Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelorarbeit

Dimitri Meier

Entwicklung und Vergleich unterschiedlicher Navigationskonzepte in einer Virtual Reality Umgebung

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Dimitri Meier

Entwicklung und Vergleich unterschiedlicher Navigationskonzepte in einer Virtual Reality Umgebung

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung

im Studiengang Bachelor of Science Angewandte Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Birgit Wendholt
Zweitgutachter: Prof. Dr.-Ing. Martin Hübner

Eingereicht am: 13. August 2018

Dimitri Meier

Thema der Arbeit
Entwicklung und Vergleich unterschiedlicher Navigationskonzepte in einer Virtual Reality Umgebung

Stichworte
Navigation, Virtuelle-Umgebung, 3D-Modell, HMD, Virtuelle Realität, Magic-Locomotion, Motion-Triggered

Kurzzusammenfassung
Diese Arbeit fokussiert sich auf die Untersuchung und Entwicklung der Navigationskonzepte in einer virtuellen Realität. Es soll untersucht werden, welche der Bewegungsformen zum Überqueren größerer Strecken im virtuellen Raum, unter dem Aspekt der Bedienbarkeit und Berücksichtigung der Cybersickness, am besten geeignet sind. Die Bewegungsformen werden anhand eines realen Flugzeugmodells für die VR Umgebung umgesetzt und in einer Evaluationsphase die entwickelten Lösungen von einer kleinen Benutzergruppe getestet. Den Abschluss dieser Arbeit bildet die Auswertung der Ergebnisse, in der die einzelnen Bewegungsformen auf die Akzeptanz überprüft werden.
Dimitri Meier

Title of the paper
Development and comparison between different concepts for navigation in a virtual reality environment

Keywords
navigation, virtual enviroment, 3D model, HMD, virtual reality, magic locomotion, motion triggered

Abstract
This thesis focuses on the investigation and development of navigation concepts in a virtual reality. It's to be investigated wich kind of motion is the best suitable for crossing of larger distances in the virtual space in terms of usability and cybersickness. The forms of motion are implemented on the basis of a real aircraft model for the VR environment and in an evaluation phase, the developed solutions are tested by a small user group. The conclusion of this work is the evaluation of the results, in which the individual forms of movement are checked for acceptance.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

1.1 Motivation

Durch die stetige Entwicklung der Virtual Reality (VR) sind populäre Anwendungsbereiche wie z.B. Spiele und Filme entstanden. Mit dem aktuellen Fortschritt der VR-Hardware wird dem Nutzer die Möglichkeit geboten, in eine fiktiv erschaffene Welt einzutauchen. Der Nutzer wird nicht mehr durch die klassischen Bildschirme einer Desktop-Umgebung eingeschränkt und kann stattdessen mit natürlichen Bewegungen die Sicht und Bewegung steuern. In 360°-Filmen übernimmt der Zuschauer im Grunde die Kameraführung und kann somit den aktiven Teil der Filmwelt mitgestalten.
Abseits der Unterhaltungsmedien wäre der Einsatz von VR für die Simulation von Gebäuden bzw. größeren Räumen denkbar, bevor man anfängt diese zu planen. Hierdurch ließen sich einerseits hohe Kosten in der Planung minimieren und andererseits Räume bzw. Gebäude untersuchen und ggf. explorieren. Mit dem Einsatz von VR ließen sich auch Szenarien, wie z.B. Evakuierungen, Simulationen von Rauchentwicklung oder Akustik testen und die Planung darauf abstimmen.
Der aktuelle Stand der Technik ermöglicht es, VR-Hardware mit Nutzertracking vergleichbar günstig anzubieten. Damit sind auch kleine und mittelständige Unternehmen in der Lage, VR-Lösungen für Großunternehmen anzubieten, die vorher nur mit großräumigen Installationen und einer teuren Hardware möglich waren. Der Nutzen von VR gewinnt auch in der Vertriebsunterstützung eine wichtige Rolle. Beispielsweise können Unternehmen ihren Kunden mit einer interaktiven VR-Applikation veranschaulichen, wie ihre Produkte im Alltag verwendet werden. Dadurch können sich Kunden viel eingehender mit dem Produkt beschäftigen und sind eher zum Kauf bereit, als gegenüber einem herkömmlichen Werbevideo.
Der Einsatz, einer vergleichsweise günstigen VR-Hardware, hat aber auch gewisse Einschränkungen. Die wesentliche Einschränkung ist der Aktionsradius, d.h. der Nutzer kann sich nur auf einen relativ kleinen physischen Raum bewegen. In Anwendungsdomänen wie z.B. Museen, Flugzeugen oder auch großen Räumen müssen Überlegungen getroffen werden, wie man diese vergleichbar großen Räume mit einem begrenzten physischen Raum vollständig erkunden

kann. Um den Konflikt zwischen der virtuellen großen Fläche und der vergleichbar kleinen Aktionsfläche überbrücken zu können, werden Navigationskonzepte benötigt.

1.2 Zielsetzung

Ziel dieser Arbeit ist, am Beispiel eines realen Flugzeugmodells, drei Navigationskonzepte in VR zu entwickeln, die eine intuitive und komfortable Bedienung ermöglichen, um größere Entfernungen in der virtuellen Umgebung zu überbrücken. Dabei soll es möglich sein, trotz der physikalischen Einschränkung im Aktionsradius der herkömmlichen Konsumenten-Hardware, die Navigation und Exploration in einer virtuellen Flugzeugkabine effizient durchzuführen. Es soll eine vollständige und funktionierende Lösung entwickelt werden, die die verschiedenen Navigationskonzepte umsetzt. In einer abschließenden Evaluationsphase sollen die entwickelten Lösungen von einer kleinen Benutzergruppe getestet und die objektiven Bewertungen anhand von messbaren Größen erfasst werden. Die gesammelten Ergebnisse sollen zeigen, wie gut sich die entwickelten Navigationskonzepte für die Navigation und Exploration einer großen virtuellen Umgebung eignen. Als Software- und Hardware-Umgebung soll die HTC-Vive in Kombination mit Unity-3D verwendet werden.

1.3 Gliederung

In Kapitel 2 werden die theoretischen Grundlagen zur Navigation in VR gelegt und anhand vergleichbarer Arbeiten die unterschiedlichen Navigationskonzepte vorgestellt. Anschließend wird die Auswahl der für diese Arbeit relevanten Navigationskonzepte auf Grundlage einer Bewertungstabelle getroffen.
Die zu entwickelnden Navigationsformen werden in Kapitel 3 ausführlich als Katalog von funktionalen und nicht funktionalen Anforderungen formuliert.
Kapitel 4 beschreibt den Systementwurf, sowie die detaillierte Umsetzung der einzelnen Navigationskonzepte unter Berücksichtigung der Ergebnisse aus Kapitel 2 und 3. Die darauf folgende Evaluierung in Kapitel 5 wird zeigen, dass keine der entwickelten Navigationsformen allein für die Fortbewegung im virtuellen Raum ausreicht, sondern eine Kombination aus einer präzisen und langsamen Bewegung mit einer intuitiven und schnellen Fortbewegung am besten geeignet ist. In Kapitel 6 werden die Ergebnisse dieser Arbeit zusammengefasst, sowie auf mögliche Erweiterungen der entwickelten VR-Anwendung eingegangen. Abschließend wird anhand eines Anwendungsfalls die Übertragbarkeit der entwickelten Konzepte gezeigt.

2 Grundlagen und vergleichbare Arbeiten

In diesem Kapitel werden zunächst die grundlegenden Probleme und Lösungen der Navigation in Virtual Reality erläutert (Abschnitt 2.1). Die Abschnitte 2.1.1 - 2.1.6 diskutieren diese Lösungen im Detail, die im Abschnitt 2.1.7 für die Eignung der Verfahren in dieser Arbeit untersucht werden. Da Navigationsanwendungen in VR auch klassische Benutzer Schnittstellen beinhalten, stellt Abschnitt 2.2 die unterschiedlichen Interface-Elemente in 3D-Anwendungen vor. Abschnitt 2.3 führt den Szenengraphen ein, die klassische Datenstruktur für die Umsetzung von VR-Anwendungen. Im letzten Abschnitt 2.4, werden die wichtigsten Inhalte der eingeführten Themen zusammengefasst.

2.1 Navigation in Virtual Reality

Der Begriff Virtual Reality (VR) bezeichnet eine vom Computer generierte Umgebung, die entweder über Großbildleinwände in speziellen Räumen (Cave Automatic Virtual Environment, CAVE) oder mittels Head-Mounted-Display (HMD) übertragen werden kann [Bendel (2018)]. Mithilfe der geeigneten Ein- und Ausgabegeräte werden die natürlichen Interaktionsmöglichkeiten in VR auch als Mensch-Maschine-Schnittstelle bezeichnet. Im Vergleich zu den traditionellen Benutzerschnittstellen (engl. User Interface, UI) bietet die VR eine natürliche und intuitive Interaktion mit der 3D-Welt [Dörner u. a. (2014)].

In der VR-Welt hat die Navigation als interaktive Aufgabe eine zentrale Bedeutung. Um die notwendige Eigenschaft „Immersion“ in virtuellen Welten beizubehalten, muss der Nutzer sich möglichst einfach und natürlich in der virtuellen Welt bewegen können [Dörner u. a. (2014)]. Hierfür sind VR-Systeme mit integrierten Trackingbereich bereits in der Lage, reale Bewegungen vom Nutzer innerhalb des Trackingbereich auf die virtuelle Umgebung abzubilden. Bei der Navigation müssen zwei grundlegende Probleme gelöst werden. Das eine Problem ist das Verhältnis zwischen der realen Spielfläche und der virtuellen Umgebung. Virtuelle Umgebungen können potentiell unendlich groß sein und lassen sich dadurch mit der realen Spielfläche nicht vollständig erkunden. Das andere Problem betrifft die Cybersickness, die bei einer Navigationsart in VR entstehen kann. Beispielsweise kann eine neu entwickelte

Bewegungsart das Problem bei der Navigation in VR-Umgebungen zwar lösen, wird aber vom Nutzer eventuell nicht akzeptiert, da eine Cybersickness bei der Verwendung auftritt.

Cybersickness wird Bewegungskrankheit oder auch Simulationskrankheit genannt, die häufig bei Bewegungen in der virtuellen Welt auftritt. Diese Krankheit entsteht dann, wenn die visuell vorgetäuschte Bewegung nicht exakt mit der Bewegung des eigenen Körper übereinstimmt. Diese widersprüchliche Information kann bei einzelnen Nutzern unter anderem zu Übelkeit, erhöhtem Speichelfluss, Benommenheit, Schwindelgefühlen und auch Erbrechen führen [Dörner u. a. (2014)]. Reaktionen dieser Art können bei der Verwendung von Head-Mounted-Displays (HMD) in Kombination mit einem Tracking-Systems auftreten, welche die Bildübertragung zu der Kopfbewegung in Echtzeit verändern. Dies geschieht häufig, wenn die Bilder asynchron zur Bewegung übertragen werden oder auch durch eine zu hohe Latenz zwischen der Verarbeitung von Bewegungsinformationen im System und der Darstellung im HMD. Auch die Darstellungsqualität vom HMD kann zu den aufgezählten Symptomen führen, wenn die übertragenen Bilder bei der Darstellung im HMD verzerrt oder auch unscharf dargestellt werden [Dörner u. a. (2014)].
Es soll möglich sein, die Symptome reduzieren zu können, indem die Diskrepanzen zwischen der simulierten und der realen Bewegung möglichst gering gehalten wird. Somit spielt auch die geringe Latenz eine entscheidende Rolle, um die Übertragung der Bild und Sensordaten möglichst synchron halten zu können [Dörner u. a. (2014)].
Im Folgenden werden die sechs unterschiedlichen Kategorien für die Fortbewegung in VR vorgestellt.

2.1.1 Magic-Locomotion

Diese Bewegungstechnik soll dem Nutzer eine einfache aber unnatürliche Bewegungsform für die Fortbewegung in VR-Umgebungen anbieten [Gieselmann (2017)]. Eine der bekanntesten Methoden ist die „Point & Teleport“ Bewegung, die es dem Nutzer erlaubt an eine beliebige Position der virtuellen Umgebung zu teleportieren [Bozgeyikli u. a. (2016b)]. Die Interaktion wird im Grunde in zwei Stufen unterteilt:

- **Zielbestimmung**
  Der Nutzer verwendet das Eingabegerät wie z.B. Maus, Joystick oder eine Geste [Bozgeyikli u. a. (2016a)], um sein Ziel in der virtuellen Umgebung zu bestimmen. Hierfür wird ein Strahl verwendet, vergleichbar mit einem Laserpointer, um den getroffenen Bereich als Ziel zu identifizieren.

- **Positionsveränderung**
  Hierbei wird die virtuelle Nutzerposition auf das zuvor selektierte Ziel geändert.

Die Ergebnisse eines Experiments von Evren Bozgeyikli [Bozgeyikli u. a. (2016b)] haben gezeigt, dass sich Point & Teleport nicht nur für schnelle Fortbewegungen in einer virtuellen Umgebung eignet, sondern auch den Nutzer ein unterhaltsames Erlebnis bietet. Zwar bietet diese Bewegungstechnik keine erhebliche Immersion, ermöglicht aber den virtuellen Raum schnell zu erkunden, ohne Cybersickness zu verursachen.

2.1.2 Artificial-Locomotion

Bei Artificial-Locomotion handelt es sich um eine klassische Fortbewegung mit einem Joystick, die überwiegend in 3D-Computerspielen verwendet wird. Die Fortbewegung in virtuellen Welten lässt sich mit der Steuerung eines Fahrzeugs vergleichen [Dörner u. a. (2014)]. Der Benutzer kann sich in vier Himmelsrichtungen bewegen und die Blickrichtung entweder mit einem Joystick steuern oder mit dem Kopf, wenn ein trackingbasiertes VR-System verwendet wird [Bozgeyikli u. a. (2016a)]. Der Vorteil dieser Steuerungs-Technik ist, dass zum einem eine hohe Genauigkeit und Präzision gegeben ist und zum anderem eine sehr geringe Latenz bei der Übertragung entsteht [Nabiyouni u. a. (2015)].
Da diese Bewegungsart das Körpergefühl bei der Bewegung nicht unterstützt, besteht eine erhöhte Gefahr auf Cybersickness. Um dem entgegenzuwirken, sollten grundsätzlich schnelle Translations- und Rotations-Bewegungen vermieden werden und stattdessen mit einem konstanten Bewegungstempo umgesetzt werden [Gieselmann (2017)]. Wenn mit unterschiedlichen Beschleunigungsstufen gearbeitet wird, empfiehlt es sich das Field of View (FOV) einzuschränken, um die Cybersickness zu verringern - dies hat eine Studie „Combating VR Sickness through Subtle Dynamic Field-Of-View Modification“ belegt [Fernandes und Feiner (2016)].

2.1.3 Reoriented-World

Anders als bei Artificial-Locomotion, wird nicht der Nutzer, sondern die virtuelle Welt bewegt. Ziel dieser Bewegungsform ist, die virtuelle Umgebung neu zu orientieren bzw. zu bewegen, ohne dass der Nutzer diese als Eigenbewegung wahrnimmt [Gieselmann (2017)]. Es gibt unterschiedliche Ansätze diese Technik in virtuellen Umgebungen umzusetzen. Am einfachsten lässt sich die Steuerung mit Artificial-Locomotion umsetzen, indem z.B. ein Joystick für die Bewegung der virtuellen Welt verwendet wird. In dem Projekt **The Holosphere** von Tomáš Mariančík wird die Bewegung mit Hilfe einer Leap Motion gesteuert, indem der Nutzer seine Hand zu einer Faust formiert, um die virtuelle Umgebung neu auszurichten oder zu bewegen

[James (2015)]. Damit der Nutzer die virtuelle Translation und Rotation nicht als Eigenbewegung wahrnimmt, wird eine virtuelle Sphäre (Holosphere) um den Nutzer projiziert. Diese virtuelle Sphäre ähnelt einem Gitternetz (siehe Abbildung 2.1) und ermöglicht dem Nutzer die virtuelle Umgebung durch die Sphäre zu sehen, wenn auch nur sehr unscharf.

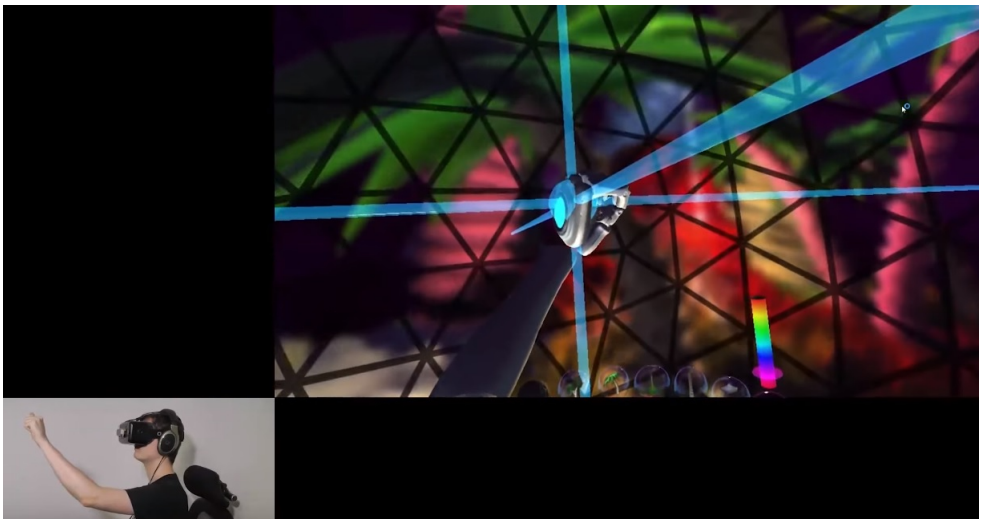


Abbildung 2.1: Die virtuelle Sphäre wird bei der Bewegung und Rotation der virtuellen Umgebung um den Nutzer projiziert, um eine Umgebungsbewegung zu imitieren. Quelle: James (2015)

2.1.4 Redirected-Walking

Die natürlichste Art sich in einem virtuellen Raum zu bewegen, ist das virtuelle Gehen (engl. Walking). Diese Bewegungstechnik unterstützt das Körpergefühl bei der Bewegung, das vom Gleichgewichtsorgan des Menschen geliefert wird [Preim und Dachselt (2015)].
Am einfachsten lässt sich die reale Bewegung auf die virtuelle übertragen, indem eine eins zu eins Abbildung verwendet wird. Da der Interaktionsraum eines Trackingsystems im realen Raum begrenzt ist, lässt sich ein potentiell unendlich großer virtueller Raum nicht vollständig erkunden. Die Technik von Redirected-Walking versucht den Nutzer zu der am weitesten entfernten Grenze des Trackingbereichs umzulenken, ohne das die Manipulation vom Nutzer wahrgenommen wird. Hierfür gibt es unterschiedliche Gains, um die Translations- oder Rotations-Bewegung zu manipulieren. Die menschliche Wahrnehmung von Bewegung und Rotation, wird auf Grundlage von unterschiedlichen Reizen ermittelt. Hierfür dienen Gleich-

gewichtssystem, Propriozeption [1], sowie visuelle und akustische Reize, um unterscheiden zu können, ob sich der eigene Körper oder die Objekte in der Umgebung bewegen [Razzaque u. a. (2001)]. Normalerweise arbeiten diese Sinne eng miteinander, um die Bewegung wahrzunehmen, doch kann es vorkommen, dass der visuelle Sinn bei der Bewegungswahrnehmung dominiert, wenn eine Diskrepanz zu den anderen Sinnen auftritt. Ein bekanntes Beispiel aus dem Alltag - Man sitzt in einen stehenden Zug und beobachtet auf dem Nachbargleis, wie sich ein Zug in Bewegung setzt. So kann man häufig auf den ersten Blick nicht unterscheiden, ob man selbst bewegt wird oder der benachbarte Zug [WikiBewegungssehen (2018)]. Redirected-Walking nutzt die visuelle Dominanz, um die Wahrnehmung der physischen Körperbewegungen zu manipulieren. Eine Manipulation der Bewegung bleibt grundsätzlich unbemerkt, solange die Bewegung als Eigenbewegung wahrgenommen wird [Riecke u. a. (2015)]. Dies kann erreicht werden, indem die Reize zu der Bewegung konsistent empfunden werden, z.B. wenn der Nutzer eine Rotation um 90° ausübt und die virtuelle Rotation mit 120° übertragen wird, so müssten die hörbaren Klänge aus der virtuellen Umgebung auch um 120° übertragen werden [Larsson u. a. (2004)]. Wie stark die Wahrnehmung der Bewegung manipuliert wird, hängt vom Gain ab. Diese sind wie folgt aufgebaut:

- **Translations-Gain**
  Dieses Gain skaliert die reale Bewegung um eine vergleichbar größere virtuelle Umgebung erkunden zu können. Es können große Schwankungen entstehen, wenn der Skalierungsfaktor zu hoch gesetzt wurde, was allein durch die Ungenauigkeit des Tracking-Systems entstehen kann [Dörner u. a. (2014)].

- **Rotations-Gain**
  Anders als beim Translations-Gain, skaliert das Rotations-Gain die Rotation des Kopfes. Da die Kopf-Rotation sehr häufig bei einem natürlichen Bewegungsablauf vorkommt, kann dieses Gain fast immer angewendet werden. Die Manipulation der Rotation ist vom Nutzer viel schwieriger zu erkennen, als bei anderen Gains, da bei einer Rotation des Kopfes das Gleichgewichtssystem angeregt wird [Razzaque u. a. (2001)].

- **Curvation-Gain**
  Das Curvation-Gain kombiniert die Eigenschaften von Rotation- und Translation-Gain, die auf die reale Bewegung des Nutzers angewandt werden. Anders als bei den zuvor genannten Gains, wird die reale Bewegung nicht verstärkt, sondern eine zusätzliche Rotation auf die virtuelle Umgebung durchgeführt, wenn sich der Nutzer im realen

[1] Der Begriff Propriozeption bezeichnet die Wahrnehmung von Körperbewegungen und Körperlage im Raum. Quelle: Wikipedia

Raum geradeaus bewegt. Bei kleinen Abweichungen zwischen der realen und virtuellen Rotation, versucht der Mensch diese unbewusst auszugleichen und bewegt sich im Kreis, während er im virtuellen Raum geradeaus läuft [Walker (2013)].

- **Displacement-Gain**
  Im Gegensatz zu Curvature-Gain, bildet das Displacement-Gain die reale Rotation des Nutzer auf die virtuelle Translation ab. Es gibt Anwendungsfälle, bei denen die Veränderung der virtuellen Position mit dem Kopf bzw. Körper benötigt wird, ohne die physikalische Position zu verändern [Steinicke u. a. (2009)].

- **Time-Dependent-Gain**
  Das Time-Dependent-Gain regelt pro vergangene Zeiteinheit die Manipulation der realen Bewegung, so dass z.B. das Rotations-Gain pro vergangene Sekunde eine virtuelle Rotation mit wenigen Grad um den Nutzer herum dreht, obwohl sich dieser nicht bewegt. Geringe Manipulationen werden vom Nutzer unbemerkt ausgeglichen, so dass sich dieser auf der Stelle dreht [Steinicke u. a. (2009)].

Der entscheidende Nachteil dieser Technik ist der enorme Platzbedarf, der für die Umsetzung benötigt wird. Dieser ist entscheidend für eine unauffällige Manipulation der Bewegung.

2.1.5 Motion-Triggered

Die Motion-Triggered Technik beschreibt die Fortbewegung in virtuellen Welten, die vom physikalischen Körper aktiviert wird. Zum Beispiel kann sich der Nutzer, durch das Wippen des Kopfes, eine natürliche Bewegung imitieren und die Fortbewegung bei konstanter Geschwindigkeit in der virtuellen Umgebung steuern, ohne den realen Körper zu bewegen [Jehan (2017)].
Alternativ dazu wurde in einer Studie die Fortbewegung mit dem menschlichen Oberkörper untersucht, indem sich die Neigung des Oberkörpers auf die Geschwindigkeit auswirkt. Es hat sich ergeben, dass eine konstante Geschwindigkeit (d.h. das Halten der Vorwärtsneigung des Oberkörpers) den Effekt auf die Selbstwahrnehmung erhöht und somit auch Anzeichen einer Cybersickness minimiert [Kruijff u. a. (2015)].
Grundsätzlich kann sich die Motion-Triggered Methode für viele Nutzer intuitiver und natürlicher anfühlen als andere Navigationsformen, schränkt jedoch die Funktionalität der Controller bzw. des HMD mit der Steuerung der Fortbewegung ein [Jehan (2017)].

2.1.6 Surrogate-Vehicle

Surrogate-Vehicle beschreibt keine konkrete Bewegungsart, sondern eine Hilfestellung für die Fortbewegung, indem ein virtuelles Cockpit als Referenzrahmen eingeblendet wird [Gieselmann (2017)], um eine vergleichbare Fortbewegung aus der Realität zu simulieren. Oftmals wird für die Steuerung das Artificial-Locomotion verwendet, vergleichbar mit einem Autorennspiel, welches mit einem Joystick oder Tastatur gesteuert wird.
Es ist aber auch möglich die Fortbewegung in einem virtuellen Fahrzeug umzusetzen, dies hat Dimitrij Schäfer in seiner Bachelorthesis umgesetzt [Schäfer (2017)]. Das Ziel dieser Hilfestellung ist, dass FOV mit dem virtuellen Cockpit einzuschränken, um die Cybersickness zu reduzieren [Förtsch (2016)].

2.1.7 Bewertung der Navigationsformen

Zuletzt soll anhand der Bewertungen ermittelt werden, welche der unterschiedlichen Navigationsformen für die Arbeit geeignet sind. Hierfür wurden die wichtigsten Eigenschaften mit einer Punktzahl bewertet. Zu den Eigenschaften zählen:

- **Entfernung** : Die virtuelle Entfernung, die in der 3D-Welt zurück gelegt werden kann.
- **Präzision** : Wie genau wird die Bewegung gesteuert.
- **Cybersickness** : Wie anfällig ist diese Bewegungsart auf die Cybersickness.
- **Platzbedarf** : Wie hoch ist der reale Platzbedarf, um diese Bewegungsform umsetzen zu können.
- **Anwendungsbereich** : In welchem Anwendungsbereich wird die Navigationsform üblicherweise verwendet.

Die Bewertungsskala geht von 1 (minimale Punktzahl) bis 3 (maximale Punktzahl). Da es sich im Anwendungsbereich um eine räumliche Anwendungsdomäne mit größerer Ausdehnung handelt, wird zum einem ein Verfahren benötigt, mit dem man sehr einfach größere Entfernungen überqueren kann und zum anderen ein Verfahren, mit dem man kurze Strecken überwinden kann. Da keine der dargestellten Bewegungsformen diese beiden Eigenschaften in einem vereint, muss eine Kombination aus zwei Verfahren gewählt werden.
Als Fortbewegung zum Überbrücken großer Entfernungen würde sich Magic-Locomotion anbieten. Magic-Locomotion ist offenbar die einzige Bewegungsform, die das Überbrücken größerer Entfernungen bei geringer Cybersickness unterstützt. Als Kandidaten für kleinere Distanzen bzw. präzisere Bewegungsformen kommen Artificial-Locomotion, Redirected-Walking

	Entfernung	Präzision	Cybersickness	Platzbedarf	Anwendungsbereich
Magic-Locomotion	3	2	1	1	sowohl für räumliche als auch für offene Umgebungen
Artificial-Locomotion	1	3	3	1	sowohl für räumliche als auch für offene Umgebungen
Reoriented-World	2	2	2	1	überwiegend für offene Umgebungen
Redirected-Walking	1	3	1	3	räumliche Umgebungen
Motion-Triggered	1	2	2	1	sowohl für räumliche als auch für offene Umgebungen
Surrogate-Vehicle	2	3	2	1	überwiegend für offene Umgebungen

Tabelle 2.1: Bewertungstabelle der Navigationsformen

und Motion-Triggered in Frage. Artificial-Locomotion scheidet aufgrund der zu hohen Wahrscheinlichkeit auf Cybersickness aus. Um die Bewegungsform Redirected-Walking umsetzen zu können, wird eine große begehbare Fläche benötigt. Somit scheidet auch Redirected-Walking aus, da in der Testumgebung diese Bewegungsform nicht anwendbar ist. Als letzter Kandidat bleibt nur noch die Bewegungsform Motion-Triggered, diese eignet sich bislang am besten für die präzise Fortbewegung, da die Steuerung mit einer natürlichen Körperbewegung übertragen wird. Allerdings sollte bei der Umsetzung darauf geachtet werden, dass die Anzeichen einer Cybersickness für den Nutzer gering gehalten werden. Somit werden die Navigationsformen Magic-Locomotion und Motion-Triggered als Kandidaten zum Überqueren kleiner und großer Strecken in dieser Arbeit verwendet.
Die genaue Funktionsweise der ausgewählten Bewegungsformen wird in Kapitel 3 anhand der funktionalen und nicht funktionalen Anforderungsanalyse beschrieben.

2.2 Klassifizierung der Interface-Elemente

Der Begriff Interface, in Bezug auf 3D-Anwendungen, wird oft im Zusammenhang mit grafischen Benutzerschnittstellen (GUI) gebracht. Normalerweise beziehen sich GUIs auf die Steuerelemente wie z.B. Schaltfläche im Menü einer Anwendung, aber auch auf Statusinformationen wie Lebenspunkte oder Munition in einem Computerspiel. Grundsätzlich sollten Interface-Elemente den Nutzer das passende Feedback geben können, um auf Änderungen in der Umgebung zur richtigen Zeit reagieren zu können. Hierfür wurde in der Masterarbeit "Beyond the HUD: User Interfaces for Increased Player Immersion in FPS Games"von Erik Fagerholt und Magnus Lorentz eine Terminologie aufgestellt, um Interface-Elemente zu er-

fassen, die sich auf die Interaktion zwischen Spieler und Spiel beziehen [Novak (2011)]. In den folgenden Unterkapiteln werden vier verschiedene Interface-Elemente beschrieben und anhand von Beispielen belegt.

2.2.1 Diegetic

Diegetic Interface-Elemente werden als Teil der 3D-Umgebung definiert und sind im Vergleich zu den klassischen GUI-Elementen immersiv [Novak (2011)]. Der Begriff Diegetic bedeutet soviel wie „innerhalb der Erzählung“ [Unity-Technologies] und wird in 3D-Szenen auf zwei Arten gehandhabt .

1. Eine Art orientiert sich an dem klassischen HUD, von dem die Interface-Elemente in die Ausrüstung des virtuellen Charakters integriert werden. Diese Art wird häufig in Spielszenen verwendet, wenn der virtuelle Avatar eine schwere Ausrüstung trägt oder als ein Roboter agiert [TvTropes (2013)]. In der Abbildung 2.2 werden zwei beispielhafte Umsetzungen der Diegentic-UI vorgestellt.

2. Die andere Art verwendet im Gegensatz zur Ersten, keine Interface-Elemente. Stattdessen sollen UI-Informationen auf natürliche Weise, über Anzeichen oder Hinweise in der Szene deutlich gemacht werden. Beispielsweise wird ein verwundeter Spielcharakter hinken, anstelle der Lebensanzeige im HUD [TvTropes (2013)].

(a) Screenshot aus dem Computerspiel Metro 2033 (b) Screenshot aus dem Computerspiel Dead Space

Abbildung 2.2: Die zwei Screenshots aus den Computerspielen (Metro 2033, Dead Space), zeigen wie Diegetic Interface-Elemente in die 3D-Szene integriert werden können. Quellen: Stonehouse (2011) , Andrews (2010)

In virtuellen Fahrzeugen werden die Diegetic Interface-Elemente oftmals im Cockpit integriert, indem die Statusinformationen und Schalter im Fahrzeug angezeigt werden, die der Spieler

aus der egozentrischen Perspektive sehen und sogar interagieren kann.
Benutzeroberflächen mit Diegetic Elementen werden im Allgemeinen dazu verwendet, um den Grad der Immersion in 3D-Szenen zu steigern. Es wirkt auf den Spieler oftmals viel realistischer, wenn sich eine virtuelle Person in einer vergleichbar realen Situation befindet und nicht mit künstlichen Symbolen wie z.B. Gesundheit oder Munition überladen wird [TvTropes (2013)].

2.2.2 Non-diegetic

Die Non-diegetic Interface-Elemente existieren nur außerhalb der 3D-Umgebung und sind auch nicht immersiv. Üblicherweise werden diese Interface-Elemente nur für den Spieler in einem Head-Up-Display (HUD) visualisiert [Novak (2011)]. Das HUD ist ein Anzeigesystem, das an die Blickrichtung des Spielers gekoppelt ist und wird verwendet, um Informationen in das Sichtfeld des Spielers zu projizieren. Es wurde ursprünglich für Piloten in Kampfflugzeugen entwickelt, um bei einer bestimmten Aufgabe wie z.B. bei der Navigation oder beim Landeanflug zu unterstützen. Aktuelle Computerspiele wie z.B. Fortnite (Abbildung 2.3) verwenden das HUD hingegen, um allgemeine Statusinformationen für den Spieler bereitzustellen[2] .

Abbildung 2.3: In einem Computerspiel werden Non-diegetic Interface-Elemente in einem HUD visualisiert. Die Informationen werden verteilt in der Blickrichtung des Spielers eingeblendet. Quelle: HoldToReset [3]

2.2.3 Spatial

Die Spatial Interface-Elemente sind nicht immersiv und existieren nur innerhalb der 3D-Umgebung, die z.B. als Teil der Geometrie eingebunden werden, um Informationen an den 3D-

[2]https://de.wikipedia.org/wiki/Head-up-Display Aufgerufen am 28.06.2018
[3]https://holdtoreset.com/a-beginners-guide-to-fortnite-battle-royale/ Aufgerufen am 28.06.2018

Objekten zu projizieren [ZEALOUSYS (2011), Novak (2011)]. Oft sind Spatial Interface-Elemente an die fiktive Handlung der Szene gebunden und ermöglichen dem Spieler Informationen innerhalb der 3D-Szene mitzuteilen [Stonehouse (2011)]. Als Beispiel wird in der Abbildung 2.4 eine aufleuchtende Wegbeschreibung in der 3D-Szene eingeblendet, um den Spieler in der Navigation zu unterstützen. Dadurch kann sich der Spieler besser orientieren und muss folglich nicht auf die Karte zurückgreifen, was zu einem höheren Grad der Immersion führen kann [Stonehouse (2011)].

Abbildung 2.4: Ein Screenshot aus einem Spiel, in dem das Spatial Interface als eine aufleuchtende Wegbeschreibung in der 3D-Szene eingeblendet wird. Quelle: Engadget[4]

2.2.4 Meta

Meta Interface-Elemente gehören zwar nicht zu der 3D-Umgebung, sind aber immersiv und werden im Allgemeinen als Teil der Spieloberfläche angesehen. Ein Beispiel wäre ein 2D-Overlay-Interface in einem Action-Spiel mit einer egozentrischen Perspektive, indem die Blutspritzer auf dem Display angezeigt werden, wenn der Spieler verletzt wurde (Abbildung 2.5). Gut umgesetzt, könnte der immersive Effekt für den Spieler im Wesentlichen derselbe sein, wie vom Diegtic Interface [Novak (2011)].

2.2.5 Auswahl geeigneter Interface-Elemente

In Kapitel 3 wird sich zeigen, dass nur zwei Formen der Interface Kategorien für die Umsetzung verwendet werden. Für die Navigationsform Miniature-Movement werden Diegetic Interface-Elemente in das Miniatur-Modell eingebaut, die dem Nutzer die aktuelle Raumlage der virtuellen Umgebung zeigen sollen. Ergänzend dazu soll eine Minimap (kleine 2D-Karte

[4]https://www.engadget.com/2008/07/16/joystiq-e3-hands-on-fable-2/?guccounter=1 Aufgerufen am 03.07.2018

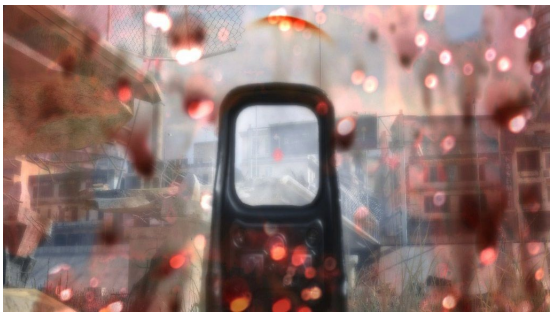


Abbildung 2.5: Das Screenshot aus dem Spiel Call-of-Duty visualisiert mit einem 2D-Overlay-Interface Blutspritzer auf der Bildfläche. Hiermit wird dem Spieler der Zustand des virtuellen Avatars mitgeteilt, anstelle der Lebenspunkte in einem HUD. Quelle: ZEALOUSYS (2011)

von der Anwendungsdomäne) in das HUD eingeblendet, um den Nutzer bei der Orientierung zu unterstützen, unabhängig von der verwendeten Fortbewegung. Die verwendeten Interface-Elemente im HUD gehören zu der Kategorie Non-Diegetic. Es wird sich im weiteren Verlauf der Arbeit zeigen, wie die Diegetic und Non-diegetic Interface-Elemente aufgebaut und im Kapitel 4 in die Anwendung integriert werden.

2.3 Szenen-Graphen

Als zentraler Bestandteil der virtuellen Welt wird eine Szene verwendet, um das äußere Erscheinungsbild und die notwendigen Informationen der Struktur zu beschreiben. Die Szene ist ein spezielles 3D-Modell, das die räumlichen und zeitlichen Beziehungen zwischen den Objekten herstellt und neben der Geometrie- und Materialbeschreibungen aller 3D-Objekte, auch Kameraeinstellungen, Audio- und die Lichtquellen definiert [Kühhirt und Rittermann (2005)]. Die bildlichen Inhalte der Szene werden für den Nutzer zur Laufzeit gerendert und auf geeigneten Ausgabegeräten, wie z.B. Monitor oder Head-Mounted-Display (HMD) dargestellt [Dörner u. a. (2014)]. Eine Szene kann sich zur Laufzeit auf zwei Arten dynamisch verändern:

Interaktiv
: Wenn 3D-Objekte einer Szene auf die Eingabe des Nutzers reagieren.

Animiert
: Wenn 3D-Objekte über die Zeit ihre Position in der Szene verändern.

Verhalten / Zustandsbasierend
: Wenn Objekte auf Ereignisse, Nutzereingaben oder Wechselwirkungen mit anderen

Objekten, mit Zustandsänderungen reagieren.

Als geeignetes Datenmodell hat sich der Szenengraph als grundlegende Beschreibungsform einer Szene durchgesetzt. Der Szenengraph ist ein gerichteter azyklischer Graph (engl. Directed Acyclic Graph - DAG), mit dem sich hierarchisch aufgebaute Szenen effizient beschreiben lassen [Dörner u. a. (2014)]. Wie in jedem Graphen, sind auch im Szenengraphen Knoten enthalten, die über gerichtete Kanten miteinander verbunden sind. Man bezeichnet A als Elternknoten und B als Kindknoten, wenn eine Kante von Knoten A zu Knoten B verläuft. Ähnlich wie bei einem Baum, hat auch der Szenengraph einen Wurzelknoten, der keinen Elternknoten besitzt und den Einstiegspunkt der gesamten Szene darstellt. Blattknoten sind Knoten aus dem Szenengraph, die keine Kindknoten besitzen und repräsentieren 3D-Objekte in der Szene. Alle anderen Knoten dienen der gruppierenden Funktion. Die meisten Szenengraphen haben, im Gegensatz zu einem Baum, mehrere Elternknoten und ermöglichen dadurch eine kompakte Repräsentation hierarchisch aufgebauter Objekte, die z.B. mehrfach in der Szene verwendet werden [Dörner u. a. (2014)].

Eine Besonderheit im Szenengraphen ist die Transformationsgruppe, die ein (lokales) Koordi-

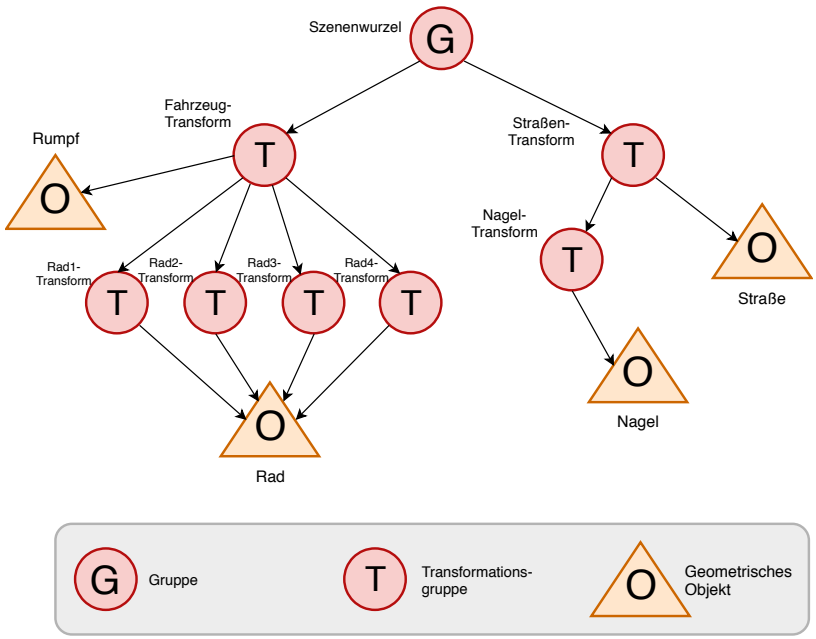


Abbildung 2.6: In der Abbildung wird ein beispielhafter Szenengraph dargestellt, indem das Fahrzeug mit Rädern modelliert wurde. Es wird nur ein 3D-Objekt für das Rad im Speicher geladen und wird mehrfach wiederverwendet. Quelle [Dörner u. a. (2013)]

natensystem für die Kindknoten definiert. D.h. alle Transformationen im geometrischen Sinne (Verschiebung, Drehung und Skalierung) werden im Kindknoten auf dem lokalen Koordinatensystem des übergeordneten Elternknotens durchgeführt werden. Abbildung 2.6 zeigt ein Beispiel einer Szene, in der ein Fahrzeug hierarchisch aufgebaut wird. Die Räder wurden in vier Transformationsknoten aufgeteilt, die wiederum auf ein definiertes 3D-Objekt verweisen. Durch die Wiederverwendbarkeit der 3D-Objekte wird der benötigte Speicher reduziert, d.h. in diesem Beispiel wird nur ein Rad für das Fahrzeug geladen und auf vier unterschiedlichen Positionen referenziert.
Szenengraphen werden unter anderem von Szenengraphsystemen wie OpenSceneGraph und OpenSG verwendet, als auch in Gameengines wie z.B. Unity 3D und Unreal-Engine. Beliebte Gameengines liefern meist eine Komplettlösung, die eine Echtzeitumgebung und einen eingebetteten Editor bereitstellen, um die Szene auf graphischer Ebene zu erweitern.
In Kapitel 4 wird sich zeigen, wie die Komponenten der Navigationsanwendung auf das Modell des Szenengraphen inhaltlich abgebildet werden.

2.4 Zusammenfassung

In diesem Kapitel wurden zunächst die unterschiedlichen Navigationskonzepte für VR vorgestellt und mithilfe einer Bewertungstabelle die geeigneten Navigationsformen (Magic-Locomotion und Motion-Triggered) für diese Arbeit identifiziert. Das wesentliche Ergebnis war, dass aus den verschiedenen Arten der Navigationskonzepte für diese spezielle Aufgabenstellung, folgende infrage kommen: Motion-Triggered und Magic-Locomotion. Im Abschnitt 2.2 wurden verschiedene Interface Kategorien kurz erläutert und diejenigen identifiziert, die für diese Arbeit relevant sind. Den Abschluss bildet eine kurze Einführung in das Modell des Szenengraphen, um ein besseres Verständnis für die Umsetzung in Kapitel 4 herzustellen. Auf der Grundlage der Ergebnisse diese Kapitels und den Anforderungen aus Kapitel 3, wird dann in Kapitel 4 eine Konzept für die virtuelle Navigation in einer Flugzeugkabine entwickelt.

3 Anforderungsanalyse

In diesem Kapitel werden zunächst im Abschnitt 3.1 die funktionalen Anforderungen eingeführt, die die verschiedenen Arten der Fortbewegungen in einer virtuellen 3D Welt formulieren. Das Kapitel schließt mit einer Eingrenzung der Anwendungsdomäne im Unterabschnitt 3.1.5 ab. Im Abschnitt 3.2 werden die nicht funktionalen Anforderungen bezüglich der Bedienbarkeit, als auch der Softwareentwurf formuliert.

3.1 Funktionale Anforderung

Das System soll drei verschiedene Arten der Navigation in einer VR-Welt implementieren, um im Nachgang einer Evaluierung, dass am meisten akzeptierte Konzept zu identifizieren. Die Funktionen werden in den folgenden Anforderungen beschrieben.

3.1.1 Anforderung 1: Drag-Movement

Das Drag Movement hat Ähnlichkeiten mit der Fortbewegung eines Skateboard Fahrers, der sich an einem Handlauf nach vorne zieht. Es soll dem Nutzer eine natürliche Art der Fortbewegung in einem virtuellen Raum bieten. Mithilfe der Eingabegeräte soll der Nutzer nach den Objekten in der Umgebung greifen können, um aktiv seine Position zu verschieben. Der Nutzer soll frei entscheiden können, ob er sich an einen Gegenstand heranzieht, oder sich von diesem abstößt. Diese Art der Fortbewegung ist ausschließlich mit den Händen zu steuern, d.h. der Anwender kann sich in der virtuellen Umgebung fortbewegen, ohne seine reale Position mit den Füßen zu verändern. Zwar ähnelt diese Art der Fortbewegung auf einem Skateboard, dennoch unterscheiden sich einige physikalische Merkmale. Wenn die Skateboard Analogie physikalisch korrekt umgesetzt werden würde, müssten weitere Eigenschaften dem Konzept beigefügt werden.

Nachroll-Effekt

Der Nachroll-Effekt entsteht dann, wenn der Skateboard Fahrer sich von einem Gegenstand wegdrückt. Würde man diese Eigenschaft in die VR-Umgebung übertragen, so könnte der Anwender innerhalb kürzester Zeit große Distanzen überbrücken. Sowohl

die Geschwindigkeit als auch die Dauer dieser Eigenschaft, könnte dem Anwender bei der Orientierung schaden. Schnelle Bewegungen werden in VR häufig als unangenehm empfunden.

Abbremsen

Sowohl in der Realität als auch in der VR muss der Fahrende die Möglichkeit zum Abbremsen besitzen. In der Realität kann der Fahrer durch z.B. das Festhalten an der Umgebung abrupt anhalten oder aktiv mit dem Board bremsen. Übertragen auf die VR müssten die Eingabegeräte in der Lage sein die Fußbewegung des Teilnehmers zu erkennen, um das aktive Bremsen mit dem Board zu simulieren.

Kollision

Grundsätzlich führen Kollisionen mit der realen Umgebung zum sofortigem Stillstand. In der VR sind Kollisionen durch die Eingabegeräte nur schwer zu vermeiden. Unbewusste Kollisionen zwischen Eingabegerät und der virtuellen Umgebung würden den Nutzer immer wieder zum plötzlichen Anhalten zwingen. Solch unerwartete Ereignisse können den Nutzer verwirren oder sogar eine Cybersickness auslösen.

Diese genannten Phänomene könnten dazu führen, dass diese Art der Fortbewegung vom VR-Nutzer nicht akzeptiert wird. Aus diesem Grund werden die aufgezählten Eigenschaften in das Konzept nicht aufgenommen.
Prinzipiell soll sich der Nutzer uneingeschränkt in alle Richtungen bewegen können. Dies würde bedeuten, dass sich der Nutzer sowohl auf der Fläche, als auch in die Höhe bewegen kann. Es soll aber auch möglich sein die Freiheitsgraden der Bewegungsrichtung einschränken zu können, um in der anschließenden Evaluierung, die Konfiguration der Freiheitsgraden zu ermitteln, die zu der Bedienbarkeit am meisten beiträgt.
Es soll möglich sein für die jeweilige Anwendungsdomäne, die greifbaren Objekt-Typen frei zu definieren, so dass Drag Movement bestmöglich unterstützt wird. Zum Beispiel in einem korridorartigen Durchgang einer Flugzeugkabine eignen sich die Sitze und die Gepäckablage.

3.1.2 Anforderung 2: Miniature-Movement

Nach dem Konzept von Worlds in Miniature (WIM) [Stoakley u. a. (1995)] soll eine Lösung zur Orientierung und Fortbewegung in einer VR umgesetzt werden. Dabei spielt die Miniatur in diesem Konzept eine wichtige Rolle und bildet das Zentrum der Funktionalität. Das miniaturisierte Modell der Anwendungsdomäne muss soweit abstrahiert werden, dass einerseits die Orientierung noch ermöglicht wird und andererseits der Detailgrad ausreicht, um einen

erkennbaren Bezug zwischen Modell und Miniatur herstellen zu können. Hierfür sollte ein angemessener Level of Detail (LOD) gewählt werden. Im Gegensatz zu einer Mini-Map, die in der Regel eine 2D Aufsicht darstellt, ist die Miniatur ein reduziertes 3D Modell, die das große Modell in eine überschaubare Anzahl von Sektoren unterteilen soll. Diese Sektoren sind die Gebiete aus der virtuellen Umgebung bzw. der Anwendungsdomäne, in denen sich der Nutzer aufhalten kann.
Das Konzept fokussiert sich auf die Interaktion mit der Miniatur und soll für die Nutzung in drei unterschiedliche Phasen unterteilt werden.

Positionieren

Mithilfe der Eingabegeräte muss der Anwender die Miniatur in seine Umgebung platzieren können. Die Positionierung der Miniatur und die folgenden Phasen der Interaktion sollen nur mit einem Eingabegerät durchgeführt werden. Das bedeutet, dass jede Phase nur nacheinander bedient werden kann. Zuerst soll der Nutzer die Miniatur im sichtbaren Bereich platzieren können. Diese Interaktion soll mit dem gedrückthalten und loslassen der Taste am Eingabegerät gesteuert werden. Solange die Taste zum Platzieren gedrückt wird, soll die Miniatur an der virtuellen Hand gekoppelt bleiben, wie eine Spatial-UI (siehe Abbildung 3.1). Dadurch wird eine neue Perspektive für die Orientierung geboten und ermöglicht dem Nutzer eine Wegplanung durchzuführen. Die Miniatur bleibt bis zum Loslassen der Taste an der virtuellen Hand gekoppelt.

Scannen

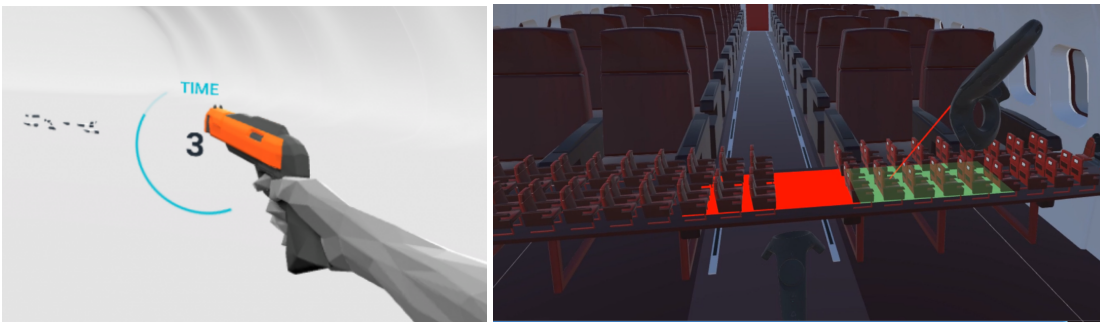
Mit einem Eingabegerät soll es dem Nutzer möglich sein die aufgelisteten Sektoren der Miniatur zu scannen. Ein Laserpointer soll einen verlängerten Zeigestock simulieren, mit dem der Nutzer die Sektoren in der Miniatur überqueren kann. Der getroffene Sektor soll farblich gekennzeichnet werden, um den Nutzer ein visuelles Feedback geben zu können.

Selektieren

Die Selektion setzt einen gültigen Scan aus der vorherigen Phase voraus. Dabei muss der Nutzer mit dem Laserpointer eins der Sektoren treffen, um selektieren zu können. Mit der Selektion ist die Auswahl des Sektors gemeint, für das sich der Nutzer im Scan Vorgang entschieden hat. Eine gültige Selektion soll den Nutzer mittig in den Sektor teleportieren und die zuvor gezeigte Miniatur ausblenden.

Um eine möglichst gute Orientierung gewährleisten zu können, soll in der Miniatur der Sektor, in dem sich der Nutzer aktuell aufhält, farblich hervorgehoben werden. Als Beispiel

wird in der Grafik (Abbildung 3.1) der aktuelle Sektor in rot und das gescannte Sektor in einer transparenten Farbe hervorgehoben. Die drei verschiedenen Interaktionsformen müssen mit dem Eingabegerät deutlich unterscheidbar realisiert werden, um einen versehentlichen Positionswechsel zu vermeiden.



(a) Ein Beispiel einer Spatial UI (b) Selektion an einem Miniaturmodel

Abbildung 3.1: (a) In der Abbildung wird ein Spatial Interface definiert. Das übliche HUD-Element für die Munitionsanzeige wurde direkt an die Waffe gekoppelt. Quelle: Unity3d [1]
(b) Selektion an einer Miniatur, um den Ziel Sektor aus dem Modell zu bestimmen. Als Hilfsmittel der Orientierung soll die rot eingefärbte Fläche dem Nutzer den aktuellen Sektor einblenden, in dem er sich aktuell aufhält. Mit einem Laserstrahl scannt der Nutzer die Sektoren der Miniatur. Der getroffene Sektor wird in grün hervorgehoben.

3.1.3 Anforderung 3: Teleportation

Zu Kontrollzwecken der entwickelten Konzepte und um die Qualität evaluieren zu können, soll als vergleichbare Lösung die klassische Teleportation als Möglichkeit der Navigation umgesetzt werden. Die frei wählbaren Positionen für die Teleportation sollen durch geeignete Constraints in Abhängigkeit der Domäne eingeschränkt werden. Es soll nicht möglich sein sich aus der Virtuellen Welt zu teleportieren bzw. Bereiche zu betreten, die dem Nutzer keinen Mehrwert bieten. Als ein weiterer Punkt der Einschränkung, soll die Reichweite bzw. die maximale Distanz eingeschränkt werden, vergleichbar mit der Wurfreichweite eines Menschen. Dabei soll anstelle eines klassischen Strahls ein Bogen verwendet werden. Um die maximale Distanz nicht auf die verwendete Domäne festzulegen, sollte der Winkelgrad für den Bogen konfigurierbar gehalten werden.

[1] https://unity3d.com/de/learn/tutorials/topics/virtual-reality/user-interfaces-vr Aufgerufen am 06.04.2018

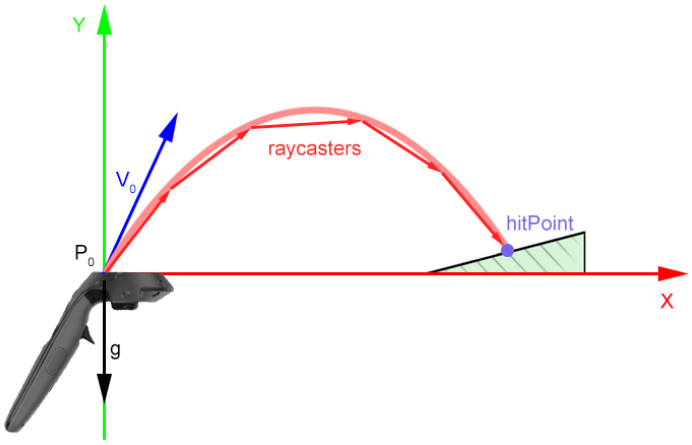


Abbildung 3.2: Ein Controller kann verwendet werden um eine Teleportation durchzuführen. Der rote Strahl dient der Anvisierung. [2]

3.1.4 2D Mini-Map

Unabhängig von den bestehenden Anforderungen soll eine 2D Mini-Map bzw. eine Karte der virtuellen Umgebung für die Orientierung in VR beigefügt werden. Die Karte sollte möglichst abstrakt Information enthalten und nur die Umrisse aus der Domäne darstellen. Diese soll in Sektoren unterteilt werden, vergleichbar mit den Sektoren der Miniatur in Anforderung 2. D.h. die Anzahl der Sektoren in der Anwendungsdomäne und Mini-Map müssen identisch sein. In der Mini-Map soll der Sektor gekennzeichnet werden, indem sich der Nutzer aktuell aufhält. Die Mini-Map soll immer nur dann eingeblendet werden, wenn der Nutzer einen neuen Sektor in der Anwendungsdomäne betritt.

3.1.5 Anwendungsdomäne

Für die Umsetzung sämtlicher Anforderungen, wird zuletzt eine Anwendungsdomäne benötigt. Hierfür soll eine nicht triviale realistische Flugzeugkabine mit einer großen Ausdehnung in der 3D-Umgebung umgesetzt werden. Der Detaillierungsgrad aus dem Modell sollte dabei möglichst hoch liegen, um eine vergleichbar reale Umgebung simulieren zu können.

[2] https://blog.mozvr.com/developing-an-aframe-teleport-component Aufgerufen am 06.04.2018

3.2 Nicht Funktionale Anforderung

In diesem Abschnitt werden die nicht funktionalen Eigenschaften als weitere Anforderungen aufgestellt. Diese sollen die verschiedenen Konzepte der Navigation in VR unterstützen und zu einer höheren Akzeptanz führen.

3.2.1 Performanz

Die Umsetzung der aufgestellten Anforderungen erfordert ein Tracking-System, welches sowohl Ein- als auch Ausgabegeräten stellen sollte. Jedes Tracking-System nimmt eine gewisse Reaktionszeit in Anspruch, beispielsweise durch das Abwarten der nächsten Abtastung der Signale. Diese Verzögerung wird Latenz genannt und wird als minimales Maß angesehen, welches aufgrund der Hardware Eigenschaften vom Tracking-System bestimmt wird. Somit kann die VR Anwendung, aufgrund der Verarbeitung von Algorithmen oder graphischen Berechnungen, die Latenz nicht unterbieten, sondern nur erhöhen. Aus diesem Grund sollte eine möglichst geringe Latenz von der Anwendung erzeugt werden, um die Reaktionszeit zwischen dem Abtasten der Hardware-Signale und den verarbeiteten Bildern zum Ausgabegerät niedrig zu halten.

3.2.2 Bedienbarkeit

Um eine möglichst intuitive Bedienbarkeit der unterschiedlichen Techniken zur Fortbewegung in der virtuellen Welt gewährleisten zu können, sollten diese keine hohe Lernkurve erfordern. Zusätzlich sollte visuelles- oder auch haptische-Feedback über die Ein- und Ausgabegeräte den Nutzer bei der Bedienung unterstützen. Abhängig von dem verwendeten Konzept sollte ein geeignetes Feedback für den Nutzer gewählt werden. Beispielsweise wäre ein haptisches Feedback angemessen, wenn der Nutzer mit einem greifbaren Objekt kollidiert bzw. nach einem Objekt greift.

3.2.3 Adaptierbarkeit

Die vorgestellte Lösung sollte mit möglichst wenig Programmieraufwand auf andere Anwendungsdomänen übertragbar sein.

3.3 Zusammenfassung

Als funktionale Anforderungen wurden drei Navigationsarten mit den entsprechenden Rahmen und Randbedingungen formuliert. Das Kapitel der nicht funktionalen Anforderungen

beschreibt die Umsetzung von den funktionalen, wie nicht funktionalen Anforderungen in einem geeigneten Entwurf.

4 Entwurf

In diesem Kapitel wird zunächst im Abschnitt 4.1 der Aufbau und die Verteilung der Hardwarekomponenten im Systemüberblick eingeführt und im Folgeabschnitt als Software-Entwurf detailliert für die Umsetzung geplant. Das Kapitel schließt mit der Abbildung des Software-Entwurfs auf das Programmiermodell von Unity 3D im Abschnitt 4.4 ab. Im darauf folgenden Abschnitt 4.5 werden die Navigationskonzepte aus der zuvor definierten Anforderungsanalyse (Kapitel 3), detailliert in der Umsetzung beschrieben.

4.1 Systemüberblick

Abbildung 4.1 zeigt die Verteilung der Softwarekomponenten, sowie die Kommunikation zwischen den Hardwarekomponenten. Die Komponenten der Hard- und Software wurden in vier verschiedene Gruppen unterteilt (grau, grün, blau und orange). Orange zeigt die Hardwarekomponenten, die als Ein- und Ausgabegeräte verwendet werden, um mit der VR-Umgebung zu interagieren. Die Komponenten (in blau) bilden die Softwarebibliotheken um die VR-Komponenten bedienen zu können. Die grüne Komponente zeigt die selbst entwickelten Navigationskonzepte, die zur Übersicht als eine Komponente zusammengefasst wurden.
Die Umsetzung einer VR-Anwendung erfordert zum einem ein Hardware System (**Tracking System**), das die räumliche Positionserkennung der Controller und vom Head-Mounted-Display (HMD) erfasst, zum anderen Softwarebibliotheken oder API's, die das Ansprechen der Hardwarekomponenten bzw. der Sensoren erlaubt. Für die Hardware-Umgebung wurde das HTC-Vive System gewählt. Mithilfe der enthaltenen Basisstationen (**Vive Base Stations**), wird Room-Scale-Tracking [1] vom HMD, als auch von den Controllern ermöglicht. Die **Linkbox** unterstützt die Kommunikation zwischen der **Working Machine** und dem HMD. Alle anderen Hardwarekomponenten vom Tracking-System kommunizieren kabellos. Der genauen Ablauf der Kommunikation, sowie die dahinter liegende Arbeitsweise wird vom Hersteller nicht bekannt gegeben.
Als Bibliotheken für den Zugriff auf die Sensordaten der VR-Hardware sowie die Schnittstelle

[1] https://en.wikipedia.org/wiki/Room_scale Aufgerufen am 25.05.2018

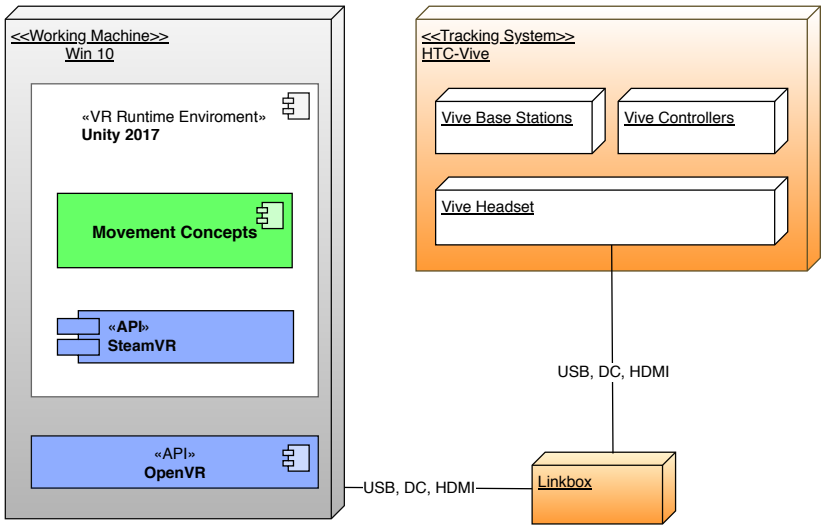


Abbildung 4.1: Das Verteilungsdiagramm zeigt die Kommunikation zwischen den Hardwarekomponenten, sowie eine grobe Übersicht der Softwarekomponenten.
Alle zugehörigen Hardware-Teile des Tracking-Systems für die VR wurden gelb eingefärbt. Die blauen Komponenten bilden die Software für den Zugriff auf das Tracking-System ab. Alle selbst entwickelten Konzepte wurden zur Übersicht in einer grünen Komponente Movement-Concepts zusammen gestellt.
Quelle: Eigene Arbeit

zum VR-Display, wurde das **SteamVR** API ausgewählt, das bereits in die Unity 3D Umgebung (**VR Runntime Enviroment**) integriert ist und von den **OpenVR** Zugriffen auf die Sensor- und Aktor-Komponenten abstrahiert.
Die Komponente **Movement-Concepts** fasst alle selbst entwickelten Komponenten zusammen, die in Unity 3D Programmiermodell die Anforderungen aus Kapitel 3 umsetzen.

4.2 Software Entwurf

In diesem Abschnitt werden die einzelnen Komponenten der Navigationskonzepte erläutert ohne auf die grundlegenden Details der Implementierung in Unity 3D einzugehen. Die Komponenten aus der Abbildung 4.2 wurden in drei verschiedene Kategorien in den Farben blau, grün und orange unterteilt. Die blaue Komponente bezeichnet den Player. Ein Player ist der virtuelle Repräsentant für die Person, die sich in der VR Umgebung bewegt. Die Komponenten in orange stellen den Content für die VR Umgebung bereit und werden unterteilt in ein Full-Scale-Model und ein Miniature-Model. Zuletzt gibt es noch die grüne Komponente Movement-Concepts. In

dieser sind folgende Komponenten der Navigationslogik enthalten:
Die **Player-Komponente** ist der virtuelle Repräsentant für eine Person, die sich in der VR-Welt bewegt und kapselt den Zugriff auf die Sensordaten und die Darstellung auf dem HMD. Es enthält zum einem ein Kamera-Objekt, welches den Kopf unifiziert und Rückschlüsse auf die Positions- und Rotations-Daten des Kopfes erlaubt. Das HUD ist eine virtuelle Leinwand und wird an das Kamera-Objekt gebunden, so dass die Kamerabewegung auf die virtuelle Leinwand übertragen wird. Es wird üblicherweise benutzt, um den Player mit Zusatzinformationen zu versorgen. Die Area (VR-Area) bezeichnet den realen Bewegungsraum des Anwenders, indem die Positions- und Rotations-Informationen der Ein- und Ausgabegeräte erfasst werden und auf die VR-Umgebung eins zu eins übertragen werden. Die letzten beiden Objekte, Hand-left und Hand-right, bilden die Referenzen auf die linke und rechte Hand. Üblicherweise werden diese in der VR-Umgebung mit den linken und rechten Controllern unifiziert. Da die Komponente Player den Zugriff auf die SteamVR API kapselt, entfallen Abhängigkeiten von der Movement-Concepts auf die verwendete API bzw. auf die Hardware.
Die zwei Komponenten **Full-Scale-Model** und **Miniature-Model** bilden den Content der VR-Anwendung. Diese besitzen keine eigenständigen Funktionalitäten für die Navigationskonzepte, sie dienen lediglich als Daten- und Visualisierungs-Objekte. Das Full-Scale-Model ist das maßstabsgetreue Flugzeugmodell mit einer Kabine und enthält einen relativ hohen Detailgrad bezüglich der Darstellung der Objekte. Interactive-Objects werden für die Umsetzung der Drag-Movement Komponente benötigt, um die Kollisionserkennung mit den virtuellen Controller erkennen zu können. Da die Dimension des Full-Scale-Model deutlich größer ist als die real begehbare Spielfläche, wird das Modell in Sektoren unterteilt. Jeder Sektor entspricht der realen Spielfläche. Das Miniature-Model ist eine verkleinerte Version der Flugzeugkabine, ein 3D-Modell mit einer niedrig aufgelösten Grafik. Die Farben werden zum Hervorheben der Player-Position (Position-Highlight) und der Selektion (Selection-Highlight) im Miniature-Model verwendet. Dieses Modell wird für die Interaktion im Miniature-Movement benötigt und enthält wie das Full-Scale-Model Sektoren. Dabei muss eine eins zu eins Beziehung zwischen den Sektoren aus dem Miniature-Model und dem Full-Scale-Model existieren.
Die **Teleport-Komponente** stellt eine Sammlung von Skripten zusammen, welche in der SteamVR API vordefiniert wurden. Diese Komponente verändert die Player-Position relativ zur dargestellten 3D-Welt und benötigt den Zugriff auf die virtuelle Hand, um Richtung und Winkel des Teleportations-Strahls bestimmen zu können. Die Teleport Komponente greift aktiv auf die virtuelle Position des Players zu, wenn der Player mit dem Teleportations-Strahl einen Sektor trifft. Der getroffene Sektor wird anhand der Kollisionserkennung zwischen dem Strahl und dem Sektor ermittelt und bildet das Ziel der Teleportation. Mit Hilfe der Sektoren aus dem

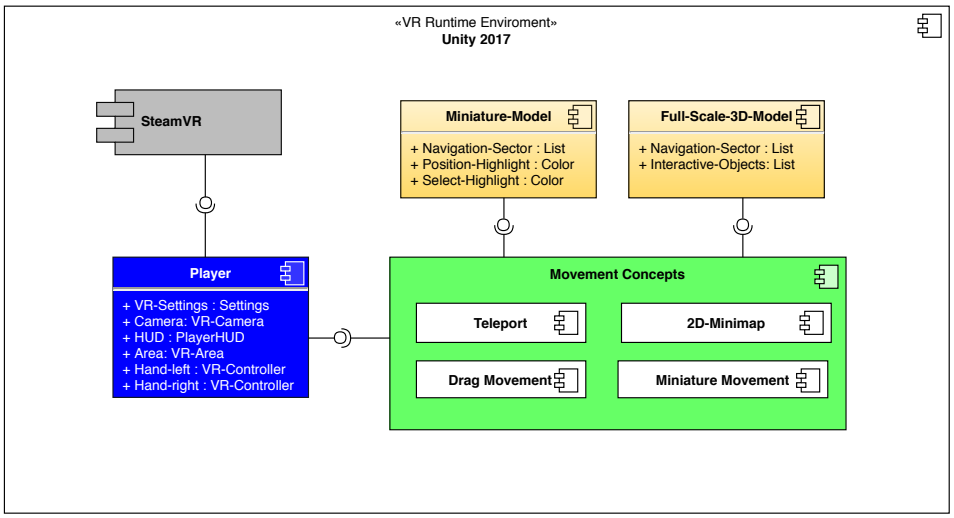


Abbildung 4.2: Überblick der Komponenten und deren Schnittstellen für den Software-Entwurf.
Quelle: Eigene Arbeit

Full-Scale-Model lassen sich ungültige Ziele filtern, so dass sich der Player nicht außerhalb der Sektoren teleportieren kann.
Die **2D-Mini-Map** dient dazu, den Übergang zwischen zwei Sektoren deutlich zu machen. Beim Wechsel der Sektoren wird die Mini-Map im HUD eingeblendet, um die Orientierung im virtuellen Raum zu unterstützen. Anhand der Player Position ermittelt diese Komponente den aktuellen Sektor, in dem sich der Player aufhält und blendet diese aktuelle Position als Mini-Map bzw. kleine Karte im HUD für einen kurzen Zeitraum ein. Ausreichend für diese Orientierung ist eine stark vereinfachte Außenaufsicht auf das Flugzeugmodell und eine Aufteilung in Sektoren, die den Sektoren aus der Innenansicht konsistent ist. In Abbildung 4.3 wird die Außenaufsicht in ein 2D-Modell reduziert und als Beispiel für einen Sektor farblich gekennzeichnet.

4.3 Anwendungsdomäne

Die Umsetzung der Anforderungen benötigt zunächst eine geeignete virtuelle Umgebung bzw. Modell. Bei der Auswahl des geeigneten Modells spielt die Größe und die Komplexität eine wichtige Rolle. Eine virtuelle Umgebung muss groß genug sein, um die Problemstellung der Fortbewegung in VR zu erschaffen. Dabei sollte beachtet werden, wie komplex der Aufbau von Knoten- und Kantenpunkte im Polygon konstruiert wurde. Je komplexer das Modell,

(a) Eine Bildaufnahme aus der Vogelperspektive vom 3D-Model

(b) Detailreduziertes 2D-Flugzeug

(c) Hervorhebung eines Sektors im 2D-Flugzeug

Abbildung 4.3: In dieser Abbildung wird eine Bildaufnahme vom 3D-Flugzeugmodel aus der Vogelperspektive auf 2D reduziert, so dass ein Sprite daraus entstehen kann. Quelle: Eigene Arbeit

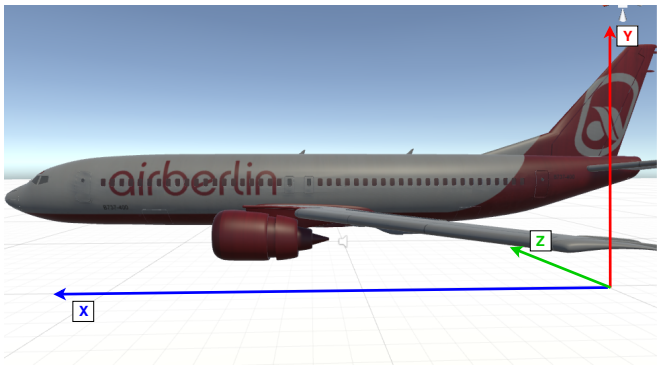
umso mehr Rechenleistung wird vom Echtzeitsystem für die Visualisierung beansprucht. Folglich kann es die Performanz und Reaktionszeit beeinflussen, was sich negativ auf die Wahrnehmung der umgesetzten Navigationskonzepte auswirken kann. Aus diesem Grund wurde die Auswahl auf ein käuflich erworbenes Flugzeug Modell gesetzt, mit dem Fokus auf eine geringe Polygon Anzahl, um die Performanz der Echtzeitumgebung gewährleisten zu können. Es handelt sich um einen maßstabsgetreuen Nachbau der Boing 737-400 (Abbildung 4.4). Das erworbene Model wird in drei unterschiedlichen 3D Formaten angeboten: MAX-Format, 3DS-Format und OBJ-Format. Lediglich die letzten zwei Formate können von der Echtzeitumgebung Unity 3D importiert werden. Gewählt wurde das 3DS-Format, dort mussten folgende Dinge nachgearbeitet werden:

- Die Texturen wurden zwar als Unity-Materialien erkannt, aber teilweise unvollständig zum 3D-Modell zugewiesen. Texturen werden in Unity 3D als Materialien definiert, um Eigenschaften und Verhalten der Texturen steuern zu können. Die nicht erkannten Materialien wurden nachträglich zum 3D-Modell zugewiesen.

- Im käuflich erworbenen Modell wurden Normal-Maps als visuelle Erweiterung der Beleuchtung definiert. Normal-Maps werden üblicherweise dazu verwendet, um Beleuchtungseffekte zu simulieren. Dies ist ganz entscheidend für den realistischen Eindruck vom Modell. Um Normal-Maps in Unity 3D darstellen lassen zu können, muss das Unity-Material passend konfiguriert werden. Im Importprozess wurden die Textu-

[2] https://www.wingsnews.org/end-of-the-journey-for-air-berlin Aufgerufen am 06.04.2018

(a) Air Berlin Boing 737-400 Original



(b) Ausschnitt vom importierten Modell

Abbildung 4.4: Die Abbildung zeigt den direkten Vergleich zwischen dem Original und dem Nachbau der Boing 737-400 von Air Berlin, sowie die Positionierung und Orientierung des importierten Modells in der 3D Umgebung.
Quellen: (a) wingsnews[2], (b) Ausschnitt aus der Entwicklungsumgebung Unity 3D

ren der Normal-Maps am Unity-Material fehlerhaft konfiguriert und erforderten einen manuellen Eingriff.

- Zuletzt waren Anpassungen an der Skalierung notwendig, um ein realistisches Verhältnis zwischen Spieler und Flugzeugmodell zu erhalten.

Es hat sich herausgestellt, dass der Import im OBJ-Format lediglich das Mesh erkannt wurde. Um den Import zu vervollständigen, müssten Materialien in Unity 3D für die Texturen und Normal-Maps von Hand nachgebaut, auf die einzelnen Objete zugewiesen und konfiguriert werden. Aufgrund der vielen Anpassungen für den Import im OBJ-Format wurde der Import im 3DS-Format gewählt. Wie in der Abbildung 4.4(b) dargestellt, wurde das 3D-Modell entlang der X-Achse im Koordinaten-System importiert.

4.3.1 Erweiterung des 3D-Modells für die Applikation

Zusätzlich wurden Erweiterungen am 3D Modell für die Umsetzung der Navigationskonzepte vorgenommen:

1. Definieren und Platzieren einer unsichtbaren Bodenplatte
2. Unterteilung der Bodenplatte in Sektoren
3. Erweitern jedes Sektors mit einer Kollisionserkennung
4. Erweiterung der Gepäckablage und der Flugzeugsitze mit einer Kollisionserkennung. Diese werden als interaktiven Objekte für Drag-Movement benötigt.

Die Bodenplatte im Flugzeugmodell wurde in 5 Sektoren unterteilt, wobei jeder Sektor in etwa der realen Spielfläche entspricht (Abbildung 4.5). Zusätzlich wird pro Sektor ein Skript

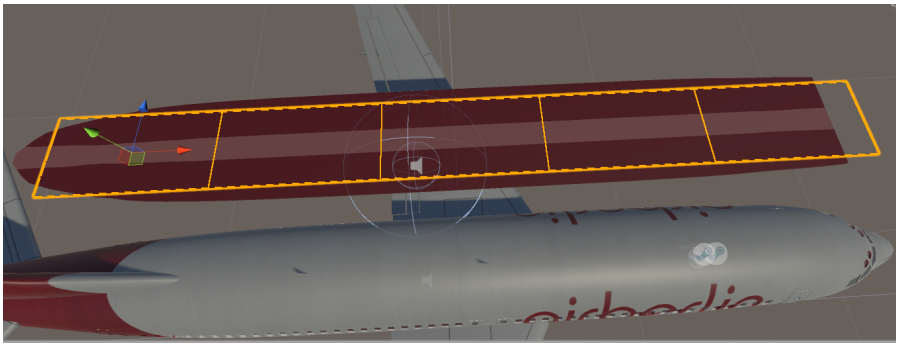


Abbildung 4.5: Die Abbildung zeigt einen Ausschnitt von der Bodenfläche aus der Flugzeugkabine. Auf die Bodenfläche wurde eine unsichtbare Bodenplatte platziert und in fünf Sektoren unterteilt. Quelle: Eigene Arbeit

aus der SteamVR API beigefügt, welches für die Teleportation vorausgesetzt wird. Das Skript manipuliert das Material des Sektors, um es farblich hervorzuheben.

4.4 Abbildung vom Softwareentwurf auf Unity 3D

In diesem Kapitel wird die Komponenten-Struktur aus 4.2 - Software-Entwurf auf das Programmiermodell in Unity 3D abgebildet.
Wie bereits im Grundlagen Kapitel 2.3 erläutert, verwendet die Echtzeitumgebung Unity 3D einen Szenengraphen als Datenmodell für die Szene. Zunächst werden die Erweiterungen am Unity 3D Szenengraphen, sowie die unterschiedliche Benennungen gegenüber der erwähnten

Definition vorgestellt.
Der Szenengraph in Unity 3D entspricht genau dem definierten Szenengraphen aus dem Grundlagen Kapitel, allerdings werden die Knotengruppen (Transformationsgruppen die nur die geometrische Informationen beinhalten) in Unity 3D **Gameobjects** genannt. Das Verhalten der Gameobjects entspricht der Knotengruppe aus der Definition. Zusätzlich können die Blattknoten des Szenengraphen in Unity 3D nicht nur 3D-Modelle, sondern auch funktionale Erweiterungen wie z.B. Kollisionserkennung und Skripte beinhalten. Diese werden in Unity 3D **Components** genannt und können auf andere Components des selben Gameobjects direkt zugreifen. Der Zugriff auf Components, die an unterschiedlichen Gameobjects hängen, kann in Unity 3D nicht automatisch durchgeführt werden. Hierfür bietet die Echtzeitumgebung mehrere Möglichkeiten auf Components im Szenengraphen zuzugreifen. Beispielsweise können Components anhand des Namen oder der zugewiesenen Tags gefunden und zugegriffen werden. Um den Zugriff eindeutig und sicher gegen Namensänderungen zu gestalten, wurden die selbst entwickelten Skripte an die Gameobjects angehägt, die den notwendigen Zugriff für die Umsetzung bieten. D.h. wenn ein Skript den Zugriff auf die virtuelle Hand benötigt, wird es an dem gleichen Gameobject der virtuellen Hand angehängt, wie das Handskript. Da die Player-Komponente auch dem Szenengraphen unterliegt, sind die virtuellen Hände als Kinderknoten des Players definiert. Der Zugriff auf die Sensordaten der jeweiligen Controller müsste direkt über die Components der virtuellen Händen geschehen. Die Komponenten Miniature-Movement und Drag-Movement erfordern solch einen Zugriff, um die Interaktion mit den Händen durchführen zu können. Aus diesem Grund wurden die beiden Navigationskonzepte jeweils an das Gameobject der virtuellen Hände angehängt. Die Programmierlogik der selbst entwickelten Komponenten wurde in Form von Skripten umgesetzt. In Unity 3D wird die Programmierlogik in der Programmiersprache C# beschrieben. Dadurch, dass jede virtuelle Hand die Navigationsskripte hält und diese unabhängig voneinander agieren, muss die Konfiguration an den Skripten für jede virtuelle Hand durchgeführt werden. Um den Konfigurationsaufwand zu minimieren, wurden die Navigationsskripte aufgeteilt in Konfiguration und Programmierlogik. Die Abbildung der Komponenten aus dem Software-Entwurf auf den Szenengraphen, wird beispielhaft anhand der selbst entwickelten Komponenten Movement-Concept und Miniature-Movement, in der Abbildung 4.6 gezeigt. Die konfigurierbaren Inhalte aus den Navigationsskripten wurden in die neue Komponente (Configuration) ausgelagert und dem Gameobject des Players angehängt. Diese Aufteilung löst das Problem der doppelten Konfiguration und ermöglicht zusätzlich eine zentrale Verwaltung der Navigationsskripte.

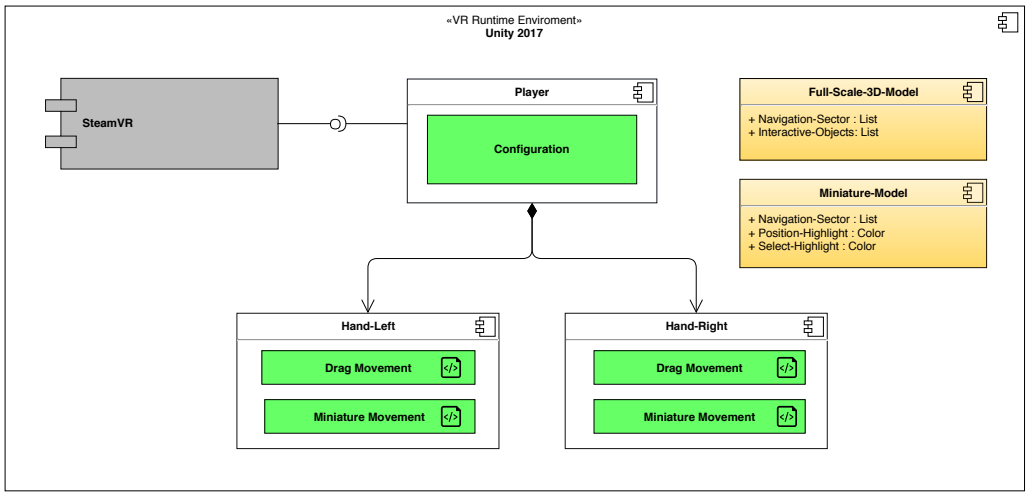


Abbildung 4.6: Die Komponenten Miniature- und Drag-Movement wurden in Konfiguration und Programmlogik aufgeteilt, um den Konfigurationsaufwand zu minimieren. Quelle: Eigene Arbeit

4.5 Umsetzung

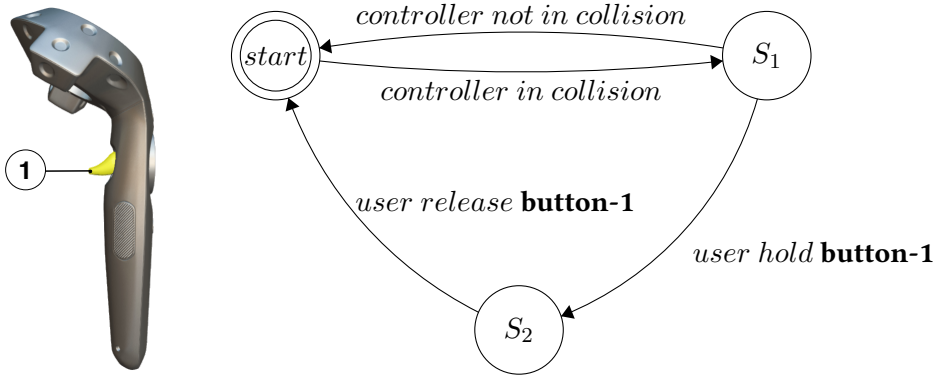
4.5.1 Drag-Movement

Das Drag-Movement Konzept ermöglicht dem Anwender nach den Flugzeugsitzen oder der Gepäckablage zu greifen, um sich voran zu ziehen oder abzustoßen (Abbildung 4.7). Dazu müs-

Abbildung 4.7: In der Abbildung wird die umgesetzte Bewegungsform *Drag-Movement* anhand einer Bildfolge gezeigt.

sen zunächst die physikalischen Abläufe, die sich bei Greifen, Ziehen oder Abstoßen ereignen, auf die Pseudo-Physik einer 3D-Engine abgebildet werden. Die virtuelle Greifbewegung wird in zwei Phasen unterteilt, Greifen und Festhalten. Üblicherweise wird die Greifbewegung in den Physik-Engines von 3D Umgebungen als Kollision zwischen zwei Objekten realisiert. Das Drag-Movement Skript verwendet die Referenz auf die virtuelle Hand, um die Kollision mit einem Objekt der virtuellen Umgebung festzustellen und die Controller-Events für das Greifen

und Festhalten. Die Interaktion des Greifens wird mithilfe der Controller umgesetzt, indem der Anwender eine Kollision zwischen dem virtuellen Controller und einem Objekt hervorruft (virtuelles Greifen) und dann die Controller-Taste gedrückt hält (virtuelles Festhalten).
Zunächst muss die Zieh-Bewegung in der virtuellen Welt aufgestellt werden, um sich an Objekte heranziehen oder abstoßen zu können. In der realen Welt zieht man den eigenen Körper an einen Gegenstand heran. Das lässt sich in der virtuellen Welt ohne mechanische Hilfsmittel nicht umsetzen, so dass diese Interaktion nachempfunden werden muss. Eine mögliche Lösung wäre die umgekehrte Logik, indem man den virtuellen Gegenstand an den realen Körper zieht oder von ihm abstoßt. Dabei wird die reale Strecke, die beim Ziehen oder Abstoßen zurückgelegt wird, eins zu eins auf die virtuelle Strecke übertragen. Die virtuelle Strecke bestimmt sich aus der Veränderung der virtuellen Controller-Position, während das Objekt festgehalten wird. Erzielt wird dieser Effekt, indem die Player-Position relativ zum Modell verschoben wird.
Es soll möglich sein die virtuelle Ziehbewegung in den Freiheitsgraden der vertikalen und horizontalen Achse einzuschränken. Dies macht man üblicherweise so, dass die Subtraktion der Positions-Vektoren auf den Eingeschränkten Achsen ausgeschlossen wird. Dadurch dass die Positionsveränderung in den vertikalen und horizontalen Achsen ignoriert wird, kann sich die virtuelle Hand vom gegriffenen Objekt entfernen (das gegriffene Objekt zieht weder in der horizontalen, noch in der vertikalen Achse nach), so dass die Kollision unterbrochen wird, was zum sofortigen Stillstand der Interaktion führt. Aus diesem Grund wurde das Drag-Movement mit drei unterschiedlichen Zuständen modelliert (Abbildung 4.8). Der Zustand *start* definiert



(a) Tastenbelegung am Controller

(b) Zustandsautomat der Interaktion - Drag-Movement

Abbildung 4.8: In der Abbildung werden die Zustände aus Drag-Movement als Zustands-Automat abgebildet. S_1 beschreibt den Zustand der Controller Kollision und S_2 den Zustand der aktiven Fortbewegung. Quelle: Eigene Arbeit

den Start- und Endzustand. Wenn im $start$ Zustand eine Kollision erkannt wird, wechselt der Zustand zu S_1. Dieser Zustand ist für die Überwachung der Kollision zuständig und setzt keine Interaktion um. In diesem Zustand werden zwei Fälle unterschieden:

1. Die Kollision wird verlassen und man landet wieder bei Zustand $start$. Dort wartet man auf die nächste Kollision.

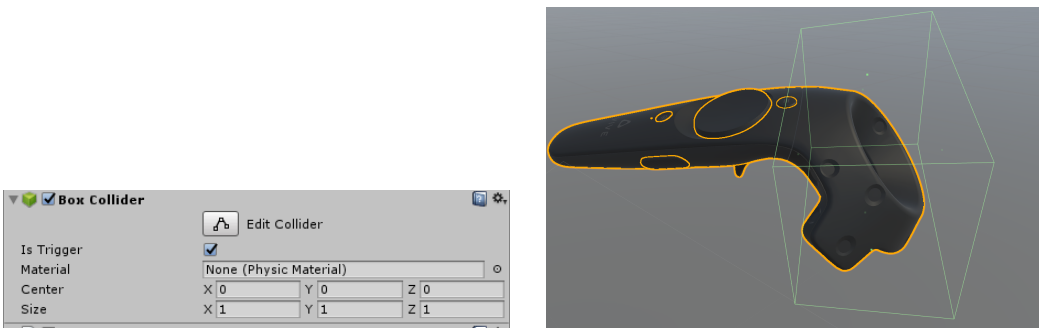
2. Wenn nach der Kollision die Controller Taste (**button-1**) gedrückt wurde, wechselt der Zustand zu S_2. Dieser Zustand definiert das Greifen und aktiviert die Interaktion.

Im Zustand S_2 wird die Zieh- und Abstoß-Bewegung übertragen, solange die Controller-Taste gedrückt wird. Somit kann die Interaktion durch das Verlassen der Kollision nicht unterbrochen werden, sondern erst mit dem Loslassen der Controller Taste.
Das geschilderte Verhalten wurde in dieser Arbeit realisiert indem:

1. Die Elemente der virtuellen Umgebung (Flugzeugsitze und Gepäckablage) mit Kollisionserkennung erweitert wurden.

2. Das Kollisionsvolumen der Controller mit einer unsichtbaren Box erweitert wurde.

Wie in Abbildung 4.9 dargestellt, wurde diese unsichtbare Box an die Spitze der virtuellen Controller befestigt und kann vom Kollisionsvolumen im Editor der Unity 3D Echtzeitumgebung konfiguriert werden. So lässt sich je nach Größe der realen Person, die Greifreichweite mit der unsichtbaren Box anpassen.



(a) Ausschnitt vom konfigurierbaren Kollisionsbereich im Editor

(b) Kopplung der Kollisionsbox am Controller

Abbildung 4.9: In der Abbildung (a) wird der konfigurierbare Kollisionsbereich in Unity 3D Editor eingestellt. Die Kollisionsbox (b) wurde an die Spitze des Controllers gekoppelt. Quelle: Eigene Arbeit

4.5.2 Miniature-Movement

Das Konzept vom Miniature-Movement bietet dem Anwender eine neue Perspektive sich mit einem 3D-Miniaturmodell zu orientieren. Mithilfe der Selektion im 3D-Miniaturmodell wird eine gezielte Teleportation in den entsprechenden Sektor der Flugzeugkabine ermöglicht (Abbildung 4.10).

Da das 3D-Miniaturmodell eine zentrale Rolle in der Interaktion spielt, muss es einerseits

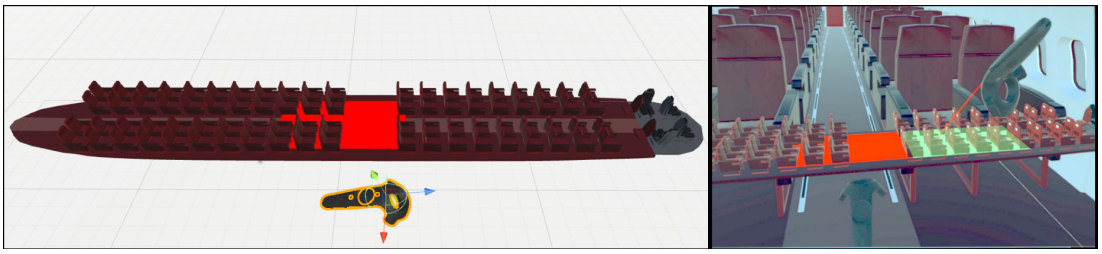
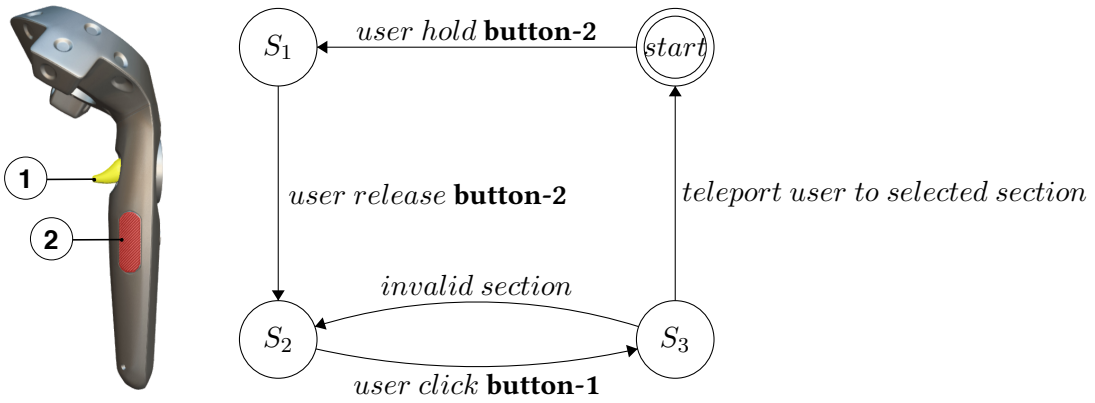


Abbildung 4.10: In der Abbildung links wird das umgesetzte 3D-Miniaturmodell und rechts die Interaktion mit dem 3D-Miniaturmodell gezeigt.

abstrakt genug sein, um die Orientierung gewährleisten zu können und andererseits im Detailgrad ausreichen, um einen erkennbaren Bezug zwischen Miniaturmodell und Full-Scale-Model herstellen zu können. Aus diesem Grund wurden die Flugzeugsitze, Sektoren und die begehbare Fläche aus der Flugzeugkabine als Miniaturmodell ausgewählt und soweit herunter skaliert, dass der Player die Fläche der Miniatur überschauen und in der virtuellen Hand festhalten kann.

Die Interaktion zwischen Miniaturmodell und virtuellen Controller erfolgt in 3-Phasen: Positionierungs-Phase, Scan-Phase und Selektions-Phase. Für die einhändige Interaktion müssen Zustände definiert werden, die diese Phasen unterscheiden. In der Abbildung 4.11 werden die 3-Phasen auf vier Zustände abgebildet. Die **Positionierungs-Phase**(S_1) wird mit dem Halten der Controller Taste (**button-2**) gestartet und als Zustandswechsel $start- > S_1$ abgebildet. Solange die Controller Taste gehalten wird, kann der Nutzer das Miniaturmodell an der gewünschten Position platzieren. Der Zustandswechsel $S_1- > S_2$ passiert dann, wenn die Controller Taste losgelassen wird.

S_2 definiert die **Scan-Phase** und benötigt einen Zeiger bzw. einen Laserpointer, um die Sektoren am Miniaturmodell präzise treffen zu können. Ähnlich wie im Unterabschnitt 4.5.1 - Drag-Movement wird der getroffene Sektor als Kollision zwischen zwei Objekten erkannt. Der getroffene Sektor wird farblich hervorgehoben und beim Verlassen der getroffenen Fläche auf



(a) Tastenbelegung am Controller

(b) Zustandsautomat der Interaktion - Miniature-Movement

Abbildung 4.11: Der Zustandsautomat (b) definiert vier Zustände für die Interaktion im Miniature-Movement. Der $start$ Zustand definiert den Start- und Endzustand und die Zustände S_1, S_2 und S_3 bilden die drei Phasen der Interaktion. Quelle: Eigene Arbeit

die Ursprungsfarbe zurückgesetzt.[3] Zusätzlich wird der aktuelle Sektor, indem sich der Player aufhält, im Miniaturmodell in einer anderen Farbe gekennzeichnet. Dazu wird ermittelt, in welchem Sektor des Full-Scale-Models sich der Player aufhält, um den Sektor auf das Miniatur-Modell zu mappen.
Sobald die Controller Taste (**button-1**) gedrückt wurde, erfolgt der Zustandswechsel $S_2 -> S_3$, um den selektierten Sektor zu evaluieren. S_3 definiert die **Selektions-Phase** und überprüft die übermittelte Selektion aus dem vorherigen Zustand. Hierbei gibt es zwei unterschiedliche Fälle:

1. Die Selektion ist fehlerhaft und liefert keinen Sektor aus dem Miniaturmodell. So wird wieder der Zustand S_2 angenommen und die Scan-Phase wird wiederholt.

2. Die Selektion ist gültig, so erfolgt der Zustandswechsel $S_3 -> start$.

Mit einem gültigen Sektor aus dem Miniaturmodell kann der Ziel-Sektor im Full-Scale-Modell berechnet werden, um anschließend die Teleportation durchzuführen. Der Player wird dabei mittig vom Ziel-Sektor platziert, weil die Sektoren aus dem Full-Scale-Modell mittig auf die begehbare Fläche positioniert wurden. Im Endzustand $start$ wird zuletzt das Miniaturmodell und der Laserpointer wieder entfernt, um die Interaktion abzuschließen.

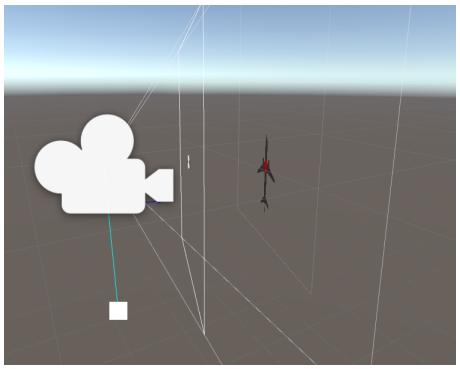
[3] Es kann maximal ein Sektor zugleich getroffen werden.

4.5.3 Teleportation

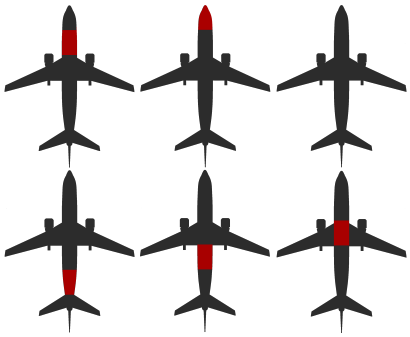
Diese Art der Fortbewegung soll zu Kontrollzwecken der entwickelten Bewegungsarten dienen. Die Umsetzung dieses Konzepts erfordert eine unterteilte Bodenfläche in Sektoren. Wie im Unterabschnitt 4.3.1 bereits beschrieben, wird in jedem Sektor eine Kollisionserkennung (Collider), sowie ein Teleport-Skript aus der SteamVR API benötigt. Alle notwendigen Funktionalitäten zum teleportieren werden von den Skripten Teleport und Teleport-Arc geboten und lassen sich bis ins Detail konfigurieren. Im Teleport-Arc Skript können Reichweite, Winkel, sowie optische Einstellungen konfiguriert werden. Die Reichweite wurde auf 60 Segmente eingestellt, dies entspricht maximal 3 Sektoren pro Teleportations-Schritt.

4.5.4 2D Mini-Map

Die Mini-Map ist eine abstrakte 2D-Grafik vom Flugzeugmodell und wird während der Fortbewegung dem Anwender in das HUD eingeblendet. Es bietet die Möglichkeit sich unabhängig von den Navigationskonzepten in der VR-Umgebung zu orientieren.
Für die Mini-Map werden abstrakte Informationen über die Umrisse des 3D-Flugzeugmodells benötigt. Hierfür wird eine Aufnahme vom 3D-Flugzeugmodell aus der Vogelperspektive erstellt, um anschließend den Detailgrad soweit zu reduzieren, bis eine einfarbige 2D-Grafik entsteht. Zusätzlich wird zu jedem Sektor aus dem 3D-Flugzeugmodell eine eigenständige Mini-Map benötigt, die den jeweiligen Sektor farblich hervorhebt. Bei fünf Sektoren im Modell, werden sechs Mini-Maps erstellt, wobei die sechste ausschließlich für den Fehlerfall benötigt wird, falls der Player keinem Sektor zugeordnet werden kann (Abbildung 4.12(b)). Bevor die Karte eingeblendet werden kann, muss zuvor ermittelt werden, in welchem Sektor sich der Player aktuell aufhält. Hierfür wird die Sektorermittlung aus dem Kapitel 4.5.3 Miniature-Movement verwendet. Die Zuordnung von Sektor auf Karte erfolgt durch das Mapping, welches zuvor konfiguriert werden muss.
Mit einem HUD wird die ermittelte Mini-Map dem Nutzer im Sichtfeld eingeblendet. Da diese Information die normale Interaktion behindert, wird die Anzeigedauer zeitlich begrenzt. Um Rücksicht auf die Orientierungszeit der Anwender zu nehmen, ist die Anzeigedauer individuell einstellbar. Die Mini-Map wird immer nur dann eingeblendet, wenn der Player einen neuen Sektor betritt.



(a) Positionierung eines HUD vor die virtuelle Kamera



(b) Aufstellung der 6 2D Flugzeuge im Sprite

Abbildung 4.12: (a) : Das HUD wird vor die virtuelle Kamera positioniert und mit den Sprites konfiguriert.
(b) : Die Sprites von der Flugzeugkabine wurden in einem Objekt zusammengestellt. Quelle: Eigene Arbeit

4.6 Zusammenfassung

In diesem Kapitel wurde die VR-Umgebung anhand eines Systemüberblicks im Abschnitt 4.1 vorgestellt. Anhand eines Softwareentwurfs wurden die Softwarekomponenten der einzelnen Navigationskonzepte im Abschnitt 4.2 erläutert. In Abschnitt 4.3 wurde das Flugzeugmodell der Anwendungsdomäne vorgestellt, das für den Import in Unity-3D aufbereitet werden musste, um eine realistische und detailreiche 3D-Umgebung in VR darstellen zu können. Die entworfenen Softwarekomponenten wurden auf das Programmiermodell von Unity-3D abgebildet und von den genutzten Bibliotheken klar abgegrenzt. In Abschnitt 4.5 wurde die detaillierte Umsetzung der einzelnen Navigationskonzepte beschrieben, die im Folgekapitel als Grundlage der Evaluierung dienen sollen.

5 Evaluierung

In diesem Kapitel werden die drei verschiedenen Navigationskonzepte für die Orientierung und Fortbewegung in einer virtuellen Umgebung evaluiert. Dazu wird im Abschnitt 5.1 das Ziel dieser Evaluierung definiert. Das Ergebnis soll zeigen, welches der zuvor genannten Konzepte für die Suche in einer virtuellen Umgebung am besten geeignet ist. Als Grundlage der Evaluation werden die drei umgesetzten Navigationskonzepte für die Fortbewegung und Orientierung verwendet, die im Kapitel 3 und 4 beschrieben wurden. Diese sollen bei der Evaluierung gleichwertig behandelt werden, um sie im späteren Verlauf miteinander vergleichen zu können. Darauffolgend wird im Abschnitt 5.2 der Versuchsaufbau und die Aufgaben beschrieben, die den Probanden gestellt wurden. Die gesammelten Ergebnisse aus der Evaluierung werden im Abschnitt 5.3 mithilfe von Balkendiagrammen visualisiert und ausgewertet. Im letzten Abschnitt wird eine Zusammenfassung über die gesammelten Ergebnisse, sowie die resultierende Bewegungsformen aus der Evaluierung wiedergegeben.

5.1 Ziel Definition

Ziel ist es, die objektive Bewertung anhand von messbaren Größen zu erfassen, als auch die subjektive Eindrücke der Probanden zu berücksichtigen. Die gemessenen Ergebnisse sollen zeigen, wie gut die Bewegungsarten zum Suchen bzw. Explorieren der VR-Umgebung geeignet sind. Zusätzlich soll herausgefunden werden, ob die theoretischen Annahmen zur intuitiven Benutzbarkeit mit den subjektiven Erfahrungen der Probanden übereinstimmen. Neben den Bewertungsskalen soll den Teilnehmern die Möglichkeit gegeben werden, Feedback und Verbesserungsmöglichkeiten zu erfassen. Bei der Auswertung besteht die Möglichkeit, dass mehrere Konzepte als vergleichbar positiv empfunden werden. In solch einem Fall kann eine Überlegung getroffen werden, die positiv empfundenen Konzepte bzw. die besten Eigenschaften zu kombinieren um davon zu profitieren.

5.2 Aufbau der Evaluation

Um die Ergebnisse der Evaluation messen zu können, wurden den Teilnehmern Aufgaben gestellt, die sie im virtuellen Raum mit den drei Navigationsformen bewältigen müssen. Diese Aufgaben müssen die Teilnehmer mit allen drei Navigationsformen einzeln lösen und im letzten Schritt der Durchführung darf eine beliebige Kombination der Navigationsformen verwendet werden. Für die ersten drei Durchführungen soll die Zeit gemessen werden, die zum Lösen der Aufgabe benötigt wurde. Im letzten Teil soll herausgefunden werden, welche präferierte Navigationsform für die Suche verwendet wurde und ob sich eine klar erkennbare Kombination aus mehreren Navigationsformen erkennen lässt.
Die Aufgaben bestanden darin, ein verstecktes 3D-Objekt in der virtuellen Umgebung zu finden, das offensichtlich nicht zum Kontext der Szene gehört. Für jede Navigationsform wurde ein neues 3D-Objekt in der virtuellen Umgebung versteckt, was nur in unmittelbarer Nähe gesehen werden kann. Im Anschluss der Durchführung wurden die subjektiven Eindrücke und gemessenen Ergebnisse anhand eines Fragebogens erfasst.
Der Fragebogen wurde mit sechs Fragen ausgelegt, die für jede Navigationsform gleichgestellt wurden. Eine der Fragen erlaubte Freitext für konkretes Feedback zur Verbesserung der Bedienung. Zusätzlich wurden über alle drei Navigationskonzepte die bevorzugte Bewegungsart zum Überqueren größerer Distanzen erfragt und die verwendete Kombination der Navigationskonzepte im letzten Schritt der Durchführung. Die Abbildung 5.1 zeigt die sechs Fragen aus dem Fragebogen, die für jede Navigationsart gestellt wurden.

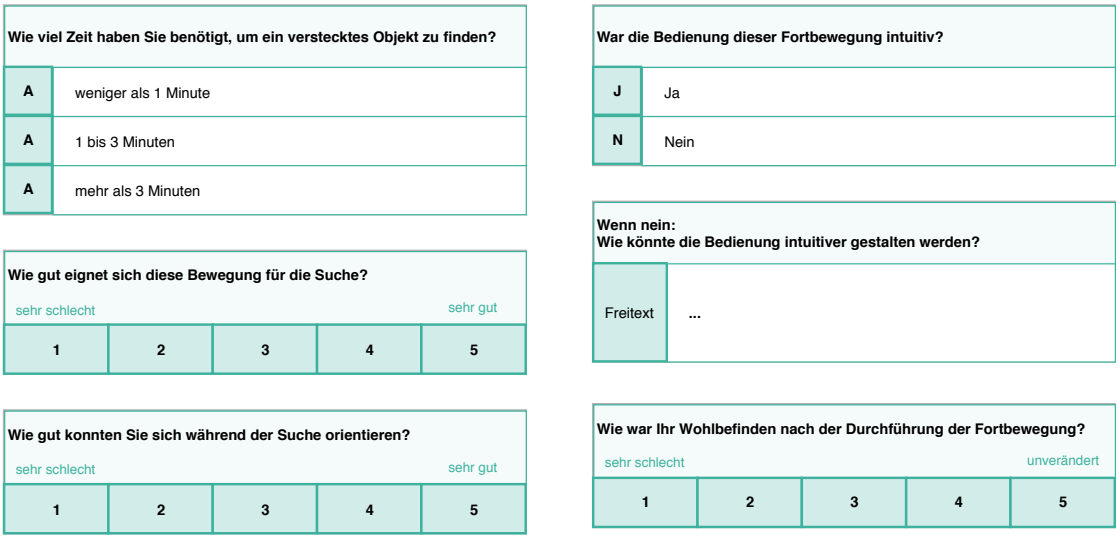


Abbildung 5.1: Sechs Fragen wurden für jeweils eine Navigationsart entworfen und in der Evaluierung von den Probanden ausgefüllt.

5.3 Auswertung

In der Auswertung wurden alle Ergebnisse von 15 Probanden, die in den Zeitraum von zwei Wochen gesammelt wurden, zusammengefasst. Im Folgenden werden die Ergebnisse in Form von Balkendiagrammen abgebildet und auf mögliche Rückschlüsse analysiert.

5.3.1 Frage 1

Die erste Frage bezieht sich auf die gemessene Zeit, die für die Suche mit der jeweiligen Bewegungsart benötigt wurde. Um die Ergebnisse miteinander vergleichen zu können, wurden Zeitabschnitte als Antwortmöglichkeiten angeboten. Mit dieser Unterteilung soll es möglich sein, eine Tendenz zu erkennen, die auf die Bedienung oder auch der Eignung für die Suche rückschließen lässt.

Aus dem Balkendiagramm (Abbildung 5.2) kann man deutlich entnehmen, dass die meisten Probanden weniger als eine Minute für die Suche benötigt haben. Auffällig dabei ist, dass drei Probanden für die Suchaufgabe mit der Teleportation länger als drei Minuten benötigt haben. Daraus könnte man schließen, dass Teilbereiche in der virtuellen Umgebung versehentlich übersprungen wurden, was bei der Teleportation häufiger vorkommen kann. Grundsätzlich wirken die Ergebnisse der Auswertung im Durchschnitt sehr positiv, was die Vermutung nahe

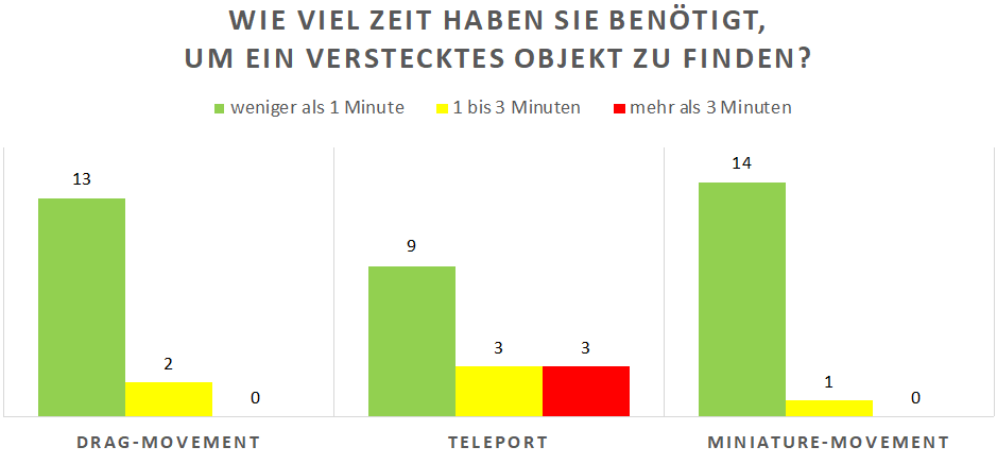


Abbildung 5.2: Auswertung der benötigten Zeit für die Suchaufgabe

legt, dass sich alle drei Bewegungsarten für eine schnelle Suchaufgabe eignen.

5.3.2 Frage 2

Bei der zweiten Frage sollte herausgefunden werden, wie gut sich die jeweilige Bewegungsart für die Suche eignet. Die Vermutung wiegt nah, dass die Ergebnisse dieser Auswertung mit den Ergebnissen der vorherigen Auswertung korrelieren.

Aus den Ergebnissen im Balkendiagramm (Abbildung 5.3) kann entnommen werden, dass die Bewegungsart Drag-Movement mit den vorherigen Ergebnissen der Auswertung korrelieren und somit die Vermutung bestätigen. Für die Fortbewegung mittels Teleport ist eine positive Tendenz zu erkennen, allerdings liegt der Durchschnitt bei ca. 3-Punkten und unterscheidet sich zu den Ergebnissen der vorherigen Auswertung. Aus diesem Grund lässt sich die Auswertung für Teleport nicht eindeutig auf die aufgestellte Vermutung zurückschließen. Dies könnte allerdings den gleichen Grund haben, wie bereits in der vorherigen Auswertung vermutet, dass die Teilbereiche in der virtuellen Umgebung übersprungen werden und somit die Suche erschwert wird.

Die Ergebnisse der Fortbewegung Miniature-Movement sind dagegen unterdurchschnittlich ausgefallen und lassen sich nicht auf die Vermutung zurückführen. Möglicherweise könnten diese Ergebnisse mit dem verwendeten Flugzeugmodell zusammenhängen, in dem alle Sitze in der Reihe identisch wirken und somit die Orientierung mindern, wenn ein Sprung in den nächsten Sektor durchgeführt wird. Aus diesem Grund könnte die nächste Auswertung für Miniature-Movement, im Bezug auf die Orientierung, ähnlich ausfallen.

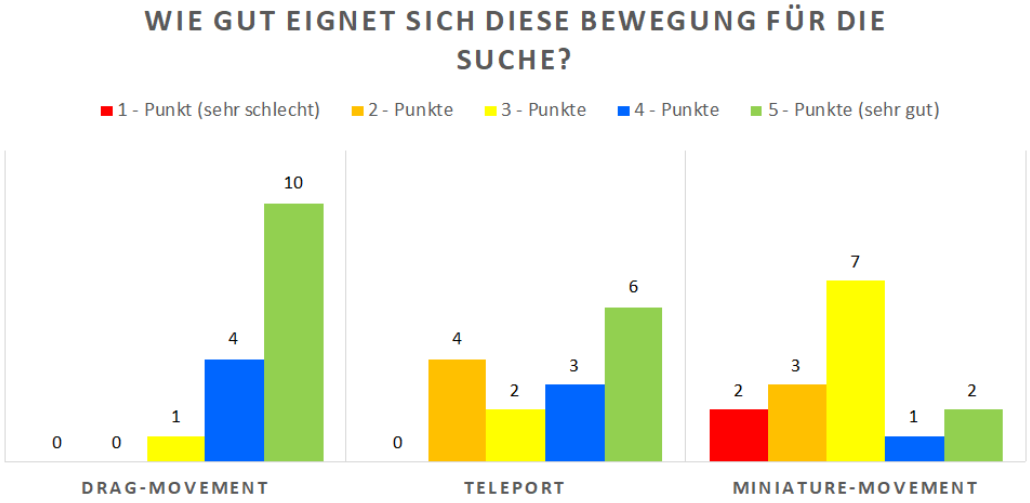


Abbildung 5.3: Auswertung über die Eignung der Navigationsarten für die Suche

5.3.3 Frage 3

Die Ergebnisse aus der Abbildung 5.4 entstanden aus der Frage: „Wie gut konnten Sie sich während der Suche orientieren?“
Je nach dem welche Bewegungsart verwendet wurde, sollten die Probanden ihre subjektiv wahrgenommene Orientierung bewerten. Die Orientierung kann durch die Bewegungsart bestärkt oder auch gemindert werden.
Ähnlich wie in der vorherigen Auswertung von Drag-Movement, sind auch die Ergebnisse dieser Auswertung sehr positiv ausgefallen. Vermutlich hängt die Einschätzung über die Eignung der Bewegungsart für die Suche und die Bewertung der subjektiv wahrgenommenen Orientierung miteinander zusammen. Die Ergebnisse der Bewegungsart Teleport sind von der durchschnittlichen Bewertung ähnlich mit der vorherigen Auswertung ausgefallen. Dagegen können die Ergebnisse von Miniature-Movement auf die vorherigen Auswertung nicht eindeutig zurückgeschlossen werden, da die Orientierung von den meisten Probanden besser als unterdurchschnittlich bewertet wurde. Möglicherweise konnten die Probanden mithilfe der Sektoren aus dem Miniatur-Modell die aktuelle Lage im virtuellen Raum besser einschätzen und durch die Aufsicht auf die miniaturisierte Umgebung, eine bessere Orientierung wahrnehmen.

5.3.4 Frage 4

Mit der Frage „War die Bedienung dieser Fortbewegung intuitiv?“ (siehe Abbildung 5.5) sollte die nicht funktionale Anforderung der Bedienbarkeit bewertet werden, die im Kapitel 3

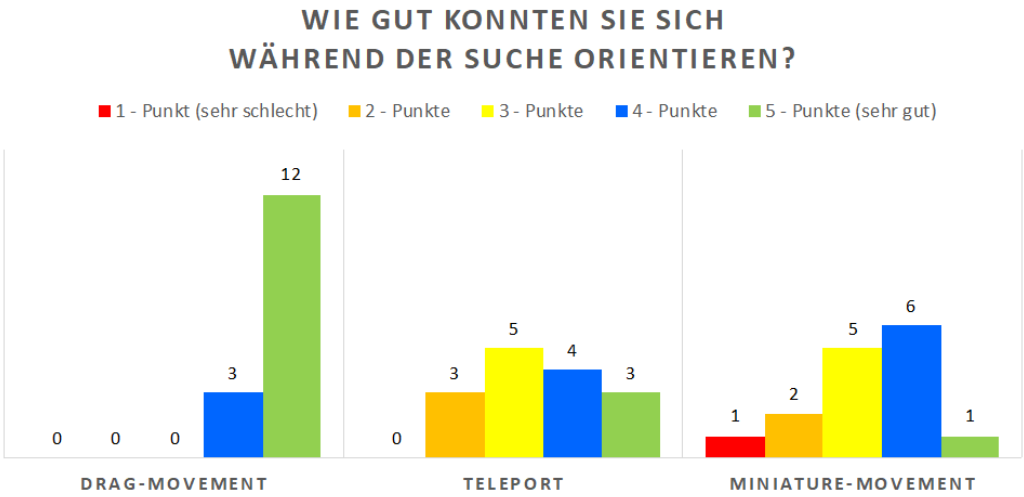


Abbildung 5.4: Im Balkendiagramm wird für jede Navigationsart die subjektiv empfundene Orientierung bewertet, die bei der Fortbewegung wahrgenommen wurde.

formuliert wurde. In dem Fragebogen wurde eine zusätzliche Option geboten, mögliche Verbesserungen in Form von Freitext zu beschreiben, wenn diese Frage verneint wurde.
Die Ergebnisse der beiden Bewegungsarten Drag-Movement und Teleport sind sehr positiv und eindeutig ausgefallen, was auf die intuitive Bedienung deuten lässt. Da diese Bewegungsarten von der Funktionalität sehr einfach aufgebaut wurden, erforderte die Bedienung auch keine aufwendige Lernkurve. Somit erfüllen Drag-Movement und Teleport die nicht funktionale Anforderung der intuitiven Bedienbarkeit.
Dagegen bietet die Auswertung von Miniature-Movement kein eindeutiges Ergebnis, was auf die Komplexität dieser Bewegungsart hinweist. Fast die Hälfte der Probanden bewerten die Bedienung als nicht intuitiv. Im Folgenden werden die Gründe für die negativen Ergebnisse, sowie die Verbesserungsvorschläge aus dem zur Verfügung gestellten Freitext zusammengefasst.

- Die Interaktion mit dem Miniatur-Modell erfordert zu viele Schritte und wirkt dadurch sehr kompliziert, was eine schnelle Bedienung erschwert.
  Als Verbesserungsvorschlag wurde häufig die Beidhändige Interaktion vorgeschlagen, bzw. die automatische Platzierung des Miniatur-Modells um die Fortbewegung mit nur einem Interaktionsschritt durchführen zu können.

- Das Miniatur-Modell hat eine zu hohe Abstraktionsstufe gegenüber dem realen Flugzeugmodell. Es ist nur schwer erkennbar, wo der vordere Teil des Miniatur-Modells sein soll.

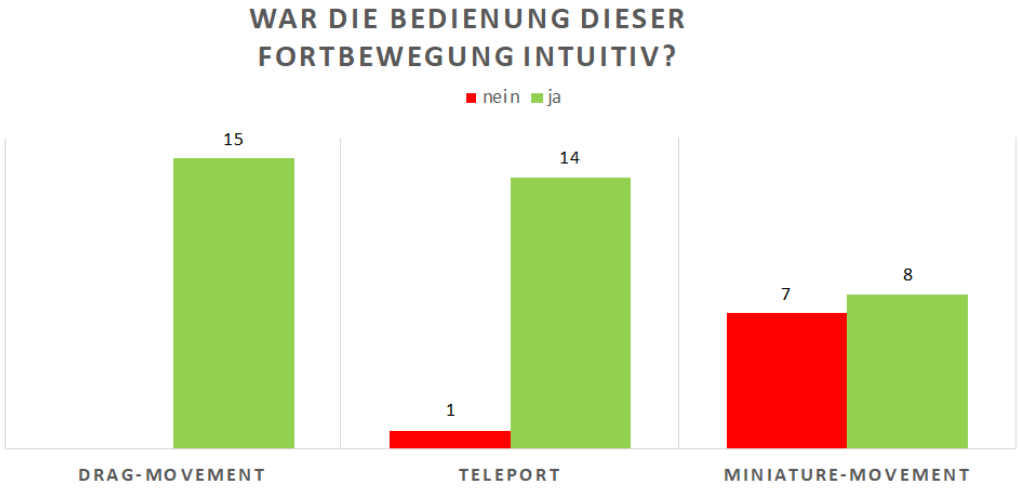


Abbildung 5.5: Das Balkendiagramm zeigt die Beurteilung der intuitiven Bedienbarkeit für jede Navigationsform

Eine Verbesserung wäre möglich, indem die Flügel vom Rumpf und vom Heck des Flugzeugmodells abstrahiert und miniaturisiert zum bestehenden Miniatur-Modell hinzugefügt werden. Dadurch hätte der Nutzer einen klaren Bezug zu der Ausrichtung der Miniatur.

Somit konnte die nicht funktionale Anforderung aufgrund der Komplexität in der Interaktion und einen zu geringen Grad von LOD (Level-of-Detail) nicht erreicht werden.

5.3.5 Frage 5

Mit der Frage aus der Abbildung 5.6 sollte herausgefunden werden, ob Anzeichen einer Cybersickness bei der Fortbewegung mit der jeweiligen Navigationsform aufgetreten sind. Das Wohlbefinden kann nach der Durchführung als unverändert bewertet werden, wenn keine Anzeichen einer Cybersickness wahrgenommen wurden und wiederum kann eine schlechte Bewertung auf starke Symptome wie z.B. Übelkeit hinweisen.
Grundsätzlich sind die Ergebnisse aller Navigationsformen sehr positiv ausgefallen und weisen darauf hin, dass so gut wie keine Anzeichen einer Cybersickness entstanden ist. Die Navigationsformen Teleport und Miniature-Movement haben im Durchschnitt eine sehr geringe bis keine Anzeichen auf Cybersickness. Die Manipulation der Nutzerposition erfolgt in beiden Navigationsformen auf die gleiche Weise, was möglicherweise auch den Grund für die ähnlichen Ergebnisse haben könnte. Zwar zeigen die Ergebnisse von Drag-Movement höhere

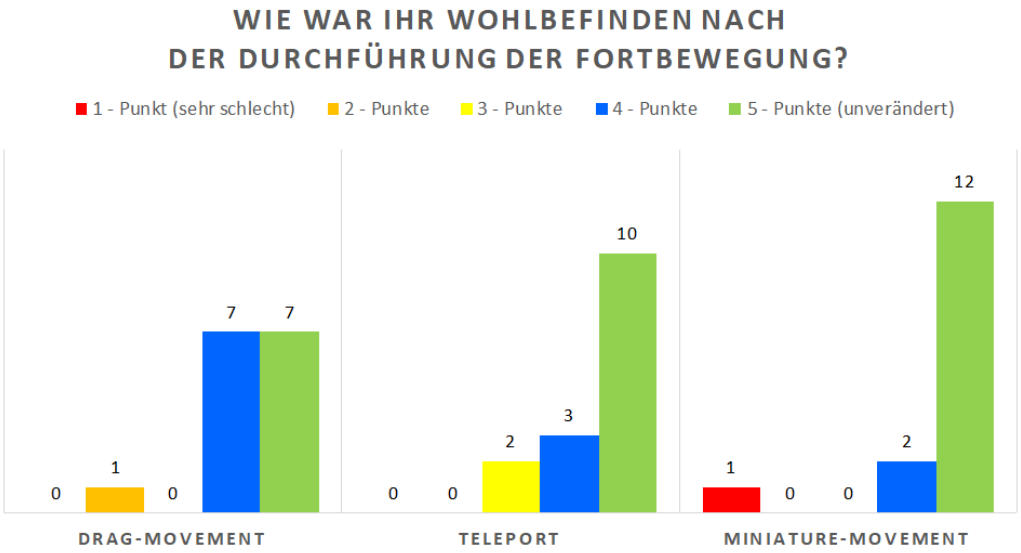


Abbildung 5.6: In dieser Auswertung wird die wahrgenommene Cybersickness für jede Navigationsform bewertet, die nach der jeweiligen Durchführung entstanden ist.

Anzeichen einer Cybersickness im Vergleich zu Teleport und Miniature-Movement, diese sind aber nur geringfügig. Einige Benutzer haben in der Durchführung davon berichtet, dass die Fortbewegung mit Drag-Movement sich so anfühlt, als ob das Flugzeug unter den Füßen bewegt wird. Möglicherweise wurde mit dem automatisch eingeblendeten Hilfsgitter, welches dem Nutzer in der virtuellen Umgebung das reale Spielfeld repräsentiert, der Eindruck einer Umgebungsbewegung vermittelt, was bei einigen Probanden zu geringen Anzeichen einer Cybersickness geführt hat. Dieses Phänomen könnte mit einer größeren Testumgebung gelöst werden, so dass das Hilfsgitter nicht frühzeitig eingeblendet wird.

5.3.6 Frage 6

Mit der Frage „Welche Fortbewegung bevorzugen Sie zum Überqueren größerer Strecken im virtuellen Raum“ sollte die präferierte Navigationsform herausgefunden werden, die sich auch auf anderen bzw. größeren virtuellen Umgebungen anwenden lässt(Abbildung 5.7). Die Vermutung liegt nahe, dass die Navigationsformen Teleport und Miniature-Movement in den Ergebnissen besser ausfallen, als die Fortbewegung mit Drag-Movement, da Teleport und Miniature-Movement zum Überquere größerer Distanzen konstruiert wurden.
Wie bereits vermutet, sind die Ergebnisse der Fortbewegung Teleport eindeutig ausgefallen.

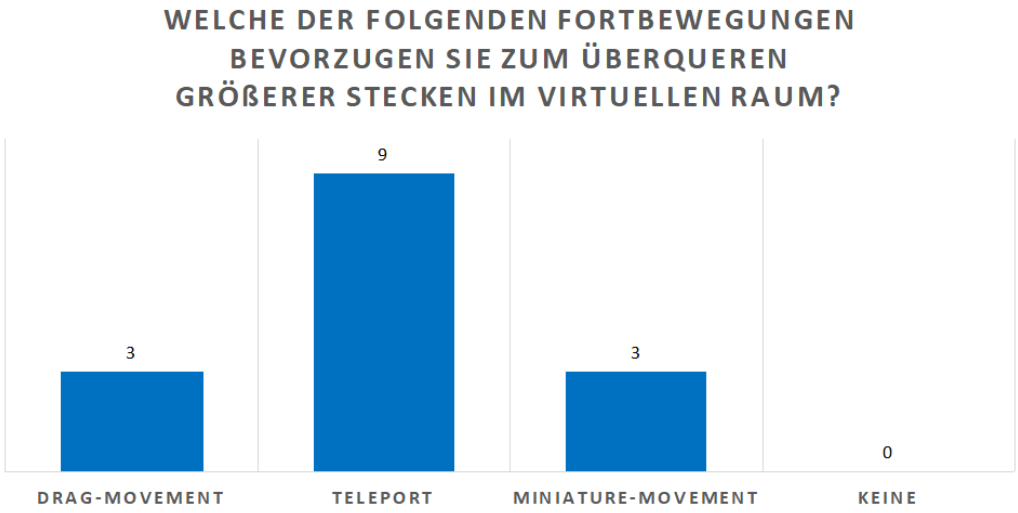


Abbildung 5.7: In diesem Balkendiagramm werden Ergebnisse einer Abstimmung gezeigt, die eine präferierte Navigationsform zum Überqueren größerer Distanzen bestimmen soll.

Diese Navigationsform wurde als eine präferiert, um eine größere Distanz im virtuellen Raum zu überqueren. Allerdings ist das Ergebnis von Miniature-Movement unerwartet negativ ausgefallen und wurde von der Anzahl der Bewertungen mit der Fortbewegung Drag-Movement identisch bewertet. Möglicherweise könnten die Ergebnisse der Auswertung aus dem Unterabschnitt 5.3.4 die Gründe für dieses unerwartete Ergebnis sein. Vermutlich wären die Ergebnisse für Miniature-Movement besser ausgefallen, würde man die vorgeschlagenen Verbesserungen in die Navigationsform einbauen.

5.3.7 Frage 7

Die letzte Frage richtet sich an die verwendete Bewegungsart bzw. eine Kombination dessen, die im letzten Schritt der Durchführung vom Nutzer frei gewählt wurde (siehe Abbildung 5.8). In der Auswertung soll ermittelt werden, ob nur eine bestimmte Navigationsart oder auch eine Kombination für die Fortbewegung genutzt wird, die auch in größeren bzw. komplexeren virtuellen Umgebungen verwenden lässt. Auf Grundlage der letzten Auswertung liegt die Vermutung nahe, dass eine Kombination mit der Fortbewegung Teleport verwendet wurde, da diese zum Überqueren größerer Strecken sich sehr gut eignet.
Betrachtet man nur die Ergebnisse der einzeln verwendeten Navigationsformen, so stellt man fest, dass die Bewegungsart Teleport stark dominiert, ähnlich wie in der letzten Auswertung.

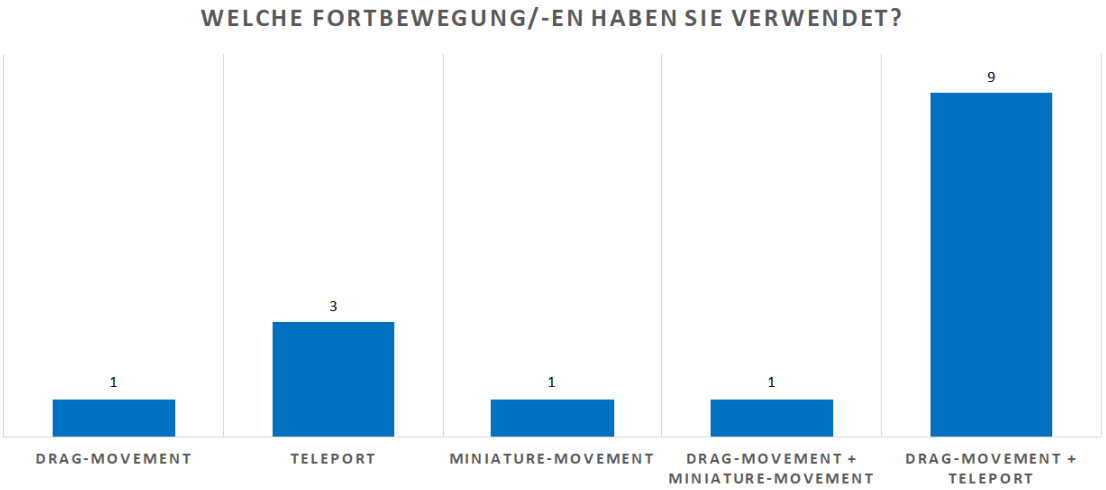


Abbildung 5.8: Das Balkendiagramm zeigt die Ergebnisse der gewählten Navigationsart/en, die im letzten Schritt der Durchführung vom Nutzer frei gewählt wurden.

Die Ergebnisse der kombinierten Navigationsformen zeigen deutlich, dass die Bewegungsart Teleport in Kombination mit Drag-Movement für die Fortbewegung im virtuellen Raum am besten eignen. Möglicherweise wurde das Miniature-Movement nicht in Kombination mit Teleport verwendet, da beide Navigationsformen zum Überqueren für größere Distanzen zuständig sind und das Miniature-Movement zu komplex in der Bedienung erscheint. Eine weiterer Grund für die ausgeschlossene Kombination könnte die Genauigkeit der Bewegungsübertragung sein. Die Fortbewegung mit Teleport ermöglicht es zwar dem Nutzer seine Position mit einem gebogenen Strahl zu bestimmen, eignet sich aber am besten für größere Sprünge, da der Kontext zu der Umgebung bewusst durch das kurzzeitige Aus- und Einblenden der Umgebung verloren geht, um die Anzeichen einer Cybersickness zu minimieren. Da die Fortbewegung mit Teleport bereits das Überbrücken größerer Strecken mit intuitiver Bedienung abdeckt, wird eine Kombination mit Miniature-Movement, welche eine ungenauere Bewegungsübertragung und eine komplexere Bedienung im Gensatz zu Teleport bietet, keinen Mehrwert für die Navigation bieten.
Auf Grundlage der gesammelten Ergebnisse aus der Evaluierung kann auf eine Akzeptanz der neuen Bewegungsform Drag-Movement geschlossen werden. Da die Suchaufgaben im großen Raum durchgeführt wurden, ist es nicht verwunderlich, dass die Fortbewegungsarten mit denen man größere Strecken überwinden kann, bessere Ergebnisse liefern. Für die Aufgabenstellung in dieser Domäne wurde eine Kombination aus der entwickelten Form und der Teleportation gewählt, die es dem Nutzer erlauben gezielte Bewegungen für kurze Strecken durchzuführen

und auch größere Distanzen schnell zu überwinden.

5.4 Zusammenfassung

In diesem Kapitel wurde auf Grundlage der entwickelten Navigationsformen, die Ziele in Abschnitt 5.1 definiert und in einem Versuchsaufbau die Aufgaben und Bedingungen der Evaluation im Abschnitt 5.2 beschrieben. Im Abschnitt 5.3 wurden die gesammelten Ergebnisse in Form von Balkendiagrammen visualisiert und ausgewertet. Die Auswertung zeigt, dass sich eine Kombination aus zwei Navigationsformen (Teleport und Drag-Movement) für eine gezielte Suche in einem Flugzeugmodell, welches mit Sektoren auf der Bodenfläche erweitert wurde, am besten eignet. Die Fortbewegung mit Miniature-Movement hat sich als nicht praktikabel herausgestellt, was vermutlich daran liegt, dass die einhändige Interaktion zu viele Schritte erfordert, um eine schnelle Bewegung durchführen zu können. Eine Erweiterung auf eine beidhändige Interaktion lässt bessere Akzeptanz vermuten, die allerdings mit einer erneuten Evaluierung überprüft werden muss.

6 Zusammenfassung und Ausblick

In dieser Arbeit wurde eine vollständige Lösung für die Navigation in einem realen Flugzeugmodell in VR mit drei unterschiedlichen Navigationskonzepten entwickelt. Jedes der entwickelten Konzepte wurde zuvor konzeptuell aufgebaut und in die Entwicklungs- und Echtzeitumgebung Unity-3D integriert, so dass diese Bewegungsformen mit der Verwendung von HTC-Vive vollständig und fehlerfrei funktionieren. Abschließend wurde mit einer kleinen Gruppe von Benutzern versucht, die aufgestellten Annahmen zu überprüfen.
In Kapitel 2 wurden die unterschiedlichen Navigationskategorien vorgestellt, aus denen sich drei Varianten für die Umsetzung in dieser Arbeit ergeben haben. Aus der Diskussion der Interaktions-Konzepte wurde schnell klar, dass eine UI für diese Art der Applikation von der Kategorie Diegetic sein muss. Außerdem wurden die theoretischen Grundlagen für die Interaktionsformen in VR, sowie das Datenmodell für 3D-Szenen gelegt. In Kapitel 3 wurden die 3 Navigationsarten ausführlich spezifiziert und die funktionalen, sowie nicht funktionalen Anforderungen formuliert, die für diese Arbeit relevant sind. In Kapitel 4 wurde der Entwurf für das zu entwickelnde System vorgestellt und die Anwendungsdomäne für die Umsetzung aufbereitet. Da das Programmiermodell von Unity-3D gewisse Einschränkungen mit sich bringt, wurde der Softwareentwurf dementsprechend abgebildet. Abschließend wurde die Umsetzung der einzelnen Navigationskonzepte vorgestellt. Kapitel 5 hat gezeigt, dass die neu entwickelte Navigationsform *Drag-Movement*, in Kombination mit *Teleport*, als eine sehr akzeptierte und brauchbare Lösung für die Fortbewegung in großen virtuellen Umgebungen eignen. Die Ergebnisse der Evaluierung haben aber auch gezeigt, dass die umgesetzte Navigationsform *Miniature-Movement* von der Bedienung zu komplex aufgebaut wurde. Das Feedback der Probanden lässt darauf schließen, dass die beidhändige Interaktion einen großen Schritt zur Verbesserung der intuitiven und schnellen Bedienung beitragen kann.

Bezüglich der nicht funktionalen Anforderung Performanz und der Konfigurierbarkeit der Navigationsformen, wären folgende Optimierungen wünschenswert. Dazu gehört zum einen die Verbesserung der Konfigurationsverwaltung. Möglich wäre dies, indem die konfigurierbaren Parameter außerhalb von Unity gehalten werden - denkbar wäre die Auslagerung in eine

Datei. Zum anderen könnte die Performanz der entwickelten Anwendung optimiert werden. Mit der Auslagerung der konfigurierbaren Parameter wird der Unity-3D Editor nicht mehr für die Konfiguration benötigt. Unity-3D bietet eine Betriebssystem optimierte Kompilierung an, um die Performanz der Anwendung zu steigern. Hierdurch gehen unter anderem Entwicklungsfunktionen, sowie die Konfiguration im Unity-3D Editor verloren.

Mit dem gesammelten Feedback in der Evaluierung könnte die Interaktion für die Navigationsform *Minature-Movement* ebenfalls erweitert werden, indem die Interaktionsschritte minimiert werden. Denkbar wäre eine Umstellung auf die beidhändige Interaktion, um die Bedienung zu vereinfachen. Es wäre damit zu rechnen, dass die Ergebnisse in einer zukünftigen Evaluierung, durch die Erweiterung der Interaktion besser ausfallen.
Weiterhin wäre es denkbar, die drei entwickelten Navigationsformen auch auf komplexere Anwendungsdomänen zu übertragen, wie z.B. U-Bote oder auch Kreuzfahrtschiffe. Diese sind deutlich komplexer in der Raumaufteilung, da sie sich über mehrere Ebenen verteilen. Um auf das Beispiel des Kreuzfahrtschiffes einzugehen, würde sich die Fortbewegung mittels *Miniture-Movement* sehr gut zum Überqueren der unterschiedlichen Ebenen eignen. Dagegen könnten lange Flure im Kreuzfahrtschiff mit der gewohnten Bewegungsform *Teleport* überbrückt werden und mit der präzisen Zieh- und Abstoßbewegung von *Drag-Movement* die einzelnen Kabinen erkundet werden.
Ähnliche Anwendungsfälle ergeben sich auch für Krankenhäuser, Pflegeheime aber auch Szenarien einer Kanalwartung, so dass die Arbeit nicht nur speziell für Navigation in Flugzeugkabinen tragfähig wäre.

Literaturverzeichnis

[WikiBewegungssehen 2018] *Bewegungssehen.* https://de.wikipedia.org/wiki/Bewegungssehen. Juli 2018. – Aufgerufen am 29.07.2018

[Andrews 2010] Andrews, Marcus: *Game UI Discoveries: What Players Want.* https://www.gamasutra.com/view/feature/4286/game/_ui/_discoveries/_what/_players/_.php. Februar 2010. – Aufgerufen am: 28.06.2018

[Bendel 2018] Bendel, Oliver: *Virtuelle Realität.* https://wirtschaftslexikon.gabler.de/definition/virtuelle-realitaet-54243/version-277293. Februar 2018. – Aufgerufen am 24.07.2018

[Bozgeyikli u. a. 2016a] Bozgeyikli, Evren ; Raij, Andrew ; Katkoori, Srinivas ; Dubey, Rajiv: Locomotion in Virtual Reality for Individuals with Autism Spectrum Disorder. In: *Proceedings of the 2016 Symposium on Spatial User Interaction.* New York, NY, USA : ACM, 2016 (SUI '16), S. 33–42. – URL http://doi.acm.org/10.1145/2983310.2985763. – ISBN 978-1-4503-4068-7

[Bozgeyikli u. a. 2016b] Bozgeyikli, Evren ; Raij, Andrew ; Katkoori, Srinivas ; Dubey, Rajiv: Point & Teleport Locomotion Technique for Virtual Reality. In: *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play.* New York, NY, USA : ACM, 2016 (CHI PLAY '16), S. 205–216. – URL http://doi.acm.org/10.1145/2967934.2968105. – ISBN 978-1-4503-4456-2

[Dörner u. a. 2014] Dörner, Ralf ; Broll, Wolfgang ; Grimm, Paul ; Jung, Bernhard: *Virtual und augmented reality (VR/AR): Grundlagen und Methoden der Virtuellen und Augmentierten Realität.* Springer-Verlag, 2014

[Dörner u. a. 2013] Dörner, Ralf ; Geiger, Christian ; Oppermann, Leif ; Paelke, Volker: Interaktionen in Virtuellen Welten. In: *Virtual und Augmented Reality (VR/AR).* Springer, 2013, S. 157–193

[Fernandes und Feiner 2016] Fernandes, Ajoy S. ; Feiner, Steven K.: Combating VR sickness

through subtle dynamic field-of-view modification. In: *3D User Interfaces (3DUI), 2016 IEEE Symposium on* IEEE (Veranst.), 2016, S. 201–210

[Förtsch 2016] Förtsch, Michael: *So wollen Entwickler und Forscher die VR-Übelkeit besiegen.* https://www.wired.de/collection/life/so-wollen-entwickler-und-forscher-die-vr-uebelkeit-besiegen. August 2016. – Aufgerufen am 21.07.2018

[Gieselmann 2017] Gieselmann, Hartmut: *Bewegungen in Virtual Reality jenseits des Teleporters.* https://www.heise.de/newsticker/meldung/Bewegungen-in-Virtual-Reality-jenseits-des-Teleporters-3637164.html. Februar 2017. – Aufgerufen am: 13.07.2018

[James 2015] James, Paul: *Sightline Dev Wants Your Feedback on Experimental Holosphere Locomotion.* https://www.roadtovr.com/sightline-dev-wants-your-feedback-on-experimental-holosphere-locomotion/. Dezember 2015. – Aufgerufen am 17.07.2018

[Jehan 2017] Jehan: *Design: Exploring Teleportation and Locomotion in VR.* http://jehansgamedesignblog.blogspot.com/2017/03/design-exploring-teleportation-and.html. März 2017. – Aufgerufen am 17.07.2018

[Kruijff u. a. 2015] Kruijff, Ernst ; Riecke, Bernhard ; Trekowski, Christina ; Kitson, Alexandra: Upper body leaning can affect forward self-motion perception in virtual environments. In: *Proceedings of the 3rd ACM Symposium on Spatial User Interaction* ACM (Veranst.), 2015, S. 103–112

[Kühhirt und Rittermann 2005] Kühhirt, Uwe ; Rittermann, Marco: *Interaktive audiovisuelle Medien.* Hanser, 2005

[Larsson u. a. 2004] Larsson, Pontus ; Västfjäll, Daniel ; Kleiner, Mendel: Perception of self-motion and presence in auditory virtual environments. In: *Proceedings of seventh annual workshop presence*, 2004, S. 252–258

[Nabiyouni u. a. 2015] Nabiyouni, Mahdi ; Saktheeswaran, Ayshwarya ; Bowman, Doug A. ; Karanth, Ambika: Comparing the performance of natural, semi-natural, and non-natural locomotion techniques in virtual reality. In: *3D User Interfaces (3DUI), 2015 IEEE Symposium on* IEEE (Veranst.), 2015, S. 3–10

[Novak 2011] Novak, Jeannie: *Game development essentials: Game Interface Design.* Cengage Learning, 2011

[Preim und Dachselt 2015] Preim, Bernhard ; Dachselt, Raimund: *Interaktive Systeme: Band 2: User Interface Engineering, 3D-Interaktion, Natural User Interfaces.* Springer-Verlag, 2015

[Razzaque u. a. 2001] Razzaque, Sharif ; Kohn, Zachariah ; Whitton, Mary C.: Redirected walking. In: *Proceedings of EUROGRAPHICS* Bd. 9 Citeseer (Veranst.), 2001, S. 105–106. – Vergleichbare Arbeit

[Riecke u. a. 2015] Riecke, Bernhard E. ; Freiberg, Jacob B. ; Grechkin, Timofey Y.: Can walking motions improve visually induced rotational self-motion illusions in virtual reality? In: *Journal of vision* 15 (2015), Nr. 2, S. 3–3

[Schäfer 2017] Schäfer, Dimitrij: *Untersuchung der Bewegungsmethoden im virtuellen Raum mit Bezug auf Cybersickness.* 2017

[Steinicke u. a. 2009] Steinicke, Frank ; Bruder, Gerd ; Hinrichs, Klaus ; Jerald, Jason ; Frenz, Harald ; Lappe, Markus: Real walking through virtual environments by redirection techniques. In: *JVRB-Journal of Virtual Reality and Broadcasting* 6 (2009), Nr. 2

[Stoakley u. a. 1995] Stoakley, Richard ; Conway, Matthew J. ; Pausch, Randy: Virtual reality on a WIM: interactive worlds in miniature. In: *Proceedings of the SIGCHI conference on Human factors in computing systems* ACM Press/Addison-Wesley Publishing Co. (Veranst.), 1995, S. 265–272. – Konzept für die Navigation ,Orientierung

[Stonehouse 2011] Stonehouse, Anthony: *User interface design in video games.* http://yuting-jiang.blogspot.com/2011/11/user-interface-design-in-video-games.html. November 2011. – Aufgerufen am: 03.07.2018

[TvTropes 2013] TvTropes: *Diegetic Interface.* https://tvtropes.org/pmwiki/pmwiki.php/Main/DiegeticInterface. Juni 2013. – Aufgerufen am: 03.07.2018

[Unity-Technologies] Unity-Technologies: *User Interfaces for VR.* https://unity3d.com/de/learn/tutorials/topics/virtual-reality/user-interfaces-vr. – Aufgerufen am: 28.06.2018

[Walker 2013] Walker, James: Redirected Walking in Virtual Environments. In: *Michigan Technological University* (2013)

[ZEALOUSYS 2011] ZEALOUSYS: *Unity Virtual Reality: Types Of UI Placements.* https://www.zealousys.com/blog/unity-virtual-reality-types-ui-placements/. Mai 2011. – Aufgerufen am: 03.07.2018

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 13. August 2018 Dimitri Meier