UNIVERSITÄT
KOBLENZ · LANDAU

Fachbereich 4: Informatik

Grenzen und Möglichkeiten der HoloLens im Bereich Gaming

Bachelorarbeit

zur Erlangung des Grades Bachelor of Science (B.Sc.)
im Studiengang Computervisualistik

vorgelegt von

Alexander James Deeming

Erstgutachter:	Prof. Dr.-Ing. Stefan Müller (Institut für Computervisualistik, AG Computergraphik)
Zweitgutachter:	Anna Katharina Hebborn, M.Sc. (Institut für Computergrafik, Mitarbeiterin der Arbeitsgruppe Müller)

Koblenz, im November 2017

Erklärung

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

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

.....

(Ort, Datum) (Unterschrift)

Institut für Computervisualistik
AG Computergraphik
Prof. Dr. Stefan Müller
Postfach 20 16 02
56 016 Koblenz
Tel.: 0261-287-2727
Fax: 0261-287-2735
E-Mail: stefanm@uni-koblenz.de

UNIVERSITÄT KOBLENZ · LANDAU

Fachbereich 4: Informatik

Aufgabenstellung für die Bachelorarbeit
Alexander James Deeming
(Mat. Nr. 213200345)

Thema: Grenzen und Möglichkeiten der HoloLens im Bereich Gaming

Die technische Entwicklung im Bereich Augmented Reality (AR) ist in den letzten Jahren deutlich gestiegen. Es existieren bereits eine Vielzahl an Systemen, mit beeindruckenden Fähigkeiten. Eine der neusten und interessantesten Entwicklungen stellt die AR-Brille *HoloLens* von Microsoft dar.

Vor allem im Bereich des AR-Gaming äußert sich der Wunsch nach grenzenloser Kombination virtueller und realer Welten. Hierbei sollen virtuelle Objekte nicht nur korrekt in der realen Welt dargestellt werden, sondern der Benutzer soll mit diesen interagieren können. Grundlegend stellt sich somit die Herausforderung die Grenzen üblicher Computerspiele durch die visuelle Erweiterung und eine besonders intuitive Form der Interaktion zu überwinden. Bislang blieb die Interaktion im Bereich AR aufgrund der hohen Anforderungen an die Umsetzung oder technischer Einschränkungen meist auf der Strecke. Einsatzfähige AR-Brillen, wie die *HoloLens*, eröffnen neue Möglichkeiten zur Interaktion. Derzeit finden sich überwiegend Smartphone Applikationen, welche die Interaktion durch die Belegung der Hände stark einschränken.

Diese Bachelorarbeit beschäftigt sich mit verschiedenen Möglichkeiten zur Interaktion mit virtuellen, dreidimensionalen Objekten in der realen Umgebung des Nutzers. Ziel dieser Arbeit ist die Konzeption und prototypische Realisierung einer spielerischen Applikation für die *HoloLens*, die verschiedene Konzepte zur Interaktion beinhaltet.

Des Weiteren soll eine Evaluation anhand von Probanden zeigen, ob diese Möglichkeiten eine geeignete Interaktion zwischen Anwender und virtuellen Komponenten ermöglicht. Insbesondere sollen hierbei die Fragen beantwortet werden, ob sich die Interaktionsmöglichkeiten der *HoloLens* in der Praxis behaupten können und welche Arten der Interaktion dem Nutzer am meisten zusprechen.

Die inhaltlichen Schwerpunkte der Arbeit sind:

1. Recherche über bestehende Interaktionskonzepte im Bereich AR
2. Einarbeitung in das *HoloLens* SDK
3. Konzeption einer spielerischen Applikation mit verschiedenen Möglichkeiten zur Interaktion
4. Prototypische Implementation der Applikation
5. Evaluation anhand von Probanden
6. Dokumentation der Ergebnisse

Koblenz, 24.05.2017

– Alexander James Deeming – – Prof. Dr. Stefan Müller –

Zusammenfassung

Diese Arbeit befasst sich mit verschiedenen Möglichkeiten zur Interaktion mit dreidimensionalen, virtuellen Objekten in der realen Umgebung des Nutzers. Im Vordergrund stehen Interaktionsmöglichkeiten, welche durch neue *AR-Technologien* aufkommen.

Dazu wird ein spielerischer Prototyp einer Applikation für die von *Microsoft* entwickelte *HoloLens* konzipiert und implementiert. Der Prototyp des Spiels besteht aus drei Phasen. Die erste Phase ist die Aufnahme der realen Umgebung des Nutzers. In der zweiten Phase kann der Nutzer die reale Umgebung mit der Hilfe von virtuellen Objekten erweitern. In der dritten Phase muss der Nutzer einen virtuellen *Avatar* durch die reale Umgebung navigieren.

Die Interaktionsmöglichkeiten der *HoloLens* wie *Gaze, Gesture* und *VoiceInput* werden in den Kategorien *Menüführung*, *Positionierung von virtuellen Objekten im dreidimensionalen Raum* und *Steuerung eines Avatars* einer Evaluation unterzogen.

Abstract

This thesis deals with the exploration of different interaction possibilities for three-dimensional, virtual objects in a real environment. The focus lies especially on interaction possibilities from new *AR-technologies*.

A playful prototype of an application for *Microsofts HoloLens* will be designed and implemented. The prototype consists of three parts. The first part is the scan-process of the real environment of the user. In the second part the user can augment the real environment with three-dimensional, virtual objects. In the third part the user is supposed to navigate a virtual *avatar* through the real environment.

The interaction possibilities of the *HoloLens* like *Gaze, Gesture* and *VoiceInput* will be evaluated in the following categories *menu navigation, positioning of three-dimensional objects in a real environment* and *controlling an avatar*.

Inhaltsverzeichnis

6 Anhang E

1 Grundlagen

Dieses Kapitel beschäftigt sich mit dem Begriff der *Augmented Reality* (*AR*), dabei werden seine Einsatzgebiete sowie einige Technologien veranschaulicht.

1.1 Augmented Reality

AR befasst sich mit der Präsentation von virtuellen Objekten in einer realen Umgebung. Die reale Umgebung des Nutzers wird durch eine *Augmented Reality Technologie* (Abschnitt 1.2) mit virtuellen Objekten überlagert. Der Begriff *AR* ist nicht zu verwechseln mit dem Begriff *Virtual Reality (VR)*. In der *VR* besteht das Sichtfeld des Nutzers nicht, wie bei *AR* , aus dem realen Umfeld, überlagert mit virtuellen Objekten, sondern vollständig aus einer virtuellen Umgebung. Milgram definierte das sogenannte *conitnuum of real-to-virtual environments*, um den Zusammenhang, als auch den Unterschied zwischen *AR* und *VR* aufzuzeigen [1] (Abbildung 1).

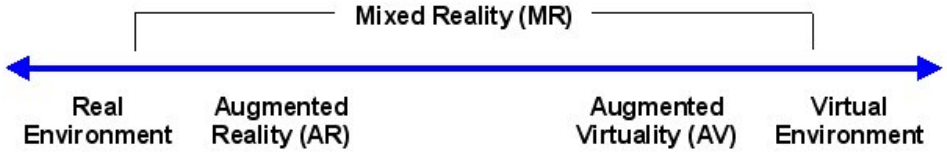

Abbildung 1: Paul Milgram erstellte dieses Mixed-Reality-Continuum bereits im Jahre 1994. (Quelle: [2])

Real Environment ist die reale Umgebung, welche den bekannten, physikalischen Gesetzen unterliegt.

AR ist die Überlagerung der realen Welt mit virtuellen Objekten, wobei versucht wird diesen Objekten die physikalischen Eigenschaften der realen Umgebung zu geben. Rollt ein virtueller Stein beispielsweise an den Rand eines realen Tisches, so sollte dieser auch von dem Tisch fallen und auf dem Boden landen.

Augmented Virtuality ist die Fähigkeit einer virtuellen Umgebung mittels realer Objekte, wie z.B. den Händen des Nutzers, manipuliert zu werden. Dabei geht es bei diesem Begriff nicht hauptsächlich um die Visualisierung der realen Objekte in der virtuellen Welt, sondern um die Interaktion zwischen realen und virtuellen Objekten.

Virtual Environment ist die virtuelle Welt, welche eine Nachahmung von realen, physikalisch unterworfenen Umgebungen oder fiktionalen, physikalisch unterworfenen Umgebungen sein kann.

Ein weiterer Begriff, welcher für die *AR* und *VR* eine zentrale Rolle spielt, ist der Begriff *Immersion*. Bei der *Immersion* geht es um das Eintauchen des Nutzers in die ihm dargestellte Umgebung und dessen Akzep-

tanz. In Bezug auf *AR* ist eine perfekte *Immersion* eine Technologie, welche es dem Nutzer nicht erlaubt den Unterschied zwischen einem realen und einem virtuellen Objekt zu erkennen. Somit wird das virtuelle Objekt als Teil des realen Raumes akzeptiert.

Die Geschichte von *Virtual Reality*, und damit auch der Anfang von *AR* beginnt im Jahr 1968, in dem Ivan Sutherland den ersten *Head Mounted Display (HMD)* baute und somit die erste Technologie entwarf, mit der ein Mensch in einer virtuellen Welt eintauchen konnte. Ein *HMD* ist ein Bildschirm, welcher auf dem Kopf und somit vor den Augen des Nutzers platziert wird. Während der erste *HMD* noch so schwer war, dass er an der Decke befestigt werden musste, wurden im Laufe der Zeit immer leichtere und leistungsstärkere Technologien entwickelt. Die unterschiedlichen Technologien und einige Beispiele werden in dem nächsten Abschnitt 1.2 aufgezeigt.

Das Einsatzgebiet von *AR* kann sehr breit gefächert sein. *AR* könnte z.B. in der Bildung eine neue, intuitive Art der Informationsvermittlung darstellen, oder in medizinischen Bereichen einen Chirurg visuell unterstützen [3][4]. Auch für industrielle Zwecke ist *AR* sehr interessant. Bei der Überwachung oder Wartung von großen Maschinen, kann *AR* eine nützliche Art der Informationsdarstellung sein. Die größte Schwierigkeit *AR* in solchen Bereichen unterzubringen, liegt darin die Applikationen so *benutzerfreundlich* wie möglich zu gestalten, um dem Nutzer eine einfache und intuitive Bedienung zu ermöglichen [3].

Jedoch muss auch die Kehrseite neuer Technologien, mögliche Auswirkungen und Gefahren für den Nutzer, untersucht werden. Bei der Verwendung einer solchen Technologie könnte der Nutzer abgelenkt werden und die Aufmerksamkeit auf seine Umgebung vernachlässigen. Um solche Situationen zu umgehen, müssen sich früh genug Gedanken über eventuelle Gefahrenquellen gemacht werden [5].

1.2 Ausgabegeräte

In diesem Kapitel werden einige Verfahrensweisen und Technologien, welche hinter dem Begriff *AR* stehen aufgezeigt. Zunächst kann gesagt werden, dass jeder Bildschirm in Zusammenhang mit einer Applikation, welche die reale und die virtuelle Welt zusammenbringt, eine Art von *AR* ist. Die Technologien lassen sich grob in drei Arten aufteilen.

Head Wearable Diese Art der Technologie wird von dem Nutzer auf dem Kopf getragen, wobei sich der Bildschirm, oder auch ein Bildschirm für jedes Auge, direkt vor den Augen des Nutzers befindet. Dabei kann das Bild für den Nutzer auf zwei verschiedene Arten dargestellt werden. Im Falle des *video see-through*, wird ein Video mittels einer Kamera in ungefährer

Augenhöhe aufgenommen, das Video wird mit virtuellen Objekten überlagert und auf einem undurchsichtigen Display ausgegeben.

Beim *optical see-through* ist das Display transparent, sodass der Nutzer die reale Welt weiterhin sieht. Die Vorteile des *optical see-through* gegenüber dem *video see-through* liegen z.B. darin, dass es durch den transparenten Bildschirm realer wirkt und es bei einem Fehler des Gerätes kein schwarzes Bild gibt. Jedoch ist z.B. die Platzierung von Objekten sowie die Überlagerung der virtuellen Objekte beim *video see-through* besser.

Handheld Bei *Handheld* Geräten handelt es sich meist um Tablets oder Smartphones. Applikationen für mobile Geräte, wie zum Beispiel *Face Swap* [6], gehören zur Kategorie *AR* , da sie die reale Welt, in diesem Fall die Gesichter der Nutzer mit virtuelle Objekten wie z.B. Hundeohren verbinden. Diese Art der Technologie bedient sich dem *video see-through* mithilfe der in den Geräten befindlichen Kameras.

Projective Hier werden eine oder mehrere Projektoren zur Informationsdarstellung verwendet. Ein Beispiel für dieses Verfahren wäre eine *Sandbox* [7]. Dabei handelt es sich um einen realen Sandkasten, auf dessen sandiger Oberfläche, mithilfe eines Projektors und einer Tiefenkamera, eine Art Höhenkarte dargestellt wird. Werden die Sandhaufen mit der Hand verändert, entstehen so auch entsprechende Farben für die neuen Höhenzüge.

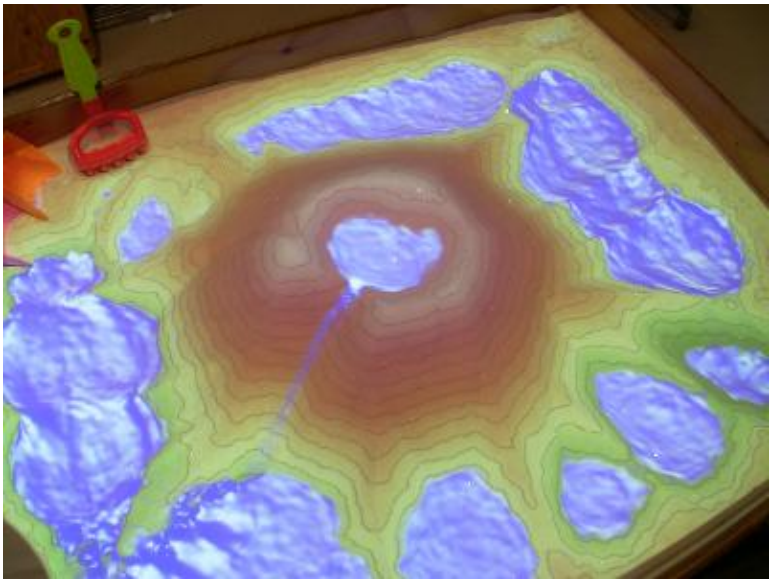

Abbildung 2: Ein Beispiel für eine *Sandbox*.(Quelle: [7])

1.3 Head Wearables

In diesem Abschnitt werden die derzeit verwendeten *Head Wearables* dargelegt. Insbesondere wird auf die, in dieser Arbeit verwendete, *HoloLens* und ihre Eigenschaften eingegangen.

1.3.1 Smart Helmet

Die im Jahr 2010 gegründete Firma Daqri ist spezialisiert auf die Entwicklung von industriellen *AR* Technologien sowie deren Applikationen. Der *Smart Helmet* soll einem Arbeiter in einer Fabrik, z.B. bei der Wartung, Nutzung und Reparaturen von großen Maschinen als Experte zur Seite stehen [8]. Gleichzeitig soll er für seine Sicherheit sorgen, indem er den Arbeitshelm sowie die Sicherheitsbrille ersetzt. Er kann sowohl ortsgebundene Informationen außerhalb des Kontrollraumes darstellen, als auch dem Arbeiter Schritt für Schritt Anweisungen für bestimmte Aufgaben erteilen. Außerdem kann er durch eine Internetverbindung als Kommunikationsschnittstelle dienen. Dabei begleitet ihn eine Menge an Technik, verpackt in einem ungefähr ein Kilogramm schweren Helm. Er besitzt einen leistungsstarken Intel Core m7 Prozessor, eine Hochleistungsweitwinkelkamera mit einem geeigneten Prozessor für AR-Anwendungen, ein *see-through* Display mit einem großen *field-of-view* und einer hohen Helligkeit für eine indoor und outdoor Nutzung. Zudem kann der *Smart Helmet* mittels der bereits erwähnten Kamera, einer Stereo-Infrarotkamera sowie einem Infrarot-Lichtprojektor die Tiefe des Raumes ermitteln. Ein virtueller *Button* wird betätigt indem der Nutzer diesen für eine kurze Zeit fokussiert. Eine akustische Steuerung oder eine Gestensteuerung ist nicht vorhanden.

1.3.2 Daqri Smart Glasses

Diese *AR-Brille* ist ebenfalls für industrielle Zwecke gedacht. Die *Smart Glasses* helfen dabei Maschinen zu warten und zu überwachen. Sie können in einer Logistikstätte den Arbeiter anweisen oder bei Ausbildungstätigkeiten eine interessante und einprägende Art der Informationsvermittlung bieten [8]. Die *AR-Brille* hat einen Intel Core m7 Prozessor, eine weitwinkel AR-Trackingkamera, welche die Bewegungen des Nutzers verfolgt und sie verfügt über die gleichen Sensoren wie der *Smart Helmet*, um die Tiefe des Raumes zu bestimmen. Ebenso hat sie einen gleich großen *field-of-view* und eine gleich hohe Helligkeit für indoor und outdoor. Zudem ist sie deutlich leichter als der *Smart Helmet*. Die *Smart Glasses* soll ab frühem Herbst 2017 für einen Preis von 4995 US Dollar erhältlich sein.

1.3.3 Meta 2

Eine interessante *AR-Brille* stellt die *Meta 2*, wegen ihres geringen Vorbestellungspreises von 949 US Dollar dar [9]. Die *Meta 2* ist kein eigenständiger Rechner, weshalb der Nutzer keinen freien Bewegungsradius hat. Sie muss an einen Computer angeschlossen werden, welcher einige Anforderungen erfüllen muss. Ihr *field-of-view* soll dabei ungefähr bei 90° liegen, während der *field-of-view* der *HoloLens* irgendwo zwischen 30° und 40° liegt. Dabei hat sie eine Auflösung von 2550×1440 Pixeln bei 60 Hz, ein Audioausgabegerät, eine 720p Frontkamera sowie einen Sensor für Gestensteuerung. Zudem soll die *Meta 2* einen angenehmen Tragekomfort bieten und ein moderates Gewicht haben.

1.3.4 HoloLens

Die Microsoft *HoloLens* ist die derzeit weitverbreiteste *AR-Brille* . Zwei Versionen mit unterschiedlichen Preisen und Features sind auf dem öffentlichen Markt erhältlich. Die Technik der beiden Versionen ist identisch, wobei die *Developer Edition* für einzelne Entwickler und die *Commercial Suite* für den Einsatz in Firmen gedacht ist. Die *Developer Edition* mit einem Preis von 3299 Euro ist die günstigere der beiden Versionen, wobei alle zugehörigen Dienste bisher nur auf Englisch erhältlich sind. Mit einem Preis von 5489 Euro ist die *Commercial Suite* um einiges teurer. Jedoch bringt diese Version einige, für den Gebrauch in einem Unternehmen sehr praktische, Softwarefeatures mit sich. Zum Beispiel ist die Bit-Lock Datenverschlüsselung in der *Commercial Suite* aktiviert, außerdem kann der Entwickler Softwareupdates verhindern. Es gibt einige weiter Vorteile.

Optik In der Optischen Einheit der *HoloLens* sind zwei durchsichtige, holografische Linsen (Wellenleiter), zwei HD-16:9-Licht-Generatoren und eine automatische Pupillen-Abstands-Kalibrierung integriert, die eine holografische Auflösung mit insgesamt 2,3 Millionen Lichtpunkten ermöglicht [10].

Sensoren Vier umgebungserfassende Kameras, eine Tiefenkamera, ein Trägheitselement, eine 2-Megapixel Foto-/HD-Video-Kamera, ein Umgebungslichtsensor, vier Mikrofone und eine Einheit für die Erfassung der gemischten Realität liefern ausreichend Daten um ein akkurates Tracking zu ermöglichen

Die Steuerung der *HoloLens* erfolgt über die Erfassung der Ausrichtung des Kopfes, der Gesten sowie der Stimme des Nutzers. Zudem ist die *HoloLens* in der Lage das räumliche Klangbild der Umgebung des Nutzers nachzuahmen [10].

Gaze ist die erste und einfachste Form der Eingabe, welche die *HoloLens* bietet. Der *Gaze* ist vergleichbar mit dem *Maus-Cursor*, mit dessen Hilfe auf dem Desktop navigiert und interagiert werden kann. Die *HoloLens* verwendet die Position und Orientierung des Kopfes des Nutzers als Navigationseingabe. Hierbei wird ein nicht sichtbarer Strahl von der Position zwischen den Augen des Nutzers waagerecht in die Welt gesandt. Der Tiefenwert, an welchem dieser Strahl die reale Umgebung trifft, liefert die Position des *Gaze*. Visualisiert wird dieser in den meisten Fällen mit einem kleinen weißen Punkt. Im Gegensatz zur realen Welt, in der ein Objekt von Interesse mit den Augen fixiert wird, wird bei der *HoloLens* das Objekt mit dem *Gaze* fixiert. Der Nutzer muss also seinen Kopf bewegen um ein Objekt von Interesse zu fixieren und somit für weitere Interaktionsmöglichkeiten auszuwählen. Wie bei einem Desktop zeigt der *Cursor*, die Position, welche fixiert ist [10].

Gesture oder auch Handgesten, werden von der *HoloLens* für eine Interaktion mit virtuellen Objekten bzw. Hologrammen genutzt. Sie bieten den Vorteil, dass der Nutzer keine weiteren technischen Geräte benötigt, um mit Hologrammen interagieren zu können. Die *HoloLens* ist in der Lage entweder eine oder beide Hände gleichzeitig zu erfassen. Dabei muss sich die Hand im sogenannten *Gesture Frame* befinden, damit sie von der *HoloLens* erfasst werden kann. Der *Gesture Frame* ist ein rechteckiges Volumen im vorderen, unteren Bereich der *HoloLens* (Abbildung 3).

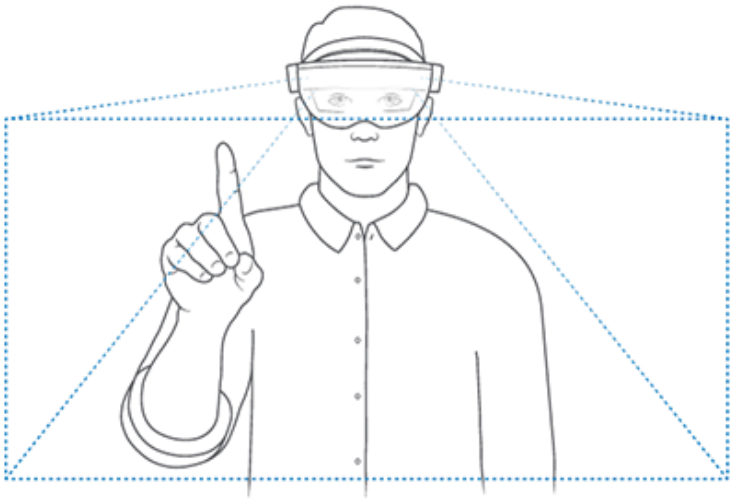

Abbildung 3: Dargestellt ist der *Gesture Frame* in welchem die *HoloLens* die Hand erkennt. (Quelle: [11])

Die Hände des Nutzers werden dann für eine mögliche Interaktion in

Betracht gezogen, wenn sie einen von zwei Zustände erfüllen. Ein Zustand ist der *ready state*, bei dem der Handrücken zur *HoloLens* , und der Zeigefinger nach oben zeigt. Der andere Zustand ist der *pressed state*, bei dem der Handrücken zur *HoloLens* und der Zeigefinger nach unten zeigt.

Für jede Hand, welche die *HoloLens* erkennt, kann der Status und seine Position abgefragt werden. Zudem kann die *HoloLens* die Richtung anzeigen, in welche sich die Hand bewegen muss, falls sie sich am Rand des *Gesture Frames* oder außerhalb befindet.

Der nächste Abschnitt beschäftigt sich mit den *Low-Level Interaktions*, bestehend aus der *Select-Geste* und der *Home-Geste*. Die *Select-Geste* ist der primäre Weg etwas auszuwählen oder zu aktivieren. Dabei wird zunächst eine Faust innerhalb des *Gesture Frames* gebildet und der Zeigefinger wird gehoben. Wenn nun der Zeigefinger nach unten (*press*) und dann wieder nach oben (*release*) bewegt wird, so wird die *Select-Geste* ausgeführt.

Die *Home-Geste* ist das Pendant zum *Home-* bzw. *Start-Button* eines Windows-PCs, mit dem der Nutzer zu jeder Zeit in das Startmenü gelangen kann. Ein anderer Name für diese Geste ist der sogenannte *Bloom*. Die Hand wird dabei mit den Fingern nach oben gehalten, wobei sich die Fingerspitzen berühren. Wird die Hand nun geöffnet, so öffnet sich das Startmenü.

Durch eine Kombination von *press* und *release* sowie der Handbewegung lassen sich andere Gesten herleiten. Für die *Hold-Geste* muss die *Select-Geste* beispielsweise über einen kurzen Zeitraum hinaus gehalten werden. Die *Hold-Geste* kann z.B. für eine Positionsveränderung von Hologrammen verwendet werden, indem das Hologramm zunächst mit dem *Gaze* anvisiert und dann die *Hold-Geste* ausgeführt wird. Nun kann das Hologramm für die Dauer des *Hold* hinsichtlich seiner *Transloation, Rotation* und *Skalierung* manipuliert werden [10].

VoiceInput oder auch Sprachsteuerung ist eine weitere Art der Eingabe. Die *HoloLens* bietet nur eine englische Sprachsteuerung, welche für eine U.S. amerikanische Aussprache optimiert ist. Wie bei *Gesture* wird das Hologramm erst mittels *Gaze* anvisiert und dann mit einem zugehörigen Sprachbefehl manipuliert. Die zugehörigen Sprachbefehle sollten bei dem Anvisieren der Hologramme im visuellen Sichtfeld des Nutzers erscheinen. Die *HoloLens* benutzt ein *"see it, say it"Model* für die Sprachsteuerung, nach dem die Beschriftung eines *Buttons* auch gleichzeitig der Sprachbefehl ist, um diesen auszuführen. Die Sprachsteuerung erlaubt es dem Nutzer mit dem System zu interagieren, ohne Gesten oder andere technische Steuerungselemente zu benutzen.

Der Sprachbefehl *"Select"* bewirkt wie die *Select-Geste* das Auswählen eines Hologrammes oder einer Funktion. Dieser Befehl basiert auf einem *Low-Level Sprachmustererkennungsalgorithmus* und ist somit wegen seinem geringen Aufwand zu jeder Zeit anwendbar. Weitere Sprachbefehle lassen

sich über das Einschalten von *Cortana* mittels *"Hey Cortana"* aktivieren. Dazu gehören Befehle wie *"Go home"*, welcher den Nutzer zum Startmenü weiterleitet, *"Launch"*, welcher eine anvisierte App startet, oder *"Increase the brightness"*, mit dessen Hilfe die Helligkeit erhöht werden kann.

Ein weiterer Vorteil von *VoiceInput* ist es, dass bei der Eingabe von Wörtern darauf verzichtet werden kann, jeden Buchstaben mittels einer *Select-Geste* einzugeben, da der Nutzer das gewünschte Wort diktieren kann.

Darüber hinaus liefert die *HoloLens* drei verschiedene Input-Möglichkeiten für einen Sprachbefehl, welche für unterschiedliche Situationen geeignet sind [10].

Spatial Sound Die *HoloLens* bietet dem Entwickler die Möglichkeit Applikationen zu entwerfen, in welchen der Nutzer eine dreidimensionale Geräuschkulisse um sich herum wahrnehmen kann. Während der visuelle Radius des Menschen ungefähr bei $217°$ liegt und die visuellen Effekte der *HoloLens* lediglich zwischen $30°$ und $40°$ liegen, kann der Nutzer seine komplette Umgebung mit dem Gehör wahrnehmen. *Spatial Sound* ermöglicht dem Nutzer Hologramme in seiner Nähe auszumachen, ohne diese zu sehen. Dadurch entsteht eine größere *Immersion* und die virtuellen Objekte erscheinen realer. So könnte anstelle einer visuellen Warnung nun eine akustische Warnung für die neue Ausrichtung des Kopfes verwendet werden.

Mittels der Ausrichtung des Kopfes, den Distanzen zu den Geräuschen sowie einer Simulation der Umgebung, wird der *Spatial Sound* ermittelt. Die *Head Related Transfer Function (HRTF)* simuliert dabei, wie der Klang von einer Quelle durch einen Raum zu beiden Ohren gelangt. Es können 10 bis 12 *Spatial Sound* Quellen für eine Applikation verwendet werden. Hierbei werden etwa 12% der CPU beansprucht [10].

Spatial Mapping wird von der *HoloLens* genutzt um eine virtuelle Nachbildung eines realen Raumes zu erstellen. Es besteht aus zwei grundlegenden Bestandteilen. Ein Bestandteil ist der *spatial surface observer*, welcher eine oder mehrere *Bounding Volumes* beinhaltet, welche den Raum darstellen, in dem die *spatial map* angelegt wird. Die *Bounding Volumes* können dabei stationär sein, d.h. sie sind an einer absoluten Position der realen Welt, oder sie sind an die Position der *HoloLens* gebunden. Im zweiten Fall orientieren sie sich an der Position der *HoloLens* und bewegen sich mit ihr durch den Raum, jedoch lassen sie sich nicht rotieren. Wird die reale Welt nun durch die *HoloLens* erfasst, so werden die dafür vorgesehenen *Bounding Volumes* mit sogenannten *spatial mapping surfaces* befüllt, welche Teile realer Oberflächen darstellen. Zum einen können die Daten für einen Raum bzw. eine Umgebung einmalig aufgenommen und abgespeichert werden. Zum anderen können die Daten für einen Raum dynamisch aufgenommen werden,

sodass dieser permanent erfasst wird und Veränderungen in die *spatial map* eingehen.

Mithilfe des *spatial mapping* ergeben sich viele Möglichkeiten Hologramme in der realen Welt zu verarbeiten. Sie können nun an realen Gegenständen fixiert werden, sodass sie ihre Position permanent beibehalten. Somit können sich nun Hologramme und reale Objekte gegenseitig überdecken und zu einer möglichst *immersiven* Erfahrung des Nutzers beitragen. Von realen Objekten überdeckte Hologramme können ebenfalls visualisiert werden, damit der Nutzer sie jederzeit detektieren kann. Jedoch sollten sie anders dargestellt werden, damit der Nutzer zwischen verdeckten und nicht verdeckten Hologrammen unterscheiden kann.

Reale physikalische Gegebenheiten können simuliert werden, indem die Hologramme mit realen Objekten interagieren. Fällt z.B. ein virtueller Ball, so kann er von dem realen Boden wieder abprallen.

In der Regel werden die *spatial mapping surfaces* nicht visualisiert, da sie zu viel Rechenleistung in Anspruch nehmen würden. Es ist jedoch von Vorteil sie bei bestimmten Situationen anzeigen zu lassen. Während der Nutzer seine Umgebung aufnimmt, ist es zum Beispiel von Vorteil, die *spatial map* zu visualisieren. So hat der Nutzer die Möglichkeit die Genauigkeit der Aufnahme zu kontrollieren.

Wichtig bei der Erstellung der *spatial map* ist es, dass der Nutzer sich für die Aufnahme Zeit lässt, damit möglichst alle Oberflächendaten gesammelt werden und keine Lücken entstehen [10].

Eine Erweiterung des *spatial mapping* stellt das *spatial understanding* dar.

Spatial Understanding Aufgrund der Unebenheit der *spatial map* wurde eine Erweiterung des *spatial mappings* entwickelt. Sollen Hologramme in der realen Umgebung positioniert werden, um *immersive* Applikationen zu erstellen, so ist ein besseres Verständnis der realen Umgebung von großem Vorteil. *Spatial understanding* erweitert das *spatial mapping* zunächst, indem es aus den unebenen *spatial map Daten* einen einheitlichen, flachen *Mesh* erstellt. Bei diesem ist es um einiges leichter zusammengehörige, ebene Flächen zu detektieren. Wird der *Scanvorgang* beendet, so wird das *spatialUnderstandingCustomMesh*, in dem sich das *Mesh* befindet, komplett geschlossen. Dadurch wird dem Entstehen eines lückenhaften *Meshes* entgegengewirkt.

Sobald das *spatialUnderstandingCustomMesh* fertig gestellt wurde, werden seine Oberflächen intern mit *floor, wall* und *ceiling* markiert (*gelabelt*). Daraufhin können mit den *Topology Queries* und *Shape Queries* individuelle Bereiche von Interesse detektiert werden. Will der Nutzer z.B. ein Hologramm in Form eines Plakates automatisch an einer Wand positionieren, können mittels der *Topology Queries* geeignete Positionen gefunden werden. Auf die selbe Weise wird in der vorgestellten Applikation die *Lava* auf

dem Boden positioniert [12].

1.4 Interaktion im Bereich AR

AR Technologien können viele, unterschiedliche Interaktionstechniken verwenden. In diesem Kapitel werden einige dieser Techniken vorgestellt.

Egal, ob die *Graphical User Interface (GUI)* auf einem Desktop PC, einem Smartphone oder einem Projektor dargestellt wird, sie muss immer mittels physischer Eingabegeräte bedient werden. Kommt nun *AR* ins Spiel, so ergibt sich eine komplett neue Art der Eingabe, nämlich die Eingabe ohne technische Hilfsmittel. Damit überwindet *AR* die Kluft zwischen der physischen und der virtuellen Welt [13]. Zudem verringert die Nutzung von physischen Hilfsmitteln, wie z.B. einer Tastatur, Maus oder planare Marker zum Tracking, die *AR* Erfahrung [13][14]. Zusätzlich ermöglicht dies nun auch mit mehreren Geräten gleichzeitig zu arbeiten. Der Nutzer hat den Vorteil weiterhin die reale Welt zu sehen und mit dieser, in diesem Fall einem anderen Desktop oder ähnlichem, zu interagieren, ohne dafür die *AR-Brille* abzunehmen [15][13]. In den Anfängen der *AR* ging es bei den meisten Prototypen viel mehr um eine gute Präsentation von Informationen und weniger um die Interaktion mit diesen [15].

Herkömmliche Eingabegeräte Unter herkömmlichen Eingabegeräten sind technische Hilfsmittel gemeint. Dazu gehören alle Knöpfe und Steuerungssticks, wie sie z.B. an einem *GamePad* zu finden sind. Ebenfalls zählen hier die Eingabe und Manipulation mittels Markern zu den herkömmlichen Eingabegeräten, da sie für den Nutzer sichtbare Hilfsmittel sind und eine *Immersion* verschlechtern.

Gestensteuerung Schon in den 1980er Jahren wurde versucht mit Gesten zu arbeiten. Damals wurden sogenannte Datenhandschuhe verwendet, um die Bewegungen einer Hand oder der Finger zu deuten. Obwohl mittlerweile, auf Bildanalyse basierende, Gestensteuerungsverfahren existieren, wie zum Beispiel die Gestensteuerung der *HoloLens*, sind die Datenhandschuhe noch nicht ganz aus der Mode. Heute wird versucht die damaligen Problematiken der steifen Finger, Unbequemlichkeit, etc. aus der Welt zu schaffen [16].

Der Vorteil einer Steuerung der virtuellen Oberfläche mittels der Hände ergibt sich aus einer natürlichen und intuitiven Steuerung [17]. Zu einer Gestensteuerung gehören folgende Elemente. Zunächst müssen die Hände bzw. eine Hand des Nutzers erkannt werden. Dabei wird die Position sowie die Ausrichtung eines Gegenstandes in einem 3D Raum ermittelt. Nachdem die Hand erfasst wurde, wird nun die Haltung der Hand und der Finger interpretiert, um den dafür entsprechenden Befehl auszufüh-

ren. Dieser Vorgang wird als Klassifikation der Geste bezeichnet. Da die Hand und die Finger frei bewegt werden können, stehen der Gestensteuerung eine große Menge an unterschiedlichen Gesten für eine Eingabe zur Verfügung.

Nachteile in der Gestensteuerung liegen darin, dass der Nutzer die Gesten kennen und nicht vergessen darf. Eine intuitive Nutzung von Gesten wird erst dann möglich, wenn der Nutzer schon Erfahrung hat. Aufgrund der Seltenheit dieser Interaktion muss den Nutzern Zeit gegeben werden, sich mit diesen auseinander zusetzen. Zudem können Menschen mit einer Behinderung der Hände diese Art der Interaktion nicht verwenden.

S.Hegde, R.Perla und R.Hebbalaguppen haben es geschafft eine Echtzeit Gestenerkennung für Smartphones zu implementieren. Diese ist bezüglich des Energieverbrauchs effizienter als herkömmliche Gesternerkennungsverfahren und somit interessant für mobile Einsätze [18].

Im Bereich des *Gaming* in der *AR* kann eine natürliche Gestensteuerung zu der *Immersion* des Nutzers beitragen, wobei die Steuerung so flüssig wie möglich ablaufen sollte [19].

Sprachsteuerung Eine Steuerung eines Systems mittels Sprachbefehlen kann sehr nützlich sein, da sie komplett ohne den Einsatz von technischen Hilfsmitteln oder der Hände abläuft. Die Autoren J. Bose u.a. beschreiben eine freihändig bedienbare, mobile Browser-Applikation, welche unter anderem mittels Sprachsteuerung bedient werden kann. In dem Falle, dass der Nutzer gerade ein Auto fährt und somit in diesem Moment keinen Browser bedienen kann, oder falls der Nutzer seine Hände z.B. auf Grund einer Behinderung nicht nutzen kann, wäre eine freihändige Bedienung sehr nützlich [20]. Diese Szenarien können für viele andere Aufgabenbereiche ausgeweitet werden. R,Sarikaya u.a. beschreiben, dass eine natürliche Sprachsteuerung einen größeren Umfang an Informationsaustausch zwischen Nutzer und System bieten kann und Aufgaben somit schneller erledigt werden können, wodurch die Produktivität des Nutzers steigt [21].

N. Nasari u.a. erläutern, dass eine Spracherkennung, eingebaut in ein seriöses Spiel, kleinen Kindern mit Sprachproblemen helfen könnte, ihre Sprache zu verbessern. Eine Applikation, welche den Kindern auf eine spielerische, aber doch didaktische Art etwas beibringt, den Kindern erlaubt eigenständig zu lernen und gleichzeitig den Fortschritt ihres Lernens messen kann, könnte solchen Kindern weiterhelfen [22][23].

1.5 Bestehende Interaktionskonzepte im Bereich AR-Gaming

Momentan existieren noch wenige, auf dem öffentlichen Markt, erhältliche *HoloLens* Spiele. Bei den Spielen *RoboRaid* und *Young Conker* muss zu Beginn die Umgebung des Nutzers aufgenommen werden. Dabei werden

nach der Fertigstellung der Aufnahme automatisch Hologramme in der generierten Umgebung platziert.

RoboRaid In diesem Spiel muss der Nutzer mit seinem ganzen Körper interagieren [24]. Während des Spiels fliegen kleine Roboter durch den Raum und feuern ab und zu virtuelle Feuerbälle auf den Nutzer ab. Dieser kann den Feuerbällen durch die Bewegung des Körpers ausweichen. Visiert der Nutzer die kleinen Roboter mit dem *Gaze* an und führt die *Select-Geste* aus, so kann er sie abschießen und zerstören. Mithilfe des *Sprachbefehls "XRay"* kann er seine *Fähigkeit*, welche alle Roboter in der Nähe zerstört, aktivieren. Dieses Steuerungskonzept versetzt den Nutzer in die Rolle des *Protagonisten* des Spiels.

Das Menü lässt sich durch das Anvisieren der *Buttons* mit dem *Gaze* und der Ausführung der *Select-Geste* , für das Betätigen der *Buttons*, steuern.

Dieses Spiel bringt alle Interaktionsmöglichkeiten der *HoloLens* zusammen, wobei jede nur einer einzigen Funktion zugeordnet ist.

Young Conker Während der Nutzer in dem Spiel *RoboRaid* der *Protagonist* ist, ist es in dem Spiel *Young Conker* ein kleiner Eichhörnchen ähnlicher *Avatar* [25]. Dieser lässt sich durch die Umgebung navigieren, indem der Nutzer den Sprachbefehl *"Lets Go"* äußert oder die *Select-Geste* ausführt. Dadurch folgt der *Avatar* dem *Gaze*. Sobald die *Select-Geste* abermals ausgeführt wird oder der Sprachbefehl "*Stop"* geäußert wird, bleibt der *Avatar* wieder stehen. Zudem kann der *Avatar* mittels des Sprachbefehls *"Jump"* springen und mittels des Sprachbefehls *"Faster"* seine Bewegungsgeschwindigkeit kurzzeitig erhöhen.

Die Steuerung des Menüs funktioniert genau wie in dem Spiel *RoboRaid*. Hinzu kommt, dass hier die *Buttons* auch betätigt werden können, indem der Nutzer deren Aufschrift als Sprachbefehl äußert.

Actiongram Bei der Applikation *Actiongram* handelt es sich in erster Linie nicht um ein Spiel [26]. Mithilfe dieser Applikation kann ein Nutzer Hologramme in der realen Umgebung platzieren und diese in Scene setzten um ein Video zu erstellen. Interessant bei dieser Applikation ist die Platzierung der Hologramme. Aus einer großen Sammlung kann der Nutzer ein Hologramm auswählen. Jedes Hologramm hat ein eigenes Menü, über das der Nutzer die *Translation, Skalierung, Rotation* und mehr manipulieren kann. Jede dieser manipulierbaren Eigenschaften des Hologramms besitzt wieder ein Untermenü. Mithilfe der *Hold-Geste* lassen sich die Hologramme manipulieren. Dazu wählt der Nutzer eine Eigenschaft wie z.B. die *Translation*. Wenn er nun die *Hold-Geste* ausführt, ist die *Translation* des Hologrammes an die *Translation* der Hand geheftet. Somit kann der Nutzer das Hologramm auf der *x-Achse, y-Achse* und *z-Achse* durch die Bewegung

der Hand im Raum verschieben. Dabei ist die *Translation* des Hologramms während der Ausführung der *Hold-Geste* auch an die Bewegung des Kopfes des Nutzers geheftet. So bewegt sich das Hologramm mit dem Nutzer durch den Raum.

Das Menü lässt sich mittels eines Betrachtens und der Ausführung der *Select-Geste* steuern. Zudem können die *Buttons*, welche beschriftet sind auch mit dem gleichnamigen Sprachbefehl aktiviert werden.

1.6 Verwendete Programme und Tools

In diesem Abschnitt werden die verwendeten Programme und *Tools* kurz aufgezeigt. Auf einzelne Funktionen dieser *Tools* wird später in ihrem jeweiligen Verwendungsbereich detaillierter eingegangen.

1.6.1 Unity

Unity ist eine Entwicklungsumgebung für Spiele mit dessen Hilfe Applikationen für unterschiedliche Plattformen entwickelt werden können [27]. Zu diesen Plattformen gehören *PCs*, *Spielkonsolen*, *Webbrowser* und mobile Geräte wie *Smartphones, Tablets, HMDs (z.B. HoloLens)*.

Die grafische Oberfläche von *Unity* besteht aus mehreren Fenstern, welche je nach Bedarf individuell strukturiert werden können. Zu den wichtigsten Fenstern gehört das *Game-Fenster*, welches beim Ausführen der Applikation die *Scene* im Endzustand grafisch darstellt. Das *Scene-Fenster* zeigt die verwendeten, virtuellen Objekte und ihre Positionierung im 3D-Raum an. Zudem werden in diesem Fenster auch *Lichter, Kameras,* etc. visualisiert. Das *Project-Fenster* verfügt über eine Ordnerstruktur, in welcher die *Tools, Scripts, 3D-Objekte, etc.* zu finden sind, welche in diesem Projekt eingebunden wurden. Im *Hierarchy-Fenster* werden die *GameObjects* für die aktuelle *Scene* strukturiert. *GameObjects* sind die fundamentalen Objekte in einer *Unity-Applikation*. Sie repräsentieren Teile aus dem Spiel wie z.B. einen *Avatar*, *Lichter* und mehr. Die *GameObjects* können mit unterschiedlichen *Components* ausgestattet werden.

Über das *Inspektor-Fenster* werden die *Components* eines *GameObjects* veranschaulicht. Zu den *Components* gehören z.B. die *Transform*, bestimmte *Collider*, *Scripts* und vieles mehr.

Ein *GameObject* kann einem *Layer* zugewiesen werden. Ein *Layer* kann z.B. bezwecken, dass bestimmte Objekte durch einen *Ray* erfassbar sind, während andere Objekte nicht von diesem *Ray* erfasst werden. Während *Layer* dazu verwendet werden, *GameObjects* mit gleichen Eigenschaften zu gruppieren, werden sogenannte *Tags* dazu eingesetzt einzelne Objekte im Spiel zu identifizieren.

Das Herzstück einer jeden Applikation bilden die *Scripts*. Mithilfe der *Scripts* können die *Components* der *GameObjects* manipuliert und so ihr Ver-

halten kontrolliert werden. Es besteht die Möglichkeit die *Scripts* entweder in *C#* oder in *UnityScript*, welches *JavaScript* nachempfunden wurde, zu schreiben. In dieser Arbeit wurden alle *Scripts* mit *C#* geschrieben.

Ein weiterer, wichtiger Bestandteil von *Unity* sind die sogenannten *Prefabs*. Das *Prefab* ist vergleichbar mit einer Skizze. Falls ein *GameObject* existiert, welches immer die selben *Components* und Werte hat und oft in der *Scene* vorkommt, so muss dieses nicht jedes mal neu editiert werden. Ein *Prefab* wird erzeugt, welches als Vorlage dient und beliebig oft instanziiert werden kann.

Unity bietet somit die Grundlagen zum Entwickeln einer *immersiven* Applikation für die *HoloLens* , mit welcher die Konzeption umgesetzt werden kann.

Verwendete Components

Transform ist die *Transformation*, *Rotation* und *Skalierung* eines Objektes im 3D-Raum.

Camera ist das *Component*, durch welche der Nutzer die virtuelle Welt betrachtet.

Mesh Filter Dieses *Component* erhält einen bestimmten *Mesh* z.B. ein *Cube* und leitet diesen an den *Mesh Renderer* weiter.

Mesh Renderer erhält das vom *Mesh Filter* weitergeleitete *Mesh* und visualisiert dieses an der *Position*, welche im *Transform* des *GameObjects* zu finden ist.

Mesh Collider nimmt das vorgegebene *Mesh* des *GameObjects* und baut auf dieser Basis einen *Collider*.

Box Collider ist ein primitiver *Collider* in Form eines Rechtecks.

Rigidbody Wird dieses *Component* einem *GameObject* zugewiesen, so wird die Bewegung dieses *GameObjects* unter die Kontrolle von *Unitys physics engine* gegeben. Somit wird dieses Objekt z.B. von der virtuellen Schwerkraft angezogen.

Text Mesh generiert eine 3D-Geometrie, welche einen *Text String* visualisiert.

AudioSource kann an ein *GameObject* angehängt werden, um Audiodateien im 3D-Raum abzuspielen.

1.6.2 MixedRealityToolkit-Unity

Das *MixedRealityToolkit-Unity* bietet die softwareseitige Grundlage, um die Funktionen der *HoloLens* nutzen zu können [28]. Es handelt sich um eine Kollektion von *Scripts*, *Components* und *Prefabs*, mit dessen Hilfe *immersive* Applikationen für die *HoloLens* in *Unity* entwickelt werden können. Das *MixedRealityToolkit-Unity* liefert z.B. *Skripts*, welche dem Entwickler unter anderem helfen Eingabemöglichkeiten, wie *Gaze, Gesture* und *VoiceInput* zu verarbeiten oder die Aufnahme von *spatial maps* und *spatialUnderstanding-CustomMeshs* umzusetzen.

1.6.3 MRDesignLab

Das *MRDesignLab* ist eine Sammlung von Beispielen in Form von *Scripts, Components, Prefabs, etc.*, welche dem Entwickler helfen sollen visuell anschauliche *Mixed Reality Applikationen* zu entwickeln [29]. Dieses *Repository* wird von Microsofts *Mixed Reality Design Team* publiziert. Das *MRDesignLab* hat zum Teil ähnliche Funktionen wie das *MixedRealityToolkit-Unity*, dabei überschneiden sich einige und werden doppelt ausgeführt. Dazu gehört ein doppeltes Abfangen der *Gesten-Events*. Weitere Einzelheiten und die Behebung dieses Fehlers werden in Abschnitt 3.2 genauer erläutert. Dieses *Tool* wurde zusätzlich zu dem *MixedRealityToolkit-Unity* integriert, da es mit seiner Hilfe möglich ist, eine angemessene und anschauliche Menüführung zu erstellen.

1.6.4 HoloLensXboxControllerInput

Die Integration dieses *Tools* ermöglicht einer *HoloLens Applikation* die Nutzung eines *Microsoft Xbox One S Wireless Controllers* [30]. Dabei funktioniert das *GamePad* lediglich mit der *HoloLens* selbst und nicht mit dem *HoloLens Emulator* oder im *Unity-Editor*.

2 Konzept

Zum Testen der unterschiedlichen Interaktionsmöglichkeiten, welche die *HoloLens* bietet, soll ein prototypisches Spiel entwickelt werden. Dabei sollen die Interaktionen in unterschiedlichen Bereichen einer Anwendung im Anschluss einer Evaluation unterzogen werden. Die Bereiche setzen sich aus der Interaktion mit einem Menü, der Positionierung von virtuellen 3D-Objekten in der realen Umgebung, sowie der Navigation eines virtuellen *Avatars* in der realen Umgebung zusammen (Abbildung 5). Die entwickelten Konzepte zur Steuerung des *Avatars* orientieren sich dabei zum Großteil an den Steuerungskonzepten des Spiels *Young Conker*. Für die Positionierung der Hologramme im Raum wurde ein neues Konzept entwickelt. Das realisierte Spiel besteht aus drei Phasen (Abbildung 4).

2.1 Phase 1: Aufnahme der Spielumgebung

Zunächst muss der Nutzer das Menü, welches sich zu Beginn der Anwendung am *Gaze* befindet, an einer beliebigen Position im Raum mittels der *Select-Geste* ablegen. Sobald das Menü positioniert wurde, sind seine zwei *Buttons* bereit zur Verwendung. Aktivierbar sind die *Buttons* als Sprachbefehl oder über das Anvisieren mit dem Gaze und dem Ausführen der *Select-Geste*. Wird der obere *Button "Create Surrounding"* betätigt, so wird der *Scanvorgang* gestartet. Dabei wird ein *spatialUnderstandingCustomMesh* angelegt und für den Nutzer visualisiert, damit dieser seine Umgebung detailliert aufnehmen kann. Sobald der Nutzer den *Scanvorgang* beenden möchte, kann er den unteren *Button "Finalize Scan"* auswählen. Dadurch wird das *spatialUnderstandingCustomMesh* fertiggestellt, der Boden wird zu *Lava*, das *StartMenü* wird durch das *SpielMenü* ersetzt und die grüne *Startflagge* an den *Gaze* gesetzt. Dieser Vorgang kann einen kurzen Moment dauern.

2.2 Phase 2: Erweiterung der Spielumgebung

Nun muss der Nutzer nacheinander die grüne *Startflagge* und daraufhin die rote *Zielflagge*, welche sich am *Gaze* befinden, mittels der *Select-Geste* an beliebigen Positionen im Raum ablegen. Die Flaggen sollten dabei möglichst weit auseinander und nicht auf den selben, realen Objekten stehen, da der Nutzer später die Aufgabe hat, den *Avatar* von der *Startflagge* zur *Zielflagge* zu navigieren. Befinden sich die beiden Flaggen auf dem selben, realen Objekt so ist die Aufgabe sofort zu bewältigen. Befinden sie sich jedoch auf unterschiedlichen Objekten, welche nicht mit einander verbunden sind, so muss der Nutzer einen Weg bauen, um die *Zielflagge* zu erreichen. Nach dem Positionieren der roten *Zielflagge* erscheint eine rechteckige *Box* am *Gaze*. Nun können *Boxen* an beliebigen Positionen, durch das Ausfüh-

ren der *Select-Geste* oder durch den Sprachbefehl *"Place Box"* positioniert, instanziiert werden. Durch das Anvisieren einer *Box* und die Ausführung der *Hold-And-Release-Geste* oder dem Äußern von *"Delete Box"* kann die *Box* gelöscht werden. Mithilfe dieser *Boxen* kann der Nutzer nun einen Weg bauen, um später die *Zielflagge* mit dem *Avatar* zu erreichen.

Über das *SpielMenü*, mit den beiden Operationen "*Activate Avatar*" und "*Activate Build*", hat der Nutzer jederzeit die Möglichkeit, zwischen dem *Aufbau der Spielumgebung* und der *Navigation des Avatars* zu wechseln.

2.3 Phase 3: Navigation des Avatars

Sobald der *Button* "*Activate Avatar*" betätigt wurde, verschwindet die *Box* vom *Gaze* und eine weiße Sphäre, der *Avatar*, erscheint an der *Startflagge*. Das Ziel des Nutzers ist es den *Avatar* zur *Zielflagge* zu navigieren.

Der *Avatar* kann nun auf zwei unterschiedliche Steuerungsarten durch die reale Umgebung navigiert werden (Abbildung 5).

Steuerung 1: Follow My Gaze Dieses Steuerungskonzept bedient sich der Gestensteuerung, der Sprachsteuerung sowie dem *Gaze*. Gestartet wird die Steuerung mittels dem Sprachbefehl *"Follow My Gaze"*, womit der *Avatar* anfängt, den *Gaze* geradlinig zu verfolgen. Hier findet keine Pfadverfolgung, sondern lediglich eine Bewegung in die Richtung des *Gaze* statt. Durch das Ausführen des Sprachbefehls *"Jump Up"* oder der *Select-Geste* springt der *Avatar*. Mit dem Sprachbefehl *Stop Following* bleibt der *Avatar* wieder stehen.

Steuerung 2: GamePad Dieses Steuerungskonzept bedient sich einer herkömmlichen *GamePad-Steuerung*. Mit dem linken *Joystick* des *GamePads* lässt sich der *Avatar* in *x-* und *y-Richtung* bewegen und durch Betätigen des *A-Buttons* springt er. Dabei sind die *x-,y-Achsen* des *Avatars*, die *x-,y-Achsen* des Nutzers, und somit die *Achsen* der *HoloLens* aufeinander abgestimmt, sodass sich der *Avatar* bezüglich der Ausrichtung der *HoloLens* bewegt.

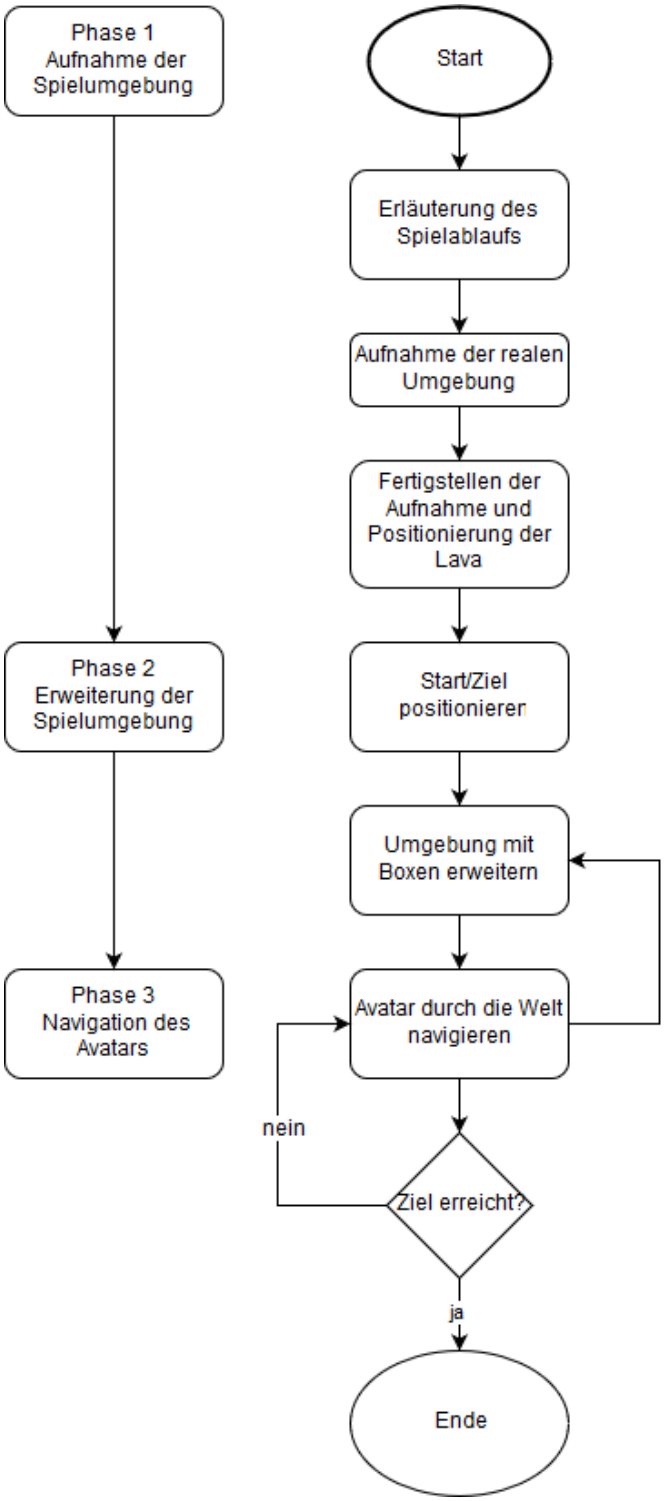

Abbildung 4: Dies ist eine grobe Skizze des Spielablaufs.

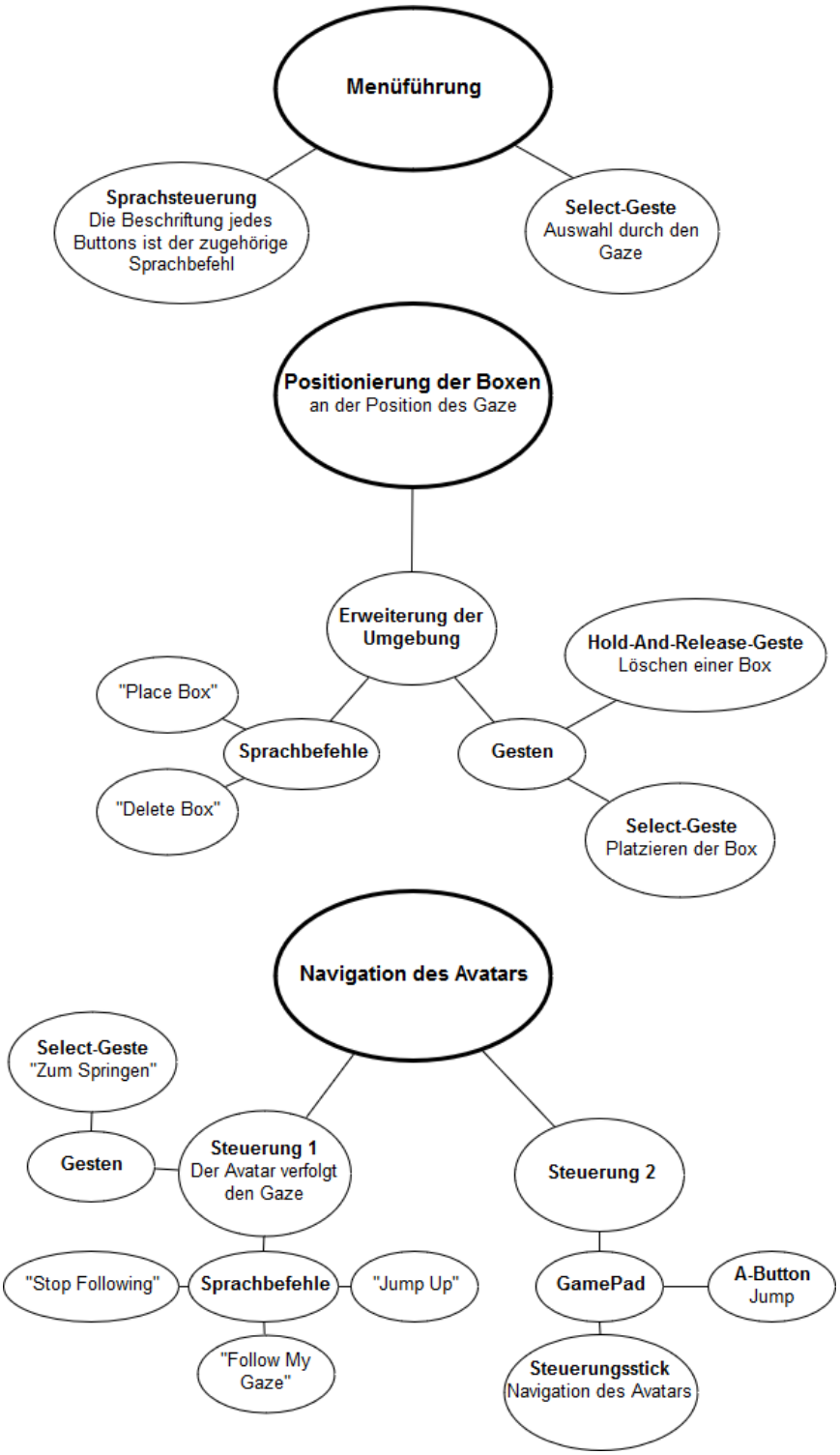

Abbildung 5: Diese Diagramme zeigen die Interaktionskonzepte.

3 Implementation

Der strukturelle Aufbau der Applikation besteht grob aus fünf Bereichen.

Der erste Bereich *"HoloLens"* ist ein *Prefab* aus dem *MRDesignLab*. Dieses *GameObject* bringt alle notwendigen Klassen aus dem *MRDesignLab* zusammen, um eine Grundlage für die Nutzung der *HoloLens* zu schaffen. Zum Beispiel wird hier die *Kamera* oder der *Focus Manager*, welcher die Berechnung des *Gaze* übernimmt, implementiert. Das *MRDesignLab* wurde aufgrund der einfachen Implementierung eines visuell, ansehnlichen Menüs später in die Applikation integriert. Teilweise wurden bestimmte Methoden von Klassen aus dem *MixedRealityToolkit-Unity* durch Methoden von Klassen aus dem *MRDesignLab* abgelöst. Dabei handelt es sich überwiegend um Funktionen der Behandlung von Interaktionen. Die beiden *Frameworks* sind zu diesem Zeitpunkt noch nicht vollständig kompatibel. Zum Beispiel werden *Events*, welche durch Gesten ausgelöst wurden, an zwei Stellen abgefangen. Dies wird in Abschnitt 3.8 genauer erläutert. Der Großteil der Applikation läuft im *Unity Editor*. Dadurch fällt das Testen von neuen Funktionen oder Änderungen wesentlich leichter.

Der "*InputManager*" ist ein *Prefab* aus dem *MixedRealityToolkit-Unity*. Alle Eingabemöglichkeiten, abgesehen vom *Gaze* und der Gestensteuerung für die Menüs, werden von Klassen aus dem *MixedRealityToolkit-Unity* verarbeitet. Wichtig ist der *Keyword Manager* aus dem *InputManager*, welcher die *Sprachbefehle* organisiert und der *InputHandler*, welcher den Gesten eine Funktion zuweist.

Ein weiteres *Prefab* aus dem *MixedRealityToolkit-Unity* ist "*Spatial*". In diesem Bereich der Applikation befinden sich alle notwendigen Klassen für das Erstellen von *spatial maps* und *spatialUnderstandingCustomMeshs*.

Der vierte Bereich *"GameManager"* beinhaltet die eigentlichen Inhalte des Spiels, ausgenommen der Menüs. Hier befinden sich alle virtuellen Objekte, sowie die *Components*, welche dem Spiel und den Objekten Eigenschaften verleihen.

Das "*Menu*" ist der letzte Bereich des Spiels. In diesem *GameObject* befinden sich die Menüs, welche mithilfe der *Buttons* aus dem *MRDesignLab* erstellt wurden. Das *Script MenuManager* implementiert die Funktionen der einzelnen *Buttons*.

3.1 Gaze

Im Laufe des Spiels werden unterschiedliche Objekte an den *Gaze* angeheftet. Zu Beginn wird dem *Gaze* das *StartMenü* und später werden dem *Gaze* die *BuildBox* oder die Flaggen zugewiesen. Während die Klasse *FocusManager* des *MRDesignLabs* die Position und Ausrichtung des *Gaze* liefert, kümmert sich die Klasse *CursorManager* um die Platzierung der entsprechenden Objekte zum entsprechenden Zeitpunkt. *CursorManager* enthält die Objek-

te *BoxOnHologramm* und *BoxOffHolgramm*. *BoxOnHologramm* ist das Objekt, welches angezeigt wird, wenn der Nutzer sich in der *Build-Phase* befindet und den *Gaze* über ein virtuelles Objekt bewegt. Bewegt sich der *Gaze* nicht über ein virtuelles Objekt, so wird die *BoxOffHologramm* angezeigt. Da die Menüs sowie die Flaggen zusätzlich instanziiert werden, befinden sich ihre Objekte nicht in dieser Klasse. Ihre Positionierung am *Gaze* findet jedoch weiterhin in dieser Klasse statt.

Bei der Positionierung und Ausrichtung der Objekte ist es sehr wichtig, dass sich die Koordinatensysteme der Objekte im Mittelpunkt ihres Bodens befinden, wobei der *Up-Vector* in das Objekt selbst zeigt. Dies ist notwendig, da der Ursprung des Koordinatensystems des Objektes an den *Gaze* gesetzt wird. Würde sich das Koordinatensystem eines Würfels beispielsweise in der Mitte des Volumens befinden, so würde der Würfel bei der Verschiebung zum *Gaze* in das anvisierte Objekt ragen.

3.2 Menü

Das Menü besteht aus dem *StartMenu* und dem *GameMenu*, welche wiederum jeweils aus zwei *Buttons* bestehen.

Ein *Button* ist ein *Prefab* des *MRDesignLabs*. Er hat einige *Components*, wie einen *Collider* oder einen *Renderer*. Diese *Components* werden durch mehrere *Scripts* organisiert und manipuliert. Wichtig sind die *Components Compound Button Text* und *Compound Button Icon*. Wird das *Compound Button Text Component* im *Inspector-Fenster* eines *Buttons* geöffnet, so kann unter *Button text:* eine beliebige Zeichenfolge vom Typ *String* eingegeben werden. Diese stellt den angezeigten Text auf dem *Button* dar.

Compound Button Icon organisiert die *Icons* der einzelnen *Buttons*. Hier muss für jeden *Button* mit einem eigenen *Icon* ein eigenes Profil angelegt werden. Jeder *Button* in dieser Anwendung hat ein eigenes *Icon*. Somit werden vier dieser Profile angelegt und dem jeweiligen *Compound Button Icon Component* zugewiesen.

Um den *Buttons* eine Funktion zuzuordnen, werden *InteractionReceiver* eingesetzt. *InteractionReceiver* ist eine abstrakte Klasse aus dem *MRDesignLab*. Für jeden *Button* wird eine eigene Klasse angelegt, welche von *InteractionReceiver* erbt. In diesen Klassen wird die Methode

```
1 protected virtual void OnTapped(GameObject obj,
      InteractionManager.InteractionEventArgs eventArgs) { }
```

überschrieben. Diese Methode wird aufgerufen, wenn die *HoloLens* eine *Select-Geste* registriert und sich gleichzeitig der *Gaze* auf dem jeweiligen Button befindet. Den *InteractionReceivern*, werden *Interactibles* zugeordnet. In diesem Fall sind die *Interactibles* die *Buttons* selbst. Diese *Interactibles* können jegliche, virtuelle Objekte sein. Wird ein virtuelles Objekt einem

Receiver als *Interactible* zugeordnet, so werden die in diesem *Receiver* enthaltenen Funktionen aktiviert bzw. erreichbar, sobald sich der *Gaze* über dieses Objekt bewegt.

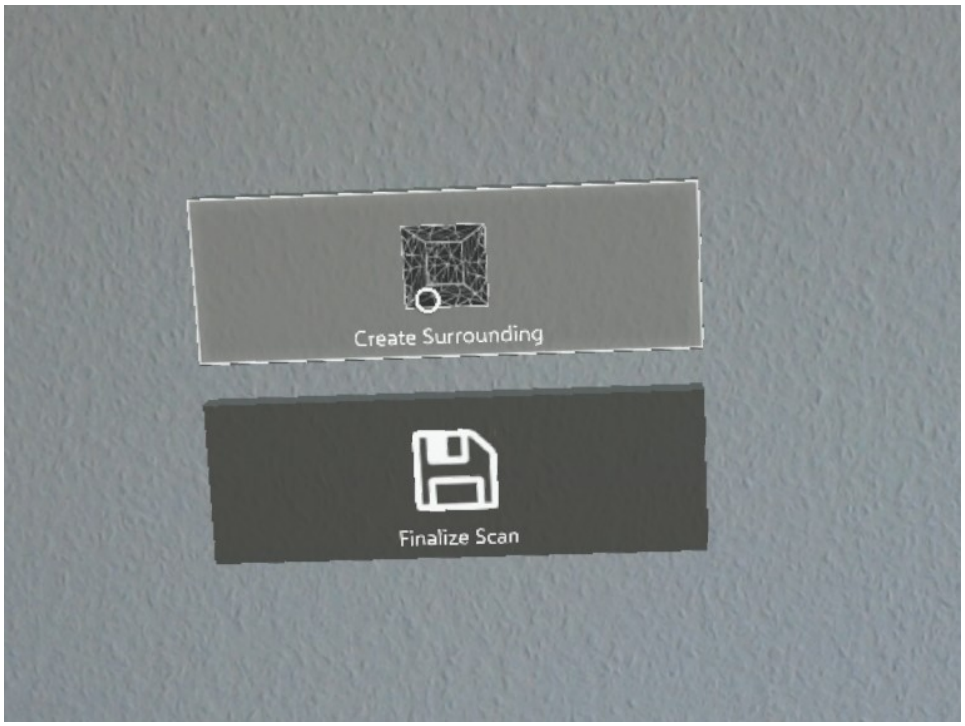

Abbildung 6: Dieses Bild zeigt das *StartMenu*.

Die *Events*, welche beim Registrieren einer Geste geworfen werden, werden sowohl von den *InteractionReceivern*, als auch von dem *InputManager* abgefangen. Der *InputManager* ist eine Klasse aus dem *MixedRealityToolkit-Unity*. Um zu verhindern, dass die *Events* doppelt verwendet werden, findet im *InputHandler* eine Prüfung auf den *Tag* des vom *Gaze* erfassten Objektes statt. Hat dieses Objekt den *Tag = "Button"*, so wird das *Event* vom *InputHandler* ignoriert. Wenn diese Prüfung nicht stattfinden würde, dann würde z.B. eine *Box* auf dem *Activate Avatar Button* abgelegt werden, falls dieser mit der *Select-Geste* betätigt wird.

Das *StartMenu* muss zu Beginn des Spieles an einer vorgegebenen Position im Raum abgelegt werden. Dabei wird dieses Menü solange an den *Gaze* geheftet, bis der Nutzer es mittels der *Select-Geste* ablegt. Die Menüs haben ein *Component* namens *Billboard*, welches dafür sorgt, dass sich die Menüs bei der Positionierung immer dem Nutzer zuwenden. Die *Buttons* haben einen *Box Collider*, welcher standardmäßig dem *UserInterface(UI) Layer* zugeordnet ist. Dieses *Layer* wird von dem *Gaze* wahrgenommen, sodass sich das *StartMenu* permanent auf den Nutzer zubewegt. Um dies zu verhindern, werden die *Layer* der *Buttons* des *StartMenus* bei ihrer Initialisie-

rung auf *2 = ignoreRaycast* gesetzt. Nachdem das *StartMenu* positioniert wurde, werden sie wieder standardmäßig auf *5 = UserInterface(UI)* gesetzt.

Sobald die Umgebung aufgenommen wurde und der Nutzer den *Button "Finalize Scan"* betätigt, wird zum einen die Spielumgebung fertiggestellt und zum anderen wird das *StartMenu* deaktiviert und das *GameMenu* an der selben Position aktiviert.

3.3 Aufnahme der SpatialUnderstanding Map

Die Aufnahme der *spatial map* wird beim Starten der Applikation aktiviert. Dabei ist die Visualisierung des *Meshs* deaktiviert. Betätigt der Nutzer nun den *Button "Create Surrounding"*, so wird die folgende Methode

```
1   public void createSurrounding()
2     {
3         if (MenuManager.Instance.fixMenuPos)
4         {
5             SpatialMappingManager.Instance.DrawVisualMeshes
                  = false;
6             SpatialUnderstanding.Instance.
                  RequestBeginScanning();
7             StartCoroutine(HologramManager.Instance.
                  ShowMessage("Look␣Around␣to␣scan␣your␣
                  PlaySpace.",5));
8         }
9     }
```

ausgeführt. Zunächst findet an dieser Stelle eine Überprüfung statt, ob das *StartMenu* schon abgelegt wurde. Somit wird sicher gestellt, dass die Methode erst ausgeführt wird, nachdem das *StartMenu* positioniert wurde. Ist dies der Fall, so wird die Aufnahme für das *spatialUnderstanding* gestartet und der Nutzer bekommt die Nachricht *"Look around to scan your PlaySpace"* einen kurzen Moment lang angezeigt. Das *understandingCustomMesh* wird automatisch visualisiert, damit der Nutzer den Vorgang beobachten und etwaige Fehler in der *understandingCustomMesh* durch längeres Betrachten aus unterschiedlichen Perspektiven beheben kann.

Sobald der Nutzer mit der Aufnahme zufrieden ist, kann er den *Button "Finalize Scan"* betätigen.

```
1   public void finalizeSurrounding()
2     {
3         if (MenuManager.Instance.fixMenuPos && !GameManager
              .Instance.surroundingLoaded &&
              SpatialUnderstanding.Instance.ScanState ==
              SpatialUnderstanding.ScanStates.Scanning)
4         {
5             //Beendet den Scanvorgang.
```

```
6               SpatialUnderstanding.Instance.StartCoroutine("
                    RequestFinishScan");
7
8               SpatialMappingManager.Instance.SurfaceObserver.
                    StopObserving();
9               LavaPlacement.Instance.readyToPlaceLava = true;
10
11              //Das Material der UnderstandingCustomMesh wird
                     auf transparent(schwarz) gesetzt. Somit ist
                     sie weiterhin aktiv,
12              //aber nicht zu sehen, da schwarz von der
                    HoloLens nicht dargestellt werden kann.
13              SpatialUnderstanding.Instance.
                    UnderstandingCustomMesh.MeshMaterial =
                    SpatialUnderstanding.Instance.
                    UnderstandingCustomMesh.transparentMaterial;
14
15              //Das UnderstandingCustomMesh muss gerendert
                    werden, da sonst die komplette Plane der
                    Lava zu sehen ist.
16              SpatialUnderstanding.Instance.
                    UnderstandingCustomMesh.DrawProcessedMesh =
                    true;
17
18                          //surroundingLoaded wird auf true
                                gesetzt, sodass der Build und
                                die Steuerung des Avatars
                                aktivert werden.
19              //Das StartMenu wird deaktiviert und das
                    GameMenu aktiviert.
20              GameManager.Instance.surroundingLoaded = true;
21              GameObject.Find("Menu/Menus/StartMenu").
                    SetActive(false);
22              GameObject.Find("Menu/Menus/GameMenu").
                    SetActive(true);
23
24              StartCoroutine(HologramManager.Instance.
                    ShowMessage("␣Your␣PlaySpace␣will␣be␣
                    finalized␣in␣a␣few␣Seconds.", 3));
25
26              //Warte 3 Sekunden bis die Flaggen positioniert
                     werden koennen.
27              Invoke("flagsCanNowBePlased", 3.0f);
28          }
29      }
```

Die oben dargestellte Funktion *finalizeSurrounding()* beendet den Scanvorgang und stellt das *understandingCustomMesh* fertig. Dabei wird das *Mesh* vollständig geschlossen. Der *SurfaceObserver* wird angehalten, da er Auf-

grund der *understandingCustomMesh* nicht mehr benötigt wird.

Nun können die *PlacementQuerys* zum Platzieren von Objekten an bestimmten Orten im Raum verwendet werden. Der *bool readyToPlaceLava* wird in der Klasse *LavaPlacement* auf true gesetzt. Dadurch wird ein *PlacementQuery* für die Positionierung der Lava auf dem Boden ausgeführt. Die Farbe des Materials des *understandingCustomMesh* wird auf Schwarz gesetzt. Die *HoloLens* ist nicht in der Lage die Farbe *Schwarz* auszugeben. Das *understandingCustomMesh* muss weiterhin visualisiert werden, da sonst die *Plane* der *Lava* komplett angezeigt wird und der Eindruck verloren geht, dass der Boden aus *Lava* besteht. Auf den Abbildungen 8 und 9 ist die *understandingCustomMesh* weiterhin zu sehen, da das Bild mit dem *Live Preview* des *Device Portals* der *HoloLens* über den *Browser* gemacht wurde. Der Nutzer der *HoloLens* sieht dieses *Mesh* nicht.

Weiterhin findet ein Austausch des *StartMenus* durch das *GameMenu* statt. Der *GameManager* erhält die Nachricht, dass die Umgebung erstellt wurde, sodass das Spiel losgehen kann. Zudem muss der Nutzer sich drei Sekunden lange gedulden, bis er die *Startflagge* ablegen kann. Dies hat zum einen den Grund, dass es einen kurzen Augenblick dauert bis die *Lava* auf dem Boden positioniert wurde. Zum anderen wird damit sicher gestellt, dass die Flagge nicht mit der selben *Select-Geste* positioniert wird, welche den *Button "Finalize Scan"* ausgelöst hat.

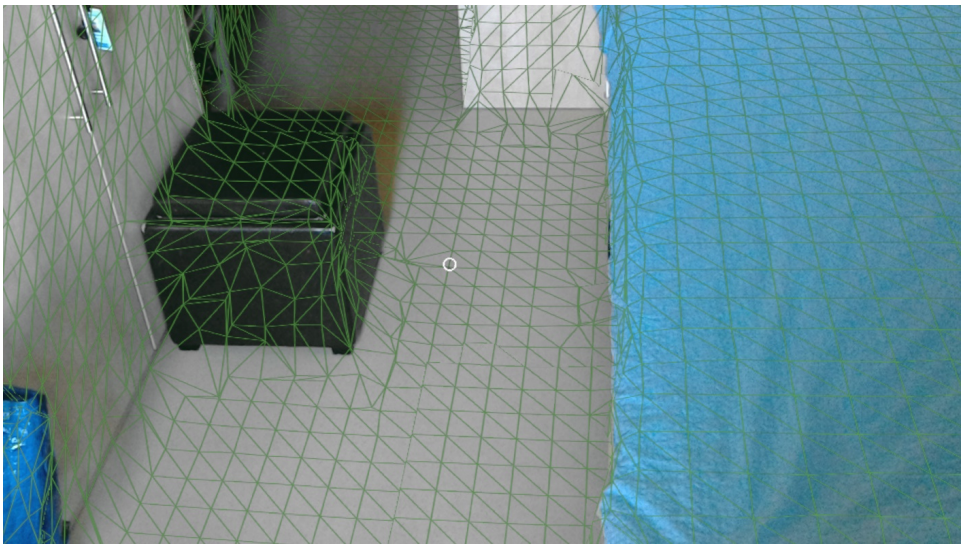

Abbildung 7: Dieses Bild zeigt die Applikation während der Aufnahme des *SpatialUnderstandingCustomMesh*.

3.4 Gestensteuerung

Die *Select-Geste* sowie die *Hold-And-Release-Geste* wurden beide implementiert.

Bei der Gestensteuerung muss zwischen der Gestensteuerung der Menüs und der Gestensteuerung des restlichen Spiels differenziert werden, da sie unterschiedlich verarbeitet werden. Die Gestensteuerung des Menüs wird in Abschnitt 3.2 beschrieben.

Die Klasse *InputHandler* kümmert sich um die Verarbeitung der Gesten zum Positionieren und Löschen von Hologrammen und um die Steuerung des *Avatars*. Sie implementiert die beiden Schnittstellen *IInputClickHandler* und *IHoldHandler*. Zudem wird ein *GlobalListener* hinzugefügt, welcher die Klasse über jedes, durch eine Geste ausgelöstes, *Event* informiert. Dabei spielt es keine Rolle, ob die *Events* schon einmal verwendet wurden.

IInputClickHandler liefert die Methode *OnInputClicked()*, welche bei der Registrierung der *Select-Geste* aufgerufen wird. Innerhalb dieser Methode finden einige Überprüfungen statt, welche je nach Zustand des Spiels zu einer unterschiedlichen Verwendung der *Select-Geste* führen. Zum Beispiel wird die *Select-Geste* für die Platzierung einer *Box* verwendet, wenn der *Gaze* ein virtuelles Objekt erfasst hat, das anvisierte Objekt kein *Button* ist und die Umgebung fertiggestellt wurde. Wichtig an dieser Stelle ist die Überprüfung, ob das *Event* schon verwendet wurde, da es sonst zu einer zweifachen Ausführung kommt. Leider kann diese Überprüfung nicht mit den Gesten für das Menü verwendet werden, da die Typen der *Events* unterschiedlich sind.

Die *Hold-And-Release-Geste* wird zum Löschen einer bereits gebauten *Box* verwendet. Während die *Select-Geste* nur eine Methode benötigt, benötigt die *Hold-And-Release-Geste* drei. Die Schnittstelle *IHoldHandler* implementiert diese drei Methoden. Die erste Methode *OnHoldStarted()* wird aufgerufen, wenn die *Select-Geste* über einen gewissen Zeitraum gehalten wird und ein *Hold* entsteht. Falls der *Gaze* eine *Box* erfasst hat, wird diese rot gefärbt. Wenn der *Hold* gelöst wird, dann wird die Methode *OnHoldComplete()* aufgerufen, welche wiederum die Methode zum Löschen eines Hologrammes aufruft. Falls der *Hold* misslingt, z.B. indem der Nutzer während des *Holds* die Hand aus dem *Gesture Frame* nimmt, wird die Methode *OnHoldCanceled()* aufgerufen. Wurde eine Box rot gefärbt, so wird ihre Farbe nun wieder auf die abgespeicherte, ursprüngliche Farbe zurückgesetzt.

3.5 Sprachsteuerung

Eine Integration von Sprachbefehlen ist dank des *Keyword Managers* sehr simpel. Ein neuer Sprachbefehl lässt sich im *Inspector-Fenster* des *Keyword Managers* mit drei Schritten erstellen. Zunächst muss *Size* um eins erhöht werden. *Size* gibt die Anzahl der Sprachbefehle des *Keyword Managers* an.

Nun kann dem neuen Feld ein *Keyword* als *String*, welcher den *Sprachbefehl* darstellt, übergeben werden. Eine Methode wird diesem zugeordnet und eine neuer Sprachbefehl existiert. Um die Funktionalität zu gewährleisten ist es wichtig, dass unter *Player Settings* das *Microphone* aktiviert ist und es einen *Audio Listener* in der Applikation gibt. Insgesamt wurden neun Sprachbefehle erstellt. Vier davon sind für die *Buttons* der Menüs und die restlichen fünf Sprachbefehle sind für die Platzierung, Löschung von *Boxen* und die Steuerung des *Avatars* bestimmt. Beim Erstellen der Sprachbefehle wurde auf die Einhaltung der Design-Prinzipien von *Microsoft* geachtet. Zum Beispiel bestehen sie immer aus mindestens zwei Silben und ähneln sich nicht in ihrem Klang.

3.6 Positionierung der Lava

Nachdem die Aufnahme und somit die endgültige Erstellung des *understandingCustomMeshs* beendet wurde, wird die Methode *placeLava()* aus der Klasse *LavaPlacement* aufgerufen. *LavaPlacement* beinhaltet das *Lava GameObject*, welches aus einem geplätteten Würfel mit einem *Lava Material* besteht.

In der Methode *placeLava()* wird ein *PlacementSolver* aus dem *MixedRealityToolkit-Unity* verwendet. Der *PlacementSolver* sucht bestimmte Bereiche in dem *understandingCustomMesh*. Diese Bereiche lassen sich durch eine Vielzahl von Parametereinstellungen genau definieren.

Zum Beispiel kann mit der *ObjectPlacementDefinition* eingestellt werden, an welchen Orten (Boden, Wand, Möbel, etc.) nach der Position gesucht wird. Darüber hinaus kann mit den *ObjectPlacementRules* definiert werden, wie weit die Position von anderen Orten, Objekten, etc. minimal entfernt sein soll. Mit dem *ObjectPlacementConstraint* können weitere Einstellungen vorgenommen werden, die die Entfernung bestimmen, in welcher die Position im Bezug auf bestimmte Orte gesucht werden soll. Wenn alle Parameter eingestellt wurden, kann die Position mit Hilfe der Methode *Solver_PlaceObject()* gesucht werden. Wurde eine Position gefunden, welche die Bedingungen erfüllt, so ist der zurückgegebene *Integer-Wert* > 0. Das Ergebnis wird dann als *ObjectPlacementResult* zurückgegeben. Darin befindet sich unter anderem die Position und Ausrichtung.

Im Falle der Positionierung der *Lava* wurden die Parameter so gewählt, dass die *Lava* auf dem Boden positioniert wird. Die *Lava* wurde so groß angelegt, dass sie für unterschiedliche Räume passt. Daher sind genauere Einstellungen von den Parametern des *PlacementSolvers* an dieser Stelle nicht nötig.

3.7 Positionierung und Löschung von Hologrammen im realen Raum

Während der Phase des *Builds*, in welcher der Nutzer sich eine Brücke mithilfe der *Boxen* baut, können diese positioniert und gelöscht werden. Aufgerufen durch die jeweiligen Gesten oder Sprachbefehle werden die Methoden dazu aus der Klasse *HologramManager* ausgeführt. Diese Klasse beinhaltet alle *GameObjects*, welche instanziiert oder gelöscht werden können.

Bei der Positionierung der *Boxen* wird dem Nutzer zunächst die *BoxOnHologram* oder die *BoxOffHologram* angezeigt. Diese beiden *Boxen* sind identisch mit der *BuildBox*, jedoch haben sie keinen *Collider*. Führt der Nutzer nun die *Select-Geste* oder den *Sprachbefehl "Build Box"* aus, so wird für die Geste zunächst im *InputHandler* geschaut, ob alle erforderlichen Bestimmungen erfüllt sind. Wenn dies der Fall ist, wird die Methode *PlaceHologram()* im *HologramManager* aufgerufen. Hier finden weitere Prüfungen auf den Zustand des Spiels statt. Wenn diese Zustände eingehalten wurden, dann wird die *BuildBox* an der Position des *Gaze* instanziiert. Dieser *Box* wird eine zufällige Farbe aus einem bestimmten Bereich aus dem *HSV Farbraum* zugewiesen. Die erzeugten Farben sind hell und stark gesättigt, damit sie für den Nutzer gut sichtbar sind.

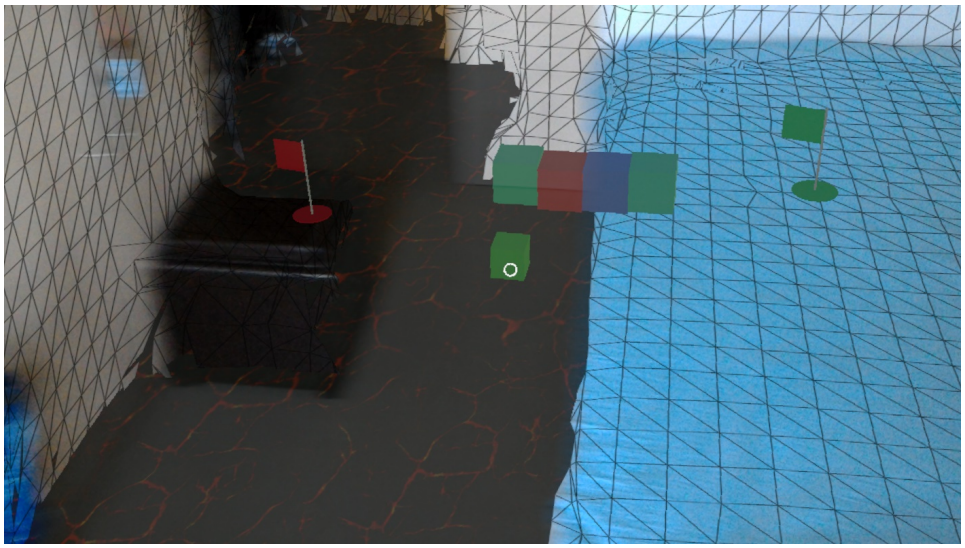

Abbildung 8: Dieses Bild zeigt die Phase des *Builds*.

Die Positionierung der *Startflagge* sowie der *Zielflagge* haben, aufgrund der besseren Übersichtbarkeit, eigene Methoden. Da der Nutzer die Möglichkeit haben soll, die Position der Flaggen während des *Builds* zu verändern, wurde der *bool startAndEndPointPlaced* eingeführt. Ist dieser Wert *true*, so können *Boxen* platziert werden. Solange er jedoch *false* ist, also eine

oder beide Flaggen nicht positioniert sind, wird dem Nutzer die entsprechende Flagge angezeigt. Diese muss positioniert werden, bevor die *Boxen* gebaut, oder der *Avatar* gesteuert werden kann.

Hat der Nutzer vor, eine *Box* oder eine Flagge zu löschen, so kann der *Sprachbefehl "Delete Box"* oder die *Hold-And-Release-Geste* ausgeführt werden. Die Methode zum Löschen stammt wieder aus dem *HologramManager*. Die einzigen *GameObjects*, welche gelöscht werden können, sind die *BuildBox* die *StartFlag* und die *Endflag*, welche mit *DeletableBox, DeletableStartFlag* oder *DeletableEndFlag* gekennzeichnet (*tagged*) sind. Wird diese Methode für eine *Box* aufgerufen, so wird sein instanziiertes *GameObject* zerstört. Wenn es sich um eine Flagge handelt, so werden zusätzlich einige Parameter geändert, sodass der Nutzer die Flagge zuerst wieder positionieren muss.

3.8 Steuerung des Avatars

Beide Konzepte aus Abschnitt 2.3 zur Steuerung des *Avatars* wurden umgesetzt. Sobald der Nutzer den *Button "Activate Avatar"* betätigt oder den entsprechenden Sprachbefehl nennt, verschwindet die *Box* vom *Gaze* und der *Avatar*, eine weiße Kugel, erscheint an der *Startflagge*.

In der Klasse *AvatarManipulation* befinden sich alle Methoden für das erste Steuerungskonzept. Dieses wird gestartet, indem der Nutzer den *Sprachbefehl "Follow my Gaze"* äußert. Dadurch wird der *bool followGaze* auf *true* gesetzt. In der *Update()-Methode*, welche jeden *Frame* ausgeführt wird, wird die Methode *MoveFollow()* aufgerufen, solange *followGaze = true* ist. In dieser Methode wird die Bewegung von der Position des *Avatars* zur Position des *Gaze* berechnet. Somit bewegt sich der *Avatar* nun gleichmäßig und geradlinig auf die Position des *Gaze* zu. Während die Bewegung in *x-Richtung* und *z-Richtung* nur eine Veränderung der *Translation* ist, wird bei dem Sprung des *Avatars* der *Rigidbody* manipuliert. Sobald der *Avatar* den Befehl zum Springen bekommt, wird dessem *RigidBody* ein Kraftimpuls in *y-Richtung* gegeben, sodass der *Avatar* nach oben springt und kurz darauf wieder hinunterfällt. Wichtig an dieser Stelle ist, dass Funktionen, welche einen *Rigidbody* beeinflussen, in *FixedUpdate()* aufgerufen werden müssen. Der Grund dafür ist, dass die Berechnungen in *Unitys physics engine* in bestimmten Zeiteinheiten ausgegeben werden, die nicht mit jedem *Frame* synchron sind.

Die Klasse *AvatarManipulation* sorgt ebenfalls dafür, dass der *Avatar* nicht springen kann, solange er sich in der Luft befindet. Erst wenn eine Kollision aufgetreten ist, kann er wieder springen.

Die *Velocity* beschreibt in einem *Vector3* die Bewegungsgeschwindigkeit, welche in *Unitys physics engine* berechnet wird. Diese wird während einer Kollision des *Avatars* mit einem anderen Objekt auf *(0,0,0)* gesetzt. Sobald eine Kollision aufgehoben wird, wird die *Velocity* wieder normal

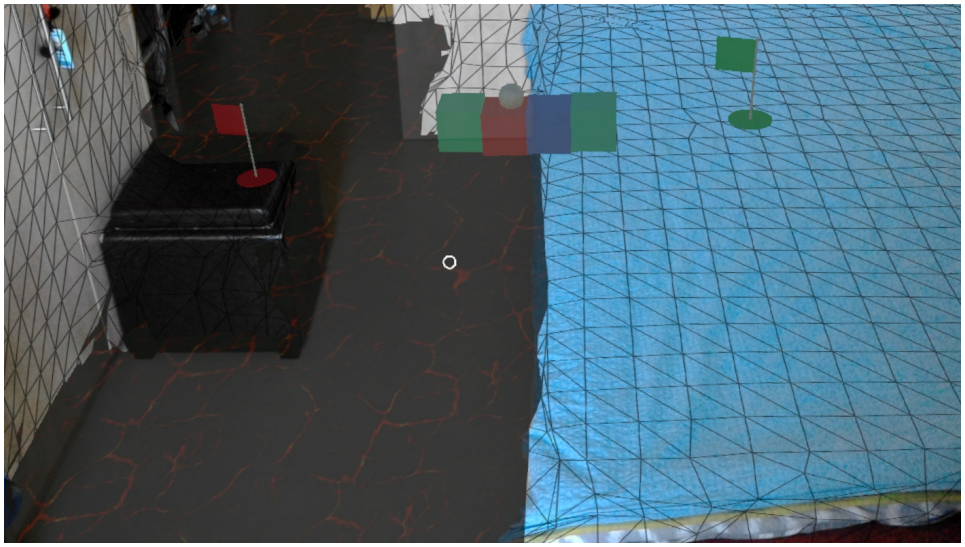

Abbildung 9: Dieses Bild zeigt die Steuerung des *Avatars*.

berechnet, sodass der *Avatar* zu Boden fallen kann. Der Nutzer hatte, während die *Velocity* normal berechnet wurde, keine angemessene Kontrolle mehr über den *Avatar*. Dieser konnte schon geringe Neigungen nicht bewältigen. An den Rändern von Objekten wie Tischen, wo die Neigung der Dreiecke des *understandingCustomMeshs* abfällt, ist der *Avatar* abrupt hinuntergefallen.

Deswegen findet während der Bewegung des *Avatars* in *X-Richtung* und *Z-Richtung* lediglich eine Translation des Koordinatensystems statt. Die Bewegung des *Avatars* ist somit keine realistisch, rollende Bewegung, jedoch hat der Nutzer ein höheres Maß an Kontrolle.

Für die Steuerung mit dem *GamePad* wurde eine eigene Klasse namens *ControllerManipulator* geschrieben. *HoloLensXboxControllerInput* liefert die Grundlage, die Knöpfe und Steuersticks eines entsprechenden *GamePads* auszulesen und zu verwenden. Dazu muss ein Objekt von *controllerInput* angelegt werden. Über dieses Objekt können nun die Zustände der Knöpfe und Steuerungssticks abgefragt werden.

Für den Sprung des *Avatars* wird die gleiche Methode wie im ersten Steuerungskonzept verwendet. Der Befehl

controllerInput.GetButtonDown(ControllerButton.A) liefert *true* zurück, falls der Button gedrückt wird und *false*, falls er nicht gedrückt wird. Dies bedeutet, dass *true* zurückgegeben wird, solange der Knopf gedrückt wird. Damit der *Avatar* beim Drücken des *Buttons* nur einmal springt, wird ein *bool* namens *used* auf *true* gesetzt. Sobald der Knopf losgelassen wird, wird *used* wieder auf *false* gesetzt.

Für die Bewegung in *x-* und *y-Richtung*, werden zunächst die *float-Werte* der Steuerungssticks ausgelesen. Durch die Multiplikation mit einem Fak-

tor *0.01* wird die spätere Bewegung abgeschwächt. Die Achsen der Steuerungssticks werden auf den Achsen der Kamera angeglichen, damit der *Avatar* sich immer in Bezug zur *HoloLens* bewegt. Nun wird der erzeugte *Vector3* auf die Position des *Avatars* addiert und der *Avatar* bewegt sich.

4 Evaluation

Der zentrale Punkt dieser Evaluation ist es ein Verständnis dafür zu erlangen, ob sich die Interaktionsmöglichkeiten der *HoloLens* in der Praxis behaupten können und welche Arten der Interaktion dem Nutzer am meisten zusagen. Zum Einen sollen die Interaktionsarten wie *Gaze*, *Gesture* und *VoiceInput* auf die *Leichtigkeit* ihrer Anwendung geprüft werden. Zum Anderen sollen diese Interaktionsarten auf ihre *Anforderungsgerechtigkeit* in unterschiedlichen Bereichen des Spiels untersucht werden. *Anforderungsgerechtigkeit* soll an dieser Stelle bedeuten, in wie weit die Art der Interaktion zu ihrem jeweiligen Einsatzgebiet passt. Darüber hinaus wird untersucht, wo die Probanden die Zukunft dieser Interaktionsarten sehen.

4.1 Vorgehen

Zunächst werden die Probanden über den Ablauf sowie über den Hintergrund der Evaluation unterrichtet. Dabei werden einige Begriffe wie z.B. *AR* oder *Gaze* erklärt. Die Gesten werden kurz demonstriert und geübt. Zu Beginn muss der Proband den ersten Teil des Fragebogens beantworten. Nachdem dieser ausgefüllt ist, wird die *HoloLens* auf dem Kopf des Probanden positioniert und die Testphase beginnt. Die Aufgaben werden von dem Testleiter nacheinander vorgelesen, damit der Proband seine Hände für die Gestensteuerung frei bewegen kann. Sobald der Proband mit einer Aufgabe fertig ist, gibt er Bescheid, sodass die nächste Aufgabe bearbeitet werden kann. Mithilfe des *Live Previews* des *Device-Portals* der *HoloLens* hat der Testleiter die Möglichkeit den Fortschritt des Probanden zu beobachten. Es wird über eine WLAN-Verbindung über den Computer in einem *Webbrowser* abgespielt. Wobei das *Live Preview* mit einer Verzögerung von zehn Sekunden angezeigt wird. Sobald die Aufgaben bewältigt worden sind, muss der Proband den restlichen Fragebogen bearbeiten. Nachdem dieser vollständig ausgefüllt wurde, ist die Testphase beendet.

4.2 Fragebogen

Der Fragebogen setzt sich aus drei Teilen zusammen. Der erste Teil des Fragebogens soll vor der Evaluation mit der *HoloLens* ausgefüllt werden. Er befasst sich mit grundlegenden Fragen z.B. zum Alter und Geschlecht des Probanden. Zudem werden schon gezielt Fragen über die Vorerfahrung im Bereich der *AR* gestellt. Der zweite und dritte Teil des Fragebogens soll ausgefüllt werden, nachdem der Proband die Testphase mit der *HoloLens* abgeschlossen hat. Im zweiten Teil muss der Proband Fragen über die Interaktionsarten, zum Einen im Allgemeinen und zum Anderen zu expliziten Bereichen des Spiels, beantworten. Im dritten Teil des Fragebogens geht es um die Umsetzung der Aufgaben sowie um die Einschätzung der Zu-

kunftsaussichten der Steuerungsarten. Der Proband hat an einigen Stellen im Fragebogen die Möglichkeit Anmerkungen zu hinterlassen 6.

4.3 Aufgabenstellung

Im Raum werden drei Bilder zur Hilfestellung angebracht. Ein Bild zeigt den Schriftzug *Menü* und wird an einer Wand befestigt. Die beiden Anderen zeigen jeweils einen *Start-* sowie einen *Zielpunkt* an und werden auf zwei auseinander stehenden Tischen platziert. An die Positionen der Bilder sollen die jeweiligen Objekte positioniert werden. Am Anfang muss der Proband das *StartMenu* an der vorgegebenen Position an der Wand ablegen. Bevor und während der Proband die Aufnahme der Umgebung vornimmt, weist der Testleiter darauf hin, wie die Aufnahme am besten gelingt. Ist die Aufnahme beendet, so wird die Spielumgebung fertiggestellt.

Nun muss der Nutzer die virtuellen Flaggen an der jeweiligen Position im realen Raum ablegen. Danach besteht die Aufgabe des Nutzers darin, sich mithilfe von *Boxen* einen Weg zu bauen, um den *Avatar* zum *Ziel* zu befördern. Dabei darf der *Avatar* nicht den Boden berühren. Berührt der *Avatar* den Boden, so erscheint er wieder an der *Startflagge*. Die Testperson wird darauf hingewiesen, mit welchen Gesten und Sprachbefehlen er die Brücke bauen kann. Erscheint dem Probanden die Brücke als fertig, so kann die Steuerung des *Avatars* beginnen. Das Ziel besteht, darin den *Avatar* mit den beiden unterschiedlichen Steuerungskonzepten vom *Start* zum *Ziel* zu befördern (Abschnitt 2.3).

Die Konzepte werden erklärt und der Proband kann die Navigation beginnen. Durch das Menü hat er die Möglichkeit zur *Build-Phase* zurück zu gelangen und mögliche Fehler auszubessern. Während der Ausführung der Aufgaben wird der Proband darauf hingewiesen, alle Steuerungskonzepte mehrere Male zu verwenden. Danach kann er intuitiv selbst entscheiden, mit welcher er weitermachen möchte.

4.4 Hypothesen

Die im Vorfeld aufgestellten Hypothesen sind die Folgenden.

Hypothese 1 Die Steuerung des *Avatars* mithilfe des *GamPads* wird den Probanden am leichtesten fallen, da diese Steuerungsart durch ihr langes Dasein intuitiv ist.

Hypothese 2 Die Steuerung des Menüs mithilfe der Gesten und Sprachsteuerung wird von dem Nutzer schnell akzeptiert und intuitiv verwendet.

Hypothese 3 Die neuen Interaktionsmöglichkeiten werden mit Interesse getestet und positiv von den Probanden aufgenommen.

4.5 Ergebnisse

Insgesamt wurde die Evaluation mit 14 Probanden durchgeführt. Drei dieser 14 Probanden waren weiblich (Abbildung 18). Das Durchschnittsalter lag bei 23 Jahren, wobei der älteste Proband 32 Jahre und der jüngste Proband 19 Jahre alt war (Schriftliche Antworten 6). 64.29% der Probanden gaben an *häufig* Computerspiele zu spielen, womit diese zur Zielgruppe der *Gamer* gehören (Abbildung 19). 85.71% haben den Begriff *AR* oder *VR* schon einmal gehört und acht der 14 Probanden haben bereits zuvor einen *HMD* getragen (Abbildung 21, 15). Somit hat mehr als die Hälfte der Probanden Erfahrung mit Technologien aus der *AR* bzw. *VR*.

Trotz der Bedenken im Vorfeld, dass die Probanden die Aufnahme des *understandingCustomMesh* nicht selbst machen sollten, da der Fortlauf der Evaluation unter Anderem von der Qualität der Aufnahme abhängt, konnte jeder Proband die Aufnahme selbst und ordentlich vornehmen. Kleine Fehler in der *understandingCustomMesh* hatten dabei keinen negativen Einfluss auf den Verlauf des Tests.

Die Testpersonen waren, im wesentlichen alle in der Lage die geforderten Aufgaben umzusetzen. Lediglich drei der 14 Probanden haben es nicht geschafft den *Avatar* mittels des Steuerungskonzeptes *Follow My Gaze* von der *Startflagge* zur *Zielflagge* zu navigieren. In zwei dieser drei Fälle ist dies auf die Bauart der Brücke zurückzuführen. Einer dieser Probanden hatte im Allgemeinen sehr viele Schwierigkeiten, die geforderten Aufgaben umzusetzen. Dabei waren nicht nur die Steuerungskonzepte ein Hindernis, sondern die *HoloLens* bzw. eine solch neuartige Technologie im Allgemeinen. Mit ein paar Hilfestellungen konnte diese Testperson dennoch alle restlichen Aufgaben erfüllen.

Leichtigkeit

GamePad Die Steuerung mithilfe des *GamePads* ist den Probanden *am leichtesten* gefallen (Abbildung 10). Alle 14 Probanden haben es innerhalb kürzester Zeit geschafft, den *Avatar* zum Ziel zu befördern. Selbst den drei Probanden, die angaben, sich nicht mit einem GamPad auszukennen, ist die Steuerung leicht gefallen (Abbildung 20). Somit ist *Hypothese 1* eingetroffen. Keine Testperson empfand das *GamePad* als störend. Dies lässt darauf schließen, dass bei einer Navigation eines virtuellen Objektes durch die reale Welt, ein physisches Eingabegerät keinen Einfluss auf eine *immersive* Erfahrung hat.

	1.		2.		3.		4.			
	Σ	%	Σ	%	Σ	%	Σ	%	Ø	±
Gaze	2x	14,29	5x	35,71	4x	28,57	3x	21,43	2,57	1,02
Sprachsteuerung	-	-	2x	14,29	7x	50,00	5x	35,71	3,21	0,70
Gestensteuerung	1x	7,14	5x	35,71	2x	14,29	6x	42,86	2,93	1,07
Gampad	11x	78,57	2x	14,29	1x	7,14	-	-	1,29	0,61

Abbildung 10: Frage 12: Bitte ordnen Sie die unterschiedlichen Steuerungen nach ihrer Leichtigkeit.

Während sich die Probanden, zum Großteil einig sind, dass die Steuerung mit dem *GamePad am leichtesten* ist, so trifft dies auf die übrigen Steuerungsarten nicht zu.

Gaze Mit einem Durchschnitt von 2.57, auf einer Skala von 1 bis 4, wobei 1 *am leichtesten* und 4 *am wenigsten leicht* bedeutet, empfanden die Probanden die Steuerung mit dem *Gaze* am *zweit leichtesten* (Abbildung 10). Alle Bereiche von *am leichtesten* bis *am wenigsten leicht* sind vertreten. Auffällig bei der Evaluation war, dass sich die Probanden, abgesehen vom *Scanvorgang*, kaum bewegt haben, obwohl die *HoloLens* dies durch ihre Unabhängigkeit von Kabeln zulässt. Bei der Bewältigung der Aufgaben haben die meisten versucht alles aus einer Perspektive zu lösen. Nur einem geringen Teil ist aufgefallen, dass durch das Ändern der Perspektive die Aufgaben teilweise deutlich einfacher zu lösen waren.

Gestensteuerung Die Probanden fanden die Gestensteuerung mit einem Durchschnitt von 2.93 am *dritt leichtesten* (Abbildung 10). Wieder sind alle Antwortmöglichkeiten von *am leichtesten* bis *am wenigsten leicht* vertreten. Das Schwierigste bei der Gestensteuerung ist die richtige Ausführung der Geste. Der Nutzer muss diese zusätzlich an der richtigen Positi-

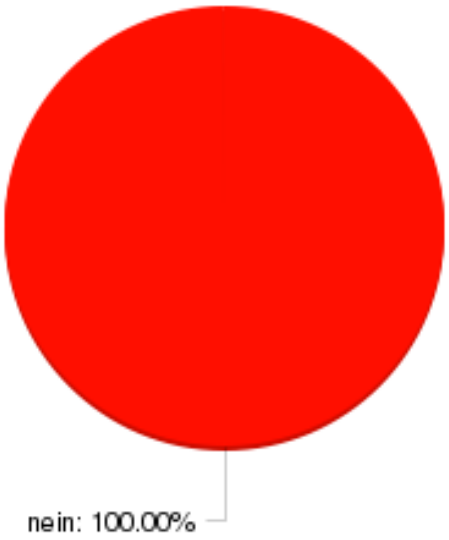

Abbildung 11: Frage 16: Haben Sie das Gamepad als störend empfunden?

on, sprich im *Gesture Frame*, ausführen, damit diese erkannt wird. Einige Probanden ist es unerwartet schnell gelungen, die *Select-Geste* korrekt auszuführen, während die meisten anfangs Schwierigkeiten bei der richtigen Ausführung hatten. Hinzu kam, dass die Probanden die Gesten teilweise richtig ausgeführt haben, sich die Hand jedoch nicht im *Gesture Frame* befunden hat. Daraus kann geschlossen werden, dass es eine gewisse Zeit und Übung braucht, bis ein unerfahrener Nutzer die Gestensteuerung gut anwenden kann. Für die Probanden war es schwer, die *Select-Geste* oder die *Hold-And-Release-Geste* richtig auszuführen, darauf zu achten, dass sich die Hand im *Gesture Frame* befindet und gleichzeitig die Aufgaben gewissenhaft zu bearbeiten. Dennoch wurden schon in der kurzen Zeit der Evaluation Fortschritte in diesem Prozess beobachtet.

Spracheingabe Obwohl 64.29% die Frage *Sind Sie mit einer Interaktion über eine Spracheingabe vertraut (z.B. Siri, Cortana, Alexa)?* mit *Ja* (Abbildung 16) beantwortet wurden und immerhin 50% eine Spracheingabe schon einmal verwendet haben, ist sie dennoch als *am wenigsten leicht* empfunden worden (Abbildung 10). Die Standardabweichung ist hier relativ gering, somit stimmen die Meinungen der Probanden zur Spracheingabe im wesentlichen überein. Da die Spracherkennung der *HoloLens* für eine U.S. amerikanische Aussprache ausgelegt ist, benötigt der Nutzer eine ähnlich gute Aussprache, damit die Spracherkennung funktioniert. Diese Tatsache hat bei manchen Sprachbefehlen zu Komplikationen geführt. Der Sprachbefehl *"Activate Avatar"* hat sich als besonders problematisch her-

ausgestellt. Bei vielen Probanden hat er selbst nach mehrmaliger Nennung nicht funktioniert. An dieser Stelle wäre entweder eine bessere Spracherkennung oder angemessenere Sprachbefehle von Nöten.

	sehr gut (1)		gut (2)		mittel (3)		eher weniger (4)		gar nicht (5)			
	Σ	%	Σ	%	Σ	%	Σ	%	Σ	%	Ø	±
Allgemein	2x	14,29	8x	57,14	3x	21,43	1x	7,14	-	-	2,21	0,80
in Bezug auf die Menüführung	6x	42,86	7x	50,00	-	-	1x	7,14	-	-	1,71	0,83
in Bezug auf die Positionieru…	5x	35,71	7x	50,00	1x	7,14	1x	7,14	-	-	1,86	0,86
in Bezug auf die Steuerung d…	2x	14,29	8x	57,14	1x	7,14	3x	21,43	-	-	2,36	1,01

Abbildung 12: Frage 15: Inwiefern sehen Sie die Steuerung mit dem Gaze als Anforderungsgerecht?

Anforderungsgerecht In den folgenden Tabellen hatten die Probanden die Möglichkeit den Interaktionsarten Noten, in Bezug auf die *Anforderungsgerechtigkeit*, für die unterschiedlichen Bereiche des Spiels zu geben. Die Noten reichen hier von 1 *sehr gut* bis 5 *gar nicht*.

Beim Vergleich der drei Tabellen 12, 13 und 14 wird deutlich, dass die Menüführung mit allen Steuerungsarten gut abgeschnitten hat. Somit kann an dieser Stelle die zweite *Hypothese* als teilweise eingetroffen betrachtet werden. Wobei eine intuitive Steuerung des Menüs nicht der Fall ist. Auffällig ist hier der Proband mit den vielen Schwierigkeiten, da er sich oft als Einziger für *eher weniger* entschieden hat.

Gestensteuerung Allgemein hat die Gestensteuerung mit einer Durchschnittsnote von 1.71 abgeschlossen. In Bezug auf die Menüführung erreicht die Gestensteuerung die beste Note mit 1.50. Auch in den Bereichen

	sehr gut (1)		gut (2)		mittel (3)		eher weniger (4)		gar nicht (5)			
	Σ	%	Σ	%	Σ	%	Σ	%	Σ	%	Ø	±
Allgemein	5x	35,71	8x	57,14	1x	7,14	-	-	-	-	1,71	0,61
in Bezug auf die Menüführung	9x	64,29	4x	28,57	-	-	1x	7,14	-	-	1,50	0,85
in Bezug auf die Positionieru...	3x	21,43	7x	50,00	3x	21,43	1x	7,14	-	-	2,14	0,86
in Bezug auf die Steuerung d...	4x	28,57	5x	35,71	3x	21,43	2x	14,29	-	-	2,21	1,05

Abbildung 13: Frage 13: Inwiefern sehen Sie die Gestensteuerung als Anforderungsgerecht?

	sehr gut (1)		gut (2)		mittel (3)		eher weniger (4)		gar nicht (5)			
	Σ	%	Σ	%	Σ	%	Σ	%	Σ	%	Ø	±
Allgemein	4x	28,57	5x	35,71	5x	35,71	-	-	-	-	2,07	0,83
in Bezug auf die Menüführung	6x	42,86	7x	50,00	-	-	1x	7,14	-	-	1,71	0,83
in Bezug auf die Positionieru...	4x	28,57	4x	28,57	5x	35,71	1x	7,14	-	-	2,21	0,97
in Bezug auf die Steuerung d...	5x	35,71	3x	21,43	4x	28,57	2x	14,29	-	-	2,21	1,12

Abbildung 14: Frage 14: Inwiefern sehen Sie die Sprachsteuerung als Anforderungsgerecht?

der Positionierung der *Boxen* und der Steuerung des *Avatars* hat die Gestensteuerung mit *gut* abgeschnitten. Wobei die Meinungen wieder etwas weiter auseinander gehen. In Bezug auf die Steuerung des *Avatars* sind die unterschiedlichen Meinungen besonders bemerkbar. Die etwas schwierigere *Hold-And-Release-Geste* wurde unerwartet gut ausgeführt und von manchen Probanden sogar dem Sprachbefehl *"Delete Box"* vorgezogen.

Sprachsteuerung Die Ergebnisse der Sprachsteuerung ähneln größtenteils denen der Gestensteuerung, wobei die Meinungen geringfügig weiter auseinander gehen und die Durchschnittsnoten geringfügig schlechter sind. Am auffälligsten ist die allgemeine Note, welche mit 2.07 um einiges schlechter als die der Gestensteuerung und dennoch ziemlich gut ist.

Gaze Der Gaze hat im Allgemeinen mit einer Durchschnittsnote von 2.21 abgeschlossen. In Bezug auf die Positionierung der *Boxen*, hat er die beste Note mit 1.86. Bezüglich der Steuerung des *Avatars* schneidet der *Gaze* am schlechtesten mit 2.36 ab.

Zukunftsaussichten Acht der 14 Probanden würden in Zukunft gerne mehr Gebrauch von der Gestensteuerung und der Sprachsteuerung ma-

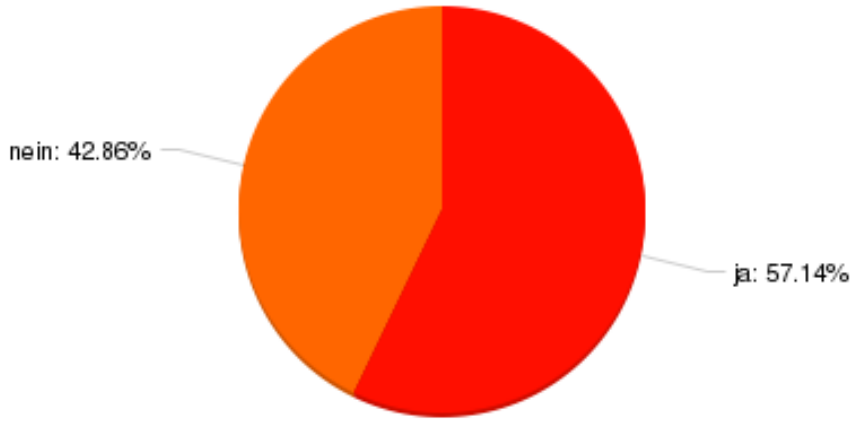

Abbildung 15: Frage 7: Haben Sie schon einmal einen Head-Mounted Display getragen?

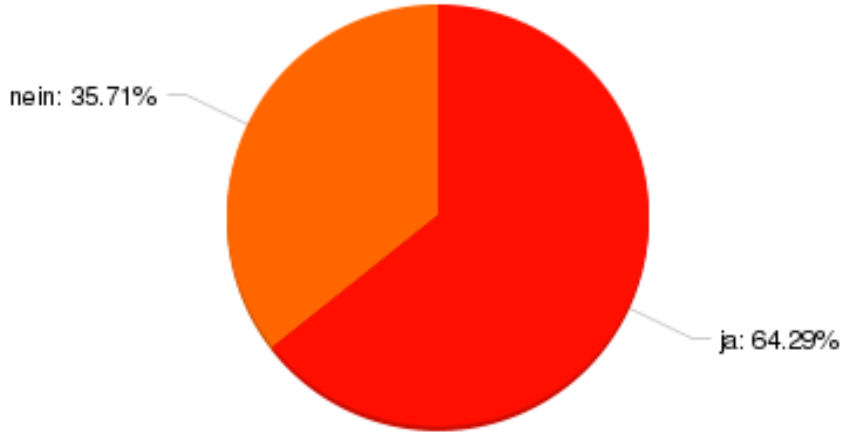

Abbildung 16: Frage 8: Sind Sie mit einer Interaktion über eine Spracheingabe vertraut (z.B. Siri, Cortana, Alexa)?

chen, wobei nur sechs den *Gaze* zukünftig verwenden möchten 28. Ein Proband merkte an, dass er sich wünschen würde, dass der *Gaze* nicht auf Augenhöhe, sondern etwas tiefer liegen sollte 6. Ein Anderer schrieb, dass es ihn gestört hat, dass sich der Gaze beim nach vorne Schauen bewegt hat 6. Während der Evaluation hat der Großteil der Probanden die unterschiedlichen Steuerungsarten mit Interesse ausprobiert. Bei der Positionierung der *Boxen* und der Menüführung haben sich viele entweder für die Sprachsteuerung oder für die Gestensteuerung entschieden. Dabei ist keine der beiden Steuerungsarten auf ersichtliche Weise bevorzugt worden. Somit ist *Hypothese 3* teilweise eingetroffen. Die Probanden haben die Interaktionsmöglichkeiten mit Interesse getestet, jedoch unterschiedlich gut aufgenommen.

Spiel Allgemein Der Großteil der Probanden fand das Spiel *sehr gut* oder *gut*. Lediglich einem Probanden hat das Spiel *nicht so gut* gefallen, da er große Schwierigkeiten in allen Bereichen hatte (Abbildung 17). Die Umsetzung der Aufgaben ist den Probanden im Groben entweder *leicht* oder *mittel schwer* gefallen (Abbildung 27). Dabei fanden zwei der 14 Testpersonen die Aufgaben sogar *sehr leicht*. Nur ein Proband fand die Aufgaben *schwer*.

Einige Probanden haben zu der Frage *Was hat Ihnen besonders an dem Spiel gefallen oder was empfanden Sie als besonders störend?* Anmerkungen hinterlassen 6. Während eine Person die Gestensteuerung und den *Gaze* als störend empfunden hat, schrieben andere, dass ihnen z.B. die Gestensteuerung, die Neuartigkeit des Spiels oder die unterschiedlichen Bedienungsmöglichkeiten gefallen haben. Zudem wurde angemerkt, dass leichte Kanten zwischen den Boxen die Steuerung des *Avatars* beeinträchtigt haben und dass eine Drehfunktion bei der Positionierung der Boxen hilfreich sein könnte.

4.6 Zusammenfassung der Ergebnisse

Allgemein ist der Test ohne große Schwierigkeiten abgelaufen.

Die Sprachsteuerung wurde zwar mit Interesse getestet, ist jedoch wegen ihrer unzureichenden Ausführbarkeit nicht so gut ausgefallen, obwohl der Großteil der Probanden mit einer Sprachsteuerung vertraut war (Abbildung 22). Dieses Ergebnis ist auf die schwierige Umsetzung der Sprachbefehle zurückzuführen, was wiederum auf die Aussprache und damit auf die Spracherkennung selbst zurückzuführen ist. Eine geeignetere Spracherkennung oder simplere Sprachbefehle wären von Nöten.

Bei allen Steuerungsarten ist die Meinung bezüglich der Steuerung des *Avatars* weiter auseinander gegangen als in den anderen Bereichen. Dies könnte ein Zeichen dafür sein, dass neue Navigationskonzepte Zeit brauchen, um sich zu etablieren. Die Zielgruppe der *Gamer* wird wahrschein-

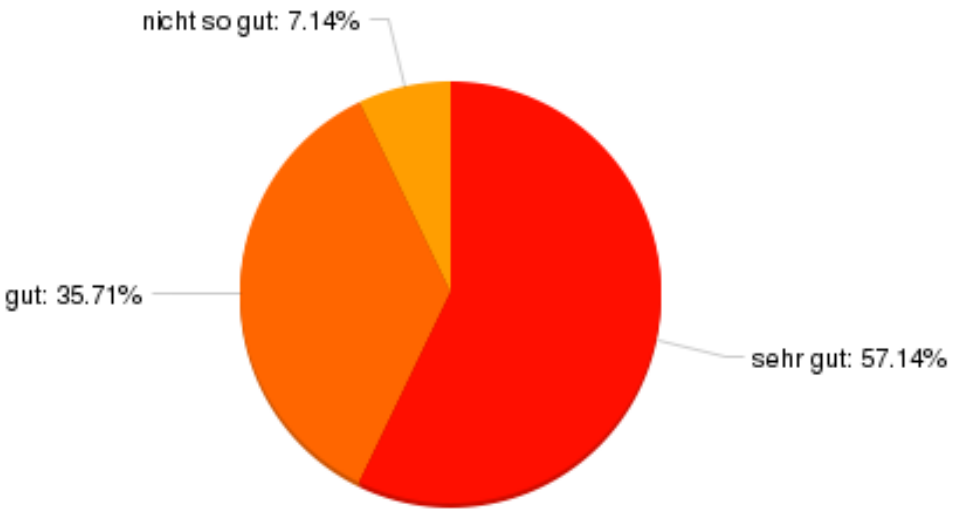

Abbildung 17: Frage 20: Wie gut hat Ihnen das Spiel gefallen?

lich mehr Zeit brauchen, da in ihrem Bereich bereits viele ausgereifte Navigationskonzepte existieren. Jedoch könnte auch vermutet werden, dass es genau anders herum ist, da sich gerade die *Gamer* mit unterschiedlichen Steuerungsarten auskennen und es ihnen deswegen leichter fallen könnte Neue zu erlernen. Die Ergebnisse des *GamePads* bezüglich der *Leichtigkeit* sowie der Frage *Welche Steuerungskonzepte haben Ihnen besonders gefallen?* bestärken die erste Vermutung (Abbildung 25). Bei dieser Frage hat das *GamePad* die meisten Stimmen bekommen.

Obgleich die einzelnen Steuerungsmöglichkeiten unterschiedlich aufgenommen und unterschiedlich gut umgesetzt worden sind, haben alle mit einer *guten* Note bezüglich der *Anforderungsgerechtigkeit* abgeschlossen. Auffällig ist zudem, dass die Probanden alle Steuerungsarten nach der absteigenden Reihenfolge *Menüführung*, *Positionierung der Boxen* und dann *Steuerung des Avatars* gewählt haben. Hier wird wieder erkennbar, dass ein gänzlich neues Konzept zur *Steuerung eines Avatars* Zeit braucht, um akzeptiert zu werden.

Während der Evaluation konnte beobachtet werden, dass die Probanden von der *HoloLens* sehr beeindruckt waren. Eine Ablenkung durch die *HoloLens* selber kann also nicht ausgeschlossen werden. Im Gegenteil, die Neuartigkeit der *HoloLens* die Vielzahl der unterschiedlichen Steuerungsmöglichkeiten sowie das Verstehen des Spielkonzepts haben den Probanden gefordert. Jedoch ist dies nicht gänzlich negativ zu betrachten, da dadurch auch das Interesse und der Ehrgeiz der Probanden geweckt wurde, die Aufgaben zu bewältigen.

Der *Gaze* hat insgesamt gut abgeschlossen, jedoch hat er nur vier Probanden besonders gefallen und nur sechs würden den *Gaze* in Zukunft nut-

zen wollen (Abbildung 25, 28). Wie auch die anderen Interaktionsmöglichkeiten benötigt der Nutzer eine gewisse Zeit, um den *Gaze* zu akzeptieren. Obwohl ein paar Probanden schon einen *HMD* verwendet haben, werden sie kaum Erfahrung mit einem Auswahlwerkzeug haben, welches sie sowie durch die Bewegung des Kopfes als auch des ganzen Körpers steuern können.

Somit kann gesagt werden, dass sich die Interaktionsmöglichkeiten der *HoloLens* in der Praxis behaupten können, auch wenn sie nicht von jedem Nutzer auf Anhieb akzeptiert werden. Besonders in der Menüführung wurden die neuen Möglichkeiten positiv aufgenommen. Die Gestensteuerung wurde insgesamt am besten bewertet, wobei die Sprachsteuerung nicht wesentlich schlechter abgeschnitten hat.

5 Fazit und Zukunftsaussichten

In dieser Arbeit wurde der spielerische Prototyp einer Applikation für die *HoloLens* entwickelt und im Anschluss einer Evaluation unterzogen, um neue Interaktionsarten aus dem Bereich der *AR* und *VR* zu untersuchen.

Im Allgemeinen wurden die entwickelten Interaktionskonzepte und die damit verbundenen, neuen Interaktionsmöglichkeiten der *HoloLens* mit Interesse getestet.

Während der Nutzung der Applikation haben diverse Faktoren und Eindrücke auf den Probanden eingewirkt. Dazu gehört die Neuartigkeit der *HoloLens* sowie das Verstehen des Spielprinzips, der unterschiedlichen Interaktionen und der Interaktionskonzepte. Somit kann an dieser Stelle noch nicht von einer gänzlich intuitiven Interaktion gesprochen werden. Besonders die Interaktionen selbst wie *Gaze, Gesture* und *VoiceInput* sind noch zu fremd. Jedoch konnte beobachtet werden, dass der Großteil der Probanden die Interaktionen schnell verstanden hat und ordentlich ausführen konnte. In Anbetracht des guten Lernprozesses der Probanden sowie deren Interesse für die neuen Interaktionen sind die Zukunftsaussichten dieser Interaktionsarten positiv zu betrachten.

Applikationen, welche in Zukunft für die *HoloLens* entwickelt werden, sollten wohl überlegte und einfache Interaktionskonzepte verwenden.

Bei der Erstellung von Sprachbefehlen ist es wichtig, auf eine einfache Aussprache zu achten. Somit wird sicher gestellt, dass diese auch leicht verwendbar sind. Ebenfalls sollte bei der Positionierung von virtuellen Objekten auf eine Rotationsfunktion und eventuell auf eine Skalierungsfunktion nicht verzichtet werden.

Die Gestensteuerung ist sehr robust und gut anwendbar, solange der Nutzer auf die richtige Ausführung der Geste im *Gesture Frame* achtet.

Konventionelle Eingabegeräte wie z.B. das *GamePad* sollten für eine Interaktion mit virtuellen Objekten im realen Raum nicht vernachlässigt werden. So kann es beispielsweise bei der Steuerung eines *Avatars* sehr zweckmäßig sein und ist intuitiv verwendbar. Zudem konnte eine Einschränkung für eine *immersiven* Erfahrung aufgrund des *GamePads* nicht festgestellt werden.

Abbildungsverzeichnis

Literatur

[1] A. Utsumi F. Kishino P. Milgram, H. Takemura. Augmented reality: A class of displays on the reality-virtuality continuum. Paper, ATR Communication Systems Research Laboratories 2-2 Hikaridai, Seika-cho, Soraku-gun Kyoto 619-02, Japan, 1994.

[2] Grafik des continuum of real to virtual environments. https://en.wikipedia.org/wiki/Reality(November 2017).

[3] G. Akçayir M. Akçayir. Advantages and challenges associated with augmented reality for education: A systematic review of the literature. *The Journal of the European Association for Research on Learning and Instruction (EARLI)*, 2016.

[4] C. Lagman S. Tenn J. V. Demos S. J. Lee T. T. Bui N. E. Barnette N. S. Bhatt N. Ung A. Bari N. A. Martin I. Yang P. E. Pelargos, D. T. Nagasawa. Utilizing virtual and augmented reality for educational and clinical enhancements in neurosurgery. *Journal of Clinical Neuroscience*, 2016.

[5] E. E. Sabelman and R. Lam. The real-life dangers of augmented reality, 2015.

[6] Face swap applikation. https://play.google.com/store/apps/details?id=com.fotoable.faceswap.c403&hl=de (November 2017).

[7] Augmented reality beispiel: Sandbox. http://idav.ucdavis.edu/(tilde)okreylos/ResDev/SARndbox/ (November 2017).

[8] Daqri. https://daqri.com (November 2017).

[9] Meta2. https://buy.metavision.com/?hsCtaTracking=c21ccbfc-840a-453c-9793-eabfa3dabd1d(November 2017).

[10] Hololens features. https://developer.microsoft.com/en-us/windows/mixed-reality/development (November 2017).

[11] Gesture frame. https://support.microsoft.com/de-de/help/12644/hololens-use-gestures (November 2017).

[12] Hololens sptaialunderstanding. https://developer.microsoft.com/en-us/windows/mixed-reality/spatial_mapping_in_unity#holotoolkit.spatialunderstanding (November 2017).

[13] W. Li Z. Huang P. Hui Senior Member IEEE S. Lin, H. F. Cheng and C. Peylo. Ubii: Physical world interaction through augmented reality, 2017.

[14] R. V. Krevelen. Augmented reality: Technologies, applications, and limitations. Technical report, VU University Amsterdam, 2007.

[15] R.Behringer R.Azuma, Y.Baillot. Recent advances in augmented reality, 2001.

[16] J. Hsu. Vr glove powered by finger motions, 2015.

[17] M. S. Bouhlel S. Ameur, A. B. Khalifa. A comprehensive leap motion database for hand gesture recognition, 2016.

[18] R. Hebbalaguppe E. Hassan S. Hegde, R. Perla. Gestar: Real time gesture interaction for ar with egocentric view, 2016.

[19] G. Klinker K. Moustakas M. Bikos, Y. Itoh. An interactive augmented reality chess game using bare-hand pinch gestures, 2015.

[20] A. Trisal V. Keshav U. Dubey J. Bose, A. Singhai. A hands free browser using eeg and voice inputs, 2015.

[21] A. Marin M. Jeong J.P. Robichaud A. Celikyilmaz Y.B. Kim A. Rochette O. Z. Khan X. Liu D. Boies T. Anastasakos Z. Feizollahi N. Ramesh H. Suzuki R. Holenstein E. Krawczyk V. Radostev R. Sarikaya, P. A. Crook. An overview of end-to-end language understanding and dialog management for personal digital assistants, 2017.

[22] A. Rashed N. Nasiri, S. Shirmohammadi. A serious game for children with speech disorders and hearing problems, 2017.

[23] J. Magalhaes S. Cavaco A. Grossinho, I. Guimaraes. Robust phoneme recognition for a speech therapy environment, 2016.

[24] Roboraid (hololens applikation). https://www.microsoft.com/de-de/hololens/apps/roboraid (November 2017).

[25] Young conker (hololens applikation). https://www.microsoft.com/de-de/hololens/apps/young-conker (November 2017).

[26] Actiongram (hololens applikation). https://www.microsoft.com/de-de/hololens/apps/actiongram (November 2017).

[27] Unity. https://unity3d.com/de (November 2017).

[28] Mixedrealitytoolkit-unity. https://github.com/Microsoft/MixedRealityToolkit-Unity (November 2017).

[29] Mrdesignlabs_unity. https://github.com/Microsoft/MRDesignLabs_Unity (November 2017).

[30] Hololensxboxcontrollerinput. https://www.assetstore.unity3d.com/en/#!/content/70068 (November 2017).

6 Anhang

Aufgabenstellung

1) Kurze Erläuterung des Ablaufs.
2) Bearbeitung des ersten Teils des Fragebogens.
3) Einweisung in die Begrifflichkeiten und den Hintergrund der Evaluation.
   1. Kopf ist die Maus (*Gaze*).
   2. Gestensteuerung (*Select, Hold-And -Release-Geste, Gesture Frame*)
   3. Sprachsteuerung
4) Sie können jederzeit Fragen stellen.

5) Erstellen des SpatialUnderstandingCustomMesh´s:

1) Da die Umgebung zu Beginn des Spieles aufgenommen werden muss, kann es sein, dass sich das Menü noch nicht über die Oberflächen realer Objekte bewegen lässt. Warten Sie eine kurze Zeit, und bewegen Sie dabei das Menü über eine Wand. Sobald Sie feststellen, dass sich das Menü entlang dieser bewegt, können Sie fortfahren.
2) Legen Sie das Menü mithilfe der *Select-Geste* an der markierten Position im Raum ab.
   Hinweis: Die *Buttons* des Menüs können durch das anvisieren mittels *Gaze* und der gleichzeitigen Ausführung der *Select-Geste* ausgeführt werden. Zudem können sie auch ohne ein Anvisieren betätigt werden, indem der gleichnamige Sprachbefehl geäußert wird.
3) Bevor Sie nun ihre Spielumgebung aufnehmen, hier noch ein paar Tipps:
   1. Lassen Sie sich Zeit bei der Aufnahme der Umgebung.
   2. Falls Ihnen etwas an einer Stelle falsch vorkommt, betrachten Sie es länger und aus verschiedenen Perspektiven.
   3. Achten Sie darauf den gesamten Raum zu scannen.
   4. Sobald Sie den Eindruck haben, dass der Raum richtig aufgenommen wurde, können Sie fortfahren.
4) Betätigen Sie nun den Button mit der Beschriftung *„Create Surrounding“* und nehmen Sie den Raum auf.
5) Betätigen Sie den Button mit der Aufschrift *„Finalize Scan“*, um die Aufnahme abzuschließen.

   Nun wird die entstandene Aufnahme, sowie die Spielumgebung fertiggestellt.
   Dies kann einen kurzen Moment in Anspruch nehmen.
   Sobald die *Lava* auf dem Boden positioniert ist und sich eine grüne Flagge an Ihrem *Gaze* befindet, können Sie fortfahren.

6) Aufbau der Spielumgebung

1) Die grüne Flagge, welche sich am *Gaze* befindet symbolisiert die *Startflagge*. Positionieren die *Startflagge* an der vorgegeben Position mithilfe der *Select-Geste*.
2) Die nun sichtbare rote Flagge symbolisiert die *Zielflagge*, welche Sie ebenfalls an der vorgegeben Position platzieren sollen.
3) Nun können Sie sich mithilfe der *Boxen* einen Weg von der *Startflagge* zur *Zielflagge* bauen.
   Select-Geste / *„Place Box“* : Platziert eine *Box* an der Position des *Gaze*.
   Hold-Release-Geste / *„Delete Box“* : Löscht eine anvisierte *Box* nach einigen Sekunden.
   Hinweis: Verwenden Sie jede dieser Interaktionen zu Beginn mindestens einmal, danach können Sie, wie es Ihnen beliebt, fortfahren.
4) Sobald Sie die Brücke ausprobieren möchten, können Sie über das Menü den *Avatar* aktivieren.

Hinweis: Sie können zwischen dem Aufbau und der Steuerung des *Avatars* beliebig wechseln.

7) **Steuerung des Avatars:**

Sobald der *Button* zum aktivieren des *Avatars* betätigt wurde, verschwindet die *Box* am *Gaze* und der *Avatar* erscheint an der Position der *Startflagge*.
Nun gibt es zwei unterschiedliche Arten den *Avatar* durch den Raum zu navigieren.
Die Aufgabe ist es den *Avatar* zur *Zielflagge* zu navigieren, wobei beide Steuerungsarten verwendet werden sollen.
Nachdem Sie das Ziel mit beiden Steuerungsarten erreicht haben ist die Testphase beendet.

Steuerung 1) Navigation mithelfe des *Gaze*.

Starten: Sprachbefehl *„Follow My Gaze“*. Somit wird die Navigation aktiviert.
Navigation: Nun verfolgt der *Avatar* den *Gaze* und kann so durch die Welt navigiert werden.
Springen: Sprachbefehl *„Jump Up“* / *Select-Geste*
Stoppen: Sprachbefehl *„Stop Following“*. Somit verharrt der Avatar an seiner jetzigen Position

Steuerung 2) Navigation mithilfe eines herkömmlichen *GamePads*.

Navigation: Durch den oberen, linken *Steuerstick* kann man den *Avatar* Steuern.
Springen: *A-Button*

Hinweis: Während der *Avatar* sich in der ersten Steuerung befindet und den *Gaze* verfolgt, kann das *GamePad* nicht für die Steuerung verwendet werden. Zuerst muss der Sprachbefehl *„Stop Following“* geäußert werden, um die erste Steuerung zu deaktivieren.

8) Nach dem erfolgreichen Abschluss der Evaluation mit der *HoloLens* muss der Proband den Rest des Fragebogens bearbeiten.
9) Ende.

Interaktion im Bereich der Augmented Reality

Seite 1

Zunächst müssen Sie ein paar grundlegende Fragen beantworten, bevor die eigentliche Evaluation mit der HoloLens losgehen kann.

Test Nr.

Wie alt sind Sie?

Geschlecht

- ◯ Männlich
- ◯ Weiblich

Wie oft spielen Sie jegliche Art von Computerspielen(Handy,Konsole,Computer, ...)?

- ◯ häufig
- ◯ selten
- ◯ nie

Sind Sie mit der Steuerung eines herkömmlichen Gamepads(Xbox, Playstation, etc.) vertraut?

- ◯ ja
- ◯ nein

Haben Sie von den Begriffen der Augmented Reality(AR) und/oder der Virtual Reality(VR) vor dieser Evaluation schon einmal gehört?

- ◯ ja
- ◯ nein

Haben Sie schon einmal einen Head-Mounted Display getragen?

- ◯ ja
- ◯ nein

Sind Sie mit einer Interaktion über eine Spracheingabe vertraut (z.B. Siri, Cortana, Alexa)?

- ◯ ja
- ◯ nein

Wie oft verwenden Sie Spracheingaben wie Siri (Apple), Cortana (Microsoft), Alexa (Amazon), oder andere?

- ◯ oft
- ◯ selten
- ◯ nie

Sind Sie mit einer Steuerung über Gesten vertraut?

- ◯ ja
- ◯ nein

Seite 2

Die folgenden Fragen beziehen sich auf Ihre Erfahrung mit den unterschieldichen Steuerungen in den verschiedenen Bereichen des Spiels, welche Sie nach der Bearbeitung des Tests gesammelt haben.

Haben Sie den Scanvorgang des Raumes selber vorgenommen?

- ◯ ja
- ◯ nein

Bitte ordnen Sie die unterschiedlichen Steuerungen nach ihrer Leichtigkeit.

1 = am leichtesten, 4 = am wenigsten leicht

- Gaze
- Sprachsteuerung
- Gestensteuerung
- Gampad

Inwiefern sehen Sie die Gestensteuerung als Anforderungsgerecht?

Anforderungsgerecht: Wie gut passt diese Art der Interaktion zu seinem Einsatzgebiet?

	sehr gut	gut	mittel	eher weniger	gar nicht
Allgemein	◯	◯	◯	◯	◯
in Bezug auf die Menüführung	◯	◯	◯	◯	◯
in Bezug auf die Positionierung der Boxen	◯	◯	◯	◯	◯
in Bezug auf die Steuerung des Avatars	◯	◯	◯	◯	◯

Inwiefern sehen Sie die Sprachsteuerung als Anforderungsgerecht?

Anforderungsgerecht: Wie gut passt diese Art der Interaktion zu seinem Einsatzgebiet?

	sehr gut	gut	mittel	eher weniger	gar nicht
Allgemein	◯	◯	◯	◯	◯
in Bezug auf die Menüführung	◯	◯	◯	◯	◯
in Bezug auf die Positionierung der Boxen	◯	◯	◯	◯	◯
in Bezug auf die Steuerung des Avatars	◯	◯	◯	◯	◯

Inwiefern sehen Sie die Steuerung mit dem Gaze als Anforderungsgerecht?

	sehr gut	gut	mittel	eher weniger	gar nicht
Allgemein	◯	◯	◯	◯	◯
in Bezug auf die Menüführung	◯	◯	◯	◯	◯
in Bezug auf die Positionierung der Boxen	◯	◯	◯	◯	◯
in Bezug auf die Steuerung des Avatars	◯	◯	◯	◯	◯

Haben Sie die das Gamepad als störend empfunden?

◯ ja

◯ nein

Welche Steuerungskonzepte haben Ihnen besonders gefallen?

☐ Sprachsteuerung (Menüführung)

☐ Sprachsteuerung (Positionierung der Boxen)

☐ Sprachsteuerung (Bewegung des Avatars)

☐ Gestensteuerung (Menüführung)

☐ Gestensteuerung (Positionierung der Boxen)

☐ Gestensteuerung (Bewegung des Avatars)

☐ Gaze

☐ GamePad

Welche Steuerungskonzepte haben Ihnen gar nicht gefallen?

- [ ] Sprachsteuerung (Menüführung)
- [ ] Sprachsteuerung (Positionierung der Boxen)
- [ ] Sprachsteuerung (Bewegung des Avatars)
- [ ] Gestensteuerung (Menüführung)
- [ ] Gestensteuerung (Positionierung der Boxen)
- [ ] Gestensteuerung (Bewegung des Avatars)
- [ ] Gaze
- [ ] GamePad

Hier können Sie Anmerkungen zu den Steuerungskonzepten hinterlassen.

Seite 3

Sie haben es fast geschaft.
Die nächsten Fragen beziehen sich auf das Spiel im allgemeinen und die Zukunftsaussichten der neuen Steuerungsarten.

Wie gut hat Ihnen das Spiel gefallen?

- ◯ sehr gut
- ◯ gut
- ◯ mittel
- ◯ nicht so gut
- ◯ gar nicht

Wie leicht fanden Sie die Umsetzung der Aufgaben?

- ◯ sehr leicht
- ◯ leicht
- ◯ mittel
- ◯ schwer
- ◯ sehr schwer

Was hat Ihnen besonders an dem Spiel gefallen oder was empfanden Sie als besonders störend?

Von welchen Steuerungsarten würden Sie in Zukunft mehr gebrauch machen wollen?

Sie können merh als eine Steuerung markieren.

- ☐ Gaze
- ☐ Sprachsteuerung
- ☐ Gestensteuerung

Hier können Sie Anmerkungen zum Spiel oder zu den Zukunftsaussichten der Steuerungsarten hinterlassen.

» **Umleitung auf Schlussseite von Umfrage Online** (ändern)

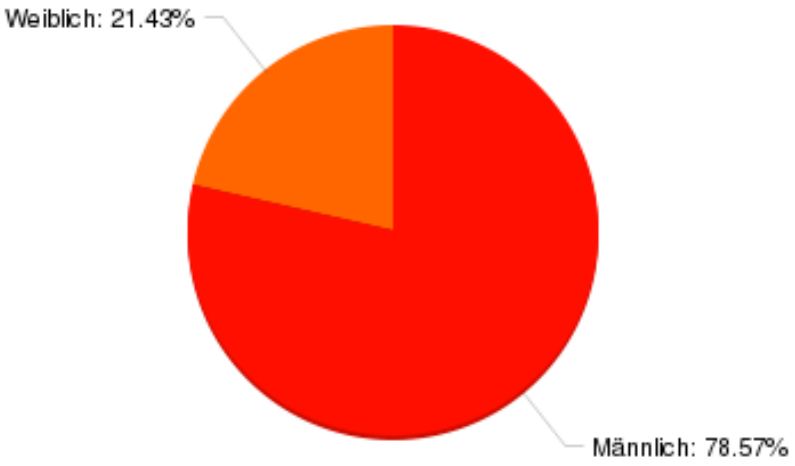

Abbildung 18: Frage 4: Geschlecht

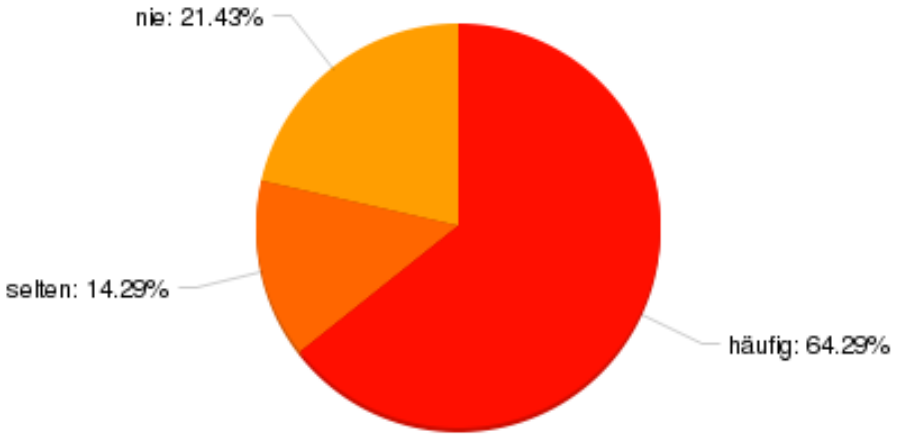

Abbildung 19: Frage 4: Wie oft spielen Sie jegliche Art von Computerspielen(Handy,Konsole,Computer, ...)?

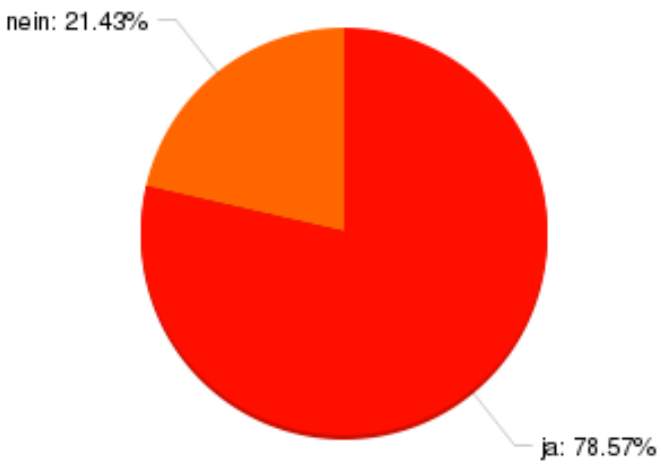

Abbildung 20: Frage 5: Sind Sie mit der Steuerung eines herkömmlichen Gamepads(Xbox, Playstation, etc.) vertraut?

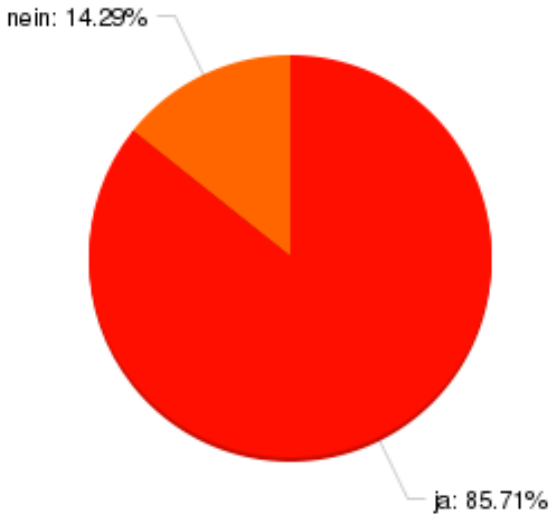

Abbildung 21: Frage 6: Haben Sie von den Begriffen der Augmented Reality(AR) und/oder der Virtual Reality(VR) vor dieser Evaluation schon einmal gehört?

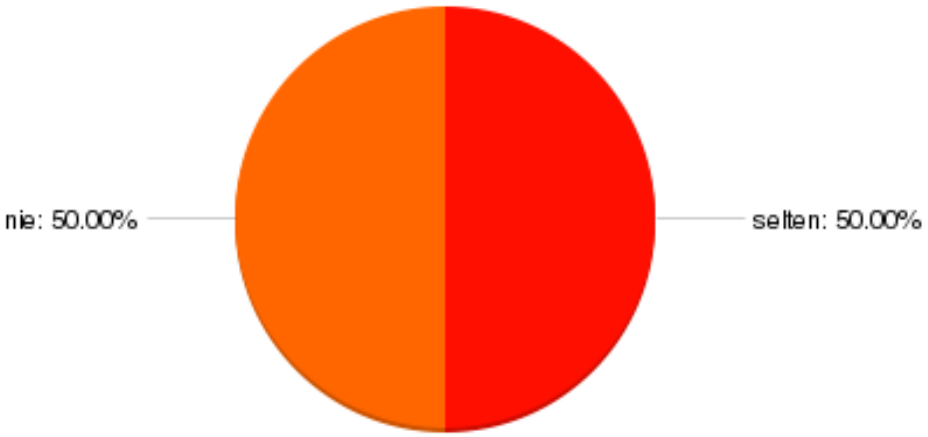

Abbildung 22: Frage 9: Wie oft verwenden Sie eine Spracheingaben wie Siri (Apple), Cortana (Microsoft), Alexa (Amazon), oder andere?

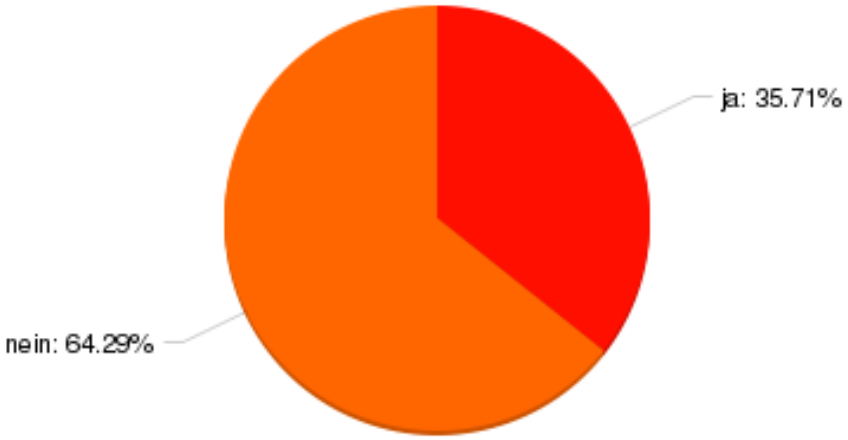

Abbildung 23: Frage 10: Sind Sie mit einer Steuerung über Gesten vertraut?

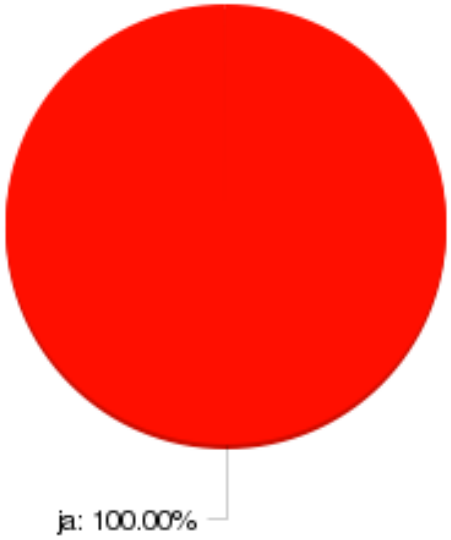

Abbildung 24: Frage 11: Haben Sie den Scanvorgang des Raumes selber vorgenommen?

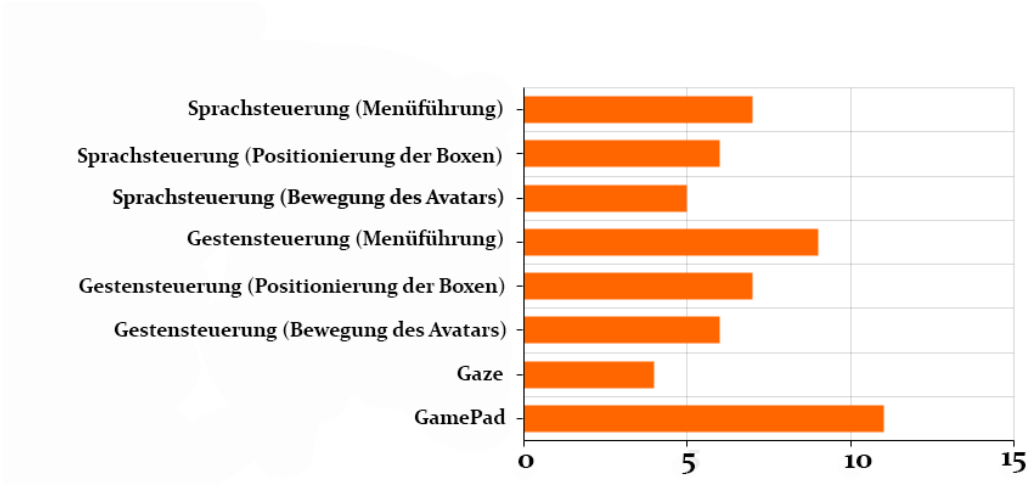

Abbildung 25: Frage 17: Welche Steuerungskonzepte haben Ihnen besonders gefallen?

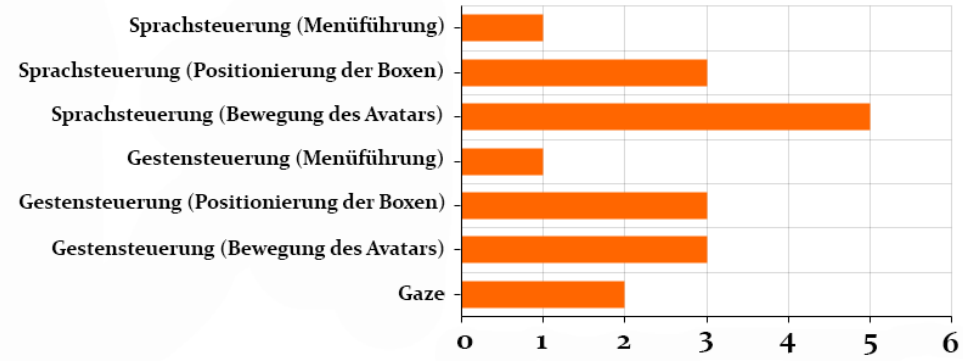

Abbildung 26: Frage 18: Welche Steuerungskonzepte haben Ihnen gar nicht gefallen?

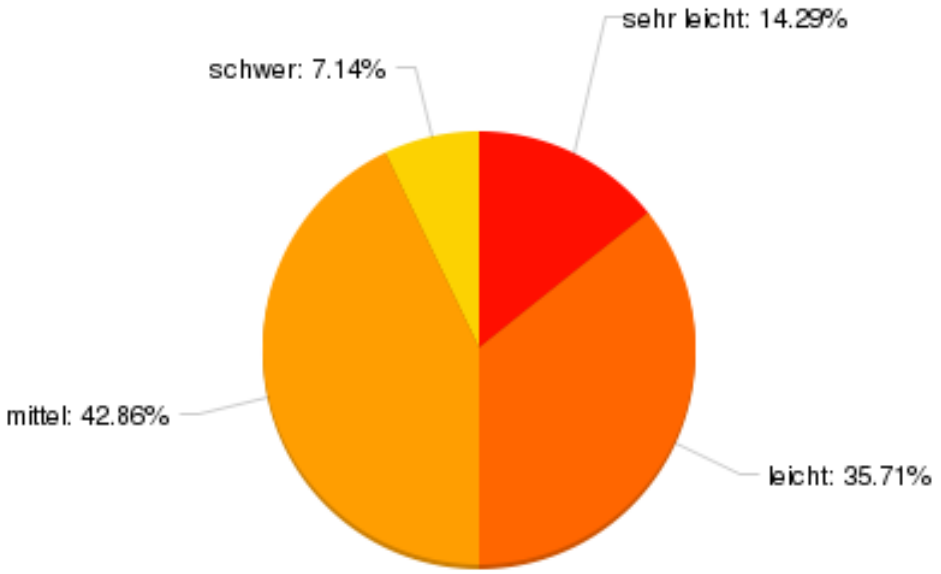

Abbildung 27: Frage 21: Wie leicht fanden Sie die Umsetzung der Aufgaben?

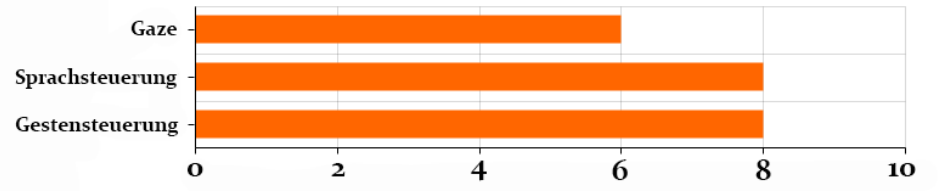

Abbildung 28: Frage 23: Von welchen Steuerungsarten würden Sie in Zukunft mehr gebrauch machen wollen?

Schriftliche Antworten

Frage 2: Wie alt sind Sie?

23,22,23,24,26,32,23,24,25,23,19,20,19,20

Frage 19: Hier können Sie Anmerkungen zu den Steuerungskonzepten hinterlassen.

- Gaze für die Steuerung des Balls hat mir besser gefallen als ich erwartet habe. Eine in diesem Kontext erstaunlich gute Alternative zum Joystick.
- Steuerung per Gaze (für mich neu) fände ich besser wenn es tiefer läge.
- Die Sprachsteuerung des Avatars fand ich passend um Befehle zu äußern, wie bspw. "Follow my Gaze". Allerdings war die Sprachsteuerung eher unpassend für die Bewegung an sich, sprich Richtung oder das Springen. Ansonsten wären mehr funktionen wünschenswert, zB für die Gestensteuerung, damit man mehr ausprobieren kann.
- Das Konzept der VR-Umgebung und die Steuerung via Gaze und Gesten ist eine sehr stimmige Kombination, welche viele Stunden Spielspass bringen kann/wird.
- Schwieirg mit der Gestensteuerung zurechtzukommen. Ungewohnte Steuerungsmethode.
- gaze springt beim nach vorne schauen nach oben
- Ich fand das Programm überaus interessant. Die Sprachsteuerung fand ich sehr gut, allerdings ist es schwierig diese mit Reaktionszeit und der Steuerung des Avatars zu verbinden. Vielleicht wäre eine Gestensteuerung mit einem Finger ganz praktisch um den Avatar schnell zum halt zu bringen.

- Sowohl mit Gestensteuerung, als auch mit Sprachsteuerung ist mir die Kontrolle über die Bausteine und den Avatar sehr leicht gefallen. Durch das Gamepad fühlt man sich als Benutzer auch gleich vertraut mit den Steuerungskonzepten, was besonders für Einsteiger, welche noch nie mit der Hololens gearbeitet haben, den Anfang erleichtern kann.

Frage 22: Was hat Ihnen besonders an dem Spiel gefallen oder was empfanden Sie als besonders störend?

- Gefühlt ist eine Kugel eine Steigung hochgerollt, was physikalisch nicht plausibel war.
- Die Neuartigkeit
- Die Boxen wurden manchmal nicht senkrecht/waagerecht plaziert. Besser wäre eine "Drehfunktion".
- störend: Gaze, Gestensteierung
- die gestensteuerung hat mir gefallen
- Ich fand gut, dass es verschiedene Bedienungsmöglichkeiten gibt, wie für Springen per Sprache oder Geste etc. Auch fand ich gut, dass die Objekte farblich so heraus stechen. So steigert sich die gute Erfahrung beim Spiel
- Die Idee, den Boden als Lava zu gestalten, hat mir sehr gut gefallen. Das freie Setzen der Bausteine lässt einem zudem sehr viel gestalterischen Spielraum, sodass ich mir auch direkt vorstellen konnte, ausgefallenere Parkours zu erstellen. Was eine schöne Erweiterungsmöglichkeit wäre, wäre das Benutzen unterschiedlicher Bausteinformen, um neue Strecken zu bauen.
- leichte Kanten zwischen den Boxen haben den Avatar beinträchtigt

Frage 24: Hier können Sie Anmerkungen zum Spiel oder zu den Zukunftsaussichten der Steuerungsarten hinterlassen.

- Alle Steuerungsarten sind praktisch in jeweils spezialisierten Gebieten (also nicht äquivalent einsetzbar usw)
- siehe Anmerkungen vorher
- Das Prinzip wird sich in Zukunft definitiv durchsetzen
- innovative Spielart, jedoch schwierige Umsetzung, wenn man diese Spielmethode nicht gewohnt ist.

- höchstens spielerei
- Besonders durch den Sprachbefehl "Follow my gaze"wurde das Spiel sehr viel lebendiger, was mir persönlich am besten gefallen hat.