NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE

TOMÁS BARROS SANTOS

B.Sc. in Computer Science and Engineering

MARC.MD- MULTIMEDIA AUGMENTED REALITY COLLABORATION IN MOBILE DEVICES

EXPLORING COLLABORATIVE AR INTERACTIONS

MASTER IN COMPUTER SCIENCE

NOVA University Lisbon
November, 2023

NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE

MARC.MD- MULTIMEDIA AUGMENTED REALITY COLLABORATION IN MOBILE DEVICES

EXPLORING COLLABORATIVE AR INTERACTIONS

TOMÁS BARROS SANTOS

B.Sc. in Computer Science and Engineering

Adviser: Rui Pedro da Silva Nóbrega
Assistant Professor, NOVA School of Science and Technology, Lisbon

Examination Committee:

Chair: Vasco Amaral
Associate Professor, NOVA School of Science and Technology, Lisbon

Rapporteur: Rui Rodrigues
Associate Professor, Faculdade de Engenharia da Universidade do Porto, Porto

Adviser: Rui Pedro da Silva Nóbrega
Assistant Professor, NOVA School of Science and Technology, Lisbon

MASTER IN COMPUTER SCIENCE

NOVA University Lisbon
November, 2023

MARC.MD- Multimedia Augmented Reality Collaboration in Mobile Devices

Acknowledgements

This dissertation proved as an opportunity, that I never had before, to have relative creative reign over the design of a thing I wholly believe in. For this opportunity, and the offering of a guiding hand whenever I needed it, I thank my advisor, Rui Nóbrega, very much.

I also thank my parents, Margarida Barros and António Santos, for the necessary love, care, and mental fortitude to reach this place. They were the first line of defense when I slacked off a bit too much, but also they were the first to the trenches when I was too entrenched in my work.

An important and underrated factor for any dissertation is a healthy amount of procrastination, and I have to thank my friends, Tomás Tavares, Paulo Matos, António Matos, Diogo Barradas, Donovan Oliveira, Francisco Cardoso, Nelson Moreira, Diogo Ferreira, whether sharing in the struggle of writing a dissertation with me or not, for making that procrastination all the more enjoyable. A special thanks goes to my friends over in room 248, Rodrigo Mesquita, André Costa and Henrique Ferreira.

Lastly, I would like to thank all the faculty in my department for giving me the know-how to be able to represent the university in any capacity. A special thanks goes to Professor Nuno Correia for lending me a tripod, so I could better debug my application, and Professor João Leitão for giving a hand in correcting some errors in my academic writing.

To a very specific person who will cite this document in the future, a special thanks to you too, of course.

“Pure will in every moment. There are no good ideas. ” (Rory Allan Philip Ferreira)

Abstract

In an in-person setting, using smartphones to share multimedia is hindered by their small display size. Furthermore, multimedia sharing applications, such as social media, are not designed for exclusive in-person interactivity, leading to a lack of simple methods for doing this action in the presence of friends or other collaborators. Besides sharing, there is a lack of options for in-person multimedia organization, which is an increasingly important aspect of collaborative multimedia-focused workflows, such as social media marketing and design thinking.

We propose a platform that uses Mobile Augmented Reality (MAR) to support collaborative multimedia sharing and organization. MAR provides the possibility of viewing and interacting with virtual elements in physical space. We explored various types of interactions that enjoy the unique characteristics of the medium to encourage effective collaboration, and, from these, we developed an interaction taxonomy. This interaction taxonomy provides categories for the different types of interaction that should be included in an application designed for this context, such as sharing and multimedia spatial manipulation. Furthermore, we implemented and evaluated a multimodal photocentric application composed of interactions derived from the taxonomy. We found that the interaction set derived from the taxonomy allowed users to, collaboratively, achieve complex organization goals and that its implementation was adequate at supporting it.

Keywords: Augmented Reality, Collaboration, Human-Computer Interaction, Multimedia

Resumo

Num contexto presencial, usar *smartphones* para partilhar conteúdo multimédia é um processo prejudicado pelo tamanho pequeno dos ecrãs destes dispositivos. Aplicações para partilha de multimédia, como redes sociais, não são feitas com o propósito de serem usadas exclusivamente em contextos presenciais, o que leva a uma falta de métodos simples para partilhar presencialmente multimédia.

Para além de partilhar, faltam também alternativas para a organização presencial de conteúdos multimédia, tarefa que faz parte de vários fluxos de trabalho focados em multimédia, tal como *marketing* em redes sociais e *design thinking*.

Nós propomos uma plataforma que usa Realidade Aumentada Móvel (RAM) que suporta a partilha e organização colaborativa de multimédia. RAM providencia a possibilidade de ver e interagir com elementos virtuais, mantendo o contexto do mundo real. Explorámos interações que aproveitam as caracteristicas únicas da tecnologia para motivar colaboração efetiva, e, destas interações, desenvolvêmos uma taxonomia de interações. Esta taxonomia de interação contribui categorias para as diferentes interações que podem ser incluidas numa aplicação que trabalha neste contexto, tal como manipular conteúdo multimédia espacialmente ou partilha-lo. Conjuntamente, desenvolvêmos e avaliámos uma aplicação que implementa a taxonomia através de uma interface multimodal no contexto de fotos.

Vimos que o conjunto de interações proposto derivado da taxonomia, no contexto implementado, permitiu aos utilizadores atingir, colaborativamente, objetivos de organização não triviais.

Palavras-chave: Realidade Aumentada, Colaboração, Interação Pessoa-Máquina, Multimédia

Contents

List of Figures

List of Tables

Glossary

extended reality A catch-all term referring to AR and VR. xviii, 28

kinaesthetics The ability to know where the parts of your body are and how they are moving. 21

viewport The visible space captured by a visual sensor and presented on a display. 22

Acronyms

6DOF Six Degrees of Freedom xi, 22, 23, 62

AR Augmented Reality ix, xi, xii, xiv, xvi, 1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 44, 45, 47, 48, 54, 60, 61, 62, 63, 65, 66, 72, 73, 83, 84, 86, 87

CLI Command Line Interface 19

GUI Graphical User Interface 2, 19, 20, 28

HCI Human-Computer Interaction 19, 30

HMD Head Mounted Display xi, 11, 18, 24

IMU Inertial Measurement Unit 23, 25

MAR Mobile Augmented Reality xi, xii, 3, 4, 10, 11, 12, 15, 16, 18, 20, 21, 22, 23, 24, 27, 28, 30, 31, 32, 34, 42, 43, 44, 52, 86, 87

OS Operating System 19

PDA Personal Digital Assistant 11

SLAM Simultaneous Localisation and Mapping 9, 10, 25

TLP Transport Layer Protocol 29

TUI Tangible User Interface 3, 12, 13, 20

UI User Interface 2, 22, 52, 54, 55, 56, 57, 58, 59, 62, 64

UMPC Ultra mobile personal computer 11

VE Virtual Environment xi, xii, 1, 2, 3, 6, 7, 9, 10, 12, 13, 15, 18, 19, 21, 28, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 45, 46, 48, 49, 50, 51, 52, 54, 55, 56, 57, 58, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 80, 83, 84

VR Virtual Reality xi, xvi, 1, 14, 15, 18, 19

XR extended reality 13, 15, 28, 29

1

Introduction

Augmented Reality (AR) allows users to interact, in real-time, with virtual objects that compose a Virtual Environment (VE), while still being able to perceive the real world [3]. This property of AR is achieved by superimposing the VE onto a view of the real world, unlike Virtual Reality (VR). For this superimposition to be effective, the location of the VE must be anchored to a location in the real world. Applications that use AR to augment a physical space with virtual information can be collaborative, in which multiple users can affect a shared VE or other users' VEs. In AR collaborative applications, users can exchange data spatially, through the VE [44].

AR, as a technology, has developed rapidly [7] following the success of various applications. Pokemon Go[1] allows users to share a VE through the *Share AR Experience*[2]. Ikea Place[3] can be used to preview accurately scaled furniture models in a markerless setting. Filters in Instagram and TikTok through Meta Spark[4] and Effect House[5] respectively prove that AR is already being used and succeeding in a multimedia content creating/sharing environment. Although these applications were not made to be collaborative, the use of AR, particularly in social media, raises the question of how can we expand the currently narrow, in scope and scale, use of Virtual Environments in order to create better interactive multimedia sharing experiences.

Also in recent years, there has been an effort to digitalize workflows that involve the collaborative use of multimedia. The success of tools such as *Miro*[6] and *Figma*[7] shows that synchronous spatial manipulation of multimedia content aids in collaboration, particularly in creative fields.

AR, as a technology, can allow for effective collaboration by introducing a spatial

[1] Pokemon Go: https://pokemongolive.com/en/,Last Accessed: 2023
[2] Share AR Experience: https://www.youtube.com/watch?v=PMTC7vbdA9Y,Last Accessed: 2023
[3] Ikea Place: https://www.ikea.com/global/en/newsroom/innovation/ikea-launches-ikea-place-a-new-app-that-allows-people-to-virtually-place-furniture-in-their-home-170912/,Last Accessed: 2023
[4] Spark Studio: https://sparkar.facebook.com/ar-studio/, Last Accessed: 2023
[5] Effect House: https://effecthouse.tiktok.com/, Last Accessed: 2023
[6] Miro: https://miro.com/index/ Last Accessed: 2023
[7] Figma: https://www.figma.com/ Last Accessed: 2023

element to the interactions present in collaborative systems, as well as encouraging communication and teamwork, especially in an in-person setting.

1.1 Motivation

As smartphone usage increases in recent years, it takes an increasingly bigger part of our daily lives and interactions. It's not uncommon to be in an in-person setting with other people and the interactions with the group being assisted with smartphones. In this context, the way of sharing multimedia content is either: through communication platforms, such as direct messaging applications or social media, or by physically handing out your mobile device. The latter option leads to an inefficient and non-interactive endeavor, mainly due to the small display size of smartphones, which doesn't easily allow showing virtual information to more than two users. An example of this problem would be a person sharing their vacation photos with a group of friends. A collaborative AR mobile application that allows multiple smartphone users to share a VE and manipulate multimedia would bypass the restriction of the physical display in this context.

Although AR is a well-researched technology, most AR experiences currently offered to users are single-user experiences, where any collaboration/interaction with other users is offered in the form of non-AR User Interfaces (UI), if at all, which signals a lack of collaborative AR interfaces.

Collaborative AR systems can work in in-person/face-to-face and remote/online contexts, allowing them to work under the shift of the context of collaboration that has occurred due to COVID-19 [54].

File sharing is a widespread practice in modern collaborative projects, due to the high degree of digitalization of information. Multimedia resources are especially digital, sometimes with no real-world counterpart, such as in the case of 3D models. Representations of multimedia files in an AR VE can allow users to share, display and organize them in the context of a physical space. This, in turn, allows for spatial interactions that are not present in 2D Graphical User Interfaces (GUI), as well as providing a preview of how these multimedia resources look in the real world, as is the case with *IKEA Place*.

1.2 Research Questions

The following research questions were formulated to explore the most important aspects of collaborative AR interfaces and how to implement them in the context of multimedia manipulation. The questions are further derived from the main problem this thesis intends to solve: *How can a group of users collaborate on a set of multimedia resources by using a synchronous VE and what interactions should an AR interface provide to allow for engaging collaboration?*

RQ1: What means of interaction with multimedia content should a collaborative AR system implement?

AR systems provide a 3D interaction space where users can interact with virtual elements, and possibly real elements (in the case of Tangible User Interfaces (TUI)) to influence the VE's state. Given the characteristics of the MAR paradigm and how MAR devices have different input modules and sensors that can be used to register input from the user, this question, in summary, points to two concepts: which input methods does the system use to interact with the multimedia content, and how to design the multimedia content's virtual representation and their actions, focusing on their manipulation in 3D space.

RQ2: How can users collaboratively use the multimedia content in the VE to reach a shared goal?

Sharing multimedia content is an essential part of some fields, such as education, collaborative 3D modeling and UI development, and workflows, such as video conferencing. This question explores how users can, through the collaborative AR system's interactions with the multimedia content, allow for the sharing, manipulation and organization of multimedia content.

1.3 Objectives

To answer the questions raised in Section 1.2, the objectives of this dissertation are:

- To create a mobile AR system that allows for collaborative, real-time interaction on multimedia content, specifically images. The implemented system must be able to support: the sharing of multimedia content, 3D spatial organization of multimedia objects.
- Find AR collaborative means of interaction for the multimedia content. Using already known patterns of interaction and collaboration, the implemented system must include sharing and organization actions. These actions and their appropriate inputs must be designed within the context of mobile AR.
- Measure the efficacy of the above-mentioned system to share and spatially organize all the file types mentioned, in the context of photo categorization.

1.4 Solution Description

The developed solution consists of an in-person, collaborative AR system that runs on smartphones and tablets, allowing for synchronous collaboration through interactions with the VE and between collaborators. The AR tracking is done through a marker shared by the collaborators. The system allows users to spatially manipulate images in the VE

and interact with other users' virtual representations. This system's implementation is discussed in detail in Chapter 4 and Figure 4.7 shows the implementation's interface.

The system achieves abstraction from networking concepts by pairing a collaborative session's marker with the information needed to connect to that session. Thus, users can connect to an already existing session simply by pointing their devices' cameras to that session's marker.

Collaborators have corresponding virtual representations which are used to show what the user is currently doing as well as providing feedback as to where that user is looking. These virtual representations can be used by others to share photos with that specific user. Users can choose a specific photo to present, highlighting it to all other users present in that session. Furthermore, each collaborator has access to a personal folder, which all users can use to organize photos. Users can manipulate photos by moving them along a parallel plane to the marker.

1.5 Contributions

This dissertation provides various contributions to the area of collaborative, multimedia-focused MAR applications:

Interaction taxonomy which aids in the development of applications in this domain by categorizing AR interactions based on their characteristics and goals. This taxonomy offers three main categories, which are:

- Organizational interactions, pertaining to interactions that provide users with ways to enforce an organizational system upon a set of multimedia items. Organizational interactions can be used, for example, to move an image across the pages of a virtual photo album, or to create folders to help in video categorization efforts.
- Sharing interactions, which describe interactions that allow the exchange of information, in the form of multimedia content, between two or more users. These interactions range from a user sharing an image with another user to a user presenting a piece of multimedia to everyone they are collaborating with.
- Multimedia-specific interactions, a category which describes interactions that manipulate or interact with the multimedia content in ways specific to that type of multimedia content. This category describes interactions such as starting video playback or zooming in images.

Marker-based application design principles were formulated for the interactions described in the taxonomy, describing the unique characteristics of the medium and how multimedia content can be displayed and interacted with in it. An important factor mentioned is how various users organize themselves in physical space in order to get the best view of the multimedia content in the AR virtual scene.

Collaborative MAR interface which implements a set of interactions derived from the taxonomy, in a marker-based, in-person context. We implemented a multimodal interface (which contains different modes of interaction, each with different functionalities) to evaluate the taxonomy, as well as how this type of interface can be used to minimize visual clutter. This application allows a set of users to synchronously share a virtual environment and collaborate, using the taxonomy's features, in a photo categorization workflow.

User study which allows for the evaluation of the implemented interface using qualitative and quantitative metrics. This protocol includes tasks pertaining to singular actions in the application, as well as a task that allows users to achieve a complex goal using their preferred methods of organizing, sharing and interacting with the photos present in the virtual environment.

1.6 Document Structure

This document is separated into six chapters:

Introduction: This chapter of this document portrays, in a summarized manner, the context behind this thesis and its objectives.

State-of-the-art: This chapter provides the technological and academic background for this thesis. Each subsection focuses on a specific topic, with the main topics being Augmented Reality, Collaboration and Human-Computer Interaction, with a focus on how their general concepts relate to the context of real-time AR applications.

Collaborative Interaction Design: This chapter discusses the system developed to answer the presented research questions, focusing on its task requirements, the interaction design decisions made to fulfill those requirements and the principles that guided them.

MARC.MD Implementation: This chapter details how, technologically, the system's implementation was achieved, focusing on the system's architecture and the implementation of the application's functionalities.

Evaluation: This chapter describes the system's user testing process and the choice of metrics to evaluate the usability of the system. This chapter also shows the results obtained from the user testing and compiles them to conclude the system's capacity to answer the research questions.

Conclusion: This chapter ponders the work done in the context of this thesis, as well as describes what can be done in the future to further the research in the field.

2

State of the Art

This chapter explores the underlying concepts that are used for the development of this thesis, by analyzing the technological and theoretical components of its main topics: Augmented Reality (Section 2.1), Collaboration (Section 2.2) and Human-Computer Interaction (Section 2.3). It will also address the tools and frameworks that are currently used for the development of collaborative AR systems in Section 2.4.

2.1 Augmented Reality

Augmented Reality, defined as the real-time, interactive combination of real word and virtual environments [3], is a concept that shares its design philosophy with other technologies, this philosophy being the combination or replacement of some or all elements of the virtual and real worlds [46]. Due to this shared objective, these 'reality technologies' are often compared amongst themselves, leading to the need of a basis of comparison. Schnabel et al. proposes the axis of 'reality-virtuality' (present in Figure 2.1) to be this point of comparison [46].

This axis is used to determine how close the reality technology's presentation and interactivity are to either the ones of a real environment or a virtual one. This is determined by two attributes, or dimensions: the extent of interaction with real entities, or how many of the interactions the systems offer are with the real environment, and correlation between action and perception, or how are the action space, where the system's interactions are done, and the perception space, where we receive our visual or tactile feedback, correlated to each other.

AR, in this spectrum, stands closer to a interactive system operating in reality than one operating in virtuality. This is attributed to the high correlation between action and perception spaces, since the action space is the real world and the perception space is the real world seen through a mediating device, or AR device, and in the fact that real objects can be used for interaction in these systems (Section 2.1.3). To implement these characteristics, AR systems must keep track of where the VE is supposed to be in relation to the real environment (Section 2.1.1) to allow for effective spatial interaction

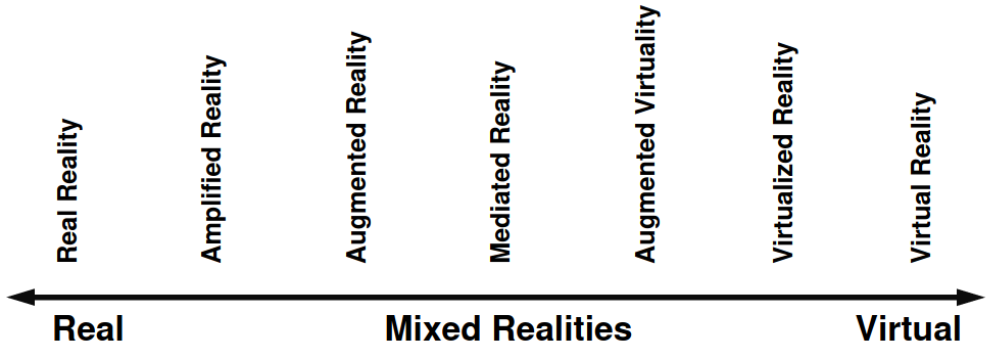


Figure 2.1: The Reality-Virtuality Axis, where multiple 'reality technologies' are rated [46].

(Section 2.1.3,Section 2.3.3).

2.1.1 Tracking

Currently, the various techniques used to track virtual location and its relation to the real world space are divided into two categories: marker-based techniques which mainly rely on fiducial markers to calculate the location of the Virtual Environment, and markerless techniques which include sensor-based approaches and natural feature processing approaches to determine the location information.

Marker-based Tracking: It relies on computer vision algorithms to recognize and track markers and use them as a spatial reference, so that a Virtual Environment can be superimposed onto physical space.

There's a number of different algorithms used to detect fiducial markers and track their keypoints, such as SIFT, SURF, ORB and KAZE [30].

One of the disadvantages of using a fiducial marker AR system is the fact that the markers themselves are visually obtrusive [49] due to the characteristics they should have to maximize their effectiveness. These marker-based systems, however, have a lighter computational load, which makes them attractive for AR systems which run on low power hardware, such as some mobile device platforms (Section 2.1.2).Additionally, fiducial markers allow the AR systems to leverage the speed and precision they provide in regards to camera pose estimation, since their design properties allow for quick keypoint recognition, without ambiguity. These properties include [55]:

- Black and white color palette. This maximizes contrast between shapes, allowing for quicker determination of a shape's edges using edge detection algorithms, a step needed for the execution of most of the keypoint detector algorithms mentioned prior.
- No axes of symmetry. This guarantees no ambiguity regarding the marker's orientation, and consequently, the orientation of the virtual environment.

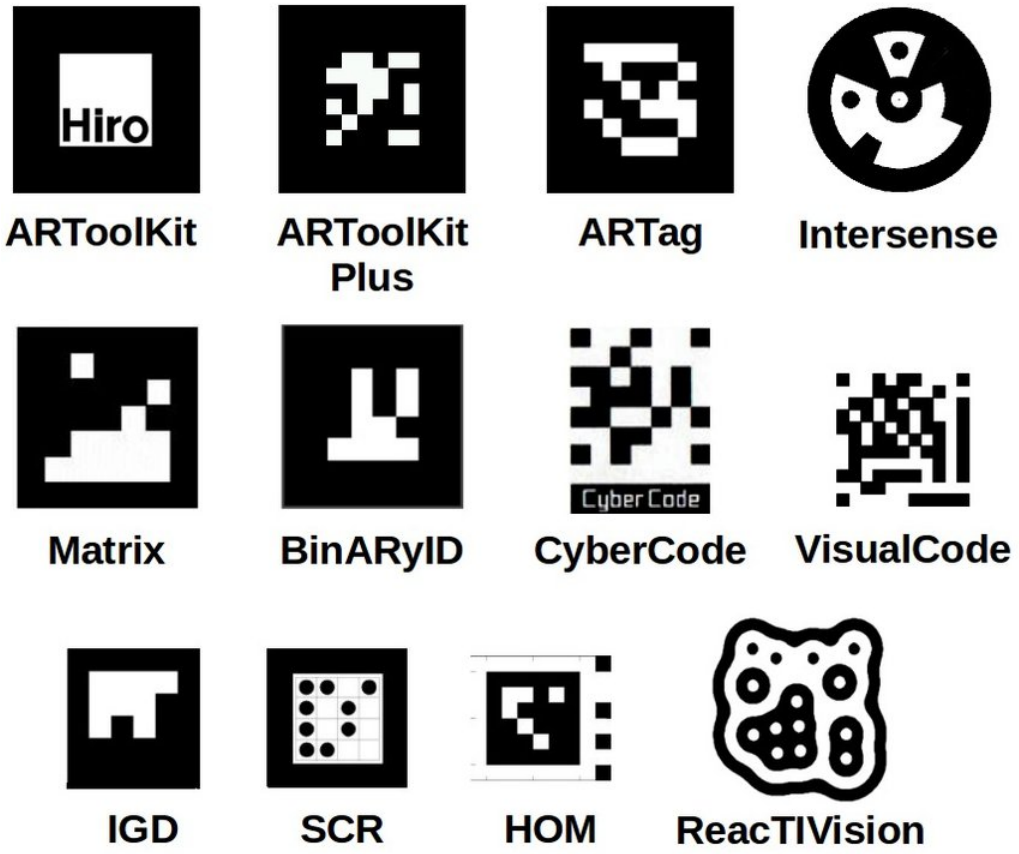


Figure 2.2: Various fiducial markers used in different marker-based systems [18].

- High number of keypoints. This characteristic guarantees increased precision under occlusion. These are detected differently based on the algorithm used, so it is a common approach to design a marker and the algorithm that detects it in tandem.

Markerless tracking: These methods can be divided into two categories: sensor-based, where the camera's pose (location and rotation) is estimated using sensors and positioning technologies, and vision-based, where videos or image sequences are used to estimate the camera's pose through computer vision and image processing algorithms [11, 43]. Included in the sensor-based tracking approaches are:

Inertial-based tracking: It leverages linear acceleration measuring modules (accelerometers) and angular rotation measuring modules (gyroscopes) to obtain the relative position and angle to its previous pose. To obtain the camera's absolute pose, referring to its real position and angle, prior knowledge of the initial pose of the camera is needed. This approach has a drift problem, due to the accumulation of sensor errors and jitters originating from external interference. This can be mitigated by employing periodic re-calibrations.

Magnetic-based tracking: It uses magnetometers to determine the location, orientation, or both, relative to either a created magnetic field, or Earth's magnetic field. An example of a magnetic sensor which is widely used is the compass, integrated into smartphones and tablets. This sensor allows to determine the orientation relative to Earth's magnetic field. This method of tracking may suffer from interference from other magnetic fields present in the magnetometer's vicinity. The magnetic field used to calculate the sensor's pose can suffer distortion from the presence of metal, which can lead to erroneous tracking.

Vision-based markerless tracking can be subsequently decomposed into two subcategories: model-based tracking or no-model tracking [43]. Model-based approaches require prior knowledge of the real environment through the use of 3D models, which can approximate the whole environment or specific objects, for camera pose estimation and tracking purposes. No-model approaches don't require prior environmental information to function. Model-based tracking approaches include:

Edge-based tracking: This method projects the 3D model of the environment into an image, sourced from a video stream, and matches it with the edge features of that image by 'minimizing the displacement between the projected model and imaged object' [47]. This approach was the first to achieve real-time 3D object tracking, and due to its low complexity, features good computational performance and is easy to implement, but can't track fast camera movements.

Optical flow tracking: It tracks physical points in space by tracking them in the camera's video sequence, while measuring their velocity at each pixel's location. It doesn't handle sudden changes in illumination and camera position well.

Depth Imaging: It is used to compute the camera's pose in markerless AR systems by using depth images containing the distance of the objects in the environment. The reference 3D models are created offline, but the system allows for runtime model adjustments [43].

In use cases where the environment is unknown, these previously discussed approaches can't be used. In these situations, a no-model approach should be used. This approach tracks the camera's pose and registers the environment simultaneously [43].

Simultaneous Localisation and Mapping (SLAM)-based approaches are a popular option for vision-based no-model AR systems. The SLAM problem "(...) has become well-defined in the robotics community as the question of a moving sensor platform constructing a representation of its environment on the fly while concurrently estimating its ego-motion." [13]. While SLAM has been used mainly for the registration of real objects in physical space, it can also be used for the display of a VE in an AR system. It's possible to have a working real-time SLAM algorithm working on a mobile device with a single camera sensory setup, as shown in Davison's MonoSLAM [13] and, more recently, Qing's

proposed framework [17], which expands the ORB-SLAM method devised by Mur-Artal et al. [35].

There can be systems that implement multiple markerless tracking methods in order to compensate for the disadvantages of the methods in a standalone implementation. This approach can be seen in vision-based systems which use sensor information to make better camera pose estimation, such as SLAM approaches using GPS tracking to partially mitigate the vision-based approach's poor performance in outdoors settings [43].

Vision-based markerless tracking incurs a bigger computational load when comparing it with marker-based tracking, or strictly sensor-based markerless approaches.

GPS-based Tracking: Augmented reality, in its most encompassing definition as being the interactive superimposition of virtual elements into physical space, can be achieved by tracking the location of the user through a GPS system, instead of processing a image stream to register and track either fiducial markers or natural features in the approaches described prior, and present a VE based on the user's geographical location. This approach falls under the category of sensor-based markerless tracking.

This AR tracking paradigm has had success in Mobile Augmented Reality (MAR), due to the almost ubiquitous presence of GPS systems implemented in smartphones and tablets, and is a popular approach for AR games, shown by the success of Niantic's games - Ingress[1] and Pokemon GO[2].

GPS-based tracking suffers from poor location determining performance indoors, a large location estimation error and relatively long latency, leading to it not being applied in scenarios where high indoor tracking precision or real-time, latency-free tracking are needed [27].

2.1.2 Mobile Augmented Reality

MAR is a field of augmented reality research and development which focuses on AR applications made for mobile device platforms. This paradigm of AR has been emboldened by the upwards trend of mobile devices' computational performance.

Chatzopoulos et al. expands Azuma's definition of AR [3] to define the characteristics of a MAR system as a interactive, real-time system, which combines real and virtual objects in the context of the real world, as well as running or being displayed in a mobile device [11].

The device platforms used for MAR applications are the following [11]:

Laptops/Notebooks: This class of devices has high computing and rendering power, allowing for the development of complex AR applications, but is lacking in portability and battery life. Setups using these devices also need additional display devices,

[1] Ingress Website: https://www.ingress.com/,Last Accessed: 2023
[2] Pokemon Go Website: https://pokemongolive.com/en/,Last Accessed: 2023

Figure 2.3: Notebook being used as a backpack computing platform displaying debug info, coupled with HMD [11].

since they are used only as backpack computing platforms to increase user mobility. An example of one of these AR setups can be seen in Figure 2.3.

Tablet PCs: Tablet PCs are personal mobile computers which include large touchscreens. They feature enough computational power to run MAR systems, but are too heavy for long single-handed use, which in turn limits the interactions that can be done in the MAR systems to be either touch-based or motion-based (Section 2.3.3).

Smartphones/Tablets: This class of devices provides high portability and endurance, but lower computing power when compared to other classes. The usage of smartphones is also almost ubiquitous nowadays, which results in a bigger target market for MAR applications.

Head Mounted Displays (HMD) / AR Glasses: This class of devices provide maximum portability, since they allow for hands-free interaction. They provide high endurance as well, at the cost of computing power. The computing power disadvantage can be mitigated by using other devices as computing platforms, such as laptops or UMPCs.

Personal Digital Assistants (PDA): These devices afford high portability and endurance, but similarly to smartphones, lack computational power. However, unlike smartphones, they are not used ubiquitously, and it can be said that their use as a highly portable device capable of storing and presenting digital information has been deprecated and superseded by smartphones. These devices were used in the *medie.welten* project [45].

Ultra mobile personal computers (UMPC): UMPCs offer enough computational power and endurance to run MAR systems, but their high price impedes their widespread. They also are limited, in regards to interactable space, by their small display.

2.1.3 Interactions in AR systems

According to a literature analysis done by Hertel et al. [25], interactions in AR systems can be broken down into tasks, or actions, such as object creation (through the placement of an already existing asset or through the act of creation from scratch), selection and manipulation, the latter being further classified as either geometric (as in manipulating an object's geometric properties, such as rotation, scale and position) or abstract (interactions which don't affect directly any object in the VE, such as filtering some subset of objects or accessing menus).

In this taxonomy model, interactions also may have either a single modality or several, in the case of multi-modal interactions, that describe how a certain task is achieved. These modalities include the following [25]:

- Tactile Interaction, defined as any interaction that involves touching, whether that be a physical object, in the case of AR TUIs, or a virtual object through an input device, such as a touchscreen on a smartphone, for example. This modality has been used in several studies [8, 14, 21] and is prevalent in MAR systems, as many of the mobile devices possess touchscreens.

- Gestures, the most common being face and hand gestures, due to their inherent intuitiveness that allows for users to become proficient in utilizing the system quicker. Hand gesture tracking (further explored in Section 2.3.3) has been used in [28, 32].

- Voice, frequently used in conjunction with other modes to execute simple tasks, such as creating or deleting virtual objects. This type of multi-modal interaction can be seen in [53].

- Gaze, which can be broken down into head gaze and eye gaze, and is used primarily for the tracking of virtual objects for selection purposes or to improve collaboration efficiency by providing the collaborators with an estimation of the intent of the other users by their gaze [50].

Tangible User Interfaces (TUI) are a part of tactile interactive interfaces which can be defined as an interface type that allows the coupling of virtual information and physical spatial information through the association of physical objects with virtual ones. This results in 3D interfaces where the interface elements are located in the real world, and actions that affect these elements are then parsed to have an effect on their virtual counterpart in the VE. These interfaces lends well to AR, since, when using marker-based approaches, this relationship between virtual and physical objects is formed intrinsically, through the use of fiducial markers. These typically are used as anchors for the VE or separate virtual objects [41], and thus their location in the real world is directly correlated with the VE's state. This is specially apparent in marker-dependant AR games, where the field of play, and consequently, of interaction, is defined by the presence of a marker.

Manipulating the marker in physical space can be considered a form of interaction, and thus, can be utilized as such to execute the tasks previously described in Section 2.1.3.

2.1.4 AR Applications

Augmented Reality has seen use in various applications from different fields of study. In this section, we will focus on AR uses in the entertainment, cultural heritage, medicine, engineering and education fields.

AR in Entertainment. AR has been used in several areas of entertainment, mainly the interactive ones. AR has been used to develop video games [48], such as *Invizimals*, a handheld console game which used marker-based tracking to define the play space for its turn-based battles and 'capture' sequences, where users play mini-games based on the '*invizimal*' they're capturing, which can involve using motion gestures (further explained in Section 2.3.3). Before the 'capture' sequence is initiated however, an exploratory phase needs to be completed, which involves finding a color in the real environment, which promotes user exploration of their surrounding space. For specific '*invizimals*', their specific marker must be registered to allow their appearance in-game, which is a concept that is the basis of AR card games. *EyePet* is an AR video game which uses a stationary camera and a mobile marker to defined its play space. The marker is both used for initialization of the VE, which stays stationary for the course of the play session, and interaction, by using it coupled with mid-air gestures (further explained in Section 2.3.3) to play a variety of minigames. This use of the marker as a interaction medium can be classified as a TUI [9].

AR in Cultural Heritage. AR has been used to develop applications regarding cultural heritage [5, 15, 34]. AR can enrich cultural heritage sites and artefacts through added information [5]. XR technologies can are also be used to recreate lost heritage, a useful property for fields of cultural heritage preservation such as archaeology, where the cultural heritage sites and artefacts can suffer from extended time periods of exposure to natural elements and thus, be in a state of disrepair. A distinct advantage of the virtual recreation of heritage sites is the fact that it allows for the remote visualization of the cultural heritage sites or artefacts [15], which can be used to augment museum exhibitions, and still be used as a way to improve visitor's immersion and engagement with the cultural heritage site in on-site applications [5].

AR in Medicine. AR provides a framework for the visualization of a virtual scene in the context of the real world. AR can be used in medicine to adapt current display techniques to be more effective at conveying information to either medical professionals or patients. A field where this becomes apparent is the treatment of phantom pain.

Phantom pain is a type of pain that occurs on amputees where their amputated limb used to be. A common treatment to phantom pain is mirror therapy, where the patient is

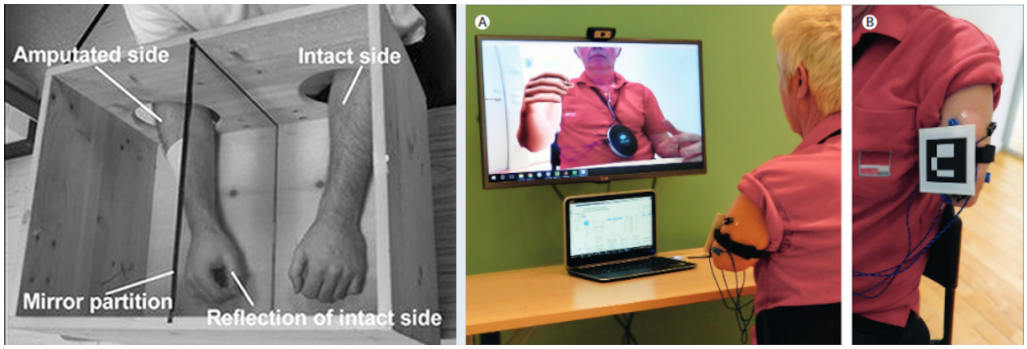


Figure 2.4: Mirror therapy setup for phantom pain treatment (left) [16], AR missing limb visualization using marker-based approach with fiducial markers (right) [39].

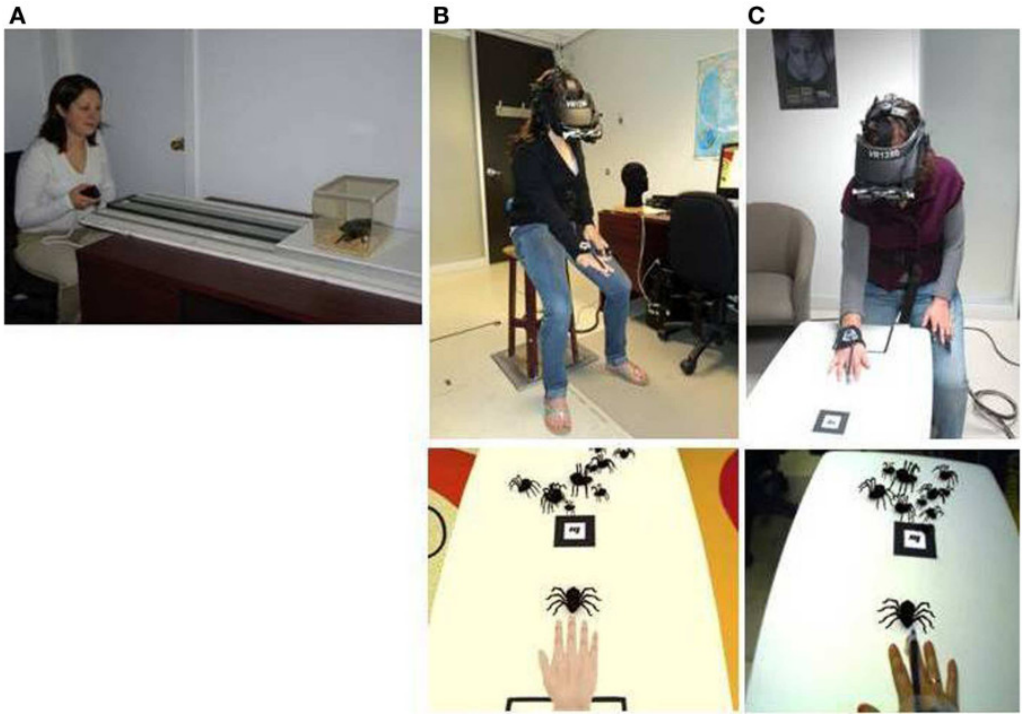


Figure 2.5: (A): *in vivo* exposure to phobia stimulus (spider), (B): VR exposure to spider interacting with virtual representation of patient's body, (C): AR exposure to spider stimuli interacting with patient's real body [4].

seated in front of a mirror so that, for the patient, the reflection of the remaining limb is where the amputated limb used to be and perform physical exercises. This treatment can, however, be improved by featuring better missing limb visualization. Ortiz et al. used AR paired with machine learning to create a treatment plan for phantom pain, using machine learning for garnering input from the amputated arm, and marker-based AR to place a virtual arm on top of it. This treatment was tested on fourteen participants and the results feature less intense pain felt with less frequency after using this treatment. [39] Figure 2.4 shows a side-by-side comparison of the two procedures' (mirror therapy and AR-aided treatment) setups.

AR has seen use in the field of exposure-based therapy [4], a field of therapy where

patients which have phobias are gradually exposed to the object of their specific phobia in a controlled and safe environment. Without using XR technologies, the exposure to the anxiety or fear inducing object can be done *in vivo*, where the patient is in presence of their phobia inducing object or *in imago*, where the phobia stimulus is imagined. By using XR technologies, the exposure can be done within a VE, guaranteeing the controlled and safety requisites of this medical procedure and allowing for more fine-tuned control of the stimuli given to the patient. A distinct advantage that AR has over VR in this field is using the real world as the context of the VE. This allows to decrease development costs, since there's no need to create the whole virtual environment, only the phobia inducing stimuli, and allows for the patient to see their own body interact with the phobia stimuli, contrary to the VR approach where patients see a virtual representation of their body, which can elicit better judgment of reality, and thus, better results on their rehabilitation [4]. The setups for *in vivo*, VR and AR exposure are shown in Figure 2.5.

AR in Engineering. In the field of engineering, specifically product design and manufacturing, product assembly simulation is an essential step for guaranteeing assembly quality and efficiency. This process involves the construction of a physical prototype, which incurs a increase in costs and in the time needed for the products to enter the market. AR is used in the field of engineering to help in the process of product assembly simulation [42] by offering a much cheaper alternative to the physical construction of prototypes, the construction of a digital twin (DT). A DT is a 'virtual representation that provides simulation, or runtime products with a virtual reflection of the real world throughout multiple phases of the product or plant lifecycle' [12].

AR can be used coupled with DT technology to create a Virtual Environment similar in functionality to a real assembly plant and test its performance [42]. It can also be used to augment existing physical assembly and maintenance lines, an example being Haritos et al.'s AR system for aircraft maintenance training [24].

AR in Education. AR in the field of education has been mainly explored through AR educational games and simulations.

Schmalstieg et al. explored applications of Mobile Augmented Reality in a museum educational setting [45], where participants, in a case study titled *medie.welten*, had to complete a number of tasks by scanning fiducial markers located in different exhibits and solving problems related to an overarching narrative relating to espionage in World War II, under a time constraint. This time constraint resulted in increased motivation if the participant was proficient with the AR aspects of the experiment, but increased frustration when participants were not. Although the results were positive, it was found that a subset of *medie.welten*'s participants were not familiar with how marker tracking works, provided that it's not an intuitive system by default. For their following case study - *Expedition Schatzsuche*, these two aspects were fixed, by removing the time constraints and providing a more thorough explanation of how the marker tracking works, whilst

also providing in-system feedback for the tracking, in the form of an outline that appears around the marker when it is detected.

In this analysis of the potential of MAR in education [37], Nincarean et al. concluded that besides AR not providing, by itself, an increase in learning efficiency, and instead having to be paired with good game design and teaching principles when designing the application to achieve positive results, most participants of the previously mentioned experiments showed increased motivation and willingness to engage with the taught material, which can itself translate into better learning performance.

2.2 Collaboration

The field of Computer-Supported Cooperative Work (CSCW) studies how persons cooperate with the aid of computer systems, and helps develop collaborative computer tools and frameworks. This section is based on this field of study and, within it, the two paradigms of modern collaboration, these being online collaboration and in-person collaboration, are explored, as well as how they relate to AR systems.

2.2.1 Online Collaboration

With the increase in remote work and remote learning needs during the COVID-19 pandemic [54], a lot of traditionally in-person workloads were adapted to be online. These include collaborative workloads, the most prominent being meetings, which were adapted to the online environment through videoconferencing applications, such as Zoom[3] and Google Meet[4]. Videoconferencing allows for effective verbal communication between collaborators, but is not as efficient for more complex workloads, such as e-learning [36].

Online collaborative AR can be used to improve videoconferencing by allowing remote annotation on physical space [19, 33]. This AR paradigm mainly applies to workloads that involve two parties, each with complementary functions: the 'technicians' or on-site collaborators, which provide the video stream to be augmented, and the 'experts' or remote collaborators, that create the AR content overlay in the form of spatial annotations. These systems can work either in a synchronous way [19], or asynchronously [33]. In the former approach, due to the unstable nature of a hand-recorded video feed and the fact that annotations could be associated with the wrong object, the annotation process must accommodate this restriction. Gauglitz et al. resolved this problem by implementing a frame 'freeze' function available to the remote participants, in which the remote collaborators can pause their view of the video stream and annotate on a single image frame [19], returning to the live video feed when the annotations where done. In the latter approach, the local collaborators can manually extract a image frame and annotate

[3]Zoom Website: https://zoom.us/,Last Accessed: 2023
[4]Google Meet Website: https://meet.google.com/, Last Accessed: 2023

Figure 2.6: Collaborators interacting with Sandscape TUI.

it, send it to the remote collaborator, which can reuse annotations and provide new ones, allowing for effective asynchronous communication through annotations [33].

Collaborative AR in an online environment comes across two major restrictions, which are the loss of the concrete position and direction of each collaborator, which affects collaborative AR system design, and the loss of the non-verbal communication that is present in an in-person setting. For some workloads, such as analyzing and annotating a 3D model collaboratively, the concrete pose of each collaborator can be substituted with the relative position and direction of each collaborator to the model and a virtual object representing each collaborator, providing no loss in functionality [29].

2.2.2 In-Person Collaboration

When discussing in-person collaboration, it's important to note that, given the spatial proximity of all collaborators, it's possible to have a collaborative system that runs on a single device, and so, it's important to make the distinction between multi-device collaborative systems and single-device ones.

The latter approach is attractive to the field of collaborative tangible interfaces. An example would be SandScape [5] (visible in Figure 2.6) and informational digital kiosks, such as TOMI World [6]. The main advantage of these systems is not having to synchronize state between devices, leading to a more fluid collaborative endeavor, which can work for large collaboration groups [52], if the device in question has the capabilities to support them.

The former approach allows each collaborator to access, individually, their interfaces, which can result in greater work parallelism, when the collaboration objective demands

[5]Tangible Media Group's Sandscape: https://tangible.media.mit.edu/project/sandscape/, Last Accessed: 2023

[6]Tomi World Website: https://tomiworld.com/pt/, Last Accessed: 2023

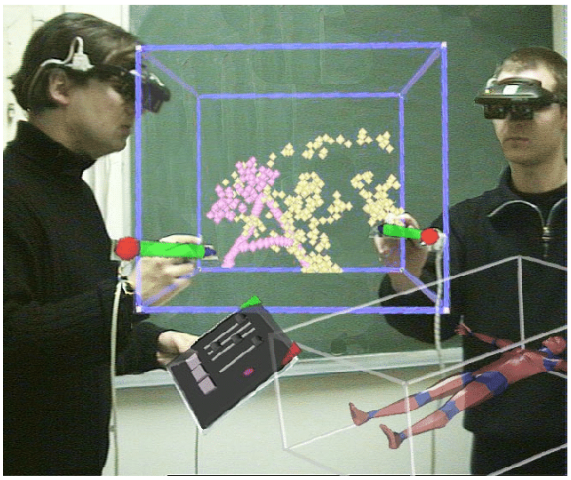


Figure 2.7: Users interacting with shared Virtual Environment, using HMDs and tangible controllers [44].

it, and in the capability of having different interfaces for different user types or technologies, prominent examples being teacher and student specific interfaces with different functions [26] and AR and VR interfaces affecting the same VE [40]. Multi-device collaborative systems lends better to AR than their counterpart due to the ubiquotous presence of AR-capable mobile devices, such as smartphones and tablets, which are better operated by a single collaborator, and due to the fact that, to have a large group collaborate on a single device, this device's interactable space needs to be large itself, hindering the portability of the system, meaning these collaborative platforms are, many times, immobile [52].

This approach is also compatible with online collaboration, meaning that any MAR multi-device system that is implemented for an in-person setting could also work for online collaboration, in the case that the system in question regards the online collaboration problems described in Section 2.2.1. This is the case for ARCritique [29], where a AR task often done in-person (multiple people interacting with a 3D object) is adapted to be done online by divorcing the virtual object being analyzed of its concrete spatial location and relating each collaborator's pose to their local view of the object, whilst placing a a smartphone 3D model in this relative pose, simulating where their smartphone would be if the collaborators where collaborating in-person. It is also of note that there can be overlap of online and in-person collaboration in a system (for example, a system can recognize two collaborators that are situated in site A and two other collaborators that are situated in site B).

In-person collaboration allows for the usage of a broad range of communication means between collaborators, such as speech and gestures similar to its online counterpart, but has the added component of the spatial information of each collaborator, such as their location in space or the direction of their gaze. This component can be used to enrich the system itself, for example, by using these to calculate the quality of the view each

Figure 2.8: 3D Model annotation through remote AR collaboration. View of collaborators (left and right), shared VE (middle) [29].

collaborator has of an object, such as what was done in VR by Wang et al. [51], or by integrating gaze direction feedback within the system [50].

2.3 Human-Computer Interaction

Human-Computer Interaction (HCI) is the field of study of how humans interact with computers and technology in general. This involves the design and evaluation of interaction systems and interfaces with the objective being improving interactive systems in usability, accessibility and efficiency.

AR, as an interactive medium, can be explored through the lens of HCI to derive interaction and interface patterns.

2.3.1 AR Interfaces

Background on GUI Design: Graphical User Interfaces (GUI) were first featured in the release of the Xerox Alto, in 1973. GUIs consist on interfaces that are 'graphical widget'-based (as opposed its text-based counterpart in CLIs) and allow interaction through manipulation of these 'graphical widgets', whether this manipulation is spatial (dragging windows to different places in screen, make windows fullscreen, minimize windows) or logical (clicking on a folder opens a window containing the folder's contents). Navigation through these interfaces is often done with a pointing device, such as a mouse, and a keyboard. The 'desktop' as an interface metaphor has its origin in GUIs, and serves as the centerpiece of modern Operating Systemes (OS). Agarawala et al. explored how to improve this interface metaphor by approximating its functionality to the functionality of a real world desktop, adding depth, physics, and more literal representations of files (images are represented by themselves, as opposed to icons) [1].

Smartphones and tablets also implement GUIs, with the added notion that the touchscreen is the input method and the size of the touchscreen is not as large as the size

Figure 2.9: Collaborative tanglible interface, combining virtual elements with real world annotations.Non-augmented view of collaborative AR interface (left), augmented view of collaborative AR interface (right) [41].

of the display of a computer, leading to different means of interaction. These means of interaction are described, in the context of MAR in Interaction Techniques for MAR Systems.

Interface Design in AR: AR has a 3D interaction space, meaning that its interface objects are located in 3D space, unlike computer GUIs. While the user interaction space, in an AR setting, is the user's surroundings, a vision-based AR system only shows its field of view, so some interaction objects can be occluded in a given view of the AR interface. A solution to this problem is the indication via graphical elements of where the object is relating to the field of view, as seen in Figure 2.10.

Vision-based AR systems can use TUIs. Poupyrev et al. described a collaborative AR interface using fiducial markers, or 'tiles', as interactive interface elements. Figure 2.9 shows the augmentation of the scene by comparing the non-augmented view and the augmented view. The system uses different markers, each with different augmented interface elements associated with them.

2.3.2 Interaction Modalities

The way a user provides a system with input and processes the given output can be categorized through a decomposition of basic senses, as defined by Aristotle in his essays *De Anima* and *Parva Naturalia*, these being vision, audition, touch, olfaction and gustation. The way a computer perceives the world can also be classified using a similar decomposition of the senses, albeit with the notion that the human sensory system is more advanced than its computer counterpart. In the same way that we experience acceleration forces or balance, a computer system can do the same with accelerometers and gyroscopes, for example [2].

Thus, interactions between a human and a computer system can be modeled to be interactions between systems which possess the five senses. Interactions between systems

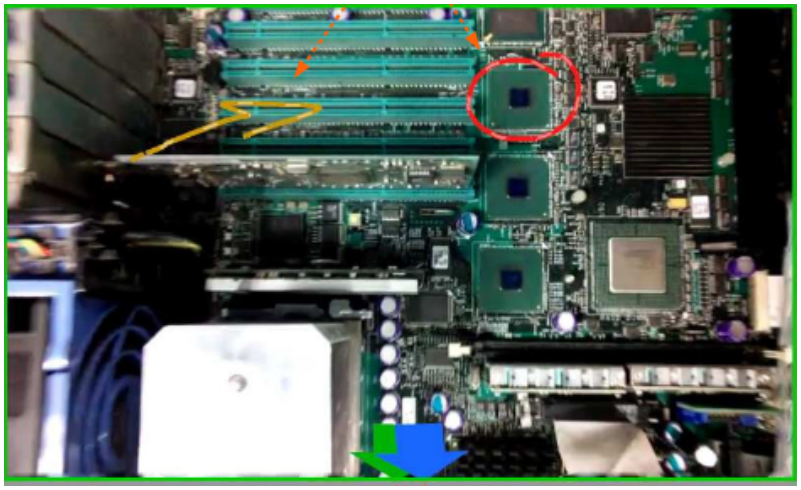


Figure 2.10: View of a PC motherboard with AR annotations. Arrows point to annotations outside the current field of view [19].

consist in the relation between a output provided by one system, and the input sense of another system, which consumes the output [2]. An example of an interaction viewed through this lens could be a person talking to another person; here, the interaction is determined as the relation between the output- the sounds produced by the person talking, and the input- the sense of audition allowing the other person to hear.

For AR interaction system design, only a subset of the senses, composed of vision and touch, is required, and other senses, such as kinaesthetics and audition are often used. An AR system needs to be both an vision input (through its camera) and output (through its display) provider and a touch input provider through its input devices (touchscreen, touchpads, buttons). It can also be a audio output provider to actualize audio feedback. An AR system can also implement kinaesthetics input to be able to use sensor-based tracking and interaction approaches.

2.3.3 Interaction Techniques for MAR Systems

Benoit et al. introduced the definition of gestures as input, whether they apply to a 2D medium or a 3D one, as one of the modalities of user input in 1999 [6], which are represented in Figure 2.11. As of this year of 2023, this definition continues to be useful to classify methods of user input, specially in the realm of AR.

In the context of MAR, this definition can be used to classify three gesture-based modes of interaction with a VE[21]:

- 2D Touch-based Gestures
- 3D Motion Gestures
- 3D Mid-air Gestures

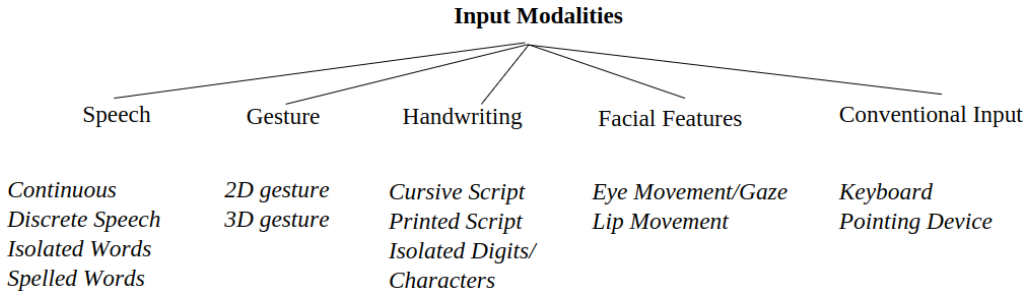


Figure 2.11: Input modalities [6].

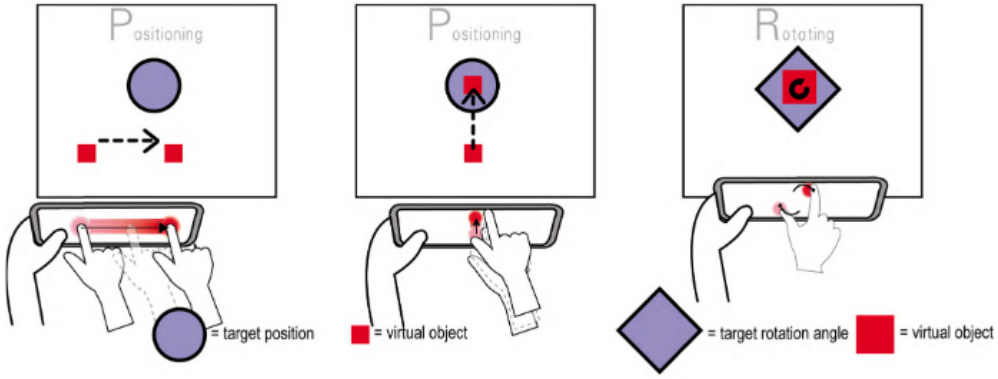


Figure 2.12: Graphical representation of object manipulation with surface gestures in a MAR application [21].

Touch-based gestures. Sometimes referred as surface gestures, these methods are a common way to define interactions in applications for smartphones and tablets, as well as other devices that include touchscreens or touchpads (Section 2.1.2).

They consist of 2D gestures drawn with one or more fingers over a touchable surface. The most commonly used ones for mobile non-AR applications are one finger swipes and two finger pinches. While these commonly used gestures can be used to create robust interactive systems, the fact that they are limited in number conditions AR applications to have an increased number of graphical widgets to allow for the increased number of interactions some AR systems possess, such as 3D object manipulation, which occupy screen real-estate [38]. Figure 2.13 depicts a subset of surface gestures. Screen real-estate is an important commodity in the development of UIs for AR applications, since a clear view of the camera viewport is needed to interact with the virtual environment effectively.

It's important to note that when manipulating 3D virtual objects in Six Degrees of Freedom (6DOF), a 2D input isn't enough to represent 6 possible manipulation controls: A 2D screen can only represent manipulations on two axis based on a single finger swipe, leaving four axis unrepresented. Multi-touch inputs can be used to fill in the remainder, for example using two finger pinches to move the object forward or backward, or swiveling two fingers in opposite directions to represent rotation in a single axis. A possible touch-based interaction framework for 6DOF object manipulation can be seen in Figure 2.14 [21].

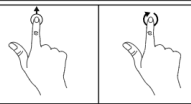
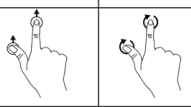
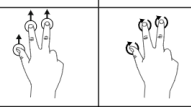
Constraint	FREE				ANCHORED			
Reference	EXTERNAL		INTERNAL		EXTERNAL		INTERNAL	
Shape	LINEAR	CIRCULAR	LINEAR	CIRCULAR	LINEAR	CIRCULAR	LINEAR	CIRCULAR
1 CP								
2 CP								
3 CP								
4 CP	...	...	...	...				
5 CP	...	...	...	...				

Figure 2.13: Breakdown of a subset of multi-finger gestures [38].

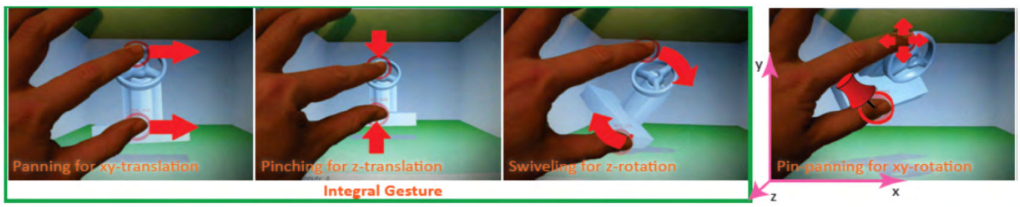


Figure 2.14: Touch-based interactions for 6DOF 3D object manipulation [31]

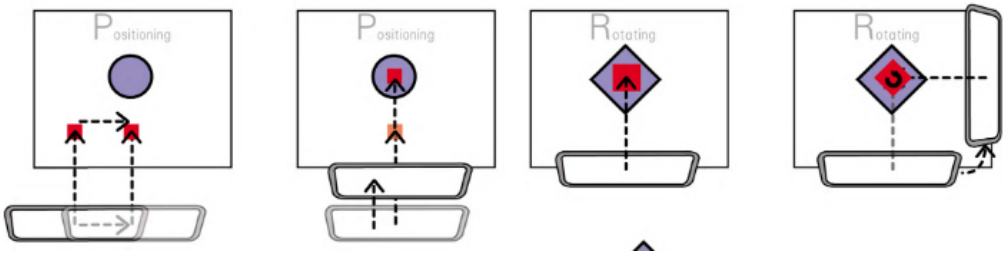


Figure 2.15: Graphical representation of 3D object manipulation through motion gestures in a MAR application [21].

Motion Gestures. An alternative to the previously explored touch gestures is motion gestures. Motion gestures refer to 3D gestures that are registered using the mobile device's Inertial Measurement Unit (IMU) sensors.

The added spatial dimension of this family of interaction methods allows for a breadth of unique gestures, such as rotating the Augmented Reality device, moving it forward, backwards and side-to-side, that can allow for more engaging interaction metaphors. This interaction paradigm was explored in the context of MAR by Dong et al. [14]. It was found that participants, who were asked to perform tasks such as manipulating virtual objects, either using surface gestures or motion gestures, were more engaged with the motion gestures, but thought that surface gestures were easier to use, as some participants claimed that they're more used to use surface gestures than motion gestures. A problem presented in this study with the motion gestures explored is the fact that movement of the device implicates movement of the display, since the devices used were smartphones,

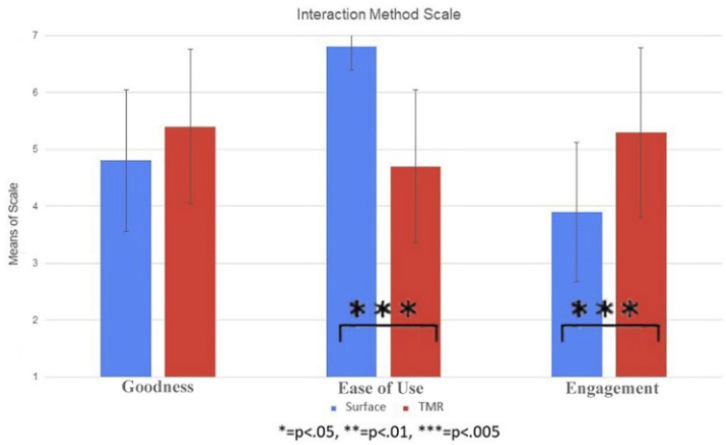


Figure 2.16: Bar graph depicting the evaluation of different qualitative metrics of motion gestures (TMR) and surface gestures (Surface), in a 7-point *Likert* scale [14].

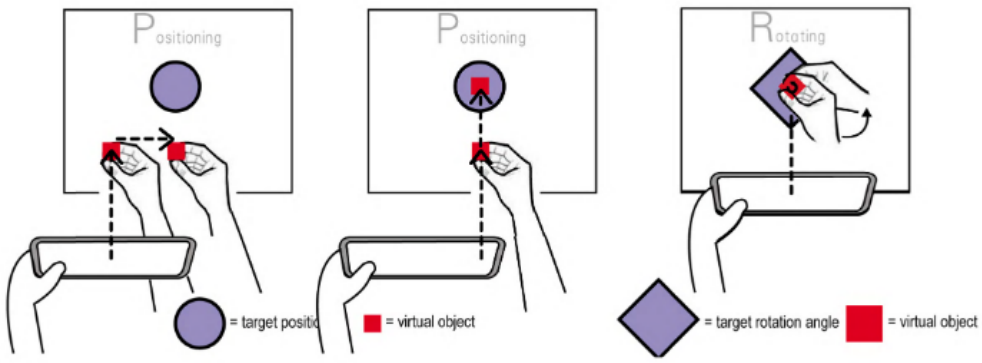


Figure 2.17: Graphical representation of 3D object manipulation using mid-air gestures in a MAR application [21].

which limits the visual feedback of users. The qualitative metrics that resulted from this user study are graphically represented in Figure 2.16.

Mid-air Gestures refer to gestures that occur mid-air, without any direct registering device. Mid-air gestures can be registered in a system by way of human motion recognition through cameras or motion sensors. In the context of Mobile Augmented Reality, this is usually done by holding the MAR device, such as a smartphone, in one hand, pointing its camera at other hand, thus capturing the action being done, or, while wearing an HMD, interacting with both hands. This interaction technique provides a very intuitive basis for interaction, since an interaction with a virtual object is the same as if the object was real, *sans* the tactile feedback.

The main problem of this gesture paradigm consists in the fact that, unlike the other methods presented, input recognition isn't trivial: touch-based gestures are registered in a touchscreen, in which the input can be formatted into a pair of coordinate values (or multiple pairs of these coordinate values, if accounting for multiple touches) and motion gestures are registered by parsing the concrete values of the velocity and acceleration

given by the IMU sensors. Mid-air gesture recognition achieved through computer vision infers hand and finger tracking, which has a high computational load coupled with problems of self-occlusion and inability to handle rapid hand movement [21].

2.4 Technologies

Currently, AR application development is aided by several different tools, frameworks and platforms. In this section, these technological tools will be explored, highlighting their capabilities in the context of AR development.

2.4.1 AR Frameworks

AR frameworks provide the necessary functionalities to make an Augmented Reality system, including keypoint detection algorithms for marker based approaches and SLAM algorithms for markerless ones, with some exceptions.

Some of the more prominent frameworks are:

- Vuforia Engine[7]

  Vuforia Engine is a closed source framework that allows for the development of marker-based and markerless AR systems. It is widely used and is compatible with iOS, Android and UWP (Universal Windows Platform) devices[8]. It also offers a Unity extension for development within the game engine.

  It implements "Extended Tracking"when tracking markers, which uses a device's pose (gathered from their IMU sensors) to estimate its relative position to a marker when it's out of view.

- ARCore[9]

  ARCore is a open-source AR framework developed by Google. It is compatible with Android versions over 7.0 (Nougat) and a subset of its features is also available in iOS, such as the Augmented Faces API, which recognizes face features and constructs a three dimensional mesh in which virtual objects can be mapped.

- ARKit[10]

  ARKit is a closed source AR framework developed by Apple for use in iOS devices. It features marker-based and markerless tracking as well as implementing various optimizations based on LiDAR (**Li**ght **D**etection **A**nd **R**anging) sensors present in some Apple products.[11]

[7] Vuforia Engine Homepage:https://developer.vuforia.com/ Last Accessed: 2023

[8] List of recommended target devices for Vuforia: https://library.vuforia.com/platform-support/recommended-devices. Last Accessed: 2023

[9] ARCore Homepage:https://arvr.google.com/arcore/ Last Accessed: 2023

[10] ARKit Homepage: https://developer.apple.com/augmented-reality/arkit/ Last Accessed: 2023

[11] Apple promotional article referencing LiDAR usage: https://www.apple.com/newsroom/2020/03/apple-unveils-new-ipad-pro-with-lidar-scanner-and-trackpad-support-in-ipados/ Last Accessed: 2023

Table 2.1: Breakdown of AR frameworks.

Name	Open-Source?	Free?	Markerless Tracking?	Target Platforms	Development Environments
Vuforia Engine	N	Y (free tier)	Y	Android, iOS, Windows	Windows, macOS
ARCore	Y	Y	Y	Android (some features on iOS)	Windows, Linux
ARKit	N	N	Y	iOS	macOS
ARToolKit (X/Plus)	Y	Y	N	Android, iOS	Windows, Linux
Wikitude	N	N	Y	Android, iOS	Windows

- ARToolKit(Plus/X)[12,13]

  ARToolKit was first developed in 1999.[14]. It allowed for the recogntion and tracking of fiducial markers and overlapping of virtual objects unto them. The original ARToolKit has since been discontinued and forked, originating ARToolKitPlus and ARToolKitX.

  ARToolKitPlus, much like the original, only allows for the development of marker-based AR systems, but provides a easier C++ API than its predecessor. It also has been deprecated.

  ARToolKitX was first made available as a open source project in 2018, following the gradual disinterest on the original ARToolKit and variants. It is the most updated 'branch' of the ARToolKit family. It also only allows for marker-based AR systems.

- Wikitude[15]

  Wikitude is a paid, closed source AR framework offering development solutions for marker-based and markerless AR systems. It has a considerable market share considering its competitors are free or have free tiers of their software. It features, much like Vuforia, an 'Extended Tracking' functionality. It allows for the development of AR systems utilizing AR Glasses, mainly the Snapdragon Spaces.

A breakdown of the important characteristics of each AR framework can be seen in Table 2.1.

[12]ARToolKit Plus Repository: https://github.com/paroj/artoolkitplus Last Accessed: 2023
[13]ARToolKitX Repository: https://github.com/artoolkitx/artoolkitx Last Accessed: 2023
[14]History of ARToolKit: http://www.hitl.washington.edu/artoolkit/documentation/history.htm Last Accessed: 2023
[15]Wikitude Homepage: https://www.wikitude.com/ Last Accessed: 2023

Table 2.2: AR development platforms. The modern target platforms mentioned are: Windows, macOS, Linux, Android and iOS.

Name	Category	Open-Source?	Graphical API Abstraction?	Target Platforms	Development Environments	Programming Language
Android Studio	Mobile Development	Y	N	Android	Windows, Linux, MacOS, ChromeOS	Java, C++, Kotlin
XCode	Mobile Development	N	N	iOS	macOS	Swift
Unity	Game Engine	N (some components are)	Y	modern target platforms	Windows, macOS, Linux	C#
Unreal Engine	Game Engine	N	Y	modern target platforms	Windows, macOS, Linux	C++
Godot Engine	Game Engine	Y	Y	modern target platforms	Windows, macOS, Linux, Android (Experimental)	C#, C++, GDScript

2.4.2 Development Platforms

The currently used development platforms for AR system development can be divided into two categories: mobile development platforms, used for the development of mobile applications, and game engines. A breakdown of the characteristics of each development platform can be seen in Table 2.2.

Mobile development platforms enable the development of MAR applications, which make up a large subset of AR applications. For *Android*, the most used development platform is Android Studio and, for iOS, the most used development platform is *XCode*. These platforms include emulation of their target devices, as well as streamlined GUI programming when compared to game engines.

These development platforms don't have any abstractions regarding graphical APIs, which hinders MAR development.

Game Engines: In the mid 1990s, games such as DOOM popularized a different architectural approach to game making than what had existed up until then: separating the more general/reusable aspects of a game, or the 'core': the rendering engine, collision systems and audio system from the more specific components: the game rules and art assets. This new approach allowed for faster and more efficient development of games,

since the 'core' could be ported over to other games, instead of being written from scratch, resulting in the game making process being faster [22].

This design philosophy is what defines a game engine. A game engine is a software suite which includes various tools to aid in game development as well as providing the 'core' functionalities: rendering, physics, audio rendering and input recognition.

The most prominent modern game engines, in a large majority, also include some kind of extended reality (XR) support built-in, which qualifies them as AR development platforms, with some specific advantages and disadvantages. The main advantages of using modern game engines for AR development are:

- Overlap of desirable features

  Game engines and AR systems have some common requirements. AR systems necessitate the rendering of virtual objects that occupy a VE, which is a feature that all game engines provide. For the inherent interactivity of AR systems, input recognition is essential, and all game engines offer it, for a variety of input devices. For collaborative augmented reality systems, most game engines include networking frameworks to synchronize each collaborator's view.

- Ease of use

  Game engines are streamlined for the creation of virtual environments and many of them offer project templates for XR development, which cut down development time. Rendering and graphics related code is also often abstracted, which makes XR development with minimal interaction with graphical APIs, such as OpenGL, possible.

Disadvantages of these systems include:

- Non-standard GUIs

  Most game engines don't include the option for the use of Android or iOS standard GUI elements, which results in more time dedicated to designing user interfaces, which themselves can work less effectively on their respective target devices.

- MAR Debugging

  Game engines that are not designed specifically for the development of mobile games (such as Unity, Unreal and Godot) don't offer easy ways to test on target devices. The most common way to debug mobile game builds is to emulate their displays on the development system, but this approach doesn't handle mobile interactions such as motion input. In Unity, it's possible to use a smartphone or tablet as a testing device by using the 'Unity Remote' app, which allows for more efficient debugging. 'Unity Remote' is limited by the fact that the target device must be connected to the development device by USB cable, which hinders MAR debugging specially, considering that user movement is limited by a physical connection.

Currently, the most prominent game engines are:

- Unreal Engine[16]

  Unreal Engine is a free 2D/3D game engine. It uses C++ as the main programming language and allows exporting to a large number of target platforms, including Android, iOS, Windows and the current generation of game consoles.

  In terms of AR, Unreal Engine has built-in tools that follow the OpenXR[17] standard. OpenXR is a royalty-free open standard that aims to simplify XR development by standardizing device-specific code, allowing for the creation of cross-platform XR applications.

- Unity[18] Unity is a game engine that offers 2D and 3D game development tools. It uses C# as the main programming language and supports development for an increased number of target platforms, such as Windows, iOS, Android, WebGL and the current generation of game consoles. It includes a user-generated asset store, which allows for the sharing of third-party assets, including but not exclusive to 3D models, 2D sprites, source code and plug-ins. It also includes a free version which doesn't restrict any game making features, but imposes an annual income restriction and forces a 'Made with Unity' screen to appear on game startup.

  - Networking capabilities

    Unity offers Netcode for GameObjects[19](formerly known as Unity MLAPI), a first-party open-source networking library targeted for the development of small-scale cooperative experiences. It is available from version 2020.3 onward. It provides profiling and monitoring tools, including runtime network monitoring.

    Outside the first-party sphere exist other networking frameworks, namely Photon PUN[20] and Mirror[21]. Photon PUN (**P**hoton **U**nity **N**etworking) is a closed source networking framework which offers dedicated Photon servers in the cloud to host your multiplayer app, but still allows for the hosting of user hosted Photon servers. It can be configured to support various Transport Layer Protocols (TLP).

    Mirror is a open-source networking library which started development as a fork of Unity's discontinued networking API, UNET. It originally used only TCP, but has since supported other TLPs, such as KCP, an UDP based TLP, which is now the default.

[16] Unreal Engine Homepage: https://www.unrealengine.com/en-US/ Last Accessed: 2023
[17] OpenXR Homepage: https://www.khronos.org/openxr/ Last Accessed: 2023
[18] Unity Homepage: https://unity.com/ Last Accessed: 2023
[19] Unity Netcode Homepage: https://unity.com/products/netcode Last Accessed: 2023
[20] Photon Pun Homepage: https://www.photonengine.com/en-US/PUN Last Accessed: 2023
[21] Mirror Homepage: https://mirror-networking.com/ Last Accessed: 2023

  - AR Support

    Unity supports AR through various plug-ins and first party implementations: Vuforia , ARCore and Wikitude offer Unity plug-ins, and Unity itself offers ARKit support through their 'ARKit XR Plug-in', which can be used in version 2019.4.15f1 and onward and it's part of 'AR Foundation', a cross-platform AR framework.

    While Unity does include OpenXR support, it doesn't include OpenXR's AR support.

- Godot Engine[22] Godot Engine is an open-source game engine which allows for the development of 2D and 3D games. It includes support for development in GDScript, a language specially developed for use in Godot which features 'Python-like scripting', C#, C++ and through visual scripting[23].

  In terms of AR support, Godot offers basic out-of-the-box AR/VR *nodes* which can be used to develop either marker-based or markerless AR games.

2.5 Summary

In this chapter, the main concepts of this thesis: AR, collaboration and HCI are explored, while mentioning the current technologies that support them.

Regarding AR, its place in the context of other reality technologies was described, as well as were its foundational concepts, this being tracking and interaction. Furthermore, the MAR paradigm and the devices that support it were mentioned. The applications of AR were explored in the context of different fields.

While exploring the concept of collaboration, a distinction was made between online/remote collaboration and in-person collaboration, and it was described how this distinction in setting creates system design challenges. Remote collaboration was explored in the context of how COVID-19 fundamentally changed the normal paradigm of collaboration from being in-person to being remote, and how AR systems were adapted accordingly. In-person collaboration was explored by describing its advantages, the main one being the physical proximity of the collaborators, which can originate collaborative single-device interfaces, which can support AR. These single-device interfaces were compared with multi-device interfaces in regards on how well they support collaborative AR.

The HCI section started by exploring a general taxonomy on how humans interact with computers, and how we can use this taxonomy to describe typical AR and MAR interactions. Further taxonomies were used to classify the three main paradigms of MAR

[22] Godot Engine Homepage: https://godotengine.org/en Last Accessed: 2023

[23] Godot features Webpage, mentioning programming language support:https://godotengine.org/features Last Accessed: 2023

gesture interaction, these being surface gestures, motion gestures and mid-air gestures. The advantages and disadvantages of each paradigm were explored.

Finally, the current technologies supporting these three core concepts were mentioned: AR frameworks and development platforms (mobile development platforms and game engines). The differences between mobile development platforms and game engines were explored.

In the next chapter, we considered topics explored here to design a MAR collaborative, multimedia-focused system focusing on a marker-based, in-person environment.

3

Collaborative Interaction Design

To address the research questions described in section 1.2 and as a part of the thesis objectives mentioned in section 1.3, a set of design principles were formulated for collaborative, multimedia-focused, MAR systems. In this chapter, we discuss the requirements, characteristics, and interaction means for this type of systems. Additionally, we explore the system's characteristics in the context of three use cases, analyzing how can the system's features be applicable in each one of them.

3.1 User Requirements

To address the proposed research questions and objectives, we outline the following requirements for the interactions users can do in this type of system :

- Users should interact through smartphones.
- Users should be able to view multiple multimedia assets through an AR VE.
- Users' actions that manipulate the multimedia content should be synchronous.
- Users should be able to share specific multimedia assets with other users.
- Users should be able to create and join collaborative sessions without knowing computer networking concepts.

The specification to run on mobile devices, more specifically, smartphones, is derived from the motivation presented in section 1.1. Smartphones are widely used for the sharing of multimedia, but given a large enough group of people, the process of in-person sharing and collaboration becomes exceedingly challenging. This is evident in Figure 3.1, which presents how the capability of users to see the multimedia in a smartphone decreases with how many users there are. Similarly, the difficulty a single user experiences when trying to manipulate the multimedia content is positively correlated with the number of collaborators. Thus, developing a MAR system that provides multimedia sharing would allow us to analyze AR's capacity to provide a solution to problems of this nature. Notably,

Figure 3.1: The spatial distribution of users when looking at the same smartphone. On the left, only two users are in collaboration, while on the right there are five.

AR allows us to transform, in this context, a single-device collaborative effort into a multi-device one. Furthermore, we can better leverage the different means of user input that smartphones provide for implementing AR interactions when they are being used by a single user.

The VE allows the placement of multiple virtual objects, and, in the context of multimedia organization and sharing, the simultaneous display of multiple multimedia assets. As such, the VE allows users to organize multiple multimedia assets. In order to simplify collaboration in this context, these organizational actions should be synchronized among users, since different states for each collaborator would hinder the user's perception of their state in relation to their collaborators'. The difference of states, in this case, would be further worsened by the real-time nature of AR interaction.

Users can use this type of system to share and organize multimedia assets with other users. We derived these types of interaction from the motivation present in section 1.1. Since this motivation represents the possible users as anyone who wants to share multimedia assets with other people, we considered that they shouldn't need prior networking knowledge to use the system. Since this type of system is, by nature, multi-device and collaborative, thus involving networking, we need to abstract the concept of connecting to and hosting networked sessions. We created the concept of collaborative sessions to achieve this goal.

3.2 System Requirements

From the previously described user requirements, we derived the following requirements for this type of system's design:

- The system is a synchronous multi-user system, meant for in-person use.
- The system is a marker-based MAR system.
- The system allows for the display, organization, spatial manipulation, and sharing of photos in an AR VE.
- The system abstracts any networking concept from its users, allowing people to connect through the AR marker.

For a system to be collaborative, it should allow a group of users the ability to make changes to content and to perceive each other's changes. In the case of an AR system, the content that users are altering is the VE. As such, collaboration in a real-time context, with multiple people manipulating a shared VE, as is the case with AR, should try to achieve synchrony. To explore how users would interact directly with each other when using this type of system, these can be designed with a focus on in-person use. This, however, indirectly limits the number of collaborators who can contribute simultaneously to small groups. Marker-based AR is used to allow in-person usage with a specific anchor point, which aids in a user's ability to localize their collaborators in virtual space.

To be able to explore how different types of AR interaction can be used in the context of multimedia, a system's set of functionalities should be designed to acknowledge the spatial dimensionality of their multimedia content types. For example, in this case, where the multimedia assets are two-dimensional (2D) images, some spatial transformations, such as user-driven rotation, can be limited without hindering the interactive experience.

Since the users of this type of system (described in the use cases in section 3.4) are not required to know any computer networking concepts, using a streamlined process for creating and joining synchronous collaborative sessions is important. In non-AR collaborative applications, networking can be abstracted via unique codes for open collaborative sessions, or URLs that a session's host can create and send to other collaborators. Since we have established this type of system to be a marker-based, in-person AR system, we can use the common anchor point of a session, the marker, as a way to connect to it. This pairs any marker with the network identification of a session, allowing, via an external entity that stores all of these pairings, for a client to connect to a session simply by scanning it.

3.3 Virtual Environment Design

The AR VE consists of all visible virtual elements that are superimposed onto the real-world view. The design of these elements can help a user's understanding of the elements' functions, as well as help communicate the system's state.

3.3.1 Feedback Mechanisms

VE implementations should provide ample feedback for a user's actions, as well as their collaborators. This feedback is essential for being able to interpret other users' actions

and their effect on the VE's state. The relation a user forms mentally between an action and its results can be reinforced by triggering the same feedback mechanisms whether the user or any other collaborators do it. An example would be: in a collaborative jigsaw puzzle game, when users select a piece to move it, that piece "floats" above the puzzle. If a user has previously moved a piece, they would know that the triggered feedback mechanism is related to this action. If that same user observes a piece "floating" without their input, that user would logically conclude that another user is triggering that feedback mechanism, meaning that they are moving the piece. As such, for operations on the VE, such as manipulating the position of a virtual element, the resulting feedback should be synchronous. This feedback can also be part of the VE itself, which is especially useful to indicate spatial interactions.

In collaborative real-time systems, where other people can affect its state at any time, the VE should clearly communicate the current actions being taken by other users, as well as provide indicators as to what other users may do. In the case of multimodal interfaces, these indicators can show what specific actions a user can do based on their current mode. The indicators can also be predictive, hinting towards possible user intention, an example being showing where a user is looking, which can indicate an intention to act on the virtual elements present in that location. Avatars provide a way to encapsulate these indicators in a single virtual element.

3.3.2 Avatars

Avatars are virtual representations of a user in a VE and, in their design, they should communicate what a user is doing currently and what can be done to interact with them. In the case of a symmetric interaction set for all collaborators, avatars also help users learn the system's functions. For example, if a user sees other users' avatars, which resemble hands, that user would interpret that their avatar also resembles a hand, and, consequently, the interaction set available to them is similar to that of hands, which is typically grabbing and pointing. This concept relates to the theory of affordances as described by Gibson [20].

Avatars should be able to be used to identify their respective user. In an AR in-person context, this can be uniquely done by relating them to the position of their user in the real world.

3.3.3 Spatial Arrangement

When developing multimedia-focused AR applications, developers need to take into account how the multimedia assets are displayed in the screen, as well as how they are spatially arranged. The design decisions involved in this process can heavily influence the user experience. For example, if a set of multimedia assets is arranged in the VE such that a single asset is placed in the center of the arrangement and is larger than the other ones, a user will first focus on the larger, centered element. As such, if an AR

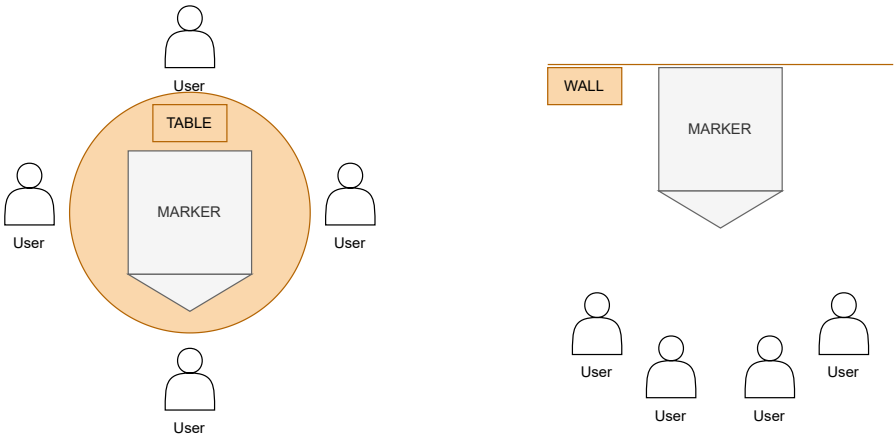


Figure 3.2: User placement depending on marker placement. On the left, users are arranged when the marker is on a table. On the right, users are arranged when the marker is on a wall.

application doesn't want to communicate the relative importance of a multimedia asset, it should opt for a neutral spatial arrangement. This approach should be taken in the case of an application's initial state, but should not be kept throughout the course of a collaborative session. Users should be able to use spatial manipulation of multimedia assets to communicate the relative importance of elements.

3.3.4 Guidelines for VE Design in Marker-based AR

In a marker-based application, where the gaze of all collaborators should be directed at the marker to be able to engage with the AR system, it's important to not place VE elements far away from the marker. In purely marker-based applications, this results in loss of tracking, and, consequently, loss of functionality. Also, in hybrid marker-based approaches, long periods of time without tracking could lead to an accumulation of position offset errors, which hinders AR interactivity and collaboration. VE element arrangements which force a user to view the marker at a grazing angle also cause loss of tracking. If a collaborative AR application's focus is 2D multimedia assets, such as images and videos, their placement in the VE should take into account the user's viewing angle. These types of multimedia typically are observed with more quality when viewed head-on, as opposed to at a grazing angle. As such, these assets should be placed so that their optimal viewing angle corresponds with the marker's optimal viewing angle.

In a synchronous setting, it is impossible for 2D assets to have synchronized positions for all users where the viewing quality is equal for all users. To solve this, the synchronous requirement can be relaxed, by not synchronizing the 2D asset's rotation, thus allowing each photo to face each user in their local view. The marker's location in the real world can also mitigate this problem. If a marker is placed on a horizontal surface, such as a table, users will arrange themselves to give all other users space to operate, resulting in circling the marker. However, if a marker is placed on a vertical surface, such as a wall,

users will form a semicircle facing the marker. This phenomenon, which is displayed in Figure 3.2, results in users having a larger difference between their viewing angle and the marker's direction in the horizontal marker placement. As such, if a collaborative multimedia AR application depends on the marker's direction, to maximize the quality of viewing angle, the marker should be placed on a vertical surface.

3.3.5 Multimedia Representation for Multiple Types

When designing how the multimedia assets appear in an AR VE, the developer should consider what are the multimedia types their system should implement. In systems that only focus on a single multimedia type, this representation can just be the multimedia content. For example, if a system only uses images, the image's representation can be just the image itself. However, if multiple multimedia types are included in a single system, representing them as their content can lead to different types having, at a glance, the same representation. For example, in a VE composed of a paused video and an image, a user cannot, at a glance, differentiate between the types of the presented multimedia content if the virtual representations of the multimedia assets are just their content. As such, additional information needs to be contained in the representations to inform users of their associated multimedia types. The previous situation could be solved by adding a pause symbol atop the paused video, for example.

3.4 Use Cases

In this section, we explore three use cases and analyze the interaction requirements of each one. These use cases are: casual photo album sharing, concept board creation for design fields, and photography categorization and selection for social media marketing team workflows.

3.4.1 Photo Album Sharing

Since users can see and manipulate a set of multimedia assets, which may include images, this type of system can be seen as a virtual, AR counterpart to a photo album. As such, one of the possible use cases is mimicking a photo album in function, allowing various collaborators to see a set of photos prepared in advance by other users. Simply by having the capacity to see the set of photos prepared by their fellow collaborators, the users engage in a sharing activity similar to real-life photo album presentations.

This type of system can also be used for the collaborative creation of a photo album, where photos may be, in advance, organized spatially, mimicking how photos are organized on photo album pages. In order to achieve distinguishability between the pages, this type of system can implement mechanisms that enforce a certain degree of visual separation between sets of elements.

3.4.2 Concept Board Creation

In the context of design fields, concept boards are graphical representations of concepts that follow two characteristics: multiple visual elements are spatially organized, and these elements can be annotated to provide auxiliary information. These characteristics help convey the board's main idea. Concept boards are typically made in mediums that facilitate the manipulation of assets and annotations, such as whiteboards, whether physical or virtual, as is the case with *Miro*[1]. One of the reasons for these mediums' continued usage in design fields is the ease of collaboration that they provide, which can help with brainstorming and concept refinement through the feedback of the collaborators.

Concept boards are achieved by representing their ideas through text and drawings and relating these ideas through their spatial layout and visual indicators, such as arrows. Considering the primary focus of the solution being multimedia collaborative manipulation, its feature set doesn't need to explicitly include the placement and use of these visual indicators, since its AR nature allows the real-world placement of these elements to be seen, which can itself enhance the virtual scene. This case is evident when using the solution to augment an already existing physical concept board, present on a writable surface. Placing the marker on top of this surface would allow the superimposition of the VE, and, consequently, the placement of virtual images on the concept board, which is useful if details of these images are important support for the concept. It also provides more versatility regarding image display, since physical photos need to be printed to be included in a fully physical concept board, while virtual ones can be easily replaced. In the making of concept boards, this type of system provides spatial manipulation and the ability to view images in detail.

Although the domain of Figure 3.3 is the subjective comparative analysis of different photos, an activity comparable to image categorization and selection (section 3.4.3), we can see how writable surfaces can work in tandem with AR VEs to allow for multimedia collaborative action.

3.4.3 Image Categorization and Selection

Multimedia, mainly images and videos, has become an essential part of modern social media. Social media have, in turn, become platforms for not only the online representation of individuals but also brands and organizations. As such, social media presence is increasingly important for these brands, which invest heavily in social media marketing. In conclusion, due to social media, teams that handle large amounts of multimedia assets are being formed. For these teams, collaboration is essential, since the quantity of produced content, which needs to fit a specific branding and design language, is increasing.

In this context, collaboration is achieved by group decisions regarding how and what multimedia assets should be used. For example, after a sports match, teams typically

[1] Miro Website: https://miro.com/index/ Last Accessed: 2023

Figure 3.3: AR-enhanced photo organization. The VE is superimposed onto a whiteboard, which allows for the drawing of elements.

post on social media regarding the match. In multimedia-centric social media, such as Instagram, this post is typically paired with a photo of the event, of which there is a high amount. As such, social media teams need to choose a subset of these photos to be paired with the post. The problem increases in complexity when considering that a single event can equate to various posts. As such, a categorization effort should be made to address this problem.

The collaborative solution allows for the categorization and selection of photographs via its organization interactions. In this context, the collaborators place the set of photos in the AR scene and begin to categorize them. In this step, the workload can be parallelized by assigning a subset of photos to each collaborator for categorization. This categorization can be done explicitly with folders or implicitly with spatial organization. The latter approach has the advantage of allowing a collaborator to define their own organization system, such as creating a spatial hierarchy where photos at the top are considered more suitable than others. Organization systems can be complemented through a physical medium, as is the case in Figure 3.3. Since collaborators share the workspace, they can get feedback on the suitability of specific photos during the process, increasing productivity. After categorization efforts, the specific photos for each post can be voted on by the group via sharing.

Table 3.1: Feature usability in each use case. 0 - Not useful, 1 - Useful, 2 - Most useful.

Features / Use cases	Sharing with specific collaborator	Presenting	View In Detail	Create/Storing in Folder	Moving photos
Photo Album	2	2	1	1	1
Concept Board	1	1	1	0	2
Categorization	1	0	2	1	1

3.4.4 Interaction types

From the three use cases, we compiled the following set of common types of interaction based on the use cases' characteristics:

- Sharing photos with a specific collaborator
- Presenting photos to all collaborators
- Viewing photos in detail
- Storing photos in folders
- Moving photos in the VE

Table 3.1 summarizes the applicability of the types of interactions for each use case. Detailed descriptions of each type of interaction can be found in section 3.5.

Since the discussed use cases are photo-centric, we decided that, for the implementation of the system, the system would also be photo-centric. Mainly, the implemented solution focuses on the photo categorization and the photo album sharing use cases.

3.5 Taxonomy of Multimedia Collaboration in AR

In this section, we propose a taxonomy for the interactions present in multimedia-focused, collaborative AR systems. The interaction types are divided into three classes: multimedia-specific interaction types, which act on the characteristics of the multimedia itself, organizational interaction types, which allow users to form an implicit or explicit form of organization, and sharing interaction types, which allow users to share multimedia assets with specific users or all users present in the session. A graphical representation of the taxonomy can be seen in Figure 3.4.

3.5.1 Multimedia-Specific Interaction Types

Multimedia-specific interactions types acknowledge the characteristics of multimedia content that make it different from other forms of virtual information. This class of interactions includes types of interaction exclusive to multimedia manipulation and displaying. Since different multimedia types have different methods of manipulation and displaying, the number of interactions present in this category increases according to the number of multimedia types represented. In this class, there's only one interaction, due to the domain being only images, which is viewing an image in detail.

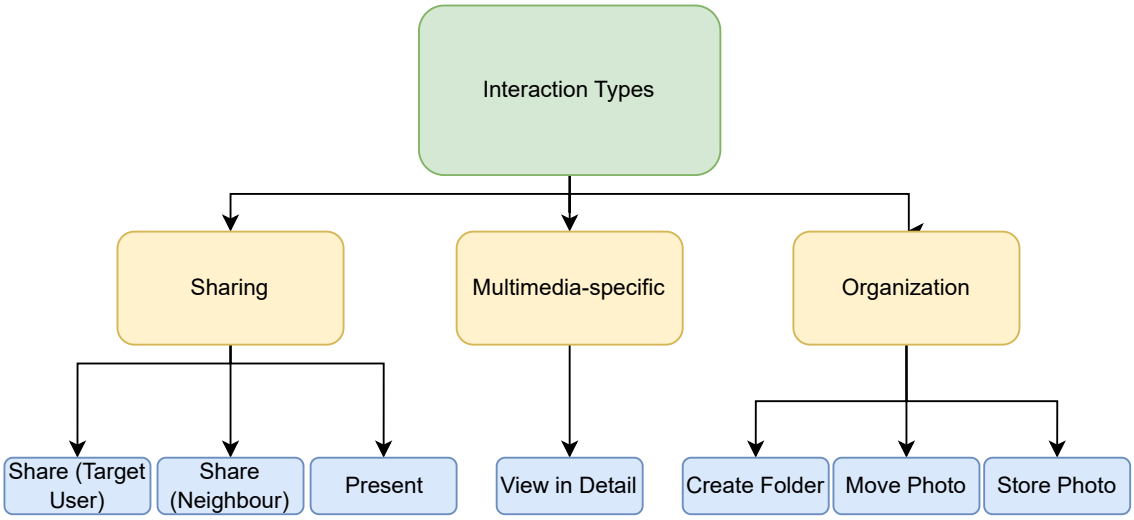


Figure 3.4: Proposed collaborative AR interaction taxonomy. Classes of interactions are pictured in yellow and interactions for photo-centric systems are pictured in blue.

Typically, to see an image in detail in an AR environment, a user moves the image closer to them or moves closer to the image. This approach doesn't let other users participate, since the examining user either has the image close to them, and in turn, farther from the other users, or they are occupying the location that would allow other users to examine the image. Essentially, this is the same problem that happens when multiple users try to see multimedia content on a single smartphone. To bypass this situation, viewing an image in detail should be a specific interaction in AR collaborative image-centric systems, which does not affect other user's views. Viewing an image in detail should provide mechanisms to examine an image's details, as well as remove any possible distractions from the examining action. In a collaborative context, viewing an image in detail can even be synchronized between users, resulting in the possibility of collaborative analysis, if coupled with an annotation toolset.

3.5.2 Organizational Interaction Types

Organizational interaction types refer to user actions that can be used to impose an organizational order in the VE. In multimedia-focused applications, organizing different multimedia assets is often done to categorize and relate multimedia assets. To this effect, we propose that AR collaborative applications should provide users two ways of organizing multimedia objects: implicitly, via moving the objects present in the scene, or explicitly, via a folder system. Having both types of organization allows users to easily express a specific organization system. Folders allow the forming of an explicit barrier of items inside the folder and outside the folder. Also, folders can be used to visually hide their contents to minimize visual clutter in the scene. The former functionality expresses the capacity for relating concepts, while the latter lets users solve categorization problems better. A folder being able to hide its contents also lends to the possibility of

Table 3.2: Organization types and their characteristics.

Organization type	Visual separation between organized structures?	Is scalable with Nº of images?	Spatial layout manipulation?	Can hide items?
Spatial Organization	✗	✗	✓	✗
Folders	✓	✓	✗	✓

Table 3.3: Sharing methods characteristics.

Sharing Method	Nº of receiving users	Nº of photos shared	Needs identification?	Depends on user position?	Interrupts target user's activities?
Through Neighbour	1	1	✗	✓	✓
Through Avatar	1	1	✓	✗	✓
Presenting	All	1*	✗	✗	✗

*Only a single photo can be presented at a time. If a user presents a photo while another is being presented, that one is overwritten.

being used as a filtering tool. Table 3.2 presents the two different types of organization systems and the characteristics that distinguish them. In the context of marker-based AR applications, it's important to note that their VEs' space is limited, which can indirectly limit the number of spatial organization layouts that can fit in the VE.

3.5.3 Sharing Interaction Types

Sharing interaction types are actions that provide users with ways to "give" things to other users. Considering that all collaborators have the same goal, the concept of ownership doesn't need to exist in a collaborative system, making the concept of "giving" obsolete as well. As such, we can redefine sharing, not as a transfer in ownership, but as pointing the attention of other users towards a specific element in the scene. This definition of sharing is only possible if the "shareable" objects are always visible in the scene.

We can classify sharing interactions according to the number of users they affect: interactions that highlight elements to all users in the collaborative session and interactions that highlight elements to a specific user. Users, in the context of being possible sharing targets, are characterized by their visual representations and by their physical, real-world position relative to the marker. Via the marker, we can determine the direction a user is in relation to any other user. Given, that information, we can determine the closest users, or neighbours, of a user. For effective use of this interaction, we can take advantage of the in-person setting, which allows users to easily verify who is on their left and right, for example. Each sharing interaction and its characteristics are presented in Table 3.3.

3.6 Summary

In this chapter, we explored the main considerations that should be taken into account when designing collaborative multimedia-focused MAR systems. These considerations

were consolidated into an interaction set which was categorized, forming a taxonomy of collaborative MAR interactions.

In the following chapter, we provide insight into how we implemented an application that leverages the characteristics of the collaborative solution discussed so far, as well as provide insight into some implementation alternatives that were prototyped during the development of this dissertation.

4

MARC.MD Implementation

In this chapter, we discuss the architecture and technologies that support our implementation of the collaborative solution described in the previous chapter, exploring in detail its interface and the implemented interactions. This implementation uses images as its singular multimedia type, as the use cases that incited it, described in section 3.4, were photo-centric. We also explore different types of prototypes of the collaborative platform, highlighting various conclusions about the design of collaborative, multimedia-focused MAR applications that were taken along its development cycle.

4.1 Application Architecture

Considering that the domain of the application is multimedia, collaboration and MAR, we compartmentalized its various basic functionalities, discussed in section 3.1 into modules, in order to simplify the development process. The organization of these modules is given in Figure 4.1. The modules are:

The AR module, which includes all the logic regarding the registration and recognition of the markers. This logic, in turn, uses the device's camera feed together with its IMU sensors to accomplish marker-based tracking.

The networking module, which provides the synchrony requirement mentioned in section 3.2. This module is explained in-depth in section 4.1.2.

The internet request module, which communicates with a multimedia auxiliary server to provide the multimedia assets present in the collaborative sessions. This module is also explained in-depth in section 4.1.2.

The mobile interaction module, which leverages the user input methods of touchscreen and IMU sensors to allow the registration of surface and motion gestures, respectively. These gestures are the basis for the application's interaction means.

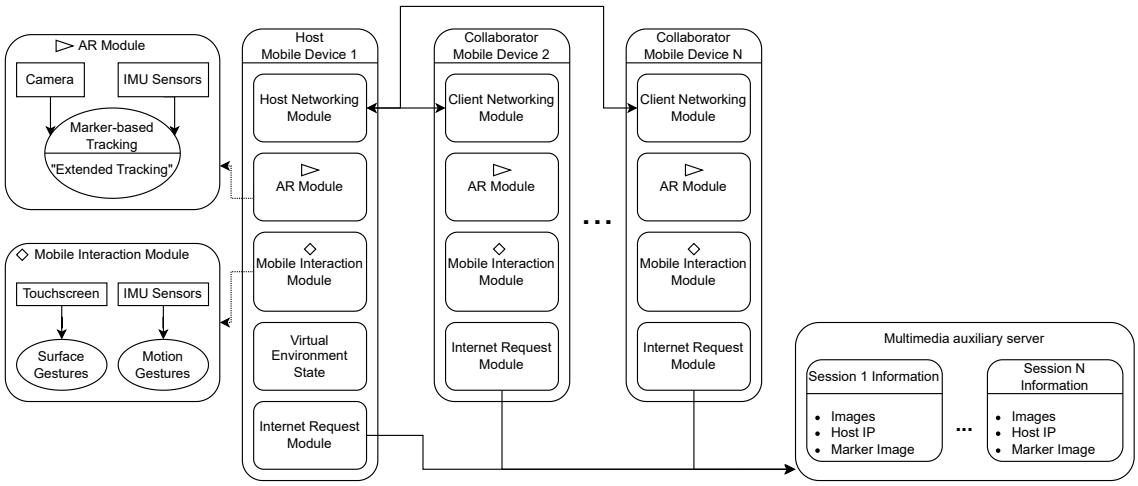


Figure 4.1: Diagram depicting the implemented application's architecture.

4.1.1 Technologies

To implement the application, we chose a game engine that provides fast prototyping and is designed for three-dimensional applications. Also considering the mobile platform requirement, we chose *Unity3D* as the game engine for our application. *Unity3D* simplifies the mobile development process by implementing "patching", which only deploys the changes since the last present build in the target device, which accelerates the building process. In turn, this provides a tighter feedback loop, which helps in prototype iteration and testing. Since the application has AR, testing in the target devices is essential to evaluate the quality of the marker tracking in realistic scenarios, as well as touchscreen actions that can't be easily simulated, such as multitouch interactions.

In the AR module, we used *Unity3D*'s *Vuforia* implementation, as it provides marker tracking with support for "Extended Tracking", which allows the visualization of the VE after losing view of the marker. This effectively increases the interaction area and adds the possibility of seeing virtual objects located outside the marker plane, which can be leveraged to implement avatars that follow a device's position, as mentioned in section 3.3.2. This tracking technique also mitigates the effects of loss of tracking, which can happen when looking at the marker from grazing angles, improving the user experience. Figure 4.2 presents a demonstration of this tracking technology in effect when tracking a VE with multiple images spread along the marker plane. We see that, even if the marker isn't visible by the device, the VE is still visible and its correct place relative to the marker.

4.1.2 Network Architecture

Synchronous collaboration among various devices implies the need for networking. To allow for effective collaboration in a synchronous VE, its state needs to be shared on all

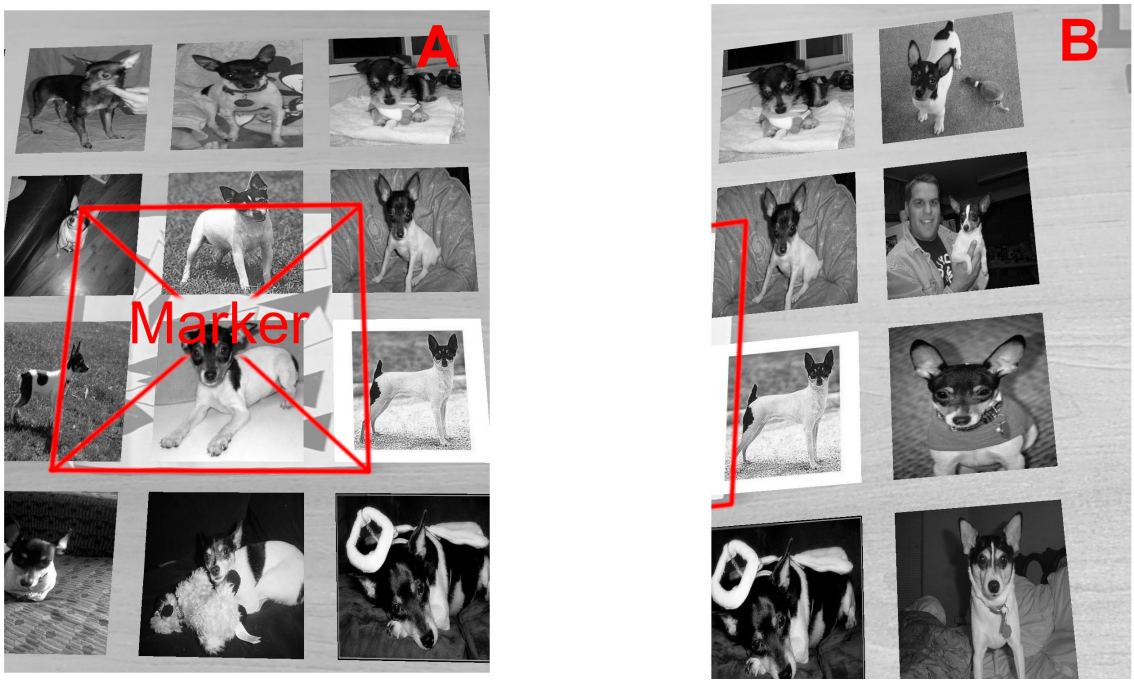


Figure 4.2: Demonstration of "Extended Tracking". Pictured in grayscale to increase visibility.The marker's location is outlined in red. In A, marker is completely in view, and in B, only a section of the marker can be seen in the left of the frame.

collaborators' devices. Given that our system implementation focuses on photos, in a single collaborative session, the state that needs to be communicated between collaborators is:

- The position of the photos.
- The multimedia content associated with the photos.
- The position of all users' mobile devices relative to the marker.

Within a single session, the state is being updated in real-time, thus creating a need to minimize the quantity of data passed through the different collaborators. The architecture that was chosen for this effect is a client-host model with an auxiliary multimedia server, which implements a REST architecture. A host keeps the VE's state, with the multimedia content being represented as URLs to the auxiliary server, which hosts all multimedia content. This approach effectively eliminates the need for the host to constantly send multimedia content, which is much larger than the other state components, by assigning the client's multimedia transfer operations to the auxiliary multimedia server, instead of the host. In this situation, the host only needs to communicate the URLs that correspond with the photos present in the scene, leading to the host having to send less data over the network to the clients.

The multimedia auxiliary server can host the information of various collaborative sessions, given the host's IP address and an identifying factor. This identifying factor is an

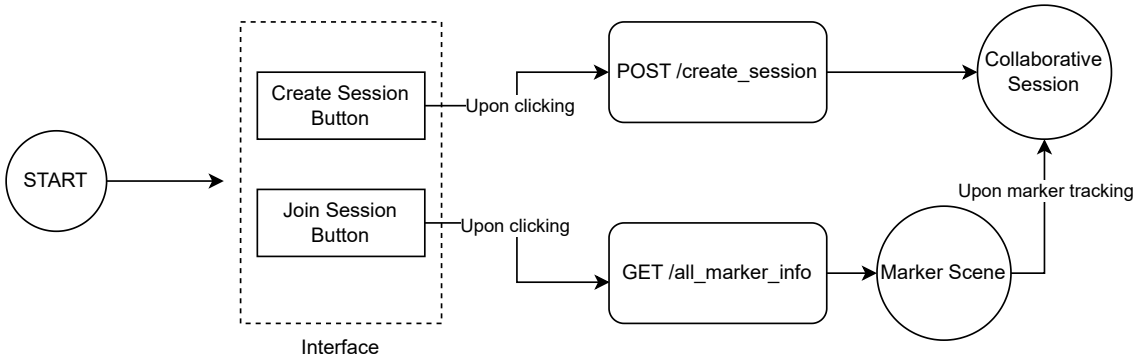


Figure 4.3: Flowchart depicting the process of connecting to or starting a new session.

abstraction that the users interact with, instead of the host's IP. Since the AR tracking implemented in the applications is *Extended Tracking*, which uses a marker for tracking, we can also use that marker as the identifying factor. Thus, creating a collaborative session involves the host's application sending the host's local IP address to the multimedia auxiliary server, paired with the desired AR marker for the session. Consequently, connecting to a session corresponds to downloading all IP-marker pairs from the auxiliary server, instantiating those markers in a *Unity3D* scene, and upon marker registration, using that host's IP address to connect to the session. This approach leads to a user experience that doesn't have either the host or the clients directly interact with IP addresses, effectively substituting their functionality with the session's marker.

For in-session networking, we use *Netcode for GameObjects*, a first-party networking framework for *Unity3D* which supports the client-host model. It provides useful network programming abstractions, such as *NetworkVariables*, which allow for the synchronization of individual variables between all connected clients and the host, and *NetworkTransforms*, which synchronize the positions of their respective virtual elements. It also provides runtime network component removal, which allows for temporary local-only object manipulations. Although *Netcode for GameObjects* is, by default, server authoritative, meaning that only the server can manipulate the synchronized state, some objects can be made into "Player Objects", which allows for their respective player to manipulate their state in a client-authoritative way. This feature simplifies the process of updating the state of user avatars, which can directly rely on that user device's position and direction, as mentioned in section 3.3.2.

In order to differentiate between users who wish to create a new collaborative session and users who wish to join an already existing one, the application's initial interface is a menu that presents that choice.

Figure 4.3 presents a simplified, schematized form of how the application allows users to create or join collaborative sessions.

Users who want to create a session send a POST request to the multimedia auxiliary server to start a new collaborative session. In the latest version of the application, the

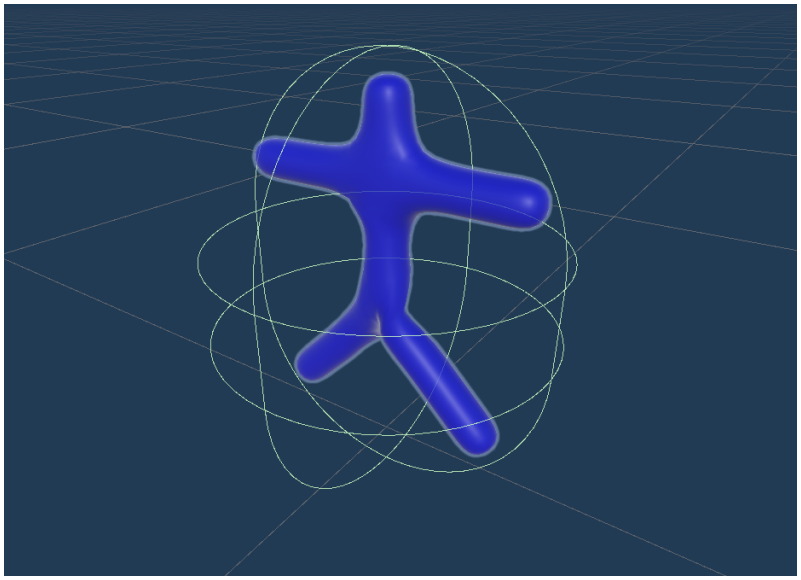


Figure 4.4: A user's anthropomorphic avatar in *Unity3D*'s Editor. Avatar's *hitbox* is depicted in green.

markers associated with the collaborative sessions and their multimedia content are hard-coded. When the collaborative session is created, the user automatically joins it as its host.

If a user chooses to join an already existing session, it sends a GET request to the multimedia auxiliary server for all the active IP-marker pairings. Upon receiving them, for each pairing, an AR marker is instantiated such that, when it is tracked, it connects the user to the corresponding host IP address.

4.1.3 AR and Avatars

Within a collaborative session, users should be able to view the effects of other users' actions. Although the application implements the interaction requirement of actions that change the state of multimedia assets being synchronous, this by itself doesn't allow users to determine the author of an action. As explained in section 3.3.2, avatars are virtual representations of users that consolidate feedback mechanisms. In this context, since avatars are specific to one user, they can be used to identify the author of an action.

In our application, avatars are distinguishable by a unique color, assigned when joining the collaborative session, based on a unique *clientID* initialized upon connecting. However, this isn't enough for users to be able to identify the author of actions, since users have no way of knowing which color corresponds to what user. For this purpose, each user has two avatars associated with them: one that follows the smartphone's position, and one that follows where that user is pointing towards in the VE. The smartphone-following avatar, which we named "phone avatar", allows users to identify what color a user is by looking at that user's smartphone position, which is known in an in-person setting. Still, users will typically not move their device's camera away from the marker,

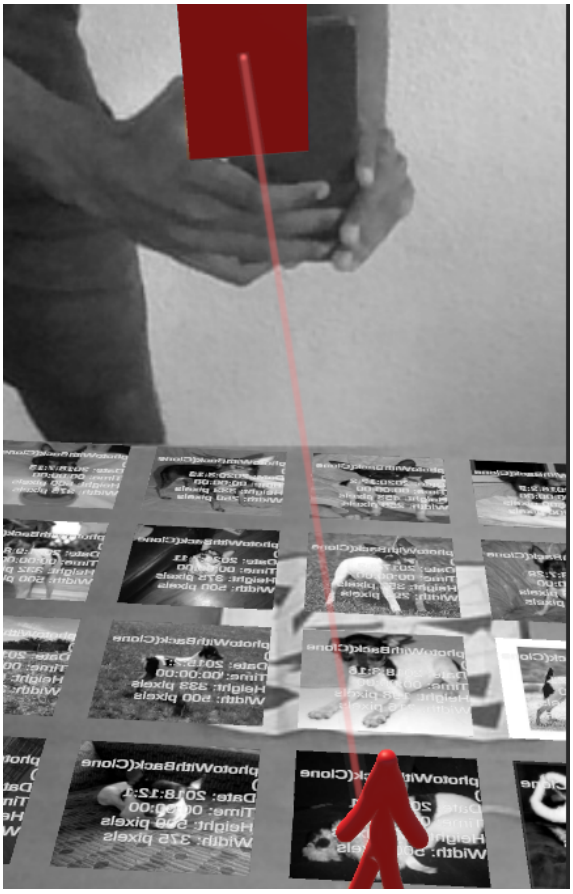


Figure 4.5: A user's avatars present in the application. Elements without relevance are in grayscale to increase visibility.

as explained in section 3.3.4. Consequently, it is possible that users do not see this avatar when interacting with the application. As such, the gaze-following avatar, presented in Figure 4.4, which we named "anthropomorphic avatar" (due to its design), takes the possible tendency of users to focus solely on the marker's close surroundings into account. As it follows where the user is pointing in the VE, the anthropomorphic avatar is always in a location where other users are likely to look. By relating these two avatars visually, via linking them with a line, we ensure that users know their collaborators' colors since they can visually follow the line from the anthropomorphic avatar back to its respective phone avatar, and, consequently, their collaborator's smartphone. Figure 4.5 displays this visual relation between the two avatars.

4.1.4 Photo Virtual Environment

The VE consists of a configurable number of photos placed on a plane that's parallel to the marker. By placing the photos in this parallel plane, the users, to view the photos at their optimal viewing angle, coincidently look at the marker in its optimal viewing angle, as explained in section 3.3.4. These photos are initially arranged in a grid formation, similar to how photos are organized in smartphones' photo gallery applications. This

initial state provides a clear view of all photos in a neutral arrangement, as mentioned in section 3.3.3. The VE's area is determined by the distance where a smartphone, situated at a distance of 1.5 meters from the marker, can still track the marker.

Since the application's only multimedia type is images, we can represent these images with plane meshes with the photo texture applied, as explained in section 3.3.5.

To initialize the VE, the host first requests the collaborative session's set of photos from the multimedia auxiliary server, and, for each photo, creates a photo object with that image as its texture. Each of these objects also permanently stores its image's URL in a *NetworkVariable* to be able to synchronize their state with late-joining clients. These clients, upon sensing a *NetworkVariable* change in value, automatically update the respective object's texture. The image downloads are handled asynchronously, such that a user can still interact with a scene if the multimedia auxiliary server doesn't respond to a specific request.

After their creation and URL assignment, the image objects are organized in the grid layout, according to the configured number when the host created the session.

4.2 Interface Architecture

The application implements the full interaction set defined in section 3.5. These interaction types can be decomposed into a set of input parameters, and an output system state. For example, the interaction type "Moving Photo" implies two input parameters: a photo and a location where that photo will be moved, and results in a system state where that photo was moved to the location.

The VE's state, dubbed as system state in this section, is defined by a set of users, the collaborators, a set of images, the photographs, and the presented image, used for the "Presenting" interaction type, explained in-depth in section 4.3.4. Users are composed of their position in the VE, the image they have currently selected, and their folder. An image is characterized by its texture and position in space. As such, the system's state and its components can be represented by:

- $\mathtt{SystemState} = \mathtt{Set}\langle\mathtt{User}\rangle \times \mathtt{Set}\langle\mathtt{Image}\rangle \times \mathtt{Image}$
- $\mathtt{User} = \mathtt{Position} \times \mathtt{Image} \times \mathtt{Folder}$
- $\mathtt{Folder} = \mathtt{Set}\langle\mathtt{Image}\rangle \times \mathtt{Position}$
- $\mathtt{Image} = \mathtt{Position} \times \mathtt{Texture}$

In the process of implementing the interaction types, knowing their "interaction signatures" is essential to organizing them into an interface. This implementation uses the following "interaction signatures" for the taxonomy's interaction set:

- $\textsc{ViewInDetail}(\textsc{myUser}, \textsc{image}) \rightarrow \textsc{dtlContext}$,

  $\mathtt{dtlContext} = \{\mathtt{zoom}, \mathtt{pan}, \mathtt{exit}\}$

- Present(targetImage,systemState)→ $systemState_p$,
  $SystemState_p$ = {SystemState.image = targetImage}
- ShareWithNeighbour(myUser,image,direction,systemState)→ $systemState_{sn}$,
  $SystemState_{sn}$ = {myUser.image = null, getNeighbour(myUser, direction).image = image}
- ShareWithUser(myUser,image,targetUser,SystemState)→ $SystemState_{su}$,
  $SystemState_{sn}$ = {myUser.image = null, targetUser.image = image}
- PlaceFolder(myUser,targetLocation,SystemState)→ $SystemState_f$,
  $SystemState_f$ = {myUser.folder.position = targetLocation}
- MovePhoto(myUser,image,targetLocation,SystemState)→ $SystemState_m$,
  $SystemState_m$ = {image.position = targetLocation}
- StorePhoto(image,targetFolder,SystemState)→ $SystemState_{st}$,
  $SystemState_{st}$ = {targetFolder.images = [..., image]}

The View in Detail operation is the only operation that forces a specific state that isn't based on the previous state. The dtlContext contains three operations: zooming in and out, panning the image, and exiting the mode, which resets the interface to its previous state. This operation set allows the user to analyze a photo's details in a separate context from the VE, which leverages the advantages of minimizing distractions, as mentioned in section 3.5.1. All the other interaction types only modify the current state of the session: sharing operations let users highlight an object for other users, and organizational operations affect the target object's position, and visibility, in the specific case of storing photos.

From this set of signatures, we can see that some of the parameters required for the application's interaction types are not directly obtainable through this interaction set. To fetch these parameters, we implemented another set of interaction types:

- getImage(position)→ Image
- getUser(avatarPosition)→ User
- getFolder(position)→ Folder

Since all the interactive objects (images, avatars, folders) are present in the VE, they must have a position associated with them. As such, we can spatially select them through their positions. In this implementation, there are various ways to select a position, which are used in different contexts.

After analyzing all the interactions, we can categorize them as part of two main groups: interactions that depend on having an image, and those that don't. We can further define the image-depending interactions as either being spatial, meaning that they depend on

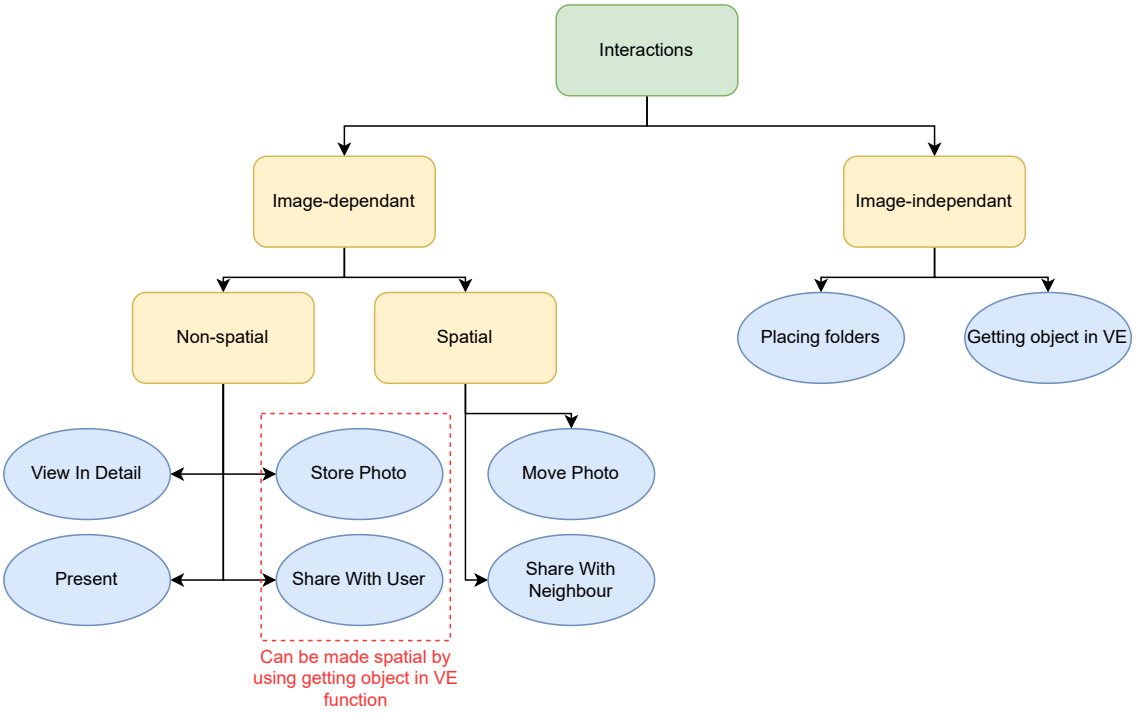


Figure 4.6: Application's interaction taxonomy of image and position dependency.

one or more positions in the VE to work, or being non-spatial. Figure 4.6 schematizes these categories and conveys the possibility of making certain non-spatial actions spatial by using the fetching interactions. In our application, we converted the storePhoto and shareWithUser interactions to be made spatial, thus making them depend on a folder's position and an avatar's position, respectively. This was done to standardize operations that depend on an object in the VE, making it easier for a user to interpret the interface. We can relate this taxonomy with the interaction taxonomy defined in Figure 3.4 by classifying all organizational interactions as spatial and all multimedia-specific and sharing interactions as image-dependant.

4.3 Interaction Type Design In Multimodal Interfaces

The application organizes its different interaction types through a multimodal interface. A multimodal interface is an interface that has a state used to determine what actions can be done at a specific moment. Each state typically has a specific UI to communicate the available actions to the user. Due to the small available space for UI that MAR platforms offer and to the differing contexts in which the interactions occur, the application implements a multimodal interface, dividing the essential UI elements across multiple modes. Each mode's UI is represented in Figure 4.7.

Multimodal interfaces require actions that transition between the different modes. To facilitate its usage, we equate each mode with a context, which relates to the categories presented in Figure 4.6. As such, the resulting multimodal interface is composed of four

Figure 4.7: Modal interfaces, with labels on each screenshot. From left to right: the detail interface, the general interface, the spatial interface and the object interface.

modes:

The general mode, which encompasses the image-independent interactions- selecting an object (image or folder) and manipulating folders.

The detail mode, which represents the separate context in which users can do detailed image analysis through zooming in or out, and panning the image.

The spatial mode, which contains all the image-dependent spatial interactions, except sharing with neighbours: sharing image with a user, moving an object and storing an image in a folder.

The object mode, which contains the non-spatial, image-dependent interactions (presenting and enabling detail mode), as well as sharing with neighbours.

From these modes, we can define transitional actions that switch between them and their contexts. The main transitional action is selecting an image, which switches the context to be image-dependent. There are, however, two modes that operate in image-dependent contexts: the spatial mode and the object mode. As such, there needs to be two different input methods for selecting an image. Also, these methods need to be simple, since selecting an image is essential to be able to access the rest of the application's features. The input methods chosen for selecting an object are: tapping the screen where an image is located, and if it is a long press, switching to object mode, otherwise, switch to spatial mode. This method of selection was chosen in part to take advantage of smartphone users' expectations of quick and long presses- quick presses normally determine the default action, such as opening an app when clicking its icon, while long presses tend to represent a set of other actions. Since we considered the spatial mode to be the most

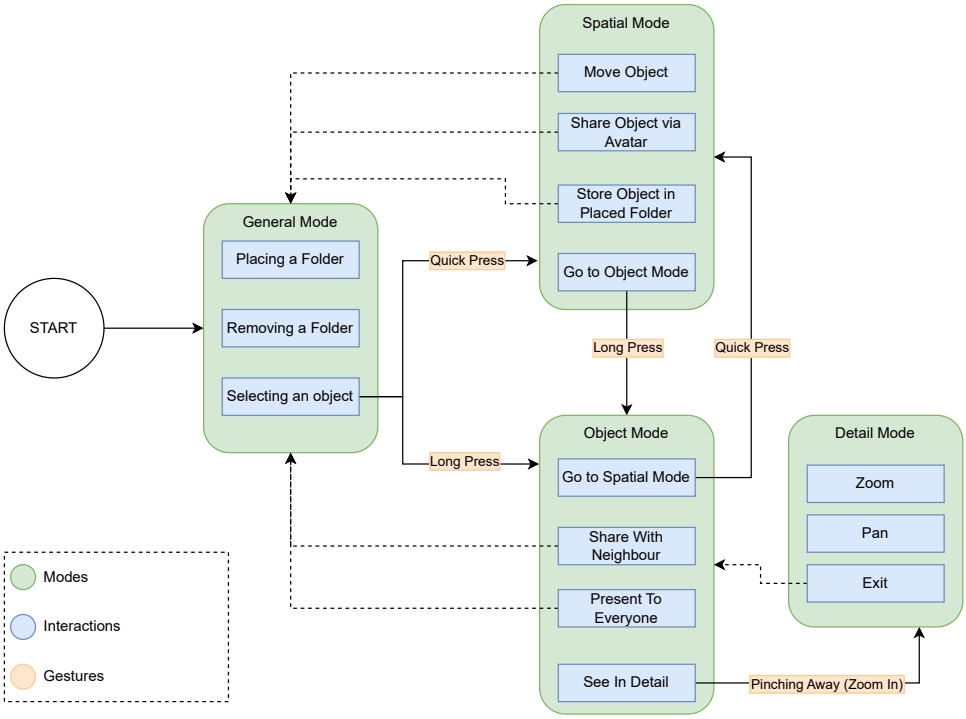


Figure 4.8: Interface mode flowchart.

important mode of the two, we assigned it to the quick press selection method. It's also possible to switch between these modes by doing their respective input methods while in the other mode.

Another transitional action is going into detail mode, which can be done through the object mode. For this transition, we again took advantage of smartphone users' expectations when using image visualization applications, by making its input method the gesture of pinching two fingers away from each other. This gesture is commonly used to symbolize the action of zooming in, which is a gesture users do when they want to see an image in detail. Fundamentally, this gesture carries an expectation that aligns with the purpose of viewing an image in detail.

Figure 4.8 represents the application's interactions, their respective modes, and the actions that transition between the modes. This approach allows for minimal UI, which results in a clearer view of the VE, and, consequently, better usage of spatial interactions.

The following sections describe the UI of their respective interface mode and provide the implementation details for each of that interface mode's interaction types.

4.3.1 Detail Mode

The detail mode UI consists of a black, opaque background that obscures the AR scene, the target of analysis, which is an image, and an exit button. Figure 4.9 shows this interface with added annotations. The image is placed slightly below the center of the screen to allow users to more easily use their thumbs when interacting with it. Panning

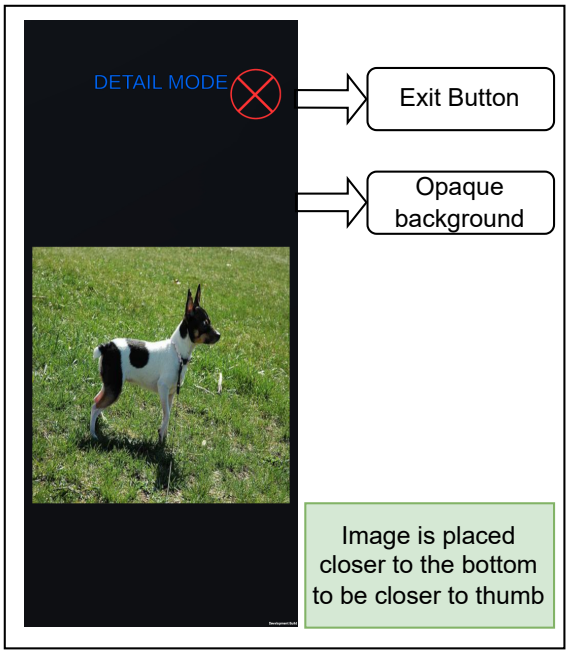


Figure 4.9: Detail mode UI with annotated relevant elements.

only has an effect on the position of the image if the resulting movement doesn't leave the image outside the viewport. The opaque background serves as an implementation of the concepts explored in section 3.5.1 and contributes to communicating that the current context is separate from the VE, and, in turn, includes different interaction types.

This interface mode includes the same interaction set that photo visualization applications typically have: **zooming and panning**. These features are implemented with the same gestures typically used in these applications: pinching two fingers away for zooming in, pinching two fingers together for zooming out and single finger dragging for panning. In this interface, there is also an exit button, which returns the user to the object mode interface.

4.3.2 General Mode

General mode's UI is characterized by its main action- selecting images. This action is only done effectively when the user executing it has a clear view of the VE. As such, the UI for this mode is minimal, only including the UI needed for its other action- placing folders in the VE. Figure 4.10 shows the interface users see when in general mode.

Selecting photos, as previously mentioned in section 4.3, is a transitional interaction that either switches the mode of interaction to the spatial mode or object mode. It is done by either quick tapping on a photo to go into spatial mode or long pressing it to go into object mode. When selecting a photo, whether it's to transition into spatial or object mode, the object is elevated vertically. This type of feedback accomplishes the goal of increasing its visibility, which helps the other users distinguish the selected object

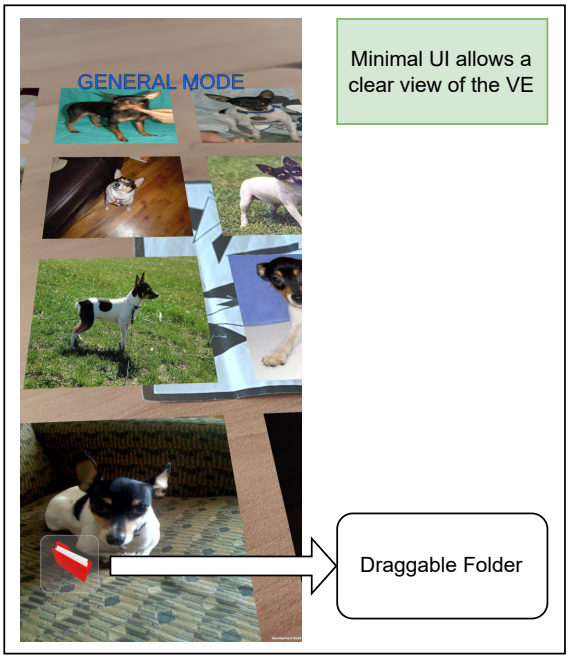


Figure 4.10: General mode UI with annotated relevant elements.

from the unselected ones, contributing to the feedback mechanism concepts presented in section 3.3.1. However, when moving the photo, this vertical elevation doesn't allow the controlling user to see the object's location when it is placed. To this effect, when a photo is subject to this vertical elevation in the context of a spatial mode transition, a helper object, which allows users to know the target position of the photo in the marker plane, is instantiated. In object mode, this vertical elevation is aided by a small amplitude bobbing movement. This movement helps in drawing attention to the selected object. This additional feedback mechanism is derived from the concepts discussed in section 3.3.1 pertaining to the ability of users to infer another user's current interface mode simply by looking at whether the elevated photo is bobbing or not.

Placing folders can be done by dragging them from the UI element present in this mode's UI. After dragging them completely out of the UI element, the folder follows the dragging finger and is elevated with the same elevation offset present in photo selection. The placing process is also helped by the same helper object present in photo selection and it is concluded when the user lifts the dragging finger. After this, the UI element changes color to reinforce the absence of the folder in it, and, consequently, the folder's presence in the VE. Then, there are two methods for removing the folder from the VE: inverting the motion that was done to place the folder, meaning dragging the folder from its current position in the VE to the UI element, or clicking the empty UI element. To be able to do the inverted placing motion for removing, the object needs to be moved in the VE through dragging, an input type that typically starts with a long press. As such, this movement implementation doesn't correspond with the quick pressing needed to

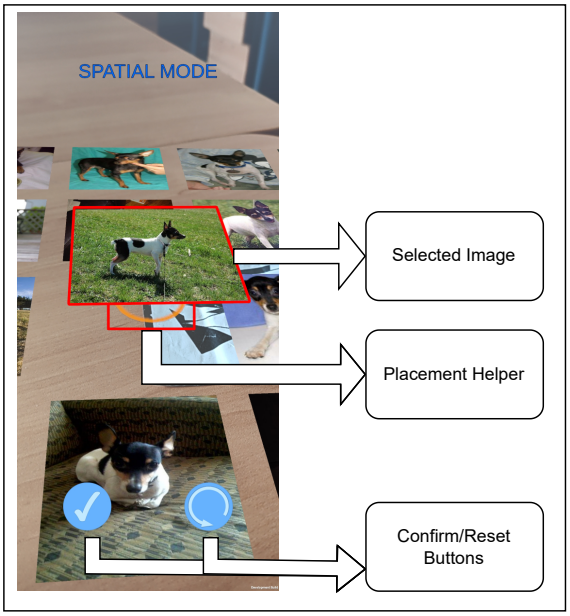


Figure 4.11: Spatial mode UI with annotated relevant elements.

normally access the spatial manipulation features, but we considered that the inverted action was intuitive enough to warrant this trade-off. Placed folders can be opened and closed by quick tapping them, toggling between the two states. The properties of closed and open folders were derived from the discussion in section 3.5.2, which suggests that having these two folder states can be used as a way for minimizing visual clutter, which can be useful if the configurable number of photos in the application is too high. Closed folders are displayed similarly to their representation in the UI element, while open folders are represented by a semi-transparent plane in which the photos are arranged automatically, according to the order they were stored in. This plane, when instantiated, moves away the photos that would occupy its space in order to draw attention to the folder's opening, as well as increase clarity as to what photos are in the folder.

4.3.3 Spatial Mode

Spatial mode comprises the context of moving a photo. Similarly to selecting images, users should have a clear view of the VE while moving photos. The UI is composed of two buttons, one to confirm the movement and one to reset the photo to its previous position. With these buttons, the process of moving photos doesn't require any fingers to be on-screen, which contributes to the clarity of the view.

Moving photos is the interaction that a user is automatically placed in when selecting a photo and transitioning to this mode. First, the photo is moved towards the center of the screen, fixing it there, and allowing the user to move it across the marker plane by pointing their device at the target location. This classifies the input method for moving photos as a motion gesture. As previously stated, the way for putting the object down is

through two buttons: one that sets it in the marker plane in the original position it was taken from, and one that sets it where it is currently, minus the elevation offset carried from the photo selection. The reset button is needed since the object, in the moment of selection, is instantly moved to the center of the user's screen to allow for the motion gesture method. Without having the reset button, a user could not easily move the object to its original location.

Storing photos and sharing through avatars are implemented identically in order to standardize the notion of interacting with user-related objects. Since avatars serve as a representation of users within the application, they can be used to provide an interface for interacting with their respective users, as explained in section 3.3.2. When a user is moving an image, by pointing their device towards an avatar or a folder, the image snaps atop the hovered avatar or folder. This feedback signals that confirming the move by using the previously mentioned button will store or share the image, depending on the hovered object. After confirming the move, in the case of sharing with an avatar, the photo returns to its original position in the VE, and the receiving user receives a notification. This notification presents the user with the choice to either view the received image in detail, or to disregard it. According to section 3.5.3, we can classify this method of sharing as being targetted and virtual representation driven. In the case of storing the image in a folder, if the folder is closed, the image disappears from the VE, as it is now part of the closed folder's contents. However, if the folder is open, the image is instead moved from the outside to the inside of the folder, and the folder's layout is updated accordingly.

4.3.4 Object Mode

The object mode interface features a UI where the selected image is represented in a UI element in the center of the screen- the "pointer", and all interactions except viewing in detail are accessed by dragging this UI element in different directions. The actions are displayed in the form of a radial menu. Sharing with neighbours is accomplished by moving the "pointer" left or right, depending on the neighbour's side. Presenting is done by moving the "pointer" up. Figure 4.12 shows the UI with their most relevant elements highlighted through annotations.

Presenting allows users to share a single image with everyone in the session. When the action is selected in the UI, a copy of the selected photo is created and gradually moved from the location of the original to the center of the marker. This copy has the added property of its rotation not being synchronized in the VE, allowing every client to change it locally, making it possible to set the image to face them individually, independently of their viewing angle, as explained in section 3.3.4.

Sharing with neighbours , upon its completion, makes the target neighbour receive the same notification from sharing with avatars, but, in this case, the image moves across

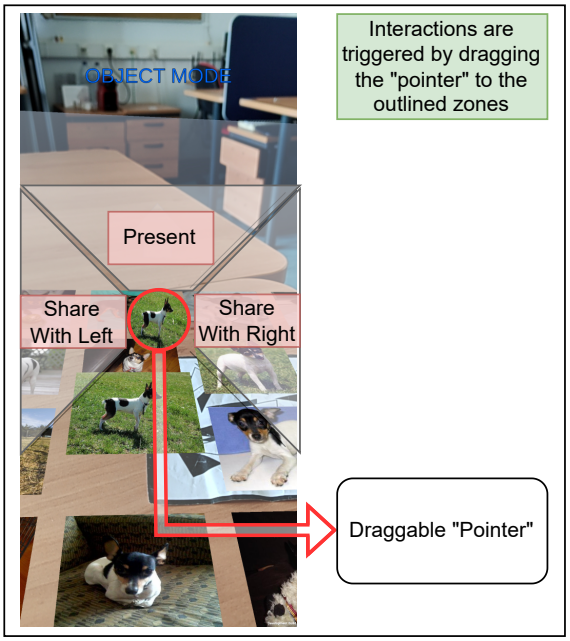


Figure 4.12: Object mode UI with annotated relevant elements.

the screen from the side that sent it. For example, if a user receives an image from the person on their right, the photo will horizontally move from the right side of the screen to the middle.

Going into detail mode is done, as previously mentioned, by doing the zoom in gesture, defined by pinching away two fingers.

4.3.5 Interface Summary

As previously mentioned, smartphones and tablets offer various input methods that can be used to trigger the application's interaction types. We established the specific control schema defined in the final version of the application after testing different input and action mappings, a process that was accelerated by separating the action code from the input method code, simplifying the act of changing mappings. This also simplifies the process of implementing composite input actions, which use two or more separate input methods, such as "Moving Photos", which uses motion gestures to define the target location and finger taps in specific areas in the UI to confirm the action. Table 4.1 summarizes the interaction type implementations discussed in this section.

4.4 Interface Prototype Evolution

We implemented the application and its interface iteratively to test the different aspects presented in the requirements. This iterative process consisted of developing a subset of the final features, followed by a small-scale user testing phase, with one to two users

Table 4.1: Interaction types and respective user inputs.

Interaction	Mode	User input	Resulting Mode	Side-effects
Selecting Object (Spatial)	General	Quick press on image	Spatial Mode	Image follows the center of the screen
Selecting Object (Object)	General	Long press on image	Object Mode	Image bobs above the marker plane
Placing Folders	General	Start drag on folder icon	General	Folder is placed where drag ends
Removing Folders	General	Drag placed folder onto icon	General	Folder is removed from the scene, maintaining its contents
Sharing with Avatars/ Storing in Folders	Spatial	Aiming the held image to hover an avatar or folder	General	Image is either shared with respective user or stored in folder
Presenting	Object	Swiping up and releasing	General	Image copy is made and placed atop the marker
Sharing with Neighbours	Object	Swiping left or right, depending on target neighbour	General	Image is shared with target neighbour
Viewing In Detail	Object	Pinching two fingers away	Detail	None

presenting feedback specific to that prototype's purpose. This method of prototyping results in many iterations over the same concept, but we will focus on three distinct phases of prototyping which yielded interesting conclusions: the conceptual phase, the spatial manipulation phase and the collaboration phase. Table 4.2 provides a summary of the different phases.

4.4.1 Conceptual Prototype

We developed prototypes to see how applicable certain design principles are for AR applications. The main tested design principle was the concept of the design of an interactive object communicating its means of interaction to a user. This concept was applied in the context of a multi-multimedia type application, where it is essential to communicate the specific type of a multimedia asset through its design, as explained in section 3.3.5. The scope of this prototype was collaborative spatial manipulation and organization of various types of multimedia, consisting of audio, 3D models, videos, text, and images. As such, for the representation of these elements to work in tandem, there should be some design abstraction to communicate that these elements can be spatially manipulated in the same way, even though they are different types of multimedia.

This prototype focused mainly on two aspects: the design of the different multimedia elements and how to organize them on an AR VE. Different types of multimedia have different visualization methods: audio files can be seen as a waveform, images are seen as themselves, and three-dimensional models can be seen in wireframe or with shaders. These visualization methods have different visual footprints on a scene as well, such as

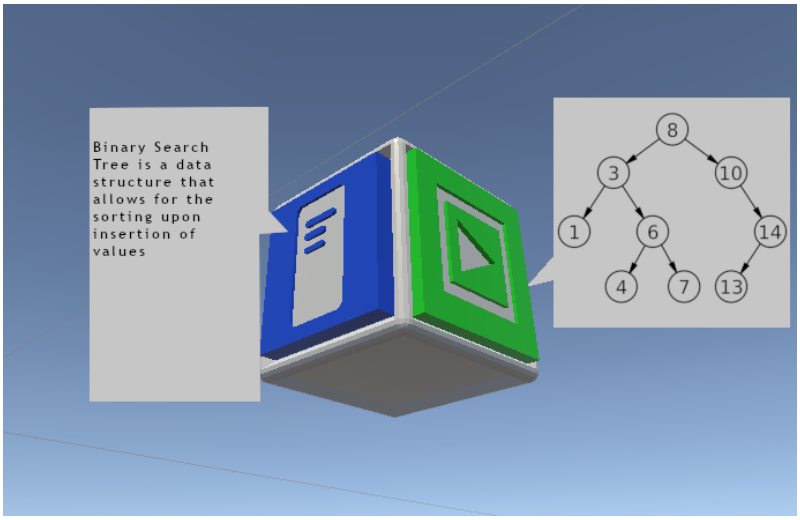


Figure 4.13: Folder representation in conceptual prototype holding a textual asset and an image asset.

images and videos being two-dimensional, while three-dimensional models occupy an extra dimension. For all of these multimedia types to be able to be interpreted by a user as having the same set of interactions, these objects need to be abstracted to a simpler, unified representation. In this prototype, it was defined that 3D icons succeeded at representing these different multimedia types. Through their design, they communicated their multimedia type, as well as the actions that are allowed to be done to them, consisting of being moved across the VE. A disadvantage of using representative icons is not being able to see their multimedia content at a glance, requiring a specific action to display their multimedia content.

This prototype also focused on spatial organization through explicit mechanisms, namely folders. Folders are useful in a context where multiple types of multimedia are present because they allow users to relate different multimedia types to each other, which can be used to represent concepts better than a single type of multimedia would. While folders can be designed to look like real-life folders, or other storage methods, resulting in a visual metaphor, folders in this prototype were designed to resemble an object where the 3D icons specifically were meant to be stored. This design resulted in a cube, where each face could be used to store a single multimedia asset. This design accomplished in communicating its purpose clearly, but had inherent limitations: it could only store six items at a time, and due to the AR nature of the application, users would not be able to see all faces of a cube at the same time. The result was users having to pick the cube up and rotate it to see all the assets stored, essentially making another action to check the folder's contents. Figure 4.13 shows a view of the cube in the *Unity3D* editor being used for relating different types of information (in this case, textual and visual) to communicate a concept.

After this prototype, we focused on a single type of multimedia, images, removing the need for the 3D icon abstraction. As mentioned in section 3.3.5, this also unifies the

design of the image representations by default, since they're the only type of multimedia present in the application.

4.4.2 Spatial Manipulation Prototype

This phase of prototyping focuses on how can multiple objects be spatially manipulated in an AR VE, and, consequently, how can objects be selected.

We started with creating a control schema that allowed users to manipulate objects with 6DOF, and evaluating what degrees of freedom the users used the least. This initial iteration used kept the selected object at a set distance when grabbed, allowing users to freely manipulate it on the three degrees of translation. The object was also set to face the grabbing user, resulting in users manipulating two degrees of rotation by moving their device. The remaining degree of rotation was manipulated through a UI element. For selecting objects, this initial variation allowed users to aim their devices to select an object. To help convey this method of selection, a crosshair was placed in the center of the screen, and an outline was activated when objects were hovered on. The user can then press and hold a button on the UI to grab that hovered object. This approach has the advantage of the user's finger placement not being essential to any movement actions, allowing it to move to manipulate the rotation manipulating UI element.

After the development of the initial prototype, degrees of freedom were incrementally removed from the variations to determine how many were needed to allow for expressive spatial organization. This process provided the advantage of incrementally simplifying the control schema, which would be needed to accommodate the remaining features in the final application.

A use case was developed to test this prototype's variations. This use-case was a game of *Tangram*, a puzzle where users have to, from a set of polygons, arrange them to form a predetermined shape. In this case, the shape was a square. A prototype that featured 2DOF, one for translation and another for rotation, and the previously described aim selection method can be seen in Figure 4.14.

One of these variations provided 2DOF, one being translation along the marker plane, and another being rotation on the y-axis. The conclusions derived from this variation come from its selection method, which was clicking on an object to select it, and, upon being selected, the object would then be moved to the center of the screen, where it would move to where the point of intersection between the marker plane and the mobile device's direction vector. This selection method had a disadvantage in that, upon selecting an object, its former position would be discarded, making the action of beginning an object manipulation irreversible. When solving a *Tangram* puzzle, rotating a piece in place is useful, and this method of selection made that action increasingly difficult.

The final prototype variation in this phase allowed manipulation on 1DOF, being translation along the marker plane. Although this variation, by definition, doesn't allow the solving of a *Tangram* puzzle, it further simplifies the control schema, and, in the

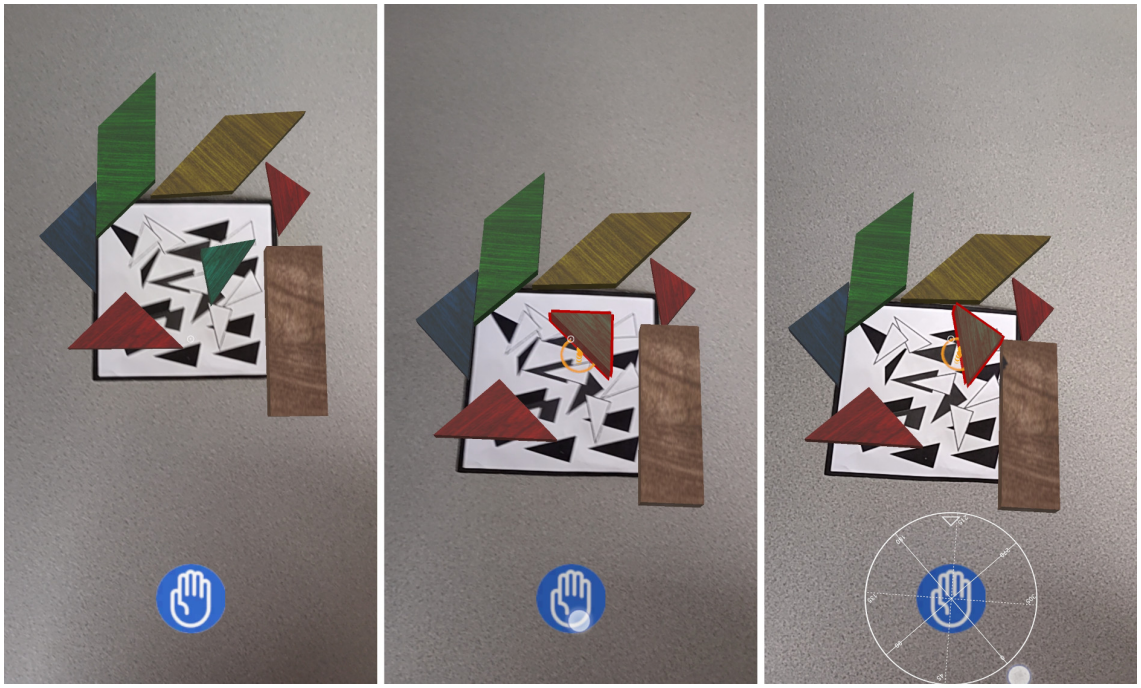


Figure 4.14: A spatial manipulation prototype's interface for *AR Tangram*. The white circle represents the placement of fingers on-screen. On the left, no *Tangram* block is selected, in the middle, a block is selected, and on the right, a block is being rotated.

context of image manipulation, if all users are viewing the scene from the same direction, the rotation doesn't need to be manipulated.

4.4.3 Collaboration Prototype

The focus of this prototype was collaborative interactions between users. In this prototype, multiple photos were placed in a plane and users were tasked with using that variation's set of features to signal other users of a specific photo.

In the first variation of this prototype, photos were able to be shared with the user's neighbours, as described in section 3.5.3. Upon selecting a photo and holding it, a user could drag it to either the right or left edge of the screen, and upon releasing the hold, the photo would appear on the respective neighbour's screen, as if they selected it. This feature succeeded in trivializing sharing when two or three users were in a session, but introduced unnecessary complexity when more users were to collaborate. This complexity occurs due to having users that are not direct neighbours. In those cases, there's a need for mechanisms that allow users to share photos with each other regardless of their relative position around the marker. To be able to identify who a user is within the AR scene, there needs to be an identifying characteristic for each user. One of these characteristics is the mobile device's position, which, in an in-person setting, can be used to identify a user.

On the following variation, users could be identified with two characteristics: their position on the VE, as well as a unique color. A virtual phone-like object was placed in

the real device's relative position to the marker, and assigned a color, just as previously shown in Figure 4.5, which shows the avatar in the integrated solution. Users could share a photo with another user by dragging the photo to the phone-like object. Although this sharing method fixed the issue of sharing with indirect neighbours, it required the users to, sometimes, remove the device's gaze from the marker, which would lead to loss of tracking, and, consequently, loss of the position of the other users, as mentioned in section 3.3.4.

To allow sharing with indirect neighbours without losing track of the marker, either all users are accessible through a constant element in the UI, such as a user list, or there's a virtual representation of the user which operates solely close to the marker plane. Due to the limited space for UI elements that smartphones and tablets provide, and the possibility of needing to present system information with UI elements, the latter approach was taken. The specification for this virtual user representation is an identifiable visual element that is located in the marker plane, which can be used for sharing, similarly to the phone-like object. Considering these design requirements, a visual metaphor was implemented. This metaphor was an anthropomorphic 3D avatar color-coded the same as the phone-like object, which was shown in Figure 4.5 in the context of the integrated solution. This design accomplishes in transmitting to the user that the object is a representation of a user by mimicking the human form, and provides identifying features through its color. It also provides a similar visual footprint to the phone-like object since they have similar scales. This shared characteristic, along with the shared color, unifies their design, resulting in users interpreting that they allow the same action, this being sharing a photo with the associated user.

Although having two virtual representations occupying different locations in the VE provides users with different ways of interacting with each other, the 3D avatar doesn't have any relation to the user, besides its color. Without the relation between the mobile device's location and its color, it becomes difficult for users to determine which color is associated with each user. To solve this problem, a ray is drawn between the mobile device's virtual position and the 3D avatar, which relates the phone-like object to the avatar, as well as provides an estimation of the phone's location by looking at the avatar. This estimation, in an in-person setting, allows users to, simply by looking at the marker plane, know approximately all users' locations and their associated colors.

The following variation introduced the feature to present a photo, allowing all users to see it simultaneously, which would be an increasingly difficult action to do with the previous feature set. Since this operation has a global effect, using the receiver paradigm used until this point would hinder parallel work, since it would be visually disruptive for a photo to appear in the center of every collaborator's screen. As such, presented objects need to occupy a different location. They could be present in the UI, occupying minimal space, but this approach would invalidate detailed viewing without interacting with that UI element. The other approach is situating the presented photos somewhere on the VE. The most suitable place in the VE for this placement is the center of the marker plane,

Table 4.2: Description of prototype phases.

Prototype	Purpose	Manip. DOF	Multimedia Types	Organization Types	Has Sharing?
Conceptual	Multimedia AR representations	5DOF	Images,Video,Audio,3D models, Text	Implicit & Explicit (folders)	✗
Spatial Manip.	Control schema for object manip.	2DOF	✗(*Tangram* blocks)	Implicit	✗
Collaboration	Defining sharing interactions	1DOF	Images	Implicit	✓
Final	Study research questions	1DOF	Images	Implicit & Explicit (folders)	✓

due to the marker-based nature of the AR application, which forces users to look at the marker to achieve tracking. This maximizes the chances users see the presented item without disrupting their current activity.

A problem introduced by presenting is the movement of the presented item, which removes it from its previous location and consequently could also remove it from a spatial organization layout. As a result of this, presenting creates a copy of the photo instead of moving the original. This approach has the additional advantage of allowing a relation between a copy and the original to be formed, which can be used to enhance the presenting act. For example, if there's visual feedback on the original photo when it's being presented, the presenting user could point to its place on a spatial hierarchy organized by the group.

4.4.4 Integrated Version

For the integrated version of the application, the feedback from the several prototype variations was considered in its design. In the initial stage of this phase of development, some features were still not implemented, namely folders. Since images have a real-world counterpart and their virtual representation uses that visual metaphor, folders could also use that approach. As such, folders were implemented as a representation of their real-world counterpart, as shown in the general mode interface in Figure 4.7, while simplifying the act of storing a photo. Storing a photo is done by placing the photo atop a placed folder in the VE, similarly to how sharing with a specific user is done. Due to these interaction similarities, the folders were made visually distinguishable by their user's color. Unlike the folders implemented in the conceptual phase, these folders' contents are not visible by default, which implies an "open folder" action. After opening, a folder's contents should be visually distinguishable from the photos outside of folders and photos inside other folders, as mentioned in section 3.5.2. Since users will be interacting with the items in the marker plane most of the time, the folders are limited to being placed in the marker plane. To support the visual separation of items inside and outside the folder, whose importance is explained in section 3.5.2, within the marker plane, open folders "expand" and push away all other elements while open.

5

Evaluation

To be able to answer the questions presented in section 1.2, we conducted user studies with the developed application. The user studies also gave us insight into how well the implementation applied the proposed concepts in chapter 3.

In order to obtain meaningful results, and, consequently, evaluate effectively, we developed a testing protocol for the user studies. In this protocol, we decided to evaluate each interaction individually by giving users very specific tasks to accomplish. To test the interface as a whole, and how complementary the interaction set's interactions are to each other, users were also tasked to accomplish an open-ended, complex task, which can be completed through a variety of methods.

The focus of this testing process is evaluating the interface's capability to communicate its interaction set and the capability of the interaction set to provide the tools to accomplish complex tasks.

5.1 Protocol

Each test session is composed of two people: the tester and the test subject/user. At the start of the test session, the test subject reads and fills out the consent form, present in Annex I. After agreeing to it, the test subject is given a smartphone, with the application already running, and given a short explanation about what the app is designed to do, as well as some essential technical aspects, such as how to interact with marker-based AR applications.

The application's state, when the user receives it, is inside a collaborative session, which was created prior. The state of the VE wasn't changed in any way beforehand. The photos present in the VE are of different dogs. The tester also possesses a smartphone with the application running, inside the same collaborative session. This allows for the test subject to be able to do the sharing operations.

After giving the smartphone to the user, the tester explains briefly the marker's function in the application and asks them to use the smartphone to look at the dog photos present in the VE. Afterwards, if the user was successful, the tester explains succinctly

how avatars work, and where they are in the VE. The user is then asked to look at them. In this initial phase of the user's interaction with the application, we keep track if they looked at the marker after being explained its function in the application and if they looked at the avatars after being provided a succinct explanation of them.

After being given some minutes to freely explore the VE, the user is given small tasks to do, such as selecting objects or sharing images through the avatar. We evaluate the execution of these tasks according to the attempts that it took to complete them and the time it took. An attempt, in this case, is considered a conscious attempt to interact with the interface. Each task has to be completed within a time and an attempt limit, otherwise, the task is to be considered not completed. The tracking of these two different metrics and their correlation allow us to conclude if the interface mode changes are easily reversible. Time, in this context, not only informs us of the difficulties of doing specific actions, but also the time users take to revert wrong actions. Attempts, in this context, are a time-independent way to measure how intuitive the interface is. For example, if a user wrongfully executes an interaction and the resulting state is hard to exit from, that would mean that wrong attempts take a lot of time, but not that the interaction is necessarily hard to do.

If a user fails to complete an interaction task, the tester explains the interaction to the user in order to allow them to understand interactions that depend on that one. Not explaining the interaction would lead to compounding difficulties for the user, and would compromise the independence of the tasks, and, consequently, their resulting study variables.

After doing every interaction the application offers in an isolated setting, the user is tasked to observe actions done by the tester. The interactions the tester does are a subset of the ones the user did previously. This process allows to test the feedback mechanisms offered by the system. For each interaction done by the tester, the user must say what interaction they think the tester is doing. From this process, we can extract the correctness of the user's observation, and the time it took them to interpret the tester's action.

5.2 Phase 0: Interface exploration

This preliminary phase is done to give users a basic understanding of the mechanisms behind the VE which are essential for effective application usage. In this phase, users are tasked to:

1. Point their device at the virtual dog photos.
2. Point their device to the anthropomorphic avatar.
3. Point their device to the phone-like avatar.

After that, users are given two minutes to explore the interface before doing the rest of the study. The tasks are as follows:

Task 0A- Looking at the marker. After being explained the purpose and goals of the application, the user is explained how to interact with marker-based applications and asked to look at the dog photos in the VE. The following observation was formulated to determine if the user understands the basic action of looking at the marker to perceive the VE:

Observation 0A1- Does the user complete the task? Answer is Yes or No.

Task 0B- Looking at the avatars. The tester explains the concept of the avatars and gives a basic description of the two avatars present in the system. Then, the user is tasked to look at the tester's avatars in the VE and point to them on their screen. The following observations were made:

- **Observation 0B1- Does the user recognize the anthropomorphic avatar?** Answer is Yes or No.
- **Observation 0B2- Does the user recognize the phone-like avatar?** Answer is Yes or No.

These observations allow us to know if the design of the avatars conveys that they are the user's virtual representations. The correlation between the results of these observations also tests if the visual relation between them is effective.

5.3 Phase 1: Individual Interaction Tasks

This phase of evaluation has the user do the basic interaction types the application provides, without an overarching goal. This phase allows the user to increase their familiarity with all the system's functionalities and start to think about their applicability in collaborative scenarios. To test the interface's ability to communicate its functionalities and its intuitiveness, the users must complete a task described verbally by the tester. This description is very brief and emphasizes the result of an action, such as "Move an image", or "Share an image with me through my avatar". With each task, the tester would also annotate any unique behaviours the users would exhibit, such as a predisposition to do a certain interaction type in a specific way. The first task is the following:

Task 1A- Selecting an image for moving. The tester instructs the user, who is now in general mode, to select an image in order to move it. The user is tasked to express verbally when they think the task is completed. The following observations were made:

- **Observation 1A1- Was the user able to do the task within 15 seconds?** Answer is Yes or No.
- **Observation 1A2- How much time did the user take to do the task?** Answer is expressed in seconds.

- **Observation 1A3- How many attempts did the user need to do the task?** Answer is either 1,2 or 3.

These observations allow us to quantitatively infer if this action has two properties: it is intuitive and any wrong actions are easily reversible.

These observations are repeated for each interaction task but with some variations on the time and attempt limits. The differences, along with the interaction types' descriptions to the user, are expressed in Table 5.1.

5.4 Phase 2: Observation game

In this phase, the user is instructed to observe the actions the tester does. The actions selected for this phase affect the other user's view of the VE so that their interpretation is possible. We designed this phase of testing to evaluate how effective are the feedback mechanisms at communicating their respective actions. If a user can interpret the action correctly, that indicates that its feedback mechanisms are adequate. The tester does the following actions in the described order:

1. Moving an image.
2. Placing a folder.
3. Storing an image.

Table 5.1: Tester prompt, time and attempt limits for all interaction tasks.

Interaction	Description	Time Limit (seconds)	Attempt Limit
1A-Selecting Image	"Select an image for moving"	15	3
1B-Placing & Resetting Image	"Place that image in the VE"	5	2
1C-Placing a Folder	"Place your folder"	10	2
1D-Removing Folder	"Remove the placed folder"	10	2
1E-Storing Object In Folder	"Store an image in your folder"	20	4
1F-Opening & Closing Folder	"Now try opening that folder... now try closing it"	10	2
1G-Sharing via an avatar	"Now try sharing a image with me using my avatar"	15	3
1H-Entering Object Mode	"Now try to select a image to access more of its options"	20	4
1I-Sharing To A Neighbour	"Now try so share that image with the person on your right/left"(depending on what side the tester is)	15	2
1J-Presenting	"Now try presenting that photo such that all users are able to see it"	15	3
1K-Viewing In Detail	"Now try seeing the image in detail without moving closer to it physically"	10	2
1L-Zooming and Panning	"Now, in this mode, try zooming in and out and pan around"	10	3
1M-Exiting Detail Mode	"Now exit that detail view"	5	1

4. Sharing via avatar.

5. Sharing with neighbour.

All of the tasks of this phase have an observation that determines the effectiveness of the feedback mechanisms, which is: **Observation 2X1- Was the user correct in their observation?** (X is used here as a placeholder for the task's associated letter), with answers being Yes or No. The descriptions of the tasks in this phase are the following:

Task 2A- Observe moving an image. The tester starts by moving an object and asks the user to describe what they are doing. Then, the tester places that object in the marker plane. Then the tester starts moving another object, but later resets the object's position.

Task 2B- Observe placing a folder. The tester first places the folder in the VE. After receiving a response, the tester then moves the folder to another place in the VE.

Task 2C- Observe storing an image. The tester uses their folder from the previous task to store an image from the scene.

Task 2D- Observe sharing. In this task, the tester will share an image with the user via the two available methods: avatar sharing and neighbour sharing. These two methods of sharing have similar notifications on the receiver side, but with the clear distinction that, in neighbour sharing, the image scrolls from the sender's side, unlike avatar sharing, where it appears from the top of the screen.

This task has an additional observation associated with it to evaluate if the sharing feedback mechanisms are dissimilar enough for users to notice, which is **Observation 2D2- How many specific sharing methods could the user distinguish?**, with answers being either 0, 1 or 2.

5.5 Phase 3: Image categorization

The marker is placed atop a large sheet of paper containing boxes labeled with different dog breeds. There are a total of three boxes, two large ones, labeled "Husky" and "Chihuahua", and a small one, labeled "Other". In the VE, there are various dog photos of a set of breeds that contain the labeled ones, as well as other ones. The user is then informed of the purpose of this task, which is to place the dog photos in their respective boxes, and that they should be able to, at any time, share specific photos requested by the tester. The purpose of these questions is for them to use folders when necessary, especially in the case of the "Other" box, which only has space to house either a single photo or a folder. This task's evaluation goal is not to measure the efficacy of users at distinguishing between different dog breeds, but to see if they use the full interaction set of the application to

organize them to the best of their ability. To this effect, we gathered data about whether users used specific interactions while organizing. This phase's tasks are:

Task 3A- Organize the dog photos according to their breed. This task evaluates two aspects:

- The users' tendency to use a specific kind of organizational action to categorize the dog photos
- If the users leveraged the benefits of viewing a photo in detail to help in their categorization decision.

As such, the following observations were made:

- **Observation 3A1- Did the user use folders in their organization system?** Answer is Yes or No.
- **Observation 3A2- Did the user view any photos in detail while organizing?** Answer is Yes or No.

Task 3B- Share specific photos. This task serves to evaluate the tendency of users to use a specific method of sharing, and as such, is associated with **Observation 3B1- How did the user share the photo?**, with answers being via avatar, via neighbour or via presenting.

5.6 User Questionnaires

After the study, users are asked to answer a questionnaire about the tasks they have done. The questionnaire helps us gather qualitative data about a user's experience. The questionnaire also serves to characterize the users.

5.6.1 Phase 1 Questionnaire

For each interaction task in Phase 1, the user is asked three *7 point Likert-scale* questions, which are questions where a user is given an affirmation and is asked to rank how much they agree with that affirmation on a scale from one to seven. The questions are as follows (**X** is used as a placeholder for the tasks' respective letters):

- **Question 1X4**- I think the action was easy to do.
- **Question 1X5**- I think that I would need the help of a technical person to be able to do this action.
- **Question 1X6**- My expectations of this action's input were different from the actual one.

The first two questions are a part of the *System Usability Scale* (SUS) questionnaire, which is a standardized questionnaire, developed by John Brooke [10] for evaluating interfaces. We formulated the third question to complement the time and attempt metrics gathered during the user study.

5.6.2 Phase 2 Questionnaire

For each interaction interpreted in Phase 2, the user is asked a 7-point Likert-scale question, which is **Question 2X3- This action was easy to interpret**.

5.6.3 System Questionnaire

To evaluate the user's qualitative experience of the system, we formulated a questionnaire whose domain is the system as a whole. In this questionnaire, we ask the user all the questions present in the *SUS*, as well as a subset of the questions present in *NASA-TLX* [23], which is a questionnaire that evaluates a user's comfort, mental effort and stress while using the application. From this questionnaire, we took the following questions:

- **Question S1- How mentally demanding was the study?**. The answer is a scale from 1 to 7, where 1 is "Not at all" and 7 is "Very much".
- **Question S2- How physically demanding was the study?**. The answer is a scale from 1 to 7, where 1 is "Not at all" and 7 is "Very much".
- **Question S3- How irritated, stressed, or annoyed were you during the study?**. The answer is a scale from 1 to 7, where 1 is "Not at all" and 7 is "Very much".

5.6.4 User Profile Questionnaire

When formulating this questionnaire, we considered that the most important user characteristics were their familiarity with AR, their familiarity with collaborative applications and their familiarity with synchronous systems. Additionally, we also profiled the user's age and gender. The profile questions are as follows:

- **U1- Gender** Answers received were Male or Female.
- **U2- Age** Answers ranged from 21 to 25 years old.
- **U3- How familiar are you with smartphones/tablets?** Answer is the familiarity 7-point categorical Likert-scale, ranging from "Not at all" to "Very familiar with".
- **U4- How familiar are you with Augmented Reality (AR)?** Answer is the familiarity 7-point categorical Likert-scale.
- **U5- How frequently did you use AR applications in the last year?** Answer is a 5-point categorical Likert-scale, ranging from "Never"to "Daily".

Gender	Count	% of Population
Male	9	75%
Female	3	25%

Table 5.2: Gender distribution.

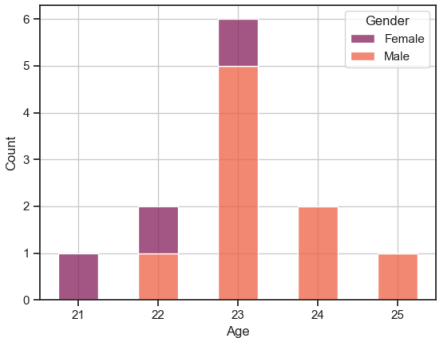


Figure 5.1: User age histogram.

- **U6- What are the platforms of the AR applications you used?** Answer is Smartphone/Tablets, Computers, None, or "Other".
- **U7- How familiar are you with collaborative applications?** Answer is the familiarity 7-point categorical Likert-scale.
- **U8- How familiar are you with distributed systems?** Answer is the familiarity 7-point categorical Likert-scale.

5.7 Results

In this section, the results of the observations and the user questions obtained from the user studies are presented.

5.7.1 Population Characteristics

The user population was composed of 12 participants. The demographics extracted from the user questionnaires are represented in Table 5.2 (population gender distribution), Figure 5.1 (population age distribution) Figure 5.2 (familiarity with system technologies and concepts), and Figure 5.3 (Frequency of AR usage).

Based on the answers to the user profile questionnaire, we can divide the user population into two groups: experienced users, who answered to ***Question U4*** with a 4 or higher, and inexperienced users, who answered with a score lower than 4. In total, there were seven experienced users ($\approx 58\%$) and five inexperienced ones ($\approx 42\%$) among the population.

5.7.2 Phase 1 Results

The ***Question 1X4- I think the action was easy to do*** provides us with insight about the task's relative difficulty in the form of a 7-point Likert-scale. Across all tasks, the mean score for answers to this question is 6.13. Although this metric indicates users think that each of the interactions is easy to do individually, it doesn't provide insight into the

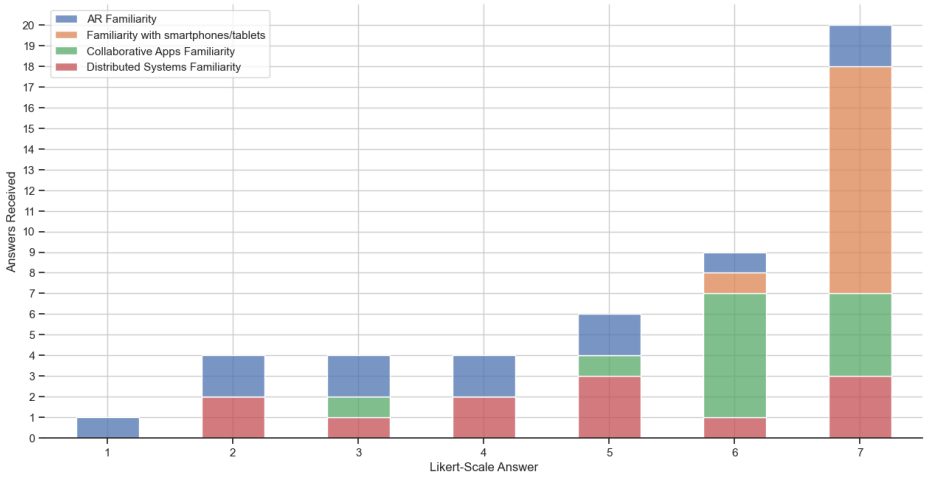


Figure 5.2: User familiarity histogram. Answers are from 7-point Likert-scale questions.

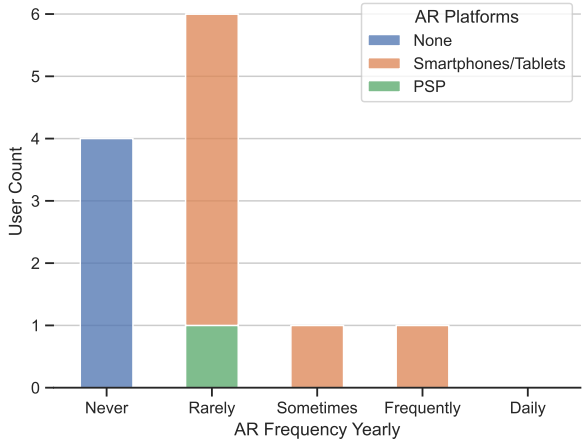


Figure 5.3: Frequency of population's AR usage. Colors denote the different AR platforms.

difficulty of the interactions when operating as a set, or the difficulty of the interactions relative to each other. To analyze this latter context, we calculated the variation between the mean score of all tasks and the mean score of the individual tasks. This metric, presented in Figure 5.4, better represents the tasks users thought were more difficult. The figure depicts the tasks which are easier than usual in green and the ones that are harder than usual in red.

Figure 5.5 shows the average times users needed to complete the tasks. The completion times are a quantitative metric that allows us to analyze a task's relative difficulty. We can also analyze this difficulty in the context of the interface mode the interaction is done in. The times were measured from a neutral state, in this case, the general mode. This means that the completion times of image-dependant interactions, such as sharing with an avatar and storing an image in a folder, include the process of first selecting the

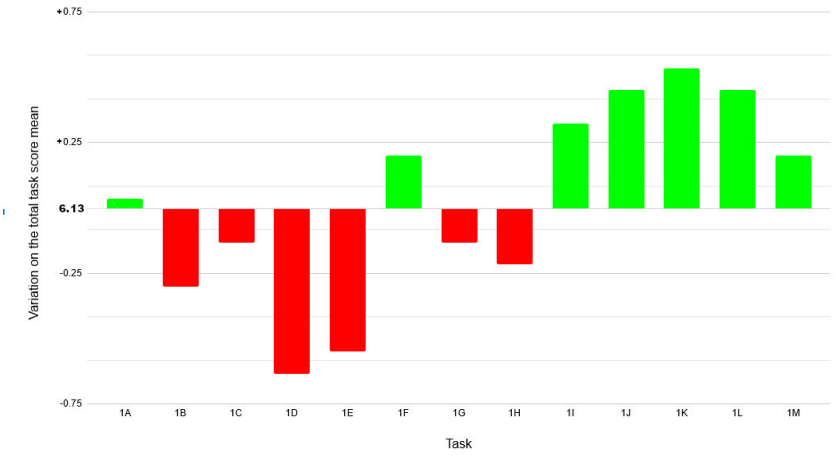


Figure 5.4: Phase 1 easiness Likert score mean variation graph.

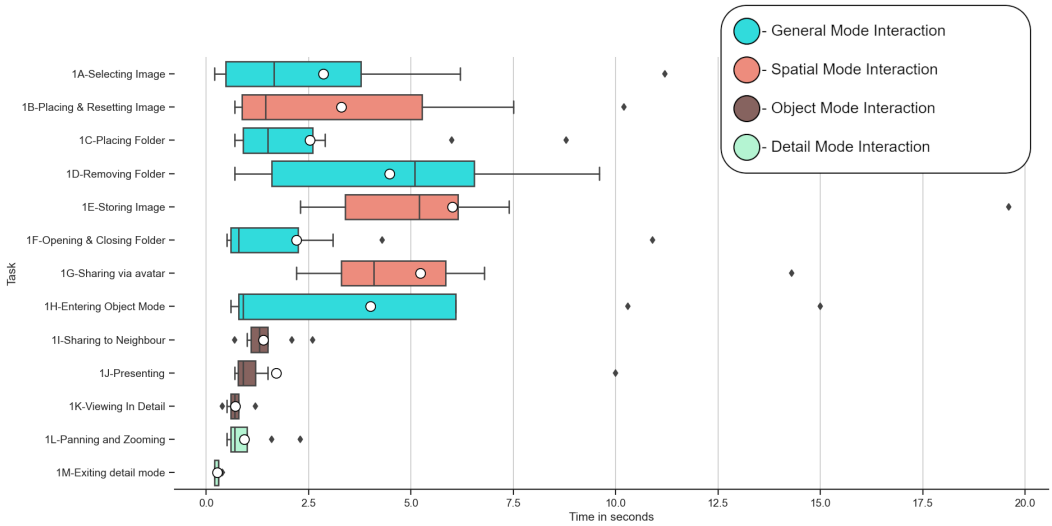


Figure 5.5: Phase 1 interactions' completion times. Colors represent the interface mode of the interaction.

image. We decided upon this method of measuring because the changing from an image-independent context to a dependent one is not explicitly shown to the user, meaning that users didn't know *a priori* that selecting an image would be a needed step to complete these tasks.

From the completion times, we can conclude that users had negligible difficulty when using object mode and detail mode interactions relative to the difficulty users faced in general and spatial mode. By analyzing the characteristics that differ between these two sets of modes, mainly the difference in the quantity of information displayed by the UIs in the two sets of modes (with detail mode and object mode displaying more information), and the spatial nature of the spatial and general mode interactions, which differs from the non-spatial interactions present in the object and detail modes.

Task 1A- Select an image(Tables 5.3): From the easiness score and the completion rate, we can conclude that users thought this interaction was easy to do, although the quartiles on the different input expectations score indicate that there is variation in the user's opinion of the action's intuitiveness. Notably, this variation in user opinion is corroborated by the number of users (n=6) who needed multiple attempts to do this interaction. We can also prove that there's a significant difference between the average time per attempt for users who took multiple attempts and the users who did it correctly on the first attempt. This discrepancy is expected since, upon making a mistake, such as switching to object mode, users need to revert to the previous state of the interface to be able to repeat the task. However, given the size of the discrepancy between the means, we can conclude that users, in this phase of testing, did not find this interaction to be easily reversible.

The low score median and low variation in the answers to **Q5** provide insight into how this specific action can be found easily, even if the method for finding it was trial-and-error.

Table 5.3: Task 1A results. Quantitative results on left, Qualitative results on right and T-test between multi-attempt and single-attempts time-per-attempt on bottom.

Task 1A Observation	Result
O1- Completion(%)	100%
O2- Time to complete ($\overline{X} of\ seconds$)	$2.86 \pm 3.36s$
O3- Attempt count (Median)	1.5

Task 1A Question	Score Median	Score Q_1 and Q_3
Q4- Task Easiness	6.0	[5.75 , 7.0]
Q5- Need of Help	1.0	[1.0 , 1.25]
Q6-Different Input Expectations	1.5	[1.0 , 2.25]

Task 1A- Select an Image				
Attempt Category	N	Mean	Std. Deviation	T-Test Significance
=1	6	0.47	0.20	p=0.032
>1	6	2.45	1.66	p<0.05

Task 1B- Placing & Resetting images(Tables 5.4): The completion rate indicates that some users couldn't do this interaction within the time and attempt limits. Also, users thought this interaction wasn't as intuitive as selecting an image, as we can conclude from the variation in answers to **Q6**. However, after learning it, users thought the action was easy to do. We can also conclude from the attempt times that, this action, although significantly long to recover from when making wrong attempts, is more reversible than ***Task 1A***. This can be due to the reachable interface states from wrong attempts at this action being more manageable for users.

Task 1C- Placing folders(Tables 5.5): This action presented a high completion rate, as well as positive results in the user questionnaire. Here, one user mentioned that they took more time and attempts due to the lack of feedback from the button, which only indication that the drag action had started was a change in color and size. Since the user's finger was over the button, they couldn't see the feedback. This user comment can justify the divergence between the attempt times present in the data.

Table 5.4: Task 1B results. Quantitative results on left, Qualitative results on right and T-test between multi-attempt and single-attempts time-per-attempt on bottom.

Task 1B Observation	Result
O1- Completion(%)	75%
O2- Time to complete ($\overline{X} of\ seconds$)	$2.09 \pm 1.67s$
O3- Attempt count (Median)	1.0

Task 1B Question	Score Median	Score Q_1 and Q_3
Q4- Task Easiness	6.0	[5.75, 7.0]
Q5- Need of Help	1.0	[1.0, 2.25]
Q6-Different Input Expectations	1.0	[1.0, 2.75]

Task 1B- Placing & Resetting Images

Attempt Category	N	Mean	Std. Deviation	T-Test Significance
=1	6	1.03	0.37	p=0.03
>1	3	2.10	0.44	p<0.05

Table 5.5: Task 1C results. Quantitative results on left, Qualitative results on right and T-test between multi-attempt and single-attempts time-per-attempt on bottom.

Task 1C Observation	Result
O1- Completion(%)	91.67%
O2- Time to complete ($\overline{X} of\ seconds$)	$2.53 \pm 2.59s$
O3- Attempt count (Median)	1.0

Task 1C Question	Score Median	Score Q_1 and Q_3
Q4- Task Easiness	6.0	[6.0 , 7.0]
Q5- Need of Help	1.0	[1.0 , 2.0]
Q6-Different Input Expectations	1.0	[1 , 2.25]

Task 1C- Placing Folders

Attempt Category	N	Mean	Std. Deviation	T-Test Significance
=1	7	1.11	0.48	p=0.161
>1	4	2.50	1.50	p>0.05

Task 1D- Removing folders(Tables 5.6): To complete this task, users could either move the folder back to the folder button or click it. A majority of users (n=8) first tried moving the folder back, essentially doing the motion of placing it but reversed. However, the application's detection for long-pressing, which is the input needed to start moving the folder, took long enough to, in the case of some users (n=5), disregard it as the right answer. These users then found the other way to remove the folder. This can justify the significantly high discrepancy of attempt times. After the task, users were informed of both methods of folder removal, meaning that the questionnaire answers regard both methods.

Table 5.6: Task 1D results. Quantitative results on left, Qualitative results on right and T-test between multi-attempt and single-attempts time-per-attempt on bottom.

Task 1D Observation	Result
O1- Completion(%)	91.67%
O2- Time to complete ($\overline{X} of\ seconds$)	$4.47 \pm 2.97s$
O3- Attempt count (Median)	2.0

Task 1D Question	Score Median	Score Q_1 and Q_3
Q4- Task Easiness	5.0	[4.0 , 7.0]
Q5- Need of Help	1.5	[1.0 , 3.0]
Q6-Different Input Expectations	2.5	[2.0 , 3.25]

Task 1D- Removing Folders

Attempt Category	N	Mean	Std. Deviation	T-Test Significance
=1	3	0.87	0.21	p=0.01
>1	8	2.91	1.11	p<0.05

Task 1E- Storing images(Tables 5.7): This action presents a high discrepancy of attempt times and a relevant number of users taking multiple attempts (n=6). However, **Q4** and **Q5** still present good score modes. A factor that contributed to these results is that users needed to press the move button while hovering over the folder to complete

the action. However, when clicking the button, sometimes the users moved their holding hand enough to no longer be hovering over the folder, resulting in users only placing the image close to the folder, when intending to do the correct interaction.

Table 5.7: Task 1E results. Quantitative results on left, Qualitative results on right and T-test between multi-attempt and single-attempts time-per-attempt on bottom.

Task 1E Observation	Result
O1- Completion(%)	91.67%
O2- Time to complete ($\overline{X} of seconds$)	$6.01 \pm 4.78s$
O3- Attempt count (Median)	2.0

Task 1E Question	Score Median	Score Q_1 and Q_3
Q4- Task Easiness	6.5	[5.0 , 7.0]
Q5- Need of Help	1.5	[1.0 , 2.0]
Q6-Different Input Expectations	2.0	[1.0 , 2.75]

Task 1E- Storing an Image

Attempt Category	N	Mean	Std. Deviation	T-Test Significance
=1	5	3.26	0.67	p=.079
>1	6	8.30	5.61	p>0.05

Task 1F- Opening and Closing Folders(Tables 5.8): This action presents good qualitative results, although its completion rate and time to complete are not what is expected from a task that takes two clicks on top of the same object to do. Partly, this is due to the tester's prompt to close the folder being interpreted as a repetition of an interaction users had done before at this point- removing the folder. This interpretation is corroborated by the easiness score being high.

Table 5.8: Task 1F results. Quantitative results on left, Qualitative results on right and T-test between multi-attempt and single-attempts time-per-attempt on bottom.

Task 1F Observation	Result
O1- Completion(%)	91.67%
O2- Time to complete ($\overline{X} of seconds$)	$2.13 \pm 2.89s$
O3- Attempt count (Median)	1.0

Task 1F Question	Score Median	Score Q_1 and Q_3
Q4- Task Easiness	7.0	[5.75 , 7.0]
Q5- Need of Help	1.0	[1.0 , 1.25]
Q6-Different Input Expectations	1.0	[1.0 , 2.0]

Task 1F- Opening and Closing Folders

Attempt Category	N	Mean	Std. Deviation	T-Test Significance
=1	7	0.66	0.11	p=.034
>1	4	2.35	1.86	p<0.05

Task 1G- Sharing via avatar(Tables 5.9): This action presents good qualitative and quantitavive metrics, for an action which involves aiming the center of the screen to a moving target. This action also presents a significant equality between the means of the multi-attempt attempt times and the single ones, meaning that successive wrong attempts don't carry an overhead time cost.

Task 1H- Entering object mode(Tables 5.10): This action presents the lowest completion rate (tied with **Task 1B**). This task was hindered by the inability of the tester's prompt to communicate the concept of selecting an image to access a sub-menu, without hinting at the method of image selection. As such, users took a long time to interpret the prompt. This explains the phenomenon of having a high time to complete with a low attempt count median.

Table 5.9: Task 1G results. Quantitative results on left, Qualitative results on right and T-test between multi-attempt and single-attempts time-per-attempt on bottom.

Task 1G Observation	Result
O1- Completion(%)	91.67%
O2- Time to complete ($\overline{X} of seconds$)	$5.23 \pm 3.34s$
O3- Attempt count (Median)	1.0

Task 1G Question	Score Median	Score Q_1 and Q_3
Q4- Task Easiness	6.0	[5.75 , 7.0]
Q5- Need of Help	1.0	[1.0 , 2.0]
Q6-Different Input Expectations	1.0	[1.0 , 2.0]

Task 1G- Sharing via Avatar

Attempt Category	N	Mean	Std. Deviation	T-Test Significance
=1	8	3.79	1.14	p=0.95
>1	3	3.73	0.91	p>0.05

Table 5.10: Task 1H results. Quantitative results on left, Qualitative results on right and T-test between multi-attempt and single-attempts time-per-attempt on bottom.

Task 1H Observation	Result
O1- Completion(%)	75%
O2- Time to complete ($\overline{X} of seconds$)	$4.01 \pm 5.33s$
O3- Attempt count (Median)	1.0

Task 1H Question	Score Median	Score Q_1 and Q_3
Q4- Task Easiness	6.0	[5.0 , 7.0]
Q5- Need of Help	1.0	[1.0 , 2.0]
Q6-Different Input Expectations	1.5	[1.0 , 2.0]

Task 1H- Entering Object Mode

Attempt Category	N	Mean	Std. Deviation	T-Test Significance
=1	6	0.78	0.16	p=.083
>1	3	4.06	1.76	p>0.05

Task 1I- Sharing to neighbour(Tables 5.11): This action, together with the rest of the object mode interactions, presented high overall completion rates, good qualitative scores and low times to complete. However, this action in particular features a higher than usual time to complete when compared to the following object mode interactions. Considering it is the first interaction that is asked of the user when first observing this mode, it is sensible to assume that users have a higher degree of knowledge in the following tasks, which justifies the times for object mode interactions being progressively faster. Due to the very low, or non-existent, number of multi-attempt tries within the following tasks, this distinction doesn't inform us of anything meaningful, and thus, the table containing that information is ommited.

Table 5.11: Task 1I results. Quantitative results on left and qualitative results on right.

Task 1I Observation	Result
O1- Completion(%)	91.67%
O2- Time to complete ($\overline{X} of seconds$)	$1.4 \pm 0.53s$
O3- Attempt count (Median)	1.0

Task 1I Question	Score Median	Score Q_1 and Q_3
Q4- Task Easiness	7.0	[5.75 , 7.0]
Q5- Need of Help	1.0	[1.0 , 1.25]
Q6-Different Input Expectations	1.0	[1.0 , 2.25]

Task 1J- Presenting(Tables 5.12): From the information gathered by the previous task, users who correctly interpret the presenting feature to be within the object mode menu can make the correct assumption that it is accessed by swiping either up or down. The tester observed that users almost exclusively first tried swiping up in this task, corroborating the effectiveness of the design and placement of this option in the object mode menu.

Table 5.12: Task 1J results. Quantitative results on left and qualitative results on right.

Task 1J Observation	Result
O1- Completion(%)	100%
O2- Time to complete ($\overline{X} of seconds$)	$1.72 \pm 2.62s$
O3- Attempt count (Median)	1.0

Task 1J Question	Score Median	Score Q_1 and Q_3
Q4- Task Easiness	7.0	[6.0 , 7.0]
Q5- Need of Help	1.0	[1.0 , 1.25]
Q6-Different Input Expectations	1.0	[1.0 , 2.0]

Task 1K- Going to detail mode(Tables 5.13): Users, at this moment, have exhausted all options in the object mode menu, and as such, when asked to view an image in detail, do the correct gesture to trigger the action in a small time frame.

Table 5.13: Task 1K results. Quantitative results on left and qualitative results on right.

Task 1K Observation	Result
O1- Completion(%)	91.67%
O2- Time to complete ($\overline{X} of seconds$)	$0.71 \pm 0.21s$
O3- Attempt count (Median)	1.0

Task 1K Question	Score Median	Score Q_1 and Q_3
Q4- Task Easiness	7.0	[6.0 , 7.0]
Q5- Need of Help	1.0	[1.0 , 2.0]
Q6-Different Input Expectations	1.0	[1.0 , 2.0]

Task 1L- Zooming and panning(Tables 5.14): These actions use a very common interaction metaphor in smartphone photo manipulation. As such, the results presented here, both qualitative and quantitative, are good.

Table 5.14: Task 1L results. Quantitative results on left and qualitative results on right.

Task 1L Observation	Result
O1- Completion(%)	100%
O2- Time to complete ($\overline{X} of seconds$)	$0.93 \pm 0.53s$
O3- Attempt count (Median)	1.0

Task 1L Question	Score Median	Score Q_1 and Q_3
Q4- Task Easiness	7.0	[6.75 , 7.0]
Q5- Need of Help	1.0	[1.0 , 1.0]
Q6-Different Input Expectations	1.0	[1.0 , 2.0]

Task 1M- Exiting detail mode(Tables 5.15): Since this interaction involves pressing a × button, which is neither a difficult action to do nor interpret, the results, both qualitative and quantitative, are naturally good.

Table 5.15: Task 1M results. Quantitative results on left and qualitative results on right.

Task 1M Observation	Result
O1- Completion(%)	100%
O2- Time to complete ($\overline{X} of seconds$)	$0.275 \pm 0.08s$
O3- Attempt count (Median)	1.0

Task 1M Question	Score Median	Score Q_1 and Q_3
Q4- Task Easiness	7.0	[5.75 , 7.0]
Q5- Need of Help	1.0	[1.0 , 1.0]
Q6-Different Input Expectations	1.0	[1.0 , 1.25]

5.7.3 Phase 2 Results

During the phase 2 tasks, users were asked to observe and verbally communicate the actions that the tester was executing within the shared VE. This process was possible due to the order of the phases which enabled users to know how to verbally communicate their actions since they were asked previously to do the reverse- interpreting actions from verbal communication. To be able to conclude that the feedback mechanisms in the

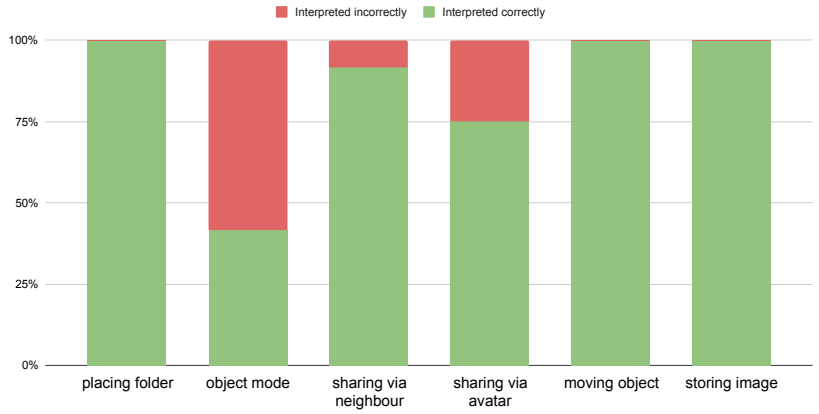


Figure 5.6: Interpretation correctness chart. Vertical axis represents percentage points.

system were appropriate, we extracted the correctness of the user's interpretations. The results are shown in Figure 5.6.

From this figure, we can derive insight about the effectiveness of the used feedback mechanisms, mainly that, feedback that appears both when the user is doing it and when the tester is doing it is highly effective. This perspective is corroborated by the poor performance of the object mode feedback, which, although symmetrical, wasn't noticed by users when doing it themselves, since the object mode menu covers a substantial portion of the screen. Additionally, a great majority of users (n=11) were able to distinguish the sharing via neighbour feedback, although when asked about the sharing via avatar feedback, two users were not able to answer correctly.

5.7.4 Phase 3 Results

The purpose of this evaluation phase was to verify what interactions the user would tend to use in an open-ended, categorization problem. This problem was designed to be analogous to the photo categorization use case presented in Section 3.4. Table 5.16 and Figure 5.7 show the results for **Observation 3A1- Did the user use folders in their organization system?** and **Observation 3A2- Did the user view any photos in detail while organizing?** of **Task 3A- Organize the dog photos according to their breed** and the sharing methods used in **Task 3B- Share specific photos**, respectively. The results show that the great majority of users (n=11) organized solely using the spatial manipulation features, while only one user used both organizational actions. Considering the difference in time to complete these actions, it is sensible to conclude that users, in a situation where organizing via folders isn't necessarily advantageous, defaulted to the faster action. A majority of users (n=10) used viewing in detail throughout this phase when in doubt about the category of a specific photo.

Regarding the Task 3B sharing method tendencies, users tended to sharing with their neighbour a majority of the time (n=7), followed by presenting (n=3), followed by sharing

Interaction	Usage Rate
Storing images	8.3%
Viewing in Detail	83.3%
Moving images	100%

Table 5.16: Task 3A Results.

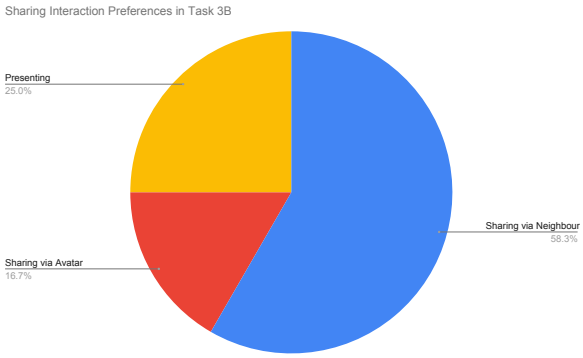


Figure 5.7: Task 3B Results.

via avatar (n=2). We can analyze this user tendency by evaluating the number of actions it takes for a user to reach the desired goal from a general mode, which can be considered the default interface state of the application. Sharing with neighbours and presenting each needs two actions to be reached from general mode- long pressing the designated photo, and dragging it up, in the case of presenting, or to one of the sides, in the case of sharing with neighbour. Sharing with avatar, in comparison, needs three actions to be executed: firstly, select an image, then aim at the avatar, and finally, confirm the sharing via a button. Also, when comparing the **Phase 1** time for completion results for the object mode interactions and sharing with avatar, we can see that, again, users preferred the faster actions. The difference between the sharing with neighbour usage and presenting can be due to the correctness of the action. Since the user was asked to show a specific photo **to the tester**, the most correct option would be to share it directly with him. Given the already discussed predisposition to choose the faster action, the majority of users preferred to use the share with neighbour interaction.

5.7.5 System Questionnaire Results

This questionnaire included the SUS questionnaire and questions to determine a user's well-being during the evaluation process. We can calculate the SUS of the application through the data gathered in this questionnaire. The SUS answers statistics are present in Table 5.17 We obtained an average SUS score of 83.54. This SUS score indicates that the application has a high degree of usability. This information is corroborated by the answers to the **Phase 1** user questionnaire.

The well-being questionnaire results are presented in Table 5.18. These answers provide insight into how well a user was feeling during the evaluation process. The results were positive, meaning that the evaluation process, and, consequently, the application's usage didn't demand any great amount of mental or physical effort, and didn't cause frustration or stress for the user.

Table 5.17: SUS questionnaire results.

SUS Question	Median	Q_1 and Q_3
I think that I would use this system frequently	3.5	[3.0, 4.0]
I found the system unnecessarily complex	1	[1.0, 1.0]
I thought the system was easy to use	4	[4.0, 4.0]
I think that I would need the help of a technical person to use this system	1	[1.0, 1.0]
I found the various functions of the application to be well integrated	4	[4.0, 4.0]
I thought there was too much inconsistency in the system	1	[1.0, 1.0]
I imagine that most people would be able to learn to use this system very quickly	4	[3.75, 5.0]
I found the system very awkward to use	1	[1.0, 1.0]
I felt very confident using the system	4	[4.0, 4.25]
I needed to learn a lot of things before I could get going with this product	1	[1.0, 1.25]

Table 5.18: Well-being during the evaluation process user questionnaire results.

Question	Median	Q_1 and Q_3
SQ1- Mental Demanding	1	[1,1]
SQ2- Physical Demanding	1	[1,1]
SQ3- Stress and Frustration	1	[1,1]

5.8 Discussion

We start this section by repeating the research questions introduced in Section 1.2:

- **RQ1:**What means of interaction with multimedia content should a collaborative AR system implement?
- **RQ2:**How can users collaboratively use the multimedia content in the VE to reach a shared goal?

RQ1 aims to outline a specific set of interaction types that manipulate the display of multimedia assets to achieve non-trivial collaborative, organizational goals. Through this perspective and in the context of our user studies, we can interpret the question as a question of user tendency: *What means of interactions with multimedia content* ***did the users tend to do to achieve a collaborative goal?*** We can further specify the question by clearly defining what "means of interaction with multimedia content" means in the context of the implemented application. In accordance with the question's aim, we define this term as interaction types that change the state, whether local or synchronized, of an image's virtual representation. As such, sharing operations are not included in the discussion of this question, since they only manipulate temporary copies of those representations in order to highlight certain multimedia assets to other collaborators. Consequently, this question refers only to the implementations of the organizational interaction types and multimedia-specific interaction types of the interaction taxonomy presented in section 3.5.

In order to provide insight into **RQ1**, we interpreted the results gathered from ***Task 3A- Organize the dog photos according to their breed*** (section 5.7.4). This task presents the user with a non-trivial collaborative, organizational goal that is achieved by moving the photos into their respective places. In this context, the action of moving the photos is essential, but using folder organization is also possible. Also, since some dog photos are ambiguous in representing that dog's breed, users are incentivized to analyze the dog photos in close detail. By gathering the usage rate of each action described (moving objects, using folders and viewing photos in detail), regarding **RQ1**, we can conclude that collaborative, multimedia-focused AR systems should include spatial organizational tools that let users manipulate the position of multimedia content, and interactions that let users analyze in detail the multimedia content.

RQ2 ponders what ways a user has of collaboratively using multimedia content in a AR system. Collaboratively using multimedia content describes the process of interacting with the content such that the resulting action provides not only progress toward the collaborative goal but also the exchange of information or resources between two or more collaborators. In the context of the goals of the use-cases (section 3.4), we consider the collaborative usage of photos to be showing them to other collaborators, or in other words, sharing them. Similarly to **RQ1**, we can reformulate **RQ2** to be based on a question of tendency: *How do users **tend to share** multimedia content in the VE to reach a shared goal?*

To provide insight into **RQ2**, we leveraged the variety of possible solutions that ***Task 3B- Share specific photos*** offers and interpreted its results, present in section 5.7.4. Since the implemented application offers various ways of sharing, we can effectively measure which ways users tend to do the most by comparing their usage rates. As such, regarding **RQ2**, we can see that users tend to share multimedia content via their relative location, in the context of a marker-based in-person application, and users tend to present that content to all other users in a manner that allows each user to see it in the best way for that multimedia type to be seen.

To validate the application's implementation, we can conclude from the results gathered from the system questionnaire (section 5.7.5) that the application has a significantly higher-than-average SUS score, which validates the application's implementation as a whole. However, in the context of **RQ1**, due to the low average scores of the folder-related interactions in **Phase 1** testing, and their relative absence in **Task 3A**, we can't conclude with a high degree of confidence that the folder's implementation was adequate, and, as such, we can't conclude about the tendency of users regarding folders. We can, however, conclude that users thought the spatial manipulation actions were easy, and that corroborates their extensive usage in **Task 3A**. We can also conclude that viewing in detail as a multimedia-specific interaction is adequate, since it was used in a majority of cases in **Task 3A**, and presented good individual task scores in **Task 1K,1L and 1M**.

Furthermore, during or after the evaluation process, some users provided feedback on the application. Some users (n=2) mentioned the application lacking information regarding the avatars, specifically being able to see their own anthropomorphic avatars.

Knowing about their respective avatar's whereabouts is mainly useful in two situations: when previewing another user sharing images with them and when the avatar's location needs to be communicated.

A subset of users (n=4) mentioned that they experienced a lack of feedback when sharing images with the tester in the context of ***Task 1I- Sharing to a Neighbour***. Although there were feedback mechanisms implemented, users in this group said that their activation and display were either too subtle or in a location within the interface where they were not looking at the time. Some users (n=2) reported the same lack of feedback when doing **Task 1H- Sharing via an avatar**: one chose the anthropomorphic avatar to share, while the other chose the phone-like avatar. Considering the distribution of avatars chosen by users, $\approx 9\%$ of users who chose the anthropomorphic avatar reported it, while 100% of users who chose the phone-like avatar reported the lack of feedback.

The tester observed that, generally, users had more difficulties when interacting with the general mode and spatial mode interfaces compared to the other interface modes. One user explained their performance was hindered by the lack of a tutorial within the application and the lack of visual indicators for spatial mode and general mode interactions.

A subset of users (n=5), when doing **Phase 0**, would move their phone closer to the virtual location of a photo to take a better look at it. Most of these users (n=4) didn't apply this behaviour in **Task 3A**, and used the viewing in detail interaction to achieve the same goal. This further legitimizes the usefulness of the view in detail interaction, and, consequently, the need for multimedia-specific interactions in this type of application.

More than 50% of test subjects (n=7) tried to drag the images when trying to move them in ***Phase 0***. This behaviour remained when doing ***Task 1F- Storing an image in a folder*** and ***Task 1H- Sharing via an avatar***. The tester also observed users (n=3) who previously didn't have this behaviour develop it in the context of the previously mentioned tasks. This phenomenon may indicate that users relate the dragging motion with storing things. This could be the result of a predisposition that originated from smartphone home screen application organization, which uses this gesture to move application shortcuts, and create folders if two shortcuts are to occupy the same place. However, it could also be an indication that users prefer the precision that dragging, as a gesture, offers for movement, and in a case where that precision is needed, such as the mentioned tasks, users default to this method of item movement.

6

Conclusion

This thesis presents an interaction taxonomy based on the needs of relevant use cases in collaborative multimedia-focused areas and workflows, as well as a set of characteristics to support collaborative work in a marker-based, in-person MAR system. After evaluating one possible marker-based, in-person implementation of this taxonomy, we can conclude that a great majority of the taxonomy's features, when implemented in a synchronous environment, contribute to collaborative multimedia work. We explored the effects of avatars of users within a MAR setting and can conclude that their inclusion can aid collaboration efforts through the encapsulation of feedback mechanisms and providing mediums of interaction. Specifically, we observed that, in a real-time AR system, avatars provide a way of increasing the interpretability of spatial manipulation actions and the ability to identify the author of those actions. We can also conclude that the introduction of multimedia-specific interactions within this kind of system can help users analyze multimedia content better. Even though the implemented application provided only photo manipulation, we explored possible implementations of AR manipulation of multiple multimedia types. Furthermore, we have provided insight into possible spatial manipulation methods for multimedia types, highlighting their applicability for two-dimensional multimedia assets. We developed a user study protocol that leverages the simultaneous presence of a virtual environment and a real-world one that AR offers to represent possible uses of the application in photo categorization. Finally, based on the results of the user study, we observed that MAR provides high efficacy in improving already existing collaborative multimedia workflows.

6.1 Future Work

Moving forward, a point of exploration that should be further developed in future works on this topic is the research of AR display and interaction techniques for different multimedia types, primarily video. Video, being a temporal medium, needs special consideration when implementing it in synchronous systems. Following this thought process, the study of interfaces that allow for the coexistence of multiple multimedia types within

the same AR system should also be considered. Another aspect that could provide future studies is the evaluation of the influence that the AR marker's position in the real world has upon collaborative, multimedia-focused marker-based applications.

In the context of spatial manipulation of AR objects, alternative selection methods, especially ones that select multiple objects, should be considered for future research, since it could lead to higher user expressiveness and efficiency when organizing objects. These methods of selection could lead to facilitating interaction in AR scenes with a higher number of objects.

In the context of implementing marker-based collaborative systems, a methodology for runtime marker creation should be explored, as it increases the real-world applicability of the concepts detailed throughout this document since it would remove the restriction of users having a predetermined marker with them to use the application.

Throughout this dissertation, we explored the unique characteristics and challenges of the MAR collaborative multimedia-focused paradigm, providing readers with an overview of interface and interaction design philosophy regarding this paradigm. We hope that the work developed here serves to inspire new design principles, methodologies, and applications for encouraging collaboration through the sharing of multimedia. We hope to have provided the tools to increase the informational throughput of any multimedia-focused collaborative effort, after all, a picture is worth a thousand words.

Bibliography

[1] A. Agarawala and R. Balakrishnan. *Keepin' It Real: Pushing the Desktop Metaphor with Physics, Piles and the Pen*. 2006. URL: www.dgp.toronto.edu (cit. on p. 19).

[2] M. Augstein and T. Neumayr. "A Human-Centered Taxonomy of Interaction Modalities and Devices". In: *Interacting with Computers* 31 (1 2019), pp. 27–58. ISSN: 09535438. DOI: 10.1093/iwc/iwz003 (cit. on pp. 20, 21).

[3] R. T. Azuma. *A Survey of Augmented Reality*. 1997, pp. 355–385. URL: http://www.cs.unc.edu/~azumaW: (cit. on pp. 1, 6, 10).

[4] O. Baus and S. Bouchard. "Moving from virtual reality exposure-based therapy to augmented reality exposure-based therapy: A review". In: *Frontiers in Human Neuroscience* 8 (MAR 2014). ISSN: 16625161. DOI: 10.3389/fnhum.2014.00112 (cit. on pp. 14, 15).

[5] M. K. Bekele, R. Pierdicca, E. Frontoni, E. S. Malinverni, and J. Gain. "A survey of augmented, virtual, and mixed reality for cultural heritage". In: *Journal on Computing and Cultural Heritage* 11 (2 Mar. 2018). ISSN: 15564711. DOI: 10.1145/3145534 (cit. on p. 13).

[6] C. Benoit, J.-C. Martin, C. Pelachaud, L. Schomaker, and B. Suhm. *Audio-visual and Multimodal Speech Systems*. Esprit/BRA project MIAMI - Multimodal Integration in Multimedia Interfaces, 1999 (cit. on pp. 21, 22).

[7] M. Billinghurst, A. Clark, and G. Lee. "A survey of augmented reality". In: *Foundations and Trends in Human-Computer Interaction* 8 (2-3 2014), pp. 73–272. ISSN: 15513963. DOI: 10.1561/1100000049 (cit. on p. 1).

[8] M. Billinghurst, R. Grasset, and J. Looser. "Designing augmented reality interfaces". In: *Computer Graphics (ACM)* 39 (1 2005), pp. 17–22. ISSN: 00978930. DOI: 10.1145/1057792.1057803 (cit. on p. 12).

[9] M. Billinghurst, H. Kato, and I. Poupyrev. "Tangible Augmented Reality". In: (2008) (cit. on p. 13).

[10] J. Brooke. "SUS: A quick and dirty usability scale". In: *Usability Eval. Ind.* 189 (Nov. 1995) (cit. on p. 72).

[11] D. Chatzopoulos, C. Bermejo, Z. Huang, and P. Hui. "Mobile Augmented Reality Survey: From Where We Are to Where We Go". In: *IEEE Access* 5 (2017), pp. 6917–6950. ISSN: 21693536. DOI: 10.1109/ACCESS.2017.2698164 (cit. on pp. 8, 10, 11).

[12] L. Damiania, M. Demartini, P. Giribone, M. Maggiani, R. Revetria, and F. Tonelli. "Simulation and Digital Twin Based Design of a Production Line: A Case Study". In: 2018. ISBN: 9789881404886 (cit. on p. 15).

[13] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. "MonoSLAM: Real-time single camera SLAM". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29 (6 2007), pp. 1052–1067. ISSN: 01628828. DOI: 10.1109/TPAMI.2007.1049 (cit. on p. 9).

[14] Z. Dong, J. Zhang, X. Bai, A. Clark, R. W. Lindeman, W. He, and T. Piumsomboon. "Touch-Move-Release: Studies of Surface and Motion Gestures for Mobile Augmented Reality". In: *Frontiers in Virtual Reality* 3 (2022). ISSN: 26734192. DOI: 10.3389/frvir.2022.927258 (cit. on pp. 12, 23, 24).

[15] D. Eggert, D. Hücker, and V. Paelke. *Augmented Reality Visualization of Archeological Data*. 2014, pp. 203–216. DOI: 10.1007/978-3-642-32618-9_15 (cit. on p. 13).

[16] C. Faure, A. Limballe, and H. Kerhervé. "Fooling the Brain, Fooling the Pain: The Role of Mirror Therapy and Modern Uses in Virtual Reality". In: *Frontier for Young Minds* (2019). DOI: 10.3389/frym.2019.00091 (cit. on p. 14).

[17] Q. H. Gao, T. R. Wan, W. Tang, and L. Chen. "A Stable and Accurate Marker-less Augmented Reality Registration Method". In: (2017) (cit. on p. 10).

[18] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez. "Automatic generation and detection of highly reliable fiducial markers under occlusion". In: *Pattern Recognition* 47 (6 2014), pp. 2280–2292. ISSN: 00313203. DOI: 10.1016/j.patcog.2014.01.005 (cit. on p. 8).

[19] S. Gauglitz, B. Nuernberger, M. Turk, and T. Höllerer. "In touch with the remote world: Remote collaboration with augmented reality drawings and virtual navigation". In: Association for Computing Machinery, 2014, pp. 197–205. ISBN: 9781450332538. DOI: 10.1145/2671015.2671016 (cit. on pp. 16, 21).

[20] J. J. Gibson. "The theory of affordances". In: *Hilldale, USA* 1.2 (1977), pp. 67–82 (cit. on p. 35).

[21] E. S. Goh, M. S. Sunar, and A. W. Ismail. "3D object manipulation techniques in handheld mobile augmented reality interface: A review". In: *IEEE Access* 7 (2019), pp. 40581–40601. ISSN: 21693536. DOI: 10.1109/ACCESS.2019.2906394 (cit. on pp. 12, 21–25).

[22] J. Gregory. *Game Engine Architecture*. 3rd ed. Taylor & Francis, 2018. DOI: https://doi.org/10.1201/9781315267845. URL: http://taylorandfrancis.com (cit. on p. 28).

[23] H. P. R. Group. *TASK LOAD INDEX (NASA-TLX) v 1.0* (cit. on p. 72).

[24] T. Haritos and N. D. Macchiarella. *A MOBILE APPLICATION OF AUGMENTED REALITY FOR AEROSPACE MAINTENANCE TRAINING*. 2005 (cit. on p. 15).

[25] J. Hertel, S. Karaosmanoglu, S. Schmidt, J. Bräker, B. Bräker, M. Semmann, and F. Steinicke. *A Taxonomy of Interaction Techniques for Immersive Augmented Reality based on an Iterative Literature Review*. 2021 (cit. on p. 12).

[26] H. Kaufmann and D. Schmalstieg. "Mathematics And Geometry Education With Collaborative Augmented Reality". In: (2003) (cit. on p. 18).

[27] H. Koyuncu and S. H. Yang. *A Survey of Indoor Positioning and Object Locating Systems*. 2010, p. 121 (cit. on p. 10).

[28] T. Lee and T. Hollerer. "Handy AR: Markerless Inspection of Augmented Reality Objects Using Fingertip Tracking". In: *2007 11th IEEE International Symposium on Wearable Computers*. 2007, pp. 83–90. DOI: 10.1109/ISWC.2007.4373785 (cit. on p. 12).

[29] Y. Li, D. Hicks, W. S. Lages, S. W. Lee, A. Sharma, and D. A. Bowman. "ARCritique: Supporting remote design critique of physical artifacts through collaborative augmented reality". In: Institute of Electrical and Electronics Engineers Inc., 2021, pp. 585–586. ISBN: 9780738113678. DOI: 10.1109/VRW52623.2021.00175 (cit. on pp. 17–19).

[30] C. Liu, J. Xu, and F. Wang. "A Review of Keypoints' Detection and Feature Description in Image Registration". In: *Scientific Programming* 2021 (2021). ISSN: 10589244. DOI: 10.1155/2021/8509164 (cit. on p. 7).

[31] J. Liu, O. K. C. Au, H. Fu, and C. L. Tai. "Two-finger gestures for 6DOF manipulation of 3D objects". In: vol. 31. Eurographics Association, 2012, pp. 2047–2055. DOI: 10.1111/j.1467-8659.2012.03197.x (cit. on p. 23).

[32] T. Madeira, B. Marques, J. Alves, P. Dias, and B. S. Santos. *Exploring Annotations and Hand Tracking in Augmented Reality for Remote Collaboration*. 2020 (cit. on p. 12).

[33] B. Marques, S. Silva, A. Rocha, P. Dias, and B. S. Santos. "Remote asynchronous collaboration in maintenance scenarios using augmented reality and annotations". In: Institute of Electrical and Electronics Engineers Inc., 2021, pp. 567–568. ISBN: 9780738113678. DOI: 10.1109/VRW52623.2021.00166 (cit. on pp. 16, 17).

[34] L. Marques and J. Roca. "Three-Dimensional Modelling for Cultural Heritage". In: Apr. 2021, pp. 418–443. DOI: 10.4018/978-1-7998-2249-3.ch014 (cit. on p. 13).

[35] R. Mur-Artal, J. M. Montiel, and J. D. Tardos. "ORB-SLAM: A Versatile and Accurate Monocular SLAM System". In: *IEEE Transactions on Robotics* 31 (5 2015), pp. 1147–1163. ISSN: 15523098. DOI: 10.1109/TRO.2015.2463671 (cit. on p. 10).

[36] S. A. Nikou. *Web-based videoconferencing for teaching online: Continuance intention to use in the post-COVID-19 period*. 2021 (cit. on p. 16).

[37] D. Nincarean, M. B. Alia, N. D. A. Halim, and M. H. A. Rahman. "Mobile Augmented Reality: The Potential for Education". In: *Procedia - Social and Behavioral Sciences* 103 (2013), pp. 657–664. ISSN: 18770428. DOI: 10.1016/j.sbspro.2013.10.385 (cit. on p. 16).

[38] H. Olafsdottir and C. Appert. "Multi-touch gestures for discrete and continuous control". In: Association for Computing Machinery, 2014, pp. 177–184. ISBN: 9781450327756. DOI: 10.1145/2598153.2598169 (cit. on pp. 22, 23).

[39] M. Ortiz-Catalan, R. A. Guðmundsdóttir, M. B. Kristoffersen, A. Zepeda-Echavarria, K. Caine-Winterberger, K. Kulbacka-Ortiz, C. Widehammar, K. Eriksson, A. Stockselius, C. Ragnö, Z. Pihlar, H. Burger, and L. Hermansson. "Phantom motor execution facilitated by machine learning and augmented reality as treatment for phantom limb pain: a single group, clinical trial in patients with chronic intractable phantom limb pain". In: *The Lancet* 388 (10062 2016), pp. 2885–2894. ISSN: 1474547X. DOI: 10.1016/S0140-6736(16)31598-7 (cit. on p. 14).

[40] V. Pereira, T. Matos, R. Rodrigues, R. Nóbrega, and J. Jacob. "Extended Reality Framework for Remote Collaborative Interactions in Virtual Environments". In: *2019 International Conference on Graphics and Interaction (ICGI)*. 2019, pp. 17–24. DOI: 10.1109/ICGI47575.2019.8955025 (cit. on p. 18).

[41] I. Poupyrev, D. S. Tan, M. Billinghurst, H. Kato, H. Regenbrecht, and N. Tetsutani. "Developing a generic augmented-reality interface". In: 35 (3 Mar. 2002), pp. 44–50. ISSN: 00189162. DOI: 10.1109/2.989929 (cit. on pp. 12, 20).

[42] C. Qiu, S. Zhou, Z. Liu, Q. Gao, and J. Tan. "Digital assembly technology based on augmented reality and digital twins: a review". In: *Virtual Reality and Intelligent Hardware* 1 (6 2019), pp. 597–610. ISSN: 26661209. DOI: 10.1016/j.vrih.2019.10.002 (cit. on p. 15).

[43] A. Sadeghi-Niaraki and S. M. Choi. "A survey of marker-less tracking and registration techniques for health & environmental applications to augmented reality and ubiquitous geospatial information systems". In: *Sensors (Switzerland)* 20 (10 2020). ISSN: 14248220. DOI: 10.3390/s20102997 (cit. on pp. 8–10).

[44] D. Schmalstieg and A. Fuhrmann. "Concept and Implementation of a Collaborative Workspace for Augmented Reality". In: *GRAPHICS '99* 18 (3 1999) (cit. on pp. 1, 18).

[45] D. Schmalstieg and D. Wagner. "Experiences with Handheld Augmented Reality". In: (2007). URL: http://www.studierstube.org (cit. on pp. 11, 15).

[46] M. A. Schnabel, X. Wang, H. Seichter, and T. Kvan. "From Virtuality to Reality and Back". In: 2007 (cit. on pp. 6, 7).

[47] V. Teichrieb, J. P. Silva, M. Lima, E. L. Apolinário, T. Souto, M. C. D. Farias, M. Augusto, S. Bueno, J. Kelner, and I. H. F. Santos. *A Survey of Online Monocular Markerless Augmented Reality*. 2007. URL: http://www.cin.ufpe.br/ (cit. on p. 9).

[48] H. Tyni, A. Kultima, and F. Mäyrä. "Dimensions of Hybrid in Playful Products". In: *Proceedings of International Conference on Making Sense of Converging Media*. AcademicMindTrek '13. Tampere, Finland: Association for Computing Machinery, 2013, pp. 237–244. ISBN: 9781450319928. DOI: 10.1145/2523429.2523489 (cit. on p. 13).

[49] D. Wagner, T. Langlotz, and D. Schmalstieg. "Robust and unobtrusive marker tracking on mobile phones". In: 2008, pp. 121–124. ISBN: 9781424428403. DOI: 10.1109/ISMAR.2008.4637337 (cit. on p. 7).

[50] H. Wang and B. E. Shi. "Gaze awareness improves collaboration efficiency in a collaborative assembly task". In: Association for Computing Machinery, 2019. ISBN: 9781450367097. DOI: 10.1145/3317959.3321492 (cit. on pp. 12, 19).

[51] L. Wang, X. Liu, and X. Li. "VR collaborative object manipulation based on viewpoint quality". In: Institute of Electrical and Electronics Engineers Inc., 2021, pp. 60–68. ISBN: 9781665401586. DOI: 10.1109/ISMAR52148.2021.00020 (cit. on p. 19).

[52] L. Westendorf, O. Shaer, P. Varsanyi, H. V. D. Meulen, and A. L. Kun. "Understanding collaboration around large-scale interactive surfaces". In: *Proceedings of the ACM on Human-Computer Interaction* 1 (CSCW 2017). ISSN: 25730142. DOI: 10.1145/3134745 (cit. on pp. 17, 18).

[53] A. S. Williams, J. Garcia, and F. Ortega. "Understanding Multimodal User Gesture and Speech Behavior for Object Manipulation in Augmented Reality Using Elicitation". In: *IEEE Transactions on Visualization and Computer Graphics* 26 (12 2020), pp. 3479–3489. ISSN: 19410506. DOI: 10.1109/TVCG.2020.3023566 (cit. on p. 12).

[54] L. Yang, D. Holtz, S. Jaffe, S. Suri, S. Sinha, J. Weston, C. Joyce, N. Shah, K. Sherman, B. Hecht, and J. Teevan. "The effects of remote work on collaboration among information workers". In: *Nature Human Behaviour* 6 (1 2022), pp. 43–54. ISSN: 23973374. DOI: 10.1038/s41562-021-01196-4 (cit. on pp. 2, 16).

[55] Z. Zhang, Y. Hu, G. Yu, and J. Dai. "DeepTag: A General Framework for Fiducial Marker Design and Detection". In: (2021). URL: http://arxiv.org/abs/2105.13731 (cit. on p. 7).

I

User Consent Form

In order to take this test you must be aware of what will happen, what data will be collected and how it will be processed and, if you agree, you consent to this testing session.

Health Concerns: The testing session will have you use an AR application on a smartphone. All sessions use the same smartphone, leading to other participating parties touching the smartphone. To this effect, the phones are disinfected between each session. Since the application is AR, some people may experience nausea during its usage. During any point in the session, you can withdraw from it and rest.

Data Concerns: All data collected will be used only in the context of this thesis. It will be anonymized before processing. Collected information during the session consists of my visual observations, our verbal communication, data collected by the system (timings, positions, etc.) the answers to the questionnaire below, as well as screen recordings of the smartphone during the session.

Anytime during the session you can ask questions regarding our session or quit.

By choosing "I consent" below, you agree to voluntarily participate in this academic study, agree to the collection and processing of your data and that you are informed about the health risks of participating.

[] - Do Not Consent [] - I Consent