HAW HAMBURG

Masterarbeit

XcapeRoom: Cross Reality Interactions in a hybrid Escape-Room Environment

Eingereicht am: 12.09.2022

von: Jörg Peter | [redacted]

Matrikelnr.: [redacted]

Erstprüferin: Dipl. Des., M.Sc. Anke von der Heide

Zweitprüfer: Prof. Dr. Ing. Roland Greule

HAW Hamburg

Fakultät Design, Medien und Information

Department Medientechnik

Studiengang: Zeitabhängige Medien – Sound-Vision

Thema der Arbeit

Können Escape Rooms von XR-Technologie und Remote Collaboration profitieren? Um diese Frage zu beantworten wird der Prototyp eines XR-Escape-Rooms gebaut und in Interviews mit Probanden ausgewertet.

Stichworte

Virtual Reality – VR, Mixed Reality – MR, Cross Reality – XR, Remote Collaboration, Immersion, Präsenz, Soziale Präsenz, LIDAR, Pointcloud, Escape Room, Unity, Avatar, Tracking, Game, Challenges, Puzzle, Rätsel, Asymmetrisches Gameplay

Kurzzusammenfassung

Escape Rooms und Virtual Reality sind moderne und populäre Themen in der Unterhaltungsindustrie. Obwohl es Escape Rooms auch als virtuelle Erfahrungen gibt, fehlen Untersuchungen, wie die Faszination von stationären Escape Rooms mit der Remote-Fähigkeit von Virtual Reality kombiniert werden können. In dieser Arbeit wird der Prototyp eines Escape Rooms mit Cross-Reality-Interaktionen entwickelt. Der Prototyp besteht aus zwei Räumen, einem real, einem virtuell, bei denen durch Avatare die Spielenden aus dem jeweils anderen Raum sichtbar gemacht werden – eine Pointcloud für die Personen im realen, ein auf den Boden projizierter Lichtpunkt für die Person in VR. Das Ziel ist die Untersuchung der Frage, ob diese Darstellung geeignet ist, eine befriedigende Escape-Room-Erfahrung für eine örtlich verstreute Gruppe zu bieten.

Für den theoretischen Hintergrund werden die Begriffe Virtual Reality, Cross Reality und Escape Room kurz beleuchtet. Es erfolgt eine theoretische Betrachtung zum Escape Room Design und eine ausführliche Beschreibung des Prototypen und seiner Bestandteile. Die atomare Taxonomie von Lekhov et al. wird genutzt, um die Rätsel des Raums zu analysieren. Desweiteren werden Interviews mit insgesamt 18 Proband*innen herangezogen, die den Escape Room absolvierten.

Die Auswertung der Avatardarstellung fällt positiv aus und die verwendete Technik ist geeignet ein Gefühl sozialer Präsenz zu erschaffen. Zum Schluss der Arbeit werden Vorschläge präsentiert, wie zukünftige Iterationen eines XR-Escape-Rooms verbessert werden könnten und mögliche Anwendungen der Technik vorgebracht.

Subject of Thesis

Can escape rooms profit from XR-Technology and remote collaboration? To answer this question a prototype of an XR escape room is constructed and evaluated through interviews with participants.

Keywords

Virtual Reality – VR, Mixed Reality – MR, Cross Reality – XR, Remote Collaboration, Immersion, presence, social presence, LIDAR, pointcloud, escape room, Unity, avatar, tracking, game, challenges, puzzle, asymmetric gameplay

Abstract

Escape rooms and virtual reality are modern and popular topics of discussion in the entertainment industry. Even though escape rooms also exist as virtual experiences, there is a lack of studies that explore how the fascination of a stationary escape room can be combined with the remote usability of virtual reality. In this project the prototype of an escape room with cross reality interactions is developed. The prototype consists of two rooms, one real, one virtual, in which the people from one are made visible to those in the other by means of avatars – a pointcloud for the people in the real space and a circle projected onto the floor for the person in the virtual space respectively. The aim is to study the question whether this representation is suitable to provide a satisfying escape room experience for a spatially distributed group.

To provide a theoretical background, the terms virtual reality, cross reality and escape room are briefly highlighted. A theoretical exploration of escape room design and an extensive description of the prototype is carried out. The atomic puzzle taxonomy of Krekhov et al is used to analyze the puzzles of the room. Furthermore, interviews of 18 participants of the escape room are consulted.

The evaluation of the avatar representation is positive, and the utilized technology is suitable to elicit a feeling of social presence. At the end of the thesis, suggestions are provided, how an XR escape room can be improved in future iterations and possible further applications of the technology are presented.

Inhaltsverzeichnis

1 Einleitung ... 1

1.1 Ziel der Arbeit ... 1

1.2 Methodik ... 1

1.3 Struktur der Arbeit ... 2

2 Theoretischer Rahmen ... 3

2.1 Escape Rooms ... 3

2.2 Virtual Reality ... 5

2.2.1 Mixed Reality ... 6

2.2.2 Cross-Reality-Interaktionen ... 6

3 Konzept und Durchführung ... 7

3.1 Gameplay Walkthrough ... 7

3.1.1 Walkthrough ... 7

3.2 Konzeption ... 11

3.2.1 High Level Framework von Krekhov et al. ... 11

3.2.2 Storytelling ... 13

3.3 Generelle Bemerkungen zur VR ... 14

3.3.1 Wahl des Headsets und physische Gestaltung des Spielbereichs ... 14

3.3.2 Wahl der Engine und des VR-Frameworks ... 16

3.3.3 Interaktionen ... 16

3.4 Generelle Bemerkung zum physischen Raum (P-Raum) ... 16

3.5 Lidarsensor und Pointcloud ... 17

3.6 Netzwerk ... 19

3.7 Lichtpunkt ... 21

3.8 Erster Prototyp ... 22

4 Die Rätsel ... 22

4.1 Rein Physisch ... 23

4.1.1 VR-Box-Rätsel ... 23

4.1.2 Lichtschalter-Rätsel ... 24

4.1.3 DNA-Scanner-Task ... 25

4.1.4 Digital-Clean-Up-Besen-Rätsel ... 25

4.1.5 PolRev-Rätsel ... 26

4.1.6 Tür-Rätsel ... 28

4.2 Rein Virtuell ... 28
4.2.1 VR-Lichtschalter-Task ... 29
4.2.2 Bewegbare-Wand-Task ... 29
4.2.3 Coderätsel ... 30
4.2.4 Partymodus-Schalter-Task ... 30
4.2.5 Passwort-Rätsel ... 31
4.3 Cross Reality ... 31
4.3.1 LED-Wand-Tresor ... 31
4.3.2 Pressure-Plates-Rätsel ... 33
4.3.3 Schlüsselrätsel (3D-Drucker) ... 34
4.3.4 Hochschulmodell-Rätsel ... 35
4.4 Rätselüberblick ... 36
5 Evaluation ... 37
5.1 Auswertung der Durchläufe ... 37
5.2 Optimierungsmöglichkeiten ... 42
5.3 Weitere Anwendungsmöglichkeiten und Ausblick ... 44
6 Fazit ... 45
Literaturverzeichnis ... 47
Anhang ... i

1 Einleitung

Escape Rooms und Virtual Reality (VR) erlebten in den letzten Jahren einen Aufschwung. Erstere blieben dabei bis zum Beginn der Coronapandemie in einem Aufwärtstrend der seit ihrer Erfindung andauerte,[1] letztere werden mit dem Anwachsen erschwinglicher Endverbraucher-Produktlinien von Firmen wie Meta immer verbreiteter [12]. Auch eine Verbindung der beiden Konzepte gab und gibt es mittlerweile schon, doch was bisher nicht als Gameplay-Element etabliert wurde, ist eine asymmetrische Kooperation zwischen Personen in der VR und Personen in der Realität, die zusammen den gleichen Escape Room bewältigen. Asymmetrisch bedeutet in diesem Fall, dass Fähigkeiten, Wissen und Aufgaben ungleich auf die Spielenden verteilt werden. Asymmetrisches Gameplay hat lange Tradition, sowohl in Escape Rooms als auch den spirituellen Vorgängern [22], die später in dieser Arbeit noch Erwähnung finden. Spiele wie Keep Talking and Nobody Explodes [27] nutzen bereits die Interaktion zwischen Personen in einer VR-Umgebung und solchen außerhalb, um asymmetrisches Gameplay zu erzeugen, indem die Person in der VR eine Bombe entschärfen muss, während eine Person in der Realität eine Anleitung mit allen nötigen Informationen dazu hat.

Die Forschung zu Remote Collaboration (also Kollaboration zwischen räumlich getrennten Personen), sozialer Präsenz und Cross Reality (XR)[2] offenbart interessante Möglichkeiten, durch die Nutzung von digitalen Repräsentationen der Mitspielenden (Avataren) eine spannende und einzigartige Escape Room Erfahrung zu kreieren. Gleichzeitig könnte ein solcher Escape Room andersherum der Erforschung von Remote Collaboration dienen.

1.1 Ziel der Arbeit

Ziel dieser Arbeit war es, einen Prototypen eines solchen Escape Rooms zu entwickeln und in der Auswertung Erfahrungen zu sammeln, wie die soziale Kopräsenz durch verschiedenartige Avatare angenommen wird. Dies ist dabei als Startpunkt für weitere Forschung zu verstehen. Daraus ergibt sich die Forschungsfrage: Ist eine soziale Kopräsenz in einer Mixed-Reality-Situation, erzeugt durch volumetrische Avatare auf der einen und durch einen projizierten Avatar auf der anderen Seite, geeignet, um eine befriedigende Escape-Room-Erfahrung für eine über mehrere Orte verteilte Gruppe zu erzeugen? Die Hypothese ist, dass die Spieler*innen die Avatare als Repräsentation ihres Teammitglieds annehmen und eine reibungslose Remote Collaboration entsteht.

1.2 Methodik

Unter Einbeziehung von Erkenntnissen aus der Forschung zur Remote Collaboration und vorherigen Erfahrungen mit Virtual- und Cross-Reality-Installationen wurde ein Escape Room (XcapeRoom) gebaut und untersucht, wie gut der Aspekt der Cross-Reality-Interaktion von Teilnehmer*innen angenommen wird. Der Aufbau

[1] Vgl. hierzu Daten aus den USA, die ein Wachstum bis 2019 aufzeigen [16] In den USA sind laut Daten von Vorhaus Advisors, hier zitiert nach Statista [33] Escape Rooms sogar die beliebteste immersive Freizeitaktivität unter 18-34-Jährigen noch vor Museen und Live-Musik. Nicholson [2015], dessen Untersuchung von Escape Rooms auf der ganzen Welt später noch referenziert wird, spricht von einem schnellen weltweiten Wachstum, ausgehend von Asien [22].

[2] Cross Reality bezeichnet hier die Interaktion zwischen Personen, die sich an unterschiedlichen Punkten auf dem Kontinuum zwischen Realität und Virtualität befinden – siehe Abschnitt 2.2.2.

erfolgte in Räumen der Hochschule für Angewandte Wissenschaften Hamburg im Mediencampus Finkenau und nahm zwei Seminarräume sowie mehrere Monate Zeit in Anspruch. XcapeRoom richtet sich an Spielgruppen von drei Personen, wobei sich davon jeweils zwei Personen im realen Raum und eine Person in einem anderen Raum und dort in einer VR-Umgebung befinden. Die Rätsel sind dabei in solch einer Form gestaltet worden, dass die zuerst völlig unterbrochenen Interaktionsmöglichkeiten zwischen den Realitäten nach und nach wieder aufgebaut werden können und so mehr und mehr Kooperation zwischen den Teilnehmenden entsteht. Die Cross-Reality-Interaktionen wurden durch die Spieleengine Unity, eine Oculus Quest im Airlink Modus sowie zahlreichen Mikrocontrollern, die in den physischen Rätseln installiert sind, gewährleistet. Als Herzstück dient auf der einen Seite eine hochaufgelöste Pointcloud, die von einem Intel Realsense Lidar-Sensor (L515) generiert wird und die Spielenden im physischen Raum in ihrer korrespondierenden Position für die Person in der VR anzeigt, und auf der anderen Seite ein Lichtpunkt, der im physischen Raum auf den Boden projiziert wird und die korrespondierende Position der Person in der VR für die anderen Spieler*innen sichtbar macht.

Die Entwicklung begann Anfang 2022 mit dem Prototyping der einzelnen Elemente und der Erforschung des Realsense Sensors. Der erste Prototyp wurde im Mai ausgestellt, doch der Ausfall des Lidar-Sensors, verunmöglichte die geplante Durchführung des Experiments. Nach einer Iterationsphase, bei der unter Berücksichtigung von Feedback aus dem ersten Prototypen auch der Spielablauf verbessert wurde, wurde ein zweiter Versuch im August gestartet.
In der zweiten Iteration wurden nach den Spieldurchläufen Interviews mit 18 Teilnehmenden in 6 Gruppen aus je 3 Personen durchgeführt, um die Erfahrungen, insbesondere mit dem zentralen Element der XR-Interaktion, der Spielenden abzufragen. Die Interviews wurden frei gehalten, um auf die individuellen Spielverläufe eingehen zu können, und mit einem Smartphone zur späteren Auswertung mitgeschnitten.

Für die Evaluation des Prototypen wurden sowohl die informellen Interviews sowie die während der Spielabläufe gemachten Beobachtungen herangezogen, als auch eine Analyse anhand einer Taxonomie von Escape Room Rätseln von Krekhov et al. [2021] durchgeführt. Die Arbeit stellt damit keine quantitative Studie dar, sondern dient in erster Linie dazu, die während der praktischen Arbeit gesammelten Erfahrungen zu dokumentieren, um das Kreieren zukünftiger XR-Escape Rooms zu erleichtern.

1.3 Struktur der Arbeit

Diese Arbeit bietet in Kapitel 1 einen theoretischen Rahmen zu Escape Rooms und diskutiert hierbei insbesondere die Ergebnisse von Krekhov et al. sowie Nicholson. Danach wird ein Überblick über Virtuelle Realität, im speziellen Mixed- und Cross-Reality-Interaktionen, gegeben.

Um zu veranschaulichen, wie der Aufbau von XcapeRoom erfolgte, wird in Abschnitt 3, Konzept und Durchführung, eine detaillierte Beschreibung der praktischen Arbeit, dem Gestalten und Bauen des hybriden Escape Rooms, gegeben. Nach einer Erläuterung des Playthroughs folgt eine Erklärung des Gesamtkonzepts anhand des von Krekhov et al. entwickelten High Level Frameworks. Eine Vorstellung allgemeiner Aspekte des Escape Rooms schließt sich an. Im folgenden Kapitel 4 werden die einzelnen Rätselelemente und ihr Design kleinschrittig beschrieben und mithilfe der von Krekhov et al. vorgeschlagenen Taxonomie analysiert.

In Abschnitt 5, Evaluation, werden die Erkenntnisse beschrieben, die aus den Interviews mit den Teilnehmenden gesammelt werden konnten. Die Interviews haben das Erleben der Zusammenarbeit mit physisch voneinander getrennten Mitspielenden als Fokus. Des Weiteren wird das Optimierungspotenzial des aktuellen Standes des Projekts beleuchtet und danach weiterführende Anwendungsmöglichkeiten für die in der Cross-Reality-Interaktion verwendete Struktur und Technologie betrachtet sowie ein Ausblick für die weitere Verwendung von XcapeRoom gegeben.
Die Ergebnisse der vorliegenden Arbeit werden in einem abschließenden Fazit zusammengefasst.

2 Theoretischer Rahmen

2.1 Escape Rooms

Es scheint schwierig, eine allgemeingültige Definition eines Escape Rooms aufzustellen. Relevante Papers beginnen oft mit einer Beschreibung einer typischen Escape Room-Erfahrung [22, 24]. Dabei widersprechen die Ergebnisse von Nicholsons Untersuchung [2015] von Escape Rooms aus der ganzen Welt der einleitenden Beschreibung von Papadopoulos und Tenta [2021], die schreiben:

> „Each escape room is based on a thematic story and therefore the players are introduced to this story and the rules by the game master."

So haben von den 224 Räumen, zu denen Nicholson Informationen von den Betreibenden eingeholt hat, 27% „no narrative" und 13% sogar „no theme" [22]. Selbst der sehr kurzen Definition, mit der Krekhov et al. [2021] ihren Text einleiten:

> „Escape rooms are games where one or multiple players solve puzzles to exit from a locked room."

wird von Nicholsons Ergebnissen widersprochen, da etwa 30% der abgefragten Räume, trotz des Namens Escape Room, nicht erforderten, einem Raum zu entkommen, um das Spiel zu gewinnen [22]. Nicholsons eigene Definition, die er seiner Ablaufbeschreibung voranstellt, ist angesichts seiner Untersuchungen stichhaltig:

> „Escape rooms are live-action team-based games where players discover clues, solve puzzles, and accomplish tasks in one or more rooms in order to accomplish a specific goal (usually escaping from the room) in a limited amount of time." [22]

Erweitert man den Begriff jedoch um virtuelle Escape Rooms und verpackte Sets, die ein Spiel ortsungebunden ermöglichen sollen, so ist sein Punkt hinfällig, dass es sich um teambasierte Spiele handelt, denn diese beiden Formen des Escape Rooms benötigen nicht zwangsweise Teams. Genau diese Erweiterung nehmen Krekhov et al. vor, weshalb in ihrer Definition von „one or multiple players" die Rede war [15]. Das einzige Element der Definitionen, das nicht weggekürzt oder zur Allgemeingültigkeit erweitert werden muss, scheint zu sein, dass Rätsel[3] gelöst werden müssen. Krekhov et al. argumentieren, dass der „unique selling point" von Escape Rooms, der sie von üblichen, „standalone"-Rätseln unterscheidet, die hohe Diversität der Rätsel ist, die innerhalb eines solchen Anwendung finden [15]. Trotz dieser sehr offenen Definition scheint der Begriff seine Berechtigung zu haben. Obwohl die Art Spiele, die gemeint ist, mithin auch unter anderen Begriffen bekannt ist (etwa

[3] In der englischen Fachliteratur wird der Begriff „puzzle" verwendet. Um Verwechslungen mit dem deutschen Wort „Puzzle" zu vermeiden, wurde für diese Arbeit die deutsche Übersetzung „Rätsel" gewählt.

"exit games," "breakout games," "unlock games" [15]), und Nicholson sogar vorschlägt, den Namen für die Zukunft anzupassen, um die Realität der gebotenen Erlebnisse besser abzubilden und das Marketing zu verbessern [22], ist der Begriff Escape Room gängig genug, um Einzug in die Titel zahlreicher wissenschaftlicher Betrachtungen zu finden. Auch im Lebensumfeld des Autors wird der Begriff allgemein verstanden und für die Art Erlebnis, die in dieser Arbeit behandelt wird, verwendet. Die in Hamburg ansässigen Firmen TeamEscape [1] und Escapediem [6] benutzen beispielsweise beide den Begriff Escape Room, um für sich zu werben.

Die noch kurze Geschichte der Escape Rooms scheint bereits jetzt nebulös. Während Warmelink et al. [2017] von einer Geschichte von etwa 35 Jahren sprechen – sprich einem Beginn um 1982 – zweifelt Nicholson in seinem Kapitel darüber bereits die Angabe von Wikipedia an, dass der erste Escape Room 2006 durchgeführt wurde und verweist stattdessen auf belastbare Quellen für das erste sogenannte Escape Game im Folgejahr in Kyoto [22]. Nicholson erweitert seine Betrachtung auf Vorgänger des Escape Rooms, die er in älteren Formen interaktiver und auch linearer Medien findet, besonders Live-Action Rollenspiel, Point-and-Click-Adventure-Spiele, Schnitzeljagden, Interaktives Theater, Abenteuer-Spieleshows und -Filme sowie Geisterbahnen [22]. Eine detaillierte Erläuterung all dieser Inspirationen würde den Rahmen dieses Kapitels sprengen, doch von besonderem Interesse für diese Arbeit ist vor allem das Beispiel der britischen Abenteuer-Spieleshow „Knightmare" aus den 80er Jahren des letzten Jahrtausends, in dem bereits mit asymmetrischem Gameplay gearbeitet wurde, indem ein*e Spieler*in vor einem Bluescreen interagierte und Hinweise von Mitspieler*innen erhielt, die eine vollständige Szene sehen konnten [22]. Ein anderes Beispiel ist Entros, eine interaktive Attraktion in Seattle, wo ein*e Spieler*in einen Helm mit einer Kamera trug und aus der Ferne Anweisungen von anderen Spieler*innen erhielt, um einfache physische Aufgaben zu bewältigen [22].

Seit ihren Anfängen in Kyoto sind Escape Rooms sehr populär geworden [15]. Dies zeigt sich nicht zuletzt auch darin, dass, wie Nicholson berichtet, viele Räume sich in einem gesättigten Markt wiederfinden [22]. Über die stationären Räume hinweg laufen unter dem Namen Escape Room diverse analoge Spiele, wie Bücher oder verschlossene Kästen, sowie digitale Spiele auf dem PC oder Smartphone [15]. Besonders interessant im Rahmen dieser Arbeit sind natürlich Virtual Reality-Räume, die sowohl für den Hausgebrauch als auch als stationäre Einrichtung existieren [15]. Im Gegensatz zu anderen digitalen Escape Rooms ist die Erfahrung bei VR-Räumen durch die immersiven Interaktionsmöglichkeiten näher an echten Escape Rooms, kann aber um ansonsten unmögliche Interaktionen und Effekte erweitert werden [15]. Der Mixed Reality Escape Room AMELIO, der Gegenstand einer Untersuchung zum Teambuilding war, ermöglicht zum Beispiel eine asymmetrische Rollenverteilung der Spielenden, indem sie im gleichen physischen Raum Unterschiedliches sehen und auf unterschiedliche Weise mit den virtuellen Elementen interagieren können [34]. Dieses Beispiel zeigt auch, dass, neben der kommerziellen Nutzung, Escape Rooms mittlerweile Forschungsgegenstand geworden sind. Arbeiten wie Nicholsons geben dabei einen Überblick über das gegebene Umfeld kommerzieller Räume, während andere, wie das AMELIO Paper, die mögliche Anwendung für Teambuilding erforschen. Auch die Erprobung, inwiefern Escape Rooms als immersive Lehrmittel genutzt werden können, rückt immer mehr in den Fokus [14, 32].

2.2 Virtual Reality

Virtual Reality ist mittlerweile zu einem geflügelten Wort geworden. Als theoretische Basis erfolgt hier eine kurze Definition der VR, bevor die konkret relevante Thematik der Cross Reality angegangen wird. Steuer [1995] definiert Virtual Reality wie folgt: „A virtual reality is defined as a real or simulated environment in which a perceiver experiences telepresence." Da sie sich über den Begriff der Telepräsenz definiert, ist diese Definition offen und enthält neben dem, was landläufig als VR bezeichnet wird, Medien von Büchern über Filme bis Radio – die alle Telepräsenz in verschiedenem Maße entlang der Achsen „Vividness" und „Interactivity" erzeugen können [29]. Diese perzeptive Definition mag ihre Berechtigung für die Diskussion moderner Medien in Politik und Wissenschaft haben, doch für den Zweck dieser Arbeit muss sie konkretisiert werden. Eine technische Definition bietet sich hierfür besser an. Eine solche erfolgt etwa in Steinickes „Being Really Virtual" [2016]:

> „Virtual Reality (VR) requires three *real* features: (i) *real*-time rendering with viewpoint changes as head moves, (ii) *real* space, i.e., either concrete or abstract 3D virtual environments, and (iii) *real* interaction, i.e., possible direct manipulation of virtual objects."

Um dies zu erreichen, wird üblicherweise ein Stereo-Displaysystem, das die Sicht der Anwender*innen völlig mit computergenerierten Inhalten füllt, mit einem Headtrackingsystem kombiniert, welches diese Inhalte der aktuellen Kopfposition anpasst [26]. Seit dem Release der Oculus Rift, die ein solches System massenmarkttauglich machte, im Jahr 2016, hat VR-Technologie ein breites Publikum erreicht. In dieser Arbeit findet ein Nachfolgemodell der Oculus Rift, die Oculus Quest, Anwendung, die durch getrackte Controller zusätzlich zu den oben genannten Features die manuelle Interaktion mit virtuellen Objekten ermöglicht.

Ein wichtiges Element von VR ist Presence, oder Präsenz, das Phänomen, dass Personen in einer VR-Umgebung agieren, als befänden sie sich an einem echten Ort und würden an echten Ereignissen teilnehmen [26]. Slater und Sanchez-Vivez [2014] unterscheiden dabei, im Gegensatz zu früheren Ansätzen, die Präsenz als das Gefühl von „being there" beschrieben, zwischen Place Illusion (PI) und Plausability Illusion (Psi). Ersteres beschreibt die Illusion, sich an einem virtuellen Ort tatsächlich zu befinden und wird stark durch die Qualität des Head- oder Eye-Trackings bestimmt [26]. Letzteres ist abhängig von der Glaubhaftigkeit der Umgebung und wie sehr die virtuelle Welt auf die Person eingeht [26]. Eine weitere wichtige Dimension der Präsenz ist die soziale Präsenz (social presence), die von Oh et al. [2018] in ihrem Systematic Review of Social Presence wie folgt beschrieben wird:

> „Social presence refers to the subjective experience of being present with a 'real' person and having access to his or her thoughts and emotions."

Dieser Aspekt ist im Rahmen eines kollaborativen Unterfangens wie einem Escape Room besonders interessant. Matthes et al. [2019] sprechen in diesem Kontext von telepresence (Telepräsenz), wenn sie ihr System beschreiben, das dazu dient, Gruppen von mehreren Personen über die Distanz (Tele) in einem virtuellen Raum (präsent) zusammenzuführen. Sie benutzen nicht den Begriff sozialer Präsenz, aber referenzieren die für diese wichtigen [23] Marker von Gesichtsausdrücken und Gestiken [17]. Um diese realitätsgetreu wiederzugeben, wählen sie für die Teilnehmenden 3D-Video-Avatare als Darstellung [17]. Ein Avatar ist eine Repräsentation einer Person in der virtuellen Welt und kann verschiedene Formen annehmen – von einem einfachen 3D-Model in einem Videospiel über durch Motion Tracking korrekt animierte detaillierte Körper, bis hin zu den detail-

lierten per Live-Aufnahme erzeugten 3D-Video-Avataren von Matthes et al. Bei Avataren, deren visuelle Darstellung dabei der gefühlten Position der User entspricht, entsteht leicht ein Gefühl von „embodiment", das Gefühl, dass der virtuelle Körper, dessen Perspektive man einnimmt, der eigene Körper ist, mit vielschichtigen Implikationen für soziale Interaktionen im virtuellen Raum [26].

2.2.1 Mixed Reality

Zwischen einer rein virtuellen und einer rein realen Umgebung (Environment) besteht ein Kontinuum, das von Milgram und Kishino 1994 als Virtuality Continuum definiert wurde [21]. Sie definieren den Begriff Mixed Reality (MR) als jenen Bereich, der zwischen den beiden Extremen liegt, wo also virtuelle und reale Anteile gemischt werden [21]. Milgram und Kishinos Paper wird in diesem Kontext bis heute oft zitiert [8, 25]. Über den Begriff der Mixed Reality hinaus liefern sie auch Definitionen für Augmented Reality (AR) und Augmented Virtuality (AV), die für die Einordnung des XcapeRooms eine Rolle spielen. Augmented Reality sind hauptsächlich reale Environments, die virtuelle Elemente enthalten, Augmented Virtuality umgekehrt hauptsächlich virtuelle Environments, die reale Elemente enthalten [21].

2.2.2 Cross-Reality-Interaktionen

Während May [2020] in seinem Paper über Cross-Reality-Kollaboration den Begriff Cross Reality (XR) in den Bereich des Gamings verortet, wird er von Simeone et al. [2020] ohne Verweis auf einen solchen Ursprung verwendet. Er definiert bei ihnen Systeme, bei denen Interaktion zwischen Personen an unterschiedlichen Punkten des MR-Spektrums oder Zusammenarbeit zwischen Personen, die sich am gleichen Ort in einer MR-Situation befinden, stattfindet. Für diese Arbeit ist das Konzept von Simeone et al. besonders interessant, da es sich auf asymmetrische Interaktionen konzentriert, bei denen „non-equivalent VEs [Virtual Environments]" Gebrauch finden, also virtuelle Umgebungen, die sich unterscheiden, bei denen verschiedene User jeweils Zugriff auf ihre eigene Realität haben, aber mit der Realität anderer User interagieren [25]. Als einen der zwei Hauptaspekte dieses Konzepts nennen die Autoren hier Kollaboration, in dessen Zuge auch der Begriff Collaborative Virtual Environment (CVE) eingeführt wird. Eben diese Kollaboration steht im Zentrum von XcapeRoom. Andere Ansätze, wie das Collaborative Virtual Reality Neurorobotics Lab [17], verfolgen ebenfalls das Ziel der Kollaboration, wobei hier für alle menschlichen Teilnehmenden gleichgeartete VEs vorgesehen sind. Obwohl die Autoren dies nicht tun, wäre dennoch eine Einordnung als XR-Plattform sinnvoll, da sie sich eignet, um gleichzeitig mit anwesenden (collocated) und dazugeschalteten (remote) Teilnehmer*innen zusammenzuarbeiten. Für XcapeRoom interessant ist dabei, neben den Begriffen Collocated Collaboration und Remote Collaboration, die Art der Darstellung der Remote-Teilnehmer, nämlich als Live 3D-Video-Avatar, basierend auf RGBD-Geräten [17], wie sie auch bei XcapeRoom Anwendung fanden.

3 Konzept und Durchführung

Damit die Lesenden einen groben Einblick in den Ablauf von XcapeRoom bekommen, schildert dieser Abschnitt zunächst das Gameplay des Escape Rooms, d.h. die Rätsel und ihre Lösungen werden in der idealen Reihenfolge beschrieben. Anschließend wird in Abschnitt 3.2 die Konzeption näher erklärt. Dies geschieht anhand des High Level Frameworks von Krekhov et al., das in diesem Zuge erläutert wird. Die Story des Spiels erhält ihren eigenen Abschnitt. Anschließend werden in den Abschnitten 3.3 bis 3.8 verschiedene Aspekte der Entwicklung näher beleuchtet.

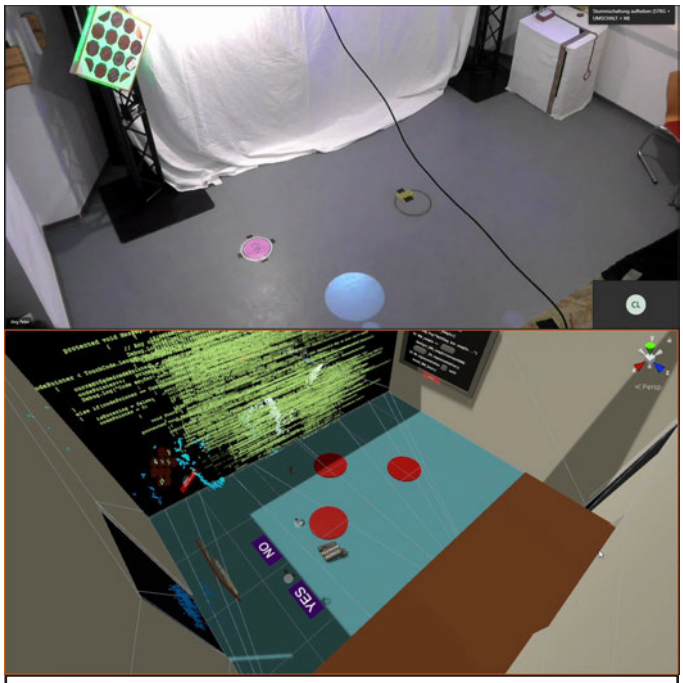


Abb. 1: P-Raum und VR-Raum aus der gleichen Perspektive

3.1 Gameplay Walkthrough

Es folgt die idealisierte Beschreibung des Spielablaufs. Der Mittelteil des Spiels ist dabei nicht strikt linear und die verschiedenen Rätsel können auch in abweichender Reihenfolge gelöst werden.

Für Präzision und einfaches Verständnis ist dieser Abschnitt im Präsens verfasst. Ebenso wurde auf eine detaillierte Beschreibung der Rätsel verzichtet, um Wiederholungen zu vermeiden und den Spielablauf verständlicher zu gestalten. Eine korrespondierende Erläuterung ist jeweils in Kapitel 4 zu finden. Teile der Geschichte des Spiels, die zum Verstehen notwendig sind, wurden kursiv geschrieben, um klarzustellen, was gesagt und wie es den Spielenden vermittelt wird. Wichtige Elemente sowie die Rätselnamen sind fettgedruckt.

Im Anschluss an den Text zeigt eine Flowchart systematisch die Interaktionen und Konditionen für die Ereignisse.

3.1.1 Walkthrough

Drei Spielende betreten den Raum zusammen mit dem Host, der eine kurze Einleitung in das Setting gibt: *Meine gute Freundin Professor Fernanda Martinez hat ein Smart Home entwickelt, das Gedanken lesen kann. Bevor sie es jedoch fertigstellen konnte, ist sie plötzlich verschwunden. Da ich in Besitz des Schlüssels bin, dachte ich, dass ich der Welt diese zwar unfertige, aber dennoch beeindruckende Erfindung nicht vorenthalten kann. Um den Gedankenleseprozess zu starten, muss eine Person ihre Hirnwellen scannen lassen, damit das Haus sich darauf einstellen kann.*

Es wird nach einer freiwilligen Person gefragt, wobei erwähnt wird, dass sie diejenige sein sollte, die das meiste Interesse an VR hat. Jene Person wird alsdann auf einen als Sessel getarnten Rollstuhl in einen abgetrennten Raumteil vor eine zweite Tür gesetzt und der Host hilft ihr, das **VR-Headset**, das als *Hirnwellen-Scanner* ausgegeben wird, aufzusetzen. Danach schließt der Host den Vorhang vor dem abgetrennten Teil und verlässt den Raum, angeblich *um nicht den Scanprozess durch die eigenen Hirnwellen zu stören*. Während die Spielenden darauf warten, dass der Hirnwellen-Scan abgeschlossen wird, startet der Host ein Video, das auf

einem Fernseher im Raum abgespielt wird. Das Video suggeriert zusammen mit einem Subwoofer im Raum ein *Gewitter*, bei dem das Smart Home von einem *Blitz* getroffen wird. Während dieser Ablenkung wird die Person mit dem VR-Headset (ab jetzt: VR-Person) von hinten aus der zweiten Tür gerollt, ohne dass die beiden Spielenden im physischen Raum (ab jetzt: P-Personen) es bemerken. Zusammen mit dem *Blitzeinschlag* geht das Licht im physischen Raum (P-Raum) aus, dazu ist ein Alarm zu hören. Der Host erklärt durch die Tür, dass *die VR-Person irgendwie in die VR gezogen wurde und dass die P-Personen sie nun retten müssen*. Die zweite Tür, durch die die VR-Person entführt wurde, ist elektrisch verschlossen, doch die P-Personen können daneben eine **Klinkenbuchse** finden, allerdings ist das Licht noch aus. Eine kleine Kiste – **die VR-Box** – leuchtet rot. Zieht man den Stecker und steckt ihn wieder ein, geht der Alarm aus, doch es bleibt dunkel. Zur gleichen Zeit wird die VR-Person in einen anderen Raum im Gebäude gefahren, während sie eine virtuelle Weltraumumgebung sieht. Im VR-Raum angekommen (ein komplett dunkles Zimmer, in dem es nicht von Nutzen ist, die VR Brille abzunehmen), erscheint ihr eine Wolke von Codeschnipseln. Versuchen die P-Personen den **Lichtschalter** neben der Tür zu bedienen, passiert im P-Raum zuerst nichts, doch in der VR lichtet sich der Code und ein Schalter ist zu sehen. Wird dieser betätigt, verschwindet er und eine virtuelle Repräsentation (ab jetzt: VR-Raum) des P-Raums wird sichtbar, außerdem geht das Licht im P-Raum selbst an. Die Position der VR-Person wird nun im P-Raum mit einem sich bewegenden Lichtpunkt auf dem Boden angezeigt, und die VR-Person kann nun den Raumsound aus dem P-Raum hören. Eine rudimentäre Kommunikation wird durch auf dem Boden markierte Felder mit der Beschriftung „YES" und „NO" gewährleistet.

Auf dem Screen im P-Raum steht jetzt die Anweisung „PRESS", die auf einen Knopf verweist, der den **DNA-Scanner** aktiviert. Durch Betätigen des Knopfes wird ein Tutorial zur Benutzung des Scanners abgespielt, das erklärt, dass *durch die Eingabe der eigenen DNA Avatare in der VR erzeugt werden*. Zur gleichen Zeit muss die VR-Person das **Code-Rätsel** lösen, indem sie Blöcke mit Wörtern in einen Code platziert. Wenn die Rätsel auf beiden Seiten gelöst sind, werden die P-Personen als **Pointcloud** analog ihrer tatsächlichen Position im P-Raum auch im VR-Raum dargestellt. Ebenfalls werden die drei Knöpfe auf dem Boden im VR-Raum dargestellt, die zum Lösen des **Pressure-Plates-Rätsels** benötigt werden. Einer der Blöcke für das **Code-Rätsel** ist hinter einer **bewegbaren Wand** im VR-Raum versteckt. Die Wand kann einfach an einem Griff heruntergezogen werden, was gleichzeitig eine **bewegbare Wand** an der gleichen Stelle im P-Raum nach unten fahren lässt. Im P-Raum kann hinter der Wand der **PolRev** gefunden werden, eine kleine Plastikbox mit einem großen Klinkenkabel, einem Platz für einen **9V-Block**, einem Schalter, einem Slot für einen speziell aussehenden **Schlüssel** und mehreren Widerständen. Die Teile zur Vervollständigung sind hinter den verschiedenen Rätseln im Raum versteckt.

Die P-Personen berühren nun die Symbole des **LED-Wand-Tresors** in der korrekten Reihenfolge. Diese ist so auf einer Wand vermerkt, dass man sie nicht gleichzeitig lesen und eingeben kann, was die P-Personen dazu zwingt, einander die Symbole vorzulesen. Durch diese Kommunikation wird die VR-Person darauf aufmerksam gemacht, was wichtig ist, da das Rätsel nur kooperativ gelöst werden kann, denn einige der Symbole sind nur auf der Entsprechung des Wand-Rätsels in der VR zu sehen. Berührt die VR-Person die Symbole auf ihrer Wand, so leuchten sie auch in der VR auf, so kann das Rätsel zu dritt gelöst werden. Hierfür erhalten die P-Personen den **9V-Block** für den **PolRev** und die VR-Person ein **Radial Menu**, mit dem sie voreingesprochene Wörter an die P-Personen schicken kann. Diese Wörter beinhalten "Here", "Stay" und "Go", mit denen die VR-Person die P-Personen an spezifische Punkte leiten kann. Dies wird im nächsten Rätsel, dem **Pressure-Plates-Rät-**

sel, benötigt. Sind alle Personen korrekt platziert, erhält die VR-Person einen **Schalter zum Aktivieren des Partymodus**. Wird dieser aktiviert, ändert sich in beiden Räumen das Licht und es wird Musik gespielt. Im P-Raum geht außerdem eine **UV-Lampe** an, die eine bisher unsichtbare Information beleuchtet: Die benötigte **Widerstandsangabe**, um den PolRev zu betreiben. Gleichzeigt erscheint an der dazu analogen Stelle im VR-Raum ein **Passwortblock**, den die VR-Person wie die Blöcke im Code-Rätsel nutzen kann.

Drückt eine P-Person den roten Knopf auf dem **Digital-Clean-Up-Besen** (Besen mit Knopf und LED) erscheint in der VR hinter einer weiteren Wand mit fliegendem Code ein **Passwortfeld**, in das der **Passwortblock** gesetzt werden muss, um Zugriff auf das **VR-Telefon** zu bekommen. Dieses befand sich schon die ganze Zeit im P-Raum und kann jetzt von der VR aus angerufen werden, um eine Sprachkommunikation in beide Richtungen zu etablieren.

Die P-Personen nutzen jetzt die **Widerstandsangabe**, um das **Widerstandsrätsel** auf dem **PolRev** zu lösen.

Über das **VR-Telefon** kann nun kommuniziert werden, um den passenden **Schlüssel** für den **PolRev** im **Schlüsselrätsel** mit dem **3D-Drucker** auszudrucken. Ist der korrekte **Schlüssel** ausgegeben worden, können die P-Personen ihn in den **PolRev** einführen. Nun müssen die einzelnen Teile zusammengebracht werden: Die Widerstände müssen korrekt verbunden sein, der Schalter muss auf „Polarity" gestellt sein, der **9V-Block** muss angeschlossen sein und der **Schlüssel** muss heruntergedrückt werden. Sind alle diese Dinge gegeben, leuchtet der **PolRev** grün. Wird er nun mit dem Klinkenkabel an die Tür angeschlossen, öffnet sich diese.

Nun können die P-Personen der VR-Person *in die VR* folgen, um sie zurückzuholen. Die VR-Person hat eine **dreidimensionale Karte**, die vom P-Raum bis in den VR-Raum führt, und einige Symbole enthält, die in der Realität zu finden sind. Anhand dieser Symbole erklärt die VR-Person den P-Personen am **VR-Telefon**, wohin sie gehen müssen. Sobald die P-Personen am VR Raum ankommen, können sie den mit Molton abgedunkelten Bereich betreten und die *VR-Person rechtzeitig befreien, bevor sich ihre Gedanken dauerhaft in der VR verlieren*. Damit ist das Spiel gewonnen.

3.1.2 Schematische Darstellung

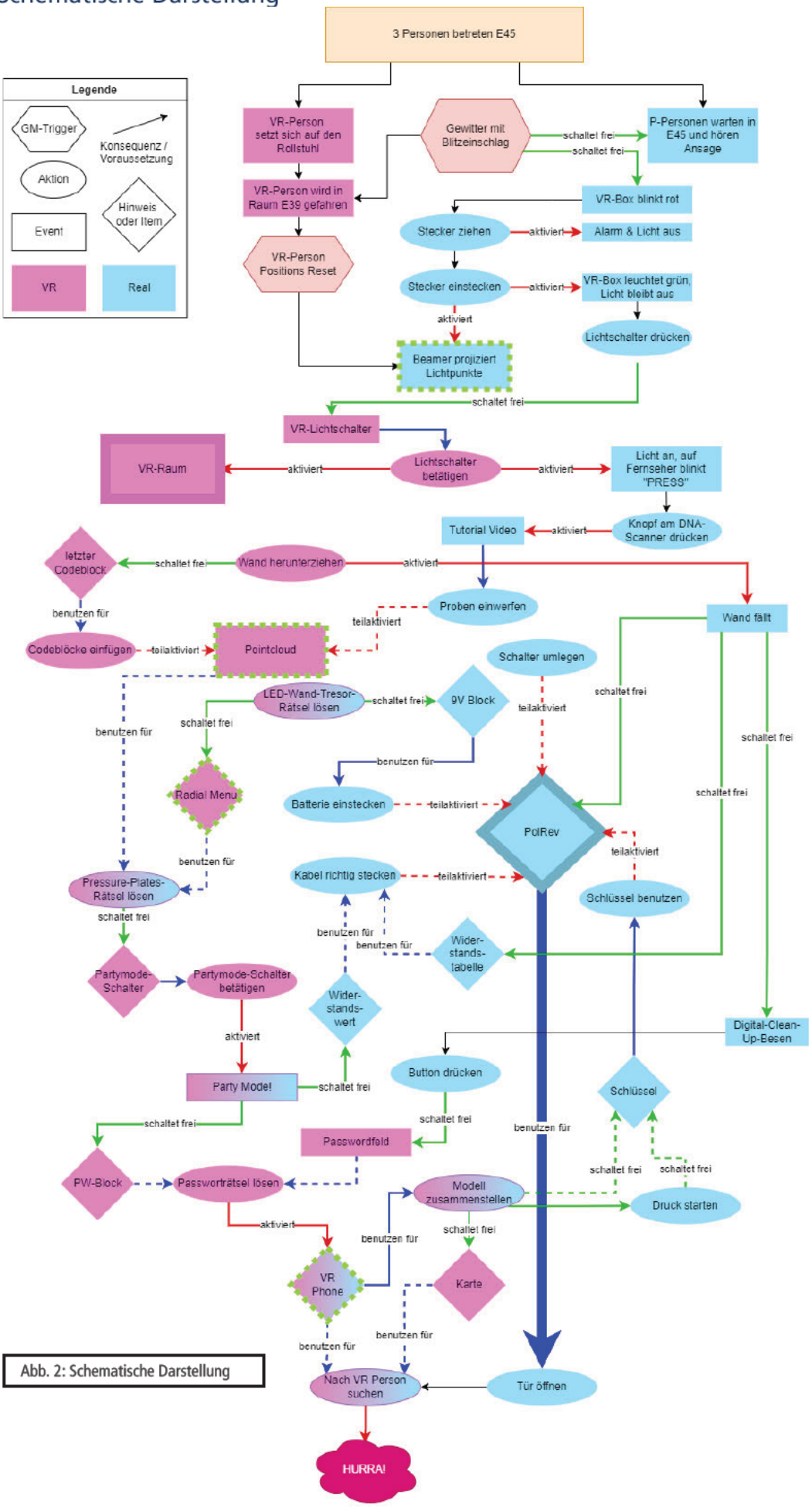


Abb. 2: Schematische Darstellung

3.2 Konzeption

Bei der Konzeption stand die Entwicklung des Gameplay-Ablaufs im Vordergrund. Ausgehend von der Anforderung, reale mit virtueller Interaktion zu kombinieren, und der persönlichen Erfahrung des Autors mit Escape Rooms und Videospielen, wurden sowohl die Handlung als auch die Rätsel in einem Fluss erdacht. Leichte Anpassungen erfolgten im Laufe der Entwicklung: So wurden Elemente gekürzt, um einerseits die Entwicklung zu beschleunigen und andererseits die Durchläufe des Spiels für die Teilnehmer*innen zu verkürzen, um eine größere Anzahl an Durchläufen zu ermöglichen.

Auf theoretische Grundlagen wurde nicht zurückgegriffen, da der geplante Ablauf bereits sehr schnell sehr konkrete Züge annahm und das Ziel der Arbeit ein Prototyp war, der als Grundlage zur Weiterentwicklung des XR-Escape-Room-Konzeptes dienen sollte. Es war daher geboten, umgehend mit der Umsetzung anzufangen, die sich als sehr arbeitsintensiv herausstellte. Bei der Konzeption wurden bereits die vorhandenen Fähigkeiten des Teams berücksichtigt, um die Entwicklung der nötigen Technik im Rahmen des Machbaren zu halten. Jörg Peter war dabei für die Softwareseite sowie die Konzeption der elektronischen Schaltungen zuständig und Charlotte Lehr für die physische Konstruktion und visuelle Gestaltung der notwendigen Elemente.

3.2.1 High Level Framework von Krekhov et al.

Das Design lässt sich anhand der Kriterien von Krekhov et al. in ihrem High Level Framework [15] analysieren. Dies ermöglicht eine Grundlage, auf der sich die Erfahrungen aus den Testläufen einordnen lassen. Das Framework besteht aus sechs Designaspekten:

> „(1) target group and team composition, (2) theme and narrative, (3) modalities and platform, (4) puzzle organization, (5) puzzle design, and (6) hint system and failure handling." [15]

Im Folgenden werden die Punkte in Bezug auf XcapeRoom erläutert.

1) Target Group and Team Composition

Dieser Punkt dreht sich um die Zielgruppe im Sinne von Interessen und Vorwissen sowie die Größe des teilnehmenden Teams [15]. Im Falle des vorliegenden Projekts war die Zielgruppe hauptsächlich auf Studierende der HAW Hamburg sowie Freunde und Bekannte des Teams beschränkt. Dadurch war durchschnittlich von einer gewissen Vorerfahrung mit Virtual Reality und Spielen allgemein auszugehen – wobei sich diese Vermutung nur teilweise bewahrheitet hat, wie in Teil 5.1 über die Auswertung der Durchläufe gezeigt werden wird. Es wurde davon ausgegangen, dass die Teilnehmenden alle volljährig und geschlechtlich gemischt sein werden. Die Teamgröße wurde, im Kontrast zu regulären kommerziellen Escape Rooms, wo üblicherweise eine gewisse Spanne von Mindest- und Maximalteilnehmer*innenzahl angegeben wird [15], auf genau 3 Spieler*innen festgelegt. Dies diente dazu, das Design zu vereinfachen und die Erfahrung für die Befragungen zu vereinheitlichen.

2) Theme and Narrative

Beim Thema (Theme) handelt es sich um das Setting des Spiels, während das Narrativ (Narrative) die Geschichte beschreibt, die als Hintergrundinformation oder durch das Spiel hindurch erzählt wird [15]. Krekhov et al. definieren wie folgt:

> "The theme describes the general setting of the game and mainly influences the aesthetic audio-visual design of the room […] A narrative provides further story information on how players ended up in the current room/situation and explains the main goal and motivation." [15]

Ihnen zufolge ist ein Narrativ nicht notwendig, hilft jedoch dabei, die Immersion und Motivation zu steigern. Für XcapeRoom fiel die Entscheidung, das Thema passend zu einer Interaktion mit einem Computersystem (was durch die VR-Person dargestellt wird) festzulegen – solche Interaktionen sind im Bereich des Smarthomes geläufig, weshalb dieses als Setting gewählt wurde. Daraus entwickelte sich dann passend das Narrativ. Auf diesen Aspekt wird in Abschnitt 3.2.2 Storytelling weiter eingegangen.

3) Modalities and Platform

Da Escape Rooms, wie im Abschnitt 2.1 erwähnt, vielfältige Erscheinungsformen haben, ist die Wahl der spezifischen Modalität (Modality) und Plattform von großer Bedeutung für das Design [15]. Die Plattform ist dabei die Wahl zwischen realen stationären, realen verpackbaren oder digitalen Escape Rooms, wobei digitale Räume die Möglichkeit verschiedener Input-Modalitäten mit sich bringen [15]. Im Falle von XcapeRoom stand diese Entscheidung ganz am Anfang. Die zentrale Technik sollte eine interaktive Verbindung von rein realen und rein virtuellen Spielenden sein, da diese Lücke im bestehenden Escape Room Ökosystem bemerkt wurde. Bei XcapeRoom war also eine hybride Mischung aus stationärem realem Raum und VR-Raum erforderlich.

4) Puzzle Organization

Rätsel können in Escape Rooms in verschiedenen Strukturen organisiert werden. Krekhov et al. definieren dabei in Bezug auf Wiemker et al. drei mögliche Strukturen: linear, offen (open) und multi-linear [15]. Linear bedeutet dabei, dass die Rätsel ohne mögliche Variation der Reihenfolge aufeinander folgen, offen, dass die Rätsel den Spieler*innen alle gleichzeitig zur Verfügung stehen und multi-linear stellt eine Mischung dar, bei der mehrere lineare Pfade gleichzeitig zur Verfügung stehen [15]. Die Organisation ist von verschiedenen grundlegenderen Überlegungen, wie beispielsweise Narrativ und Teilnehmer*innenzahl abhängig – lineare Pfade eignen sich besser zum Erzählen einer linearen Handlung, offene Designs ermöglichen es mehreren Spielenden gleichzeitig, verschiedene Rätsel zu lösen – und Escape Rooms kombinieren üblicherweise die verschiedenen Ansätze und kreieren so komplexe Mischstrukturen [15]. Im Falle von XcapeRoom liegt ebenfalls eine solche Mischstruktur vor, bei der einige Abschnitte linear verlaufen, während andere Abschnitte offen bleiben. Durch die asymmetrische Rollenverteilung entsteht sogar eine weitere Dimension der Komplexität. Da die Spieler*innen in den verschiedenen Modalitäten verschiedene Rätsel zur Verfügung haben, deren Verfügbarkeit wiederum teilweise durch die Lösung von Rätseln in der jeweils anderen Modalität bestimmt wird, während andere Teile die Zusammenarbeit aller Spieler*innen erfordern, lässt sich diese Struktur nur schwer abbilden. Ein Versuch einer Darstellung durch ein Flussdiagramm findet sich in Abschnitt 3.1.2.

Ein besonderer Faktor in der Struktur ist im Fall von XcapeRoom, dass die Kommunikation zwischen den Spielenden zunächst unterbrochen und dann Stück für Stück wieder aufgebaut wird. Dies sollte einerseits dazu dienen, zu sehen, welche Kommunikationskanäle bereits für eine Zusammenarbeit genügen, andererseits sollte es bei den Spielenden das Gefühl erzeugen, einander durch das Lösen der Rätsel näher zu kommen.

5) Puzzle Design

Dieser Punkt bezieht sich auf das Design spezifischer Rätsel im Escape Room. Zur besseren Untersuchung haben Krekhov et al. ihre atomare Taxonomie entwickelt [15]. In Kapitel 4 dieser Arbeit werden anhand dieser Taxonomie die Rätsel von XcapeRoom aufgeschlüsselt.

6) Hint System and Failure Handling

Um sicherzustellen, dass die Spieler*innen nicht stecken bleiben und so das Spiel nicht beenden können, benutzen Escape Rooms unterschiedlich geartete Hilfestellungssysteme [15]. Krekhov et al. bemerken, dass sich Escape Rooms hier grundlegend von anderen Spielen unterscheiden, die die Spielenden langsam an die Spielmechaniken heranführen, da sie aus einer Vielzahl unterschiedlichster Herausforderungen bestehen und in einem begrenzten Zeitrahmen gelöst werden sollen [15]. In XcapeRoom wurde ein sehr offenes System eingesetzt, bei dem die Spielleiter*innen den Spieler*innen jederzeit Hinweise zurufen konnten – entweder über den im realen Raum befindlichen Fernseher, oder direkt im Falle der VR-Person, die sich im gleichen Raum wie die Spielleiter*innen befand. Um die Immersion zu wahren, wurde der Ansatz verfolgt, dass die Hilfe von den fiktiven Assistent*innen der fiktiven Entwicklerin des Raums stammt. Aufgrund technischer Probleme, die eine direktere Einmischung durch die Spielleiter*innen erforderte, konnte dies jedoch in der Praxis schlecht umgesetzt werden, was dazu führte, dass das Hilfssystem sehr prominent wurde und sogar einige Proband*innen störte.

Failure Handling beschreibt, wie im Spiel damit umgegangen wird, wenn die Spielenden etwas falsch machen. In regulären Computerspielen wird hierfür oft ein Teil der Lebenspunkte abgezogen, in Escape Rooms wird manchmal eine Zeitstrafe verhängt. In XcapeRoom wird zwar am Anfang durch die Story ein Zeitlimit von 30 Minuten eingeführt, dies wurde allerdings von den Spielleitenden nicht streng durchgesetzt – auch, weil technische Probleme immer wieder den eigentlichen Spielablauf unterbrachen, was eine Bestrafung der Spielenden unfair gemacht hätte. Abgesehen davon besitzt XcapeRoom keine Elemente, bei denen es sinnvoll wäre, eine Strafe zu erteilen (wie z.B. ein Laserlabyrinth, bei dem man immer 5 Minuten verlieren könnte, wenn man einen Strahl berührt). Die Bestrafung für das Nichterledigen der Rätsel ist somit nur das Verlängern der Zeit, die man mit dem Rätsel verbringt.

3.2.2 Storytelling

Für den XcapeRoom wurde als Thema das "Smart Home" gewählt, da es sich als Schnittpunkt digitaler Technik und physischer Interaktion anbietet, denkt man zum Beispiel an appgesteuerte Backöfen oder ähnliches. Dass Objekte in einem realen Raum von einer Person in der VR beeinflusst werden, war die Kernidee des Rätseldesigns und das Thema Smart Home sollte diese Verbindung intuitiver machen. Des Weiteren gibt es eine gewisse Nähe der beiden Bereiche virtuelle Realität und Smart Homes. Beides sind junge Technologien, die immer noch als neue Entwicklungen gesehen werden und im öffentlichen Bewusstsein eher vage vertreten sind. Das Instrument des "in die VR gesaugt Werdens" hat lange Tradition in der Science-Fiction-Literatur und kann mindestens bis zum letzten VR-Hype in den 1980er und 90ern zurückdatiert werden, wo es Filme wie Tron [30] inspirierte. Erfahrungsgemäß ist das Storytelling in Escape Rooms oft wenig subtil, weshalb die Nutzung einer derart absurden Trope angemessen für dieses Projekt wirkte. Zusammen mit Details wie einem Blitzeinschlag als Grund für die Fehlfunktion erinnert die Geschichte sogar an noch ältere Science-Fiction, wie die Filmadaption von Frankenstein aus dem Jahre 1931 [11]. Das Damoklesschwert der Möglichkeit, dass die VR-Person für immer in der virtuellen Welt verloren gehen könnte, dient dabei als Mechanismus zur Erzeugung von Dringlichkeit, um die Immersion weiter zu verstärken.

Die abstraktere, gameplay-getriebene Geschichte handelt vom abrupten Verlust von Kommunikation zwischen den Spielenden und dem schrittweisen Zurückgewinnen dieser. Aus der Perspektive der P-Personen vollzieht sich der Wandel von keiner Kommunikation über das Sehen einer abstrakten Positionsangabe (durch den projizierten Punkt) und die „YES"/"NO"-Felder, über das Hören von vorausgewählten Nachrichten, bis hin zur regulären Sprachkommunikation über das Telefon. Die VR-Person auf der anderen Seite hört zuerst nur die Gespräche der P-Personen, sieht dann ihre Positionen, bekommt dann die Möglichkeit, Nachrichten zu senden und kann dann ebenfalls mit dem Telefon regulär kommunizieren. Am Ende wird die Präsenz wieder vollständig hergestellt, wenn die Spielenden sich physisch wiedertreffen.

3.3 Generelle Bemerkungen zur VR

Die virtuellen Elemente sind teilweise Nachbildungen von Elementen im P-Raum, teilweise Substitutionen und teilweise völlig unabhängig. Von besonderer Bedeutung ist die Pointcloud, die die realen Spielenden im virtuellen Raum darstellt. Aufgrund des besonderen Status hat diese daher ein eigenes Segment, in dem näher auf ihre Entwicklung eingegangen wird. Entgegen der Bezeichnung VR-Raum handelt es sich mit Einschalten der Pointcloud klar um eine Form von Augmented Virtuality. In der größtenteils virtuellen Umgebung werden die realen Personen aus dem P-Raum „sampled and then resynthesized" [21] – stellen also ein reales Objekt in der Virtualität dar.

3.3.1 Wahl des Headsets und physische Gestaltung des Spielbereichs

Abb. 3: Die Oculus Quest im Einsatz

Als VR-Headset wurde die Oculus Quest gewählt. Diese Wahl erfolgte aufgrund der Möglichkeit, diese unkompliziert kabellos zu betreiben. Ein Kabel hätte von den Spieler*innen Vorerfahrung im Umgang mit VR-Headsets, oder aber ein*e Betreuer*in erfordert, um zu verhindern, dass das Kabel zur Stolperfalle wird. Zudem stand bereits früh in der Entwicklung fest, dass die VR-Person zu Beginn mit dem Headset von einem zum anderen Raum gebracht werden sollte. Ein Kabel hätte dies verunmöglicht, ohne den ganzen PC mitzunehmen. Die Oculus Quest funktioniert darüber hinaus ohne früher übliche Basisstationen – Kameras, die die am Headset befindlichen IR-LEDs tracken. Bei der Oculus Quest sind Kameras im Headset verbaut, die Kontrastpunkte in der Umgebung verfolgen (Inside-Out Tracking), was den mobilen Einsatz ermöglicht.

Zunächst war angedacht, das Spiel nativ auf dem Headset laufen zu lassen, um auf dem Weg zwischen den Räumen unabhängig von einer ständigen Netzwerkverbindung zu sein. Zusätzlich hätte dies die Qualität der Interaktion und des Bilds unabhängig von WLAN-Übertragungsschwankungen gemacht. Die Interaktion wäre dann durch ein Multiplayer-Framework für Unity, Photon Unity Networking (PUN),[4] erfolgt. Aufgrund von Problemen beim Streamen der Pointcloud wurde dies jedoch verworfen (siehe Abschnitt 3.5) und die Funktion „AirLink" der Oculus Quest verwendet. Dabei besteht eine WLAN-Verbindung zu einem Host-Rechner, auf

[4] PUN ist als kostenloses Plugin für Unity unter folgender Webadresse verfügbar: https://www.photonengine.com/pun (zuletzt überprüft 07.09.2022). Die Multiplayer-Funktionalität wird über Server der Firma Photon ermöglicht, die bis zu einer gewissen Kapazitätsgrenze ebenfalls kostenlos nutzbar sind, was sie ideal für dieses Projekt machte, bei dem nicht mehr als zwei Verbindungen zum Server bestehen.

dem die Oculus-Software läuft. Von dieser werden die entsprechenden Bilddaten als Video zum Headset gestreamt, während das Headset seinerseits die Trackingdaten an den Host streamt, um korrekte Interaktion zu ermöglichen. Dies stellte das Team vor die Herausforderung, dass eine konstante WLAN-Verbindung in beiden Räumen sowie dazwischen notwendig wurde, da bei einer Verbindungsunterbrechung die AirLink-Funktion komplett ausfällt, was den Spielenden nicht zugemutet werden sollte. Die Räume lagen dabei zu weit auseinander, um beide von einem üblichen WLAN-Router abgedeckt zu werden. Da die AirLink-Funktionalität sehr störungsanfällig ist, wurde es nach Rücksprache mit Netzwerkadministratoren der Hochschule ausgeschlossen, dass die Verbindung über mehrere Router hinweg aufrecht gehalten werden könnte (durch sogenannte Handovers können Geräte im Betrieb zwischen verschiedenen Accesspoints wechseln, was jedoch von Hard- und Software sowohl der Accesspoints als auch des Headsets unterstützt werden müsste.). Daher wurde der Entschluss gefällt, den Router an einem langen Ethernetkabel (30m) zusammen mit der VR-Person von Raum 1 zu Raum 2 zu transportieren. Dies lief nach einigen Versuchen auch reibungslos.

Abb. 4: Der VR-Spielbereich

Der Raum, in dem sich die VR-Person während des Hauptteils des Spiels physikalisch befand, war dabei ein regulärer Seminarraum, dessen Größe ausreichte, um den realen Escape Room in voller Größe abzubilden. Für den Fall, dass der*die Spieler*in das Headset abnimmt, wurde eine Traversenkonstruktion in entsprechender Größe vollständig mit Molton verhangen. Durch die dadurch entstehende Dunkelheit sollte verdeutlicht werden, dass außerhalb der Virtualität hier kein Spiel stattfindet. Außerdem diente der Molton so als Raumtrenner, um die Spielleiter*innen von der VR-Person zu separieren. Dieses Verfahren brachte das Problem mit sich, dass dem Headset die Anhaltspunkte für das Tracking fehlten. Da das Inside-Out-Tracking der Oculus Quest darauf angewiesen ist, per Kamera Kontraste in der Umgebung zu erfassen, waren sowohl das fehlende Licht als auch die komplette Einfarbigkeit des Moltons ein Problem. Um dem entgegenzuwirken, wurden zwei Infrarot-Lampen (850 nm) und Elemente aus Kreppband und Alufolie angebracht, die das Infrarotlicht reflektieren und so Kontrastpunkte für das Headset darstellen konnten. Krepp- und Aluminiumelemente wurden Stück für Stück hinzugefügt und immer wieder das Tracking des Headsets getestet, bis es stabil funktionierte. Als Nebeneffekt erzeugten die, bewusst in bestimmten Formen angebrachten, Elemente für eine besondere Atmosphäre, wenn zum Ende des Spiels die Tür geöffnet wurde und die Spielenden sich wiederfanden.

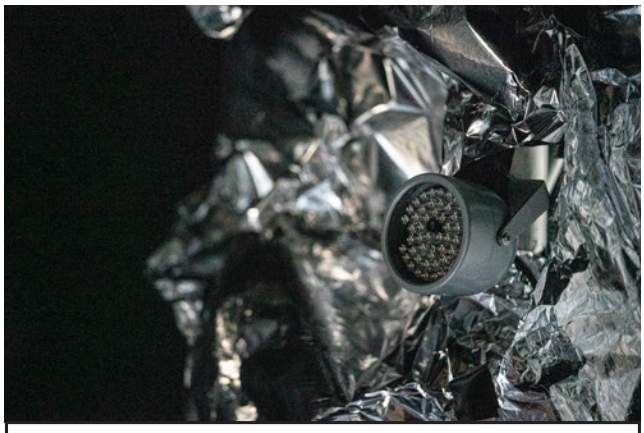


Abb. 5: Infrarotlampe im VR-Spielbereich

3.3.2 Wahl der Engine und des VR-Frameworks

Um den virtuellen Teil des Escape Rooms sowie dessen Interaktion mit dem realen Teil zu verwirklichen, wurde die Spieleengine Unity[5] herangezogen. Die Wahl erfolgte aufgrund vorheriger Erfahrung des Teams mit der Engine, sowie angesichts des Fakts, dass sie für kleine Projekte kostenlos nutzbar ist. Um den Input des Headsets zu verwalten, wurde das Oculus Integration SDK,[6] bereitgestellt vom Hersteller Oculus, verwendet. Diese Wahl erfolgte, da keine Kompatibilität mit anderen Headsets gewährleistet werden musste und eine klare Anleitung für den Einstieg vorhanden war. Im Laufe der Entwicklung stellte sich heraus, dass die Dokumentation lückenhaft ist und andere Frameworks möglicherweise eine bessere Wahl gewesen wären – ein Wechsel zu späterer Zeit wurde jedoch nicht vorgenommen, um bereits funktionierende Elemente nicht zu gefährden. Das Framework kommt mit vorgefertigten Elementen und Beispielen, die die Verwendung von Features des VR-Headsets ermöglichen. So gibt es vorgefertigte Komponenten, um etwa aufhebbare Gegenstände („Hand Grab Interactable") und Interaktionen wie das Drücken von Knöpfen („Poke Interactable") oder das Verschieben von Elementen auf einer vorgegebenen Strecke ermöglichen. Teilweise mussten diese jedoch für das gewünschte Verhalten modifiziert werden.

3.3.3 Interaktionen

Alle Interaktionen innerhalb des VR-Teils beschränken sich auf das Greifen und Bewegen von Objekten mit Hilfe des dedizierten „Greifen"-Buttons im unteren Teil des Oculus-Controllers, das „Berühren" von Elementen, indem eine sichtbare Interaktionssphäre um die Hand herum in Kontakt mit einem Objekt gebracht wird, und das „Hineindrücken", ähnlich eines großen Knopfes, das im weiteren Verlauf in Anlehnung an das entsprechende Script aus dem Oculus Framework „Poke" genannt wird. Es wurde bewusst auf die Verwendung weiterer Controllerbuttons verzichtet, um das Erlebnis möglichst intuitiv verständlich, auch für weniger VR-erfahrene Personen zu gestalten. Die Benutzung des „Greifen"-Buttons anstelle des „Triggers", der sich weiter oben am Controller in Höhe des Zeigefingers befindet, hat dabei jedoch zeitweise Irritationen ausgelöst und in weiteren Iterationen sollte dieser zusätzlich zu jenem eine „Greifen"-Interaktion ausführen.

3.4 Generelle Bemerkung zum physischen Raum (P-Raum)

Ein großer Teil der Arbeit war die Fertigung der physischen Elemente. Hierbei wurde der Aufwand unterschätzt, was dazu führte, dass die Tests zum Netzwerk und dem Zusammenspiel der Elemente zu kurz kamen. Die Konzeption des Raumes lies es jedoch nicht zu, die Elemente einfacher zu bauen, und völlige Streichungen hätten den Raum zu sehr reduziert.

Ein nicht unerheblicher Teil des Aufwands entfiel auf die Aufbauten im Raum selbst. Eine große Holzkonstruktion für die Tür mit Magnetschloss und die herablassbare Wand wurde gebaut und ein Tor aus Traversen errichtet, das durch ein großes Stück weißen Molton den Hauptrechner sowie das Audiosystem versteckte, den Raum verdunkelte und außerdem als Aufhängung für den Beamer diente, der den Lichtpunktes projizierte (siehe Teil 3.7).

---

[5] Verwendet wurde die Long-Term-Support (LTS)-Version 2020.3.28f1, die zu Beginn des Projekts die aktuellste LTS-Version war. Unity ist kostenlos erhältlich unter: https://unity.com/ (zuletzt überprüft am 07.09.2022).

[6] Kostenlos im Unity Asset Store verfügbar. Dokumentation unter:
https://developer.oculus.com/downloads/package/unity-integration/ (zuletzt überprüft am 07.09.2022).

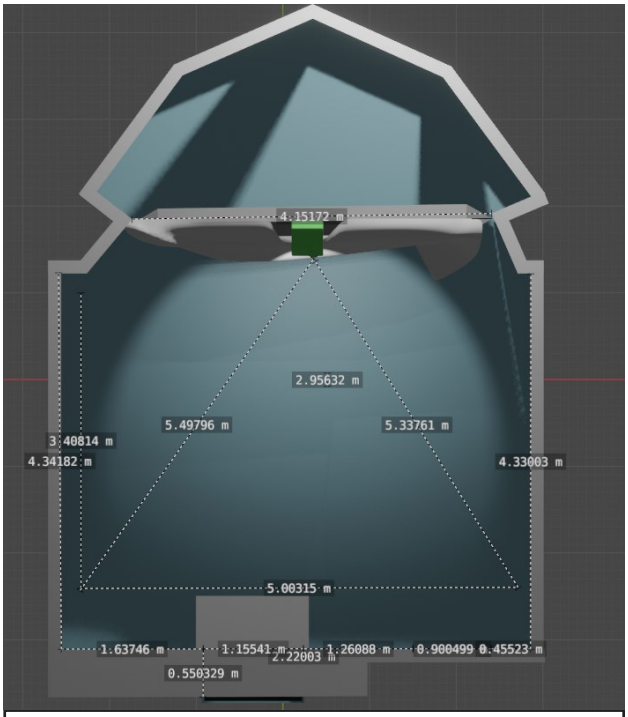


Abb. 6: 3D Modell von E45 mit Originalmaßen

Auf dem Virtuality Continuum von Milgram und Kishino ist der P-Raum weit auf der Seite der Realität einzuordnen. Möglicherweise müsste er als völlig real betrachtet werden, da das einzige medial dargestellte Element die Darstellung der VR-Person ist, die ja nicht simuliert, sondern „sampled and then resynthesized" [21] ist, was eine hinreichende Bedingung für die Realität der dargestellten Sache ist. Handelte es sich um ein Video der VR-Person, würde es wohl kaum in Frage kommen, das Display, das der Boden des P-Raums durch den Projektor darstellt, als Mixed Reality Display anzusehen. Da die Position auf dem Boden jedoch nur per Proxy der Position der VR-Person entspricht (so könnte sie etwa durch Verschieben in der Gameengine in Relation zu der Repräsentation des Raums verändert werden) ist es zumindest zweifelhaft, ob es sich nicht doch um eine Art Augmented Reality handelt: Der Lichtpunkt auf dem Boden hat simulierte Aspekte und wäre somit zumindest teilweise als virtuelles Objekt zu betrachten. Dennoch macht es Sinn, diesen Raum im Sinne der Collaborative Virtual Environments zu betrachten, da definitiv Kollaboration zwischen verschiedenen Punkten auf der MR-Skala besteht.

3.5 Lidarsensor und Pointcloud

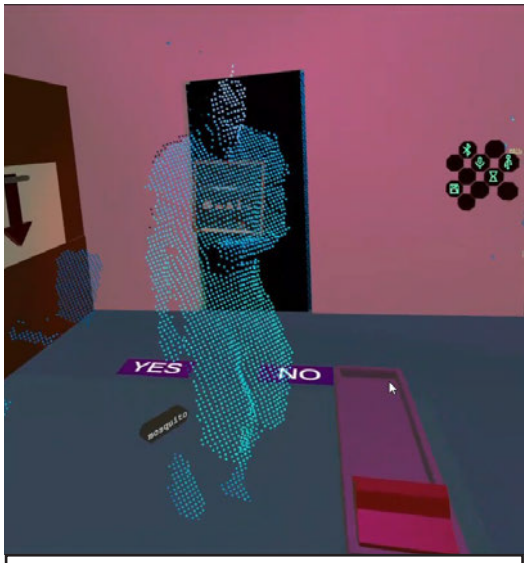


Abb. 7: Pointcloud aus Sicht der VR-Person

Die Punktwolke (Pointcloud), die zum Darstellen der P-Personen im VR-Raum diente, stand ganz am Anfang der Entwicklung. Bei der Mitarbeit im Bachelorprojekt Jonas Plümkes, bei dem eine Theatererperformance per VR einem Remote-Publikum zugänglich gemacht wurde [13], wurden bereits Erfahrungen mit RGBD-Sensoren von Intel und den dazugehörigen Software-Frameworks gemacht. Unter der Marke RealSense vertreibt Intel eine Reihe solcher Sensoren und stellt auch eine Anbindung an die Unity-Engine bereit. Aufgrund der höheren Reichweite und Auflösung wurde für XcapeRoom statt der auf Stereoskopie basierenden RealSense D435i das Modell L515 gewählt, das die Tiefeninformation per Lidar[7] sammelt. Zunächst wurde in einer kurzen Probephase die Art der Implementierung in Unity erprobt. Intel gibt eine Frameauflösung von bis zu 1024x768 Pixeln bei einer Tiefengenauigkeit von 5-14mm an. In der Erprobung waren diese Werte ausreichend, um eine Punktwolke in Unity zu erzeugen, in der sogar die Gesichtszüge und Mimik der Testperson erkennbar waren. Daher fiel die Entscheidung, statt,

[7] Während bei Stereoskopie die Tiefeninformationen ähnlich dem menschlichen Sehapparat, durch die Verschie-bung zwischen zwei Blickpunkten errechnet werden, werden sie bei einem LIDAR-System, wie es beispielsweise auch für KI-gesteuerte Fahrzeuge verwendet wird, durch ein Raster aus Infrarot-Laserstrahlen gesammelt, bei dem von jedem Punkt die Zeit zwischen Schicken des Signals und Ankunft des reflektierten Lichts gemessen und so die Entfernung relativ genau bestimmt wird.

Abb. 8: Intel RealSense L515 an einem Magic Arm an der Traverse

wie bei dem früheren Projekt, wo mit Skeleton-Tracking gearbeitet wurde, die Pointcloud-Darstellung für die P-Personen zu nutzen, um eine naturalistischere Darstellung zu erzielen, bei der die VR-Person ihre Mitspieler*innen auch wiedererkennen kann. Zudem gab es beim vorigen Versuch immer wieder Tracking-Aussetzer, die bei der Pointcloud-Darstellung nicht passieren, da diese eine quasi rohe Darstellung der Sensordaten ist. Außerdem bestanden Befürchtungen, dass zwei Person gleichzeitig zu tracken, die Trackingprobleme noch verschlimmern könnte, zumal es sich nicht vermeiden ließe, dass die Personen sich zweitweise überlappen – die VR-Person würde im Gegensatz zu einem Computer in der Lage zu sein, bei teilweiser Überlagerung der Personen in der Pointcloud diese weiterhin auseinanderzuhalten. Zu diesem Zeitpunkt bestand darüber hinaus noch der Plan, die Anwendung nativ auf der Oculus Quest auszuführen und die vorhandenen Tracking-Softwares ließen sich nicht ohne Weiteres für Android kompilieren. Um die technischen Anforderungen, sowohl an die Hard- als auch die Software, im Rahmen zu halten, wurde darauf verzichtet, die RGB-Daten der Kamera zu integrieren. Auf diese Weise konnte das Pointcloud-Script verwendet werden, das Teil der Unity-Integration ist, die von Intel bereitgestellt wird. In späteren Versuchen wäre die Nutzung der RGB-Daten aber durchaus in Erwägung zu ziehen. Man würde dadurch einen 3D-Video-Avatar erhalten, ähnlich der Arbeit von Matthes et al. [17]. Um Objekte, die ebenfalls vom Sensor erfasst werden, ausblenden zu können und so die Personen klarer hervorzuheben, wurde ein Processing Block (s. Anhang 5 S. xvi) geschrieben, mit dem Punkte innerhalb eines definierten Quaders von der weiteren Berechnung ausgeschlossen werden konnten. Außerdem wurde das Script zur Darstellung der Pointcloud leicht verändert, um eine Messung der Anzahl der Punkte, ebenfalls in gedachten Quadern (Unity Collider), zu ermöglichen, was für das Pressure-Plates-Rätsel relevant war. Um diese Berechnungen zu vereinfachen, wurde die Auflösung der Pointcloud reduziert – da dies, in Verbindung mit einem der mitgelieferten Shader, der die Punkte in Abhängigkeit ihrer Entfernung zur Kamera in unterschiedlich hellen Blautönen erscheinen ließ, auch ästhetisch interessant war und an Science-Fiction-Hologrammdarstellungen erinnerte, die zum Theme des Raumes passten, wurde diese Reduzierung auch für die Darstellung beibehalten. Neben der Einbindung des RGB-Feeds liegt hier das größte Potenzial zur Erhöhung der Visual Fidelity der Pointclouddarstellung.

Beachtlicher Aufwand floss in die Recherche dazu, wie sich eine kabellose Übertragung der Pointcloud vom P-Raum auf die Oculus Quest bewerkstelligen ließe, da hierfür keine vorgefertigte Lösung existierte. Es gibt zwar eine Lösung für die direkte Integration von Intel Realsense auf der Oculus Quest,[8] doch diese sieht eine direkte Kabelverbindung vor, die im Fall von XcapeRoom ungeeignet war. Der vielversprechendste Ansatz für das Streamen der Tiefeninformation war ein vorhandenes Repository für die Hardware-Encodierung des RGBD-Streams.[9]

Da jedoch keine Lösung für das Decodieren des Streams auf der Android-basierten Oculus Quest Hardware existierte, und dem Team die nötige Fachkenntnis für diese Art der Programmierung fehlte, wurde von dieser Lösung abgesehen, um stattdessen die Pointcloud auf dem Hauptrechner laufen zu lassen und die ganze Anwendung per Oculus AirLink (siehe Kapitel 3.4.1) zu streamen. Hier besteht die Möglichkeit zur Weiterentwicklung, um ein vielseitigeres und stabileres System zu schaffen.

[8] Github repository: https://github.com/GeorgeAdamon/quest-realsense [7].

[9] Github repository: https://github.com/bmegli/realsense-network-hardware-video-encoder [2].

Beim ersten Aufbau im Mai 2022 fiel der Lidar-Sensor während der Benutzung mit einem kritischen Fehler aus und ließ sich nicht mehr betreiben. Eine kurze Internetrecherche deutete darauf hin, dass auch andere User auf dieses Problem gestoßen sind. Die Produktion und der Softwaresupport für den Sensor wurden von Intel bereits im Februar 2022, kurz nach der Markteinführung im vierten Quartal 2019, [9] wieder eingestellt. [10]

3.6 Netzwerk

Die genaue Struktur des notwendigen Netzwerks wurde beim ersten Aufbau durch sukzessives Einbauen der Bestandteile erarbeitet. Dabei wurde vorhandene Infrastruktur der Hochschule genutzt. Im Mittelpunkt des Netzwerks stand der Hauptrechner, auf dem die Unity-Anwendung lief. Er befand sich im P-Raum und war mit einem Router verbunden, der alle weiteren Verbindungen innerhalb des Raums steuerte und weiterhin über eine Ethernetverbindung mit einem weiteren Router im VR-Raum verbunden war. Dieser zweite Router wiederum stellte die Drahtlosverbindung zur Oculus Quest her. Um eine ausreichende Bandbreite für die AirLink-Verbindung zu erreichen, wurde, wie von Oculus spezifiziert, das 5GHz-Band für diese Verbindung verwendet. Außer mit der Oculus, war der zweite Router nur mit dem Smartphone der Spielleitung verbunden, die damit über OSC-Messages[10] die Spielsequenz einleiten, die Position der VR-Person adjustieren und als Backup das Lichtschalter- oder das Digital-Clean-Up-Signal schicken konnte. Der Router im P-Raum war außer mit dem Hauptrechner und dem zweiten Router noch mit einer Hue Bridge per Kabel verbunden. Diese ist Kernbestandteil des Hue-Systems von Philips, das die Ansteuerung mehrerer Smart-Lampen über das Netzwerk ermöglicht. Auch hierfür existiert eine vorgefertigte Implementation in Unity. Per WiFi war der Router darüber hinaus mit allen WEMOS D1-Modulen,[11] die zur Ansteuerung der verschiedenen Elemente im Raum verwendet wurden, verbunden und tauschte Daten über OSC aus. Der im Hochschulraum fest installierte Präsentations- und Videokonferenzrechner wurde zweckentfremdet und ebenfalls per WiFi mit dem Router verbunden. Über diese Verbindung empfing er einerseits OSC-Signale, um per VVVV[12] verschiedene Videos abzuspielen, und sendete andererseits über Voicemeeter[13]

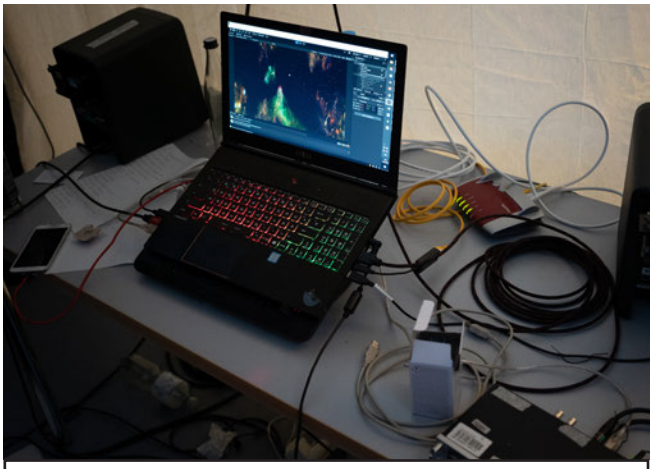


Abb. 9: Der Hauptrechner

[10] OSC (Open Sound Control) ist ein sehr flexibles Netzwerkprotokoll, das es ermöglicht, unkompliziert kleine Datenpakete per UDP zu verschicken. Entsprechende Apps für Smartphones und fertige Libraries sowohl für Arduino als auch Unity sind ohne Weiteres auffindbar. Da die Datenpakete frei definierbar sind, eignet es sich für die schnelle Ansteuerung einfacher Elemente, wie sie in XcapeRoom oft vorkommen und ist dabei vielseitig genug, um etwa die Position des Schlittens im KIBO, oder auch die nötige Spannung für das Herablassens der Fake-Wand zu übermitteln. Ein Nachteil ist, dass per UDP im Gegensatz zu TCP kein Handshaking stattfindet und so kein Schutz vor Datenverlust oder anderen Unzuverlässigkeiten wie doppelte Übermittlung oder falsche Reihenfolge der Datenpakete besteht. Für kritische Elemente, wie etwa die Fake-Wand, die genau einmal ihr Signal erhalten müssen, wurde daher in der Software ein Frage-Antwort-Mechanismus eingebaut.

[11] Beim Wemos D1 handelt es sich um ein Entwicklerboard mit einem ESP8266 Mikroprozessor. Dieser ist mit ei-nem WLAN-Modul ausgestattet und kann über das Entwicklerboard per USB-Anschluss über die Arduino-Plattform programmiert werden.

[12] VVVV ist eine visuelle Programmierumgebung ähnlich etwa MAX-MSP, die die einfache Ansteuerung von Medi-en ermöglicht. Sie ist für nicht-kommerzielle Zwecke kostenfrei nutzbar und zu finden unter folgender Domain: https://vvvv.org/

[13] Voicemeeter ist ein virtueller Audiomixer von VB-Audio mit eingebauter Netzwerkfunktion, die das Streamen von Au-

einen Audiostream vom im Raum verbauten Konferenzmikrofon an den Hauptrechner, zur Verwendung als Raumsound in der VR-Umgebung. Das einzige Element, das nicht per WiFi angesteuert wurde, war der LED-Wand-Tresor, der per Serial Port direkt über USB mit dem Hauptrechner verbunden war.
Unabhängig vom Hauptrechner bestand zudem eine Verbindung zwischen dem zuvor erwähnten Präsentationsrechner und einem weiteren Rechner im VR-Raum, der nur der Überwachung des Spielgeschehens im P-Raum diente. Hierfür wurde aus Praktikabilitätsgründen ein Microsoft Teams Videoanruf über die Internetanbindung der Hochschule zwischen den beiden Rechnern getätigt.
Nicht über ein tatsächliches Netzwerk, aber dennoch über das OSC-Protokoll lief zusätzliche Kommunikation zwischen Unity und einem VVVV-Programm auf dem Hauptrechner, das für die Darstellung des Lichtpunkts (siehe folgendes Unterkapitel) und das Abspielen von Soundsamples über das Soundsystem zuständig war.

Es folgt eine grafische Darstellung der Netzwerkstruktur. Der Vollständigkeit halber enthält sie auch die Verbindung zum Soundsystem (per USB am Hauptrechner), Beamer (HDMI am Hauptrechner) und die unabhängige Verbindung zwischen PolRev und Tür (per Klinkenkabel, hergestellt durch die Spieler*innen).

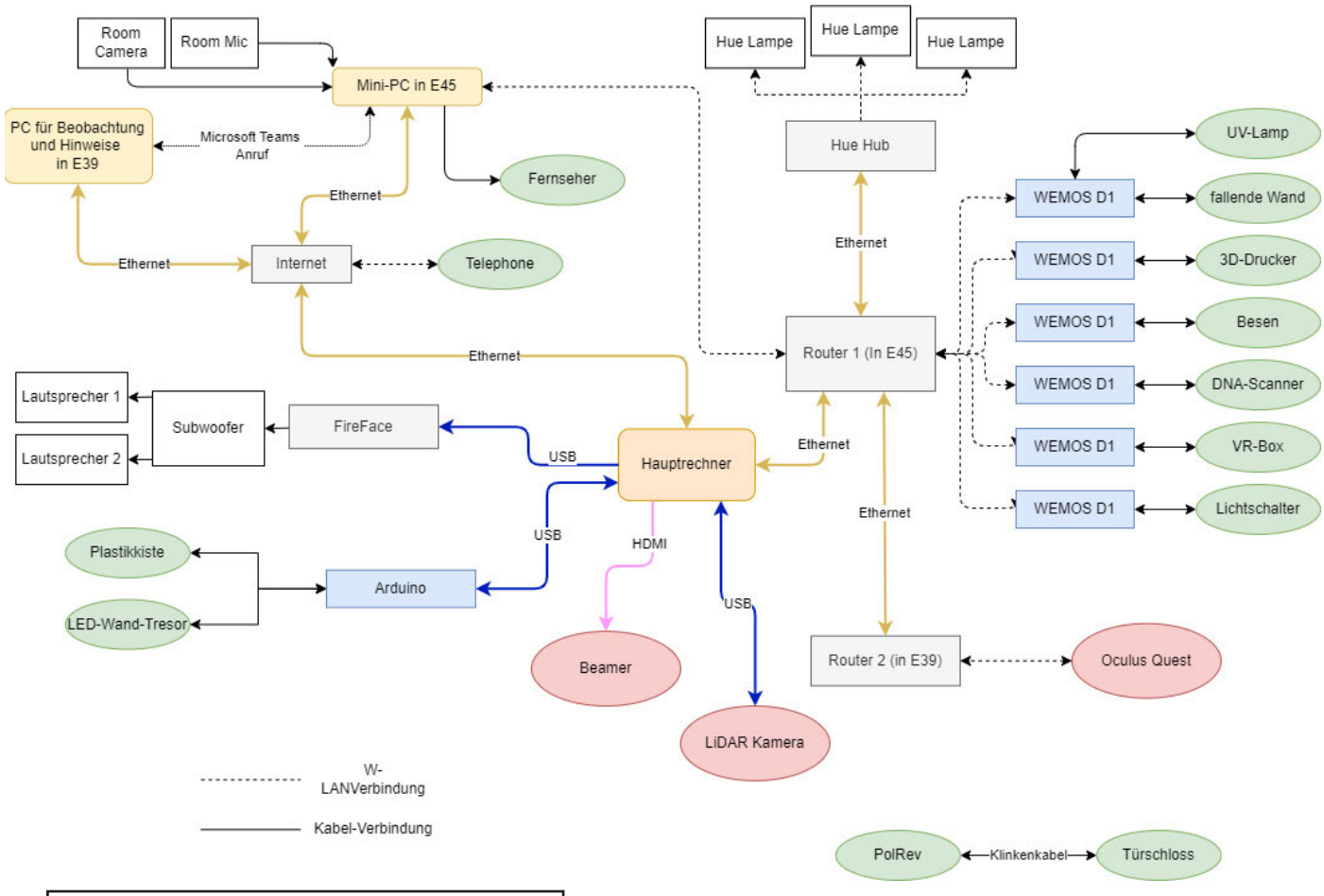


Abb. 10: Schematische Darstellung der Netzwerkverbindungen

diosignalen innerhalb eines LAN ermöglicht. Für nicht-kommerzielle Nutzung ist es ebenfalls kostenfrei erhältlich unter: https://vb-audio.com/Voicemeeter/
Ergänzend wurde auch die Software Virtual Audio Cable vom selben Anbieter verwendet. Mit ihr ist es unkompliziert möglich, durch einen virtuellen Audio-Input am Computer eine beliebige Audioquelle in Unity als (virtuelles) Mikrophon zu verwenden.

3.7 Lichtpunkt

Abb. 11: Beamer, aufgehängt an der Traverse in E45

Als Repräsentation der VR-Person im P-Raum wurde, in Anlehnung an das frühere Theaterprojekt [13] ein einfacher Lichtpunkt gewählt, der durch einen Beamer auf den Boden projiziert wird. Dies geschah aus zwei Überlegungen heraus: Zum einen hatte der Schauspieler aus dem früheren Projekt berichtet, dass die Lichtpunkte in ihrer Bedeutung als Zuschauer ihm das Gefühl gegeben hatten, für ein Publikum zu spielen [13] – sich also eignen, um die Präsenz eines Menschen darzustellen. Zum anderen sollte es Teil der Escape-Room-Erfahrung sein, dass die P-Personen herausfinden müssen, was der Lichtpunkt ist. Eine weniger abstrakte Darstellung wäre hier womöglich abträglich gewesen.

Im Lauf der Entwicklung wurde zusätzlich zur Spieler*innenposition noch die Position der Handcontroller als eigene Punkte der Lichtpunkt-Repräsentation hinzugefügt. Dadurch sollte eine weitere Interaktionsmöglichkeit für die VR-Person geschaffen werden und zusätzlich eine visuell interessantere Darstellung erreicht werden, ohne den Abstraktionsgrad dramatisch zu verringern. Diese Anpassung stellte sich als sehr gewichtig heraus, da sie dem Lichtpunkt plötzlich eine sehr viel lebendigere Anmutung gab. Diese Einschätzung entstammt den Beobachtungen des Autors und bedarf weiterer Untersuchung. Auf der VR-User-Seite lässt sich auf jeden Fall feststellen, dass durch Tracking der Gliedmaßen, besonders der Hände, die Präsenz stark verbessert [19] wird. Auf der Seite der P-Personen lässt sich sagen, dass es zumindest naheliegt, dass die zusätzliche Dimension der, wenn auch abstrakten, Gesten wahrscheinlich dazu beiträgt, die soziale Präsenz zu erhöhen, da solche (zusammen mit anderer, hier nicht dargestellter nonverbaler Kommunikation wie Mimik) eine wichtige Rolle in der Erzeugung sozialer Präsenz spielen [23].

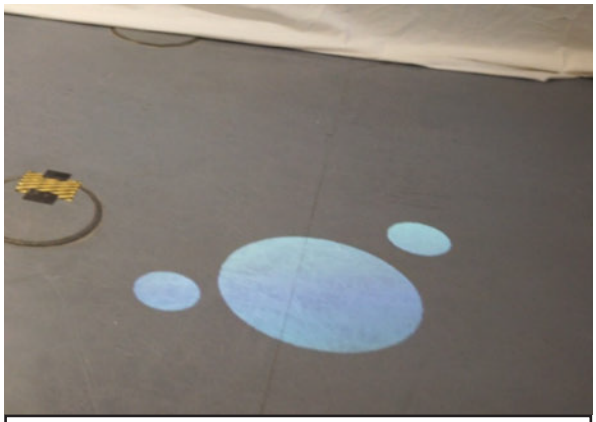


Abb. 12: Der Lichtpunkt, inkl. zwei kleiner Handpositionspunkte

Die technische Umsetzung erfolgte über ein einfaches VVVV-Programm, das per OSC-Message 30-mal die Sekunde die x- und y-Koordinaten der VR-Kamera sowie der beiden Handcontroller empfing. Um die Abbildung der Dimensionen des VR-Raums auf den P-Raum zu gewährleisten, wurde der Nullpunkt des Koordinatensystems in der Mitte des virtuellen Raumes gesetzt und ebenfalls in der Mitte des realen Raumes der Ursprung des projizierten Punkts definiert. Nach einer Keystone-Entzerrung, um eine plane Projektion zu ermöglichen, wurden die Maxima des Koordinatensystems festgelegt, indem sich eine Testperson mit dem Headset im P-Raum zu den vier Kanten der Projektionsfläche begab (die unbeabsichtigt und lediglich aufgrund der geometrischen Gegebenheiten und den Spezifikationen des verfügbaren Beamers den Boden des Raumes nur unvollständig abdeckte) und die gesendeten Koordinaten als die Endpunkte entlang der entsprechenden Achse definiert wurden. Nach dieser Erstkalibrierung erfolgte eine Kontrolle über die Pointclouddarstellung. Die Test-Person ging in den anderen Raum und eine zweite Person stellte sich im realen Raum auf den Lichtpunkt. Durch einen Offset, der für beide Achsen einstellbar war, wurde die Position des Lichtpunkts so lange adjustiert, bis in der VR die auf dem Lichtpunkt stehende Person die Testperson überlappte.

3.8 Erster Prototyp

Der ursprüngliche Plan für diese Arbeit war es, Spielende zu interviewen und die Vor- und Nachteile des Escape Rooms in einer Iteration zu analysieren. Während des geplanten ersten kompletten Durchlaufs des Prototyps versagte der Intel Realsense Sensor allerdings plötzlich und unerwartet den Dienst. Zusätzlich zeichneten sich einige Probleme mit dem Netzwerk und dem Aufsetzen des Spiels für die nächste Runde ab. Das größte Netzwerkproblem war, dass die Verbindung zwischen den zwei Räumen, die in den vorherigen Tests verlässlich funktioniert hatte, wiederholt abbrach, was dazu führte, dass die VR-Person wieder in den P-Raum gebracht wurde, wo das notwendige Netzwerk des VR-Headsets funktionierte. Da es zur Zeit der Aufführung keine Möglichkeit gab, den Lidar Sensor zu ersetzen und da die Pointcloud, die damit erzeugt wurde, essentiell für das Experiment war, wurde den eingeladenen Gruppen eine Präsentation des Konzepts und der funktionierenden Elemente angeboten und andere, kleinere Probleme in den Zeiten zwischen den Aufführungen repariert. Während die Netzwerkprobleme sich als unerklärlich erwiesen und zum Teil in den folgenden Tagen von selbst verschwanden, konnte für die anderen Elemente eine viel stabilere Funktion gewährleistet und eine besser geeignete Setup-Prozedur zur Minimierung von Problemen während des Durchlaufs entwickelt werden. Am letzten Tag war es machbar, einer Gruppe einen nahezu stabilen Durchlauf zu ermöglichen, wobei es zwar keine Pointcloud gab, aber alle anderen Verbindungen wie vorgesehen funktionierten. Die Probleme, die mit den einzelnen Elementen auftraten, bzw. dafür angedachte Verbesserungen werden in Abschnitt 5.2 näher behandelt.

4 Die Rätsel

Der Hauptbestandteil eines Escape Rooms sind die Rätsel. In diesem Kapitel werden die Rätsel von Xcape-Room detailliert beschrieben und analysiert. Für die Analyse wird die atomare Taxonomie von Krekhov et al. genutzt, um die Rätsel in ihre Bestandteile aufzuteilen und zu klassifizieren. Diese Taxonomie dient unter anderem dazu, die Verteilung der verschiedenen Aufgaben innerhalb eines Escape Rooms zu analysieren und etwaige Lücken zu identifizieren und soll nach Intention der Autor*innen allgemeingültig für alle Arten der Escape Rooms anwendbar sein, was es ideal für die Anwendung eines XR-Escape Rooms macht, der verschiedene Modalitäten gleichzeitig bedient. In der folgenden Beschreibung werden die englischen Originalbegriffe verwendet. Krekhov et al. identifizieren drei Arten von Herausforderungen (Challenges): Mental Challenges, Physical Challenges und Emotional Challenges [15]. Diese wiederum enthalten eine größere Anzahl von Unterkategorien, deren Namen größtenteils selbsterklärend sind, weshalb hier auf eine genaue Erörterung verzichtet wird. Bei der Beschreibung der Rätsel wird jeweils begründet, warum die Bestandteile der jeweiligen Kategorie zugeordnet werden. Jedes Rätsel besteht dabei aus einem Set von Challenges.

Welche Bestandteile des Escape Rooms genau als Rätsel definiert werden, ist bei diesem Ansatz offen. Man könnte nach ihrem System jede Handlung, die innerhalb des Escape Rooms stattfindet, als Rätsel klassifizieren, da kein Mindestlevel der Herausforderung etwa für Aufgaben, bei denen Objekte bewegt werden müssen (*Object Movement/Alignment*), vorausgesetzt wird. Nicholson beschreibt in Unterscheidung zu Rätseln „Tasks" als Tätigkeiten, bei denen die Spieler*innen wissen, was zu tun ist und es nur ausführen müssen [22]. Als Beispiel führt er ein Laserlabyrinth an, das bei Krekhov et al. definitiv unter die Klassifikation *Self Motion* fallen würde. Für dieses Kapitel wurde im Sinne der Vereinfachung beschlossen, dass triviale Aufgaben, wie etwa das Einwerfen eines Plastikröhrchens nach Anleitung in ein großes Loch, als Tasks klassifiziert und nicht als Rätsel in ihre Bestandteile zerlegt werden.

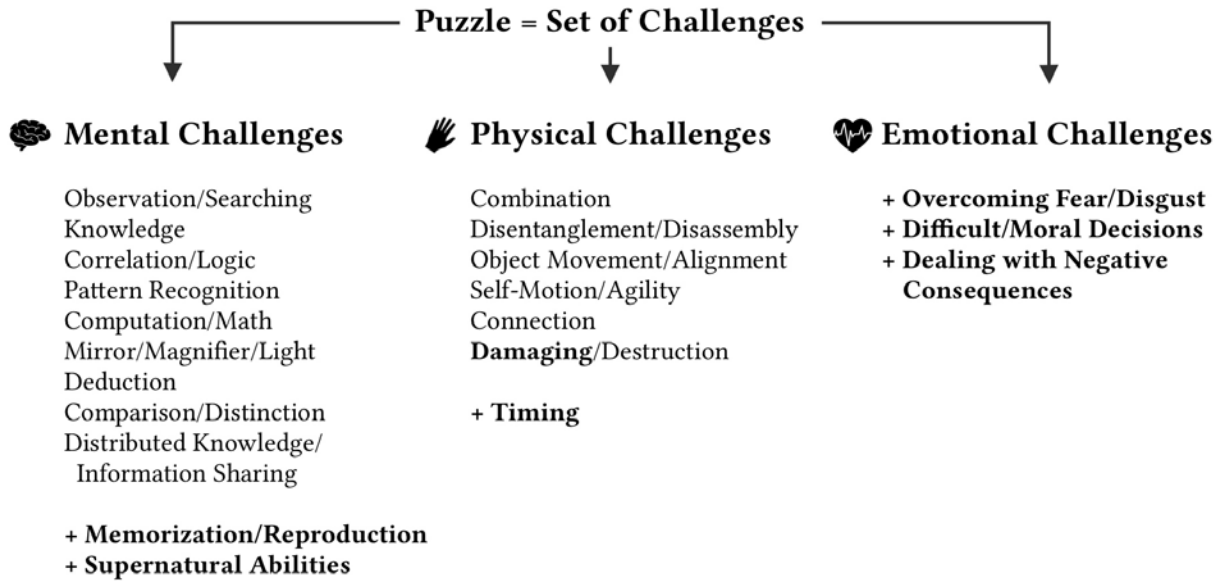


Abb. 13: Taxonomie der Challenges nach Krekhov et al. [2020]

Die Übergruppe aller Elemente ist ihr entsprechendes Rätsel oder Task, unter dem Rätsel/Task werden die einzelnen Bestandteile und ihre konkreten Funktionen und Effekte beschrieben, bevor das Rätsel an sich betrachtet und mit dem atomaren Modell von Krekhov et al. analysiert wird. Zum Teil werden auch Elemente beschrieben, die weder Rätsel noch Tasks sind, sondern nur Bestandteile der Rätsel, die aber sonst an keiner Stelle Erwähnung finden und der Vollständigkeit halber hier aufgeführt werden. Das ganze Kapitel gliedert sich dabei in drei Teile, wobei der erste Teil „rein physische" Rätsel behandelt, was bedeutet, dass ihre Lösungen allein im P-Raum vollzogen werden, auch wenn sie eine Auswirkung auf die VR haben mögen. Es folgen „rein virtuelle" Rätsel, die ohne Input von den P-Personen allein im VR-Raum gelöst werden, ebenfalls ohne Auswirkungen auf die Realität auszuschließen. Im letzten Teil werden die „Mixed-Reality-Rätsel" angeschaut, bei denen sowohl Input von der VR als auch der Realität gebraucht wird, um sie zu lösen.

4.1 Rein Physisch

4.1.1 VR-Box-Rätsel

Die VR-Box ist lediglich eine kleine Kiste, die mit einem Netzteil mit dem Strom verbunden wird und so einen Arduino und ein 4er-Set farbiger LEDs betreibt. Im regulären Modus leuchtet die Box grün, bekommt sie ein Signal vom Hauptrechner, dass der Alarm aktiv ist, beginnt sie, rot zu blinken. Trennt man die Box vom Strom und verbindet sie danach wieder, wechselt die Box wieder in den regulären Modus und der WEMOS D1 sendet eine Nachricht an den Hauptrechner, woraufhin der Alarm gestoppt wird.
Es handelt sich hierbei um ein Rätsel der Art *Disentanglement/Disassembly*, wobei es absichtlich ausgesprochen einfach gehalten ist. Die Beschäftigung mit der Box soll lediglich etwas Puffer einbauen, während die VR-Person in einen anderen Raum gebracht wird und eine Kooperation zwischen den Spielenden beginnen kann.

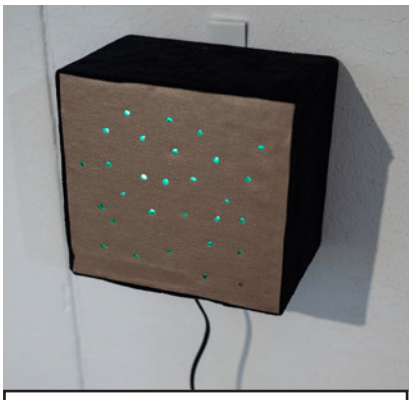


Abb. 14: VR-Box in grünem „Ruhezustand"

4.1.2 Lichtschalter-Rätsel

Abb. 15: Kleine Pappkiste mit falschem Lichtschalter

Dieses Element hat eine Doppelfunktion: Zum einen ist damit der tatsächliche Lichtschalter des Raumes verdeckt und somit nicht-verwendbar gemacht und zum anderen befindet sich in der Box, auf der der Lichtschalter montiert ist, ein mit einem 9V-Block betriebener WEMOS D1, der bei Betätigung des Schalters eine Nachricht an den Hauptrechner schickt, die **Codewolke** zu entfernen und mit dem VR-Lichtschalter-Rätsel zu ersetzten.

Man könnte den Lichtschalter nach Krekhov et al. als *Observation/Searching* Rätsel klassifizieren, da der Lichtschalter erst gefunden werden muss. Da er sich an der üblichen Stelle für Lichtschalter, neben der Eingangstür, befindet, und die Spieler*innen so ihr Vorwissen über die Welt nutzen können, könnte man auch von einem *Knowledge*-Rätsel sprechen, oder von einem *Logic/Correlation*-Rätsel, wobei die Logik ist, dass auf Drücken des Lichtschalters das Angehen des Lichts folgt, was in diesem Fall aber nicht passiert. Es mag fraglich erscheinen, ob das Schalten eines Lichtschalters überhaupt als Rätsel gewertet werden kann, doch aus den praktischen Erfahrungen zeigt sich, dass nicht alle Teilnehmenden unmittelbar den Schalter drückten. Hierzu folgt eine kurze Diskussion im Auswertungsteil 5.2.

Codewolke

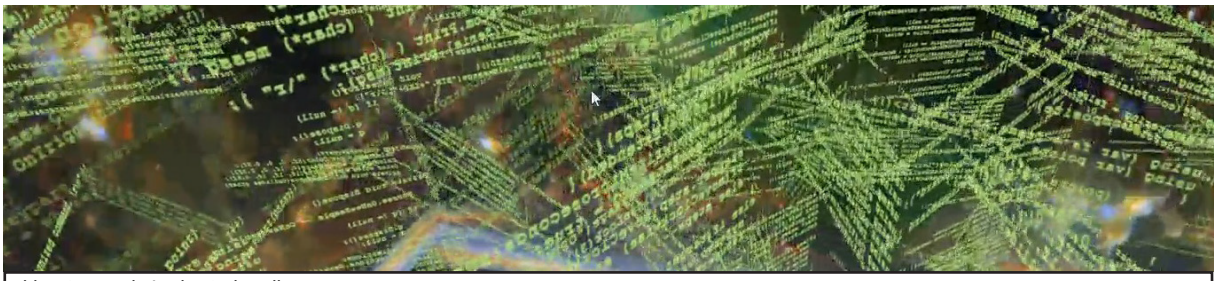


Abb. 16: Ausschnitt der Codewolke

Die Codewolke ist ein rein visuelles Element ohne Interaktionsmöglichkeiten. Sie besteht aus einer Partikel-Effekt-Wolke aus grünen Code-Schnipseln. Die VR-Person beginnt das Spiel sitzend im Rollstuhl und wird durch einen Gang zu einem zweiten Raum gebracht. Dabei war zunächst geplant, eine Animation eines Tunnels zu zeigen, durch den die Person „fällt", um das „in die VR gesaugt werden" darzustellen. Aufgrund zeitlicher Beschränkungen konnte dies nicht verwirklicht werden. Daher befindet sich die Person von Anfang an in der Codewolke, die eigentlich erst nach diesem Tunnel erscheinen sollte. Durch das Tracking des VR-Headsets zwischen den Räumen bewegt sich die VR-Person relativ weit und verlässt so die Codewolke. Beim Zurücksetzen der Position bei Erreichen des Raums kann erfragt werden, ob die VR-Person sich in der Codewolke befindet, und so verifiziert werden, dass die Ursprungsposition erreicht wurde.

4.1.3 DNA-Scanner-Task

Abb. 17: DNA-Scanner mit Knopf, Einwurfloch und Röhrchenhalter

Der DNA-Scanner besteht aus 2 Teilen: Einem im Raum fest installierten Minicomputer und seinem Display sowie einer darunter stehenden selbstgebauten Tisch-Konstruktion. Auf dem Computer läuft VVVV, welches vom Hauptcomputer OSC-Messages erhält, um zu bestimmten Zeiten bestimmte Videos anzusteuern, die auf dem Display laufen sollen. Das betrifft auch das Video, das ganz zu Anfang läuft und Bilder von Gewitter und Blitzen zeigt. Zu diesem Zeitpunkt ist der DNA-Scanner noch inaktiv. Er aktiviert sich in dem Moment, in dem in der VR der Lichtschalter betätigt wird. Diese Aktion löst ein Video aus, bei dem ein blinkender Pfeil mit der Aufschrift „PRESS" auf einen Button an einer Konstruktion unter dem Display verweist. Wird der Button nun gedrückt, löst das ein weiters Video aus, in dem erklärt wird, wie der DNA-Scanner zu benutzen ist: Ein Röhrchen mit einem Wattestäbchen wird aus einem Ständer genommen, mit dem Stäbchen ist nun durch Berührung mit dem eigenen Körper eine DNA-Probe zu nehmen, wonach das Stäbchen wieder zurück in das Röhrchen und dann alles zusammen in ein Loch über dem Button geworfen werden soll. Dies ist für beide Personen im Raum durchzuführen.

Im Kern der Konstruktion befindet sich eine Waage, die wahrnimmt, wenn ein Röhrchen eingeworfen wird. Die Waage ist mit einem WEMOS D1 verbunden, der beim Eingang der „DNA-Proben" zuerst ein Video auslöst, dass den Erhalt der ersten Probe und dann den der zweiten Probe bestätigt. Nach dem Eingang der zweiten Probe wird ebenfalls eine Audionachricht abgespielt, die darauf verweist, dass die Daten in der VR verarbeitet werden müssen. Wird das Code-Rätsel in der VR gelöst, gibt es noch eine letzte Nachricht, die die Teilnehmer*innen in der Realität darauf hinweist, dass sie ab jetzt in der VR sichtbar sind. Außerdem wird in der VR die Pointcloud freigeschaltet.

Da die P-Personen hier lediglich den sehr detaillierten Anweisungen im Video folgen müssen, wird diese Interaktion als Task klassifiziert.

4.1.4 Digital-Clean-Up-Besen-Rätsel

Abb. 18: Verkabelung im Besen

Hierbei handelt es sich um einen kleinen Besen, in dessen Stiel eine LED eingearbeitet wurde, mit der Aufschrift „Digital Clean-Up". Auf der Borstenseite befindet sich ein WEMOS D1, der mit der LED, einer 6V Batterie und einem Button verbunden ist. Drückt man den Button, schickt der Arduino ein Signal an den Hauptrechner, der die **Codewand** in der VR verschwinden lässt. Außerdem leuchtet die LED in der Realität als Feedback erst lila und nach erfolgreichem Senden des Signals grün, bevor sie gänzlich erlischt. Der Besen hat ansonsten keine Wirkung auf die Realität und dient lediglich als interessanter Gegenstand, zu dem sich die Spielenden ihren Teil denken und eventuell während des Froth' mit der VR-Person

sprechen können.
Es handelt sich hierbei um ein *Observation/Searching*-Rätsel: Der Knopf muss bemerkt werden, um gedrückt werden zu können.

Codewand

Wo im realen Raum der Molton, der die Technik versteckt, hängt, befindet sich in der VR eine schwarze Wand, vor der sich eine Partikel-Effekt-Wolke aus grünen Code-Schnipseln bewegt. Durch den Digital-Clean-Up-Knopf wird diese Wand durch das Video einer ruhigen Meeresbrandung ersetzt. Und gibt ebenfalls die Sicht auf eine Passwort-Eingabemaske frei.

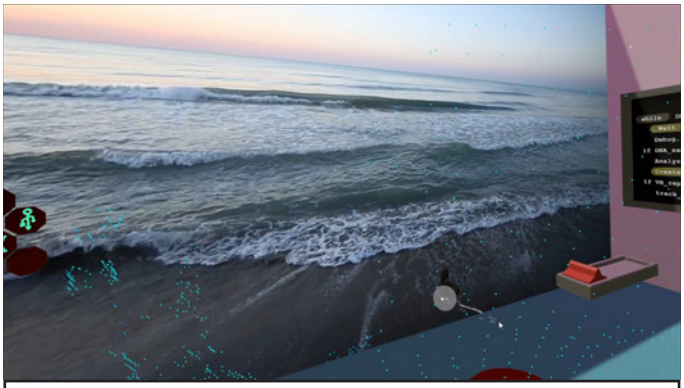


Abb. 19: Meeresbrandung nach Aufräumen der Codewand

4.1.5 PolRev-Rätsel

Kernstück und Fokuspunkt des realen Raums stellt der „Polarity Reverser" (PolRev) dar. Dieser ist eine kleine Plastikbox, die aus 3D-gedruckten Teilen und Plexiglas besteht und mit einem Kippschalter, einem großen Klinkenkabel, 3 kleinen Tastern, einem mit Widerständen versehenen Breadboard, einem Batteriefach und einer mehrfarbigen LED bestückt ist. Die Logik, die die Steuerung des Rätsels übernimmt, läuft dabei auf einem Arduino Nano Every, der aufgrund seines niedrigen Preises ausgewählt wurde. Er ist durch das Plexiglas sichtbar verbaut, um die Fiktion des prototypenhaften Smart Homes zu unterstützen. Ebenfalls im Innern des Kastens befindet sich die ansteuerbare RGB-LED (WS2812D), um visuelles Feedback zum Fortschritt des Rätsels zu geben. Um den PolRev für seinen finalen Zweck einsetzen zu können, müssen vier Schritte erledigt sein, die als Teil-Rätsel beschrieben werden:

1. Batterie-Teilrätsel

Eine Batterie muss gefunden und in die dafür vorgesehene Halterung gesteckt werden.

2. Widerstands-Teilrätsel

Auf der Oberseite des PolRevs befinden sich Widerstände, eingelötet auf einem steckbaren Breadboard. Diese müssen derart durch Steckerkabel verbunden werden, dass ihre Summe einem bestimmten Widerstandswert entspricht. Hierfür müssen die Werte der Widerstände mit Hilfe einer Tabelle herausgefunden und ausgerechnet werden, wie sie zu kombinieren sind. Die Widerstände selbst sind tatsächliche Widerstände, die allerdings für bessere Sichtbarkeit und Vereinfachung des Rätsels im zweiten Durchlauf mit Kreppband umwickelt und bemalt wurden. Die Umwicklung diente zur Vergrößerung, da die Originalwiderstände nur einen Durchmesser von etwa 1,5 mm hatten. Die umwickelte Variante kommt hingegen auf 4-5 mm Durchmesser. Das eigenhändige Bemalen gab zusätzlich die Möglichkeit, Komplikationen zu vermeiden. So haben die Originalwiderstände

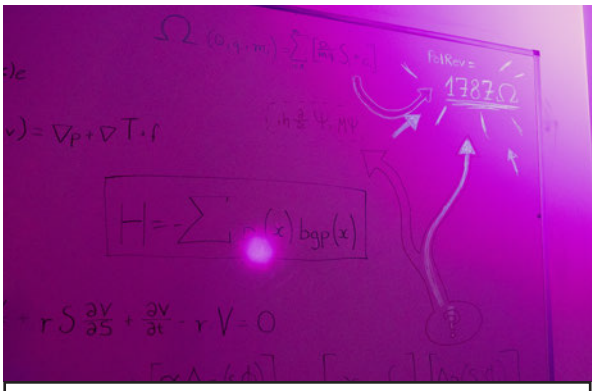


Abb. 20: Der Ohmwert in UV-Farbe unter UV-Licht

drei Streifen zur Angabe ihres Wertes, einen vierten Streifen, der als Multiplikator dient sowie einen fünften Streifen, der die Toleranz des Widerstandes angibt. Auf die letzten beiden Streifen wurde verzichtet, wodurch alle Widerstände nun lediglich dreistellige Werte haben konnten, was für den Zweck allerdings irrelevant war. Die Steckerkabel sind teilweise bereits auf einer Seite festgelötet, um das Rätsel weiterhin zu erleichtern und den Spielenden einen Anhaltspunkt zu geben. Die LED gibt optisches Feedback zum Status der Widerstände: Der PolRev leuchtet rot, solange die Widerstände nicht oder nicht richtig verbunden sind und gelb-grün sobald das Rätsel gelöst wurde.

3. Schlüssel-Teilrätsel

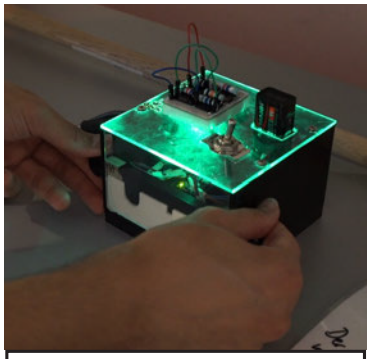


Abb. 21: Schlüssel gedrückt

An der Vorderseite befindet sich ein Schlitz, in den ein Schlüssel (aus Plastik, ebenfalls 3D-gedruckt) eingeführt werden kann, mit dem sich drei kleine Taster betätigen lassen, die in einer 3D-gedruckten Form so verbaut sind, dass nur mit dem richtig geformten Schlüssel alle drei gleichzeitig aktiviert werden können.

Passiert dies und ist der in Punkt 4 erwähnte Kippschalter korrekt gestellt, wird eine Spannung von 5V auf das mit dem Arduino verbundene Klinkenkabel angelegt (der entsprechende Pin wird auf HIGH geschaltet), die dazu geeignet ist, das Relais, das das magnetische Türschloss deaktiviert, zu schalten.

4. Schalter-Teilrätsel

Weiterhin ist der Kippschalter auf dem Deckel auf „Polarity" zu stellen. Anfangs steht dieser Schalter in seiner Mittelposition auf „OFF" und der PolRev ist tatsächlich ausgeschaltet, leuchtet also nicht. Um Feedback für das Widerstandsrätsel zu erhalten, muss der Schalter in eine der beiden Richtungen gekippt werden, hierbei ist die Polung nicht relevant. Erst wenn Punkt 3 abgehandelt wurde, spielt die Richtung eine Rolle. Ein Hinweisschild zu den Bestandteilen des PolRev weist dabei darauf hin, dass, um in die VR zu gehen, der PolRev auf „Polarity" und um sie zu verlassen auf „Reverse Polarity" gestellt sein muss. Da die P-Personen der VR-Person in die VR folgen wollen, muss also „Polarity" eingestellt sein. Ist dem so, leuchtet der PolRev nach Betätigen des Schlüssels tiefgrün, anderenfalls leuchtet er blau.

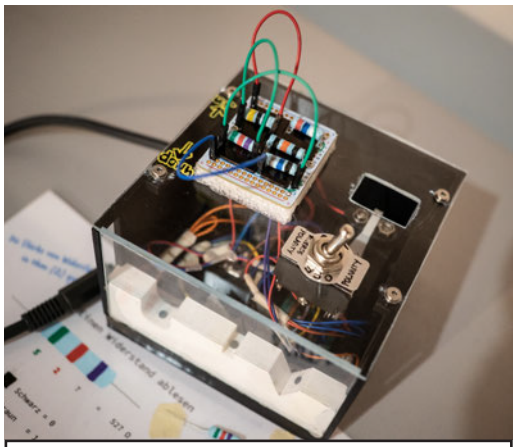


Abb. 22: Der PolRev, noch ohne Batterie und Schlüssel

Ziel des Designs des PolRev war, dass man durch bloßes Betrachten dessen ungefähr verstehen konnte, welche Teile man braucht und wozu man den PolRev benutzt. Wobei trotzdem Hinweise ausgelegt wurden, welche Teile für das vollständige Zusammenbauen des RolRevs gebraucht werden und die Klinkenbuchse neben der Tür ebenfalls beschriftet war und auf den PolRev verwies.

Im realen Raum sind die Belohnungen für das Lösen von Rätseln meist Teile oder Informationen, die zusammen mit dem PolRev benutzt werden müssen, um am Ende die Tür „in die VR" zu öffnen. Damit ist der PolRev ein Metarätsel, bei dem die Erfolge der anderen Rätsel zusammenlaufen, er enthält damit auch den Aspekt *Combination*.

Bei Teilrätsel 1 und 3 handelt es sich um rein physische Rätsel, die an dieser Stelle nur noch einen *Object Movement/Alignment*-Aspekt besitzen, da die eigentlichen Rätsel an anderer Stelle gelöst wurden. Wobei sich bei Teilrätsel 1 das Stecken in die Halterung für Menschen ohne Erfahrung mit 9V-Blockbatterien als *Combination*-Rätsel gestaltet.

Teilrätsel 2 ist sowohl physisch als auch mental anspruchsvoll und involviert die Elemente *Correlation/Logic*, da die Widerstände mit der Widerstandstabelle und dem Ohmwert am Whiteboard verbunden werden müssen, *Mirror / Magnifier / Light*, weil der Ohmwert erst durch ein Schwarzlicht sichtbar wird,[14] *Pattern Recognition*, da die Tabelle korrekt abgelesen und die Farbstreifen der Widerstände in Ziffern „übersetzt" werden müssen, *Computation/Math*, weil aus den entzifferten Widerständen ihre richtige Zusammensetzung zum Erreichen des korrekten Ohmwerts errechnet werden muss, *Connection*, da die Kabel gesteckt werden müssen, und nicht zuletzt *Object Movement/Alignment*, da eben dieses Stecken der Kabel nicht wenig Fingerspitzengefühl erfordert.

Teilrätsel 4 erfordert hauptsächlich *Correlation/Logic*, um die Informationen über den Schalter, korrekt mit dem Schalter selbst in Verbindung zu bringen und einen winzigen Teil *Object Movement/Alignment*, da der Schalter auch tatsächlich bewegt werden muss.

4.1.6 Tür-Rätsel

Vor einer der tatsächlichen Zimmertüren ist eine falsche Tür in einer OSB-Plattenkonstruktion eingehängt. Diese Tür wird über ein elektronisches Türschloss verschlossen, das an ein Relais gekoppelt ist. Wird der korrekt zusammengebaute PolRev über eine Klinkenbuchse neben der Tür an das Relais angeschlossen, wird der Stromkreis zum Schloss unterbrochen und es öffnet sich.

Die P-Personen müssen die Tür untersuchen, die Buchse finden (*Observation/Searching*) und mit dem Klinkenstecker des PolRevs mental und physisch in Verbindung bringen. Der mentale Anteil ist dabei *Pattern Recognition*, da erkannt werden muss, dass Buchse und Stecker zusammengehören. Beim Stecken handelt es sich um ein simples physisches Rätsel der Kategorie *Object Movement/Alignment*. Da an dieser Stelle normalerweise das Spiel vorbei ist und dem Raum entkommen wurde, sollte durch das befriedigende Benutzen des Klinkensteckers ein Gefühl des Triumphs entstehen, besonders in Anbetracht der sehr komplexen Aufgabe, den PolRev korrekt zusammenzubauen.

4.2 Rein Virtuell

Hier gibt es nur zwei tatsächliche Rätsel und drei zusätzliche Tasks im Gegensatz zu den rein physischen Aufgaben, die fünf Rätsel und eine Task beinhalten. Dies war bewusst so konzipiert, da die P-Personen zu zweit sind, um die Rätsel zu lösen und davon ausgegangen wurde, dass die VR-Person sich erstmal in der VR orientieren lernen müsste.

[14] Obwohl hier nicht etwa selbstständig mit einem Licht umhergeleuchtet wird, muss das Licht dennoch erst angeschaltet werden. Da dies durch die VR-Person erfolgt, könnte dieses Rätsel als Cross Reality gewertet werden, doch da dies das einzige PolRev-Element ist, das XR-Interaktion erfordert, wurde der gesamte PolRev und damit auch dieses Teilrätsel als rein physisches Rätsel eingestuft.

4.2.1 VR-Lichtschalter-Task

Nachdem die Codewolke verschwunden ist, erscheint ein virtueller Lichtschalter. Dieser ist jedoch nicht als üblicher Lichtschalter gestaltet, sondern als großer Schieberegler. Dies dient der größeren Auffälligkeit sowie einer befriedigenderen Interaktion in der VR. Mit einer großen Bewegung muss der Regler gegriffen und zur Seite gezogen werden. Die Bewegung des roten Schiebeelements dient dabei als klares visuelles Feedback. Wenn der Regler auf über die Hälfte gezogen und dort losgelassen wird, erscheint die virtuelle Repräsentation des Escape Rooms und im echten Raum geht das Licht an. Zudem beginnt ab diesem Moment, der Lichtpunkt im realen Raum die Position der VR-Person darzustellen.

Es handelt sich hierbei um eine Task, da der Lichtschalter in diesem Moment das einzige sichtbare Element ist und die VR nur das Schieben zur Seite zulässt. Für VR-unerfahrene Person dient dieses Element ebenfalls als Tutorial für die Greifen-Funktion, ohne die die Task nicht abgeschlossen werden und dementsprechend keine Partei mit dem Lösen anderer Rätsel fortfahren kann.

4.2.2 Bewegbare-Wand-Task

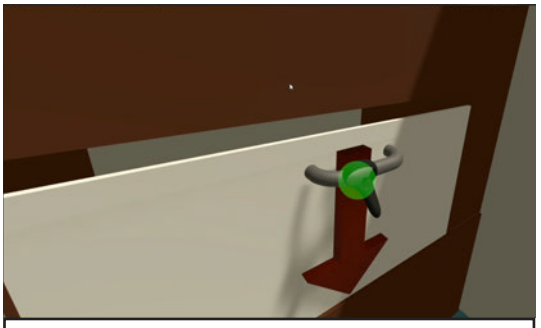


Abb. 23: Die bewegbare Wand während des Herunterziehens

An der Stelle, an der sich im realen Raum die fallende Wand befindet, befindet sich eine Repräsentation selbiger in der VR, markiert mit einem Pfeil nach unten und einem griffartigen Element. Dies soll dazu dienen, die VR-Person darauf aufmerksam zu machen, dass man diese Wand herunterziehen kann. Die Mechanik ist dabei grundlegend dieselbe, wie bei den Schiebereglern des Lichtschalters und des Partymodus-Schalters, nur, dass die grafische Repräsentation eine andere ist und beim Loslassen unterhalb einer bestimmten Grenze die Wand langsam nach unten in ihre finale Position gleitet. Gleichzeitig wird dadurch die **fallende Wand** im P-Raum ausgelöst. Da diese sich nicht ohne Beaufsichtigung wieder hochziehen lässt, erfolgt keine weitere Interaktion, sollte die Wand im VR-Raum wieder hochgezogen werden. Hinter der Wand ist einer der Blöcke für das Coderätsel versteckt, um sicherzustellen, dass die Wand früh im Spiel heruntergezogen wird.

Wie angemerkt, basiert diese Task auf dem Lichtschalter. Zwar gibt es zu diesem Zeitpunkt auch andere interaktive Elemente, doch der Griff und der Pfeil machen die vorgesehene Nutzung sehr eindeutig, weswegen von einem Rätsel maximal die Rede sein kann, wenn man die Wand als *Observation/Searching*-Rätsel wertet.

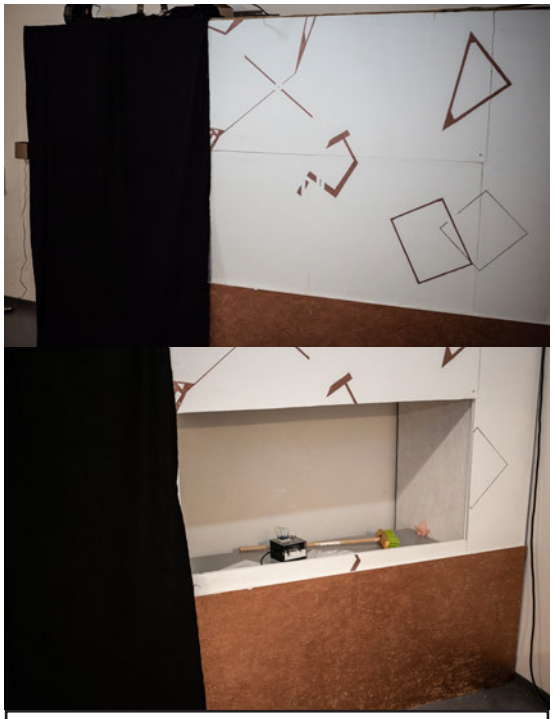


Abb. 24: Die Fallende Wand im P-Raum, oben in geschlossenem, unten in offenem Zustand

Fallende Wand

An der OSB-Plattenkonstruktion im P-Raum befindet sich eine „Wand" aus Styroporplatten, die mit einem dünnen Seil rechts und links an zwei Rollen aufgehängt ist. Die Rollen werden jeweils von einem DCMotor betrieben, der das Seil aufrollen (und damit die Wand hochziehen) oder abrollen (und damit die Wand herunterlassen) kann. Hinter der Wand befinden sich der Besen, der PolRev sowie einige Hinweise zu seiner Benutzung. Angesteuert werden die DC-Motoren über einen WEMOS D1 Chip, der mit dem Hauptrechner in Verbindung steht.

4.2.3 Coderätsel

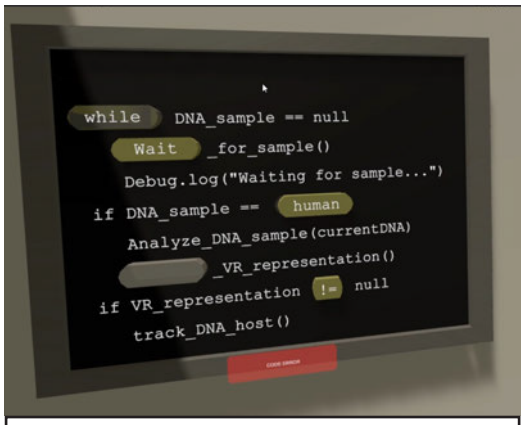


Abb. 25: Das fast vollständig gelöste Coderätsel – es fehlt noch der Block „Create"

Das Coderätsel befindet sich räumlich an der Stelle, an der im realen Raum der Screen ist, auf dem die Nachricht zum Erstellen des Avatars dargestellt wird. Analog ist dies die zweite Komponente, die gelöst werden muss, um die Pointcloud freizuschalten. Virtuell besteht das Rätsel aus einem statischen Textteil im Stil von Programmcode, bei dem einzelne Begriffe fehlen. Diese liegen als Blöcke im Raum verteilt und können mit den VR-Handcontrollern aufgehoben und in der entsprechenden Lücke platziert werden. In der Entwicklung wurde zunächst das Verhalten der Blöcke beim Schlüsselrätsel kopiert, da dort bereits eine Mechanik für das Einsetzen von Blöcken in vorgesehene Bereiche existierte. Da das Rätsel jedoch nur eine korrekte Lösung hat und alle anderen Möglichkeiten keinen Erfolg darstellen, musste hier die Logik angepasst werden. Da die Blöcke hier nicht auf eine zweidimensionale Ebene beschränkt waren und zudem als physische Objekte, mit auf sie wirkender Schwerkraft, funktionieren sollten, war außerdem eine umfangreichere Anpassung des Codes zum Verwalten der Kollision mit den vorgesehenen Lücken notwendig.

Die korrekte Position der Blöcke erschließt sich, wenn man den „Code" liest und versucht, sinnvolle Sätze zu erstellen. Es handelt sich also um ein *Correlation/Logic*-Rätsel. Da die Blöcke im Raum verstreut und zum Teil sogar versteckt sind, gibt es ebenfalls eine *Observation/Searching*-Komponente. Da ein Vorwissen in Programmiersprachen zwar nicht notwendig, aber doch hilfreich ist, kann man das Rätsel ebenfalls in der Kategorie *Knowledge* ansiedeln. Zu guter Letzt ist auch *Object Movement/Alignment* notwendig, um die Blöcke an die richtigen Positionen zu befördern.

4.2.4 Partymodus-Schalter-Task

Wurde das Pressure-Plates-Rätsel gelöst, erscheint der Schalter für den Partymodus. Er funktioniert genau wie der Lichtschalter zu Beginn des Spiels, ist jedoch persistent und kann jederzeit nach seiner Freischaltung an- und ausgeschaltet werden. Wird der Knopf auf seine „An"-Position bewegt, beginnt der Partymodus. Dieser kennzeichnet sich in der VR durch buntes Licht und das Erscheinen des Passwortblocks in einer Ecke, die der oberen rechten Ecke des Whiteboards im P-Raum entspricht. Im P-Raum wird ebenfalls das Licht farbig, eine bunte Einblendung wird auf dem Screen abgespielt, fröhliche Musik ertönt und die **UV-Lampe** wird aktiviert.

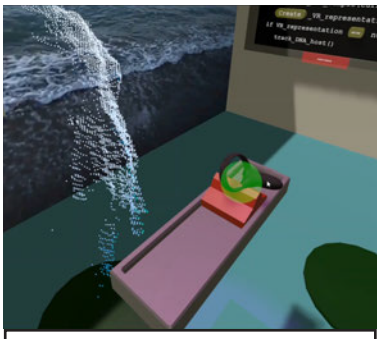


Abb. 26: Der Partymodus-Schalter während des Schaltens

Wie der Lichtschalter zuvor ist auch der Partymodus-Schalter lediglich eine Task. Die Simplizität wird davon verstärkt, dass man das Prinzip des Schiebereglers exakt so schon angetroffen hat.

UV-Lampe

Der Chip, der die fallende Wand kontrolliert, steuert ebenfalls per DMX-Protokoll eine UV-Lampe an, die auf der OSB-Plattenkonstruktion steht. Wird sie aktiviert, strahlt sie auf das Whiteboard, auf dem mit UV-Marker die richtige Ohmzahl für den PolRev vermerkt ist.

4.2.5 Passwort-Rätsel

Um dieses Rätsel lösen zu können, muss in der Realität das Digital-Clean-Up-Besen-Rätsel gelöst sein, damit die Passwort-Eingabemaske sichtbar ist. Das dort enthaltene Passwortfeld ist eine vereinfachte Version des Coderätsels mit nur einem Block. Ebenfalls braucht man die Belohnung der Partymodus-Schalter-Task, den Passwortblock. Dieser Block ist von seiner Funktion identisch mit den Codeblöcken, ist aber mit dem Wort „P@sSw0rD" beschriftet. Wird der Block eingesetzt, verschwindet die Eingabemaske und ein Anruf von der VR in den realen Raum wird gestartet. Hierbei wird das ansonsten gesperrte Smartphone, dass im P-Raum liegt, aktiviert: Auf dem Smartphone läuft eine Unity-Anwendung, die es über Photon-Unity-Networking (PUN) mit dem Hauptrechner verbindet. Dieser kann darüber ein Signal senden, das einen Anruf simuliert (visuell und auditiv). Wird der dabei erschienene grüne Button auf dem Telefon gedrückt, wird eine Voicechat-Verbindung zwischen den Clients (Telefon/Hauptrechner) etabliert, worüber der Input des Mikrofons der Oculus zu hören ist. Gleichzeitig wird das Abspielen des Raumsounds auf der Occulus gestoppt, damit es nicht zu einer Feedbackschleife kommt, und stattdessen nur der Sound des Mikrofons des Smartphones ausgegeben. Dieser Anruf konnte leider bis zum Ende des Experiments aufgrund technischer Probleme nicht verwirklicht werden und er wurde stattdessen durch einen tatsächlichen Anruf von einem Smartphone auf das P-Raum-Smartphone ersetzt. Der Spielleiter tätigte dabei den Anruf und reichte dann das Telefon an die VR-Person weiter.

Das Passwort-Rätsel erfordert ganz klassisch *Observation/Searching*, *Correlation/Logic* und *Object Movement/Alignment*, denn der Passwortblock muss zuerst entdeckt, dann mit der Passwort-Eingabemaske in Verbindung gebracht und zuletzt noch physisch dorthin bewegt werden.

4.3 Cross Reality

4.3.1 LED-Wand-Tresor

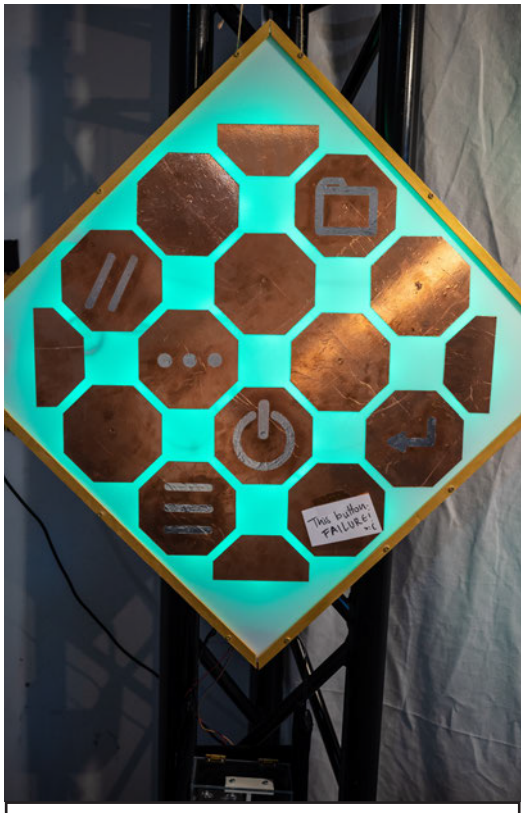


Abb. 27: Der LED-Wand-Tresor nach korrekter Eingabe des Codes (grün leuchtend)

Hierbei handelt es sich im P-Raum, um eine gerahmte Milch-Plexiglasscheibe, auf der Hexagons aus Kupferfolie kleben. Durch Löcher im Plexiglas, sind diese mit Kabeln verlötet, wodurch sie als kapazitive Sensoren fungieren. Die Kabel laufen in zwei MPR121 Touch-Sensor-Modulen zusammen, die wiederum an einen Arduino Nano angeschlossen sind. Der Arduino Nano steuert gleichzeitig eine Reihe mehrfarbige LEDs an, die hinter dem Plexiglas mit etwas Abstand auf ein dünnes Holzbrett aufgeklebt sind. Jeweils ein 5er-Set LEDs beleuchten dabei das Hexagon, das berührt wurde, weiß. Der Arduino ist per USB mit einem Serial Port des Hauptrechners verbunden und kann darüber auch die Tastendrucksignale der VR empfangen, die ebenfalls von den LEDs angezeigt werden, allerdings in Pink.

Auf der Kupferfolie kleben außerdem Symbole aus Aluminiumfolie. Jedoch sind nicht alle Hexagone beschriftet. Die fehlenden Symbole sind im VR-Äquivalent zu finden, wo andersherum die Symbole fehlen, die auf der realen LED-Wand existieren.

Werden in der Realität die Tasten in der korrekten Reihenfolge, welche an einer Wand in der Realität notiert ist, berührt, öffnet sich eine kleine Plastikkiste. Diese Kiste ist ebenfalls mit dem Arduino verbunden und enthält ein elektrisches Schloss, das durch einen Riegel den Deckel der Kiste verschlossen hält. In der Kiste liegt eine 9-Volt-Blockbatterie.

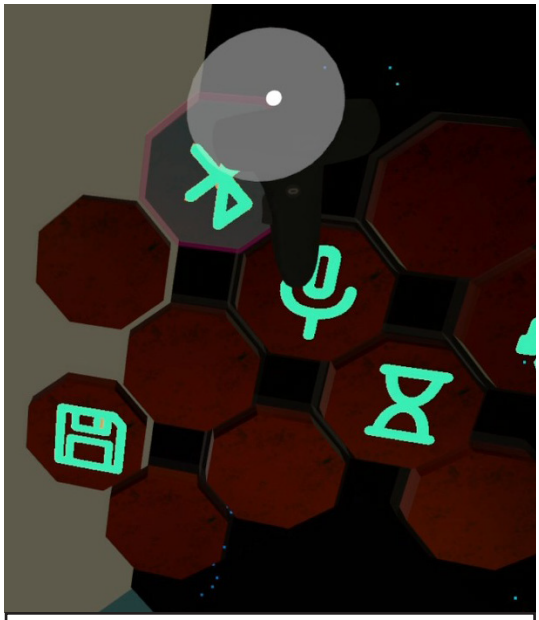


Abb. 28: Interaktion mit dem LED-Wand-Tresor in der VR

An der entsprechenden Stelle des LED-Wand-Tresors befindet sich ein digitales Faksimile desselben. Wie beschrieben, werden hier die Symbole dargestellt, die nicht in der Realität vorhanden sind. In der ersten Iteration war die Interaktion hier als Poke-Element gestaltet. Wenn eines der Hexagone gedrückt wurde, leuchtete es in der Realität wie auch in der VR auf, bis das gleiche Hexagon in der Realität gedrückt wurde. Dies führte jedoch dazu, dass die Spieler*innen gezwungen waren, das leuchtende Feld zu berühren, um weitere Hinweise bekommen zu können, was unter Umständen dazu führte, dass ein falsches Hexagon berührt und die Sequenz von vorne eingegeben werden musste. Daher wurde in der zweiten Iteration in der VR die neue Interaktion des „Berührens" eingeführt. Wenn ein Element berührt wird, so leuchtet es für die Dauer der Berührung sowohl virtuell als auch real auf, während zusätzlich, als weiteres Feedback, der Controller vibriert. Berührungen in der Realität führen simultan zu einem Aufleuchten des entsprechenden Hexagons in der VR.

Wenn die P-Personen den vollständigen Code eingegeben haben, wird ein rundes, mit „Menu" beschriftetes Element, das **Radial Menu**, freigeschaltet, das wiederum durch „Poke" aufgenommen werden kann.

Dieses Rätsel hat im P-Raum den Aspekt *Observation/Searching*, weil der richtige Code ein wenig versteckt in Bodennähe auf der Wand steht und zuerst gefunden werden muss. Zudem muss eine *Pattern Recognition* stattfinden, wobei die P-Personen lediglich die exakt gleichen Zeichen mental zusammenbringen müssen, während die VR-Person das Symbol durch eine auditive Beschreibung mit dem visuellen Zeichen in Verbindung bringen muss. Hierzu gehört zum Teil auch *Knowledge*, denn es wussten zum Beispiel nicht alle Teilnehmer*innen, was das Bluetooth-Symbol ist. Als asymmetrisches, Mixed-Reality-Rätsel erfordert es zudem *Deduction*, da ein Zusammenhang zwischen den pink leuchtenden Hexagonen und der VR-Person komplett eigenständig erkannt werden muss, um die Informationen aus der VR sinnvoll nutzen zu können. Gleichzeitig muss die VR-Person folgern, dass ihre Hilfe benötigt ist, wenn die P-Personen die Symbole vorlesen und eine zur anderen beispielsweise sagt: „Es gibt kein Mikrofon." Die korrekte Reihenfolge ist, extra um diesen Schritt zu ermöglichen, so weit weg von der LED-Wand entfernt angebracht, dass die P-Personen gezwungen sind, den Code einander vorzulesen und so unbeabsichtigt die VR-Person einbinden, die sie die ganze Zeit hören kann. Für alle Beteiligten besteht also ebenfalls der Aspekt von *Distributed Knowledge/Information Sharing*, wobei eine Person den Code mitteilt, die zweite bemerkt, welche Informationen fehlen und die dritte Person diese Informationen weitergibt.

Radial Menu

Das Radial Menu wird durch Lösen des LED-Wand-Tresor-Rätsels in der VR freigeschaltet. Nachdem es aufgenommen wurde, haftet es am linken Controller der VR-Person und kann durch Poke-Interaktion bedient werden. Nach dem ersten „Poke" klappen aus dem zentralen Element sternförmig acht Elemente aus, die beschriftet sind mit jeweils: „Yes", „No", „It's me!", „Stay!", „Go!", „Here!", „Help!" und „Argh!". Wenn diese wiederum gedrückt werden, ertönt im P-Raum die entsprechende Nachricht per Sprachsynthese.

Abb. 29: Das Radial Menu

4.3.2 Pressure-Plates-Rätsel

Bei diesem Rätsel müssen drei Knöpfe (Pressure-Plates) auf dem Boden gedrückt werden. Ein Drücken erfolgt, wenn entweder die VR-Person oder eine der P-Personen auf dem Knopf steht und lässt diesen grün statt rot erscheinen. Ob die VR-Person auf einem Knopf steht, wird durch einen Collider, der sich etwas hinter der virtuellen Kamera befindet, bestimmt. Die nach hinten verlagerte Position ergab sich daraus, dass es sich bei Selbstversuchen falsch angefühlt hat, wenn das Drücken nicht durch die Körpermitte, sondern sozusagen durch die Augen – den Sitz der VR-Brille – erfolgte. Ob eine reale Person auf einem der Knöpfe steht, wird erfasst, indem das Programm die Punkte der Pointcloud, die sich innerhalb eines Colliders befinden zählt, und beim Erreichen einer gewissen Schwelle für wenige Sekundenbruchteile (um Schwankungen auszugleichen) den Knopf entsprechend betätigt. Sind alle Knöpfe gleichzeitig grün, so verschwinden sie und der Partymodus-Schalter wird freigeschaltet.

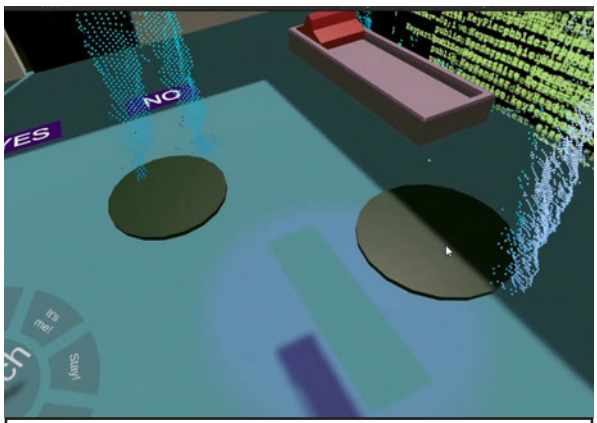


Abb. 30: Das gelöste Pressure-Plates-Rätsel. Man sieht, wie beide P-Personen auf den entsprechenden Positionen stehen und den bereits eingeblendeten Partymode-Schalter

Das Erfassen der realen Positionen durch die Pointcloud-Punkte funktionierte zwar, erforderte aber einen aufwendigen Kalibrierungsprozess und war störungsanfällig. Ein zusätzliches anderes System, mit einem softwaregestützten Skeleton-Tracking ähnlich dem erwähnten Theaterprojekt [13] oder Positionserkennung wäre hier für zukünftige Iterationen zu bevorzugen.

Dieses Rätsel basiert wie alle Mixed-Reality-Rätsel in XcapeRoom auf dem *Distributed Knowledge/Information Sharing* Konzept. Wobei hier die VR-Person erst einmal durch *Observation/Searching* herausfinden muss, was eigentlich zu tun ist. Sie muss bemerken, wann die Punkte grün werden und erkennen, dass es erstrebenswert ist, alle drei Knöpfe gleichzeitig zu aktivieren. Hierfür ist *Knowledge* über Rätsel dieser Art, wie sie in Computerspielen häufig vorkommen, hilfreich, wenn auch nicht zwingend notwendig, anderenfalls kann durch *Correlation/Logic* die Verbindung zwischen den Knöpfen aufgrund ihrer Gleichartigkeit bemerkt werden. Da die VR-Person zu diesem Zeitpunkt noch nicht in direktem Kontakt mit den P-Personen steht, sondern behelfsweise das Radial Menu verwenden muss, um mit den anderen zu kommunizieren, ist auf deren Seite ebenfalls *Deduction* gefordert, da sie erkennen müssen, dass es sich bei den ominösen Sprachprompts um Nachrichten

der VR-Person handelt und sie die Motivation haben müssen, diesen „Befehlen" auch zu folgen. Auch dass der Lichtpunkt auf dem Boden von der VR-Person kontrolliert wird, ist an dieser Stelle eine notwendige Information, die man durch *Deduction* hergeleitet haben muss. Da die P-Personen keinerlei Anhaltspunkte für die richtigen Position haben, muss der VR-Person quasi blind vertraut werden, dass das bloße Herumstehen im Raum einen tieferen Sinn hat. Alle Spieler*innen müssen zudem sehr einfache *Self-Motion/Agility* absolvieren, da sie sich selbst im Raum platzieren müssen.

Abb. 31: Das Innenleben des „KIBO"

4.3.3 Schlüsselrätsel (3D-Drucker)

Der 3D-Drucker (KIBO) ist eine schwarze Kiste mit einem Auswurfschlitz an der unteren Vorderseite und wird hauptsächlich in der VR bedient. Das Bedienelement in der Realität ist lediglich ein wenige Zentimeter großer Touchscreen, der 4 verschiedene Nachrichten anzeigt „No 3D Model", „Ready to print <Start><Abbr.>", „[Countdown] <Abbr.>" und „DONE <Okay!>". Solange die erste Nachricht angezeigt wird, kann man in der Realität nichts mit dem Drucker machen. Wird in der VR ein Schlüsselmodell zusammengestellt und bestätigt, wird in der Realität die zweite Nachricht angezeigt, der Schlüssel"druck" wird gestartet und die dritte Nachricht wird aktiviert, bei der ein Countdown die Fertigstellung des Drucks ankündigt. Nach Ablauf des Countdowns wird der in der VR erstellte Schlüssel aus der Kiste ausgeworfen und die vierte Nachricht erscheint, mit welcher man den VR Teil des Druckers erneut freischalten kann, um einen weiteren Schlüssel zu „drucken".

Im Inneren der Kiste sind die 12 möglichen Kombinationen der Elemente aus der VR aufgehängt. Sie klemmen zwischen zwei Holzbrettern und werden von einem kleinen Plastikstift gehalten. Hinter den Schlüsseln ist eine Zahnschiene, auf der ein kleiner Stepper-Motor die gesamte Länge der Kiste abfahren kann. Der Motor ist in eine Halterung montiert, die von einer durch die Kiste verlaufenden Stange aufrecht gehalten wird. Auf der Halterung ist ebenfalls ein kleiner linearer Servo-Motor montiert, der mit Hilfe eines darauf angebrachten Hakens die jeweiligen Stifte herausziehen kann, die die Schlüssel fixieren. Mit einem Flachbandkabel sind die beiden Motoren mit einem WEMOS D1 verbunden, der einerseits mit dem Touchscreen auf der Vorderseite der Kiste in Kontakt steht und andererseits eine WLAN-Verbindung zum Hauptrechner hält. Arrangiert nun die VR-Person die Formen in eine beliebige Kombination und wird diese in der Realität bestätigt, steuert der Controller den Steppermotor vor den entsprechenden Schlüssel. Wenn der Countdown abgelaufen ist, zieht der Servomotor den Stift und der Schlüssel rutscht über eine Rampe aus der Kiste.
Die Alternative zum fahrenden Steppermotor wäre eine Konstruktion gewesen, bei der jeder Stift seinen eigenen Servomotor erhalten hätte. Diese Variante wurde aber aus Kostengründen verworfen.

Im VR-Raum muss die VR-Person drei von fünf Blöcke in die dafür vorgesehenen Felder einordnen. Die Blöcke sind mit dem „Grab Interactor"-Verhalten des Oculus Frameworks ausgestattet und zudem in ihrer Bewegung

auf eine zweidimensionale Ebene begrenzt. Zusätzlich besitzen sie ein spezielles Verhalten, um die Interaktion mit den Platzhalterfeldern zu steuern. Diese haben einen Collider und wenn eine Kollision zwischen den Objekten stattfindet, wird es direkt an die richtige Stelle teleportiert und der Kontrolle der VR-Person entzogen. Erst durch erneutes „Greifen" erfolgt eine neue Interaktion und das Element lässt sich wieder bewegen. Es gibt zwei große und zwei kleine Slots sowie vier verschiedene große Elemente und ein kleines Element. Das kleine Element passt nur in die kleinen Slots, die großen Elemente nur in die großen Slots. Die beiden großen und einer der kleinen Slots müssen gefüllt sein, um einen validen Schlüssel zu erzeugen. Auf diese Weise gibt es 24 unterschiedliche Kombinationsmöglichkeiten, wobei 12 davon spiegelsymmetrisch zu jeweils einer Kombination aus den übrigen 12 sind, wodurch es in der Realität nur 12 unterschiedliche Teile geben muss, da dort die gespiegelte Version durch Drehen des Schlüssels erzeugt werden kann. Wenn eine valide Kombination eingestellt wurde, kann durch „Poke"-Interaktion ein Knopf gedrückt werden, der mit „Press to Confirm Model" beschriftet ist. Daraufhin wird eine Nachricht an den „3D-Drucker" gesendet und weitere Interaktionen gesperrt (mit dem Warnhinweis „Sending…"), bis der Schlüssel „gedruckt" und bestätigt wurde. Sobald das richtige Schlüsselmodell in der VR bestätigt wurde, erscheint das Hochschul-Modell in der Mitte des VR-Raums.

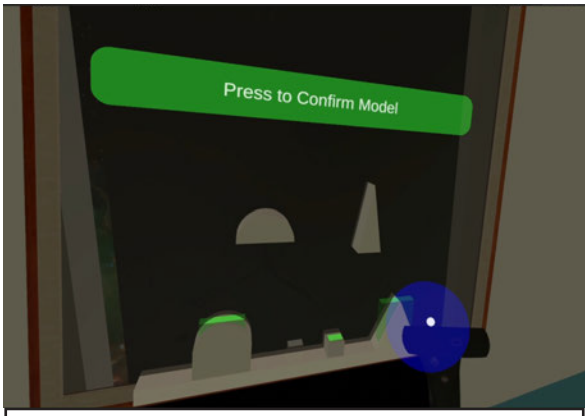


Abb. 32: Der virtuelle Teil des Schlüsselrätsels. Der korrekte Schlüssel ist eingegeben und der Schalter oben freigeschaltet

Um das Rätsel erfolgreich zu lösen, müssen die P-Personen der VR-Person über das Smartphone, das im Passwort-Rätsel freigeschaltet wurde, kommunizieren, wie das "Schlüsselloch" im PolRev aussieht und die VR-Person kann entsprechend ein Schlüsselmodel zusammenstellen. Nach dem Druck können die Spielenden den Schlüssel testen: Passt er nicht, muss die VR-Person erneut ein modifiziertes Modell zusammenstellen. Es ist also ein *Distributed Knowledge/Information Sharing*-Rätsel, da an dieser Stelle wieder mit einer asymmetrischen Informationslage gearbeitet wird: Die P-Personen können analysieren (*Pattern Recognition*), wie der Schlüssel aussehen muss, aber nur die VR-Person weiß, welche Elemente zum Bau des Schlüssels zur Verfügung stehen. Außerdem ist es für die VR-Person ein *Combination*-Rätsel, da die einzelnen Teile des Schlüssels zusammengeführt werden müssen. Nach dem Beschreiben des Schlüsselloches müssen die P-Personen noch den Druck starten (eine Task) und den Schlüssel testen (*Object Movement/Alignment*).

4.3.4 Hochschulmodell-Rätsel

Abb. 33: Eine VR-Person betrachtet das Hochschulmodell

Das Hochschulmodel ist eine kleine dreidimensionale Karte des relevanten Bereichs der Gänge der Hochschule. Der Ort, an dem die P-Personen den realen Raum verlassen können, sowie der Aufenthaltsort der VR-Person sind markiert. Die Karte ist leicht abgeändert (die Länge der Gänge ist verändert, teilweise wurde der Boden ausgespart, um nicht-durchquerbare Bereiche zu markieren) und einige Orientierungspunkte (Bänke, geschlängelte Linie auf dem Boden,

Bild eines Kaninchens, Türen), die eine Entsprechung in der Realität haben, sind hervorgehoben. Die Karte ist ein simples 3D-Modell, das durch Greifen manipuliert werden kann. Es wirkt keine Physik auf das Objekt, wodurch es beim Loslassen statisch im Raum schwebt. Die VR-Person kann sich durch die Karte einen Überblick verschaffen und die P-Personen mittels des Smartphones durch die Gänge (auf einem Umweg) zu sich selbst führen, wodurch das Spiel gewonnen ist.

Auch bei diesem letzten Rätsel handelt es sich um eine Situation, in der asymmetrische Informationsverteilung herrscht: Die VR-Person kennt den ungefähren Weg, die P-Personen befinden sich in den realen Gängen. Die VR-Person muss also *Information Sharing* betreiben, um die anderen zu sich zu führen. Die P-Personen befinden sich in der Position, *Observation/Searching* anwenden zu müssen, um die von der VR-Person beschriebenen Objekte zu finden und sich orientieren zu können. Zudem müssen sie durch *Self-Motion/Agility* die Strecke zurücklegen, um zum Ziel zu gelangen.

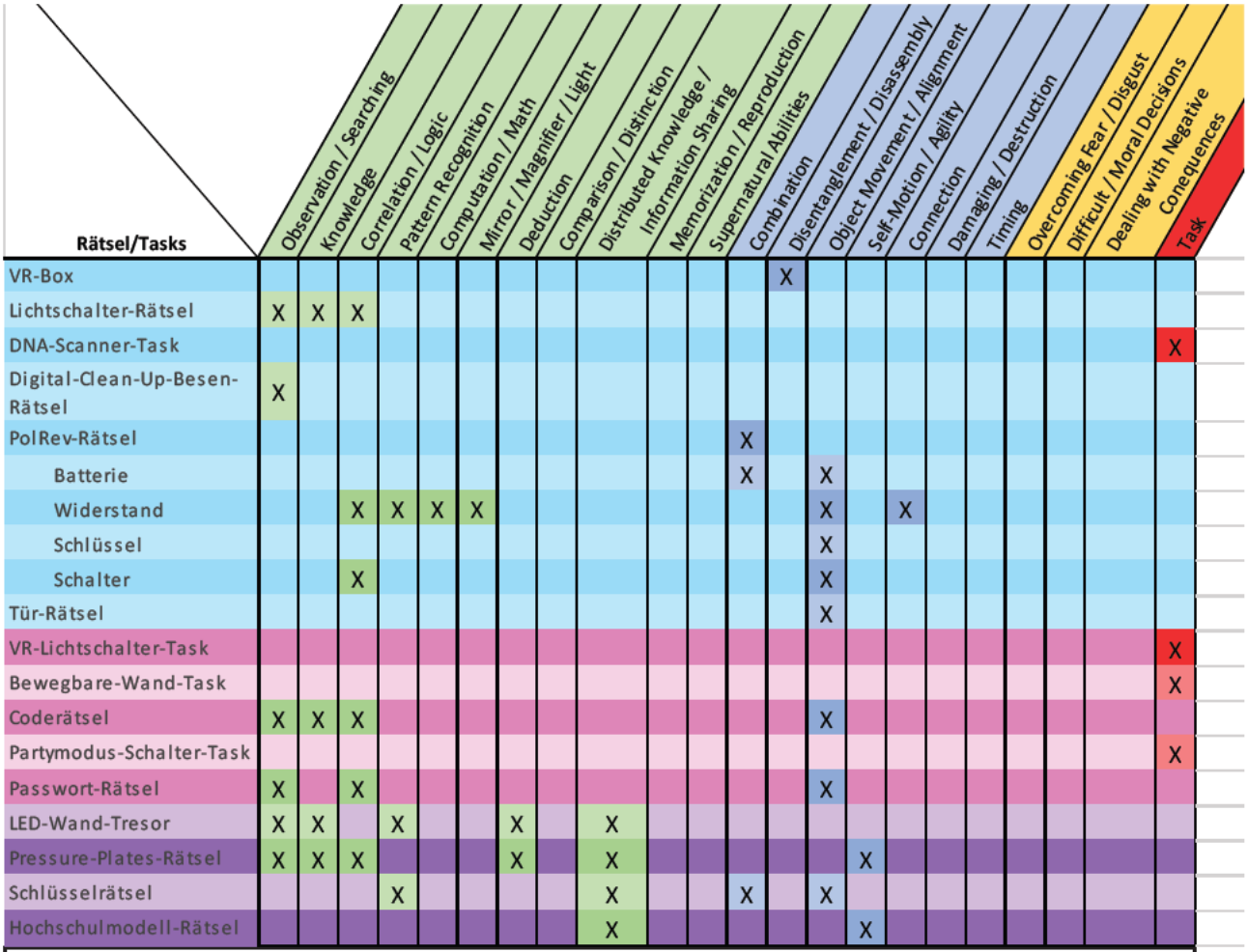
4.4 Rätselüberblick

Um einen Überblick zu ermöglichen, folgt an dieser Stelle eine Tabelle, die die Rätsel und ihre analysierten Challenges darstellt. Daraus wird schnell ersichtlich, welche Arten von Challenges fehlen oder überrepräsentiert sind. In der Kopfzeile finden sich alle Challenges aus der Taxonomie, wobei die Überkategorien Mental Challenges, Physical Challenges und Emotional Challenges farblich voneinander abgesetzt sind. Tasks werden der Vollständigkeit halber auch aufgelistet. Die erste Spalte enthält die Rätsel – ebenfalls farblich abgesetzt, in Cyan für rein physische Rätsel, Magenta für rein virtuelle und Lila für XR-Rätsel. Die Challenges, die für die entsprechenden Rätsel festgestellt wurden, sind in der Tabelle mit einem X markiert und in der Farbe der Challenge-Art eingefärbt.

In dieser Darstellungsform fällt sofort auf, dass einige Challenges überhaupt nicht verwendet werden. Dies betrifft sämtliche Emotional Challenges,[15] *Timing*, *Damaging/Destruction*, *Supernatural Abilities,*[16] *Memorization/Reproduction* und *Comparison/Distinction*. Andere kommen nur im realen Teil vor: *Computation/Math*, *Mirror/Magnifier/Light* und *Disentanglement* sind abwesend vom VR-Teil. Besonders stark vertreten ist die Challenge *Object Movement/Alignment* – wobei die Herausforderungen hier alle relativ trivial sind. Es wäre zu überlegen, ob zukünftige Forschung eine Dimension der Schwierigkeit der einzelnen Challenges einführt, um die Gewichtigkeit und nicht nur die bloße Anzahl von Challenges in die Auswertung von Escape Rooms einfließen zu lassen.

[15] Dies ist konsistent mit der Beobachtung von Krekhov et al., dass diese Art der Rätsel (noch) sehr selten Anwen-dung findet. In Teil 5.1 wird erwähnt, wie das Feature der Telepräsenz für eine besondere Art der Emotional Challenge eingesetzt werden könnte.

[16] Ein solches Rätsel war ursprünglich geplant: Die VR-Person sollte, anstatt auf den LED-Wand-Tresor zu zeigen, den Raum um 90° drehen und sich auf Plattformen mit den entsprechenden Symbolen stellen, die an der Wand – jetzt dem Boden – befindlich gewesen wären. Dieses Rätsel wurde zu dem jetzigen LED-Wand-Tresor-Rätsel vereinfacht, da einerseits die Umsetzung für den engen Zeitrahmen zu anspruchsvoll gewesen wäre und andererseits diese Interaktion aufgrund ihrer Ungewöhnlichkeit eine ausführliche Erklärung oder Einführung für die VR-Person erfordert hätte, die auch den Umfang des Spiels stark erhöht hätte.



Rätsel/Tasks	Observation / Searching	Knowledge	Correlation / Logic	Pattern Recognition	Computation / Math	Mirror / Magnifier / Light	Deduction	Comparison / Distinction	Distributed Knowledge / Information Sharing	Memorization / Reproduction	Supernatural Abilities	Combination	Disentanglement / Disassembly	Object Movement / Alignment	Self-Motion / Agility	Connection	Damaging / Destruction	Timing	Overcoming Fear / Disgust	Difficult / Moral Decisions	Dealing with Negative Consequences	Task
VR-Box												X										
Lichtschalter-Rätsel	X	X	X																			
DNA-Scanner-Task																						X
Digital-Clean-Up-Besen-Rätsel	X																					
PolRev-Rätsel											X											
Batterie											X		X									
Widerstand			X	X	X	X							X		X							
Schlüssel													X									
Schalter			X										X									
Tür-Rätsel													X									
VR-Lichtschalter-Task																						X
Bewegbare-Wand-Task																						X
Coderätsel	X	X	X										X									
Partymodus-Schalter-Task																						X
Passwort-Rätsel	X		X										X									
LED-Wand-Tresor	X	X		X			X		X													
Pressure-Plates-Rätsel	X	X	X				X		X					X								
Schlüsselrätsel				X					X		X		X									
Hochschulmodell-Rätsel									X					X								

Abb. 34: Tabellarische Darstellung aller Rätsel und Tasks von XcapeRoom, in Challenges aufgeschlüsselt

5 Evaluation

5.1 Auswertung der Durchläufe

Die Durchläufe der Gruppen wurden durch die Spielleiter*innen über eine im Raum verbaute Kamera beobachtet. Das Video dieser Kamera wurde zudem zu Dokumentationszwecken und für die weitere Auswertung aufgezeichnet. Bei einer Gruppe fiel die Aufzeichnung leider aus, weshalb es nur fünf Aufzeichnungen gibt, zudem fehlen bei zweien der Ton, was das Nachvollziehen der genauen Handlungen erschwert. Wo immer möglich, wurden die Angaben hier aus der Erinnerung der Spielleiter*innen ergänzt. Vor den aufgezeichneten Durchläufen gab es noch zwei Probedurchläufe mit Proband*innen, von denen Aspekte aus der Erinnerung hier Anwendung finden. Die Sicht der VR-Person konnte während der Durchführung nicht beobachtet werden, wurde aber durch eine Bildschirmaufnahme des Hauptcomputers ebenfalls zur weiteren Auswertung aufgezeichnet. Zudem konnten die Spielleiter*innen durch die Kommunikation mit den Teilnehmenden weitere Erfahrungen sammeln, die ebenfalls in die Auswertung einfließen. Bei einer der aufgezeichneten Gruppen musste das Spiel aufgrund schwerwiegender technischer Probleme neugestartet werden. Insgesamt sind die Aussagen in diesem Teil stark von der Prototypenhaftigkeit des Escape Rooms geprägt. Wo dies spezifische Auswirkungen auf die Beurteilung hat, wird dies erwähnt. Allgemein lässt sich sagen, dass das Hilfestellungssystem teilweise zur Klärung von technischen Mängeln benutzt wurde, und gelegentlich ein*e Spielleiter*in

den Raum betreten musste, um die Funktionalität der Rätsel wiederherzustellen. Dadurch wurde der Spielablauf gestört und eine Atmosphäre geschaffen, in der die Spieler*innen hinterfragen mussten, ob sie etwas falsch machen, oder ob ein Rätsel kaputt war. Dies führte dazu, dass noch mehr Hilfestellungen gegeben werden mussten, sodass Teile der Durchläufe sehr geleitet stattfanden. Die Aspekte der Rätsel, in denen es darauf ankommt, herauszufinden, was zu tun ist, lassen sich daher nur in eingeschränktem Maße beurteilen.

Um die Meinungen der Teilnehmenden zu erfassen, wurde aufgrund der geringen Anzahl an Teams im Sinne einer qualitativen Auswertung ein informelles Interview gewählt, bei dem jeweils auf die Aussagen der Teilnehmenden eingegangen wurde. Hierbei wurde die unverbindliche Situation des „Froth"[17] genutzt, um die Reaktionen zu sammeln, solange sie noch frisch in der Erinnerung waren und ein direkter Bezug zu den Gegenständen im Raum hergestellt werden konnte. Ein besonderer Fokus bei den Fragen lag auf den Aspekten der Pointcloud und der Interaktion und Kooperation über die Realitätsmodalität hinweg. Es stellte sich als schwierig heraus, die Frage nach der Qualität der Interaktion in einer Art zu stellen, die nicht auf eine bestimmte Antwort hinleitet und dennoch den relevanten Aspekt der Präsenz hervorhebt. Formulierungen wie „hat es sich angefühlt, als wäre die andere Person mit im Raum" waren ungenügend und die Antworten wurden daher im Gespräch gesucht. Durch die offene Herangehensweise konnten Aspekte berücksichtigt werden, die erst im Laufe der Tests offenkundig sichtbar wurden, etwa, dass bei einigen Teilnehmenden eine Hemmung bestand, auf den Lichtpunkt im realen Raum zu treten, was dann bei späteren Gruppen auch abgefragt wurde.

Insgesamt wurden sechs Interviews geführt, die jeweils mit einem Smartphone aufgezeichnet wurden. Die Gruppen stammten alle aus dem universitären Umfeld oder dem Bekanntenkreis des Autors. Jede Gruppe bestand aus drei Personen. Sie meldeten sich (bis auf eine) vorher zusammen in einem Online-Formular an. Wegen der kleinen Probengröße wurde auf demographische Erhebungen verzichtet. Aufgrund der großen Nähe der Teilnehmenden zum Forschungsteam sind Geschlecht, Altersgruppe und Vorerfahrung mit Escape Rooms, Videospielen oder Virtual Reality bekannt und – wo möglich – werden vorsichtige Vermutungen formuliert, wie sich diese Faktoren auf die Erfahrung auswirken. Im Sinne des iterativen Designprozesses, den Krekhov et al. für Escape Rooms empfehlen [15], wurden zwischen den Durchläufen Verbesserungen aufgrund des Feedbacks durchgeführt. Wegen der geringen Anzahl an Durchläufen konnte jedoch in den Interviews der Erfolg der Anpassungen nicht klar ausgewertet werden, zumal die kurze Zeitspanne keine großen Änderungen erlaubte und das Lösen technischer Probleme im Vordergrund stand. Besonders prävalent war das Problem der starken Verzögerung des Audiosignals zwischen dem realen Raum und der VR-Umgebung. Dieses konnte erst bei der letzten Gruppe behoben werden und wurde von allen vorigen Gruppen als sehr störend bewertet. Da dieses Problem direkten Einfluss auf die Kooperationserfahrung, die im Mittelpunkt der Betrachtung stand, hatte, ist es bemerkenswert, dass dennoch vier von den fünf betroffenen Gruppen attestierten, dass sich ein Gefühl der Kooperation einstellte. Die VR-Person der fünften Gruppe merkte an, dass „for me […] cooperation wasn't really possible at a lot of times", also dass Kooperation oft nicht möglich war. Dies wurde mit der mangelnden Qualität der Kommunikationsmöglichkeiten begründet. Interessanterweise wurde hier als Beispiel das Pressure-Plates-Rätsel herangezogen, das strenggenommen von der Verzögerung nicht betroffen

[17] Nicholson zieht den Begriff „Froth" aus der Live Action Role Playing Community heran. [22] Dieser beschreibt das Diskutieren der gerade durchlebten Ereignisse im Zustand großer Ereiferung,welcher sich beim Escape Room nach dem finalen Rätsel einstellt. Nicholson geht sogar so weit, das Erzeugen desselben zum Ziel guter Escape Room Designer zu erklären und erwähnt, dass die meisten Escape Rooms den Spieler*innen Zeit geben, zusammen mit den Angestellten der Einrichtung das Erlebte zu besprechen [22].

war, da die Kommunikation aus der VR durch das Radial Menu erfolgt, welches keiner Verzögerung unterliegt, und das Feedback durch die Aktionen der Spieler*innen im realen Raum durch die Pointcloud ebenfalls ohne Verzögerung sichtbar wird. Jedoch hört die VR-Person die Kommandos, die sie über das Radial Menu geben kann, nur, indem diese im realen Raum abgespielt und über das Raummikrofon in die VR übertragen werden. Obwohl die Kommandos also direkt ankommen, entsteht für die VR-Person der Eindruck einer Eingabeverzögerung, selbst wenn die Reaktion der Mitspieler*innen sofort erfolgt. Erschwerend kommt hinzu, dass bei dieser Gruppe das Pressure-Plates-Rätsel zufällig gelöst wurde: „It just aligned by accident basically." Dieses Problem trat bei mehreren Gruppen auf und führte dazu, dass die Kooperation über das Radial Menu und Lichtposition mit Pointcloud-Feedback nicht bis zum Ende ausprobiert wurde. Da die VR-Person hier allerdings auch nicht viel Einfluss hat, wenn die Mitspieler*innen nicht adäquat kooperieren, wurde das zufällige Lösen von den betroffenen Gruppen als positiv empfunden.

Umgekehrt wurde gerade das Pressure-Plates-Rätsel von anderen VR-Personen als besonders kooperativ empfunden, da man in direktem Kontakt mit den P-Personen steht, die unmittelbar von der VR-Person mit Audiosignalen und ihrem Lichtpunkt angewiesen werden. Auch die extreme Einschränkung der Kommunikation auf die vorbestimmten Kommandos wurde von dieser VR-Person als besonders stimulierend hervorgehoben: „Gerade die Machtlosigkeit in dem Moment, wo [P-Person 1] sitzengeblieben ist und ich halt wirklich überhaupt keine Sprachauswahloption hatte, die irgendwie sagt: ‚du musst jetzt auch mal mit dazukommen', das war schon irgendwie genial […] der Versuch dann über Gestiken das anders nochmal zu lösen, das fand ich gut." Ebenfalls hilfreich bei diesem Rätsel war, dass die Pointcloud aktiv genutzt werden konnte. Diese wurde von den Proband*innen durchweg als positiv empfunden, wenn auch in unterschiedlichem Maße als Repräsentation der P-Personen: Alle VR-Personen konnten die P-Personen wiedererkennen und hatten das Gefühl, sich mit ihnen einen Raum zu teilen. Manche VR-Personen berichteten davon, dass es ein interessantes, erschreckendes oder intimes Gefühl war, sich durch die Punktwolke hindurchzubewegen, andere Personen hatten an dieser Stelle allerding keine besonderen Emotionen. Eine VR-Person beschrieb die Pointcloud als „a little bit like a ghosty thing".

Abgesehen von der Notwendigkeit der Pointcloud für das Pressure-Plates-Rätsel war es auch hilfreich für die VR-Personen zu sehen, mit welchem Rätsel sich die P-Personen gerade beschäftigten, um daran mitzuhelfen. Diese Hilfestellung funktionierte auch umgekehrt, denn auch der Lichtpunkt auf dem Boden konnte gelegentlich den P-Personen dabei assistieren, welches Rätsel als nächstes gemacht werden musste und auch an der Stelle, an der das UV-Licht angeht, bemerkte eine Gruppe den Ohmwert an der Tafel erst, als der Lichtpunkt sich dorthin bewegte (da die VR-Person an derselben Stelle ihren Passwortblock abholen musste). Die Wahrnehmung der Repräsentation der VR-Person als Lichtpunkt war dabei sehr divers. Lediglich zwei Gruppen gelang es, ohne jeden Hinweis, den Punkt als Darstellung der fehlenden Person zu erkennen, wobei eine der Gruppen durch Vorwissen belastet war. Die Hinweise waren meist darauf beschränkt, durch Kommentare wie „Was ist das denn für ein Punkt auf dem Boden bei euch?", innerhalb der Fiktion des Spiels das Interesse am Punkt zu wecken. Eine der Gruppen dachte zunächst, dass sie den Lichtpunkt durch Drücken der Hexagons auf der LED-Wand steuern kann und wurde von diesem Irrweg abgebracht. Als nächste Eskalationsstufe von Hinweisen wurde gemutmaßt „vielleicht ist das ja eine Person." Nur einer Gruppe musste direkt gesagt werden, dass der Punkt die VR-Person darstellt. Vor den Hinweisen wurde der Punkt von einigen Gruppen für eine Art Bildschirmschoner oder Ähnliches gehalten und ignoriert. Obwohl das Konzept offenbar schwer

selbst zu erschließen war, konnten die Gruppen es alle sehr gut annehmen. Die Gruppe, der es ohne Vorwissen und Hinweise gelungen war, den Punkt zu identifizieren, nutzte sogleich die „Yes" und „No"-Felder auf dem Boden, um ihre Vermutung zu überprüfen und das weitere Vorgehen zu koordinieren. Insgesamt benutzten alle der aufgezeichneten Gruppen das Yes/No-System. Manche hielten die Flächen zunächst für Schaltflächen, mit denen interagiert werden müsste. Dies war nach Aussage einer Probandin durch die Fiktion des Smarthome-Systems begünstigt: „Weil, ich dachte auch erst, das ist irgendwas, was wir dann im Smarthome aktivieren oder deaktivieren." Durch die graphische Darstellung in der VR ergab sich auch dort für manche VR-Personen der Eindruck, dass es sich um interaktive Elemente handeln müsste.

Ein interessanter Aspekt der Wahrnehmung des Lichtpunktes, der sich erst im Laufe der Durchläufe zeigte und nicht vorgesehen war, war, dass einige Teilnehmende Hemmungen hatten, auf den Lichtpunkt zu treten. Hier kommt das Pressure-Plates-Rätsel wieder ins Spiel, das so vorgesehen war, dass die VR-Person durch ihre Position den genauen Ort anzeigt, an dem die P-Personen sich platzieren sollen, was erfordert, dass in der Realität auf den Lichtpunkt getreten wird. Hier mussten die VR-Personen andere Lösungen finden (Gestikulieren mit den Händen, weitergehen und erneut „Here"-Kommando auslösen) um die P-Personen an die richtigen Stellen zu bewegen, wenn diese den Punkt nicht betreten wollten. Eine Teilnehmerin begründete im Interview ihre Aversion damit, dass sie vermeiden wollte, einen Schatten zu werfen, der den Punkt verschwinden lässt. Interessant für die Frage nach der Cross-realen Interaktion ist die Begründung einer anderen Probandin: „ich hätte mir jetzt noch [VR-Person]s Brille vorgestellt und dann war das für mich halt die Repräsentation von [VR-Person] – wenn ich mit ihm spreche, dann stell' ich mich halt nicht drauf." Dies scheint auf eine starke Personifizierung der doch sehr abstrakten Darstellung hinzudeuten. Eine Probandin sagte im Interview über den Lichtpunkt zur VR-Person: „Für mich warst du's dann irgendwann." Weitere Untersuchungen, die sich spezifisch dieser Darstellungsform widmen, bieten sich an, um etwa die Grenze der Abstraktion auszutesten, ab der der Effekt der Personalisierung für die Mehrheit der Menschen auftritt. Hier sei nochmal auf die Tatsache verwiesen, dass die Nähe, bzw. das Überschneiden der Pointcloud-Darstellung mit der gefühlten eigenen Position offenbar ebenfalls für einige Leute auf der VR-Seite Hemmungen erzeugt, da sie erschreckend wirkt, und andere sogar von einem Gefühl der Intimität sprechen. Unter Betrachtung dieses Aspekts scheint man Interaktionen dieser Art zwischen nicht-physisch-präsenten Personen nicht ohne Weiteres wie in einem Multiplayer-Computerspiel lösen zu können, wo es mitunter üblich ist, dass mehrere Charaktere sich überschneiden und sich so simultan am gleichen Ort befinden. Vor dem Hintergrund von embodiment und sozialer Präsenz ist auch schon festgestellt worden, dass Menschen selbst in Interaktion mit virtuellen Avataren sensibel für die (kulturell abhängige) soziale Distanz sind, die Komfortzone, die für angenehme zwischenmenschliche (nicht intime) Begegnungen nicht unterschritten werden sollte [5]. Es kann sogar durch rein visuelle Simulation von Berührung ein subjektives Berührungsgefühl entstehen [20]. Eine Art virtuelle Etikette bildet sich bereits in rein virtuellen Räumen wie VR-Chat heraus [31], und eine cross-reale Etikette wird fraglos notwendig werden, sollte sich hier eine breitere Anwendung finden. Für Escape Rooms interessant wäre wiederum das Brechen solcher Etiketten. So ist es denkbar, als Emotional Challenge [15] die Spieler*innen nach einer längeren Phase der Gewöhnung vor eine Herausforderung zu stellen, bei der sie in die Punktwolke hineintreten müssen oder etwas – möglicherweise „gefährliches", wie ein Schwert – durch sie hindurchstecken. Dies passt nicht perfekt in die Kategorien, die Krekhov et al. etablieren, da es nicht direkt Fear/Disgust ist, was hier überwunden werden muss, aber weitere Versuche würden hier womöglich sogar eine Anpassung dieser Kategorie nahelegen.

Eine der Gruppen merkte im Interview an, dass die Hinweise der Spielleitung störend waren, da sie teilweise dazu aufforderten, eine Handlung zu unterbrechen und etwas anderes zu tun. Dies war größtenteils den besprochenen technischen Mängeln geschuldet, da der Spielfluss stark gestört wurde, um etwa die Aufmerksamkeit der VR-Person durch Kommentare der Spielleiter*innen für die Dauer des Delays an der richtigen Stelle zu halten, oder um die Position der VR-Person zu adjustieren, wenn diese an die Grenzen des real vorhandenen Spielbereichs geriet. Teilweise war es aber auch ein Problem des Designs, da an mehreren Punkten des Spiels aufgrund fehlender auditiver oder visueller Feedbacks die Aufmerksamkeit der Spieler*innen nicht organisch durch das Spiel selbst geleitet wurde, auch wenn dies teilweise notwendig war, etwa um die gegenseitige Interaktion am LED-Panel aufzuzeigen. In weiteren Iterationen müssten, zusätzlich zur Lösung der technischen Schwierigkeiten, diese Aspekte genauer unter die Lupe genommen und durch entsprechende Feedbacks ergänzt werden, um die Abhängigkeit von Hilfestellungen durch das Hilfesystem zu reduzieren, damit dieses nur für die eventuelle Hilfe beim Lösen der tatsächlichen Rätsel notwendig wird. Ebenso müsste mit den Timings der Freischaltung der möglichen Kommunikationswege, oder der Einführung komplett neuer Kommunikationskanäle experimentiert werden. Zwei der Interviewten schlugen etwa ein System vor, wodurch die VR-Person mit dem Controller zeichnen kann, was entweder auf dem Bildschirm im Raum oder auf dem Boden dargestellt würde.

Des Weiteren ließ sich beobachten, dass die Erfahrung mit Escape Rooms bzw. regulären Computerspielen einen erheblich stärkeren Einfluss auf alle Beteiligten hatte als zuerst antizipiert. Mangelnde Erfahrung mit VR führte bei einigen Teilnehmer*innen zu einer anfänglichen kompletten Starre, bevor ihnen erklärt wurde, dass sie sich durch das reguläre Bewegen ihrer realen Füße durch den virtuellen Raum bewegen können. Diese „Starre" konnte allerdings auch in der Realität beobachtet werden, wo eine Spielerin den Escape Room mit folgenden Worten kritisierte: „[…] dieses Anzunehmen, dass wenn man in ‚nem Raum steht, [man] einfach irgendwelche Knöpfe drückt."

Eine Gruppe freute sich sogar explizit über das Pressure-Plates-Rätsel, in dem konkrete visuelle und auditive „Befehle" gegeben wurden: „[…] irgendwie war das eher erleichternd, dass wir endlich wussten, [lacht] was wir machen sollten."

Diese Probleme, die bei Anfängern auftauchten, sorgten teilweise für einen sehr zähen Spielablauf, wobei alle Teilnehmenden trotz aller technischen Probleme ein generell positives Feedback für das Erlebnis gaben, was aber natürlich an der sehr persönlichen Atmosphäre und der dadurch kaum ausschließbaren sozialen Erwünschtheit liegen könnte.

Gleichzeitig waren erfahrenere Spieler*innen weniger zurückhaltend und fühlten sich sogar vom kompliziert aussehenden Widerstandsrätsel nicht abgeschreckt. Dieses – in der ersten Version des Raumes sogar noch um einiges kompliziertere – Rätsel führte bei den Escape Room-Profis sogar zu verstärktem Engagement, da die Herausforderung sie reizte.

Was sich zwischen den Gruppen ebenfalls stark unterschied war der Grad, zu dem sie sich auf die Geschichte und den Spiel-Aspekt einließen. Einige VR-Personen forderten die P-Personen auf, auf der geschlängelten Linie im Keller zu balancieren, da sie keinen Boden auf der Karte sahen. Die P-Personen von zwei Gruppen kamen dieser Aufforderung auch nach, obwohl sie natürlich sehen konnten, dass keine Gefahr drohte. Dieses

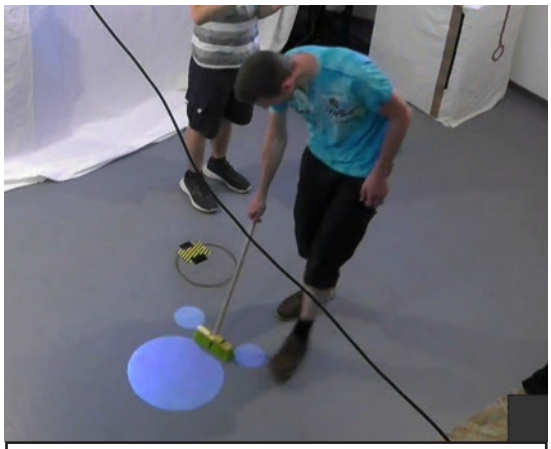


Abb. 35: Spaß am Spiel - eine P-Person fegt den Lichtpunkt

Eingehen auf das Spiel, ein spielerisches Element, schien den Proband*innen Spaß zu bereiten und ließ sich etwa auch im Fegen mit dem Besen wiederfinden. Hier ist möglicherweise ein positiver Aspekt der vagen Feedbacks zu sehen – da die Spieler*innen nicht genau wussten, was von den Sensoren erfasst wird, neigten sie zu einer Art magischem Denken und konnten sich leicht darauf einlassen, dass ja vielleicht doch etwas passieren könnte, wenn man fegt oder sich nicht auf der geschlängelten Linie hält. Dieses fast rollenspielartige Verhalten spielt auch eine Rolle beim Finale des Spiels: Den meisten Gruppen musste von den Spielleiter*innen gesagt werden, dass das Spiel nach Finden der VR-Person zu Ende ist, und ein eiferndes Feiern nach dem letzten Rätsel (der Froth) kam nicht immer zustande. Gruppen, die sich besser auf die thematischen Aspekte einlassen konnten – „mitspielten" – hatten hier eine bessere Erfahrung. Es wäre zu untersuchen, wie sich dieses Verhalten gezielter herbeiführen lässt.

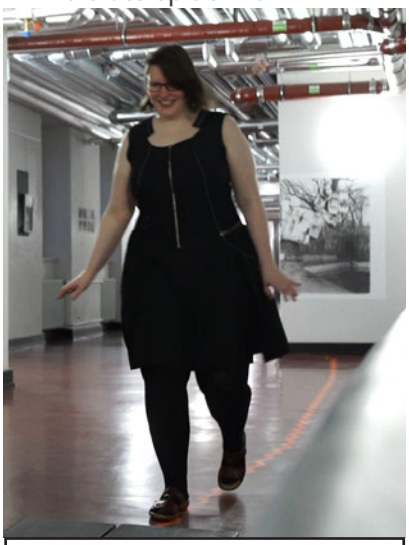


Abb. 36: Eine Spielerin balanciert auf der Linie

Einige Erwartungen des Teams wurden nicht bestätigt. So konnten selbst erfahrene Teams nicht immer ohne Hinweise die Funktion des PolRevs anhand des Aussehens erschließen. Die Druckknöpfe für den Schlüssel etwa sind zu unauffällig, um erkennbar zu machen, dass der Schlüssel nach dem Einführen noch nach unten gedrückt werden muss. Auch das Farbfeedback war uneindeutig – nach Lösen des Widerstandsrätsels leuchtet der PolRev gelb statt rot, doch die spezifische Farbe der LEDs erscheint sehr grün, was es für viele Teilnehmende so erschienen lies, als sei das Rätsel PolRev bereits komplett gelöst. Tatsächlich schalteten auch, trotz Anleitung, nicht alle Proband*innen den Schalter ein, um überhaupt ein Feedback durch das Licht bekommen zu können. Auch die Batterie wurde, vermutlich von Spielenden ohne viel Erfahrung mit 9V-Blockbatterien, nicht immer vollständig eingesteckt, sondern nur vorsichtig auf ihren Steckplatz gelegt, was dazu führte, dass der PolRev unzuverlässig anging und das visuelle Feedback erschwert wurde.

5.2 Optimierungsmöglichkeiten

Aus den Erlebnissen und Feedbacks lassen sich einige Verbesserungen der Rätsel und des Gesamtkonzepts ableiten. Zuerst ist hierfür sicherzustellen, dass zwischen den Audiogeräten, wie im letzten Durchlauf, weiterhin kein Delay herrscht, da es sich dabei um eine der zwei größten Störungen des Spielflusses handelte. Die zweite wichtige Optimierung wäre, den virtuellen Raum kleiner zu skalieren, falls keine ausreichend große Fläche als VR-Raum zur Verfügung steht, um zu gewährleisten, dass die VR-Person zu keinem Zeitpunkt die Rätsel nicht erreichen kann. Sollte man den Raum skalieren, müsste auch die Pointcloud kleiner skaliert und der Lichtpunkt auf dem Boden im P-Raum in seinen Bewegungen hochskaliert werden, um weiterhin den korrekten Standpunkt der VR-Person anzuzeigen.

Viele der anderen technischen Probleme ergaben sich teils aus hastiger Programmierung und konnten teilweise noch zum Ende des Projekts berichtigt werden (hier sei besonders der Fehler zu nennen, dass die VVVV-Nachrichten nicht korrekt weitergeleitet wurden und der Mini-Computer in E-45 gelegentlich die Videos nicht abspielte).

Ebenfalls den Lichtpunkt betreffend, ist bei der Beobachtung aufgefallen, dass der Projektionsbereich an einer Stelle in das vertikale Element des DNA-Scanner-Tasks ragte, wodurch sich bei Annäherung der Lichtpunkt nach oben verschob. Es wäre interessant auszuprobieren, ob solche vertikalen Elemente gezielt eingesetzt werden können, um interessantere Interaktionen zu ermöglichen. So ist es vorstellbar, dass eine vertikale Projektionsfläche vor dem LED-Wand-Tresor platziert wird, damit es bei Berührung in der VR so erscheint, als würden die Hand-Punkte nach oben zum Tresor greifen. Dadurch könnte die Aufmerksamkeit der P-Personen besser auf sich gezogen werden, und es wäre klarer, dass das Aufleuchten auf dem LED-Wand-Tresor durch den Lichtpunkt erzeugt wird.

Weiterhin müsste das Besen-Rätsel überarbeitet werden, um dauerhafte Funktionsfähigkeit zu garantieren und die aktuell lose hängende Technik vor Zerstörung, Manipulation und Wackelkontakten zu schützen. In diesem Zusammenhang könnte ebenfalls ein gyroskopischer Sensor sowie ein Trackingsystem verbaut werden, die es erlauben, das tatsächliche Benutzen des Besens festzustellen und in das Rätsel zu integrieren. Der Enthusiasmus, den die Teilnehmenden beim Fegen mit dem mysteriösen Gegenstand an den Tag legten, lässt darauf schließen, dass eine solche konkrete und haptische Nutzung sehr positiv aufgenommen werden würde.

Unter Betrachtung der Challenge-Analyse in Teil 4.4 wäre es zu untersuchen, ob die Rätsel teilweise umdesignt werden können oder neue Rätsel hinzugefügt, um die Verteilung der Challenges etwas gleichmäßiger zu gestalten. So könnte beispielsweise das Coderätsel einen mathematischen Anteil enthalten und dafür weniger auf *Observation/Searching* aufbauen, um die Art der Herausforderungen für P- und VR-Personen in ähnlicheren Verhältnissen zu gestalten. Ein Rätsel, das die fehlenden Challenges *Memorization/Reproduction* und *Comparison/Distinction* einbringt, wäre ebenfalls wünschenswert – in der VR wäre dies auch mit *Supernatural Abilities* kombinierbar, etwa in einem Rätsel, in dem sich der gesamte Raum magisch verändert und die Veränderungen Hinweise auf die Lösung geben. Der VR-Person müsste ohnehin mindestens ein zusätzliches Rätsel gegeben werden, da während des Lösens des Widerstandsrätsels sonst ein extremer Leerlauf auf der VR-Seite entsteht. Außerdem wäre es sinnvoll, in der VR Hinweise zum Lösen des Widerstandsrätsels zu hinterlegen, damit die VR-Person quasi als Kontrolle fungiert, die im Fall, dass das Widerstandsrätsel bis zum Erhalt des Smartphones immer noch nicht gelöst wurde, gegebenenfalls an Stelle der Spielleitenden den P-Personen Unterstützung bieten kann. Generell wäre es hilfreich, für das Widerstandsrätsel in einer neuen Iteration ein anderes Design zu finden, das weniger abschreckend ist, da viele Spieler*innen einen initialen Impuls hatten, zu sagen, dass sie das Rätsel nicht lösen können, es dann allerdings doch konnten, nachdem dieser anfängliche Stressmoment überwunden wurde.[18] Auch die Integration von Emotional Challenges, etwa wie in 5.1 besprochen durch Ausnutzen der sozialen Kopräsenz wäre wünschenswert.

[18] Ebenso müsste womöglich das Coderätsel simpler gestaltet oder besser unterfüttert werden, da hier ein ähnlicher Effekt auftritt – obwohl der Text eigentlich als einfache Sprache verständlich ist, wurde er von programmierunerfahrenen Proband*innen meist gar nicht richtig analysiert und die Blöcke quasi zufällig eingesetzt.

Von diesen technisch wichtigen Dingen abgesehen wären auch kosmetische Änderungen besonders am VR-Raum in Nachfolgeprojekten sinnvoll, da es besonders auf dieser Ebene noch erhebliches ungenutztes Potenzial gibt. Viele Teilnehmende waren (oft aufgrund mangelnder VR-Erfahrung) von dem bisher sehr spärlichen Design sehr beeindruckt, doch für Menschen, die schon häufiger VR-Anwendungen genutzt haben, ist z.B. das Fehlen einer runden Skybox extrem offensichtlich und wirkt generell unprofessionell. Teilweise ist das graphische Design auch für die Interaktion relevant: Die Poke-Interaktion des Radial Menus müsste klarer als Button dargestellt werden, die YES-NO-Felder auf dem Boden ebenso klar als nicht interaktiv und das Aufleuchten der von den P-Personen gedrückten Hexagons beim LED-Rätsel müsste deutlicher erscheinen, um nicht als Fehler missverstanden zu werden. Auch auf der realen Seite wäre bei diesem Rätsel ein zusätzliches auditives Feedback wünschenswert, das signalisiert, wann ein Symbol als „eingeloggt" gilt (nämlich nur dann, wenn es auch in im P-Raum berührt wurde und nicht etwa schon, wenn die VR-Person es berührt).

Eine optische Einheitlichkeit sowie ein dedizierter Raum – im Gegensatz zu einem temporär umgestalteten Seminarraum – wäre ebenfalls sinnvoll, damit sich die P-Personen sicher sein können, dass alle nicht explizit gekennzeichneten Elemente auch wirklich zum Raum gehören. Denn wie bereits erwähnt, wollten einige Spieler*innen den Lichtschalter zuerst nicht bedienen, wobei eine Gruppe dies damit begründete, dass sie dachten, der Schalter gehöre nicht zum Spiel.

Nicht zuletzt könnte man darüber nachdenken, wie man den Spielenden klarer macht, welche Rätsel gelöst werden müssen, um weiterzukommen und welche Rätsel noch weitere Elemente oder Informationen benötigen, die erst im Laufe des Spiels freigeschaltet werden. Insgesamt war ein oft bemerktes Element das fehlende Feedback – wobei auch zusätzliche audiovisuelle Mittel helfen könnten, wie etwa die erwähnten vertikalen Elemente oder zusätzliche Lautsprecher, die im Raum verteilt ermöglichen auditives Feedback von den entsprechenden Positionen in VR zu erhalten. Auch die Einheitlichkeit der bereits verwendeten Feedbackmethoden wäre zu überarbeiten. Eine VR-Person merkte etwa an, dass es irritierend war, dass nur bei der Interaktion mit dem LED-Wand-Tresor Controllervibration verwendet wurde.

5.3 Weitere Anwendungsmöglichkeiten und Ausblick

Die positiven Reaktionen der Teilnehmer*innen trotz der technischen Mängel des Prototypen lassen darauf schließen, dass eine weiterentwickelte Form des XcapeRooms sich in einem sich wirtschaftlich selbsttragenden Konzept als kommerzieller Escape Room bewähren könnte. Die Erkenntnisse aus dieser Arbeit, sowohl praktische Herangehensweisen aus der Planungs- und Konzeptionsphase als auch die Ergebnisse der Interviews lassen sich dabei verwenden, um ähnliche Escape-Room-Erfahrungen mit Cross-Reality-Interaktionen zu gestalten.

In diesem Kontext ließe sich auch die soziale Präsenzerfahrung zwischen den unterschiedlichen Realitätsmodalitäten weiter beobachten. Der Rahmen als fest installierter Escape Room würde es ermöglichen, eine größere Anzahl Proband*innen einzuladen und verschiedene Variablen zu iterieren. Eine methodische Untersuchung der verwendeten Avatare könnte wertvolle Ergebnisse für die Forschung an telepräsenter Kooperation beitragen. Insbesondere das Verfahren, die VR-Person als Lichtpunkte mit Hand-Repräsentation darzustellen, birgt aufgrund ihrer einfachen technischen Umsetzbarkeit großes Potenzial, etwa für Messen oder Kulturveranstaltungen.

Die verwendete Tiefenkamera wurde allerdings bereits vom Markt genommen und müsste durch eine adäquate Alternative ersetzt werden. Da die Forschung an Avataren aus volumetrischen Aufnahmen vielversprechende Ergebnisse für soziale Präsenz in VR birgt [3], kann davon ausgegangen werden, dass weitere Entwicklungen in diese Richtung folgen werden, die in zukünftigen Iterationen Anwendung finden würden.

Da sich Teile des XcapeRooms als besonders erfolgreich herausstellten, und als einzelne Rätsel aus dem Escape-Room-Kontext ausgekoppelt werden können, bietet sich zudem die Option der Präsentation dieser Elemente auf Fachmessen an. Insbesondere das LED-Wand-Tresor-Rätsel und das Pressure-Plates-Rätsel eignen sich aufgrund ihrer hohen Kooperativität und Mobilität, um in einem solchen Rahmen präsentiert zu werden.

Nicht zuletzt gibt es relevante Forschung dazu, ob und wie Escape Rooms in einem Bildungskontext Anwendung finden können [4, 24, 32]. Auch hier birgt XcapeRoom Potenzial, wenn die Rätsel wie Tutorials gestaltet werden, die aufeinander aufbauen. Bei diesen wäre das Ziel am Ende, zu verstehen, was z.B. ein Widerstand macht und wie man ihn liest und einbaut. Oder wenn das Coderätsel noch realistischer gestaltet würde und es dafür Scaffolding gäbe (was sowohl Übersetzungen der englischen Vokabeln als auch Hinweise zur Benutzung von Satzzeichen beim Programmieren sein könnten).

6 Fazit

Diese Arbeit widmete sich der Frage: Ist eine soziale Kopräsenz in einer Mixed-Reality-Situation, erzeugt durch volumetrische Avatare auf der einen und durch einen projizierten Avatar auf der anderen Seite, geeignet, um eine befriedigende Escape-Room-Erfahrung für eine über mehrere Orte verteilte Gruppe zu erzeugen? Um diese zu klären, wurde zunächst ein Überblick über den Stand des Wissens zu Escape Rooms und Mixed Reality gegeben. Escape Rooms wurden definiert als eine Ansammlung diverser Rätsel, bei denen häufig ein Thema im Mittelpunkt steht und deren herausragende Eigenschaft eben die Diversität der Rätsel und der für die Lösung selbiger notwendigen Fähigkeiten ist. Es wurde aufgezeigt, dass Escape Rooms in vielen Varianten, unter anderem auch Mixed und Virtual Reality existieren, und das Fehlen von Escape Rooms bemerkt, welche die herkömmliche stationäre Variante mit Virtual Reality kombinieren. Mixed Reality und besonders Cross-Reality-Interaktionen wurden beleuchtet und die Herangehensweise, volumetrische Avatare zu nutzen, als vielversprechend identifiziert.

Auch vor der Recherche existierte aufgrund von Vorwissen in den entsprechenden Bereichen der Verdacht, dass es einen Mangel an Cross-Reality-Interaktions-lastigen Escape Rooms gäbe – was sich in der theoretischen Untersuchung bestätigte – und es wurde ein Prototyp für einen Escape Room mit XR-Interaktionen entwickelt und gebaut. Anschließend wurde dieser mit Proband*innen aus dem universitären Umfeld erprobt und aus den informellen Interviews, die im Anschluss an die Probedurchläufe geführt wurden Rückschlüsse auf Stärken und Schwächen des Konzepts gezogen. Das zentrale Element der sozialen Präsenz durch die Avatare wurde dabei durchwegs positiv aufgenommen. Insbesondere die volumetrische Darstellung der P-Personen als Punktwolke war, trotz reduzierter Auflösung und abstrakter Farbgebung, in der Lage starke Gefühle der Kopräsenz auszulösen. Der Lichtpunkt im P-Raum war initial schwierig zu verstehen, aber nachdem er als VR-Person erkannt wurde, war auch er in der Lage, ein Gefühl der Kopräsenz zu erzeugen, und Kollaboration zu ermöglichen. Besonders das Hinzufügen von Lichtpunkten für die Position der Hände vermochte hier der doch

sehr abstrakten Darstellung einen natürlichen Charakter zu verleihen. Die positiven Rückmeldungen erlauben, die Forschungsfrage vorsichtig positiv zu beantworten.

Die Hypothese, dass die Spieler*innen die Avatare als Repräsentation ihres Teammitglieds annehmen und eine reibungslose Remote Collaboration entsteht, kann teils als bestätigt angesehen werden. Der erste Teilsatz, dass die Avatare angenommen werden, hat sich klar bestätigt, doch die Kollaboration verlief nicht immer reibungslos. Dies ist jedoch nach persönlicher Einschätzung, die sich durch die Gespräche mit den Proband*innen ergibt, in erster Linie den technischen Schwächen des Prototypen geschuldet. Eine weitere Iteration, idealerweise an einem festen Ort oder ein reduzierter Aufbau, der sich auf eine Kollaborationssituation konzentriert, könnte den zweiten Teil der Hypothese besser untersuchen. Hierfür käme auch eine quantitative Studie in Frage, bei der nur eine einzige konkrete Situation betrachtet und dafür eine Vielzahl an Teilnehmer*innen gefunden würde. Wobei hier für die Variable „Unterschiede im Spielverhalten" derart kontrolliert werden könnte, dass tatsächlich immer nur eine Person jeweils in der Realität oder VR ist, während ein*e Studienleiter*in die übrige Position einnehmen würde.

Für zukünftige Iterationen wurden Hinweise aufgeführt, die der Verbesserung des Escape-Room-Erlebnisses dienen. Insbesondere eine optimale Netzwerkverbindung sowie ein reibungsloses Bewegen für die VR-Person im realen Raum sind hierbei zu beachten, da alle anderen Design- und Technikfragen in Anbetracht solch derartig vordergründiger Probleme verblassen.

Insbesondere wurde bemerkt, dass sich die Cross-Reality-Interaktion spezifisch dafür eignet, emotional Challenges zu kreieren, die eine Übertretung der sozialen Distanz in einem sicheren Rahmen erfordern, da dies nicht tatsächlich, sondern nur in mediierter Form ablaufen würde.

Zusammenfassend lässt sich feststellen, dass XcapeRoom spannende Ansätze liefert, wie stationäre Escape Rooms um eine Dimension der Cross-Reality-Interaktion bereichert werden können.

Literaturverzeichnis

[1] 2CP GmbH. *TeamEscape Hamburg*. Retrieved September 7, 2022 from https://teamescape.com/hamburg/

[2] bmegli. 2021. *RNHVE Realsense Network Hardware Video Encoder. Realsense hardware video/depth encoding and streaming over custom MLSP protocol.* (2021). Retrieved September 11, 2022 from https://github.com/bmegli/realsense-network-hardware-video-encoder

[3] SungIk Cho, Seung-wook Kim, JongMin Lee, JeongHyeon Ahn, and JungHyun Han. 2020. Effects of volumetric capture avatars on social presence in immersive virtual environments. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, [S.l.], 26–34. DOI: https://doi.org/10.1109/VR46266.2020.1581170537418

[4] Samantha Clarke, Sylvester Arnab, Luca Morini, Oliver Wood, Kate O'Leary, Alex Masters, and Aikaterini Bourazeri. 2016. EscapED: A Framework for Creating Live-Action, Interactive Games for Higher/Further Education Learning and Soft Skills Development. In *Proceedings of the 10th European Conference on Games Based Learning*, 968–972.

[5] Letizia Della Longa, Irene Valori, and Teresa Farroni. 2021. Interpersonal Affective Touch in a Virtual World: Feeling the Social Presence of Others to Overcome Loneliness. *Frontiers in psychology* 12, 795283. DOI: https://doi.org/10.3389/fpsyg.2021.795283

[6] EscapeReality GmbH. *EscapeDiem*. Retrieved September 7, 2022 from https://escapediem.de/

[7] GeorgeAdamon. 2020. *Oculus Quest - Intel RealSense Integration. Integration of Intel RealSense depth cameras with the Oculus Quest standalone VR headset using Unity.* (2020). Retrieved September 11, 2022 from https://github.com/GeorgeAdamon/quest-realsense

[8] Matt Gottsacker. 2022. Balancing Realities by Improving Cross-Reality Interactions. In *2022 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*. IEEE, [S.l.], 944–945. DOI: https://doi.org/10.1109/VRW55335.2022.00323

[9] Intel. *Intel® RealSense™ LiDAR Camera L515*. Retrieved September 8, 2022 from https://ark.intel.com/content/www/us/en/ark/products/201775/intel-realsense-lidar-camera-l515.html

[10] Intel. *Message to Customers*. Retrieved September 8, 2022 from https://www.intelrealsense.com/message-to-customers/

[11] James Whale. 1931. *Frankenstein*

[12] Jason Tsai. 2022. *With an Assist from Oculus Quest 2, 2022 AR/VR Device Shipments Revised Up to 14.19 Million Units, Says TrendForce* (2022). Retrieved September 11, 2022 from https://www.trendforce.com/presscenter/news/20220216-11122.html

[13] Jonas Plümke. 2021. *Nutzungs- und Gestaltungsmöglichkeiten von XR-Techniken. Experiment einer*

Theaterperformance. Bachelorthesis. Fakultät Design, Medien und Information. Hochschule für Angewandte Wissenschaften, Hamburg.

[14] Zoi Karageorgiou, Eirini Mavrommati, Panagiotis Fotaris, and Eleni Christopoulou. 2020. Escape room in education: Evaluating using Experience Pyramid Model. In *2020 5th South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference (SEEDA-CECNSM)*. IEEE, 1–8. DOI: https://doi.org/10.1109/SEEDA-CECNSM49515.2020.9221798

[15] Andrey Krekhov, Katharina Emmerich, Ronja Rotthaler, and Jens Krueger. 2021. Puzzles Unpuzzled. *Proc. ACM Hum.-Comput. Interact.* 5, CHI PLAY, 1–24. DOI: https://doi.org/10.1145/3474696

[16] Lisa Spira. 2022. *US Escape Room Industry Report – July 2022* (2022). Retrieved September 11, 2022 from https://roomescapeartist.com/2022/07/30/escape-room-industry-report-2022/

[17] Carl Matthes, Tim Weissker, Emmanouil Angelidis, Alexander Kulik, Stephan Beck, Andre Kunert, Anton Frolov, Sandro Weber, Adrian Kreskowski, and Bernd Froehlich. 2019. The Collaborative Virtual Reality Neurorobotics Lab. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR 2019)*. *Osaka, Japan, 23-27 March 2019*. IEEE, Piscataway, NJ, 1671–1674. DOI: https://doi.org/10.1109/VR.2019.8798289

[18] Dominik May. 2020. Cross Reality Spaces in Engineering Education – Online Laboratories for Supporting International Student Collaboration in Merging Realities. *Int. J. Onl. Eng.* 16, 03, 4. DOI: https://doi.org/10.3991/ijoe.v16i03.12849

[19] Mel Slater and Martin Usoh. 1994. Body Centred Interaction in Immersive Virtual Environments. In *Artificial Life and Virtual Reality*. John Wiley and Sons, 125–148.

[20] Manuel Mello, Martina Fusaro, Gaetano Tieri, and Salvatore M. Aglioti. 2022. Wearing same- and opposite-sex virtual bodies and seeing them caressed in intimate areas. *Quarterly journal of experimental psychology (2006)* 75, 3, 461–474. DOI: https://doi.org/10.1177/17470218211031557

[21] Paul Milgram and Fumio Kishino. 1994. A Taxonomy of Mixed Reality Visual Displays. *IEICE Trans. Information Systems* vol. E77-D, no. 12, 1321–1329.

[22] S. Nicholson. 2015. *Peeking behind the locked door: A survey of escape room facilities.* (2015). Retrieved from http://scottnicholson.com/pubs/erfacwhite.pdf

[23] Catherine S. Oh, Jeremy N. Bailenson, and Gregory F. Welch. 2018. A Systematic Review of Social Presence: Definition, Antecedents, and Implications. *Frontiers in robotics and AI* 5, 114. DOI: https://doi.org/10.3389/frobt.2018.00114

[24] Ioannis Papadopoulos and Eirini Tenta. 2021. Escape Rooms as a Collaborative Problem-Solving Environment. *International Journal of Game-Based Learning* 11, 4, 57–71. DOI: https://doi.org/10.4018/IJGBL.2021100103

[25] Adalberto L. Simeone, Mohamed Khamis, Augusto Esteves, Florian Daiber, Matjaž Kljun, Klen Copic

Pucihar, Poika Isokoski, and Jan Gugenheimer. 2020. International Workshop on Cross-Reality (XR) Interaction. In *Companion Proceedings of the 2020 Conference on Interactive Surfaces and Spaces*. ACM Digital Library. Association for Computing Machinery, New York,NY,United States, 111–114. DOI: https://doi.org/10.1145/3380867.3424551

[26] Mel Slater and Maria V. Sanchez-Vives. 2014. Transcending the Self in Immersive Virtual Reality. *Computer* 47, 7, 24–30. DOI: https://doi.org/10.1109/MC.2014.198

[27] Steel Crate Games. 2015. *Keep Talking and Nobody Explodes*. Steel Crate Games

[28] Frank Steinicke. 2016. *Being Really Virtual*. *Immersive Natives and the Future of Virtual Reality*, Cham.

[29] Jonathan Steuer. 1995. Defining Virtual Reality. Dimensions Determining Telepresence. In *Communication in the Age of Virtual Reality*, Frank Biocca and Mark Levy, Eds., Hoboken, 33–56.

[30] Steven Lisberger. 1982. *Tron*

[31] Straszfilms. 2021. *Identity, Gender, and VRChat*. *(Why is everyone in VR an anime girl?)* (2021). Retrieved September 7, 2022 from https://www.youtube.com/watch?v=5v_Dl7i4Bcw&t=672s

[32] Alice Veldkamp, Johanna Rebecca Niese, Martijn Heuvelmans, Marie⊠Christine P. J. Knippels, and Wouter R. van Joolingen. 2022. You escaped! How did you learn during gameplay? *Brit J Educational Tech* 53, 5, 1430–1458. DOI: https://doi.org/10.1111/bjet.13194

[33] Vorhaus Advisors. 2022. *Engagement with immersive activities among adults in the United States as of December 2021, by age group* (2022). Retrieved September 11, 2022 from https://www.statista.com/statistics/1310556/immersive-activities-engagement-age/

[34] Harald Warmelink, Igor Mayer, Jessika Weber, Bram Heijligers, Mata Haggis, Erwin Peters, and Max Louwerse. 2017. AMELIO. In *Extended Abstracts Publication of the Annual Symposium on Computer-Human Interaction in Play*. ACM, New York, NY, USA, 111–123. DOI: https://doi.org/10.1145/3130859.3131436

Anhang

Anhangsverzeichnis

1) GameMaster Code i

2) LED-Wand-Tresor Code ix

3) Coderätsel Code xv

4) OSC Code xv

5) Processing Block zur Pointcloud Bereinigung xvi

6) Pointcloud Code xviii

7) VVVV Sketch zur Steuerung von Lichtpunkt und Audiosamples xxi

8) Eidestattliche Erklärung xxiii

Digitale Anhänge

1) Pointcloud in Bewegung (Videofile)

2) Lichtpunkt in Bewegung (Videofile)

1) GameMaster Code

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using extOSC;
using System.Net.Sockets;
using System.IO.IsolatedStorage;
using UnityEditor.Rendering;
using Oculus.Platform;
using OVR.OpenVR;
using System.Threading.Tasks;
using System.Linq;
using extOSC.Core;
using UnityEngine.Assertions.Must;
using Intel.RealSense.Math;
using System.Xml.Linq;

namespace XcapeRoom
{
  public enum GameLanguage
  {
    de,
    en
  }
  public class GameState
  {
    public GameState()
    {

    }
    public GameState(bool setAll)
    {
      Initialized = setAll;
      VRBoxOn = setAll;
      Alarm = setAll;
      VRBoxReset = setAll;
      LightsOn = setAll;
      ScaleButtonPressed = setAll;
      SwabsInserted = setAll;
      CodePuzzleSolved = setAll;
```

```
      PointcloudOn = setAll;
      WallDown = setAll;
      BroomActivated = setAll;
      TouchBoardSolved = setAll;
      WristMenuPickedUp = setAll;
      PartyMode = setAll;
      PhoneActive = setAll;
      KeyPrinted = setAll;
    }
    public bool Initialized = false;
    public bool VRBoxOn = false;
    public bool Alarm = false;
    public bool VRBoxReset = false;
    public bool LightsOn = false;
    public bool ScaleButtonPressed = false;
    public bool SwabsInserted = false;
    public bool CodePuzzleSolved = false;
    public bool PointcloudOn = false;
    public bool WallDown = false;
    public bool BroomActivated = false;
    public bool TouchBoardSolved = false;
    public bool WristMenuPickedUp = false;
    public bool PartyMode = false;
    public bool PhoneActive = false;
    public bool KeyPrinted = false;
  }
public class GameMaster : MonoBehaviour
{
  private enum CommandTarget
  {
    lightswitch
  }
  public GameLanguage Language;
  public OSCReceiver MasterReciever;
  public OSCTransmitter TVTransmitter;
  public OSCTransmitter SamplerTransmitter;
  public OSCTransmitter LightswitchTransmitter;
  public OSCTransmitter VRBoxTransmitter;
   public static GameMaster Instance { get; private set; }
  public OVRCameraRig m_OVRCameraRig;
  public GameObject InteractionSampleRig;
  public GameObject PasswordField;
  public GameObject SeaWall;
  public GameObject ParticleTrash;

  public RoomMicrophoneListener RoomMicrophoneListener;
  public GameObject Pointcloud;
  public HueLamp WhiteHueOne;
  public HueLamp WhiteHueTwo;
  public HueLamp ColorHue;
  public GameState State {get; private set;} = new GameState();
  public GameObject PartySlider;
  public GameObject Room;
  public GameObject LightswitchObject;
  public GameObject CodeCloud;
  public TouchboardMaster SensorboardController;
  public GameObject SchoolMap;
  private int vrBoxActive = 0;
  private bool LightswitchConfirmed;
  private OSCBundle samplerBundle;

   private void Awake()
    {
      if(Instance == null)
      {
        Instance = this;
        DontDestroyOnLoad(gameObject);
      }
      else
      {
        Destroy(gameObject);
      }
    }


  // Start is called before the first frame update
  void Start()
  {
    MasterReciever.Bind("/recenter", ResetPositionOSC);
    MasterReciever.Bind("/idle", IdleOSC);
    MasterReciever.Bind("/lichtschalter", LightswitchOSC);
    MasterReciever.Bind("/thunder", ThunderOSC);
    MasterReciever.Bind("/lightswitch", LightswitchOSC);
    MasterReciever.Bind("/vrbox", VRBoxOSC);
    MasterReciever.Bind("/height", HeightOSC);
    MasterReciever.Bind("/x", XOSC);
    MasterReciever.Bind("/z", ZOSC);
    MasterReciever.Bind("/broom", ManualBroomOSC);
    MasterReciever.Bind("/roomsound", RoomSoundOSC);
```

```
    LightswitchObject.SetActive(false);
    Room.SetActive(false);
    SensorboardController.SetAllOff();

    CodeCloud.SetActive(true);

    var thunderMessage = OSCMessage.Create("/Thunderstorm");
    thunderMessage.AddValue(OSCValue.Int(0));
    var alarmMessage = OSCMessage.Create("/facility-alarm");
    thunderMessage.AddValue(OSCValue.Int(0));
    var voiceMessageDE = OSCMessage.Create("/IntroVoicetrack_de");
    thunderMessage.AddValue(OSCValue.Int(0));
    var voiceMessageEN = OSCMessage.Create("/IntroVoicetrack_en");
    thunderMessage.AddValue(OSCValue.Int(0));
    samplerBundle = new OSCBundle(thunderMessage, alarmMessage, voiceMessageDE, voiceMessageEN);
    //ScreenIdle();

  }

  // Update is called once per frame
  void Update()
  {

    if(Input.GetKeyDown("r")){
      //OVRManager.display.RecenterPose();
      ResetPosition();
      Debug.Log("Recenter");
    }
    if(Input.GetKeyDown("a")){
      RoomMicrophoneListener.StartRoomAudio();
    }
    if(Input.GetKeyDown("s")){
      RoomMicrophoneListener.StopRoomAudio();
    }
    if(Input.GetKeyDown("t")){
      Thunder();
    }
    if(Input.GetKeyDown("q")){
      StartAlarm();
    }
    if(Input.GetKeyDown("v")){
      GameManager.Instance.CallPhone();
    }

    if(Input.GetKeyDown(KeyCode.Alpha0)){
      SendScreenMessage(0);
    }
    if(Input.GetKeyDown(KeyCode.Alpha1)){
      SendScreenMessage(1);
    }
    if(Input.GetKeyDown(KeyCode.Alpha2)){
      SendScreenMessage(2);
    }
    if(Input.GetKeyDown(KeyCode.Alpha3)){
      SendScreenMessage(3);
    }
    if(Input.GetKeyDown(KeyCode.Alpha4)){
      SendScreenMessage(4);
    }
    if(Input.GetKeyDown(KeyCode.Alpha5)){
      SendScreenMessage(5);
    }
    if(Input.GetKeyDown(KeyCode.Alpha6)){
      SendScreenMessage(6);
    }
    if(Input.GetKeyDown(KeyCode.Alpha7)){
      SendScreenMessage(7);
    }
    if(Input.GetKeyDown(",")){
      SendSamplerBundle(Sounds.ConfirmationMessage);
    }
    if(Input.GetKeyDown(".")){
      SendSamplerBundle(Sounds.WaitingMessage);
    }
  }

    public void ResetPosition()
  {
    float currentRotY = m_OVRCameraRig.centerEyeAnchor.eulerAngles.y;
    float targetRotY = 90.0f;
    float difference = targetRotY - currentRotY;
    m_OVRCameraRig.transform.Rotate(0, difference, 0);
    Vector3 currentPosition = m_OVRCameraRig.centerEyeAnchor.localPosition;
    m_OVRCameraRig.trackingSpace.localPosition = new Vector3(OffsetValue(currentPosition.x), 0, OffsetVal-
ue(currentPosition.z));
  }
  private void ManualBroomOSC(OSCMessage incoming)
  {
    Debug.Log("Manual Broom Called");
```

```
  PasswordField.SetActive(true);
  SeaWall.SetActive(true);
  ParticleTrash.SetActive(false);
  SetBroomActivated();

}
private void RoomSoundOSC(OSCMessage incoming)
{
  Debug.Log("RoomSoundOSC called");
  if(incoming.Values[0].IntValue == 0)
  {
    RoomMicrophoneListener.StopRoomAudio();
  }
  if(incoming.Values[0].IntValue == 1)
  {
    RoomMicrophoneListener.StartRoomAudio();
  }

}

private void HeightOSC(OSCMessage incoming)
{
  Debug.Log("Height called");
  Debug.Log("HeightOSC incoming: "+incoming);
  var value = incoming.Values[0].FloatValue;
  InteractionSampleRig.transform.Translate(new Vector3(0, value,0));
}
private void XOSC(OSCMessage incoming)
{
  Debug.Log("XOSC called");
  Debug.Log("XOSC incoming: "+incoming);
  var value = incoming.Values[0].FloatValue;
  InteractionSampleRig.transform.Translate(new Vector3(value, 0,0));
}
private void ZOSC(OSCMessage incoming)
{
  Debug.Log("ZOSC called");
  Debug.Log("ZOSC incoming: "+incoming);
  var value = incoming.Values[0].FloatValue;
  InteractionSampleRig.transform.Translate(new Vector3(0, 0,value));
}
private void VRBoxOSC(OSCMessage incoming)
{
  Debug.Log(incoming);
  if(incoming.Values[0].StringValue == "restarted")
  {
    var outgoing = OSCMessage.Create("/vrbox");
    outgoing.AddValue(OSCValue.Int(0));
    VRBoxTransmitter.Send(outgoing);
    // if(!State.VRBoxReset && State.Alarm)
    // {
    //     StartCoroutine("CountdownVRBoxActive");
    // }
    // else
    // {
      if(State.Alarm)
      {
        Debug.Log("VRBox fully reset!");
        CancelInvoke("StartAlarm");
        State.Alarm = false;
      }

    // }
  }
  if(incoming.Values[0].StringValue == "active")
  {
    vrBoxActive = 3;
  }

}
// private IEnumerator CountdownVRBoxActive()
// {
//     vrBoxActive = 3;
//     while(vrBoxActive > 0)
//     {
//         vrBoxActive--;
//         yield return new WaitForSecondsRealtime(1f);
//     }
//     State.VRBoxReset = true;
// }
private void ResetPositionOSC(OSCMessage incoming)
{
  Debug.Log("Recenter");
  ResetPosition();

}
private void LightswitchOSC(OSCMessage incoming)
{
  Debug.Log(incoming);
```

```
  if(incoming.Values[0].StringValue == “button_pressed”)
  {
    Lightswitch();
  }
  if(incoming.Values[0].StringValue == “shutting_down”);
  {
    LightswitchConfirmed = true;
  }
}
private async Task SendOSCUntilConfirmedAsync(CommandTarget target, OSCMessage message)
    {
      bool confirmation = false;
      while(!confirmation)
      {
        switch(target)
        {
          case CommandTarget.lightswitch:
            if(LightswitchConfirmed)
            {
              confirmation = true;
            }
            LightswitchConfirmed = false;
            LightswitchTransmitter.Send(message);
            break;

        }
        await Task.Delay(1000);
      }

    }

private async void Lightswitch()
{
  var message = OSCMessage.Create(“/lightswitch”);
  message.AddValue(OSCValue.Int(0));
  //LightswitchTransmitter.Send(message);
  LightswitchObject.SetActive(true);
  await SendOSCUntilConfirmedAsync(CommandTarget.lightswitch, message);
  await FadeoutParticles();
}

public void SendScreenMessage(int videoNumber)
{
  var message = OSCMessage.Create(“/tv”);
  message.AddValue(OSCValue.Int(videoNumber));//button
  TVTransmitter.Send(message);
  Debug.Log(“Sent Screen Message: “ + videoNumber);
}
public async void LightswitchVR()
{
  Debug.Log(„Licht Schalted!“);
  //ResetPosition();

  SendScreenMessage(0);
  Room.SetActive(true);
  LightswitchObject.SetActive(false);
  RoomMicrophoneListener.StartRoomAudio();
  LightsOn();

}
private async Task FadeoutParticles()
{
  var particleSystem = CodeCloud.GetComponent<ParticleSystem>();
  var main = particleSystem.main;
  main.maxParticles = 0;
  while(particleSystem.particleCount > 0)
  {
    await Task.Delay(100);
  }
  CodeCloud.SetActive(false);
}
private void IdleOSC(OSCMessage incoming)
{
  ScreenIdle();
}

public void ScreenIdle()
{
  SendScreenMessage(4);
  //LightsOn();

}
void LightsOn()
{
  State.LightsOn = true;
  StopCoroutine(“DimLights”);
  WhiteHueOne.on = true;
  WhiteHueTwo.on = true;
  ColorHue.on = true;
```

```
    WhiteHueOne.color = Color.white;
    WhiteHueTwo.color = Color.white;
    ColorHue.color = Color.white;
  }

  private void ThunderOSC(OSCMessage incoming)
  {
    Thunder();
  }

  private void Thunder()
  {
    Debug.Log("Thunderstorm Called");
    SendScreenMessage(5);
    // message = OSCMessage.Create("/Thunderstorm");
    // message.AddValue(OSCValue.Int(1));// 1 = play
    SendSamplerBundle(Sounds.Thunderstorm);

     // (samplerBundle.Packets.First(p => p.Address == message.Address) as OSCMessage).Values[0] = mes-
sage.Values[0];
    // SamplerTransmitter.Send(samplerBundle);
    StartCoroutine("DimLights");
    InvokeRepeating("StartAlarm", 80f, 2f);
    State.Alarm = true;
    Invoke("StartVoiceMessage", 93f);

  }

  public enum Sounds
  {
    Thunderstorm,
    FacilityAlarm,
    IntroVoicetrack_de,
    Introvoicetrack_en,
    Partysong,
    ConfirmationMessage,
    WaitingMessage
  }

  public void SendSamplerBundle(Sounds sound)
  {
    var thunderMessage = OSCMessage.Create("/Thunderstorm");
    thunderMessage.AddValue(OSCValue.Int(sound == Sounds.Thunderstorm ? 1 : 0));
    var alarmMessage = OSCMessage.Create("/facility-alarm");
    alarmMessage.AddValue(OSCValue.Int(sound == Sounds.FacilityAlarm ? 1 : 0));
    var voiceMessageDE = OSCMessage.Create("/IntroVoicetrack_de");
    voiceMessageDE.AddValue(OSCValue.Int(sound == Sounds.IntroVoicetrack_de ? 1 : 0));
    var voiceMessageEN = OSCMessage.Create("/IntroVoicetrack_en");
    voiceMessageEN.AddValue(OSCValue.Int(sound == Sounds.Introvoicetrack_en ? 1 : 0));
    var partyMessage = OSCMessage.Create("/partysong");
    partyMessage.AddValue(OSCValue.Int(sound == Sounds.Partysong ? 1 : 0));
    var confirmationMessage = OSCMessage.Create("/xconfirmation_msg");
    confirmationMessage.AddValue(OSCValue.Int(sound == Sounds.ConfirmationMessage ? 1 : 0));
    var waitingMessage = OSCMessage.Create("/waiting_msg");
    waitingMessage.AddValue(OSCValue.Int(sound == Sounds.WaitingMessage ? 1 : 0));
    var localSamplerBundle = new OSCBundle();
    localSamplerBundle.AddPacket(thunderMessage);
    localSamplerBundle.AddPacket(alarmMessage);
    localSamplerBundle.AddPacket(voiceMessageDE);
    localSamplerBundle.AddPacket(voiceMessageEN);
    localSamplerBundle.AddPacket(partyMessage);
    localSamplerBundle.AddPacket(confirmationMessage);
    localSamplerBundle.AddPacket(waitingMessage);
    SamplerTransmitter.Send(localSamplerBundle);
    Invoke("ResetSampler",0.5f);
  }

  private void ResetSampler()
  {
    var thunderMessage = OSCMessage.Create("/Thunderstorm");
    thunderMessage.AddValue(OSCValue.Int(0));
    var alarmMessage = OSCMessage.Create("/facility-alarm");
    alarmMessage.AddValue(OSCValue.Int(0));
    var voiceMessageDE = OSCMessage.Create("/IntroVoicetrack_de");
    voiceMessageDE.AddValue(OSCValue.Int(0));
    var voiceMessageEN = OSCMessage.Create("/IntroVoicetrack_en");
    voiceMessageEN.AddValue(OSCValue.Int(0));
    var partyMessage = OSCMessage.Create("/partysong");
    partyMessage.AddValue(OSCValue.Int(0));
    var confirmationMessage = OSCMessage.Create("/xconfirmation_msg");
    confirmationMessage.AddValue(OSCValue.Int(0));
    var waitingMessage = OSCMessage.Create("/waiting_msg");
    waitingMessage.AddValue(OSCValue.Int(0));
    var localSamplerBundle = new OSCBundle();
    localSamplerBundle.AddPacket(thunderMessage);
    localSamplerBundle.AddPacket(alarmMessage);
    localSamplerBundle.AddPacket(voiceMessageDE);
    localSamplerBundle.AddPacket(voiceMessageEN);
```

```
    localSamplerBundle.AddPacket(partyMessage);
    localSamplerBundle.AddPacket(confirmationMessage);
    localSamplerBundle.AddPacket(waitingMessage);
    SamplerTransmitter.Send(localSamplerBundle);
}
public void StopSampler()
{
    var stopMessage = OSCMessage.Create("/Stop");
    stopMessage.AddValue(OSCValue.Int(1));
    SamplerTransmitter.Send(stopMessage);
    Invoke("StopSamplerReset",0.5f);
}

private void StopSamplerReset()
{
    var stopMessage = OSCMessage.Create("/Stop");
    stopMessage.AddValue(OSCValue.Int(0));
    SamplerTransmitter.Send(stopMessage);
}

private void StartAlarm()
{
    Debug.Log(„ALARM!");
    // var message = OSCMessage.Create("/facility-alarm");
    // message.AddValue(OSCValue.Int(1)); // 1 = play
    // SamplerTransmitter.Send(message);
    SendSamplerBundle(Sounds.FacilityAlarm);
    if(State.VRBoxReset == false)
    {
        var vrBoxMessage = OSCMessage.Create("/vrboxAlarm");
        vrBoxMessage.AddValue(OSCValue.Int(1));
        VRBoxTransmitter.Send(vrBoxMessage);
    }
    ColorHue.on = false;
    WhiteHueOne.on = false;
    WhiteHueTwo.on = false;

}

private void StartVoiceMessage()
{
    // string oscAddress = "/IntroVoicetrack_";
    if(Language == GameLanguage.de)
    {
        // oscAddress.Concat("de");
        SendSamplerBundle(Sounds.IntroVoicetrack_de);
    }
    else if(Language == GameLanguage.en)
    {
        // oscAddress.Concat("en");
        SendSamplerBundle(Sounds.Introvoicetrack_en);
    }
    // var message = OSCMessage.Create(oscAddress);
    // message.AddValue(OSCValue.Int(1)); // 1 = play
    // SamplerTransmitter.Send(message);
}

private IEnumerator DimLights()
{
    float value = 1f;
    while(value > 0)
    {
        var color = new Color(1f,1f,1f,value);
        WhiteHueOne.color = color;
        WhiteHueTwo.color = color;
        ColorHue.color = color;
        value -= 0.01f;
        yield return new WaitForSeconds(0.5f);

    }
    WhiteHueOne.on = false;
    WhiteHueTwo.on = false;
    ColorHue.on = false;
    yield return new WaitForSeconds(0.5f);
    WhiteHueOne.on = false;
    WhiteHueTwo.on = false;
    ColorHue.on = false;
    yield return new WaitForSeconds(0.5f);
    WhiteHueOne.on = false;
    WhiteHueTwo.on = false;
    ColorHue.on = false;
}

float OffsetValue(float value)
{
    if (value != 0)
    {
        value *= -1;
```

```
    }
    return value;
  }

  public void SetSwabsInserted()
  {
    if(State.SwabsInserted == true)
    {
      return;
    }
    Debug.Log("SetSwabsInserted() called");
    State.SwabsInserted = true;
    SendSamplerBundle(Sounds.WaitingMessage);
    if(State.CodePuzzleSolved)
    {
      SetPointcloudOn();
    }
    Invoke("ScreenIdle", 20f);
  }
  public void SetCodePuzzleSolved()
  {
    Debug.Log("SetCodePuzzleSolved() called");
    State.CodePuzzleSolved = true;
    if(State.SwabsInserted)
    {
      SetPointcloudOn();
      SendSamplerBundle(Sounds.ConfirmationMessage);
    }
  }
  public void SetPointcloudOn()
  {
    Debug.Log("SetPointcloudOn() called");
    State.PointcloudOn = true;
    Pointcloud.SetActive(true);
  }
  public void SetWallDown()
  {
    Debug.Log("SetWallDown() called");
    State.WallDown = true;
  }
  public void SetBroomActivated()
  {
    Debug.Log("SetBroomActivated() called");
    State.BroomActivated = true;
    //TODO: Enable Password-Input
  }
  public void SetWristMenuPickedUp()
  {
    Debug.Log("SetWristMenuPickedUp() called");
    State.WristMenuPickedUp = true;
  }
  public void SetPartyMode()
  {
    Debug.Log("SetPartyMode() called");
    State.PartyMode = true;
    PartySlider.SetActive(true);
  }

  public void SetTouchBoardSolved()
  {
    Debug.Log("SetTouchboardSolved()) called");
    State.TouchBoardSolved = true;
  }
  public void SetPhoneActive()
  {
    Debug.Log("SetPhoneActive() called");
    State.PhoneActive = true;
    GameManager.Instance.CallPhone();
    //RoomMicrophoneListener.StartRoomAudio();
    RoomMicrophoneListener.StopRoomAudio();
  }
  public void SetKeyPrinted()
  {
    SchoolMap.SetActive(true);
    State.KeyPrinted = true;
  }
  public void SetScaleButtonPressed()
  {
    if(State.LightsOn)
    {
      State.ScaleButtonPressed = true;
      SendScreenMessage(1);
    }
  }
}
}
```

2) LED-Wand-Tresor Code

```
#include <FastLED.h>
#include <SerialCmd.h>

#include <Wire.h>
#include “Adafruit_MPR121.h”

#ifndef _BV
#define _BV(bit) (1 << (bit))
#endif

#define LED_PIN 3
#define NUM_LEDS 72

#define LOCK_PIN 12

//#define DEBUG

enum colors{
  OFF, GREEN, RED, WHITE, VIOLET
};

CHSV colorValues[]{
  {0,0,0},
  {100,255,200},
  {12,255,255},
  {0,0,200},
  {180,255,255}
};

struct button {
  colors color;
  uint8_t lights[5];
  bool stayOn;
};

CRGB leds[NUM_LEDS];

bool isSwitched = false;
bool ledOn = false;
short ledColor = 0;

button Buttons[] = {
  {WHITE, {7,8,9,13,14}, false},
  {WHITE, {26,27,28,29,30}, false},
  {WHITE, {4,5,6,15,16}, false},
  {WHITE, {23,24,25,31,32}, false},
  {WHITE, {46,47,48,49,50}, false},
  {WHITE, {20,21,22,33,34}, false},
  {WHITE, {43,44,45,51,52}, false},
  {WHITE, {17,18,19,35,36}, false},
  {WHITE, {40,41,42,53,54}, false},
  {WHITE, {63,64,65,69,70}, false},
  {WHITE, {37,38,39,55,56}, false},
  {WHITE, {60,61,62,71,72}, false},
  {WHITE, {1,2,3,0,0}, false},
  {WHITE, {10,11,12,0,0}, false},
  {WHITE, {57,58,59,0,0}, false},
  {WHITE, {66,67,68,0,0}, false}
};

short EckeLinks[] = {1,2,3};
short EckeOben[] = {10,11,12};
short EckeUnten[] = {57,58,59};
short EckeRechts[] = {66,67,68};

bool isFading = false;
bool fadeUp = false;
bool isHalfFade = false;
button currentButton;
char currentColor;
int currentSpeed;
CHSV color; //current Color


Adafruit_MPR121 cap[2] = {Adafruit_MPR121(), Adafruit_MPR121()};

// Keeps track of the last pins touched
// so we know when buttons are ‘released’
uint16_t lasttouch = 0;
uint16_t currtouch = 0;
uint16_t lasttouchTwo = 0;
uint16_t currtouchTwo = 0;

SerialCmd SeCom ( Serial );

void setup() {
```

```
  // put your setup code here, to run once:
  Serial.begin(9600);
  while ( !Serial ) delay ( 500 );

  FastLED.addLeds<WS2812, LED_PIN, GRB>(leds, NUM_LEDS);
  #ifdef __AVR__
  SeCom.AddCmd( F (“BTN”), SERIALCMD_FROMALL, SerialButton );
  SeCom.AddCmd( F (“HSV”), SERIALCMD_FROMALL, HSVButton );
  SeCom.AddCmd( F (“RB”), SERIALCMD_FROMALL, RainbowButton );
  SeCom.AddCmd( F (“IRB”), SERIALCMD_FROMALL, Rainbow );
  SeCom.AddCmd( F (“HSVALL”), SERIALCMD_FROMALL, HSVAll );
  SeCom.AddCmd( F (“OPEN”), SERIALCMD_FROMALL, OpenLock );
  SeCom.AddCmd( F (“CLOSE”), SERIALCMD_FROMALL, CloseLock );
  SeCom.Print ( F ( “INFO: Program running on AVR ... \r\n” ) );
  #else
  SeCom.AddCmd( “BTN”, SERIALCMD_FROMALL, SerialButton );
  SeCom.Print ( ( char * ) “INFO: Program running ... \r\n” );
  #endif

  if (!cap[0].begin(0x5A)){
    #ifdef DEBUG
    Serial.println(“MPR121 not found, check wiring?”);
    #endif
    while (1);
  }
  if (!cap[1].begin(0x5C)){
    #ifdef DEBUG
    Serial.println(“Second MPR121 not found, check wiring?”);
    #endif
    while (1);
  }

///Sets Thresholds - first for touch, second release - the value can be in range 0x05~0x30 cap[1] is set
very easy to release so 11 works, cap[1] rather hard, so 2 doesn’t fire accidentally
  cap[1].setThresholds(0x05, 0x00);
  cap[0].setThresholds(0x15, 0x15);

  pinMode(LOCK_PIN, OUTPUT);

  LedFun();
  LedOff();
}

void loop() {
  // put your main code here, to run repeatedly:
  int8_t ret;

   ret = SeCom.ReadSer();
   #ifdef DEBUG
   if ( ret == 0 )

      SeCom.Print ( ( char * ) “ERROR: Urecognized command. \r\n” );
   #endif

   currtouch = cap[0].touched();

   if((currtouch & (1 << 0)) && !(lasttouch & (1 << 0)))
   {
    FadeButton(Buttons[6],colors(WHITE),0);
    SeCom.Print ( ( char * ) “6\r\n” );
   }
   if((currtouch & (1 << 1)) && !(lasttouch & (1 << 1)))
   {
    FadeButton(Buttons[4],colors(WHITE),0);
    SeCom.Print ( ( char * ) “4\r\n” );
   }
   if((currtouch & (1 << 2)) && !(lasttouch & (1 << 2)))
   {
    FadeButton(Buttons[1],colors(WHITE),0);
    SeCom.Print ( ( char * ) “1\r\n” );
   }
   if((currtouch & (1 << 3)) && !(lasttouch & (1 << 3)))
   {
    FadeButton(Buttons[3],colors(WHITE),0);
    SeCom.Print ( ( char * ) “3\r\n” );
   }
   if((currtouch & (1 << 4)) && !(lasttouch & (1 << 4)))
   {
    FadeButton(Buttons[2],colors(WHITE),0);
    SeCom.Print ( ( char * ) “2\r\n” );
   }
   if((currtouch & (1 << 5)) && !(lasttouch & (1 << 5)))
   {
    FadeButton(Buttons[0],colors(WHITE),0);
    SeCom.Print ( ( char * ) “0\r\n” );
   }

   currtouchTwo = cap[1].touched();
```

```
//   if((currtouchTwo & (1 << 0)) && !(lasttouchTwo & (1 << 0)))
//   {
//    FadeButton(Buttons[11],colors(WHITE),0);
//    SeCom.Print ( ( char * ) "11\r\n" );
//   }
   if((currtouchTwo & (1 << 1)) && !(lasttouchTwo & (1 << 1)))
   {
    FadeButton(Buttons[9],colors(WHITE),0);
    SeCom.Print ( ( char * ) "9\r\n" );
   }
   if((currtouchTwo & (1 << 2)) && !(lasttouchTwo & (1 << 2)))
   {
    FadeButton(Buttons[8],colors(WHITE),0);
    SeCom.Print ( ( char * ) "8\r\n" );
   }
   if((currtouchTwo & (1 << 3)) && !(lasttouchTwo & (1 << 3)))
   {
    FadeButton(Buttons[5],colors(WHITE),0);
    SeCom.Print ( ( char * ) "5\r\n" );
   }
    // FOLLOWING ONE USED TO BE 10, but now it is broken - the original 7 turned into 10 and 7 is on
(0<<6) now...
//   if(!(currtouchTwo & (1 << 4)) && (lasttouchTwo & (1 << 4)))
//   {
//    FadeButton(Buttons[10],colors(WHITE),0);
//    SeCom.Print ( ( char * ) "10\r\n" );
//   }
   if((currtouchTwo & (1 << 5)) && !(lasttouchTwo & (1 << 5)))
   {
    FadeButton(Buttons[10],colors(WHITE),0);
    SeCom.Print ( ( char * ) "10\r\n" );
   }
   if((currtouchTwo & (1 << 6)) && !(lasttouchTwo & (1 << 6)))
   {
    FadeButton(Buttons[7],colors(WHITE),0);
    SeCom.Print ( ( char * ) "7\r\n" );
   }


   lasttouch = currtouch;
   lasttouchTwo = currtouchTwo;

   if(isFading)
   {
    DoFade();
   }
}

void OpenLock() {
  digitalWrite(LOCK_PIN, HIGH);
}
void CloseLock() {
  digitalWrite(LOCK_PIN, LOW);
}

void LedOn(){
  for(int i = 0; i < NUM_LEDS; i++)
  {
    leds[i] = CRGB(125,0,0);
  }
  #ifdef DEBUG
    Serial.println("LedOn()");
  #endif
  FastLED.show();
  ledOn = true;
}

void LedOff(){
  for(int i = 0; i < NUM_LEDS; i++)
  {
    leds[i] = CRGB(0,0,0);
  }
    //Serial.println("LedOff()");
  FastLED.show();
  ledOn = false;
}

void LedGreen(){
  for(int i = 0; i < NUM_LEDS; i++)
  {
    leds[i] = CRGB(0,255,0);
  }
  #ifdef DEBUG
    Serial.println("LedGreen()");
  #endif
  FastLED.show();
  ledOn = true;
```

```
}

void LedRed(){
  for(int i = 0; i < NUM_LEDS; i++)
  {
    leds[i] = CRGB(255,0,0);
  }
  #ifdef DEBUG
    Serial.println("LedRed()");
  #endif
  FastLED.show();
  ledOn = true;
}
void LedFun(){
  leds[0] = CRGB(255,0,0);
    for(int i = 1; i < NUM_LEDS; i++)
  {
    leds[i-1] = CRGB(0,0,0);
    leds[i] = CRGB(255,0,0);
    FastLED.show();
    delay(20);
  }

    leds[NUM_LEDS] = CRGB(0,0,255);
    for(int i = NUM_LEDS-1; i >= 0; i--)
  {
    leds[i+1] = CRGB(0,0,0);
    leds[i] = CRGB(0,0,255);
    FastLED.show();
    delay(20);
  }
  LedGreen();
    #ifdef DEBUG
    Serial.println("LedRed()");
    #endif
  ledOn = true;
}

void SerialButton() {
  #ifdef DEBUG
  Serial.println("SerialButton called");
  #endif
  char * btn;
  char * clr;
  btn = SeCom.ReadNext();
  clr = SeCom.ReadNext();

  short btnnr = (short) strtoul ( btn, NULL, 0 );
  short clrnr = (short) strtoul ( clr, NULL, 0 );

  if(btn == NULL)
  {
    return;
  }
  if(clr == NULL)
  {
    return;
  }
  #ifdef DEBUG
  //Serial.println("Buttonnr: " + btnnr-1);
  SeCom.Print ( (char *) "Buttonnr = " );
  SeCom.Print ( (short)btnnr );
  SeCom.Print ( (char *) "\r\n" );
  #endif
  HalfFadeButton(Buttons[btnnr-1], clrnr,0);

}

void HSVButton() {
  char * btn;
  char * hParam;
  char * sParam;
  char * vParam;
  btn = SeCom.ReadNext();
  hParam = SeCom.ReadNext();
  sParam = SeCom.ReadNext();
  vParam = SeCom.ReadNext();

  short btnnr = (short) strtoul ( btn, NULL, 0 );
  short h = (short) strtoul ( hParam, NULL, 0 );
  short s = (short) strtoul ( sParam, NULL, 0 );
  short v = (short) strtoul ( vParam, NULL, 0 );

  button butt = Buttons[btnnr -1];



  if(hParam == NULL || sParam == NULL || vParam == NULL)
  {
    SeCom.Print ( (char *) "Wrong Number of Parameters! \r\n" );
```

```
    return;
  }
  //Serial.println("Buttonnr: " + btnnr-1);
  #ifdef DEBUG
  SeCom.Print ( (char *) "HSV = " );
  SeCom.Print ( (short)h );
  SeCom.Print ( (char *) ", " );
  SeCom.Print ( (short)s );
  SeCom.Print ( (char *) ", " );
  SeCom.Print ( (short)v );
  SeCom.Print ( (char *) "\r\n" );
  #endif

  for(int i=0; i < 5; i++)
  {
    leds[butt.lights[i]-1].setHSV(h,s,v);
  }
    FastLED.show();


}

void HSVAll() {
  char * hParam;
  char * sParam;
  char * vParam;
  hParam = SeCom.ReadNext();
  sParam = SeCom.ReadNext();
  vParam = SeCom.ReadNext();

  short h = (short) strtoul ( hParam, NULL, 0 );
  short s = (short) strtoul ( sParam, NULL, 0 );
  short v = (short) strtoul ( vParam, NULL, 0 );

  if(hParam == NULL || sParam == NULL || vParam == NULL)
  {
    #ifdef DEBUG
    SeCom.Print ( (char *) "Wrong Number of Parameters! \r\n" );
    #endif
    return;
  }

  for(int i=0; i < NUM_LEDS; i++)
  {
    leds[i].setHSV(h,s,v);
  }
    FastLED.show();


}


void LightButton(button butt, char col){
  for(int i=0; i < 5; i++)
  {
    leds[butt.lights[i]-1] = colorValues[col];
  }
    FastLED.show();
 #ifdef DEBUG
  for(int i = 0; i<5; i++)
  {
    SeCom.Print ( (uint8_t) butt.lights[i] );
    SeCom.Print ( (char *) "\r\n" );
  }
  #endif
  //Serial.println(col);
}

void DoFade()
{
  if(color.v <= 251 && fadeUp)
  {
    for(int i=0; i < 5; i++)
    {
      leds[currentButton.lights[i]-1] = color;
    }
      FastLED.show();
      color.v = color.v + 4;
      delay(currentSpeed);
      return;
  }
  if(isHalfFade)
  {
    isFading = false;
    isHalfFade = false;
    currentButton.stayOn = true;
    return;
  }
    fadeUp = false;
```

```
  if(color.v >= 5 && !fadeUp)
  {
    for(int i=0; i < 5; i++)
    {
      leds[currentButton.lights[i]-1] = color;
    }
      FastLED.show();
      color.v = color.v - 4;
      delay(currentSpeed);
      return;
  }

  color.v = 0;
  for(int i=0; i < 5; i++)
    {
      leds[currentButton.lights[i]-1] = color;
    }
      FastLED.show();

  isFading = false;
  currentButton.stayOn = false;
}

void FadeButton (button butt, char col, short spd){
  if(!currentButton.stayOn)
  {
    color.v = 0;
    for(int i=0; i < 5; i++)
    {
      leds[currentButton.lights[i]-1] = color;
    }
    FastLED.show();
  }
    currentButton = butt;
    currentColor = col;
    currentSpeed = spd;
    color = colorValues[currentColor];
    isFading = true;
    fadeUp = true;
    color.v = 20;
}
void HalfFadeButton(button butt, char col, short spd)
{
    FadeButton(butt, col, spd);
    isHalfFade = true;
}

void RainbowButton(){
  char * btn;
  btn = SeCom.ReadNext();
  short btnnr = (short) strtoul ( btn, NULL, 0 );
  button butt = Buttons[btnnr -1];

  for(int hue = 0; hue < 256; hue++)
  {
    for(int i=0; i < 5; i++)
    {
    leds[butt.lights[i]-1].setHSV(hue,255,255);
    }
    FastLED.show();
    delay(10);
  }
  LedOff();
}
void Rainbow(){
  char * spd;
  spd = SeCom.ReadNext();
  int spdnr = (int) strtoul ( spd, NULL, 0 );
  int hue = 0;
   while(true)
    {
      for(int i=0; i < 72; i++)
      {
      leds[i].setHSV(hue,255,255);
      }
      FastLED.show();
      if(spd != NULL)
        delay(spdnr);
      else
        delay(10);

      if(hue == 254)
        hue = 0;
      else
        hue++;
    }

}
```

3) Coderätsel Code

```
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using Oculus.Interaction;
using Photon.Voice.Unity.Demos;
using TMPro;
using UnityEngine;

namespace XcapeRoom
{

  public class Code
  {

    public Code(CodeBlock top, CodeBlock side, CodeBlock bottom, CodeBlock small)
    {
      CodeBlocks[(int)CodeSection.Top] = top;
      CodeBlocks[(int)CodeSection.Side] = side;
      //CodeBlocks[(int)CodeSection.Middle] = middle;
      CodeBlocks[(int)CodeSection.Bottom] = bottom;
      CodeBlocks[(int)CodeSection.Small] = small;
    }
    public CodeBlock[] CodeBlocks = new CodeBlock[5];
  }
  public class CodePuzzleHandler : APuzzleHandler
  {
    public TextMeshPro OutputText;
    public GameObject ErrorNotification;
    public GameObject CorrectNotification;
    public Renderer[] PlaceholderRenderers;
    Color green = new Color(0f, 1f, 0, 0.5f);
    public Code AssembledCode {get; private set;} = new Code(CodeBlock.None, CodeBlock.None, CodeBlock.
None, CodeBlock.None);
   public Code CorrectCode {get; private set;} = new Code(CodeBlock.Prepare, CodeBlock.Human, CodeBlock.
Create, CodeBlock.NotEquals);

    public override void RegisterCodeBlock(CodeBlock codeBlock, CodeSection codeSection)
    {
      AssembledCode.CodeBlocks[(int)codeSection] = codeBlock;
      Debug.Log("Current CodeBlocks: " + AssembledCode.CodeBlocks.ToString());
      foreach(var block in AssembledCode.CodeBlocks)
      {
        Debug.Log(block.ToString());
      }
      if(AssembledCode.CodeBlocks.SequenceEqual(CorrectCode.CodeBlocks))
      {
        GameMaster.Instance.SetCodePuzzleSolved();
        ErrorNotification.SetActive(false);
        CorrectNotification.SetActive(true);
        foreach(Renderer renderer in PlaceholderRenderers)
        {
          renderer.material.SetColor("_Color", green);
        }
        if(GameMaster.Instance.State.SwabsInserted)
          OutputText.text = "Code correct - generating Avatars";
        else
          OutputText.text = "Code correct - awaiting samples";

      }
      else
      {
        CorrectNotification.SetActive(false);
        ErrorNotification.SetActive(true);
      }
    }
  }
}
```

4) OSC Code

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using extOSC;

namespace XcapeRoom{
  public class OscPosSender : MonoBehaviour
  {
    public string AddressX = "/x";
    public string AddressY = "/y";
    public float OffsetX;
    public float OffsetY;
    private Coroutine sendThirtyFPS;
```

```
    [Header("OSC Settings")]
    public OSCTransmitter Transmitter;
    // Start is called before the first frame update
    void Start()
    {

    }
    void OnEnable()
    {
      sendThirtyFPS = StartCoroutine(SendThirtyFPS());
    }
    void OnDisable()
    {
      StopCoroutine(sendThirtyFPS);
      sendThirtyFPS = null;
    }

    // Update is called once per frame
    void Update()
    {
      //SendPosition();
    }

    private IEnumerator SendThirtyFPS()
    {
      while (true)
      {
        SendPosition();
        yield return new WaitForSecondsRealtime(0.03f);
      }
    }

    private void SendPosition()
    {
      if(this.transform.hasChanged){
        var messageX = new OSCMessage(AddressX);
        messageX.AddValue(OSCValue.Float(transform.position.x + OffsetX));
        var messageY = new OSCMessage(AddressY);
        messageY.AddValue(OSCValue.Float(transform.position.z + OffsetY));
        Transmitter.Send(messageX);
        Transmitter.Send(messageY);
        //Debug.Log(„Message sent“);
        transform.hasChanged = false;
      }
    }
  }

}
```

5) Processing Block zur Pointcloud Bereinigung

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Runtime;
using System.Runtime.InteropServices;
using System.Runtime.Remoting;
using Intel.RealSense;
using Intel.RealSense.Extensions;
using Intel.RealSense.Math;
using UnityEngine;

[ProcessingBlockData(typeof(RemoveCoordinates))]
public class RemoveCoordinates : RsProcessingBlock
{
  public ushort Decimation;

  ushort[] depthData;
  public Bounds[] Bounds;
  [Space (30)]
  public Rotator[] RotOffset;

  // [Range(1, 5)]
  // public int GuiFiller;

  bool isInit = false;
  private Intrinsics intrinsics;
  private int[] dimensions;

  void OnDisable()
  {
    depthData = null;
    isInit = false;
  }
```

```
  private void Init()
  {
      float[] myCoeffs = {0.0f,0.0f,0.0f,0.0f,0.0f};
     intrinsics = new Intrinsics() {width = 640, height = 480, ppx = 286.67578125f, ppy = 249.482421875f,
fx = 465.76953125f, fy = 465.16015625f, model = 0, coeffs = myCoeffs};
      Debug.Log(intrinsics);
      isInit = true;
  }

  Frame ApplyFilter(DepthFrame depth, FrameSource frameSource)
  {
    using (var p = depth.Profile)
    {
      var count = depth.Width * depth.Height;
      if (depthData == null || depthData.Length != count)
        depthData = new ushort[count];

      depth.CopyTo(depthData);

      RemoveWithinBounds(depthData);

      // for (int i = 0; i < count; i++)
      // {
      //     if (depthData[i] > Distance)
      //         depthData[i] = 0;
      // }

     var v = frameSource.AllocateVideoFrame<DepthFrame>(p, depth, depth.BitsPerPixel, depth.Width, depth.
Height, depth.Stride, Extension.DepthFrame);
      v.CopyFrom(depthData);

      return v;
    }
  }

  public override Frame Process(Frame frame, FrameSource frameSource)
  {
    if(!isInit)
      Init();

    if (frame.IsComposite)
    {
      using (var fs = FrameSet.FromFrame(frame))
      using (var depth = fs.DepthFrame)
      {
        var v = ApplyFilter(depth, frameSource);
        // return v;

        // find and remove the original depth frame
        var frames = new List<Frame>();
        foreach (var f in fs)
        {
          using (var p1 = f.Profile)
            if (p1.Stream == Stream.Depth && p1.Format == Format.Z16)
            {
              f.Dispose();
              continue;
            }
          frames.Add(f);
        }
        frames.Add(v);

        var res = frameSource.AllocateCompositeFrame(frames);
        frames.ForEach(f => f.Dispose());
        using (res)
          return res.AsFrame();
      }
    }

    if (frame is DepthFrame)
      return ApplyFilter(frame as DepthFrame, frameSource);

    return frame;
  }
  private ushort[] RemoveWithinBounds(ushort[] depthData)
  {
    UnityEngine.Quaternion[] rotation = new UnityEngine.Quaternion[RotOffset.Length];
    for(int j = 0; j < RotOffset.Length; j++)
    {
      rotation[j] = UnityEngine.Quaternion.Euler(RotOffset[j].x, RotOffset[j].y, RotOffset[j].z);

    }
    Vector2D pixel = new Vector2D
        {
          X = 0,
          Y = 0
        };
        Vector3 vertex;
```

```
      for(int i = 0; i < depthData.Length; i++)
      {
        var depth = Convert.ToSingle(depthData[i]);
        vertex = CoordinateMapper.Map2DTo3D(intrinsics, pixel, depth/4000f, Decimation);
        for(int j = 0; j < Bounds.Length; j++)
        {
          if(Bounds[j].Contains(rotation[j] * vertex))
          {
            depthData[i] = 0;
          }
        }
        if(pixel.X < intrinsics.width/Decimation -1 && pixel.Y < intrinsics.height/Decimation)
          pixel.X += 1f;
        else if(pixel.X >= intrinsics.width/Decimation -1 && pixel.Y < intrinsics.height/Decimation)
        {
          pixel.X = 0f;
          pixel.Y += 1f;
        }

      }
      return depthData;
  }
  [System.Serializable]
  public struct Rotator
  {
    public float x;
    public float y;
    public float z;
  }
}
```

6) Pointcloud Code

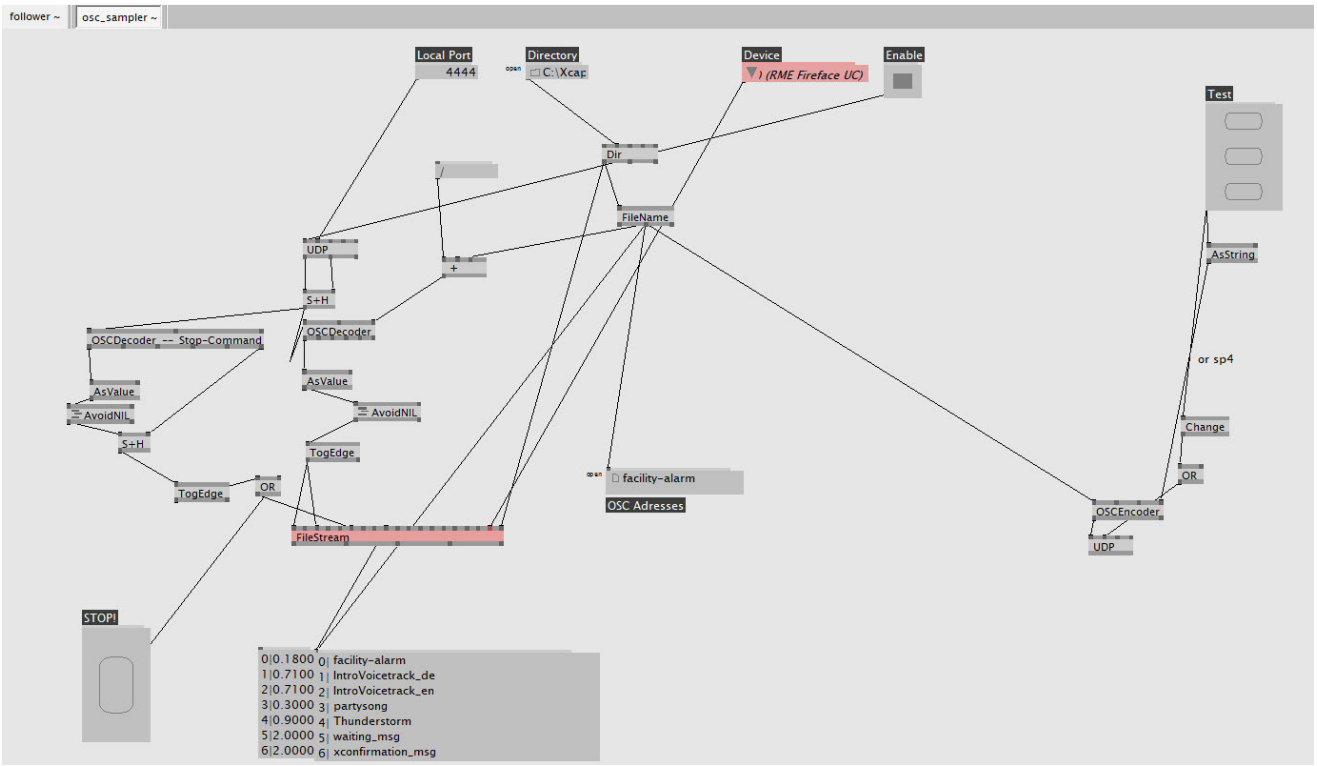
```
using System;
using UnityEngine;
using Intel.RealSense;
using UnityEngine.Rendering;
using UnityEngine.Assertions;
using System.Runtime.InteropServices;
using System.Threading;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using UnityEditor;
using Intel.RealSense.Math;

namespace XcapeRoom {

  [RequireComponent(typeof(MeshFilter), typeof(MeshRenderer))]
  public class XrPointCloudRenderer : MonoBehaviour
  {
    public RsFrameProvider Source;
    private Mesh mesh;
    private Texture2D uvmap;

    //private Vector3 bin = new Vector3(0,0,0);

    [NonSerialized]
    public Vector3[] vertices;
    private Vector3[] threadVertices;
    private List<Vector3> vertexBuffer = new List<Vector3>();
    public int[] PointThreshold;
    public double[] TimeWithPoints;
    public double[] TimeWithoutPoints;
    public int[] pointNumbers;
    bool isVolumeThreadFinished = true;
    bool[] isSomethingInVolume;
    public Transform[] volumeBoxes;
    public Bounds OneOneBounds;
    Matrix4x4 worldMatrix;
    Matrix4x4[] volumeMatrixes;

    FrameQueue q;

    void Awake()
    {
      worldMatrix = transform.localToWorldMatrix;
      isSomethingInVolume = new bool[volumeBoxes.Length];
      TimeWithPoints = new double[volumeBoxes.Length];
      TimeWithoutPoints = new double[volumeBoxes.Length];
      volumeMatrixes = new Matrix4x4[volumeBoxes.Length];
    }
    void Start()
    {
```

```
      Source.OnStart += OnStartStreaming;
      Source.OnStop += Dispose;

    }

    private void OnStartStreaming(PipelineProfile obj)
    {
      q = new FrameQueue(1);

      using (var depth = obj.Streams.FirstOrDefault(s => s.Stream == Stream.Depth && s.Format == Format.
Z16).As<VideoStreamProfile>())
        ResetMesh(depth.Width, depth.Height);

      Source.OnNewSample += OnNewSample;
    }

    private void ResetMesh(int width, int height)
    {
      Assert.IsTrue(SystemInfo.SupportsTextureFormat(TextureFormat.RGFloat));
      uvmap = new Texture2D(width, height, TextureFormat.RGFloat, false, true)
      {
        wrapMode = TextureWrapMode.Clamp,
        filterMode = FilterMode.Point,
      };
      GetComponent<MeshRenderer>().sharedMaterial.SetTexture("_UVMap", uvmap);

      if (mesh != null)
        mesh.Clear();
      else
        mesh = new Mesh()
        {
          indexFormat = IndexFormat.UInt32,
        };

      vertices = new Vector3[width * height];

      var indices = new int[vertices.Length];
      for (int i = 0; i < vertices.Length; i++)
        indices[i] = i;

      mesh.MarkDynamic();
      mesh.vertices = vertices;

      var uvs = new Vector2[width * height];
      Array.Clear(uvs, 0, uvs.Length);
      for (int j = 0; j < height; j++)
      {
        for (int i = 0; i < width; i++)
        {
          uvs[i + j * width].x = i / (float)width;
          uvs[i + j * width].y = j / (float)height;
        }
      }

      mesh.uv = uvs;

      mesh.SetIndices(indices, MeshTopology.Points, 0, false);
      mesh.bounds = new Bounds(Vector3.zero, Vector3.one * 10f);

      GetComponent<MeshFilter>().sharedMesh = mesh;
    }

    void OnDestroy()
    {
      if (q != null)
      {
        q.Dispose();
        q = null;
      }

      if (mesh != null)
        Destroy(null);
    }

    private void Dispose()
    {
      Source.OnNewSample -= OnNewSample;

      if (q != null)
      {
        q.Dispose();
        q = null;
      }
    }

    private void OnNewSample(Frame frame)
    {
      if (q == null)
```

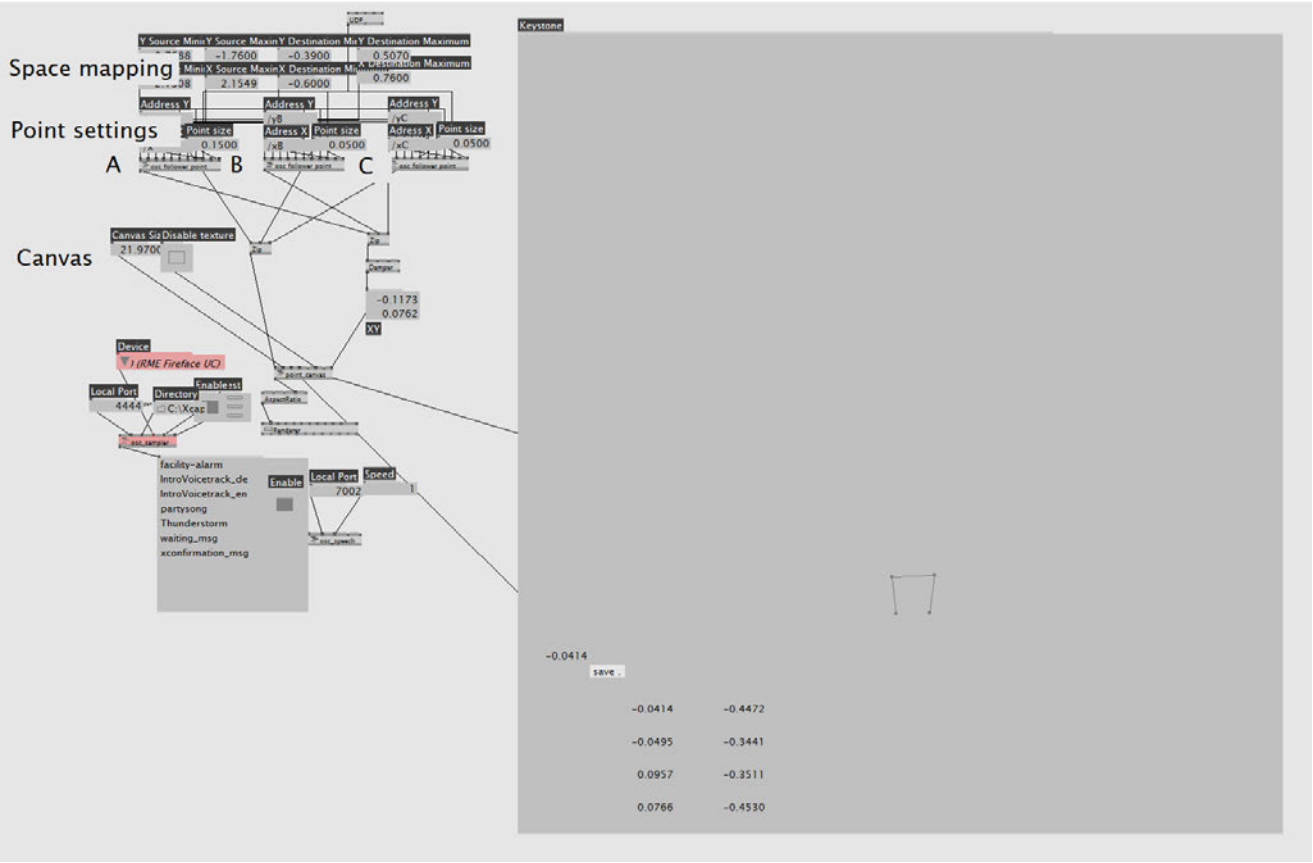
```
    return;
  try
  {
    if (frame.IsComposite)
    {
      using (var fs = frame.As<FrameSet>())
      using (var points = fs.FirstOrDefault<Points>(Stream.Depth, Format.Xyz32f))
      {
        if (points != null)
        {
          q.Enqueue(points);
        }
      }
      return;
    }

    if (frame.Is(Extension.Points))
    {
      q.Enqueue(frame);
    }
  }
  catch (Exception e)
  {
    Debug.LogException(e);
  }
}

protected async void LateUpdate()
{
  if (q != null)
  {
    Points points;
    if (q.PollForFrame<Points>(out points))
      using (points)
      {
        if (points.Count != mesh.vertexCount)
        {
          using (var p = points.GetProfile<VideoStreamProfile>())
            ResetMesh(p.Width, p.Height);
        }

        if (points.TextureData != IntPtr.Zero)
        {
          uvmap.LoadRawTextureData(points.TextureData, points.Count * sizeof(float) * 2);
          uvmap.Apply();
        }

        if (points.VertexData != IntPtr.Zero)
        {
          points.CopyVertices(vertices);
          threadVertices = vertices;
          if(isVolumeThreadFinished)
          {
            for(int i = 0; i < volumeBoxes.Length; i++)
            {
              volumeMatrixes[i] = volumeBoxes[i].worldToLocalMatrix;
            }
            Thread checkVolume = new Thread(CheckPoints);
            isVolumeThreadFinished = false;
            checkVolume.Start();

            for(int i = 0; i < isSomethingInVolume.Length; i++)
            {
              if(isSomethingInVolume[i])
              {
                TimeWithPoints[i]++;
                TimeWithoutPoints[i] = 0;
              }
              else
              {
                TimeWithPoints[i] = 0;
                TimeWithoutPoints[i]++;
              }

            }
          }

          mesh.vertices = vertices;
          mesh.UploadMeshData(false);
        }
      }
  }
}

async void CheckPoints(){
  isSomethingInVolume = new bool[volumeBoxes.Length];
  pointNumbers = new int[volumeBoxes.Length];
  foreach(var vert in threadVertices)
```

```
        {
          if(vert != Vector3.zero)
          {
            var transformedPoint = worldMatrix.MultiplyPoint(vert);
            for(int i = 0; i < volumeMatrixes.Length; i++)
            {
              var volumePoint = volumeMatrixes[i].MultiplyPoint(transformedPoint);
              if(OneOneBounds.Contains(volumePoint))
              {
                pointNumbers[i]++;
              }
            }
          }
        }
        for(int i = 0; i < isSomethingInVolume.Length; i++)
        {
          if(pointNumbers[i] > PointThreshold[i])
              {
                isSomethingInVolume[i] = true;
              }
        }
        isVolumeThreadFinished = true;
        }
    }
}
```

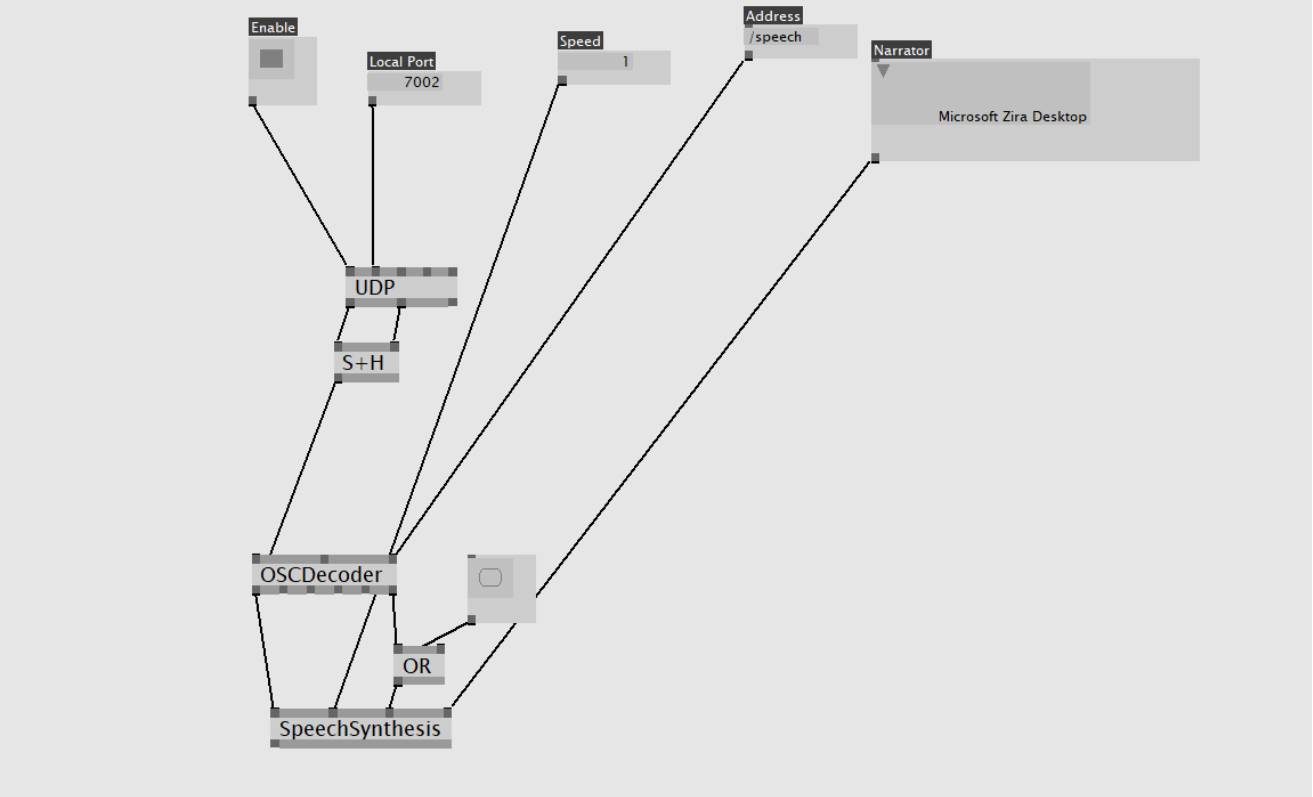
7) VVVV Sketch zur Steuerung von Lichtpunkt und Audiosamples



Ansteuerung der Audiosamples über OSC Messages



Hauptsketch inkl. Ansteuerung der Lichtpunktprojektion



Sprachsynthese über OSC Messages

8) Eidestattliche Erklärung

Eigenständigkeitserklärung

Hiermit versichere ich, dass ich die vorstehende Masterthesis mit dem Titel

XcapeRoom: Cross Reality Interactions in a hybrid Escape-Room Environment

selbstständig ohne fremde Hilfe gefertigt und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quelle kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Hamburg den 12.09.2022