VRIJE
UNIVERSITEIT
BRUSSEL

Master thesis submitted in partial fulfilment of the requirements for the degree of
Master of Science in Applied Sciences and Engineering: Computer Science

MIXED REALITY-BASED INTERACTION FOR THE WEB OF THINGS

Elena Zambon
0561862

Academic Year 2020/2021

Promotor: Prof. Dr. Beat Signer
Science and Bio-Engineering Sciences

"*This master's thesis came about (in part) during the period in which higher education was subjected to a lockdown and protective measures to prevent the spread of the COVID-19 virus. The process of formatting, data collection, the research method and/or other scientific work the thesis involved could therefore not always be carried out in the usual manner. The reader should bear this context in mind when reading this Master's thesis, and also in the event that some conclusions are taken on board.*"

Abstract

This work aims to analyse and summarise the interaction techniques used for the different classes of the augmented reality (AR) devices available on the market, such as smartphones, head-mounted displays, smart glasses and spatial displays. The outcome of this research has been useful to design the interface we want to propose as a tool to control smart objects present in the physical environment and create dependencies and rules between them. In particular, we could exploit these foundings to make sure that the techniques to interact with the application were the most efficient and straightforward possible to enable users to easily access its functionalities. Given that the interface works in an augmented reality setting and takes advantage of the information retrieved directly from the appliances in the Internet of Things (IoT), these functionalities consist of manipulating the superimposed objects in AR in order to alter the behaviour of the real smart objects in the surroundings. By defining personalised rules, the user can control the properties of Things in an implicit manner. Hence, performing some explicit action on an object might trigger the system to carry out some other action on a set of smart devices behind the scenes. An example could be to define a rule that makes sure that when the user turns on the TV (explicit interaction), the light is automatically switched off by the system (implicit interaction). Explicit interactions can be performed either through the interface itself or by using the Things in the physical surroundings. By enhancing their characteristics, this work brings the worlds of AR and IoT together to improve the experience of both experts and novice users in their smart environments. In fact, by proposing an intuitive interface running on any camera-equipped device, we aim to bridge the gap between non-technical users and automation systems.

Acknowledgments

I devoted most of my time between September 2020 and June 2021 to the draft of this master's thesis, which is the highest amount of time it has ever taken me to complete any other project in my life. Looking back at my whole period here at the Vrije Universiteit Brussel, I realise how fast these two years have past, and how intense and challenging this master's degree has been. The lack of social relationships and the fact that lectures and meetings were held online due to the Covid-19 pandemic have made everything harder than usual, and the fear of failure was persistent. This is why, maybe for the first time in my life, I am sincerely proud of myself and what I achieved.

This is also the first occasion in which I would like to open the acknowledgments by thanking myself, for not giving up, being determined, putting my heart and soul in every project, test, exam I underwent, and for the "go big or go home" mentality. However, I could have never done it without the support of my friends, who have always been there to listen to me, offer some help and advise, and who believe in me when I lack motivation and self-esteem. But most importantly, they constantly remind me that I am loved and, no matter the circumstances or the distance that separates us, they show me that they care for me, which is the fuel that got me this up. I am using this thesis as a means to say that I love you all.

Commitment, perseverance, determination and sacrifice. Ever since I started university in 2015, I have always thought that these are the four ingredients necessary to accomplish any goal one wants to reach. And this thesis is another proof of it. It represents not only the end of this Master of Computer Science Engineering, but also the conclusion of the chapter of my life called *University*.

It is time to leave the academic world and discover what the future holds for me.

Contents

1 Introduction

Augmented reality (AR) is a technique that consists of superimposing some computer-generated geometric models, images, sounds and text to the physical world [1]. It allows the user to see the real environment surrounding them, but with the addition of digital content visualised by means of devices such as see-through head-mounted displays (HMD), desktop and mobile interfaces, and monocular systems [2, 3].

While virtual reality (VR) replaces the real world with a completely artificial setting, AR supplements it, enabling the user to still have a perception of living in and dealing with their reality, but with the extension of virtual content. Hence, not only does AR enhance the connection between human and environment, but it also facilitates human-computer interactions (HCI). With this regard, throughout the last two decades, AR has been exploited in many different domains, ranging from health care, manufacturing and navigation, to museums, education, entertainment, sports and gaming [4].

With the spread of small Internet-connected devices all over the world, everyone has access to the Internet everywhere and at any time [5], leading to the notion of *Internet of Things* (IoT) or *Web of things* (WoT). The IoT is revolutionising industries such as manufacturing, transportation and energy, by interconnecting digital devices, and retrieving and exchanging information among them.

An example of an application field that has been emerging recently in the IoT context is *home automation*. A home automation system enables users to control and interact with the technological appliances (e.g. dish washer, microwave and tv) of their so-called *smart homes* [6], through the Internet.

The leading research and advisory company Gartner[2], forecast that *14.2 billion connected things would be in use in 2019, and that the total will reach 25 billion by 2021, producing an immense volume of data*[3]. However, as per Security Today[4],

[2] `https://www.gartner.com/en`

[3] Gartner Identifies Top 10 Strategic IoT Technologies and Trends. `https://www.gartner.com/en/newsroom/press-releases/2018-11-07-gartner-identifies-top-10-strategic-iot-technologies-and-trends`

[4] `https://securitytoday.com/Home.aspx`

in 2019 the number of active IoT devices reached 26.66 billion, and every second 127 new IoT devices are connected to the Web. They also predicted that by 2021 35 billion IoT devices will be installed worldwide, against the 25 billion foreseen only two years ago[1]. Comparing the numbers, the Internet of Things is growing much faster than expected, increasing the possibility for people, things, and services to become closely and rapidly connected [7].

Some of the most recent research has been tackling the challenge of integrating AR and IoT [8]. In fact, although AR and IoT might seem unrelated, given that they focus on different concepts, they can indeed be complementary one to the other.

In this thesis we aim to investigate a method to combine AR and IoT, in a way that allows users to detect the electronic devices in their environment and interact with individual devices via an AR interface as well as define implicit interactions (via rules) between multiple IoT devices.

A first step is to analyse the current state-of-the-art research to compare the displays and techniques that are used in AR settings, together with their advantages and disadvantages. In this way we will get a better understanding of which approaches might be most suitable for the above-mentioned idea.

Subsequently, the focus is on studying a way to connect multiple devices together in an IoT setting. Currently the Web of Things applications allow the user to control one appliance at a time and the interconnection between smart devices is rather limited. This means that the Things' properties cannot be altered automatically in case some events and situations (e.g. the user is on the couch or the TV is currently on) take place.

Finally, we combine augmented reality and Internet of Things by implementing the interface previously described. The idea is to exploit the Internet of Things to detect and control the surrounding appliances through camera-equipped technologies such as headsets, smart glasses and smartphones, and the augmented reality to superimpose graphical hints or objects that allow the user to link them and make them communicate. Therefore, by collecting the data from one device and sending it to another, the functionality of the first could be used by the second, pushing the AR concept of enhancing reality even further. This also entails that the interaction between different devices might not only be implicit by applying some predesigned rules (e.g. if the tv is on, the music on the stereo is turned off), but it is also possible for the user to explicitly define an interaction by following the suggestions

[1]The IoT Rundown For 2020: Stats, Risks, and Solutions. `https://securitytoday.com/Articles/2020/01/13/The-IoT-Rundown-for-2020.aspx?Page=2`

in AR with their hands (e.g. drag a document and drop it to the printer through the gesture recognition of the AR interface, in order to print that document).

1.1 Problem Statement

The rapid evolution that is happening every day in the IoT domain and the spread of more sophisticated devices and technologies to interact with, have raised many questions and new needs. As described above, the Web of Things aims to control and monitor the equipment we use in our daily lives through the Internet [9]. However, the objects we interact with are constantly changing, improving or even being replaced by other up-to-date refined ones. Hence, the need of new guidelines and solutions as to how we could interface with smart objects is a natural outcome.

One of the main problems that we try to address in this thesis is the fact that, to our knowledge, all the existing developed solutions to engage with IoT appliances focus on the interaction with one single object at a time, rather than multiple ones. We can control whether the light in the living room is on, and we can turn off the coffee machine, but we cannot define a rule that would control both objects simultaneously. In fact, as stated in [10], *"we are witnessing an explosive growth of the number and density of powerful mobile devices around us. However, their great majority are still blind to the presence of other devices and performing tasks among them is usually tedious and lacks recognizable guiding principles". Home Assistant* is slightly moving toward this direction, adding logic and dependencies between smart objects. However, controlling and customising the application are tasks that need to be performed by someone with enough technical knowledge, which might be intimidating for those approaching this domain for the first time.

Linked to the previous issue is the fact that different objects are not able to communicate with each other, which represents an important limitation in terms of usability. We live in a world fully connected, where we can engage with devices that complement each other in relation to their functionalities, but we still do not have the chance to combine appliances together in order to make them exchange information and resources.

Furthermore, for what concerns the AR side, existing solutions have only one fixed set of interactions, independently of what particular device one is currently using [11]. In other words, the available implementations do not take into consideration that, for different AR solutions, different types of interaction might be required. A head-mounted display might work better with a certain interaction technique, which in turn might not be very suitable for a handheld device.

Finally, what is missing in current automation systems (e.g. Amazon Alexa and Google Assistant) is a proper graphical user interface that makes the control and manipulation of smart objects more efficient and intuitive. Potential end users, such as the elderly, people with little knowledge of Computer Science or with physical disabilities, still have no access to an easy and straightforward way to visualise their smart appliances on a display.

Our research addresses the issues just presented by (1) comparing the different AR devices available and identify the most effective interaction techniques, (2) investigating a way that allows the interaction between multiple objects in an IoT setting, and (3) develop a simple interface that enables the user to interact with multiple appliances by combining their functionalities via implicit human computer interaction.

1.2 Contributions

Given the problem statement presented in the previous section and the corresponding proposed solution, we can identify the main contributions of this work as follows:

- The delineation of some guidelines specific to each augmented reality display in a way that their usability and potential are maximised. As mentioned above, currently only one set of interaction techniques is available for all devices, without differentiating which one is more suitable for a given type of augmentation. Identifying how to best deal with a specific AR hardware when manipulating the virtual objects is expected to increase the performance of the model and satisfaction of the user.
- The exploration of the possibility to interconnect multiple devices in an IoT setting. This is achieved through the definition of rules that outline implicit interactions by checking whether the appliances are allowed or not allowed to connect, and the definition of rules created by the users while utilising the application, which allow them to make the interaction across smart objects explicit.
- The implementation of an AR interface which brings together the two points above. This prototype allows the user to detect the smart objects in their room and interact with them logically. This is possible through the definition of personalised rules that create connections behind the scenes (implicit interactions), and through rules made concrete by users themselves (explicit interactions). In fact, the interface shows some hints regarding which devices

can be combined, preventing the user to try to connect appliances that are not expected to exchange data. By following these patterns, the user can complement the functionality of a smart object with the functionality of another.

- The illustration of some use cases in which the interface described above might come in handy in real world situations.

1.3 Methodology

The completion of this thesis was possible by collecting and analysing state-of-the-art work in the domains of mixed reality, Internet of Things and human-computer interactions. Studying how the research was carried out and what steps were taken during the last decades helped us understand the evolution of the above-mentioned fields throughout the history, and where the results of the current state come from. By putting all the relevant findings together, we could make comparisons and draw conclusions that led to the outcome of this thesis.

In particular, the *Research through Design* (RtD) methodology [12] was applied, which is often used where technology is designed with the purpose of illustrating how the future will or should look like. In fact, with this thesis we aim to propose a new interface that allows users to deal with smart objects in the most efficient and straightforward way possible, by means of particular interaction techniques. This is something that has not been implemented yet. To this date, the Internet of Things already plays an important role in our lives, but in the future it will probably be accessible by a larger number of people, making it even more omnipresent in our routine and in every environment. Hence, an important outcome of the research method adopted for this thesis is the definition of operations and guidelines necessary to show the utility and the validity of the application, as well as future research questions.

2 Background

In this chapter, we portray the relevant information concerning the topic in question. Thus, we explore the background in the domains of augmented reality (and mixed reality in a broader way), and the Web of Things. This preliminary step represents the foundation of the outcome of our research, which will be described in the next chapters.

As explicitly stated in the introduction, this thesis aims to look for a solution that brings the worlds of augmented reality and the Web of Things together. Hence, the first part of our work focused on exploring these two domains in terms of background and main research contributions that have led to the current state. However, we will not provide an insight of the history of these two fields because this would be out of the scope of what we want to achieve in this thesis.

2.1 Mixed Reality and Augmented Reality

Augmented reality (AR) has faced astonishing advances since the implementation of the first AR system in 1968 [13]. Since then, experts came up with many alternative or complementary definitions to describe what AR really is. However, the most popular, and probably most straightforward, one still remains the one proposed by Azuma in 1997: *"Augmented reality is a field in which 3D virtual objects are integrated into a 3-D real environment in real time"* [2]. Furthermore, Azuma added that AR is a variation of Virtual Environments (VE) [3] which brings us to the definition of virtual reality, and mixed reality in a broader way.

As per Costanza et al. [14], *virtual reality* (VR) is a technology that enables users to be immersed in a virtual, computer-generated world, losing the perception of the real environment surrounding them. This represents the main difference between AR and VR. The former, in fact, always happens in the physical space the user is currently in, which does not allow a fully immersive VR experience. The fact that the virtual objects are superimposed to the real environment entails that the user is not isolated from their familiar surroundings. On the other hand, VR systems create a new artificial reality that replaces the real world.

Both virtual and augmented reality belong to a broader concept called *mixed reality* (MR). Even though this notion has been around for a while, there is still no shared understanding of what MR actually represents and experts have not found a universal definition to describe this concept [15]. For instance, many consider mixed reality as a synonym for AR, but others disagree responding that MR enables walking into and manipulating a scene, whereas AR does not. In MR systems, users perceive both the physical (real) environment around them and the computer-generated elements presented through, for example, the use of semitransparent displays.

A definition of mixed reality which is commonly adopted to describe the concept is the one provided by Milgram and Kishino, who are the first to introduce this term in 1994 [16]. In their vision, mixed reality is *"a particular subset of Virtual Reality (VR) related technologies that involve the merging of real and virtual worlds somewhere along the "virtuality continuum" which connects completely real environments to completely virtual ones"*. This definition is visualised in Figure 2.1.

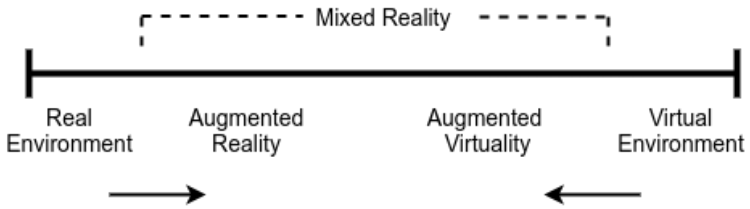


Figure 2.1: Graphical representation of the Reality-Virtuality continuum described by Milgram and Kishino [16]. It ranges from completely real to completely virtual environments and comprises Augmented Reality and Augmented Virtuality.

Therefore, MR systems are designed to give their users the illusion that digital and physical objects coexist in the same space. To achieve this, it is fundamental to precisely position digital objects into the real environment and align them with the real objects in real time, as stated in Azuma et al.'s work [17]. In particular, they declare that this alignment or registration of elements in virtual and real environments is a definitive characteristic of augmented reality systems. Thus, AR is often considered to be a branch of MR. In fact, both systems in which the virtual aspects are dominant and systems in which the physical reality is dominant fall under the definition of mixed reality. This means that, even though augmented reality has a higher physical aspect than a virtual one, it is still considered part of mixed reality [14].

Finally, the last concept comprised under the umbrella term mixed reality is *augmented virtuality* (AV). AV refers the inclusion of real world elements into a

virtual environment [18]. The difference between AR and VR boils down to where the user interaction takes place: while AR occurs in the real world, AV happens in a virtual environment.

Azuma [3] states that an AR system, to be defined as such, should present the three following properties:

- combination of real and virtual objects in a real environment;
- interactive and real-time execution; and
- registration (alignment) of real and virtual objects with each other.

Even though in this thesis we focus on augmented reality, we thought it was important to provide a short overview of what the broader concept of mixed reality comprises of and the differences with other related technologies.

2.1.1 Elements of Augmented Reality Systems

It should be clear by now that augmented reality (and mixed reality in a broader way) has a wide meaning that can have different interpretations. For this reason, it is not always clear what is meant or intended when people use AR. Liang [19] identifies six elements that should be included in an AR system and that ascertain the options available in the use of AR for particular activities. He points out that an architecture that comprises these elements might significantly facilitate the development and the innovation of the actual AR application.

The first element is the *user*. Users are in fact the core component of the AR architecture because they represent the direct beneficiary of any AR system. They are able to control and manipulate the AR system themselves, both individually and in group.

The second element is the *interaction*, where *inter* means the state between (or among) things, and *action* means that something has been concretely done, typically to achieve an aim. Hence, we refer to interaction when one entity performs an action and the other entity responds in a particular way. In the user-based AR system, usually the user does something that triggers some changes in terms of AR content visualised. This response might occur between users and AR device, or users and virtual content, both explained below. However, it is also true that in some cases this process of interaction does not exist anymore. In other words, users do not necessarily need to interact with the augmented reality device and the virtual content in order to see the augmentations in the real world.

The third element to be included in an AR architecture is the *device*, which is the hardware component of the AR architecture. Craig [20] identifies three hardware functions that every AR device must comprise: sensors, processors and displays. Sensors are responsible for the recognition of the state of the physical world where the application is deployed (e.g. GPS to identify the location and the orientation of the user). In fact, it is essential for the application to retrieve information of the real environment in real time, in order to properly respond to the user's actions. Craig identifies three primary categories of sensors in AR systems: (1) sensors used for tracking, (2) sensors for gathering environmental information, and (3) sensors for gathering user input. Processors are used to evaluate the data obtained by sensors, execute the instructions specified in the application program, and finally generate the signals required to drive the display. Processors can be seen as the brain of the technological system. Finally, displays represent the means with which users have the impression that the real and virtual worlds coexist. Thus, a display consists of a device that provides the signals perceived by our senses. Many different kinds of display are available, such as visual, audio, haptic and stereo displays.

The fourth element is the *virtual content*, which represents the digital information presented by the AR device. 3D animation, 2D image, text, audio information and vibration are some methods through which the digital content is visualised, and more generally communicated, to the user. Since we are in an AR setting, an important feature of virtual content to be pointed out is that the virtual information can be changed dynamically.

The fifth element is the *real content*, which is nothing more than the real-world information presented by the device without any rendering. This component allows the user to still have the perception of the real environment surrounding them.

The sixth and last element of AR systems is *tracking*, which is the way of generating virtual content based on the real content. This comprises three different features: synchronicity, because when the real content changes, the virtual content should change as well; antecedent, because the real content happens before the virtual content (if it was the other way around, the virtual element would be meaningless since it cannot be interpreted in the real world); and partial one to one, because there is a one-to-one correspondence between real and virtual content [21]. However, in some case, the real content is associated with multiple pieces of virtual information. Optical, acoustical, electromagnetic and mechanical tracking are some methods to perform tracking nowadays.

From a design perspective, Liang [22] adds that, in an AR system, the most critical problem is to select an appropriate physical-world context, identifying the method of transmission and creating different modalities of virtual content.

2.1.2 Object Positioning and Accuracy of Virtual Objects Perception

Both VR and AR have developed incredibly rapidly and find their application in many different domains. However, Ping et al. [23] identified two critical problems. The first one concerns the correctness of the virtual object's position in the scene; the second one regards the accuracy with which the user perceives the virtual objects in relation to the other (virtual and physical) objects in the scene. This is the reason why researchers and experts consider perceiving the position of virtual objects correctly in AR and VR a critical challenge.

A solution as to how to properly position virtual objects in AR is the use of visual markers or computer vision techniques, in order to create an understanding of the real environment and its physical elements [24].

Initially, AR content originated from marker-based tracking toolkits (e.g. ARToolkit or ARTag) that aim to determine tracking and registration (in other words, where to display the digital content) and the media content itself (what information needs to be displayed) [25]. However, marker-based tracking has slowly been replaced by markerless tracking. This is due to the fact that AR has evolved significantly over the last decades, registering huge technological advancements in terms of both hardware and software. Another outcome of this progress led to the diffusion of mobile context-aware methods (e.g. Layar, Junaio, and Wikitube) that can bring AR into mobile and contexts.

With regard to the two above-mentioned problems of implementing AR applications, Wang et al. [25] state that the "*biggest single obstacle to building effective AR systems is the requirement of accurate, long-range sensors and trackers*", identifying in optical tracking the tracking technology most used by developers.

Many different possibilities for implementing object tracking in AR are available, based on the nature of the tracking algorithms [26]:

i ID markers, which are rectangular 2D markers used for simple AR applications. They all follow the same structure with a wide black border and a distinctive pattern on the inside, such as the ones in Figure 2.2a. By exploiting all the different combinations of patterns, one can configure a few hundreds of markers with different encoded information. They represent the simplest but at the same time most efficient and robust method to detect and track an object. A comparison of ID markers used by some marker-based tracking toolkits is shown in Figure 2.2a.

ii 1D barcodes and Quick Response (QR) codes. They are optical 2D representations of data items that are machine-readable. One-dimensional barcodes have

been around since the 1960s and are still widely used for packaging, labeling and identifying retail and commercial products worldwide [27]. QR codes are read by imaging devices (e.g. any camera-equipped device with an installed QR reader application) and interpreted to extract information from the patterns. Examples of a linear barcode and a QR code is shown in Figures 2.2b and 2.2c respectively. As one can notice, ID markers and QR codes are similar.

iii Picture markers, that are somewhere in between ID markers and markerless tracking. A similarity with the former is that they have an easily distinguishable rectangular border as well. However, they can contain any arbitrary image (containing enough visual content) inside the boundary, which is not possible with ID markers. Additionally, the presence of this border enables the marker to be detected faster than with borderless markers, which represents a quite important advantage.

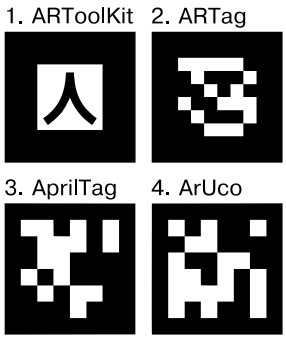
iv Markerless is the term used when referring to 2D markers that do not present an explicit rectangular boundary as the markers described above. An example is shown in Figure 2.2d.

v Markerless 3D tracking does not use any physical marker to identify an object. Instead, it uses a map of distinctive 3D features (e.g. point descriptors). This technique is less immediate than the one presented previously and the object needs to be scanned from different perspective in order to determine its own visual features. A concrete example is shown in Figure 2.2e

vi Computer-Aided Design (CAD) edge model tracking requires real-time six-degrees-of-freedom (DOF) pose estimation of the camera/objects. The AR system estimates in real time the location and orientation of the camera (pose) and subsequently superimposes computer-generated objects on the captured sequence of a real environment [28]. A 3D CAD model is used to initialise the pose of an object through its edges. This is achieved in order to enable an accurately scaled and localized augmentation. As soon as the pose of the objects is initialised, the tracking method is usually switched to markerless 3D tracking (see above). CAD edge model tracking requires extra attention to be prepared because it does not rely on any physical signs or picture. The user is responsible for the initialisation of the very first pose. This process is done once and for all manually.

2.1.3 Design Principles in AR Setting

Dünser et al. [29] state that a consequence of adopting an approach focused on technology might be the development of applications and prototypes without



(a) Examples of ID markers[1]

(b) Example of 1D barcode[2]

(c) Example of QR code[3]

(d) Example of borderless image[4]

(e) Example of markerless 3D tracking[5]

(f) Example of CAD edge model tracking[6]

Figure 2.2: AR tracking techniques

having a complete understanding of the problem they aim to solve. By taking into consideration the core purpose of the application and the public it is designed for, the development of (in this case) AR systems can significantly improve, becoming more successful and reaching a wider audience. This is called user-centred design.

Furthermore, not only defining guidelines might be beneficial for developers in an early stage, but it also prevents usability problems that may become evident later. Even though it sounds tempting, applying Graphical User Interface (GUI) design and evaluation principles to AR interfaces is not ideal. In fact, the two of them present some fundamental differences, especially in terms of interaction with the system itself.

[1] https://en.wikipedia.org/wiki/Augmented_reality
[2] https://commons.wikimedia.org/wiki/Barcode
[3] https://commons.wikimedia.org/wiki/Barcode
[4] https://redshift.autodesk.com/borderless-world
[5] https://www.youtube.com/watch?v=PtdxNAvso08
[6] https://www.jvrb.org/past-issues/4.2007/1159

We can therefore consider general Human-Computer Interaction (HCI) principles and knowledge derived from VR. In their work, Dünser et al. [29] identify eight design principles that might be useful in an AR setting.

Affordance is first design principle to be applied in AR. This concept "*suggests that there is an inherent connection between a user interface and its functional and physical properties*". In other words, affordance offers a conceptual model that describes a subject-object relationship. Tangible User Interfaces (TUIs) are an example of technology that reverses or negates meaning perceived from physical objects. Hiroshi and Ullmer [30] describe TUIs as a new type of HCI that "*augment the real physical world by coupling digital information to everyday physical objects and environments*". It is therefore clear that since the relation subject-object can change, it is necessary to precisely define it for a user study.

The second design principle is *reducing cognitive overhead*. The goal is to allow the user to focus as much as possible on the task they are carrying out, and therefore to reduce the effort used to interact with the application itself. In their work, Rizzo et al [31] explain how this extra non-automatic cognitive overhead on the interaction with the system might in fact represent a distraction to the user, limiting the quality of the VR (and AR) experience. Additionally, Dede et al. [32] prove that factors such as fatigue and cognitive overhead in mastering the interface influence negatively the outcome of the task fulfilled through virtual reality.

Linked to the previous principle, another important thing to take into consideration when designing an AR interface is to ensure that the task is carried out with *low physical effort*. The idea is to enable the user to reach the goal with the minimum number of interaction steps. In other words, the application should not ask the user for unnecessary interventions that might cause them fatigue and frustration. In his Ph.D. thesis, Kaufmann [33] demonstrates through some experiments and surveys how wearing a heavy or uncomfortable Head-Mounted Display (HMD) for a long time might be counterproductive in terms of success of the task to be accomplished. His findings show that the effects of wearing a mixed-reality device are subjective to the person using it, and they should all be taken into consideration when designing an AR interface. Fatigue, headache, nausea and dizziness are some words used by the participants of her study to describe their feelings during the experiment. These symptoms might in fact significantly reduce the usability of the system. A well-know example of fatigue manifestation is the so-called *gorilla arm*. We refer to gorilla-arm effect when mid-air interactions might cause fatigue and lead to a feeling of heaviness in the upper limbs [34].

The fourth HCI design principle that can be applied to AR development is *learnability*. It should be easy for a user to learn how to use the AR system in question.

Sometimes, designers might come up with new different interaction techniques that users are not familiar with. Since these new methods need to be learned by users in order to enable them to use the system properly, they must be as much intuitive as possible. Designing techniques similar to what users are used to in the real world or designing them in a way that reminds users of something they have already acquired through another application/technology (e.g. traditional computer interactions), might significantly reduce the time spent to learn them. Self descriptiveness is an important prerequisite that facilitates learnability. In fact, the more self-explanatory the interaction element is, the less time the user spends on understanding what it is for and how to use it. Another factor that improves learnability is consistency. An interface is considered inconsistent when different design choices are applied throughout the application (e.g. referring to a functionality in different ways or associating a feature or design element with different colours). This might cause confusion and uncertainty on the user side.

User satisfaction could come across as an obvious concept but the truth is that if a designer does not adopt a user-centred approach, they might forget taking into consideration this principle. It is in fact fundamental to always bear in mind that satisfaction is not an objective feeling and users may have a different opinion on an application compared to the one of who designed it in the first place. For this reason, both objective and subjective measurements should be made in order to offer an as much satisfactory as possible user experience.

Furthermore, *flexibility in use* is an aspect to be considered when designing AR interfaces. As stated above, all users are unique and have different opinions and preferences that the designer should try to accommodate. For instance, since the AR technology allows the integration of different kinds of input and output devices, users can choose one interaction modality rather than the other. More concretely, an AR system might include both gesture and speech options to carry out a task, thus leaving to the user the opportunity to pick the one they prefer or that simply come more in handy in that particular moment.

For what concerns *responsiveness and feedback*, the system should respond rather quickly to the user input and if it is not the case, it should at least provide an explanation of the reasons behind the delay. In this way, the user is informed and always have a clear idea as to what is happening. Additionally, not only the feedback should be sent by the system to help minimise problems caused by poor responsiveness, but it should also be provided any time the user performs an action, as a way to communicate that the input has been acknowledged and taken in charge. Thereby, the user is aware that they have triggered some functionality and they expect something to happen soon.

Finally, last but not least, the principle of *error tolerance* is essential to remind the designer to predict all scenarios where things can go wrong. Building an AR interface capable of managing errors whenever they occur (especially in terms of tracking, since we are in an AR setting) represents an effective way to gain the trust of the user with regard to the system reliability.

In addition to the eight HCI design principles above mentioned, Rolim et al. [35] propose further guidelines relative to the AR setting. A designer should bear in mind the importance of indicating movement by the application. In other words, an instruction must show the correct path (which sets the trajectory of the movement), the correctness of the movement (the right way to achieve the goal) and, in some cases, the velocity or acceleration. Moreover, if the action the user is expected to perform requires the exploitation of some part of the body or object in the scene, the system should be really clear in conveying the exact element involved in the task must be moved. Thus, misunderstandings about how to perform a gesture or instruction should be avoided. The AR interface should also allow different kinds of visual appearance attributes, which depend on the conditions of the environment, and which are controlled by users themselves. The last AR guideline proposed in the paper [35] is making sure to manage occlusion and depth. The former is the situation in which an object blocks the view of another object, whereas the latter is the distance between an object and the user. Handling occlusion in AR environments means comparing the depth of the pixels of the virtual objects with the depth of the pixels corresponding with real objects, which is not a trivial operation [36]. A system should provide all the necessary information and cues to the user to make them aware of objects hidden behind other objects, the distance that separates different elements in the scene, and the depth estimation of each virtual and real component, in order to have a clear perception of the environment surrounding them.

Dabor et al. [37] further suggest that the interface should be easy to use both for novice and for expert users (so-called *universal usability*). Designer should also support user control, meaning that users should have control of the next action to be performed, rather than the system carrying out pre-defined ones. This guideline, together with the possibility for the user to personalise the visual display, might considerably reduce the stress imposed on the user. Dabor et al. also add that the short-term memory should not be overloaded (e.g. low amount of steps to accomplish a task and minimise distracting audio output). Finally, as last guideline they suggest that context should be used to provide information, in order to convey the application messages in the most clear and straightforward way possible.

2.1.4 Mixed Reality Technologies and Displays

The mixed reality space can be designed and developed through two main types of technologies: optical see-through (OST) and video-see-through (VST). The main difference between the two is that with the former, a user can see the real world through the transparent glasses while the virtual objects are formed on LCD (Liquid Crystal Display); whereas with the latter, a user sees both real world and virtual objects on LCD [38, 39]. Technically, this happens in the following way. In VST displays, one or two video cameras are mounted near the user's eyes and are used to capture the real world. These real-world images are then combined with the computer-generated graphics and displayed to the user. On the other hand, in OST displays the semitransparent screens are placed in front of the user's eyes and optically merged with the digital graphics. Thus, with this technology, the perception of superimposed graphical element colours is affected by the colour and brightness of the real environment, which is not the case of VST displays [40].

In practice, OST devices are preferred over VST devices because considered simpler to be set up and because they give a better awareness of the real environment. In fact, since with VST displays the real world is not seen directly but as a digital capture, there might be some problems of latency and low resolution. This might lead to situations in which images may not appear "attached" with the real objects they are supposed to correspond to, appearing unstable, jittering, or swimming around [41]. These issues do not normally occur with OST devices which, however, require the human eye to adjust the focus between different planes, the screen and the real environment (effort that is not needed with VST). Given the above-mentioned comparison, Rolland et al. [42] conclude that "*the fundamental trade-off, then, is whether the additional features afforded by the more invasive approach justify the loss of the unobstructed real-world view*".

There are essentially three classes to categorise the display tools in mixed reality: head-mounted display (HMD), handheld display (HHD), and spatial displays. As already mentioned above, when designing a mixed reality application, one of the principles to bear in mind is reducing as much as possible the physical effort to be made when carrying out the task. This also anticipates the two concepts to pay particular attention to in this phase: comfort and immersion (e.g. field of view) [38]. Both of them are in fact related to the MR device that is used by the user and can therefore be different for each category of display the MR application can support.

As its name suggests, a head-mounted display is worn on the head (or as part of a helmet) and shows real and virtual world images over the user's view. HMDs can be either OST or VST, and can have a monocular or binocular display optic. As already explained above, video-see-through displays expect the user to wear

two cameras on their head and both captures are combined to digitally recreate the real environment with integrated computer-generated images. This can be quite demanding compared to OST displays which allow the view of the physical world to pass through a half-silver mirror technology, and graphically overlay information that is then reflected in the user's eyes. In Figure 2.3 the reader can find some examples of head-mounted displays. Oculus Rift (2.3a) is used for virtual reality, HTC Vive is used for both virtual and augmented reality (2.3b), whereas Magic Leap is a HMD used for augmented reality purposes (2.3c).

(a) Oculus Rift[1]

(b) HTC Vive[2]

(c) Magic Leap[3]

Figure 2.3: Examples of head-mounted displays

The term "handheld" is self-explanatory, too. A HH display consists of small device that the user can hold in their hands. This class of displays uses video-see-through for the superposition of computer-generated images to the real world, and sensors (e.g. digital compasses and GPS units) for their six-degree of freedom tracking. HH devices represent a superior alternative for AR, "*especially for untrained users in unconstrained and non-supervised environments*" [43]. There are essentially three categories of handheld displays for AR settings: mobile phones, Personal Digital Assistants (PDAs) and Tablet PCs. The trade-offs here is about their size, weight, computing power and cost [44]. Nowadays handheld displays represent the most

[1] `https://www.oculus.com/rift-s`
[2] `https://www.vive.com/eu`
[3] `https://www.magicleap.com/en-us`

accessible type of AR technology for the general public, and they are more robust than HMDs. For these reasons, users are more comfortable operating with them. One last note on handheld displays is that, since at least one of the user's hands are occupied with the device, the designer should consider that interface metaphors developed for SAR displays and HMD based systems may not be appropriate for this class of displays [45].

Finally, with Spatial Augmented Reality (SAR) users do not need to wear or carry any kind of device. In fact, computer-generated information is displayed directly onto physical objects, by means of technologies such as video-projectors, optical elements, holograms, radio frequency tags, and other tracking technologies, most of which integrated into the environment [41]. This enables collaboration between users in wide areas. The environment can be augmented differently according to the approach to SAR adopted. We can distinguish essentially three of them: video-see-through, optical-see-through and direct augmentation. Spatial VST displays are screen based, whereas spatial OST displays generate images that are aligned within the physical environment. Because of their spatially aligned optics and display technology, spatial OST displays do not support mobile applications. Lastly, projector-based spatial displays take advantage of front-projection to throw computer-generated images directly into physical objects' surfaces, instead of displaying them on an image plane in the visual field of the viewer [46]. An example of spatial augmented reality is the SunCAVE in San Diego that the reader can see in Figure 2.4.

Figure 2.4: SunCAVE in San Diego[1]

[1] https://ucsdnews.ucsd.edu/pressrelease/from-star-to-sun-the-qualcomm-institutes-cave-expands

If on one hand, according to Carmignani et al. [41], we identify these three major types of displays used in AR (head mounted displays, handheld displays and spatial displays), Peddie [47] states that within dedicated visual see-through augmented reality systems there are seven classes of displays. These classes are more specific than the categories reported above and they further subdivide the augmented reality displays developed so far. The user should also bear in mind that these classifications are not strict and inflexible, and a device can be categorised under one or multiple classes. Thus, some overlapping is present. We can identify the following classes:

- Contact lens
- Helmet
- Head-up Display (HUD)
- Headset (Smart-glasses)
  - Integrated (Indoors and Outdoors)
  - Add-on display and system for conventual, sun, or safety glasses (Indoors and Outdoors)
- Projectors (other than HUD)
- Specialised and other (e.g., health monitors, weapons, etc.)

The idea of contact lenses for augmented reality is that "*by building the infrastructure directly on top of the eye, the eye is allowed to move or rotate freely without the need of exit pupil expansion nor eye tracking*" [48]. However, they are still under development and nothing is available on the market yet.

Helmets are devices that cover the user's ears, total head and most of the face. Head-mounted displays can be used as part of a helmet.

Head-up Displays (HUDs) superimpose information to physical surfaces or objects, without the need for the user to wear particular devices or perform specific actions. HUDs are mostly used in cars (see Figure 2.5a), where they are positioned in the dashboard and connected either to the automobile's on-board diagnostics system or to the user's smartphone. By projecting the vehicle's information such as speed, water temperature and battery voltage, to the inside surface of the windshield, the user can keep their head up looking at the road, having complete awareness of the car's data [47]. Another application is the aviation context, as shown in Figure 2.5b.

(a) HUD in cars (b) HUD in aviation

Figure 2.5: Examples of head-up displays[1]

Peddy [47] defined *integrated smart glasses* those that include and integrate lenses and other elements (such as a microphone, camera, or earphones), which constitute a full-fledged augmented reality headset (see Figures 2.6b and 2.6c). Smart glasses can be more suitable for indoor or outdoor environments, depending on the level of brightness of the display that determine the possibility to overcome the ambient light from outside. Smart glasses are worn in the same way as normal glasses and, as any type of augmented reality interface, they overlay extra computer-generated information to the user's real-world scenes and project it to the lenses or the user's eye sight. On the other hand, add-on augmented reality display devices, such as Varia Vision by Garmin in Figure 2.6a, can be attached to sunglasses or prescription glasses, but the basic idea is the same as the one described above. This kind of device is usually limited to monocular presentation and suppliers seem to favour the right eye, possibly influenced by Google Glass.

Finally, examples of specialised displays are glasses with limited or no information display capability that are commonly used for fitness tracking and health monitoring, and special weapons used by the military.

2.2 Internet of Things

The objects in our surroundings are more and more equipped with tiny sensors and processors that allow the transmission and the reception of data. This ability to autonomously obtain and apply knowledge is the reason why we call them *smart* objects [49]. Therefore a smart environment is capable of exchanging knowledge to adapt according to its inhabitants' needs. Likewise, we refer to smart homes, smart cities and smart transportation, when the aim is to automate them

[1] `https://magic-holo.com/en/all-about-head-up-displays-hud`

(a) Varia Vision by Garmin[1]

(b) Raptor by Everysight[2]

(c) Microsoft HoloLens 2[3]

Figure 2.6: Examples of smart glasses

and manage everything via the Internet [50].

Kevin Ashton was the first who, in 1999, coined the term "Internet of Things" (IoT) to describe this concept of connecting smart objects with the Internet [49]. In his vision, computers would use technologies such as sensors and Radio Frequency Identification (RFID) to gather data without human help and render it into useful information. In other words, they would enable computers to observe, identify and understand the world [51]. Since then, the notion itself has evolved and new technologies have been introduced.

2.2.1 IoT Components

Considering that "Things" can be literally anything from appliances and buildings to people and trees, a broader definition has been provided by some companies, such as Cisco [52]. They refer to IoT as the Internet of Everything (IoE) with four key components:

- people, because the main idea is always to connect people together, who take action based on notifications from connected applications;

[1] `https://buy.garmin.com/en-US/US/p/530536`
[2] `https://everysight.com`
[3] `https://www.microsoft.com/en-us/hololens`

- data, which represent the information exchanged between devices;
- processes, that guarantee interoperability; and
- Things, physical devices connected to the Internet and able to send and retrieve information intelligently.

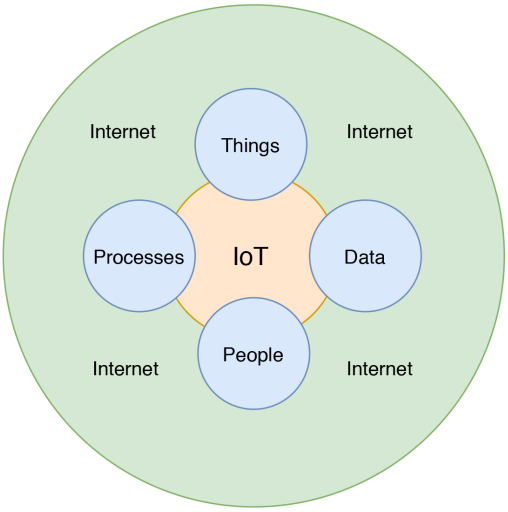


Figure 2.7: Graphical representation of IoT components.

All in all, with this in mind, a more complete definition of IoT can be drawn: *"IoT is the network of "Things", with device identification, embedded intelligence, and sensing and acting capabilities, connecting people and Things over the Internet"* [52].

This network consists of physical elements empowered by (1) sensors, to collect the information, (2) identifiers, that identify the source of data, (3) software, used to analyse the data, and (4) Internet connectivity, in order to communicate information and notify a user or an object.

The two main requirements for "Things" in IoT are sensing and addressing. The former is essential to identify and collect key parameters for analysis, but it lacks the ability to control or repair the Things. The latter, on the other hand, is necessary to uniquely identify Things over the Internet. In order to be able to control things through the Internet, actuators have been introduced. Thus, the key requirements for "Things" in IoT consist of sensing, actuating, and unique identification. We can now merge these requirements with the IoT definition provided above, and obtain a high level graphic representation of IoT (Figure 2.8).

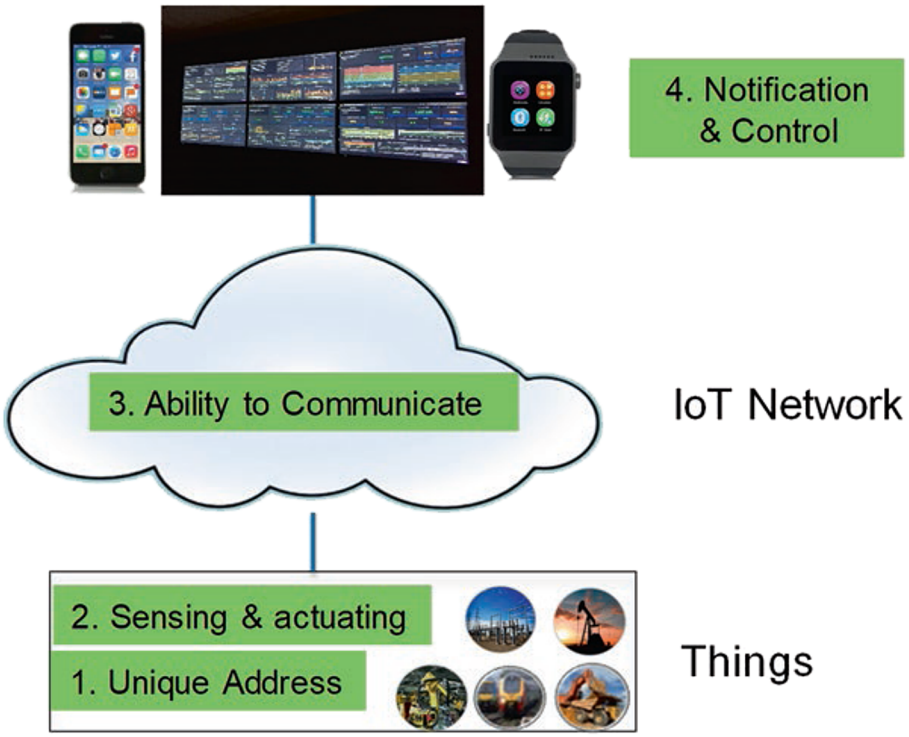


Figure 2.8: IoT requirements [52]

2.2.2 Mediums of Deployment

For what concerns the mediums of deployment, many different solutions are available. However, even though initially the IoT was implemented in wired communication (because the Internet itself used only wired connections), that does not seem to be a good idea nowadays, both in terms of cost and mobility issues, as reported in Suresh et al.'s work [9]. The wireless medium is preferred instead, and five main mediums of deployment are identified.

i Radio Frequency Identification (RFID), a technology that transmits data through radio frequencies. Thanks to its small size, it can be included in any object (e.g. car keychains and mobile cardiac telemetry systems) and can transmit data remarkably fast (less than 100 milliseconds). A concrete example is shown in Figure 2.9a.

ii IEEE 802.11, also known as Wi-Fi, is a wireless medium used to send and receive data, signals, commands and any other information emitted by an electronic device. Nowadays, Wi-Fi networks are used in most public and private areas, which represents a huge advantage for the IoT, given that its goal is to connect smart objects that might be distant from each other. Additionally,

as mentioned above, the number of Things increases every day, and the fact that Wi-Fi allows adding new devices without installing new ones, makes it a great means of deployment in IoT scenarios.

iii Linear or 1D barcodes, and 2D barcodes. They consist of a series of lines and spaces of variable width to encode data, and they are read horizontally. Two-dimensional barcodes, on the other hand, are read in two dimensions because the data is encoded based on both the vertical and horizontal arrangement of the pattern. They can contain any types of binary data, such as images, audio and website addresses [53]. Barcodes are probably the most economical, accurate and reliable medium for electronic data interchange, and they can be deployed with every thing easily. Using 2D barcodes such as Quick Response (QR) codes allows the consumer to connect to the IoT with the simple scan of a smartphone or tablet. Examples of a linear barcode and a QR code are depicted in Figures 2.2b and 2.2c respectively.

iv ZigBee IEEE 802.15.4. Zigbee is a wireless network standard for small, low power radios based on the IEEE 802.15.4 Standard for Personal Area Network (WPAN), which has a typically limited range of transmission (10 to 100 meters) [54]. Since it uses very low data rate, has a long battery life, and is cheaper than Wi-Fi and Bluetooth, Zigbee is widely useful in monitoring and controlling applications [55]. Thus, it is considerably adopted in IoT systems. For a concrete example, see Figure 2.9b.

v Bluetooth is an open wireless technology for Personal Area Networks. Introduced in 1998, Bluetooth has faced substantial advances in terms of technology. Nowadays, Bluetooth finds its application in multiple environments and domains ranging from printers and smartphones to wearable devices such as smart watches, headsets and shoes.

(a) Examples of keys using RFID [56]

(b) Zigbee transmitter and receiver [57]

Figure 2.9: Some graphical representations of IoT mediums of deployment

2.2.3 Home Automation

In this section we briefly analyse the current state of the home automation domain, providing some examples of applications available for the public. As already anticipated, the number of intelligent electronic objects exploited by humans has largely increased in the last years, up to a point that even appliances such as the toaster and the fridge can emit and receive data through the Internet. Their computing capabilities are made possible by the electronic components embedded into these physical devices. Even though these smart devices can be found anywhere (e.g. industry and urban environments), with all probability, private smart homes represent the most common and investigated scenario where to control the IoT devices by means of one single application. This application is most likely installed on tablets or smartphones, which can be carried anywhere and used at any moment. Users provide some commands to the controlling devices that in turn, thanks to the hardware elements listed above (usually Bluetooth and Wi-Fi), enable the communication with the smart appliances.

The most common home automation systems in commerce are *Amazon Echo*[1] and *Google Nest Hub*[2]. The former uses Amazon's proprietary Alexa voice-powered technology. It comprises speakers and microphones around the room to easily capture the user's voice commands and answer them back. Everything that can be retrieved from the web or that is connected to Amazon Echo can be requested or controlled. The Echo can work as a standalone device, connecting to a cloud-based service to make calls or send text messages, provide music, request weather and traffic information, set up alerts and times, or simply ask questions. On the other hand, Google Nest Hub uses Google Assistant to carry out tasks using voice commands, but works mainly as a visual interface. It can display local weather information, events, notifications, daily schedules, as well as play music and video from YouTube and other services such as Netflix.

Even though they represent a big support to users in their homes, we could identify two main limitations of these home automation systems. The first is the lack of a proper interface that allows users to have a visual understanding and control of the smart objects in the surroundings. In particular, a way to quickly check their position in the space and to perform some actions directly when they appear in front of the camera. The second limitation concerns the impossibility to add logic to the domotic system. For instance, when the user wants to turn on the TV, turn off the light and open the couch leg extension, they are currently obliged to provide three separate voice commands, such as "Hey Google, (or Alexa) turn on the TV", then "turn off the light" and finally "open the couch leg extension". Whereas it

[1] `https://www.amazon.com/smart-home-devices/b?ie=UTF8&node=9818047011`
[2] `https://store.google.com/category/connected_home`

could come in handy if the system automatically performs these actions when the user sits on the couch (thus, when the sensors detects the user laying or sitting on it). These two limitations are addressed and tackled in the application we propose in Chapter 6.

Finally, another important contribution for the home automation field is provided by *Home Assistant*[1], which is an open source project powered by a worldwide community of tinkerers and do-it-yourself enthusiasts. This systems is oriented to the control and the privacy of users' homes, and can be run on a Raspberry Pi or a local server. Figure 2.10 shows a demonstration of how Home Assistant would look like after being installed and set up. Here we can notice that this automation system has a rich and elaborated interface that enables end users to control and personalise their homes. However, the target audience consists of people who are very familiar and skilled at programming. In fact, novice users might not be able to set up a Raspberry Pi by themselves and might find the interface provided quite overwhelming.

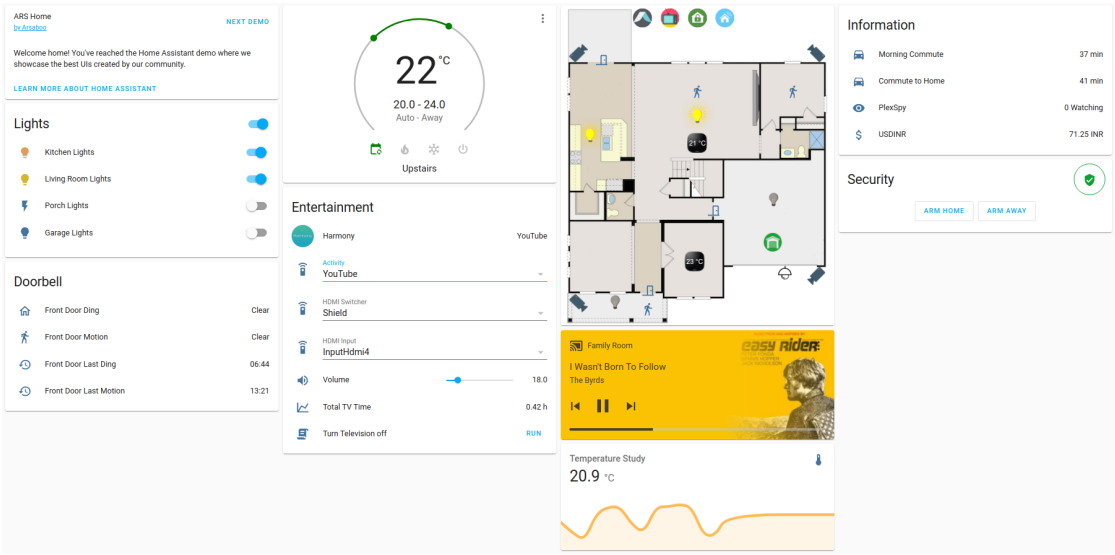


Figure 2.10: Home Assistant demo

[1] `https://www.home-assistant.io`

3 Related Work

During the last decades, extensive work has been done in the domains of augmented reality, human-computer interaction and Internet of Things. In this section we present some of the major contributions that has led to the outcome of this thesis.

One of the first implementations combining different input techniques to manipulate the graphical interface is Bolt's Put-that-there, presented in 1980. The idea was that, by means of speech recognition and position sensing in space, voice and gesture inputs could be used to draw and move objects projected in a large screen. The user could also exploit joysticks or touch controls to zoom in and navigate to where specific caches of information reside. In this way, the two spatial orders (virtual graphical space and the user's immediate real space) could converge to become one continuous interactive space [58].

In 1997, Jun Rekimoto proposed a new field of user interfaces called multi-computer direct manipulation where the user could transfer data between different computers or within the same computer through the usage of a pen. Each pen is assigned a unique ID, used to recognise which pen holds an object. When the user touches the display with the pen, the pen manager asks the first computer to transfer the data to the second one. The proposed implementation takes the name of *Pick-and-Drop*, and allows a user to pick up an object on a display and drop it on another display as if they were manipulating a physical object [59].

Related to this work, two years later, Rekimoto presented augmented surfaces, a spatially continuous work space for hybrid computing environments. The idea was that by "*using an interaction technique called hyperdragging, users can transfer information from one computer to another, by only knowing the physical relationship between them*" [60]. This can been seen as dragging icons from one screen to another in a single computer supporting multiple screens. But what is particularly interesting to the matter of this thesis is that the system supported links between digital information and physical objects. In fact, by attaching visual markers to the objects, the system could read and recognise them in order to display the associated digital data.

As a solution to the difficulty of transferring information between different devices, Marquardt et al. propose the gradual engagement design pattern, capable of con-

necting devices and exchange capabilities as a function of inter-device proximity [61]. In other words, when people move and orient their personal devices (e.g. smartphones and tablets) towards other surrounding devices, the interface increases continuously in three stages: (1) awareness of device presence and connectivity, (2) reveal of exchangeable content, and (3) interaction methods for transferring content between devices tuned to particular distances and device capabilities. This mitigates the problem of knowing which devices can communicate with each other, what information they contain is exchangeable, and how to exchange information in a controlled way.

In the same context of cross-device interactions, but now with the addition of user gesture tracking, Rädle et al. propose HuddleLamp, "*a desk lamp with an integrated RGB-D camera that precisely tracks the movements and positions of mobile displays and hands on a table*" [10]. The camera detects and identifies displays such as smartphones and tablets situated on the table, and tracks the users' hands to enable the interaction between the devices. By just opening a URL on the device and putting it on the table, the user can easily add it to the space and use it to transfer and receive data to and from the others.

Espada et al. [62] proposed a system that allows web applications to access the device communication hardware elements (e.g. Bluetooth and Wi-Fi), enabling the communication with physical devices. As explained in their article "*these communication elements use a group of actions to interact with other devices or networks, actions such as scanning the network, connecting to a device, sending data, receiving data, and so on*". This implies that web applications need to use a specific language to specify the action that has to be executed. For this reason, the application proposed by Espada et al. eliminates any platform dependency and can work on a wider range of mobile platforms with lower development and maintenance costs. Furthermore, the language adopted does not require the use of specific web technologies, libraries, plugins or other components.

In 2017, Roels et al. presented a framework for cross-device information exploration and exchange called INFEX [63]. By communicating via various protocols and in an extensible manner, the INFEX framework allows the user to read content from one device and copy it to another device even though they use different communication methods. Hence, the framework acts as a mediator to handle device detection and information exchange.

Yang and Nakajima [64] proposed a cross-platform middleware infrastructure that "*supports remote monitoring and control functionalities based on remote streaming for networked intelligent devices such as smartphones, computers and smart watches, and home appliances such as smart refrigerators, smart air-conditioners*

and smart TVs". The authors started from the issue that screens in our daily lives are mostly isolated from each other. To control them, users have to directly exploit input widgets connected to the device, and to watch them, users necessarily need to stand in front of the output screens. Given that hanging around these devices and home appliances is still laborious and irritating, Yang and Nakajima's goal is to propose a platform whose connection of multiple smart objects over the display screen would considerably save time and efforts, constructing a convenient IoT environment indoors and outdoors. In other words, the middleware overcomes the constraints of "one display per device" and "one operating system per device" and is particularly thought to find its application in a smart home context.

Some of the recent research focused on browsing the Web of Things in mobile augmented reality (MAR). Zachariah and Dutta propose a model capable of allowing users to identify new devices and to easily access the ones used regularly in the physical environment [65]. More precisely, through this model, users can open a "browser" on their smartphone or tablet, and use the camera to identify objects and discover their associated web interfaces in physical space. Once open, the interface uses a JavaScript Bluetooth API or network protocol to interact with the device.

Another important architecture helping the interaction with the Internet of Things using augmented reality is smAR^{2}t (Smart Model At Runtime Augmented Reality Tracking). As Bezerra and de Souza explain in their paper [66], smAR^{2}t allows users to point their smartphones directly at Things, adapting them to the information acquired in the surrounding environment, by simply interacting with 3D augmentations. The system exploits the World Wide Web Consortium (W3C) Web of Things standard, which suggests the use of a formal model called Thing Description (TD) [67] that allows to express in detail how to interact with a Web Thing (WT). Furthermore, these TDs can constitute guides to create and manipulate models in different contexts.

In the context of adding conditions and statements in an IoT scenario, *IFTTT*[1] has been developed. IFTTT, which stands for "If This Than That", helps people connect services together through the usage of Applets. Examples of services are Amazon Alexa, Twitter, Evernote, Dropbox and Uber. In this way, you can get Alexa talking to Google applications, change the hue of the bulb when Uber arrives or send Evernote to Slack. All in all, IFTTT brings services together and adds logic to them in form of conditions and dependencies.

Guided by the purpose of reducing energy consumption at home and improving the safety and security of home equipment, Vishwakarma et al. [68] developed an IoT-based home automation system that uses both voice and web-based services to

[1] `https://ifttt.com`

control all smart appliances. The former are provided by the support of an existing home assistant such as Google Assistant. The latter uses an IoT based application that uses (1) *NodeMcu*[1], which is an open-source firmware and development kit that helps prototype the IOT product within a few Lua script lines; (2) IFTTT, used to make easier for the device to work based on the mobile application using the conditional statements; (3) *Adafruit*[2], responsible for sending and receiving the feed data; (4) *Arduino*[3], used to compile the code.

Finally, a work that helped us design our interface conceptually is the context modelling toolkit (CMT) [69]. In their work, Trullemans et al. point out the importance of having a balance between the level of control and automation. For this reason, the context-aware system they built is particularly helpful in a smart home scenario, where by taking into consideration the individuals' behaviours, the system adapts and controls the smart objects accordingly. Thus, users can define context rules in the form of "IF this THEN that" or "IF situation THEN action" and therefore instruct the system into taking a certain action only in case specified events or situations happen. The context modelling toolkit is not only accessible to programmers who are experienced with this logic and low-level programming, but also to expert users, capable of creating template for the rules, and those end users who only need to fill the templates with data instances without any programming knowledge whatsoever. In the same way, our interface aims to assist any type of user to access and control all the smart objects in the environment in the most intuitive and straightforward way possible.

[1] `https://www.nodemcu.com/index_en.html`
[2] `https://www.adafruit.com`
[3] `https://www.arduino.cc`

4 Interaction Techniques

In this chapter, we explore the different interaction techniques available for each augmented reality category of display. As stated above, each type of device might be more suitable for a certain kind of interaction technique rather than another. This is due to various factors such as the size of the display, the field of view (FOV), and the engagement of one or both hands. Furthermore, an important aspect to take into consideration is that the interaction should be performed in a natural way, which also depends on the display used for the task. In other words, if the user interacts with the device in an unnatural way, it will appear awkward to use it in public places. For instance, speech recognition is an interaction technique that, besides suffering from noisy conditions, which is already a factor of unsuitability, is also not socially acceptable, therefore often not applicable [70].

As explained in Section 2.1.4, in the mixed reality domain, we count three main categories of displays: head-mounted displays (HMD), handheld displays (HHD), and spatial displays. However, further classes can be identified as stated by Peddie [47] (e.g. smart glasses and head-up displays). They all have different characteristics and thus require the use of particular parts of the user's body or of the device itself.

4.1 Head-mounted Displays

Recently, Whitlock et al. conducted a study [71] in which they explored AR interactions at a distance, in particular in the category of head-mounted displays. Three interaction modalities – multimodal headtracking and voice, embodied headtracking and freehand gesture, and pointer-based handheld remote – have been considered to test actions such as selection, rotation and translation, on objects situated between 8 and 16 feet away from the user's position. Three main conclusions emerged from the experiment:

- Voice interaction was the least efficient and least preferred modality,
- Embodied gestures were perceived as the most usable and strongly preferred,
- Both handheld remotes and embodied gestures enabled fast and accurate interactions.

More precisely, even though objectively no efficiency differences was registered between embodied gestures and handheld remotes, participants preferred the former to the latter, considering them more usable and effective. This is also due to the fact that locating and maintaining a cursor was much easier with embodied gestures, where the cursor is located in the centre of the field of view.

Despite the original hypothesis that "*multimodal voice interactions would be robust to distance, while embodied freehand gestures and handheld remotes will be slower and less accurate as distance increases*", Whitlock et al. found out that, with increased distance, the performance improved for voice interactions and degraded for handheld remotes and freehand gestures when in combination with other factors.

Finally, for what concerns voice-based interaction, users performed the actions worse (both objectively and subjectively) with this modality, whereas embodied freehand gestures were perceived more usable and intuitive than the handheld remote.

Gesture-based Interaction Song et al. [72] identify two approaches to perform mid-air interactions: the first requires a handheld controller device, such as the Nintendo Wiimote[1], and the second is a controller-free approach where the user can manipulate the 3D objects directly with their bare hands. The former allows to form high-level gestures to support the interaction by means of button clicks and accelerometer-based motion sensing. The latter uses an image and/or a depth sensor to keep track and analyse the user's hands. Unfortunately, the user might encounter the occlusion problem where some content in the environment is hidden by their hands or the remote controller. To tackle this problem, the bar metaphor was introduced, enabling the user to manipulate single or multiple objects along the line described by both hands [73]. Song et al. [72] call it the *handle bar* metaphor and it consists of virtually piercing a virtual bar through the selected 3D object. In this way, the user can translate and rotate the object exploiting this bar with both hands in a more intuitive and visible manner, as shown in Figure 4.1.

Gaze-based Interaction To date, eye-tracking technology is still mostly used in research but it is not available in products commercialised in the market [74]. What is more currently used is head-gaze, which is exploited to aim at the UI elements. Practically, the user moves their head towards an element in the scene in order to make the virtual cursor in the display overlap with it. However, Blattgerste et al. [74] proved that a solution that "*outperforms head-gaze in terms of speed, task load, required head movement and user preference*" is eye-tracking, which also

[1] `https://www.nintendo.be/nl/Wii/Accessoires/Accessoires-Wii-Nintendo-626430.html`

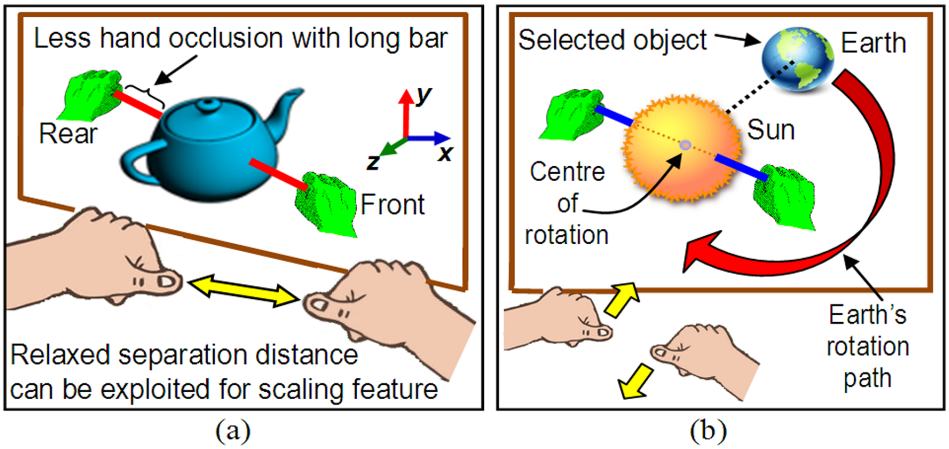


Figure 4.1: Handle bar metaphor. (a) Scaling is done by varying the distance between the two hands. (b) Rotation of a selected object (Earth) performed about the centre of the handle bar placed inside another object (Sun) [72]

appears to be more natural and less exhausting than head-gaze. By using eye-gaze, the trigger event can be either the click of a controller or a dwelling threshold.

Compared to other interaction techniques, for a simple operation such as selection, eye gaze is slower than hand-based pointing, and less accurate (even though a bit faster) than controller-based pointing. Furthermore, when used in a virtual environment, mouse selection is outperformed by eye-gaze in terms of task-completion times. Finally, when users cannot use their hands for the interaction because they are busy in completing a task (e.g. maintenance tasks), eye-gaze has proven to be the potential best solution, and it is even more accurate than head-gaze both in VR and AR settings in all FOV conditions. What needs to be done is further research to improve it and integrate it to the current devices.

Elmadjian et al. [75] conducted a study to explore eye tracking as a means to interact with 3D objects. They took advantage of two methods: a geometric one and an appearance-based one. The former provides gaze vectors through a rigid transformation of the eyeballs, assuming many simplifications such as the sphericity of the eyeball that causes the shift of the gaze estimates by a certain degree from the user perspective. The latter consists on tracking specific features on the eye image, such as the projected pupil center. Here a nonlinear regression function with the aim of mapping tracked features to targets on the scene camera is needed. This procedure appears to be more accurate than the geometric one, since it also takes into consideration noise and sensing biases. The most evident result of Elmadjian et al.'s study is the proof that gaze estimation is still an open challenge which requires

particular attention especially in terms of depth. Furthermore, they discovered that even with the simplest method (geometric), a single-plane calibration is not enough to provide accurate gaze estimation in the scene volume [75]. Gaze interaction is indeed a field that demands further research and investigation in order to provide a more accurate result.

4.2 Smart Glasses

As shown in the examples in Figure 2.6, smart glasses have a limited display size and viewing angle, which corresponds to a narrow field of view (FOV). This makes it difficult to visualise all the information on the AR display and limits the user-centric task assistance [11]. Moreover, using full hand gestures as a means to interact with the interface might be quite challenging.

Kim et al. [11] recognise two main types of gesture-based interactions in wearable AR: hand gesture-based interaction and multi-touch interaction. The former allows the user wearing AR smart glasses to use both their hands to manipulate virtual objects. To enable this type of interaction, it is therefore necessary to include a depth sensor to the front of the glasses in order to consistently track and recognise the hands. The latter, on the other hand, requires the usage of a touchpad (e.g. smartphone or tablet) in combination with the smart glasses. It is now evident that bi-manual operations are no longer supported, since one of the user's hands is occupied.

What emerged from the experiment run by the authors, for all the tasks (except one) the hand gesture interaction method was preferred over the multi-touch method, because considered more natural and effective, and hence, faster. However, a downside was that participants found wearing another sensor for detecting hand gesture uncomfortable. For what concerns the second method, participants stated that carrying out the tasks with multi-touch interaction was more difficult and challenging. Moreover, the narrow FOV is indeed a limitation for the augmented information to be displayed, which sometimes fell out of the FOV of the screen.

Finally, the results showed that manipulating 2D virtual objects through selecting and dragging was more feasible with the multi-touch interaction, whereas the hand gesture interaction is more efficient for manipulating 3D models (translating and rotating are more efficient with this method). Therefore a hybrid user interface might be the best solution. In this way, the user can decide to opt for one technique rather than the other, according to the type of task they need to complete.

Given the characteristics and limitations of smart glasses, in their study [76], He and Yang reported that three main design principles should be applied:

- No cumbersome sensors on hand, meaning that even though wearing sensors on hands (or gloves) would make the tracking of hands easier, it would be an uncomfortable and unnatural technique. Thus, bare-hand is preferred.
- Keep few gestures, because having a large number of gestures to memorise would be frustrating and counterproductive for the user.
- Keep obvious status switching, which would avoid any kind of confusion to the user. It is important that they have the situation clear at any moment in time when using the device to interact.

4.3 Handheld Displays

Handheld displays comprise every computer device that can be held in one hand, such as smartphones and tablets. In comparison with other AR displays such as traditional desktops or tabletops, this category of augmented reality interfaces is more limited in terms of screen size, graphic support and activity time due to the battery duration [45, 77]. Additionally, the input mediums are different as well, since no mouse and keyboard are available. As a consequence, handheld displays require different interaction techniques. Even though handheld mobile devices represent the most popular output medium for augmented reality (especially smartphones which take part in everyone's daily life), research in this field is still lacking. However, in their exhaustive study, Goh et al. [78] were able to identify three main categories of 3D object manipulation in handheld mobile AR:

1. touch-based interaction, which involves all uses of on-screen touch inputs;
2. mid-air gesture-based interaction, made possible by the tracking of bare hands of finger gestures as inputs; and
3. device-based interaction, that includes techniques enabling the handheld mobile device's physical attributes to be tracked.

These interactions are graphically shown in Figure 4.2. Being significantly different one from the other, each type has its own limitations and strengths.

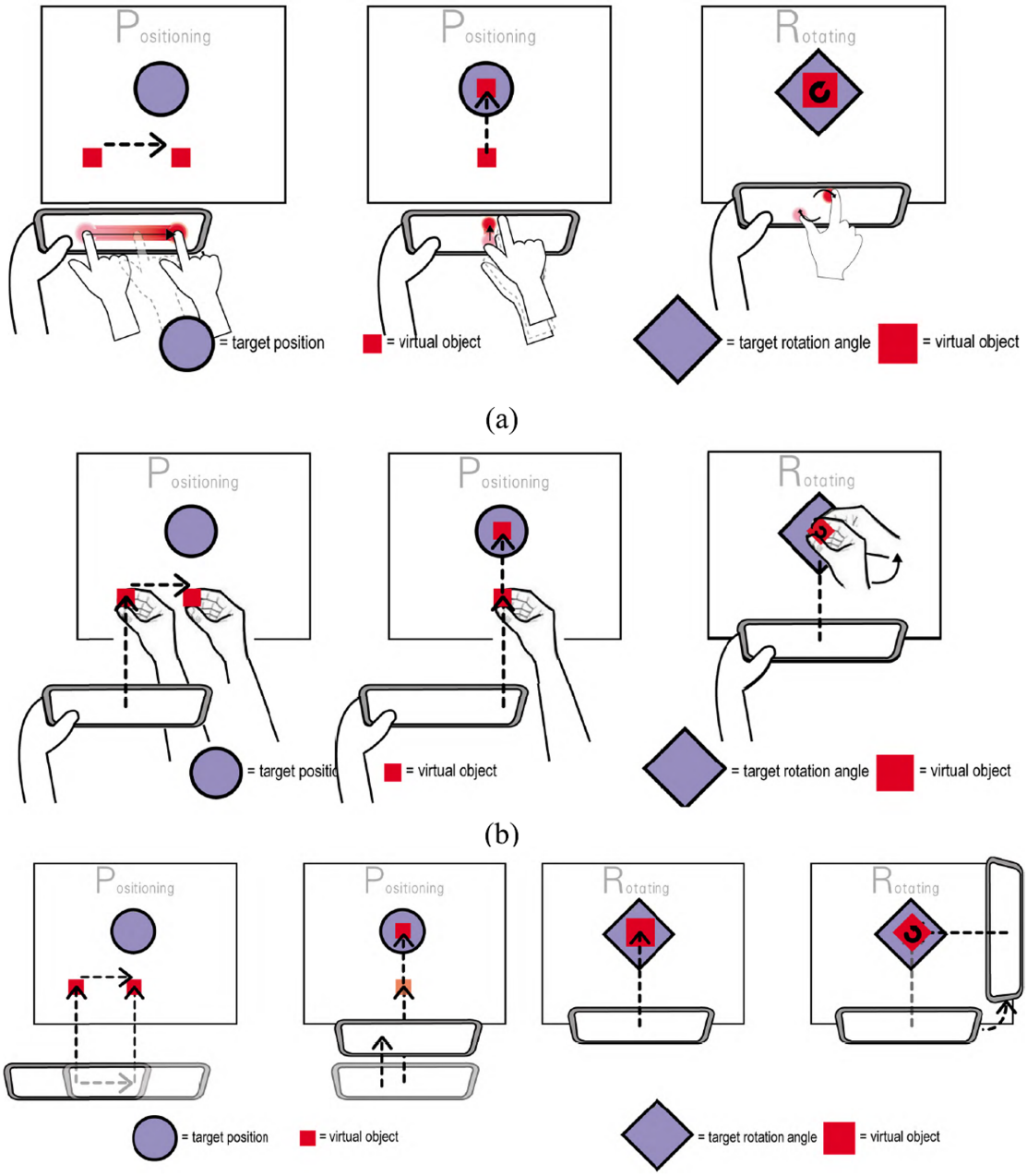


Figure 4.2: The three categories of 3D object manipulation in handheld mobile AR: (a) touch-based interaction, (b) mid-air gesture-based interaction, and (c) device-based interaction [78]

Touch-based Interaction In multi-touch interaction, the number of degrees of freedom (DOF) that can be simultaneously controlled on a device is high. In fact, given that each finger specifies a 2D position, using the whole hand theoretically allows the control of 10 DOF (5 fingers * 2D) [79]. However, this count represents only an upper bound because the actual number is lower (between 4 and 6). In fact, currently no multi-touch interaction uses the position of all the five fingers to control the object. The reason behind this is that "*complex gestures involving all fingers are often unstable, and the time it takes to perform them would be prohibitive for an interactive use*" [79]. Finally, for tasks that can be completed with global hand gestures such as translation, rotation, and scaling of an object, two or three fingers would be sufficient to do the job.

Furthermore, as reported in [80], given that only one hand is available for interactions in typical handheld AR devices, only simple touch gestures of one hand for interaction are suitable. Additionally, usability can suffer not only from using two-finger touch on such small displays for 3D object manipulation (and especially for rotation), but also because users occlude the object with their fingers. This issue in turn can cause the *fat finger* effect, which happens when the users' fingers cover part of the content on the touch screen [78].

To facilitate the touch-based interaction in handheld devices and to tackle some the problems above mentioned, the *freeze view* has been introduced. This technique allows the user to easily manipulate the AR scene without being affected by shaky views [81]. For instance, if the interface is used while walking and the user touches its screen while trying to maintain visual tracking, it is hard to keep it still and steady. However, thanks to the freeze view, users can freeze the current camera frame (hence, the real world view) and interact just with the virtual objects in the AR scene. In this way, they can move the device without updating the visual tracking. When they successfully complete the selected task on the scene, they can free the frozen view by releasing it and letting it update with the new real world data [82].

Mid-air Gesture-based Interaction As mentioned above, there are essentially two approaches to perform mid-air interactions: the first requires a handheld controller device, such as the Nintendo Wiimote, and the second is a controller-free approach where the user can manipulate the 3D objects directly with their bare hands [72]. Since here we are in a handheld display context in which one of the user's hands is occupied by holding the device, we focus on the second mid-air interaction approach. By means of a depth sensor, the user's hand gestures are continuously sensed and analysed via real-time processing techniques. This type of interaction represents a more natural and intuitive mechanism for the user. However, self-occlusion is still a problem that requires special attention. Moreover,

the computational tracking information that needs to be calculated on the depth axis, and the additional hardware that is necessary to obtain the spatial data could significantly exhaust the battery capacity of the device [78, 83].

Bai et al. [84] conducted a study with eighteen participants to compare touch-based and gesture-based interactions in handheld AR. What emerged was that users found the touch-based technique not intuitive because it did not correspond to the common interaction pattern in their real life. Hence, they felt in need for instructions and tips in order to manage to complete the task. Furthermore, as already proved in other studies, the reduced screen size represents a limitation that entails the fat finger phenomenon and the occlusion issue, aggravated by touching the interface erroneously and shaking problems. These drawbacks did not appear in the gesture-based interaction technique that users found more enjoyable and more intuitive, even though participants could feel physical stress by using both hands to hold the interface. Finally, what these two techniques appear to have in common is the difficulty in perceiving depth. Users, in fact, realised that the 3D objects were positioned in a misleading way after checking from another viewpoint. The reasons behind this issue are essentially two: visual occlusion and depth occlusion, which make it more difficult to understand the spatial relationship in the environment.

Another problem related to hand-gestures interaction is the the gorilla arm. Chun and Höllerer's experiment [85] showed how participants were affected by the phenomenon of "tiring shoulder" when carrying out tasks that required up to ten seconds. Holding the handheld mobile for such a long time might lead to frustration and discomfort. Combining hand gestures with other techniques, such as touch interaction, or avoid long tasks by splitting them into shorter, easier ones might solve the problem.

Finally, Nanjappan et al. [86] recently published a study in which they explain the reasons why tracking the hand and finger movements with the device's camera might not always be feasible. First of all, the tracking may not be very accurate. Then, the handheld display should be kept at a certain distance from the user's eye, in order to be comfortable for the sight and to actually see captured view. However, on the other hand, since the human arm is not very long, it could be very difficult to have enough space between the device itself and the hand performing the gestures.

Device-based Interaction We refer to device-based interaction when the user exploits the attributes of the device to control the interaction with the virtual objects. What emerges from Mossel et al.'s [80] and Marzo et al.'s [87] studies is that device-based interaction appears to be a natural and intuitive technique which allows to translate 3D objects faster than touch-based interaction. However,

large-range 3D rotation tasks are still hard to be completed. "*By mapping the position of the handheld mobile device's built-in camera with the 3D object registered on the AR marker, the user can use the handheld mobile device as a moving and rotating tool for manipulation*" [78]. A downside of using the device movement as the same input for both 3D translation and rotation is the position deviation, which leads to a slower and less accurate positioning when performing 3D object manipulation tasks. Despite this inconvenient, device-based techniques are more suitable than mid-air gestures to be applied in handheld mobile AR because the device movement in the depth axis (z-axis) is mapped with the 3D object (contrary to mid-air gesture-based techniques where depth data needs to be calculated). This means that when the user lifts the device, the 3D object selected is lifted up, too, reducing additional calculations on the depth axis and hence, avoiding the exhaustion of the battery capacity.

As a summary to the issues relative to touch-based (TBI), mid-air gesture-based (MBI) and device-based interaction (DBI) techniques for 3D object manipulation in handheld mobile AR, the reader can find Table 4.1 partially retrieved from [78].

Issues	Interaction Techniques		
	TBI	MBI	DBI
Occlusion	Yes	Yes	No
Fatigue phenomenon	Serious	Serious	Slight
Prior knowledge needed	Yes	No	No
Intuitiveness/naturality	Low	High	High
Low precision	No	Yes	No
Position mismatch	No	Yes	No
Orientation mismatch	No	Yes	No
Position and orientation deviations	Yes	Yes	Yes

Table 4.1: Issues comparison between the touch-based (TBI), mid-air gesture-based (MBI) and device-based interaction (DBI) techniques for 3D object manipulation in handheld mobile AR [78]

Speech-based Interaction Another possibility for the interaction between users and handheld devices is speech. In the same way as gaze-based interaction, voice enables the user to operate with the virtual objects without using their hands. However, even though speech interaction appears faster than other modalities since

humans use their voice naturally to communicate with people, this technique is less accurate than other means such as multi-touch. To enhance the usability and the performance of voice interaction, an excellent solution would be to use it in combination with another modality. In fact, as per Kondratova [88], "*multimodal interfaces allow speedier and more efficient communication with mobile devices, and accommodate different input modalities based on user preferences and the usage context*".

Moreover, in outdoor settings the number of challenges to face is higher, not only with HH displays, but with any AR device. Noise and distractions are the main problems a user might encounter in an open environment. The noise can be in the form of traffic, industry, animals, or wind speed, and so on. Whereas what is considered distraction are movements of people, animals, information overload, or forgetting system keywords [89].

4.4 Spatial Displays

In Spatial Augmented Reality (SAR), computer-generated images are integrated directly in the user's environment and not simply in their visual field. If this digital content is aligned on a flat display surface, it appears in 2D, otherwise, in case it floats above a planar or non-planar surface, it appears in 3D [90]. Roo and Hachet [91] identified the basic issues to address in SAR:

- *Geometry.* In order to align the virtual and real information, it is paramount to know the geometry of the surface. This is achievable by generating a 3D model of the environment, either manually or through a scanner.
- *Position.* It is also necessary to know where the geometry lies in relationship with the real scene. By taking advantage of sensors or tracking (with or without markers), we can acquire the position of the scene.
- *Material.* The observed colour of a surface or object can be affected by its material depending on the colour and reflectance where the pixel is projected.
- *Light condition.* The ambient light is another factor capable of affecting the final result and it competes with the projector luminosity.
- *Shadow casting.* Shadows are produced when a pixel path from the projector reaches an object. A way to ease this problem is to use more than one projector.
- *Dynamic environments.* Spatial environments can be dynamic, meaning that all the problems listed above can change over time. This can be tricky and

might lead to unpredictable results. A way to mitigate this issue could be to predict the change.

Spatial augmented reality brings several advantages among which the most important one is probably the fact that users do not need to wear any heavy equipment, meaning that the system is less cumbersome and more ergonomic. Since the interface is not limited to a mobile screen or a head-mounted display, but the content is displayed on the wide surfaces present in the real environment, the field-of-view is less limited. These two benefits suggest that SAR is the preferred option for multi-user scenarios [46, 92]. In fact, as Thomas et al. [93] state, projecting the information directly onto the physical objects enable the user to observe them from different points of view by physically walking around them. Furthermore, when picking up a 3D element from the scene, users can move, rotate and resize them however they wish, in order to analyse them from different faces.

Moreover, if on the one hand in most of human-machine systems, interaction is facilitated by means of accessories such as the mouse, keyboard, remote controllers or by using touch displays, on the other hand, spatial AR requires something different than fixed controls, since the users move around the room while interacting with screens and projectors [94]. In order to assure users to have freedom of mobility at any time and to enable them to interact with the system from any distance, touchless interaction seems to be the most suitable option.

With regard to the interaction techniques in the context of spatial augmented reality, Roo and Hachet [91] detected three main categories: (1) manipulation, (2) body and speech, and (3) pointing. Manipulation is used in Tangible User Interfaces (TUIs), where by applying human natural skills for interaction with physical objects users are able to give physicality to virtual information and operations. However, manipulation here is not limited to manipulate rigid bodies, but it is also extended to touching surfaces or objects, detected by means of sensors placed in the environment or in the objects themselves, or vision techniques. As one might imagine, a disadvantage of this technique is that when objects are close together, their shadows could inevitably create confusion. The second interaction technique suggests that the best options in a SAR setting are gaze, gestures and speech, whereas pointing is achievable by controlling a cursor, either using tools or fingers. The same conclusions were drawn by Elepfandt and Sünderhauf who identify three main touchless interaction means in the context of spatial augmented reality: gaze, speech and gesture [94].

Gaze-based Interaction Given that it is the only reliable predictor of the area of visual attention, gaze is often used for people with motor disabilities. It represents a good means to point at people or objects, and selection appears to be faster than

hand-eye coordination (e.g. when using a mouse with their hand, the user needs to look at the target and reach it with the cursor simultaneously). Furthermore, users do not need to learn something new, since the movement of the eye is a natural consequence of the direction the individual is looking in. Different events can trigger the selection of an object. Some examples are when a certain amount of time elapses without moving the eye, or when the user blinks or any other keystroke is detected. The dwell-time is the most common technique used in the gaze interaction. However, defining the most convenient duration of the fixation is not a trivial task. If the dwell-time is too short, the system detects commands everywhere the user looks at (this phenomenon is called *Midas Touch* and it is fully explained by Jacob in [95]), which might be annoying for the user who would keep activating functionalities unintentionally. However, on the other hand, if the dwell-time is too long, there would not be any advantages in terms of rapidity of using eye movements instead of other techniques such as the mouse. Additionally, it might be difficult for a user not to blink or not to look away for a relatively long time.

Speech-based Interaction If hands and eyes are busy, speech represents an excellent interaction technique that gives even more freedom to users compared to gaze and gestures. Speech is in fact completely independent from input devices and works in any situation and from any distance [94]. For all these reasons, speech can indeed enhance productivity because users are able to instruct the system while in motion or doing something else at the same time [96]. However, this technique has also some drawbacks. The most obvious one is the fact that users might not want to be heard by others, either because of privacy or for a disturbance policy of the environment in question. Furthermore, systems are usually not that clever to understand everything the user says. Thus, voice commands need to be learned and memorised in order to correctly use the application. Due to these disadvantages, most of the time speech is used in multimodal settings, rather than on its own.

Gesture-based Interaction In human-computer interaction, gestures are very often used in combination with other modalities and they convey different messages according to the country or the cultural context they are adopted in. Pointing appears to be the only natural gesture shared by everyone, whereas all the other gestures in human-machine interactions might be obvious for some people and unintuitive for others. Hence, learning all the gestures could be quite demanding, both to learn and to reproduce fluently. For this reason, Jetter et al. [97] differentiate two classes of interaction: symbolic gestures and manipulations. The former are close to the keyboard shortcuts of the common WIMP (Windows, Icons, Menus and Pointers devices) systems and they are not continuous. This means that the user at any moment can execute them and the system detects them providing no

explicit feedback. The latter are continuous between manipulation and completion. Some typical examples of manipulations are the dragging, resizing, and rotating of objects. Since manipulations are natural and gestures are not, Jetter et al. "*consider non-symbolic direct manipulation in a model-world as the key to truly natural interfaces*" [97].

Touch-based Interaction Finally, touch-based interaction is another option. Bimber and Raskar [46] illustrated some scenarios and prototypes where 3D objects can be manipulated by means of a mouse, a transparent touch screen mounted in front of the holographic plate (a touchpad), or a 6-DOF force-feedback device. Thus, the user is able to virtually touch and feel the hologram and all the virtual models integrated to it. Furthermore, in a tabletop SAR system (see Figure 4.3) different interaction techniques are brought together. In addition to mid-air gestures, the user(s) might hold a handheld device with an integrated push button for selection and a scroll wheel for 2D zooming, whereas physical objects are used for tangible interactions [93].

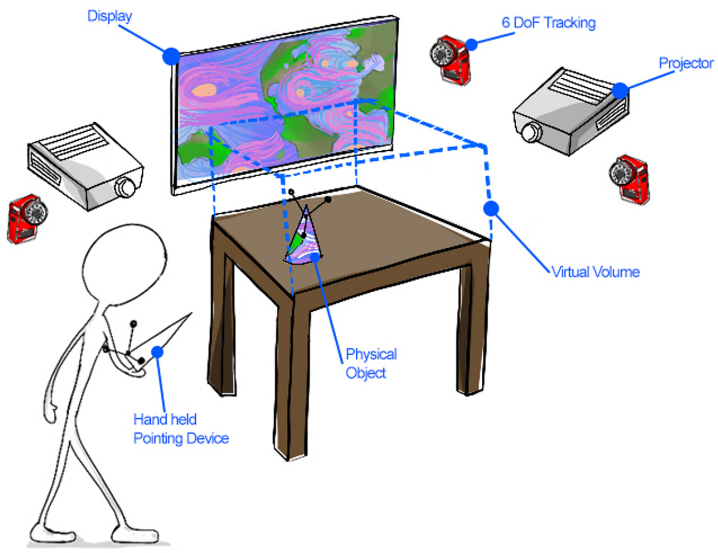


Figure 4.3: A common setting of Tabletop SAR [93]

The major characteristics of the interaction techniques introduced above are summarised in Table 4.2.

Table 4.2: Summary of speech, multi-touch, gestures, gaze and device-based interaction techniques for the different display categories

	HMDs / Glasses	HHD	SAR
Speech	Enhances productivity, but less efficient than other modalities. Privacy and disturbance issues. Preferred in multimodal settings. Specific set of commands needs to be learned.		
Multi-touch	By using a touchpad. Bi-manual operations not supported.	Only one hand available. Two or tree fingers are enough. Fat finger problem.	Via a mouse, a transparent touch screen mounted in front of the holographic plate (a touchpad), or a 6-DOF force-feedback device.
Mid-air gestures	Either by using a controller or depth sensors.	By tracking one hand with depth sensors. Occlusion problem. Battery capacity quick exhaustion.	Specific set of gestures needs to be learned.
Gaze	Eye-gaze outperforms head-gaze. Gaze depth is still a huge challenge.	Gaze is used in combination with touch on the device screen.	Defining dwell-time can be challenging.
Device-based	Not possible	By exploiting device attributes to interact with objects. More intuitive. Less computationally demanding.	Not possible

5 Interactions

What we aim to do in this chapter is offering the reader a description of the available interactions for each of the categories of displays discussed in the previous chapter, and a comparison of how a task is carried out with different devices. While Chapter 4 describes the existing means of interaction used in the various AR settings, we now dig in a bit deeper, focusing on how the user can practically perform actions on 3D objects. In his book, Craig [98] provides an accurate overview of augmented reality, its application and the possible interactions in this domain. Interactions are defined as any action taken by the users during the AR experience.

Common interaction schemes such as the WIMP paradigm may or may not be possible in AR settings, especially for specific types of interface. While interactions on handheld devices are similar to the desktop ones, mouse, trackpad or touchscreen to select an object might not be available on another AR display such as smart glasses. Therefore, new interaction techniques need to be discovered and designed.

Sherman and Craig [99] identify three main key interactions in virtual (and augmented) reality: manipulation, navigation, and communication. Manipulation is the action through which the user can modify the objects present in the virtual world. Navigation allows the user to explore and walk through the virtual world. Sometimes navigation is the only available interaction. Communication can be performed either with other users or with some agents in the virtual world. "*Each of the user's interactions involves a primary task (e.g. grabbing, pushing, or talking), that is often made up of subtasks such as selecting (items, directions, correspondents), and then a means of activation*" [99]. We are going to describe each of these key interactions, focusing more on the manipulation one, given the scope of this thesis. Moreover, we will analyse the differences between performing interactions with one interaction category rather than the other.

5.1 Manipulation

In the virtual context, most manipulations include two phases: (1) selecting the object in order to instruct the system to know what needs to be modified,

and (2) performing an action on that object [99, 100]. It is also possible to execute these actions simultaneously. In an early study about virtual reality environments, Mine [101] identified three major categories of manipulation, to which Sherman and Craig [99] have added a fourth one:

1. *Direct user control.* This includes the use of hand tracking, gesture recognition, pointing, gaze direction, and so on, to mimic real-world interaction.
2. *Physical control.* This includes buttons, sliders, dials, joysticks, steering wheels, trackballs or any device the user can physically touch to manipulate the objects.
3. *Virtual control.* This includes any control we can implement, therefore any virtual version of a physical control that the user can virtually touch to manipulate the objects.
4. *Agent control.* This happens when participants give commands to some agent in the virtual world to carry out the action on their behalf.

Direct User Control As anticipated above, most of the time direct user interactions combine the object selection with the actual manipulation, and they use either gaze or gesture interactions. For instance, with gesture-based interaction, the user can perform the *grab with fist* technique to select an object by closing their hand into a fist (grab) and subsequently moving the object around by maintaining the fist and changing its position is space. In case of gaze interaction, the user could select an object by blinking, carry it around by exploiting gaze tracking, and finally release it by blinking again.

Physical Control Since the user exploits real-world devices to control the virtual world, the participants receive haptic feedback from pressing buttons and performing other actions. Buttons, switches with multiple position settings, slider and dial valuators, 2-degree of freedom (DOF) valuator controls like joysticks or trackballs, represent examples of controls integrated in tracked props. These controls can act independently or in combination with the prop's position. For instance, using the buttons on a controller to scroll forward and backward, or left and right, or to select a menu option (usually the middle button) constitute a setting where controls act independently. On the other hand, an example of integrating the prop's position would be to point the prop at an object and press a button to select it.

Nowadays, physical controls are seldom used because displays such as head-mounted devices or contact lenses are more and more widespread and, as should be clear by now, the ideal situation is to leave the user's hands free. In some cases, hand controllers are still used as a means of interaction in the virtual world. However, detecting hand gestures and, more generally, any action performed by the user will

replace any physical tracking device. Thus, virtual controls are without any doubt becoming more important for AR applications [100].

Virtual Control We talk about virtual controls when physical controls such as those listed above are emulated in virtual representations. In many cases, virtual controls are preferred as a means of interaction, even though at some point it is necessary for the user to physically do something to activate a virtual control. For instance, having virtual controls reduces the number of devices needed by the interface to operate. The most evident example is the use of the mouse in a desktop setting, which has 2D movements and can also be used for virtual controls, such as moving sliders, rotating dials or pressing buttons.

Furthermore, a virtual control can be hidden and only showed on-demand. The reader could think about a pop-up menu similar to the one in the desktop metaphor, which is specifically retrieved when the user presses a button. In such cases, what is common in today's interfaces is to "grey down" the appearance, meaning that the content is temporarily toned down to move the focus to the virtual control in foreground.

Agent Control With agent controls, "*the user is in direct communication with an "intelligent" agent who will then perform the requested action*" [99]. The communication can be performed via voice or gestures, where gestures can be simple body language commands or more formal languages such as naval semaphores or American Sign Language (ASL). On the other hand, tools such as Apple's Siri[1] or Microsoft's Cortana[2], now available on any mobile device and computers, could be exploited to instruct the system to do something for us in the virtual world.

5.1.1 Selection

Selection is probably the most basic and at the same time essential operation in any interactive system. We can distinguish three main categories: choosing a direction, choosing an item, and direct input of numeric or alphabetic values. Mine [101] states that each type of selection requires 1) a way to identify the object to be selected, and 2) a signal or a command to indicate the actual act of selection.

Direction Selection This method is used both for item selection and as a directional indicator for travel control. It is not necessary for items to be within reach: they can be placed faraway from the user. Sherman and Craig [99] distinguish seven different direction selections:

[1] `https://www.apple.com/siri`
[2] `https://www.microsoft.com/en-us/cortana`

- Pointer-directed, which uses hand posture or gesture to indicate a direction. For this purpose, direct tracking of the hand position or the use of controllers are exploited.
- Gaze-directed, which depends on the user's attention, using the direction the user is looking in. Eye-gaze is still under development in the mixed reality context, therefore gaze-based pointers take into consideration the direction the user's nose is pointing.
- Reticle-directed, which is the product of combining pointer-directed and gaze-directed selection styles. Despite being an easy and accurate technique, reticle selection requires the user to use their head and hand, which might result too demanding.
- Torso-directed, which is a natural option to select direction of travel. There is also the possibility to combine the torso position to the rest of the body in order to provide a full-body pose used for direction and activation gesture. However, due to the fact that it requires additional tracking hardware, it is very often discarded in AR/VR systems.
- Device-directed, which uses multiple valuators, such as a joystick (2-DOF valuators) or Spaceball (6-DOF valuators). By assuming that the valuator is always held in the hands oriented in a certain way with respect to the user's body, the user can manipulate the valuator to indicate a direction relative to the control location.
- Coordinate-directed, which uses azimuth and elevation values to specify a direction relative to some reference frame. This is accomplished by providing numerical coordinates. For instance, the user can use their voice to say "East" and the application would be oriented in that way.
- Landmark-directed, which performs the selection by taking into consideration a point of reference in the environment. For instance, with speech interaction, the user could say "move toward the red tea pot".

Select an Item Practically speaking, selecting an item (both in mixed-reality and real-world applications) is the process of picking a component from an enumerated list. These selectable components can be objects, locations or iconic representations of those items. In some cases, it is also possible to select multiple items. Consequently, the user should be able to deselect objects that are erroneously marked. Sherman and Craig [99] identify seven different ways of selecting items:

- *Contact-select.* The user's avatar makes contact with the object. The contact can automatically activate an action (from a specific body part or part of the avatar's body) or a separate trigger activation needs to be executed.
- *Point-to-select.* As explained above, the user selects an item by pointing directly at it, by means of a controller, their finger or their gaze.
- *3D-cursor-select.* A 3D cursor indicates the object selected. It is the 3D equivalent of selecting an item on a 2D surface using a mouse or trackball. A method often used is the *Go-Go* interaction technique, which uses "*a non-linear mapping between the controlled motion of the user's hand in the real world and the effected motion of the virtual hand in the immersive environment*" [102]. Basically, it uses the metaphor of interactively growing the user's arm, by mapping the position of a 6-DOF sensor onto the position of a virtual hand so the user can touch, grasp and manipulate virtual objects with their own hands.
- *Aperture-select.* The space between two fingers (thumb and forefinger) creates an aperture and objects that appear in the aperture are selected. The user can then manipulate the items using finger gestures, as already described above. This technique uses the image that is formed, which combines the real world, the virtual world, and the participant's fingers [100]. Aperture-select is also used to move and delete a virtual object. In fact, by closing the fingers together, the object becomes smaller and smaller until it disappears completely.
- *Menu-select.* A list of items is presented for selection, similar to the traditional desktop WIMP interface. Some solutions could be to use eye-gaze (but it is not very accurate), gestures or finger-contact gloves. What differentiates menu-select from contact-select is that it is not necessary for the choices to be nearby or visible to the user.
- *Select-in-miniworld.* Selecting items by contact-select in a miniature representation of the world, such as a map. While the direct methods, such as contact-select and point to select, provide the primary representations of the item, select in miniworld provides smaller replicas of objects.
- *Name-to-select.* A user selects the desired item by providing its name. This can be achieved with voice or by typing the name with a virtual keyboard. Since speech recognition is not an exact science and therefore might misunderstand what the user says, it is important for the system to provide some feedback to verify whether what has been processed is exactly what the participant intended. A downside could be that the user needs to know

the precise name of each object to be selected. However, this technique could be combined with the point-to-select method, which might reduce the overhead. For instance, instead of saying "under the green square" it would be possible to point at that location and say "there", which is the basic idea of Bolt's Put-That-There [58].

Alphanumeric Value Selection Sometimes, specific numeric or alphabetic information can significantly enhance the mixed-reality experience. A common technique to enter such data into the system is by using a pen-like handheld device. Voice and physical or virtual devices are some valid alternatives. Physical input devices should not occlude the real world and should always be visible to the user, whereas virtual input devices are the digital representations of the physical ones. Voice is an example of agent control which represents an effective means to feed the system with the alphanumeric values, by speaking out loud the entire phrase or number, or each letter or digit at a time.

5.1.2 Manipulation Operations

Above we have explored the various possibilities to select an object in a mixed-reality context. However, to actually manipulate the desired element, many other operations can be performed. In their work, Sherman and Craig [99] propose the following most common forms of manipulation: positioning and sizing objects, exerting force on a virtual object, modifying object attributes, modifying global attributes, altering state of virtual controls, and controlling travel.

Positioning and Sizing Objects This operation enables the user to change the location, orientation, and size of a virtual object. Any of the direct, physical, virtual, or agent controls described above can be used to alter the position and the size of the object. For instance, the *grab with fist* allows the user to change the object's position by "carrying" it around in their fist, and the object's size by opening and closing the hand. The virtual element's centroid and the position of the user's hand represent two alternatives of axes or points to be used when rotating or resizing an object. A common technique to resize an object is to use both hands to virtually stretching or shrinking it. It is easier and more straightforward to use both hands for this kind of operation. Thus, handheld devices might not be the preferred option. However, when only one hand is available and the device includes a touch screen, it has become a standard practice to use two fingers to modify the object's size. By using physical or virtual controls, the repositioning of an object is constrained to only one axis, which can constitute a limitation in some cases and a perfect suit for others (e.g. when the alignment of the object is important). Scaling the size of an object could also be accomplished by considering the distance between the finger aperture and the eyes.

Exerting Force on a Virtual Object Pushing, hitting, and supporting objects are kinds of interaction that require the exertion of force. Since there are different degrees or force that might be applied, the system must provide some feedback (e.g. visual, sound or haptic) to the user to notify them about their action and let them decide whether to increase or decrease the force. Not necessarily exerting force is used to move virtual objects, but it can be a means to hold an object in place or to cut, puncture or deform the object itself.

Modifying Object Attributes, which means changing the parameters used to control the rendering or the behaviours of an object. Mass, density, colour and light are examples of attributes that can be modified. Some of parameters are linked to others. Thus, changing a property might inevitably change another characteristic of the object (e.g. altering the density can affect the shape).

Modifying Global Attributes By changing global parameters, the rendering and/or simulation parameters for the virtual world are adjusted, leaving unaltered the attributes of a specific object. However, global attributes are related to all the objects in the environment. Hence, if the user reduces the global light, the elements would become a bit darker.

Altering the State of Virtual Controls Since virtual controls are not physically present in the scene, but instead are computer-generated elements added to the environment, their state can be modified on-demand. For instance, the user might trigger the visualisation of a virtual control by clicking or pointing at a button.

Controlling Travel Users can also change their own position in the virtual world, by travelling from one position to another. This technique fall under the second interaction key that we present below: navigation.

As a visual representation of the main gestures used in AR we propose the experiment carried out by Piumsomboon et al. [103]. In this study, the participants were asked to perform some mid-air interaction with the application in order to complete some different tasks. The results showed a total of 800 gestures, that were then classified according to their similarity. By using the "similar gesture" constraint to group those static pose and path gestures that were identical or behaved in the same way, the original 800 gestures were reduced into 320 unique gestures. The authors were able to distinguish eleven major variants of observed hand poses, showed in Figure 5.1, which can be used interchangeably. Finally, the 44 most used gestures were selected to make the so-called consensus set. These final gestures are depicted in Figure 5.2. As the reader can notice, to scale an object, different alternatives are available and proved to be efficient: both one- and two-hand gestures might be adopted. Hence, the user can recur to head-mounted displays as well as handheld devices to accomplish the task.

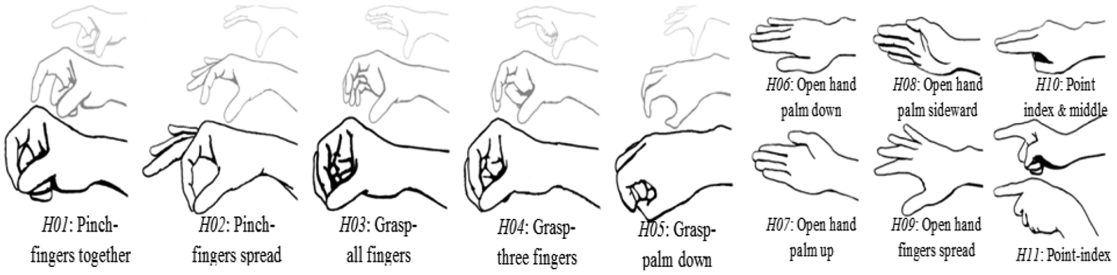


Figure 5.1: Variants of hand poses that Piumsomboon et al. observed among all gestures, identified with the codes H01-H11 [103]

5.2 Navigation

There exists a number of different alternatives to move from a place to another in a mixed-reality context. Navigation involves two phases: wayfinding, which is the awareness of where the user is where they want to go, and travelling, the actual act of moving in a certain position in space. Sometimes wayfinding is not performed at all. For instance, when the user just moves around to explore or understand their location, or when there is simply no need to think about where to go (so-called *maneuvering*). It is important to provide the necessary tools to inform the user about their position in the environment and allow them to build their mental model. A common way to assist the user in this process is through maps, which probably constitute the best way to offer situational awareness. Other methods are using a compass, displaying some signs (e.g. arrows) on the path, and guiding the user via the system's assistance (e.g. superimposition of text or voice directions).

Unfortunately, as Craig states, a problem of travelling in augmented reality settings is that the user might end up in places where the application cannot track them [100]. For this reason, many systems have designed different strategies to warn the users that they are leaving the boundaries of the tracked world by showing a virtual fence, activating an alarm or any warning signal, or displaying some text on the user's view.

5.3 Communication

In some circumstances, interacting with others represents a fundamental aspect of the a mixed-reality experience. In fact, it is often desirable to have multiple participants both in the virtual and in the real world (and hence in the AR experience) [100]. When multiple users interact with the purpose of completing a

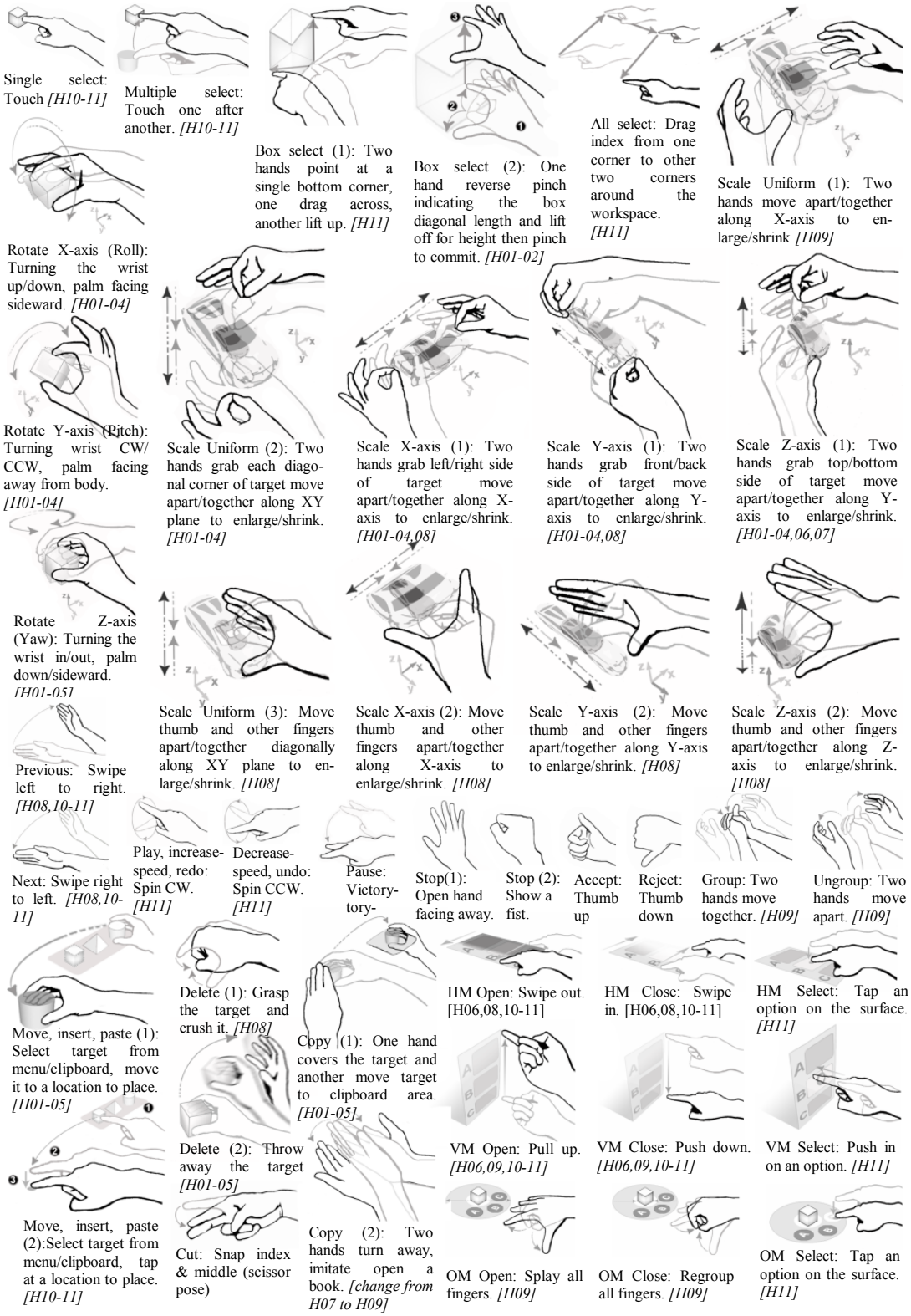


Figure 5.2: The user-defined gesture set for AR. The number showed between parentheses indicates multiple gestures in the same task. The codes in square brackets indicate the eleven hand pose variants (see Figure 5.1) that can be used for the same gesture [103]

task or solving a problem, we refer to *collaborative experience*. However, users might interact with each other not to achieve a common goal, but just to have a conversation and socially entertain a relationship. Therefore, the experience is only *shared*. In this regard, Facebook, one of the most popular social networks, has recently bought Oculus (virtual reality device presented earlier), as a proof that VR (as well as AR) is increasingly becoming a social computing platform.

It is important to note that a collaborative experience is shared, but a shared experience is not necessarily collaborative. Ideas, annotations, the virtual world itself and users' avatars are examples of elements that can be shared in mixed-reality experience. Furthermore, in a *full multipresence experience* all users live the experience in immersively, whereas in a nonimmersive experience, participants have more the role of viewers. Hence, instead of sharing the control of the whole virtual world, users share only the control of the viewport.

Probably, the best way to communicate with someone is aurally, which is also the quickest method to receive and process information from the system. Another efficient means used to convey actions to other participants is by exploiting body gestures. This technique of communication is often preferred because simple body gestures can be transmitted using much less bandwidth than even low-quality speech [99]. A commonly adopted solution that integrates both visual and aural information is video conferencing, which can be easily incorporated in mixed-reality systems. However, as one can imagine, the most intuitive and efficient way to collaborate with someone is by physically being in the same room. This is possible with spatial augmented reality systems, such as a CAVE, where all users can benefit from all the advantages of a room size system with the addition of the extremely high resolution imagery on the walls [93], as showed in Figure 5.3.

Finally, the communication might occur in a synchronous or asynchronous way. According to the task in question, it may be paramount or not essential that the information is conveyed instantaneously. The two main methods to interact with others asynchronously are annotations and experience playbacks. The former consist of notes placed on the virtual world by participant to explain, ask questions or reviewing some content. The latter consist of storing the participants' actions over time, enabling their consultation or replay in a second moment. Experience playbacks are particularly helpful in training scenarios where the learner has access to the teacher's instructions whenever they need.

We were able to identify five main operations in the domain of augmented reality interaction: selection, drag, rotation, resize and zoom. These actions can then be combined with each other in order to perform some more complex tasks. An example is the drag-and-drop feature which can be seen as selection, followed by

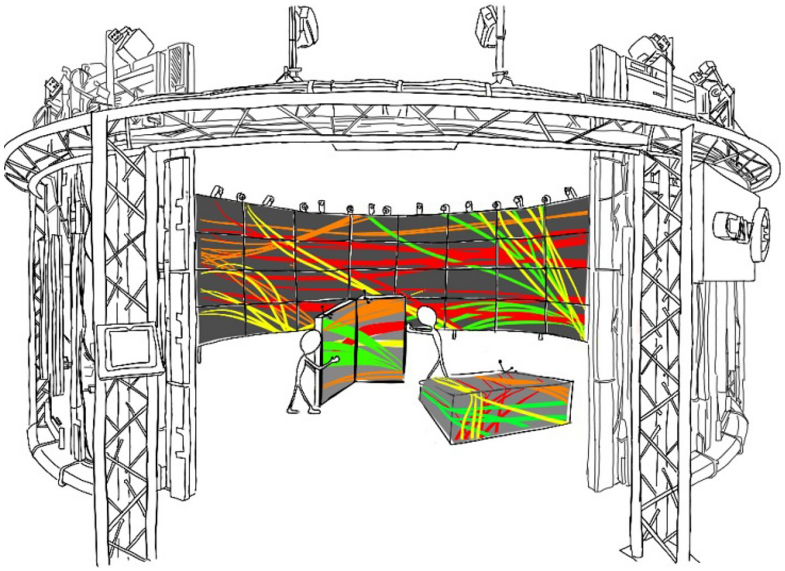


Figure 5.3: Representation of SAR in a CAVE setting [93]

dragging, and deselection at the desired destination. We also noticed that the techniques used for resizing an object are the same (or very similar) to the ones for zooming in or out of an area. Hence, we categorised these two operations under the same interaction method. Table 5.1 summarises what has been discussed in this chapter.

Table 5.1: Summary of the different methods to perform the major interaction operations

<table>
<tr><th></th><th>Voice</th><th>Touch</th><th>Gestures / Body</th><th>Gaze</th><th>Controller / Device</th></tr>
<tr><td rowspan="2">Selection</td><td rowspan="2">Provide coordinates
Provide position
in relation to a landmark</td><td rowspan="2">Single or double tap
Type on virtual keyboard
Select from menu list</td><td>Finger pointer
Grab with fist
Torso-directed
Contact-select (avatar)
Go-Go
Aperture-select
Select from menu list
Select-in-miniworld</td><td>Dwell-time looking at item
Blink while looking at item
Select from menu list
by blinking</td><td rowspan="2">Click button
Type on keyboard</td></tr>
<tr><td colspan="2">Reticle-directed</td></tr>
<tr><td>Drag</td><td>Provide destination to
the system</td><td>Drag the finger(s) on the
screen until destination</td><td>Grab with fist</td><td>Gaze tracking while carrying
the grabbed object</td><td>Move controller to destination
and release or click again the button
to drop the item</td></tr>
<tr><td>Rotation</td><td>Provide voice command</td><td>Move fingers or hands
clockwise/anticlockwise
without detaching from
the screen</td><td>Handle bar metaphor</td><td>Select rotation choice from
menu</td><td>Handle bar metaphor
Select rotation choice from menu</td></tr>
<tr><td>Resize /
Zoom</td><td>Provide voice command</td><td>Vary distance between
hands or fingers</td><td>Grab with fist</td><td>Select desired size from
menu</td><td>Handle bar metaphor
Select desired size from menu</td></tr>
</table>

6 Augmented Reality Interface to Control the IoT

As anticipated in Section 1.2, we aim to design an interface that allows the user to detect the smart objects in their room and interact with them naturally. The interaction can be of two types: explicit, when the user deliberately performs some actions on a specific smart object (e.g. turning on/off an appliance or changing the temperature of the oven), or implicit, when the actions performed on a smart object are carried out by the system as a consequence of the existence of a certain rule. For instance, the user might define a rule that specifies that when the window is open, the heating must be turned off. Hence, if the heating is on and the user opens the window, they also implicitly turn the heating off, if it was on, otherwise its state remains unchanged.

In this chapter we propose some methods to interact with smart objects in an IoT setting. Thus, possible techniques to manipulate smart appliances by means of an augmented reality interface. The ways to carry out these different operations can change significantly from one type of device to another, and from one category of interaction (e.g. gaze or touch) to another. This is where the content of Chapter 4 and Chapter 5 comes in handy. Subsequently, we describe some use cases where the proposed interface might be beneficial to the user in order to reach the goal they have in mind. Finally, we will show how the above-mentioned functionalities work practically, that is by building a prototype as a proof of concept.

6.1 Functionalities of the Interface

The interface we propose is designed to be used with each type of device available. All the possible operations the user can perform on the interface are therefore described in detail, by taking into consideration the characteristics of the different categories of displays widely explained above.

Search for smart objects in the real environment

The basic operation offered by our interface is searching for all the smart appliances surrounding the user in the physical environment. As anticipated earlier (see Section 2.2.2 about the mediums of deployment in the IoT), there exists different techniques to connect and control the Things in the Internet of Things. For instance, RFIDs and Wi-Fi networks are the most used ones. Therefore, when the user requests to see all the smart devices present in the room, the system would be able to show them on the interface by means of these mediums. Specifically, the information retrieved from the smart objects in the IoT includes the location where this Things are installed in the real environment. This enables the application to place the relative images on top of the physical appliances, in a way that the user can identify immediately the kind of smart object in question and its position in relation to the room and the surroundings. An example of how this would look like is depicted in Figure 6.1. Furthermore, not only can the user retrieve all the available objects in the room, but they can also search for a particular Thing or group of Things.

Figure 6.1: Search for all smart objects in the surroundings

When the room is scanned the first time, the interface takes approximately thirty seconds to receive the data from the Things in the IoT, and consequently build and save the room in the system. The following times the user uses the application in the same room, it is not necessary to scan it again because all the information has already been registered. In case the user installs a new smart appliance or wants to check whether something new has been applied to the room since the last time, they can scan the room again in order to update the data previously stored.

In the event that the application is launched from a handheld device, the user has the possibility to tap on a button on the screen to begin the search. The system would then show some digital content, such as an image of the type of Thing in question, on top of each smart object present in the room, as a sort of placeholder. Consequently, it is intuitive and straightforward for the user to understand at a glance the type of appliance in question and its position in relation to the real world. Of course, since the size of the screen is limited, it is likely that not all the Things would be shown to the user at the same time. For this reason, the application provides some hints to inform the user that, even though not immediately visible, there are some other objects out of the field of view. An efficient way to convey this information is by superimposing some arrows on the edges of the screen and showing a small icon of the object in question. For instance, if the system identifies a coffee machine on the left of the user and the camera of the handheld device is facing ahead, the appliance would not fit on the screen. Hence, an arrow pointing toward the left of the screen serves as an indicator that the object depicted in the small icon (the coffee machine) will appear on the screen if the user follows the indication with their camera (that is by moving it on the left). Figure 6.2 shows an example in which the user is pointing their smartphone toward the kitchen and the interface displays some arrows as a notification that by moving the camera in those directions, new smart devices can be discovered. In particular, by moving the phone up the user could see the kitchen light, and on the right side another light and the toaster are available, as the small icons next to the arrows suggest. An alternative could be to explicitly instruct the user by using voice signals. In other words, the system emits some predefined sentences (e.g. "move your device to the left to discover more objects") notifying the user about the hidden Things. Given that sound is not always the preferred option, the user is able to select from a menu which interaction means suits them the best.

Users could also start the application with a head-mounted display or smart glasses. In this case, it would be possible to click on the search button via the controllers held in the participant's hands or by using mid-air gesture commands. Touch interaction is an alternative when the device is used in combination with a touchpad. Here the user would simply touch a button likewise the scenario mentioned above. Gaze represents another option, but nowadays other techniques are preferred because considered more efficient. On HMDs and glasses, the field of view is wider than on HHDs but it still cannot cover the whole real environment. Hence, the same hints described above are necessary in these scenarios as well.

Finally, whatever device the user resorts to, voice commands are always available as a means to instruct the system to search for Things in the room. We could imagine of short sentences such as "search smart objects in the room", to which

Figure 6.2: When the interface detects some objects in the surrounding that cannot fit on the screen, some arrows on the edges are displayed

the interface would reply by displaying the digital content. Of course the user can hide the available devices by repeating the same procedure adopted to show them (that is taping the same button again with touch interaction or with controllers) or by providing the opposite voice commands used to show the objects (e.g. "hide smart objects in the room").

On the other hand, as introduced earlier, the user might also decide to look for a particular Thing in the environment. For instance, if someone is using the interface in a room they are not familiar with, this option would enable them to quickly check whether the desired appliance is present or not, without scanning all of them. For this purpose, a button would be available on the interface, ideally with a magnifying glass. The user could then have the choice of typing the name of the smart object on the keyboard, or use their voice to instruct the system to search for it in the environment. This functionality is useful if the user is interested in one particular Thing, such as the coffee machine, or a group of Things, in case there are more of them. Figure 6.3 shows an example where the user is in the kitchen and wants to look for the thermostat. They type the word "thermostat" on the keyboard; subsequently the interface shows an arrow on the left edge of the screen, therefore the user follows the arrow until the living room where the thermostat finally appears on screen.

Since this interface has the goal to enhance the functionalities of what is currently available for the public to control the IoT (e.g. home automation) by adding augmented reality for the visual graphics, the tasks can still be achieved in the "traditional" way. This means that all smart objects or only the specific ones the user is interested in can not only be retrieved and displayed in AR through the

Figure 6.3: Search for a smart object by typing its name on the keyboard

red digital images described above, but they can also be visualised in form of a list that the user can consult without actually using their camera and walking around the room.

Visualise all existing rules between smart objects

Similarly to the previous operation, the user can select a button or provide a voice command to obtain a visual representation of all the rules defined. Rules can involve multiple smart objects as source and as targets. But they can also be triggered by events not necessarily linked to Things (e.g. at a certain time, the alarm is set to on) or situations such as "going to sleep". This distinction will be explained thoroughly in the operation concerning the definition of a rule.

Graphically speaking, in the first case, when a rule involves smart objects, the interface visualises an outgoing arrow from one appliance to another, as a logic representation of the fact that the behaviour of the first one affects the second one, according to some predefined control rule. This coincides with an incoming arrow to the second Thing, coming from the first one it has a relation with. If a rule involves objects that are distant from each other and hence they currently do not fit in the viewport, the arrow is still visualised, enabling the user to follow where it leads to or comes from. Figure 6.4 shows a graphical representation of this operation.

In the second case, when a rule has only a target object because triggered by certain circumstances instead of an actual explicit action performed on a Thing by the user, the visual representation on the interface corresponds to a green button on the left side of the object. This button, once clicked, opens a pop-up list of all the events and situations that need to happen in order to execute some actions, by altering the properties of the target object of the rule itself. For instance, the rule that says "if it is 11pm and everyone is in bed, then set the home security system to on", would be visualised as depicted in Figure 6.5. On the left of the house alarm, the user can see a green button to be consulted to check what events and situations modify the behaviour of the alarm without any human interaction. In this case,

Figure 6.4: Visualise all existing rules between smart objects

the list includes two rules responsible for turning on the alarm: (1) the first uses the event that it is past 11pm as an event, and the situation that "everyone is sleeping"; (2) the second rule checks whether the door is locked (event fired by the smart door) and that no one is at home (information sent by the sensors around the house).

As one can guess, the presence of a high number of arrows would lead to an overcrowded AR environment, which would be confusing and not very practical, especially in a limited size screen such as the one on a smartphone. For this reason, as a default option, no rules are displayed to the user, but they can easily retrieved on-demand by a simple click/tap/voice command. As an alternative, the user might decide to hide the overall rules of all smart devices and only display the ones specific to one Thing (see next operation).

Finally, another interesting feature in the context of rules that might increase the usability of the application is the visualisation of an object's rules when it is captured by the camera in the centre of the screen. In this way, the user does not need to perform further actions to check the predefined rules of the Thing they are currently looking at. Additionally, since the target objects might fall out of the field of view, it would not be practical to move the device's camera in order to discover where the arrow leads to. This in fact would mean not to have the desired smart object in front of the camera anymore, therefore the rules would not be visualised (otherwise of course the user explicitly chooses to visualise the rules from the menu, as explained in the next operation). For this reason, we thought of a new technique that enables users to have a sort of preview of all appliances involved in a Thing's rules, by showing their miniatures around the source object. Their disposition suggests the direction the smart object is located in the room,

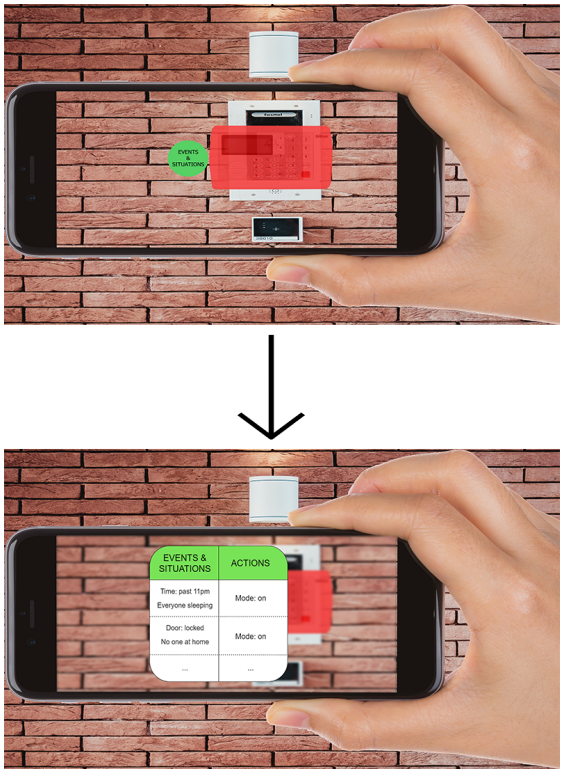


Figure 6.5: The user first selects the home security system or points at it with their camera, and then presses the green button in order to visualise events and situations

in relation to the source object in question. The user can recognise whether a miniature refers to a smart device out of the field of view of the camera by its dashed arrow. On the other hand, plain regular arrows are used for Things closeby the source object. In the example in Figure 6.6, the reader can notice how the presence of the TV in the centre of the smartphone's screen triggers the visualisation of all the rules where the TV is the source object, even though the targets are not in front of the camera. In this case, since the TV has some dependencies with the speakers, the light, the fireplace, the door and the lamp, these Things are displayed as miniatures all around the source (TV). Since the light, the fireplace and the door's miniatures are linked to the TV through some dashed arrows, the user can immediately understand that these smart Things are not closeby the TV, but by moving the camera in those directions, they can discover their exact positions in the physical environment.

On the other hand, the events and situations in place when the rule is triggered are still visualised when clicking on the green button on the left side of the central

Figure 6.6: When an object is captured in the centre of the camera, the visualisation of all its rules is triggered, together with the miniatures of all the smart objects involved

object, as described above. This is depicted in Figure 6.7, where we propose a zoomed-in image of the TV in the centre of the screen. This miniature feature can increase the usability of the application because without performing any explicit interaction, the user can consult the rules of the desired Thing, by simply capturing it in the centre of the screen.

Figure 6.7: Events and situations button on the left side of the object in the centre of the screen

Select a smart object to get information about its properties and display rules with other objects

When controlling an object, the user must first select it and then manipulate its state and behaviour as they wish. As explained in Section 5.1.1, selection is

an essential operation that is used as a starting point to perform most types of "complex" operations, such as the classic drag-and-drop, where an object is firstly selected, then dragged to the destination and finally deselected (dropped). In our interface, it is possible to select a smart object displayed in the augmented environment, and retrieve all the information related to it.

Different methods are available to select an object. For instance, in case of a handheld device, the easiest and most straightforward way is the touch interaction, with which the user tap on a smart object on screen in order to open up a menu. At this point, it is possible to consult and alter the properties of the Thing selected, such as the temperature of a thermostat or an oven, the timer of an electric induction hotplate, or simply the on/off mode of a TV. Similarly, the properties can be consulted and altered from the list of all smart objects present in the surroundings. From there, the user selects the desired object and modifies the properties they are interested in. This allows the user to manage their smart home remotely, without physically standing in front of the smart object, which is the other new option offered by our application. Furthermore, the user can choose to display all the rules defined for that particular appliance, by ticking the "show rules" option. In this way, the environment would not be overcrowded with plenty of arrows, but only the ones concerning the selected Thing. This functionality is particularly useful when using an AR display with a limited screen size, such as the handheld ones. In the same way, the user has the possibility to hide the selected object's rules from the menu by toggling the same choice. In addition to the arrows that lead to the target objects of the selected Thing's rules, the interface also shows a green button on the left side of the device in question that enables the user to consult which events and situations not directly related to specific smart objects' properties act as sources in one or more rules. Figure 6.8 shows the result of selecting a smart object (in this case a TV). The application visualises a pop up window with the Thing's properties and the option to display its rules. In this case, the user is interested in displaying its rules, therefore as soon as they tick that option, the outcoming arrows populate the interface. When deselecting the device or simply touching outside the pop up menu, the user can clearly see the scene with all the requested arrows, as depicted in Figure 6.9.

If a head-mounted display is used instead, the preferred technique to select a Thing would probably be by means of controllers. Once the menu shows up, the user can then decide to visualise the properties and their states, as well as get the rules where the object is involved. Mid-air gestures represent another method to select an object by pointing at it and then picking the desired choice from the menu. Similar techniques are adopted in a spatial augmented reality setting, such as a CAVE.

Figure 6.8: When selecting a Thing, a pop up window with its properties and the option to show its rules appears

Figure 6.9: All existing rules where the source object is the one selected by the user

What differentiates this operation from the one described above is that when the user captures a Thing in the centre of the screen, its rules are momentarily visualised with the miniatures of the target objects involved. However, when the user moves around their device, the superimposed arrows disappear because the Thing is no longer in the centre. On the other hand, when the user selects a smart object and chooses to display its rules, the arrows appear on screen permanently until another action is performed or the user explicitly decides to hide the rules from the menu.

Control a smart object by altering its properties in accordance with the predefined rules

While the previous operation aims to consult the information related to a selected smart object, the purpose of this one is the manipulation of the Thing itself by altering the state of its properties, making sure that this action is allowed by the existing rules. This is what we call "explicit interaction", since the action on the object is performed explicitly by a human being. As mentioned previously, properties might be the temperature, the volume, the timer and so on, and their values can be modified by the user by means of the popup menu showed on the proposed interface. These properties can be accessed and altered from the list of smart objects in the surroundings (the user selects the desired smart Thing from the list, open its properties and modifies the one(s) they are interested in), which is what is offered by current applications to control the IoT. This allows the user to perform some actions remotely, without being physically in front of the smart appliance. The other option is to tap on the corresponding AR image in the scene to select the smart device and alter its behaviour from a popup menu, such as the one in Figure 6.8. In a HH setting, the user can tap on a specific property and type a certain value (e.g. type 20 as the number of degrees the air conditioner should be set to, by using a virtual keyboard). In an HMD setting, controllers are definitely the most suitable way to enter some numeric data or switch a button from on to off, and vice versa. Voice commands can also be used to instruct the system to change a property to a desired value, especially in case the user's hands are occupied (e.g. the user is cooking and wants to switch the oven off without touching anything with their dirty hands). In this case, mid-air gestures with head-mounted displays, smart glasses or spatial displays might come in handy.

The alteration of some property on an object coincides with an event that may or may not have a corresponding rule responsible of modifying, in turn, properties of another Thing. This is what we call "implicit interaction", because it is not performed directly by the user, but it is a consequence of a rule triggered by a source event or situation. In other words, changing a property can entail a series of adjustments on other objects' properties simultaneously, without breaking any other existing rule. This helps avoid unwanted situations in the environment. For instance, let us take into consideration the case in which the user has defined a rule that involves the smart window in the kitchen as a source object and the heating as the target one, in a way that when the user wants to open the window, the heating turns off. If, at any moment, the user wants to open the window while the heating is on, the system would show a warning on the screen to convey that the action carried out by the user on the window has provoked the shutting down of the heating. In this way, the user is aware all the time of what is happening

in their smart home, even when the actions have not been performed directly by them, but from the system itself following some predefined rules.

Finally, by checking the state of the objects' properties and by retrieving information from the sensors around the environment, different situations are created, which can be used in rules to trigger some actions. As an example, by exploiting the sensors in the house's beds, the system can know if all people living there are actually laying on them. Through this information, plus the event that it is past 11pm, the situation "everyone is sleeping" is activated. As a consequence, the rule "if everyone is sleeping then turn on the home security system" is triggered and the alarm is set to on. Additionally, the fact that the house alarm is in place might represent the source event of another rule responsible of turning off the TV and all the lights. Thus, from a situation or event, a chain of other events and situations might follow.

Define rules regarding one or more smart Things

Probably, the most innovative functionality of the interface we are proposing is the possibility to define rules between two or more smart objects, and rules triggered when certain events or situations are fired. So far we have talked about searching and managing smart objects, and what happens when a rule is triggered. However, we have not described how the user can create their own rules in order to personalise the digital appliances in the real world according to their schedule and preferences. There are essentially three methods to achieve this: (1) choosing the "new rule" option from the menu, (2) performing the drag-and-drop operation from the source object to the target one and (3) tapping the source object once, moving the camera toward the target object and finally double tap it to create the new rule. These operations are explained in detail below.

A user can create a rule by selecting the "new rule" button on the interface by performing one of the possible techniques available for a particular AR display (HHD $\rightarrow$ touch, HMD $\rightarrow$ gestures/controllers, ...). Rules are N:N associations, meaning that the user can select one or more source objects, and one or more target objects. These relations take advantage of the smart appliances' properties whose values are continuously checked by the system. For instance, in a home automation scenario where people care about the environment, the user can select the smart window in the kitchen as a source element, and the central heating as a target. However, as depicted in Figure 6.10, since the heating is not among the recently used Things, the user can click on "browse Things" in order to look for other smart elements in the system. A popup window such as the one in Figure 6.11 will appear, allowing the user to type the name of what they are searching and

select as many Things as they want. In this case only the heating is selected. In Figure 6.12, the reader can notice that now the heating is in the list of Things to be added to the new rule, and the user can select it as the only target. The rule would exploits the properties "open" of the window and "off mode" of the heating making sure that when the user opens the window, the heating shuts down. The status of the properties is defined by the user directly from the rule settings menu that appears on the interface, as shown in Figure 6.13.

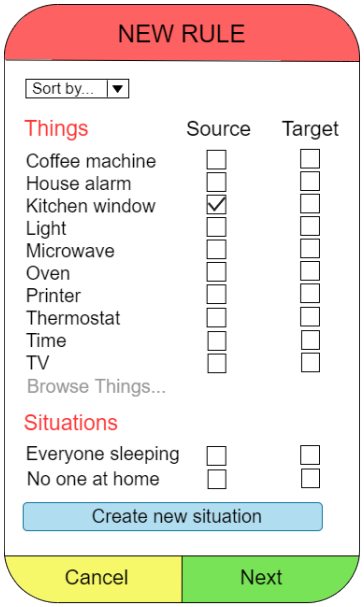


Figure 6.10: The user creates a new rule where the kitchen window is the source

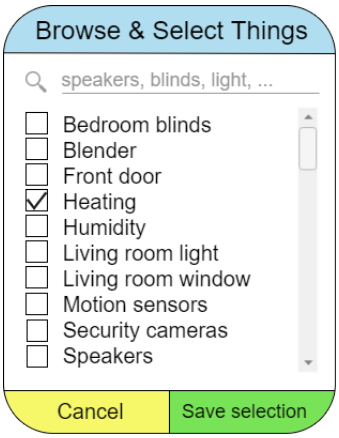


Figure 6.11: The user browses the available Things and selects the heater

Figure 6.12: The user can now select the heater as the rule's target

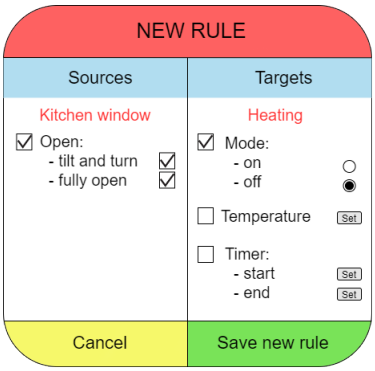


Figure 6.13: The user can select the properties of the objects involved in the new rule

Furthermore, as anticipated above, a rule might involve only one or multiple target objects, whereas the source is represented by certain information retrieved from the sensors placed around the house (e.g. user laying in bed) or by a particular situation. In fact, the context modelling toolkit [69] on which we based ourselves clearly explains that events and situations are responsible for the execution of certain actions. Since in our case we are in an IoT scenario where smart Things are involved, events can be triggered by any alteration of the objects' properties or any data received from sensors and actuators. Situations, on the other hand,

are defined by users themselves as a result of combinations of events happening simultaneously. For instance, the user might define a situation "watching TV" which takes place when the two following events happen: (1) the user is sitting or laying on the couch (event triggered by sensors) and (2) the TV is on (the event is the "mode" property of the TV set to on).

As anticipated earlier, a rule might present both events and situations as sources, as shown in Figure 6.14. In particular, they select the time event and the "everyone sleeping" situation as sources, and the TV, the light and the house alarm as targets. The user can also notice that they have the possibility to add new events and situations, in case they are not in the list of the ones used recently. The next step in the creation of a rule consists of selecting the desired properties to be checked (sources) or changed (targets). In this case, the user selects the property "after" and set the time 11pm, as the event necessary to trigger the rule together with the situation "everyone sleeping", and as actions to be performed they set the TV and light mode to off, and the house alarm to on. The representation of this second step is shown in Figure 6.15. The reader can also notice that the situation "everyone sleeping" is displayed as a button because by clicking it, the user can actually modify the situation as they wish.

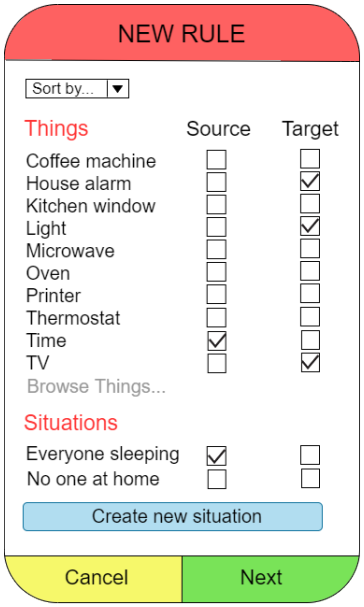


Figure 6.14: The user selects the time event and the situation that everyone is sleeping as sources, and the light, the TV and the alarm as targets

The other modality used to create a rule is by dragging a source object (or better, its virtual AR representation) and dropping it to the target destination, by means

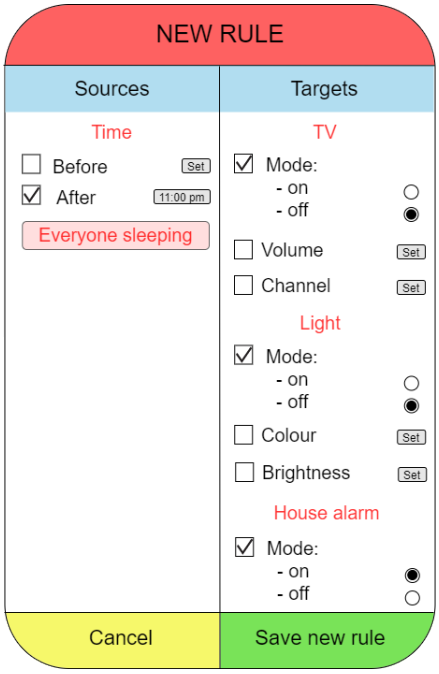


Figure 6.15: The user selects the properties of the sources and targets involved in the rule

of touch interaction or controllers. Basically it is the equivalent of reproducing with gestures the arrow from source to target that will be eventually be drawn by the system once the rule is saved. This method is a shortcut of the previous one. When the drag-and-drop has been performed, a pop-up menu appears, allowing the user to choose which properties of the two objects are involved in the rule. Practically speaking, a user might decide to create a rule where the source is the TV and the target is the living room light, in a way that if the TV is on, the light is off. Hence, they point the camera toward the TV, they drag it to the living room light and finally drop it. When the rule settings menu pops up, the user sets the property "mode" of the TV to on, and the property "mode" of the light to off. When saving the new rule, an arrow appears from the TV to the light.

The drawback of recurring to this modality is that if a rule involves multiple objects, the user must perform the drag-and-drop multiple times, one for each arrow. An example might be the definition of a rule where the house alarm controller is the source object that leads to the TV, the speakers and the oven. The user would have to drag and drop the TV three times, one for each target, creating one arrow for each gesture: (1) the arrow from the house alarm controller to the TV, (2) the arrow from the house alarm controller to the speakers, and (3) the arrow from the house alarm controller to the oven. For this reason, under these circumstances, the

creation of the rule from the menu explained above is the preferred option.

Finally, if the source and target objects are not closeby and the user does not want to drag the source on screen for a long time while moving the device toward the destination, possibly because of the fatigue (gorilla arm), instead of using the first method by selecting "new rule" from the menu, they can recur to a third option. The user can double tap the AR image corresponding to the source Thing, then move their device in order for the camera to capture the desired target, and finally double tap on this second digital image. The same window described above, with the first double-tapped object selected as source, and the second double-tapped object selected as target is presented to the user, where they can confirm the choice and pick the properties of the Things in question. The difference from the previous operation is that here the user is not obliged to drag the object and thus to keep their finger on screen until the destination is reached, which can be quite an advantage if the Things involved are spatially far from each other. However, once again, if more than one source or targets are involved, the user should perform the double tap multiple times. Hence, the "new rule" from the menu might represent the quickest option.

6.2 Implementing the Interface: a Proof of Concept

In this section we talk about the process of building the actual interface, implemented in a strictly demonstrative way. Furthermore, we will illustrate the features that have been implemented in this prototype, which were only described from a conceptual point of view up to this stage.

The idea was to exploit a technology capable of rendering augmented reality content on the browser, no matter the device from which the application is launched. Our initial choice was *WebXR*[1], which is an API designed to support virtual reality and augmented reality on the Web. Any device with three or six degrees of freedom, and with or without external positional sensors can be used to exploit the different features offered by WebXR. However, we soon learnt that WebXR only supports movement detection and no computer vision or marker detection. It hides the camera to the application due to privacy reasons, meaning that it cannot use any reference in the real environment to display the augmented reality content. Consequently, we decided to use another technology, whose purpose is the same, but with fewer restrictions. As reported in the official documentation,"*AR.js is a lightweight library for Augmented Reality on the Web, coming with features such as image tracking, location-based AR and marker tracking.*" [104].

[1] `https://developer.mozilla.org/en-US/docs/Web/API/WebXR_Device_API`

Ideally, the application should be able to detect the IoT smart objects automatically, for instance by means of WiFi or bluetooth. However, since what we wanted to build was a proof of concept with the only purpose of showing the general functionalities described in the previous section, we resorted to other simpler means, and leave the integration of the whole IoT-related architecture for future work.

Our first approach involved markers, in particular ArUco markers, which are called "pattern markers" in the official AR.js documentation. By placing markers all around the room, we were able to superimpose some digital content to them when detected by the camera. Hence, when the smartphone was used in the direction of the TV, a red image of a TV appeared in correspondence of the position of the relative marker, enabling the user to manipulate the real TV, by applying changes on the virtual one. We also noticed that the interactions with the images worked better when they were integrated inside a 3D model, for instance a cube. However, since the content was showed only when markers were found, the interface could not display arrows between objects that could not fit in the device's screen. In fact, the system did not know a priori the position of the markers in the room, nor it was able to remember their locations after detecting them the first time. As illustrated in Figure 6.16, the objects need to be captured by the camera simultaneously in order to display the rule between them. Hence, we had to change our approach to the problem.

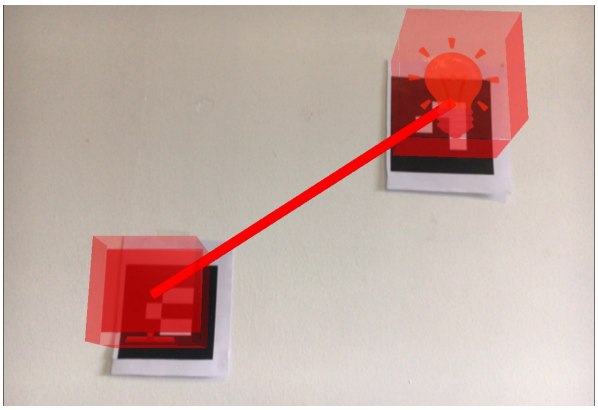
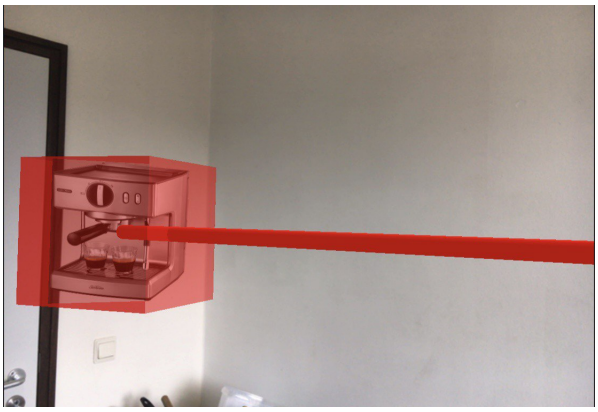


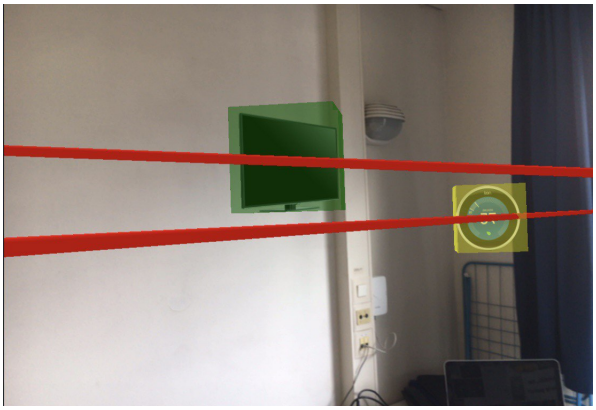
Figure 6.16: Two markers representing respectively a TV and the light, connected by a rule

What we tried next was the location based augmented reality, which is used both for indoor (less accurate though) and outdoor geopositioning of AR content. The official documentation presents all the different choices to load places. They can be loaded statically, from HTML or from JavaScript, or data can be loaded through local/remote json, or even through API calls. For simplicity, we used

static coordinates that work in relation to the device's position. Therefore, we defined some simulated latitude and longitude and we assigned to each element in the scene some coordinates slightly different from the starting point. This means that when opening the application, the user could move around the smartphone and discover the objects' images surrounding them. With this approach, we were able to display arrows between objects even though they were not close to each other and not in front of the camera, as depicted in Figure 6.17. The inaccuracy of the GPS is showed once again in Figure 6.17b, where the arrows do not stop in correspondence of the target objects, but they continue, even though the precise coordinates of the floating figures are used as destination points.



(a)



(b)

Figure 6.17: Rules are displayed even though the source or target object(s) are not in front of the camera

However, as already mentioned, the GPS inside the buildings is still not very accurate and, since the use case of this application is inside the individuals' homes, this solution was not successful. In fact, we were not able to precisely identify the

exact position where we wanted to show our content, because depth and distance from the starting point were difficult to set and manipulate.

For this reason, we decided to go back to the marker-based AR, accepting the limitation that the content we wanted to show had to fit necessarily in the same field of view in order to be able to see the arrows representing the rules. The only obstacle was the interactions needed to select a smart object to retrieve and alter its properties. With one marker in the scene, AR.js and touch interactions worked perfectly: when the user touched the TV in AR, a menu on the top-left corner of the screen would appear to show the properties (see Figure 6.18), and by touching it again, the menu would disappear as a consequence of the deselection. The problem was that we needed multiple markers in the scene (one for the TV, one for the light, one for the stereo, ...) and this seemed to create many issues in the application. By contacting the maintainer of AR.js, we were informed that, to this date, it is still not possible to interact with multiple markers in the scene. The library was released in 2018 and it is currently under development. They said that they would implement the possibility to add listeners to different markers in the future but, since it will not be anytime soon, we had to change strategy once again.

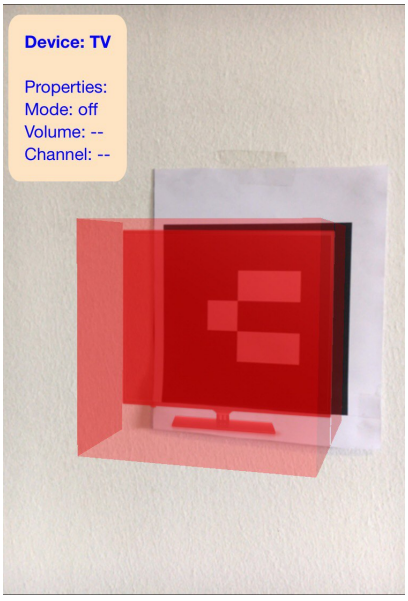


Figure 6.18: When selecting an object, its properties appear as a menu on the top-left corner of the screen

Finally, we switched to *Unity*[1], the most popular platform to create interactive, real-time content, such as 2D, 3D and VR games and apps. We could therefore simulate an augmented reality environment where the camera captures the real environment and the interface superimposes some images on it, as shown in Figure 6.19. Here we can see a bulb that superimposes the actual lamp in the room, and a TV that is used as a placeholder for the physical TV in the room (which in reality is not there because the real environment was not equipped with that Thing). We can notice an arrow between the TV and the light, meaning that there is a rule in place where the TV is the source object and the light is the target; and an outgoing arrow on the bottom of the image. By following this arrow with the camera-equipped device used (this screenshot was taken from a smartphone), the user can discover which is the target object included in that rule.

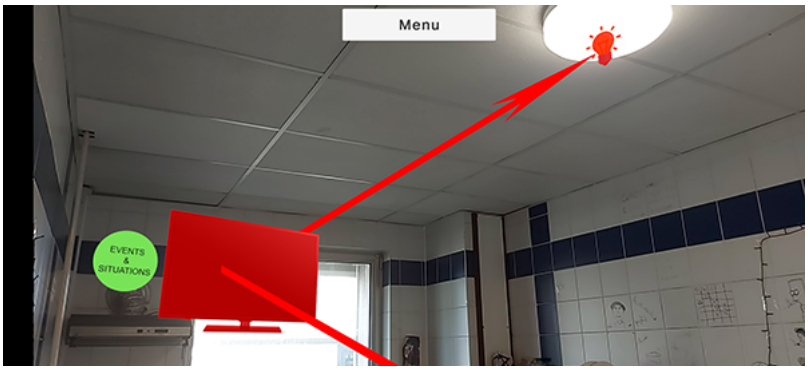


Figure 6.19: The images of the TV and the light are superimposed to the real environment, as well as the arrow that links these two Things

Initially, we decided to implement the application in a way that it could scan the room, detect the surfaces and the disposition of objects, and eventually place the AR images in specific locations. Hence, the disposition of the digital content would have been related to the actual physical surroundings. However, when going down to this road, we realised that the application was unstable and not very accurate at identifying the distance between the walls and the angles in the room. Therefore, we opted to anchor the digital images to some coordinates in the 3D space and we assumed their presence worked as a placeholder for the real objects. Normally, as explained in the previous chapter, the images representing the Things in the IoT would correspond to the right positions of the objects in the physical space (e.g. where the TV is installed in the room, we superimpose a red image of a TV over it), but unfortunately the house where the application was implemented and tested did not present any smart appliances. Hence, the AR images are displayed in the

[1] `https://unity.com`

space without being linked to actual objects. We could also draw arrows between Things. In particular, we took into consideration the TV and we showed only the rules that included the TV as a source object, in order to avoid an overcrowded scene. The touch on each of the images toggles the appearance and disappearance of some text as depicted in Figure 6.20. This enables the user to see the object's name, check and control its properties, choose whether to display the rules on screen in form of arrows, and edit the rules where the TV is the source or one of the sources.

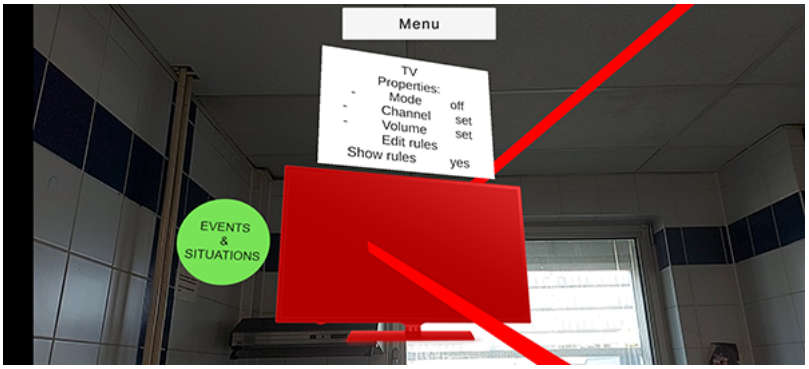


Figure 6.20: When taping on the TV, a popup window with its properties appears. Here the user can also edit its rules and decide whether to show or hide them

By tapping the "edit rules" voice, the user can check the rules where the TV is involved in an active way (as a source) and can modify them as they wish. The green button on the left side of the device in question enables the user to consult which events and situations not directly related to specific smart objects' properties act as sources in one or more rules. For instance, as shown in Figure 6.21, if it is eight pm and the TV is on (meaning that the user is actually home), the channel on the TV is set to 5 in order to show the newscast. Another situation that can change the TV's behaviour is when everyone is sleeping. Here we can think of a situation in which the user forgot to turn off the TV before going to bed or fell asleep while watching it. In those circumstances, this rule would come in handy because the application would take care of turning off the TV automatically after checking the presence of these events and situations.

Finally, from the menu the user can create a new rule, search for a particular smart Thing in the surroundings and show or hide the objects in the scene and their rules.

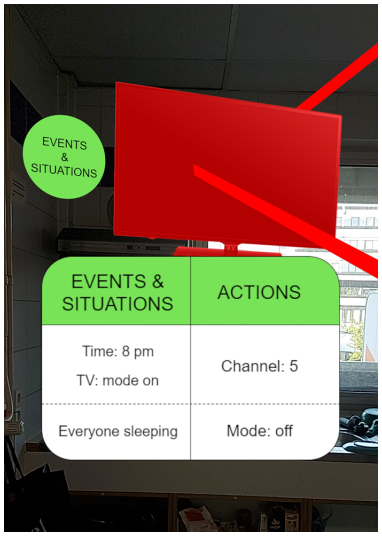


Figure 6.21: Tapping on the "events & situations" button the user can consult which events and situations trigger some actions on the TV

6.3 Validation

In this section we validate the application described above, by considering both the idea, purpose and functionalities, and the actual prototype implemented with Unity that supports what we would like to achieve ideally. Due to the current Covid-19 pandemic, it was not possible to perform a physical presentation of the interface to some potential users of the target audience, and neither were we able to let participants try the interface by themselves to get familiar with it. This means that certain aspects of the application, such as the usability, could not be evaluated. However, we managed to perform the inquiry by organising multiple video calls with four to six participants each, where we first presented the application in a more theoretical way and then showed a demo video of the working implementation to make the explanation more concrete. We took advantage of the mock-up screens used in Chapter 6.2 to support the concepts described with words, such as *augmented reality application that superimposes digital images to the real objects by receiving data from them by means of Bluetooth or Wi-Fi networks.*

The survey was created with *Qualtrics*[1] and consisted of a total of twenty three questions, most of which to be answered on a seven-point scale, where the adjective on the left has a negative meaning and the adjective on the right describes the value more positively, as depicted in Figure 6.22. They were divided into six main blocks, each focused on a different value of the product we wanted to investigate. We took

[1] `https://www.qualtrics.com/uk`

them from the extended version of the *User Experience Questionnaire (UEQ)*[1], a fast and reliable questionnaire to measure the user experience of interactive products, and to these, we added some open questions. The questions users were asked concerned the attractiveness, perspicuity, novelty, usefulness, visual aesthetics and intuitive use of the application in question. Figure 6.22 represents an example of questions asked to investigate the perspicuity value of the application. The second part of the question aims to understand how important the value taken into consideration is for the participant, in order to weight the four answers given in the first part. In fact, if the participant thinks that handling and using the application is understandable, easy to learn, easy and clear, and the perspicuity is a very important value to them (basically, they evaluate everything with seven points or values toward the right side), then we can conclude that their positive opinion is deeply relevant for our validation. On the contrary, if they think that handling and using the application is not understandable, difficult to learn, complicated and confusing, but then the perspicuity is completely irrelevant for them (values on the left side of the scale), then we know that their negative feedback should not affect much the way that we evaluate the application. The same modality has been adopted for all six properties investigated. The complete survey can be consulted in annex A of the Appendix.

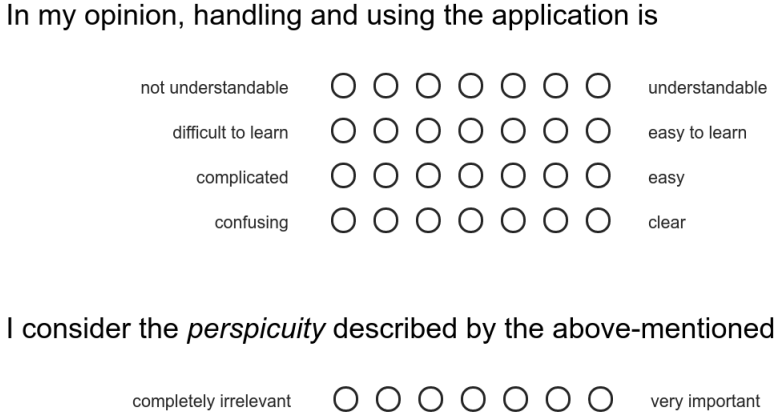


Figure 6.22: Example questions for the perspicuity value used in our survey

In addition to these application-oriented questions, the very first three questions of the questionnaire aimed to learn some general information about the person filling it out. In particular, we were interested in their age range, gender and familiarity with Computer Science, in order to understand whether there is any direct relation between these factors and the answers given by the participants. Given that we

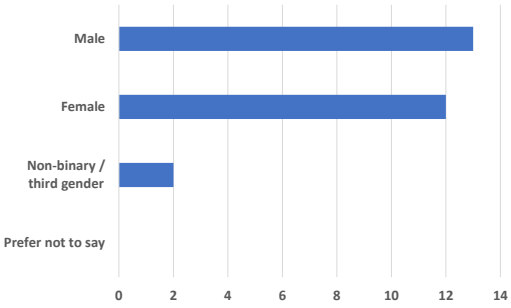
[1] `https://ueqplus.ueq-research.org`

designed the application to be accessible by anyone, whether they are old or young, or expert in the technological field or novices, we tried to have a balanced group of participants, especially for what concerns their age and the background knowledge in Computer Science.

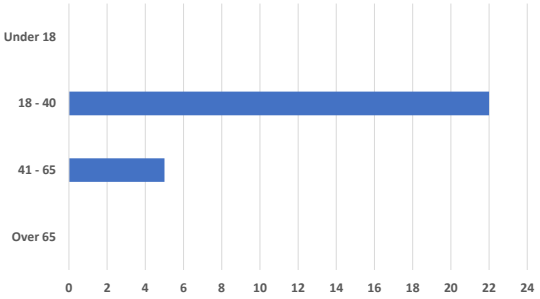
The UEQ also provides a tool to analyse the data retrieved from the participants' answers. We will present the most interesting and relevant information in the form of graphics and mean values, and we will include some of the suggestions and critics that we learnt from the answers to the open questions of the survey. In total, we presented the application to 27 people, which is also the same amount of replies we registered. Figure 6.23 provides a visual representation of the participants' personal information. As the reader can see, of these 27 participants, 12 are female, 13 are male and 2 are non-binary. For what concerns the age, 22 are between 18 and 40 years old, and 5 are between 41 and 65. Finally, on average, the participants are overall moderately familiar with Computer Science. 15 of them are equally distributed between very familiar, extremely familiar and slightly familiar, whereas 2 of them are not all familiar. This suggests a good variety of people with different backgrounds, which is a positive factor since we want our application to be as accessible as possible to anyone.

As anticipated above, there were six properties of our application that we wanted to investigate: attractiveness, perspicuity, novelty, usefulness, visual aesthetics and intuitive use. In order to calculate the means for the scales (that is the means over all items in a scale), standard deviations and confidence intervals are calculated. Since all our questions were constructed based on a seven-point scale, we transformed the mean values from a 1 to 7 range to a –3 to +3 range, where intuitively the neutral value zero is between the negative and positive ones. The results of the calculations are summarised in Table 6.3. The fact that none of the means of each scale (or property value) is negative tells us that the participants of our survey did not find any particular problem or aspect they did not appreciate regarding the application itself. In particular, the perspicuity, usefulness and intuitive use of the proposed interface present values higher than 2 (out of 3), which is a promising sign for our intention to make the application as straightforward and easy to use as possible in order to be accessible by all kind of audience. The lowest mean value is the visual aesthetics of the interface, but this is not an alarming result.

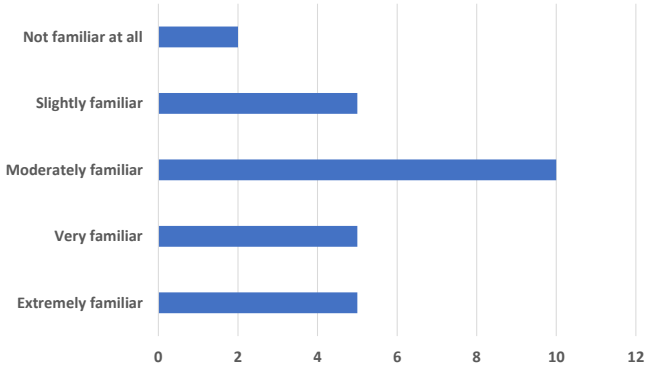
Figure 6.24 highlights the means and interval values just reported in a graphical manner. It is immediately visible that none of the property values goes below zero and that participants were not particularly enthusiastic about the visual aesthetics of the application.



(a) Participants' gender summary



(b) Participants' age summary



(c) Participants' Computer Science familiarity summary

Figure 6.23: Participants' personal data summaries

Scale	Mean	Variance	Std. Dev.	N	Confidence	Confidence Interval	
Attractiveness	1.89	0.75	0.86	27	0.33	1.56	2.21
Perspicuity	2.19	0.77	0.87	27	0.33	1.86	2.51
Novelty	1.75	1.31	1.14	27	0.43	1.32	2.18
Usefulness	2.08	0.79	0.88	27	0.33	1.75	2.42
Visual Aesthetics	0.56	2.19	1.47	27	0.56	0.00	1.11
Intuitive Use	2.22	0.72	0.84	27	0.32	1.90	2.54

Table 6.1: Mean and confidence interval per scale

On the other hand, the graph in Figure 6.25 shows the importance ratings of each of the application property investigated according to the participants of the survey. The most important values taken into consideration by our participants when judging an application are its perspicuity, usefulness and intuitive use. It is also relevant for them the novelty of the interface in question, whereas its attractiveness and visual aesthetics are overall less important.

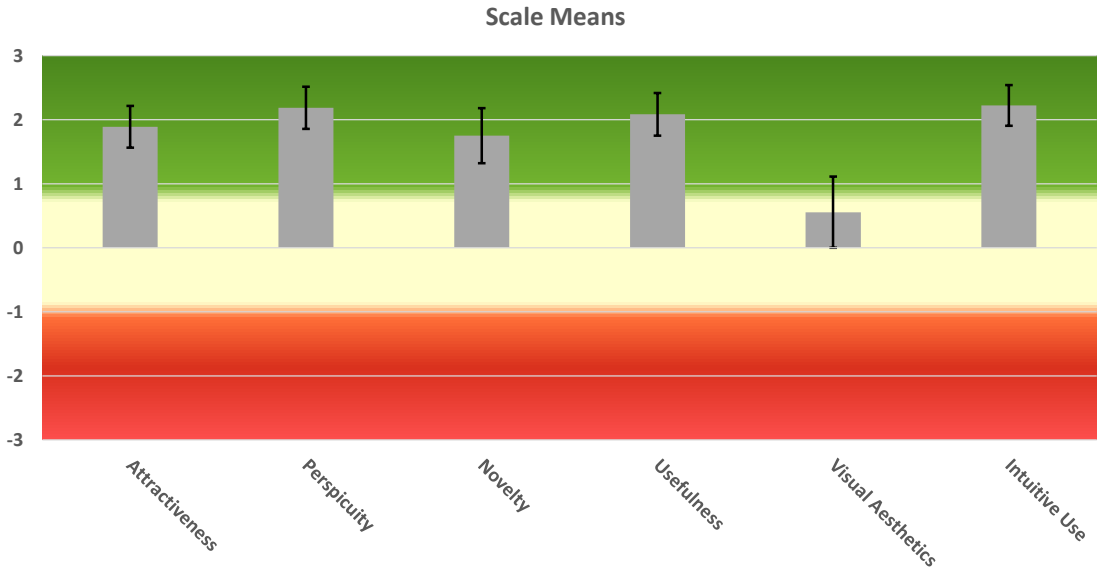


Figure 6.24: Scale means of the six investigated application properties

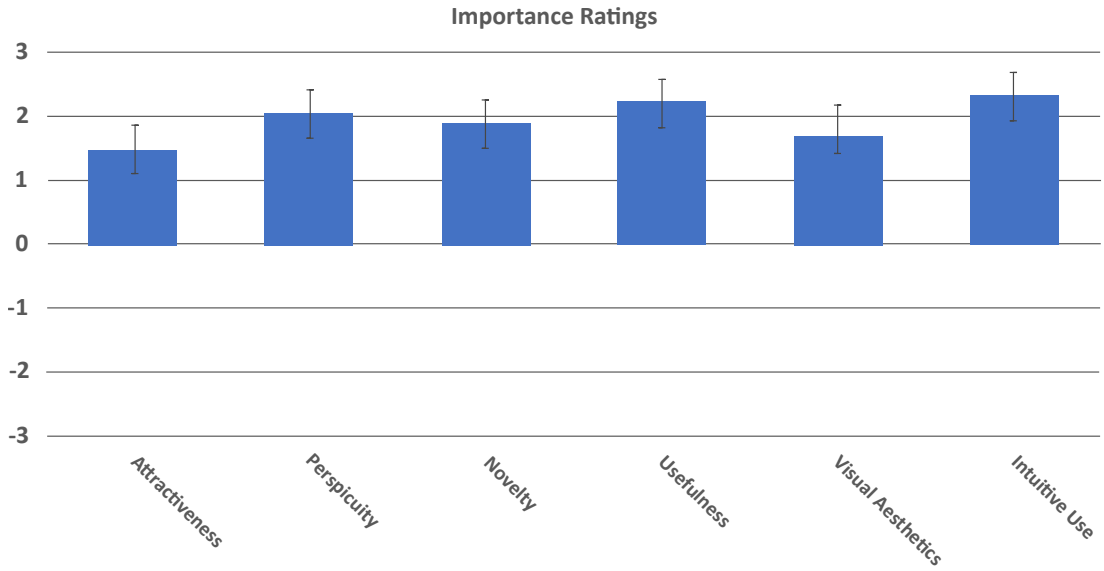


Figure 6.25: Importance ratings of the six investigated application properties

We also realised that some participants considered the attractiveness and visual aesthetics as referring to the same concept, which can be true if the user perceives something attractive according to its graphics and visual effects, rather than its usefulness and effectiveness. We discovered these different conceptions of the term in the open question asked, whose most interesting answers are presented below.

How would you make the application more attractive?

1. "*User interface: bigger buttons, brighter colours, glassmorphism and other modern graphics improvements. The arrows may be softer, with a lighter colour and a less stretched shape. The user experience is globally good, everything is easily reachable.*"
2. "*The application seems awesome and very user-friendly, but I didn't mark the max value in the "friendly" field since I'm a little concerned about the application aspect when many devices have rules assigned. Moreover, the menu could be hard to navigate on devices, due to the small buttons. Overall, looks great!*"
3. "*The arrows should not be stretched but with the line elongated and the arrow remaining small. Less strong and more muted colors.*"
4. "*Improve arrow graphics and enlarge menu keys before releasing the application.*"
5. "*Instead of using arrows, maybe make a list of which devices are connected to the source. And make like a drop down list. If you have 1 source and 10 targets, maybe it is a little bit messy to look at with all the arrows.*"
6. "*I really enjoyed the create new rule screens. A global overview of all devices (maybe located on a map of the house) could be useful. How is conflict resolution handled? E.g. when two rules want opposite results? Also the ability to add certain rules only for specific people could be useful.*"
7. "*The arrows currently appear rather large and clunky, the same as the devices, I think smaller less obtrusive icons would improve the look and feel of the application.*"
8. "*I suggest a prettiest graphic, try to simplify some action and for example have some 'suggest' from the app for the routines to apply, what can you also do with the objects.*"
9. "*To attract more people, a way could be to propose it to those with limited mobility.*"
10. "*Focusing on the fact that the application would represent an actual help for people in their home (especially if they are old and alone, or with disabilities, or simply people with not much knowledge of Computer Science) and it's accessible to anyone. Also, advertising the idea as a concrete and efficient way to improve safety controls in people's houses.*"

11. "*Proposing the application to people who are not at home most of the time, so they can control everything remotely, even in accordance to my engagements.*"

12. "*Adding a badge associated to each device that shows its primary properties (current status).*"

13. "*Advertising the application with daily-life examples in order to make people realise how often and easily they can use the application, and how much simpler the tasks can be carried out compared to situations in which the application is not available.*"

14. "*Smaller icons. Instead of identifying the TV thanks to a full size TV icon for example, I would put just a little TV icon as a "point of interest". This is to have less mess in the smartphone screen, which is already populated by the arrows.*"

What emerged from these responses is that some users consider an application attractive if it can come in handy to as many people as possible, which was more our intention. Therefore, they suggested some ways to present our application to different categories of users, such as people with physical constraints and those who are rarely at home, in order to reach the needs of anyone (see answers 9, 11, 12 and 13). More that once, participants stated that they would change the appearance of the arrows to visualise the rules between all smart objects (1, 3 and 7). They also seemed concerned about the case in which a source object is linked to a high number of target objects, making the augmented environment highly overcrowded in terms of arrows displayed. Answer 5 mentions a list of devices instead of the arrow visualisations, which is something that we included in the application but, since it was not implemented in the proof of concept, participants were not informed about this alternative for the consultation and visualisation of rules and smart objects. Finally, answers 6 and 8 propose to include a suggest button to create new rules, taking into consideration the conflicts that there might be between different smart appliances.

Furthermore, for what concerns the usefulness of the application, we asked users if they think they would use this application in their daily life. As shown in Figure 6.26, 20 participants answered positively and 7 replied "maybe". None of them thought that they would never exploit the application in their routine, which is a very promising result.

To those who answered "yes", we asked an open question to know what real-life scenarios they could think about in support of their positive statement. The most interesting answers are listed below.

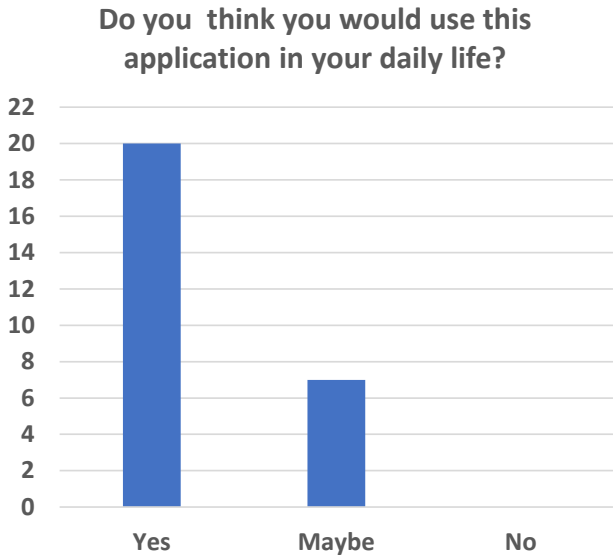


Figure 6.26: Usefulness of the application according to our study's participants

Can you think of a situation in which this application would come in handy? When would it be useful for you to exploit the application?

1. "*I'd use this application to prevent the power from falling. If I have some appliances working simultaneously and I want to turn on another one, the application would stop me in case this action would cause a blackout.*"
2. "*Automatically switch off the lights, turn on the Wi-Fi when I arrive at home, turn on the oven with an app button.*"
3. "*I think I would exploit the information concerning the weather conditions to automatically take some actions in the house or to warn me that it will rain or there will be a thunderstorm soon (and therefore I can consequently control the IoT in my smart home)*"
4. "*Netflix that opens when I pick a beer from the fridge would be a dream, otherwise I'd like to have the radio to start when I enter the shower.*"
5. "*Automate the heating temperature based on the presence or absence of people in the rooms. Close the windows if it rains. Making coffee when you get out of bed. Turn off various types of devices when you leave the house by locking the door.*"
6. "*In the scenario where the alarm goes off, the shutters go up, I get out of bed and the coffee machine prepares my coffee.*"

7. "*As a table tennis trainer, this application could be very useful to configure all devices in the sporting halls. E.g we can turn on light in different halls depending on the number of players, and we can prevent energy lose from devices that aren't turned off. The same applies for buildings such as schools.*"

8. "*I would definitely see this as a beneficial application to use in an office environment in which complex rules can be defined based on the specific people entering. E.g. your computer could turn on when you arrive at work, push notifications could be send, lights could be turned on automatically. And it would still be understandable to the people on the floor instead of only to the IT administrators.*"

9. "*Yes, I would use this application in my daily life. It would come in handy in situations of emergency, for instance when I'm not at home after a certain time, the system will turn the lights on automatically (I currently do it to prevent burglars from coming in) and will turn on the heating or the AC some time before I reach home.* [Translated from Italian]"

10. "*For example when I'm entering my home (so when I unlock the door and somebody in the house), the shutters will open, or when the sun goes down the lights will turn on automatically. Similarly, when I'm leaving home, the system will shut the shutters and the light will be switched off. Or for example if I'm at work and it's starting to raining, the shutters will be shut automatically.*"

11. "*When I step into the house, the light automatically turns on and the AC or the heating turns on (after the system has checked that the temperature is higher or lower than a certain limit. Hence, if it's summer or winter).*"

12. "*As an integration to domotic systems and to help people with disabilities in their homes to control their appliances.*"

13. "*I think that this application would be useful to reduce energetic consumption in anyone's home. In fact, by programming the exact time when the system is supposed to turn on and off the smart appliances, we can significantly reduce their usage time. Furthermore, by defining one rule, one simple action on the interface would be enough to control the situation on the smart home remotely, which is useful in case of emergency and it's not possible to come home.*"

14. "*When there are many appliances working simultaneously, the application could calculate how many kWs are currently consumed and check whether the power would fall, when I want to turn on another smart object.*"

15. "*To create a rule that checks if it's past of a certain time (e.g. 8 pm) and no one is at home (maybe because I had an inconvenient or an accident so I can't control the app personally), then the dispenser for the dog's food opens and releases some food.*"

16. "*I'm used to watching YouTube videos during my breakfast and dinner. This app seems useful for me to set the rules: If I sit on the chair, and the dish is on the table, then play "Watch Later" videos from YT or a specific news channel.*"

These answers helped us realise that the participants would actually exploit the proposed application, given that they provided their own scenarios in which the app would come in handy in their daily life. In particular, weather conditions have been mentioned more than once to trigger some automatic behaviours in the house, such as shutting the blinds. The calculation of the energy currently being consumed in the house in order to prevent the power from going out when turning on an additional device is an interesting situation that could potentially be implemented by using the information retrieved from the objects in the IoT.

Finally, only 3 people stated that in their opinion, tasks might be too difficult to achieve with this application, against the 24 participants who said that they would carried out the actions without any issue. We asked to those three

Can you think of some ways to make the application's tasks more intuitive to carry out?

and we received the following answers:

1. "*If I hadn't watched the demo, I wouldn't know the meaning of source and target when creating a new rule. Maybe adding some virtual guide to the application, such as "?" button, would help the user to know better how to use the application.*"

2. "*There should be an easy way to show conflicts between rules that allows for debugging when things do not go as planned.*"

3. "*I suggest to include in the application some examples of usage and some suggestions on how to combine the appliances with each other, in order to help the user set the application and create new rules.*"

Again, conflicts between rules is a feature that needs to be implemented, not only as a background check by the application when creating dependencies between Things, but also in the form of a visual message or warning to inform the user of their options. It is in fact important that at any moment users are aware of what is happening in the application and why, in order to improve its usability and

accessibility. Similarly, answer 1 suggests a valid way to make the interface easier to learn and use, especially for users that are not used to achieve daily-life tasks through their phones (or any other device). As a matter of fact, some participants claimed that the demo showed when presenting the app was very helpful to convey the concepts explained theoretically at the beginning (e.g. meaning of rule between Things, how to alter a property of an objects, how to select multiple sources and targets, etc.). Hence, we could include it in the interface to be played on demand by the user whenever they need help or do not remember how to perform an action. Of course, the demo would be recorded with the final version of the working application, and maybe split into different pieces, each describing a specific task to achieve. In this way, the user would not be obliged to open a two-minute long video and skip through it to find the part they are interested in. That would satisfy answer 3, making sure that users feel constantly in control of the application by being assisted at any step.

7 Discussion and Future Work

In this chapter we discuss the results obtained during the validation process, and compare what we achieved with what we set as a goal at the beginning. Furthermore, we present some future work to fix and improve the application.

Even though what we implemented was only a proof of concept including the basic functionalities of the application, it represents a good starting point that (1) visually supports the goal we aimed to achieve in the first place, and (2) served as a concrete idea to some potential end users in order to know what they think about it and their first impressions. Unfortunately, our intention of adopting a technology that would enable users to run the application on the Web, no matter the device, was not satisfied because AR.js and WebXR are still in their early stages. This means that in the future, when they might become more stable and reliable and less limited in terms of features, the application presented in this thesis could be successfully implemented with a tool of augmented reality on the Web, such as AR.js. Having known this since the beginning, we could have immediately discarded this option and focused on the investigation of an alternative technology since the early stages of the process, such as Unity that we ended up adopting rather late. But this is research and sometimes it is necessary to invest some time into one route to eventually realise it led nowhere, in order to achieve the predefined goal. The results, whether positive or negative, can always be useful for future work to save some time and speed up the whole process.

As reported in Section 6.3, participants were overall satisfied and excited about the proposed interface, up to a point where they could easily imagine themselves using it in their current daily lives. In particular, it was important for us to receive some positive feedback from those with little familiarity of Computer Science and those in their middle age, who do not usually have a strong technological background. In fact, we aim to provide a tool that is not only accessible to expert users, as it is currently happening with the most recent applications such as Home Assistant (see Section 2.2.3), but to anyone who wishes to have the complete control of their home via an app. From the answers received and the individual interviews carried out with the participants, we were informed that they found the augmented reality side of the interface intuitive enough to be able to successfully use the application by themselves already at their first attempt. One of the participants revealed that

they had been using Home Assistant to control their home for a few months, and stated that they managed to use it because highly skilled at programming and connecting devices. "*If you do not have any knowledge of Arduino and writing code, you would not be able to set up your Home Assistant and customise the application. The graphical interface is particularly difficult to model because you need to use Python, HTML and JavaScript to get the job done*", they said, adding that "*this is why I think that an interface such as the one that you are proposing could simplify the life of many and could bridge the gap between automation systems and any kind of user*".

As some participants stated, when the smart objects and the arrows between them are largely present in the scene, the augmented reality environment could become convoluted and difficult to read and a list of devices with their rules might be a better solution. This is indeed another option offered by the interface, which we did not implemented in our prototype. The idea that drove us through this project is not to substitute the current home automation solutions offered to the public, but rather to enhance them with an AR interface and further functionalities.

Furthermore, we received some positive feedback about the interaction technique that we included in the application to create new rules, which is the drag-and-drop of a smart object to another. A participant aged between 41 and 60 years old affirmed "*selecting a TV and dragging it toward the light where I can drop it is actually a very intuitive way to add some logic between the two devices because the direction of the performed gesture indicates what is the source and what is the target in the new desired rule. It makes sense*".

What is to be taken care of (as reported by some participants in the open questions) is that the conflicts between rules need to be prevented. This means that when the user wants to create a new rule between specific smart objects or events and situations, and exploits some properties that would break the logic of other rules, the system should not allow this new rule to be defined. For instance, if the user is using the drag-and-drop feature to create a new rule, once the source object is selected, we could display the possible target destinations in the same red colour as they are always presented, and the prohibited destination in a more opaque red or in a completely different colour. In the event that the target object is available without breaking anything else, but some properties might contradict another rule, we should not give the user the possibility to select them, and maybe show a warning message stating that "*if you wish to select this property, you should edit or delete the rule between Thing X and Thing Y in order to be consistent with the implicit actions in place*".

Another improvement that needs to be realised in the graphical aspect of the interface is the way how arrows appear on the screen, which seems to bother more than one participant. Someone suggested a "help" button to be consulted in case the user has difficulties when trying to achieve a task. For instance, a "?" icon on one of the edges of the screen that once tapped would trigger a virtual guide tour of the interface, with short sentences for each button and functionality, might be enough to solve the problem. Some participants were not particularly enthusiastic about the overall appearance of the app (neutral values for the visual aesthetics property), suggesting some more stylish look and 3D models to represent smart objects. This is indeed an aspect that needs to be improved, but we knew since the beginning that what implemented would only be a simple prototype to visually show what we meant in the application's description, rather than the final product. On the other hand, more than one participant appreciated the minimal look of the app because "*sometimes making stuff more fancy could lead to confusion and difficulty in performing the actions*". One of them added "*the plain red images clearly convey the objects they represent in the real environment. My parents could easily use the application by themselves without my help for once!*". A final improvement on the visual aesthetics of the application that we learnt from the evaluation study might be to use smaller images as smart object placeholders. A few participants mentioned to use icons instead of images covering the entire size of the real smart appliance they refer to. This must be further investigated because most of the participants did not seem bothered by our current visual choice.

All in all, the feedback received was positive enough to suggest that we are moving toward the right direction. Even though, as explained above, there is still some work to be done to fix and increase some aspects of the application, the idea, the design and the functionalities all constitute a solid basis for the final product we wish to implement and distribute. In fact, as mentioned multiple times, the implementation presented to the participants had the only purpose of supporting what we described with words and aimed to achieve. It does not represent an accurate realisation of the complete application. Besides the improvements regarding the graphical side of the interface that emerged through the survey (e.g. stylish arrows, smaller images as placeholders, bigger buttons, virtual guide), the whole IoT aspect needs to be developed. The related work in Chapter 3 might significantly come in handy to exploit some existing tool to make the exchange of data between smart objects, and between smart objects and users, possible. Hardware such as Bluetooth and Wi-Fi is installed in all devices used nowadays, therefore it represents the easiest option to adopt when sending and receiving information. To take care of the implicit interactions working behind the scenes between the smart Things in the environment, the context modelling toolkit could be integrated in the implementation in order to handle the "IF condition THEN action" logic.

8 Conclusion

To the best of our knowledge, an augmented reality application that enables users to monitor and manipulate all smart objects in the real environment through both explicit and implicit interactions has not been proposed yet. For this reason, the gap between users and home automation is still rather wide, and the actions that can be performed consist mainly of voice commands (e.g. Amazon Alexa). By letting all different smart Things in the environment exchange data between each other and the user, and by offering users a tool to intuitively visualise where these devices are and how their properties can be controlled to connect them intelligently, not only we can enhance the features of the IoT, but we can also offer a tool that allows anyone to master their automation system. Driven by these necessities, the goal of this thesis was to design such an interface by exploiting the interaction techniques studied in the research of the sate of the art. Even though there is still some work to do in this direction, we could successfully address the problems identified at the beginning of this project, as well as the goals we set initially as contributions for the IoT field.

First of all, we provided some general guidelines concerning the interaction techniques to be adopted when using each of the different augmented reality displays. For instance, gestures are preferred when using head-mounted displays, whereas touch interactions are more effective with smartphones and tablets. Furthermore, we summarised the most efficient and straightforward ways to execute the universal operations, such as the selection and the drag-and-drop, with a specific AR device rather than the other. We also came up with a solution capable of enhancing the current state of the art in the field of home automation. In particular, we proposed a way to add logic and conditions between smart objects in the IoT in order to enable not only interactions carried out explicitly by humans, but also implicit interactions performed by the system behind the scenes by means of personalised rules. Finally, we designed our own interface that brings together the augmented reality and Internet of Things worlds, making a step further in the visualisation and control of automation systems. By proposing this new interface, we also designed new intuitive interaction techniques to make the application more accessible and easy to learn for people with disabilities or not much technological background. In particular, (1) by dragging the instance of a smart object in AR and dropping it on

another smart object, the user can easily create a rule between the two, (2) double tapping the source object and subsequently performing the same action on the target object represents another way to create a rule with these two smart Things involved, and (2) pointing the camera of the device in front of a Thing triggers an automatic selection of the smart object in question by the interface itself, without any explicit human action. While describing the interface and its functionalities, we also presented some scenarios in which the user can easily identify themselves in real life, and where the proposed application could come in handy.

The results obtained from interviewing some potential end users and analysing the answers received from the participants of our survey have been positive enough to conclude that the work done in this thesis constitutes an important addition to what is currently available for the public in the domains of the Internet of Things, human-computer interaction, automation systems and augmented reality. In particular, since a commercial graphical user interface in support of the IoT is currently missing, our solution could be beneficial for both novice and expert users who wish to have a practical tool to manage and secure their homes with just one click, voice command or gesture. The solution would in fact be accessible from any camera-equipped device chosen to complete the desired tasks.

Appendix A: Survey Questions

Beginning of survey

Mixed Reality-based Interaction for the Web of Things

Elena Zambon

Prof. Dr. Beat Signer

This survey has the purpose of validating the interface that we propose in our work, which will be presented as a Master's thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Applied Sciences and Engineering: Computer Science.

In order to protect your opinions, your answers are treated confidentially in accordance with the Belgian and European privacy legislation (Cf. AVG or GDPR). All your answers will be processed anonymously, so your identity is never revealed.

The application is presented to the participant prior to takingthis survey, with the support of a demonstration video to show its functionality in a concrete way. The video can be found here and you should watch the video before answering the questions of this survey.

The results of this survey will be collected and used in the report itself, in order to validate the research that has led to the design and implementation of this interface. In particular, we will investigate the following properties:

- attractiveness
- perspicuity
- novelty
- usefulness
- visual aesthetics
- intuitive use

The survey will take approximately 10 minutes of your time.

If you have any questions, please contact
elena.zambon@vub.be

Participant's information

Participant's personal data

Please select your gender

- ◯ Male
- ◯ Female
- ◯ Non-binary / third gender
- ◯ Prefer not to say

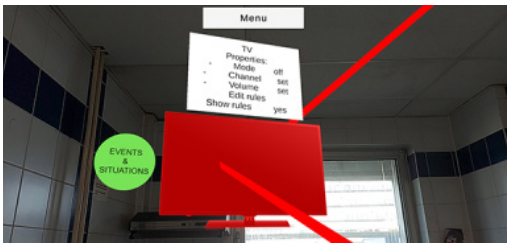
Please select your age

- ◯ Under 18
- ◯ 18 - 40
- ◯ 41 - 65
- ◯ Over 65

How familiar are you with the Computer Science domain?

- ◯ Extremely familiar
- ◯ Very familiar
- ◯ Moderately familiar
- ◯ Slightly familiar
- ◯ Not familiar at all

Attractiveness



In my opinion, the application is generally

annoying	○ ○ ○ ○ ○ ○ ○	enjoyable
bad	○ ○ ○ ○ ○ ○ ○	good
unpleasant	○ ○ ○ ○ ○ ○ ○	pleasant
unfriendly	○ ○ ○ ○ ○ ○ ○	friendly

I consider the *attractiveness* described by the above-mentioned terms as

completely irrelevant	○ ○ ○ ○ ○ ○ ○	very important

How would you make the application more attractive?

Perspicuity

In my opinion, handling and using the application is

not understandable	○ ○ ○ ○ ○ ○ ○	understandable
difficult to learn	○ ○ ○ ○ ○ ○ ○	easy to learn
complicated	○ ○ ○ ○ ○ ○ ○	easy
confusing	○ ○ ○ ○ ○ ○ ○	clear

I consider the *perspicuity* described by the above-mentioned terms as

completely irrelevant	○ ○ ○ ○ ○ ○ ○	very important

Novelty

In my opinion, the idea behind the application and its design are

dull	○ ○ ○ ○ ○ ○ ○	creative
conventional	○ ○ ○ ○ ○ ○ ○	inventive
usual	○ ○ ○ ○ ○ ○ ○	leading edge
conservative	○ ○ ○ ○ ○ ○ ○	innovative

I consider the *novelty* described by the above-mentioned terms as

completely irrelevant	○ ○ ○ ○ ○ ○ ○	very important

Usefulness

I consider the possibility of using the application as

useless	○ ○ ○ ○ ○ ○ ○	useful
not helpful	○ ○ ○ ○ ○ ○ ○	helpful
not beneficial	○ ○ ○ ○ ○ ○ ○	beneficial
not rewarding	○ ○ ○ ○ ○ ○ ○	rewarding

I consider the *usefulness* described by the above-mentioned terms as

completely irrelevant ○ ○ ○ ○ ○ ○ ○ very important

Do you think you would use this application in your daily life?

○ Yes
○ Maybe
○ No

Can you think of a situation in which this application would come in handy? When would it be useful for you to exploit the application?

Visual Aesthetics

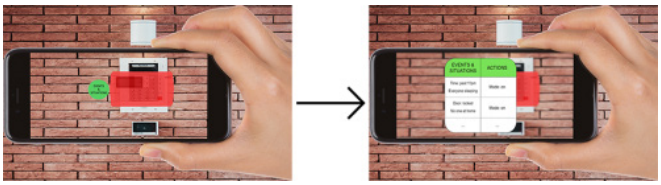
In my opinion, the visual design of the application is

ugly	○ ○ ○ ○ ○ ○ ○	beautiful
lacking style	○ ○ ○ ○ ○ ○ ○	stylish
unappealing	○ ○ ○ ○ ○ ○ ○	appealing
unpleasant	○ ○ ○ ○ ○ ○ ○	pleasant

I consider the *visual* aesthetics described by the above- mentioned terms as

completely irrelevant ○ ○ ○ ○ ○ ○ ○ very important

Intuitive Use



Novelty

In my opinion, using the application is

difficult	○ ○ ○ ○ ○ ○ ○	easy
illogical	○ ○ ○ ○ ○ ○ ○	logical
not plausible	○ ○ ○ ○ ○ ○ ○	plausible
inconclusive	○ ○ ○ ○ ○ ○ ○	conclusive

I consider the *intuitive* use described by the above-mentioned terms as

completely irrelevant	○ ○ ○ ○ ○ ○ ○	very important

Do you think the tasks are too difficult to achieve with this application?

- ○ Yes
- ○ Maybe
- ○ No

Can you think of some ways to make the application's tasks more intuitive to carry out?

Bibliography

[1] M. Berger. Resolving Occlusion in Augmented Reality: A Contour Based Approach Without 3D Reconstruction. In *Proceedings of IEEE CVPR 1997, Computer Society Conference on Computer Vision and Pattern Recognition*, Jun 1997.

[2] H. Shoaib and S. W. Jaffry. A Survey of Augmented Reality. In *Proceedings of ICVAR 2015, International Conference on Virtual and Augmented Reality*, Singapore, Singapore, Jan 2015.

[3] R. T. Azuma. A Survey of Augmented Reality. *Presence: Teleoperators and Virtual Environment*, 6(4), Aug 1997. doi: 10.1162/pres.1997.6.4.355.

[4] I. Sicaru, C. Ciocianu, and C. Boiangiu. A Survey on Augmented Reality. In *Proceedings of Journal of Information Systems & Operations Management*. Romanian-American University, Dec 2017.

[5] T. Andrade and D. Bastos. Extended Reality in IoT Scenarios: Concepts, Applications and Future Trends. In *Proceedings of exp.at 2019, 5th Experiment International Conference*, Funchal (Madeira Island), Portugal, Jun 2019. doi: 10.1109/EXPAT.2019.8876559.

[6] M. Ayaskanta, K. Sayan, B. Ankush, and D. Ankita. Design and Development of IoT-based Latency-optimized Augmented Reality Framework in Home Automation and Telemetry for Smart Lifestyle. *Journal of Reliable Intelligent Environments*, 6, Sep 2020. doi: 10.1007/s40860-020-00106-1.

[7] H. Lee and M. Kim. The Internet of Things in a Smart Connected World. In *Internet of Things - Technology, Applications and Standardization*, chapter 5. Springer, Aug 2018. doi: 10.5772/intechopen.76128.

[8] D. Jo and G. Kim. AR Enabled IoT for a Smart and Interactive Environment: A Survey and Future Directions. *Sensors*, 19(19), Oct 2019. doi: 10.3390/s19194330.

[9] P. Suresh, J. V. Daniel, V. Parthasarathy, and R. H. Aswathy. A State of the Art Review on the Internet of Things (IoT) History, Technology and Fields of Deployment. In *Proceedings of ICSEM 2014, International Conference on*

Science Engineering and Management Research, Chennai, India, Nov 2014. doi: 10.1109/ICSEMR.2014.7043637.

[10] R. Rädle, H.C. Jetter, N. Marquardt, H. Reiterer, and Y. Rogers. HuddleLamp: Spatially-Aware Mobile Displays for Ad-Hoc Around-the-Table Collaboration. In *Proceedings of ITS 2014, 9th ACM International Conference on Interactive Tabletops and Surfaces*, Dresden, Germany, Nov 2014. doi: 10.1145/2669485.2669500.

[11] M. Kim, S. Choi, K. Park, and J. Lee. User Interactions for Augmented Reality Smart Glasses: A Comparative Evaluation of Visual Contexts and Interaction Gestures. *Applied Sciences*, 9(15), Aug 2019. doi: 10.3390/app9153171.

[12] J. Zimmerman, E. Stolterman, and J. Forlizzi. An Analysis and Critique of Research through Design: Towards a Formalization of a Research Approach. In *Proceedings of DIS 2010, 8th ACM Conference on Designing Interactive Systems*, Aarhus, Denmark, Aug 2010. doi: 10.1145/1858171.1858228.

[13] I. E. Sutherland. A Head-Mounted Three Dimensional Display. In *Proceedings of AFIPS 1968, Fall Joint Computer Conference, Part I*, San Francisco, USA, Dec 1968. doi: 10.1145/1476589.1476686.

[14] E. Costanza, A. Kunz, and M. Fjeld. Mixed Reality: A Survey. *Lecture Notes in Computer Science*, 5440, Jan 2009. doi: 10.1007/978-3-642-00437-7_3.

[15] Speicher, M. and Hall, B. D. and Nebeling, M. What is Mixed Reality? In *Proceedings of CHI 2019, Conference on Human Factors in Computing Systems*, Glasgow, Scotland, UK, May 2019. doi: 10.1145/3290605.3300767.

[16] P. Milgram and F. Kishino. A Taxonomy of Mixed Reality Visual Displays. *IEICE Transactions Information Systems*, E77-D(12), Dec 1994.

[17] R. Azuma, Y. Baillot, R. Behringer, S. Feiner, S. Julier, and B. MacIntyre. Recent Advances in Augmented Reality. *IEEE Computer Graphics and Applications*, 21(6), Dec 2001. doi: 10.1109/38.963459.

[18] Z Pan, A. D. Cheok, H. Yang, J. Zhu, and J. Shi. Virtual Reality and Mixed Reality for Virtual Learning Environments. *Computers & Graphics*, 30(1), Feb 2006. doi: 10.1016/j.cag.2005.10.004.

[19] S. Liang. Research Proposal on Reviewing Augmented Reality Applications for Supporting Ageing Population. *Procedia Manufacturing*, 3, Dec 2015. doi: 10.1016/j.promfg.2015.07.132.

[20] Alan B. Craig. Chapter 3 - Augmented Reality Hardware. In *Understanding*

Augmented Reality. Morgan Kaufmann, Jun 2013. doi: 10.1016/B978-0-240-82408-6.00006-0.

[21] S. Liang and C.R. Roast. Five Features for Modeling Augmented Reality. In *Proceedings of HCI International 2014, Posters' Extended Abstracts*, Heraklion, Crete, Greece, Jun 2014. doi: 10.1007/978-3-319-07857-$1_1$07.

[22] S. Liang. Design Principles of Augmented Reality Focusing on the Ageing Population. In *Proceedings of HCI 2016, 30th International BCS Human Computer Interaction Conference: Fusion!*, Swindon, GBR, Jul 2016. doi: 10.14236/ewic/HCI2016.2.

[23] J. Ping, Y. Liu, and D. Weng. Comparison in Depth Perception between Virtual Reality and Augmented Reality Systems. In *Proceedings of IEEE VR 2019, 26th Conference on Virtual Reality and 3D User Interfaces*, Osaka, Japan, Mar 2019. doi: 10.1109/VR.2019.8798174.

[24] C. Diaz, M. Walker, D. A. Szafir, and D. Szafir. Designing for Depth Perceptions in Augmented Reality. In *Proceedings of IEEE ISMAR 2017, International Symposium on Mixed and Augmented Reality*, Nantes, France, Oct 2017. doi: 10.1109/ISMAR.2017.28.

[25] W. Xiangyu, M. J. Kim, P. Love, and S. Kang. Augmented Reality in Built Environment: Classification and Implications for Future Research. *Automation in Construction*, 32, Jul 2013. doi: 10.1016/j.autcon.2012.11.021.

[26] C. Koch and M. Neges and M. König and M. Abramovici. Natural markers for augmented reality-based indoor navigation and facility maintenance. *Automation in Construction*, 48, Dec 2014. doi: 10.1016/j.autcon.2014.08.009.

[27] A. Sharif, G. Zhai, J. Jia, X. Min, X. Zhu, and J. Zhang. An Accurate and Efficient 1D Barcode Detector for Medium of Deployment in IoT Systems. *IEEE Internet of Things Journal*, PP(99), Jul 2020. doi: 10.1109/JIOT.2020.3008931.

[28] P. Han and G. Zhao. A Review of Edge-based 3D Tracking of Rigid Objects. *Virtual Reality & Intelligent Hardware*, 1(6), Dec 2019. doi: 10.1016/j.vrih.2019.10.001.

[29] A. Duenser, R. Grasset, H. Seichter, and M. Billinghurst. Applying HCI principles to AR systems design. In *Proceedings of MRUI 2007, 2nd International Workshop at the IEEE Virtual Reality 2007 Conference*, Charlotte, USA, Jan 2007.

[30] H. Ishii and B. Ullmer. Tangible Bits: Towards Seamless Interfaces between

People, Bits and Atoms. In *Proceedings of CHI 1997, ACM SIGCHI Conference on Human Factors in Computing Systems*, Atlanta, USA, Mar 1997. doi: 10.1145/258549.258715.

[31] Development of a Benchmarking Scenario for Testing 3D User Interface Devices and Interaction Methods. In *Proceedings of HCI International 2005, 11th International Conference on Human Computer Interaction, author = Rizzo, A. and Kim, G. J. and Yeh, S. and Thiebaux, M. and Hwang, J. and Buckwalter, J.*, Las Vegas, NV, Jul 2005.

[32] C. Dede, M. C. Salzman, and R. Bowen Loftin. ScienceSpace: Virtual Realities for Learning Complex and Abstract Scientific Concepts. In *Proceedings of VRAIS 1996, Virtual Reality Annual International Symposium*, NW Washington, USA, Mar 1996.

[33] H. Kaufmann. *Geometry Education with Augmented Reality*. PhD thesis, Institut für Softwartechnik und Interaktive Systeme, Mar 2004. URL `http://publik.tuwien.ac.at/files/PubDat_138490.pdf`.

[34] J. D. Hincapié-Ramos, X. Guo, P. Moghadasian, and P. Irani. Consumed Endurance: A Metric to Quantify Arm Fatigue of Mid-Air Interactions. In *Proceedings of CHI 2014, SIGCHI Conference on Human Factors in Computing Systems*, Toronto, Ontario, Canada, Apr 2014. doi: 10.1145/2556288.2557130.

[35] C. Rolim and D. Schmalstieg and D. Kalkofen and V. Teichrieb. Design Guidelines for Generating Augmented Reality Instructions. In *Proceedings of IEEE ISMAR 2015, International Symposium on Mixed and Augmented Reality, year=2015, month=Nov, doi=DOI 10.1109/ISMAR.2015.36, address=Fukuoka, Japan.*

[36] J. Gimeno, S. Casas, C. Portalés, and M. Fernández. Addressing the Occlusion Problem in Augmented Reality Environments with Phantom Hollow Objects. In *Proceedings of IEEE (ISMAR-Adjunct) 2018, International Symposium on Mixed and Augmented Reality Adjunct*, Munich, Germany, Apr 2018. doi: 10.1109/ISMAR-Adjunct.2018.00024.

[37] O. Dabor, E. Longford, and S. Walker. Design Guidelines for Augmented Reality User Interface: A Case Study of Simultaneous Interpretation. In *Proceedings of CEEC 2019, 11th Computer Science and Electronic Engineering*, Sep 2019.

[38] S. Rokhsaritalemi, A. Sadeghi-Niaraki, and S. Choi. A Review on Mixed Reality: Current Trends, Challenges and Prospects. *Applied Sciences*, 10(2), Jan 2020. doi: 10.3390/app10020636.

[39] I. Potemin, A. Zhdanov, Nikolay N. Bogdanov, D. D. Zhdanov, I. Livshits, and Y. Wang. Analysis of the visual perception conflicts in designing mixed reality systems. In *Proceedings of SPIE 2018, Optical Design and Testing VIII*, Beijing, China, Oct 2018. doi: 10.1117/12.2503397.

[40] S. Debernardis, M. Fiorentino, M. Gattullo, G. Monno, and A. Uva. Text Readability in Head-Worn Displays: Color and Style Optimization in Video versus Optical See-Through Devices. *IEEE Transactions on Visualization and Computer Graphics*, 20(1), May 2014. doi: 10.1109/TVCG.2013.86.

[41] J. Carmigniani and B. Furht and M. Anisetti and P. Ceravolo and E. Damiani and M. Ivkovic. Augmented Reality Technologies, Systems and Applications. *Multimedia Tools and Applications*, 51, Dec 2010. doi: 10.1007/s11042-010-0660-6.

[42] J. Rolland, R. Holloway, and H. Fuchs. Comparison of Optical and Video See-Through, Head-Mounted Displays. In *Proceedings of SPIE 1994, The International Society for Optical Engineering*, Jan 1994. doi: 10.1117/12.197322.

[43] D. Schmalstieg and D. Wagner. Experiences with Handheld Augmented Reality. In *Proceedings of IEEE ISMAR 2007, 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, NW Washington, USA, Nov 2007. doi: 10.1109/ISMAR.2007.4538819.

[44] D. Wagner and D. Schmalstieg. Handheld Augmented Reality Displays. In *Proceedings of IEEE VR 2006, Virtual Reality Conference*, Alexandria, USA, Mar 2006. doi: 10.1109/VR.2006.67.

[45] A. Henrysson, M. Billinghurst, and M. Ollila. Virtual Object Manipulation Using a Mobile Phone. In *Proceedings of ICAT 2005, International Conference on Augmented Tele-Existence*, Christchurch, New Zealand, Dec 2005. doi: 10.1145/1152399.1152430.

[46] O. Bimber and R. Raskar. *Spatial Augmented Reality Merging Real and Virtual Worlds*. A. K. Peters LTD, Aug 2005. doi: 10.1201/b10624.

[47] Jon Peddie. Types of Augmented Reality. In *Augmented Reality : Where We Will All Live*. Springer International Publishing, Apr 2017. doi: 10.1007/978-3-319-54502-8_2.

[48] J. Chen, L. Mi, C. Chen, H. Liu, J. Jiang, W. Zhang, and Y. Liu. A Foveated Contact Lens Display for Augmented Reality. In *Proceedings of SPIE, Optical Architectures for Displays and Sensing in Augmented, Virtual, and Mixed Reality (AR, VR, MR)*, San Francisco, USA, Feb 2020. doi: 10.1117/12.2545972.

[49] E. Ahmed, I. Yaqoob, A. Gani, M. Imran, and M. Guizani. Internet-of-things-based smart environments: state of the art, taxonomy, and open research challenges. *IEEE Wireless Communications*, 23(5), Nov 2016. doi: 10.1109/MWC.2016.7721736.

[50] I. Lee and K. Lee. The Internet of Things (IoT): Applications, Investments, and Challenges for Enterprises. *Business Horizons*, 58(4), Apr 2015. doi: 10.1016/j.bushor.2015.03.008.

[51] K. Ashton. That 'Internet of Things' Thing. *RFID Journal*, Jun 2009.

[52] A. Rayes and S. Salam. *Internet of Things From Hype to Reality - The Road to Digitization*. Springer, Nov 2016. doi: 10.1007/978-3-319-99516-8.

[53] A. M. Al-Ghaili, F. Abdul Rahim, F. Azman, and H. Kasim. Efficient Implementation of 2D Barcode Verification Algorithm for IoT Applications. In *Proceedings of IEEE BigDataSecurity 2019, 5th Intl Conference on Big Data Security on Cloud, IEEE HPSC, Intl Conference on High Performance and Smart Computing, and IEEE IDS, Intl Conference on Intelligent Data and Security*, Washington, USA, May 2019. doi: 10.1109/BigDataSecurity-HPSC-IDS.2019.00059.

[54] V. Deep and T. Elarabi. Efficient IEEE 802.15.4 ZigBee standard hardware design for IoT applications. In *Proceedings of ICSigSys 2017, International Conference on Signals and Systems*, Sanur, Indonesia, May 2017. doi: 10.1109/ICSIGSYS.2017.7967053.

[55] Y. Zhou, X. Guo, M. Zhou, and L. Wang. A Design of Greenhouse Monitoring Control System Based on ZigBee Wireless Sensor Network. In *Proceedings of 2007, International Conference on Wireless Communications, Networking and Mobile Computing*, Sep 2007. doi: 10.1109/WICOM.2007.638.

[56] Wikipedia. *RFID Key Fobs*. `http://www.surerfid.com/rfid-key-fobs.html`, 2020. Accessed: 22 Nov 2020.

[57] Digi. *Digi XBee Zigbee*. `https://www.digi.com/products/embedded-systems/digi-xbee/rf-modules/2-4-ghz-rf-modules/xbee-zigbee`, 2020. Accessed: 22 Nov 2020.

[58] R. A. Bolt. "Put-That-There": Voice and Gesture at the Graphics Interface. In *Proceedings of SIGGRAPH 1980, 7th Annual Conference on Computer Graphics and Interactive Techniques*, Seattle, USA, Jul 1980. doi: 10.1145/800250.807503.

[59] J. Rekimoto. Pick-and-Drop: A Direct Manipulation Technique for Multi-

ple Computer Environments. In *Proceedings of UIST 1997, 10th Annual ACM Symposium on User Interface Software and Technology*, Banff, Alberta, Canada, Oct 1997. doi: 10.1145/263407.263505.

[60] J. Rekimoto and M. Saitoh. Augmented Surfaces: A Spatially Continuous Work Space for Hybrid Computing Environments. In *Proceedings of CHI 1999, SIGCHI Conference on Human Factors in Computing Systems*, Pittsburgh, USA, May 1999. doi: 10.1145/302979.303113.

[61] N. Marquardt, T. Ballendat, S. Boring, S. Greenberg, and K. Hinckley. Gradual Engagement: Facilitating Information Exchange between Digital Devices as a Function of Proximity. In *Proceedings of ITS 2012, ACM Conference on Interactive Tabletops and Surfaces*, Cambridge, USA, Nov 2012. doi: 10.1145/2396636.2396642.

[62] J. Espada, V. García Díaz, R. Gonzalez Crespo, O. Sanjuán, B. Pelayo García-Bustelo, and J. Cueva Lovelle. Mobile Web-Based System for Remote-Controlled Electronic Devices and Smart Objects. *Mobile Networks and Applications*, 19, Jul 2014. doi: 10.1007/s11036-014-0510-2.

[63] R. Roels, A. Witte, and B. Signer. INFEX: A Unifying Framework for Cross-Device Information Exploration and Exchange. *Human-Computer Interaction*, 2(2), Jan 2017. doi: 10.1145/3179427.

[64] Yang, Z. and Nakajima, T. Connecting Smart Objects in IoT Architectures by Screen Remote Monitoring and Control. *Computers*, 7(4), Sep 2018. doi: 10.3390/computers7040047.

[65] T. Zachariah and P. Dutta. Browsing the Web of Things in Mobile Augmented Reality. In *Proceedings of HotMobile 2019, 20th International Workshop on Mobile Computing Systems and Applications*, Santa Cruz, USA, Feb 2019. Association for Computing Machinery. doi: 10.1145/3301293.3302359.

[66] J. D. H. Bezerra and C. T. de Souza. SmAR2t: A Models at Runtime Architecture to Interact with the Web Of Things Using Augmented Reality. In *Proceedings of SBES 2019, XXXIII Brazilian Symposium on Software Engineering*, Salvador, Brazil, Sep 2019. doi: 10.1145/3350768.3353818.

[67] W3C Recommendation. *Web of Things (WoT) Thing Description*. `https://www.w3.org/TR/wot-thing-description/`, 2020. [Accessed: 20 Nov 2020].

[68] S. K. Vishwakarma, P. Upadhyaya, B. Kumari, and A. K. Mishra. Smart Energy Efficient Home Automation System Using IoT. In *Proceedings of IoT-SIU 2019, 4th International Conference on Internet of Things: Smart*

Innovation and Usages, Ghaziabad, India, Apr 2019. doi: 10.1109/IoT-SIU.2019.8777607.

[69] S. Trullemans, L. Van Holsbeeke, and B. Signer. The Context Modelling Toolkit: A Unified Multi-Layered Context Modelling Approach. *Proceedings of the ACM on Human-Computer Interaction*, 1(7), Jun 2017. doi: 10.1145/3095810.

[70] B. Furht and J. Carmigniani. Chapter 1 - Augmented Reality: An Overview. In *Handbook of Augmented Reality*. Springer New York Dordrecht Heidelberg London, Jan 2011. doi: 10.1007/978-1-4614-0064-6.

[71] M. Whitlock, E. Harnner, J. R. Brubaker, S. Kane, and D. A. Szafir. Interacting with Distant Objects in Augmented Reality. In *Proceedings of IEEE VR 2018, Conference on Virtual Reality and 3D User Interfaces*, Reutlingen, Germany, Mar 2018. doi: 10.1109/VR.2018.8446381.

[72] P. Song, W. Boon Goh, W. Hutama, C. Fu, and X. Liu. A Handle Bar Metaphor for Virtual Object Manipulation with Mid-Air Interaction. In *Proceedings of CHI 2012, SIGCHI Conference on Human Factors in Computing Systems*, Austin, USA, May 2012. doi: 10.1145/2207676.2208585.

[73] D. Mendes, F. Fonseca, B. Araùjo, A. Ferreira, and J. Jorge. Mid-air interactions above stereoscopic interactive tables. In *Proceedings of IEEE 3DUI 2014, Symposium on 3D User Interfaces*, Minneapolis, USA, Mar 2014. doi: 10.1109/3DUI.2014.6798833.

[74] J. Blattgerste, P. Renner, and T. Pfeiffer. Advantages of Eye-Gaze over Head-Gaze-Based Selection in Virtual and Augmented Reality under Varying Field of Views. In *Proceedings of COGAIN 2018, Workshop on Communication by Gaze Interaction*, Warsaw, Poland, Jun 2018. doi: 10.1145/3206343.3206349.

[75] C. Elmadjian, P. Shukla, A. Diaz Tula, and C. H. Morimoto. 3D Gaze Estimation in the Scene Volume with a Head-Mounted Eye Tracker. In *Proceedings of COGAIN 2018, Workshop on Communication by Gaze Interaction*, Warsaw, Poland, Jun 2018. doi: 10.1145/3206343.3206351.

[76] Z. He and X. Yang. Hand-Based Interaction for Object Manipulation with Augmented Reality Glasses. In *Proceedings of VRCAI 2014, 13th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry*, Shenzhen, China, Nov 2014. doi: 10.1145/2670473.2670505.

[77] M. Billinghurst, H. Kato, and S. Myojin. Advanced Interaction Techniques for Augmented Reality Applications. In *Proceedings of VMR 2009, Virtual*

and Mixed Reality, Third International Conference, San Diego, USA, Jul 2009. doi: 10.1007/978-3-642-02771-0_2.

[78] E. S. Goh, M. S. Sunar, and A. W. Ismail. 3D Object Manipulation Techniques in Handheld Mobile Augmented Reality Interface: A Review. *IEEE Access*, 7, Mar 2019. doi: 10.1109/ACCESS.2019.2906394.

[79] R. Brouet, R. Blanch, and M. Cani. Understanding Hand Degrees of Freedom and Natural Gestures for 3D Interaction on Tabletop. In *Proceedings of INTERACT 1990, IFIP Conference on Human-Computer Interaction*, Cape Town, South Africa, Sep 2013. doi: 10.1007/978-3-642-40483-2_20.

[80] A. Mossel, B. Venditti, and H. Kaufmann. 3DTouch and HOMER-S: Intuitive Manipulation Techniques for One-Handed Handheld Augmented Reality. In *Proceedings of VRIC 2013, Virtual Reality International Conference: Laval Virtual*, Laval, France, Mar 2013. doi: 10.1145/2466816.2466829.

[81] G. A. Lee, U. Yang, Y. Kim, D. Jo, K. Kim, J. H. Kim, and Jin S. Choi. Freeze-Set-Go Interaction Method for Handheld Mobile Augmented Reality Environments. In *Proceedings of VRSR 2009, 16th ACM Symposium on Virtual Reality Software and Technology*, Kyoto, Japan, Nov 2009. doi: 10.1145/1643928.1643961.

[82] H. Bai, G. A. Lee, and M. Billinghurst. Freeze View Touch and Finger Gesture Based Interaction Methods for Handheld Augmented Reality Interfaces. In *Proceedings of IVCNZ 2012, 27th Conference on Image and Vision Computing New Zealand*, Dunedin, New Zealand, Nov 2012. doi: 10.1145/2425836.2425864.

[83] A. Dey, M. Billinghurst, R. Lindeman, and J. Swan. A Systematic Review of 10 Years of Augmented Reality Usability Studies: 2005 to 2014. *Frontiers in Robotics and AI*, 5, Apr 2018. doi: 10.3389/frobt.2018.00037.

[84] H. Bai, G. A. Lee, M. Ramakrishnan, and M. Billinghurst. 3D Gesture Interaction for Handheld Augmented Reality. In *Proceedings of SIGGRAPH Asia 2014, Mobile Graphics and Interactive Applications*, Shenzhen, China, Nov 2014. doi: 10.1145/2669062.2669073.

[85] W. H. Chun and T. Höllerer. Real-Time Hand Interaction for Augmented Reality on Mobile Phones. In *Proceedings of IUI 2013, International Conference on Intelligent User Interfaces*, Santa Monica, USA, Mar 2013. doi: 10.1145/2449396.2449435.

[86] V. Nanjappan, R. Shi, H. Liang, H. Xiao, K. Lau, and K. Hasan. Design of Interactions for Handheld Augmented Reality Devices Using Wearable Smart

Textiles: Findings from a User Elicitation Study. *Applied Sciences*, 9(15), Aug 2019. doi: 10.3390/app9153177.

[87] A. Marzo, B. Bossavit, and M. Hachet. Combining Multi-Touch Input and Device Movement for 3D Manipulations in Mobile Augmented Reality Environments. In *Proceedings of SUI 2014, 2nd ACM Symposium on Spatial User Interaction*, Honolulu, USA, Oct 2014. doi: 10.1145/2659766.2659775.

[88] I. Kondratova. Multimodal Interaction for Mobile Learning. In *Proceedings of UAHCI 2009, Universal Access in Human-Computer Interaction. Intelligent and Ubiquitous Interaction Environments*, San Diego, USA, Jul 2009. doi: 10.1007/978-3-642-02710-9$_3$6.

[89] R. Pascoal, R. Ribeiro, F. Batista, and A. de Almeida. Adapting Speech Recognition in Augmented Reality for Mobile Devices in Outdoor Environments. In *Proceedings of SLATE 2017, 6th Symposium on Languages, Applications and Technologies*, volume 56, Jun 2017. doi: 10.4230/OASIcs.SLATE.2017.21.

[90] R. Raskar and K. Low. Interacting with spatially augmented reality.

[91] J. Sol Roo and M. Hachet. Interacting with Spatial Augmented Reality. working paper or preprint, Mar 2016. URL `https://hal.archives-ouvertes.fr/hal-01284005`.

[92] F. Zhou, H. Duh, and M. Billinghurst. Trends in augmented reality tracking, interaction and display: A review of ten years of ISMAR. In *Proceedings of IEEE/ACM ISMAR 2008, 7th International Symposium on Mixed and Augmented Reality*, Cambridge, UK, Sep 2008. doi: 10.1109/ismar.2008.4637362.

[93] B. Thomas, M. Marner, R. Smith, N. Sayed, S. Itzstein, K. Klein, M. Adcock, P. Eades, A. Irlitti, J. Zucco, T. Simon, J. Baumeister, and T. Suthers. Spatial Augmented Reality - A Tool for 3D Data Visualization. In *Proceedings of IEEE VIS 2014, International Workshop on 3DVis (3DVis)*, Paris, France, Jul 2015. doi: 10.1109/3DVis.2014.7160099.

[94] M. Elepfandt and M. Sünderhauf. Multimodal, Touchless Interaction in Spatial Augmented Reality Environments. In *Proceedings of ICDHM 2011, Third International Conference of Digital Human Modeling*, Orlando, USA, Jul 2011. doi: 10.1007/978-3-642-21799-9$_3$0.

[95] R. J. K. Jacob. The Use of Eye Movements in Human-Computer Interaction Techniques: What You Look at is What You Get. *ACM Transactions on Information Systems*, 9(2), Apr 1991. doi: 10.1145/123078.128728.

[96] G. L. Martin. The Utility of Speech Input in User-Computer Interfaces. *Inter-*

national Journal Man-Machine Studies, 30(4), Apr 1989. doi: 10.1016/S0020-7373(89)80023-9.

[97] C. Jetter, J. Gerken, and Harald Reiterer. Natural User Interfaces: Why We Need Better Model-Worlds, Not Better Gestures. In *Proceedings of CHI 2010 Workshop, Natural User Interfaces: The Prospect and Challenge of Touch and Gestural Computing*, Atlanta, USA, Apr 2010.

[98] A. B. Craig. *Understanding Augmented Reality: Concepts And Applications*. Morgan Kaufmann, Jun 2013. doi: 10.1016/B978-0-240-82408-6.00001-1.

[99] W. R. Sherman and A. B. Craig. Chapter 7 - Interacting With the Virtual World. In *Understanding Virtual Reality (Second Edition)*. Morgan Kaufmann, second edition, Dec 2018. doi: https://doi.org/10.1016/B978-0-12-800965-9.00007-6.

[100] A. B. Craig. Chapter 6 - Interaction in Augmented Reality. In *Understanding Augmented Reality*. Morgan Kaufmann, Jun 2013. doi: 10.1016/B978-0-240-82408-6.00006-0.

[101] M. R. Mine. Virtual Environment Interaction Techniques. Technical report, May 1995.

[102] I. Poupyrev, M. Billinghurst, S. Weghorst, and T. Ichikawa. The Go-Go Interaction Technique: Non-Linear Mapping for Direct Manipulation in VR. In *Proceedings of UIST 1996, 9th Annual ACM Symposium on User Interface Software and Technology*, Seattle, USA, Nov 1996. doi: 10.1145/237091.237102.

[103] T. Piumsomboon, A. Clark, M. Billinghurst, and A. Cockburn. User-Defined Gestures for Augmented Reality. In *Proceedings of CHI EA 2013, Extended Abstracts on Human Factors in Computing Systems*, Paris, France, Apr 2013. doi: 10.1145/2468356.2468527.

[104] E. Jerome and N. Carpignoli. *AR.js - Augmented Reality on the Web*. `https://ar-js-org.github.io/AR.js-Docs/`, 2020. [Accessed: 1 Mar 2021].