School of Electronic Engineering

Integrated wireless control of multiple olfaction dispensers in Digital Twinning for improved VR experience

Project Portfolio

Marouane Ben Zineb
Student ID: 23266913

August 2024

MEng in Electronic & Computer Engineering

Supervised by Gabriel-Miro Muntean

ACKNOWLEDGMENTS

I would like to express my warmest gratitude and appreciation to my supervisor Prof. Gabriel Miro-Muntean as well as Dr. Anderson Augusto Simiscuka for their precious help and advice all throughout this project's duration. A special thanks to the School of Electronic Engineering at Dublin City University for providing the necessary resources, and the participants who gladly volunteered for the user testing experience.

CONTENTS

Integrated wireless control of multiple olfaction dispensers in Digital Twinning for improved VR experience

Marouane Ben Zineb
School of Electronic Engineering, Dublin City University, Dublin, Ireland

***Abstract*—This study explores integrating wireless olfactive dispensers into a Unity-based digital twin environment to enhance user experience in a 6 Degrees of Freedom (DoF) Virtual Reality (VR) setting. Using a Meta Quest 2 headset and "Inhalio" dispensers (smell diffusers), synchronized scent triggers were implemented based on user location and interaction, monitored by Unity C# scripts. The approach synchronizes the real-world devices with their virtual counterparts (assets), triggered by user interactions and proximity within a VR laboratory. Cloud-based (Adafruit IO servers) and local-based (WiFi) network synchronization methods were compared. Perceptual user testing assessed user preferences, scent quality, visual display, and overall immersion. Results indicate that the olfaction cue integration significantly enhances user experience, with proximity-based triggers preferred and high immersion levels reported. A technical analysis concluded with a clear performance differences between the tested scenarios.**

***Index Terms*—6 DoF Virtual Reality, Digital Twin, Olfaction, Meta Quest 2, Adafruit IO, Synchronization, MQTT, Unity.**

I. Introduction

VIRTUAL REALITY (VR) has known a rapid development over the last years, but the progress quickly started slowing down in terms of pushing the boundaries of immersion and realism. Although VR engages a large panel of senses (sight, hearing and touch), the multi-sensory spectrum remains under-explored. One particular sense, olfaction or commonly known as smell, plays a crucial role in daily life, and although many studies attempt to integrate it in the VR experience, this innovation continues to face obstacles. This project aims to address this challenge by developing a solution that integrates wireless olfaction dispensers within a Unity-based digital twin environment which will be a virtual replica of the DCU laboratory (S220 in the Stokes Building). The solution will compare three synchronization techniques with the aim of identifying the most efficient one. The first one is a local network-based approach that will establish a direct communication between the dispensers and the VR headset. The second and third methods are cloud-based and rely on the Adafruit IO service as a data intermediary. The two cloud-based approaches will differ in the communication protocol used, where one uses HTTP while the other uses MQTT.

II. Related work

A. Existing Olfaction devices types

Integrating the olfaction cue in VR can have multiple forms. For instance, Nakamoto *et al.* rely on a wearable device that can be attached to the headset. This method uses the Surface Acoustic Wave (SAW) technology which reduces the issue of smell persistence. But this version has some constraints as it is hard to dispense many scents at the same time without them mixing together. Moreover, the results of the SAW technology turned out to be unsatisfactory and smell persistence still hindered the user's experience. [1].

A better solution proposed by Liu *et al.* is the usage of miniaturized odor generators in the form of adhesive skin patches that can be placed directly on the user. The small devices rely on a small electrical module that will take care of the data transfers between the simulation from the headset and the resistors that will heat the paraffin wax encapsulating the scent. This solution is very portable and economical but can create discomforts for the user if the heating exceeds a certain temperature threshold [2].

The solution chosen for the study is the one used by Simiscuka *et al.* where the olfaction source is stationary in the setting rather than mobile with the user. The static dispensers provide a better scent distribution in the room and avoid the challenges associated with mixing scents and managing smell persistence. By placing multiple dispensers around the environment and twinning them with virtual counterparts in the virtual world, it is possible to create a more immersive and controlled olfactory experience [3].

B. Digital twinning

Digital twinning in the context of Virtual Reality is the process of creating a virtual replica of a physical object or environment. The aim is to mirror the real-world counterparts with high precision to ensure a certain immersion of the user. Furthermore, it enables the dynamic interaction between the physical and virtual worlds through a real-time synchronization. The best tool to carry out such a task is the Unity game engine which provides a wide range of high-quality realistic 3D assets [4].

C. Communication protocols & Synchronization methods

In an attempt to establish a real-virtual IoT device synchronization in VR, Simiscuka *et al.* rely on the Hypertext Transfer Protocol (HTTP) and the Queuing Telemtry Protocol (MQTT), the most widely used communication protocols. HTTP is a request-response protocol and can introduce latency due to its nature of requiring a complete handshake for each data

transfer. MQTT is popular in the IoT world for its lightweight aspect and a publish-subscribe scheme that is optimized for low-bandwidth devices which makes it the ideal choice for handling large volumes of small messages. In the study, Simiscuka *et al.* compared two different approaches: a local-based method and a cloud-based method using the Adafruit IO service. Performance testing showed lower latency in the local network approach compared to the cloud-based approach and for the cloud-based approach, MQTT showed a better performance than REST API in terms of delay and data traffic [5].

III. Design & Implementation

A. Overview

The following section will describe in details the hardware and software components as well as some important concepts that are crucial for the implementation of the solution.

B. Synchronization approaches

One local network-based approach and two cloud-based approaches were tested in the study. The general architecture of the systems are outlined in Figure 1.

1) Local network-based method:
The first synchronization method involves establishing a direct communication link between the VR headset and the olfaction dispensers using a local WiFi network. The headset sends straightforwardly activation orders to the device whenever a listener is triggered using the GET method in the form of a URL link: http://192.168.0.102/diffuse?duration=1&intensity=high.

2) Cloud-based methods:
HTTP-Based Synchronization: In the first cloud-based method, HTTP is used as the communication protocol to send commands from the VR headset to the olfaction dispensers via the Adafruit IO service using HTTP POST requests. The data (The device IP address, the duration and intensity) is sent to the designated Adafruit IO feed in the JSON form. An Adafruit listener Python script running on a local computer, periodically checks for updates by sending HTTP GET requests to the Adafruit feed. When new data is detected, the script transmits commands to the dispenser through HTTP GET requests.

MQTT-Based Synchronization: The second cloud-based method has the same architecture as the first one but utilizes MQTT instead. In this scenario, the VR headset publishes messages to the Adafruit MQTT broker hosted on the Adafruit IO platform. Each message contains the necessary parameters and is published to the designated topic. Another Adafruit listener script that is running on a local computer, is connected to the MQTT broker and is subscribed to the topic, continuously checking for new messages. Upon receiving new data, the script orders the corresponding olfaction dispenser to activate using HTTP GET requests.

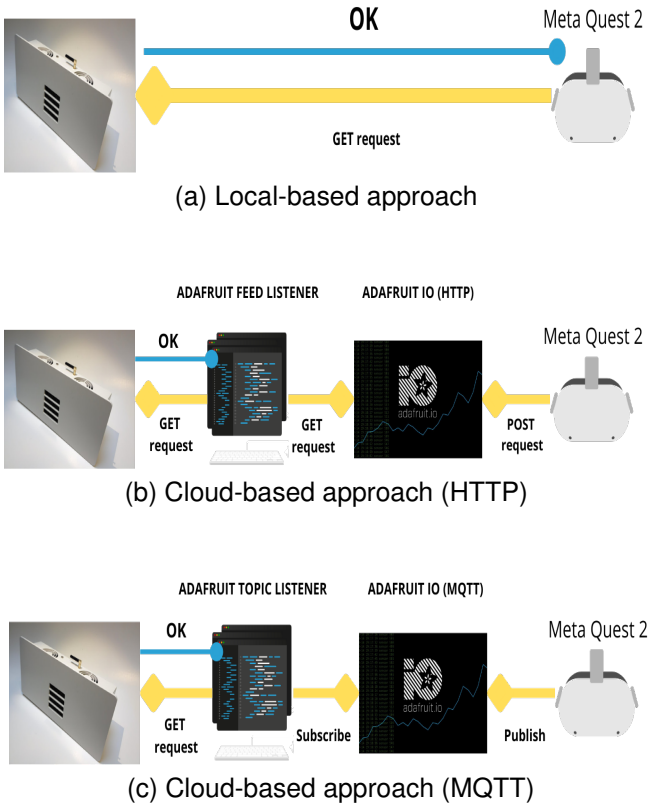


Fig. 1. Synchronization methods schematic

C. Components & Hardware

1) Olfaction dispensers:
The smell diffusion is managed by scent dispensers designed and manufactured by “Inhalio”, a company that specializes in digital scent systems. The device as seen in Figure 2, relies on a dry-air diffusion technology where scents are encapsulated in paraffin wax beads and propelled through fans. The air leaving from the device carries the scent from the beads. Each dispenser is equipped with its own integrated network adapter granting it WiFi connectivity and a framework with an integrated RESTful API which makes it controllable through HTTP requests (GET method). The API gives control over parameters including duration and intensity. The scents used for the study are: Cookie, Perfume and Apple [6].

Fig. 2. “Inhalio” dry-air scent dispenser.

2) VR headset:
The choice of the VR headset has a direct impact on the quality of the overall experience in terms of immersion and realism. The Meta Quest 2 headset provides a balance between performance, affordability and quality. A resolution of 1832 x 1920 pixels Per Eye and a LCD display with a 90 Hz refresh rate contribute to reducing motion sickness and improving visual clarity. Moreover, the standalone (Wireless) aspect of the headset adds a note of authenticity where the user can move around his surroundings without having to worry about cable shortness. A 3D positional audio system is also built directly into the headset, contributing to the multi-sensory experience. However, the headset is limited by its processing power as it is not connected to an external unity. Therefore, the challenge is to create a Unity digital twin scenario of the laboratory that does not overload the headset's capacities [7].

3) WiFi router:
The WiFi router ensures the transmission of the orders to the dispensers. Therefore, the quality of the router has a direct correlation with the performance of the scent distribution and by extent the immersion of the users. The router used for this study is the TP-Link AC1200 WiFi Router with a frequency of 5GHz (Also works at 2.4 GHz) and a maximum speed of 867 Mbps. It is compatible with the following WiFi standards: 802.11n, 802.11b, 802.11ac [8].

D. Digital Twinning

The digital twinning of the laboratory environment involves creating a detailed and accurate 1:1 scale virtual replica of the physical room using the Unity game engine. By comparing the original laboratory and its VR digital twin in Figure 3, it is evident that they share the exact same measurements and closely resemble each other in appearance. The primary difference is the removal of the chairs in the virtual version, a decision made to optimize the space and facilitate the navigation of the users in the room.

The olfaction dispensers are also digitally twinned, with each one linked to a virtual counterpart asset that will represent a specific scent. In this study, the assets used were a cookie, an apple and a perfume bottle, each twinned to a dispenser that will hold their respective scent. The cookie and apple trigger the scent when the user approaches them (trigger by distance), and the perfume is triggered when the user interacts with it using the VR hand controller. In Figure 4, the large dispenser next to the desktop computer is replaced by a small cookie in the virtual world. (See Appendix L)

(a) Physical laboratory room

(b) Digital twin of the physical laboratory room

Fig. 3. VR Laboratory scene (Room S220)

(a) Cookie scent dispenser (b) VR cookie counterpart

Fig. 4. Digital twinning of the olfaction dispensers

E. Software Architecture

1) Unity:
Unity is the most popular game engine used to develop 2D, 3D and VR/AR application. It is built on the C# programming language, enabling developers to script complex interactions and behaviors within virtual environments. Unity also provides a native toolkit called OpenXR offering compatibility with many VR headsets including the Meta Quest 2 headset which is used for this study. Unity provides a great amount of external packages and a digital marketplace called the Unity Asset Store that offers a large collection of assets and templates.

2) Adafruit IO (Cloud-based methods only):
Adafruit IO is a cloud-based service provided by the company Adafruit Industries. Users can send, receive, and monitor data from connected devices through a user-friendly web interface or via an API. It supports popular IoT communication protocols such as MQTT (Message Queuing Telemetry Transport) and HTTP. Data streams received from connected

devices are stored in feeds.

3) Integrated C# scripts:
For each scenario, a VR asset that will dispense a scent must have an instance of the following classes: *OlfactionObject.cs* and *DatanSender.cs* (See Appendix I). Although the implementation differs between the synchronization methods, the structure is the same.

- **OlfactionObject.cs**: This class acts as a listener and detects when a player interacts or approaches the object. It then initiates an activation request for the corresponding dispenser from the datasender instance.
- **DatanSender.cs**: This class handles the communications either directly between the headset and the dispensers (for the local-based method) or between the headset and the Adafruit IO service. In the HTTP case, It sends the function 'UnityWebRequest' to either send a JSON payload to the Adafruit IO server through the POST method, or trigger the GET request to activate a dispenser over the local network.

4) Adafruit IO Listeners (Cloud-based methods only):
The objective of the listener scripts is to gather the activation commands transmitted from the headset to the Adafruit IO feed and initiate the dispenser's activation. The scripts are written in Python and must run on a device connected to the same local network as the dispensers. (See Appendix K)

IV. Testing & Results

A. Performance testing

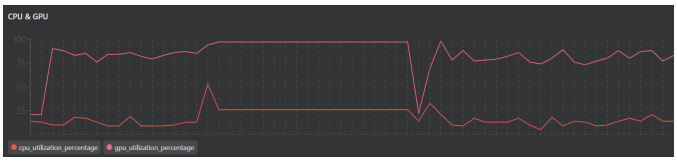
To evaluate the efficiency of each synchronization method and determine the best one, multiple metrics were tested. The most important one is the network latency (the delay measured as the time taken for a packet to travel from the VR headset to the olfaction dispenser), as it has a direct impact on the immersion of the user and his perception of the scents. Other metrics include hardware performance.

1) Latency:
To calculate latency, the trigger-by-interaction type asset (perfume) was used to track the exact moment of the activation of the twin dispenser. Using a modified version of the C# scripts and Adafruit listener scripts, it is possible to get the exact timestamps (precision to the milliseconds) at which the requests are sent from the headset and received by the olfaction dispensers. The packet analyzer software "WireShark" was also used in parallel to verify the veracity of the timestamps obtained. Results are listed in Table I.

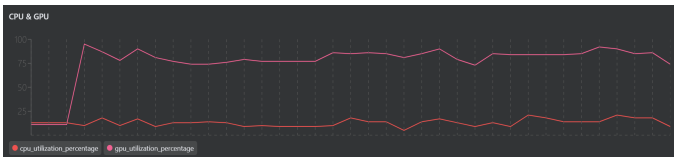
TABLE I
Latency (in seconds) Comparison

Latency	Local-based	cloud-based HTTP	cloud-based MQTT
Maximum	0.289 s	0.884 s	0.436 s
Minimum	0.041 s	0.249 s	0.186 s
Average	0.166 s	0.462 s	0.268 s
Standard dev.	0.006 s	0.207 s	0.082 s

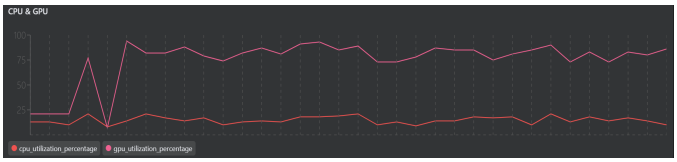
2) Headset Metrics:
The VR headset metrics are a great indicator for the quality of the implementation as the performance of the headset directly impacts the synchronization with the olfaction dispensers. The following results were obtained using the "OVR Metrics Tool" on the Meta Quest Developer Hub software, and following a standardized test. The obtained results in Figure 5 show a similar tendency and approximately the same percentages for the cloud-based approaches (between 75% and 95% for the GPU and between 5% and 25% for the CPU). However, the values for the local-based method are slightly more elevated.



(a) Local-based method



(b) Cloud-based method (HTTP)



(c) Cloud-based method (MQTT)

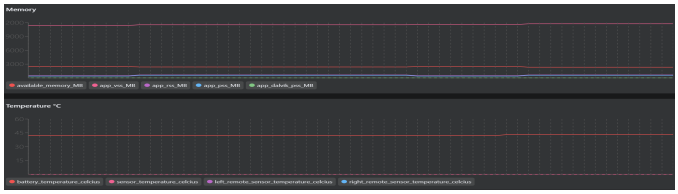
Fig. 5. CPU & GPU Utilization comparison

On the other hand, in Figure 6, we observe that the memory and temperature metrics are exactly equal in the three cases (around 3000 MB of available memory and 44°C in temperature).

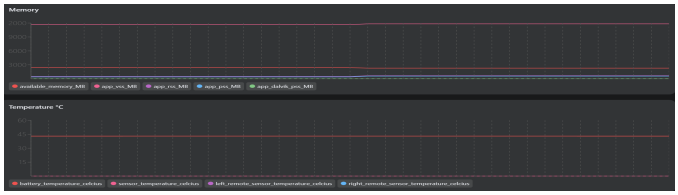
B. User test experience

To evaluate the effectiveness and user perception of the different synchronization methods, a user test was conducted. The aim is to validate the performance test results and conclude which method is the most efficient from the users' point of view. Participants' feedback will also determine if the integration of the olfaction cue in VR impacted positively the overall immersion and user satisfaction in the virtual experience.

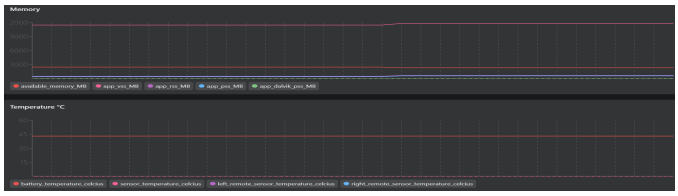
1) Experiment overview:
Each user undergoes the same test three times (one for each synchronization method) without knowing the order of the scenarios. The user has to navigate the room, locate the virtual assets that will emit a scent and interact with them either by approaching them (trigger by proximity) or interacting with them using the VR hand controller (trigger by interaction). At the end of each scenario, the user answers a questionnaire.



(a) Local-based method



(b) Cloud-based method (HTTP)



(c) Cloud-based method (MQTT)

Fig. 6. CPU & GPU Utilization comparison

At the end of the experiment, the user answers a general questionnaire about the overall experience. (See Appendix F)

2) Participants:

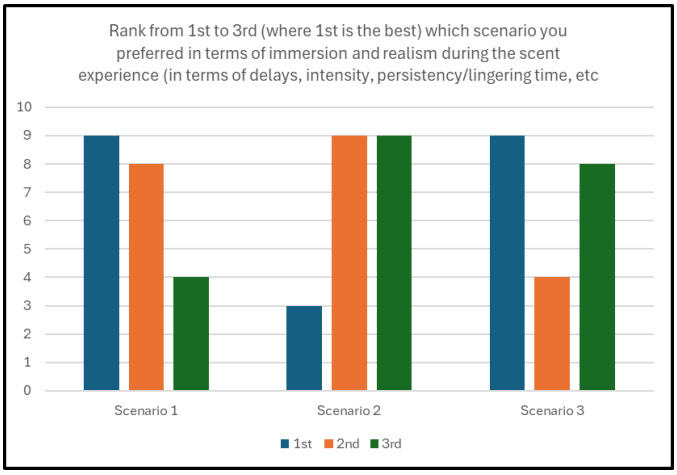
21 random individuals volunteered to participate in the user test including 19 men, 4 women and 1 unspecified, with ages ranging from 19 to 52. One-third of the participants had never used VR before, while the rest had at least a first experience. To avoid potential biases, they were split into two groups, experiencing the scenarios in a different order as detailed in Table II.

TABLE II
GROUP DIVISION

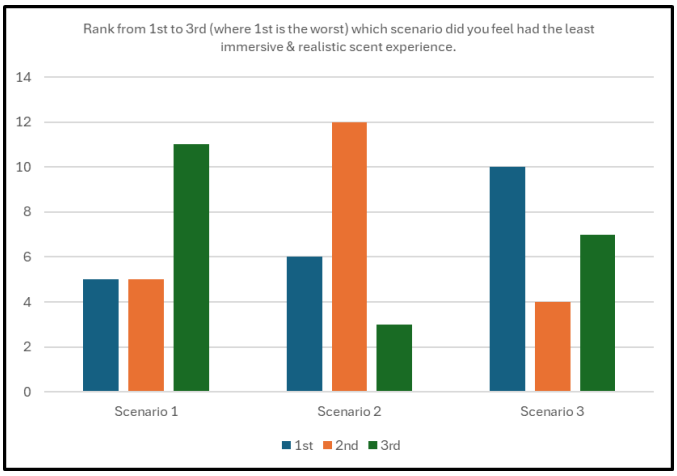
GROUP	METHOD 1	METHOD 2	METHOD 3
A	LOCAL HTTP	ADAFRUIT HTTP	ADAFRUIT MQTT
B	ADAFRUIT MQTT	ADAFRUIT HTTP	LOCAL HTTP

3) Questionnaire:

Before the beginning of the experiment, each participant signed a consent form and completed a demographics questionnaire. After each scenario, testers were asked to fill a questionnaire of 9 questions which were the same each time (27 questions in total). At the end of the experiment, they filled a final questionnaire of 14 questions about the general experience, comparing the three scenarios altogether. Most questions are answered by choosing an option between the following: *Strongly Disagree, Disagree, Neutral, Agree, Strong Agree*; or choosing between *scenario 1, scenario 2* and *scenario 3*. (See Appendix G)



(a)



(b)

Fig. 7. Scenarios classification according to users

4) Results (See Appendix H):

Overall, 100% of the volunteers enjoyed the use of smells during the experience, and the majority (over 90%) agreed that the smells enhanced the realism of the simulation and helped making the experience more immersive and enjoyable which correlates with the feedback observed in the ”OmniScent” study of Simiscuka *et al.* [3]. Most of the participants enjoyed the quality of the scents and did not find them disturbing. Over 90% agreed that the digital twin of the laboratory resembles the original one in terms of visual quality and over 60% preferred the proximity-trigger type objects over the interaction-trigger type. When participants were asked to rank the scenarios from best to worst as seen in Figure 7, the local-based method (scenario 1) was the most favored, receiving the highest number of 1st place rankings. The cloud-based HTTP (scenario 2) method followed with the majority of participants placing it in 2nd or 3rd place. However, The feedback on the cloud-based MQTT method (scenario 3) was more polarized. Some volunteers ranked it as being the best scenario, while an equal number found it to be the least preferred.

V. Analysis & Results interpretation

The performance tests showed that the local-based method had the lowest average latency (0.166s), followed by cloud-based MQTT (0.268s), and cloud-based HTTP (0.462s). These results join the conclusion drawn by Simiscuka *et al.* when comparing the synchronization approaches [5]. This technical performance correlates with user preferences, as the local-based method was ranked as the best scenario by the majority of participants. This correlation suggests that users can perceive the latency differences to a certain extent, even if they're not explicitly aware of the technical details.

While the local-based method showed slightly elevated GPU and CPU utilization compared to the cloud-based approaches, this difference did not negatively impact the users' preference. In fact, the local-based method was still voted the best by users, showing that the minor increase did not affect the overall experience.

The technical tests showed that the cloud-based MQTT method outperformed the cloud-based HTTP method in terms of latency. Surprisingly, user feedback on these two methods was more varied. While the HTTP method was consistently ranked second or third, the MQTT method received more divergent responses, with some users ranking it as the best and others as the worst. This discrepancy suggests that other factors, apart from latency, may have influenced the users' perception. Despite the divided opinions on the synchronization scenarios, the majority of the participants agreed on the benefits of adding the sense of smell in VR reinforcing the conclusion reported by Simiscuka *et al.* [3].

VI. Conclusions

In conclusion, the technical analysis and the user test results revealed a strong correlation. The local-based method, which demonstrated the shortest latency timing, was also the most favored by users. The polarized responses to the cloud-based MQTT method, despite its superior performance compared to HTTP, suggest that other factors also play a role in user perception. Overall, the integration of the olfaction cue in VR significantly increased the users' immersion and their enjoyment of the experience.

References

[1] T. Nakamoto, T. Hirasawa, and Y. Hanyu, "Virtual environment with smell using wearable olfactory display and computational fluid dynamics simulation," in *Proceedings of the Institute of Electrical and Electronics Engineers (IEEE)*, Jun. 2020, pp. 713–720. doi: 10.1109/vr46266.2020.00094.

[2] Y. Liu *et al.*, "Soft, miniaturized, wireless olfactory interface for virtual reality," *Nature Communications*, vol. 14, no. 1, Dec. 2023, doi: 10.1038/s41467-023-37678-4.

[3] A. A. Simiscuka, D. A. Ghadge, and G. M. Muntean, "OmniScent: An Omnidirectional Olfaction-Enhanced Virtual Reality 360° Video Delivery Solution for Increasing Viewer Quality of Experience," *IEEE Transactions on Broadcasting*, vol. 69, no. 4, pp. 941–950, Dec. 2023, doi: 10.1109/TBC.2023.3277215.

[4] Z. Wang, K. Han, and P. Tiwari, "Digital twin simulation of connected and automated vehicles with the unity game engine," in *Proceedings 2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence, DTPI 2021*, Institute of Electrical and Electronics Engineers Inc., Jul. 2021, pp. 180–183, doi: 10.1109/DTPI52967.2021.9540074.

[5] A. A. Simiscuka, T. M. Markande, and G. M. Muntean, "Real-Virtual World Device Synchronization in a Cloud-Enabled Social Virtual Reality IoT Network," *IEEE Access*, vol. 7, pp. 106588–106599, 2019, doi: 10.1109/ACCESS.2019.2933014.

[6] Inhalio. (2024). "Scent Diffusion Technology." [Online]. Available: https://inhalio.com/tech/scent-diffusion/.

[7] Meta. (2024). "Meta Quest 2 - Technical Specifications." [Online]. Available: https://www.meta.com/quest/products/quest-2/tech-specs/.

[8] TP-Link. (2024). "Archer C6 AC1200 Wireless MU-MIMO Gigabit Router - Specifications." [Online]. Available: https://www.tp-link.com/us/home-networking/wifi-router/archer-c6/v2.8/#specifications.

Appendix A
Literature Review

Integrating wireless control of multiple olfaction dispensers in Digital Twinning for improved VR experience: A comprehensive review

Marouane Ben Zineb

***Abstract*— Virtual Reality (VR) has revolutionized the interactions with the virtual world by immersing the users in a simulation detached from physical reality while enhancing their perception of their digital surroundings. This is done by focusing on the human senses such as vision, audition and equilibrioception. However, the full spectrum of human senses is underutilized, particularly the sense of smell. This literature review focuses on the novel integration of olfactory dispensers in digitally twinned environments to improve the VR experience. It will discuss olfactory techniques, their design challenges, and the synchronization mechanisms between the virtual and real worlds. The concept of digital twinning is dissected to understand its role and application in enhancing sensory experiences within virtual environments.**

***Index Terms* — Digital Twinning, IoT, Multi-sensory Integration, Olfaction, Synchronization mechanism, Unity, Virtual Reality, Wireless control**

I. INTRODUCTION

The realms of Virtual Reality (VR) have significantly altered our engagement with digital environments. By mainly leveraging visual, auditory, and "equilibrioception" (balance-related) senses, these technologies offer immersive experiences that blur the lines between the real and the virtual. However, the sensory spectrum in VR is not exploited to the fullest limiting the potential applications in which the technology can be used. Hence the importance of developing the multi-sensory integration (MSI) in the virtual domain. In the context of VR, MSI refers to the process of fusing the different [2]sensory inputs dispatched by the sensory systems installed with the objective of enhancing the user's immersion or manipulating his perception. For the MSI to be effective, it relies on other factors such as the quality of the VR equipment used, the realism of the sensory data delivered to the user (quality of the audio and visuals, accuracy of synthetic tastes and odours) and the speed of data transfer between the different compounds [1]. A particularly underestimated sensory modality in VR is olfaction. The integration of olfactory feedback can significantly enrich virtual experiences, yet it remains relatively undeveloped. Olfaction plays a critical role in human perception, influencing emotions, memories, and even decision-making processes [2]. In VR, the incorporation of scent can add a layer of realism and depth, enhancing the user's engagement. This is especially relevant in scenarios where environmental interaction is key such as therapeutic treatments or interactive education [2].

The focus of this review is to compare the latest findings in an attempt to design and implement a prototype system where olfactive devices are triggered based on recognized content. In a VR scenario, using a digital twin of a real-world environment, scent effects will be activated in relation to the user's location and interaction with specific 3D elements or assets. The challenge lies not only in understanding the optimal choice of the olfactive devices within the VR environment but also in the development of a synchronization mechanism within the context of establishing a continuous communication between the user (VR headset), the "digitally twinned" simulation and the scent emitters.

II. VR EQUIPMENT

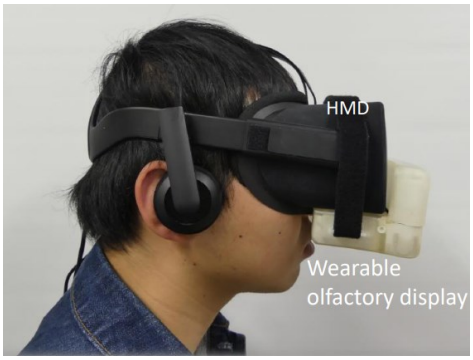
The quality and capabilities of VR headsets and controllers fundamentally shape the user's experience. High-resolution displays, accurate motion tracking, and responsive controls are among the key features that enhance the realism of the virtual environment. Therefore, the choice of the equipment is a fundamental step. Different VR headsets vary in their features and capabilities such as the display quality and the tracking accuracy. But the main splitting difference is the reliance on an external computer. Some, like the Oculus Quest 2, offer a standalone experience without the need for an external computer, emphasizing ease of use and portability. Others such as the Oculus Rift, provide a more high-end experience with superior tracking and display quality, often requiring a connection to a powerful monitor [3].

Multi-sensory integration (MSI) includes the interactive aspect of VR. These interactions with simulated objects allow the user to engage physically with the virtual environment, providing a more immersive and realistic experience. VR controllers are crucial for this part as they translate the physical movements into virtual actions that can trigger events in the simulation. The VR controllers are usually equipped with different buttons and motions sensors.

III. OLFACTION INTEGRATION

Integrating an olfactory cue in a VR environment can be done through various techniques. Nakamoto et al. relied on a wearable olfactory display mounted strategically to the VR headset with the idea of increasing the speed of delivery of the odour while still being able to control the intensity and the scattering. This is done with a technology called SAW (Surface Acoustic Wave) which atomizes the liquid containing the scent making the switching between different odours rapid. The

device is equipped to be able to recollect most of the odours. However, the system is not very efficient. During the experiment which consisted in locating the odour's source in



the simulation, only half of the participants were able to recognise the right odours and the researchers encountered an issue of smell leakage around the users when using the device. [4].

Fig. 1. Side view of a user wearing olfactory display attached to a head mount display (HMD). [4]

A better alternative would be the usage of miniaturized odour generators (OGs) as described by Yiming Liu et al. where the whole system is skin-integrated. As seen in figure 2, it is in the form of a skin patch containing an electrical module responsible for the data transfer between the simulation and the odour generators and protected by a layer of Polydimethylsiloxane (PDMS). The device is attached to the skin through adhesives installed at the bottom of the interface. The OGs liberate the smell through the heating of odorous wax made of paraffin [2]. This solution is described as highly performant in terms of energy savings and response rate, but smell directivity and persistency was not discussed. However, it is limited in terms of odours selection as only 2 OGs can be placed at the same time in each device.

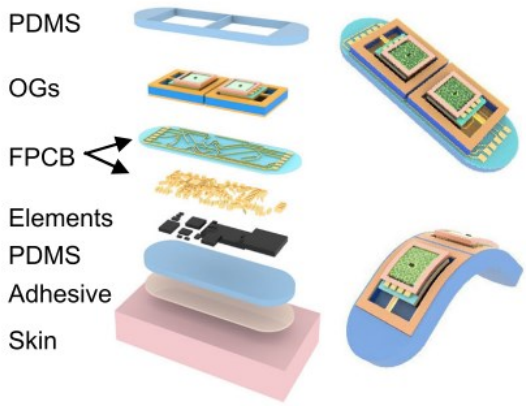


Fig. 2. Exploded view of the skin-integrated wireless olfaction interface based on two odour generators. [2]

A completely different approach is described by Simiscuka et al. in their attempt of integrating olfactory dispensers with a 360° VR video delivery system to enhance viewer experience, where the olfactory interface is a diffuser placed statically in the physical environment rather than attached to the user. The device used in this experiment was developed by the company "Inhalio". Each sensor contains a cartridge with paraffin wax beads carrying a specific smell. The device uses a fan to propel the odours unidirectionally. This alternative fixes the odour persistency issue around the user but could potentially lead to the scent not dissipating around the dispenser's location if the space is too enclosed [5].

IV. DIGITAL TWINNING

"Digital Twinning" is a concept that emerged in the early 2000s in the domain of data transfer between the physical and the virtual worlds. In this context, it refers to the creation of a dynamic virtual representation of the physical environment in VR enabling the user not only to engage with and manipulate elements within the virtual scene but also to influence their real-world counterparts [6], [7], [8]. Such a two-way interaction bridges the gap between the tangible and digital worlds.

A Digital Twin simulation typically comprises two distinct layers. The first layer consists of the physical environment, filled with objects destined to be simulated and replicated in the virtual reality with attributed characteristics or triggering call-to actions. The other layer is the simulated world which replicates exactly the physical one, with the possibility of modifying the texture or appearance of some simulated objects [6], [8]. For instance, a simulated odour diffuser from the physical world can be displayed in the simulation as a bottle of perfume that can be triggered through the VR controller to add a note of realism. The virtual layer is divided into different Unity components (API and other tools). A third layer that must be considered in most cases is the information exchange between the two layers through a synchronized communication mechanism [8].

A. Unity software

Unity is a versatile and powerful game engine and development platform, renowned for its wide application in creating interactive 3D content, particularly in video games, simulations, and virtual reality (VR) environments. It supports the C# programming language allowing the integration of custom scripts to control [7]. In the context of digital twinning, Unity's capabilities are utilized to create accurate and interactive replicas of physical spaces and objects. The process of creating a digital twin of a physical space often starts with the collection and integration of data from various sources to create a detailed 3D digital model. For example, publicly available mapping data from sources like OpenStreetMap can be used combined with additional data for enhanced accuracy, or the Unity Asset Store which is a digital marketplace that provides a large collection of assets and templates that are ready to use. This integrated data is then processed using external tools to blend different data formats into a unified model. Once an initial Digitally twinned model is created, Unity provides enhancing tools to add dynamic features, animations or scenarios and even applying physics to objects [9]. The final model is then either uploaded to the memory of the standalone VR headset or simulated

directly from a tethered one using the computational power of an external computer.

V. SYNCHRONISATION AND WIRELESS CONTROL

A stable continuous communication between the Digital Twin environment and the physical world must be initiated to make the integrated olfactive devices interactive with the player's movements and actions. This synchronization requires real-time data transfer and processing to ensure that the sensory outputs from the olfactory dispensers match the user's interactions and movements within the virtual environment. Effective synchronization mechanisms involve precise timing to align the release of scents with specific actions or locations in the VR simulation. Simiscuka et al. relied on a novel technique called VR-IoT Environment Synchronization Scheme (VRITESS) to continuously synchronise the states of physical objects and their digitally twinned versions in the virtual reality. VRITESS establishes a two-way connection system between the IoT Integration Platform (ITINP), which receives operations executed in the real world, and the VR IoT Platform (VRITIP), which handles operations in the simulated world. Both ITINP and VRITIP utilize the Network Time Protocol (NTP) to ensure precise synchronization based on timestamps of these actions. The timestamps and actions are registered in a database that is either local using a local network or cloud-based which turned out to be a slower solution. The MQTT (Message Queuing Telemetry Transport) and REST (Representational State Transfer) protocols were used for the data transfer as they are designed for real-time communication. The MQTT protocol is more suitable for lightweight data and employs a publish/subscribe system while the REST protocol relies on the HTTP protocol with a system of request/response [10].

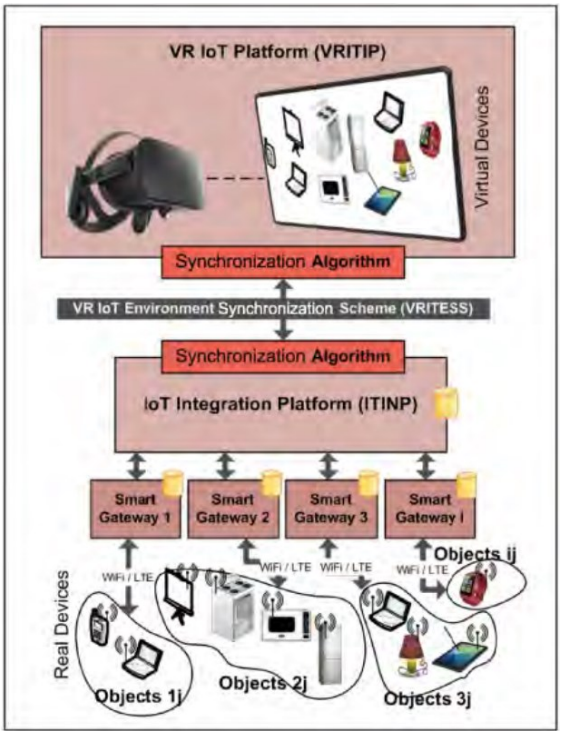


Fig. 3. The VR-IoT Environment Synchronization Scheme (VRITESS) architecture in the context of a Digital Twinning case. [10]

Wang et Al. also described the usage of TCP/IP locally and MQTT/HTTP with cloud to enable data transfer between their Unity simulation and their external tools in their attempt to develop a Digital Twin simulation of connected vehicles [6].
In the context of developing an Extended Reality (XR) Application Framework for a Digital-Twin-Based Smart Crane, Yang et Al. employed a combination of OPC UA (Open Platform Communications Unified Architecture) combined with MQTT (Message Queuing Telemetry Transport) and GraphQL for efficient data acquisition and transmission. OPC UA ensures secure, reliable, and platform-independent communication, while MQTT offers lightweight messaging, ideal for IoT devices. GraphQL enhances data-querying capabilities [8].

VI. APPLICATIONS

Incorporating the sense of smell into virtual reality (VR) environments opens up a wide range of exciting possibilities in diverse sectors. For instance, in an educational setting, an olfactory VR experience could be used to teach botany where understanding the wide variety of plant species, their characteristics, and their environments is crucial. By incorporating VR with olfactory elements, educational institutions can create immersive, multi-sensory learning experiences that are not feasible in a traditional classroom setting [2].
In the medical domain, this technology could be used to treat some disorders related to eating by focusing on retro nasal olfaction phenomenon [1]. Additionally, it can aid in memory recall and cognitive therapy for patients with dementia, where familiar scents coupled with the corresponding visual display can trigger memories and emotional responses [2].

VII. PROJECT DESIGN

The project adopts a novel approach diverging from the dynamic scent solutions reviewed, and opting for a static olfactory dispenser solution as described by Simiscuka et al. [5]. This method presents unique challenges in ensuring timely and contextually relevant scent release. To address these challenges, the usage of the MQTT protocol for data transfer through a cloud-based network can be effective in reducing the delays of communication. The publish/subscribe system employed by the protocol is ideal in this scenario where data must be distributed from one source (VR headset) to various recipients (olfactive dispensers). However, choosing a simple local network implementing the HTTP protocol for communication is probably a much reliable and simpler approach as the Unity platform supports this type of communication by providing native tools for an easy setup. Utilizing timestamps is crucial to coordinate the timing of scent release with specific trigger events. The choice of the VR headset is also an important metric, and a standalone version could provide a degree of realism compared to a tethered one which needs to be plugged to a computing system. This allows

the user to move freely in both the physical realm and the digital twin simulation.

VIII. CONCLUSIONS

In conclusion, the integration of olfactory senses into VR represents a significant leap in the evolution of immersive technologies. A key aspect discussed is the technological advancements in olfactory integration, examining various methodologies like wearable olfactory displays, miniaturized odor generators, and static dispensers. These technologies, each with their unique advantages and limitations, build the framework for more nuanced and contextually relevant olfactory experiences in the VR domain. Additionally, the concept of digital twinning, was introduced especially in the context of VR and using the Unity software to generate a twin virtual world. The literature also illustrates the various communication protocols to ensure the synchronization and data transfer between the physical and virtual layer. Future research should focus on developing more sophisticated mechanisms for scent synthesis and delivery in VR environments. Moreover, expanding the spectrum of olfactory stimuli that can be accurately and efficiently reproduced will be crucial for achieving a fully immersive sensory experience.

REFERENCES

[1] F. Weidner, J. E. Maier, and W. Broll, “Eating, Smelling, and Seeing: Investigating Multisensory Integration and (In)congruent Stimuli while Eating in VR,” *IEEE Trans Vis Comput Graph*, vol. 29, no. 5, pp. 2423–2433, May 2023, doi: 10.1109/TVCG.2023.3247099.

[2] Y. Liu *et al.*, “Soft, miniaturized, wireless olfactory interface for virtual reality,” *Nat Commun*, vol. 14, no. 1, Dec. 2023, doi: 10.1038/s41467-023-37678-4.

[3] Vladislav Angelov, Emiliyan Petkov, Georgi Shipkovenski, and Teodor Kalushkov, “Modern Virtual Reality Headsets,” in *2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, 2020.

[4] T. Nakamoto, T. Hirasawa, and Y. Hanyu, “Virtual environment with smell using wearable olfactory display and computational fluid dynamics simulation,” Institute of Electrical and Electronics Engineers (IEEE), Jun. 2020, pp. 713–720. doi: 10.1109/vr46266.2020.00094.

[5] A. A. Simiscuka, D. A. Ghadge, and G. M. Muntean, “OmniScent: An Omnidirectional Olfaction-Enhanced Virtual Reality 360° Video Delivery Solution for Increasing Viewer Quality of Experience,” *IEEE Transactions on Broadcasting*, vol. 69, no. 4, pp. 941–950, Dec. 2023, doi: 10.1109/TBC.2023.3277215.

[6] Z. Wang, K. Han, and P. Tiwari, “Digital twin simulation of connected and automated vehicles with the unity game engine,” in *Proceedings 2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence, DTPI 2021*, Institute of Electrical and Electronics Engineers Inc., Jul. 2021, pp. 180–183. doi: 10.1109/DTPI52967.2021.9540074.

[7] W. Piper, H. Sun, and J. Jiang, “Digital Twins for Smart Cities: Case Study and Visualisation via Mixed Reality,” in *IEEE Vehicular Technology Conference*, Institute of Electrical and Electronics Engineers Inc., 2022. doi: 10.1109/VTC2022-Fall57202.2022.10012753.

[8] C. Yang *et al.*, “Extended Reality Application Framework for a Digital-Twin-Based Smart Crane,” *Applied Sciences (Switzerland)*, vol. 12, no. 12, Jun. 2022, doi: 10.3390/app12126030.

[9] S. Rundel and R. De Amicis, “Leveraging digital twin and game-engine for traffic simulations and visualizations,” *Front Virtual Real*, vol. 4, 2023, doi: 10.3389/frvir.2023.1048753.

[10] A. A. Simiscuka, T. M. Markande, and G. M. Muntean, “Real-Virtual World Device Synchronization in a Cloud-Enabled Social Virtual Reality IoT Network,” *IEEE Access*, vol. 7, pp. 106588–106599, 2019, doi: 10.1109/ACCESS.2019.2933014.

Appendix B
Project Design Plan

Project Design Plan

Student Name	Marouane Ben Zineb
Student ID	23266913
Project Title	Integrated wireless control of multiple olfaction dispensers in Digital Twinning for improved VR experience
Programme	MECE
Supervisor Name	Prof. Gabriel-Miro Muntean

RESEARCH QUESTION

How can the integration of wireless olfaction dispensers in a Unity-based digital twin environment enhance user experience, in terms of scent triggering via asset interaction and user proximity; and how different network & synchronization scenarios for the solution compare against each other performance-wise?

PROJECT SCOPE

The project involves these areas of research:

- 6 DoF Virtual Reality – involves immersive environments where users can move freely in three-dimensional space.
- Digital Twinning – A virtual representation of the lab.
- Synchronisation – Designing a synchronization approach to ensure real-time alignment between the interactions with the simulation and scent dispensing.
- Olfactive dispensers – integrating the olfactive cue to enhance sensory experience.
- Perceptual user testing – evaluating how users perceive and interact with the system created.
- Technical testing – Ensuring all components works correctly and evaluating several metrics.

The following technologies are used:

- Standalone VR headset (Meta Quest 2).
- Unity VR Game Engine.
- "Inhalio" Olfactive devices + odour beads of different scents.
- WiFi Routers (2,4 GHz / 5 GHz).

Excluded Areas/Technologies:

- As the ethical approval only allows Post Doctoral researchers and Phd students to perform perceptual user testing, I will guide some members of the performance engineering lab to conduct these tests and will perform the test analysis afterwards.

DESIGN APPROACH

The idea of the project is to synchronize the triggering of olfactive dispensers between the real world and the digital twin VR world:

- The player can move around the room with the standalone headset where different olfactive objects are placed.
- The room in Virtual Reality is a realistic twin of the actual DCU laboratory (S220) made with the Unity Game Engine.
- The olfactive objects are synchronized with real life olfactive dispensers using the Unity C# scripts that transmits orders in the form of HTTP GET and POST requests.
- The scripts are responsible for constantly monitoring the state of the VR olfactive objects & synchronizing the changes with the corresponding dispensers when a listener is triggered (distance / interaction).
- Some objects will be triggered through:
    - action (by interacting with the object using the Oculus controllers) and;
    - user distance, when the player is close enough (the intensity of the smell will vary depending on how close the player is from the object).
- When a trigger listener is activated, the corresponding request is sent to the endpoint of the olfactive devices (using a local API). The devices adapt accordingly.

The approach will be tested with different scenarios, such as:

- Network conditions (e.g background vs no-background traffic)
- Synchronized cloud-based (e.g Adafruit, MQTT) vs synchronized local network-based (WiFi) vs non-synchronized (scents constantly diffused) scenarios

The following metrics will be analysed in a technical test:

- Latency
- Throughput
- VR headset metrics (GPU/CPU usage, memory usage, maximum temperature)

A test will be carried out with multiple participants where the followings metrics will be evaluated with the aid of the feedback (via questionnaires) collected such as:

- Ranking which type of trigger is best (By distance or by interaction with the object) based on user preferences.
- Quality and realism of the scents.
- Quality of the visual display and the degree of similarity with reality.
- Impact of adding scents to the user enjoyment.
- Evaluation of any disturbance caused by the smells
- Immersion and realism overall.

The test consists in users taking turns going around the room and testing each olfactive-triggering object (3D elements in the VR scenario) by interacting with them.

SUCCESS CRITERIA

This project will be considered successful if the following criteria are met:

- Successful integration and synchronization of at least three wireless olfactive dispensers within the VR environment.
- Analysis of the different synchronous and non-synchronous scenarios as well as the different network scenarios in terms of tested metrics.
- All the submissions are completed (Literature Review, Project Presentation, Project Design Plan, Research Log, Research Paper, Final Portfolio).

DETAILED TIMELINE

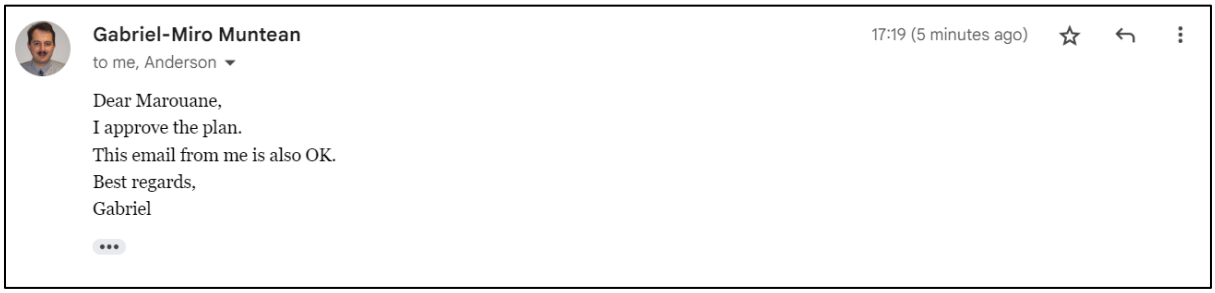
Task	Duration (days)	MAY 2024																									
		W1 (06/05)							W2 (13/05)							W3 (20/05)							W4 (27/05)						
		M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S
PROJECT DESIGN PLAN																													
Creating a Digital Twin of the Lab	21																												

Task	Duration (days)	JUNE 2024																									
		W1 (03/06)							W2 (10/06)							W3 (17/06)							W4 (24/06)						
		M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S
Writing the C# scripts for synchronization	7																												
Setting up the olfaction devices	2																												
Testing (Bug fixing & improvements)	5																												
Technical testing (metrics analysis)	5																												
Preparing user test & survey	4																												
Research paper redaction (start)	7																												
Search for participants for the test	21																												

Task	Duration (days)	JULY 2024																									
		W1 (01/07)							W2 (08/07)							W3 (15/07)							W4 (22/07)						
		M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S
USER TEST PERIOD	1																												
Research paper redaction (continue)	28+																												
User test data analysis	7																												

Task	Duration (days)	AUGUST 2024																									
		W1 (29/07)							W2 (05/08)							W3 (12/08)							W4 (19/08) +						
		M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S
Research paper redaction (finalize)	7																												
Additional time to account for delays	7																												
Building the final portfolio	7																												
PORTFOLIO SUBMISSION DEADLINE																													
Preparing for the final interview	-																												

SUPERVISOR APPROVAL



Gabriel-Miro Muntean
to me, Anderson
17:19 (5 minutes ago)

Dear Marouane,
I approve the plan.
This email from me is also OK.
Best regards,
Gabriel

Appendix C
Research Log

*

Masters Project Research Log

Masters in Electronic and Computer Engineering 2023/2024

Student Name: Marouane Ben Zineb
Student ID: 23266913
Project Title: Integrated wireless control of multiple olfaction dispensers in Digital Twinning for improved VR experience

Please read before making entries in this log

The purpose of this Project Research Log is to capture concise, focused summaries of research materials you read, as you progress through your project. The emphasis is to record (i) how the material you have read will determine or influence your project solution approach and (ii) your assessment of the key strengths and weaknesses of the solutions, methods, technologies, etc. proposed in the material you have read.

In the first stage of your project, the literature review, use the Log to capture this information for the key papers you have read (for example, the three most important papers of your 10 literature review references). As your project progresses into the design and implementation phases, you will need to continue to search the literature so you can review, revise and refine your initial thinking and the details of your approach to a project solution. Use this Research Log to capture your continued research reading and its influence on your project design and implementation.

Be selective about what you record in this log. Do not use it as an informal notebook while you are reading a new paper. Only make an entry after you have read a paper that you consider important to the development of your project solution. It is expected that, by the end of the project, you will have made **between 10 and 20 entries (20 maximum)**.

Share your log with your supervisor for viewing throughout the project. You will submit the final version of the log for grading, at the end of the project implementation period. It will be assessed on the basis of how well you have used your analysis of the literature to inform your project design, implementation and the evaluation of your project results. The Research Log contributes **5%** to the overall project mark.

Note: All entries you make in this log must use the prescribed format shown on the next page. You will maintain other notes as you progress through your project but they should not be recorded here. Fill in the details where the *** signs are.

Statement of project problem / research question (maximum 200 words)
This statement should be periodically reviewed and updated, as necessary, as your project progresses and you gain further insight into the detailed project challenges, requirements and objectives as your project work moves from background reading, literature review, initial project design planning and detailed design and implementation. Initially, start by stating your current understanding of the project objectives. After each meeting with your supervisor, review and refine your project problem statement, as required.

How can the integration of wireless olfaction dispensers in a Unity-based digital twin environment enhance user experience, in terms of scent triggering via asset interaction and user proximity; and how different network & synchronization scenarios for the solution compare against each other performance-wise?

A complete reference for the paper

T. Nakamoto, T. Hirasawa, and Y. Hanyu, "Virtual environment with smell using wearable olfactory display and computational fluid dynamics simulation," Institute of Electrical and Electronics Engineers (IEEE), Jun. 2020, pp. 713–720. doi: 10.1109/vr46266.2020.00094.

Summary of paper (maximum 100 words)

This paper presents an innovative system to introduce smell in Virtual Reality using a wearable olfactory display and computational fluid dynamics (CFD) simulation. The researchers developed a two-story virtual building with dynamic odor distribution calculated using CFD. They created a wearable olfactory display using micro dispensers and SAW atomizers, capable of quickly switching between scents.

How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)

The paper addresses the integration of the olfactory cue in Virtual Reality which is one of the main objective of my current project. The idea of a mobile solution integrated directly in the VR headset is an interesting alternative rather than having a local scent dispensing solution.

What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)

Strengths:
- Innovative and portable design.
- Compact technology.
- Comprehensive evaluation including comparison with desktop olfactory display.

Weaknesses:
- The system is limited in the number of scents that can be distributed at once.
- Efficiency needs improvement (Results not satisfactory)

A complete reference for the paper
Y. Liu *et al.*, “Soft, miniaturized, wireless olfactory interface for virtual reality,” *Nat Commun*, vol. 14, no. 1, Dec. 2023, doi: 10.1038/s41467-023-37678-4.
Summary of paper (maximum 100 words)
This article describes a more advanced and miniaturized olfactory interface that can be directly attached to the skin or integrated into face masks. It uses arrays of flexible, miniaturized odor generators (OGs) that are wirelessly controllable to deliver programmable scents during Virtual Reality experiences.
How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)
The paper is relevant to my project as it describes an original, compact, lightweight and mobile olfactory system which could be an interesting alternative to my solution.
What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)
Strengths: - Lightweight and compact system. - Wireless and programmable. - Integrated with full motion tracking in Virtual Reality. Weaknesses: - Limited to 9 base scents in the larger array. - Discomfort issues for skin-integrated version. - Complexity of the system

A complete reference for the paper
A. A. Simiscuka, D. A. Ghadge, and G. M. Muntean, "OmniScent: An Omnidirectional Olfaction-Enhanced Virtual Reality 360° Video Delivery Solution for Increasing Viewer Quality of Experience," *IEEE Transactions on Broadcasting*, vol. 69, no. 4, pp. 941–950, Dec. 2023, doi: 10.1109/TBC.2023.3277215.
Summary of paper (maximum 100 words)
This paper presents "OmniScent", a system to combine olfaction with 360° videos in Virtual Reality. The solution uses four scent dispensers placed around the user to release smells from different directions corresponding to the location of objects/scenes in the 360° video.
How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)
This paper is highly relevant to my project on integrating wireless olfaction dispensers in a digital twin in Virtual Reality. It demonstrates a working implementation of static directional scent delivery systems that can be synchronized with a specific event. The user study design and results offer insights into evaluating user experience with olfactory-enhanced Virtual Reality. The challenges and solutions presented, like synchronization and smell direction perception, are the same addressed in my project.
What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)
Strengths: - An innovative way of integrating the olfaction cue in Virtual Reality. - A user study with positive results on immersion and enjoyment. Weaknesses: - The synchronization of the olfaction dispensers is manual with the usage of timestamps. - The study does not address issues such as scent mixing or smell persistence.

A complete reference for the paper

A. A. Simiscuka, T. M. Markande, and G. M. Muntean, "Real-Virtual World Device Synchronization in a Cloud-Enabled Social Virtual Reality IoT Network," *IEEE Access*, vol. 7, pp. 106588–106599, 2019, doi: 10.1109/ACCESS.2019.2933014.

Summary of paper (maximum 100 words)

This paper introduces VRITESS, a synchronization scheme for integrating Virtual Reality and Internet of Things (IoT) environments. The authors implemented and tested two approaches: a local network-based solution and a cloud-based solution using Unity. Performance testing showed lower latency in the local network approach compared to the cloud-based approach and for the cloud-based approach, MQTT showed a better performance than REST API in terms of delay and data traffic.

How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)

This paper provides a framework for synchronizing IoT devices with virtual representations in Virtual Reality. The comparison of local network vs cloud-based approaches (Adafruit IO) and different communication protocols (MQTT vs REST) offers useful information for my implementation.

What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)

Strengths:

- Practical implementation and testing of both local and cloud-based approaches.
- Detailed performance comparison of MQTT and REST protocols.
- Detailed latency and data traffic analysis.

Weaknesses:

- Cloud-based approach limited by Adafruit IO rate limits (1 request/second).
- Lacks a user study to evaluate the effectiveness of the VR interface for IoT control.

A complete reference for the paper
Z. Wang, K. Han, and P. Tiwari, "Digital twin simulation of connected and automated vehicles with the unity game engine," in *Proceedings 2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence, DTPI 2021*, Institute of Electrical and
Summary of paper (maximum 100 words)
This paper proposes a Digital Twin simulation architecture for connected and automated vehicles (CAVs) using the Unity game engine. The architecture consists of a physical world layer and a digital world layer with three sub-layers: Unity game objects, Unity scripting API, and external tools.
How is this paper relevant to solving your project problem or addressing your research question? (maximum 100 words)
This paper provides an understanding of how to create a digital twins and environment using Unity. The paper's approach to integrating cloud services and external tools with Unity could be helpful in my study that requires the synchronization between Virtual Reality and the olfaction dispensers
What are the strengths and weaknesses of the solutions/methods/technologies proposed in this paper? (maximum 100 words)
Strengths: - Good explanation of the digital twin architecture and the integration of external tools in Unity Weaknesses: - Does not address the challenges in the synchronization between physical and virtual worlds.

Appendix D
Unity VR APK building

Building a VR Unity app on the Quest 2 headset

DEPENDENCIES:

- The headset used must be in developer mode. (See last step).
- The account to which is connected the headset must be in developer mode
- The computer used to build must have the Meta Quest Developer Hub installed (not the normal hub but THE DEVELOPER one). Normally installing the Developer Hub will come with the ADB driver included, but in case it doesn't you can install the driver manually from this link: https://developer.oculus.com/downloads/package/oculus-adb-drivers/
- The account used to connect on the Developer Hub on the computer must be the same as the account logged on the headset as administrator.
- Activate the developer mode of the headset from the Developer Hub after connecting it to the app.

STEPS:

Import the world folder:

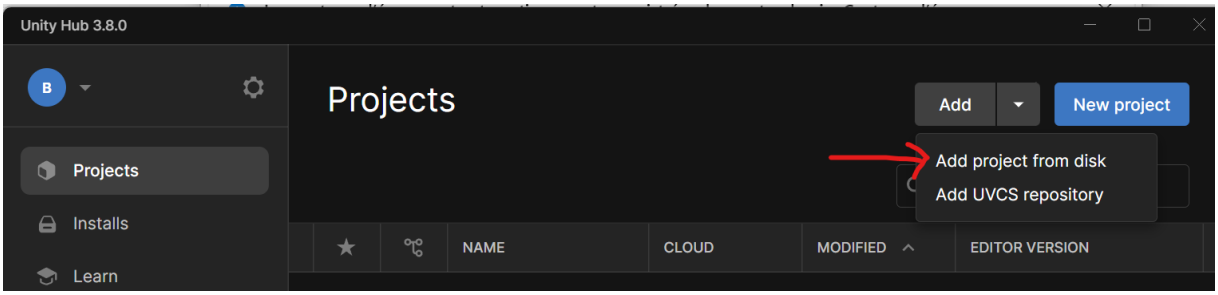
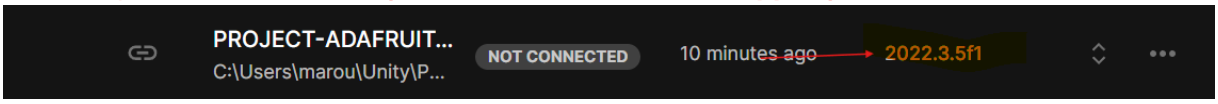


Figure 1: Adding project to Unity Hub

Use Unity version 2022.3.5f1 (Must have Android Build Support)



After opening the project, in Project settings verify these settings:

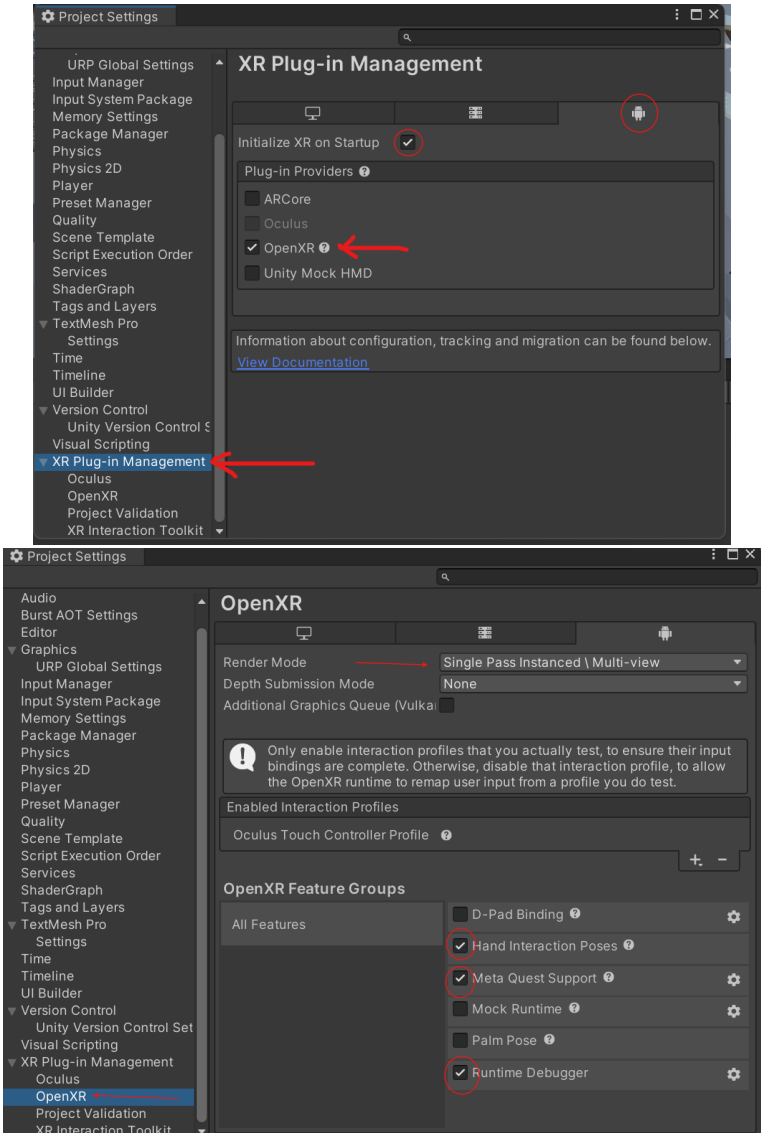
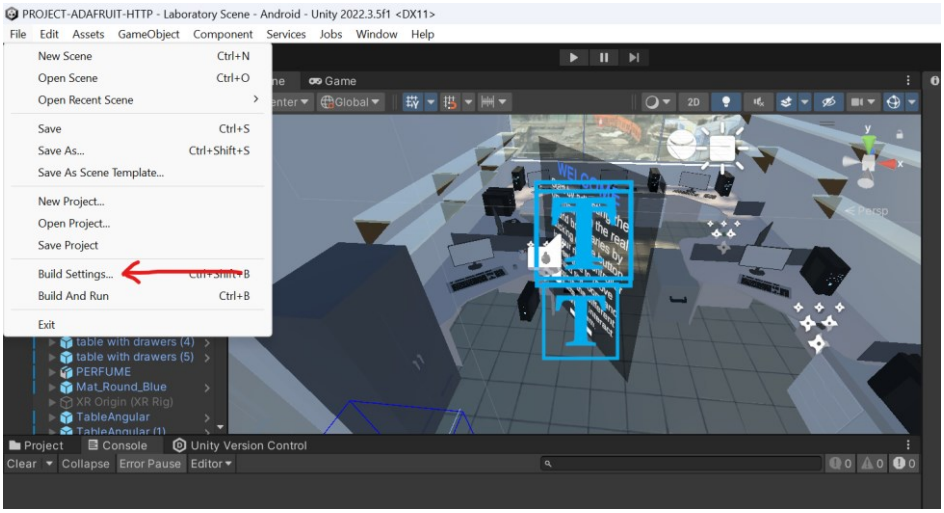


Figure 2: Project Settings

Go to build settings



Verify that the parameters are the same as in the picture below:

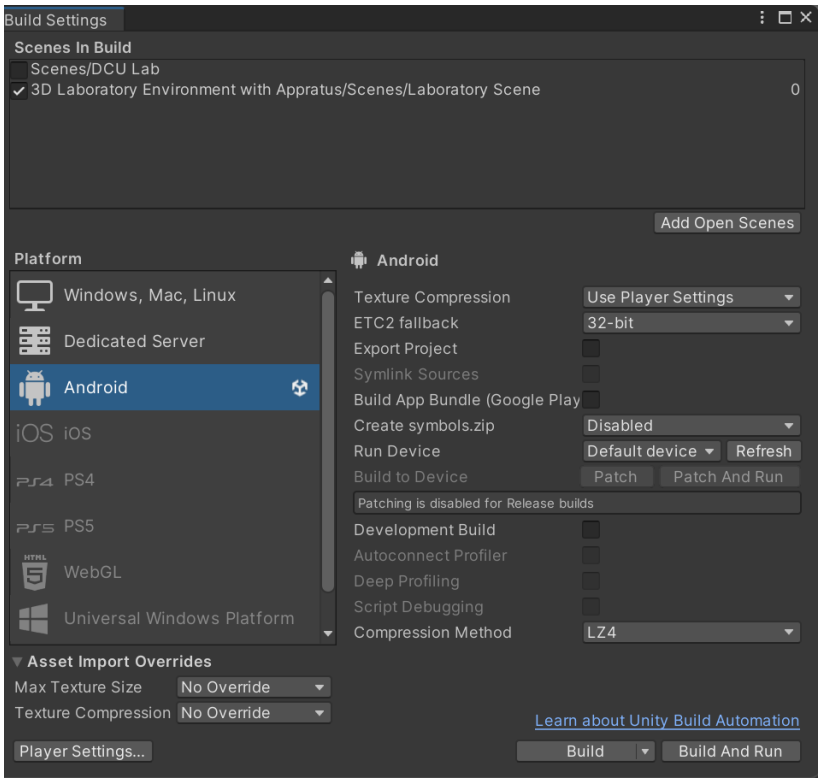
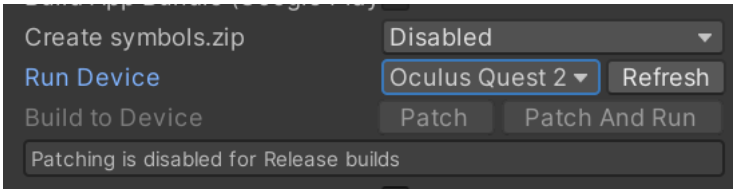


Figure 3: Build Settings

The headset must be plugged to your computer with developer mode set, and USB debugging activated and should appear in the Run Device slot instead of "Default device".



If after refreshing, the device still does not appear, that means that something is missing: either the headset is not in developer mode, or the USB debugging was not activated or the ADB driver is not installed.

If everything is set, click on Build & run.

Appendix E
Olfaction Dispensers
&
VR Assets Twinning

TWINNING OLFACTION DISPENSERS TO VR ASSETS

1. Local-based method:

- Attach an **OlfactionObject.cs** instance & a **DataSender.cs** instance to the object you would like to twin to an olfaction dispenser.
- Specify the olfaction device IP on the local network and the trigger type (distance or interaction), as well as the trigger distance in meters (if it is distance-type).
- Attach the **DataSender** instance to the "Data Sender" Slot of the **OlfactionObject** instance. Particles can be added to the "Perfume Particles" slot for more realism (recommended).

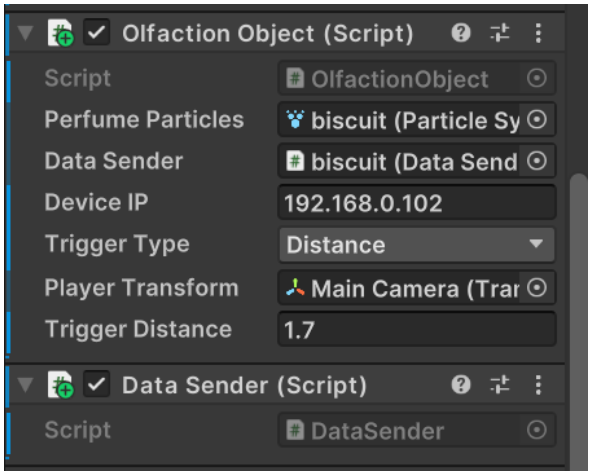


Figure 1: Olfaction Object instance

The "Player Transform" slot takes the Main Camera instance from the XR Origin as a parameter.

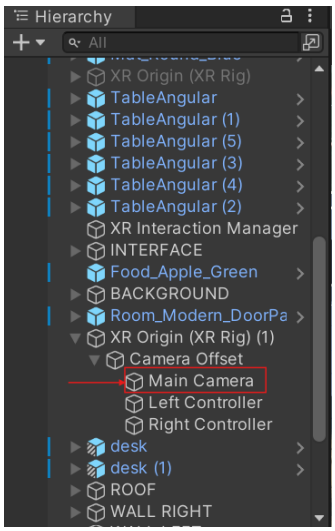


Figure 2: Main Camera instance location in hierarchy

2. Cloud-based method (Adafruit HTTP & MQTT):

The **OlfactionObject** instance parameters are the same as in the first case.

However, the **DataSender** instance now needs configuration.

- For both the HTTP & MQTT method, an Adafruit Key and a username (available in the Adafruit IO account settings) must be specified, as well as the feed name that will collect the data.

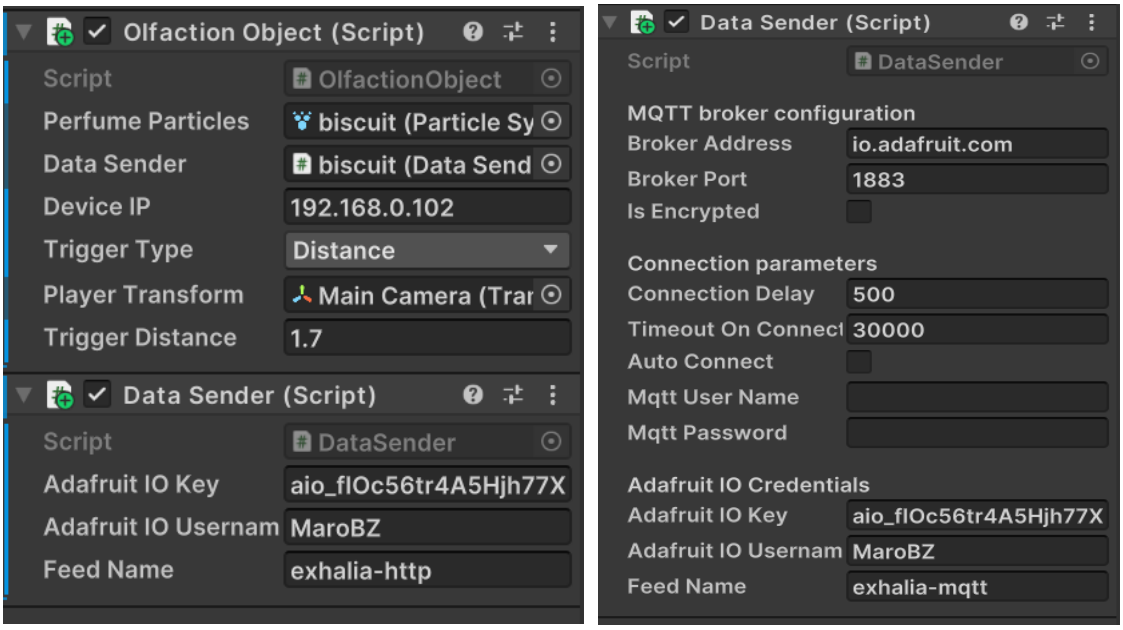


Figure 3: DataSender instance configuration for cloud-based methods

Appendix F
User Test Tutorial

USER TEST TUTORIAL

BEFORE EXPERIMENT:

- Remove all obstacles present in the boundaries of the room (The chairs and the bin).
- Check that the boundaries did not disappear and are still precise. (If not: redraw the boundaries by following all the free floor space (do not include the tables or any non-walkable surface in the boundaries and make it as precise as possible).
- Do not forget to have the headset charged.

THE EXPERIMENT:

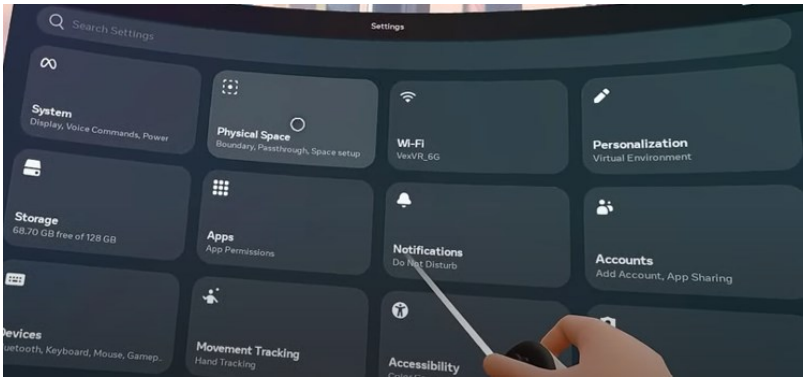
Each participant will have 3 scenarios:

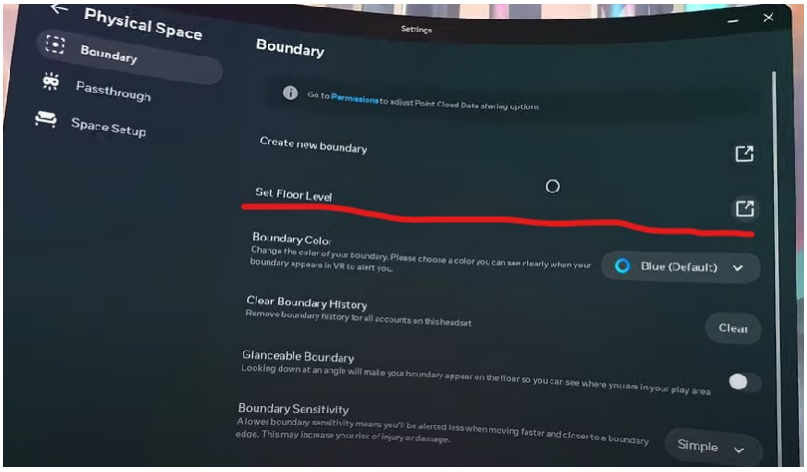
- PROJECT-LOCAL-HTTP

- PROJECT-ADAFRUIT-HTTP

- PROJECT-ADAFRUIT-MQTT

For each participant, set the floor level before starting





FOR THE LOCAL SCENARIO:

- Headset must be connected to the same WiFi as the dispensers.
- No other manoeuvre is needed.

FOR THE ADAFRUIT SCENARIOS:

- Connect headset to a different WiFi than the dispensers

- Open the python virtual environment for each scenario accordingly on the computer connected to the local WiFi of the dispensers (HTTPServer.py or MQTTServer.py depending on the scenario)

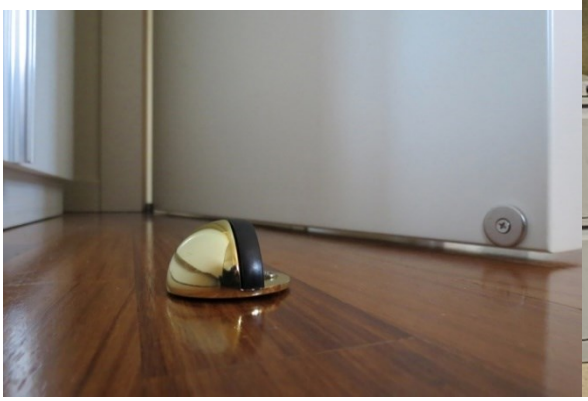
FOR EACH SCENARIO:

- Close the door of the room.

- The boundaries of the room are supposed to be already configurated and must not be changed.

- Now the participants need to calibrate those boundary lines of the real room to correspond exactly to the VR version of the lab. (the calibration must be done again between each scenario)

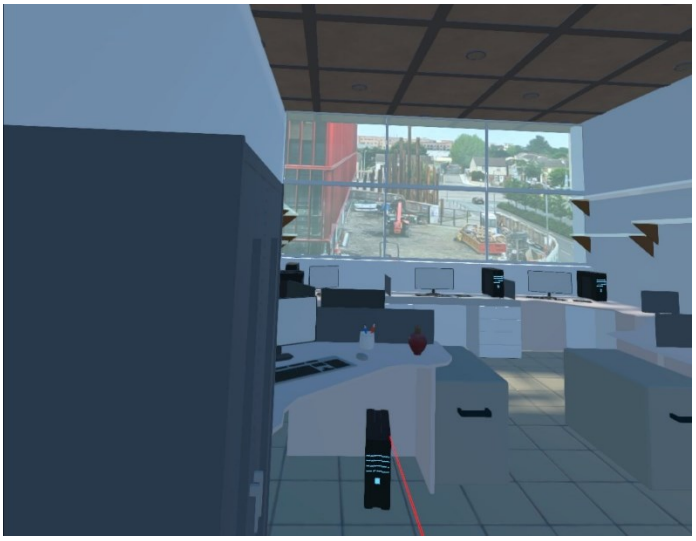
- Participants have to adjust the boundaries of the room with the VR world by starting from the black door stopper, putting their legs joint together and looking forward straight.



The boundaries will appear in a blue grid-like wall

If the boundaries are a little off, the participant can adjust using the Oculus button on the right controller.

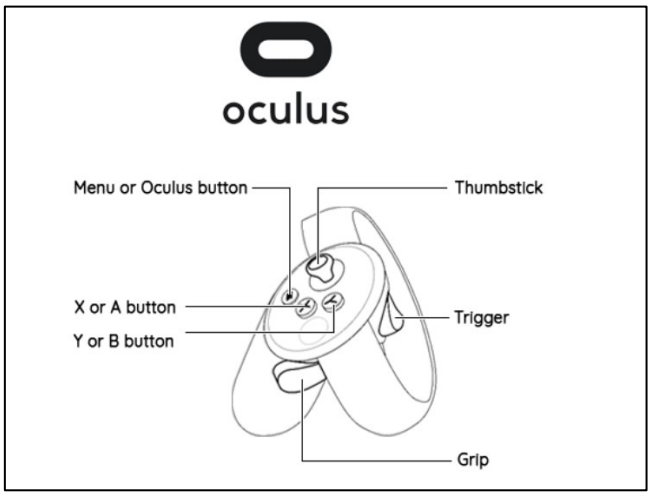
The best indicator to know if the boundaries are in the good place is to look at the closet and see if the grid lines (corresponding to the limits of the real closet) are aligned with it. Other indicators are the table that has the perfume on it and the 2 drawers. Compare the grid lines with the VR environment and adjust until it is precise.



Once experiment starts, the participant MUST NOT USE THE OCULUS BUTTON AGAIN. (It will mess the boundaries).

- To remove the text displayed, Use the index finger button (Trigger)

- To click on the clickable object, Use the middle finger button (Grip)



Now the participant is free to move in the room

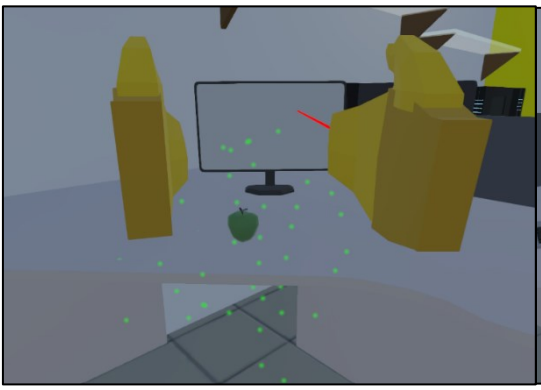
AFTER EACH SCENARIO:

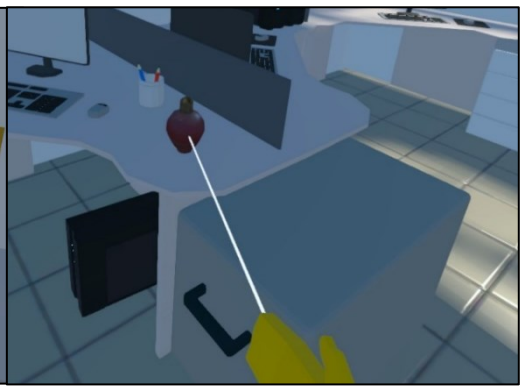
- Fill the corresponding scenario questionnaire.

AFTER EXPERIMENT:

- Fill the General Experience questionnaire.

The objects to interact with are the following:





Appendix G
Consent Form
&
Questionnaire

Informed Consent Form

(Participants over 18, parents and/or legal guardians of participants below the age of 18, or people over 18 who have a guardian)

(What participation in the study involves)

First of all, we will ask you for some basic information keeping in mind anonymity of all data. Then we'll start the activity and at the end we will report back on the activity just performed.

In this research study, DCU researchers focus on delivering an immersive VR experience with the addition of smells throughout the visuals. These smells are related to the content presented. The opinions of yours in relation to the smells used will be asked in a questionnaire, after the consent form is signed.

(Duration)

The duration of this activity may vary from 30 mins to an hour.

(Risks and benefits)

Participation involves no risks of any kind. This study will give us a better understanding of olfaction and VR interaction.

(Compensation)

In this case, no compensation is envisaged for taking part.

(Confidentiality)

If you consent to the participation, your identity will be anonymised. Your name will never be used if case studies need to be presented.

This informed consent form will be kept in a safe place by the principal investigators and will be destroyed in December 2027, which is five years after the end of the project.

(Voluntary participation)

Participation in this study is completely voluntary. There is no penalty for opting not to take part.

(Right to withdraw from the study)

You can withdraw from the activity at any time without giving explanations and with no negative consequences: just by letting us know through any communication channel. As well as this, you can exercise your rights under the European General Data Protection Regulation by making a request to the Irish Data Protection Commission, as well as contacting the researcher (Dr. Anderson Simiscuka (andersonaugusto.simiscuka@dcu.ie)) and DCU Data Protection Unit (data.protection@dcu.ie).

You have the rights to withdraw consent by contacting the researcher (Anderson Simiscuka (andersonaugusto.simiscuka@dcu.ie) or DCU Data Protection Unit (data.protection@dcu.ie).
Contact of DCU's Data Protection Officer – Mr. Martin Ward (data.protection@dcu.ie - Ph: 7005118 / 7008257)

In all cases a written response will be received within the legal time limit, stating what action has been taken.

(Subsequent publication/re-use/other processing of the basic data and conservation period)

The anonymised research data will be made available to other researchers.

(Use of contributions made by the child, who can also give or withhold consent)

I consent to my contributions being quoted literally, provided there is no mention of my name.

Contact person for queries about the study)

If you have any queries, you can contact the following:

Dr. Anderson Simiscuka (andersonaugusto.simiscuka@dcu.ie), Dublin City University, School of Electronic Engineering, Room S220, Glasnevin, Dublin 9, Ireland

(Consent)

- I have read the information about the research project and I have had the opportunity to ask questions, which have been answered to my satisfaction.
- I understand that the anonymised information on this project will be placed at the disposal of other researchers.
- I consent to my voluntary participation and I have received a copy of this consent form.

Full name of the participant: _____

Participant Signature _____Date: _____

Researcher Signature _____Date _____

Data Management Plan

Introduction

A number of ethical and legal requirements apply to the management of research data, particularly where the research involves human subjects. Ethical considerations include the purpose and nature of the research itself, the nature of consent obtained (e.g. opt-in versus opt-out participation), what data need to be safeguarded during analysis, etc.

Much research data about people – even sensitive data – can be shared ethically and legally if researchers employ strategies of informed consent, anonymization and controlled access to data.

Informed consent is an ethical requirement for most research and must be considered and implemented throughout the research lifecycle, from planning to publication. Gaining consent must include making provision for sharing data and take into account any immediate or future uses of data. The procedures implemented to obtain informed consent are reported in the Annex to this document.

Research which involves collecting or processing of personal data, regardless of the method by which they are collected should comply with the EC decisions, the directives of the European Parliament and of the Council as well as on laws on privacy and data protection of the partner countries involved in the project.

Next, information is provided on the procedures that will be implemented for data collection, storage, protection, retention and destruction and confirmation that they comply with national and EU legislation.

Roles and Responsibilities

The primary responsibility for managing the data will belong to the partner which collected the data. If not available, the coordinator will be able to access data and continue the data management process. It is the goal to store the data in secured servers located at the partner premises. The partners back up the data periodically. The data will be collected at the completion of the grant and then the responsibility for the information management will be transferred to the coordinator. The partners will continue to have access to the data via the coordinator. Any data not deemed public will be kept private under strict security.

Expected Data

The data for this project will include but is not limited to: interviews with participants, questionnaires to be answered by community members, focus groups and survey data. The interviews will be stored with video, audio and transcript text files. A summary will be included in English language. Results from user interaction with the systems will include network performance data, quality of service and quality of experience data, generated content, etc. The surveys will include survey answers and demographic information. Transcribed text will be associated to the video files. No fewer than two copies of the data will be kept at all times. In addition, data for public access will be anonymised and stripped of identifying information. The process of anonymization will be documented.

Period of Data Retention

The new data collected by partners will be available for a 5-year period after completion of the grant, in December 2027. After this period the data will be destroyed, both in electronic and paper format.

Data Formats and Metadata

There will be two file types: text and spreadsheets. The files will include anonymisation of participant information using IDs. We will use a naming system for the files that will be associated with metadata. For the files uploaded

in the coordinator servers, we will include a complete descriptor with each file. Participant responses will be collated in spreadsheets.

Data Dissemination and Policies for Public Access, Sharing and Publication Delays

It is the goal to store the data on servers at a secured location in the premises of the partner who was in charge with collecting the data. The data will be archived at grant completion. The individual's privacy will be ensured by removing any identifiable information. Some of the data collected will be publicly available via our websites. The access to the data will comply with the all national and EC requirements. Although we will publish findings based on the data, it is hoped that the data will be a rich source of inquiry for anyone interested in the research. Any reference to data should be made explicit in any third party publication or dissemination, including grant agreement number.

Data Storage and Preservation of Access

The data will be stored on medium term in the partner-responsible secure servers and on long term on coordinator servers.

Plain Language Statement for Participants, Tutors, Parents

I. Introduction to the Research Study

In this VR experience, DCU researchers focus on the assessment of the environment enhanced with olfaction technology.

II. Details of involvement in the Research Study

In this research study a VR environment was enhanced with the addition of smells throughout the experience. These smells are related to the content presented, so the participants can relate the scenarios to the scents of the environment. The opinions of yours in relation to the experience and the smells used in the environment will be asked in a questionnaire, after the consent form is signed.

III. Potential risks to participants from involvement in the Research Study

The research will mainly require participants to view the 3D environments using a VR headset, with the added olfaction technology and completion of questionnaires. Therefore, it is not envisaged that there are any risks to participants arising from involvement in these research studies.

IV. Benefits (direct or indirect) to participants from involvement in the Research Study

The participants will be exposed to a highly interactive material, with an experience for young people that is thought provoking, exciting and challenging. The use of multisensorial technology aims to enhance the experience with an extra layer of immersion.

V. Data Protection Advice and GDPR Compliance

- Participants' privacy will be ensured by removing any personal identifiable information.
- The data will be stored on DCU-located secure servers. Data collected via paper questionnaires will be stored DCU premises, in a cabinet drawer locked with key, in a room with restrict access via swipe card.
- DCU is the Data Controller (Researcher: Dr. Anderson Simiscuka (andersonaugusto.simiscuka@dcu.ie), part of the Performance Engineering Laboratory, School of Electronic Engineering).
- Contact of DCU's Data Protection Officer – Mr. Martin Ward (data.protection@dcu.ie - Ph: 7005118 / 7008257)
- The data collected in this questionnaire will provide input to the effectiveness of the use of scents in VR experiences, if it increases the interest in the technology and is well received by the participants.
- The data is anonymised, including the personal data related to health and ethnicity.
- The retention period of the data is until December 2027 (The duration of the EU project, plus 5 years after it).
- Participants have the right to lodge a complaint with the Irish Data Protection Commission, as well as the right to access their own personal data by contacting the researcher and DCU Data Protection Unit (data.protection@dcu.ie).
- You have the rights to withdraw consent by contacting the researcher (Dr. Anderson Simiscuka (andersonaugusto.simiscuka@dcu.ie) or DCU Data Protection Unit (data.protection@dcu.ie).
- By contacting the researcher, you can also access the findings of the research (with data processed, anonymised and summarised) in the form of a scientific publication

VI. Statement that involvement in the Research Study is voluntary

You may withdraw from the participation of the activity at any point. There will be no penalty for withdrawing before all stages of the Research Study have been completed.

VII. Other relevant information

The progress made on the project and any updates that relate to this project can be followed by the participants via the lab website www.eeng.dcu.ie/~pel or Twitter page: https://twitter.com/pelatdcu or by contacting the researcher directly (Dr. Anderson Simiscuka (andersonaugusto.simiscuka@dcu.ie))

If participants have concerns about this study and wish to contact an independent person, please contact: The Secretary, Dublin City University Research Ethics Committee, c/o Research and Innovation Support, Dublin City University, Dublin 9. Tel 01-7008000, e-mail rec@dcu.ie

DEMOGRAPHICS QUESTIONNAIRE

PARTICIPANT ID:_____

1. **Gender**

□ Male
□ Female
□ Other
□ Prefer not to say

2. **Town/City:** ……

3. **Age:** …….

4. **In terms of ethnicity, how would you define yourself?**

□ Irish
□ White / Caucasian
□ Black / African American / Black Irish
□ Hispanic or Latino
□ Asian / Asian American / Asian Irish
□ Eastern Asian (Chinese, Japanese, Korean, etc.)
□ South Asian (Indian, Pakistani, Bangladeshi, etc.)
□ Southeast Asian (Filipino, Vietnamese, Thai, etc.)
□ Arab / Middle Eastern
□ Other, including mixed group/background
□ Prefer not to say

5. **Are you a person with disabilities?**

□ Yes
□ No
□ Prefer not to say

6. **What is your job/profession?**

□ Student
□ ICT* professional
□ Retired
□ N/A
□ Other: ……

*ICT = Information Communication Technology

7. **Is today the first time you use Virtual Reality?**

□ Yes
□ No

USER QUESTIONNAIRE (VIA GOOGLE FORMS)

Three Scenarios: Communications between VR headset and olfaction devices using 1) Local Network, 2) Cloud using REST and 3) Cloud using MQTT, in different orders.

Participant ID: _____

Please complete the following questionnaire after finishing different parts of the experiment. You will have to answer each question by choosing ONE item among the options. Circle or highlight your answers.

A. Fill after Scenario 1:

1. Scents were properly synchronised with my interactions in the VR environment.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

2. Scents started quickly after interacting with or approaching the scent-related objects.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

3. The scents seemed realistic to me in terms of Delay.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

4. The scents seemed realistic to me in terms of Intensity.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

5. The scents seemed realistic to me in terms of Persistency/Lingering time.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

6. I felt some scents were triggered when they shouldn't.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

7. I found that the smells were diffused at the right moments.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

8. I enjoyed that some scents were triggered by distance (apple and cookie)?

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

9. I enjoyed that some scents were triggered by interaction (perfume)?

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

B. Fill after Scenario 2:

1. Scents were properly synchronised with my interactions in the VR environment.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

2. Scents started quickly after interacting with or approaching the scent-related objects.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

3. The scents seemed realistic to me in terms of Delay.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

4. The scents seemed realistic to me in terms of Intensity.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

5. The scents seemed realistic to me in terms of Persistency/Lingering time.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

6. I felt some scents were triggered when they shouldn't.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

7. I found that the smells were diffused at the right moments.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

8. I enjoyed that some scents were triggered by distance (apple and cookie)?

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

9. I enjoyed that some scents were triggered by interaction (perfume)?

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

C. Fill after Scenario 3:

1. Scents were properly synchronised with my interactions in the VR environment.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

2. Scents started quickly after interacting with or approaching the scent-related objects.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

3. The scents seemed realistic to me in terms of Delay.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

4. The scents seemed realistic to me in terms of Intensity.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

5. The scents seemed realistic to me in terms of Persistency/Lingering time.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

6. I felt some scents were triggered when they shouldn't.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

7. I found that the smells were diffused at the right moments.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

8. I enjoyed that some scents were triggered by distance (apple and cookie)?

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

9. I enjoyed that some scents were triggered by interaction (perfume)?

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

General experience (Fill after all scenarios and previous questionnaires):

1. I enjoyed the use of the smells during the VR experience.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

2. The smells are relatable to the visuals.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

3. I could easily recognize the smells that were diffused.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

4. I found the following scents to be **disturbing or unpleasant**:

Apple.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

Cookie.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

Perfume.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

5. Which trigger type you found to be the most immersive and realistic (in terms of delays, intensity, scent lingering, etc...)?

Scent triggered by distance	Scent triggered by interaction

6. The virtual room closely resembles the actual lab in terms of visual quality.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

7. The addition of scents enhanced the realism of the VR environment.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

8. The smells were a distraction or hindrance from the visual experience.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

9. Rank from 1st to 3rd (where 1st is the best) which scenario you preferred in terms of immersion and realism during the scent experience (in terms of delays, intensity, persistency/lingering time, etc...)

   Scenario 1 _____ Scenario 2 _____ Scenario 3 _____

10. Rank from 1st to 3rd (where 1st is the best) which scenario did you feel had the least immersive & realistic scent experience.

    Scenario 1 _____ Scenario 2 _____ Scenario 3 _____

11. I felt dizziness / motion sickness during the virtual experience.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

12. The smells helped in making the VR experience more immersive.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

13. The smells helped in making this experience enjoyable.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

14. I would recommend this experience to other people.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

Open Questions : (Optional)

15. What did you like most about the experience? (Optional)

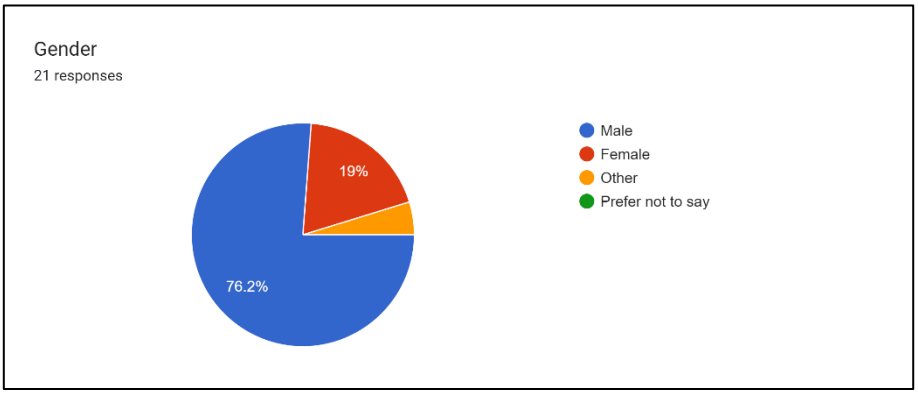
16. What did you dislike the most about the experience? (Optional)

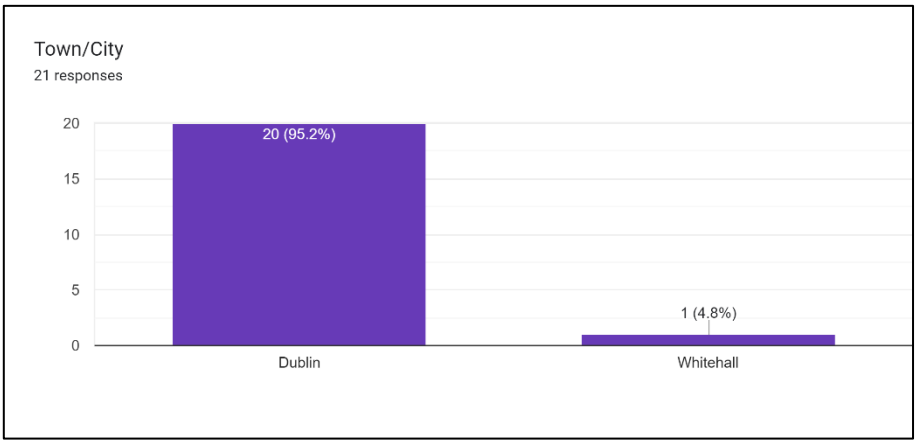
17. Do you have any suggestions for improving the integration of scents in VR? (Optional)

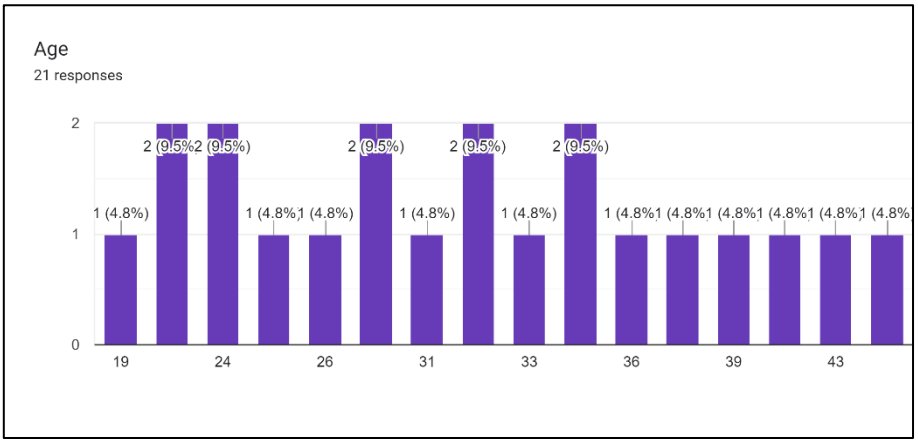
Appendix H
Questionnaire results

QUESTIONNAIRE RESULTS

DEMOGRAPHICS QUESTIONNAIRE RESULTS

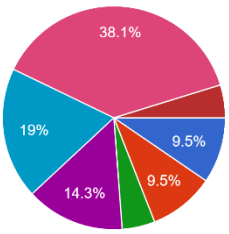






In terms of ethnicity, how would you define yourself?

21 responses

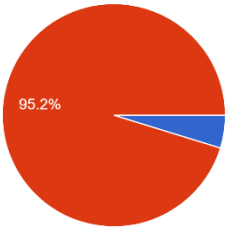


- Irish
- White/Caucasian
- Black / African American / Black Irish
- Hispanic or Latino
- Asian / Asian American / Asian Irish
- Eastern Asian (Chinese, Japanese, K…
- South Asian (Indian, Pakistani, Bangla…
- Southeast Asian (Filipino, Vietnamese…

▲ 1/2 ▼

Are you a person with disabilities?

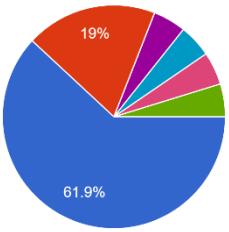
21 responses



- Yes
- No
- Prefer not to say

What is your job/profession?

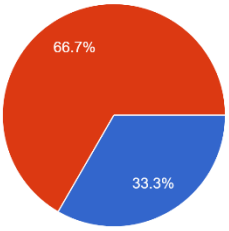
21 responses



- Student
- Information Communication Technology (ICT) professional
- Retired
- N/A
- Postdoctoral Researcher
- Lecturer
- Administrator
- Operations

Is today the first time you use Virtual Reality?

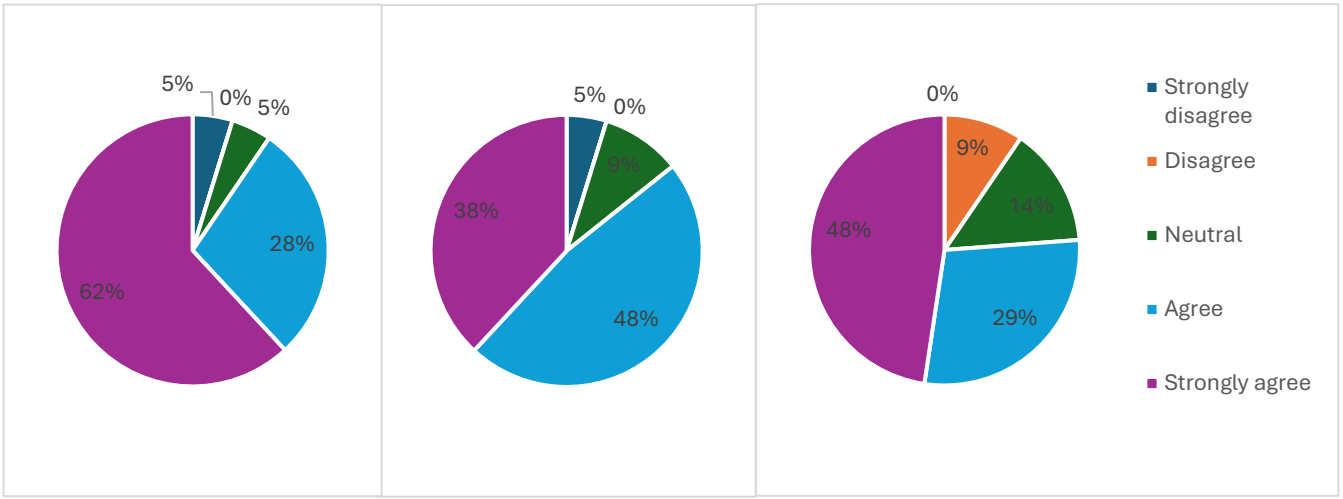
21 responses



- Yes
- No

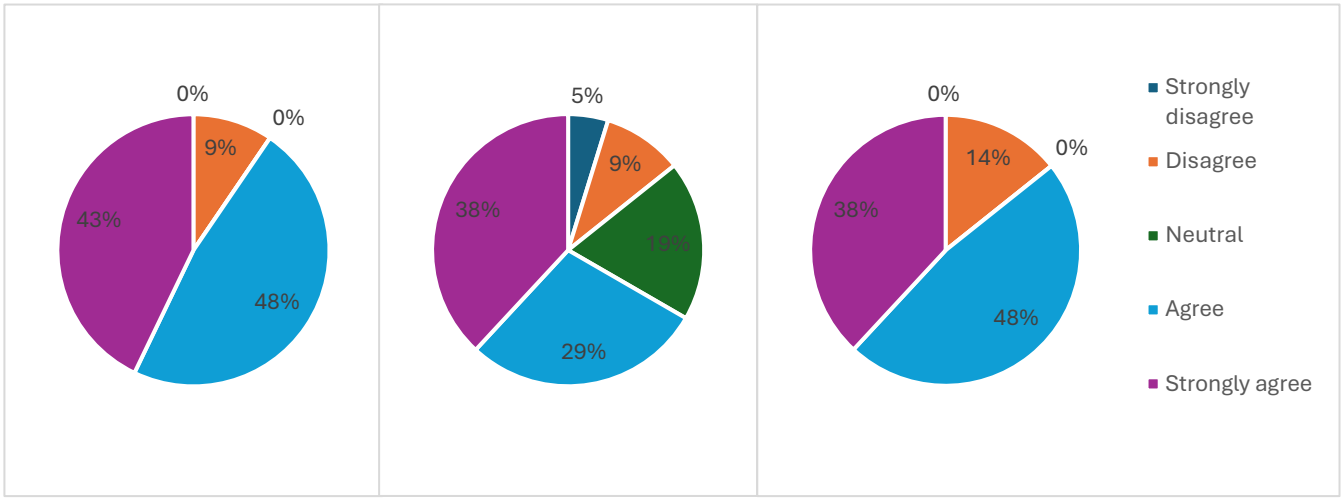
SCENARIOS RESULTS

Scents were properly synchronised with my interactions in the VR environment.



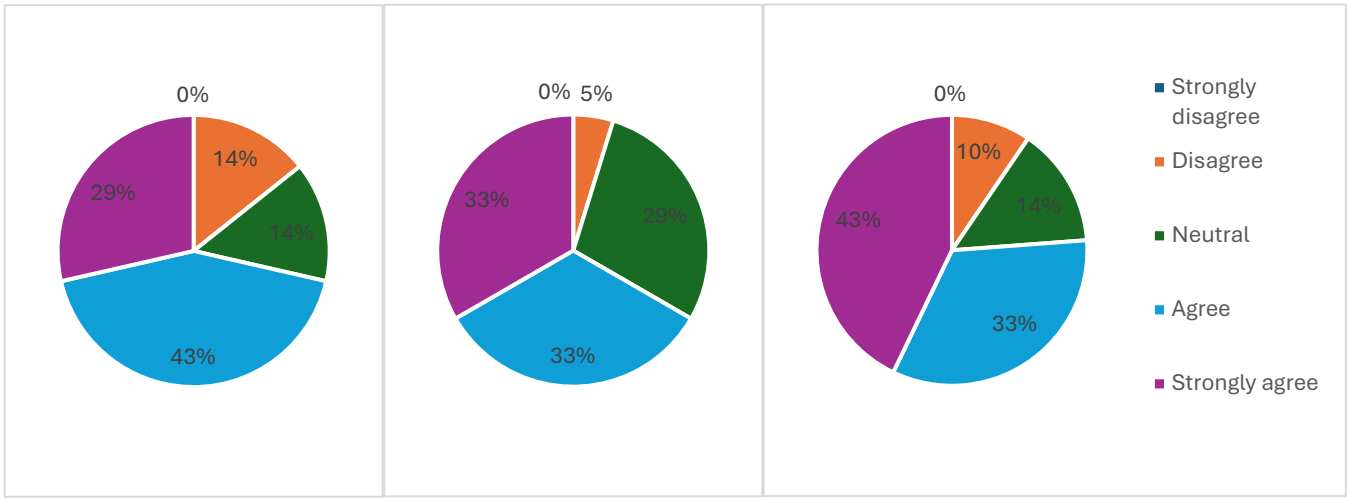
Local-based **Cloud-based (HTTP)** **Cloud-based(MQTT)**

Scents started quickly after interacting with or approaching the scent-related objects.



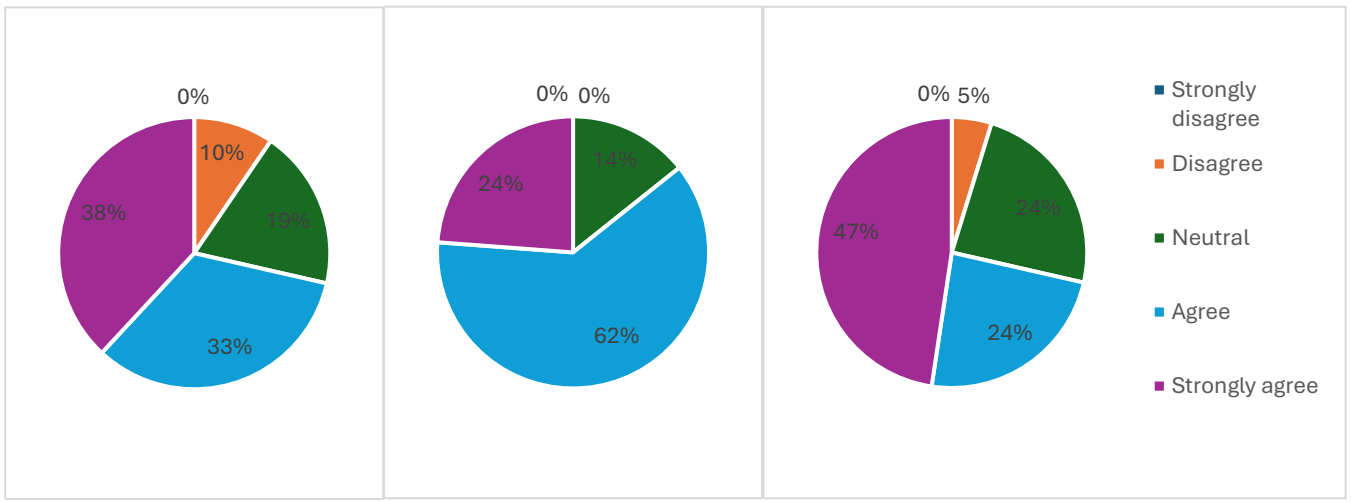
Local-based **Cloud-based (HTTP)** **Cloud-based(MQTT)**

The scents seemed realistic to me in terms of Delay.



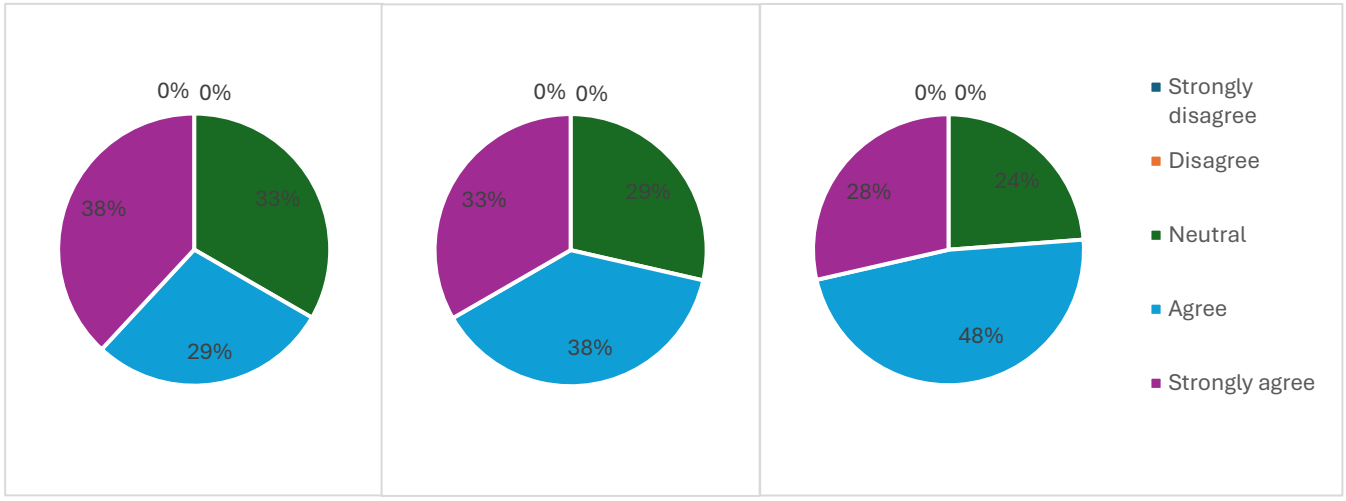
Local-based **Cloud-based (HTTP)** **Cloud-based(MQTT)**

The scents seemed realistic to me in terms of Intensity.



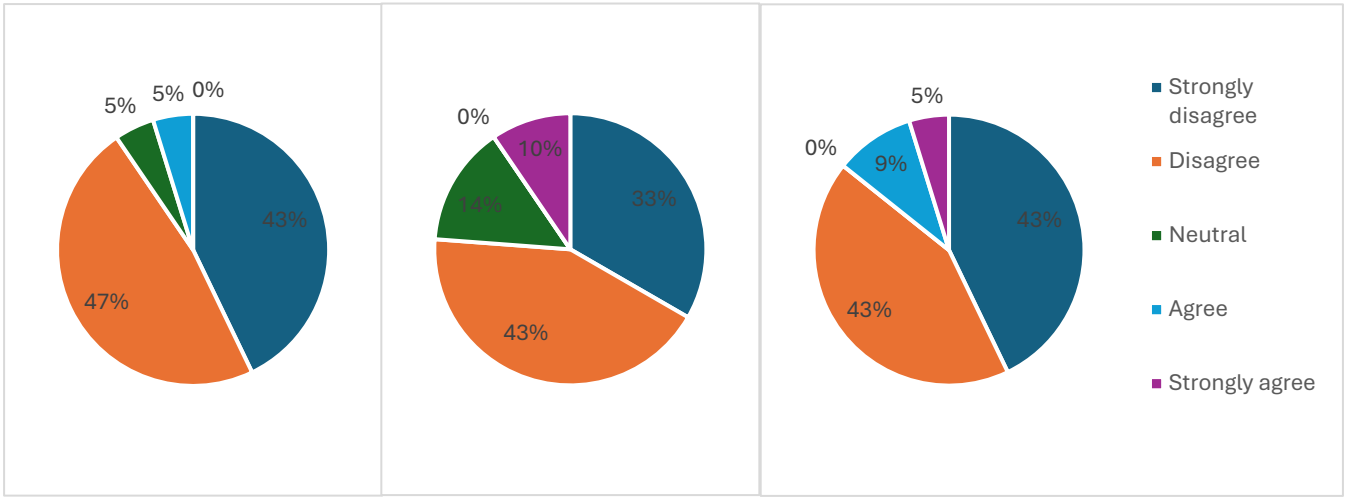
Local-based **Cloud-based (HTTP)** **Cloud-based(MQTT)**

The scents seemed realistic to me in terms of Persistency/Lingering time.



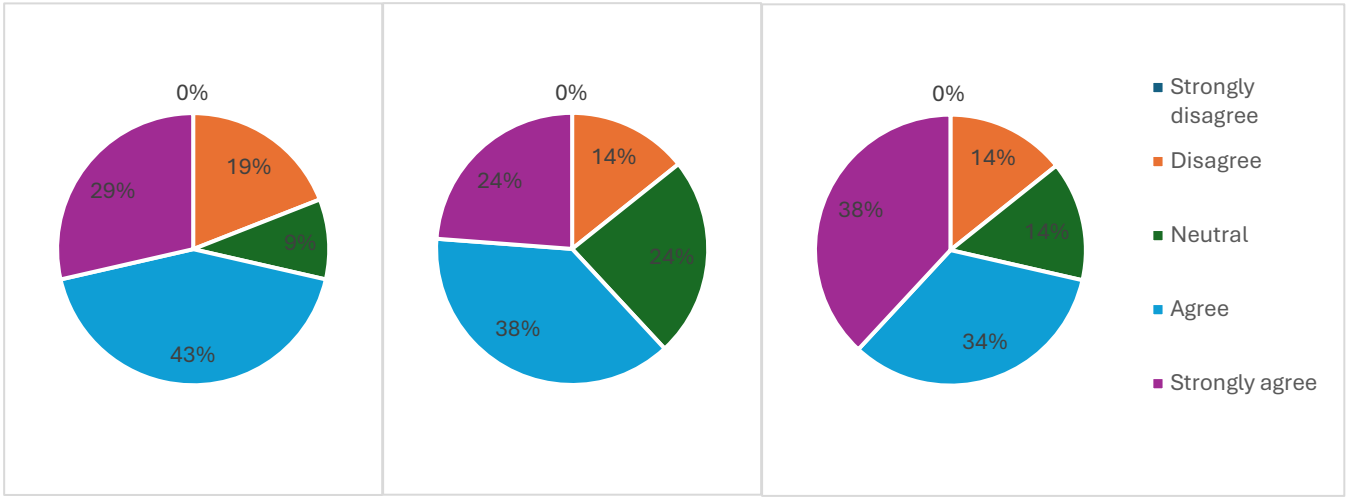
Local-based **Cloud-based (HTTP)** **Cloud-based(MQTT)**

I felt some scents were triggered when they shouldn't.



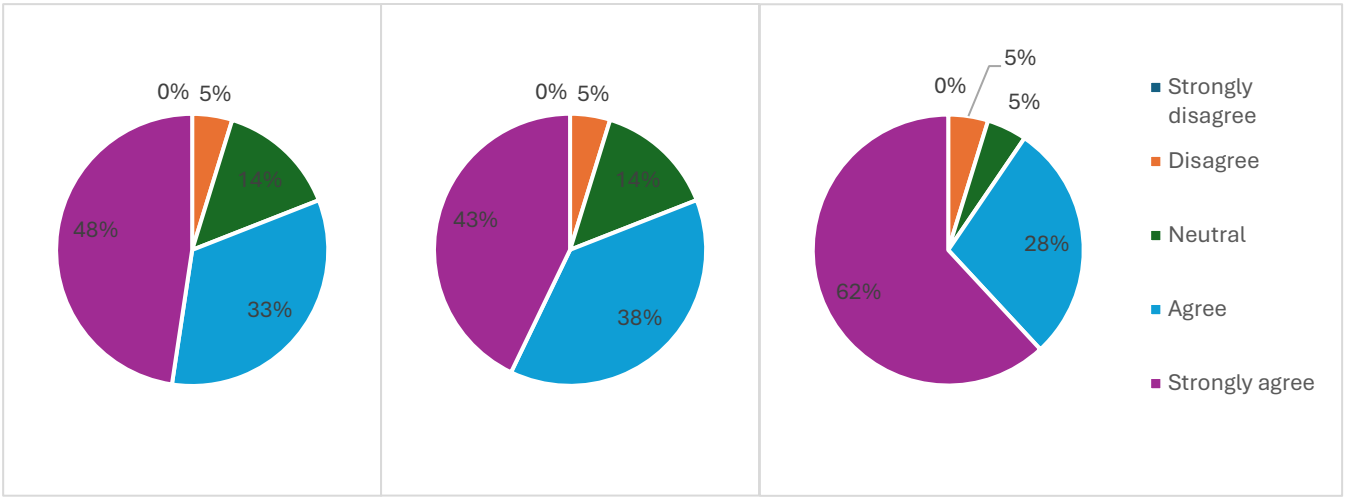
Local-based **Cloud-based (HTTP)** **Cloud-based(MQTT)**

I found that the smells were diffused at the right moments.



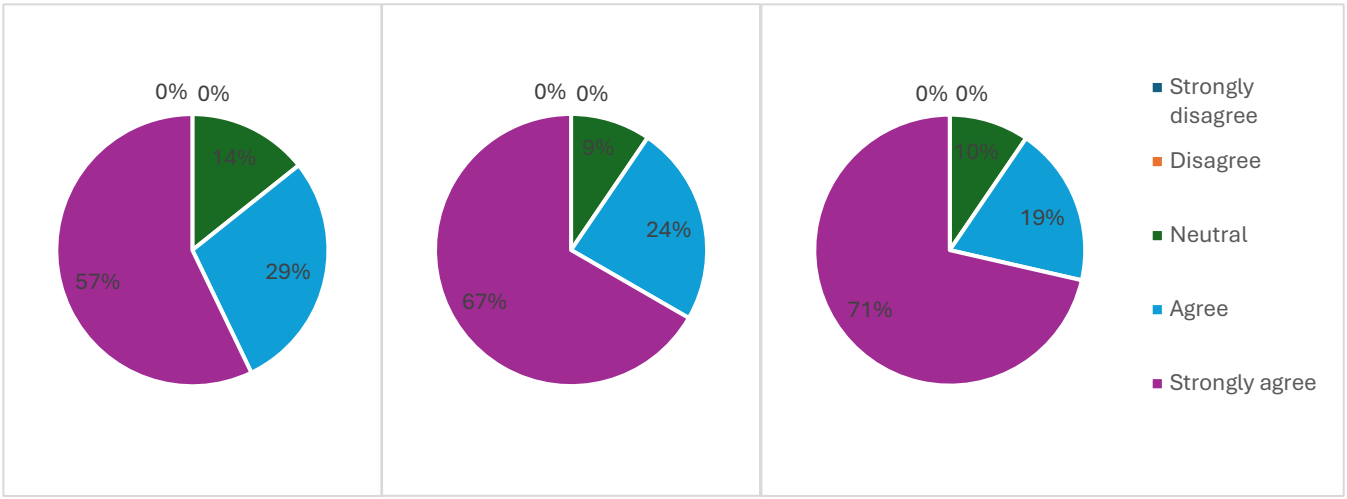
Local-based **Cloud-based (HTTP)** **Cloud-based(MQTT)**

I enjoyed that some scents were triggered by distance (apple and cookie)?



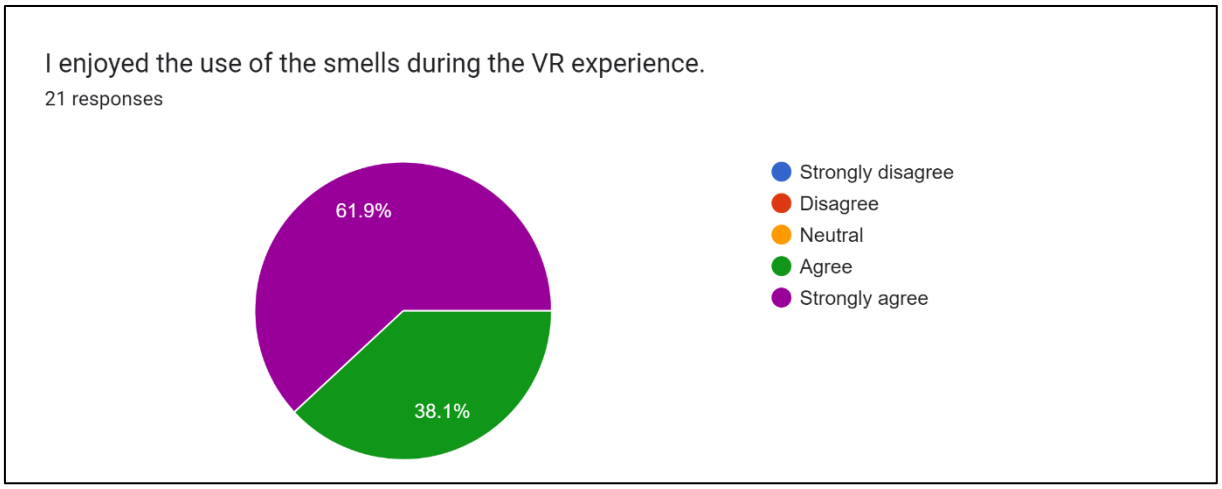
Local-based **Cloud-based (HTTP)** **Cloud-based(MQTT)**

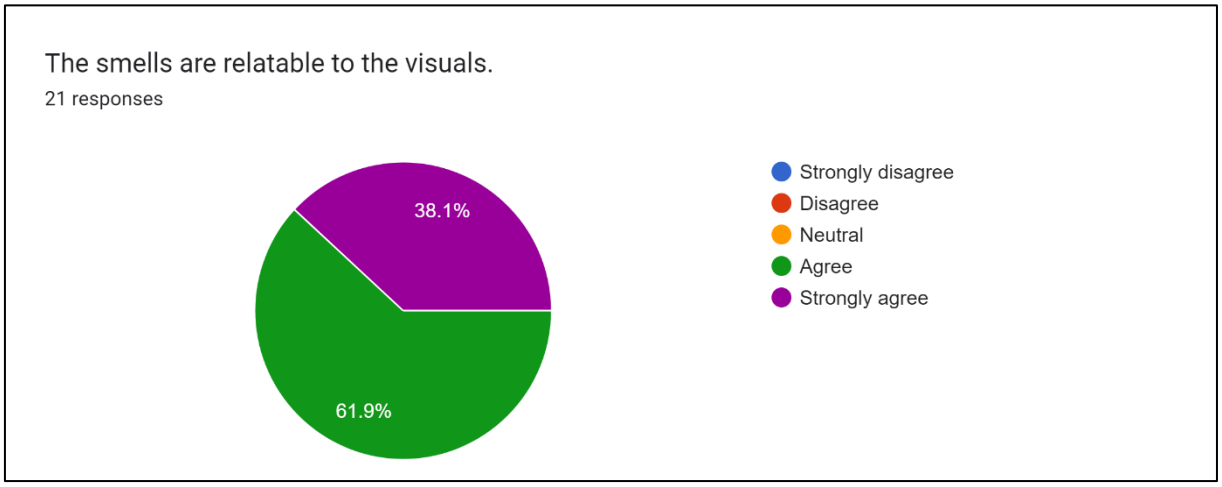
I enjoyed that some scents were triggered by interaction (perfume)?



Local-based **Cloud-based (HTTP)** **Cloud-based(MQTT)**

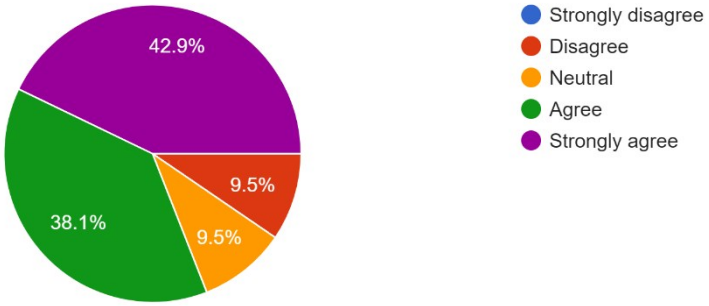
GENERAL EXPERIENCE QUESTIONNAIRE RESUTLS



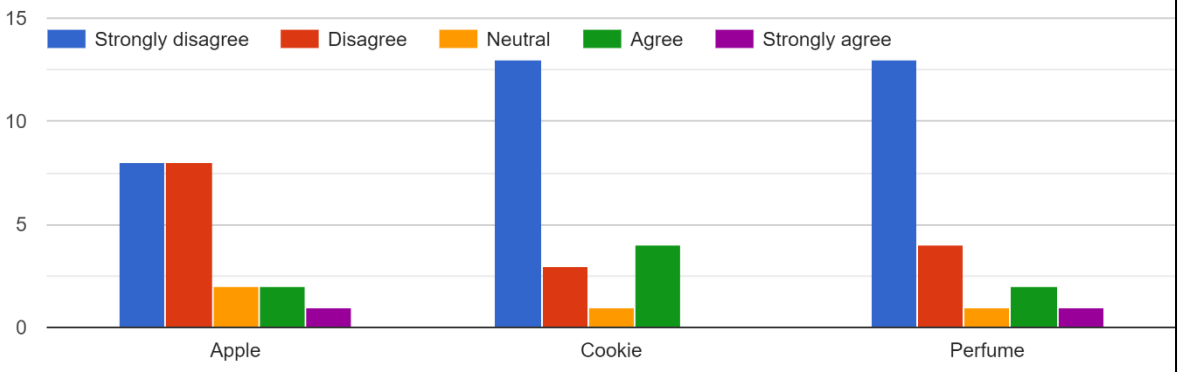


I could easily recognize the smells that were diffused.

21 responses

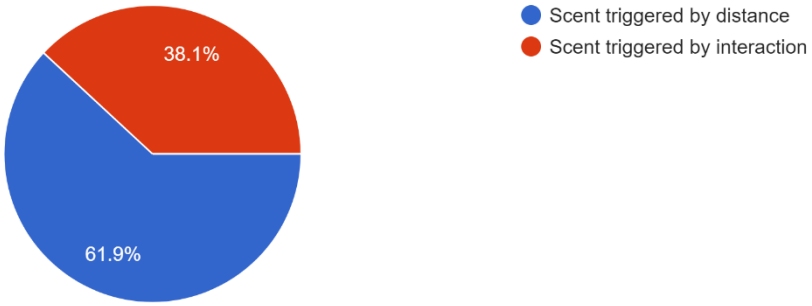


I found the following scents to be disturbing or unpleasant:



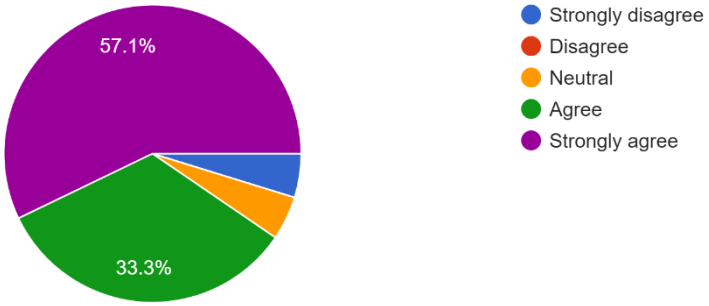
Which trigger type you found to be the most immersive and realistic (in terms of delays, intensity, scent lingering, etc…)?

21 responses



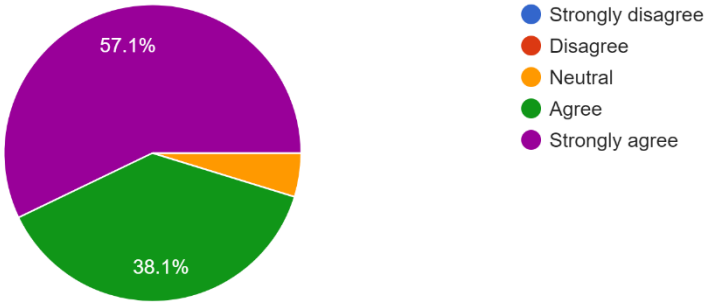
The virtual room closely resembles the actual lab in terms of visual quality.

21 responses



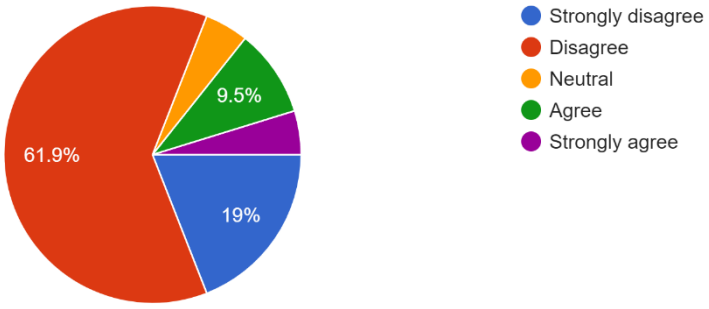
The addition of scents enhanced the realism of the VR environment.

21 responses



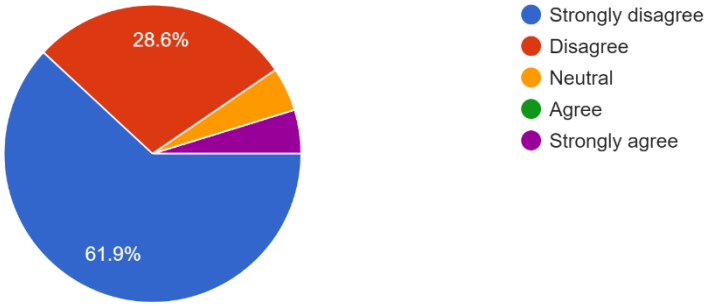
The smells were a distraction or hindrance from the visual experience.

21 responses



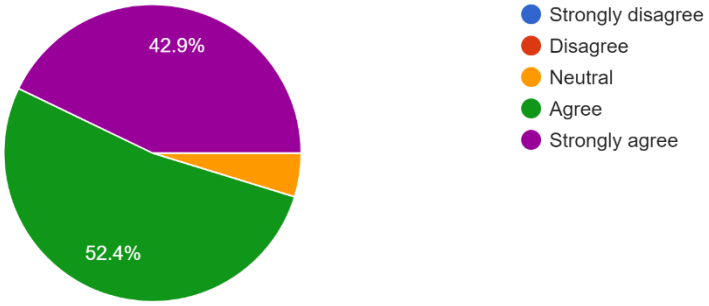
I felt dizziness / motion sickness during the virtual experience.

21 responses



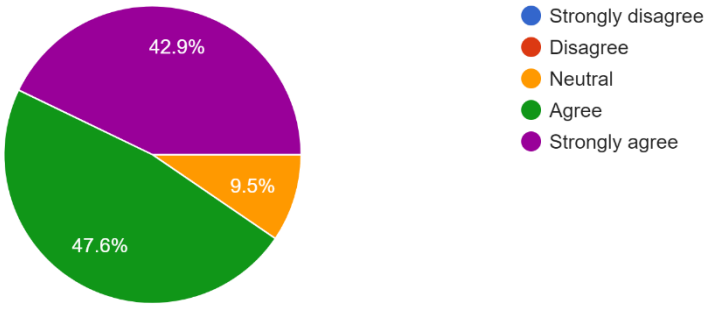
The smells helped in making the VR experience more immersive.

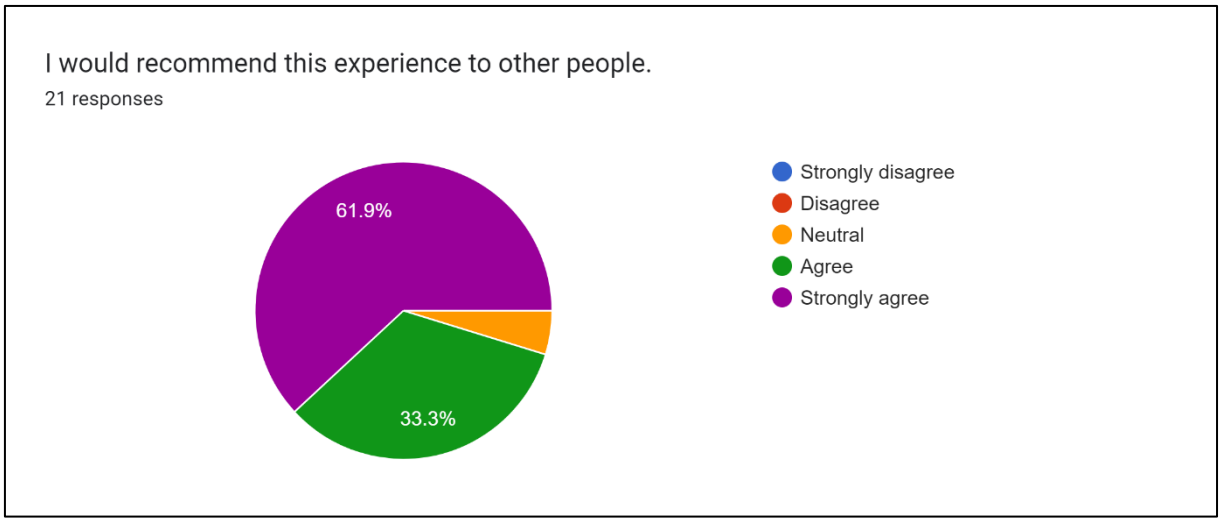
21 responses

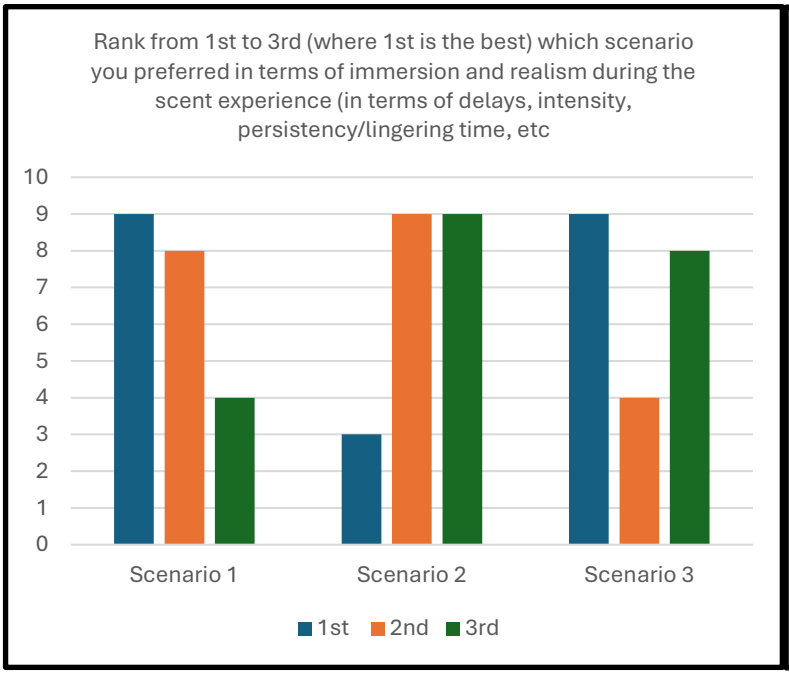


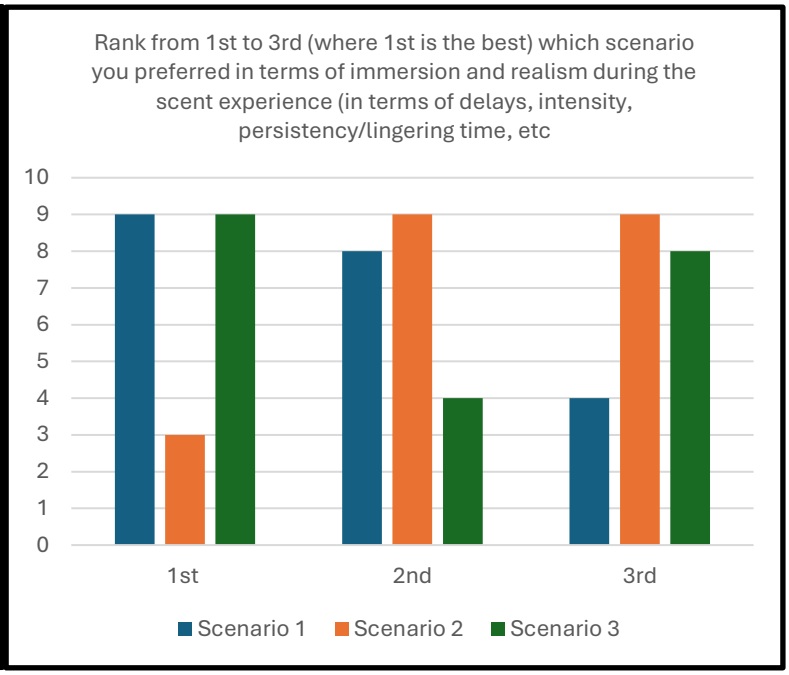
The smells helped in making this experience enjoyable.

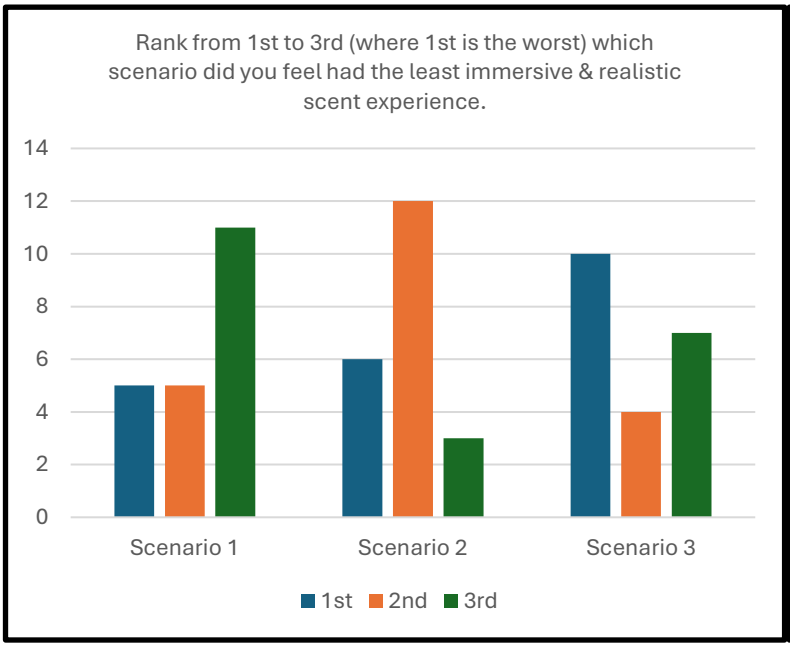
21 responses

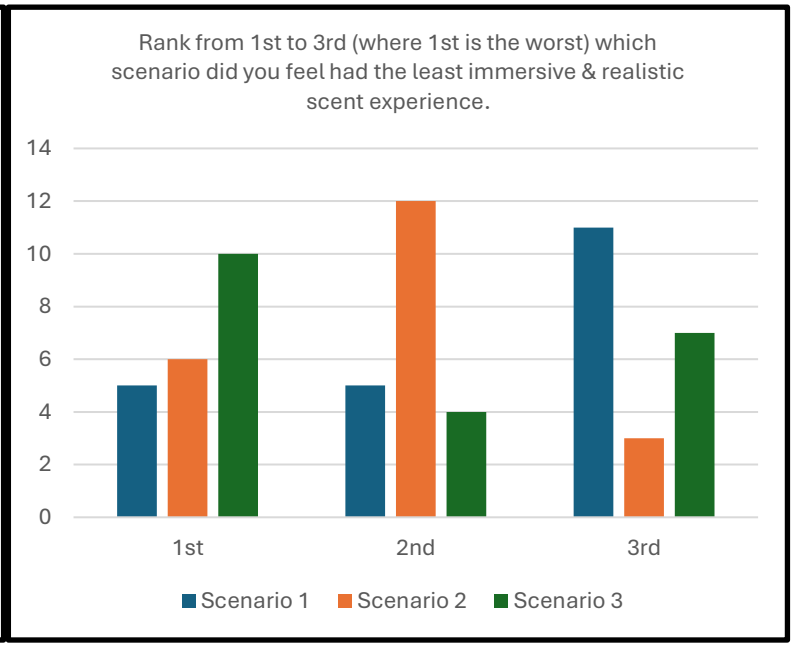



I would recommend this experience to other people.
21 responses
61.9%
33.3%
Strongly disagree
Disagree
Neutral
Agree
Strongly agree


Rank from 1st to 3rd (where 1st is the best) which scenario you preferred in terms of immersion and realism during the scent experience (in terms of delays, intensity, persistency/lingering time, etc
10
9
8
7
6
5
4
3
2
1
0
Scenario 1
Scenario 2
Scenario 3
1st
2nd
3rd


Rank from 1st to 3rd (where 1st is the best) which scenario you preferred in terms of immersion and realism during the scent experience (in terms of delays, intensity, persistency/lingering time, etc
10
9
8
7
6
5
4
3
2
1
0
1st
2nd
3rd
Scenario 1
Scenario 2
Scenario 3


Rank from 1st to 3rd (where 1st is the worst) which scenario did you feel had the least immersive & realistic scent experience.
14
12
10
8
6
4
2
0
Scenario 1
Scenario 2
Scenario 3
1st
2nd
3rd


Rank from 1st to 3rd (where 1st is the worst) which scenario did you feel had the least immersive & realistic scent experience.
14
12
10
8
6
4
2
0
1st
2nd
3rd
Scenario 1
Scenario 2
Scenario 3

Appendix I
C# scripts
source code

Local-based method

OlfactionObject.cs:

```
using System;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.XR.Interaction.Toolkit;

public enum TriggerType
{
    Click,
    Distance
}

public class OlfactionObject : MonoBehaviour
{
    private XRBaseInteractable interactable;
    public ParticleSystem perfumeParticles;

    [Tooltip("The olfactive object needs a DataSender component to send the scent parameters to the dispenser")]
    public DataSender dataSender; // DataSender instance

    [Tooltip("IP Address of the dispenser")]
    public string deviceIP; // Setting DeviceIP manually
    public TriggerType triggerType; // Type of trigger to use (proximity or interaction)

    [Tooltip("Player's transform (Main camera in the XR Rig)")]
    public Transform playerTransform; // Player's transform for distance calculation

    [Tooltip("Minimum distance at which distance trigger type is activated")]
    public float triggerDistance; // Trigger distance for distance-based interactions


    // Function to calculate intensity and duration based on distance
    public string ScentParametersUrl(float distance)
    {
        // Calculate intensity and duration based on distance
        int intensity = Mathf.CeilToInt(3 - 2 * (distance / triggerDistance));
        Debug.Log("Intensity: " + intensity);
        float duration = 2;
        string intensity_level = "";
        if (intensity == 1) {
            intensity_level = "low";
        }
        if (intensity == 2) {
            intensity_level = "medium";
        }
        if (intensity == 3){
            intensity_level = "high";
        }
        string deviceURL = "http://"+deviceIP+"/diffuse?duration=" + duration + "&intensity=" + intensity_level;
        return deviceURL;
    }

    void Start()
    {
        if (playerTransform == null)
        {
            Debug.LogWarning("Player Transform is not set. Please assign it in the inspector.");
        }
        if (dataSender == null)
        {
            // Attempt to find an object with a DataSender component
```

```
            dataSender = FindObjectOfType<DataSender>();
            if (dataSender == null)
            {
                Debug.LogWarning("Data Sender is not set. Please assign it in the inspector.")
                    ;
            }
        }
        if (deviceIP == null)
        {
            Debug.LogWarning("Device IP is not set. Please assign it in the inspector.");
        }

        // Get the XRBaseInteractable component for handling interactions
        if (triggerType == TriggerType.Click) {
            interactable = GetComponent<XRSimpleInteractable>();
            if (interactable != null)
            {
                // Subscribe to the SelectEntered event to detect when interaction-based
                    object is selected
                interactable.selectEntered.AddListener(OnSelectEntered);
            }
        }

    }

    void Update()
    {
        if (triggerType == TriggerType.Distance)
        {
            // Calculate the distance between the player and this object
            float currentDistance = Vector3.Distance(playerTransform.position, transform.
                position);
            // Check if the distance is less than the trigger distance
            if (currentDistance <= triggerDistance)
            {
                Debug.Log("Distance: " + currentDistance);
                dataSender.GetRequest(ScentParametersUrl(currentDistance));
                perfumeParticles.Play();
                Debug.Log(ScentParametersUrl(currentDistance));
            }
        }
    }

    void OnSelectEntered(SelectEnterEventArgs args) // This function is called when the user
        clicks on the object
    {
        if (triggerType == TriggerType.Click)
        {
            string deviceURL = "http://"+deviceIP+"/diffuse?duration=3&intensity=high";
            Debug.Log(deviceURL);
            dataSender.GetRequest(deviceURL);
        }
    }

}
```

DataSender.cs:

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

public class DataSender : MonoBehaviour
{
    private bool isSendingRequest = false;

    public void GetRequest(string url)
```

```
    {
        if (!isSendingRequest)
        {
            StartCoroutine(GetRequestCoroutine(url));
        }
    }

    IEnumerator GetRequestCoroutine(string url)
    {
        isSendingRequest = true;

        using (UnityWebRequest request = UnityWebRequest.Get(url))
        {

            yield return request.SendWebRequest();

            if (request.result == UnityWebRequest.Result.ConnectionError || request.result ==
                UnityWebRequest.Result.ProtocolError)
            {
                Debug.LogError(request.error);
            }
            else
            {
                Debug.Log("Response: " + request.downloadHandler.text);
            }
        }

        yield return new WaitForSeconds(0.7f); // Wait for 0.7 seconds before allowing another
             request to avoid overload

        isSendingRequest = false;
    }
}
```

Cloud-based method (Adafruit IO HTTP)

OlfactionObject.cs:

```
using System;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.XR.Interaction.Toolkit;


public enum TriggerType
{
    Click,
    Distance
}

public class OlfactionObject : MonoBehaviour
{
    private XRBaseInteractable interactable;
    public ParticleSystem perfumeParticles;

    [Tooltip("The olfactive object needs a DataSender component to send the scent parameters to the dispenser")]
    public DataSender dataSender; // DataSender instance

    [Tooltip("IP Address of the dispenser")]
    public string deviceIP; // Setting DeviceIP manually
    public TriggerType triggerType; // Type of trigger to use (proximity or interaction)

    [Tooltip("Player's transform (Main camera in the XR Rig)")]
    public Transform playerTransform; // Player's transform for distance calculation

    [Tooltip("Minimum distance at which distance trigger type is activated")]
    public float triggerDistance; // Trigger distance for distance-based interactions


    // Function to calculate intensity and duration based on distance
    public (string, string) ScentParametersUrl(float distance)
    {
        // Calculate intensity and duration based on distance
        int intensity = Mathf.CeilToInt(3 - 2 * (distance / triggerDistance));
        Debug.Log("Intensity: " + intensity);
        string duration = "2.2";
        string intensity_level = "";
        if (intensity == 1) {
            intensity_level = "low";
        }
        if (intensity == 2) {
            intensity_level = "medium";
        }
        if (intensity == 3){
            intensity_level = "high";
        }
        return (duration, intensity_level);
    }

    void Start()
    {
        if (playerTransform == null)
        {
            Debug.LogWarning("Player Transform is not set. Please assign it in the inspector.");
        }
        if (dataSender == null)
        {
            // Attempt to find an object with a DataSender component
            dataSender = FindObjectOfType<DataSender>();
```

```
            if (dataSender == null)
            {
                Debug.LogWarning("Data Sender is not set. Please assign it in the inspector.");
            }
        }
        if (deviceIP == null)
        {
            Debug.LogWarning("Device IP is not set. Please assign it in the inspector.");
        }

        // Get the XRBaseInteractable component for handling interactions
        if (triggerType == TriggerType.Click) {
            interactable = GetComponent<XRSimpleInteractable>();
            if (interactable != null)
            {
                // Subscribe to the SelectEntered event to detect when interaction-based object is selected
                interactable.selectEntered.AddListener(OnSelectEntered);
            }
        }

    }

    void Update()
    {
        if (triggerType == TriggerType.Distance)
        {
            // Calculate the distance between the player and this object
            float currentDistance = Vector3.Distance(playerTransform.position, transform.position);
            // Check if the distance is less than the trigger distance
            if (currentDistance <= triggerDistance)
            {
                var (duration, intensity) = ScentParametersUrl(currentDistance);
                dataSender.GetRequest(deviceIP, duration, intensity);
                perfumeParticles.Play();
            }
        }
    }


    void OnSelectEntered(SelectEnterEventArgs args) // This function is called when the user clicks on the object
    {
        if (triggerType == TriggerType.Click)
        {
            string duration = "3";
            string intensity = "high";
            dataSender.GetRequest(deviceIP, duration, intensity);

        }
    }

}
```

DataSender.cs:

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

public class DataSender : MonoBehaviour
{
    private bool isSendingRequest = false; // Flag to prevent multiple messages from being sent at the same time

```

```
    // Adafruit IO credentials
    public string adafruitIOKey = "aio_fIOc56tr4A5Hjh77X0SXwA5VBapw";
    public string adafruitIOUsername = "MaroBZ";
    public string feedName = "exhalia-http";

    public void GetRequest(string deviceIP, string duration, string intensity)
    {
        if (!isSendingRequest)
        {
            StartCoroutine(SendToAdafruitIO(deviceIP, duration, intensity));
        }
    }

    IEnumerator SendToAdafruitIO(string deviceIP, string duration, string intensity)
    {
        isSendingRequest = true;

        string url = $"https://io.adafruit.com/api/v2/{adafruitIOUsername}/feeds/{feedName}/
            data";
        string innerJson = $"{{\"deviceIP\":\"{deviceIP}\",\"duration\":\"{duration}\",\"
            intensity\":\"{intensity}\"}}";
        string json = $"{{\"value\":\"{innerJson.Replace("\"", "\\\"")}\"}}";
        Debug.Log("URL: " + url);
        Debug.Log("JSON: " + json);

        using (UnityWebRequest request = UnityWebRequest.PostWwwForm(url, "POST"))
        {
            byte[] jsonToSend = new System.Text.UTF8Encoding().GetBytes(json);
            request.uploadHandler = (UploadHandler)new UploadHandlerRaw(jsonToSend);
            request.downloadHandler = (DownloadHandler)new DownloadHandlerBuffer();
            request.SetRequestHeader("Content-Type", "application/json");
            request.SetRequestHeader("X-AIO-Key", adafruitIOKey);

            yield return request.SendWebRequest();

            if (request.result == UnityWebRequest.Result.ConnectionError || request.result ==
                UnityWebRequest.Result.ProtocolError)
            {
                Debug.LogError(request.error);
            }
            else
            {
                Debug.Log("Response: " + request.downloadHandler.text);

            }
        }

        yield return new WaitForSeconds(2f);
        isSendingRequest = false;
    }
}
```

Cloud-based method (Adafruit IO MQTT)

requires the following extended library to work (placed in the Assets folder):
https://github.com/gpvigano/M2MqttUnity

OlfactionObject.cs:

```
using System;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.XR.Interaction.Toolkit;

public enum TriggerType
{
    Click,
    Distance
}

public class OlfactionObject : MonoBehaviour
{
    private XRBaseInteractable interactable;
    public ParticleSystem perfumeParticles;

    [Tooltip("The olfactive object needs a DataSender component to send the scent parameters
        to the dispenser")]
    public DataSender dataSender; // DataSender instance

    [Tooltip("IP Address of the dispenser")]
    public string deviceIP; // Setting DeviceIP manually
    public TriggerType triggerType; // Type of trigger to use (proximity or interaction)

    [Tooltip("Player's transform (Main camera in the XR Rig)")]
    public Transform playerTransform; // Player's transform for distance calculation

    [Tooltip("Minimum distance at which distance trigger type is activated")]
    public float triggerDistance; // Trigger distance for distance-based interactions


    // Function to calculate intensity and duration based on distance
    public (string, string) ScentParametersUrl(float distance)
    {
        // Calculate intensity and duration based on distance
        int intensity = Mathf.CeilToInt(3 - 2 * (distance / triggerDistance));
        string duration = "2.2";
        string intensity_level = "";
        if (intensity == 1) {
            intensity_level = "low";
        }
        if (intensity == 2) {
            intensity_level = "medium";
        }
        if (intensity == 3){
            intensity_level = "high";
        }
        return (duration, intensity_level);
    }

    void Start()
    {
        if (playerTransform == null)
        {
            Debug.LogWarning("Player Transform is not set. Please assign it in the inspector."
                );
        }
```

```
        if (dataSender == null)
        {
            // Attempt to find an object with a DataSender component
            dataSender = FindObjectOfType<DataSender>();
            if (dataSender == null)
            {
                Debug.LogWarning("Data Sender is not set. Please assign it in the inspector.")
                    ;
            }
        }
        if (deviceIP == null)
        {
            Debug.LogWarning("Device IP is not set. Please assign it in the inspector.");
        }

        // Get the XRBaseInteractable component for handling interactions
        if (triggerType == TriggerType.Click) {
            interactable = GetComponent<XRSimpleInteractable>();
            if (interactable != null)
            {
                // Subscribe to the SelectEntered event to detect when interaction-based
                    object is selected
                interactable.selectEntered.AddListener(OnSelectEntered);
            }
        }

    }

    void Update()
    {
        if (triggerType == TriggerType.Distance)
        {
            // Calculate the distance between the player and this object
            float currentDistance = Vector3.Distance(playerTransform.position, transform.
                position);
            // Check if the distance is less than the trigger distance
            if (currentDistance <= triggerDistance)
            {
                var (duration, intensity) = ScentParametersUrl(currentDistance);
                dataSender.SendCommand(deviceIP, duration, intensity);
                perfumeParticles.Play();
            }
        }
    }


    void OnSelectEntered(SelectEnterEventArgs args) // This function is called when the user
        clicks on the object
    {
        if (triggerType == TriggerType.Click)
        {
            string duration = "3";
            string intensity = "high";
            dataSender.SendCommand(deviceIP, duration, intensity);
        }
    }
}
```

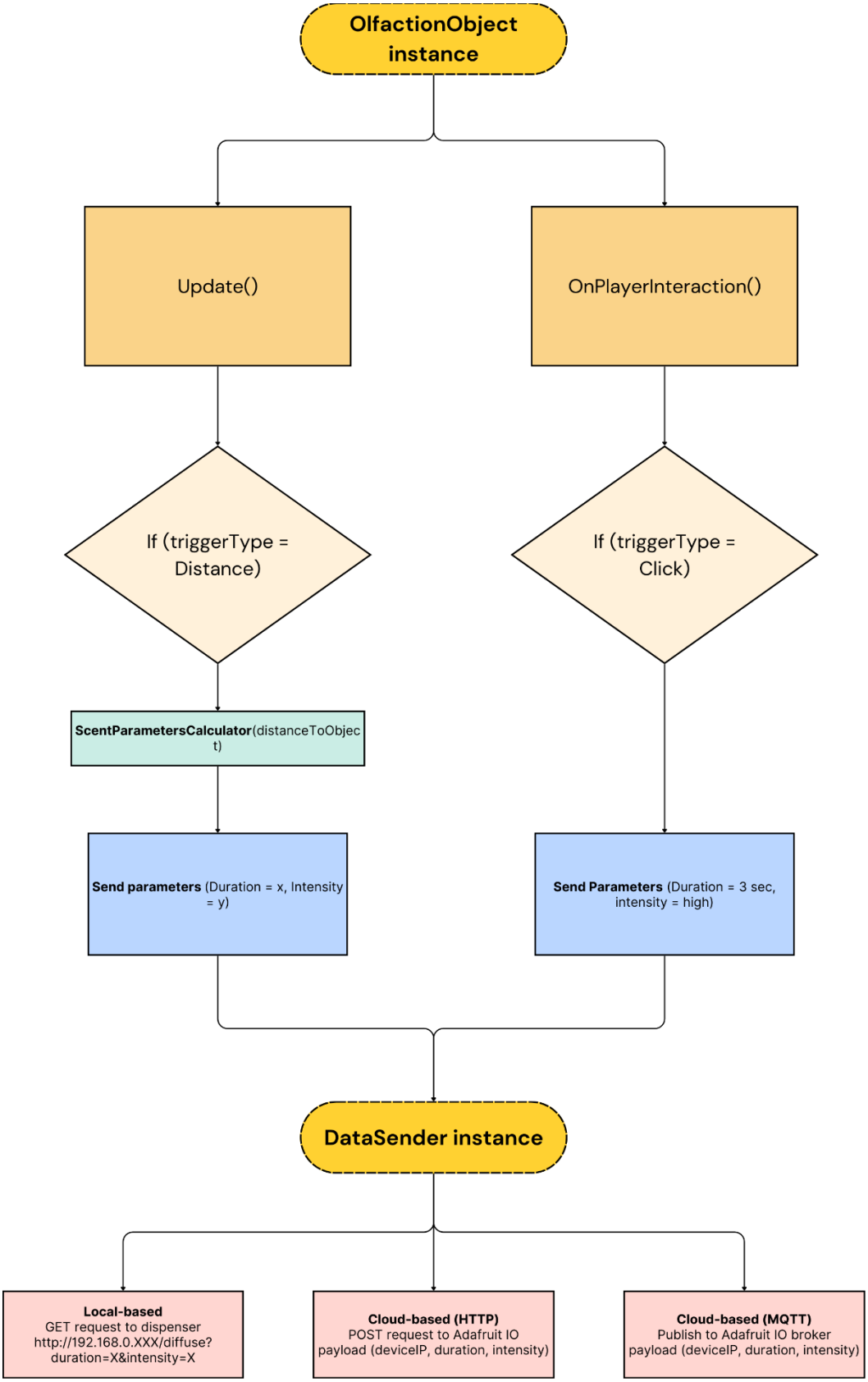
DataSender.cs:

```
using System;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using uPLibrary.Networking.M2Mqtt;
using uPLibrary.Networking.M2Mqtt.Messages;
using System.Text;

public class DataSender : M2MqttUnity.M2MqttUnityClient
{
```

```
    [Header("Adafruit IO Credentials")]
    public string adafruitIOKey = "aio_fIOc56tr4A5Hjh77X0SXwA5VBapw";
    public string adafruitIOUsername = "MaroBZ";
    public string feedName = "exhalia-mqtt";

    private bool isPublishing = false; // Flag to prevent multiple messages from being sent at
         the same time

    private string topic;

    protected override void Awake()
    {
        base.Awake();
        mqttUserName = adafruitIOUsername;
        mqttPassword = adafruitIOKey;
        topic = $"{adafruitIOUsername}/feeds/{feedName}";
    }

    protected override void Start()
    {
        brokerAddress = "io.adafruit.com";
        brokerPort = 1883; // Default MQTT port
        autoConnect = true; // Automatically connect on startup

        base.Start();
    }

    public void SendCommand(string deviceIP, string duration, string intensity)
    {
        string message = $"{{\"deviceIP\":\"{deviceIP}\",\"duration\":\"{duration}\",\"
            intensity\":\"{intensity}\"}}";
        if (!isPublishing)
        {
            StartCoroutine(PublishRequest(message));
        }
    }

    private IEnumerator PublishRequest(string message) {
        isPublishing = true;

        if (client != null && client.IsConnected)
        {
            client.Publish(topic, Encoding.UTF8.GetBytes(message), MqttMsgBase.
                QOS_LEVEL_AT_LEAST_ONCE, false);
            Debug.Log("Message sent: " + message);
        }
        else
        {
            Debug.LogWarning("Client not connected. Cannot send message.");
            yield return null;
        }
        yield return new WaitForSeconds(2); // Wait for 2 seconds before sending the next
            message
        isPublishing = false;

    }


    protected override void OnConnected()
    {
        base.OnConnected();
        Debug.Log("Successfully connected to Adafruit IO.");
    }

    protected override void OnConnectionFailed(string errorMessage)
    {
        base.OnConnectionFailed(errorMessage);
        Debug.LogError("Connection to Adafruit IO failed: " + errorMessage);
    }

    protected override void OnDisconnected()
    {
        base.OnDisconnected();
        Debug.Log("Disconnected from Adafruit IO.");
    }

}
```

Appendix J
C# Scripts
Diagram Explanation

C# SCRIPTS DIAGRAM EXPLANATION



Appendix K
Adafruit IO Python Listener Scripts Source Code

HTTP feed listener

```
from flask import Flask, request
import requests
import json
from Adafruit_IO import Client, Feed, Data
import time
from datetime import datetime
import csv
import os

app = Flask(__name__)

ADAFRUIT_IO_KEY = 'aio_fIOc56tr4A5Hjh77X0SXwA5VBapw'
ADAFRUIT_IO_USERNAME = 'MaroBZ'
FEED_NAME = 'exhalia-http'

aio = Client(ADAFRUIT_IO_USERNAME, ADAFRUIT_IO_KEY)


last_timestamp = None  # Variable to store the timestamp of the last processed data

@app.route('/command', methods=['POST'])
def handle_command():
    data = request.json
    device_ip = data['deviceIP']
    duration = data['duration']
    intensity = data['intensity']

    url = f'http://{device_ip}/diffuse?duration={duration}&intensity={intensity}'

    try:
        response = requests.get(url)
        response.raise_for_status()
        return {'status': 'success', 'response': response.
```

```
            text}, 200
    except requests.RequestException as e:
        return {'status': 'error', 'error': str(e)}, 500

def check_feed_updates():
    global last_timestamp
    while True:
        try:
            data = aio.receive(FEED_NAME)
            if data is not None:
                current_timestamp = data.created_at
                if current_timestamp != last_timestamp and
                  last_timestamp != None:
                    last_timestamp = current_timestamp
                    print(f'Received data at {
                      current_timestamp}: {data.value}')

                    command = json.loads(data.value)
                    device_ip = command['deviceIP']
                    duration = command['duration']
                    intensity = command['intensity']

                    url = f'http://{device_ip}/diffuse?
                      duration={duration}&intensity={
                      intensity}'
                    try:
                        response = requests.get(url)
                        response.raise_for_status()
                        print(f'Success: {response.text}')
                    except requests.RequestException as e:
                        print(f'Error: {e}')
                elif last_timestamp == None:
                    last_timestamp = current_timestamp


        except Exception as e:
            print(f'Error receiving data: {e}')

        time.sleep(0.2)  # Poll every 0.2 seconds for updates

if __name__ == '__main__':
```

```
    # Start a separate thread to continuously check for feed updates
    import threading
    update_thread = threading.Thread(target=check_feed_updates, daemon=True)
    update_thread.start()

    app.run(port=5000)
    # Run the Flask web server
```

MQTT topic listener

```
import json
import requests
import paho.mqtt.client as mqtt

ADAFRUIT_IO_USERNAME = 'MaroBZ'
ADAFRUIT_IO_KEY = 'aio_fIOc56tr4A5Hjh77X0SXwA5VBapw'
MQTT_FEED_NAME = 'exhalia-mqtt'

MQTT_BROKER = 'io.adafruit.com'
MQTT_PORT = 1883
MQTT_TOPIC = f'{ADAFRUIT_IO_USERNAME}/feeds/{MQTT_FEED_NAME}'


def on_connect(client, userdata, flags, rc):
    print(f"Connected with result code {rc}")
    client.subscribe(MQTT_TOPIC)


def on_message(client, userdata, msg):
    print(f"Received message: {msg.payload.decode()}")
    command = json.loads(msg.payload.decode())

    device_ip = command['deviceIP']
    duration = command['duration']
    intensity = command['intensity']

    url = f'http://{device_ip}/diffuse?duration={duration}&
      intensity={intensity}'

    try:
        response = requests.get(url)
        response.raise_for_status()
        print(f'Success: {response.text}')
    except requests.RequestException as e:
        print(f'Error: {e}')
```

```
35
36
37 client = mqtt.Client()
38 client.username_pw_set(ADAFRUIT_IO_USERNAME, ADAFRUIT_IO_KEY)
39 client.on_connect = on_connect
40 client.on_message = on_message
41
42 client.connect(MQTT_BROKER, MQTT_PORT, 60)
43
44 client.loop_forever()
```

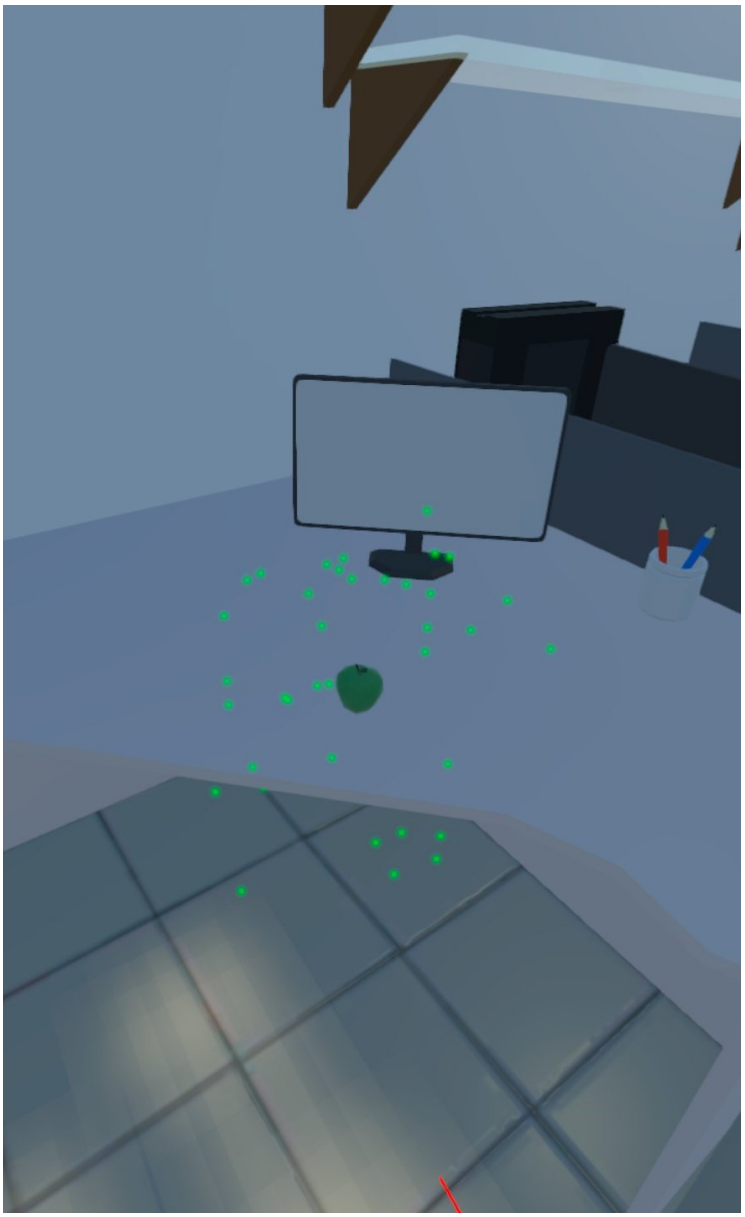
Appendix L
Digital Twin
Comparison

DIGITAL TWIN vs REALITY COMPARISON

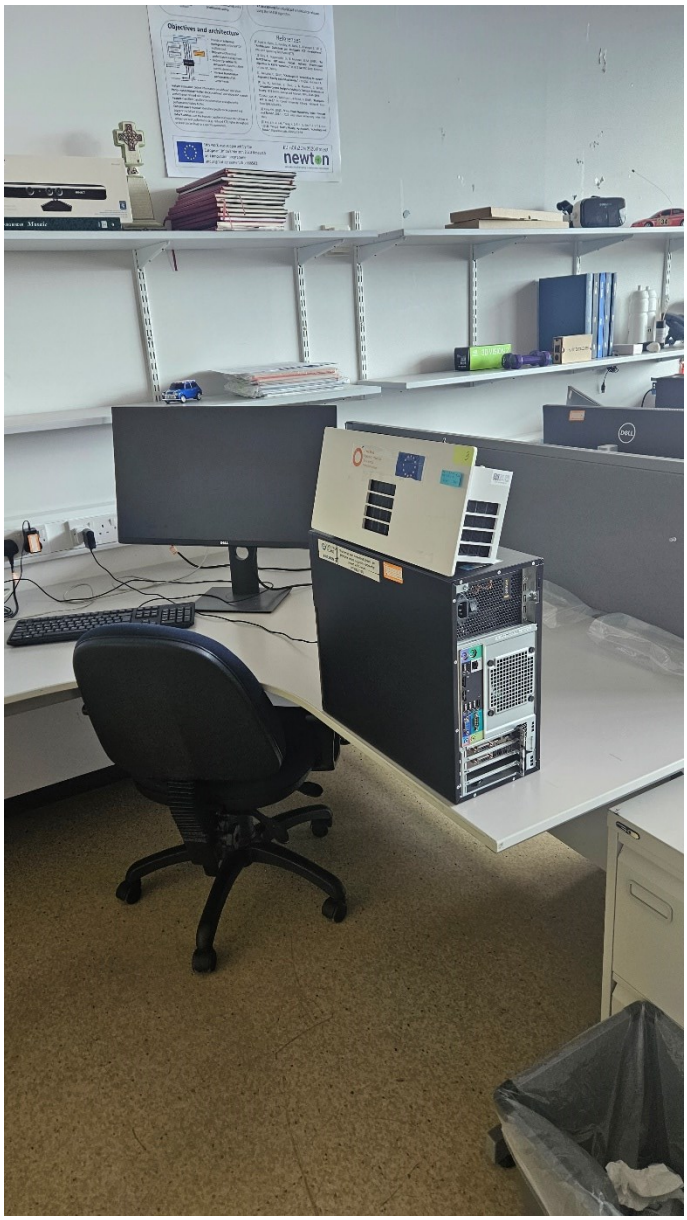
LABORATORY OVERVIEW:

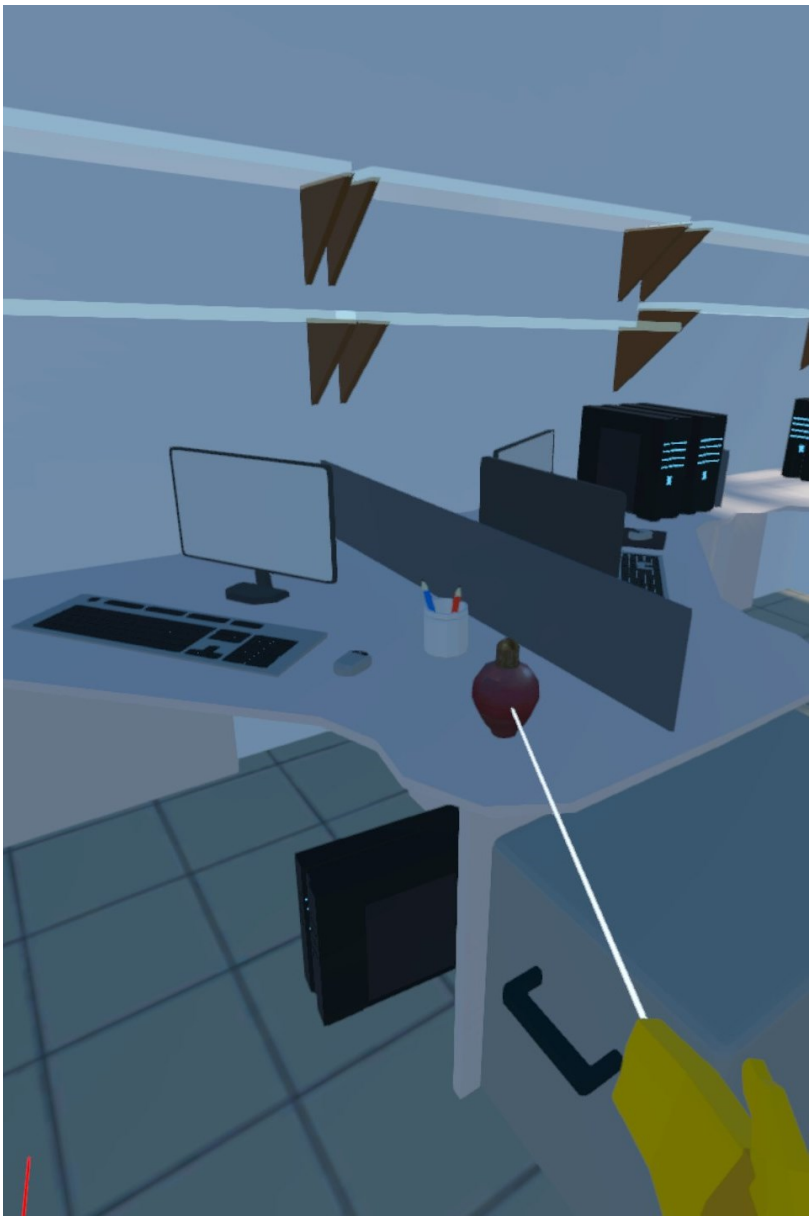

XR Device Simulator

APPLE:



PERFUME:





COOKIE :

Appendix M
Software
&
Tools List

SOFTWARE & TOOLS LIST

Software & tools used:

- Meta Quest Developer Hub, version 4.8.0
- Unity Game Engine, editor version 2022.3.5f1
- Unity Asset Store (https://assetstore.unity.com/)
- Adafruit IO cloud server (https://io.adafruit.com/)
- Visual Studio Code, version 1.92.0
- Python 3.11+ environment

Programming languages:

- C#
- Python

All documents, source codes and other materials available on the google drive link:

https://drive.google.com/drive/folders/1KEjLZ2RL8yvbfIGmYrpXreJP_ufZsyQT