Room of Errors - Feasibility and Design of a Simulation Training Concept in VR with Intentional Simulation Errors

Kristoffer Nome

Master's thesis in Software Engineering at

Department of Computer science, Electrical engineering and Mathematical sciences,
Western Norway University of Applied Sciences

Department of Informatics,
University of Bergen

June, 2022

Abstract

In 2020, 13.1 percent of somatic hospital stays in Norway led to hospital-acquired complications or injuries. One way of working to prevent these events may be found in the education of health personnel. Nursing study programs have seen increased use of simulation training and Virtual Reality (VR). One method for simulation training is the "Room of Errors," which asks participants to look for pre-established errors. Implementing this concept in VR introduces additional challenges: How should such an application be designed so that users recognize and accept the errors as part of the simulation and do not see them as the result of a faulty application? Previous studies have not considered cooperative Room of Errors simulation training in VR, which has added benefits in non-VR simulations. How feasible is it to support multiple concurrent users?

This thesis addresses these questions by developing a VR Room of Errors application. Nine lecturers for Nurse Anesthetist, Intensive Care Nursing, and Operating Room Nursing study programs evaluated the application. They confirm its usability and that they would consider using the application as part of their simulation-based education. Importantly, the virtual environment is accepted by users, including intended and unintended simulation errors. Evaluations with two cooperating participants shows an added layer of communication and confirms the feasibility of multiple concurrent users. The concept is viable in VR, and development is practically and financially feasible.

Note: Some text in this thesis has been used by the author in a research article accepted for publication at the 1st IEEE International Conference on Cognitive Aspects of Virtual Reality 2022 [1]:

K. Nome, M. V. Olsen, H. Soleim, A. Geitung, and T. Johnsgaard, "Usability of a VR Simulation Training Concept with Intentional Simulation Errors," in 1st IEEE International Conference on Cognitive Aspects of Virtual Reality, (Online), IEEE, May 2022

At the time of submitting this thesis, the research article is accepted but not yet published. The IEEE copyright form states:

- Authors/employers retain all proprietary rights in any process, procedure, or article of manufacture described in the Work.
- Authors/employers may reproduce or authorize others to reproduce the Work, material extracted verbatim from the Work, or derivative works for the author's personal use or for company use, provided that the source and the IEEE copyright notice are indicated, the copies are not used in any way that implies IEEE endorsement of a product or service of any employer, and the copies themselves are not offered for sale
- Although authors are permitted to re-use all or portions of the Work in other works, this does not include granting third-party requests for reprinting, republishing, or other types of re-use.The IEEE Intellectual Property Rights office must handle all such third-party requests.

Since the relevant text in the article that also appears in this thesis may be considered "material extracted verbatim from the Work,", the following copyright notice for the article is included in this thesis:

The IEEE copyright does not extend to the entire thesis.

Preface

Five years ago, I worked day, evening and night shifts as an operating room nurse. I loved my job but could never shake the thought that I should give this computer thing a proper attempt before thirty years had suddenly passed and it would be too late. This thesis is the result of that "proper attempt".

My deepest gratitude to Harald Soleim and Atle Birger Geitung, my supervisors at the Western Norway University of Applied Sciences (HVL), for their advice, insight, and guidance, and for generally believing in me from start to finish. They have pushed me to greater heights and at the same time kept me grounded.

The project was initiated by Marit Vassbotten Olsen, a nurse anesthetist and simulation facilitator at SimArena, HVL. We met when she was looking for IT students in HVL's computer graphics course to develop a VR application for her. We recognized each other from my OR nursing studies, when she was a lecturer for nurse anesthetists. I jumped on the opportunity to use my OR nursing background.

I am extremely grateful to Marit for an amazing experience. Even though we were able to skip some hurdles due to my background, it was not always entirely straight-forward —– but we pulled through together. I hope you will be able to get some use out of the application we have created together, and I look forward to reading about your results in the future. I am confident our paths will cross again.

Meeting Marit led to my Bachelor's thesis project. The project described in this thesis is a continuation of that project, which was completed in June 2020, in cooperation with Adrian René Johnsen Mortensen. Adrian moved back to Trondheim after his BSc degree and left the project in my hands. He was sorely missed during development! We were able to work together physically for four days before the first Covid lock down took effect, so we had to forge some new paths and find our own solutions. I want to thank him for our teamwork and his contributions —– a special thanks for "the ray".

Thanks to Tone Johnsgaard and Petrin Hege Eide who participated in our "design team", provided invaluable insight into their professional worlds, and helped with finding willing evaluation participants. Thanks also to Tone again, Klas Karlgren, and Ilona Heldal, for their help in writing a research paper based on the project.

I want to thank my fellow Computer Graphics Master students for making the last year enjoyable —– a special thanks to Severin for "holding the lab-fort" with me every day for the last year. Having someone to discuss ideas and troubleshoot problems with has been invaluable.

Lastly, thanks to my family for always encouraging and believing in me. These past five years have been quite the experiment.

I consider myself very lucky to be able to combine my two “professional homes”, and it has been exciting to see other people showing interest in our project. The VR application has been presented at three different occasions related to HVL, and a research article based on the project has been accepted for publication at the 1st IEEE International Conference on Cognitive Aspects of Virtual Reality, which also involved a presentation of the work. In addition, several people have expressed interest in future developments and have asked when *they* can get access to the application for their students.

A “promotional” video of the application can be seen at:

https://youtu.be/9Bba-xeEbtg

Contents

Acronyms

AIO Anesthesia, Intensive care, and Operating Room. 1, *Glossary:* AIO nursing

AIO nursing nurse anesthetist, intensive care nursing, and operating room nursing. viii, 1, 3, 27, 35, 55, 74, 85, *Glossary:* nurse anesthetist, intensive care nursing, and operating room nursing

AR Augmented Reality. 14

DoF Degrees of Freedom. 16

GPU Graphics Processing Unit. viii, 11, 13, 61, 63, *Glossary:* Graphics Processing Unit

HMD Head-Mounted Displays. viii, 2, 4, 9, 11, 14, 16, 17, 34, 35, 37–39, 41, 42, 46, 48, 49, 51, 52, 55, 60, 63, 64, 69, 72, 80, *Glossary:* Head-Mounted Displays

HVL Western Norway University of Applied Sciences. 3, 10, 30, 34, 35, 60, 65, 80

MR Mixed Reality. 14

OR operating room. viii, 9, 10, 37, 40, 43, 48, 49, 61, 62, 70, 77, *Glossary:* operating room

OR nursing operating room nursing. viii, 3, 9, 43, 51, 67, 70, 73, 75, 77, 82, *Glossary:* operating room nursing

SUS System Usability Scale. 4, 34–36, 66, 69, 72–75, 82, 84

VR Virtual Reality. viii, 2–10, 13–17, 19–24, 27, 30, 31, 33–39, 41, 46, 48, 49, 53, 54, 57, 60, 61, 64, 66–70, 72–75, 77–82, 84, 85, *Glossary:* Virtual Reality

XR Extended Reality. 14, 24, 60

Glossary

asset Term that describes content or tools used in the game engine. For example, 3D models, textures, or sound effects. 24, 41, 45, 48, 51, 60, 61, 82

Component A *component* in Unity can define traditional object-oriented classes and objects, or specific traits and behaviors. 25, 26, 39, 55–58

Entity A basic "building block" in Unity. Can have one or more *Components*. See also: GameObject. 25, 56, 58

GameObject The most basic *entity* in the Unity engine and its *Entity-Component architecture*. Can contain one or more *Components*, and can have one or more GameObject-children, which themselves can have one or more GameObject-children. ix, 25, 26, 58

Graphics Processing Unit A specialized processor, often located on specialized GPU cards. Optimized for parallel polygon processing. viii, 11

Head-Mounted Displays Head-Mounted Displays let users "shut out" other visual input than is given to them through the enclosed displays. Contains one screen for each eye, which provides depth perception. Can also provide perceived movement in the virtual experience by tracking its movement in 3D space. viii, 2

intensive care nursing A patient with an acute, critical illness requires intensive care nursing 24 hours a day. The education gives the nurse knowledge and competency to care for these patients. 9, 43, 62, 67, 69, 70, 72–74, 81

Material In Unity, Materials are used in combination with shaders to (somewhat simplified) determine how objects should be colored. 41, 42, 61, 62

mesh A collection of vertices in 3D coordinate space and the polygons created by those vertices. Together, they approximate a surface. 11, 13, 17, 34, 41, 61, 62

nurse anesthetist The nurse anesthetist is the first person to meet a patient in a surgical ward, and the last person in the surgical team to leave the patient after they are transferred to a postoperative ward. A simplification: An nurse anesthetist is specialized in providing nursing to patients in general anaesthesia ("Narkose" in Norwegian.). 3, 9, 43, 51, 67, 70, 73, 74

nurse anesthetist, intensive care nursing, and operating room nursing Three of the many specializations in nursing. Master-level degrees. Requires a bachelor's degree in nursing, and minimum two years of professional practice within the last five years. The three studies contain several overlapping subjects and topics. viii, 3, 85

operating room A room where surgical procedures are performed. A surgical *ward* can contain many operating rooms. viii, 9, 37, 40, 43, 48, 61, 70, 77

operating room nursing Operating Room nursing requires knowledge about the acute and/or critically ill patient's situation and needs, and deep knowledge about surgical principles and surgical techniques and treatments. viii, 9

polygon Defined by connected vertices. A collection of polygons and the vertices that define them define a 3D mesh. The simplest polygon is a triangle. 11, 13, 61, 62

postoperative ward A hospital ward where patients are monitored after surgery until they are considered stable enough to be transferred to a regular hospital ward or be released. A postoperative ward can contain large open rooms with several beds, several closed private rooms, or a combination. 9, 10, 43, 45, 48, 61, 62, 70

Room of Errors Method for simulation training. A room containing prearranged errors or breaches of patient safety routines. One or more participants are tasked with finding as many errors as they can within a given time. 1–5, 7–9, 11, 19–21, 24, 27, 30, 32, 34, 35, 37–39, 41–43, 48, 49, 53–56, 60, 68, 69, 77–79, 81, 82, 84

Scene A logical division of a Unity game's content. In practice, a Scene defines what should be loaded into memory. 25, 26, 55

Unity Game engine. Refers to both the core runtime of a compiled game and the frameork and tools that facilitate production of the game. 4, 10, 24–26, 33, 41, 45, 55, 56, 58, 60–62, 80

vertex Connected vertices in 3D coordinate space define polygons, which together define a 3D mesh. The number of vertices in a polygon mesh determines the mesh's resolution. 11, 13, 61, 62

Virtual Reality A virtual reality gives users increased immersion in virtual experiences by using HMDs. Can refer to VR technology or the virtual experience itself. viii, 2

Chapter 1

Introduction

This chapter introduces the motivation and context for this thesis, leading to its research questions and research goals, followed by a summary of prior and related works.

1.1 Motivation

Adverse events during hospitalization can lead to severe consequences for patients. In 2020, 13.1 percent of somatic hospital stays in Norway led to hospital-acquired complications or injuries. It is considered likely that more than half of these could have been avoided [2]. Anesthesia, Intensive care, and Operating Room (AIO) wards and sections are advanced technological departments requiring high degrees of competency to give patients a comprehensive quality of care. Patient safety is an important area of focus in these departments, exemplified by the frequent use of checklists and simulation training — but adverse events of varying severity still occur. A 2008 study found that 41% of adverse events in hospitals occur in operating rooms, and only 3.1% occur in intensive care units [3]. Recognizing that these events occur, learning from them, and working toward preventing them, are important steps in improving patient safety [4, 5].

One potential avenue for preventing such events can be found in the education of health personnel, which has seen increased use of simulation training in recent years. One specific method for simulation training is the "Room of Errors" [6, 7, 8, 9, 10] (also called "Room of Horror" in some studies), which highlights patient safety. The method involves a standard patient room that has been prepared with a number of errors or breaches of patient safety routines. One or several participants enter the room with the objective of finding as many errors as possible within a given time. The purpose of the simulation is for the participants to develop their ability to discover such errors in real-life situations and, perhaps more importantly, develop their ability and confidence to communicate about errors and to speak up. Several participants entering as teams tend to discover more errors than the individual participants. Performing the simulation in teams also promotes collaboration and communication between the different professions [9] — which by itself can contribute to preventing adverse events in

Figure 1.1: Room of Errors with a single participant (Photo: Molloy & Clay [12])

Figure 1.2: Room of Errors with multiple participants (Photo: Molloy & Clay [12])

the operating room [11]. Examples of Rooms of Errors can be seen in Figures 1.1 and 1.2.

In later years, nursing study programs have also started to use simulation training aided by computer games and Virtual Reality (VR) [13]. By performing simulation training in VR, several resource needs can be reduced. For example, the simulation can be done independently of physical rooms and equipment, and the time to prepare situations can be drastically reduced [14]. Also, the need for a teacher to guide and control the simulation is potentially lessened. For the Room of Errors concept specifically, simulation in VR would allow participants to experience errors that would be infeasible in real-life versions. Participants can also go through several different rooms in one session, which lets them experience different types of errors.

A virtual Room of Errors introduces a number of challenges. In order to find something wrong in the room, there must be something *right* in the room. A perfectly realistic, 1:1 digital recreation of such advanced wards would likely be challenging and time-consuming to create. In addition, the amount of detail required would likely be graphically resource-intensive — something that could prove challenging for popular standalone Head-Mounted Displays (HMD) units such as the Meta Quest (previously Oculus[1] Quest), which have somewhat limited technical capabilities.

Without such a perfect recreation of the rooms, it would be easy for participants to find errors that were not meant by the developers to be errors but were instead an unintended imperfection or lack of detail. These "unintended" errors are discovered frequently in non-virtual versions. In one study by Turrentine et al. [10], 33 errors were prepared, but participants found **291** extra unintended errors. However, none of the articles about non-digital Rooms of Errors that have been found in literature searches see this as a problem, as the unintended errors did not seem to affect the participants' view of the simulation experience. Would this also be true for a VR version of the concept? Is it possible for participants to accept an imperfect, simulated room and be immersed in the situation

[1]Oculus officially rebranded its HMDs to the Meta brand in January 2022. However, not all official Oculus websites and documentation have been updated. Therefore, this thesis will occasionally use the Oculus name when referring to websites with Oculus branding.

while they are explicitly looking for errors and faults? Or would they instead see the imperfections and unintended errors as errors with the application itself and consider it “buggy” and poorly developed? This could severely impact a participant’s engagement and immersion in the experience — factors that seem to be important for learning outcomes [15]. Questions are also raised regarding the level of *fidelity* required — how close to reality does a simulation need to be in order to be an effective learning tool?

At Western Norway University of Applied Sciences (HVL), simulation training for the various nursing educations is performed at SimArena. SimArena is in the process of expanding their use of VR in simulation training. In addition, researchers at SimArena have initiated a larger research project focusing on patient safety and adverse events, including a PhD study by Marit Vassbotten Olsen [16], a nurse anesthetist and simulation training facilitator. At the time of writing, the study is in progress and planned to be completed in 2025. One of several sub-projects in the PhD study will examine how new technologies such as VR can contribute when acquiring new knowledge about patient safety. This process will include looking at the possible effects and learning outcomes of a virtual Room of Errors for nurse anesthetist, intensive care nursing, and operating room nursing (AIO nursing) students. The three study programs contain several overlapping subjects and topics and frequently collaborate, both during their studies and professional work.

This current thesis aims to explore how to design and develop a VR Room of Errors directed at nurse specialist students and examine if the concept is usable in VR. As the author of this thesis is both a software developer and a trained operating room nurse, we believe this can be explored from a unique perspective. The developed application will be used as an experimental intervention in the related PhD study [16].

1.2 Research questions

Studies have shown that non-digital simulation training does not necessarily require perfection and high fidelity in the simulations to improve a user’s learning outcome [15, 17]. Tun et al. [18] explicitly oppose the notion that high-fidelity simulation requires a complete and perfectly detailed representation of reality. They instead argue for focusing on other factors, such as accurate representations of real-world sensory cues and relevant stimuli, and that clarifying limitations in the simulation at an early stage could allow participants to more easily suspend their disbelief. Is this also true for VR? If so, how should this simulation be designed? How much detail in the virtual room is *good enough* for participants to immerse themselves in a virtual Room of Errors? Also, considering that performing a Room of Errors simulation using teams seems to have advantages, is it feasible to design the VR simulation so that it allows for several participants to take part at the same time?

This study aims to design a VR application that enables collaborative training in identifying errors in simulated clinical environments. This leads to this thesis’ research questions:

- RQ1: How can we design a simulation in VR that enables users to recognize errors as part of the simulation and not as the result of a faulty application?

- RQ2: How feasible is it to create a virtual Room of Errors simulation in VR, which supports multiple concurrent users?

1.3 Research goals

The main goal of this study is to understand and describe design principles, design elements, and the design process for designing immersive VR experiences. Previous related studies often describe the virtual environment that was used but rarely the process behind it [19]. Also, there does not seem to be previous research on how to design such an experience when users are explicitly looking for elements that potentially break their immersion. The resulting application can potentially be considered a specific solution to the problem of creating a virtual Room of Errors. However, the research also aims to have implications for future development processes of Serious VR Games targeting health care students regarding techniques used during development and elements that proved useful to implement. In addition, this thesis aims to understand how users' experience of a VR application is affected when they are essentially looking for something wrong with the application. Results could have implications for how much time and resources need to be spent on detail and realism in VR simulations — both in health care simulations and other professional fields where error awareness is of importance, such as air traffic.

1.4 Scope and limitations

One of the main objectives of the related PhD study is to examine the learning outcomes of the virtual Room of Errors over time. Therefore, it was decided not to study the pedagogical effects and learning outcomes of using the application in this thesis but rather to focus on the application itself and its usability.

1.5 Methodology

Answering the research questions requires investigating and studying principles and guidelines for developing Serious Games, both in terms of the development process itself and game elements that promote learning and engagement. In addition, features and elements that previous studies have indicated lead to immersive experiences are studied. Domain experts at SimArena are heavily involved in designing the VR environments, scenarios, and included errors.

This research is used to develop a VR Room of Errors simulation. User evaluations were performed to evaluate the application, aided by a mixed-method approach using the System Usability Scale (SUS) [20] in addition to observation and semi-structured interviews. The evaluations are described in chapter 3.

The application was implemented in Unity [21] for the Meta Quest [22] VR HMD. The application's design is described in chapter 4. It is intended to be used as a trial intervention as part of the ongoing PhD research project at SimArena [16] described in section 1.1, studying patient safety and adverse events.

The project can be viewed from the perspective of the Design Science methodology [23], which, in summary, uses both an established knowledge base and an environment to design or develop an IT artifact that solves a relevant problem while at the same time providing new knowledge to the knowledge base [23]. In this case, research literature serves as a knowledge base, and domain experts at SimArena serve as an environment. Design Science is further discussed in chapter 3.

1.6 Prior work

During the spring of 2020, a prototype Room of Errors in VR was developed as part of a bachelor thesis in Computing and Information Technology by Adrian R. J. Mortensen and Kristoffer Nome [24], in cooperation with Marit Vassbotten Olsen and Lars Peder Bovim from SimArena. During development of the prototype, the potential of the VR simulation became more apparent to the project's initiators from SimArena and lecturers at the relevant study programs. Also, the possibility for more dynamic environments was increasingly considered to be of special interest for the ongoing larger research project, instead of the more traditional static Room of Errors containing a given number of errors. In addition, several questions were left unanswered from a software engineering perspective, especially since the global Covid-19 pandemic prohibited user testing. Therefore, it was decided to continue and expand development of the virtual Room of Errors for further research as part of this thesis.

The prototype is further described in section 2.1.

1.7 Related work

Simulation in VR for health care professions has seen significant amounts of previous work. Computer graphics and VR have played a part for decades in diagnosing and treating patients, and VR simulation is increasingly used for training and planning surgery [25]. Several tools are available (e.g., Acadicus [26], Oxford Medical Simulation [27], SimX [28] and Osso VR [29]). In addition, applications exist that are directed at highly specific professionals, using specialized hand controllers for specific types of operations, such as LapSim [30] for training in laparoscopic surgery ("keyhole surgery" or "kikkhull-operasjoner" in Norwegian). Applications focused on training communication have also seen positive results in medical students [31].

Some commercial VR applications also specifically target nurses and nursing students. For example, UbiSim [32] is specifically directed at nursing students and provides customizable scenarios where one or several users interact with virtual patients in realistic clinical settings. One published article has been found [33] that describes the successful experiences of designing a UbiSim scenario — although it should be noted that the author of this article has acted as a consultant at UbiSim, and the work was funded by UbiSim.

There have been studies on virtual simulations in nursing studies (e.g., [34]), but these have not focused on VR. Instead, the term "virtual" has often been used to describe simulations as experiences on a "traditional" computer screen. In general, there is a paucity of academic studies on the use of VR in nursing studies [35].

1.7.1 Related work on VR and immersion

VR has also been the object of several academic studies, some of which are summarized here. The technology is further discussed in section 2.7.

Servotte et al. [36] immersed 61 healthcare students in a VR simulation of a mass casualty incident involving a bus crash in a tunnel. The authors note that limited evidence exists in literature about the immersion process and the factors that affect it. Therefore, the study aims to understand more about the topic. Validated French adaptations of Witmer & Singer's *Immersive Tendencies Questionnaire* and *Presence Questionnaire*[37] were used to measure immersion and sense of presence. In addition, qualitative interviews were performed. Elements that are reported to specifically influence immersion include pictorial realism, the ability to act relatively freely, and the fidelity of the scenario itself. Concrete suggestions for improvement include expanding the possible interactions with patients, improving the fidelity of the sounds in the simulation, and adding haptic feedback. These suggestions based on their findings strengthen other studies' claims that multisensory congruent cues such as visual, auditory, and tactile enhance participants' presence in the simulation. In addition, the authors strongly suggest including a pre-briefing before entering VR, as was also suggested by Tun et al. [18].

Radianti et al. [19], in a literature review, examined VR applications directed at higher education, which are described in research papers. Thirty-eight articles are included in the analysis. The authors identify fourteen design elements that have been used. Notably, the authors discover that the term *realistic* is not used uniformly, as some papers use it to describe high-fidelity environments with complex, high-quality graphics. In contrast, others use it to describe "realistic enough" environments, detailed enough for users to recognize them.

This study is expanded upon in a 2021 study [38], which instead examines 120 commercially available VR applications directed at higher education. The authors conclude by saying that their work reveals which design elements are the most commonly used while designing such VR applications, but that knowledge on which design elements actually support learning is lacking.

In another literature review, Neo, Won & Shapley [39] attempt to find strategies for designing immersive virtual environments. The authors note that researchers typically describe the environment they have designed but rarely describe the application's design decisions and design processes. They do, however, identify five key categories that have been reported as effective for designing VR environments intended for behavior research: The appropriate level of detail in the environment, context, social cues, participant tracking, and, lastly, rendering and non-visual sensory information.

Jennett et al. [40] investigated whether immersion can be defined and measured quantitatively, leading to a validated questionnaire. The study discovered several factors they believe can enhance users' immersive experiences. The article defines several useful terms and perspectives on immersion and describes the process of developing a questionnaire through three separate experiments, all involving non-VR computer games and interactions.

1.7.2 Related work on Room of Errors.

Numerous articles have studied traditional non-digital Rooms of Errors (e. g. [6, 7, 9, 10]). However, there does not seem to be many studies on VR implementations. Some mentions can be found about one implementation being used at American educational institutions as early as 2018 [41], but these mentions do not include information about whether this implementation is commercially available or how it works. In addition, some movies can be found from a VR implementation that seemingly was under development at UMass Chan Medical School's simulation center in Massachusetts, USA, when the movies were published during the autumn of 2019 [42]. However, attempts at finding more information or descriptions about the application have been unsuccessful. Regardless, the application seems to be directed at medical students and bachelor's degree nursing students. In addition, they take place in a traditional hospital ward and not in specialized units such as anesthesia, operating room, and intensive care wards, which is the target group for the current master's thesis.

However, one study has been found, specifically looking at a VR implementation of a Room of Errors directed at operating room nursing students by Bracq et al. [43]. Among other findings, the study looks at factors that affected how many errors users discovered. One of these factors is immersion. The study found that users reporting higher degrees of immersion found more errors than users reporting low immersion. This finding leads to a question: Did increased immersion lead to higher performance, or were the participants in the higher-performing group able to be immersed easier? At the very least, the study indicates that immersion plays a role in user engagement and performance.

The article does not mention users reporting errors that were not intended to be errors, which, as mentioned, is a frequent occurrence in non-digital versions (e.g. [6, 9, 44]). The article states:

> The errors were selected after discussion with expert scrub nurses and a review of the literature on existing tools. [...] We also took into account the representability of errors in a virtual OR environment after discussion with VR engineers and we asked scrub nurse teachers to validate their pedagogical interest. Finally, 19 errors were introduced, related to hygiene and risk of infection (n = 12), disruptions to the surgical procedure (n = 4), identity monitoring (n = 1), trophic lesions (n = 1) and medical risk (n = 1). For each one, the level of risk was evaluated by two teachers and two scrub nurses and we differentiated between moderate- (n = 12) and high-risk errors (n = 7). [43, p. 2]

Based on this, the authors seem to have done a thorough job when designing their scenario and deciding on which errors to include, so did they manage to avoid these situations altogether? This question was asked to the lead author, who quickly responded via email (M.-S. Bracq, personal communication, October 22, 2021). Users had indeed discovered errors that the authors had not imagined, in addition to the ones that were implemented. The lead author also mentions her amazement at how people interpreted the environment and situation differently according to their own points of view and experiences. As a method for handling the issue, the authors had decided not to reveal a participant's "score" to them after the simulation — meaning how many errors they discovered out of all the possible errors. Instead, they focused on the oral debriefings where selected errors were discussed — both the "intended" and "unintended" ones. This solution did lead to some feelings of frustration among participants as they were curious about their "score." However, the VR simulation was reportedly generally well accepted and motivated participants [43]. This supports the need for a debriefing after the simulation, which is common in simulation training [45]. Although unintended errors were found, these occurrences did not seem to affect user immersion or engagement, nor acceptance of the VR application itself. In other words, a VR Room of Errors seems feasible, even with unintended errors — but no studies have been found examining VR implementations of Rooms of Errors supporting multiple concurrent users.

Chapter 2

Background

In this chapter, terms and concepts are further defined, and a theoretical background is established. Overarching topics include simulation training in general, virtual reality and computer graphics, and game design.

2.1 Room of Horror — BSc prototype

The Room of Errors VR prototype that was developed as part of a Bachelor thesis [24] was meant to serve as a starting point for further development. It simulates a traditional Room of Errors simulation training, where a relatively static and stationary room is set up with a set of predefined errors. Participants are given the task of discovering as many errors as possible within a given time limit. As mentioned in Section 1.1, the simulation training method is called "Room of Horror" in some studies, which was the name used for the prototype. The name was changed for the Master's project after discussions in the design team.

The prototype consists of two main rooms: a virtual operating room (OR), which is intended to be used by operating room nursing (OR nursing) and nurse anesthetist students, and a virtual postoperative ward, which is intended to be used by intensive care nursing and nurse anesthetist students. The OR is shown in Figure 2.1, and the postoperative ward is shown in Figure 2.2. Users are first given the opportunity to go through a short tutorial for using the application, introducing them to their ability to move in the environment, and marking potential errors. After the tutorial, the HMD is given to a lecturer or simulation training facilitator, who selects the errors that are to be active in the room. The HMD is given back to users, who then explore their respective rooms and mark what they believe to be errors by using their hand controllers. After they choose to finish, or their time has expired, they are shown a "debrief" where the errors they found are marked green, and the errors they did not find are marked red. Figure 2.3 shows an error that has been marked on the left and the same error not marked on the right — and Figure 2.4 shows the result in the "debrief." Users are also shown a brief explanation of the error. If users marked objects that were not intended to be errors, the objects are marked yellow in

Figure 2.1: Operating room in the prototype

Figure 2.2: Postoperative ward in the prototype

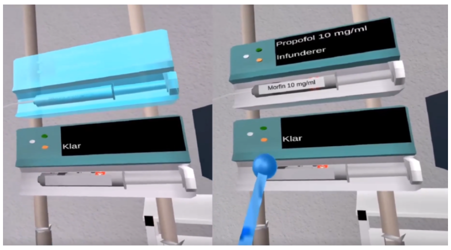

Figure 2.3: An error has been marked on the left in the prototype

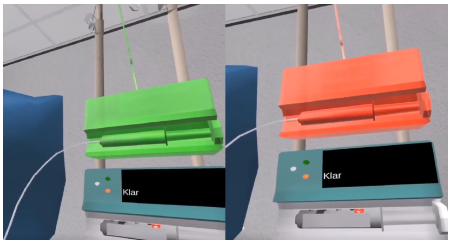

Figure 2.4: A user is shown that the error was correctly found, or not found

the debriefing.

In total, 11 different errors were implemented. Some errors can be reused in different rooms or at different locations in the same room. For example, the error "*Hand disinfectant container is empty*" was used twice in the OR as it is common to have a container on each side of the room, and twice in the postoperative ward. With the 11 errors available, ten errors can be activated in the OR, and ten can be activated in the postoperative ward. This number was considered adequate to give an impression of the errors that were possible to implement in future developments.

The resulting bachelor thesis received an award for the best bachelor thesis among the graduating bachelor level students of Computing and Information Technology of 2020 at HVL. The authors also received the NITO award from NITO Hordaland for the bachelor thesis with the highest innovation potential for 2020 [46].

At the start of the current master thesis' project, the prototype provided a starting point for development. However, several frameworks and specific elements — such as user input — that had been used were now considered outdated. At the time of developing the prototype, VR implementation in Unity and the 2019 OpenXR standard was still relatively immature. Therefore, the decision was made to use more stable frameworks, considering the short development period of around three months. In addition, the prototype had occasional frame-rate issues, which needed to be addressed. However, the prototype was developed with future further development in mind, especially when it came to the project's code and structure — meaning that most of the prototype's code would be reusable, with some changes. Notable changes in code and functionality from the prototype are discussed throughout chapter 4.

The changes made to frameworks are discussed in section 4.4. Another significant change between the prototype and the application presented in this thesis can be found in its graphical appearance and its graphical fidelity. Most of these changes are discussed in section 4.6.

2.2 Computer Graphics

This section briefly summarizes computer graphics and the process of generating an image that is displayed on computer screens. It is not meant as a complete introduction to the subject, but rather a glimpse into the number of calculations and data processing required to produce images. These operations are performed several times per second, which results in what is perceived as moving images.

Han [47, p. 1] says simply that "Computer graphics refers to the process of generating images using computers." For three-dimensional (3D) graphics, 3D mathematical representations of objects — vertices and vectors in 3D space — are given as input, various calculations are performed, and an image is produced. These images are often called *frames*, and when a sequence of changing frames is shown on the screen, the illusion of movement is created. For a Meta Quest HMD, this sequence is ideally displayed at a rate of 72 generated frames per second, which is the upper limit for the Quest HMD's screens. The required calculations are often performed on a Graphics Processing Unit (GPU), a processor specialized for graphics and optimized for parallel polygon processing.

3D objects are defined by a *polygon mesh*, which in essence describes a collection of vertices in 3D coordinate space and the polygons these vertices create, which together approximate a surface [47]. The simplest polygon is a triangle, which among other things, has the added benefit of always being planar, which contributes to the triangle being relatively easy and efficient for computers to process and render. The number of vertices in a polygon mesh — or the *resolution* of the mesh — has consequences. The more vertices, the more detailed the 3D models become, and the more time is needed by the computer to process the mesh — which, again, happens several times per second to generate sequences of frames. As Han states:

> There is a trade-off between accuracy and efficiency. As the resolution increases, the shape of the mesh becomes closer to the original smooth surface, but the time needed for processing the mesh also increases. [47, p. 19]

This trade-off is visualized in Figure 2.5. Figure 2.6 shows a polygon mesh of a doctor used in the final Room of Errors application. This mesh is advertised as *"low-poly"* [48] — a term used to describe meshes with a relatively small number of polygons [49]. These low-poly meshes are often used in computer games as they are relatively efficient to render but often have a characteristic unrealistic look. The doctor mesh appears simple but consists of 1719 vertices and 3358 triangle polygons.

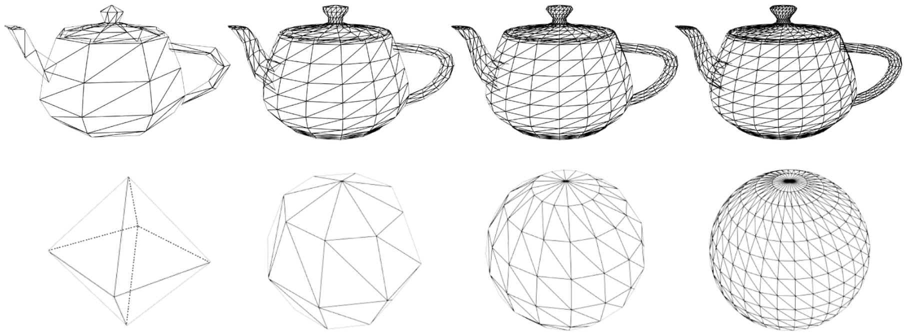

Figure 2.5: The trade-off between mesh resolutions: A low-resolution mesh is fast to render but does not approximate the original surface well [47, p. 19] (Image: ©2018 medialab-ku, MIT License [50])

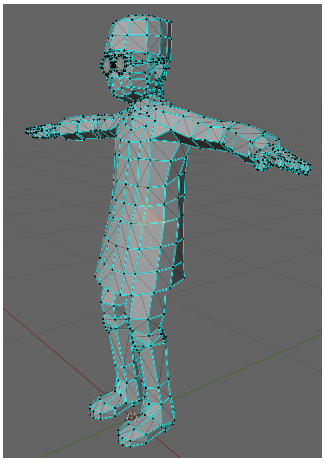

Figure 2.6: Polygon mesh of a doctor used in the Room of Errors

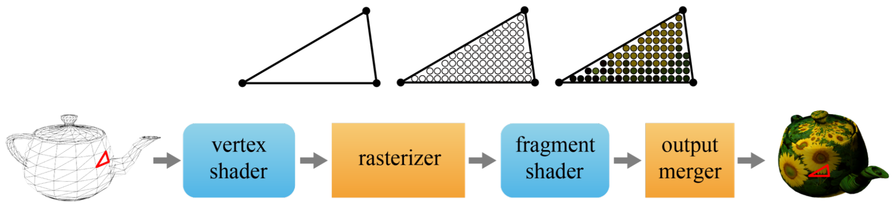

Figure 2.7: The main stages of the rendering pipeline [47] (Image: ©2018 medialab-ku, MIT License [50])

In the process of rendering a frame, these polygon meshes are passed to the GPU, where they go through the series of stages called the *rendering pipeline*. The pipeline consists of several stages, in which the output of one stage is taken as the input for the next stage [47]. The simplified main stages of the rendering pipeline can be seen in Figure 2.7. Notably missing from the figure is the stage prior to the vertex shader, which includes any operations or calculations performed on the CPU before the results are passed on to the GPU — for example, handling user inputs (including tracking the user's movement), animations, or collision detection. While not as clearly defined as the "proper" shader stages, the number of operations can significantly effect an application's performance. This stage is sometimes called the Application stage [51].

The vertex and fragment shaders in the pipeline are programs that are most often run on a GPU. The vertex shader performs operations on every vertex it receives or has stored in its vertex buffers before sending the results to the rasterizer. The rasterizer creates triangles from the vertices it receives and translates them into *fragments* — pixel locations that will potentially be translated into actual pixels seen on the computer screen. The results are passed to the fragment shader, which performs operations and calculations such as texturing and lighting on each fragment to decide its color. After being colored, the fragment is passed to the output merger stage, where more operations are performed to determine whether the fragment should be displayed at all or be combined with other fragments.

As has been repeatedly stated, this sequence is performed for every frame, several times per second — again, for the Meta Quest, 72 times per second. As also mentioned, the doctor mesh shown in Figure 2.6 consists of 1719 vertices and 3358 triangles. A typical frame can contain a large number of meshes that are more or less complex than this mesh — potentially hundreds or thousands of them — each requiring more or less complex operations. This means that every frame potentially can require performing operations on hundreds of thousands or even several millions of vertices — and this must be done a large number of times per second. In other words — to summarize — computer graphics can require large amounts of resources, a requirement that is further stressed in VR.

2.3 Virtual Reality

VR has seen significant progress in the last few years after today's generation of VR technology was made available to a broad audience with the commercial launch of HMD devices such as the HTC Vive in 2016 [52] and Meta Quest in 2019 [53]. These devices allow users to experience highly immersive environments and allow for high degrees of interaction and freedom within the surrounding virtual world. VR saw significant commercial growth in 2018 and 2019 [54], and it is expected that increased use of VR in education will strongly contribute to this growth continuing for the foreseeable future [55].

In recent years, virtual reality simulations have increasingly been used for training applications for a wide variety of professions, such as surgeons, firefighters, and pilots [56]. There is, however, a paucity of studies on the use of VR in nursing studies [35].

2.3.1 The virtuality continuum

Some definitions and disambiguations are required. In later years, the term VR has increasingly come to mean what is also called *immersive VR*, which is experienced through HMDs that block out external sensory information. However, the term VR has been used to describe several different types of computer-simulated environments [35], and there is still ambiguity and a non-homogenous understanding of what technologies can be considered "immersive" [19]. As an example, Paiva et al. [57] consistently use the terms Virtual Reality and VR when describing their application, which is interacted with on a desktop computer and is described as immersive. No mention of HMDs can be found.

In 1994, Milgram & Kishino defined the *Virtuality Continuum* [58], shown in Figure 2.8. According to Suh et al. [59], *Augmented Virtuality (AV)* in the spectrum can be used interchangeably with VR. In VR, real objects are added to virtual environments. On the opposite side of the spectrum is Augmented Reality (AR). In AR, virtual objects are added to real environments by viewing the world through a digital lens (e.g., glasses with small transparent computer screens or a mobile phone). Both VR and AR are encompassed by the term Mixed Reality (MR) as the space where physical and virtual worlds co-exist and where real and virtual objects are presented together [59]. In later years, the term Extended Reality (XR) has also seen increased use. The term defines a superset that includes the whole spectrum defined by Milgram & Kishino [58]: MR, VR, AR, and also "the complete real" and "the complete virtual" [60]. This continuum does not suffice to separate virtual content displayed on a computer screen, and more immersive virtual content experienced using HMDs.

Suh et al. [59], therefore, further define VR as *Non-immersive VR* and *Immersive VR*. Non-immersive VR is displayed via a computer screen and uses traditional media such as keyboards and mice for interaction — and immersive VR uses HMDs to encompass users in a virtual environment while blocking out visual cues from the user's physical environment. They also include the Cave Automatic Virtual Environment (CAVE) as immersive VR. The CAVE systems place users in a physical room where the actual walls act as "monitors." This effect is achieved by using projectors potentially aided by users wearing 3D

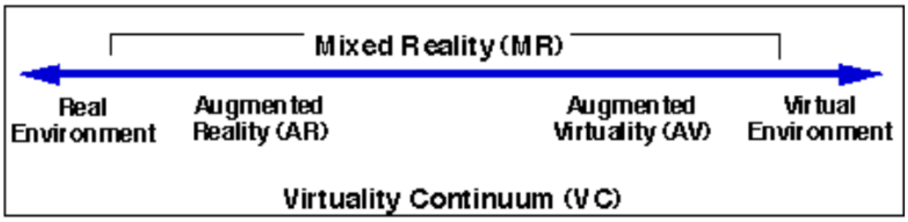

Figure 2.8: Milgram & Kishino's Virtuality Continuum [58]

Figure 2.9: Cave Automatic Virtual System (CAVE) [61]

glasses, resulting in the user being surrounded by a virtual environment. An example can be seen in Figure 2.9.

Radianti et al. [19], in a meta-review of the application of VR in higher education, explicitly consider CAVE systems as *non*-immersive because users can still recognize the screens surrounding them. Therefore, in the review, these systems are equated with virtual environments experienced on a desktop computer. It can be argued that both CAVE and desktop systems can provide immersive experiences in different contexts. As Radianti notes [19], the concept of immersion can change over time. In the early 1990s, computer games displayed on a small CRT monitor with a resolution of 320x160 pixels using 2D sprites were considered highly immersive. The concept of immersion is further discussed in section 2.7.

For this thesis, the term VR refers to virtual environments experienced using

HMDs.

2.3.2 VR technology

Bailenson [62] defines VR as working by executing three technical elements as flawlessly as possible: *Tracking*, *Rendering*, and *Display*.

Tracking describes the process of measuring body movements and rotations, and can be linked to the concept of Degrees of Freedom (DoF), which is a defining feature of HMDs. 3-DoF devices can track rotational movements but not translational movements — meaning the device can track if a user turns or tilts their head, but not if a user moves their head — while 6-DoF devices can track both rotational and translational movements [63]. Tracking can significantly impact a user's sense of presence [64].

Rendering is the process of taking a 3D model and 3D environments — which consists of mathematical data such as coordinates and vertices as described in section 2.2 — and translating it to a graphical, visual representation. This process must be repeated for every new frame, taking into account the user's tracked position [62]. As mentioned in section 2.2, this means 72 times per second on a Meta Quest HMD.

Display is the process of finally showing the rendered images to the user. Modern HMDs contain one or more display screens that the user views through two lenses — one for each eye. The two separate lenses give two different perspectives resulting in the HMD creating *Stereoscopy* — the illusion of depth [65]. Each screen must be able to display images at a high resolution, and the images must be refreshed frequently [62] — the Meta Quest HMD screens display images at a resolution of 1440x1600, with a refresh rate of 72 Hz.

Today's commercially available HMDs are roughly separated into three: 3-DoF devices, computer-powered 6-DoF devices, and standalone 6-DoF devices. 3-DoF devices include the Oculus Go [66] and smartphone systems such as Google Cardboard [67] and the Samsung Gear VR [68]. These devices are not discussed further in this thesis.

Computer-powered 6-DoF devices such as the HTC Vive [69] and HP Reverb G2 [70] are HMDs that are connected to a nearby computer through a physical wire. Tracking is performed by the HMD and potentially other external sensors. This information is sent to the computer, which performs calculations and goes through the rendering process. The resulting images are then returned and shown to the user in the HMD displays. Since the "heavy" and demanding operations of the rendering process are performed by the computer using its' technical resources, these HMDs can provide experiences of high graphical fidelity, limited only by the connected computer's technical capabilities.

Standalone 6-DoF devices such as the Meta Quest [22] and the emerging Pico Neo series [71] are HMDs that conversely do not require a connected computer. All three elements of VR — Tracking, rendering, and display — are performed on the HMD device itself. The devices themselves are essentially smartphones with an HMD built around them — in fact, the Meta Quest uses an Android-based operating system. This means the graphical capabilities of these systems are somewhat limited.

The different devices' computational capabilities must be considered, as VR experiences can have higher demands and requirements than "traditional" 3D experiences on a computer screen. If tracking, rendering, or display is not done fast enough or well enough, VR users can experience simulator sickness [62]. Essentially, the mind has an idea of how the body is positioned and how it is moving. In VR, HMDs can show the user something different. For example, the body physically moves, but the VR environment reflects the movement even a tiny bit delayed. Alternatively, the body stands still, but the VR user can see themselves moving forward in the HMDs. In these situations, the eyes and the body give the mind contradicting information. This contradiction can cause a VR user to potentially experience nausea and an uncomfortable feeling similar to motion sickness.

This challenge leads back to the discussion of 3D mesh resolution in section 2.2. Standalone HMDs have limitations when it comes to computational capabilities, which in graphical terms means it has limitations when it comes to the number of vertices and polygons that can be processed per frame before the user starts to experience a lower *frame rate*. The device cannot process all the required data fast enough, which means frames take longer to generate, so the device cannot generate them as frequently, and tracking, rendering, and display cannot be performed fast enough to prevent VR sickness. Reducing the number of meshes in a scene or the meshes' resolution is one way to reduce the complexity of rendering a frame, but this leads to lower graphical fidelity and realism. In other words, when developing applications for standalone VR HMDs, developers must consider the trade-off between accuracy and efficiency, as described in section 2.2.

2.4 Simulation training

The education of nurses and nurse specialists is in continuous development. Practical skills training with anatomic models and role-play have been a part of nursing studies for many years — the use of training mannequins and models has been described in literature from the 1800s [13]. The most notable mannequin for health personnel training — Mrs. Chase — was created in 1910, and the 1970s saw the creation of specialized mannequins with realistic respiration and heartbeats [72]. Today's training mannequins are highly advanced and can be controlled by external computers, offering a breadth of functionalities.

A relatively new addition to these studies is the concept of a more structured Clinical Simulation Training [45]. These simulations combine several forms of teaching and learning to mimic the reality of clinical environments to a higher degree. Participants go through scenarios where they perform procedures as they would in real life and practice decision-making and critical thinking through role-playing. The training often includes mannequins and other tools such as video recordings of the simulation sessions. The training method creates new opportunities to try different approaches to solve problems and tasks and gain insight into what works — and does not work — in real life [73]. Examples of clinical simulation training can be seen in Figures 2.10 and 2.11.

Clinical Simulations at SimArena follow the frameworks suggested by Jeffries in

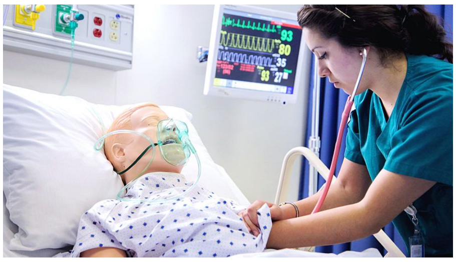

Figure 2.10: Simulation with a training mannequin (Photo: Monirb, CC BY-SA 4.0 [74])

Figure 2.11: Simulation of a surgical procedure (Photo: COD Newsroom, CC BY 2.0 [75])

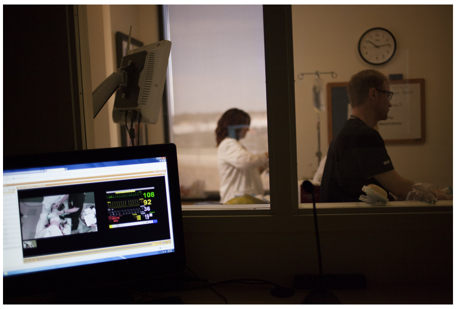

Figure 2.12: Simulation training from a facilitator's perspective (Photo: COD Newsroom, CC BY 2.0 [76])

Figure 2.13: Simulation training facilitators (Photo: COD Newsroom, CC BY 2.0 [77])

[45] and later publications. Before starting, students are given a briefing about the situation and basic information about the scenario and patient. During the scenario, the patient is represented by a mannequin, and simulated vital signs such as heart rate and blood pressure can be seen on a patient monitor. These values and the mannequin are controlled by a simulation facilitator seated in a different room, observing the session via cameras. The facilitator can communicate with participants via microphones in the mannequin's ear and a speaker in the mannequin's mouth — and at the same time communicate with lecturers present in the simulation wearing a headset with earphones and a microphone. The simulation training facilitator perspective can be seen in Figures 2.12 and 2.13. The scenario itself is recorded on video. It is common for students to go through the same scenario twice, allowing students to experiment and explore — which may cause them to "fail" the scenario, meaning that they, for example, did not perform required interventions fast enough. The scenario is stopped before any harm would come to the patient in real life.

Studies have shown that participants can remember mistakes they made in detail several months after the simulation. Making mistakes in simulations can have long-term effects on participants, both physically and cognitively, even though they are fully aware they are in a simulation [78]. Therefore, it is important not to "go too far" for the sake of the participants. After both scenario sessions, the students and teachers gather for a debriefing, where they talk about what happened and what could have been done differently as they watch back key moments from the recordings. By going through the scenario twice, the participants are given the opportunity to correct their mistakes and end the day feeling a sense of mastery instead of negative emotions.

Foronda et al. [34] show that *virtual* simulations positively impact student learning outcomes. However, they also state that there is a discrepancy in the meaning of the term between studies, most often referring to screen-based and not VR-based experiences, which this current thesis studies.

2.5 The Room of Errors

To design a virtual Room of Errors, the simulation must first be further defined beyond the core concept — digital or not. The idea of a Room of Errors seems to be relatively new. In literature searches, the earliest academic study using the concept that has been found was published in 2011 [79]. Attempts at finding recommendations or standards for setting up such a room have been unsuccessful. Several articles can be found (e. g. [6, 7, 9, 10]), but the authors rarely provide reasonings behind why they set up the room in the manner they did or explain the errors they included. There does not seem to be specific recommendations or standards for preparing the rooms. The basic idea seems clear, however: Participants enter a room where errors or breaches of patient safety routines are set up and try to find as many of the errors as they can within a given time.

Several questions remain. What instructions are the participants given beforehand? Which errors or types of errors should be included? How many errors should the room contain? Should errors be obvious and exaggerated, so participants can become aware of them and can discuss as many errors as possible? Or should errors be more hidden and vague, so participants can be surprised that they did not find them, which hopefully sparks discussion? Should the participants be told which intended errors they found or did not find after the simulation? Should participants examine the room on their own or in a group? How much time should the participants be given to search the room? Does an "optimal" way of preparing a Room of Errors simulation exist? If these questions have clear answers, do the same answers apply to a VR version?

As mentioned, several studies using a Room of Errors can be found, which give varying answers to some of these questions. The number of prepared errors varies from 9 [6] to 95 [80]. Most simulations give participants around 10-15 minutes. Most studies mention or describe a debrief following the simulation — some immediately after (e. g. [81, 82]), some after a period of time (e. g. [9]). In most of the studies, participants are told intended errors that were not discovered (e. g. [6, 9]), but this is not clear in all articles (e. g. Shekter et al. [81] state that a debrief took place, but not whether participants were told of undiscovered errors). Most studies state that the simulations were performed in groups of varying sizes (e. g. [44, 82]). Clay et al. [9] asked participants to first go through a Room of Errors alone and some time later in a group, which gave the participants opportunities for teamwork. Conversely, Farnan et al. [6], in one of their two sessions, asked participants to enter the room in groups of 3, but with observers ensuring no conversation between the participants occurred. As for the errors themselves, some studies explicitly list all implemented errors (e.g. [9, 10]), while some contain a non-exhaustive list of examples of implemented errors (e. g. [79, 81]).

As mentioned in section 1.7, only one article has been found describing a VR implementation of the concept by Bracq et al. [43]. In this version, participants were first shown a VR scenario describing movement in the application and were then briefed about their task. Participants were also given a paper version of the patient's file. Nineteen errors, selected after discussions with domain experts and a literature review, were included in the application. The specific errors are not described, but broader categories of errors are given. Participants went through the room on their own, not in groups. The VR sessions lasted for a total of fourteen minutes. Each session was followed by a short individual debriefing of 5-10 minutes, followed by a 90-minute collective debriefing. The participants themselves asked to repeat the simulation, which was organized six months later.

In Bracq et al.'s study [43], the participants were not given a "score" or told which of the intended errors they did or did not find — the objects and errors they indicated during the simulation were instead used as a basis for discussion during the later debriefings. Notably, not being given a score led to negative feelings among participants, such as confusion and frustration. In the article's conclusion, the authors state that the simulation was generally well accepted and motivating. However, participants did express difficulty with identifying some elements and identifying details and had questions regarding interpreting the errors. Participants also experienced difficulty with the controllers. The article answers some of the questions about preparing a VR Room of Errors. However, as the article's focus and field of study is not software engineering, it does not give a complete picture from that perspective.

2.6 Simulation fidelity

Fidelity in a simulation context refers to the degree to which objects in the simulation mimic reality [13]. A simple, low-fidelity, inanimate mannequin without any additional functionalities is considered of lower reality than a high-fidelity mannequin with advanced functionality such as emitting sounds from a simulated heartbeat and moving its chest to simulate breathing. If this term were to be translated to the context of serious games, one could consider, for example, graphical fidelity. A 2-dimensional pixel-graphics game would be regarded as low fidelity, and on the opposite side of the spectrum, a 3-dimensional VR application using realistic 3D models would be considered high fidelity. One could also consider fidelity in terms of a user's interaction with the virtual environment. In this context, an imagined application using abstract menus to perform actions could be regarded as low fidelity. A VR application allowing users to interact with an environment by picking objects up and pressing "physical" buttons would be considered high fidelity.

For simulation training in general, realism and fidelity have been topics of research and discussion [15, 83]. Several studies indicate that a simulation's fidelity does not necessarily affect educational effectiveness. For example, Matsumoto et al. [17] compared results between two simulations for training endourological[1] skills — a high-fidelity simulation using advanced mannequins and real endourological instruments, and a low-fidelity simulation using a coffee cup and straws. Their post-intervention ability was rated by staff examiner observers and time spent performing a similar task. There was no significant difference between participants in the two simulations, but both simulation groups improved their ability compared to a third group receiving only traditional didactic sessions. Other studies even show that high fidelity led to worse learning outcomes in some aspects than a low fidelity simulation [85]. Kim et al [86], in a meta analysis, finds that simulation training has strong educational effects, but that the effect *is not proportional* to the simulation's level of fidelity. Regardless of fidelity, a participant's ability to involve themselves and engage cognitively does seem to be important for learning outcomes [15].

2.7 Immersion and Sense of Presence

As mentioned in Section 1.1, immersion is indicated to be important for learning outcomes [15] and is considered a significant element in the VR Room of Errors simulation. *Immersion* and *Sense of Presence* are two terms that, in some studies, have been used interchangeably as synonyms [87]. For this thesis, the two are considered to be different and are used as described by Jennett et al. [40].

2.7.1 Definitions

Immersion refers to being engaged and engrossed in the experience. An immersed user can lack awareness of time progressing outside the experience, lose awareness of the real world, and experience a feeling of involvement and a sense of *being in* the virtual environment [40]. Evidence about the immersion process itself is limited: what factors affect it, and what roles do those factors play [36] — how exactly does a user become immersed?

Sense of Presence refers to the sense of actually physically being in the virtual environment [40]. Studies have also described the feeling of agency and being able to interact with the virtual environment [56], and an illusion that the virtual events happening in front of the user are real [87].

Jennett et al. [40] argue that immersion is an experience in time and that presence is a state of mind. Games with simple graphics such as Tetris do not involve presence and a feeling of "being there" but can still be highly immersive and cause you to experience time loss. Contrary to this, a sense of presence is possible without immersion — a user can feel present in a virtual environment without losing their sense of time if, for example, the environment and possible interactions are perceived as boring and uninteresting.

[1] Endourology is a branch of urology that includes all minimally invasive urological surgical procedures. These techniques use small cameras and instruments to access the relevant organs instead of the large incisions used in open surgery[84].

2.7.2 Increasing immersion and presence

For the current thesis, an important question is: How do we increase immersion and sense of presence? Makransky & Petersen [88] summarize factors or determinants affecting presence, such as representational fidelity of the environment — essentially graphical realism — and the degree of control afforded by the environment, or as previously mentioned, the ability to interact with the virtual environment. However, they note that different individuals may experience different amounts of presence in response to the same experience, indicating that elements may have different effects for different users.

Winkler et al. [89] argue that despite the growing attention in research to immersive technologies like VR, there is no common understanding of which factors affect immersion or what actually makes VR immersive. The authors, therefore, attempt to define these factors in a qualitative study and find eleven immersion predictors, of which nine have been described in earlier research: Visual and auditory inclusion; Translating actions from physical to virtual reality; Transportation; Distracting aspects of virtual reality; Concentrated attention; Losing sense of time; Affective involvement; Control; Interaction between users; Perception of other avatars; and Shared Experience. The authors explicitly mention that the study has several limitations but suggest that their findings can be useful in future research.

Servotte et al. [36] confirm some of these — the importance of multi-sensory cues such as sounds and haptic feedback — and also mention the fidelity of the scenario itself — the realism of the situation and scenario. The authors also suggest that a well-defined pre-briefing can facilitate the immersion process. This suggestion is similar to Tun et al.'s [18] study, which suggests that clarifying limitations at an early stage allows participants to more easily suspend their disbelief in non-digital simulations, as mentioned in section 1.2.

In his book "*The art of game design*" [90], Schell adds a different perspective — that of a game designer. He mentions six *Presence Breakers* and suggestions for mitigating them: Motion Sickness, Counter-Intuitive Interactions, Intensity Overload, Unrealistic Audio, Proprioceptive Disconnect (the user's movement is not represented or replicated well), and Lack of Identity. Conversely, six *Presence Builders* are defined, including suggestions on how to implement them: Hand Presence, Social Presence, Familiarity, Realistic Audio, Proprioceptive Alignment (the user's movement is represented well), and Comedy. These twelve concepts are presented as facts, seemingly without references to scientific studies or similar materials supporting their definitions. They are, however, presented alongside specific examples from commercially successful VR games, which lends them credence.

Factors affecting immersion and presence seem numerous and are presented with varying degrees of evidence. Graphical realism appears to be important for immersion, but realism in other elements such as the situation and the scenario itself and auditory cues also play a role. In addition, the ability to affect the environment and interact with it is frequently mentioned when discussing presence.

2.7.3 Social Presence

VR has a number of social applications and offers a unique level of *Social Presence* compared to other forms of technology-mediated communication [91]. The term *Social presence* refers to the subjective experience of being present with another person, whether real or artificial [91, 92].

Studies have indicated that social presence in VR is associated with higher motivation of the learner and positive results in communication and education outcomes [91, 93, 94]. Communication between users can also reduce their awareness of the technology, which allows them to focus more on the experience [95]. From the less scientific perspective of a game designer, Schell believes that social presence makes virtual environments easier to accept as real [90]. These two factors indicate that allowing for collaboration between simultaneous users in the Room of Errors simulation may help users accept the simulated errors and the environment. In other words, answering RQ2 may be important to answering RQ1.

2.8 Serious Games

The term *Serious Games* is used for digital games where the primary goal is not simply entertainment, but education, outreach, or training [96]. These games attempt to take advantage of motivational factors found in video games to improve learning [97]. The development and design of these games require different principles and methodologies than "traditional" games made for entertainment purposes, as the games themselves need to include engaging and motivating environments [98]. These elements must also align with the overarching aim of learning [99].

Many articles can be found studying the various effects of VR on, for example, learning outcome or usability. However, very few articles and very few game developers in general report details of the design frameworks [100] or methodologies used to develop their games [97]. From an academic perspective, Neo, Won & Shepley [39] conclude in a literature review that researchers typically describe the environments they have designed but rarely describe design decisions and design processes behind their applications [39]. Radianti et al. [19], in another literature review, suggest that a high contribution and impact in the field of VR for higher education can be expected from future articles with sound theoretical and technological foundations. They propose that future studies are explicit in describing the design and development process.

2.9 Game Engines

Modern computer games are not only entertainment products but also complex technical systems that aim to give users a satisfactory, often entertaining experience — and potentially educational in the case of Serious Games. From a software architecture perspective, a game's engine — the underlying software — is a complex system of intertwined layers, handling components such as input devices, graphics rendering, collision detection, and animation systems [101]. However, it is important to establish that the term *game engine* is often used to refer to both the core runtime of a compiled game and the framework or tools that facilitate content creation and production of the game. Messaoudi, Simon & Ksentini [102] propose clearly distinguishing between the two aspects: the game engine as a *framework* for game creators and the game engine as a set of software and data that run on devices to provide the game to end-users. For this thesis, the term *framework* will be used to describe the set of tools facilitating production, but the term *game engine* may be used interchangeably to describe both aspects.

2.9.1 Choosing a game engine

Developing an entire game engine "from the ground up" can be time-consuming and demanding. Therefore, it is common to use an already established and developed engine and framework, of which there are several. Wikipedia's List of game engines contains several hundred entries and is explicitly described as not exhaustive [103]. Of these, the two most popular engines and frameworks are Unity [21] and Unreal Engine [104], at least when it comes to commercial use [101, 105]. They have also been described as the most powerful and most developed game engines available [106].

Unity and Unreal Engine are both possibilities for developing a Room of Errors in VR. They are both free to use for personal and internal project development (though licensing fees or royalties are required when publishing or monetizing the product above a threshold). They both offer extensive features and capabilities while simultaneously being relatively easy to use. Importantly, they both support development of VR applications. Choosing between the two can be a difficult decision, exemplified by the several resources available (e. g. [107, 108, 109, 110]) — asking Google the question "Unity or Unreal" provides approximately 162 000 results.

For this project, it is strongly believed that using either framework could lead to similar results. However, it was decided to use Unity as the game engine and framework. Unity seems to offer slightly more learning resources, slightly more community activity (e. g., forum posts answering questions), and a slightly larger asset store (e. g., 3D models and tools to aid development) [107]. While framework familiarity is not a *deciding* factor — most concepts should be transferable, and both frameworks offer numerous tutorials and extensive documentation — it is still a factor. Having developed the prototype BSc version in Unity, the author of this thesis is more familiar with development in Unity, which should result in a more efficient initial phase of development. Familiarity with a framework's programming language is also a factor. Unity mainly uses the C# programming language, while Unreal Engine mainly uses the C++ programming language, and the author is more experienced with C# than with C++. Lastly, Unity seems to offer a more versatile and complete integration of tools for XR development — however, without more experience with Unreal Engine, this is an unconfirmed perception.

2.9.2 Unity and the Entity-Component Architecture

As mentioned in section 2.9.1, an essential aspect of a game engine is its set of tools facilitating production. For the Unity engine, the most important tool

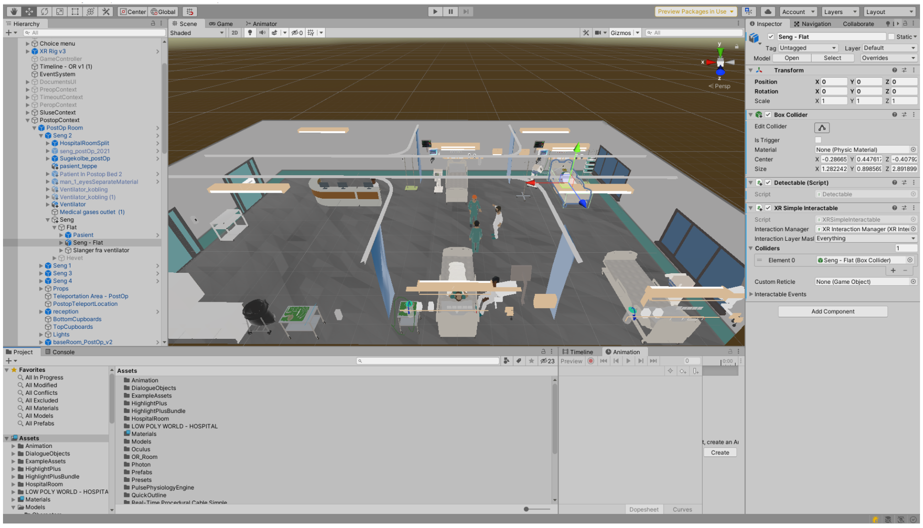

Figure 2.14: The Unity Editor

is the Unity Editor, which offers a graphical user interface used in creating and defining a *Scene*. The editor can be seen in Figure 2.14. The left section shows the Scene *Hierarchy*, which shows all objects that are in the active Scene in a Scene Graph-like structure. The right section shows the *Inspector*, which displays the selected object's properties and Components. The middle section is the *Scene view*, which displays the active Scene and allows developers to manipulate it graphically by, for example, moving or rotating an object.

Development in Unity uses an *Entity-Component* architecture, which is frequently used in game development [111]. In this architecture, developers, as much as possible, replace inheritance and polymorphy with *composition* when structuring their code [112]. The programming principle of low coupling is of high importance, leading to high degrees of reusability. Instead of several layers of inheritance, which can lead to large and confusing classes with many dependencies, code is separated into *Entities* and *Components*. Instead of Entities sharing code between two classes by having them inherit from the same class, they both own an instance of the same class — the same Component [112]. An Entity can contain one or more Components, and both Entities and Components can be used in different contexts. Components can define traditional object-oriented classes and objects, or specific traits or behaviors. Note that the Entity-Component architecture is different from an Entity-Component-System architecture, in which Components only contain data and no behavior [113] — but they both have similar motivations related to decoupling and reusability.

In the Unity engine, the *GameObject* is the most basic Entity. A GameObject Entity can have several GameObject-Entities as children, which can each have several Components and again contain several other GameObject-Entities. A GameObject child inherits its parent's transformation values in a *scene graph*-like structure, which means that a child will move or rotate in the scene if its parent is moved or rotated.

GameObjects in Unity are contained in Unity *Scenes*. A *Scene* can be viewed as a logical division of the game's contents. In practice, a Scene defines what should be loaded into the computer's or device's memory. When the Scene starts, the Scene's content is loaded in, and when a different Scene is started, the computer or device releases the resources that were needed by the previous Scene [114, 115].

Components define functionalities or behavior. Unity offers several Components, such as the basic *Transform* Component. This Component is part of all GameObjects and defines its position, rotation, and scale. If the GameObject is to be rendered, the *MeshRenderer* Component is added to it. If the game requires collision detection for the object (determining if an object has moved into another object or a raycast has hit the object), one of the several different types of *Collider* Components is added to it. In addition to these built-in Components, Components can be created by the game developer through programming code. Unity uses the C# programming language. Classes or Components that are to be used in a Scene must inherit from the Unity classes *MonoBehaviour* or *ScriptableObject*. Both of these classes contain the method `Awake()`, which is called when the script is loaded. However, only *MonoBehaviour* contains the methods `Start()`, which is called after initialization but before anything else happens, and `Update()`, which is called before each new frame is drawn in the game [116, 117].

Chapter 3

Research Methodology

To answer the thesis' research question, a methodology for research and application development was needed. As mentioned in section 1.5, the project uses a variant of the Design Science paradigm. In this chapter, this choice is explained, and Design Science is discussed. This is followed by an explanation of how the paradigm is applied to the current project.

3.1 Methodology requirements

Before starting development, a research methodology and development methodology had to be chosen. As described in section 2.5, several questions remain unanswered regarding the simulation experience and how it should work. Although preliminary evaluations of the BSc prototype and Bracq et al.'s study [43] indicated that the Room of Errors concept is viable in VR, several of the decisions made during development of the prototype were not necessarily well-founded or based on research but were instead based on what *seemed* logical at the time. This reasoning would not be sufficient to answer RQ1 regarding the VR application's design, neither in terms of the user experience and simulation flow nor the underlying software architecture.

The project would require searching for proven practices and effective VR elements in research literature; short and frequent iterations that provided working software that could be discussed and evaluated by domain experts who were part of the design team; and methods to evaluate the software more formally after a number of iterations. Ideally, the project would result in a working VR implementation of a Room of Errors simulation, which could be used as an experimental intervention in a PhD project, and potentially as part of a simulation training exercise for AIO nursing students. These requirements point to a methodology inspired by the Design Science paradigm.

3.2 The Design Science paradigm

The Design Science paradigm is fundamentally a problem-solving paradigm. When applied to Information Systems, it seeks to develop and evaluate a "purposeful IT artifact created to address an important organizational problem" [23]. The aim is not necessarily only to create an application or algorithm but to understand how to design and build them. As Hevner et al. [23, p. 98] state: "The design-science paradigm seeks to create 'what is effective,'" as opposed to other paradigms such as the behavioral science paradigm, which seeks to find "what is true."

The goal of design science research is to, as vom Brocke et al. summarize: "Generate knowledge on how to effectively build innovative solutions to important problems" [118, p. 521]. From a software engineering perspective, Engström et al. state: "The long term goal of much software engineering research is to provide useful recommendations on how to address real-world problems providing evidence for benefits and potential weaknesses of those recommendations" [119, p. 2646]. These two perspectives seem to align well.

Hevner et al. [23] establish seven guidelines for design science as applied to Information Systems, which can be seen in Table 3.1. Most importantly, the goal of design science is to design or create an *artifact* that solves **relevant** important business needs or problems. Hevner et al. [23] also emphasize that design is inherently an iterative and incremental activity, which is expanded upon by Hevner [120] in a commentary published three years later. In this commentary, he defines the three cycles of design research, which can be seen in Figure 3.1.

In summary, Hevner [120] emphasizes that constructing and evaluating the continually evolving design artifact must be based on **both** relevance and rigor. This means that the design process goes through several types of cycles and iterations. The need, problem, or application context may originate from the environment, which initiates the relevance cycle. The environment is then regularly involved in developing the artifact by providing requirements and being involved with "field testing" — where the environment studies and evaluates the artifact — which can determine whether new iterations are needed in this cycle. The rigor cycle on the other side of the diagram represents that artifacts are designed and evaluated using an existing knowledge base, which is revisited during development — and one of the goals of design science is to contribute something new, back to the existing knowledge base. The middle design cycle represents the building and development of the artifact, and this cycle sees more frequent and rapid iterations. Note the separation between the design cycle's *evaluate* activity and the relevance cycle's *field testing*. Hevner argues that the artifact must be thoroughly tested "internally" in experimental situations before being released to the more formal field testing — which means multiple iterations of the design cycle are needed before the artifact is "output" into the relevance and rigor cycles.

Guideline	Description
Guideline 1	Design-science research must produce a viable artifact in the form of a construct, a model, a method, or an instantiation.
Guideline 2	The objective of design-science research is to develop technology-based solutions to important and relevant business problems.
Guideline 3	The utility, quality, and efficacy of a design artifact must be rigorously demonstrated via well-executed evaluation methods.
Guideline 4	Effective design-science research must provide clear and verifiable contributions in the areas of the design artifact, design foundations, and/or design methodologies.
Guideline 5	Design-science research relies upon the application of rigorous methods in both the construction and evaluation of the design artifact.
Guideline 6	The search for an effective artifact requires utilizing available means to reach desired ends while satisfying laws in the problem environment.
Guideline 7	Design-science research must be presented effectively both to technology-oriented as well as management-oriented audiences.

Table 3.1: Hevner et al.'s seven guidelines for design science [23, p. 83]

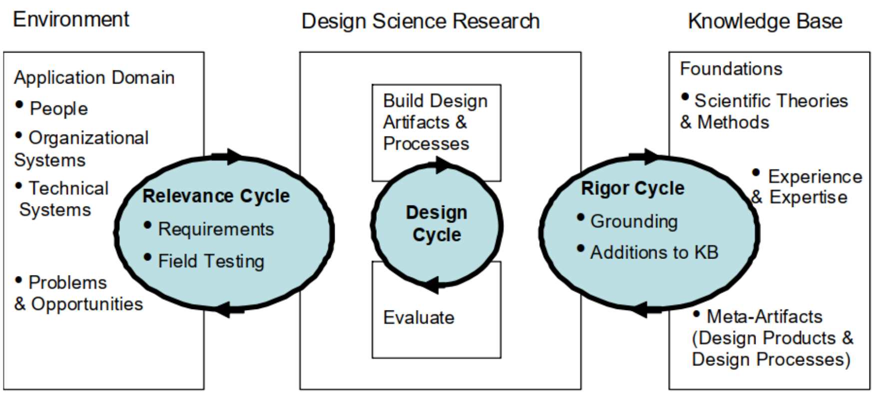

Figure 3.1: Hevner's design science research cycles [120, p. 2]

3.3 Design Science applied to the virtual Room of Errors

The project of developing a VR Room of Errors was initiated by researchers at SimArena, HVL. They were in need of VR applications that can be used to explore the use of "new technology" — in this case, VR — in the education of health care personnel. This situation fits well with Design Science's goal of designing an artifact to solve relevant and important business needs — guideline 2. For the current thesis, the environment from a Design Science perspective specifically required a VR implementation of a Room of Errors simulation. However, this was not the singular need for SimArena, as they have plans and ideas for several additional applications, which could benefit significantly from knowledge about how such applications could or should be designed. The aim of answering RQ1 is to examine how to design a specific type of VR application — but the process should also provide some insight into answering this greater, more significant problem of designing future, related VR applications.

This means that this thesis may not follow the Design Science paradigm completely but that the paradigm was adapted to fit a smaller scale problem. Therefore, SimArena and lecturers who may potentially use the VR application after development and who more rarely participated in the more formal evaluations of the application are considered part of the larger environment and as part of the relevance cycle in this adapted and scaled-down Design Science method. From this perspective, the environment — represented by SimArena — has frequently contributed with requirements and participated in smaller and larger evaluations. In other words, the environment has been frequently involved in the project, contributing to the project's relevance.

Notably, in a sense, design science's relevance cycle seems to differ from the agile development principles of involving the customer early and often, as Hevner argues that artifacts should not be output to the relevance cycle before multiple iterations and experiments in the design cycle [120]. For this thesis, the act of "outputting" to the relevance cycle for field testing is considered to be the act of more formal testing, as discussed in chapter 5 — but as mentioned, the environment was involved frequently during development, more in line with the agile development perspective. However, the two views do not necessarily need to be exclusive, as seen, for example, in Conboy, Gleasure & Cullina's Agile Design Science Research Methodology [121]. From an agile perspective, frequently involving the customer in evaluating requirements and the continually evolving software can be seen as the customer being part of the design team. From the design science perspective, however, the larger, potentially more formal field testing is output to the environment at large, not just the specific customer. With this in mind, the project's main initiator, PhD candidate Marit Vassbotten Olsen, is considered part of the design and development team as she frequently participated in "internal" iterations, as argued by Hevner [120], which included informal evaluations and iterating on requirements. From this more agile perspective, the application has been developed by frequent cooperation with domain experts over a period of approximately a year.

In a sense, the bachelor's degree project described in section 2.1 can be considered this project's first major iteration or development period. That project

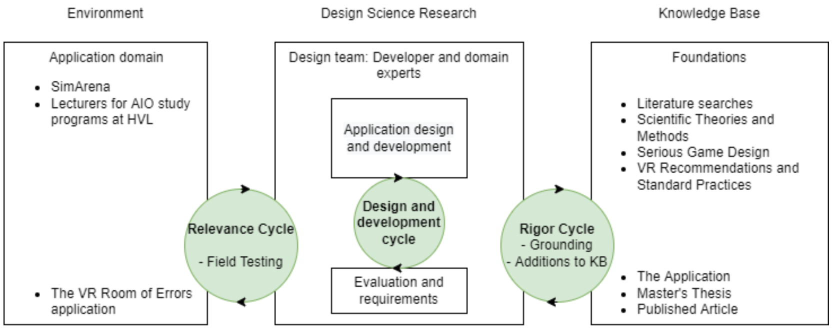

Figure 3.2: The Design Science Cycles applied to development of the Room of Errors application

was lacking in terms of research methodology and evaluation, as the Covid-19 pandemic prevented proper evaluation and field testing. As mentioned in section 3.1, the current expanded project requires more rigorous and well-founded reasons and decisions to answer RQ1. These requirements fit well with Design Science's rigor cycle and guidelines 5 and 6.

Existing literature and previous studies in the field of VR application development is used as a knowledge base. The databases used for literature searches include Google Scholar, IEEE Xplore, Engineering Village, and Web of Science. Several iterations of literature searches have been performed — initially to develop a “local” knowledge base consisting of existing design elements, principles, and techniques, and in later development iterations on a regular basis — at least once monthly — in attempts to discover newly published knowledge. The results of these literature searches are discussed throughout this thesis, most notably in chapter 4. The knowledge base has also been used to determine evaluation methods, which are discussed in section 3.5. As for the second aspect of the rigor cycle of the adapted, scaled-down Design Science — adding new knowledge to the knowledge base — the thesis aims to add knowledge by answering the thesis' research questions. An article reporting the results from the first user evaluations has been accepted for publication at the first IEEE International Conference on Cognitive Aspects of Virtual Reality [1].

Again, the Design Science paradigm is adapted and scaled down for a smaller scale problem. This thesis does not claim to be a complete literature review on how to design immersive VR experiences, nor does it claim to make a contribution of *significance* in the “areas of the design artifact, foundations, and/or methodologies,” as defined in Guideline 4. However, it does claim to make *a* contribution. The basic principles of Design Science's rigor cycle — as understood by the current thesis' author — have been followed.

The Design Science Cycles as applied to the current project can be seen in Figure 3.2. The middle design and development cycle can be seen from a different perspective in Figure 3.3. As touched upon, the design cycle has seen shorter and more frequent iterations than the other cycles of design science. Each iteration included an evaluation of the application, its design, and the development process in cooperation with the design team. The application's development is further described in section 3.4.

3.4 Application development methodology and process

Initially, the design team had plans to design and implement a reenvisioning of the Room of Errors concept. Early ideas included implementing fully simulated surgical procedures in scenarios with numerous branching paths depending on the user's actions. One of many examples is administering the correct medication A at the correct time X, leading the user down one path, itself consisting of numerous branches — while administering the correct medication A at the wrong times Y or Z would lead to different branches with different outcomes. At the same time, the user can instead decide to administer an *in*correct medication B at times X, Y, or Z, leading to completely different branches. This concept was ultimately decided to be outside the scope and objectives of the related PhD study.

However, there was still a desire to involve real-world stories and cases as a background for errors. Olsen, the PhD candidate for the related research project, is an experienced nurse anesthetist and simulation training facilitator. She has been faced several times with students who protest that events in the simulation seem far-fetched and that they would never happen in real life — "Someone would have discovered this error or said 'stop'". Being able to reply that these events did, in fact, actually happen was deemed to be important, both for learner motivation and for the sake of raising awareness.

This led to the idea of limiting the simulation to scenarios where a single adverse event occurs, and the user is given a single choice, leading to either desired or undesired outcomes. However, this idea was countered by the fact that very few — if any — adverse events can be broken down to a single, concrete action being the cause of the event, but that they more often than not happen due to several factors potentially happening all at once [4]. This was confirmed after studying numerous analyses of adverse events performed by Helsetilsynet [122] and conversations with the Section of Patient Safety at Haukeland Universitetssykehus' Department of Research and Development. In these conversations, they were more than willing to share their real-world data and reports on adverse events at the hospital (dependent on the appropriate approvals from ethics committees and people involved). However, they cautioned that their data was raw and may not provide a single, clear cause.

Therefore, after a period of planning and initial development for implementing scenarios, it was decided to scale the project down and redirect it to the traditional "static" Room of Errors concept. The real-world stories and cases were distilled and simplified to concrete errors. This decision refocused the development process and led to the application presented as part of this thesis.

At the start of development, the design team did not have a full vision of the final application. Therefore, development would require a methodology that allows for flexibility and evolving requirements. In addition, the application's design, features, and functionalities would need to be discussed and evaluated

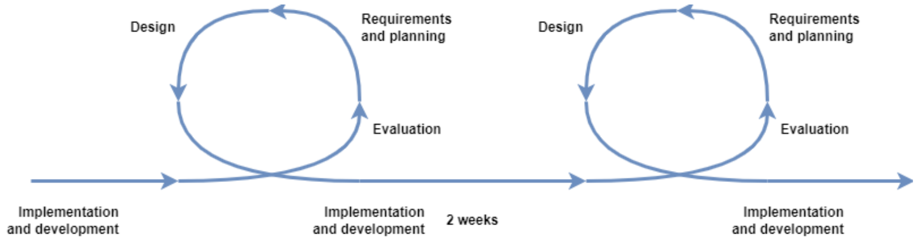

Figure 3.3: A model of the development process

frequently. The developer and other members of the design team would have to cooperate closely. Before and during development, members of the design team without a software engineering background frequently expressed that they were unsure of what could be accomplished in VR. Therefore, new ideas needed to be implemented to a functional state quickly in order to give them an impression of how the ideas could work — or if they likely would not work.

These development requirements naturally led to the principles of Agile development [123, 124]. The application was developed iteratively and incrementally. Development consisted of several small sprint cycles of typically two weeks, where new ideas and functionalities were implemented, or new intended errors in the simulation were added. The design team then met to evaluate the current state of the application. These evaluations led to deciding whether to iterate further on the new implementations or errors or discard them. This was followed by discussions of requirements and planning before making plans for the next sprint cycle, which then started. A model of the iterative development can be seen in Figure 3.3.

The application was developed using the Unity game engine [21]. Development started with version 2020.3.16f1 but was updated to version 2020.3.23f1 in November 2021. Unity was not updated further to avoid potential problems related to updating, such as incompatibilities or new issues.

Other software used:

- Microsoft Visual Studio 2019 [125] for programming
- Blender [126] for 3D modeling
- Paint.net [127] for texturing and imaging
- GIMP [128] for texturing
- Audacity [129] for editing audio

Other resources:

- Trello [130] for planning
- Unity Collaborate [131] for version control (phased out by Unity in March 2022)
- Unity Plastic SCM [132] for version control (replaced Unity Collaborate)

The project's code is submitted as attachments, described in Appendix A.

After several months of development, the first round of user evaluations was completed in February 2022. Six evaluators participated. Feedback and suggestions from these evaluations were considered and implemented in the following four weeks before a second round of user evaluations. The evaluations are further described in Section 3.5.

3.5 Evaluation methodology

To choose an evaluation methodology, the thesis' research questions were considered. Answering them requires, among other aspects, an indication of whether the designed simulation *works* in the environment context, meaning that the application is usable, that the Room of Errors concept works in VR, and that users accept the simulated environment and can immerse themselves in it.

Initial ideas for evaluation were primarily focused on evaluating users' immersion and sense of presence, which could potentially be broken by encountering errors in the virtual environment. The evaluations would have used Witmer & Singer's *Immersive Tendencies Questionnaire* and *Presence Questionnaire*[37]. However, measuring these concepts can be complex, and some researchers have argued that current methods for measuring them have been unreliable [133]. One idea also included one group of users trying a Room of Errors version with high resolution models and another group of users trying a version with low resolution models — but this quickly proved to be infeasible, especially due to concerns regarding graphical complexity when it came to the Meta Quest HMD's capabilities. Also — while this idea may have answered interesting questions related to immersion and presence, these evaluations would not have answered questions related to the artifact's usability and its viability for educational use, which were the areas of greatest concern for SimArena.

Also considered were versions of the Technology Acceptance Model (TAM) [134] or the Unified Theory of Acceptance and Use of Technology (UTAUT) [135] questionnaires. Both of these models examine technology acceptance. It can be argued that this project indeed looks at the acceptance of a relatively new technology (VR) in an area that has not yet fully embraced the technology. The interest in the technology is already there, exemplified by the HVL's Faculty of Health and Social Sciences investing in 100 VR HMDs in 2022. However, this investment at the faculty level does not necessarily indicate acceptance or interest from teachers or students. Therefore, a study of general acceptance of VR at that level could prove interesting — a study that could likely benefit from using models such as TAM or UTAUT and their iterations.

However, the current project is more interested in the design of a specific concept in VR and its acceptance and feasibility in an educational context. Therefore, it was decided to focus more specifically on the application's *usability*, which was deemed important for Marit Vassbotten Olsen's PhD research, and the application's potential to be used in nursing studies in the future. This led to choosing the SUS [20] to evaluate the project. SUS is a popular scale, often used to assess the usability of an application or service. As Bangor, Kortum & Miller [136] state: SUS is flexible enough to assess a wide range of technologies; it is quick and easy to use; it provides a single score that is easily understood by a wide range of people who are involved in development; and it is nonproprietary, meaning it does not require payment to use. This fits the requirements for the current thesis.

SUS is a *Likert scale*, which contains ten statements. Respondents are asked to indicate the degree of agreement or disagreement with the statement on a scale from 1-5 [20], where 1 indicates "strongly disagree," and 5 indicates "strongly agree." Five questions can be considered "positive" statements where the score contribution ranges from 0 to 4, and five questions are inverse statements, where the score is calculated as 5 minus the scale position. The scores are then multiplied by 2.5 to obtain the overall value, giving a range of 0 to 100. For the current thesis, a Norwegian translation by Svanes was used (obtained from Toftøy-Andersen & Wold [137]). The word "system" was changed to the word "simulation" (Norwegian "simulering"). The questionnaire used can be seen in Appendix B.

In addition to SUS, evaluation participants completed a semistructured interview. Specific questions or discussion topics were prepared, which can be seen in Appendix C.

To evaluate the artifact's usability, two sessions of user evaluations were performed with potential users from the target group — lecturers from the three AIO nursing study programs at HVL. All sessions took place in SimArena's Operating Room training lab. Meta Quest 2 HMDs were prepared before participants arrived. When two participants cooperated simultaneously, the room was divided in half to avoid participants physically colliding while in VR. This division was accomplished by altering the Meta Guardian play areas in the HMDs, which alerts the user when they move too close to the area's defined borders.

Participants were given an oral introduction to the project and asked if they were familiar with the Room of Errors concept. If they were not, it was briefly explained to them. They were also asked how much prior experience they had with VR. Afterward, they were helped with putting on a Meta Quest 2 HMD, as some participants had issues with adjusting the straps to a comfortable position. Participants were instructed to enter the application's tutorial when everyone was ready. After completing the tutorial in approximately 2-3 minutes, they were instructed to enter the Room of Errors part of the application. Participants were observed while going through the simulation itself, and written notes were taken. Observers watched what participants saw in their HMDs by the Meta Quest *casting* functionality, which streams a video of a user's perspective to a PC browser. Participants were encouraged to "speak their mind" and continually explain what they were thinking and seeing. They were allowed to ask questions, but explanations of the developers' intentions were not given. Participants were given 20 minutes in the simulation. If participants did not progress in the simulation — for example, by spending over half the allotted time in the Operating Room's first preoperative context — they were reminded that there were more rooms to see and how to move to the next room. In some instances, participants spent too much time attempting to mark an object as an error when the developers knew that the specific object was not defined as markable in the application. In these instances, participants were told that the object could not be marked, and the object was noted to be defined as such for the next development cycle. Besides these two situations, participants were not given any further instructions, which potentially could have influenced their perception of the application's usability. However, if participants asked questions about the application's controls — such as "How did I mark things again?" or "Moving is performed with the left button, right?" — these questions were answered.

As Brooke suggests [20], evaluators were asked to fill out the scale directly after using the VR application, before any debriefing or discussion occurred. Participants were encouraged to ask questions about the scale but also to try to note their immediate responses before they could overthink the statement. After filling out the SUS, the evaluators participated in a semistructured interview as a debriefing. The interviews included the current thesis' author and Olsen, the frequently mentioned PhD candidate. In most of the interviews, the conversations naturally led to the prepared topics. Written notes were taken during the interviews to document them. For the first round of evaluations, participants were asked to come back approximately a month later for a second round. All participants confirmed their interest in coming back.

In the second round of testing, participants who had tried the application in the first round were encouraged to repeat the tutorial. These evaluation sessions followed a similar structure to the first round, although some questions were modified or added in the semistructured interviews. For example, the topic of potential benefits of completing the simulation twice was added.

Evaluation results are described in chapter 5.

Chapter 4

Application Design and Solution

To answer the research questions posed in Section 1.2, a VR Room of Errors application was developed. This chapter describes the design of the application and the user experience. This is followed by a high-level overview of the application's architecture and a summary of the work that was done to make the application run efficiently on standalone HMDs. Finally, an accompanying REST API and database which stores simulation results is briefly described.

4.1 The digital Room of Errors

The core experience of participating in a traditional, non-digital Room of Errors consists of wandering through a regular room for a given time and noting objects, errors, or inconsistencies that do not follow patient safety routines and regulations. Several questions and challenges are raised when designing a virtual version of the experience.

4.1.1 Concept

As described in section 2.5, after searching for existing literature on Room of Errors implementations, many questions are unanswered regarding several aspects of the simulation. This means that when designing a digital Room of Errors, several decisions had to be made without a clear and definitive reason behind them but instead had to be made based on the development team's own ideas and thoughts. The group of health care professionals in the team — including the previously mentioned PhD candidate and lecturers for related courses — had some previous experience in this regard, having set up and completed a non-digital version for their students a few years ago. However, this was set in a regular patient room and not in the specialized operating room or intensive care ward settings that are targeted for the current thesis. Some of the errors that were included are, however, transferable and can be used in both types of settings, such as the error of there not being hand disinfectant available in the room.

Some of these decisions were made at an early stage. As mentioned, the basic idea of the core concept is clear and was agreed upon by the development team: Participants enter a room where errors or breaches of patient safety routines are set up and try to find as many of the errors as they can. The VR experience was to follow Jeffries' framework for simulation training [45], which meant participants were to receive a briefing before going into VR, and a debriefing following the simulation. The briefing and debriefing were to occur outside of VR, with a simulation training facilitator present to guide the conversations and discussions. The simulation was to be time-limited to induce a feeling of pressure — but the actual time limit was not decided upon until later stages of development, after user evaluation sessions. Other decisions made later in development include the number of errors the simulation was to contain. Errors were continuously implemented, changed, and removed as new ideas came to the development team.

4.1.2 Detecting errors

In non-digital Room of Errors simulations, participants usually have some method of noting down objects of interest — for example, writing notes with pen and paper. The idea of *writing* in VR seems cumbersome and challenging. This would likely involve either a virtual pen or a virtual keyboard, requiring hand recognition or using the controllers to point and click on a virtual keyboard, one letter at a time. At the time of writing, these methods are inaccurate, unwieldy, or immature. Another solution would be for participants to have access to a checklist of objects or a multiple-choice form to select errors — but this could introduce bias and preconceptions in a Room of Errors simulation setting [12]. A third solution would involve taking the HMD off to take physical notes, which seems even more cumbersome. A different solution altogether is to interact with the objects of interest themselves, which takes advantage of the possibilities that are gained when translating the Room of Errors to a digital version.

Looking at literature — as mentioned in Sections 1.7 and 2.5, only one article has been found involving a VR Room of Errors by Bracq et al. [43]. Regarding noting down objects of interest, the article states: "They were asked to report any surgical error they observed by indicating the nonconforming object or situation." [43, p. 2] The article does not specify *how* participants note objects of interest or what actions they take to do so. However, the article does state that the simulator recorded data in a log for, among other things, the number and nature of detected errors. So the method of indicating objects does seem to involve somehow interacting with objects in the VR application, and not by participants taking their HMD off to take physical notes with pen and paper. Also included in the article is a picture of an error in the application which seems to be marked by the user as the object is outlined in red.

The act of marking or selecting objects in VR or virtual worlds has been a subject of study for many years. Mine [138], as early as 1995, defined the *local* and *at-a-distance* selection techniques. *Local* means users are close to the object and select them by, for example, moving their hand within an object's selection region and initiating a selection signal such as a button press. *At-a-*

distance means the object is outside of the user's immediate reach, and objects are selected by a method of pointing — for example, "laser beams" from the user's hand or by gaze direction. Both the local and at-a-distance methods are followed by users initiating a selection signal or voice input (spoken commands, e. g. "Red Box — select"). These methods have been experimented with over the years in attempts to improve accuracy and speed of selection — one example is *Progressive Refinement*, where users select a volume containing the target object and then refine their selection progressively by selecting subsets of objects via menus [139]. Today, most interactions with VR applications are performed by using remote hand controllers connected to the HMD [140], although hand gestures have been increasingly used in recent years.

The at-a-distance "laser beams" described by Mine — or *ray casting* — is one of the most commonly used selection techniques in today's commercial VR systems [141] due to its simplicity and generality [139]. The technique does have some drawbacks, primarily in selecting small or distant objects, which can be difficult for users. The action is performed in free space without physical support for the hand, and users have to keep the controller steady until a selection signal is sent, by, for example, pressing a button on the controller. The button press can cause a slight movement of the hand, causing the ray to point at a different object when the button press registers in the application [141]. This is known as the "Heisenberg effect" [142].

Since the average user of the Room of Errors VR application is expected to be a novice VR user, simplicity is prioritized. Therefore, the solution used in the current thesis' VR Room of Errors application is the raycast selection technique: users essentially point and click on objects in the environment using the HMD's hand controllers. The right-hand controller emits a raycast — a ray starting at its base, extending essentially to infinity. Most importantly, this allows for identifying the object the ray hits in the game engine. By default, the ray itself is displayed to the user in a red color, indicating that it is not currently pointing to an interactable object. Figure 4.1 shows the red ray which does not hit an interactable object. When it does hit an interactable object, the ray itself changes to a white color, and the point of intersection — the point where the ray hits the object — displays a sphere, so the user can more accurately identify where the ray is pointing, and at what object they are pointing. Figure 4.2 shows the raycast hitting an interactable character. The ray does not, in fact, hit the *object itself* but a *collider* Component that is attached to the object. When an object is pointed at, and a button is pressed on the hand controller, the object changes color to indicate that it has been marked as an error or risk factor. Pointing at a marked object and pressing a button on the hand controller again causes the object to change back to its original colors to indicate that it is no longer marked. A marked object is shown in Figure 4.6.

This type of selection as a substitute for users taking notes with pen and paper is not a perfect solution. For example, it does not directly allow for errors where the error involves an object that *should* be present but is missing — nor does it directly allow for errors or risk factors related to less physically tangible concepts, such as loud noises or needlessly stressful environments. The solution also emphasizes an issue that frequently arose when designing the digital Room of Errors — visibility of the errors. Making errors highly visible, identifiable,

Figure 4.1: The ray does not hit an interactable object.

Figure 4.2: The ray hits an interactable object. A sphere shows the point of intersection.

and easy to point at and click on makes for a better user experience — but potentially removes the challenge of finding the errors, which is a large part of the experience itself.

A different challenge also presents itself. In order for an object to be interactable and able to be marked, the object must be defined as such. If the user is only able to interact with and mark objects that are related to errors, the experience can be "cheated" and reduced to simply "clicking on" all the objects that *can* be clicked on. Therefore, other objects in the scene should also be defined as markable. However, extending this to absolutely all objects in the scene does have consequences or lead to new decisions that have to be made.

One example of these decisions can be found in the operating room's anesthesia machine. In reality, this is a highly advanced machine, consisting of hundreds of different parts, each with potentially severe consequences if something were wrong with it. Ideally, users would be able to select and mark one or several of these smaller parts as potential errors — but that would require breaking the mesh into several different, smaller meshes, which has severe consequences for the application's rendering complexity and performance. These performance issues are discussed further in section 4.6. In cases such as this, it was decided to treat the machine as a single object. Users essentially stating that "there is an error related to the anesthesia machine" is considered sufficient, instead of requiring specific details. The goal of users marking objects is to replace the traditional note-taking, to be used as a basis for further discussion after the user is finished — not to create a detailed report.

In other instances, two colliders are in close proximity to each other — for example, when an interactable object lies on top of another interactable object. As previously mentioned, this is a known potential difficulty with raycast selection. In some instances, this is alleviated somewhat by making the relevant colliders larger than necessary. Making colliders slightly larger than the object itself also has the benefit of the user more easily seeing a difference when moving the ray between two objects, which makes it more evident at which object the user is currently pointing. However, this can cause the ray to seemingly stop "in the air," potentially reducing user immersion. A more ideal solution would probably include a way to more clearly communicate to the user which object is being pointed at without adding rendering complexity beyond the Quest's capabilities. In some other instances, instead of its collider, the object itself is made larger than the real-world object it represents. This technique also has advantages regarding user comfortability, as described in Section 4.1.4.

During development, an attempt was made to implement simple "tooltips" for interactable objects. When the user pointed at an interactable object, a text box appeared, showing the user the name of the currently being-pointed-to object. According to informal user testing, this functionality significantly aided usability, without affecting frame rate. However, the PhD candidate and lecturers in the development team considered this to lessen the experience and lower the challenge of a Room of Errors, as the explicit identification of objects was seen as too helpful. Therefore, the tooltip feature was disabled. In a later iteration, a different attempt was made to implement simple highlighting of objects to help identify which object was currently being pointed at by coloring the entire object a single color. This is the same technique that is used to identify objects that have been marked as errors, but with different colors. While this provided clarity for some informal testers, other informal testers with less experience in VR were confused. Using the same coloring technique for two different purposes made them unsure if they had unintentionally marked the object as an error by simply pointing at it. In addition, the design team again believed this lessened the experience as it "felt very artificial." Therefore, the functionality of highlighting objects when hovering over them was also disabled. Users only know they are pointing at *an* interactable object but are not always sure *which* object they are pointing at until they mark it. However, the decision to mark an object is not permanent, so if users mark an object in error, they can choose to un-mark it.

Several different ways of marking objects were experimented with and examined. The simple method chosen for the related bachelor project involves storing all Unity Materials that are applied to an object's mesh and changing them all to a single color Material when the object is marked. When the object is unmarked, the original Materials are put back in place. This method has very few computational consequences — in fact, coloring an object with a single color can reduce its rendering complexity as the entire mesh can be rendered through a single set of shaders. However, the result reduces the object's graphical fidelity, giving it an unrealistic appearance.

Two different Unity Assets from the Unity Asset Store were examined. Quick Outline [143] adds a simple frame or outline to the mesh's outer edges, which has a satisfying look and effect. Figure 4.4 shows Quick Outline in use. The effect is very similar to the outline effect shown in Bracq et al.'s article [43], mentioned in Section 4.1.1. However, even if the asset is relatively simple, highlighting more than a few objects proved too complex for the Quest HMD, as the frame rate started dropping after marking more than five objects. This was unacceptable when the user should be able to mark as many objects as possible.

The Highlight Plus [144] asset is a more advanced implementation, allowing for large amounts of customization to the outline effect. Figure 4.5 shows Highlight

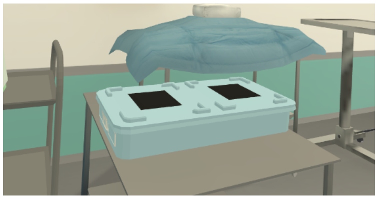

Figure 4.3: A box containing surgical instruments - unmarked

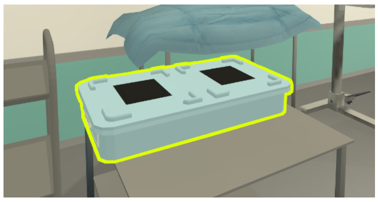

Figure 4.4: The box marked using the Quick Outline asset

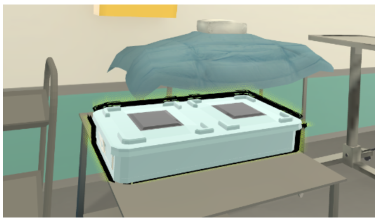

Figure 4.5: The box marked using the Highlight Plus asset

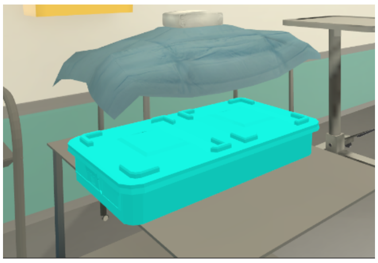

Figure 4.6: The box marked by changing its materials

Plus in use. The outlines are achieved through Post-processing — effects that can improve an application's appearance, such as bloom and motion blur [145]. Post-processing, in general, is not recommended when developing for a Quest 1 HMD [146] — Simply enabling post-processing in Unity before compilation caused the application to go from a steady 72 frames per second to around 40 frames per second on the Quest, even without actually adding any effects. Again, this is unacceptable.

Finally, some experiments were performed with changing values in the Materials already in use — for example, making an object brighter but still mostly maintaining its original colors. However, an effort has been made to reuse Materials for efficiency purposes. For example, if different objects use the same shade of blue, they can use the same identical Material for the blue areas. This allows the two different objects to be rendered in a single draw call, reducing the number of operations. In other words, making objects brighter would require either brightening other, unrelated objects that happen to be using the same Material or using entirely unique Materials for each object — which is inefficient and would have consequences for the Quest's frame rate.

As a result of these experiments, it was decided to go back to the simple method of changing Materials to a single color to indicate an object being marked. This effect is shown in Figure 4.6. A better method for marking errors is a definite area of potential improvement in future development.

4.1.3 The User Experience

When starting the application, users first encounter a simple menu with two choices: Go directly to the Room of Errors, or go through a short tutorial.

The tutorial explains to the user the actions that are available to them in the primary simulation experience and requires them to perform these actions. The first action, interacting with menu buttons, is taught by simply requiring users to click a button saying "OK." They are then moved to a different room, where they are taught about the action of marking objects. Before moving on, they are required to mark three cubes of a specific color. Cubes with other colors in the room can also be marked but do not count towards the three specific required cubes. The concept of marking is then put into practice in the next room, where users are required to mark three errors or potential sources for adverse events.

For the sake of the tutorial, the three errors are exaggerated and highlighted. Following advice from Schell [90], establishing some degree of absurdity or introducing comedy may make it easier for users to reinforce the rules of the virtual world, thus reinforcing the user's sense of presence. The three tutorial errors include a surgeon standing on top of a patient with an open surgical incision, a trashcan on top of a patient with an open incision, and an absurdly oversized pair of scissors threatening to cut an intubated patient's tubes, connecting them to their respirator.

The final tutorial room explains movement. Users are required to move into three Teleportation Anchors placed on the ground, which are fixed circles that light up when users point at them. After moving into the three anchors, the concept of free teleportation is explained to users, and they can try this method of movement freely before choosing to move on from the room. The teleportation system is further discussed in Section 4.2. Lastly, users are presented with the choice of going through the whole tutorial again from the start or going back to the main menu.

When choosing to enter the Room of Errors in the main menu, users are transferred to the main simulation experience, which contains two "*courses*" — one course for OR nursing and nurse anesthetist students and one for intensive care nursing students. Users are first asked to specify their specialization: Nurse anesthetist, OR nursing, or intensive care nursing. After making their selection, they are shown how many other users are currently connected and how many of them — if any — have also chosen their specialization, signifying that they are ready to start. If all users are ready, a large "Start" button appears, which starts the simulation and transports all users to the start of their relevant course.

The two courses require users to go through different combinations of the three available rooms: The operating room, the transfer room where patients are handed over from the operating room team to the Postoperative team, and finally, the Postoperative/Intensive Care ward. Figure 4.7 shows the operating room, and Figure 4.8 shows the postoperative ward. The operating room contains four "contexts": *Preoperative*, set before the patient is put under anesthesia; *Timeout*, set immediately before the operation starts; *Perioperative*, set during the operation; and finally, *Postoperative*, set after the operation is finished.

Users can move around in the rooms by physically walking or using the teleportation functionality described in Section 4.2. When discovering an object users believe to be related to an error, they can choose to mark it as such by

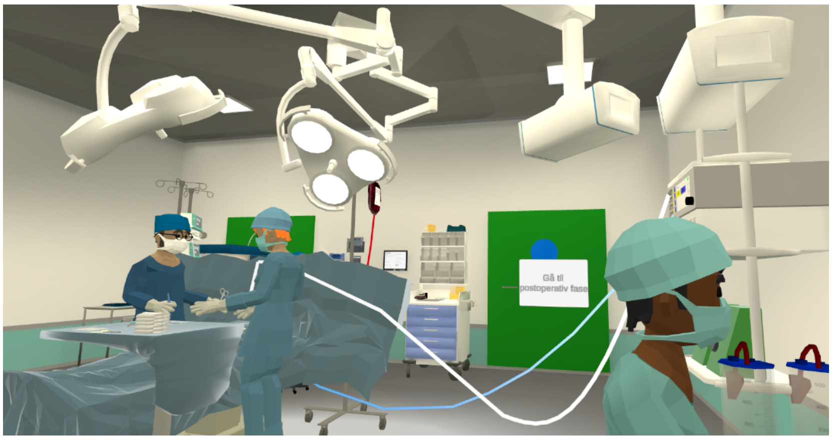

Figure 4.7: The operating room

Figure 4.8: The postoperative ward

Figure 4.9: Button to start a dialogue

Figure 4.10: Dialogue between the nurse anesthetist and patient

pointing with their right hand controller and pressing a button, as described in section 4.1.2. The OR contains 21 defined errors for participants to discover, and the postoperative ward contains 20 defined errors. In addition, most objects in the rooms can be marked even if they are not related to a predefined error. The rooms contain, in total, approximately 300 objects that can be marked. Two contexts contain a “speech bubble” conversation between the characters — the preoperative and the “timeout” contexts. The dialogue contains errors related to patient safety routines. The conversations are initiated by the user pressing a button placed in front of the character, which activates a “speech bubble” with a line pointing to the character speaking the dialogue and a “next”-button for moving to the following dialogue. The implementation uses the Unity asset VIDE Dialogues [147] to control the flow of conversation and can be extended in future development to include different user responses and conversation branches. Figure 4.9 shows the button that starts a conversation, and Figure 4.10 shows a conversation in progress.

Moving to different contexts in the rooms is performed by using buttons that have been placed on the doors, by pointing at them with the right hand controller and pressing the trigger button. The buttons were deliberately placed on doors in an attempt to invoke natural thoughts of looking for the door when a user wants to exit the room. These buttons are shown in Figure 4.11. However, preliminary testing revealed that many users stayed in the first room they entered and did not go through the different contexts. They had to be reminded that more than one context existed and how to move between them. Therefore, in the next iteration, the same type of buttons was placed in the tutorial to move users to the next tutorial area — but most users had forgotten the functionality when they entered the main simulation. Therefore, a pulsating effect was added to the buttons, causing them to grow and shrink in size periodically, to catch users’ attention. This reduced the issue to a small degree, but several users still had to be reminded that the button was there during user evaluation sessions. If there had been enough time in the project for a third round of evaluations, the pulsating effect would have been experimented with more by, for example, changing the button’s colors or making the button grow even more prominent.

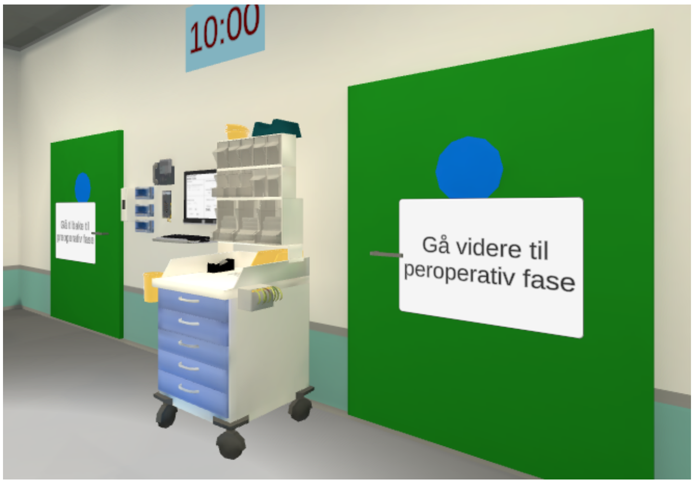

Figure 4.11: Buttons to change contexts have been placed on doors.

When users reach the last room in their course, the context button is replaced by a button that ends the simulation session. When pressing this button, users are asked to confirm their choice of ending the session. By doing so, they are teleported to a new final room. Alternatively, the same ending occurs when users run out of time. Users are notified by a speaking voice when ten minutes, five minutes, one minute, and ten seconds remain in their session. In the final room, users are shown a list of all the objects they marked during the session. In addition, a small screen on their left informs them that their choices have been submitted to a database. On their right, they can find buttons to restart the simulation, go back to the main menu, or close the VR application. An overview of the user sequence can be seen as a flowchart in Figure 4.12.

The simulation contains limited amounts of sound. Sound could potentially have been used to increase user immersion and sense of presence, as suggested by, for example, Servotte et al. [36]. However, the application is intended to be used by two participants in the same room. If a sound effect originating from a specific place or object in the room were to be played on one user's HMD, the other user present in the room would also hear that sound. The other user may be positioned at an entirely different location in the virtual room, facing an entirely different direction than the user that produced the sound effect. This experience could be disorienting for the second user and instead break their immersion. The application could potentially have benefited from sound effects that do not necessarily originate from a specific object, such as background noises from air vents or similar sounds that generally fill the room. The one set of sound effects that have been implemented consists of a generic alert noise, followed by a neutral, female voice, notifying the user that they have ten minutes, five minutes, and one minute remaining. The voice is generated by the free text-to-speech service ttsMP3.com [148]. When ten seconds remain in the simulation, the alert noise is instead followed by the sound of a clock ticking down, which concludes with a "ding" sound when the timer reaches zero. The alert noise, clock ticking, and "ding" sounds have all been acquired from the website Freesound [149, 150, 151]. These sounds all originate from the position

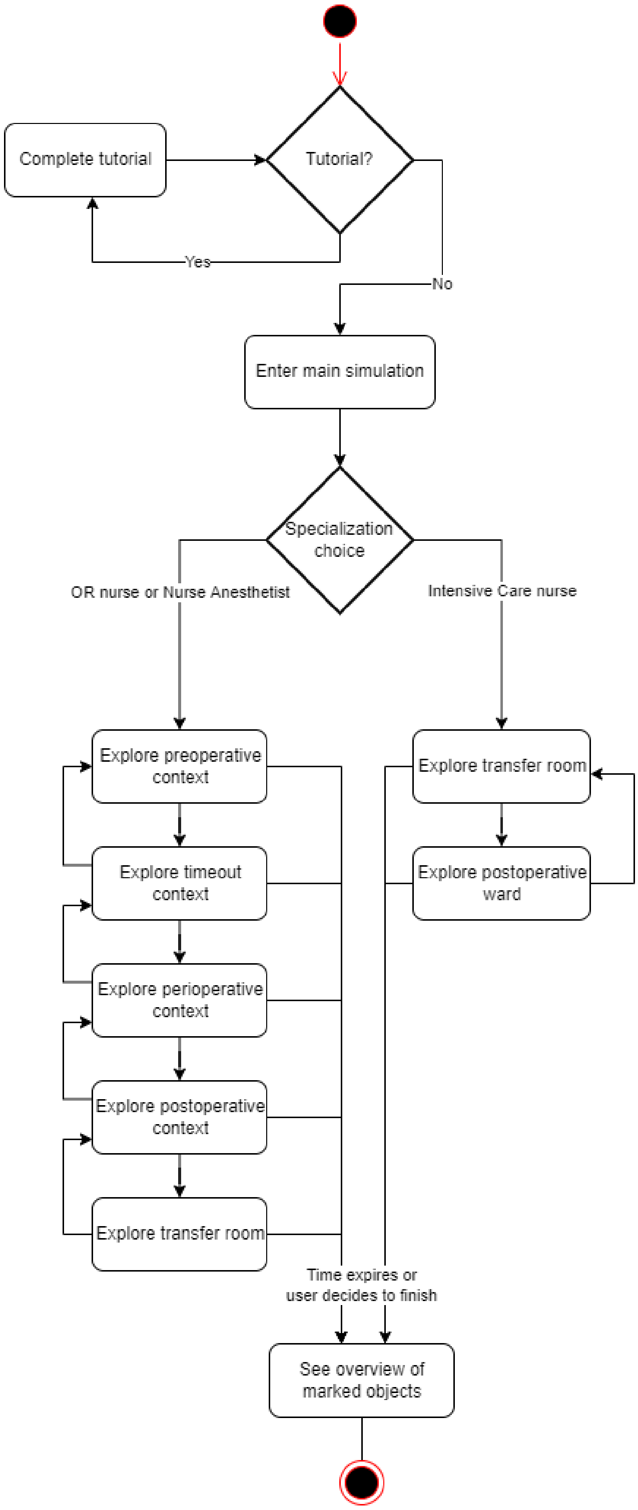

Figure 4.12: Flowchart of the user experience

of the user's HMD and play at the same time for all connected users, which should avoid the potential disorienting effect.

To create an authentic visual experience, users see and observe several 3D models in the environment. Most of these models were part of purchased packs of assets and models. The purchased packs can be seen in Appendix E. Several models also had to be created by the author using Blender [126]. These models are shown in Appendix F. Some were created by Adrian René Johnsen Mortensen for the BSc prototype, and these models have been noted as such in the listing. Some additional models have been created by modifying purchased models, but these are not listed.

4.1.4 The Errors

As mentioned in Section 4.1.3, the operating room contains 21 intended errors, and the postoperative ward contains 20 intended errors, for a total of 41 errors. Three errors are used more than once, either in the same room or in both main rooms (e.g., the error *"Hand disinfectant is empty"* is used twice in the OR and three times in the postoperative ward). Therefore, the application contains 34 *unique* intended errors. The intended errors have been chosen in different ways. Several have been chosen and defined during discussions in the design team. "External" health care personnel from Haukeland University Hospital have been involved in defining some errors. For example, an anesthesiologist was consulted regarding what medications are commonly used in specific types of operations and what medications could be used as an error related to allergies. In addition, some errors were suggested by nurses working at Haukeland, inspired by real events. Other errors were suggested by participants in the user evaluations after having tried the application. A complete list of all intended errors can be seen in Appendix D

The Room of Errors concept provides an additional challenge in VR: Visibility of the errors. Oculus explains [152] that VR creates a situation where 3D images are presented on a screen that remains the same distance from the eyes, but the different images each eye sees may require each eye to adjust differently. Therefore, objects that users will focus on should be rendered at least 0.5 meters away, and a distance of 1 meter is reported to be comfortable for menus and user interfaces. The recommendation of 0.5 meters creates a challenge for the Room of Errors simulation, as the concept may lead users to focus on and stare at specific objects up close to look for errors. The errors should be recognizable as errors from a distance — but at the same time, they cannot be too apparent to users, so the challenge of the simulation remains.

Because of the recommendation of 0.5 meters, in the current Room of Errors application, some small objects are somewhat larger than the real version they represent. In some situations, this also has advantages regarding the user's ability to mark the object, as described in Section 4.1.2. In other instances, attempts have been made to solve the challenge by allowing users to discover the related error by studying different objects.

One example is the 3D model of a stopcock with connection tubing (Norwegian "treveiskran"). The system is often used to control the flow of transfusions to the patient from one or two connected tubes. While small — the connection tubing itself is 10 cm long with a diameter of 3-5 mm — it can be the source of several different errors. For example, the wrong connection can be open, which prevents the intended medication from flowing to the patient — or a connection is open with no tube connected, allowing air to enter the patient's blood flow. If the model were scaled to the same size as its real-life version, users would likely feel the need to study the system closely for errors. Instead, the model is made slightly larger. It is used in two intended errors in the postoperative ward: One patient is receiving a mixture of two medications that should not be combined, and for another patient, the stopcock is correctly closed, but is missing a cap, which may lead to an infection. The first error can be discovered by users seeing the two connected tubes from syringe pumps which are marked with visible text. The second error is made more apparent by the missing cap (which has also been enlarged compared to its real-life counterpart) lying in the patient's bed. Users may still wish to study the systems up close, but they are not required to focus on them for long to recognize an error. Instead, the Room of Errors simulation challenge lies in them remembering to check the stopcocks at all. Braun's DiscoFix® stopcock is shown in Figure 4.13, and the 3D model is shown in Figure 4.14. The error with the missing cap is shown in Figure 4.15

An argument can also be made for making some errors less challenging to find. As mentioned in Section 2.4, making mistakes during simulation training can have long-lasting effects on participants. As also mentioned in the same section, completing the same simulation twice is common to encourage feeling a sense of mastery. In a similar vein, during the first user evaluation sessions, one participant suggested making an error easy to find. This could potentially make participants feel a sense of accomplishment and inspire them to look for more. Therefore, an attempt at an "obvious" error was added: The nurse anesthetist in both rooms is wearing an oversized necklace. Wearing jewelry is a breach of hygiene routines and is one of the first things a nurse is taught not to do. The second round of user evaluations revealed that the error was not "obvious enough" as not all participants discovered it — but all participants found at least one error.

The BSc prototype included a menu to select which specific errors should be active for the session and functionality to activate a selected number of random errors. These selections were made in VR and involved a cumbersome sequence of events. A simulation participant entered VR to complete a tutorial before giving the HMD to a simulation facilitator, who selected which errors to activate before giving the HMD back to the participant. Activating errors from a desktop or web application was a suggested improvement for the Master's project. However, after discussions within the design team, this feature was deemed unnecessary. Instead, in the current application, errors are activated based on the specialization chosen by the users.

A related discussion in the design team was regarding the need for active and inactive versions of the errors. The initial intention was for errors to appear differently to users if they are active or inactive. For example, for the error "Catheter bag is on the floor": If the error is active, the bag is placed on the floor in the OR — If the error is inactive, the bag can be seen hanging from a proper position on the operating table. The two versions can be seen in Figures 4.16 and 4.17. Late in development, this functionality was decided to

Figure 4.13: A Braun DiscoFix® [153]

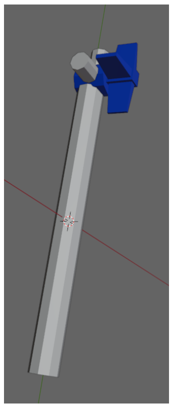

Figure 4.14: 3D model of a stopcock in Blender.

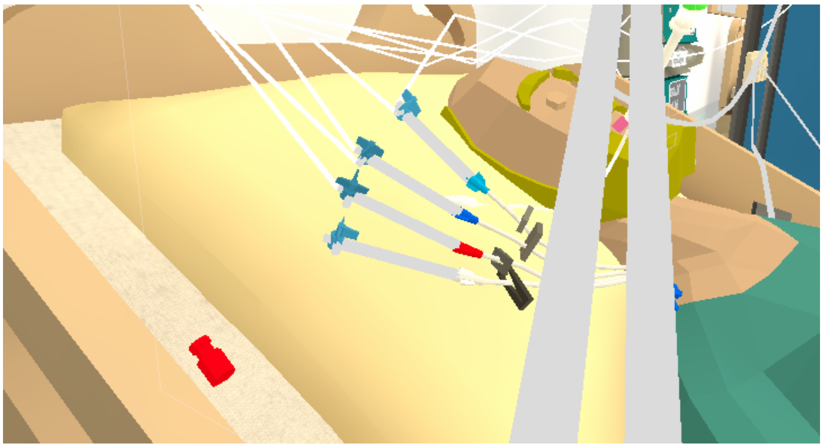

Figure 4.15: Example of stopcocks in the simulation

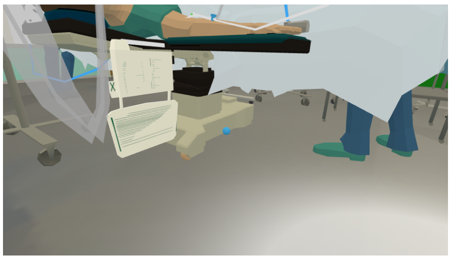

Figure 4.16: Error inactive - catheter bag hanging from a proper position

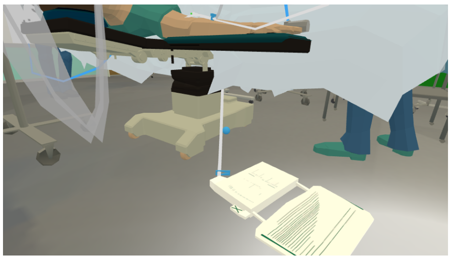

Figure 4.17: Error active - catheter bag lying on the floor

be unnecessary. An argument could be made for disabling errors targeted at OR nurses when nurse anesthetists were training and vice versa. However, the design team could not see a downside to letting all errors be visually active and present, even if they were not counted as an active intended error when determining the number of errors found. In addition, only needing to implement one version of an error allowed for faster implementation of *new* errors. Therefore, errors that were implemented in later stages of development do not offer an inactive version. However, from a code perspective, all errors are logically considered active or inactive, based on the user's specialization.

This decision was not without consequences. The second set of user evaluations revealed the benefits of participants completing the simulation twice. With this in mind, one participant suggested activating some selected errors for the first simulation completion and a different selection of errors for the second simulation. This possibility should be considered in future development. Therefore, the errors implemented near the end of development that do not offer an inactive version may be considered an instance of technical debt. It was decided to prioritize implementing new errors at the cost of not offering inactive versions of them.

4.2 User input and interface

It was decided at an early stage to keep using the Meta Quest's hand controller for user input, as in the BSc prototype. Hand tracking could also have been an option, as this functionality has seen increased use in recent years [140], and Meta's implementation is continually being improved [154]. However, at the time, this implementation required using the Oculus Unity Integration asset in Unity. Using this asset would have consequences for compatibility with other HMDs. This cross-compatibility would later prove necessary, as discussed in Section 4.5. However, hand tracking may be worth exploring in future versions of the application.

A Meta Quest HMD's hand controller offers five buttons that are usable by applications:

- a primary button, a secondary button, and a thumbstick which can be pressed down to perform a button press, are pressed with the thumb
- a trigger button which is pressed by using the index finger

Figure 4.18: The Meta Touch hand controllers. (Photo: UKER, CC BY-SA 4.0 [155])

- a “grip” button which is pressed using the middle finger

A sixth button is usually reserved for Meta or menu functionalities. The controllers can be seen in Figure 4.18. The Meta Quest 2 HMD’s hand controllers are visually slightly different but offer the same buttons.

The BSc prototype contained several button functionalities. For example, the ray to mark objects could be toggled on and off by pressing the right hand controller’s secondary button. Another example is that users were teleported to a different environment that contained a “pause” menu when they pressed the left hand controller’s menu button. Perhaps the most advanced feature was the controls for teleportation. When users moved the left thumbstick and held it pushed in a direction, a ray appeared from the user’s left hand. At the end of the ray was a circle with an arrow pointing in the direction the thumbstick was moved. Users could rotate the thumbstick to define which direction they would look after the teleportation. To perform the teleport, users had to click the thumbstick while holding a direction. In addition, users could rotate their perspective by moving the right thumbstick. This movement felt natural for users familiar with hand controllers and 3D games, where it is normal for the left thumbstick to control movement and the right thumbstick to control camera perspective. However, the movement functionality was revealed to be far too complex and confusing for novice users after initial informal user testing at the start of the Master’s thesis project. Users struggled with performing the action and remembering how to use the thumbsticks. These user tests indicated the need for reduced complexity.

In the current version, unnecessary actions have been removed. In preliminary testing, several users instinctively held the hand controllers so that their index fingers were on the controllers' grip buttons instead of their middle fingers. In addition, some users instinctively pressed the primary button to perform actions, while others instinctively pressed the trigger button to perform actions. Therefore, instead of a specific button performing a specific action, all buttons on the left hand controller perform the same action, and all buttons on the right hand controller perform the same action — all buttons meaning the primary, secondary, trigger, and grip buttons. Users can perform three distinct actions in the application: Teleportation, marking objects, and interacting with user interface elements such as buttons or checkboxes.

The application contains rooms that are larger than the rooms the simulation participants will likely be using VR in — or in the case of users cooperating in the same room, the physical room would likely be divided in two to avoid the users physically crashing into each other. Therefore, the application requires some method for *artificial locomotion*, which allows users to move through the virtual rooms. According to Oculus, a comfortable and efficient method for locomotion is "essential to the success of a VR project" [156]. Several different locomotion techniques have emerged [157], but a common and easy-to-implement method is teleportation. This method avoids the potential challenges of *vection* — the perception of motion based on visual input — which can lead to discomfort [158]. It is also easy for users to grasp without having to consider preferences or previous experience with VR. Teleportation is implemented through Unity's XR Interaction Toolkit [159].

Two modes of teleportation are available in the application. The default mode is teleportation to pre-defined locations known as *Teleport Anchors*. These anchors are represented by several blue circles that have been placed on the ground. The user's left hand continually casts a ray, and when this ray points to one of the circles, the circle changes to a green color. If the user presses any button on their left hand controller while pointing at a circle, they are teleported to the circle. Figure 4.19 shows the Anchor Teleportation. The green circle is currently targeted, and the user will move to it if a button is pressed. The blue circles are currently un-targeted anchors. The second method for teleportation has been called "Free Teleport" and is intended for users familiar with hand controllers and VR. This method can be toggled on by users while selecting a nursing specialization directly before starting the main Room of Errors simulation. If active, the user's left hand does not continually cast a ray, but an arced ray is activated when the user moves and holds the left thumbstick forward. A green circle at the end of the ray where it hits the ground indicates the location being teleported to, which can be moved freely by moving the hand controller. When the user releases the thumbstick to its neutral position, they are teleported to that location. Free teleport is shown in Figure 4.20.

For both teleportation methods, when performing the teleport action, the screen instantly turns black before fading back in over the course of a second. This fade or "blink" technique has been reported to improve comfort during teleportation [160] by, in a sense, forcing the user to blink their eyes. Also, the base principle of movement with the left hand and changing perspective with the right hand are maintained for both teleportation methods. Moving the right thumbstick

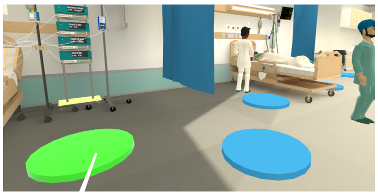

Figure 4.19: Anchor Teleportation: The green circle is currently targeted.

Figure 4.20: "Free teleportation": A user-controlled ray and circle determines where the user will teleport to.

left or right rotates the user's perspective by 45 degrees in that direction — called a "snap turn." The principle also follows best practices for mapping input functionality, as suggested by Oculus [161]. Snap turning is not a required user action, as users can instead physically rotate their bodies to achieve the same result. However, the functionality was kept from the BSc prototype in order to avoid alienating users familiar with VR and hand controllers. Moving the thumbstick has not been observed as an instinctive or unintended button press by novice users, so it is not considered a potentially disruptive or invasive functionality. In addition to being recommended by Oculus [160], teleportation, snap turns, and screen fading are recommended by Unity to make VR applications accessible and comfortable [162]. Lastly, IEEE recommends using teleportation and screen fading to reduce VR sickness [163].

The second action required by users is marking objects as errors, which is explained in Section 4.1.2. The user's right hand continually emits a ray. As with teleportation, if a user presses any button on the right hand controller while pointing at a markable object, that object is marked.

The third action, interacting with the user interface, uses the same controller technique as marking objects. The same ray being emitted from the user's right hand is used to point at user interface elements, and if a user presses any button on the right hand controller while pointing at the element, it is activated. For example, the user interface button is pressed, or the checkbox is ticked.

The user interface follows the recommendations of Oculus regarding vision [152]. Traditional desktop 3D applications often contain a "Heads-up Display" (HUD) which displays information to the user. It is usually always shown in a fixed position on the screen. This can lead to discomfort in VR. Instead, information should be "built into" the scene. For example, in the Room of Errors application, users can see their remaining time on digital clocks hanging on the wall. In a traditional desktop application, this would likely be shown on a HUD. In addition, menu buttons to move to a different room or context are placed on doors, where it would be natural to look for a way forward.

4.3 Multiplayer

Allowing for multiple concurrent users — or "multiplayer" functionality — is implemented with the Unity plugin Photon Unity Networking (PUN) 2 [164].

Photon provides network connectivity and communication through their cloud services and servers. This includes a free plan that supports up to 20 concurrent users. Paid plans are available if more users are required. At least initially, the Room of Errors application is intended to be used in a PhD study, where small groups of participants will be under observation. If a later decision is made to use the application as part of a simulation training concept in education, the AIO nursing simulation training sessions at SimArena usually include 5-10 participants. Therefore, a limit of 20 concurrent users is considered sufficient.

PUN connectivity is first separated into *rooms* and *lobbies*. Lobbies allow users to browse open and available rooms and select which of them to join. If implemented, there is also a possibility for "matchmaking" between different players based on specified filters. Rooms, where multiplayer interactions actually happen, can be customized based on properties such as a specified maximum number of players, room names, and other properties defined by developers. For the Room of Errors application, it was decided not to implement a lobby functionality. Instead, when users enter the simulation part of the application, they join a "random" available room that is open. If no open room is found, a new room is created for them. When all players in a room have chosen their AIO nursing specialization and started their simulation, the room is closed for new users, and any new users must create a new room. Again, at least initially, the application is intended to be used with small groups of participants under observation. Therefore, it was not considered necessary to implement functionalities allowing users to select which session they should connect to, but it is possible for multiple sessions to run in parallel. If needed at a later time, lobby functionality is an extension that should be relatively simple to implement.

The implementation of PUN connectivity is based on an implementation described in YouTube tutorials by Valem [165]. Most importantly, it is managed through the `NetworkManager` class. When loading the Room of Errors Unity Scene, a connection is established, and a PUN room is entered. The `NetworkManager` is subscribed to events in several different classes. One example of these events is a user selecting an AIO nursing specialization. This event leads to updating the local user's `CustomProperties` hash table, which by itself notifies other connected users. These other users can then handle the event if needed. Another method for notifying other connected users is through custom Photon events. An example of this method can be found in the `DetectableManager`. Marking an object has no direct effect on other users by itself, but if the user is connected to the network, an event is raised when the local user chooses to mark an object as an error. This tells other connected users which object was marked based on a unique persistent identifier. Their local `DetectableManager` looks up the object in their local C# Dictionary and marks the object locally for the other connected users as well.

Other connected users are represented by anonymous health personnel avatars. The avatar can be seen in Figure 4.21. The avatar's head and body movements are synchronized to the user's HMD movement, and the hands are synchronized to the user's hand controller movement. This synchronization is performed through Photon's `Photon View` and `Photon Transform View` Components, which continually transmit position and rotation transform values.

Near the end of the current project, Photon stopped development of PUN 2

Figure 4.21: Avatar representing other connected users

in favor of their new Photon Fusion [166] system. PUN 2 will be supported in Unity 2022 versions, and according to their documentation, projects using PUN 2 will continue to work and run in the future. Future development of the virtual should likely consider changing to Photon Fusion.

4.4 Architecture

As described in Section 2.9.2, development in Unity uses an Entity-Component-based architecture. The most important Entity in the Room of Errors application is the *Game Manager*, which contains the identically named `GameManager` Component. The Game Manager connects events to listeners and manages the flow and user progress of the simulation. The class contains references to all other manager components in the application — such as the `DetectableManager`, which manages all markable objects, and the `NetworkManager`, which, as the name implies, manages the network functionality. These "sub-managers" act independently and perform their task as directed by the Game Manager. Any communication between them is performed through C# events.

As mentioned in Section 2.9.2, decoupling and reuse are important motivations for using this architectural style. These principles are further strengthened by the use of C# events in the application. One example of this can be found in the Component `RaycastMarking`, which is responsible for reacting when users want to mark an object as an error, as described in Section 4.1.2. When a user points their raycast at an object and presses a controller button, the Component

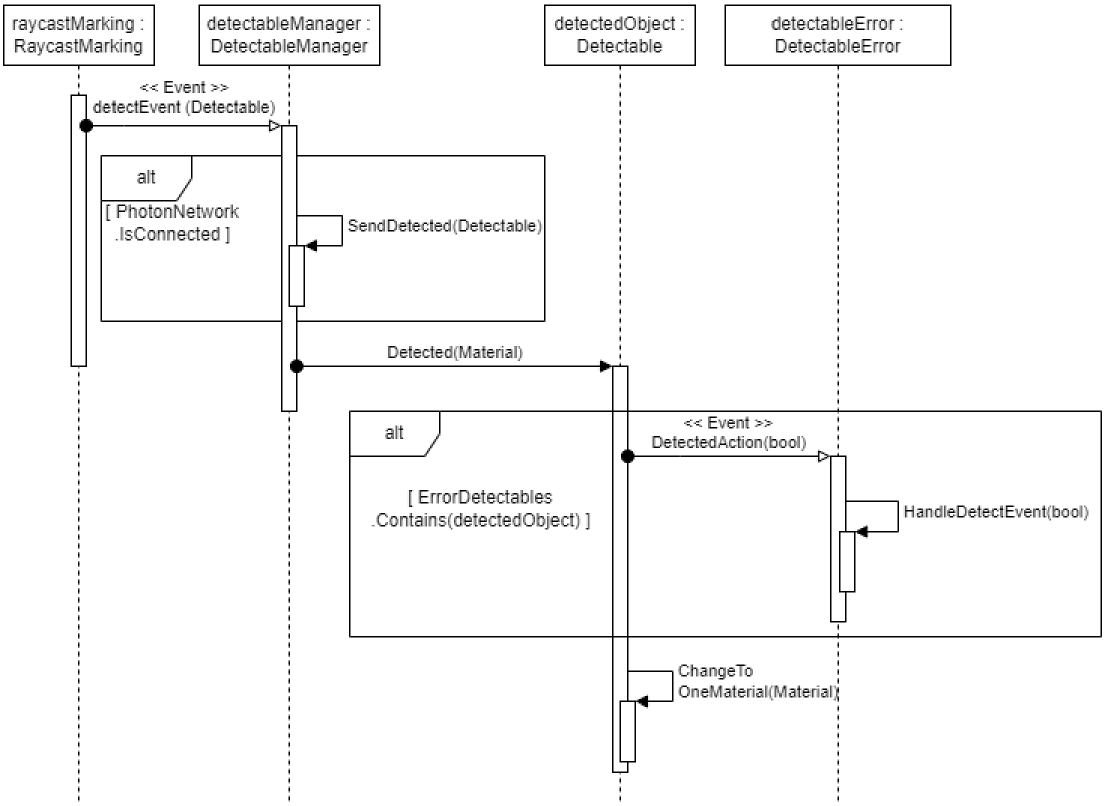

Figure 4.22: Sequence diagram of an object being marked

checks if the targeted object contains a `Detectable` Component. If so, the `DetectEvent` is invoked, with the `Detectable` object passed as an argument. The `RaycastMarking` Component has no information about or control over what happens next, nor does it need to — it simply lets any listeners know that the user has tried to mark an object as an error. The `GameManager` Component has connected the `DetectableManager` as a listener to this event. When the event is invoked, the `DetectableManager` tells the `Detectable` object received as an argument that it is now considered marked and that it should change its appearance. If the `Detectable` object is related to a `DetectableError`, the error is notified that it has been discovered through an event invoked by the `Detectable`. The sequence is shown in Figure 4.22.

The independence gained by using events also means that the `RaycastMarking` class can notify any other Component in the application without any alterations or additions to the Component's code. For example, a statistics Component that registers *when* the object was marked could have merit. On the other hand, the `DetectableManager` can be connected as a listener to any other Component that can send a notification that an object was marked. For example, the VR implementation could be changed to a mouse-and-keyboard implementation where a mouseclick invokes a similar event.

Increased use of events is also a change from the BSc prototype described in

Section 2.1. In that version, Entities related to a defined error did not have a `Detectable` Component themselves but were all defined as hierarchical children to an Entity containing a `DetectableError` Component. In the BSc prototype, this meant it was easy to determine if an error had been discovered and that all related objects should be colored as marked. However, this also meant that, for example, objects' transform values — where they were positioned in the world — were dependent on their parents' transform values. If different Entities were related to the same error, they were, in a sense, tightly coupled. This led to a rigid structure where modifying an error could create issues. In the current version, however, all `Detectable` Components are independent and instead have their own `DetectedAction` event that can be listened to by any related `DetectableError`. This means that objects related to an error can be moved, changed, or replaced without problems, and any other objects or Components can listen and react when they are marked.

Objects that can be marked — or "detectable" objects — and errors in the scene are overseen by the `DetectableManager` Component. When the application starts, this class asks the Unity engine for all GameObjects with a `Detectable` Component and all GameObjects with a `DetectableErrors` Component. These GameObjects are then stored in C# dictionary data structures. This lookup method is not efficient and could potentially lead to performance issues if used often — Unity's documentation states that the method `FindObjectsOfTypeAll(Type type)` is "very slow" [167]. However, it is only done once, and if needed in the future, any slowdown or delay at startup can be "hidden" from users by showing them the same black screens that are used when they teleport. Dynamically re-discovering all detectable objects at startup makes it easy to change or add new detectable objects and new errors. However, it does have consequences for the accompanying database, which is summarized in Section 4.7. `Detectable` objects are dynamically given a unique ID which is based on the object's name and position coordinates, which means that the object's ID changes if the object is moved. This is an instance of technical debt. A functional and easy-to-implement method was chosen over a solution that would have been more scalable and with fewer consequences — but this method would have been more time-consuming to implement. A more ideal solution would give objects a more persistent ID independent from values that can be changed. The database stores `Detectable`s and errors as they were when they were first created and will, for example, create a new table row if a detectable is altered. Development of the API has not been prioritized during the project period — but it serves its purpose.

An overview of the most important classes is shown in Figure 4.23. A significant amount of classes are not included, such as classes that define the application's menus, classes that define countdown clocks' behavior, or classes that define "speech bubble" dialogues.

4.5 VR Frameworks

As mentioned in Section 2.1, several elements and frameworks used in the BSc prototype were considered outdated.

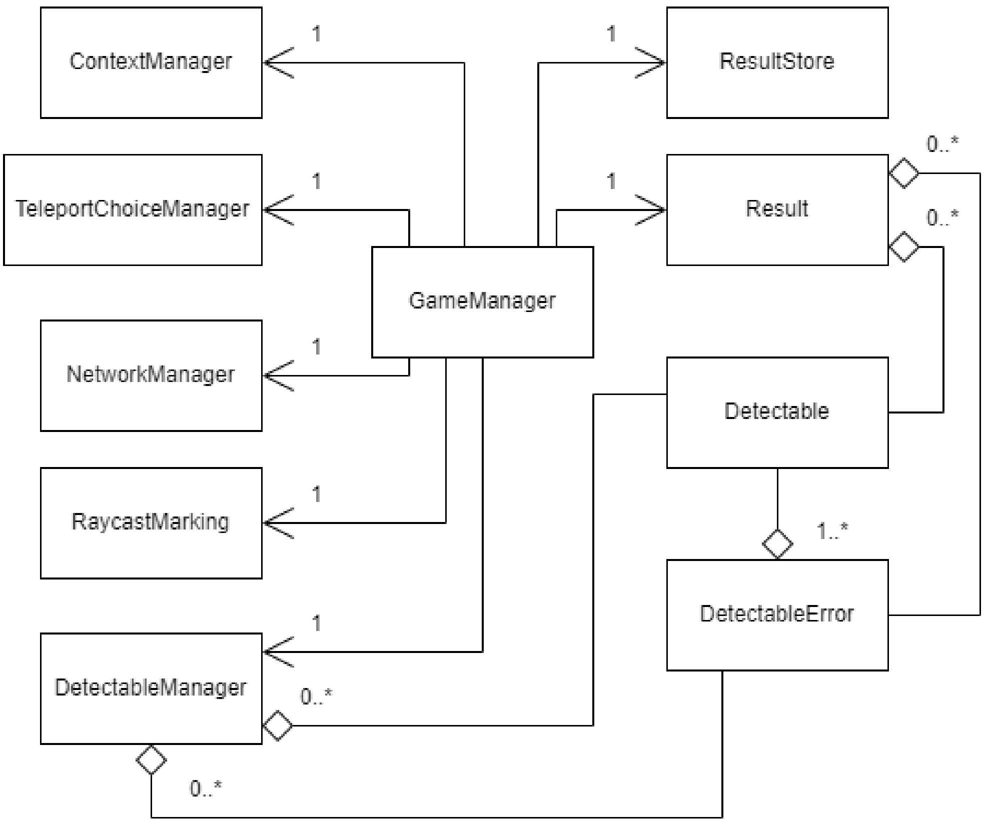

Figure 4.23: Overview of the most important classes

The BSc prototype was developed using the Unity asset Oculus Integration [168] in combination with Virtual Reality Toolkit (VRTK) [169], a collection of common solutions for VR such as movement and interaction. While this combination is functional, it is not necessarily ideal for future, further development. For example, the Oculus Integration asset provides some added functionality specific to Oculus/Meta devices (and "some Open VR supported devices"), such as support for Oculus Platform Solutions functionality, but other frameworks offer better support for cross-platform compatibility. The other half of the combination, VRTK, has undergone significant changes since version 3.0.0 — now over two years old — which was used in the prototype, seemingly to the extent that re-familiarizing oneself with the toolkit would require a time investment of a similar scale to learning other toolkits.

During development, the targeted HMD was a Meta Quest, which could have benefited from Oculus-specific functionalities. However, at the start of the development period, HVL's Faculty of Health and Social Sciences — which SimArena is a division of — had plans to buy a significant number of VR HMDs in the near future, and other HMDs were being strongly considered. Near the end of development, this became relevant when they invested in more than 100 units of the Pico Neo 3 HMD [71]. Therefore, cross-platform compatibility proved to be necessary. The Room of Errors application is compatible with these units but requires a version of the application with the Pico Neo Unity XR SDK [170] installed and activated. A separate version is needed as the application crashes on the Pico Neo 3 if support for Quest units is activated in Unity.

In 2019, OpenXR [171] was released, a standard that attempted to simplify and unify development of XR applications to a single cross-platform API. Prior to this, developers were required to use separate, proprietary APIs when developing for multiple XR HMDs. Also in 2019, Unity released their XR Interaction Toolkit [159], a toolkit that is significantly more closely integrated with Unity and has an explicit focus on cross-platform compatibility. It is actively being developed and features extensive documentation integrated with Unity's standard documentation [172]. Learning resources are plentiful, e. g. [173, 174] on Unity Learn, Unity's extensive platform for courses and tutorials. It was decided that the current thesis' project would use Unity's XR Interaction Toolkit to implement VR functionality.

4.6 Optimization

The BSc prototype described in Section 2.1 had occasional issues with its frame rate dropping to around 60-65 frames per second, which is not considered optimal. Meta requires interactive applications to maintain 72 frames per second to launch an application on their Meta Quest Store [175]. Even if there are no current plans to launch the Room of Errors application commercially, the requirement is a strong recommendation for preventing VR sickness, if nothing else. In other words, the prototype would require some amount of optimization.

Some principles were maintained from the prototype, such as limiting real-time illumination and dynamic shadows, and following best practices and recommendations from Oculus [176, 177]. This also includes, for example, using primitive colliders, such as boxes and capsules, instead of the more processor-intensive mesh colliders [178]. In addition, some performance-enhancing features that had not been utilized in the prototype were implemented. For example, Single Pass Stereo rendering (also known as Multiview) was available at the time of developing the BSc prototype, but due to lack of knowledge and experience, it was not enabled. Traditionally, VR rendering was performed sequentially, rendering first the left screen, then the right. The scene graph must be traversed twice, and many calculations must be performed twice. With Single Pass rendering, objects are rendered once and duplicated to the other eye (modified as necessary to adjust for the slightly different perspective). Most calculations need only be performed once, and rendering efficiency can be significantly increased [179, 180].

For Quest 1 applications, Oculus recommends between 350 000 to 500 000 total visible triangles at the same time and 200 to 400 *draw calls* per frame for light simulation applications [175]. A draw call essentially tells the GPU what to draw and how to draw it and can be resource-intensive. A single draw call contains meshes or vertices with similar properties or states [181]. Unity is able to combine different meshes with the same Materials to a single draw call, which increases efficiency.

The first optimization decision was to simplify the application's graphical fidelity. In the OR, the prototype displayed over 300 000 triangles, needing 780 draw calls (measured using the Unity Editor's Statistics window). In the post-operative ward, more than 700 000 triangles were rendered, needing 540 draw calls. These numbers needed to be reduced.

The prototype featured relatively realistic human 3D models. The model of a doctor consisted of 5476 vertices and 7244 triangles. These were changed to low-poly characters from the Unity asset "Low Poly World - Hospital" [48]. In this set of models, the model of a doctor contains 4324 vertices and 2166 triangles. In addition to the lower number of triangles, these models use a texture atlas which is also used by several other models in the rooms, which allows these other models to be combined with the characters in draw calls, increasing efficiency. Furthermore, the new art style avoids the "uncanny valley." It also allows actions that may reduce realism even further, such as unrealistic shadows or unrealistic character animation, without significantly impacting player perception [182]. The characters are animated by animations from Mixamo [183], a free library of motion captured animations for human characters.

Some 3D models were not replaced but simplified. For example, the Anesthesia Machine in the operating room was purchased for the BSc project. The original 3D model contained 23 778 triangles. For the BSc project, the model was simplified using Blender's model "Cleanup" functionalities [184], such as geometry decimation and vertex merging. At the start of the Master's project, the model contained 3923 vertices and 7479 triangles. The model could be simplified further, but as the machine is of great importance to nurse anesthetists, it was decided to keep the model relatively detailed. However, in Unity, the model required 130 draw calls alone. The model's primary source of complexity was that the model was divided into 52 separate sub-meshes and used 31 different Materials, including five different shades of blue and ten different shades of black. Instead of simplifying the model's resolution by further lowering the

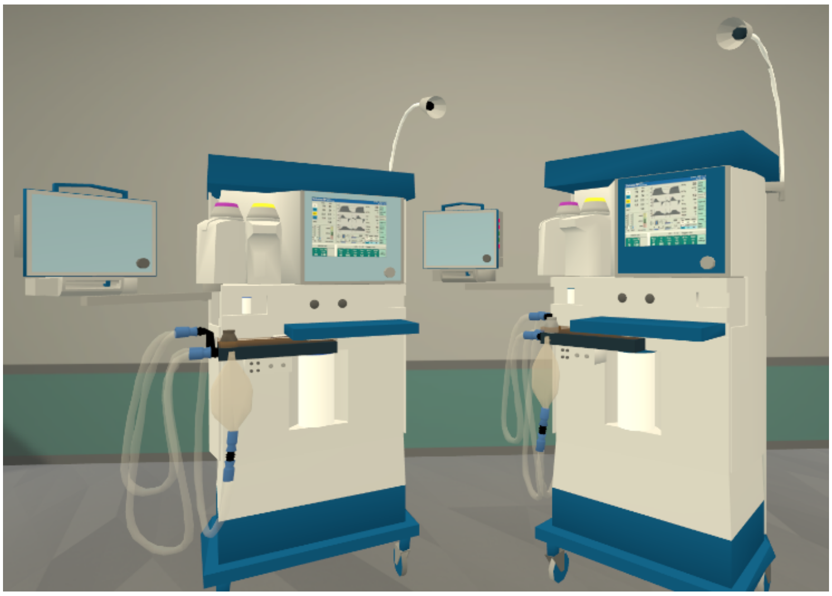

Figure 4.24: The anesthesia machine on the left requires 130 draw calls, while the machine on the right requires 41 draw calls.

number of vertices, Blender was used to merge several sub-meshes into one mesh and replace several Materials with similar colors. The current model contains 16 sub-meshes and uses 20 different Materials. In Unity, the model requires 41 draw calls. Visually, there is little difference. The two different models are shown in Figure 4.24.

Some models that were purchased for the Master's project were also simplified. For example, a table in the previously mentioned Unity asset "Low Poly World - Hospital" contained 1112 vertices and 1890 triangles. Using Blender, the model was simplified by removing unnecessary structures and details and currently contains 408 vertices and 588 triangles. In addition, its Materials were simplified to resemble a more metallic table more often used in the local hospital's surgical ward. The two different tables are shown in Figure 4.25.

In the current application, approximately 200 000 triangles are rendered in the OR, needing 380 draw calls. 330 000 triangles are rendered in the postoperative ward, needing 550 draw calls. This means that the number of draw calls is above Oculus' recommendation of 400 for Quest 1 applications.

The first set of user evaluations revealed that the postoperative ward needed more objects to study and more intended errors for users to find. At the time, the highest number of draw calls needed in the postoperative ward was 406. This was also above the recommendation, but since the application maintained a steady 72 frames per second, it was considered acceptable. Several suggestions for errors were provided by the evaluation participants and by lecturers for intensive care nursing. This led to adding a new patient to the room, which would be the second intubated patient. This type of patient requires several machines, various equipment, and several wires and tubes. After adding this

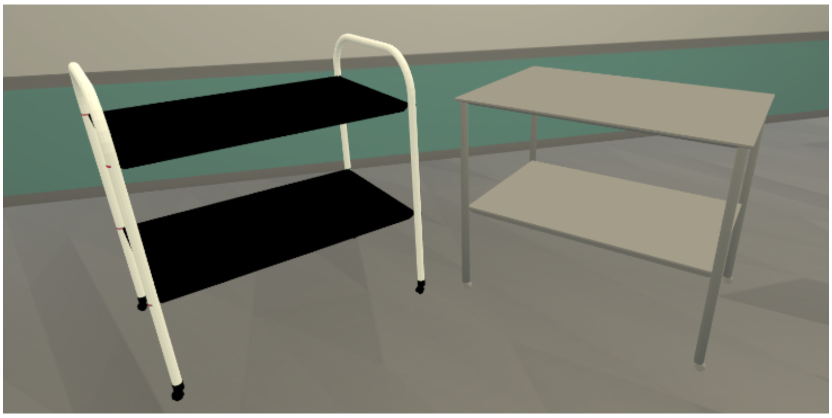

Figure 4.25: The table on the right contains less than half the number of vertices of the table on the left.

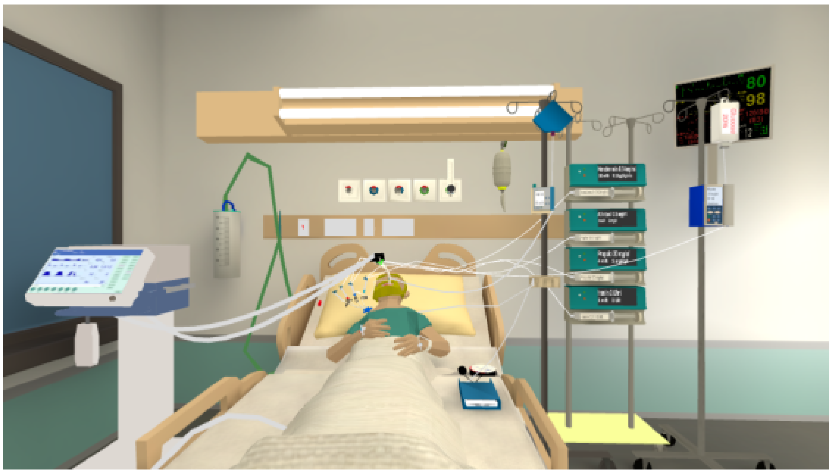

Figure 4.26: The second intubated patient in the postoperative ward

new patient and accompanying equipment, the number of draw calls had risen to over 700, and the number of frames per second had fallen to approximately 50. The added patient, devices, and machines can be seen in Figure 4.26

The most significant measure that was taken to reduce this new problem was to reduce the amount of transparency. Transparency adds significant graphical complexity, as the GPU has to make several calculations and decisions regarding what should be visible and what should be hidden. Both intubated patients in the room included 31 wires or objects that could be seen through, such as plastic wires going from a syringe pump to the access points for the patient's circulatory system. All these objects were changed to use simpler and opaque materials. In addition, some objects and details were removed. These actions were enough to raise the number of frames per second above 60 on a Quest 1 HMD. However, the application could keep a steady 72 frames per second on Quest 2 HMDs. The first set of user evaluations had been completed on Quest 2 units, and at

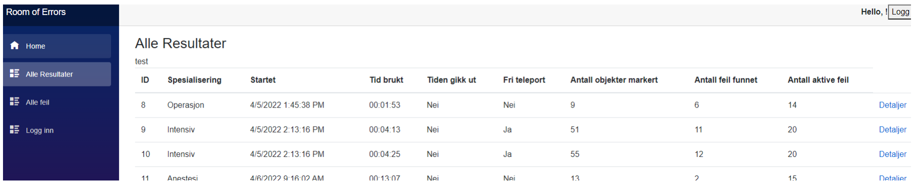

Figure 4.27: Overview of sessions in the web application

the time, the second set of evaluations was also planned to be completed on Quest 2 units. Therefore, it was decided to prioritize fixes and additions before the second set of evaluations instead of stabilizing the frame rate on Quest 1 HMDs.

4.7 The REST API and database

The REST API and database were initially added so the design team and the related PhD project could gather statistics on which errors were discovered by users. As the need for a debriefing following the simulation surfaced during development, its primary function is instead to give simulation facilitators an overview of which objects were marked by users, which gives them a basis for discussion during the debriefing. The API is not considered part of the main application, nor is it strictly necessary to answer this thesis' research questions. It is therefore summarized briefly and not described in detail.

The REST API and browser client were created with ASP.NET and developed in C#. They, and the Microsoft SQL Server database, are hosted on Microsoft's Azure cloud services. When a user runs out of time or chooses to finish the simulation in VR, the `ResultStore` class generates a JSON representation of the user session, including the user's specialization, errors that were active, which objects the user marked, and how long they took. This data is then sent as a POST request to the REST API.

The web application uses a layered structure, with Controller, Service, and Data layers. Controller classes receive the POST request. The received data is sent to the Service layer, where it is translated into C# objects. These objects are then sent to the Data layer, which finally submits them to the database. The browser client uses the service layer directly to retrieve data about VR user sessions, and Razor is used to generate web pages that are shown to the web application user. Figure 4.27 shows an overview of sessions in the web application.

The web application is intended for use by a single person to gather information, initially for the related PhD project and later potentially by simulation facilitators as part of a simulation training session. Therefore, optimization was not considered a priority during development. The stored data is not considered confidential or secret but is protected by a user login system where the user's password is encrypted using an implementation of the Scrypt algorithm. In addition, data submitted from HMDs needs to be accompanied by a "secret" 16-character string, which is stored as an environment variable in the Azure hosting service.

RQ2 considers the feasibility of the application. Evaluating the feasibility includes its financial feasibility. During development, the database and web application were hosted on Microsoft Azure, using the free solutions available to students through Azure for Students. The end of the current Master's project would mean that this Student solution is no longer viable. However, Azure offers plans for database hosting of 2GB of data for less than NOK 100 and web application service plans with 1GB memory and 60 minutes of computing per day for free. This should be sufficient to handle the expected at most 20-30 submissions per day, which is planned for the PhD study, and likely a similar number if the application was to be used in regular simulation training at HVL. If necessary, the next pricing tier includes 240 minutes of computing per day, also for less than NOK 100. If the application's use and requirements were to scale beyond this, several solutions are available, albeit at a price.

Chapter 5

Results

As mentioned in Section 3.5, two sets of user evaluation sessions were completed, separated by a month and a half. This chapter describes the results from these two sets, improvements made between the sessions, and an overview of the total results from different perspectives. Participants were observed while in VR, and after completing the simulation, they were asked to fill out a SUS form and participate in a semi-structured interview. The SUS form can be seen in Appendix B, and the topics for the interviews in Appendix C.

5.1 Notes on Statistical Values

In the following sections, several tables with raw results are provided. For comparisons, a p-value is given. In brief, in statistics, the p-value indicates the likelihood that we would get these results, given that the null hypothesis is true [185]. For example, in this thesis' results regarding multiplayer, where the p-value is 0.023: Given that it is true that multiplayer does not improve SUS scores, the likelihood that we would see the improvement of 12.0 is 2.3%. This does not necessarily mean that the improvement is confirmed beyond all doubt or that it is objectively true that multiplayer improves the SUS score. However, with an alpha value of 0.05, we can state that the results are statistically significant and therefore reject our null hypothesis. The improved score is unlikely to be due to random chance. If the opposite were true and the p-value was above 0.05, we could not state that we have confirmed that multiplayer does not improve the SUS score. Instead, we would only be able to state that we cannot reject our null hypothesis — we cannot definitively state that multiplayer does or does not have an effect on SUS score.

Where applicable, p values have been calculated with a two-tailed Student's T-test for unpaired samples with equal variances, with one exception noted in the text. The p-values have been calculated in Excel using its T-test formula (T.TEST()). Two-tailed means that we allow for the possibility of values both increasing or decreasing. Unpaired samples means that the samples have not been considered related (which may be inaccurate when comparing the first and second evaluation sessions). Equal variances means that we assume a similar degree of variance in the two compared samples.

Standard error is presented as Standard Error of the Mean (SEM). This means the standard deviations of average values - how much would the average value vary if we were to do the same tests with a different selection of the same number of participants [186]. These values have also been calculated in Excel by using Excel's formula for standard deviation (STDEV.S()), divided by the square root of the number of samples.

For some results, degrees of freedom (d.f.) are also provided. This number determines the number of independent values that can vary when estimating parameters such as the mean - how much independent information was used to calculate the parameter [187]? A higher degree of freedom leads to a more precise estimate. For two-sample T-tests that have been used for the following results, the value is calculated by N-2 since there are two parameters being estimated.

5.2 The first set of user evaluations

For the first round, six user evaluations were completed over two days at the end of February 2022. Feedback was strongly positive. Participants included one lecturer for nurse anesthetists, two for intensive care nursing, and three for OR nursing. The lecturers confirmed that they would consider using the application as part of their educational simulation training sessions.

Simulation results are shown in Table 5.1. Most participants found less than half of the errors intended by the developers. At the same time, they found several unintended errors and, in total, found more errors than were defined in the application. While participants were in VR, the unintended errors were accepted as part of the simulation and discussed similarly to the intended errors. For all unintended errors, participants gave explanations and valid reasons behind marking them as errors. Therefore, the objects they perceived as errors can be considered valid errors in the simulation context. However, some users experienced confusion related to their immersion and perception of the simulation. They were unsure whether visual elements such as colors and simplified 3D models, which are different from the same objects they are used to in their professional work, should be considered errors in the simulation context. This led to them being confused about whether they should mark the object or not. They did, however, agree that these differences can be used as a basis for discussion after future participants have gone through the simulation, which seems to confirm the need for a post-intervention debriefing. Debriefings are common in non-VR simulation training [45] and are planned to be part of the larger simulation experience. The users with less VR experience also reported a higher subjective level of stress, caused by feeling that they are expected to find every error and that they would "fail" the simulation if they do not. This indicates a need for a clear pre-briefing in future use — which, again, is planned to be part of the larger simulation experience. One such participant suggested that future users should be given a "test run" to familiarize themselves with the application and how to use it, which could lower the perceived pressure. At the time, this idea was dismissed, mainly due to practical concerns over whether

Specialization	VR experience	Time used	Marked objects	Intended errors found	Active intended errors	Unintended errors found	SUS score
Nurse Anesthetist	None	16m 18s	17	6	14	11	70.0
Intensive Care	Used once	8m 8s	7	4	11	3	80.0
Intensive Care	Used a few times	8m 35s	11	9	11	2	87.5
Operating Room	None	20m	17	5	14	12	92.5
Operating Room	None	20m	17	5	14	12	90.0
Operating Room	Used once	12m 16s	15	5	14	10	72.5

Table 5.1: Simulation results from the first evaluation sessions

students would be available for two sessions and to a smaller degree due to the belief that users would be too familiar with the simulation's errors.

Other participants remarked that they did not find the experience *realistic* but that they were still very immersed and found themselves highly involved in the task of finding errors in the room. The different experiences in terms of immersion and presence seem to be in agreement with the idea that these two concepts can be subjective and individual [133, 188]. Also notable was how different users focused on different aspects, similar to what was experienced by Bracq et al. [43] in their study using a VR Room of Errors simulation. Some evaluation participants could devote a significant amount of time and discussion to a specific object. In contrast, other participants completely ignored that same object, but both groups expressed satisfaction with the simulation experience. Some objects were also interpreted differently, which is further discussed in Chapter 6.

One user evaluation session was performed with two concurrent users cooperating to find errors. During the simulation, the two participants gradually communicated more and more as they became used to the experience of being in VR. Afterward, they commented that this communication created further opportunities for reflection and discussion, as they had seen examples of one participant discovering details the other participant had not noticed. Regarding future use for students, they remarked that they believed the possibility of cooperating in teams could also lower a user's perceived stress. They imagined many students potentially being worried about not being able to find all the errors, as had been experienced by another evaluation participant in a different session. However, in their experience, having a partner "disarmed" that worry and allowed them to lower their shoulders and focus on the simulation. These experiences seem to confirm the feasibility of a VR Room of Errors supporting multiple concurrent users.

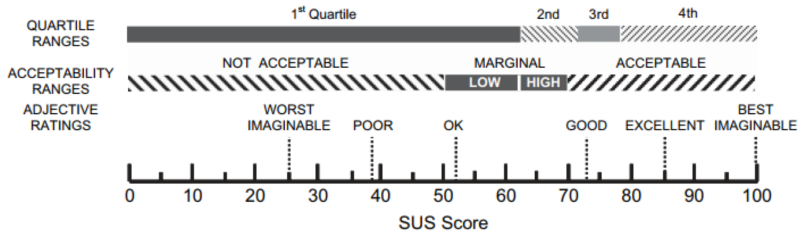

Figure 5.1: Bangor et al.'s SUS score scales [136]

After these six initial user evaluations, the Room of Errors application received an average SUS score of 82.1 by evaluators. According to Bangor et al.'s interpretation of SUS scores [136], this means the application can be considered above the “Good” adjective rating. Therefore, the application's usability is considered to be acceptable. Bangor et al.'s scales can be seen in Figure 5.1. Note that a sample size of six cannot necessarily be regarded as generalizable, and the application required further testing. The average scores for each question and the average total score can be seen in table 5.2

The score is lowered by participants consistently indicating their belief that they would require help to use the simulation on their own (SUS questions 4: ”I think that I would need the support of a technical person to be able to use this simulation” and 10: ”I needed to learn a lot of things before I could get going with this simulation” [20]). After filling out the form, several participants explained that they foresaw difficulties with setting up the HMDs and preparing VR for student use. These comments may indicate that the lowered SUS score is due to unfamiliarity with VR in general and not necessarily the application itself.

Also of note is the SUS score given by the two participants who cooperated concurrently. The total scores given by them were 92.5 and 90. By excluding these two users, the average score goes from 82.1 to 77.5. This suggests a markedly different experience for the concurrent users compared to the users who were alone. The lower score of 77,5 is still above Bangor et al.'s “Good” rating, so the application's usability is still considered acceptable if the future simulation sessions were to be performed without cooperation.

The results show that the lecturers for intensive care nursing used less time before choosing to finish and marked fewer objects as errors than lecturers for the other specializations. However, both gave SUS scores of 80 or above. This is interpreted as them seeing the application as usable but that the course for intensive care nursing students may require further development in future iterations. Prior experience with VR did not seem to affect participants' SUS scores. Participants who had previously tried using VR gave scores ranging from 72.5 to 87.5, and participants with no experience gave scores ranging from 70 to 92.5.

Question	1	2	3	4	5	6	7	8	9	10	Total (SEM)
Avg. score	3.7	3.8	3.5	1.8	3.8	3.2	3.8	3.8	3	2.3	82.1 (3.8)

Table 5.2: SUS results from the first evaluation sessions (n=6)

5.3 Improvements before the second round of evaluations

As part of the first user evaluations, participants were asked if they had suggestions for improvements or thought something in the simulation should be changed. None of the participants had specific issues with or suggestions for the application or its usability. Instead, most of them responded with suggestions for improving existing errors or ideas for new errors that could be implemented. Therefore, development in the time period between the two rounds of evaluations focused on adding or altering errors. In particular, the participants related to intensive care nursing had many suggestions for errors as the postoperative ward had relatively few errors. Also, they both chose to finish the simulation after approximately eight of the allotted twenty minutes, compared to other specializations, which all spent over fifteen minutes or even let the time run out. These suggestions led to the addition of a third patient in the postoperative ward and an additional nine errors, as described in Section 4.6.

In addition, some improvements were based on observational data from observing the participants while they were in VR. For example, many users had to be reminded that they could move between contexts in the operating room, so attempts were made to make the button to progress more visible, as described in Section 4.1.3. Also, some participants forgot that they were limited by time, so they did not pay attention to the clocks counting down and were surprised when the simulation ended. This led to the addition of the text-to-speech voices notifying users about their remaining time, also as described in Section 4.1.3.

5.4 The second set of user evaluations

For the second round, eight user evaluations were completed over two days at the start of April 2022. Participants included one lecturer for nurse anesthetists, two for intensive care nursing, and four for OR nursing. Due to a short notice cancellation, one participant from intensive care nursing participated once on the first day and once on the second day as part of a multiplayer session, for a total of three intensive care nursing evaluations when adding the second intensive care nurse. Four of the participants had also participated in the first sessions (meaning that five participants tried the application twice if the intensive care nursing lecturer who participated on consecutive days is counted as separate evaluations). The most significant finding in the second round was the potential benefits of users going through the simulation a second time. Although participants had made positive statements about the application's usability in the first round, several users needed a few minutes to get comfortable with VR and the hand controllers their first time. These minutes of acclimatization were much shorter in the second round. All users who participated for the second

Specialization	VR experience	Time used	Marked objects	Intended errors found	Active intended errors	Unintended errors found	SUS score
Nurse Anesthetist	Used several times	13m 7s	13	2	15	11	65.0
Intensive Care	None	14m 37s	20	8	20	12	97.5
Intensive Care	Used a few times	20m	42	16	20	26	97.5
Intensive Care	Used once	20m	42	16	20	26	82.5
Operating Room	Used a few times	11m 23s	20	5	14	15	80.0
Operating Room	Used twice	20m	25	8	14	17	77.5
Operating Room	Used once	20m	22	5	14	17	95.0
Operating Room	Used once	20m	22	5	14	17	87.5

Table 5.3: Simulation results from the second evaluation sessions

time explicitly recommended that future users should perform the simulation more than once since they had found their second session much more useful. Users remarked that they had a significantly easier time focusing on their task, as they knew what was expected of them and had a clearer vision of what they could expect from the simulation. One participant stated that they had a lot more fun during their second session and that the concept made more sense this time. These statements reinforced the similar suggestion from one participant in the first round of testing that an initial “test run” could reduce the perceived pressure and worries about not finding all errors.

Simulation results are shown in Table 5.3. As in the previous round of evaluations, all participants found several unintended errors and most found more errors than were available. Again, the unintended errors were accepted as part of the simulation and discussed similarly to the intended errors, and participants gave explanations and valid reasons behind marking them as errors.

Two sessions were completed with two concurrent users, for a total of four evaluations with “multiplayer.” All four were second-time participants. One session included the two users who had cooperated in the first round of testing, and the other session included users who had not previously tried multiplayer. As in the first round, participants felt that cooperation markedly improved the experience. The ability to discuss with a partner and reflect on potential errors — especially when there is not necessarily a set of “correct answers” — added significant value for them. These statements could be argued to bear more weight coming from the users who had tried both going through the simulation alone and in cooperation.

SUS scores from the second round of evaluations gave a total average score of 85.3, an increase of 3.2 from the first round. A significant outlier is one participant who gave the application a score of 65 (the next lowest being 77.5). It is suspected that this score is influenced by the participant encountering a progress-breaking bug in the game's tutorial, which ended up being resolved by the participant taking off their HMD so the developer could restart the application entirely. They encountered no further problems during the rest of their session. However, this score should not be dismissed, as this participant claimed to be fairly familiar with VR, as they own a Meta Quest privately. On the other side of the spectrum, the highest score in this round, 97.5 (note: this score was given by two separate participants), was given by a first-time participant who had never used VR or hand controllers and had some initial struggles getting used to the technology. Also of note, this same participant who gave a score of 97.5 is the participant who tried the application on consecutive days — and gave the application a score of 87.5 the following day.

Participants who completed the simulation in teams gave an average SUS score of 90.6, compared to 80 for the solo participants. The average score for solo users is lowered by the previously mentioned outlier, but even if this score is excluded, the average solo score increases to 85 — so multiplayer users still gave the application a higher score on average, which was also seen in the first round of testing.

Average SUS scores overall for the second round can be seen in Table 5.4. Note that these results contain two samples from the previously mentioned intensive care nursing lecturer who participated twice, on consecutive days. The total average score of 85.3 means that the application can be considered of "Excellent" usability according to Bangor et al.'s interpretation of SUS scores [136]. As with the first round of evaluations, questions 4 and 10 received an average score below 3, indicating users' belief that they would need help to use the application on their own. Two participants were asked if they interpreted this question as directed at VR technology or at the application itself. One of them contradicted the suspicion from the first round that these relatively low scores were directed at VR technology when they stated they believed they would handle the VR aspect well but were unsure how to start the application. The other participant responded that they felt their experience had depended on the developer being present to answer questions and give them instructions when getting started. The simulation is intended to be used as part of simulation training, which includes a briefing before getting started and the presence of a simulation facilitator who can get participants started and answer questions. Therefore, these concerns are considered an acceptable potential drawback of the application.

For the second round, question 9 ("I felt very confident using the simulation") also received an average score below 3, down from exactly 3 in the first round. This is potentially a larger concern since this could be interpreted as participants being unsure about the environment and errors in them, as described by the participants in the first round of evaluations who were unsure if, for example, simplified 3D models should be considered errors. The score also potentially has implications for the thesis' research questions and the feasibility of the concept

Question	1	2	3	4	5	6	7	8	9	10	Total (SEM)
Avg. score	3.9	3.8	3.8	2.5	3.8	3.4	3.5	4	2.8	2.9	85.3 (4.0)

Table 5.4: SUS results from the second evaluation sessions (n=8)

Question	1	2	3	4	5	6	7	8	9	10	Total (SEM)
Total average score (n=14)	3.8	3.8	3.6	2.2	3.8	3.3	3.6	3.9	2.9	2.6	83.9 (2.76)

Table 5.5: Total average score across both evaluation rounds

in VR. However, the score can also be considered another argument in favor of participants completing the simulation twice. As seen in Table 5.7, the average score for question 9 is 2.6 for first-time users and 3.4 for second-time users.

As for the improvements between evaluation rounds described in the previous section, the one participant from intensive care nursing who had participated in the first round expressed enthusiasm over the new errors that had been implemented. Their SUS score increased from 87.5 in the first round to 97.5 in the second round. Multiple users appreciated the text-to-speech voice telling them how much time was remaining. As mentioned in Section 4.1.3, attempts at making the button to progress to the next context were not as successful. Most users still had to be reminded that the button existed, although this was less frequent in second-time users.

5.5 Total results

Overall, the application has been evaluated fourteen times by nine unique users. Tullis & Stetson [189] state that sample sizes of eight SUS evaluations reach an "agreement" of 75 percent with a "correct" score, and sample sizes of twelve SUS evaluations reach an "agreement" of 100 percent — "agreement" meaning to what degree the score matches a final "correct" score from a larger sample size [190]. The difference between the first and second rounds of evaluations can be considered relatively small (82.1 vs. 85.3), and the difference is not statistically significant (p=0.583). Differences in the application between the two rounds should not be noticeable by OR nursing and nurse anesthetist lecturers. Therefore, an argument can be made to compare all fourteen samples as one selection. These "total" results are shown in Table 5.5. According to Bangor et al.'s scales [136], this score of 83.9 is above the threshold for a "Good" rating. An overall sample size of fourteen could be considered sufficient to rate the application confidently. However, the fact that some of these scores were "duplicates" from users who participated twice leads to some uncertainty and potentially inaccurate results. The score of 85.3 from the second set of evaluations may be more accurate. The application's usability is, in any case, considered acceptable.

An overview of all evaluation sessions is shown in Table 5.6. The total SUS scores

Participant	Specialization	First session SUS	Multi-player	Second session SUS	Multi-player
A	Nurse Anesthetist	70			
B	Nurse Anesthetist	65			
C	Intensive Care	80			
D	Intensive Care	87.5		97.5	With E
E	Intensive Care	97.5		82.5	With D
F	OR	90	With G	95	With G
G	OR	92.5	With F	87.5	With F
H	OR	72.5		77.5	
I	OR	80			

Table 5.6: Overview of evaluation sessions by participant

can be looked at from several different perspectives. As previously mentioned, round two of evaluations saw an increase in average score by 3.2, which is not statistically significant. The small difference does, however, suggest that results from the first round can be combined with results from the second round into one selection, as previously suggested, when looking at the results from a different viewpoint. For example, comparing results between first-time users across both evaluation rounds with second-time users shows an improvement of 6.3 when using the application for a second time. These scores are shown in Table 5.7. The difference is not statistically significant (p=0.289). In fact, excluding all users who did not complete the simulation twice gives a p-value of 1 between the first and second session. This value may be interpreted as the second session giving no improvement. However, the sample size of second-time users is small (n=5). Also, the qualitative findings should be considered. All of the second-time users strongly stated that they had a better simulation experience *because* it was their second time and provided extensive explanations for why. For example, they could focus on their task to a greater extent instead of the VR technology, which gave them more opportunities to learn and discuss what they were observing. They also experienced significantly less subjective stress over what was expected of them. Therefore, the qualitative results suggest that a second simulation completion was beneficial — but this may be an aspect that is not captured well by the SUS.

A second perspective is comparing scores between solo and multiplayer users across both evaluation rounds, which can be seen in Table 5.8. The difference of 12.0 is notable and is statistically significant (p=0.023). The sample size of multiplayer users is also small (n=6). However, again, the qualitative results support the finding. All of the multiplayer users stated that they had a better simulation experience because of the opportunities and benefits they gained from cooperation. Therefore, this aspect is also considered important.

Table 5.9 shows the difference in average SUS scores between the three AIO nursing specializations. The scores from nurse anesthetist lecturers are markedly lower than other specializations, at an average of 67.5 compared to 89 for in-

Question	1	2	3	4	5	6	7	8	9	10	Total (SEM)
First time users (n=9)	3.8	3.8	3.6	2	3.8	3.1	3.7	3.9	2.6	2.6	81.7 (3.68)
Second time users (n=5)	3.8	3.8	3.8	2.6	3.8	3.6	3.6	4	3.4	2.8	88 (3.74)

Table 5.7: Comparison between first time and second time users. (p=0.289, d.f.=12)

Question	1	2	3	4	5	6	7	8	9	10	Total (SEM)
Solo users (n=8)	3.9	3.6	3.5	1.6	3.9	3	3.5	3.9	2.4	2.3	78.8 (3.63)
Multi-player users (n=6)	3.7	4	3.8	3	3.7	3.7	3.8	4	3.5	3.2	90.8 (2.20)

Table 5.8: Comparison between solo and multiplayer users. (p=0.023, d.f.=12)

tensive care nursing lecturers and 85 from OR nursing lecturers. However, few conclusions can be gained from the scores with a sample size of only two participants. Neither of the two had the opportunity to evaluate the application a second time nor to evaluate the application while cooperating in multiplayer, both of which seem to lead to better experiences. On the other hand, the scores cannot be dismissed entirely, especially considering that one of the two was the user with the most VR experience out of all nine unique participants. This user did experience the progress-breaking bug mentioned in the previous section — but other users also encountered this bug before it was fixed, and those other users gave the application a score of 90 or more. Perhaps more importantly, the specific participant discovered several instances of inconsistency in details. For example, in the Perioperative context, where the setting is in the middle of surgery, the patient is receiving a blood transfusion. In reality, such a situation would also likely have been reflected by gauze having significant amounts of visible blood on them. However, the visible gauze in the context has minimal amounts of blood on them. The participant also found several errors that they could not directly mark as an error and stated that they had difficulties reading some of the text in the operating room. Although other users did not experience these issues to the same degree, it is valuable information, and the relatively low SUS score gives valuable insight.

VR sickness has not been quantitatively measured. However, the concept of VR sickness was explained to users before entering VR. Several users remarked in response to the explanation that they were prone to car sickness. After finishing the simulation, these users later stated that they had not experienced any nausea.

Question	1	2	3	4	5	6	7	8	9	10	Total (SEM)
Nurse Anes-thetist users (n=2)	3.5	3.5	3	0.5	3.5	1.5	3.5	4	2.5	1.5	67.5 (2.5)
Intensive care users (n=5)	4	3.8	4	3	4	4	4	4	2.8	3	89 (3.67)
OR users (n=7)	3.7	3.9	3.6	2.4	3.7	3.4	3.4	3.9	2.9	3.1	85 (3.18)

Table 5.9: Comparison between users from the three AIO specializations

Chapter 6

Discussion

In this chapter, the results presented in the previous chapter are discussed further. In addition, the application itself, potential improvements, and future work are considered.

6.1 Acceptance of errors

As described in the results, unintended errors were generally accepted as part of the VR simulation. However, some objects were also interpreted differently. For example, a model of a needle counter is included in the operating room. In reality, needle counters are used by OR nurses to keep count of how many sutures have been opened and how many needles have been returned to them by surgeons. The model can be seen in Figure 6.1. In the simulation, the model is used in an error where users can see that four sutures have been opened, but only three needles have been placed in the needle counter after the operation — a needle has been left inside the patient. One user interpreted the 3D model as a cellphone and remarked that “there should not be a cell phone in a sterile area,” and marked it as an error. The statement is true; there should not be a cell phone in a sterile area. A different user stated in the postoperative context that “that should be in the trash by now.” Again, this is a valid error, as they are usually thrown away for hygienic and safety reasons. A third user remarked that “we’re missing a needle” — the intended error.

Another object of note is a table in the operating room, covered by a sterile cloth and a sterile package. The table can be seen in Figure 6.2. The table was originally inserted as “decoration” to fill the otherwise empty left side of the room without considering its potential as an error. However, several participants spent several minutes discussing the table, asking, for example: “What is that on top of the table”; “Why is that table there, so far away from the other sterile things”; “What’s under the cloth”; or “What’s inside the package?” Even if the table was not necessarily considered an error, it caused a reaction and discussion, which, again, is an important purpose of the Room of Errors training. In that sense, the VR application works.

Based on the qualitative results, both intended and unintended errors were

Figure 6.1: 3D model of a needle counter in the operating room.

Figure 6.2: Table covered by sterile cloth in the operating room.

accepted as part of the simulation. However, the question of how to handle these situations where objects are interpreted differently is raised. In the example of the needle counter, the first two users found unintended errors *instead of* an intended error. Responding that they were wrong or that they did not find the specific error that was intended by developers would not have been constructive for the simulation experience — the point of the Room of Errors training is to raise awareness and encourage participants to speak up when something is not right. The simulation results shown in Tables 5.1 and 5.3 also show that participants generally found more than twice the number of unintended errors than active intended errors. Telling users that they, for example, found twenty out of fourteen possible errors if fourteen intended errors were active and they had marked twenty objects would not make sense. These situations support the idea presented by Bracq et al. [43], who, in their study, did not tell users the number of errors they found but instead used the marked objects as a basis for discussion in a debrief.

6.2 Improving the experience

The results indicate that, for the best possible experience, participants should complete the simulation in groups of two or more and that it should be completed more than once. These findings mirror traditional simulation training, which is usually completed more than once in teams. It was expected that participants would have a better experience when collaborating. As described in Section 2.7.3, studies have indicated that social presence in VR has positive effects on motivation and acceptance of the virtual environment. However, it was not expected that participants would recommend a second session so strongly. During development of the application, several functionalities were implemented to make the experience more accessible for users — for example, the "tooltips" described in Section 4.1.2. After informal testing in the design team, these functionalities were deemed to lessen the experience of finding errors or make it too easy for users. Therefore, the design team expected users to find it too easy their second time, as they would likely remember the errors they found their first time. However, this was not the case. As described in Section 5.4, evaluation participants had a markedly improved experience their second time and were at least as enthusiastic about their task as their first time. Participants did not recall errors they had found the first time, which is confirmed by comparing which intended errors were found. For most participants, intended errors that were found their first time were not found their second time.

Recalling errors may depend on the amount of time between the two sessions. The two sets of evaluation sessions were completed a little over a month apart, which apparently was sufficient for participants to forget errors. In simulation training at SimArena, it is common for participants to complete a second simulation immediately following their first session, which may be too short for the Room of Errors training. However, recalling errors may not be a problem. One participant completed their second session the day after their first session, and did remember many of the errors they had found the day before. They were not asked directly if they felt that recalling errors lessened the experience, but their feedback suggests it did not, as they were very positive and recommended future users complete the simulation twice. The benefits of completing the simulation twice included being more comfortable with VR and what is expected of them in the simulation. This is supported by Jeffries [45], who says that establishing ground rules related to specific simulations keeps the learning focused and that an initial investment of time for participants to orient themselves to the technology that is part of the simulation is essential. For this application, a briefing before starting VR would contribute to establishing ground rules, and a "first round" would allow participants to orient themselves to the technology.

6.3 Design considerations and consequences

RQ1 asks: How can we design a simulation in VR that enables users to recognize errors as part of the simulation and not as the result of a faulty application? Design choices are described in Chapter 4. The question is not necessarily considered to be fully answered. The current application's design is not the only way to design such an application, nor can it be definitively stated that it is the best way. However, it is *a* way that, according to the results, has led to a valid application that fulfills its purpose.

Some drawbacks still exist. As previously described, the application's graphical fidelity and realism led to some instances of confusion for participants. However, this can be mitigated by a thorough briefing before entering VR and was a notably smaller issue for participants who tried the application a second time. In addition, there is the question of marking errors, as briefly mentioned in Section 4.1.2. Several users noticed unintended errors involving an object that *should* be present but was missing. For example, in the preoperative context, a nurse anesthetist remarked that the patient should have been receiving intravenous fluids, which is common before surgery. The user then spent some time looking for something to mark. Similar situations also occasionally led to frustration as participants attempted to mark related objects that had not been defined as markable in the game engine. An ideal solution would probably include making all objects markable by default and instead defining which objects should *not* be markable. Alternatively, a different method of noting errors would be worth exploring. One idea that has been suggested during development involved users placing visible markers on or near objects, such as small flags. However, this could create different issues, such as logically identifying what the user intended when placing the marker, for the sake of the debriefing — which in the current application is accomplished by specific objects having a unique identifier that can be stored.

An entirely different question that has been asked near the end of the project is whether the application would also be viable on a desktop computer with a regular computer monitor without using VR HMDs. This question does not have a simple answer. The application itself would technically be runnable with minor changes in the Unity editor. VR was one of the initial requirements by project initiators, so the idea was never considered until the question was asked after round two of user evaluations. The application takes advantage of the natural immersion achieved by simply putting on an HMD, as this likely allows users to, for example, accept the low-resolution 3D models as part of the experience and environment. However, on a normal computer monitor, the application's relatively low graphical fidelity would become significantly more apparent compared to modern 3D games. Users may also find it more difficult to recognize virtual representations of familiar objects from their professional work, which, as previously described, has proven difficult for some objects. Another aspect to consider is the intended errors. One error, "Hole in the sterile packaging for an extra instrument," takes advantage of the freedom offered by VR HMDs. To find the error, users are required to bend down and turn their heads to study an object from the side. This action would be difficult to accomplish with traditional keyboard and mouse controls on a desktop computer. A desktop computer version could be worth exploring, but it is not considered likely that the current application would work as well. However, the concept itself would likely work. A computer more powerful than the standalone VR HMDs would allow for higher fidelity 3D models and more realistic environments, which could potentially regain some of the immersion lost by not using an HMD. However, such a version would require time and resources and would potentially have consequences regarding the practical and financial feasibility of developing the application.

6.4 Performance

As mentioned in Section 4.6, the application currently does not maintain 72 frames per second on Meta Quest 1 HMDs, but it does on Quest 2 HMDs. Furthermore, the frame rate is stable on Pico Neo 3 HMDs. These units are likely to be used in the related PhD project, as HVL's Faculty for Health and Social Sciences has purchased a large number of them for use in education. However, this means the application is in the middle of two opposites. On the one hand, the application can be optimized for use on Quest 1 HMDs. For example, initial experiments with implementing occlusion culling (not rendering objects that are behind other objects from the user's perspective) have raised the frame rate on Quest 1 units to a stable 72 in all rooms. On the other hand, the increased capabilities of Quest 2 and Pico Neo 3 units allow for greater visual detail, increasing the number of objects in the rooms and enabling some features that were disabled for the sake of the Quest 1. For example, the transparent cables and devices that were made opaque, as described in Section 4.6.

However, most importantly: the application maintains a stable frame rate of 72 frames per second on the Quest 2 and Pico Neo 3 HMDs, the units most likely to be used in the related PhD project at HVL.

6.5 Future work

Several aspects of the application can be improved in future development. Natural areas of improvement include, for example, adding more intended errors to discover or increasing and improving the level of detail on existing objects. More intended errors could also be beneficial for the purposes of activating different errors for a simulation participant's first and second Room of Errors completion. There has also been expressed interest in functionality that allows lecturers to add their own errors without the help of a software developer. This idea may be possible to implement but would require some amount of development. The lecturers would, for example, have to consider 3D models to be added, which would have to satisfy some requirements regarding graphical complexity.

There is also potential for further development in other areas. Filming and recording simulation sessions is useful in traditional simulation training for the purpose of watching the session back during the debrief. For the user evaluations of the current application, the VR perspective was cast to a browser using Meta's casting functionalities, which can also stream to smartphone apps. The perspective could have been recorded on a computer using software such as *Open Broadcaster Software* [191]. However, this solution is cumbersome for simulation training facilitators. Therefore, project initiators at SimArena have requested simpler methods of casting the VR perspective to other displays and recording it.

The Room of Errors application has significant potential for further development. Lecturers for Bachelor level nursing studies, as well as other specializations, such as Pediatric nursing, have expressed interest in a version tailored to their needs. It would take some time to create suitable environments and suitable 3D models, but the functionality of marking errors works. As was suggested in the "Room of Horrors" Bachelor's thesis [24], there is also a potential for a similar application targeting entirely different professions where details in the professional environment are important. For example, pilots or mechanics.

The simulated environments have the potential for use in other applications. During the user evaluations, one participant suggested using the current application — or a variant of it — to allow intensive care nursing students to practice a standardized check of patient status, instruments, and machines. There is also potential for using the environments in a different, related VR application that would let users practice procedures and routines, similar to the VR ABCDE training application in use at the Norwegian University of Science and Technology (NTNU) [192].

The Room of Errors concept itself also has additional potential, not necessarily linked to the current application. As described in Section 3.4, the design team initially had plans to design a reenvisioning of the Room of Errors concept to include scenarios. These scenarios would have allowed users to experience adverse events that play out before them, which could have given them a new perspective and additional insight. This concept was never fully defined before it was decided to be outside the scope and objectives of the related PhD study. However, the idea of scenarios is still considered interesting and full of potential by project initiators. As also mentioned in Section 3.4, members of the design team without a software engineering background frequently expressed that they were unsure of the possibilities in VR. In hindsight, spending more time defining ideas and clarifying what can and can not be done in VR could have been beneficial and potentially led to a concept within the PhD project's scope. Therefore, the idea of scenarios could be worth exploring further in future development.

From an academic perspective, as mentioned, the application will be used as an experimental intervention in a PhD project studying patient safety education [16]. This project is expected to produce published articles about the results of students using the application. Furthermore, as mentioned in Section 3.2, an article reporting the results from the first user evaluations has been accepted for publication [193] (not yet published at the time of submitting this thesis). The article focuses on the quantitative results. There is a possibility for additional articles focusing on qualitative results and results from the later evaluation sessions.

6.6 Limitations

SUS measures a subjective user perception of usability, which may be unreliable and does not necessarily correlate to an *objective* measure of usability. According to Tullis & Stetson [189], eight participants should be in "agreement" of at least 75 percent with a "correct" result from a larger sample size. Therefore, the quantitative results from nine participants in the current thesis are likely not entirely inaccurate. However, they are not definite. Therefore, the results are given with some reservations due to the number of samples.

The results are also potentially skewed due to bias. All but two of the participants in the evaluation sessions had met the current author previously, and some have known the author for several years in a professional capacity as an OR nurse. Therefore, the possibility that this familiarity affected their scores positively cannot be entirely ignored. However, the same familiarity can also be argued to make it easier for the participants to be honest in their feedback. This is an unknown factor that should be remembered when considering the results.

The SUS has given useful data regarding the application's usability. However, as mentioned in Section 5.5, the SUS may not be an ideal tool to reveal findings such as the benefits of completing the simulation a second time. Therefore, further research is suggested, potentially using other quantitative tools or a more qualitative approach.

6.7 Financial feasibility

RQ2 asks: How feasible is it to create a virtual Room of Errors simulation in VR which supports multiple concurrent users? One aspect of feasibility is financial feasibility. The only expenses for the project were purchasing 3D models and other assets, and the total cost for all purchases was $264. An overview of all purchases can be seen in Appendix E. This is below the planned maximum limit of NOK 10 000 determined by SimArena at the start of the project.

Most importantly, the multiplayer implementation is free for up to twenty concurrent users. As mentioned in Section 4.3, a higher number of users would infer a cost. In a similar vein, during development, the REST API and database have been hosted on Microsoft's Azure Cloud, which offers some services for free for students. As mentioned in Section 4.7, this solution will not be viable after this project, but the services provided for approximately NOK 100 should be sufficient.

Therefore, the application is considered financially feasible.

Chapter 7

Conclusion

As stated in Section 1.3, this thesis aimed to understand how users' experience of a VR application is affected when they are essentially looking for something wrong with the application. There are still aspects of this goal that are unexplored and unexplained, but some insight has been gained. More could likely be gained by studies with a more qualitative focus. Users are able to accept the "faulty" simulated environment, as well as intended and unintended errors in the simulation. The VR application that has resulted from this thesis is considered usable, based on the SUS score of 85.3 from the second set of user evaluations and qualitative data.

The research questions in this thesis were:

- RQ1: How can we design a simulation in VR that enables users to recognize errors as part of the simulation and not as the result of a faulty application?
- RQ2: How feasible is it to create a virtual Room of Errors simulation in VR which supports multiple concurrent users?

RQ1 is considered answered by Chapter 4's description of the application's design. As stated in Section 6.3, it cannot be stated that the described design is the only way or the best way to design such an application — but it is *a* way, which fulfills its purpose. The design process has also been described in Chapter 3.

RQ2 is also considered answered. Development of a virtual Room of Errors simulation in VR that supports multiple concurrent users is considered practically and financially feasible, and, according to the results, the application is viable and usable. The concept works in VR.

In summary, the Room of Errors concept works in VR. The qualitative results indicate that a high degree of graphical realism is *not* necessary for VR users to accept the simulated environment. "Multiplayer" adds a layer of communication and cooperation which has benefits for the simulation training. The results show that users should complete the simulation in cooperation with at least one other user and that they should complete the simulation more than once. Also, briefings before starting VR and debriefings following VR are recommended.

The application resulting from this project will be used in an ongoing PhD project [16] that aims to study how to introduce patient safety as a study topic for nurse anesthetist, intensive care nursing, and operating room nursing studies. Initially, the application will be part of an experiment in which students use the application as an addition to their scheduled simulation training. Depending on how it is received, the application may be included as a regular part of their studies.

Bibliography

[1] cVR, “cVR 2022 — 1ST IEEE INTERNATIONAL CONFERENCE ON COGNITIVE ASPECTS OF VIRTUAL REALITY.” https://scitope.com/cvr22/, 2022. (accessed 2022-05-25).

[2] Helsedirektoratet, “Pasientskader i Norge 2020 - Målt med Global Trigger Tool.” https://www.helsedirektoratet.no/rapporter/pasientskader-i-norge-2020-malt-med-global-trigger-tool?download=false, Sept. 2021. (accessed 2022-05-23).

[3] E. N. de Vries, M. A. Ramrattan, S. M. Smorenburg, D. J. Gouma, and M. A. Boermeester, “The incidence and nature of in-hospital adverse events: A systematic review,” *BMJ Quality & Safety*, vol. 17, pp. 216–223, June 2008.

[4] T. L. Rodziewicz, B. Houseman, and J. E. Hipskind, “Medical Error Reduction and Prevention,” in *StatPearls*, Treasure Island (FL): StatPearls Publishing, 2022.

[5] F. Oyebode, “Clinical Errors and Medical Negligence,” *Medical Principles and Practice*, vol. 22, pp. 323–333, June 2013.

[6] J. M. Farnan, S. Gaffney, J. T. Poston, K. Slawinski, M. Cappaert, B. Kamin, and V. M. Arora, “Patient safety room of horrors: A novel method to assess medical students and entering residents’ ability to identify hazards of hospitalisation,” *BMJ Quality & Safety*, vol. 25, no. 3, p. 153, 2016.

[7] A. Olson, N. Olson, J. Wilson, A. Muck, R. Garcia, and K. Balhara, “38 Room of Horrors: A Pilot Curriculum to Enhance Nurses’ Patient Safety Awareness,” *Annals of Emergency Medicine*, vol. 72, no. 4, pp. S18–S19, 2018.

[8] B. Murphy, “4 hazards medical trainees face in hospital “room of horrors”.” https://www.ama-assn.org/education/accelerating-change-medical-education/4-hazards-medical-trainees-face-hospital-room, Oct. 18. (accessed 2022-05-23).

[9] A. S. Clay, S. M. Chudgar, K. M. Turner, J. Vaughn, N. W. Knudsen, J. M. Farnan, V. M. Arora, and M. A. Molloy, “How Prepared Are Medical and Nursing Students to Identify Common Hazards in the Intensive Care

Unit?," *Annals of the American Thoracic Society*, vol. 14, pp. 543–549, Apr. 2017.

[10] F. E. Turrentine, A. T. Schroen, P. T. Hallowell, B. A. Quatrara, P. W. Smith, M. D. Williams, and J. A. Haizlip, "Enhancing Medical Students' Interprofessional Teamwork Through Simulated Room of Errors Experience," *Journal of Surgical Research*, vol. 251, pp. 137–145, July 2020.

[11] E. Ingvarsdottir and S. Halldorsdottir, "Enhancing patient safety in the operating theatre: From the perspective of experienced operating theatre nurses," *Scandinavian Journal of Caring Sciences*, vol. 32, no. 2, pp. 951–960, 2018.

[12] M. Molloy and A. Clay, ""Room of Horrors": Engaging Interprofessional Students in a Hazards of Hospitalization Simulation," in *International Nursing Association for Clinical Simulation and Learning (INACSL) Conference 2017*, (Washington DC, USA), INACSL, 2017.

[13] W. M. Nehring and F. R. Lashley, "Nursing Simulation: A Review of the Past 40 Years," *Simulation & Gaming*, vol. 40, no. 4, pp. 528–552, 2009.

[14] J. Pottle, "Virtual reality and the transformation of medical education," *Future Healthcare Journal*, vol. 6, pp. 181–185, Oct. 2019.

[15] S. J. Hamstra, R. Brydges, R. Hatala, B. Zendejas, and D. A. Cook, "Reconsidering Fidelity in Simulation-Based Training," *Academic Medicine*, vol. 89, pp. 387–392, Mar. 2014.

[16] M. V. Olsen, K. Karlgren, M. W. Nortvedt, and A.-C. L. Leonardsen, "Prosjekt #2517425 - Uønskete hendelser - læring fra praksisfelt til utdanning - Cristin." https://app.cristin.no/projects/show.jsf?id=2517425, Jan. 2022. (accessed 2022-05-23).

[17] E. D. Matsumoto, S. J. Hamstra, S. B. Radomski, and M. D. Cusimano, "The effect of bench model fidelity on endourological skills: A randomized controlled study," *Journal of Urology*, vol. 167, pp. 1243–1247, Mar. 2002.

[18] J. K. Tun, G. Alinier, J. Tang, and R. L. Kneebone, "Redefining Simulation Fidelity for Healthcare Education," *Simulation & Gaming*, vol. 46, pp. 159–174, Apr. 2015.

[19] J. Radianti, T. A. Majchrzak, J. Fromm, and I. Wohlgenannt, "A systematic review of immersive virtual reality applications for higher education: Design elements, lessons learned, and research agenda," *Computers & Education*, vol. 147, p. 103778, Apr. 2020.

[20] J. Brooke, "SUS: A "quick and dirty'usability scale," in *Usability Evaluation In Industry* (P. W. Jordan, B. Thomas, B. A. Weerdmeester, and I. L. McClelland, eds.), pp. 189–194, London ; Bristol, Pa: Taylor & Francis, 1st edition ed., June 1996.

[21] Unity Technologies, "Unity Real-Time Development Platform — 3D, 2D VR & AR Engine." https://unity.com/, 2021. (accessed 2022-05-23).

[22] Meta, "Meta Quest 2: Våre mest avanserte, nye alt-i-ett-VR-briller — Oculus." https://www.oculus.com/quest-2/, 2022. (accessed 2022-05-23).

[23] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design Science in Information Systems Research," *MIS Quarterly*, vol. 28, no. 1, pp. 75–105, 2004.

[24] A. R. J. Mortensen and K. Nome, "Room of horror – VR for økt pasientsikkerhet." https://hdl.handle.net/11250/2669986, 2020. (accessed 2022-05-23).

[25] C. Indhumathi, W. Chen, and Y. Cai, "Multi-Modal VR for Medical Simulation," *International Journal of Virtual Reality*, vol. 8, pp. 1–7, Jan. 2009.

[26] Acadicus, "Virtual Simulation Center: Immersive Education and Training." https://acadicus.com/. (accessed 2022-05-23).

[27] Oxford Medical Simulation, "Oxford Medical Simulation - Virtual Reality Healthcare Training." https://oxfordmedicalsimulation.com/. (accessed 2022-05-23).

[28] SimX, "SimX Virtual Reality Medical Simulation." https://www.simxvr.com/, 2020. (accessed 2022-05-23).

[29] Osso VR, "Osso VR." https://www.ossovr.com/, 2021. (accessed 2022-05-23).

[30] Surgical Science, "LapSim® Essence." https://surgicalscience.com/simulators/lapsim/lapsim-essence/, 2021. (accessed 2022-05-23).

[31] F. W. Kron, M. D. Fetters, M. W. Scerbo, C. B. White, M. L. Lypson, M. A. Padilla, G. A. Gliva-McConvey, L. A. Belfore, T. West, A. M. Wallace, T. C. Guetterman, L. S. Schleicher, R. A. Kennedy, R. S. Mangrulkar, J. F. Cleary, S. C. Marsella, and D. M. Becker, "Using a computer simulation for teaching communication skills: A blinded multisite mixed methods randomized controlled trial," *Patient Education and Counseling*, vol. 100, pp. 748–759, Apr. 2017.

[32] Ubisim, "The Virtual Reality Training Platform for Nursing." https://www.ubisimvr.com/, 2022. (accessed 2022-05-23).

[33] M. D'Errico, "Immersive Virtual Reality as an International Collaborative Space for Innovative Simulation Design," *Clinical Simulation in Nursing*, vol. 54, pp. 30–34, 2021.

[34] C. L. Foronda, M. Fernandez-Burgos, C. Nadeau, C. N. Kelley, and M. N. Henry, "Virtual Simulation in Nursing Education: A Systematic Review Spanning 1996 to 2018," *Simulation in Healthcare*, vol. 15, pp. 46–54, Feb. 2020.

[35] C. Plotzky, U. Lindwedel, M. Sorber, B. Loessl, P. König, C. Kunze, C. Kugler, and M. Meng, "Virtual reality simulations in nurse education: A systematic mapping review," *Nurse Education Today*, vol. 101, p. 104868, June 2021.

[36] J.-C. Servotte, M. Goosse, S. H. Campbell, N. Dardenne, B. Pilote, I. L. Simoneau, M. Guillaume, I. Bragard, and A. Ghuysen, "Virtual Reality

Experience: Immersion, Sense of Presence, and Cybersickness," *Clinical Simulation in Nursing*, vol. 38, pp. 35–43, Jan. 2020.

[37] B. G. Witmer and M. J. Singer, "Measuring Presence in Virtual Environments: A Presence Questionnaire," *Presence: Teleoperators and Virtual Environments*, vol. 7, pp. 225–240, June 1998.

[38] J. Radianti, T. Majchrzak, J. Fromm, S. Stieglitz, and J. vom Brocke, "Virtual reality applications for higher educations: A market analysis," in *Proceedings of the Annual Hawaii International Conference on System Sciences*, vol. 2020-January, pp. 124–133, 2021.

[39] J. R. J. Neo, A. S. Won, and M. M. Shepley, "Designing Immersive Virtual Environments for Human Behavior Research," *Frontiers in Virtual Reality*, vol. 2, p. 5, 2021.

[40] C. Jennett, A. L. Cox, P. Cairns, S. Dhoparee, A. Epps, T. Tijs, and A. Walton, "Measuring and defining the experience of immersion in games," *International Journal of Human-Computer Studies*, vol. 66, pp. 641–661, Sept. 2008.

[41] J. LaMantia, "Virtual-reality simulations offer medical residents hands-on practice." https://www.modernhealthcare.com/article/20180820/NEWS/180829997/virtual-reality-simulations-offer-medical-residents-hands-on-practice, Aug. 2018. (accessed 2022-05-23).

[42] Jiayi, "VR - Room of Horror." https://vimeo.com/383892248, 2019. (accessed 2022-05-23).

[43] M.-S. Bracq, E. Michinov, M. Le Duff, B. Arnaldi, V. Gouranton, and P. Jannin, "Training situational awareness for scrub nurses: Error recognition in a virtual operating room," *Nurse Education in Practice*, vol. 53, p. 103056, May 2021.

[44] C. Zimmermann, A. Fridrich, and D. L. B. Schwappach, "Training Situational Awareness for Patient Safety in a Room of Horrors: An Evaluation of a Low-Fidelity Simulation Method," *Journal of Patient Safety*, vol. Publish Ahead of Print, Mar. 2021.

[45] P. R. Jeffries, "A FRAMEWORK for Designing, Implementing, and Evaluating: Simulations Used as Teaching Strategies in Nursing," *Nursing Education Perspectives*, vol. 26, pp. 96–103, Mar. 2005.

[46] I. H. Leirvåg, "Desse vann prisar for beste bachelorprosjekt." https://www.hvl.no/aktuelt/studentprisar/, Dec. 20. (accessed 2022-05-28).

[47] J. Han, *Introduction to Computer Graphics with OpenGL ES.* Milton, UNITED KINGDOM: Taylor & Francis Group, 2018.

[48] Mixall, "LOW POLY WORLD - HOSPITAL — 3D Fantasy — Unity Asset Store." https://assetstore.unity.com/packages/3d/environments/fantasy/low-poly-world-hospital-131688, Dec. 2020. (accessed 2022-05-28).

[49] Wikipedia, "Low poly." https://en.wikipedia.org/w/index.php?title=Low_poly&oldid=1053303321, Nov. 2021. (accessed 2021-11-24).

[50] medialab-ku, "Introduction to Computer Graphics with OpenGL ES." https://github.com/medialab-ku/openGLESbook, 2018. (accessed 2022-05-30).

[51] Wikipedia, "Graphics pipeline." https://en.wikipedia.org/w/index.php?title=Graphics_pipeline&oldid=1045798798, Sept. 2021. (accessed 2021-11-24).

[52] HTC, "Vive Shipment Updates." https://blog.vive.com/us/2016/04/07/vive-shipment-updates/, Apr. 2016. (accessed 2022-05-28).

[53] Oculus, "Introducing Oculus Quest, Our First 6DOF All-in-One VR System, Launching Spring 2019." https://www.oculus.com/blog/introducing-oculus-quest-our-first-6dof-all-in-one-vr-system-launching-spring-2019/, Sept. 2018. (accessed 2022-05-28).

[54] S. Rogers, "2019: The Year Virtual Reality Gets Real." https://www.forbes.com/sites/solrogers/2019/06/21/2019-the-year-virtual-reality-gets-real/, June 2019. (accessed 2022-05-28).

[55] B. D. Nagel, "Education to Help Drive VR Growth." https://thejournal.com/articles/2020/02/26/education-to-help-drive-vr-growth.aspx, Feb. 20. (accessed 2022-05-28).

[56] S. Weber, D. Weibel, and F. W. Mast, "How to Get There When You Are There Already? Defining Presence in Virtual Reality and the Importance of Perceived Realism," *Frontiers in Psychology*, vol. 12, p. 1538, 2021.

[57] P. V. F. Paiva, L. S. Machado, A. M. G. Valença, T. V. Batista, and R. M. Moraes, "SimCEC: A Collaborative VR-Based Simulator for Surgical Teamwork Education," *Computers in Entertainment*, vol. 16, pp. 3:1–3:26, Apr. 2018.

[58] P. Milgram and F. Kishino, "A Taxonomy of Mixed Reality Visual Displays," *IEICE Trans. Information Systems*, vol. vol. E77-D, no. 12, pp. 1321–1329, Dec. 1994.

[59] A. Suh and J. Prophet, "The state of immersive technology research: A literature analysis," *Computers in Human Behavior*, vol. 86, pp. 77–90, Sept. 2018.

[60] Wikipedia, "Extended reality." https://en.wikipedia.org/w/index.php?title=Extended_reality&oldid=1034629353, July 2021. (accessed 2021-10-15).

[61] Wikipedia, "Cave automatic virtual environment." https://en.wikipedia.org/w/index.php?title=Cave_automatic_virtual_environment&oldid=1035388554, July 2021. (accessed 2021-10-18).

[62] J. Bailenson, *Experience on Demand: What Virtual Reality Is, How It Works, and What It Can Do.* W. W. Norton & Company, Jan. 2018.

[63] Google Developers, "Degrees of freedom — Google VR — Google Developers." https://developers.google.com/vr/discover/degrees-of-freedom?hl=nb, Sept. 2018. (accessed 2022-05-28).

[64] J. J. Cummings and J. N. Bailenson, "How Immersive Is Enough? A Meta-Analysis of the Effect of Immersive Technology on User Presence," *Media Psychology*, vol. 19, pp. 272–309, Apr. 2016.

[65] Wikipedia, "Stereoscopy." https://en.wikipedia.org/w/index.php?title=Stereoscopy&oldid=1050523439, Oct. 2021. (accessed 2021-10-18).

[66] Wikipedia, "Oculus Go." https://en.wikipedia.org/w/index.php?title=Oculus_Go&oldid=1053291206, Nov. 2021. (accessed 2022-05-28).

[67] Google, "Google Cardboard – Google VR." https://arvr.google.com/cardboard/, 2021. (accessed 2022-05-28).

[68] Samsung, "Gear VR with Controller (SM-R325) – finn vårt tilbud — Samsung Norge." https://www.samsung.com/no/mobile-accessories/gear-vr-r325-sm-r325nzvcnee/, 2022. (accessed 2022-05-28).

[69] HTC, "VIVE - VR Headsets, Games, and Metaverse Life — United States." https://www.vive.com/us/, 2022. (accessed 2022-05-28).

[70] HP Development Company, LP, "HP Reverb G2 VR Headset." https://www.hp.com/us-en/vr/reverb-g2-vr-headset.html, 2022. (accessed 2022-05-28).

[71] Pico Immersive Pte. Ltd., "Pico - Official Website." https://www.picoxr.com/global/, 2022. (accessed 2022-05-28).

[72] Connell School of Nursing - Boston College, "A brief history of nursing simulation." https://www.bc.edu/bc-web/schools/cson/cson-news/Abriefhistoryofnursingsimulation.html, May 2015. (accessed 2022-05-23).

[73] M. Barland, "Teknologi for livslang læring - fjernt, nært og simulert." https://teknologiradet.no/publication/teknologi-for-livslang-laering-fjernt-naert-og-simulert/, Jan. 18. (accessed 2022-05-28).

[74] Monirb, "English: Students practice on high-fidelity simulation manikins that provide real-time and crisis feedback.." https://commons.wikimedia.org/wiki/File:PHOTOS_INSIDE_THE_CLASSROOM_UPDATED014.jpg, July 2016. (accessed 2022-05-31).

[75] COD Newsroom, "College of DuPage Hosts Multi-Discipline Health Simulation 140." https://www.flickr.com/photos/codnewsroom/14087068503/, Apr. 2014. (accessed 2022-05-31).

[76] COD Newsroom, "College of DuPage Hosts Multi-Discipline Health Simulation 126." https://www.flickr.com/photos/codnewsroom/14087076223/, Apr. 2014. (accessed 2022-05-31).

[77] COD Newsroom, "College of DuPage Hosts Multi-Discipline Health Simulation 125." https://www.flickr.com/photos/codnewsroom/14067494214/, Apr. 2014. (accessed 2022-05-31).

[78] M. Breivik, T. Johnsgaard, and M. H. Reime, "Simulering er ikke til å spøke med: Helsefagstudenters erfaringer fra tverrprofesjonell teamtrening med bruk av simuleringsmetodikk," *Nordisk tidsskrift for helseforskning*, vol. 12, Dec. 2016.

[79] T. Kaplan and J. Pilcher, "Evaluating Safety and Competency at the Bedside," *Journal for Nurses in Professional Development*, vol. 27, pp. 187–190, July 2011.

[80] M. Metroyanis, P. R. Webster, and R. Panzer, "Medical Mayhem Room: Enhancing Situational Awareness of Patient Safety Risks in the Hospital Setting." https://www.psqh.com/analysis/medical-mayhem-room-enhancing-situational-awareness-of-patient-safety-risks-in-the-hospital-setting/, 18.12.20. (accessed 2022-05-28).

[81] I. Shekhter, L. Rosen, J. Sanko, R. Everett-Thomas, M. Fitzpatrick, and D. Birnbach, "A patient safety course for preclinical medical students," *The Clinical Teacher*, vol. 9, no. 6, pp. 376–381, 2012.

[82] U. Shaikh, J. E. Natale, D. A. Till, and I. M. Julie, ""Good Catch, Kiddo"—Enhancing Patient Safety in the Pediatric Emergency Department Through Simulation," *Pediatric Emergency Care*, vol. 38, p. e283, Jan. 2022.

[83] P. Dieckmann, D. Gaba, and M. Rall, "Deepening the Theoretical Foundations of Patient Simulation as Social Practice," *Simulation in Healthcare*, vol. 2, no. 3, pp. 183–193, 2007.

[84] Wikipedia, "Urology." https://en.wikipedia.org/w/index.php?title=Urology&oldid=1078136751, Mar. 2022. (accessed 2022-05-03).

[85] C. Massoth, H. Röder, H. Ohlenburg, M. Hessler, A. Zarbock, D. M. Pöpping, and M. Wenk, "High-fidelity is not superior to low-fidelity simulation but leads to overconfidence in medical students," *BMC Medical Education*, vol. 19, p. 29, Jan. 2019.

[86] J. Kim, J.-H. Park, and S. Shin, "Effectiveness of simulation-based nursing education depending on fidelity: A meta-analysis," *BMC Medical Education*, vol. 16, p. 152, May 2016.

[87] S. Grassini and K. Laumann, "Questionnaire Measures and Physiological Correlates of Presence: A Systematic Review," *Frontiers in Psychology*, vol. 11, p. 349, 2020.

[88] G. Makransky and G. B. Petersen, "The Cognitive Affective Model of Immersive Learning (CAMIL): A Theoretical Research-Based Model of Learning in Immersive Virtual Reality," *Educational Psychology Review*, vol. 33, pp. 937–958, Sept. 2021.

[89] N. Winkler, K. Roethke, N. Siegfried, and A. Benlian, "Lose Yourself in VR: Exploring the Effects of Virtual Reality on Individuals' Immersion," *Hawaii International Conference on System Sciences 2020 (HICSS-53)*, Jan. 2020.

[90] J. Schell, *The Art of Game Design: A Book of Lenses*. Boca Raton, FL, USA: CRC Press, Taylor & Francis Group, third ed., 2019.

[91] C. S. Oh, J. N. Bailenson, and G. F. Welch, "A Systematic Review of Social Presence: Definition, Antecedents, and Implications," *Frontiers in Robotics and AI*, vol. 5, 2018.

[92] F. Biocca, "The Cyborg's Dilemma: Progressive Embodiment in Virtual Environments," *Journal of Computer-Mediated Communication*, vol. 3, p. JCMC324, Sept. 1997.

[93] T. Drey, P. Albus, d. K. Simon, M. Maximilian, S. Thilo, C. Linda, R. Michael, S. Tina, and R. Enrico, "Towards Collaborative Learning in Virtual Reality: A Comparison of Co-Located Symmetric and Asymmetric Pair-Learning," in *2022 ACM CHI Conference on Human Factors in Computing Systems*, (New Orleans, Louisiana, USA), p. 19, 2022.

[94] A. L. Simeone, M. Speicher, A. Molnar, A. Wilde, and F. Daiber, "LIVE: The Human Role in Learning in Immersive Virtual Environments," in *Symposium on Spatial User Interaction*, SUI '19, (New York, NY, USA), pp. 1–11, Association for Computing Machinery, Oct. 2019.

[95] J. Mütterlein, S. Jelsch, and T. Hess, "Specifics of Collaboration in Virtual Reality: How Immersion Drives the Intention to Collaborate," *PACIS 2018 Proceedings*, June 2018.

[96] B. Stokes, "Videogames have changed: Time to consider 'Serious Games'?," *Development Education Journal*, vol. 11, p. 12, Jan. 2005.

[97] M. Carrión, M. Santórum, M. Pérez, and J. Aguilar, "A participatory methodology for the design of serious games in the educational environment," in *2017 Congreso Internacional de Innovacion y Tendencias En Ingenieria (CONIITI)*, pp. 1–6, Oct. 2017.

[98] J. M. Koivisto, E. Haavisto, H. Niemi, P. Haho, S. Nylund, and J. Multisilta, "Design principles for simulation games for learning clinical reasoning: A design-based research approach," *Nurse Education Today*, vol. 60, pp. 114–120, Jan. 2018.

[99] M.-A. Maheu-Cadotte, S. Cossette, V. Dubé, G. Fontaine, M.-F. Deschênes, A. Lapierre, and P. Lavoie, "Differentiating the Design Principles of Virtual Simulations and Serious Games to Enhance Nurses' Clinical Reasoning," *Clinical Simulation in Nursing*, vol. 49, pp. 19–23, Dec. 2020.

[100] A. E. Olszewski and T. A. Wolbrink, "Serious Gaming in Medical Education: A Proposed Structured Framework for Game Development," *Simulation in Healthcare*, vol. 12, pp. 240–253, Aug. 2017.

[101] M. Toftedahl and H. Engström, "A taxonomy of game engines and the tools that drive the industry," in *DiGRA 2019, The 12th Digital Games Research Association Conference, Kyoto, Japan, August, 6-10, 2019*, Digital Games Research Association (DiGRA), 2019.

[102] F. Messaoudi, G. Simon, and A. Ksentini, "Dissecting games engines: The case of Unity3D," in *2015 International Workshop on Network and Systems Support for Games (NetGames)*, pp. 1–6, IEEE, 2015.

[103] Wikipedia, "List of game engines." https://en.wikipedia.org/w/index.php?title=List_of_game_engines&oldid=1055358250, Nov. 2021. (accessed 2021-11-15).

[104] Unreal Engine, "Unreal Engine 5." https://www.unrealengine.com/en-US/unreal-engine-5. (accessed 2022-05-28).

[105] L. Doucet, A. P. September 02, and 2021, "Game engines on Steam: The definitive breakdown." https://www.gamedeveloper.com/business/game-engines-on-steam-the-definitive-breakdown, Sept. 2021. (accessed 2022-05-28).

[106] E. Christopoulou and S. Xinogalos, "Overview and Comparative Analysis of Game Engines for Desktop and Mobile Devices," *International Journal of Serious Games*, vol. 4, Dec. 2017.

[107] D. Gajsek, "Unity vs Unreal Engine - Which is Better for Virtual and Augmented Development?." https://circuitstream.com/blog/unity-vs-unreal/, Mar. 2021. (accessed 2022-05-28).

[108] D. Tyler, "The 10 Best Video Game Engines — 2021 Edition." https://www.gamedesigning.org/career/video-game-engines/, Nov. 2021. (accessed 2022-05-28).

[109] A. Eldad, "Unity vs Unreal, What Kind of Game Dev Are You?." https://www.incredibuild.com/blog/unity-vs-unreal-what-kind-of-game-dev-are-you, Apr. 2021. (accessed 2022-05-28).

[110] D. Buckley, "Unity vs. Unreal – Choosing a Game Engine." https://gamedevacademy.org/unity-vs-unreal/, July 2021. (accessed 2022-05-28).

[111] E. D. da Costa, "Unity with MVC: How to Level Up Your Game Development." https://www.toptal.com/unity-unity3d/unity-with-mvc-how-to-level-up-your-game-development. (accessed 2022-05-28).

[112] R. Nystrom, "Component · Decoupling Patterns · Game Programming Patterns." http://gameprogrammingpatterns.com/component.html, 2021. (accessed 2022-05-28).

[113] M. Jordan, "Entities, components and systems." https://medium.com/ingeniouslysimple/entities-components-and-systems-89c31464240d, Nov. 2018. (accessed 2022-05-28).

[114] Unity Technologies, "Unity - Manual: Scenes." https://docs.unity3d.com/Manual/CreatingScenes.html, May 2022. (accessed 2022-05-13).

[115] Unity Technologies, "Unity - Scripting API: SceneManagement.SceneManager.LoadSceneAsync." https://docs.unity3d.com/ScriptReference/SceneManagement.SceneManager.LoadSceneAsync.html, May 2022. (accessed 2022-05-13).

[116] Unity Technologies, "Unity - Scripting API: MonoBehaviour." https://docs.unity3d.com/ScriptReference/MonoBehaviour.html, Nov. 2021. (accessed 2021-11-15).

[117] Unity Technologies, "Unity - Scripting API: ScriptableObject." https://docs.unity3d.com/ScriptReference/ScriptableObject.html, Nov. 2021. (accessed 2021-11-15).

[118] J. vom Brocke, R. Winter, A. Hevner, and A. Maedche, "Special Issue Editorial – Accumulation and Evolution of Design Knowledge in Design

Science Research: A Journey Through Time and Space," *Journal of the Association for Information Systems*, vol. 21, pp. 520–544, May 2020.

[119] E. Engström, M.-A. Storey, P. Runeson, M. Höst, and M. T. Baldassarre, "How software engineering research aligns with design science: A review," *Empirical Software Engineering*, vol. 25, pp. 2630–2660, July 2020.

[120] A. R. Hevner, "A Three Cycle View of Design Science Research," *Scandinavian Journal of Information Systems*, vol. 19, no. 2, p. 7, 2007.

[121] K. Conboy, R. Gleasure, and E. Cullina, "Agile design science research," in *International Conference on Design Science Research in Information Systems*, (Dublin, Ireland), pp. 168–180, Springer, 2015.

[122] Helsetilsynet, "Avgjørelser i enkeltsaker om svikt – søkeside." https://www.helsetilsynet.no/tilsyn/tilsynssaker/, 2021. (accessed 2022-05-28).

[123] Agile Manifesto Authors, "Manifesto for Agile Software Development." https://agilemanifesto.org/, 2001. (accessed 2022-05-28).

[124] Agile Alliance, "12 Principles Behind the Agile Manifesto — Agile Alliance." https://www.agilealliance.org/agile101/12-principles-behind-the-agile-manifesto/, Nov. 2015. (accessed 2022-05-28).

[125] Microsoft, "Visual Studio: IDE and Code Editor for Software Developers and Teams." https://visualstudio.microsoft.com, 2022. (accessed 2022-05-09).

[126] Blender Foundation, "Blender.org - Home of the Blender project - Free and Open 3D Creation Software."

[127] dotPDN LLC and R. Brewster, "Paint.NET - Free Software for Digital Photo Editing." https://www.getpaint.net/. (accessed 2022-05-09).

[128] The GIMP Team, "GIMP." https://www.gimp.org/. (accessed 2022-05-09).

[129] Audacity, "Audacity." https://www.audacityteam.org. (accessed 2022-05-09).

[130] Atlassian, "Trello." https://trello.com/, 2022.

[131] Unity Technologies, "Unity - Manual: Unity Collaborate." https://docs.unity3d.com/Manual/UnityCollaborate.html, May 2022. (accessed 2022-05-09).

[132] Unity Technologies, "Plastic SCM Cloud Edition — Unity Version Control." https://unity.com/products/plastic-scm, 2022. (accessed 2022-05-09).

[133] W. M. Felton and R. E. Jackson, "Presence: A Review," *International Journal of Human–Computer Interaction*, vol. 38, pp. 1–18, Jan. 2022.

[134] F. D. Davis, "Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology," *MIS Quarterly*, vol. 13, no. 3, pp. 319–340, 1989.

[135] V. Venkatesh, M. G. Morris, G. B. Davis, and F. D. Davis, “User Acceptance of Information Technology: Toward a Unified View,” *MIS Quarterly*, vol. 27, no. 3, pp. 425–478, 2003.

[136] A. Bangor, P. T. Kortum, and J. T. Miller, “An Empirical Evaluation of the System Usability Scale,” *International Journal of Human-Computer Interaction*, vol. 24, pp. 574–594, July 2008.

[137] E. Toftøy-Andersen and J. G. Wold, *Praktisk Brukertesting*. Oslo: Cappelen Damm akademisk, 2011.

[138] M. R. Mine, “Virtual environment interaction techniques,” *UNC Chapel Hill CS Dept*, 1995.

[139] R. Kopper, F. Bacim, and D. A. Bowman, “Rapid and accurate 3D selection by progressive refinement,” in *2011 IEEE Symposium on 3D User Interfaces (3DUI)*, pp. 67–74, Mar. 2011.

[140] P. Monteiro, G. Gonçalves, H. Coelho, M. Melo, and M. Bessa, “Hands-free interaction in immersive virtual reality: A systematic review.,” *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 5, pp. 2702–2713, 2021.

[141] H. Tu, S. Huang, J. Yuan, X. Ren, and F. Tian, “Crossing-Based Selection with Virtual Reality Head-Mounted Displays,” in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pp. 1–14, New York, NY, USA: Association for Computing Machinery, May 2019.

[142] D. Bowman, C. Wingrave, J. Campbell, and V. Ly, “Using Pinch Gloves(TM) for both Natural and Abstract Interaction Techniques in Virtual Environments.” https://eprints.cs.vt.edu/archive/00000547/, May 2002. (accessed 2022-05-28).

[143] C. Nolet, “Quick Outline — Particles/Effects — Unity Asset Store.” https://assetstore.unity.com/packages/tools/particles-effects/quick-outline-115488, June 2018. (accessed 2022-05-28).

[144] Kronnect, “Highlight Plus — Particles/Effects — Unity Asset Store.” https://assetstore.unity.com/packages/tools/particles-effects/highlight-plus-134149, Jan. 2022. (accessed 2022-05-28).

[145] Unity Technologies, “Unity - Manual: Post-processing.” https://docs.unity3d.com/Manual/PostProcessingOverview.html, Jan. 2022. (accessed 2022-01-17).

[146] T. Dasch, “PC Rendering Techniques to Avoid when Developing for Mobile VR.” https://developer.oculus.com/blog/pc-rendering-techniques-to-avoid-when-developing-for-mobile-vr/, Nov. 2019. (accessed 2022-05-28).

[147] Albazcythe, “VIDE Dialogues — AI — Unity Asset Store.” https://assetstore.unity.com/packages/tools/ai/vide-dialogues-69932, Nov. 2018. (accessed 2022-05-13).

[148] ttsMP3.com, “Free Text-To-Speech for US English language and MP3 Download — ttsMP3.com.” https://ttsmp3.com/, 2022. (accessed 2022-04-11).

[149] 2887679652, "Ding_dong(Remix of 110165).wav by 2887679652." https://freesound.org/people/2887679652/sounds/171755/, 2012. (accessed 2022-04-11).

[150] michael_grinnell, "Clock_Tick_Tock_Loop.wav by michael_grinnell." https://freesound.org/people/michael_grinnell/sounds/464402/, 2019. (accessed 2022-04-11).

[151] FunWithSound, "Short Success Sound Glockenspiel Treasure Video Game.mp3 by FunWithSound." https://freesound.org/people/FunWithSound/sounds/456965/, 2019. (accessed 2022-04-11).

[152] Facebook Technologies, "Vision — Oculus Developers." https://developer.oculus.com/resources/bp-vision/. (accessed 2022-05-28).

[153] Braun, "Discofix® C 3-way Stopcock with Connection Tubing." https://www.bbraun.co.uk/en/products/b/discofix-c-dreiwegehaehnemverbindung.html. (accessed 2022-04-27).

[154] Facebook Technologies, "Set Up Hand Tracking — Oculus Developers." https://developer.oculus.com/documentation/unity/unity-handtracking/. (accessed 2022-05-10).

[155] UKER, "English: Second generation Oculus Touch controllers, as included with the Oculus Rift S and Oculus Quest.." https://commons.wikimedia.org/wiki/File:Second_generation_Oculus_Touch_controllers.jpg, July 2019. (accessed 2022-05-31).

[156] Facebook Technologies, "VR Locomotion Design Guide — Oculus Developers." https://developer.oculus.com/resources/bp-locomotion/. (accessed 2022-05-28).

[157] Facebook Technologies, "Locomotion Best Practices — Oculus Developers." https://developer.oculus.com/resources/locomotion-design-techniques-best-practices/. (accessed 2022-05-28).

[158] Facebook Technologies, "Comfort and Usability — Oculus Developers." https://developer.oculus.com/resources/locomotion-comfort-usability/. (accessed 2022-05-28).

[159] Unity Technologies, "XR Interaction Toolkit — XR Interaction Toolkit — 2.0.0-pre.6." https://docs.unity3d.com/Packages/com.unity.xr.interaction.toolkit@2.0/manual/index.html, Dec. 2021. (accessed 2022-05-28).

[160] Facebook Technologies, "Locomotion Turns and Teleportation — Oculus Developers." https://developer.oculus.com/resources/locomotion-design-turns-teleportation/. (accessed 2022-03-15).

[161] Oculus VR, "Tech Note: Touch Button Mapping Best Practices." https://developer.oculus.com/blog/tech-note-touch-button-mapping-best-practices, Nov. 2017. (accessed 2022-04-27).

[162] Unity Technologies, "3 - VR Optimization and Lighting." https://learn.unity.com/project/vr-optimization-lighting, Apr. 2021. (accessed 2022-05-03).

[163] IEEE Computer Society, "IEEE Standard for Head-Mounted Display (HMD)-Based Virtual Reality(VR) Sickness Reduction Technology," *IEEE Std 3079-2020*, pp. 1–74, Apr. 2021.

[164] Exit Games, "PUN 2 - FREE — Network — Unity Asset Store." https://assetstore.unity.com/packages/tools/network/pun-2-free-119922, 2021. (accessed 2022-05-28).

[165] Valem, "How to Make a VR Multiplayer Game - PART 1." https://www.youtube.com/watch?v=KHWuTBmT1oI, Sept. 2020. (accessed 2022-05-28).

[166] Photon, "Setting the Benchmark for Multiplayer games. — Photon Engine." https://www.photonengine.com/en-US/Fusion. (accessed 2022-05-27).

[167] Unity Technologies, "Unity - Scripting API: Resources.FindObjectsOfTypeAll." https://docs.unity3d.com/ScriptReference/Resources.FindObjectsOfTypeAll.html, Apr. 2022. (accessed 2022-04-11).

[168] Oculus, "Oculus Integration — Integration — Unity Asset Store." https://assetstore.unity.com/packages/tools/integration/oculus-integration-8 2022, Nov. 2019. (accessed 2022-05-28).

[169] VRTK, "VRTK - Virtual Reality Toolkit." https://vrtoolkit.readme.io/, 2022. (accessed 2022-05-28).

[170] Pico Interactive, "SDK - Pico Developer Platform." https://developer.pico-interactive.com/sdk. (accessed 2022-05-28).

[171] The Khronos Group, "OpenXR - High-performance access to AR and VR —collectively known as XR— platforms and devices." https://www.khronos.org/openxr/, 2022. (accessed 2022-05-28).

[172] Unity Technologies, "Namespace UnityEngine.XR.Interaction.Toolkit — XR Interaction Toolkit — 2.0.0-pre.6." https://docs.unity3d.com/Packages/com.unity.xr.interaction.toolkit@2.0/api/UnityEngine.XR.Interaction.Toolkit.html, Dec. 2021. (accessed 2022-01-25).

[173] Unity Technologies, "Create with VR." https://learn.unity.com/course/create-with-vr, Nov. 2021. (accessed 2022-05-28).

[174] Unity Technologies, "VR Beginner: The Escape Room." https://learn.unity.com/project/vr-beginner-the-escape-room, Oct. 2021. (accessed 2022-05-28).

[175] Facebook Technologies, "Performance and Optimization — Oculus Developers." https://developer.oculus.com/documentation/unity/unity-perf/. (accessed 2022-05-28).

[176] Facebook Technologies, "Best Practices for Rift and Android — Oculus Developers." https://developer.oculus.com/documentation/unity/unity-best-practices-intro/. (accessed 2022-05-28).

[177] Facebook Technologies, "Basic Optimization Workflow for Oculus Quest Apps: Unity — Oculus Developers." https://developer.oculus.com/documentation/unity/po-perf-opt-mobile/. (accessed 2022-05-05).

[178] Unity Technologies, "Unity - Manual: Introduction to collision." https://docs.unity3d.com/Manual/CollidersOverview.html, Apr. 2022. (accessed 2022-05-05).

[179] Facebook Technologies, "Using Single Pass Stereo Rendering and Stereo Instancing — Oculus Developers." https://developer.oculus.com/documentation/unity/unity-single-pass/. (accessed 2022-05-05).

[180] Unity Technologies, "Unity - Manual: Single Pass Stereo rendering (Double-Wide rendering)." https://docs.unity3d.com/Manual/SinglePassStereoRendering.html, Aug. 2018. (accessed 2022-05-05).

[181] Unity Technologies, "Unity - Manual: Optimizing draw calls." https://docs.unity3d.com/2022.1/Documentation/Manual/optimizing-draw-calls.html, May 2022. (accessed 2022-05-18).

[182] Facebook Technologies, "Art Direction for All-in-One VR Performance: Unity — Oculus Developers." https://developer.oculus.com/documentation/unity/po-art-direction/. (accessed 2022-05-05).

[183] Adobe Systems Incorporated, "Mixamo." https://www.mixamo.com/#/, 2022. (accessed 2022-05-09).

[184] Blender, "Clean Up — Blender Manual." https://docs.blender.org/manual/en/2.93/modeling/meshes/editing/mesh/cleanup.html, Apr. 2022. (accessed 2022-05-04).

[185] A. H. Pripp, "Hvorfor p-verdien er signifikant," *Tidsskrift for Den norske legeforening*, Sept. 2015.

[186] A. H. Pripp, "Nyanser av variasjon," *Tidsskrift for Den norske legeforening*, Jan. 2018.

[187] J. Frost, "Degrees of Freedom in Statistics." http://statisticsbyjim.com/hypothesis-testing/degrees-freedom-statistics/, Sept. 2017. (accessed 2022-05-23).

[188] J. Mütterlein, "The Three Pillars of Virtual Reality? Investigating the Roles of Immersion, Presence, and Interactivity," in *51st Hawaii International Conference on System Sciences*, (Hawaii, USA), Jan. 2018.

[189] T. Tullis and J. Stetson, "A Comparison of Questionnaires for Assessing Website Usability," *Usability Professionals Association Annual conference 2004*, vol. 1, pp. 1–12, June 2006.

[190] J. R. Lewis, "The System Usability Scale: Past, Present, and Future," *International Journal of Human–Computer Interaction*, vol. 34, pp. 577–590, July 2018.

[191] Open Broadcaster Software, "Open Broadcaster Software — OBS." https://obsproject.com/. (accessed 2022-05-06).

[192] H. Berg and A. Steinsbekk, "The effect of self-practicing systematic clinical observations in a multiplayer, immersive, interactive virtual reality application versus physical equipment: A randomized controlled trial," *Advances in Health Sciences Education*, vol. 26, pp. 667–682, May 2021.

[193] K. Nome, M. V. Olsen, H. Soleim, A. Geitung, T. Johnsgaard, and K. Karlgren, "Usability of a VR Simulation Training Concept with Intentional Simulation Errors," in *1st IEEE International Conference on Cognitive Aspects of Virtual Reality*, (Online), IEEE, May 2022. (Note: not yet published at the time of submission).

Appendix A

Source Code

Several files have been submitted as attachments to this document. This includes:

- Zip file containing the Unity project, including source code
- Zip file containing source code for the REST API
- APK file that can be run on Meta Quest HMDs

The Unity project contains 3D models and other assets that have been purchased. Due to licensing, these assets cannot be publicly shared. However, the Unity project can be viewed on a private GitHub repository:

https://github.com/krnome/room-of-errors

Access is provided to the evaluation committee upon request by providing a project supervisor with an e-mail address or GitHub username.

During development, Unity Collaborate and later Plastic SCM has been used for version control. Unfortunately, moving the project to GitHub did not include moving version and commit history. However, the full history can be seen in the project's Plastic SCM repository, to which access is also provided to the evaluation committee upon request.

Appendix B

The System Usability Scale

	Sterkt uenig				Sterkt enig
1. Jeg kunne tenke meg å bruke denne simuleringen ofte.	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
2. Jeg synes simuleringen var unødvendig komplisert	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
3. Jeg synes simuleringen var lett å bruke	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
4. Jeg tror jeg måtte trenge hjelp fra en person med teknisk kunnskap for å kunne bruke denne simuleringen.	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
5. Jeg synes at de forskjellige delene av simuleringen hang godt sammen.	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
6. Jeg synes det var for mye inkonsistens i simuleringen. (Det virket "ulogisk")	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
7. Jeg vil anta at folk flest kan lære seg denne simuleringen veldig raskt.	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
8. Jeg synes simuleringen var veldig vanskelig å bruke.	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
9. Jeg følte meg sikker da jeg brukte simuleringen.	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
10. Jeg trenger å lære meg mye før jeg kan komme i gang med å bruke denne simuleringen på egen hånd	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5

Appendix C

Interview Questions and Topics

- Er dette noe du ser for deg du kan bruke som en del av et undervisningsopplegg?
  - Hvis ikke, hva kan forandres på?
- Hvordan synes du det var å lete etter feil?
  - Var feilene tydelige nok eller for vanskelige å oppdage?
- Er det noe du ville forandret i simuleringen? Er det noe du savner?
- (Hvis flere deltakere) Synes du det var nyttig å være to personer?

Added topics for the second round of evaluations

- (Hvis andre gang) Synes du det var nyttig å gjøre simuleringen en gang til?
- ”Kjøpte du” at du var i en operasjonsstue/postoperativ avdeling?

Appendix D

List of intended errors

D.1 Intended errors in the operating room

- Hand disinfectant is empty on the OR nurse side of the room
- Hand disinfectant is empty on the nurse anesthetist side of the room
- Gauze sponges without X-ray thread was used
- The wrong patient is on the table
- Wrong medication in a syringe pump
- The intubation table is missing an extra blade for the laryngoscope
- Missing pillows under the patients arms
- A needle is missing after the operation
- Gauze sponge is missing after the operation
- Catheter bag is on the floor
- Patient warming has not been started
- Soiled gloves in the sterile field
- Blood transfusion has not been controlled twice
- Patient has tachycardia
- Patient is receiving Penicillin while allergic
- Nurse anesthetist is wearing a necklace and a ring
- Extra blood transfusion bag is intended for a different patient
- Hole in the sterile packaging for an extra instrument
- Missing surgical instrument after the operation
- Laerdalsbag bag has not been prepared
- Intubation tube has not been fixed in place

D.2 Intended errors in the Postoperative ward

- Hand disinfectant by the door is empty
- Hand disinfectant by bed 1 is empty
- Hand disinfectant by bed 4 is empty
- The intubation table is missing an extra intubation tube
- Oral airway is missing from bed 3
- Laerdalsbag has not been prepared
- Suction has not been connected
- Loose cap in bed 4
- Bed 2: PVC should be changed
- Bed 2: Respirator set to NIV mode
- Nurse anesthetist is wearing a necklace and a ring
- Bed 4: Missing name band
- Bed 4: Bed is too flat
- Intubation tubes have not been fixed in place
- Bed 4: Catheter is closed
- Bed 2: Nasogastric tube is on the floor
- Bed 3: Missing cap on central venous catheter line
- Bed 3: Rapifen and Propofol on the same line
- Bed 2: Transducer too high
- Bed 3: Pressure infusor lying in the bed

Appendix E

Assets

Videos used for patient monitors have been generated using the Pulse Physiology Engine asset: https://assetstore.unity.com/packages/tools/integration/pulse-physiology-engine-139773#description

Textures used for respirator screens have been generated by screen dumps on a Dräger Infinity C500 respirator: https://www.draeger.com/no_no/Products/Evita-Infinity-V500-ventilator

Name	Creator	Price	URL
Blood Bag	Reberu Game Studio	$5	https://assetstore.unity.com/packages/3d/props/blood-bag-114334
Hospital Ward	studio lab	$79	https://assetstore.unity.com/packages/3d/environments/industrial/hospital-ward-80735
LOW POLY WORLD - Hospital	Mixall	$29.99	https://assetstore.unity.com/packages/3d/environments/fantasy/low-poly-world-hospital-131688
Operating Room	Coded Illusions	$29.99	https://assetstore.unity.com/packages/3d/props/interior/operating-room-18295
Photon Unity Network 2 - Free	Exit Games	Free	https://assetstore.unity.com/packages/tools/network/pun-2-free-119922
Real-Time Procedural Cable Simple	DrinkingWindGames	Free	https://assetstore.unity.com/packages/tools/particles-effects/real-time-procedural-cable-simple-102196
VIDE Dialogues	Albazcythe	Free	https://assetstore.unity.com/packages/tools/ai/vide-dialogues-69932

Table E.1: Unity Assets

Name	Creator	Price	URL
Surgical Instruments - Medical Equipment Collection	AssetKit	$41	https://www.turbosquid.com/3d-models/3d-surgical-instruments---medical-equipment-model/1072101
3D Anaesthesia Medical Equipment (2) model	3dartmasterwork	$40	https://www.turbosquid.com/3d-models/3d-anaesthesia-medical-equipment-2-model-1213055
Anesthesia Face Mask	3_molier International	$19	https://www.turbosquid.com/3d-models/3d-model-anesthesia-face-mask/907903
Laryngoscope	Glucius	$20	https://www.turbosquid.com/3d-models/3d-model-laryngoscope-scope/763042

Table E.2: Other purchased 3D models

Name	Creator	Price	URL
Ding dong(Remix of 110165)	288767965	Free	https://freesound.org/people/2887679652/sounds/171755/
Clock_Tick_Tock_Loop	michael_grinnell	Free	https://freesound.org/people/michael_grinnell/sounds/464402/
Short Success Sound Glockenspiel Treasure Video Game	FunWithSound	Free	https://freesound.org/people/FunWithSound/sounds/456965/

Table E.3: Sound effects

Appendix F

3D Models Created for the Room of Errors

F.1 Models created by Adrian René Johnsen Mortensen for the BSc prototype

Figure F.1: Cabinet in the OR

Figure F.2: Laerdalsbag in the OR

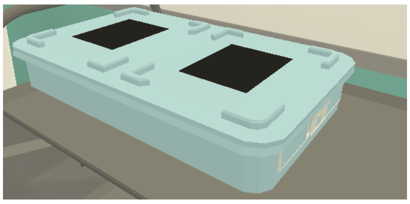

Figure F.3: Box containing surgical instruments

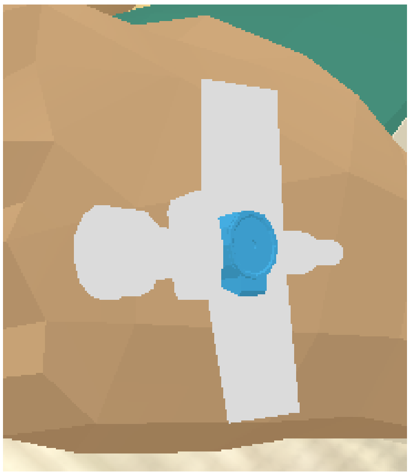

Figure F.4: Peripheral Venous Catheter (PVC)

Figure F.5: Oral airway tube

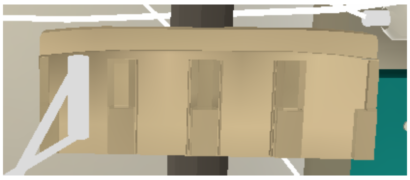

Figure F.6: Transducer

F.2 Models created by the author for the BSc prototype

Figure F.7: Patient nametag

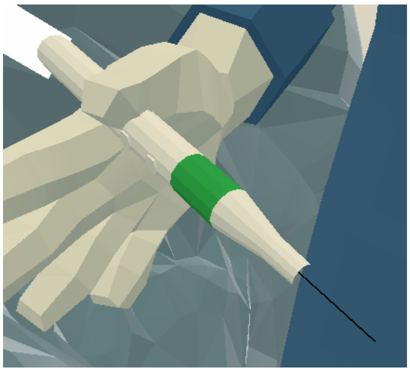

Figure F.8: Electrosurgical (ESU) knife

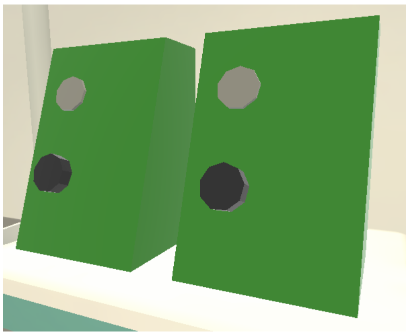

Figure F.9: Surgical suction

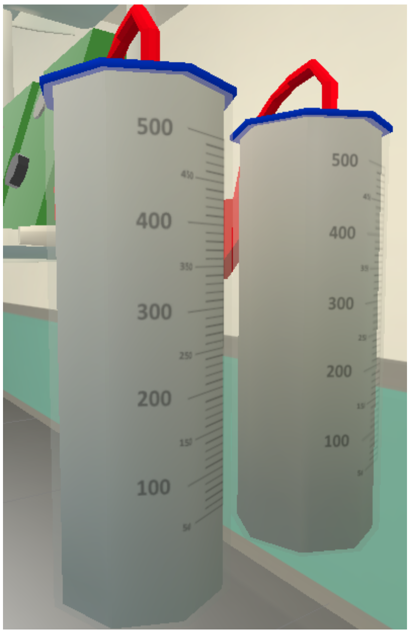

Figure F.10: Surgical suction canister

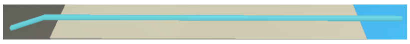

Figure F.11: Bougie for intubation

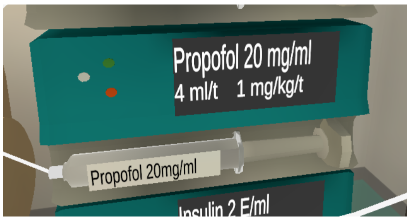

Figure F.12: Syringe pump (note: Syringe model by Coded Illusions

F.3 Models created by the author for the Master's project

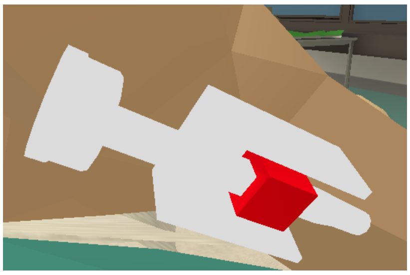

Figure F.13: Arterial catheter (Note: Modified version of Adrian's PVC model

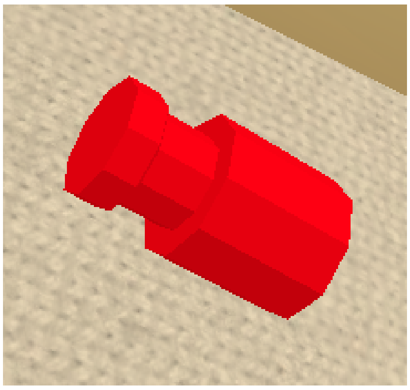

Figure F.14: "Combi-stopper", a cap for venous catheters

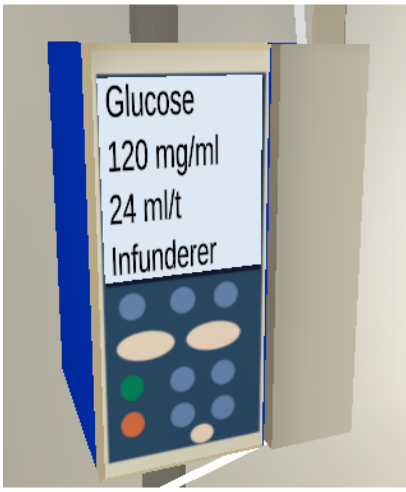

Figure F.15: Infusion pump

Figure F.16: Hand disinfectant container

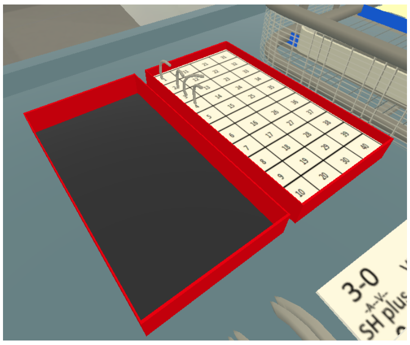

Figure F.17: Needle counter

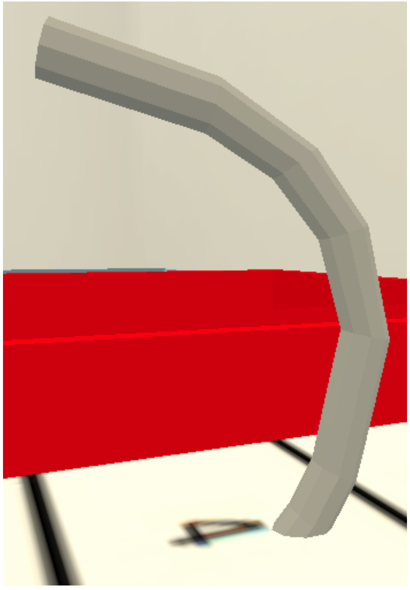

Figure F.18: Suture needle

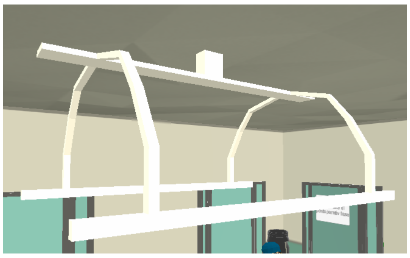

Figure F.19: Patient lifter

Figure F.20: Sterile packaging with hole

Figure F.21: Stopcock for venous catheters

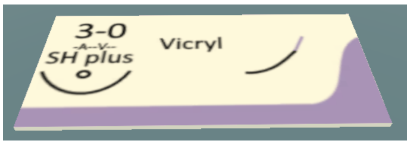

Figure F.22: Suture pack

Figure F.23: Electrosurgical (ESU) machine

Figure F.24: Smoke suction machine for the ESU machine

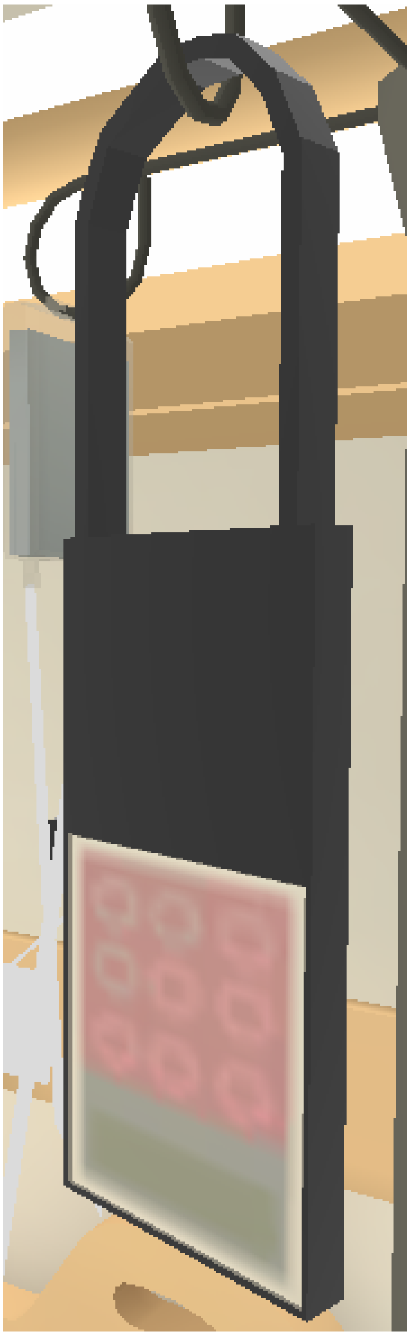

Figure F.25: CADD pump

Figure F.26: Intravenous fluid container

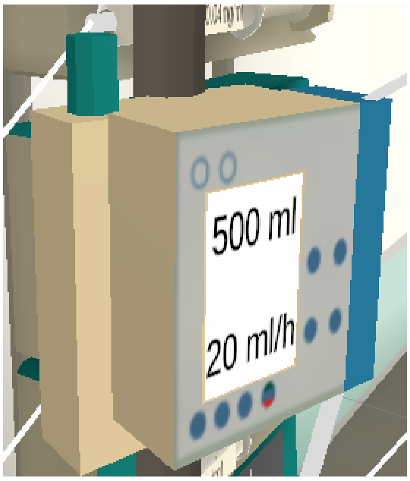

Figure F.27: Nasogastric feeding pump

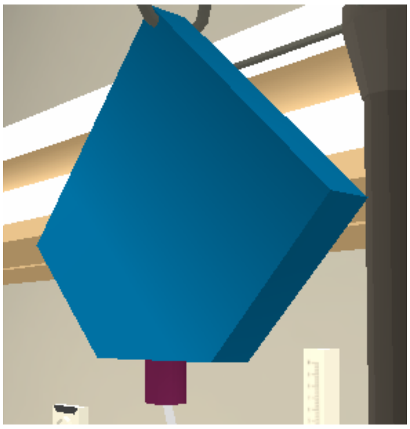

Figure F.28: Nasogastric feeding bag

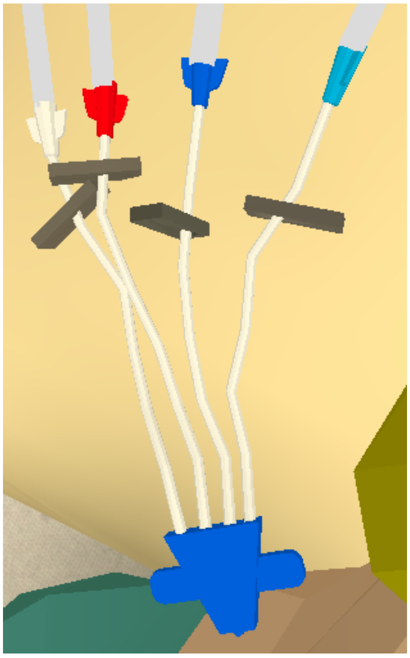

Figure F.29: Central venous catheter (CVC)

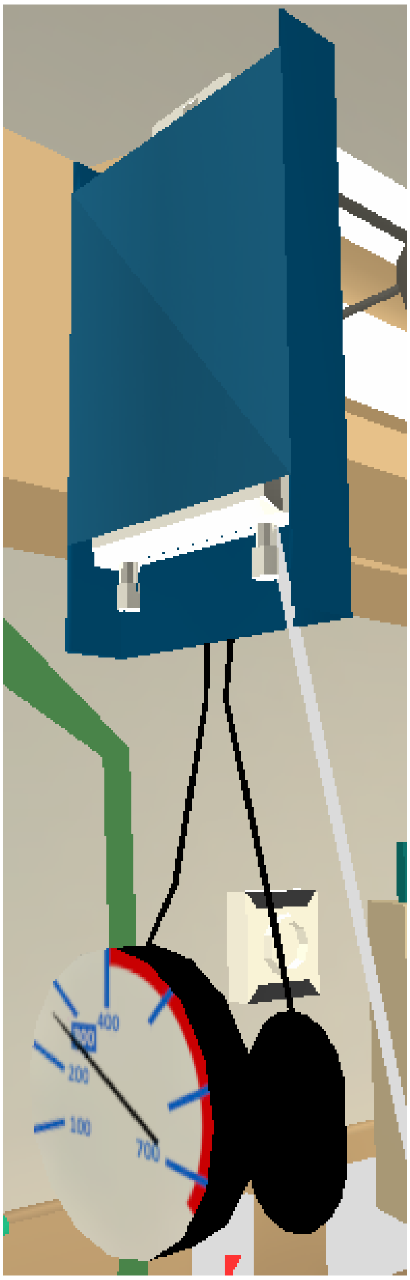

Figure F.30: Pressure infusor

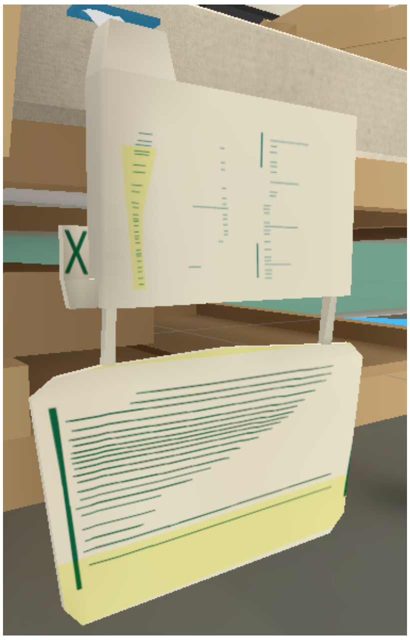

Figure F.31: Container for measuring diuresis

Figure F.32: Vacuum drain

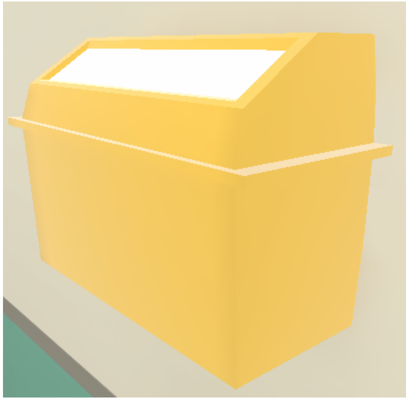

Figure F.33: Sharps container

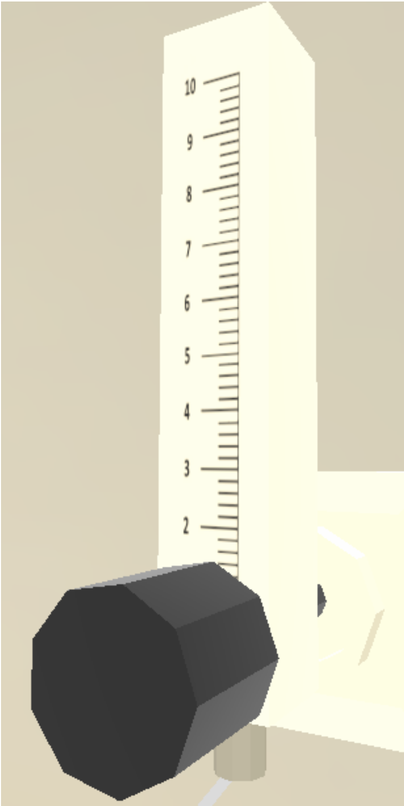

Figure F.34: Oxygen flowmeter

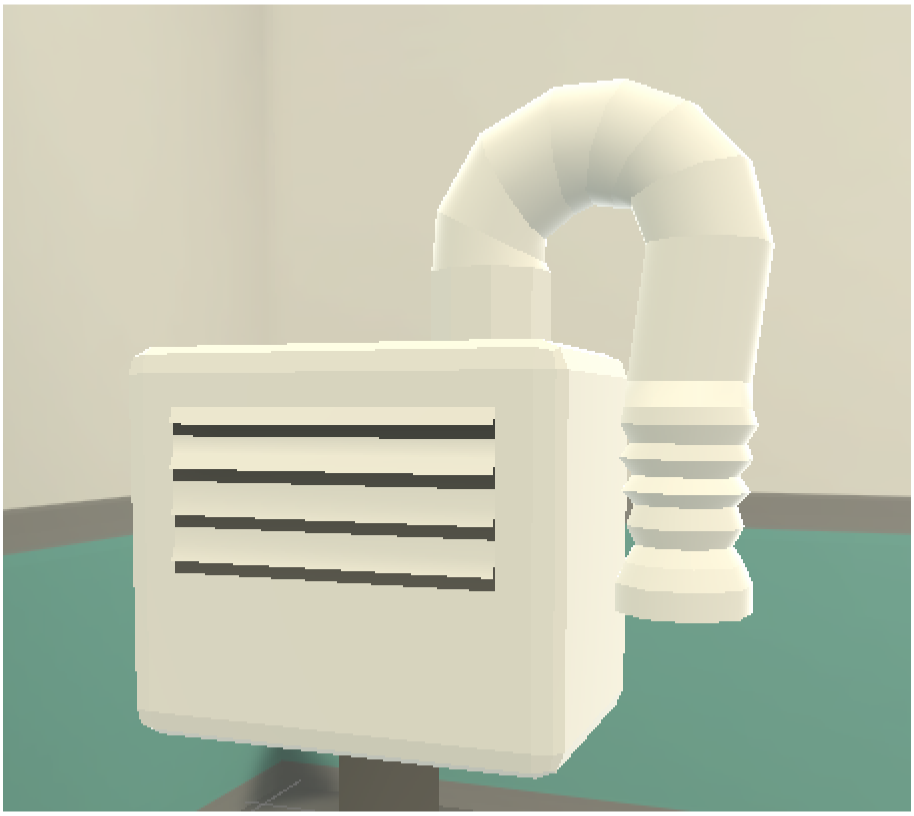

Figure F.35: Patient warmer