UNIVERSITÄT KOBLENZ · LANDAU
Fachbereich 4: Informatik

Entwicklung einer immersiven VR-Erfahrung

Bachelorarbeit

Zur Erlangung des Grades Bachelor of Science (B.Sc.)
im Studiengang Informatik

vorgelegt von

Maximilian Walter Hübel

Erstgutachter:	Prof. Dr. Stefan Müller (Institut für Computervisualistik, AG Computergraphik)
Zweitgutachter:	Nils Höhner, M.Sc. (Institut für Wissensmedien, IWM)

Koblenz, im September 2019

Erklärung

Ich versichere, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

	Ja	Nein
Mit der Einstellung dieser Arbeit in die Bibliothek bin ich einverstanden.	☑	☐

Koblenz, 25.09.2019 M. Hübel

. . . . . .

(Ort, Datum) (Unterschrift)

Abstract

This thesis is about the design and the implementation of a virtual reality experience. The goal is to answer two questions: Is it possible to create an immersive virtual reality experience which is mainly using impulses and triggers to scare and frighten users? Secondly, is this immersion strong enough to create an illusion in which the user can't separate the real world from the virtual world? To realise this project the design program Unity3D as well as Visual Studios 2017 were used. Furthermore, in order to verify that the experience is indeed immersive for the user, an experiment with a sample size of seven people was created. Afterwards the candidates were interviewed via a questionnaire how they felt during the virtual reality application.
As a result the study showed that the application has tendencies to be immersive but the users were still aware of the situation. It can be concluded that the immersion was not strong enough to fool users regarding the separation of the virtual and real world.

Zusammenfassung

Diese Bachelorarbeit beschäftigt sich mit dem Entwurf und der Implementation einer virtuellen Realitätserfahrung. Ziel ist es, zwei Fragen zu beantworten: Ist es möglich, eine immersive virtuelle Anwendung zu erschaffen, die hauptsächlich Impulse und Trigger benutzt, um Angst und Schrecken bei den Benutzern zu erzeugen? Zweitens, ist diese Immersion ausreichend, die Benutzer so zu illusionieren, dass sie die virtuelle Welt für die Reale halten. Zur Erschaffung dieser Erfahrung wurde die Programmierumgebung Unity3D sowie Visual Studios 2017 verwendet. Um festzustellen, ob diese VR-Anwendung tatsächlich immersiv für den Anwender ist, wurde ein Experiment mit sieben Probanden durchgeführt. Nach der Spieltestung wurden die Probanden zu Ihren Erfahrungen mittels eines Fragebogens befragt.
Es konnte dadurch gezeigt werden, dass diese Anwendungen Tendenzen zur Immersion aufweisen. Jedoch waren sich die Benutzer der Situation, in der sie sich befanden, stets bewusst. Daraus lässt sich schlussfolgern, dass die Immersion nicht stark genug war, um die Probanden bezüglich der virtuellen und realen Welt zu täuschen.

Inhaltsverzeichnis

1 Einleitung

Computerspiele sind eine beliebte Freizeitaktivität und in unserer heutigen Zeit nicht mehr wegzudenken. Die steigende Nachfrage nach neuen Spielen auf dem Markt ist deutlich zu spüren. Dieser Zustand verdeutlicht auch eine repräsentative Statistik von Statista (Abb.1). Daraus lässt sich erkennen, dass im vergangen Jahr ein Umsatz von knapp 4,37 Milliarden Euro auf den deutschen Märkten erwirtschaftet werden konnte. Im Jahr 2017 lag der Umsatz noch bei 4 Milliarden Euro. Dies ist eine Steigerung von knapp neun Prozent im Vergleich zum Vorjahr.

Abb.1: Statista (Brandt, 2019)

In den letzten 70 Jahren hat es eine enorme Entwicklung von Computerspielen gegeben. Begonnen mit einem Tic-Tac-Toe Spiel namens OXO in den frühen 50er Jahren ist die Entwicklung stetig vorangeschritten. Mit den rapiden technologischen Fortschritten ab den 80er Jahren wurde der Markt mit neuen Spielen überschwemmt.

Einer der neusten Errungenschaften der Computerspielindustrie ist die „Virtual Reality“ (VR). VR ermöglicht es dem Spieler vollständig in das Spiel einzutauchen und quasi ein Teil davon zu werden. Große Unternehmen wie Facebook oder Microsoft, haben das Potenzial von VR schnell erkannt und investieren seither Milliarden in die Weiterentwicklung von VR Hardwaresystemen (Solomon, 2014). Entwicklungsumgebungen, wie beispielsweise Unity3D, oder Innovationen, wie 360° Videokameras, sind Teil dieser Entwicklung und helfen bei dem Versuch das Maximum aus einem Computerspiel rauszuholen.

Ein möglicher Grund für die steigende Beliebtheit von Computer- und VR-Spielen ist die geschaffene Illusion dieser Anwendung. Dabei ist es irrelevant wie verschieden manche Spiele sein mögen oder wie sehr sie sich in ihrer Spielart unterscheiden, sie schaffen es dennoch, den Spieler von der richtigen Welt abzulenken. Sie ziehen ihn in einen Bann, bei dem der Spieler teilweise Zeit oder andere Personen um ihn herum ausblendet und vergisst (Jennett et al., 2008).

Diese Bachelorthesis beschäftigt sich damit, eine immersive Virtual Reality Anwendung zu erschaffen, die den Benutzer in ungewohnte und ungewollte Situationen versetzt. Die Anwendung zielt darauf ab, den Spieler mit seinen Ängsten oder allgemein angstauslösenden Reizen zu konfrontieren, um somit spontane Reaktionen oder stark Emotionen zu erzeugen. Dabei soll die Frage erörtert werden, ob eine VR-Anwendung so immersiv gestaltet werden kann, dass die Realität vom Spieler vergessen wird und die virtuelle Welt als neue Realität wahrgenommen wird.

2 Grundlagen

2.1 Virtual Reality

Das Verständnis einer virtuelle Realität in Bezug auf was sie definiert und ausmacht hat sich im Laufe der 90er Jahre bis heute verändert. Die virtuelle Realität (VR) wurde als ein technologisches Medium in den 90er Jahren verstanden, wie beispielsweise das Telefon oder der Fernseher. Dieses Medium wurde durch eine Kollektion von technologischer Hardware beschrieben und definiert (Computer, Datenbrille, Datenhandschuhe etc.). Somit lag der Fokus von VR verstärkt auf diversen Technologien anstatt auf Daten durch Experimente. Viele beliebte Definitionen aus diesen Jahren beinhalten eine Referenz zu einer technologischen Komponente. Im Folgenden werden zwei Beispiele aufgeführt:

„Virtual Reality is electronic simulations of environments experienced via head mounted eye goggles and wired clothing enabling the end user to interact in realistic three-dimensional situations." (Coates, 1992)

„Virtual Reality is an alternate world filled with computer-generated images that respond to human movements. These simulated environments are usually visited with the aid of an expensive data suit which features stereophonic video goggles and fiber-optic data gloves." (Greenbaum, 1992)

Diese Ansätze sind für Firmen, die VR-bezogene Hardware produzieren, äußert hilfreich. Jedoch eignen sich solche Definitionen nicht für wissenschaftliche Diskussionen, Softwareentwicklungen oder Konsumenten, da sie keine Einsicht in den Prozess oder die dahinterliegende Effekte gewähren, die durch VR-Systeme entstehen (Steuer, 1992).

Die heutige Forschung versucht, neben der technologischen Seite, auch die dahinterliegenden Charakteristika von virtueller Realität näher zu beleuchten. Prof. Dr. Müller stellte in seiner Vorlesung Eigenschaften heraus, die es zu erfüllen gilt, wenn man von VR sprechen möchte: Echtzeit, Interaktivität, Immersion und dynamisches Objektverhalten.

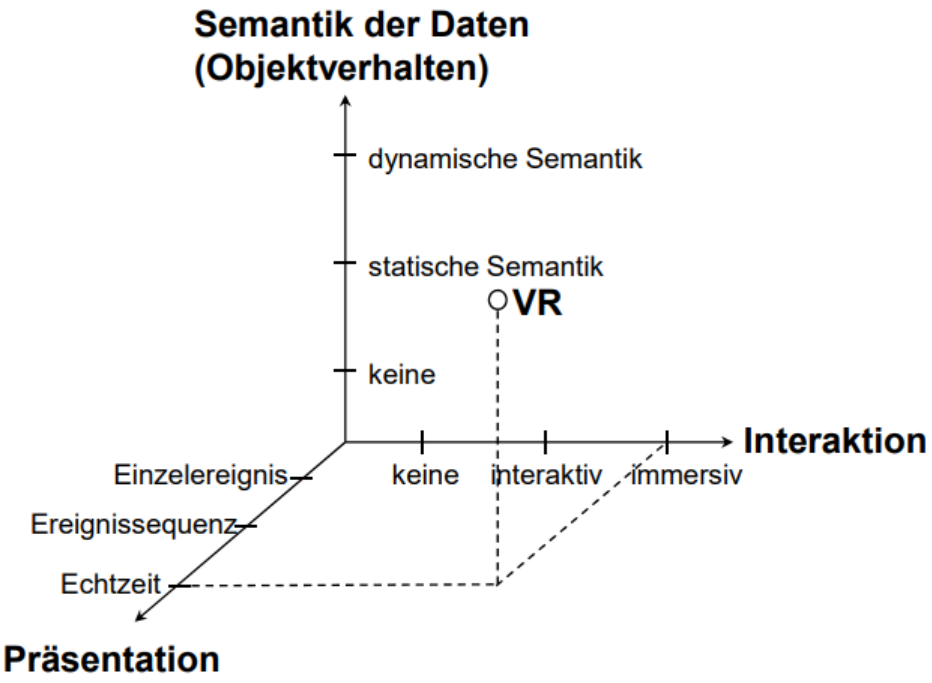


Abb.2: VR-Referenzmodell – Visuelle Darstellung von VR-Eigenschaften (Müller, 2016)

Abb.2 illustriert diese verschiedenen Indikatoren, die das Konstrukt VR beschreiben und messbar machen. Erst wenn sich Objekte dynamisch in der Welt verhalten können, die Interaktionen im Spiel als natürlich und immersiv wahrgenommen werden sowie die Präsentation des Spiels in Echtzeit abläuft, wird von einer virtuellen Realität bzw. einer virtuellen Umgebung gesprochen (Müller, 2016).

2.2 Immersion

Bis heute gibt es noch keine einheitliche Definition für den Begriff „Immersion“ (Brown & Cairns, 2004; Jennett et al, 2008; Sanders & Cairns, 2010; Slater, 2003). Im Allgemeinen steht Immersion in Verbindung mit einer intensiven Spielererfahrung. Wenn ein Spiel die Fähigkeit besitzt, den Spieler in die virtuelle Welt „hinein zu ziehen“, sodass sich der Spieler in dieser nicht reellen Welt verliert, so spricht man von Immersion (Jennett et al., 2008). Die Studie von Brown und Cairns (2004) zeigte, dass Spieler das Wort Immersion sehr ähnlich verstehen und interpretieren. Sie führten Interviews mit sieben Spielern durch, um herauszufinden was Immersion für jeden einzelnen bedeutete. Alle befragten Probanden beschrieben Immersion als Grad der Verbundenheit, die für das Spiel entwickelt wird(Brown & Cairns, 2004).

„A Zen-like state where your hands just seem to know what to do, and your mind just carries on with the story.” (Brown and Cairns, 2004, pp. 1299)

„Everything else is irrelevant, you know it’s there but it’s irrelevant.” (Brown and Cairns, 2004, pp. 1299)

Haywood und Cairns (2005) fanden heraus, dass Immersion nicht nur bei Computerspielen entstehen kann. Die Forschergruppe untersuchte das Verhalten von Kindern in einer interaktiven Museumsausstellung und entdeckten, dass die Schlüsselelemente für eine niedrige Form der Immersion die Beteiligung an einer Geschichte bzw.

einer Erzählung sowie Co-Präsenz (die Präsenz von anderen Menschen) sind. Demnach machen Menschen immersive Erfahrungen, wenn eine einfache progressive Strukturen vorliegt, die es dem Benutzer erlaubt seine eigenen Ideen umzusetzen, um das interaktive System zu verstehen (Haywood & Cairns, 2006).
In der wissenschaftlichen Arbeit von Jennett et al. (2008) wurde festgehalten, dass Immersion folgende Aspekte mit sich bringt:

- Ein Verlust des Zeitgefühls
- Ein fehlendes Verständnis für die reale Welt und für das, was um den Spieler herum passiert
- Eine Verbundenheit oder Involviertheit sowie das Gefühl in der virtuellen Umgebung präsent zu sein

2.2.1 Immersion und Präsenz

Der Begriff „Präsenz“ erfährt seit der Entwicklung der virtuellen Realität in den 90er Jahren ein großes Forschungsinteresse. Das Wortverständnis von Präsenz ist ähnlich wie bei Immersion in Fachkreisen stark diskutiert, sodass eine einheitliche Definition noch nicht gefunden wurde (Jennett et al., 2008). Allgemein wird Präsenz häufig in den Zusammenhang mit Umgebungen von virtuellen Realitäten gebracht. Damit wird ein Konzept beschrieben, dass dem Spieler ein Gefühl gibt, wahrhaftig präsent bzw. anwesend in der virtuellen Welt zu sein (Jennet et al., 2008; Robillard, Bouchard, Fournier, Renaud, 2003; Sanders & Cairns, 2010; Slater, 2003). Es gibt jedoch verschiedene Sichtweisen und Meinungen, in welchem Zusammenhang Präsenz mit Immersion, Emotionen und anderen Faktoren steht. Im Folgenden werden verschiedenen Sichtweisen von Präsenz genauer beleuchtet, um ein besseres Verständnis für dieses Konstrukt zu schaffen.

Mel Slater (2003) versucht dem Begriff Präsenz eine einheitliche Terminologie zu verleihen, um Fachleute die Grundlage zu bieten, über das gleiche Konzept zu debattieren.
Ein immersives System kann bei Menschen unterschiedliche Präsenzstärken auslösen. Genauso können verschiedene immersive Systeme bei unterschiedlichen Menschen denselben Grad an Präsenz erzeugen (Slater, 2003). Slater (2003) bedient sich einer sehr rationale Herangehensweise und beschreibt Präsenz als eine menschliche Reaktion auf die Immersion. Er unterscheidet, bezogen auf Präsenz, zwischen Form und Zufriedenheit. Die Form ist eine Beschreibung für die Möglichkeit, sich wahrhaftig an einem anderen Ort versetzt zu fühlen. Die Zufriedenheit beschreibt hingegen die subjektive Einschätzung des Spielers, ob die Person etwas interessant oder uninteressant findet. Slater (2003) ist der Meinung, dass Präsenz von der Form abhängig ist und das die Zufriedenheit keinerlei Einfluss auf Präsenz hat. Des Weiteren merkt der Forscher an, dass Präsenz und Immersion theoretisch unabhängige Konstrukte sind, jedoch empirisch einen starken Zusammenhang aufweisen.

Des Weiteren diskutiert Slater (2003) den Zusammenhang zwischen Präsenz und Involviertheit sowie Emotionen. Der Argumentation des Forschers folgend sollten Involviertheit und Präsenz als getrennte Konzepte betrachtet werden, da sie auf einem anderen logischen Level operieren. Es ist möglich, präsent zu sein aber nicht involviert

(z.B. Situationen im alltäglichen Leben). Auch Emotionen stehen orthogonal zur Präsenz. Präsenz kann auch in Abwesenheit einer emotionalen Verbindung entstehen. Eine Verschiebung des emotionalen Zustandes würde nichts an der wahrgenommenen Präsenz verändern. Demnach postuliert Slater (2003) die Meinung, dass Präsenz und emotionale Antworten logischerweise getrennt voneinander entstehen und messbar sein müssten.

Die Forschergruppe rund um Robillard et al. (2003) führte eine Studie mit phobischen Probanden sowie mit Probanden ohne jegliche Phobie durch. Die Phobiker in dieser Studie litten unter spezifischen Ängsten, welche sich entweder vor Spinnen, Höhen oder engen Räumen äußerten. Jeder phobische Proband wurde mit seiner speziellen Angst konfrontiert. Dabei wurden die Reaktionen der Phobiker mit den Reaktionen von Nicht Phobikern verglichen. Ziel dieser Studie war es zum einen herauszufinden, ob es möglich ist, trotz preiswertem Equipment Angstzustände bei den phobischen Patienten zu erzeugen. Das zweite Ziel war es Parameter wie Simulationsübelkeit und Präsenz zu erfassen und zu beurteilen. Es sollte herausgestellt werden, welchen Einfluss diese Parameter auf die phobische Effizienz der virtuellen Exposition hat. Das Ergebnis der Studie zeigte, dass die Ängste, die die Probanden äußerten, in Zusammenhang mit dem Gefühl der Präsenz standen. Phobiker zeigten vor der Immersion eine größere Tendenz, Angst zu verspüren als auch eine größere Neigung, Präsenz in der virtuellen Welt zu erfahren. Nach der virtuellen Simulation gaben die phobischen Probanden an, einen höheren Grad an Angst als auch einen höheren Grad an Präsenz verspürt zu haben. Diese Befunde legen nahe, dass Angstzustände in einem engen Zusammenhang mit Präsenz stehen.

Darüber hinaus konnte eine positiven Korrelation zwischen Ängsten und Präsenz nachgewiesen werden (Robillard et al., 2003). Dies leitet zu dem Schluss, dass es eine synergetische Verbindung zwischen Präsenz und Emotionen geben muss. Die Arbeit von Robillard et al. (2003) stimmt mit Slater (2003) über die Trennung von Emotionen und Präsenz konzeptionell überein, jedoch weisen die Resultate der Studie darauf hin, dass die Konstrukte eine empirische Verbundenheit aufweisen. Diese Ergebnisse widersprechen der Theorie von Slater (2003), dass Präsenz und Emotionen orthogonal zueinander stehen (Robillard et al., 2003).

2.2.2 Immersion und Flow

Das Flowerleben ist ein weiteres Konzept das Ähnlichkeiten mit Immersion aufweist (Sanders & Cairns, 2010). Es ist, wie die Präsenz, einer der meist verbreiteten Ideen eine einnehmende Spielerfahrung zu beschreiben (Jennet et al. 2008) und soll daher im Kontext der Immersion näher beleuchtet werden.

Flow beschreibt das Erleben der optimalen Passung zwischen den eigenen Fähigkeiten und den gestellten Anforderungen. Dieser Zustand, in dem eine Aktivität oder Aufgabe den Menschen so einnimmt, ist ebenfalls davon geprägt, dass die Welt und Zeit um die Person herum vergessen werden. Dieses Erleben ist demnach ähnlich wie bei der Immersion, wo der Spieler in eine andere Welt eintaucht und er an nichts anderes mehr denkt als an das, was vor ihm ist (Jennett et al., 2008).

Die Forscher Csíkszentmihályi und Nakamura (2005), hatten sich intensiv mit dem Konzept befasst und beschrieben acht Konditionen von Flow:

- Eine herausfordernde Aufgabe, die passend zu jemandes Fähigkeiten ist
- Klare Ziele
- Intensive Konzentration
- Den Verlust der Selbstwahrnehmung
- Gefühl von Selbstkontrolle
- Verlust von Zeitgefühl
- Direktes und sofortiges Feedback
- Selbst belohnend

Es ist offensichtlich, dass einige Überschneidungen mit dem Konzept der Immersion vorliegen (z.B. den Verlust von Zeitgefühl oder eine Aufgabe, die eine Person so stark einnimmt, dass diese nichts anderes mehr wahrnimmt). Jedoch ist Flow eine ganz spezielle Erfahrung. Sie wird als eine optimale Erfahrung und somit als eine extreme Erfahrung angesehen (Jennett et al., 2008; Sanders & Cairns, 2010). Eine Immersion ist nicht immer extrem und kann innerhalb des Spiels variieren.

2.2.3 Immersionsbeeinflussung

Da diese Bachelorarbeit das Ziel hat, eine immersive VR-Erfahrung für den Spieler zu erschaffen, wurde sich im Zuge dessen mit dem Thema befasst, wie man die Immersion beeinflussen kann. Im Folgenden werden zwei Quellen vorgestellt, die sich ebenfalls mit besagten Thema auseinandergesetzt haben.

Immersion wird oftmals intraindividuell unterschiedlich stark wahrgenommen. Je stärker die Immersion ist, desto häufiger verlieren Spieler das Gefühl für die Zeit. Jedoch unterscheiden sich Menschen darin, wie sie Immersion wahrnehmen. Allerdings besteht die Möglichkeiten Immersion, unabhängig von ihrer anfänglichen Stärke, zu beeinflussen. Diese Manipulation kann das Empfinden von Immersion reduzieren oder zu einer Steigerung beitragen (Sanders & Cairns, 2010).

Musik in Videospielen ist beispielsweise ein Faktor, der die Wahrnehmung von Immersion beeinflussen kann. Töne und Musik verleihen Spielen eine spezielle Atmosphäre, die es dem Spieler erlaubt, tiefer in das Spiel einzutauchen und stärkere immersive Erfahrungen zu machen. Experimente haben gezeigt, dass Musik nicht nur Immersion beeinflusst, sondern auch Zeitwahrnehmung verändern kann. In welche Richtung die Manipulation ausfällt ist von den Spielern selbst abhängig. Entscheidend ist dabei, ob die Musik als ansprechend oder nicht empfunden wird (Sanders & Cairns, 2010). Ein weiterer Moderator von Präsenz und Immersion ist die Latenz, mit der ein Spiel abläuft. Dabei bezieht sich die Latenz auf die End-to-End Latenz, also die zeitliche Verzögerung zwischen einer Aktion oder Bewegung des Benutzers und der visuellen Ausführung besagter Aktion oder Bewegung. In der Studie von Meehan, Razzaque, Whitton und Brooks (2003) wurde gezeigt, dass Probanden, die ein Spiel mit niedrigerer Latenz (ca. 50 ms) getestet haben, es zu einem größeren Grad an Präsenz führte als die Durchläufe mit einer höheren Latenz (ca. 90 ms).

In Stresssituationen war die Herzrate bei Spielern mit niedrigeren Latenzen signifikant höher als bei den Durchgängen mit höherer Latenz. Insgesamt lässt das den Schluss zu, dass eine möglichst niedrige End-to-End Latenz zu einer Steigerung der Präsenz und Immersion führt (Meehan, Razzaque, Whitton, & Brooks, 2003).

2.3 Anerkannte Arbeiten

Immersive Spiele und Anwendungen sind Forschungsgebiete, die bereits umfangreich in verschiedenen Bereichen der Wissenschaft beleuchtet wurden. Die vorliegende Arbeit beschäftigt sich vorrangig mit Immersion in Bezug auf eine virtuelle Anwendung, weshalb im Folgenden der aktuelle Forschungsstand in ähnlichen Bereichen aufgearbeitet wird.

Brown und Cairns (2004) untersuchten die Immersion von Spielen und versuchten die Bedeutung von „Immersion“ herauszustellen. Um dieses Vorhaben zu realisieren wurden Interviews mit sieben Spielern durchgeführt, um über deren Spielerfahrungen zu sprechen und ihren Aussagen auf Gemeinsamkeiten sowie Unterschiede zu untersuchen. Daraus ergab sich, dass wie anfänglich von Brown und Cairns vermutet, die Spieler Immersion mit einem Gard der Involviertheit beschrieben haben. Die Forscher fanden heraus, dass Immersion in drei Subgruppen unterteilt werden kann: *„Engagement“*; *„Engrossment“* sowie *„Total Immersion“*.

Engagement

Diese erste Subgruppe bildet die initiale Ebene der Immersion. Sie gilt als die schwächste Form der Immersion und muss vor den zwei anderen Gruppen durchschritten werden. Um eine Ebene zu durchschreiten müssen Barrieren, die in jeder Ebene identifiziert worden sind, durchbrochen werden.

In dieser Subgruppe wurden zwei verschiedenen Barrieren herausgestellt. Die erste Barriere wurde als „Zugang“ betitelt. Darunter wird die Präferenz des Spielers sowie die Spielsteuerung des Spiels selbst gefasst. Die Forscher führten den Punkt an, dass es zu keiner Immersion oder Einnahme eines Spielers kommen kann, wenn der Spieler selbst das Spiel als uninteressant betrachtet. Auch eine zu komplexe Spielsteuerung würde den Spaß und die Lust am Spiel erschweren. Die zweite Barriere ist Zeit sowie Bemühungen bzw. Anstrengungen, die ein Spieler bereit sein muss zu investieren. Sollten diese Barrieren überwunden werden, so entwickelt der Spieler ein zunehmendes Interesse an dem Spiel selbst und es besteht die Möglichkeit, die zweite Ebene der Immersion zu erreichen (Brown & Cairns, 2004).

Engrossment

Auf der zweiten Ebene entwickelt der Spieler Emotionen für das Spiel. Diese sind von entscheidender Bedeutung, wenn ein Spieler die höchste Stufe der Immersion erreichen möchte. Die Barriere, die diesbezüglich identifiziert worden war, ist die Spielgestaltung. Unter diesen Begriff werden die Geschichte (der Plot) des Spiels, die Aufgaben im Spiel selbst sowie die visuelle Ausgabe des Spiels gefasst. Sollten all diese Elemente positiv vom Spieler aufgenommen werden, so entsteht eine Form von Respekt, die der Spieler dem Spiel sowie dem Entwickler entgegenbringt.

Ab diesem Punkt der Immersion hat der Spieler viel Zeit, Bemühungen sowie Aufmerksamkeit für das Spiel geopfert. Daraus ergibt sich ein hoher Gard an emotionalem Investment. Dieses Investment ist der Antrieb vieler Spieler mehr und länger spielen zu wollen bis zu einem Punkt, wo sie emotional erschöpft werden können. Manche Spieler erschaffen sich eine Umgebung, in der sie noch weniger vom Spiel abgelenkt werden können. Dies ermöglicht den Spieler die höchste Stufe der Immersion zu erreichen (Brown & Cairns, 2004).

Total Immersion

Diese Stufe gilt als die höchste Form von Immersion und ist sehr selten zu erreichen. Probanden haben angegeben, dass, wenn sie sich in dieser Stufe befanden sie sich komplett von der Realität getrennt gefühlt haben. Das Spiel war das Einzige, was noch zählte und wurde der Mittelpunkt für den Spieler. Die Forscher gaben an, dass zu diesem Zeitpunkt das Spiel in der Lage war, die Gedanken und Gefühle des Spielers beeinflussen zu könnte. Um in dieser fast schon extremen Form der Immersion zu bleiben, wurden die Barrieren Empathie und Atmosphäre identifiziert (Brown & Cairns, 2004).

Eine weitere Arbeit von Jennett et al. (2008) beschäftigte sich mit der Frage, ob Immersion quantitativ gemessen werden könne. Hierfür wurden drei verschiedene Experimente durchgeführt.

Das erste Experiment untersuchte, inwiefern es den Probanden möglich war, von einer immersiven Aufgabe auf eine nicht-immersive Aufgabe zu wechseln. Dazu wurden zwei Gruppen gebildet. Die erste Gruppe sollte eine Computerspiel namens *„Half Life“* spielen, was ein 3-Deminsionales First Person Shooter Spiel ist. Die zweite Gruppe bekam eine minimalistische Computerinteraktion. Ziel der zweiten Gruppe war es, dass Involvement so gering wie möglich zu halten. Die Forscher, wollten herausfinden, ob die Immersion durch das Computerspiel erzeugt wurde oder durch jegliche Form der Computerinteraktion. Die zweite Aufgabe, die beide Gruppen zu absolvieren hatten, war eine Tangram Aufgabe. Diese Aufgabe wurde gewählt, weil die Probanden so viel wie möglich mit der realen Welt in Kontakt kommen sollten und ein Kontrast zu der Computerwelt geschaffen werden wollte.

Der Ablauf dieses Experiments sah vor, dass die Probanden zuerst die Tangram Aufgabe lösten und danach sich dem Computerspiel bzw. der minimalistischen Computerinteraktion widmeten. Im Anschluss wurden die beiden Gruppen erneut gebeten, die Tangram Aufgabe zu absolvieren. Die Zeiten, die jeder Proband für die Erledigung beider Tangram Aufgaben benötigte, wurden dabei gemessen und verglichen. Es zeigte sich, dass die allgemein empfundene Immersion bei der Gruppe mit dem Computerspiel *Half Life* höher war als die Immersion in Gruppe zwei. Darüber hinaus benötigten fast alle Probanden für die Erledigung der Tangram Aufgabe bei dem zweiten Mal weniger Zeit als sie bei dem ersten Versuch benötigten. Nur ein paar Personen aus Gruppe eins benötigten mehr Zeit für die zweite Ausführung der Tangram Aufgabe. Dies führte zu der Annahme, dass, wenn die Immersion durch ein Computerspiel steigt, die Fähigkeit mit der echten Welt erneut zu agieren und darin zu reagieren sinkt (Jennett et al., 2008).

Das zweite Experiment beschäftigte sich mit den Blickbewegungen der Probanden. Den Testpersonen wurde entweder eine nicht-immersive Aufgabe (einfaches klicken auf Boxen) oder eine immersive Aufgabe (eine virtuelle Umgebung erkunden) gestellt. In beiden Versuchen wurde die Augenbewegungen mittels eines Trackingverfahrens dokumentiert. Hierbei wurde stets der Gard an Immersion, den die Personen nach der Aufgabe angaben, kontrolliert. Auch hier zeigte sich, dass die allgemeine empfundene Immersion bei der immersiven Aufgabe höher war als bei der nicht-immersiven Aufgabe. Jedoch wurde ebenfalls festgestellt, dass die Zahl der Fixierungen von Probanden in der nicht-immersiven Aufgabe über die Zeit anstiegen, während die Fixierungen in der immersiven Aufgabe abnahmen (Jennett et al., 2008).

Es zeigte sich jedoch, dass Testpersonen teilweise die nicht-immersive Aufgabe als hoch immersiv wahrgenommen haben. Es wurde vermutete, dass die Geschwindigkeit der nicht-immersiven Aufgabe selbst Grund für diesen Effekt sein könnte, weshalb diese Möglichkeit im dritten Experiment weiter untersucht wurde. Hierbei untersuchten die Forscher die Reaktionen der Probanden, wenn sie Aufgaben unter zusätzlicher Geschwindigkeit absolvieren mussten. Das Ergebnis der Studie war, das verschiedene Angstzustände bei den Probanden gemessen wurden. Testpersonen, die in einer langsamen Geschwindigkeit operiert haben, zeigten eine signifikant niedrigere Angst, als Testpersonen, die mit einer hohen Geschwindigkeit gearbeitet haben. Den höchsten Angstzustand wurde in der Gruppe mit steigender Geschwindigkeit gemessen. Das Ergebnis dieser Experimente zeigte, dass Immersion durch eine subjektive Einschätzung als auch durch objektive Maße, wie die zeitliche Messung, erhoben werden kann. Des Weiteren wurde festgehalten, dass Immersion nicht ausschließlich als positiv wahrgenommen wird, sondern auch negative Emotionen und unangenehme Zustände als Immersionen vorkommen können (Jennett et al., 2008).

Als letzten Schwerpunkt wird im Folgenden die Forschungsarbeit von Robillart et al. (2003) näher beleuchtet. Dabei wurden die Reaktionen und Erfahrungen von Probanden während eines virtuellen Computerspiels untersucht und gemessen. Die Stichprobe bestand zur Hälfte aus Phobikern und zur anderen Hälfte aus gesunden Probanden.
Es wurde festgestellt, dass virtuelle Realität als therapeutische Maßnahme bei Phobie-Patienten eingesetzt werden kann und wirksam ist. Die Testpersonen mit einer spezifischen Phobie hatten drei virtuelle Therapiesitzung. In der ersten Sitzung wurde sie mit dem VR-Equipment vertraut gemacht und noch keine phobischen Reizen ausgesetzt. In der zweiten sowie dritten Sitzung wurden die Probanden mit dem angstauslösenden Stimulus konfrontiert und ermutigt, auf diesen Reiz zuzugehen. Alle 5 Minuten gaben die Testpersonen einen verbalen Bericht über ihr Stress bzw. Angstlevel ab. Am Ende der Sitzung berichteten die Probanden zusätzlich über den wahrgenommenen Realismus der VR-Anwendung. Die nichtphobischen Testpersonen hatten nur eine Sitzung, in der sie fünf Minuten Zeit bekamen, sich mit dem Equipment vertraut zu machen und anschließend den phobischen Reizen ausgesetzt wurden, die die Phobie Probanden als besonders schlimm empfunden haben.
Das Ergebnis dieser Studie zeigte, dass es möglich ist, die Ängste von phobischen Menschen mittels einer virtuellen Anwendung/Computerspiel zu induzieren und eine als echt empfundene Angstsituation zu erzeugen (Robillard et al., 2003).

2.4 Programmierumgebung Unity

In diesem Abschnitt soll nun die Programmierumgebung Unity näher beleuchtet und erklärt werden. Da Unity eine Vielzahl an Möglichkeiten und Optionen besitzt, wird vor allem auf die verschiedenen Elemente von Unity eingegangen, die für diese Bachelorarbeit entscheidend sind. Unity ist hauptsächlich zur Erschaffung der in dieser Arbeit verwendeten VR-Anwendung verantwortlich.

Unity ist eine Laufzeit- und Entwicklungsumgebung, in der Spiele jeglicher Art erschaffen werden können. Der Editor ist auf Windows, Mac sowie Linux verfügbar und unterstützt 2D, 3D sowie Virtual Reality Spielentwicklungen. Des Weiteren bietet Unity eine Reihe von grafikfreundlichen Tools, die für das Erschaffen von immersiven Spielerlebnissen sowie Spielwelten nützlich sind. Die *Benutzeroberfläche* ist so konzipiert, dass sie schnell und intuitiv verstanden werden kann (Technologies, 2019e).

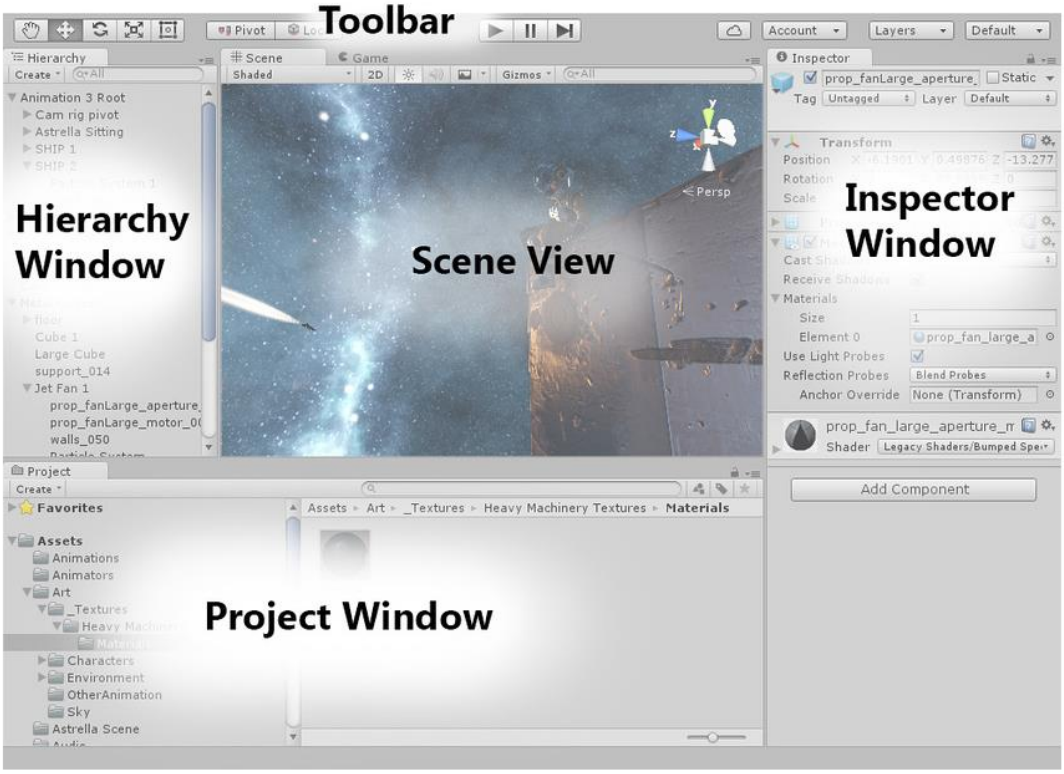


Abb.3: Unity Interface (Technologies, 2019b)

In Abb.3 ist eine Übersicht der einzelnen Komponenten des Editors von Unity zum besseren Verständnis visualisiert worden.

Ein weiterer elementarer Baustein für die Programmierumgebung Unity sind *Scripts.* Scripts sind geschriebene Codes, die bestimmen, wie sich ein Spiel oder Objekt verhalten soll. Dieser Code wird in der Programmiersprache C# geschrieben. Dabei werden häufig für die Programmierung Visual Studios oder Mono verwendet. Viele Objekte oder anderweitige Applikationen benötigen ein Script, sodass sie auf den Input beispielsweise der Spieler reagieren können. Andere Applikationen brauchen den Scriptcode um Events, die im Spiel vom Programmierer gewünscht sind, zeitlich zu bestimmten und auszuführen. Ein weiteres Feature von Scripts ist die Möglichkeit, grafische Effekte entstehen zu lassen, oder das physikalische Verhalten von Objekten

zu verändern (Technologies, 2019d). Um dem Benutzer das Programmieren zu vereinfachen, hat Unity eine Reihe von Scripting Tools (übersetzt: Programmierwerkzeuge) dem Editor hinzugefügt. Eines der wichtigsten Tools ist dabei das Konsolenfenster. Dieses Fenster erlaubt dem Benutzer in kurzer Zeit jegliche Fehlermeldungen, Warnmeldungen oder andere Arten von Nachrichten, die von Unity identifiziert wurden, zu erkennen. Das hilft dem Programmierer beispielsweise dabei einen Fehler in einem Code eines Scripts schnell zu lokalisieren. Auch eigene Nachrichten, die der Benutzer mittels den Befehlen *Debug.Log, Debug.LogWarning* oder *Debug.LogError* in sein Script implementiert, kann in dem Konsolenfenster jederzeit ausgewertet werden. Dadurch können Teile des Codes auf Korrektheit überprüft werden (Technologies, 2019a).

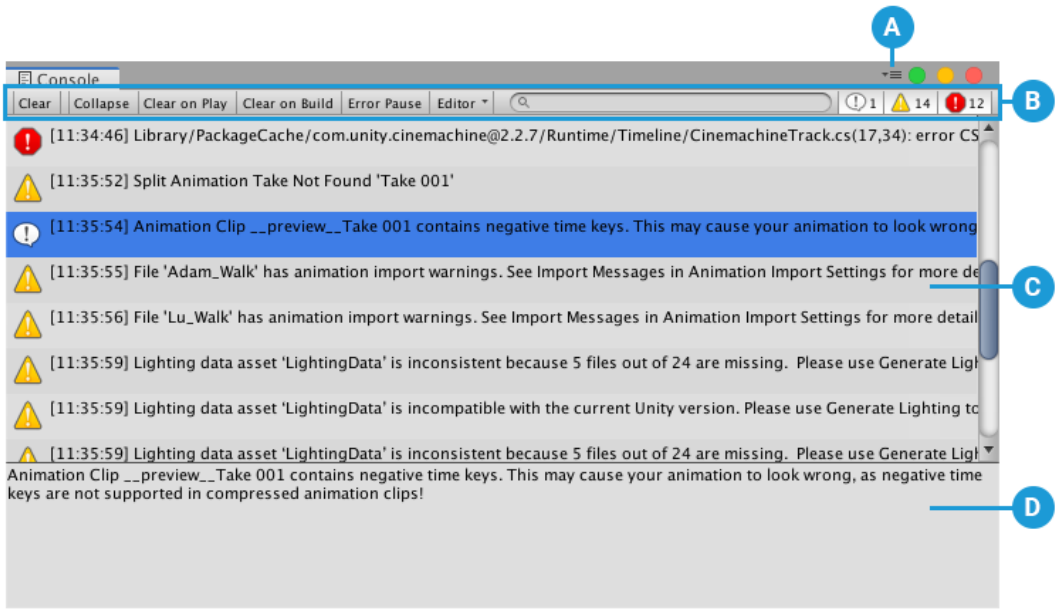


Abb.4: Konsolenfenster von Unity (Technologies, 2019a)
A: Das Menu des Konsolenfensters mit verschiedene Optionen
B: Die *Toolbar* des Konsolenfensters
C: Das Fenster der Konsole wo jede Nachricht aufgelistet wird
D: Der Detailbereich wo die Beschreibung der ausgewählten Nachricht ist

Zum Schluss ist noch der *Asset Store* zu erwähnen. Der Asset Store ist eine im Editor integrierte Online-Plattform, die es dem Benutzer ermöglicht, freie oder kommerziell erwerbliche Accessoires herunterzuladen und sie seinem Spiel beizufügen. Besagte Accessoires haben eine Spannweite von Modellen mit Animationen bis hin zu Tönen/Sounds oder Texturpakete. Diese Online-Plattform wird von Unity-Technologies aber auch von den Endnutzern der Entwicklungsplattform immer wieder erweitert und mit neuen Elementen gefüllt (Technologies, 2019f).

3 Konzept

Das Konzept dieser Bachelorarbeit entstand durch die Frage, ob und in welcher Weise es möglich ist, eine starke Immersion zu erzeugen, die hauptsächlich aus negativen Impulsen besteht. Es wurde bereits durch andere Arbeiten festgestellt, dass auch durch negative Einflüsse Immersion entstehen kann (Jennett et al., 2008, Robillard, Bouchard, Fournier, & Renaud, 2003) . Dieser Idee folgend, sollten in der vorliegenden Arbeit Gefühlen und Emotionen des Benutzers manipuliert werden, um eine Empfindung von Immersion zu kreieren und spontane Reaktionen des Benutzers hervorzurufen.

Die Virtual Reality Anwendung wurde so konzipiert, dass der Benutzer so wenig wie möglich mit dem Spiel interagieren sollte. Stattdessen sollte dieser ausschließlich Reize als auch Impulse verarbeiten und bewerten. Eine weitere Herausforderung des Spiels war die Vorgabe, dass sich der Spieler, trotz negativer Impulse, nicht von der Reizquelle entfernen kann. Der Benutzer hat die Möglichkeit via Datenbrille sich in alle Richtungen zu drehen und sich umzuschauen, jedoch waren zu keiner Zeit Fluchtmöglichkeiten vor den Impulsen möglich. Diese Manipulation der Anwendung soll zur Steigerung der negativen Emotionen und zu spontanen, emotionalen Reaktionen beitragen.

3.1 Entstehung des virtuellen Szenarios

Im Folgenden wird die Zusammensetzung bzw. die Entstehung des VR-Szenarios vorgestellt. Dabei werden für ein breiteres Verständnis auf die einzelnen, programmierten Elemente näher eingegangen. Zu Beginn befindet sich der Spieler in einem verdunkelten Raum, in der eine Musik zu hören ist. Töne sowie Musik wurden dazu genutzt, die Immersion zu steigern (Sanders & Cairns, 2010). Zwei virtuelle Arme befinden sich während der gesamten Anwendung rechts und links vom Spieler. Diese sollen später seine Bewegungen in der realen Welt imitieren, um dadurch eine höhere Immersion zu erzeugen (Abb.5). Der Raum wurde mit Absicht verdunkelt, sodass dieser eine düstere und unbehaglichere Atmosphäre wiederspiegelt. Auch die Musik, die zu hören ist, trägt zu einer bedrohlichen oder angsteinflößenden Stimmung bei. All diese anfänglichen Reize sollen den Benutzer bereits vor dem Start der Anwendung in einen angespannten oder nervösen Zustand versetzten.

Abb.5: Startszenario des Spielers

Der Spieler wird durch den Monitor dazu aufgefordert, die eigene Leertaste in der realen Welt mit beiden Zeigefingern zu drücken und gedrückt zu halten. Sobald der Spieler dieser Aufforderung gefolgt ist, simulieren die virtuellen Hände diese Bewegung und drücken ebenfalls die Leertaste. Diese Simulation, soll den Spieler noch stärker davon überzeugen, dass die virtuellen Arme zu einem Teil von ihm selbst geworden sind. Sollte der Spieler zu irgendeinem Zeitpunkt die Leertaste loslassen, weil er sich beispielsweise erschreckt, so registriert das Spiel diese spontane Reaktion und bricht die Anwendung ab. Der Benutzer hat das Spiel verloren, was ihm mit den Worten „Oh no! You Lost!!“ mitgeteilt wird (Abb.6).

Abb.6: Simulation der Arme

Sobald der Spieler die Leertaste gedrückt hat, beginnt die Anwendung. Mittels verschiedener negativer Reize und Impulsen soll der Spieler zu einer spontanen, emotionalen Reaktion gebracht werden, die sich darin äußert, die Leertaste loszulassen. Die ersten gesetzten Impulse sind Spinnen (Abb. 7).

Abb.7: Modell der Spinnen aus dem Unity Asset Store

Spinnen wurden deshalb ausgewählt, da sie Gegenstand von einer der meist verbreitetsten Phobien sind und bei vielen Menschen Angst auslösen (Pössel & Hautzinger, 2002). Evolutionär bedingt versuchen Menschen in der Regel Spinnen oder anderen Reptilien auf Distanz zu halten und verspüren Insekten gegenüber Abneigung. Dieser Impuls wird sich in der Anwendung zu Nutze gemacht. Der Spieler soll mit den Spinnen in einen engen Kontakt kommen, dem er sich schutzlos ausgeliefert fühlen soll. Nach Beginn der Anwendung soll zunächst ein Krabbel-Geräusch vom Spieler wahrgenommen werden. Nur wenige Augenblicke später erscheinen zwei Spinnen auf dem Schreibtisch, die sich einen Weg zu den virtuellen Armen des Anwenders bahnen. Die Spinnen wurden so konzipiert, dass sie an den virtuellen Armen hochlaufen und am Ende des Armes stehen bleiben. Zusätzlich zu den Geräuschen, die von den Spinnen ausgehen, sind die Spinnen mit einer sehr realistischen Laufanimation ausgestattet. Diese konnte mit den Modellen aus dem Unity Store heruntergeladen werden. Diese Maßnahme sollen die virtuellen Spinnen realistischer wirken lassen, sodass eine spontane und emotionale Reaktion bei den Benutzern hervorgerufen wird. Nachdem der Spieler die Spinnen überstanden hat, bricht die anfängliche Musik, die weiterhin im Hintergrund zu hören war, ab und es wird für einen Moment still. Dieser Effekt hat normalerweise die Wirkung, dass dem Spieler unterbewusst suggeriert wird sich in einer gefährlichen Situation zu befinden oder dass ein gefährliches Event jeden Augenblick eintritt. Das hat zur Folge, dass Stress erzeugt bzw. verstärkt wird (Schneiderman & McCabe, 2013).

Der folgende Impuls ist ein Schreckmoment (auf Englisch *Jumpscare*). Ein Schreckmoment ist ein Event, mit dem der Spieler nicht gerechnet hat oder nicht rechnen konnte und dadurch Stress sowie eine Furchtreaktion ausgelöst wird. Der Grund für diesen Anwendung ist, dass sich der Spieler während der Anwendung nicht sicher sein soll, was als nächstes auf ihn zukommt. Diese Ungewissheit, ist Teil der aufgebauten negativen Stimulation, die der Spieler während der Anwendung erfährt. Für den Schreckmoment fällt ein Objekt, ohne musikalische Vorankündigung von der Decke auf die rechte virtuelle Hand und deformiert diese (Abb.8). Das Objekt selbst ist eine gewöhnliche Kugel, die mit einem Texturpaket versehen worden ist, sodass sie an

einen Stein erinnert. Es ist wichtig, dass das Objekt plötzlich erscheint, da es ansonsten nicht den gewünschten Effekt einer Schreckreaktion erfüllen kann. Da Menschen sich und ihren Körper in der Regel vor körperlichen Verletzungen oder anderen Gefahren schützen wollen, wird mittels dieses Elementes versucht, den Spieler zu einer Schutzreaktion zu zwingen. Eine logische Schutzreaktion im Sinne der Furchtreaktion, wäre seine reelle, rechte Hand von der Tastatur wegzuziehen. Um die Szene für den Anwender noch unangenehmer zu gestalten, wird beim Kontakt des Objektes mit der rechten Hand ein Geräusch hörbar, welcher an das Brechen von Knochen erinnern soll. Auch ein männlicher Schmerzensschrei wurde dem Spiel hinzugefügt. Ziel des Ganzen ist es, die plötzliche Gefahrensituation so realistisch wie möglich darzustellen, damit der Spieler die spontane Reaktionen zeigt und die Leertaste loslässt.

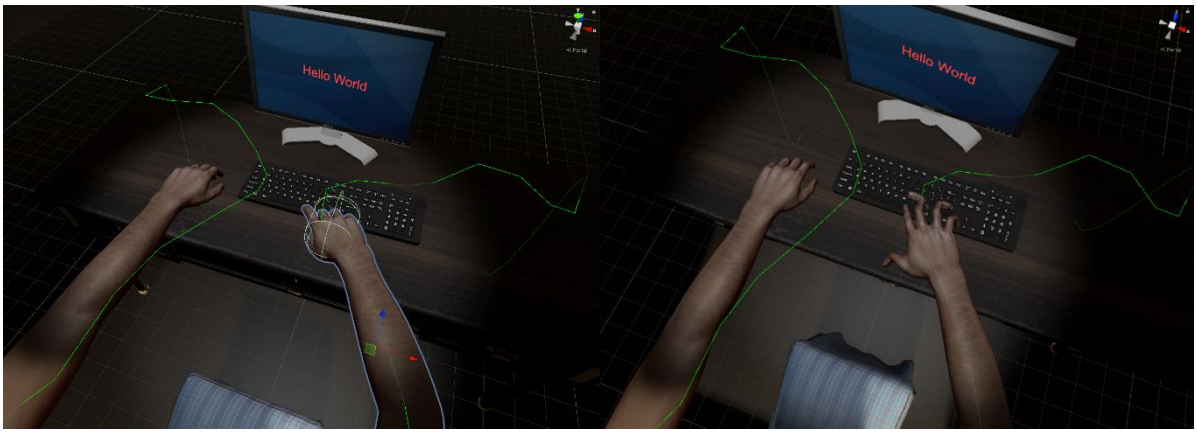


Abb.8: Linkes Bild – Rechte Hand vor Aufprall | Rechtes Bild – Rechte Hand nach Aufprall

Nachdem die Hand deformiert wurde und der Spieler sollte die Leertaste noch nicht losgelassen haben, ertönt ein neues Geräusch. Dieses suggeriert, dass ein nächster Impuls unmittelbar bevorsteht. Sekunden später zerbricht ein Fenster hinter dem Spieler. Das Zerbrechen des Fensters ist mit einem entsprechendem Geräusch versehen, dass an das Zersplittern von Glas erinnert. Auch hier soll der Sound das Geschehen glaubwürdiger machen. Das Zerspringen des Fensters soll den Spieler dazu bewegen sich umzudrehen und nachzuschauen, was hinter ihm passiert ist. Sobald sich der Spieler umdreht steht eine Vogelscheuche dicht hinter ihm, die wie die Kugel einen Schreckmoment auslösen soll (Abb.9).

Der Sinn dieses gesamten Elements ist weniger den Spieler eine solche Angst einzujagen, dass er die Leertaste loslässt, sondern soll ihn von der noch kommenden Szene ablenken. Der Spieler soll in dem Glauben gelassen werden, dass die Vogelscheuche der nächste große Schreckmoment war. Sobald der Spieler sich umgedreht und die Vogelscheuche erblickt, ertönt eine dunkle und düstere wirkende Stimme, die den Spieler auf Englisch dazu auffordert, sich wieder umzudrehen. Kommt der Spieler der Aufforderung nach, taucht eine große, monsterartige Gestalt vor dem Spieler auf (Abb.10).
Dieses Monster stößt bei Blickkontakt mit dem Spieler, ein Gebrüll aus und macht augenscheinlich Anstalten, sich auf den Benutzer zu stürzen. Das Monster wurde deshalb ausgewählt, weil bisher noch keine übernatürlichen bzw. mysteriösen Gestalten vorkamen. Somit kann davon ausgegangen werden, dass das Erscheinen einer fiktiven Gestalt den Spieler überrascht. Zudem wurde das Monster im Vergleich zu den

restlichen Elementen deutlich größer überdimensioniert und dicht vor den Spieler platziert, sodass dieser dazu gezwungen wird, dem Monster direkt ins Gesicht zu blicken. Hier ist es erneut das Ziel, dem Spieler Angst einzuflößen, indem er in eine Situation versetzt wird, in der er sich unwohl fühlen soll. Die Tatsache, dass er nicht vor dem übergroßen Monster fliehen kann, sondern ihm ins Gesicht schauen muss, soll negative Emotionen sowie die empfundene Immersion steigern.
Nach dem Erscheinen des Monsters, gilt die Anwendung als bestanden, sollte der Spieler bis dahin die Hände nicht von der Leertaste gezogen haben.

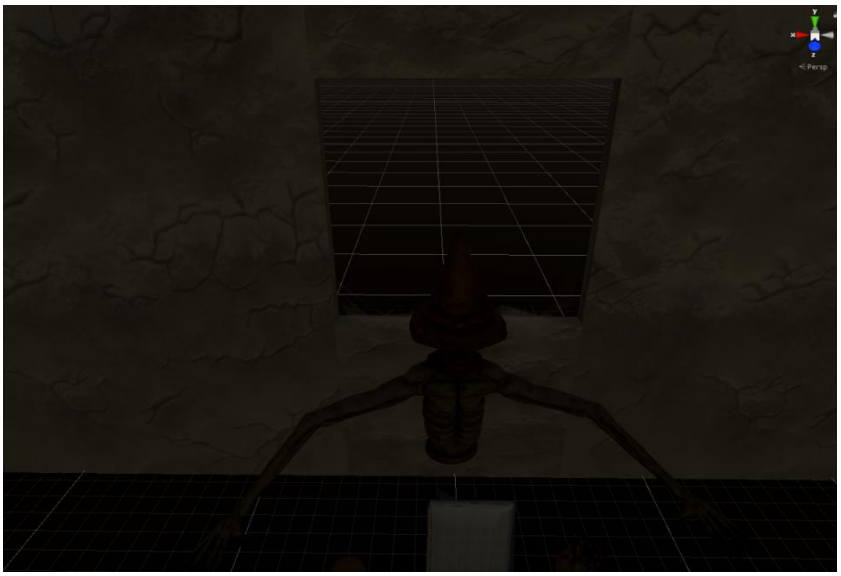


Abb.9: Modell der Vogelscheuche & Kaputtes Fenster

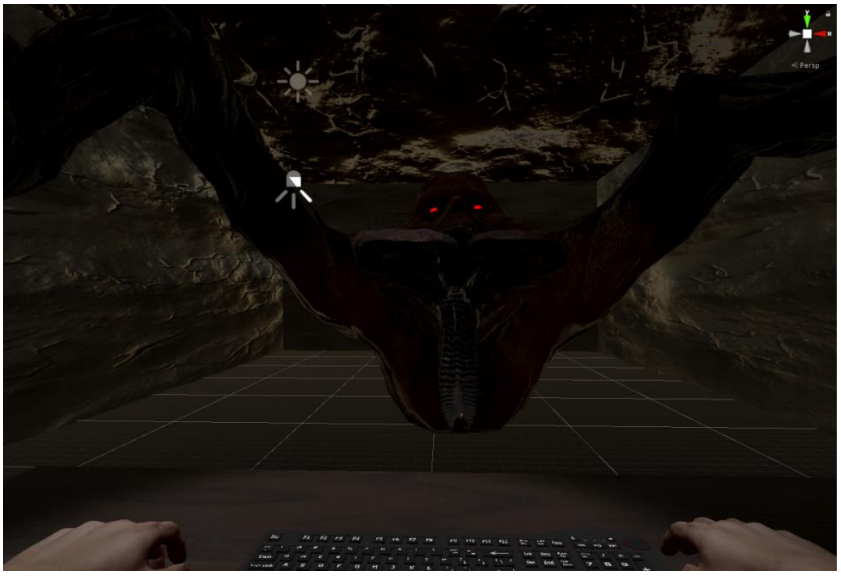


Abb.10: Sichtweise des Spielers auf das Monster

Im nächsten Schritt soll die VR-Anwendung von Probanden getestet werden. Im Anschluss an das Spiel füllen die Versuchspersonen einen Fragebogen aus, welcher die empfundene Immersion messen soll. Dazu wurde der Fragebogen von Jennet et al. (2008) verwendet und vom Autor ins Deutsche übersetzt.

4 Implementation

4.1 Material

Für die Umsetzung wurde ausschließlich die Programmierumgebung Unity verwendet. Unity bietet ein umfangreiches Angebot an Möglichkeiten, sodass auf weitere Programmierumgebungen verzichtet werden konnten. Die Scripts wurden in der Programmierumgebung Visual Studios 2017 in der Programmiersprache C# geschrieben. Jenes ist mit Unity kompatibel und daher unkompliziert und unproblematisch zu benutzten.
Zur Kontrolle der programmierten Ergebnisse wurde eine Oculus Rift benutzt. Diese wurde durch zwei Sensoren unterstützt. Die Handcontroller wurden für diese Anwendung nicht genutzt.

4.1.1 Modelle

Unter Modellen sind hier verschieden Objekte zu verstehen, die in das Spiel integriert wurden, um gewünschte Effekte zu erzielen. Diese Objekte wurden aus dem Asset Store von Unity heruntergeladen und anschließend in modifizierter Form in das Spiel integriert, sodass sie für das Ziel dieser Arbeit passend waren.

4.1.2 Ton und Musik

Der Ton sowie die Musik in dem Spiel sind einerseits frei andererseits erwerblich aus dem Internet heruntergeladen und in die Anwendung integriert worden.

4.2 Wichtige Programmierelemente

In diesem Abschnitt wird auf wichtige Scripts und Codeelemente eingegangen, die für die Anwendung entscheidend sind. Im ersten Teil des Abschnitts, wird der Grundaufbau eines Scripts in C# knapp erläutert. Darauf aufbauend werden die beiden Scripts Player und Follower vorgestellt. Das Player-Script dient als Kernelement dieser hier vorgestellten Anwendung, während das Follower-Script ein Beispiel dafür ist, wie negative Impulse im Spiel implementiert werden. Abschließend wird die Umsetzung der Handanimationen erläutert.

4.2.1 Grundaufbau eines Scripts

Allgemein gesehen, kann man ein Script als eine gewöhnlichen Klasse verstehen. Diese Besitzt verschiedene Methoden und Variablen, die am Anfang deklariert werden müssen. Im Folgenden werden zwei wichtige Methoden vorgestellt, die bereits standardmäßig in jedem Script enthalten sind. Besagte Methoden haben in der späteren Programmierung entscheidenden Einfluss auf die Anwendung.

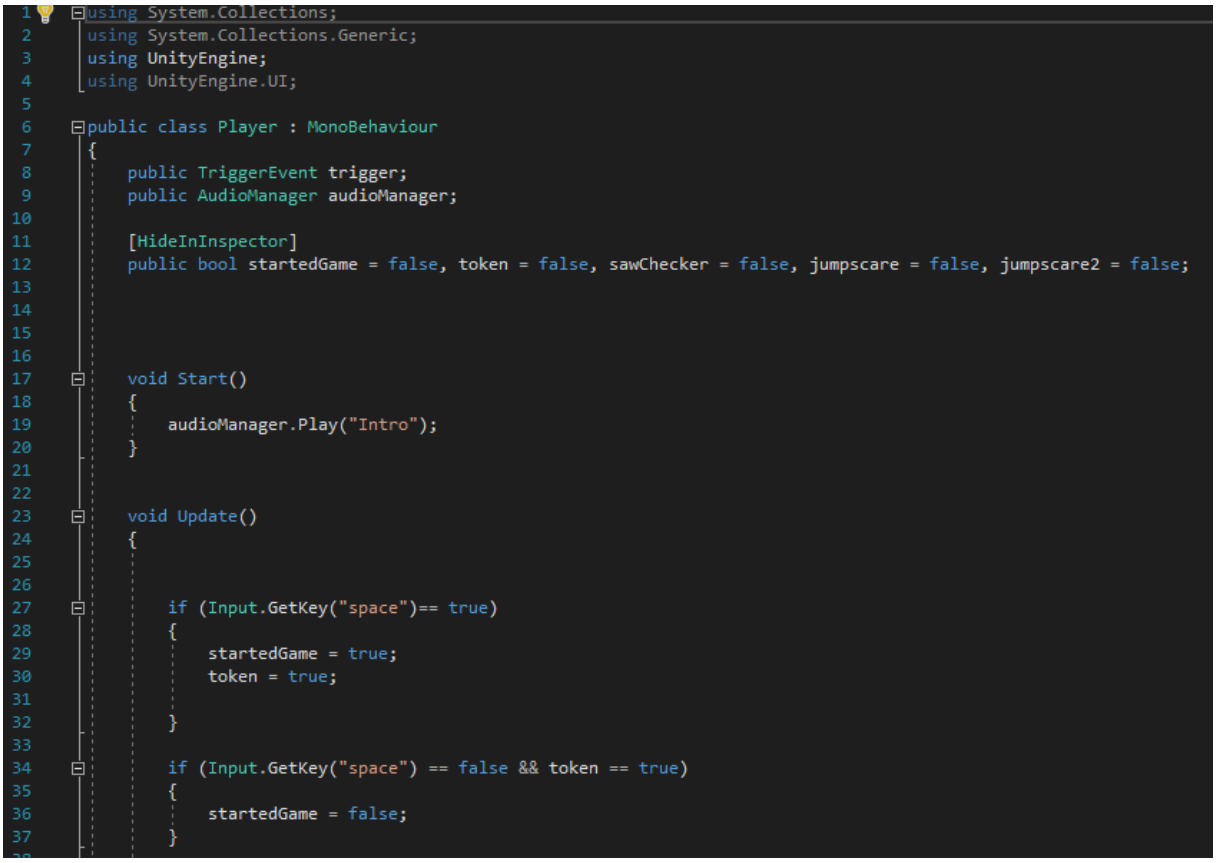
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class script : MonoBehaviour
{
    // Start is called before the first frame update
    void Start()
    {

    }

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

    }
}
```

Abb.11: Grundaufbau eines Scripts

Der Aufbau des Scripts, wie in Abb.11 dargestellt, wird bei dem Erstellen eines neuen Scripts bereits vom Programm selbst generiert. Zeile eins bis drei zeigen verschiedene Libaries, die ein Script benötigt, um mit Unity interagieren zu könne. In Zeile fünf wird der Name der Klasse festgehalten. In diesem Fall ist die Klasse als *„script“* benannt worden. Es ist entscheidend, dass der Klassenname im Editor von Unity identisch mit dem Klassennamen des Scripts ist. Andernfalls wird Unity den gewünschten Code nicht finden. Die erste Standardmethode ist die sogenannte *void Start()* Methode. Alle Anweisungen, die in dieser Methode gelistet sind, werden von Unity vor dem Start des ersten Frames des Spiels initialisiert. Die zweite notwendige Methode ist die *void Update()* Methode. In dieser wird jeglicher Code jedes Frame erneut überprüft. Mögliche Beispiele, die in der Update-Methode stehen könnte, wären Bewegungen oder Aktionen, die von dem Spieler selbst ausgelöst wurden. Erst wenn das Programm beendet wird, hört die Update-Methode auf, den Code erneut zu prüfen.

4.2.2 Player-Script

Das Player Script ist eines der wichtigsten Scripts in der Anwendung dieser Bachelorarbeit. Das Script wird der Kamera (die den Spieler in VR darstellt) angeheftet und hat zugleich mehrere Aufgaben.



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

public class Player : MonoBehaviour
{
    public TriggerEvent trigger;
    public AudioManager audioManager;

    [HideInInspector]
    public bool startedGame = false, token = false, sawChecker = false, jumpscare = false, jumpscare2 = false;




    void Start()
    {
        audioManager.Play("Intro");
    }


    void Update()
    {


        if (Input.GetKey("space")== true)
        {
            startedGame = true;
            token = true;

        }

        if (Input.GetKey("space") == false && token == true)
        {
            startedGame = false;
        }
```

Abb.12: Erster Teil des Player-Scripts

Abb.12 zeigt, wie der erste Teil des Player-Scripts aufgebaut ist. In Zeile sechs ist der Name der Klasse vermerkt: Player. Bevor nun in die Start-Methode oder die Update-Methode hineingegangen wird, werden Variablen und Referenzen zu anderen Scripts gesetzt. Die Zeilen 8 und 9 verweisen auf andere Scripts mit den Namen *TriggerEvent* und *AudioManager*. Somit besteht die Möglichkeit, dass gesetzte Scripts untereinander kommunizieren können. Der *HideInInspector* Befehl, in Zeile elf, hat die Aufgabe, Variablen in der darauffolgenden Zeile unsichtbar für den Unity-Editor zu machen. In der Start-Methode wird mittels der Variable *audioManagers* auf das AudioManager-Script verwiesen. Dieses Script hat eine Vielzahl an Tönen und Musikstücken. Die Codezeile in der Start-Methode hat die Aufgabe, durch die Notation [*.play(„Intro“)*] den Sound mit dem Titel „Intro“ aus der Liste der AudioManager- Klasse abzuspielen. Das hat zur Folge, dass sobald die Szene gestartet wird, eine Intro-Musik dem Benutzer vorgespielt wird.

In der Update-Methode wird mit der ersten if-Abfrage überprüft, ob der Spieler die Leertaste gedrückt hält und mit der zweiten if-Abfrage ob er sie zu einem beliebigen Zeitpunkt losgelassen hat. Es ist entscheidend, dass der Befehl in Zeile 27, *Input.GetKey(„space“)* lautet und nicht *Input.GetKeyDown(„space“)*, denn bei der zweiten Formulierung würde nur einmal überprüft werden, ob die Leertaste gedrückt worden ist. Es ist aber für die Anwendung von Bedeutung, dass das Programm feststellen kann, ob die Leertaste konstant gedrückt gehalten wird. Sobald der Spieler das Spiel gestartet hat, werden die Variablen *startedGame* sowie *token* auf true gesetzt. Die Variable *startedGame* ist später im Follower-Script von weiterer Bedeutung, während die *token*-Variable unmittelbar im nächsten if-Block genutzt wird. Dieser Block nutzt die *token*-

Variable, um festzustellen ob die Anwendung bereits einmal gestartet worden ist. Das Spiel soll nur dann abbrechen, wenn es schon einmal gestartet wurde.

```
38
39          RaycastHit hit;
40          if(Physics.Raycast(transform.position, transform.forward, out hit))
41          {
42              if(hit.transform.GetComponent<Monster>() !=null && trigger.visibleMonster == true)
43              {
44
45                  jumpscare = true;
46                  trigger.visibleMonster = false;
47              }
48
49              if(hit.transform.GetComponent<Hexe>() !=null && trigger.visibleHexe == true)
50              {
51                  jumpscare2 = true;
52                  trigger.visibleHexe = false;
53              }
54
55              if(hit.transform.GetComponent<CreepySounds>() !=null)
56              {
57
58                  sawChecker = true;
59              }
60          }
61
62      }
63
64
65  }
```

Abb.13: Zweiter Teil des Player-Scripts

Der zweite Teil des Codes beschäftigt sich mit der Frage, auf was der Spieler im Spiel tatsächlich blickt (Abb.13). Dazu benötigt man die Klasse *RaycastHit*. Ein Raycast, wirft einen Strahl gegen alle Kollisionspartner in der Szene und liefert detaillierte Informationen darüber, was getroffen wurde (Technologies, 2019c). Dieser geht von der Blickrichtung des Spielers, innerhalb des Spiels aus. In Zeile 40 werden die Eigenschaften des Raycasts festgelegt. Durch *transform.position* wird die Position des Spielers festgelegt, *transform.forward* beschreibt, wie sich der Strahl verhält (in diesem Fall verläuft er gerade nach vorne). Die letzte Variable, die in Zeile 39 gesetzte wird, überprüft, welches Objekt getroffen wird.

Die nächsten drei if-Abfragen behandeln drei verschieden Komponenten (Monster, Hexe und CreepySounds). Sobald der Spieler eines dieser drei Komponenten ansieht, wobei bei den Komponenten Monster sowie Hexe noch eine weitere Variable aus dem Script *TriggerEvent* auf true stehen muss, werden neue Parameter initialisiert. Diese Parameter lösen zusätzliche Scripts im Folgenden aus. Somit lassen sich Effekte und Impulse nur durch den Blick des Spielers provozieren und zeitlich abstimmen.

4.2.3 Animationen

Animationen spielen in dieser VR-Anwendung ebenfalls eine große Rolle und werden daher zum besseren Verständnis näher beleuchtet. Es gibt mehrere Objekte, denen mittels der Animationstechnik von Unity in dieser Anwendung Mobilität verliehen wird. Da das Konzept der Animation bei allen Objekten in dieser Anwendung nahezu gleich verläuft ist eine mehrfache Erklärung redundant und wird daher vermieden. Animationen sind unerlässlich, wenn eine realistische Ausführung der Objekte erwünscht ist. Per Code werden später die Animationen abgerufen und in das Programm integriert.

Im Folgenden wird die rechte virtuelle Hand betrachtet.

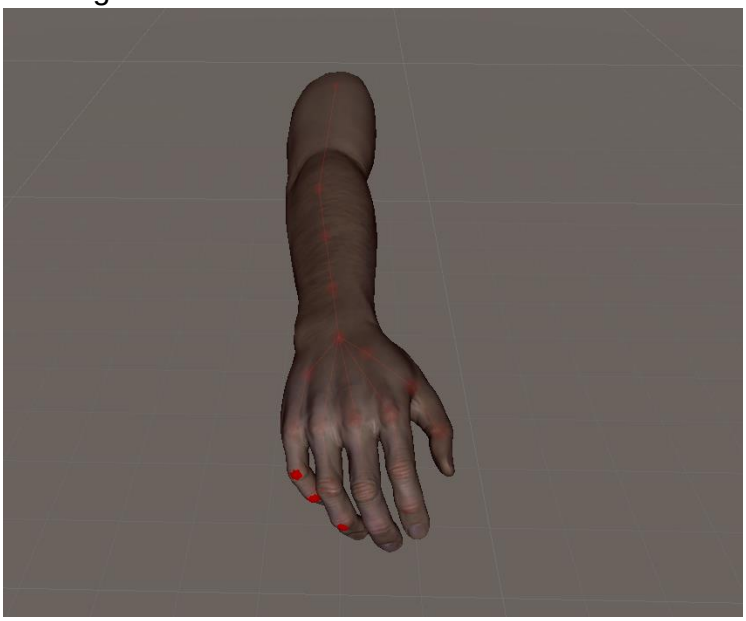


Abb.14: Aufbaustruktur des rechten Arms

In Abb.14 erkennt man ein simples Skelett, das durch den rechten Arm (hier rot markiert) verläuft. Die roten Punkte sind Stellen, auf die man in der Animation zugreifen und diese mittels verschiedener Transformationen verändern kann. Eine Transformation kann in Form einer Skalierung, Rotation und Transposition durchgeführt werden. Die Transformationen, die in der Anwendung benutzt werden, sind ausschließlich Drehungen um die besagten Knochenpunkte im Handskelett. Dadurch wird eine realistische Darstellung gewährleistet.

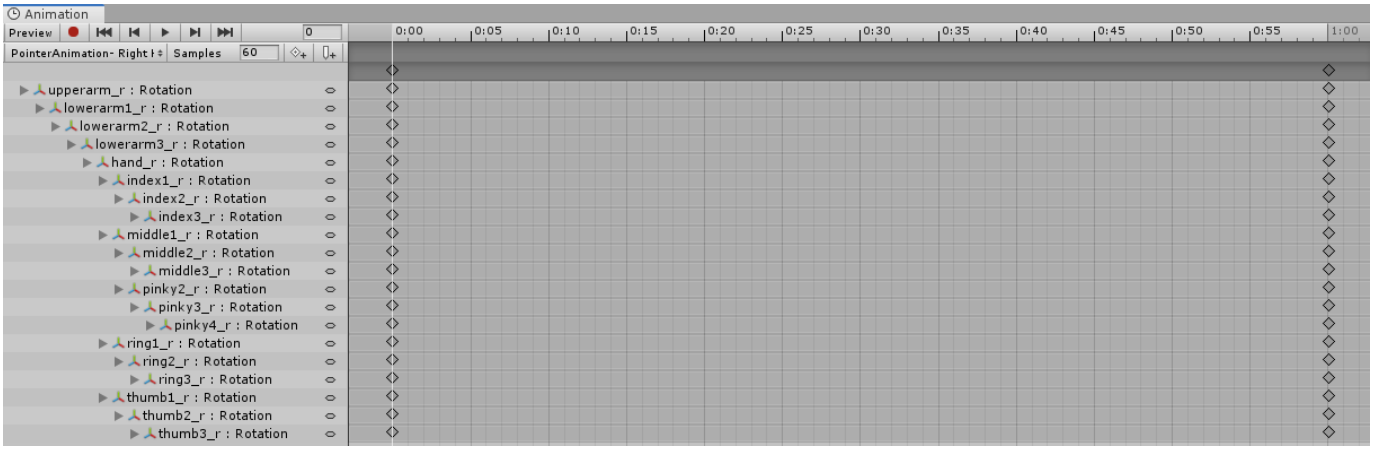


Abb.15: Animationstool von Unity

Mit der Hilfe des Animationstools von Unity (Abb.15) kann man eine gewünschte Animation erstellen. Die Knochenpunkte in der Skeletthand werden zu diesem Zweck in besagtes Tool transferiert, wo sie transformiert werden können. Es wird im Normalfall immer mit der Ausgangsposition des Objektes gestartet, welche soweit verändert wird bis die gewünschte Position erreicht ist. Diese Position wird als Endposition referenziert. Das Tool ermöglicht die Festsetzung eines zeitlichen Rahmens für die Animationen. Je kürzer man den zeitlichen Rahmen festlegt, desto schneller wird die Animation ausgeführt. Bezogen auf das Beispiel in diesem Abschnitt benötigt die rechte Hand

genau eine Sekunde, um von der Ausgangsposition in die Endposition zu gelangen (Abb.16).

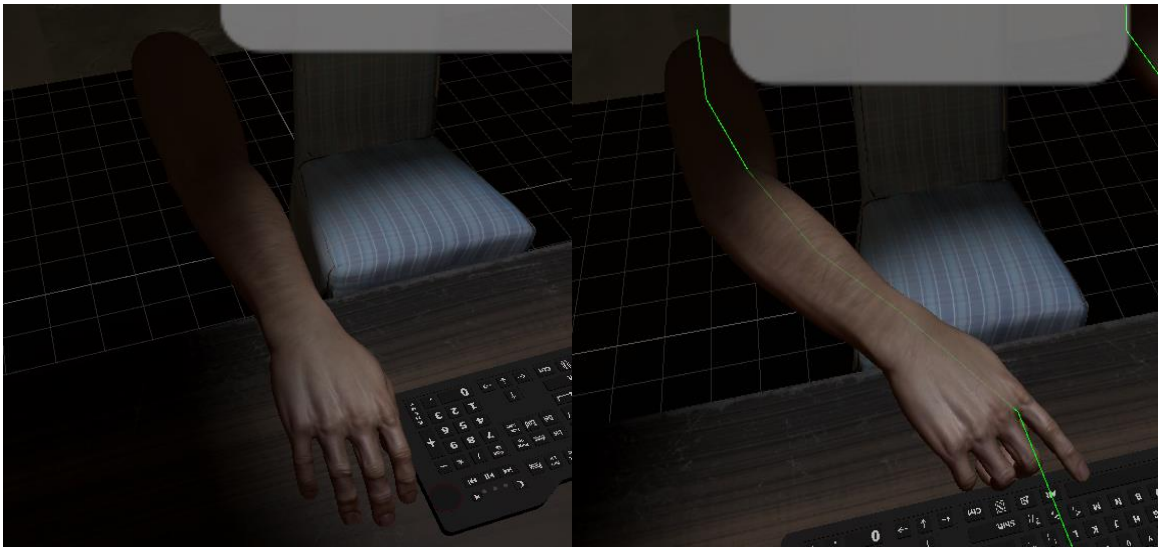


Abb.16: Ausgangsposition (links) – Endposition (rechts)

Sobald alle Transformationen abgeschlossen sind, wird die Animation mit einem Namen versehen und abgespeichert. Ein weiteres Tool von Unity namens *Animator* ist dafür zuständig, die erschaffene Animation zu dem richtigen Zeitpunkt innerhalb der Anwendung abzuspielen (Abb.17). Der Animator ermöglicht überdies den flüssigen Übergang zwischen mehreren Animationen eines Objekts. Damit ist er der Manager aller Animationen für ein Objekt. Der Wechsel zwischen zwei Animation wird mittels Trigger im Animator festgehalten. Der Trigger bewirkt, dass eine Bool-Variable, die manuell gesetzt werden muss, seinen Wert verändert. Dieser Übergang signalisiert dem Animator, dass er von einer Animation in die nächste schalten muss. Dieser Vorgang ist im Code umgesetzt.

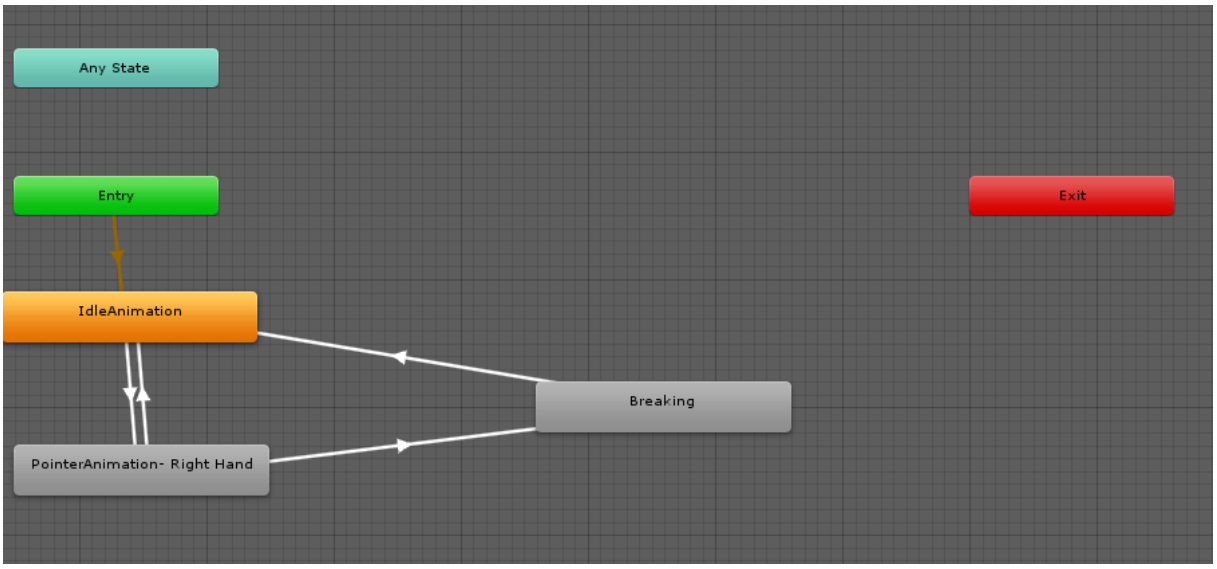


Abb.17: Animator – Illustration von Animationsübergängen im Animator

4.2.4 Follower-Script

Abschließend wird ein finales Element vorgestellt, dass für eines der negativen Reize und Impulse zuständig ist: die Spinne. Dabei wird auf Besonderheiten im Code eingegangen und erklärt, wie das Follower-Script (Spinnen-Script) mit anderen Scripts sowie dem Animator kommuniziert. Da die meisten Objekte nach einem ähnlichen Prinzip verlaufen, wird darauf verzichtet, diese ebenfalls vorzustellen.

Um den Code der Spinne besser zu verstehen, wurde diese in drei Teile unterteilt.

```
1   using System.Collections;
2   using System.Collections.Generic;
3   using UnityEngine;
4   using PathCreation;
5
6   public class Follower : MonoBehaviour
7
8   {
9       public Player player;
10      public PathCreator pathCreator;
11      public float speed = 5;
12      float distanceTravelled;
13      private Animator animator;
14      bool startWalking = false;
15      bool playing = true;
16      public EndOfPathInstruction endOfPathInstruction;
17      public AudioManager audiomanager;
18
19
20      [HideInInspector]
21      public bool endOfSpider = false;
22
```

Abb.18: Teil 1: Spinnen-Script

Damit das Script problemlos abläuft, müssen vorab Variablen sowie Pakete gesetzt werden (Abb.18). Die Zeilen eins bis vier sind Libaries, die wiederum viele verschiedene Klassen beinhalten. Ein besonderes Augenmerk ist auf das *PathCreation*-Paket in Zeile vier zu legen. Mit dieser Library wird das s.g. Pathing (Wegabläufe) umgesetzt. Die in dem Paket enthaltenen Methoden, ermöglichen es Objekten einem Pfad zu folgen.

Zu Anfang des Scipts werden eine Reihe von verschiedenen Klassen initialisiert. Das Spinnen-Script benötigt Informationen aus besagten Klassen, damit der Code korrekt ausgeführt werden kann. Der Rest sind einfache Bool bzw. Float- Variablen, die einen Wert zugewiesen bekommen haben.

```
24      void Start()
25      {
26          animator = GetComponent<Animator>();
27
28      }
29
30      // Update is called once per frame
31      void Update()
32      {
33
34
35          if (this.animator.GetCurrentAnimatorStateInfo(0).IsName("idle") && player.startedGame == true && playing == true){
36
37              playtrack();
38              playing = false;
39              StartCoroutine(spiderMovement());
40          }
41
42
43          if(startWalking == true)
44          {
45              distanceTravelled += speed * Time.deltaTime;
46              transform.position = pathCreator.path.GetPointAtDistance(distanceTravelled, endOfPathInstruction );
47              transform.rotation = pathCreator.path.GetRotationAtDistance(distanceTravelled, endOfPathInstruction);
48
49          }
50
51
52          if(transform.position == pathCreator.path.GetPoint(distanceTravelled, endOfPathInstruction))
53          {
54              animator.SetBool("isReady", false);
55          }
56
```

Abb.19: Teil 2: Spinnen-Script

Beginnend mit der void Start()-Methode wird die Variable *animator*, die in Zeile 13 (Abb.18) definiert wurde, gesetzt. Diese Variable greift nun auf die Klasse *Animator*, die die Animationen der Spinne gespeichert hat, zu. In der void Update()-Methode gibt es insgesamt vier verschieden if-Abfragen (Abb.19 & Abb.20), die konstant überprüfen, ob ein gewisser Zustand eingetreten ist oder nicht. Im Folgenden werden diese if-Abfragen näher beleuchtet.

Beginnend mit der ersten if-Abfrage in Zeile 35 hat diese die Aufgabe drei Zustände konstant zu überprüfen: Hat der Spieler das Spiel gestartet, befindet sich die Spinne in der *idle*-Position (Ausgangsposition) und wurde die Bool-Variable der Wert true zugewiesen. Bei Erfüllung dieser Voraussetzungen werden zwei Methode namens *playtrack* sowie *spiderMovement* aufgerufen und die Bool-Variable, die zunächst auf true stand, wird auf false gesetzt. Insbesondere bei dieser if-Abfrage ist die Zustandsüberprüfung interessant, welche feststellt ob der Spieler das Spiel gestartet hat oder nicht. In Zeile 35 kann man erkennen, dass dies durch den Codeausdruck *player.startedGame == true* passiert. Hier ist deutlich zu erkennen, dass die Scripte Player sowie Follower miteinander kommunizieren, wie in Abschnitt 4.2.2 erwähnt. Das Spinnen-Script, kann erst fortfahren, wenn es vom Player-Script die gewünschte Bool-Variable übergeben bekommt.

Des Weiteren soll der Grund für die Bool-Abfrage in der Condition erläutert werden. Diese wurde, wie in Abb.18 ersichtlich, bereits mit dem Wert true initialisiert. Somit erscheint die Abfrage redundant. Der Grund dafür lässt sich mit dem Wissen über die Update-Methode erklären. Da diese erste if-Abfrage in der Update-Methode geschrieben worden ist, wird diese auch jedes Frame erneut überprüft. Der Zustand *startedGame == true* sowie die Ausgangsposition der Spinne ist für längere Zeit als ein Frame der Anwendung wahr. Ohne diese Bool-Variable würden die zwei Methoden in dem ersten if-Block tausende Male erneut aufgerufen werden, was zu einem Absturz des

Systems führen könnte. Durch die Werteveränderung der Bool-Variable in dem selbigen if-Block, kann man garantieren, dass die oben genannten Methoden nur ein einziges Mal aufgerufen werden.

Die nächste if-Abfrage in Zeile 43 beschäftigt sich mit dem Pathing der Spinne. Die benötigte Bool-Variable, die das Betreten des if-Blocks erlaubt bzw. verhindert, wird in Abb.20 näher erläutert. In besagten if-Block selbst wird sich mit der Position, der Geschwindigkeit sowie der Rotation der Spinne beschäftigt. Die Methoden, die im *Path-Creation*-Paket enthalten sind, ermöglichen die Übermittlung der genaue Ausrichtung und Position des Objekts. Auch die Geschwindigkeit, in der sich das Objekt dem vorgegebenen Pfad entlangbewegt, wird in Zeile 45 festgelegt.

Die letzte if-Abfrage in Teil zwei überprüft, ob die Position der Spinne den Endpunkt des Weges erreicht hat. Die Schultern der virtuellen Arme markieren das Ende des Pfades. Sobald dieser Punkt erreicht ist, befiehlt der if-Block dem Animator die Laufanimation der Spinne zu stoppen, sodass sie wieder in die Ausgangsposition (*idle*-Position) zurückkehren kann.

```
57          if (animator.GetBool("isReady") == false && startWalking == true)
58          {
59              endOfSpider = true;
60              startWalking = false;
61              audiomanager.Stop("Spider_Idle 1");
62              audiomanager.Stop("Intro");
63          }
64      }
65
66
67      public void playtrack()
68          {
69           audiomanager.Play("Spider_Grunt 1");
70           audiomanager.Play("Spider_Idle 1");
71          }
72
73
74      IEnumerator spiderMovement()
75      {
76          yield return new WaitForSeconds(5);
77          animator.SetBool("isReady", true);
78          startWalking = true;
79
80      }
81
82
83  }
84
```

Abb.20: Teil 3: Spinnen-Script

In dem letzten Teil des Codes sollte zuerst der *IEnumerator* in Zeile 74 hervorgehoben werden. IEnumeration ist ein wiederkehrendes Werkzeug in den Codes dieser VR-Anwendung, da es die Abstimmung und das Auftreten von Ereignissen problemlos koordinieren und zeitlich planen kann. Mittels dem IEnumerator ist es möglich, eine Aktion zu einem exakten Zeitpunkt geschehen zu lassen. Methoden von IEnumerations sind immer als Routinen im Code zu deklarieren weshalb sie auch in Zeile 39 mit dem Befehl *StartCoroutine* initialisiert wurde. In diesen Methoden lassen sich Verzögerungen oder gar ein kompletter Stillstand im Code herbeiführen, bis eine Variable einen gewünschten Wert hat oder eine gewünschte Zeit vergangen ist. Mittels der Anweisung *yield return new WaitForSeconds(x)* kann man

den Code anweisen, x Sekunden zu warten, ehe er mit der nächsten Codezeile in der Methode weitermachen kann. Eine weitere Anweisung, die oftmals in den Scripts dieses Projektes Verwendung fand, war *yield return new WaitWhile(() => scriptname.scriptvariable == true/false)*. Dieser Ausdruck befiehlt dem Programm solange zu warten, bis eine Variable in einem anderen Script einen bestimmten Boolschen-Wert angenommen hat. Mittels diesen beiden Anweisungen wurde das gesamte Projekt zeitlich abgestimmt, sodass es zu keiner Überschneidung von verschiedenen Impulsen kommen kann.

Im Code selbst, erkennt man, dass eine Wartezeit von fünf Sekunden eingebaut ist (Zeile 76), ehe die Bool-Variable für die Laufanimation sowie die Bool-Variable *startWalking*, auf true gesetzt wird. Dies hat zur Folge, dass nach dem Start des Spiels, der Spieler die Spinnen hören kann, jedoch erscheinen diese erst fünf Sekunden später.
Die Methode *playtrack* wurde ebenfalls im ersten if-Block der Update-Methode aufgerufen. Dieser ist dafür zuständig, dass die Töne der Spinne sowie das Geräusch des Krabbelns zu hören ist. Beide Methoden (*playtrack* sowie *spiderMovement*) sind außerhalb der Update-Methode und werden daher nur einmal durchlaufen.

Abschließend ist der letzte if-Block der Update-Methode zu erwähnen. Dieser markiert das Ende des Follower-Scripts. Die Condition prüft, ob die Laufanimation der Spinne auf false steht sowie ob die Spinne bereits einmal aktiviert worden ist. Somit kann man garantieren, dass das Spinnen-Script erst aufhört, wenn die Spinne selbst am Ende ihres Weges angelangt ist. In besagten if-Block selbst, werden die Töne des Krabbelns sowie der Spinnen selbst gestoppt. Eine Variable, die *endofSpider* genannt wurde, wird auf true gesetzt. Diese Variable wird für das nächste Script, welche die rechte Hand des Spielers deformiert, entscheidend sein, da dieses mittels des IEnumerators darauf wartet, dass die Spinne „abgeschlossen“ ist.

5 Evaluation

5.1 Fragebogen

Ziel dieser Bachelorarbeit ist zu evaluieren, inwiefern die Spieler ein Immersionserleben bei der Anwendung des Spiels empfunden haben. Um Immersion zu messen, wurde eine Fragebogen aus der Arbeit von Jennett et al. (2008) verwendet. Das Original besteht aus 32 Items sowie einer abschließenden Frage zur empfundenen Immersion. Die Items wurden anhand einer fünf stufigen Likert-Skala erfasst, wobei 1 für „starken Widerspruch“ und 5 für „starke Zustimmung“ steht. Die Frage zu der Immersion wurde hingegen auf einer 10-stufigen Bewertungsskala gemessen, wobei 1 für „überhaupt nicht immersiv“ und 10 für „sehr immersiv“ steht. Der Fragebogen wurde für den Zweck dieser Bachelorarbeit aus dem englischsprachigen ins Deutsche übersetzt und auf 12 Items sowie die abschließende Frage reduziert. Die einzelnen Elemente wurden mit demselben Bewertungsmaßstab wie im Originalen gemessen.

Die ersten sechs Items des Fragebogens beziehen sich auf die emotionale Involviertheit, die die Probanden während des Spiels erfahren haben. Unter diesem Begriff wird das Gefühlsempfinden verstanden, das ein Spieler für das Spiel entwickelt sowie verbale oder körperliche Reaktionen des Spielers verstanden. Ein Beispielitem dieser Facette wäre: *„Ich hatte keinerlei emotionale Bindung zu dem Spiel.“*. Die folgenden sechs Items, beziehen sich auf die Manipulation der Probanden. Es soll dabei herausgefunden werden, inwieweit die Spieler das Gefühl hatten nicht mehr in der realen Welt zu sein oder ein Teil des Spiels geworden zu sein. Ein Beispielitem dieser Facette wäre: *„Ich war mir meiner realen Umgebung bewusst.“*. Die letzte Frage *„Wie hoch war die Immersion für dich?“* soll die Immersion des Spiels bewerten, die der Spieler während der gesamten Anwendungsdauer erlebt hat.

5.2 Aufbau des Experiments

Insgesamt wurden sieben Personen für das Experiment akquiriert. Diese haben zunächst die VR-Anwendung durchlaufen müssen und wurden hinterher mittels des Fragebogens zu ihrem Immersionserfahrungen befragt. Von den Probanden waren drei weiblich und vier männlich. Das Alter der Versuchspersonen lag zwischen 22 und 27 Jahren. Alle Probanden gaben an, bisher nur wenig bis keine Erfahrung mit VR-Anwendungen gemacht zu haben.

Für das Experiment wurden stets dieselbe Tastatur sowie derselbe Monitor verwendet. Nachdem die Probanden Platz genommen haben, wurde ihnen die Oculus Rift - Datenbrille aufgezogen. Nachdem alle Einstellungen in Bezug auf die Sitzhöhe, Ausrichtung des Stuhls als auch der VR-Brille zur Zufriedenheit der Probanden ausgerichtet wurden, startete der Versuchsleiter die VR-Anwendung. Während der Durchführung des Experiments wurden die Kandidaten vom Versuchsleiter gefilmt, um mögliche Reaktionen zu dokumentieren. Nach der VR-Anwendung wurden die Benutzer gebeten, den in Abschnitt 5.1 beschriebenen Fragebogen auszufüllen. Nach der Bewertung erfolgte ein kurzer verbaler Austausch über die Anwendung und mögliche Verbesserungen, die man durchführen könnte.

5.3 Auswertung

Wie bereits in Abschnitt 5.1 erläutert, umfasste der Fragebogen drei unterschiedliche Skalen zu Subfacetten der Immersion. Im Folgenden werden die Mittelwerte dieser Facetten angegeben und die Zusammensetzung des Ergebnisses wird erläutert.

Die Ausprägungen der emotionalen Involviertheit weist eine Spannweite von 3,5 bis 4,67 auf. Der Mittelwert eines jeden Probanden wurde für die ersten sechs Fragen errechnet und mit den anderen Mittelwerten zusammengefasst. Daraufhin wurde ein Gesamtmittelwert der emotionalen Involviertheit bestimmt. Das Ergebnis der Auswertung liegt bei 3,90.

Die weiteren sechs Aussagen handeln über die Manipulation der Probanden. Hier konnte eine Spannweite von 2,1 bis 4,16 festgestellt werden. Wie bei der Auswertung der ersten Subfacette wurde ein Mittelwert aller Probanden bestimmt. Anschließend wurde der Gesamtmittelwert dieser Auswertung erfasst. Das Ergebnis dieser Auswertung liegt bei 3,17.

Die letzte Frage in dem Fragebogen bezog sich auf die wahrgenommene Immersion währen der Anwendung. Die durchschnittliche empfundene Immersion der Probanden lag bei 7,14. Alle Tester haben angegeben, dass sie sich ihrer realen Umgebung während der VR-Anwendung immer noch bewusst waren, jedoch war sich keiner der Kandidaten bewusst gewesen, was um sie herum während der Anwendung geschah.

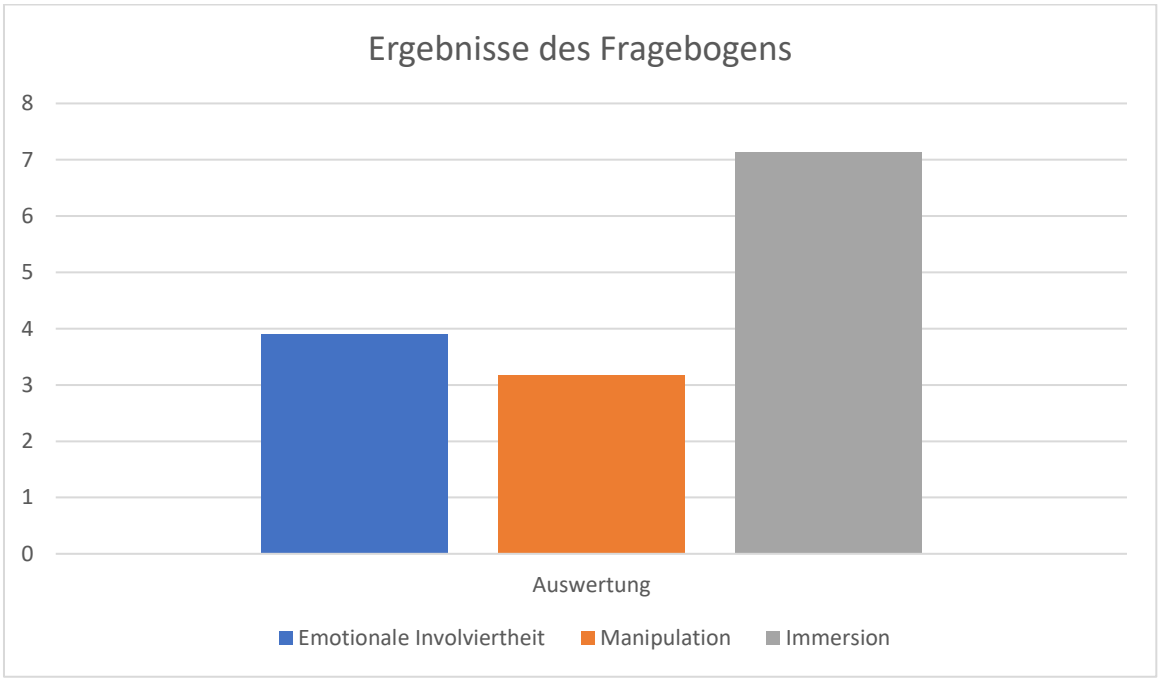


Tabelle 1: Veranschaulichung der Ergebnisse in einem Balkendiagramm

5.4 Diskussion

Zur besseren Einordnung der Befunde sollte angemerkt werden, dass sich aufgrund der kleinen Stichprobengröße nur Tendenzen ableiten lassen.
Die durchschnittliche emotionale Involviertheit mit 3,90 Punkten zeigt, dass die Probanden eine emotionale Reaktion und starke Empfindungen während der Anwendung erlebt haben. Im Gespräch nach der Anwendung gaben die Testpersonen zudem an,

dass die Emotionen, die sie während des Spiels empfanden, hauptsächlich negativ besetzt waren. Besonders im Fokus standen dabei Zeichen der Nervosität und des Stresses. Teilweise wurde sogar berichtet, dass die Anwendung Angst erzeugt hat. Auch die Aufzeichnungen des Versuchsleiters konnten Körperzuckungen sowie verbale Ausbrüche des Entsetzens festhalten. Daraus lässt sich schlussfolgern, dass das Spiel in der Lage ist, den emotionalen Zustand der Probanden zu einem gewissen Grad zu manipulieren. Diese Aussagen unterstützen den Versuch des Programmierers, eine immersive VR-Anwendung kreiert zu haben, die sich ausschließlich negativen Reizen und Impulsen bedient.

Trotz der Berichte der Angst und Nervosität gab ein Teil der Befragten widererwartend an, die Bilder und Graphiken gemocht zu haben. Die Modelle und Objekte hatten die primäre Funktion die Probanden abzuschrecken, weshalb diese Äußerungen dazu im Konflikt stehen. Auf Nachfrage im abschließenden Gespräch, gaben die Kandidaten an, die Situation selbst, nicht als positiv empfunden zu haben. Die Illustrationen in Kombination mit den verwendeten Animationen und Geräuschen wurden jedoch als ansprechend wahrgenommen. Des Weiteren ist anzumerken, dass körperliche Reaktionen, wie Zucken oder reflexartige Handbewegungen zu erkennen waren. Jedoch hat keiner der Probanden die Leertaste vollständig losgelassen.

Der zweite Teil des Fragebogens beschäftigt sich mit der Manipulation des Probanden. Diese Facette von Immersion wurde mit einem Mittelwert von 3,17 Punkten als mittelmäßig umgesetzt bewertet. Daraus lässt sich folgern, dass die wahrgenommene Präsenz in der realen Welt für die Probanden immer noch vorhanden war. Alle Testpersonen stimmten dem Item „*Ich war mir meiner realen Umgebung bewusst.*“ zu. Auch dem Item *„Ich hatte nicht das Gefühl in der realen Welt zu sein während ich die Anwendung testete.“* wurde entweder neutral oder zustimmend gewertet. Insgesamt zeichneten die Testwerten somit das Bild, dass die Anwendung nicht ausreichte, um die Personen zu täuschen und vollständig in eine andere virtuelle Welt zu versetzten. Im abschließenden Gespräch begründeten die Probanden diese Ergebnisse mit der Tatsache, dass der Versuchsleiter während der Anwendung mit ihnen kommunizieren musste. Dieser Eingriff störte die aufgebaute Illusion, weshalb die virtuelle Welt als nicht real wahrgenommen wurde. Der Versuchsleiter musste den Probanden öfters mitteilen sich zu dem zersprungenem Fenster umzudrehen, damit die Anwendung weitergeht. Ein möglicher Verbesserungsvorschlag für die Anwendung ist demnach eine Optimierung dieser Spielszene. Es zeigt sich, dass das Zerspringen des Fensters hinter dem Spieler als Impuls für eine Kopfbewegung nicht ausreicht. Durch das Ausbleiben der Drehung wird der Spielablauf gestört und läuft nicht weiter, weshalb der Versuchsleiter jeweils von außen eingreifen musste. Zukünftig sollten daher weitere und möglicherweise auch stärkere Impulse gesetzt werden, um eine Kopfdrehung der Probanden zu erreichen. Alternativ könnte notwendige Anweisungen bei Nichtzeigen der vorgesehenen Reaktion in das Spiel integriert werden. Auch diese Modifikation der Anwendung könnte eine Störung der Immersion von außerhalb verhindern.

Die letzte Frage bezog sich auf die empfundenen Immersion. Dafür wurde ein Mittelwert von 7,14 ermittelt, welcher als sehr hoch eingeschätzt werden kann. Die Probanden waren davon überzeugt, dass die Anwendung sehr gelungene negative Emotionen und angstbesetzte Reaktionen provozierte. Alle Versuchspersonen gaben im ab-

schließenden Gespräch an sich unbehaglich und unwohl gefühlt zu haben. Dieser Effekt trat trotz dem Wissen, dass sie sich in einer sicheren Umgebung zum Zeitpunkt der Anwendung befanden, auf. Zudem wurde der Einsatz von Musik und Tönen von den Probanden mehrfach als zentrales Element herausgestellt. Eine Versuchsperson gab an: „*Die Töne und Geräusche waren das schlimmste meiner Meinung nach. Ohne diese wäre alles nur halb so schrecklich gewesen.*“.

6 Ausblick

Die erste Forschungsfrage, die diese Bachelorarbeit untersuchte, war, ob eine immersive VR-Anwendung hauptsächlich aus negativen Elementen bestehen kann. Die Ergebnisse dieser Arbeit haben deutlich gezeigt, dass die geschaffene VR-Erfahrung als immersiv wahrgenommen wurde. Alle Probanden des Experiments, berichteten von einem hohen bis sehr hohen Grad an Empfundener Immersion. Manche Probanden gaben an, dass sie Emotionen wie Angst als auch Symptome von Stress oder Schock während der VR-Anwendung an sich beobachten konnten. Dabei spielten die eingesetzten Schreckmomente sowie die Implementierung von Tönen und Geräuschen eine entscheidende Rolle. Zukünftige Studien sollten demnach darauf achten, möglichst viele passende Töne und Geräusche in der Anwendung zu integrieren. Auch der Einsatz zusätzlicher Elemente würde die Dauer und demnach auch die Möglichkeit einer höher empfundenen Immersion steigern. Darüber hinaus gaben die Probanden nach der Anwendung an, dass zusätzliche externe Reize wie beispielsweise ein Kribbeln auf den Armen, während die Spinnen die virtuellen Arme hochlaufen, eine Steigerung der empfunden Immersion bewirken könnte. Dieser Idee folgend, könnten zukünftige Experimente nicht nur interne Reize betrachten, sondern auch externe Reize in ihre Anwendung aufnehmen und somit eine neue, attraktive Möglichkeit der Immersion erreichen.

Die zweite Frage, die diese Bachelorarbeit behandelt, war ob die empfunden Immersion ausreicht, um die Probanden so zu täuschen und zu manipulieren, dass sie die virtuelle Welt als die Reale wahrnehmen würden. Die Ergebnisse aus vorgenommenen Experiment zeigten, dass dies nicht möglich war. Die Probanden konnten jederzeit zwischen reeller und virtueller Welt bewusst unterscheiden. Ein entscheidender Grund dafür war, dass der Versuchsleiter teilweise korrektiv eingreifen musste, damit die Probanden mit der Anwendung fortfahren konnten. Unter dieser Berücksichtigung sollte bei einer Wiederholung des Experiments vorerst neue Impulse implementiert werden, sodass die Spieler die Anwendung ohne Eingreifen des Leiters beenden können.

Somit lässt sich zusammenfassend sagen, dass die Erschaffung einer immersiven virtuellen Erfahrung bis zu einem gewissen Grad geglückt ist. Allerdings war diese Immersion bisher nicht ausreichend, um die Probanden davon zu überzeigen, die reale Welt um sie herum vollständig zu vergessen.

7 Literaturverzeichnis

Brandt, M. (2019). Der deutsche Markt für digitale Spiele wächst. Retrieved from statista website: https://de.statista.com/infografik/15094/umsatz-auf-dem-deutschen-markt-fuer-computer-und-videospiele/

Brown, E., & Cairns, P. (2004). *A grounded investigation of game immersion*. 1297. https://doi.org/10.1145/985921.986048

Haywood, N., & Cairns, P. (2006). Engagement with an interactive museum exhibit. *People and Computers XIX - The Bigger Picture, Proceedings of HCI 2005*.

Jennett, C., Cox, A. L., Cairns, P., Dhoparee, S., Epps, A., Tijs, T., & Walton, A. (2008). *Measuring and Defining the Experience of Immersion in Games. International Journal of Human Computer Studies 66(9)*. 641–661. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.157.4129&rep=rep1&type=pdf

Meehan, M., Razzaque, S., Whitton, M. C., & Brooks, F. P. (2003). Effect of latency on presence in stressful virtual environments. *Proceedings - IEEE Virtual Reality*, *2003-Janua*, 141–148. https://doi.org/10.1109/VR.2003.1191132

Müller, S. (2016). *Virtuelle Realität und Augmented Reality*. Koblenz.

Pössel, P., & Hautzinger, M. (2002). Spinnen-Angst-Fragebogen (SAF), Validierung der deutschen Version des “Fear of Spiders Questionnaire” (FSQ). *Zeitschrift Fur Klinische Psychologie, Psychiatrie Und Psychotherapie*, *50*(2), 207–218.

Robillard, G., Bouchard, S., Fournier, T., & Renaud, P. (2003). Anxiety and Presence during VR Immersion: A Comparative Study of the Reactions of Phobic and Non-phobic Participants in Therapeutic Virtual Environments Derived from Computer Games. *CyberPsychology & Behavior*, *6*(5), 467–476. https://doi.org/10.1089/109493103769710497

Sanders, T., & Cairns, P. (2010). Time perception , immersion and music in videogames Categories and Subject Descriptors. *ACM Press*, 160–167.

Schneiderman, N., & McCabe, P. M. (2013). Biobehavioral Response to Stress in Females: Tend-and-Befriend, Not Fight-or-Flight. *Stress and Coping*, *107*(3), 13–61. https://doi.org/10.4324/9780203781722

Slater, M. (2003). A Note on Presence Terminology. *Emotion*, 1–5.

Solomon, B. (2014). Facebook Buys Oculus, Virtual Reality Gaming Startup, For $2 Billion. Retrieved from Forbes website: https://www.forbes.com/sites/briansolomon/2014/03/25/facebook-buys-oculus-virtual-reality-gaming-startup-for-2-billion/#596594522498

Steuer, J. (1992). Defining virtual reality: dimensions determining telepresence, Communication in the age of virtual reality. *Journal of Communication*, *42*(4), 73–93.

Technologies, U. (2019a). Console Window. Retrieved from https://docs.unity3d.com/Manual/Console.html

Technologies, U. (2019b). Learning the interface. Retrieved from Unity

Documentation website: https://docs.unity3d.com/Manual/LearningtheInterface.html

Technologies, U. (2019c). Physics.Raycast. Retrieved from Unity Documentation website: https://docs.unity3d.com/ScriptReference/Physics.Raycast.html

Technologies, U. (2019d). Scripting. Retrieved from https://docs.unity3d.com/Manual/ScriptingSection.html

Technologies, U. (2019e). The world´s leading real-time creation platform. Retrieved from https://unity3d.com/de/unity

Technologies, U. (2019f). Using the Asset Store. Retrieved from https://docs.unity3d.com/Manual/AssetStore.html

Anhang

Fragebogen benutzt für das Experiment dieser Bachelorarbeit

Ihre persönliche Erfahrung mit dem Spiel
Bitte bewerten Sie inwieweit Sie mit den unten aufgeführten Aussagen einverstanden sind.
SW = Starker Widerspruch; W = Widerspruch; N = Neutral; Z = Zustimmung;
SZ = Starke Zustimmung

Ich hatte keinerlei emotionale Bindung zu dem Spiel.
SW W N Z SZ

Es interessierte mich nicht herauszufinden was als nächstes in dem Spiel passiert.
SW W N Z SZ

Ich mochte die graphischen Objekte und Bilder nicht in dieser Anwendung.
SW W N Z SZ

Mir haben die Bilder und Graphiken gefallen.
SW W N Z SZ

Diese Anwendung zu testen hat keinen Spaß gemacht.
SW W N Z SZ

Ich habe es gemocht diese Anwendung zu testen.
SW W N Z SZ

Ich war mir meiner realen Umgebung bewusst.
SW W N Z SZ

Mir war unbewusst, was um mich herum in der realen Welt geschieht.
SW W N Z SZ

Zu dieser Zeit war die Anwendung meine einzige Sorge.
SW W N Z SZ

Alltägliche Gedanken und Bedenken waren immer noch präsent für mich.
SW W N Z SZ

Ich hatte nicht einmal das Bedürfnis, mit der Anwendung aufzuhören.
SW W N Z SZ

Ich hatte nicht das Gefühl in der realen Welt zu sein während ich die Anwendung testete.
SW W N Z SZ

Wie hoch war die Immersion für dich? (10 = sehr immersiv; 0 = überhaupt nicht immersiv)

0 1 2 3 4 5 6 7 8 9 10