extending the logic in *InteractionManager*. This responsibility delegation allows for objects with custom behavior when needed without changes on the *InteractionManager*.

However, it is not enough for an object to have handlers to receive inputs. An object is actually detected as selectable if it is in a layer named *Selectable*. Yet, this layer just signals the *InteractionManager* that the object is selectable, but without a script to receive its calls, the inputs do not have effect.

4.4 Object Manipulation

This section takes over from Section 4.3.1, where the selected object's menu was explained. Here, the options available in the said menu will be explored either by what they have in common and by what they differ from one another. All these available actions can be set aside in two categories: short actions, which are actions with a single and immediate effect, automatically deselecting the object right after; and long actions, represent actions that may take some time and many inputs from the user, delegating the responsibility of deselecting to the user.

4.4.1 Long Actions

All long actions are comprised of an indeterminate number of inputs from the user to conclude the action on the object. In the cases where touch gestures are used, users are allowed to make them as many times as they want before finishing the operation. All the available long actions are presented below:

- *Translate*. This operation allows the user to move the selected object to a new location. The translation is made using the gaze of the user, as the selected object follows the center of the

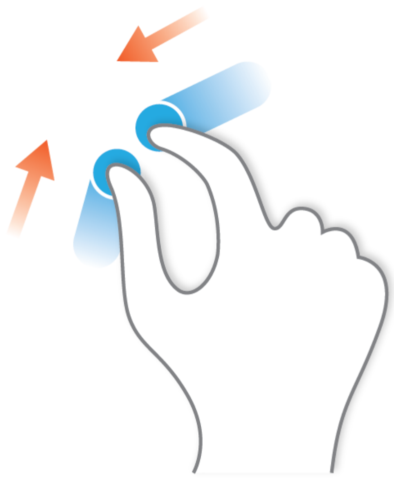


Figure 4.9: A pinch gesture, used for the scaling action. (Source: Wikimedia Commons [1])

screen's position. So, the user needs to point the screen to the location where they want the object to move to. The object maintains the orientation and scale while also maintaining its altitude during translation.

- *Scale*. This operation allows the user to change the scale of the selected object. The scaling is made by using a *pinch gesture* (see Figure 4.9), which is commonly used in multitouch devices for zooming in or out when seeing a picture or navigating a website. When the two fingers get closer, the object gets smaller; and when they get further apart, the object gets bigger.

- *Rotate*. This operation allows the user to rotate to object around its y axis (yaw). The rotation of the selected object is done by using a *rotate gesture* (see Figure 4.10), which works by using two fingers and making them rotate around an invisible point between both of them. By changing their position around the imaginary point that always exists between them, the object is rotated.

All these operations manipulate the object in a different way. However, the designed way to finish the manipulation and deselect the object is still the same, so these operations feel consistent with each other. Single and double tap were both experimented with in informal tests for deselecting objects. By the end of these experiments, the chosen solution was a double tap, which proved to be better when comparing to the single tap, as it would be more easily confused with the other interactions available in the manipulation operations.

[1]Pinch Gesture, https://commons.wikimedia.org/wiki/File:Gestures_Pinch.png

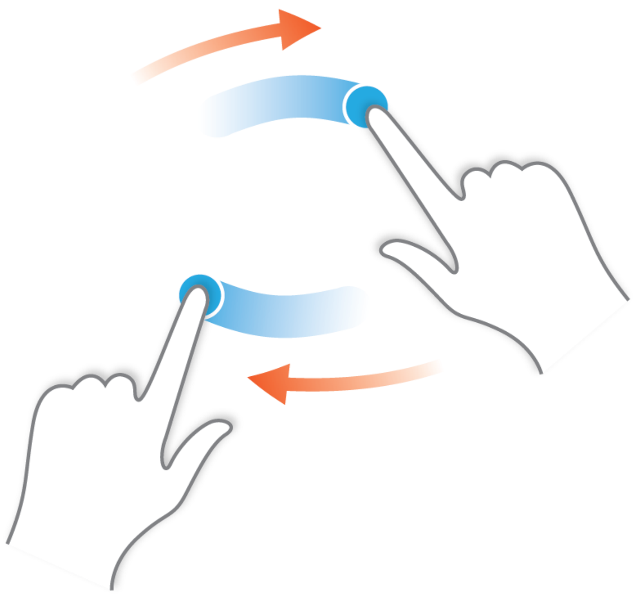


Figure 4.10: A two-handed rotation gesture, used for the rotating action. (Source: Wikimedia Commons[2])

4.4.2 Short Actions

These actions are inherently simpler than the long ones, as they require no further input after selecting an option from the menu. A short summary of what they are and other relevant information are presented down below:

- *Straighten*. This operation allows a user to completely reset an object's orientation to the Unity's default one, which is zero on all three axis. The goal of this action is to complement that of *Rotate*'s action. Even though more axis were originally considered for rotation, informal tests revealed that yaw (Unity's y axis) is the most frequent rotation axis, and the other axis' absence was more often noted when users tried to straighten objects that were tipped over. The reason for this preference might also be due to the context of the framework: interior design. In order to make up for the other axis, this option was created as a way to automate what would, most of the times, be the purpose of rotations in pitch and roll.

4.5 Markers

In this tool, AR markers are used in a similar way to the one referred in Section 2.2.1.1 as multi-markers. By using multiple markers, it was already established that the tracking space is greatly

[2]Two-Handed Rotation Gesture, https://commons.wikimedia.org/wiki/File:Gestures_Two_Hand_Rotate.png

enlarged, since these many markers work as a single big one. This works by mapping all the markers to a single object, though each of them is mapped to a different location of the object. However, tracking errors accumulate with time, which also means that the application may sometimes loose tracking after a certain time without looking at a marker or after a certain amount of movement. By using these markers on the ground, the problem of losing tracking is no longer so important, as the user does not need to go back to the marker where they started to regain or correct tracking, but rather to the one closest their zone of interest.

Another interesting application of this technique is to allow a "fake scaling" of the virtual scene to fit the real world. Consider the example of a 3D scene with $10m^2$, where markers are 2m away from the nearest walls, as seen in Figure 4.11. If these markers were placed in a $10m^2$ room with a similar setup as described, users would see a perfect fit between the real room and the 3D scene. However, the physical environment may not always have the same dimensions as the virtual one. For the purpose of the example, consider the physical room is only $8m^2$. By placing the markers closer together we can create an illusion that the virtual space fits the real one, "scaling" the scene horizontally. In this room, if the markers were placed 2m away from the nearest walls in a similar layout as the original, this effect could be achieved. A visual representation can be seen in Figure 4.12. Of course, for this to work, there must be indeed many markers placed around the space, and whenever users approach one, they need to look at it in order to keep their perception of the bounds of the 3D scene as deceived as possible. Yet, this illusion does not come without disadvantages, as it comes at the cost of users' virtual and real position seeming slightly mismatched, as the distance between two markers in the virtual and real environment may be different.

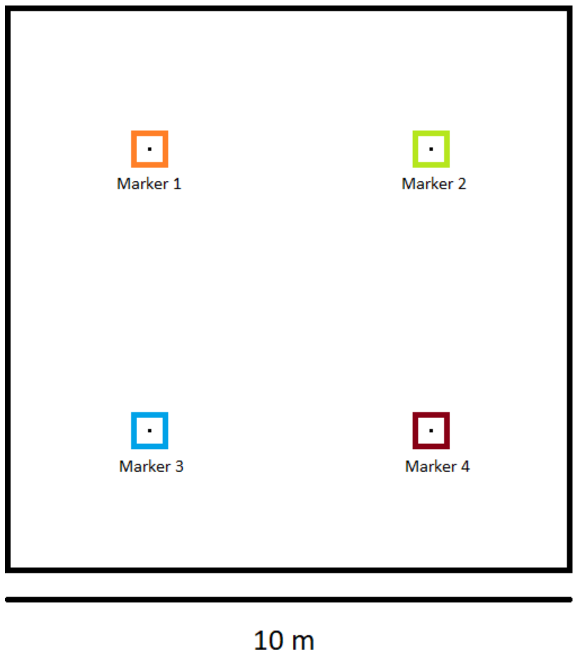


Figure 4.11: Marker setup for a 10x10m 3D scene, to be used in a real space with the same dimensions.

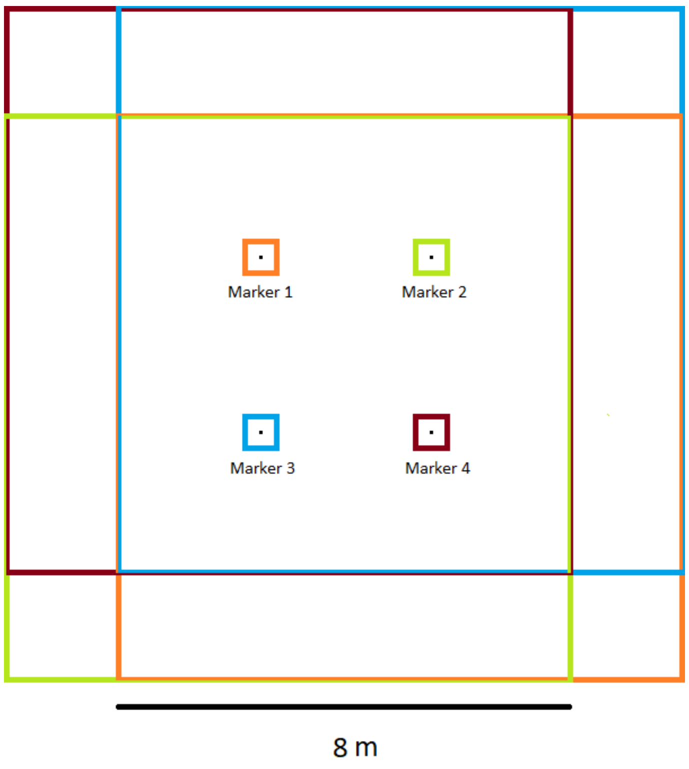


Figure 4.12: Marker setup for a 10x10m 3D scene, to be used in a real space with 8x8m.

But not even multimarkers can prevent the smartphone from losing its tracking. And even though multimarkers mitigate this problem by making more markers available to regain tracking, the user will still feel disoriented when faced with this inevitable situation. To minimize the discomfort that comes with this inevitability, an Android Toast was used to display an informative message that should help users understand what is happening. According to the Toast overview from an Android guide for developers, a "toast provides simple feedback about an operation in a small popup"[3]. As such, whenever tracking is lost or (re)gained, a Toast appears with a message informing about what is happening regarding the tracking.

Besides providing feedback about changes in tracking, there is still another feature aimed at keeping the user focused on what matters. As markers can sometimes be fickle due to many factors such as lighting conditions or camera quality, sometimes drastic changes of perception in the marker's orientation or position when the camera shakes can lead to unexpected side effects. These side effects include objects falling off to the infinity, tumbling down or other unwanted movements. Sometimes just from changing the focus from one marker to another, such events could happen. In order to restrain these sudden movements triggered by the markers, a simple script was added to each marker that would, on every Unity's FixedUpdate, reset its position and orientation to the original and intended one.

[3]Toast Overview - Android Developers, https://developer.android.com/guide/topics/ui/notifiers/toasts

4.6 Non-verbal Cues and Visual Aids

As seen in Section 2.3, the use of non-verbal cues carries great potential regarding enhancements of communication. More than that, these cues can become appropriate complements to the natural means of communication used by people. But this also means that they must be properly integrated with the remaining features so they feel as natural as possible, while requiring minimal to no explanation about what they mean, or else they may lose their purpose and hinder the rest of the experience provided by this tool. Some visual aids were also developed aiming at this natural environment, which should also make the environment more believable and mitigate feelings of discomfort towards it.

One of the first steps was to define a standard color for 3D objects used as 3D visual cues, as this should lead users to relate the color to a temporary object with which they usually will not need/want to interact. A color close to a greenish blue with some transparency was chosen, as it is not a common color to be used most of the times. This was decided based on the premise that these cues are secondary and serve merely as complements, they have a temporary role and are not supposed to be real objects like the rest of the scene, and so they should not be fully opaque and possibly hide other objects. Besides, the transparency also lets the user get some perception about what is behind the cue, withdrawing some importance from the said cue. Such examples of these are Touch Cues and Arrows, which will be further explored in the rest of this section.

4.6.1 Arrow Annotations

Regarding the annotations originally planned for this tool, arrows were implemented given they are a versatile cue that can be used in different contexts. These arrows' goal was to indicate something with accuracy, either a specific part of an object or maybe even a spot on the ground. The most versatile aspect about this annotation is that it indicates a specific point in space which should be noted by intervening users.

About the implementation itself, this was designed as an Arrow Mode, activated by pressing button "3" on the menu to the left, identified in Figure 4.6 . The button is a toggle, and so when in Arrow Mode, normal interactions are disabled to favor only the creation of arrows. With this mode active, whenever a user touches the screen, an arrow will appear. This arrow will point at the spot in the object (or ground) at which the Selection Point itself is pointing (for further information about the Selection Point, see the part about Interaction Mechanics in Section 4.3.2). The arrow orientation follows that of the screen, *i.e.* if the user has the phone tilted 90^{o} to the left, then the arrow will appear with a rotation of 90^{o} to the left, from the viewpoint of the user creating the annotation. An example is shown in Figure 4.13, where two arrows were placed with different orientations resorting to the method explained previously. It should also be noted that, while in Arrow Mode, the button appears darker in order to indicate that this mode is selected, so users have some feedback regarding the current mode.

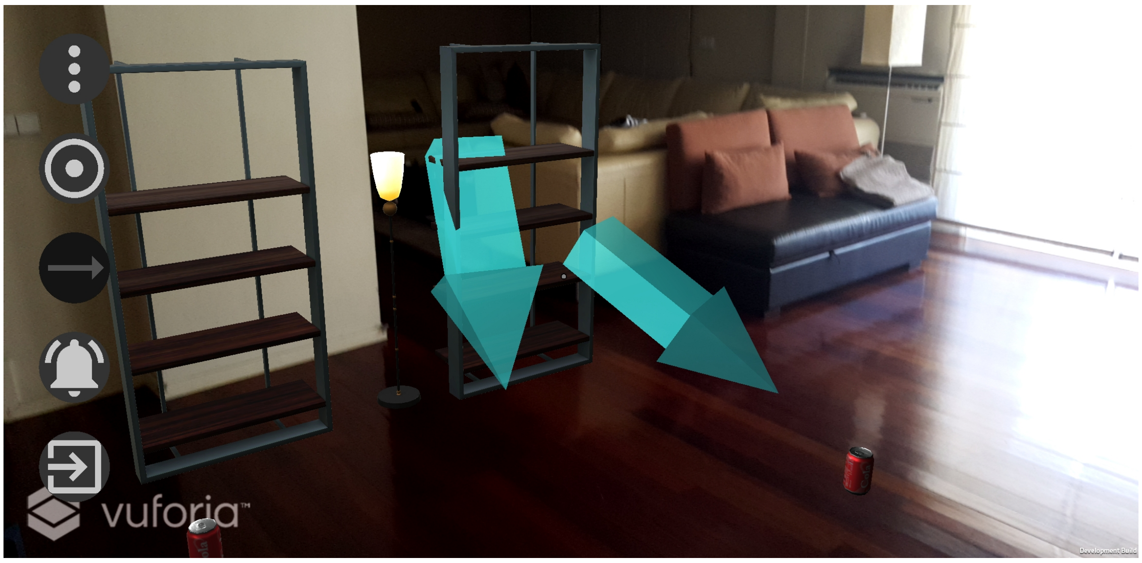


Figure 4.13: Two arrows placed with Arrow Mode. From left to right: arrow pointing at the floor, arrow pointing at a can.

4.6.2 Object Selection Cues

Some cues related to object selection were also developed. Since object selection is one of the most common events in this application, some focus and importance should be given to this mechanic.

- **Touch Cues**. These cues are triggered when the user starts touching the screen. They are used as a way to give the user some feedback about his touch. There are two cues triggered at such a moment: a Touch Sphere and an Object Color Change. The Touch Sphere is a sphere instantiated at the Selection Point (see Section 4.3.2), intended to let the user know the exact location in the 3D scene they touched. The Touch Sphere disappears 500 milliseconds after the instantiation. The Object Color Change only happens if the Selection Point hits an object, producing the following effect:

  ```
  Inputs: (h,s,v)
  s += s < 0.6f ? 0.3f : -0.2f
  v += v < 0.6f ? 0.3f : -0.3f
  ```

  The goal of this change is to let the user know which object was hit, if any at all. This effect disappears as soon as the user stops touching the screen.

- **Selection Overlay**. A Selection Overlay indicates an object is being used by an AR user. It consists of an overlay of the selected object, which has the same mesh as the object, though with a scale of 1.1f to really surround it. This overlay uses a shader from the Unity Asset Store, Baloon Shader [4], which turns it semitransparent with varying intensity according to

[4]Baloon Shader, Unity Asset Store, https://assetstore.unity.com/packages/vfx/shaders/balloon-shader-18102

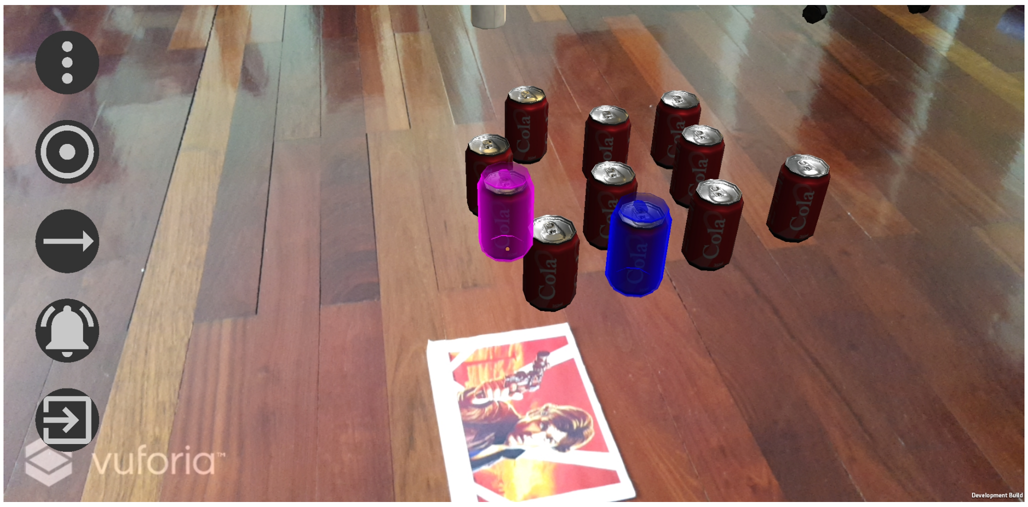


Figure 4.14: Two beverage cans selected by two different users, with magenta and blue-colored overlays.

the angle between the normal to the surface and the user's eye. The overlay also always appears in front of objects that would normally occlude the unselected object, to help users keep track of the location of the object. Each user has a color associated to them, so that when there are many selected objects from many different users everyone knows which is theirs and which are not. An object gains the overlay when it is successfully selected (which happens right after the object's ownership is transferred to the user) and loses it when the object is desselected or an action on said object is finished. An example of two overlays are shown in Figure 4.14.

- **Haptic Feedback**. This consists on using the vibration component present in smartphones to create a vibration pattern that the user feels when holding the device. This feature is triggered when an object is successfully selected. So, besides seeing an object light up with the its color, the user also literally feels the moment it happens.

- **Out-of-View Ray**. When users look away from their selected object during some time, they might forget its exact location and be forced to look around until they see the overlay. For this case and many others, this ray was added. Whenever the selected object leaves the user's field-of-view, a ray appears from the center of the screen to the object's location. This should help users keep aware of their selected object's location when looking around.

4.6.3 User Awareness Cues

However, in a collaborative environment users should not pay attention just to objects. If the context is a collaborative one, then it is of great importance for users to remain aware of each other as much as possible. And even though their physical presence in the same space greatly helps in this matter, it would not be wise to leave this weight only on the user's senses. As such, some user awareness cues were developed to increase this awareness and help users have more information about each other. This is especially helpful since users might spend a lot of time looking at the phone when using the system, and such cues presented in the screen may provide useful information. The implemented cues are displayed below:

- **Users' Virtual Representations**. This cue allows users to see virtual representation of each other's devices. These representations are displayed with a basic shape resembling what they are, displayed as smartphone-shaped boxes, and are shown in the location of their respective user and also with the orientation in which the device is. These virtual smartphones are visible to all users except the user each one is representing, so they do not clutter the screen. An example of such a representation is visible in Figures 4.15 and 4.16.
- **Users' Fields of View**. Also known as viewing frustum, the field of view encompasses the area which the user is able to see in a screen. In 3D, it can be seen as a truncated rectangular pyramid. As seen in Section 2.3.2, this frustum can be useful to increase awareness over what each user can see or not. The implementation uses dynamic mesh generation, so the mesh can really match that of each phone camera's field of view, as opposed to having a standard mesh for the purpose. The mesh possesses some transparency, as its goal is not to hide anything, but rather help users understand each other's viewing limitations. These meshes also follow the smartphone's rotation and always point to where the camera points, while also following its position so it appears in the correct location. A frustum example can be seen in Figure 4.17.

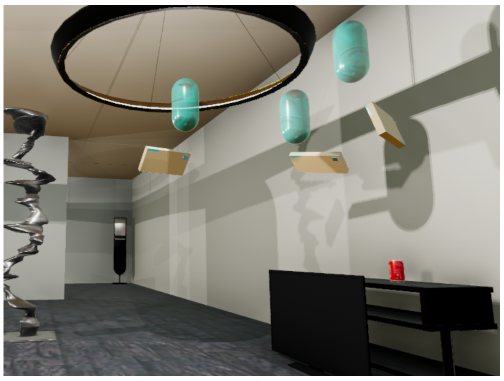


Figure 4.15: Three AR users present in the same virtual space.

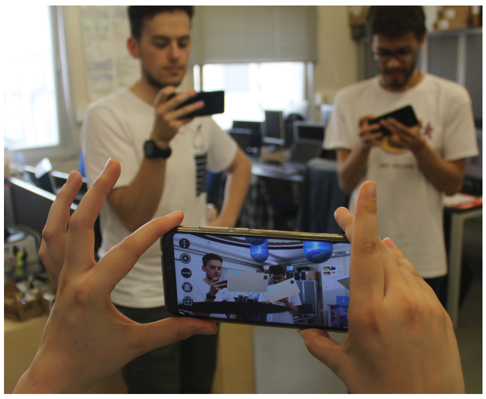


Figure 4.16: Three AR users present in the same physical space.

4.6.4 Non-Controllable Objects Visibility

When analyzing the Virtuality Continuum [MK94], a great difference can be spotted between AR and VR regarding distance to the real environment. This crucial difference is important given the

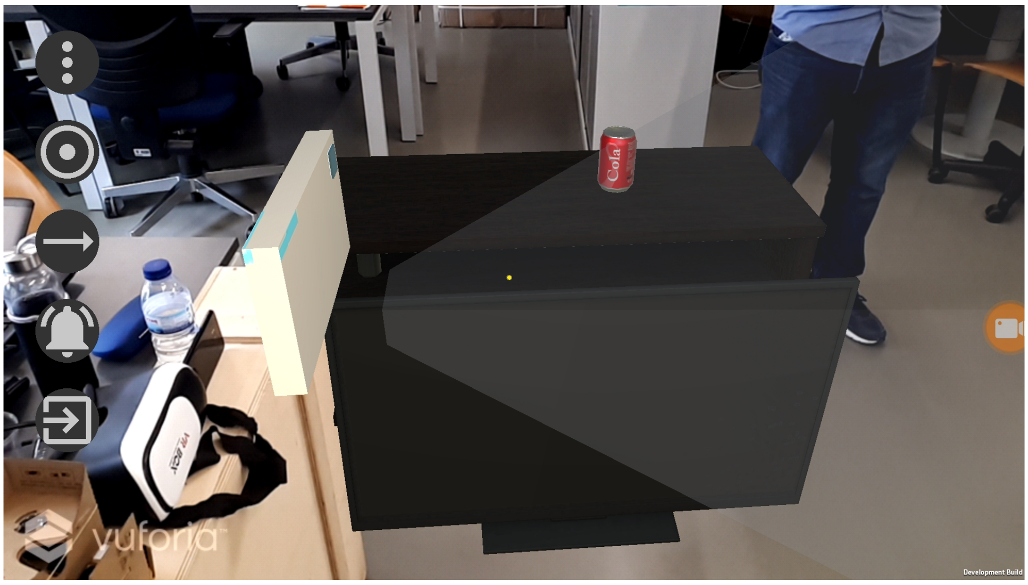


Figure 4.17: A user's field of view.

Interchangeable Reality context, as a 3D scene may be shared between AR and VR. This might be hard to reconcile because VR might, for example, want opaque walls, while AR would likely want to use the real world's walls. This divergence lead to the necessity of controlling object's visibility.

The developed mechanism for controlling visibility consists of a single script that looks for objects with certain tags to modify their opacity. Objects with the tag "AR_Invisible", turn invisible for the AR users. This is done simply by assigning them an invisible material. Such a behavior can be specially helpful for floors, ceilings and exterior walls in 3D scenes, as those can be the main objects blocking the view into the real world.

However, there may be cases where full transparency is also undesired. Interior walls, for example, can be necessary for both users to see, but the transparency is still useful for AR, so users do not loose so much track of the real world. With these edge cases in mind, semi-transparency is allowed by using tag "AR_Semitransparent". Using this tag cuts the alpha value of the material's color to 90%, and if transparency was not enabled, it is enabled. This way, the object seems mostly opaque, but some transparency exists so users can get a glimpse of what is behind.

4.6.5 Shadows

Shadows are a constant in the real world. Besides letting us know where light comes from, it also helps better perceive the location and dimension of objects relatively to each other. Such an important part of the real world should not be forgotten in the virtual world.

A simple shadow solution was devised as an alternative to Unity's, as its default shadows capabilities fall somewhat short when working with objects possessing transparency to some degree. This solution consists of an Image with a black circle, where the circle's borders have a slight gradient from opaque black to transparent. This circle is attached as a child of objects that can have shadows. The image is adapted to have the same horizontal dimensions as the parent object, so the apparent shadow has the same dimensions as the parent object.

4.7 Summary

In this chapter, an overview of the designed prototype is presented, including information from the tools and technologies used, to the many interaction mechanics implemented, while also providing a detailed explanation of why of how the feature comes to be. Informal tests are also presented alongside features whenever they happened, so decisions taken during development are better understood.

The next chapter presents the evaluation methodology used to validate this dissertation's aims and research questions, as well as details regarding the test structure and what each part of the test strives to achieve. By using the solution described in this chapter as a testing environment, some user tests will be made in order to obtain data that should help make some relevant inferences and deductions, which will be useful to effectively answer the research questions.

Chapter 5

Evaluation

The aim of the user studies is to understand the usability of the system and its influence in collaborative teamwork. This should help shed some light on the influence of such a system on collaborative teamwork and further provide answers to the research questions proposed in Section 1.2.

This chapter starts by presenting the structure for each test and for each test session, in Sections 5.1 and 5.2 respectively. Some information regarding the data gathered from 42 participants is afterwards given in Section 5.3. Section 5.4 further shows some statistical results and their analysis, which is then more heavily discussed in Section 5.5.

5.1 Evaluation Protocol

The evaluation protocol contains the tasks that users should follow while testing the system. This protocol divides these tasks in three groups, each group with a particular goal. For each group, the goal is presented, as well as the set of tasks that compose them.

The 3D scene used for these tests was specifically arranged in order to facilitate all these types of interactions. The kitchen's more open zone is mostly centred on the user's initial position, as the goal of this area is not to overwhelm the user with a large area with lots of available interactions, but instead to allow learning in a controlled environment. With this in mind, users only have to make small adjustments to complete the tasks from Group A (which happen in the kitchen), making them easier to achieve when users test the system for the first time. This room can be seen in Figure 5.1.

For the living room, the conditions are different, as this room is used for collaborative interactions from Group B tasks, which also means this room has to be more open to accommodate for two users and their simultaneous operations. As such, the space seems much wider, which also makes the tasks in Group C easier, as they involve a great amount of moving objects around.

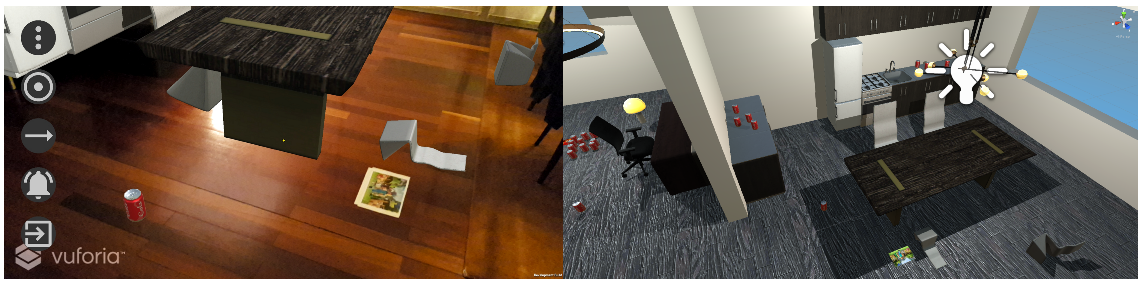


Figure 5.1: The kitchen in the real world (left) and in a virtual world (right).

For simplicity, two users are defined as generic participants for the user tests: User A and User B. These users serve as concrete examples of what each user should do in each group of tests.

- **Group of Tests A (T.A), Individual Work Tasks**. For this group of tasks, users shall tidy up the kitchen. This involves two tasks: straightening, rotating and scaling a chair; and moving a can. User A first performs those tasks and leaves the system. Then User B enters the application and performs the same tasks. The goal here is to understand how easily users interact with the system and whether they find these interactions natural. This shall be evaluated by asking users to answer some questions about the system in an usability questionnaire. Users A and B do not perform the tasks at the same time in order to allow the researcher leading the experiment to quickly clarify any doubts that may arise. Additionally, this initial separation is also due to being easier for the researcher to focus on one user at a time, allowing users to have the researcher's full attention.

- **Group of Tests B (T.B), Collaborative Work Tasks**. In the second group of tasks, users will tidy up the living room together. After finishing the previous group of tasks, and with User B still in the kitchen, User A should enter the application and move into the living room by focusing on the marker assigned to the living room. The first task these users must do is try to see each other. Whenever one user achieves this, he then alerts the other user to himself. After this, User A also moves itself to the living room. There, both users have 3 tasks they need to coordinate: move three cans; straighten a sculpture; and scale and rotate a sofa. The aim of these tasks is to make users work together and understand how this can be done with this tool.

- **Group of Tests C (T.C), Collaborative Guiding Tasks**. For this group of tasks, users shall be guided by one another in a series of 4 tasks. The first "guiding" user is chosen randomly in order to ensure that previous experiences do not influence the results from these ones. Whenever the "guiding" user is receiving instructions from the researcher, the "guided" user should not see what they are seeing. As an example, User A is chosen as the "guiding" user. For the first task, a certain beverage can is indicated to User A. User A should then explain to User B which is the beverage can and make him select it. Then User B will have the same task as User A had, but he may alert to the object before starting the task. Then User

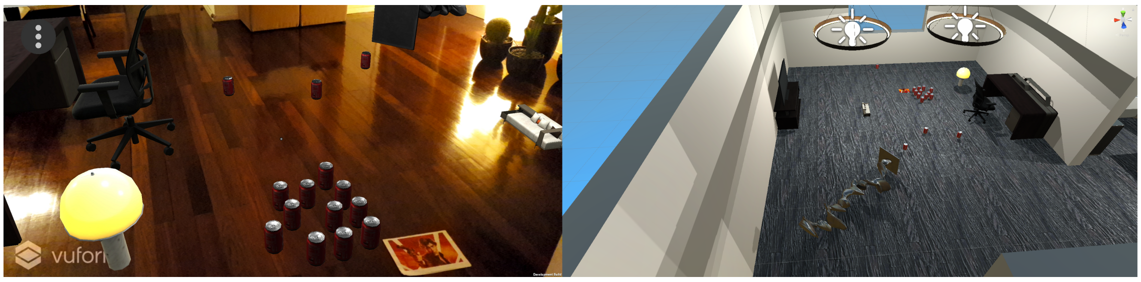


Figure 5.2: The living room in the real world (left) and in a virtual world (right).

B receives instructions to, verbally, make User A move a lamp to another place. After that, User A repeats the same task with User B, but they may use arrows and words. These tasks are used to establish a comparison between the usage of normal means of communication and AR-enhanced communication.

5.2 Test Session Structure

At the beginning of the test session, the pair of users are given a explanation regarding what this dissertation is about and further indicate the research goals underlying this experiment. Additionally, the pair of users will be informed about AR and AR collaboration so that users have a very basic understanding of the subject.

Afterwards, users are instructed to fill the demographics part of the questionnaire and also sign a consent form (see Appendix C), which states that they are aware of the context of this work and know that any information retrieved during the user tests will be treated anonymously and are for the sole purpose of this dissertation. Users are then explained the case study, where both users are requested to tidy and arrange the 3D scene for a virtual commercial filming, according to indications received. Users are also made aware that they may ask for help at any time and should be comfortable doing so.

One user shall then accompany the researcher to start the experiment. The application is opened in one of the mobile devices to explain to that user how the it works, followed by a brief demonstration. The phone is then handed to the user so he completes the tasks from Group T.A. Once the first user is finished with the tasks, the other user will undergo the same process. When the second user has finished Group T.A, the other user joins and both should complete Group T.B and Group T.C. When the experiment finishes, both users are asked to return the phones used and fill the remainder of the questionnaire.

5.3 Gathered Data

Data was gathered using two different means, both of them during the user tests. Usability questionnaires were used to gather each participant's opinion regarding their experience of the system and their personal thoughts on certain features, resulting in heavily subjective data. Besides this technique, and as a way to confirm its results, some objective notes were taken by the researcher conducting the test, containing factual data about events occurring during the user tests.

5.3.1 Usability Questionnaire

It was found necessary and important to collect the users' opinions about what they experimented, as well as their thoughts about the system. As such, two questionnaires were created: one prior to the experience and another after the experience. The first of these contains demographic questions, in order to allow the characterization of the participants. After the test's completion, the participants fill out the second questionnaire, where answers to the questions are given according to a Likert scale. These post-experiment questions are divided into three sections: the first one focuses on issues related to the application controls and the direct use of objects; the second one deals with each user's perception regarding the application's features; and a third one seeks to explore the perceived usability of the system. Answers to these post-experiment questionnaire vary from "1 - Strongly disagree" to "5 - Strongly agree". Regarding the last section of the trio, a System Usability Scale (SUS) was used [Bro96]. A SUS seeks to measure, through a set of ten questions using Likert scale (from "1 - Strongly disagree" to "5 - Strongly agree"), the users' perception about the system usability.

Finally, the questionnaire has a field entitled "Final Comment", which encourages participants to leave suggestions, critics or other form of comment whose content they feel the questionnaire has not addressed. Both full questionnaires are available in Appendix B.

5.3.2 Objective Notes Taken During Tests

As it was found necessary to adopt a qualitative evaluation methodology in order to obtain more in-depth and relevant information to the study, it was also considered essential to include a quantitative methodology, focusing in particular on the times users take to complete the various tasks of the experiment. With this in mind, it is important to mention the topics that have been taken into account in this methodology. The first aspect that required special attention on the researcher's part is that of perceiving if the participants unknowingly discovered the method for deselecting an object or not. This is done by watching for users trying to deselect an object during the experiment. If this does not happen until the end of T.B, whenever users select their indicated objects in T.C, they are asked to deselect it afterwards, providing the chance to see how they try to deselect. Other notes that the researcher found worth keeping for the investigation were also kept, though these had no defined structure. In addition, and as previously mentioned, it is also of great importance to keep track of several task completion times. These times' goal to provide objective data when interpreting the results of the questionnaire, as they can be combined with the subjective data obtained in order to verify it. More information on these notes taken during the course of the experiment can be seen in the table 5.1.

Binary Notes	D.1	Did the user understand how to deselect an object?
Descriptive Notes	D.2	Notes the researcher finds worth keeping for better results' interpretation.
Time Measurements	D.3	Individual tidying of the kitchen
	D.4	Discover other user's graphic representation
	D.5	Discover other user's graphic representation with help from the alert tool
	D.6	Cooperative tidying of the living room
	D.7	Select the indicated can
	D.8	Select the indicated can with help from the alert tool
	D.9	Move lamp to the indicated place
	D.10	Move lamp to the indicated place with help from Arrow Mode

Table 5.1: Gathered data grouped by types of data.

5.4 User Tests' Results and Analysis

After finishing all the user tests, the data from the performed user tests needs to be treated and analyzed. This section first presents the population that participated in the user tests, followed by the obtained results divided according to different areas of interest. These results provide some insight into the success and usability of the system's features and mechanics, among other useful indicators. For a complementary statistical analysis, see Appendix A.

5.4.1 Population Overview

The population which performed these user tests was invited to participate in this experiment by direct contact or via social networks, with a total of 42 people (21 pairs of people) attending these tests. The participants ages range widely from 14 to 42, although about half of the population is mostly concentrated between 19 (value for the first quartile) and 23 (value for the third quartile), with a median age of 22. These age ranges can be seen in Table 5.2.

Regarding gender, this population contains 18 female participants (43%) and 24 male participants (46%). As for the frequency of AR usage, most users said they had some experience with Augmented Reality (64%), while only a few said they never used AR (17%) and some affirming they use it many times a month (19%).

Age Range	<=16	17-19	20-22	23-25	26-28	29-31	>=32
People	1	11	10	14	1	2	3

Table 5.2: Age distribution for the population of users.

5.4.2 Interface Controls and Object Manipulation

The data presented in this subsection corresponds to the questions in the first part of the post-experiment questionnaire, thus evaluating the control interface and the object manipulation features present in the system. Regarding the Groups of Tasks presented in Section 5.1, these questions are closely linked to the individual experience of T.A, but also somewhat related to T.B, as this last one also deals heavily with object manipulation.

The first topic that should be brought to the fore is the intuitiveness of the controls, as it can provide a general overview of the entire application. As seen from statement S.1 in Table 5.3, with a median value of 5, most people found the controls highly intuitive.

Statement	**Q1**	**Median**	**Q3**	**Mode**
S.1 "Controls were intuitive"	4	5	5	5
S.2 "Selection method was useful"	5	5	5	5
S.3 "Selecting objects was easy"	4	5	5	5
S.4 "Deselecting objects was easy"	4	5	5	5
S.5 "Changing to another room was not an issue"	4	4	5	5
S.6 "Moving objects was easy"	4	5	5	5
S.7 "Scaling objects was easy"	5	5	5	5
S.8 "Rotating objects was easy"	4	5	5	5
S.9 "Straightening objects was easy"	5	5	5	5

Table 5.3: Likert-scale results for the questionnaire section about controls and object manipulation.

The next topic is about the most used interaction mechanic of the application: selecting an object. The results show no major problems with this selection method (see Interaction Modes in Section 4.3.2) which uses a crosshair as the Selection Point, with over 75% of the population heavily defining this method (see S.2 from Table 5.3) as very useful, which can also be confirmed by looking at Figure 5.3. However, these results are not completely matched with those related to S.3 in Figure 5.4. This means that some users think this selection method is not as easy as it is useful, even though the general results still point to over 50% users choosing to agree strongly that this selection method is easy to use. The reason for this discrepancy is clearer after seeing some of the comments at the end of the questionnaire. Many users reported that they might find it easier to select an object by pressing directly on it, instead of using the crosshair.

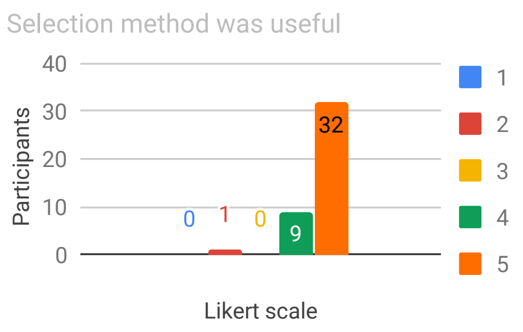


Figure 5.3: Likert-scale results for the statement “Selection method was useful” (1- Strongly Disagree 5 Strongly Agree).

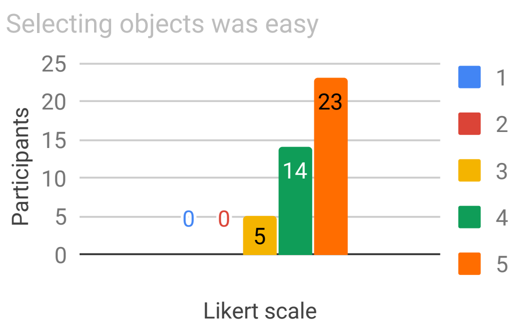


Figure 5.4: Likert-scale results for the statement "Selecting objects was easy” (1- Strongly Disagree 5 Strongly Agree).

Regarding deselecting an object, the users were not taught how to do it. Instead, the researcher conducting the experiment observed what was their instinctive response, in order to understand if the implemented deselection method (when an object was selected, but no operation had been triggered yet) had been correctly predicted as the most instinctive. As seen from Figure 5.5, 71.1% of users tried to deselect with the predicted method, confirming that the prediction was correct. 23.7% tried to deselect with a double tap, which might have happened since the method used to stop a long action (actions comprised of an indeterminate number of inputs, as seen in Section 4.4.1) was also a double tap. The remaining 5.2% asked how an object could be deselected, so it is unclear which they would have found most instinctive. These results match with those of the questionnaire, where users were asked if deselecting objects was easy. According to S.4 from Table 5.3, between 50% and 75% strongly agreed that it was easy, approximating the 71.1% obtained from the users' instinctive response.

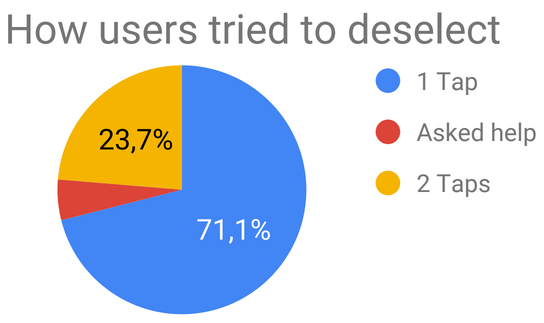


Figure 5.5: Users' instinctive response to deselecting.

The data regarding the mechanics for changing between rooms presented a different trend compared to previous mechanics. The median seen in Table 5.3 for S.5 is unique in this subsection, as even though it is still rather high at 4 and the results in the graph do not show values below 3, this is an area where there might be room for improvement. A possible reason for this may have to do with the differences between the real and virtual scale (see Section 4.5 for more details), and users may have felt confused by this difference. During the experiments, users found it easier to accept this inconsistency after its cause was explained to them, as they no longer appeared so confused after the explanation.

Regarding manipulations, users seem to have found no particular difficulties when scaling or straightening an object, as can be seen by the first quartile value of 5 in both statements S.7 and S.9 from Table 5.3, meaning that at least 75% of users did not have a problem with those two operations. However, rotating and moving seems to have been a more difficult action than scaling or straightening, as can be seen by observing S.6 and S.8, or even by directly comparing the Q1 value of both pairs. Regarding this issue, a user commented he wished he could also use his finger to drag the object on the screen. Even though only one user wrote a comment regarding this subject, there was more that one user who tried to drag the object with a dragging gesture, which might provide some insight on why users found moving an object harder than scaling or straightening. The other slightly harder interaction was rotating, which one user justified by stating that an on-screen watermark indication with the gesture could be useful. This means that the rotation gesture might not be intuitive enough or might just require further indication, as said user implied.

5.4.3 Visualization of the Virtual Environment

This subsection presents the questions in the second part of the post-experiment questionnaire, whose goal was to evaluate how the user perceived the environment around him and the results of collaborative interactions. This data is related to both to T.B and T.C.

Regarding perceptions, and similarly to the previous subsection, one of the first aspects that should be addressed is related to object selection, that is, how easy it is for users to identify selected objects. From the data presented in statements S.10 and S.11 from Table 5.4, users seem to have found this rather easy, although some discrepancy exists between perceiving the objects they have selected and objects selected by others. This might happen due to simple circumstances, such as each user knowing which selection is theirs because they are the ones triggering the action. There may also be another explanation for this, and might be related to each user's color. There were times when both users had, as their personal color, two different types of blue (cyan and regular blue). This might lead to more difficulty in perceiving other users' selections as the color may be close to the user's own color, making it more difficult to notice and differentiate.

Statement	**Q1**	**Median**	**Q3**	**Mode**
S.10 "Perceiving the other user's selected object was easy"	4	5	5	5
S.11 "Perceiving my selected object was easy"	5	5	5	5
S.12 "An alert was quickly perceived"	4	5	5	5
S.13 "Arrows were useful"	4	5	5	5
S.14 "Seeing other users was useful"	3.25	4	5	5
S.15 "Perceiving the location of another user in another room was easy"	3	4	4	4

Table 5.4: Likert-scale results for the questionnaire section about the visualization of the virtual environment.

Specifically regarding alerts, it seems that their goal was accomplished with some success. By viewing statement S.12, it is clear that the majority of users perceives these alerts quite quickly, as intended.

As for arrows, clearly more than 75% of users find them quite useful (as seen from the first quartile value associated with statement S.13, where the value 4 corresponds to an agreement with the usefulness premise), although a few may still disagree. Placing arrows in the right place is a sensitive operation, prone to errors for inexperienced or trembling users. Users further stated that sometimes it is hard to perceive where the arrow is pointing at. As a possible solution for this last problem, a user suggested replacing the arrow with a circle, which could better indicate the

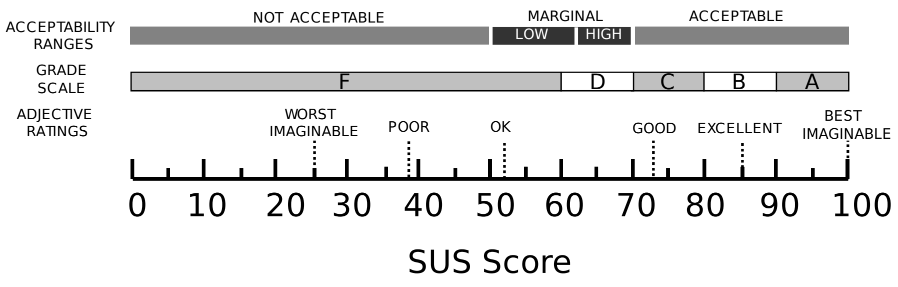


Figure 5.6: SUS rating scales, according to [BKM09a].

position. Besides the arrow pointing problem already described, one user also wished these arrows were smaller, presumably because they might have occluded some part of the environment.

Users were also asked whether they found it useful to see other users. Though the Mode presented in statement S.14 from Table 5.4 points to the highest value of agreement, indicating quite a few users see it as highly useful, the Q1 value tells us that there are also other users that disagree on different levels. A simple design for visual representation was created, but user answers seem to indicate that it might be too simple and/or might need some further improvement. The reason for these results may also have to do with the fact that the tasks were not very dependent on seeing users in the virtual world, since users shared the same physical space.

As for perceiving other users' locations when in different rooms, the results associated with statement S.15 from Table 5.4 are not as good when comparing to most others in this subsection, which means that this feature is likely to have room for improvement. Though at least 50% users agreed on several levels that it was easy to perceive the location of another user in another room, the number of users below Q1 and Q3 (which can also be seen in statement S.15) might indicate that this mechanic could be better, as Q1 is on the neutral position of the scale and Q3 did not even reach 5. The worse results might indicate two possible issues: the inability to plainly see other users in other rooms, or implementation-related problems about usage of the alert utility to call users' attention to another user (this last issue was noted when some users stated that the arrow pointing to the alert's origin was not always properly following the object).

5.4.4 System Usability

In this subsection, the results from the SUS are shown and analyzed. The SUS [Bro96] is composed by a group of ten questions, with answers following a Likert scale varying from "1 - Strongly disagree" to "5 - Strongly agree". It is a proven industry standard [BKM09a] with the goal to evaluate a system's usability.

The values from these answers are then used to calculate a SUS Score, ranging from 0 to 100. Bangor, Kortum and Miller [BKM09a] propose acceptability scores, school grading scales and adjective ranks are proposed so it is easier to better understand the result of the SUS Score. These rating scales can be further seen in Figure 5.6.

The individual results for each question can be seen in Table 5.5. However, as Brooke [Bro96] notes, these individual scores are not meaningful on their own. Instead, the main focus should be on the SUS Score. This score was calculated for each answer using the provided instructions. Afterwards, the average of these scores was calculated, with a final result of approximately 85.3. Comparing this score with Figure 5.6, the system is found as "Acceptable" in the "Acceptability Ranges" and in the "Adjective Ratings" it stands very close to "Excellent" (85.5).

SUS Question	**Median**	**Mode**
I think that I would like to use this system frequently.	4	5
I found the system unnecessarily complex.	1.5	1
I thought the system was easy to use.	5	5
I think that I would need the support of a technical person to be able to use this system.	1.5	1
I found the various functions in this system were well integrated.	5	5
I thought there was too much inconsistency in this system.	1	1
I would imagine that most people would learn to use this system very quickly.	5	5
I found the system very cumbersome to use.	1	1
I felt very confident using the system.	5	5
I needed to learn a lot of things before I could get going with this system.	1.5	1

Table 5.5: Statistics regarding individual SUS questions.

5.4.5 Time Measurements

The time measurements complement the subjectivity of the user questionnaire with objective data. These measurements were made during the tests, mostly counting the time taken for each task to be completed. The referred tasks and some statistics from the time measurements is presented in Table 5.6.

The first two presented measurements in Table 5.6, D.3 and D.4, are related to object manipulation, though in very distinct contexts. The first one refers to the tutorial task of tidying up the kitchen, which consists on scaling and rotating a chair and moving a beverage can. The second one is about the cooperative task of tidying up the living room, including straightening a sculpture, moving three beverage cans and rotating and scaling a sofa.

The next two presented measurements, D.5 and D.6, are related to discovering another user's graphical representation when users are in different rooms. This is a task that may prove difficult because users may have a semitransparent wall between them. The third and fourth entries from Table 5.6 show the results for the task without and with the help of the alert tool, respectively. By comparing both results' means, 12.56s without the alert and 10.15s with the alert, it can be seen that, in average, it is faster to notice another user when receiving help from the alert tool.

Both the first two tasks from T.C have the same goal: one user needs to help the other user identify what beverage they are referring to. This task was first performed using only verbal

Task	Mean	Std. Dev.
D.3 Individual tidying of the kitchen.	47.93s	13.85s
D.4 Cooperative tidying of the living room.	44.19s	12.31s
D.5 Discover other user's graphic representation when in a different room.	12.56s	7.99s
D.6 Discover other user's graphic representation when in a different room with help from the alert tool.	10.15s	7.11s
D.7 Select the indicated can.	31.82s	19.50s
D.8 Select the indicated can with help from the alert tool.	9.97s	8.32s
D.9 Move the lamp to the indicated place.	27.64s	17.28s
D.10 Move the lamp to the indicated place with help from Arrow Mode.	20.84s	18.48s

Table 5.6: Time measurements' statistics for tasks performed during user tests.

indications, D., and then with verbal indications and the use of the alert tool. Results for both these tasks are presented in the 5^{th} and 6^{th} rows, respectively. By comparing results from both experiments, the alert tool's usage is seen to lead to a lower mean, while also showing a lower standard deviation. This seems to indicate that using the alert tool help users find an object quicker than without the tool's help.

Finally, the last two tasks from T.C are related to one user indicating a specific virtual location, to where the second user should move a lamp. The two modalities for these lamp-related tasks are somewhat similar to the beverage can's tasks, as in the first task the user giving indications can only use verbal ones, while in the second one the guiding user is given the ability to place arrows in the environment. Both tasks' results are presented in Table 5.6 associated to D.9 and D.10, respectively. Results show a lower mean for the alternative where arrows are allowed, which means users were quicker to move the lamp when the arrows were used to help point the desired location.

5.5 Discussion

After every piece of data obtained from the tests is analyzed, a more global vision of the test results may be achieved. Subjective and objective data, from the user tests and time measurements respectively, can now be put side by side in order to better answer the research questions.

The interaction mode used during the experiments revealed itself quite useful, according to users. However, and despite showing good results, ease of use did not follow the same trend. Many users commented about the interaction mode, usually to say that they would prefer if they could touch directly the object , instead of pointing to it with the crosshair and clicking anywhere on the screen. Some users unknowingly tried to use the application with these controls, as it was their intuitive response, even though the mode was not available for use. This wished selection method was actually implemented and called Touch Mode (see Section 4.3.2), but was not included in the user tests since it was thought that by using different modes during the tests, users would experience most features differently, specially those related to object manipulation. In Touch Mode, the finger touch needs to be the more accurate the further (and thus smaller in screen) the object is, which might have proved to be unsuitable for inexperienced users and could further lead to frustration when using the system, thus provoking a different experience from one using just the Crosshair Mode. However, with many users asking for this particular feature, it is impossible to discard the Touch Mode's utility and it should be improved so it can be used even from larger distances. Crosshair Mode could also benefit from some improvements, as one user also stated they found the crosshair to be too small. With this in mind, crosshair size could be slightly enlarged or maybe even allow its customization.

About the manipulation options, and as seen in Section 5.4.2, scaling and straightening were rather easy to use and the results seem to indicate an opportunity for further enhancement. The same cannot be said for rotating and moving. Though rotating had overall good results, some difference can be seen relatively to scaling and straightening. Just by looking at statement S.8 from Table 5.3, a Q1 value of 4 indicates that at least 25% of users did not find the mechanism as easy as it can be. This might have happened due to this operation using a gesture that is not widely used in common smartphone applications, contrary to the pinch gesture used in scaling which is also very common to other uses (*e.g.* zooming on a web page or on a photo). One user even suggested that these operations could feature watermark instructions, possibly just for the first few moments so the user has some non-intrusive indication of how the operation works. As for moving, its results follow an even relatively worse trend than scaling, although the absolute results are still good. This means once again that an opportunity for improvement exists, as users thought this feature was not as easy as it could be. One immediate reason was identified as users were participating in the tests: many tried to drag the object with their fingers and not move the phone. This does not mean that the currently implemented method is ineffective, as users still rated it highly in terms of ease of use. However, these behaviors mean that many users' instinctive reaction was to drag, and so this might have been something worth implementing. Yet another reason for the worse results regarding manipulation operations might also have to do with an unexpected behavior found during tests, when users tried to move an object. When the crosshair was pointing at the object, the object should have been "ignored" and should move to the position, in the ground, pointed by the crosshair. This issue made some movements harder than they should be, most likely contributing to a decrease in the user's answers.

The Group of Tasks C used non-verbal cues to evaluate the difference between performing the tasks with or without the platform's features. By taking a look at Table 5.6, a big difference can be seen both for selecting the indicated beverage can and for moving the lamp to the right place. In the tasks related to the beverage can (D.7 and D.8 from Table 5.6), the mean time to perform the said task is 3.19 times better when using the alert cue. Besides, the fact that the standard deviation is lower when using the alert cue also means that the task performance is more consistent, being result of a more precise indication which does not depend so much on individual ability to communicate, effectively boosting cooperation efficiency. As for the lamp-related tasks (D.9 and D.10 from Table 5.6), execution time is 1.33 times better when using arrows to help guide the movement. Though this cue clearly enhances collaboration, it seems not to be so effective as alerts, which is also backed by the results from the user questionnaire, where users found alerts more useful.

By analyzing the general results of the user tests, one can better understand the impact of AR interactions on collaborative teamwork. Results show that users can use basic interaction mechanics like short actions or even the scaling long action quite easily, though more complex ones such as rotating or moving might need further instructions to have a more positive impact on the experience. This overall positive impact is further backed by the SUS result, rating this application's usability as "Excellent". Cues that were not triggered wittingly by users' commands (*e.g.* Object Selection Cues and User Awareness Cues, as seen in Sections 4.6.2 and 4.6.3, respectively) were generally well received, contributing to making interactions easier to understand.

Chapter 6

Conclusions

Augmented Reality is a type of interface that lets users see the real world with a virtual world overlayed on top of it. An interface like this is highly capable of taking intuitive interactions to a new level, enhancing natural interactions and taking advantage of the reality humans are familiar with.

Such an interface has great synergy with many application fields, like education, tourism or collaboration. Regarding collaboration, it is capable of enhancing cooperation among several users, specially when non-verbal cues are used. These cues help users in interpreting the virtual environment as well as the real world, including actions from other users present in the system. These cues are particularly important when users are only remotely present, as users cannot make use of body language or other real-world cues that are only available when users share the same physical space.

Furthermore, taking into account the research on the previously explained topics, knowledge gaps were found regarding collaborative AR. Among many other issues, works about using AR and VR devices to experience the same 3D scene simultaneously were rather scarce. These gaps raised questions, as defined in Section 1.2, regarding AR interactions in an environment where VR users may also exist. With these questions in mind, a proof of concept system was planned as a mean to further investigate this matter, including a customizable collaborative environment, integration with VR users, interaction mechanics for objects and users, as well as non-verbal cues and other visual aids.

As one of the most common devices for AR use nowadays is the smartphone, a proof of concept was developed for Android, the most common Operating System for such devices. The system implemented most of the features proposed, with a high priority on the intuitiveness of the interface and the collaboration mechanics that would allow users to better cooperate and understand each other.

User tests were also performed in order to validate the usefulness of the proposed mechanics. Users performed three different groups of tasks with different goals: understand what users find hard to use, what collaborative mechanics help users and the difference between using the system's tools and not using them. The results for these tests are presented in Chapter 5, and show that most of the implemented interaction mechanics and aids were helpful for collaboration purposes.

Regarding these document's questions, as proposed in Section 1.2, it can be said that they were answered with rather positive results. AR was found to have a positive impact on collaboration and results showed the inherent positive impact of some of the non-verbal cues and visual aids present in the system.

6.1 Future Improvements

The implemented system was developed as a proof of concept, which also means that there are some improvements that could be potentially implemented. Finishing the unfinished features is one of the first steps for improvement, followed by analysis and enhancement of the already implemented features.

One of the details mentioned by users regarding the interaction method used during the user tests was the desire to change the crosshair size. A comment stated that, in the user's opinion, the crosshair's size should be larger. However, in the interest of making future code more flexible and provide better adaptability for users, an even better solution would be to allow the crosshair's size customization (and perhaps other crosshair-related parameters).

The remaining interface could also benefit some improvements. One particular user conveyed that he thought the screen interface could be visually pleasing. This comment might be related to the grayscale colors used in buttons and menus. The use of a Material Design[1] color palette and animations might be a good approach to this problem, as it is an interface standard which is also very familiar for most people using Android devices.

Reactions to the several object manipulation operations were not all equal. Moving and rotating an object had slightly inferior reaction from users. This might be improved by following one user's suggestion about providing watermark instructions for the rotation operation, and applying such suggestion for each long action. These watermark instructions may be static, though animated ones might provide better results since it can show users the necessary gestures happening. Watermarks might stay during the whole operation's time if they are static and translucent enough to not distract the user from the operation itself, thought probably a better idea would be to show these watermarks with more relevance when a user starts a long action and make it disappear soon after.

Specifically regarding the operation where objects can be moved, it might be wise to provide an alternative method where users may drag the object through the scene, with no influence from the phone's position. Since many users tried to drag the object even after being explained how the mechanic worked, this seems to hint that this alternative should be provided and users should be able to choose which they prefer. However, the original moving method can also be further enhanced, either to fix the issue of not detecting the ground accurately or by blending its ideas with

[1]Material Design, https://material.io/design/

the drag alternative proposed above. Such a hybrid might use the method already implemented whenever the user was not touching the screen, but when a user touched the object (and while they kept touching) it would work only with the dragging gesture.

Regarding arrows, a user also commented that it would be easier to understand where an arrow is pointing at if a circle was placed in the pointed place. Further improvements on this idea might come from replacing this imagined circle by a sphere, as 3D shapes are might be easier to perceive and understand. Another user also stated that he would have preferred if arrows were smaller. For this issue, the suggested solution is the same as for the crosshair's similar issue: allow users to customize size (and perhaps more parameters) according to their liking.

During the informal tests performed alongside development, the frustum that represents a user's field of view was thought to be too obtrusive, specially as the number of AR users grew in the scene and more frustums appeared. A possible solution for this problem would consist on hiding the frustums during most of the time. When a user looked at another one's graphic representation, he would see its frustum. Whenever he looked from the user to anywhere else, the frustum would remain visible for a short amount of time to allow the user to understand where the other was looking at. However, when a user looks at another user's graphics to see its frustum, only that user would see it, as other users would probably not have interest and, above all, this could trigger the exact problem this solution tries to avoid: too many active frustums obstructing the scene.

6.2 Future Work

The developed solution for this dissertation sets a starting point for many new collaborative AR interaction mechanics. However, even outside the scope of this dissertation or the PAINTER project, there are some issues worthy of further study.

One such issue includes the study of virtual movement while using an AR device. As stated in Section 4.5, markers may be used to create a "fake scaling" by simply positioning them closer or further from each other. This line of thought can be further expanded to allow something similar to teleportation across a vast space. So, if a user wanted to move to a place far away, he might just remove the marker currently used as anchor from view and place a new one in view, which is associated with a far away place in the virtual world. It would be interesting to study to what extent this object can be applied in other AR-related works and projects and seem both useful, intuitive and easy to use.

It would also be interesting to further study different mechanics for moving objects in AR. In the previous section two other mechanics were suggested besides the implemented one: using only drag gestures and a hybrid approach merging drag gestures and the implemented method. The addition of other alternative mechanics to this study could further enrich its findings.

The differences between Touch Mode and Crosshair Mode is also another matter worth investigating. It would be relevant to understand the advantages and disadvantages of both these interaction modes, studying parameters such as accuracy, speed and users' personal preferences for each task.

Watermarks, as suggested in one user's comment, is yet another subject that can be further studied. These cues can be used to help the user understand what gestures should be used at certain times. However, there are many ways to show them: static or animated, staying during the whole interaction period or disappearing after a certain time, opaque or transparent. Different combinations of these characteristics are possible and it might be worth investigating which combinations work best.

References

[AZDA15] Luis Arenas, Telmo Zarraonandia, Paloma Díaz, and Ignacio Aedo. A Platform for Supporting the Development of Mixed Reality Environments for Educational Games. In *Learning and Collaboration Technologies*, pages 537–548. Springer, Cham, 2015.

[BGSD09] Mark Billinghurst, Raphaël Grasset, Hartmut Seichter, and Andreas Dünser. Towards Ambient Augmented Reality with Tangible Interfaces. In *Human-Computer Interaction. Ambient, Ubiquitous and Intelligent Interaction*, pages 387–396. Springer, Berlin, Heidelberg, 2009.

[BK02] Mark Billinghurst and Hirokazu Kato. Collaborative Augmented Reality. *Communications of the ACM*, 45(7):64–70, jul 2002.

[BKM09a] Aaron Bangor, Philip Kortum, and James Miller. Determining what individual sus scores mean: Adding an adjective rating scale. *J. Usability Studies*, 4(3):114–123, May 2009.

[BKM09b] Mark Billinghurst, Hirokazu Kato, and Seiko Myojin. Advanced Interaction Techniques for Augmented Reality Applications. In Randall Shumaker, editor, *Virtual and Mixed Reality*, pages 13–22. Springer, Berlin, Heidelberg, 2009.

[BKP02] Mark Billinghurst, Hirokazu Kato, and Ivan Poupyrev. Collaboration with Tangible Augmented Reality Interfaces, 2002.

[Bro96] John Brooke. Sus - a quick and dirty usability scale. *Usability evaluation in industry*, 1996.

[BSEN12] Tobias Blum, Ralf Stauder, Ekkehard Euler, and Nassir Navab. Superman-like X-ray vision: Towards brain-computer interfaces for medical augmented reality. In *2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 271–272. IEEE, nov 2012.

[CPB18] Julien Casarin, Nicolas Pacqueriaud, and Dominique Bechmann. UMI3D: A Unity3D Toolbox to Support CSCW Systems Properties in Generic 3D User Interfaces. *Proceedings of the ACM on Human-Computer Interaction*, 2(29):1–20, nov 2018.

[DB06] H. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping: part i. *IEEE Robotics Automation Magazine*, 13(2):99–110, June 2006.

[FF16] A. S. Fernandes and S. K. Feiner. Combating vr sickness through subtle dynamic field-of-view modification. In *2016 IEEE Symposium on 3D User Interfaces (3DUI)*, pages 201–210, March 2016.

[FGV+00] Mike Fraser, Tony Glover, Ivan Vaghi, Steve Benford, Chris Greenhalgh, Jon Hindmarsh, and Christian Heath. Revealing the realities of collaborative virtual reality. In *Proceedings of the third international conference on Collaborative virtual environments - CVE '00*, pages 29–37, New York, New York, USA, 2000. ACM Press.

[Gru94] J. Grudin. Computer-supported cooperative work: history and focus. *Computer*, 27(5):19–26, May 1994.

[HHPC13] Zhanpeng Huang, Pan Hui, Christoph Peylo, and Dimitris Chatzopoulos. Mobile augmented reality survey: a bottom-up approach. *Computing Research Repository*, sep 2013.

[HKBA19] Weidong Huang, Seungwon Kim, Mark Billinghurst, and Leila Alem. Sharing hand gesture and sketch cues in remote collaboration. *Journal of Visual Communication and Image Representation*, 58:428–438, jan 2019.

[HKI+12] Otmar Hilliges, David Kim, Shahram Izadi, Malte Weiss, and Andrew Wilson. HoloDesk: Direct 3D Interactions with a Situated See-Through Display. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems - CHI '12*, page 2421, New York, New York, USA, 2012. ACM Press.

[KAS10] Otto Korkalo, Miika Aittala, and Sanni Siltanen. Light-weight marker hiding for augmented reality. In *2010 IEEE International Symposium on Mixed and Augmented Reality*, pages 247–248. IEEE, oct 2010.

[KBB+18] Kangsoo Kim, Mark Billinghurst, Gerd Bruder, Henry Been-Lirn Duh, and Gregory F. Welch. Revisiting Trends in Augmented Reality Research: A Review of the 2nd Decade of ISMAR (2008–2017). *IEEE Transactions on Visualization and Computer Graphics*, 24(11):2947–2962, nov 2018.

[Low04] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[Mai17] João Pedro Maia Miranda. Interactive Collaboration Platform in Augmented Reality. Master's thesis, FEUP, 2017.

[MK94] Paul Milgram and Fumio Kishino. A Taxonomy of Mixed Reality Visual Displays. *IEICE Transactions on Information Systems*, E77-D(12):1321–1329, 1994.

[NTLY17] Minh Nguyen, Huy Tran, Huy Le, and Wei Qi Yan. A tile based colour picture with hidden QR code for augmented reality and beyond. In *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology*, number 8 in VRST '17, pages 8:1—-8:4, New York, New York, USA, 2017. ACM Press.

[PDE+17] Thammathip Piumsomboon, Arindam Day, Barrett Ens, Youngho Lee, Gun Lee, and Mark Billinghurst. Exploring enhancements for remote mixed reality collaboration. In *SIGGRAPH Asia 2017 Mobile Graphics & Interactive Applications on - SA '17*, pages 1–5, New York, New York, USA, 2017. ACM Press.

[PGR+18] Seonwook Park, Christoph Gebhardt, Roman Rädle, Anna Maria Feit, Hana Vrzakova, Niraj Ramesh Dayama, Hui-Shyong Yeo, Clemens Nylandsted Klokmose, Aaron Quigley, Antti Oulasvirta, and Otmar Hilliges. Adam: Adapting multi-user interfaces for collaborative environments in real-time. *CoRR*, abs/1803.01166, 2018.

[PHPK19] Wonjun Park, Hayoung Heo, Seongjun Park, and Jinmo Kim. A study on the presence of immersive user interface in collaborative virtual environments application. *Symmetry*, 11(4), 2019.

[SD06] Helmut Seibert and P Dähne. *System architecture of a mixed reality framework*, volume 3. INSTICC-INST SYST TECHNOLOGIES INFORMATION CONTROL & COMMUNICATION, Setubal, 2006.

[SSDP$^+$08] Beatriz Sousa Santos, Paulo Dias, Angela Pimentel, Jan-Willem Baggerman, Carlos Ferreira, Samuel Silva, and Joaquim Madeira. Head-mounted display versus desktop for 3d navigation in virtual reality: a user study. *Multimedia Tools and Applications*, 41(1):161, Aug 2008.

[SSFG98] Z. Szalavári, D. Schmalstieg, A. Fuhrmann, and M. Gervautz. "Studierstube": An environment for collaboration in augmented reality. *Virtual Reality*, 3(1):37–48, mar 1998.

[SvdH16] Hanna Schraffenberger and Edwin van der Heide. Multimodal Augmented Reality: The Norm Rather Than the Exception. In *Proceedings of the 2016 Workshop on Multimodal Virtual and Augmented Reality*, pages 1–6, New York, New York, USA, 2016. ACM Press.

[TPP18] Santawat Thanyadit, Parinya Punpongsanon, and Ting-Chuen Pong. Efficient Information Sharing Techniques between Workers of Heterogeneous Tasks in 3D CVE. *Proceedings of the ACM on Human-Computer Interaction*, 2(CSCW):1–19, nov 2018.

[WSV$^+$17] Lauren Westendorf, Orit Shaer, Petra Varsanyi, Hidde van der Meulen, and Andrew L. Kun. Understanding Collaborative Decision Making Around a Large-Scale Interactive Tabletop. *Proceedings of the ACM on Human-Computer Interaction*, 1(CSCW):1–21, dec 2017.

[ZCGTS$^+$17] Ulises Zaldivar-Colado, Samir Garbaya, Paul Tamayo-Serrano, Xiomara Zaldivar-Colado, and Pierre Blazevic. A Mixed Reality for Virtual Assembly. In *2017 26th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 739–744. IEEE, aug 2017.

REFERENCES

Appendix A

Questionnaire Statistics

Pre-Experiment: Demographic Questions

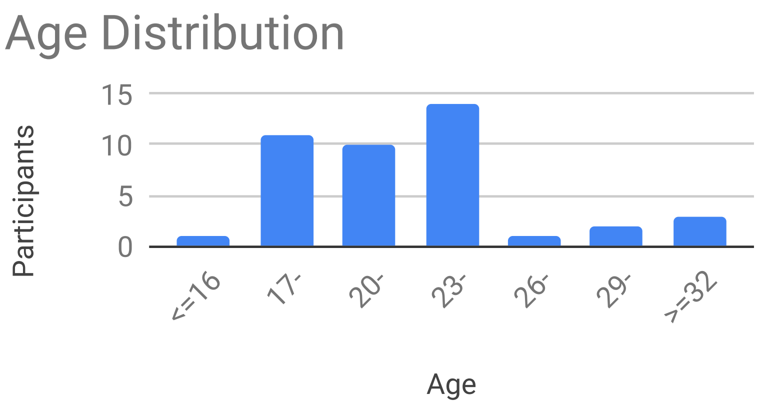


Figure A.1: Results distribution regarding users' ages.

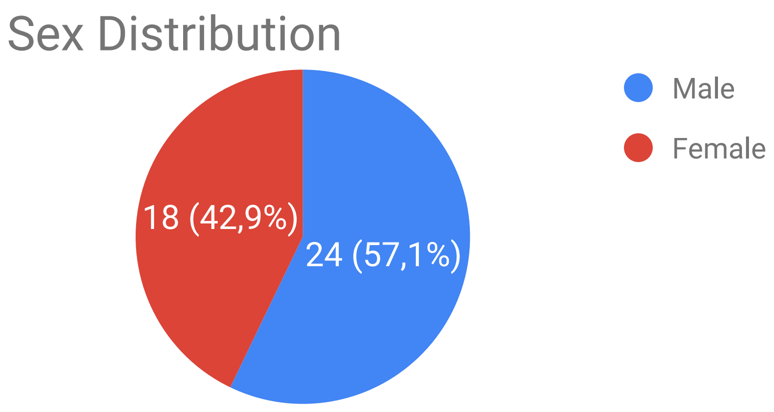


Figure A.2: Results distribution regarding users' sex.

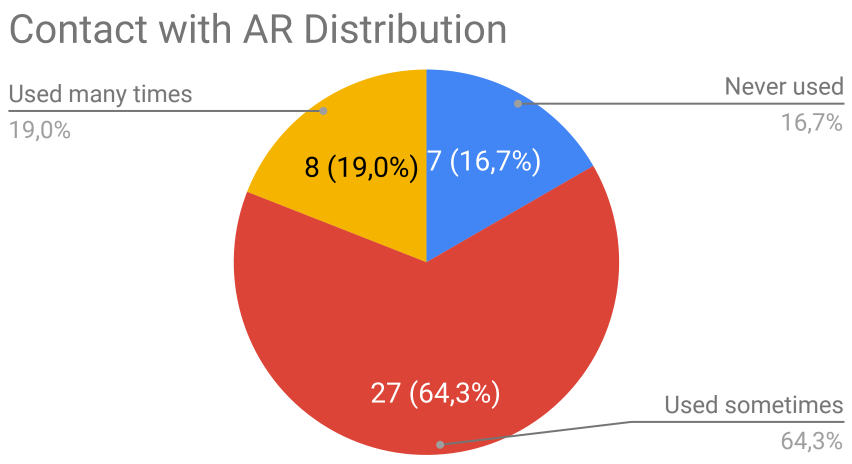


Figure A.3: Results distribution regarding users' contact with Augmented Reality.

Post-Experiment: Controls-Related Questions

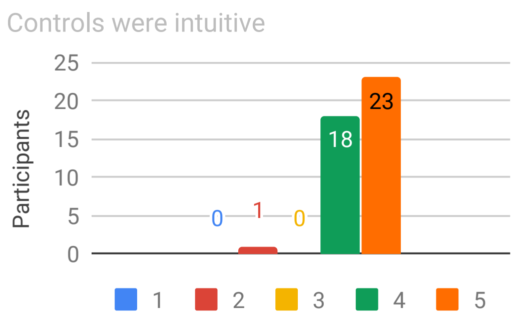


Figure A.4: Likert-scale results for the statement "Controls were intuitive" (1- Strongly Disagree 5 Strongly Agree).

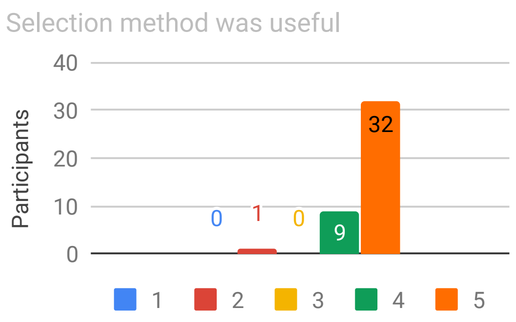


Figure A.5: Likert-scale results for the statement "Selection method was useful" (1- Strongly Disagree 5 Strongly Agree).

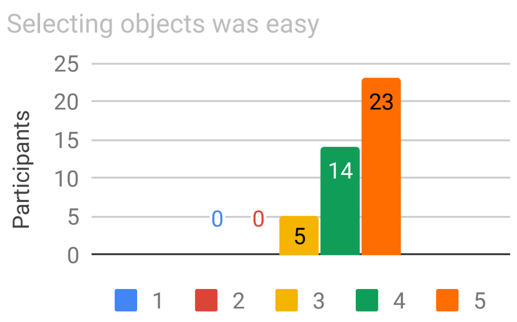


Figure A.6: Likert-scale results for the statement "Selecting objects was easy" (1- Strongly Disagree 5 Strongly Agree).

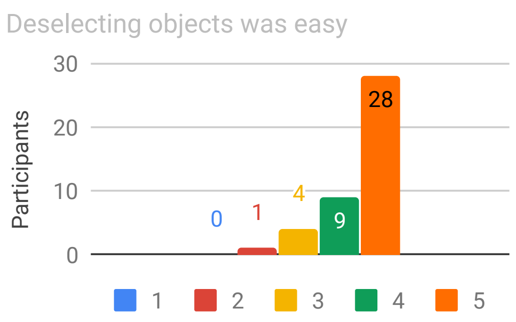


Figure A.7: Likert-scale results for the statement "Deselecting objects was easy" (1- Strongly Disagree 5 Strongly Agree).

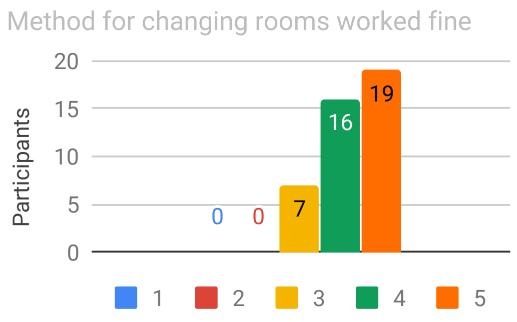


Figure A.8: Likert-scale results for the statement "Method for changing rooms worked fine" (1- Strongly Disagree 5 Strongly Agree).

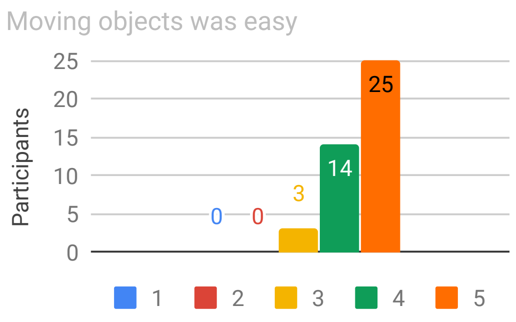


Figure A.9: Likert-scale results for the statement "Moving objects was easy" (1- Strongly Disagree 5 Strongly Agree).

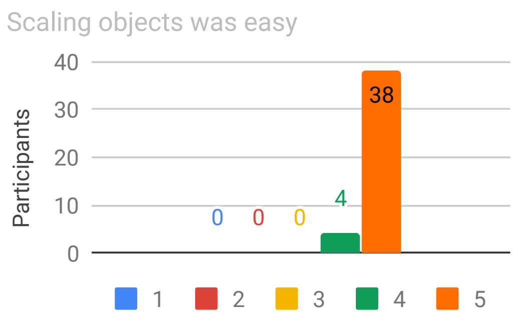


Figure A.10: Likert-scale results for the statement "Scaling objects was easy" (1- Strongly Disagree 5 Strongly Agree).

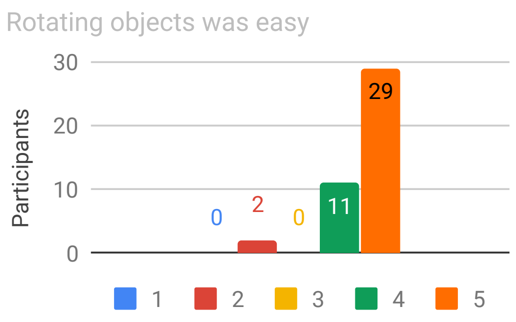


Figure A.11: Likert-scale results for the statement "Rotating objects was easy" (1- Strongly Disagree 5 Strongly Agree).

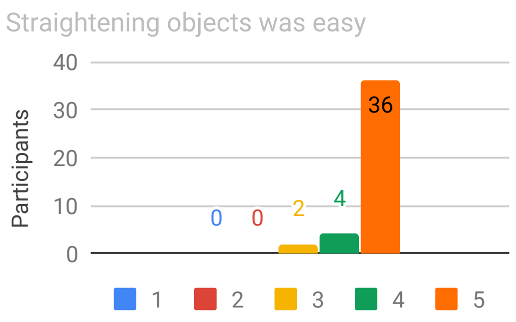


Figure A.12: Straightening objects was easy

Post-Experiment: Perceptions-Related Questions

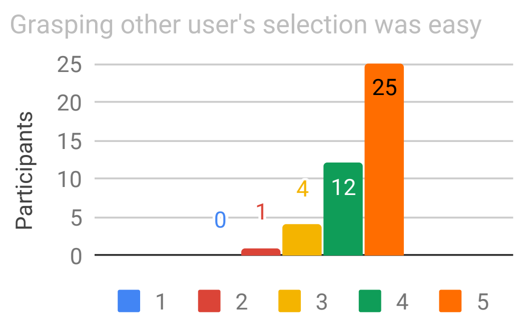


Figure A.13: Likert-scale results for the statement “Grasping other user’s selection was easy” (1- Strongly Disagree 5 Strongly Agree).

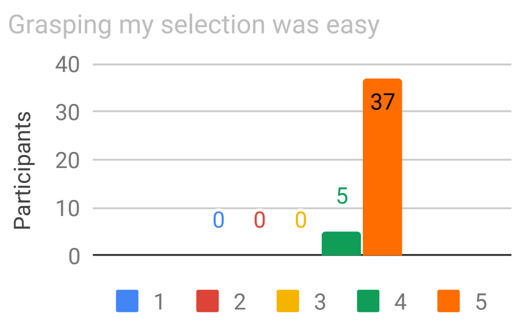


Figure A.14: Likert-scale results for the statement “Grasping my selection was easy” (1- Strongly Disagree 5 Strongly Agree).

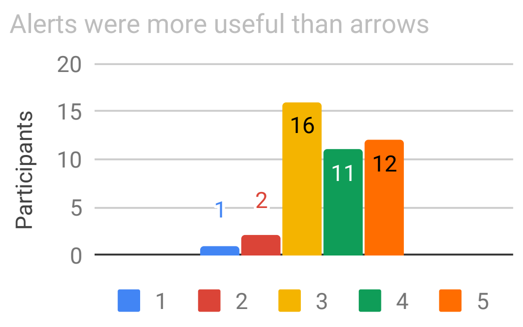


Figure A.15: Likert-scale results for the statement “Alerts were more useful than arrows” (1- Strongly Disagree 5 Strongly Agree).

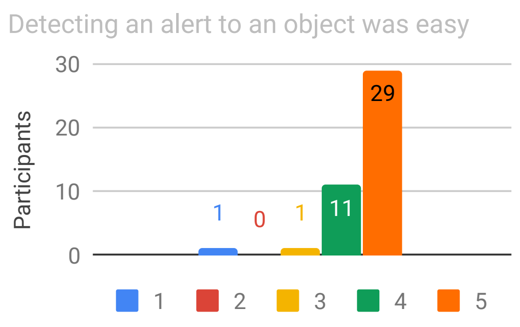


Figure A.16: Likert-scale results for the statement “Detecting an alert to an object was easy” (1- Strongly Disagree 5 Strongly Agree).

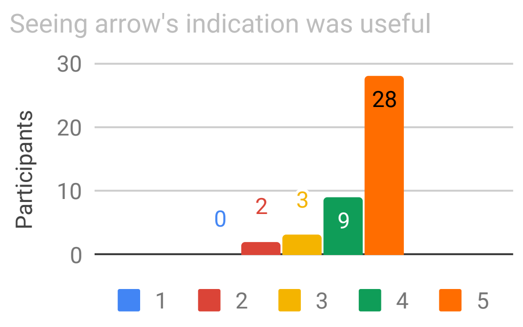


Figure A.17: Likert-scale results for the statement “Seeing arrow’s indication was useful” (1- Strongly Disagree 5 Strongly Agree).

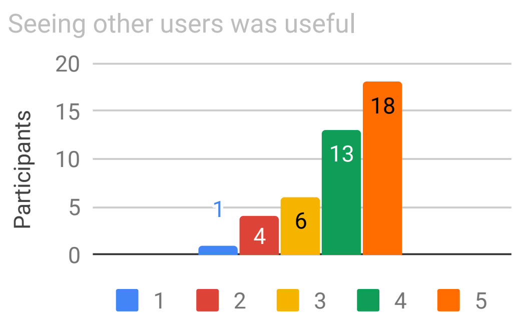


Figure A.18: Likert-scale results for the statement “Seeing other users was useful” (1- Strongly Disagree 5 Strongly Agree).

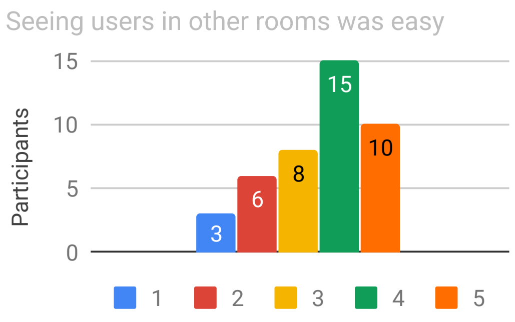


Figure A.19: Seeing users in other rooms was easy

Post-Experiment: SUS Questions

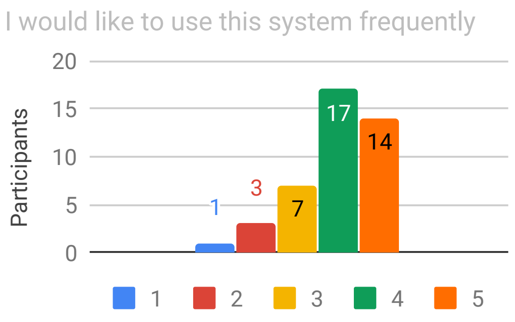


Figure A.20: Likert-scale results for the statement "I would like to use this system frequently" (1- Strongly Disagree 5 Strongly Agree).

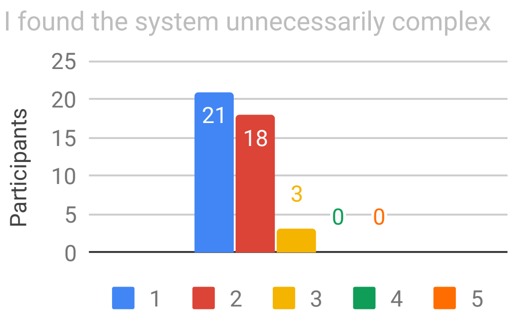


Figure A.21: Likert-scale results for the statement "I found the system unnecessarily complex" (1- Strongly Disagree 5 Strongly Agree).

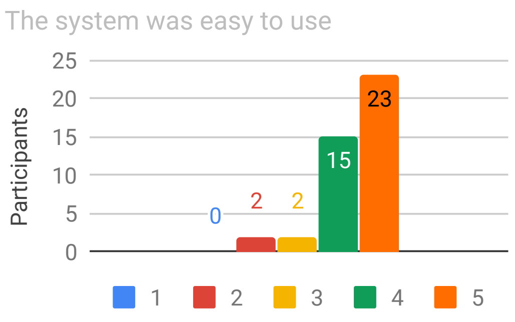


Figure A.22: Likert-scale results for the statement "The system was easy to use" (1- Strongly Disagree 5 Strongly Agree).

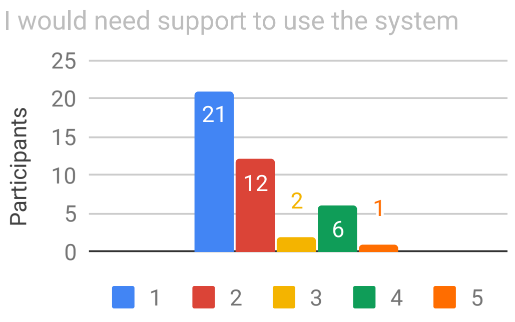


Figure A.23: Likert-scale results for the statement "I would need support to use the system" (1- Strongly Disagree 5 Strongly Agree).

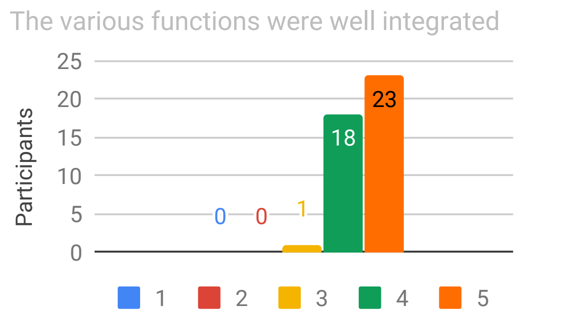


Figure A.24: Likert-scale results for the statement "The various functions were well integrated" (1- Strongly Disagree 5 Strongly Agree).

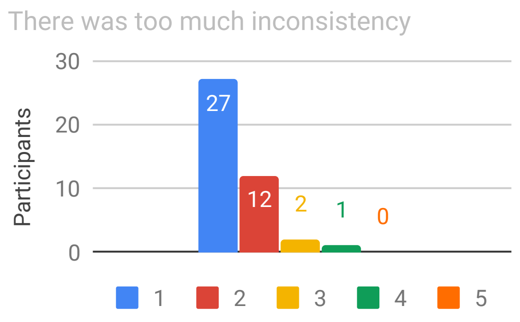


Figure A.25: Likert-scale results for the statement "There was too much inconsistency" (1- Strongly Disagree 5 Strongly Agree).

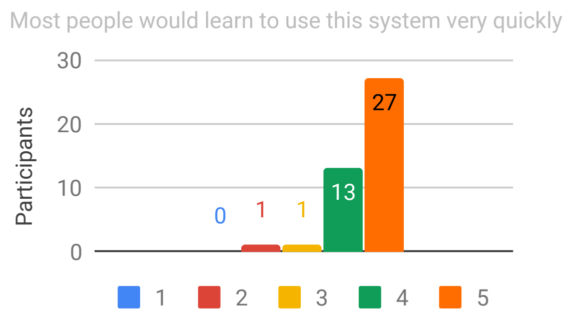


Figure A.26: Likert-scale results for the statement "Most people would learn to use this system very quickly" (1- Strongly Disagree 5 Strongly Agree).

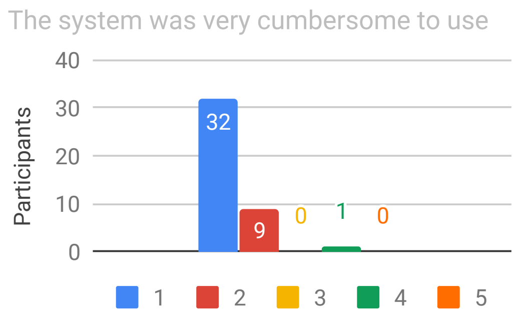


Figure A.27: Likert-scale results for the statement "The system was very cumbersome to use" (1- Strongly Disagree 5 Strongly Agree).

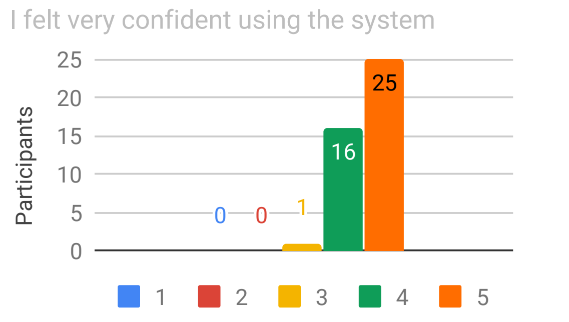


Figure A.28: Likert-scale results for the statement "I felt very confident using the system" (1- Strongly Disagree 5 Strongly Agree).

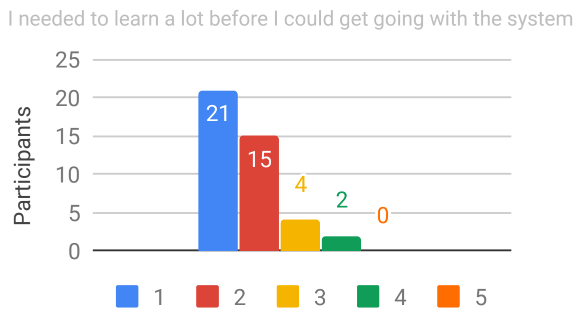


Figure A.29: Likert-scale results for the statement "I needed to learn a lot before I could get going with the system" (1- Strongly Disagree 5 Strongly Agree).

Appendix B

User Questionnaires

Pre-Experiment: Demographic Questions

Idade

____

Sexo	☐	Masculino
	☐	Feminino

Com que frequência interages com Realidade Aumentada (AR)?	☐	Algumas vezes por mês
	☐	Já usei algumas vezes
	☐	Nunca experimentei

Exemplos de AR: máscaras e outros filtros para a cara do Instagram; apanhar Pokémons no mundo real do Pokémon GO.

Post-Experiment: Controls-Related Questions

	1 - Strongly Disagree	2	3	4	5 - Strongly Disagree
De modo geral, achei os controlos intuitivos.	☐	☐	☐	☐	☐
Achei útil o modo como	☐	☐	☐	☐	☐
Achei fácil selecionar objetos.	☐	☐	☐	☐	☐
Achei fácil desseleciconar objetos.	☐	☐	☐	☐	☐
Achei que o modo usado para mudar de divisão funcionava bem.	☐	☐	☐	☐	☐
Achei fácil mover objetos.	☐	☐	☐	☐	☐
Achei fácil escalar objetos.	☐	☐	☐	☐	☐
Achei fácil rodar objetos.	☐	☐	☐	☐	☐
Achei útil endireitar objetos.	☐	☐	☐	☐	☐

Post-Experiment: Perceptions-Related Questions

	1 - Strongly Disagree	2	3	4	5 - Strongly Disagree
Achei fácil de perceber quando outro utilizador estava com um objeto selecionado.	☐	☐	☐	☐	☐
Achei fácil de perceber quando eu estava com um objeto selecionado.	☐	☐	☐	☐	☐
De modo geral, acho o "alertar" mais útil que as setas.	☐	☐	☐	☐	☐
Consegui perceber rapidamente se estava a ser alertado para um objeto.	☐	☐	☐	☐	☐
Achei útil a indicação das setas.	☐	☐	☐	☐	☐
Achei útil ver a localização e orientação dos outros utilizadores.	☐	☐	☐	☐	☐
Achei fácil de perceber a localização dos outros utilizadores quando não estamos na mesma divisão.	☐	☐	☐	☐	☐

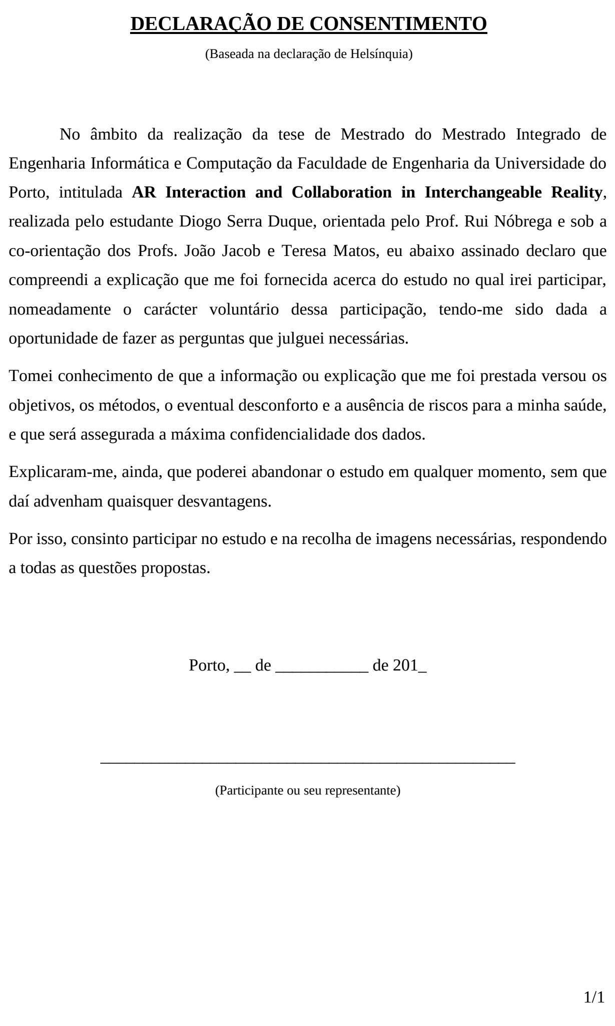
Post-Experiment: SUS Questions

	1 - Strongly Disagree	2	3	4	5 - Strongly Disagree
Acho que gostaria de usar esta aplicação frequentemente.	☐	☐	☐	☐	☐
Achei a aplicação desnecessariamente complexa.	☐	☐	☐	☐	☐
Achei a aplicação fácil de usar.	☐	☐	☐	☐	☐
Acho que precisaria de suporte de um técnico para conseguir usar a aplicação.	☐	☐	☐	☐	☐
Achei as várias funções da aplicação bem integradas.	☐	☐	☐	☐	☐
Achei que existia demasiada inconsistência na aplicação.	☐	☐	☐	☐	☐
Imagino a maioria das pessoas a conseguir aprender a usar esta aplicação muito rapidamente.	☐	☐	☐	☐	☐
Achei a aplicação muito desconfortável.	☐	☐	☐	☐	☐
Senti-me confiante ao usar a aplicação.	☐	☐	☐	☐	☐
Precisei de aprender muitas coisas antes de começar a usar a aplicação.	☐	☐	☐	☐	☐

User Questionnaires

Appendix C

Consent Form



<u>DECLARAÇÃO DE CONSENTIMENTO</u>

(Baseada na declaração de Helsínquia)

No âmbito da realização da tese de Mestrado do Mestrado Integrado de Engenharia Informática e Computação da Faculdade de Engenharia da Universidade do Porto, intitulada **AR Interaction and Collaboration in Interchangeable Reality**, realizada pelo estudante Diogo Serra Duque, orientada pelo Prof. Rui Nóbrega e sob a co-orientação dos Profs. João Jacob e Teresa Matos, eu abaixo assinado declaro que compreendi a explicação que me foi fornecida acerca do estudo no qual irei participar, nomeadamente o carácter voluntário dessa participação, tendo-me sido dada a oportunidade de fazer as perguntas que julguei necessárias.

Tomei conhecimento de que a informação ou explicação que me foi prestada versou os objetivos, os métodos, o eventual desconforto e a ausência de riscos para a minha saúde, e que será assegurada a máxima confidencialidade dos dados.

Explicaram-me, ainda, que poderei abandonar o estudo em qualquer momento, sem que daí advenham quaisquer desvantagens.

Por isso, consinto participar no estudo e na recolha de imagens necessárias, respondendo a todas as questões propostas.

Porto, __ de ___________ de 201_

____

(Participante ou seu representante)

1/1