Institut für Visualisierung und Interaktive Systeme

Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Bachelorarbeit

Benutzerinteraktion in Virtual Reality mittels Eye Tracking

Anja Groß

Studiengang:	Softwaretechnik
Prüfer/in:	Prof. Dr. Thomas Ertl
Betreuer/in:	Dr. Michael Krone, Michael Becher, M.Sc.
Beginn am:	4. Juni 2018
Beendet am:	18. Dezember 2018

Kurzfassung

In den vergangenen Jahren nahmen VR und AR eine immer bedeutendere Rolle sowohl im wirtschaftlichen, als auch forschungsorientierten Bereich ein. Die verfügbare Hardware wurde zunehmend erschwinglicher und leistungsfähiger. Im Hinblick auf den Bereich der Immersive Analytics und der auftretenden Ermüdungserscheinungen bei herkömmlichen VR Systemen, beschäftigt sich diese Arbeit mit Eye Tracking als Eingabemechanismus für VR Anwendungen. Es werden zuerst allgemeine Probleme bei System, welche Eye Tracking als Eingabemedium verwenden, dargestellt, sowie bereits entwickelte Formen der Interaktion mithilfe von Eye Tracking vorgestellt. Mithilfe des FOVE VR HMD, welches über integriertes Eye Tracking verfügt, wird im Rahmen dieser Arbeit entwickelte Prototyp zur Manipulation von Objekten im virtuellen Raum vorgestellt. Die Interaktionen bestehen dabei aus einer Kombination aus Eye Tracking Input und einfachen Eingaben in Form eines Knopfdruckes eines Controllers statt. Mithilfe einer Benutzerstudie werden die entwickelten Konzepte evaluiert und anschließend die Ergebnisse präsentiert und diskutiert. Auf dieser Basis werden mögliche Verbesserungen der Konzepte und Erweiterungen der Anwendung vorgestellt. Die aus dieser Arbeit gewonnenen Erkenntnisse können ebenfalls für AR Anwendungen verwendet werden.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Die Begriffe *Virtual Reality* und *Augmented Reality* sind in aller Munde. Sei es in der Automobilindustrie, der Spieleindustrie oder der Forschung: Die Einsatzbereiche für diese Technologien sind grenzenlos. Mittlerweile sind kommerzielle, günstige und leistungsstarke VR Geräte auf dem Markt und bieten selbst Privatpersonen eine Möglichkeit der Entwicklung von virtuellen Anwendungen. VR und AR nehmen im Bereich Immersive Analytics eine zunehmend wichtigere Rolle ein: Bei der Untersuchung von großen Datenmengen bieten diese beiden Technologien dem Benutzer eine intensive Immersion in die Datenanalyse. Große Datenmengen werden dadurch greifbarer und neue Erkenntnisse können leichter und schneller gewonnen werden.

Betrachtet man gängige VR Systeme stellt man fest, dass die meisten kommerziell verfügbaren aus einem HMD und zwei separaten Controllern bestehen. Diese Controller dienen der Interaktion innerhalb der virtuellen Umgebung. Meist besteht dabei die Interaktion daraus, dass der Benutzer entweder auf ein Element zeigt und dieses mit einem Knopfdruck auswählt, oder dass er mithilfe von Kopfbewegungen den zentral im Sichtfeld positionierten Cursor auf einem Element platziert und anschließend mit einem Controller die Auswahl durchführt. Dies sind zwar sehr natürliche Bewegungen, dennoch sind sie auf Dauer sehr anstrengend. In einem professionellen Kontext, in dem eine längere Nutzerdauer notwendig wird, führt dies zu Ermüdung. Auch wenn der Benutzer lediglich auf Objekte zeigen muss bedeutet dies, dass er ständig einen Arm ausstrecken muss; je nach Interaktionsart sogar für längere Zeit. Die Augen spielen dabei nur eine observierende Rolle, obwohl sie stets als erstes das Ziel der Interaktion erfassen bevor etwaige Kopf- oder Handbewegungen durchgeführt werden.

In dieser Arbeit wird Eye Tracking als eine Form der Eingabe in Kombination mit einer VR Brille untersucht, sowie daraus resultierende Designentscheidungen diskutiert. Auf dieser Basis werden Konzepte zur Benutzerinteraktion innerhalb eines implementierten Prototypen entwickelt und mithilfe einer Benutzerstudie getestet. Die Anwendung sollte die Möglichkeit bieten, Objekte untersuchen und manipulieren zu können. Diese Art der Interaktion findet in vielen verschiedenen Bereichen Anwendung: Sei es in der Molekularvisualisierung, im Produktionsdesign oder in der Medizin. Bei der Entwicklung sollten mehrere Aspekte berücksichtigt werden:

1. Die Interaktion sollte intuitiv für den Benutzer sein,
2. Die körperliche Anstrengung für den Benutzer sollte so minimal wie möglich sein,
3. Die Interaktion sollte auch bei kleinem Raum möglich sein, um die Anwendung auch für eine Büroumgebung zu ermöglichen.

Für die Anwendung wurde eine Kombination aus Eye Tracking und einem physischen, kabellosen Controller verwendet. Der Controller wird dabei nur für einfache Eingaben (Knopfdruck) verwendet, um die Möglichkeiten von Eye Tracking so weit wie möglich auszutesten. Dies ermöglicht eine flexible, schnelle und komfortable Interaktion mit dem System, ohne dabei durch aufwendige Gesten zu Ermüdung zu führen.

Die Arbeit ist in folgender Weise gegliedert: Im ersten Teil wird näher auf die verwendete Hard- und Software eingegangen (siehe Kapitel 3), Probleme bei der Verwendung von Eye Tracking als Eingabemechanismus bei System und typische Augenbewegungen besprochen (siehe Kapitel 4) und bereits entwickelte Konzepte der Benutzerinteraktion beschrieben (siehe Kapitel 5). Anschließend werden das Design des im Rahmen dieser Arbeit erstellten Benutzerinterfaces (siehe Kapitel 6), sowie dazugehörige Implementierungsdetails vorgestellt (siehe Kapitel 7). Die Benutzerstudie für das Testen der entwickelten Konzepte wird in Kapitel 8 erläutert, sowie die Ergebnisse präsentiert. Auf der Basis der Studie werden in Kapitel 9 mögliche Verbesserungen der Konzepte und Erweiterungen des Prototypen vorgestellt. Schließlich fasst Kapitel 10 die Ergebnisse der Arbeit zusammen und gibt einen Ausblick auf zukünftige Arbeiten.

2 Verwandte Arbeiten

Eye Tracking als Eingabemechanismus in computergestützten System wurde schon in den 1980er Jahren für körperlich eingeschränkte Menschen verwendet, damals allerdings vor allem für die Texteingabe [BKDA82]. Zu dieser Zeit fand Eye Tracking allerdings stets in Kombination mit normalen Computermonitoren statt, das Design der hier verwendeten Anwendungen stützte sich also auf die Zweidimensionalität des Bildschirms.

Im Jahre 1987 haben [WM87] mit Messungen nachgewiesen, dass Augenbewegungen schneller sind als andere zu der Zeit verfügbaren Eingabemethoden. Zudem stellten sie fest, dass die Kombination aus Eye Gaze Input und einem physischen Button als Selektionsmechanismus eine ebenfalls sehr gut funktionierende Alternative zu reinem Eye Gaze Input darstellt.

[Jac93] stellte verschiedene Konzepte zur Interaktion mit Objekten dar, darunter neben der Selektion auch das Scrollen von Texten und die Objektmanipulation. Dabei werden Objekte mithilfe einer Kombination aus Eye Gaze und Mausinput über einen Bildschirm bewegt.

[TJ00] haben mit Eye Gaze Input in virtuellen, dreidimensionalen Umgebungen experimentiert. In ihrem System besteht die Interaktion aus natürlichen Augenbewegungen, um bestimmte antrainierte Augenbewegungen für die Interaktion mit dem System zu vermeiden. Mithilfe von Histogrammen wird durch bloßes betrachten von Objekten das am längsten betrachtete Objekt vergrößert und somit selektiert, die Histogramm-Werte der restlichen Objekte hingegen verkleinern sich. Wechselt der Fokus des Benutzers, schlägt sich dies in einer Änderung im Histogramm-Wert des gerade betrachteten Objekts nieder. Diese Methode führte allerdings zu ”visual clutter”, das heißt es gab zu viele Objekte welche zu groß wurden durch betrachten dieser. Also wurde dies durch ein Zeige-System ersetzt, bei welchem nur in ein Objekt hineingezoomt wird, wenn der Benutzer darauf zeigt. Bewegt er hingegen den Zeiger weg vom Objekt, wird es wieder deselektiert und verkleinert sich.

3 Verwendete Hard- und Software

Dieses Kapitel stellt die bei der Implementierung der Anwendung verwendete Hard- und Software vor. Verwendet wurden das FOVE VR HMD für die Darstellung der virtuellen Umgebung und die Unity Engine für die Entwicklung.

Abbildung 3.1: Oben: Ein Bild des FOVE VR HMD, welche über integriertes Eye Tracking verfügt. Unten: technische Zeichnungen des FOVE. Zu erkennen sind die eingebauten Linsen. Quelle: https://www.getfove.com/ (zuletzt besucht am 12/16/2018).

3.1 Fove

Bisher ist das FOVE VR HMD[1] die einzige kommerziell verfügbare VR Brille, welche über integriertes Eye Tracking verfügt. Die FOVE besitzt ein WQHD OLED Display mit einer Framerate von 70Hz und einem Sichtfeld von bis zu 100°. Das Positionstracking basiert auf Infrarot, das Orientierungstracking auf einer IMU. Für das Eye Tracking werden Infrarot Lichter benutzt, welche ringförmig um die Linsen herum angeordnet sind. Der Anbieter gibt an, dass die Tracking Präzision mit einer Framerate von 120Hz bei weniger als 1° liegt. Das Setup besteht neben der VR Brille auch aus einer Tischkamera, welche dem Benutzer zugewandt ist. Diese muss, ebenfalls wie die VR Brille selbst, an den Computer angeschlossen werden. Durch ein separates Debug Tool können neben der Positionsdaten und der Orientierungsdaten der VR Brille auch der Zustand der einzelnen Augen des Benutzers (geöffnet oder geschlossen) eingesehen werden. Zudem werden die Bildaufnahmen der Eye Tracking Kameras dort dargestellt. Im Gehäuse selbst sind zusätzlich mehrere Infrarot LEDs angebracht, welche nach außen hin strahlen. Somit können neben den Augenbewegungen über die Eye Tracking Kameras Kopfbewegungen auch die Kopfbewegungen des Benutzers über die Tischkamera aufgenommen und im Debug Tool darfestellt werden. Dabei nimmt die Tischkamera jedoch nur Infrarotlicht wahr, das heißt es sind lediglich die Infrarotlichter im Gehäuse der Brille zu sehen.

Das FOVE VR HMD wird von bekannten Game Engines wie Unity und Unreal mithilfe von Plugins unterstützt. Das genaue Setup für die Implementierung wird in Abschnitt 7.1 beschrieben.

Während der Entwicklung fiel auf, dass das Eye Tracking nur im zentralen Bereich des Sichtfeldes einwandfrei funktioniert. Blickt der Benutzer zum Rand des Sichtfeldes, verschlechtert sich die Präzision des Eye Tracking. Ab einem gewissen Punkt werden die Augen dann schließlich gar nicht mehr von den Eye Tracking Kameras registriert. Möglicherweise liegt das am Sitz der Kameras, welche nicht zwischen den Linsen und dem Benutzer, sondern vermutlich direkt hinter den Linsen und am unteren Rand davon positioniert sind. Blickt der Benutzer weit genug vom Zentrum des Sichtfeldes weg, werden aufgrund der Perspektive die Pupillen schlechter sichtbar und können schlechter registriert werden. Dies musste ebenfalls beim Design des Benutzerinterfaces beachtet werden (siehe Kapitel 6).

Zudem ist es nicht möglich, während der Benutzung des FOVE VR HMD eine zusätzliche Brille mit Sehstärke zu tragen. Zum einen ist der Sitz der VR Brille aufgrund des Designs des Gehäuses und der Nähe der darin befindlichen Linsen zum Benutzer nicht dafür geeignet eine Brille zu tragen. Zum anderen funktionierte dabei das Eye Tracking nicht mehr korrekt. Bei Blick auf die Kameraufnahmen im Debug Tool kann man erkennen, dass beim Tragen einer zusätzlichen Brille auf dieser viele Reflektionen der Infrarotlichter auftreten. Abbildung 3.2 zeigt einen Benutzer mit und einen ohne zusätzliche Brille. Es ist möglich, dass aufgrund der Reflektionen und der daraus resultierenden schlechten Lichtverhältnisse das Eye Tracking gestört wird.

[1] https://www.getfove.com/ (zuletzt besucht am 16.12.2018)

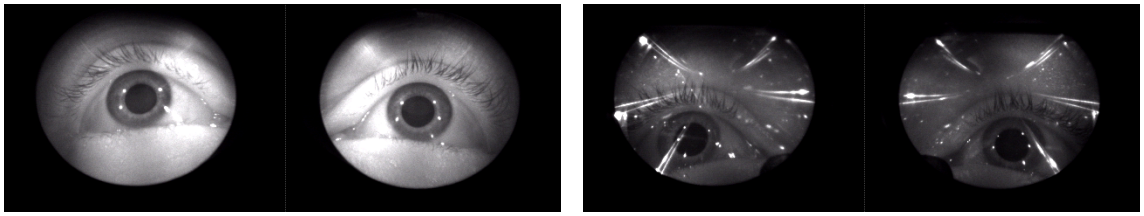

Abbildung 3.2: Screenshots der Augenaufnahmen im Debug Tool. Links: Der Benutzer trägt keine Brille mit Sehstärke. Rechts: Der Benutzer trägt eine Brille. Man kann die weißen Reflektionen auf den Linsen erkennen, sowie eine deutliche Verdunkelung.

3.2 Unity Engine

Die Unity Engine[2] ist eine Game Engine, welche ebenfalls zur Entwicklung von VR Anwendungen verwendet werden kann. Sie bietet einen grafischen Editor zur Entwicklung von zwei- und dreidimensionalen virtuellen Umgebungen. Allgemein werden solche virtuellen Umgebungen in Unity in Szenen aufgeteilt. In einer Szene können Objekte platziert werden. Auf diese Objekte können zudem Skripte (meist in der Programmiersprache C# erstellt) platziert werden. Diese Skripte können das Verhalten dieses und anderer Objekte beeinflussen.

Entwickelte Konzepte können direkt im Programm mit angeschlossener VR Brille getestet werden. Die Unity Engine bietet somit eine hilfreiche, grafische Entwicklungsumgebung für die Entwicklung von VR Anwendungen.

[2] https://unity3d.com/ (zuletzt besucht am 16.12.2018)

4 Probleme bei Systemen mit Eye Tracking als Input

Aufgrund des typischen Verhaltens der Augen des Menschen bei der Betrachtung von Objekten entstehen bei der Verwendung von Eye Tracking als Input leicht Probleme in der Anwendung.

Nach [Ohn98] machen es die folgenden Eigenschaften der Augenbewegungen unmöglich, die heute verwendeten Eingabegeräte eins zu eins durch Eye Tracking Input zu ersetzen:

1. Es ist schwierig die Augen bewusst zu bewegen. Wenn wir die Augen im Alltag bewegen, geschieht dies ohne Schwierigkeiten, doch wenn wir sie bewusst bewegen und auf einen bestimmten Punkt fixieren lassen müssen, führt dies zu Übermüdung.
2. Präzise Kontrolle der Augen ist im Vergleich zur Computermaus schwierig. Kleinste Objekte können mit einer Maus selektiert werden, aber mithilfe der Augen wird dies wesentlich schwerer.
3. Wenn ein Eye Gaze Interface verwendet wird, haben die Augen des Benutzers zwei Funktionen: Informationen über das was dargestellt ist aufzunehmen und das Interface zu bedienen. Das Eye Gaze Interface muss dazu in der Lage sein, diese beiden Funktionen voneinander zu trennen.

Deshalb ist es notwendig, ein neues Design für ein Eye Gaze kontrolliertes Interface zu entwickeln. Um dies jedoch zu bewerkstelligen ist es notwendig, Kenntnisse über die Augenbewegungen des Menschen zu haben. Im Folgenden werden die typischen Augenbewegungen, sowie die daraus resultierenden Probleme in der Mensch-Computer-Interaktion erläutert.

4.1 Typische Augenbewegungen

Nach [Jac91] verlaufen die natürlichen Augenbewegungen des Menschen nicht linear, sondern plötzlich und ruckhaft. Auf diese sogenannten *Sakkaden* folgen kurze Perioden der *Fixation* (etwa 200 bis 500 ms), in welchen der Blick des Menschen relativ stabil bleibt. In diesem Zeitintervall kann ein Objekt betrachtet werden. Während der Dauer der Fixation bewegen sich die Augen aber dennoch weiter, dabei gleichen die Bewegungen einem Zittern (aus dem engl. *jitter*).

Die allgemeinen Augenbewegungen bestehen also aus einer Aneinanderreihung von Sakkaden, jeweils gefolgt von kurzen Phasen der Fixation. Die Augen selbst sind also fast nie völlig bewegungslos. Während der Fixation denkt der Mensch für gewöhnlich, dass er ein einzelnes Objekt ruhig und fokussiert betrachtet - er ist sich selbst der kleinen, zitternden Bewegungen seines Blickes nicht

bewusst. [Jac93] diskutiert dazu die Einführung eines Mechanismus, bei dem solche unruhigen Bewegungen durch Filterung der Positionsdaten der Augen kompensiert werden, um dem Benutzer das widerzuspiegeln was er denkt zu tun, nicht das, was seine Augenmuskeln tatsächlich tun.

Daraus lässt sich ableiten, dass Benutzer einer Anwendung nicht dazu in der Lage sind, eine längere Zeit ihren Blick auf einem bestimmten Objekt ruhen zu lassen. Je kleiner dabei das betrachtete Objekt ist, desto eher bewegt sich auch während der Phase der Fixation der Blick des Benutzers vom Objekt weg. Längere Phasen des Fokussierens sollten also während des Designs des Benutzerinterfaces und der Interaktionskonzepte vermieden und Interaktionselemente nicht zu klein dargestellt werden.

Ebenfalls problematisch stellen sich Interaktionen dar, welche auf lineare und gleichmäßige Augenbewegungen des Benutzers angewiesen sind. Das Folgen eines sich linear bewegenden Elements oder beispielsweise einer vorgegebenen Linie mit dem Blick sollten entweder vermieden werden oder so gestaltet sein, dass der Blick des Benutzers nicht ununterbrochen darauf ruhen muss um die Interaktion erfolgreich durchzuführen.

Bevor der Benutzer ein physisches Eingabegerät wie eine Computermaus verwendet, betrachtet er laut [WM87] zunächst das Ziel. Erst danach findet die eigentliche Bewegung statt. Dieses Verhalten kann als Indiz für die stattfindende Bewegung und des Ziels dienen und sollte ebenfalls beim Design beachtet werden, gerade in Bezug auf das nun folgende ”Mida’s Touch Problem”.

4.2 Mida’s Touch Problem

Das *Mida’s Touch Problem* besagt, dass unbeabsichtigte Interaktionen durchgeführt werden können, wenn Eye Tracking verwendet wird. Das simple Betrachten eines interaktiven Objektes könnte vom System als Interaktion interpretiert werden, falls nicht weitere Maßnahmen getroffen werden [Jac91]. Da der Benutzer seine Augen nicht nur als Input verwendet, sondern auch zur Observierung und allgemeinen Wahrnehmung, muss dieser Aspekt bei der Entwicklung von Interaktionskonzepten mit Eye Tracking Technologie beachtet werden.

5 Benutzerinteraktion

Die typischen Interaktionen in den meisten Anwendungen sind die Selektion von interaktiven Elementen, Texteingabe und das Scrollen in abgegrenzten Bereichen, in welchen der vorgegebene Bereich zu klein ist für den Inhalt der dargestellt werden soll. In diesem Abschnitt werden verschiedene Konzepte für jede der Interaktionsarten in Systemen mit Eye Tracking vorgestellt, sowie ihre Vor- und Nachteile in der Anwendung diskutiert. Anschließend wird in Abschnitt 5.5 auf verschiedene Varianten des Benutzer-Feedbacks eingegangen.

5.1 Selektion

Die Selektion ist die am meisten getätigte Interaktion in Benutzerinterfaces, somit existiert eine Reihe an verschiedenen Eingabemethoden dafür.

Dwell *Dwell* wird oft auch in reinen VR Anwendungen verwendet, in welchen kein Eye Tracking zur Verfügung steht. Dabei wird der Cursor im Zentrum des Sichtfeldes des Benutzers mithilfe von Kopfbewegungen auf einem Interaktionselement platziert [Jac91]. Der Cursor muss ununterbrochen für eine bestimmte, vordefinierte Zeitspanne auf dem Element platziert sein um die Selektion auszulösen. Wird der Cursor während dieser Zeit vom Element wegbewegt, wird der Vorgang abgebrochen und muss bei Bedarf noch einmal durch Platzierung des Cursors auf dem Element gestartet werden. Bei Systemen mit Eye Tracking wird der visuelle Cursor allerdings nicht mit dem Kopf platziert, sondern mit dem Blick des Benutzers.

Falls also der Benutzer die Selektion des Elementes nicht durchführen will, hat er nach Betrachtung des Objektes innerhalb der Zeitspanne zur Aktivierung der Selektion die Möglichkeit, durch Wegsehen den Vorgang abzubrechen. Möchte der Benutzer jedoch das Element genauer betrachten, hat er nur sehr wenig Zeit dafür und muss eventuell mehrmals absichtlich den Blick abwenden und wieder hinsehen. Falls der Blick doch zu lange auf dem Element verweilt kann dies zu einer unabsichtlichen Selektion führen. Gerade bei der Verwendung von Eye Tracking ist der Benutzer, anders als bei einer reinen VR Anwendung ohne Eye Tracking, an die Tatsache gebunden, dass der Cursor und sein Blick miteinander Verbunden sind.

Neben dieser Variante des Dwell-Konzeptes existiert eine modifizierte Variante, das Konzept des sogenannten *Multiple Confirm* Dwell [PLW13]. Ähnlich wie bei der einfachen Variante muss der Cursor für eine bestimmte Zeitspanne ununterbrochen auf dem Interaktionselement platziert werden. Allerdings taucht nach der Zeitspanne ein weiteres Element auf, mit welchem auf dieselbe Art und Weise interagiert werden muss. Erst dann wird die Selektion tatsächlich durchgeführt. Dies dient als Bestätigung und zusätzliche Hürde, um unbeabsichtigte Selektionen zu vermeiden.

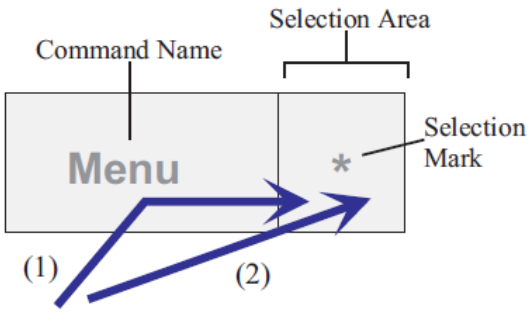

Abbildung 5.1: Ein Menüelement, welches Teil des Designs der *Quick Glance Selection Method* ist: Auf der linken Seite steht der Name des Menüelements, rechts daneben ist das Selektionsfeld. Erst bei Blick auf das Selektionsfeld wird dieses Menüelement selektiert.

Eine ähnliche Variante des Multiconfirm Dwell entwickelten bereits [WM87]: Es wurde ein zweidimensionaler, länglicher, großer Button an der linken Seite des Bildschirms platziert (diese Methode wurde für Computerbildschirme konzipiert), welcher als Bestätigung für die Selektion betrachtet werden musste. Via Dwell wurde somit durch Betrachtung dieses Buttons das Menuitem selektiert, das im Menü auf der rechten Seite als letztes betrachtet wurde).

Blinzeln und Zwinkern [Sha02] beschreibt die Notwendigkeit, normale Augenbewegungen von beabsichtigten Bewegungen zur Selektion unterscheiden zu können. Absichtliches blinzeln allein als Selektionsmethode erscheint zunächst intuitiv, doch diese Handlung findet meist unbewusst statt. Zudem kann ständiges, absichtliches Blinzeln anstrengend für den Benutzer sein: In Abhängigkeit mit der verwendeten Technologie und für die Unterscheidung zu normalem, unbewussten Blinzeln muss der Benutzer für die Selektion eventuell langsamer blinzeln und dabei die Augen deutlich öffnen und schließen. Eine Möglichkeit der Unterscheidung zu herkömmlichem Blinzeln könnte auch mehrfaches blinzeln innerhalb kurzer Zeit sein. Bei mehrmaliger Selektion hintereinander wäre dies für den Benutzer auf Dauer aber zu anstrengend. Beim Blinzeln werden beide Augen geschlossen und wieder geöffnet. Dies führt dazu, dass der Benutzer den Fokus des zu selektierenden Objekts verliert, er muss dann sozusagen wieder danach suchen.

Eine Alternative Methode stellt [Sha02] allerdings dar: Ähnlich zum Blinzeln ist das zwinkern mit einem Auge. Diese Augenbewegung ist leicht durchzuführen, wird nicht ständig unbewusst durchgeführt wie das Blinzeln und benötigt nur sehr wenig Zeit. Zwinkern ist stets eine bewusste Handlung. Anders als beim Blinzeln bleibt dabei ein Auge offen, somit verliert der Benutzer nicht den Fokus des selektierten Objekts. Das bedeutet, dass das zu selektierende Objekt betrachtet und mithilfe eines einmaligen Zwinkerns selektiert wird.

Quick Glance Selection Method [Ohn98] entwickelte die *Quick Glance Selection Method*, allerdings im Hinblick auf das Design von Menüs und der Darstellung von Hierarchien. Das Ziel besteht aus zwei Teilen: einem Kommando-Namen (wie beispielsweise Öpen") und einem Selektionsfeld (siehe Abbildung 5.1). Möchte der Benutzer mit dem Menüelement interagieren, muss er lediglich das dazugehörige Selektionsfeld betrachten. Anders als bei der Dwell-Methode muss

dieses Feld nicht über eine bestimmte Zeitspanne hinweg betrachtet werden, die Selektion wird direkt beim Blick darauf durchgeführt. Es gibt zwei Möglichkeiten das Menüitem zu selektieren: Die eine besteht daraus, zuerst auf das Kommando-Feld zu schauen, und anschließend auf das dazugehörige Selektionsfeld. Dieser Ansatz benötigt zwei Sakkaden für die Selektion. Diese Variante wird vor allem dann benutzt, wenn der Benutzer die Anwendung und die genaue Position des Menüelements noch nicht kennt. Eine Alternative dazu ist das direkte Betrachten des Selektionsfeldes. Dies ist eher für erfahrene Benutzer gedacht, welche die Position der Menüelemente bereits kennen. Dies benötigt lediglich eine Sakkade und ist somit wesentlich schneller.

Blick und physisches Eingabegerät Bereits [WM87] verwendeten eine Kombination aus Blick und Druck eines physischen Buttons zur Selektion von Interaktionselementen: Dabei muss der Benutzer auf das Interaktionselement schauen und gleichzeitig eine Taste auf der Tastatur, eine Maustaste oder einem anderen physischen Gerät (beispielsweise einem Controller) drücken. Diese Variante der Selektion ist nach der Studie von [WM87] schneller als die Dwell-Variante. Sie lässt sich zudem bei jeglicher Interaktion verwenden, sei es bei der Auswahl von Elementen innerhalb eines Menüs oder von anderen Interaktionselementen.

5.2 Scrollen

Die Scrolling-Funktion von [Jac93] ist für Fenster gedacht, deren textueller Inhalt nicht hineinpasst. Oberhalb der obersten und unterhalb der untersten Textzeile befinden sich Pfeile. Betrachtet der Benutzer diese Pfeile beginnt das Scrollen des Textes, je nachdem welche Pfeile betrachtet wurden. Die Intention dabei ist, dass die Aufmerksamkeit des Nutzers auf die Bewegung gezogen wird sobald er die Pfeile betrachtet, was das Scrollen beendet. Liest er weiter und ist an der letzten Zeile angekommen, muss er lediglich leicht nach unten auf die Pfeile blicken, um den Rest des Textes anzeigen zu lassen. Dabei werden die Pfeile nur dargestellt, wenn auch tatsächlich gescrollt und weiterer Inhalt angezeigt werden kann.

5.3 Texteingabe

Die meist verwendete Art der Texteingabe über Eye Tracking besteht aus der Betrachtung des jeweiligen Buchstabens [I S07]. Typischerweise wird hierfür eine virtuelle Tastatur dargestellt, ähnlich einer physischen. Die jeweilige Taste wird mithilfe einer Bewegung des Benutzers (Auslösung einer physischen Taste, Blinzeln, Augenzwinkern oder einer sonstigen Muskelaktivität des Gesichts) selektiert. Mit der schon bei der Selektion vorgestellten Dwell-Methode ist dies ebenfalls möglich und verhindert unbeabsichtigtes Auslösen der virtuellen Taste, kann jedoch durch auftretende Sakkaden unterbrochen werden [SR95]. Zudem dürfen die virtuellen Tasten nicht zu klein sein, sonst wird die Selektion erheblich erschwert und kann inakkurat werden. Gerade bei einer Tastatur, welche neben den 26 Buchstaben auch noch Sonderzeichen und ein Leerzeichen enthalten sollte, wird dies zunehmend schwerer.

Eine andere Variante stellen [WM02] mit ihrem System namens *Dasher* vor. Sie nutzen dabei die natürlichen Such- und Navigationsfähigkeiten der Augen aus. Die Buchstaben des Alphabets befinden sich in alphabetischer Reihenfolge an der rechten Seite des Bildschirms. Betrachtet der

Benutzer einen der Buchstaben, wird dieser vergrößert und beginnt sich nach links zu bewegen. Die horizontale Position des verwendeten visuellen Cursors (das heißt die Position des Fokuspunktes des Benutzers) wirkt sich dabei auf den Grad der Vergrößerung aus: Blickt der Benutzer nach links, verkleinert sich der Buchstabe wieder. So können eventuelle Fehler korrigiert werden. Hat ein Buchstabe die senkrecht im Bildschirm verlaufende Linie überschritten, wird er selektiert und geschrieben. Die Buchstaben, welche mit der größten Wahrscheinlichkeit als nächstes benötigt werden, werden ebenfalls vergrößert angezeigt und bewegen sich etwas nach links. Dies führt zu einer Reduzierung der Distanz und einer schnelleren Suche des nächsten Buchstabens, und somit zu einer Zeitersparnis für deren Selektion.

5.4 Objekt-Manipulation

[Jac93] entwickelte zwei Methoden für die Translation eines Objekts. Beide verwendeten bei der Selektion eine Kombination aus Maus als physisches Eingabegerät und den Blick des Benutzers. [Jac93] stellte fest, dass bei einer Manipulation mit einer Maus typischerweise zwei Operationen durchgeführt werden: Die Selektion des zu manipulierenden Objekts und die Durchführung der eigentlichen Manipulation.

Bei der ersten Methode besteht die Selektion daraus, dass der Benutzer das Objekt betrachtet und dabei eine Taste drückt. Für die Translation muss der Benutzer eine Maustaste gedrückt halten und dabei die Maus in Richtung bewegen, in die das Objekt bewegt werden soll. Es existiert dabei kein visueller Mauscursor, die Bewegung des Objekts wird relativ zur Änderung der initialen Mausposition durchgeführt. Bei loslassen der Maustaste stoppt die Translation.

Für die zweite Methode muss der Benutzer wie zuvor das Objekt per Blick darauf und gleichzeitigem Tastendruck selektieren. Anschließend drückt er eine andere Taste; während diese Taste gedrückt gehalten wird bewegt sich das Objekt mit dem Blick des Benutzers, allerdings mit einer Verspätung von 100 ms nach der Fixation. Das Objekt bleibt dabei ständig an dieser Position, trotz zitternder Augenbewegungen. Erst bei der nächsten Fixation ändert das Objekt die Position. Wird die Taste losgelassen, wird die Translation beendet und das Objekt verbleibt an seiner neuen Position.

5.5 Benutzer-Feedback

Feedback während der Interaktion mit Objekten ist für den Benutzer unerlässlich, denn nur so kann er stets beurteilen, ob die Interaktion funktioniert hat oder nicht. Je nach Funktion ist nach einer erfolgreichen Selektion keine (offensichtliche) Veränderung für den Benutzer sichtbar, umso wichtiger ist dann die Bestätigung für seine Handlung. Im Folgenden werden verschiedene Varianten vorgestellt, sowie ihre Einsatzmöglichkeiten erläutert.

Highlighting Unter dem Begriff *Highlighting* ist all jenes Feedback gemeint, welches die Farbe des Elements verändert mit welchem auf irgendeine Art und Weise interagiert wurde. Wenn der Blick des Benutzers beispielsweise auf einem Element ruht, kann die Farbe des gerade betrachteten Elements geändert werden solange der Blick darauf ruht. Somit weiß der Benutzer, dass er gerade darauf blickt und damit interagieren kann.

Diese Art des Feedbacks kann für die meisten Interaktionselemente verwendet werden, sei es für Buttons jeglicher Art oder auch dreidimensionale Objekte in VR.

Pop Out Unter *Pop Out* ist das Hervortreten von Elementen, wie beispielsweise Buttons in einem Menü, zu verstehen. Dabei bewegt sich das Objekt in virtuellen Umgebungen wortwörtlich ein kleines Stück auf den Benutzer zu, im Zweidimensionalen könnte man dies durch einfaches vergrößern des jeweiligen Objektes realisieren. Dieses Konzept findet in einem Beispielprojekt von Unity3d [Gol18] Verwendung. Für diese Art Feedback ist es jedoch von Vorteil, wenn andere Elemente von ähnlicher Größe daneben platziert sind, denn so kommt der Effekt deutlicher zum Vorschein. Gerade in kachelförmig angeordneten Menüs tritt so das gerade betrachtete Element noch deutlicher hervor.

Radialanzeige bei Dwell In gewöhnlichen VR Anwendungen wird zur Unterstützung von Dwell Interaktionen oftmals eine radiale Anzeige verwendet, um das Ablaufen der Zeit bis zur Selektion dem Benutzer visuell darzustellen [Gol18]. Dabei füllt sich ein Ring mit voranschreitender Zeit mit Farbe. Ist der Ring komplett mit Farbe gefüllt bedeutet dies, dass die Dwell-Zeit abgelaufen ist, die Radialanzeige schließt sich und die Selektion wird durchgeführt. Ohne diese Form des Feedbacks ist es sonst für den Benutzer zum einen schwer festzustellen um welche Form der Interaktion es sich bei der Anwendung handelt (wenn er nicht vorher schon instruiert wurde), und zum anderen wie lange er das Element betrachten muss bis zur erfolgreichen Selektion.

6 Benutzerinterface-Design für Eye Tracking

In diesem Abschnitt wird auf das Design des im Rahmen dieser Arbeit implementierten Benutzerinterfaces eingegangen. In Abschnitt 6.1 wird auf das allgemeine Benutzerinterface eingegangen, das heißt auf die Wahl des Cursors und das Menü-Design. In Abschnitt 6.3 wird anschließend detailiert auf das Design der Objekt-Manipulation eingegangen und die verschiedenen Konzepte vorgestellt.

Für die Interaktion habe ich mich für die Kombination aus Tastatur und Eye Tracking als Input entschieden (siehe Abschnitt 5.1). Die allgemeine Interaktion besteht darin, dass der Benutzer ein Element des Interfaces betrachtet und durch drücken einer vorher definierten Taste die Funktion auslöst. Außer der Interaktions-Taste wird mithilfe einer weiteren, separaten Taste das Hauptmenü aufgerufen. Dies geschieht unabhängig von der Eye Tracking Technologie und kann daher jederzeit ausgeführt werden. Bei den Tasten habe ich mich bewusst für die Leertaste als Interaktions-Taste entschieden, da diese aufgrund ihrer Form und Position auf der Tastatur allein durch Erfühlen sehr leicht vom Benutzer gefunden werden kann. Während die VR Brille auf dem Kopf des Benutzers sitzt, kann er seine Umgebung nicht sehen; daher ist es wichtig, dass bei Gebrauch einer Tastatur, welche sich fest auf dem Tisch vor dem Benutzer befindet, Tasten zu wählen die leicht zu finden sind. Für den Aufruf des Hauptmenüs habe ich die Enter-Taste des Nummernblocks gewählt, da sich diese bei der Implementierung verwendeten Tastatur unten rechts in der Ecke befindet und aufgrund ihrer Position ebenfalls leicht zu ertasten ist.

Die Idee war es ursprünglich, anstatt einer Computer-Tastatur, welche doch aufgrund ihres Formates und ihrer Funktion sich stets auf einem Tisch liegend befindet, für die Tasteneingabe einen kleinen Controller zu verwenden. Damit wäre der Benutzer nicht mehr an einen Tisch gebunden, sondern könnte sich flexibler und mobiler bewegen. Einfache Selektionen sind mit Controllern möglich, dennoch reichte es, für den Test des Prototypen eine Tastatur zu verwenden, da diese simpel und robust genug dafür ist.

6.1 Grundlegendes Benutzerinterface-Design

Das grundlegende Benutzerinterface besteht aus einem visuellen Cursor, einem Hauptmenü und einem Objektmenü.
Die Position des Cursors ist abhängig von der aktuellen Position des Fokuspunktes des Benutzers. Das bedeutet, dass der Cursor sich immer dort befindet wo der Benutzer in der virtuellen Umgebung hinsieht.

Das Hauptmenü bietet die Möglichkeit, ein 3D Objekt aus einer Liste von vordefinierten Objekten auszuwählen und zu öffnen, und Einstellungen für die Benutzung der Anwendung zu tätigen, wie beispielsweise die Wahl der Cursor-Form. Wurde ein Objekt geöffnet und geladen, tritt das Objektmenü in Erscheinung: Dieses dient dazu, Funktionen und Einstellungen zu tätigen, welche

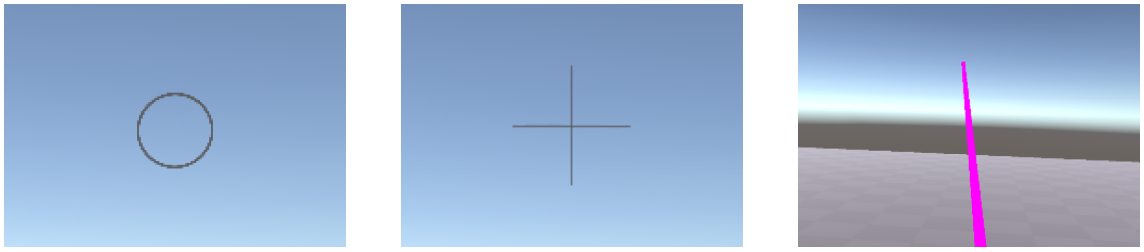

Abbildung 6.1: Cursor Varianten (von links nach rechts): Ring-Cursor, Fadenkreuz, und Laser.

sich allein auf dieses Objekt beziehen. Das bedeutet, dass damit einzelne Objekte angesprochen werden können wenn mehrere Objekte aufgerufen worden sind. Für die Einschätzung des Raumes und der Entfernung des Benutzers zu den Objekten sowie deren Größe wurde ein gekachelter Boden hinzugefügt.

Im Folgenden werden die verschiedenen entwickelten Cursor-Varianten vorgestellt, sowie die beiden Menüs.

6.1.1 Cursor Design

In gewöhnlichen VR Umgebungen ist es üblich, einen Cursor im Zentrum des Blickfeldes des Benutzers zu positionieren. Dies dient zur Visualisierung des Interaktionspunkts des Benutzers mit den Elementen in der virtuellen Umgebung, ähnlich wie bei der Computermaus. Oft werden hierfür einfache Formen wie ein oder mehrere ineinander befindliche Kreise, Punkte oder Fadenkreuze verwendet. Sinnvoll ist es im Allgemeinen, keine Form zu verwenden welche zu auffällig ist und den Benutzer von der eigentlichen Umgebung und Interaktion ablenkt oder zu groß ist und damit die Sicht versperrt.

Ohne die Verwendung von Eye Tracking müssen die Benutzer ihre Kopfbewegungen nutzen um den Cursor zu bewegen und für Interaktionen zu platzieren. Ohne einen visuellen Cursor ist es für den Benutzer sehr schwer zu wissen, wo genau sich der Interaktionspunkt befindet. Die Benutzer sind sich jedoch stets bewusst welches Objekt sie gerade betrachten. Dementsprechend kommt die Frage auf, ob unter der Verwendung von Eye Tracking die Benutzer ein visuelles Feedback in Form eines Cursors zur Interaktion benötigen.

Ring-Cursor Der Ring-Cursor ist ein zweidimensionaler, dunkelgrauer und leicht transparenter Ring. Die Farbe und Intensität wurden so gewählt, dass der Ring nicht vom Hintergrund ablenkt und möglichst wenig Fläche bedeckt. Der Interaktionspunkt (und damit auch der Fokuspunkt des Benutzers) befindet sich im Zentrum des Rings. Dies wurde so gewählt, damit versehentliche Interaktionen mit Elementen verringert werden wenn beispielsweise ein Teil des Rings auf einem Element platziert wird das sich nah an dem Interaktionselement befindet mit dem interagiert werden soll.
Dieses Design hat den Vorteil, dass man durch den Ring hindurch sehen kann und er dementsprechend nicht den Fokuspunkt des Benutzers verdeckt. Allerdings kann die Interaktion mit kleinen oder weit entfernten Objekten erschwert werden, da die Position des Interaktionspunkts im Verhältnis zum Element nur schwer eingeschätzt werden kann. Einen ähnlichen Effekt hat kann die

Wahl der Größe des Rings sein: Je größer der Ring im Verhältnis zum Interaktionselement, desto schwieriger ist es für den Benutzer zu entscheiden wo sich der Interaktionspunkt befindet. Bei einer sehr klein gewählten Ringgröße verringern sich die oben genannten Vorteile des Ring-Cursors: Der Ring verdeckt mehr von den Elementen dahinter und es lässt sich schwerer durch ihn hindurch sehen.

Fadenkreuz Das Fadenkreuz besteht aus zwei dunkelgrauen, dünnen Linien die sich in ihrer Mitte kreuzen (siehe Abbildung 6.1). Die Farbe und Farbintensität wurden ähnlich wie bei Ring-Cursor so gewählt, dass der Cursor nicht vom Hintergrund ablenkt und möglichst wenig Fläche bedeckt. Häufig in Computerspielen verwendet wird diese Variante des Cursors vielen Benutzern bekannt sein und die Bedienung der Anwendung somit potentiell erleichtern. Der große Vorteil an diesem Design ist dass der Interaktionspunkt, anders als beim Ring-Cursor, durch die Kreuzung der beiden Linien in der Mitte exakt visualisiert wird. Dem Benutzer ist dadurch stets bewusst, wo sich dieser genau befindet. Durch die schmalen Linien werden zudem die Objekte dahinter nur minimal verdeckt.

Laserstrahl Der Laserstrahl ist eine pinke Linie, welche leicht unterhalb der Position des Benutzers beginnt und im Fokuspunkt des Benutzers und somit dem Interaktionspunkt mündet. Ich habe mich für diese Farbe entschieden, da dies der Unity Standard bei der Erstellung von Linien ist und sehr auffällig ist. In der Anwendung wird diese Farbe ansonsten nirgends verwendet. Somit ist der Laserstrahl stets gut sichtbar und hebt sich vom Hintergrund ab.
Der Laserstrahl findet häufig bei VR Controllern Verwendung: Um mit seiner virtuellen Umgebung interagieren zu können muss der Benutzer mit dem Controller auf ein Element zeigen. Der Laserstrahl geht hierbei vom Controller aus und zeigt in die Richtung in die der Arm des Benutzers gerichtet ist. Das heißt der Strahl fungiert dabei wie eine Verlängerung des Arms. Das Ende des Strahls ist dabei der Interaktionspunkt.
Dieses Verhalten habe ich für diese Anwendung adaptiert und auf das Eye Tracking angepasst. Deshalb entspringt der Ausgang des Strahls auch in der Position des Benutzers, da der Strahl sich nach der Blickrichtung des Benutzers richtet. Der Laserstrahl beginnt nicht direkt an der Position der Augen des Benutzers, da dies dazu führen würde dass der Benutzer lediglich einen Punkt sehen könnte und nicht den Strahl an sich. Stattdessen wurde eine niedrigere Höhe gewählt.

Beim Ring-Cursor und dem Fadenkreuz ist es möglich sie in einem festen Abstand zum Benutzer oder am Fokuspunkt des Benutzers zu positionieren. Letzteres bedeutet eine dynamische Anpassung der Größe des Cursors, das heißt je näher der Punkt ist der betrachtet wird, desto größer ist auch der Cursor selbst. Diese dynamische Anpassung gibt dem Benutzer ein Gefühl für den dreidimensionalen Raum in dem er sich befindet. Allerdings führt dieses Verhalten auch dazu, dass der Cursor sehr klein wird wenn der Fokuspunkt weit entfernt ist. Der Cursor wird schwerer zu erkennen und Interaktionen mit Objekten werden somit deutlich erschwert.
Der Laserstrahl hingegen ändert dynamisch seine Länge wenn dieser sich am Fokuspunkt des Benutzers orientiert, das heißt je weiter entfernt sich der Fokuspunkt des Benutzers befindet, desto länger wird der Strahl. Je größer die Länge, desto schwieriger wird es hier jedoch das Ende zu sehen und Interaktionen mit Objekten werden wie bei den anderen beiden Cursor Varianten erschwert.

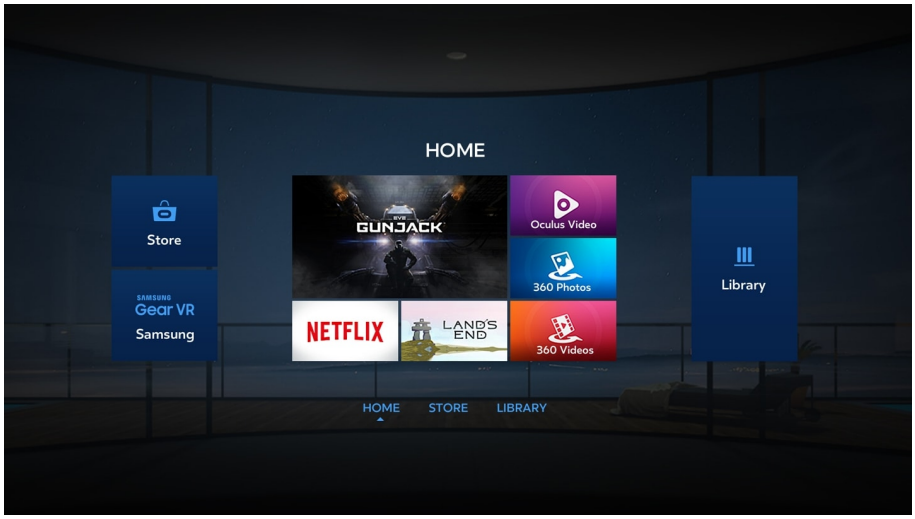

Abbildung 6.2: Der Homescreen des GearVR von Samsung. Die Menüelemente sind kachelförmig auf einer Ebene angeordnet. Quelle: https://www.samsung.com/no/wearables/gear-vr-r322/ (zuletzt besucht am 16.12.2018)

Smoothing Wie oben bereits erwähnt folgt der Cursor dem Blick des Benutzers. Die Augenbewegungen sind jedoch nicht linear, sondern ruckartig (siehe Abschnitt 4.1). Den Menschen ist dies nicht bewusst, das bedeutet dass dem Benutzer plötzliche, ruckartige Bewegungen des Cursors auffallen würde und diese ihn ablenken könnten. Deshalb habe ich einen Modus entwickelt in welchem der visuelle Cursor dem Blick des Benutzers fließend folgt anstatt sofort zum Fokuspunkt zu springen. Hierfür werden die vorangegangenen Blickpunkte des Benutzers gespeichert und der Durchschnitt über diese Menge gebildet. Je mehr Blickpunkte dabei gespeichert werden, desto mehr zieht der Cursor dem Blick des Benutzers nach. Zu starkes Nachziehen des Cursors kann jedoch dazu führen, dass der Cursor sich nicht schnell genug für den Benutzer bewegt. Die Schnelligkeit dieser Cursorbewegung hängt sehr vom jeweiligen Benutzer ab und wird unterschiedlich empfunden. In Abschnitt 8.5 wird dieses Thema noch einmal in Zusammenhang mit der Benutzerstudie aufgegriffen.

6.1.2 Menüs

Beim Design der Menüs habe ich mich an häufig verwendeten Konzepten für VR GUIs orientiert. Die Menüs bestehen aus zweidimensionalen Buttons, die kachelförmig auf einer 2D Ebene angeordnet sind (siehe Abbildung 6.3). Diese Ebene schwebt in einer festen Entfernung vor dem Benutzer in der Luft und ist ihm stets zugewandt, das heißt bei Drehungen des Benutzers ist das Hauptmenü stets in Reichweite. Eine ähnliche Anordnung der Elemente bietet das Home-Menü der Samsung Gear VR, wobei hier das Menü fest im Raum positioniert wurde.

Dadurch dass die Buttons stets dem Benutzer zugewandt sind fällt deren zweidimensionale Form nicht auf, und der Benutzer gerät nicht in die Situation sich bei Bewegung durch den Raum gegebenenfalls hinter den Buttons zu befinden. Die Buttons wurden relativ groß gehalten, um deren Selektion so einfach wie möglich zu gestalten. Wird mit einem Button im Hauptmenü interagiert,

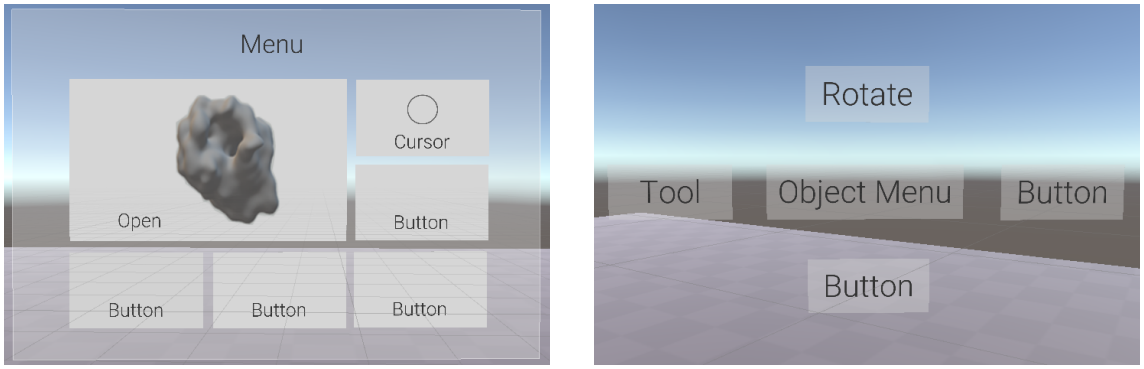

Abbildung 6.3: Links: Das Hauptmenü der Anwendung, welches Menüelemente für das Laden eines Objektes und Cursor Einstellungen bietet. Rechts: Das Objektmenü mit dessen radial angeordneten Elementen. Um das Design besser darzustellen, wurden Buttons als Platzhalter hinzugefügt.

verschwinden alle Buttons des Hauptmenüs und die Elemente des jeweiligen Untermenüs werden geöffnet. In der oberen linken Ecke des Untermenüs gibt es einen *Back*-Button, über den man zurück zum Hauptmenü gelangt.

Neben dem Design des Hauptmenüs ist es wichtig, sich über den Aufruf des selbigen Gedanken zu machen. Die initiale Idee war es, einen Button im Sichtfeld des Benutzers oder statisch im Raum zu platzieren, über den das Hauptmenü geöffnet werden kann. Platziert man einen Button statisch im Raum muss der Benutzer stets zu dieser Position zurückgehen um das Menü zu öffnen, was bei einer Anwendung in der sich der Benutzer bewegen kann unpraktisch ist. Zudem ist es möglich, dass der Benutzer den Button nicht mehr im virtuellen Raum finden kann.
Platziert man nun den Button so, dass er fest im Sichtfeld des Benutzers ist (das heißt ähnlich wie das Hauptmenü selbst immer in einem festen Abstand zum Benutzer positioniert wird und seinen Kopfbewegungen folgt), ergibt sich ein anderes Problem. Damit der Button nicht die Sicht des Benutzers auf andere Elemente beinträchtigt, sollte man ihn so weit wie möglich am Rand des Sichtfeldes platzieren. Doch das Eye Tracking der FOVE funktioniert nur im Zentrum des Sichtfeldes zuverlässig: An den Rändern des Sichtfeldes funktioniert die Technologie ab einem gewissen Punkt nicht mehr. Somit können dort positionierte Elemente mithilfe von Eye Tracking nicht zuverlässig selektiert werden.

Aus diesen Gründen habe ich mich für einen anderen Ansatz entschieden: Statt der Interaktion mit einem visuellen Button in der virtuellen Umgebung muss der Benutzer zum Öffnen des Hauptmenüs eine gesonderte Taste auf der Tastatur drücken, welche allein die Funktion des Öffnen und Schließen des Hauptmenüs innehat. Auf diese Weise kann der Benutzer stets problemlos das Hauptmenü öffnen, egal wo er sich gerade im virtuellen Raum befindet. Diese Variante ist weniger intuitiv und immersiv als die der Platzierung eines visuellen Buttons, da dem Benutzer zuvor mitgeteilt werden muss dass diese Taste gedrückt werden muss. Beobachtungen dazu wurden während der Benutzerstudie gemacht und werden in Abschnitt 8.5 diskutiert.

Jedes Objekt besitzt ein eigenes Objektmenü. Der Button zum Öffnen des Objektmenüs ist stets nahe des Objektes positioniert. Selektiert der Benutzer diesen Button, öffnet sich das Objektmenü und alle Menüelemente (ebenfalls Buttons) erscheinen. Die Menüelemente sind radial um den Objektmenü-Button angeordnet (siehe Abbildung 6.3 rechts). Wird ein Button des Objektmenüs

selektiert welcher zu einem Untermenü führt, verschwinden die anderen Buttons des Objektmenüs und die Elemente des Untermenüs erscheinen, ebenfalls radial um den zentralen Button angeordnet, welcher nun der Button des eben selektierten Buttons ist. Um zu einem höheren Level der Menü-Hierarchie zurückzukehren, muss der Button im Zentrum des Radialmenüs selektiert werden. Dieses Konzept lässt sich auf mehrere Hierarchiestufen erweitern, wobei die Hierarchietiefe zur einfachen Orientierung nicht tiefer als drei sein sollte.

6.2 Benutzer-Feedback

In der Anwendung wird als Benutzer-Feedback *Highlighting* verwendet (siehe Abschnitt 5.5). Die Buttons beim Haupt- und Objektmenü werden geld eingefärbt wenn der Blick des Benutzers darauf ruht. Dasselbe geschieht bei den Elementen des Transformations-Tools wenn diese selektiert werden - somit ist dem Benutzer stets bewusst, mit welchem Element er gerade interagiert. Endet die Interaktion, erhält das Element wieder seine ursprüngliche Färbung zurück.

6.3 Objekt-Manipulation

In der im Rahmen dieser Arbeit entwickelten Anwendung sollte neben allgemeinen Konzepten zur Benutzerinteraktion auch Konzepte für die Manipulation von Objekten entwickelt werden, welche gerade im Bereich Immersive Analytics von großer Bedeutung ist. Für die Untersuchung von Objekten soll es möglich sein, diese drehen, vergrößern beziehungsweise verkleinern und im virtuellen Raum bewegen zu können. In diesem Abschnitt werden Konzepte zur Rotation, Skalierung und Translation von Objekten vorgestellt.

Die Implementierung von komplexeren Interaktionen als der Selektierung eines Elements stellt eine Herausforderung dar, da der Benutzer über seine Augen die Manipulation steuern soll und im selben Moment Veränderungen am Objekt wahrnehmen möchte.

6.3.1 Freie Rotation

Die sogenannte *Freie Rotation* erlaubt es dem Benutzer das Objekt mithilfe seines Blicks zu rotieren. Betrachtet der Benutzer einen Punkt auf dem Objekt und hält die Leertaste (die allgemeine Interaktionstaste) gedrückt, dreht sich der betrachtete Punkt zum Benutzer hin. Verschiebt der Benutzer seinen Fokuspunkt auf dem Objekt, ändert sich dynamisch die Rotationsrichtung. Lässt er hingegen die Taste wieder los, stoppt die Rotation.

Die Freie Rotation wird über das Objektmenü, genauer über die Selektion des *Rotate*-Buttons (siehe Abbildung 6.3 rechts), aktiviert. Nach der Aktivierung erscheint ein rötlicher Kreis in der Mitte des Objektes (siehe Abbildung 6.4). Dieser Kreis ist die sogenannte ”Dead Zone”, das heißt während sich der Fokuspunkt des Benutzers innerhalb des Kreises befindet und die Leertaste gedrückt wird, ist keine Rotation möglich. Die Dead Zone wurde hinzugefügt, da während der Implementierung und der ersten Tests der Funktionalität auffiel, dass bei Blick auf den zentralen Bereich des Objektes schnelle Richtungswechsel der Rotation geschehen, selbst wenn der Benutzer dies nicht als Ziel hatte. Dies liegt daran, dass das Objektzentrum bei der Implementierung als Bezugspunkt für

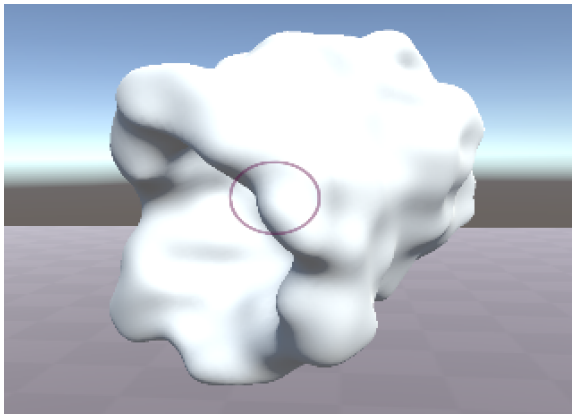

Abbildung 6.4: Ein Screenshot der die ”Dead Zone”der Freien Rotation zeigt (der rötliche Kreis). Während der Benutzer die Dead Zone betrachtet findet keine Rotation statt. Die Freie Rotation wird über das Objektmenü aktiviert und deaktiviert, ähnlich wie das Tool.

die Rotationsrichtung herangezogen wurde. Mithilfe des Objektzentrums und der Position des Fokuspunktes des Benutzers wird über das Kreuzprodukt der beiden Werte die Rotationsachse berechnet, um die das Objekt rotiert.
Der Benutzer kann nur schwer abschätzen wo sich das Objektzentrum genau befindet, weshalb bei Blick auf das Zentrum des Objektes Probleme bei der Bedienung auftreten. Durch die Dead Zone wird diese Problematik umgangen, da das Zentrum bei der Funktionalität ausgenommen wird. Zudem ermöglicht dieser rotationsfreie Bereich dem Benutzer die Rotation selbst mit seinem Blick zu pausieren, ohne dass die Leertaste losgelassen werden muss. In einer alternativen Variante ist die Rotationsgeschwindigkeit abhängig von dem Abstand des Fokuspunkts des Benutzers und des Objektmittelpunkts: Je größer der Abstand ist, desto schneller wird das Objekt rotiert. Dies führt zu einer kontrollierteren Rotationsbewegung als wenn die Rotationsgeschwindigkeit konstant bleibt.

Anders als die anderen Funktionen die in diesem Abschnitt besprochen werden, hat der Input des Blicks des Benutzers bei der Freien Rotation einen sofortigen, andauernden Effekt auf das Objekt. Das bedeutet, dass ein Wechsel der Blickrichtung eine sofortige Auswirkung auf die Rotationsrichtung des Objektes hat. Da die Augen aber nicht nur zur Steuerung der Rotationsrichtung, sondern auch zur Observierung des Objekts und der Veränderungen daran dienen, passiert es schnell, dass ungewollte Rotationen durchgeführt werden. Die Auswirkungen der Freien Rotation, gerade in Abhängigkeit mit dem sofortigen Effekt bei Änderung des Fokuspunktes, werden in Abschnitt 8.5 besprochen.

6.3.2 Transformations-Tool

Das Ziel war es, eine Möglichkeit zu finden, alle drei elementaren Funktionen (Rotation, Skalierung und Translation) einfach und schnell anzusprechen, ohne diese durch separate physische oder virtuelle Buttons zu aktivieren. Zudem war es notwendig, die Elemente unabhängig von der aktuellen Position und Rotation des Objekts stets sicht- und greifbar zu machen, ohne dass das Objekt Interaktionselemente verdeckt. Dennoch sollten die Interaktionselemente für die Funktionen nicht

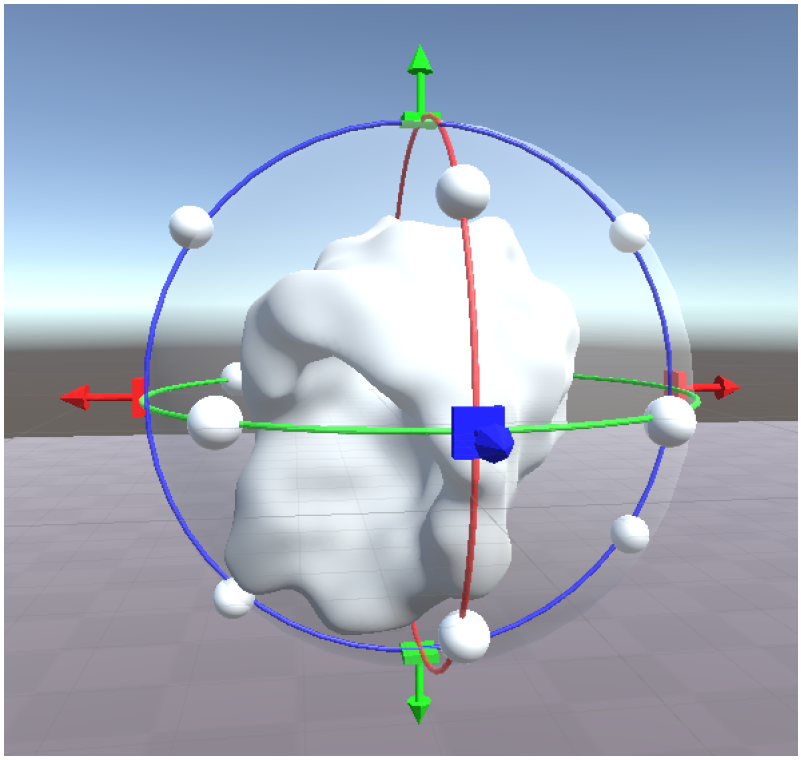

Abbildung 6.5: Das Transformations-Tool bietet visuelle Elemente zur Objekt-Manipulation: Die farbigen Reifen dienen der Rotation, die Pfeile der Translation und die weißen Sphären an den Ringen der Skalierung des Objekts.

das Objekt selbst verdecken. Das Transformations-Tool wurde ähnlich zu den Transformations-Tools der bekannten Anwendungen Blender, Maya, und sogar des Editors der Unity Engine gestaltet. Es nutzt die Dreidimensionalität des Raums aus und passt sich dem Objekt stets an. Das Tool hält folgende Funktionen in einem bereit: Rotation, Skalierung und Translation. Der Vorteil der Verwendung des Tools ist, dass der Benutzer nicht jede einzelne Funktion über ein Menü aktivieren muss; er kann die Funktionen direkt nutzen wenn er möchte. So kann er bei Bedarf problemlos und schnell zwischen den einzelnen Funktionen hin- und herwechseln.

Das Tool besteht aus farbigen Ringen zur Rotation um die einzelnen Achsen (rot für die X-Achse, grün für die Y-Achse und blau für die Z-Achse), Pfeilen in denselben Farben welche in dieselben Richtungen zeigen wie die Achsen für die Translation, und weiße Sphären die an den Ringen befestigt sind um das Objekt zu skalieren (siehe Abbildung 6.5). Im Folgenden werden die einzelnen Funktionen und Konzepte des Transformations-Tools vorgestellt.

Zwei-Klick-Rotation Diese Form der Rotation wird durch gleichzeitiges Betrachten eines farbigen Rings und einmaliges Drücken der Leertaste aktiviert. Es erscheinen daraufhin zwei pinke Linien, welche beide im Objektzentrum beginnen (siehe Abbildung 6.6). Die erste Linie schneidet den Ring an der Stelle des Rings die der Benutzer während des Drückens der Taste betrachtet hat, die andere folgt dem Blick des Benutzers. Beide Linien liegen dabei auf der Ebene, in welcher der eben aktivierte Ring liegt. Wird die Leertaste ein zweites Mal gedrückt rotiert das Objekt um

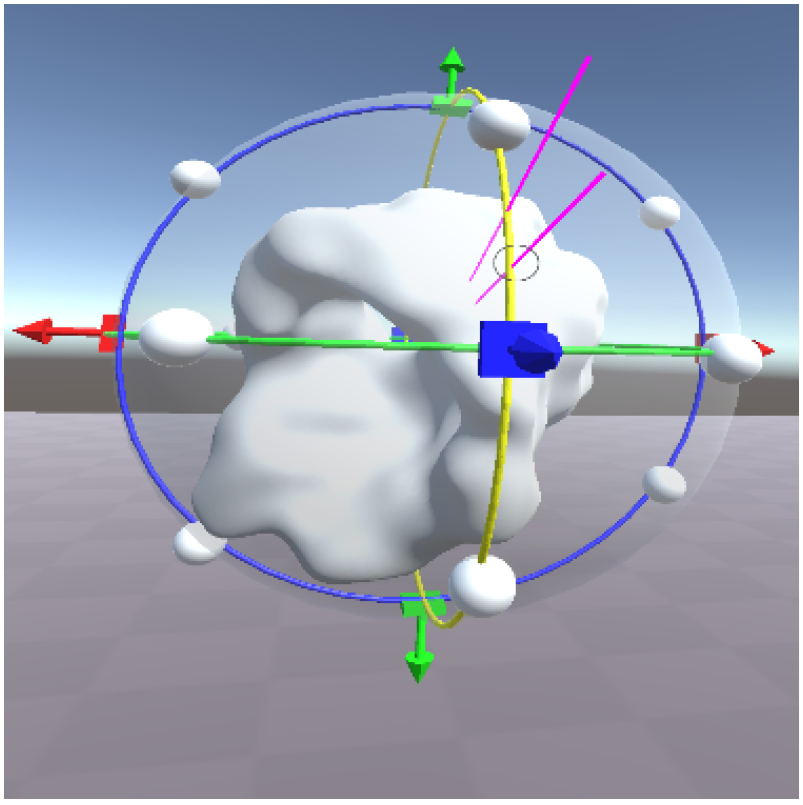

Abbildung 6.6: Die aktivierte Zwei-Klick-Rotation: Das Objekt wird nach dem zweiten Drücken der Leertaste um den kleineren der beiden Winkel rotiert den die beiden pinken Linien aufspannen. Eine der beiden Linien folgt dem Ring-Cursor.

die jeweilige Achse um den kleineren der beiden Winkel den die beiden Linien aufspannen und die Linien verschwinden anschließend. Die Rotationsrichtung ist abhängig von der Position des Fokuspunkts des Benutzers beim zweiten Drücken der Taste: Abhängig von der ersten Linie rotiert das Objekt in Richtung der vom Blick abhängigen Linie.

Während der Implementierung stellte sich die Frage, wie lang die beiden Linien sein sollten, um dem Benutzer möglichst klar zu zeigen wie sich die Rotation auf das Objekt auswirken wird. Zieht man die Linien vom Objektzentrum bis zum aktuellen Fokuspunkt des Benutzers, führt dies zu Linien, die sich in ihrer Länge stetig ändern. Das erschwert es die resultierende Rotation abzuschätzen. Setzt man die Linien auf die Länge des Radius des Objektes, reichen sie lediglich durch das Objekt hindurch bis an den Rand des Tools, die Linien an sich kann man fast nicht sehen. Deshalb entschied ich mich, ihre Länge etwas größer als den Objektradius zu wählen. So kann man, anders als bei der ersten Variante, sehen wenn sich die Linien weiter weg vom Benutzer befinden und kann die Winkelgröße in der Tiefe besser wahrnehmen. Die Linien sind immer gut sichtbar, ohne dass sie die Sicht auf das Objekt beeinträchtigen.

Die Farbe der Linien wurde so wie beim Laserstrahl Cursor gewählt (siehe Abbildung 6.1 rechts), da diese sehr auffällig ist und sich von dem Objekt und dem Tool gut abhebt.

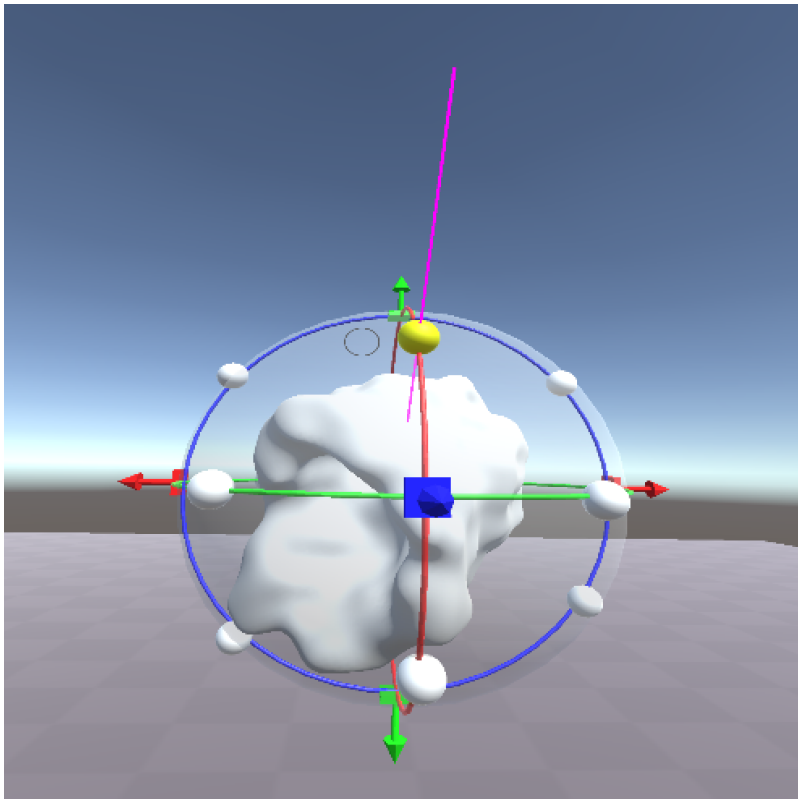

Abbildung 6.7: Die aktivierte Skalierung: Der Benutzer “zieht“ mit seinem Blick das Objekt an der weißen Sphäre größer oder “drückt“ es kleiner. Dabei gibt die pinke Linie an inwieweit das Objekt vergrößert oder verkleinert werden kann.

Kontinuierliche Rotation Diese Form der Rotation wird ebenfalls per Blick auf einen der farbigen Ringe des Tools aktiviert. In diesem Fall jedoch muss die Leertaste gedrückt gehalten werden um die Rotation um die jeweilige Achse zu starten. Die Rotationsgeschwindigkeit ist konstant. Lässt der Benutzer die Taste los, stoppt die Rotation. Die Rotationsrichtung ist abhängig davon welchen Bereich des Rings der Benutzer betrachtet hat während er anfing die Taste zu drücken, das heißt betrachtet der Benutzer den roten Ring oberhalb des blauen Pfeils während er beginnt die Taste zu drücken, dann dreht sich dieser Bereich zu ihm hin. Bei Betrachtung des Rings unterhalb des blauen Pfeils dreht sich dieser Bereich in Richtung des Benutzers.

Der große Vorteil bei dieser Art der Rotation ist es, dass der Benutzer nach Aktivierung der Rotation nicht mehr auf den Ring schauen muss, sondern frei das Objekt betrachten kann während es rotiert. Anders als bei der Zwei-Klick-Rotation und der Freien Rotation kann er also die Vorgänge direkt beobachten, denn die Rotation selbst ist dann nicht mehr am Blick des Benutzers gebunden. Zu weite Rotationen können so vermieden werden, der Benutzer verfügt über mehr Kontrolle.

Skalierung Der Skalierungsprozess wird durch Betrachtung einer der weißen Sphären und gleichzeitiges Drücken der Leertaste aktiviert. Eine pinke Linie erscheint, ähnlich wie die bei der Zwei-Klick-Rotation, welche durch die Position der Sphäre verläuft mit welcher interagiert wurde, und in Richtung des Objektzentrums zeigt (siehe Abbildung 6.7). Diese Linie zeigt an, wie sehr das

Objekt vergrößert oder verkleinert werden kann. Gesteuert wird die Skalierung durch den Blick des Benutzers: Wandert sein Blick in Richtung Objektzentrum, wird das Objekt verkleinert. Blickt er stattdessen vom Objekt weg, wird es vergrößert. Diese Methode basiert auf der aus dem Alltag gewohnten Art, Dinge mit den Händen zu verkleinern und zu vergrößern: Nimmt man beispielsweise ein zusammengeknülltes Blatt Papier, würde man mit beiden Händen daran ziehen um den Papierball zu vergrößern. Will man ihn verkleinern, drückt man ihn zusammen. Diese Form der Größenmanipulation habe ich an die Eye Tracking Technologie angepasst. Somit "zieht" der Benutzer mit seinem Blick das Objekt größer oder "drückt" es zusammen um die Größe zu manipulieren. Die Skalierung wird beendet, wenn die Leertaste nochmals gedrückt wird. Das Objekt behält dann die Größe, welche es zum Zeitpunkt des zweiten Drückens der Leertaste hat.

Da der Benutzer das Objekt unter Umständen viel zu klein oder zu groß skaliert (dies passiert besonders dann, wenn er die Interaktionsart nicht gewohnt ist), wurde ein Minimum und ein Maximum für einen einzelnen Skalierungsschritt gesetzt. Die maximale und minimale Größenveränderung ist dabei abhängig von der initialen Größe des Objekts vor Beginn der Skalierung. Dies bedeutet, dass der Benutzer mehrere Schritte benötigt um das Objekt wesentlich größer oder kleiner zu machen. Als Default Wert für die maximale Skalierung während eines einzelnen Skalierungsschritts wurde der Faktor zwei gewählt, für die minimale Skalierung 0.5. Das Objekt kann also maximal in seiner initialen Größe verdoppelt oder halbiert werden.

Die pinke Linie ist zum einen dazu da die Limits der Skalierung zu visualisieren, zum anderen dient sie als visuelle Leitlinie für den Benutzer während des Skalierungsvorgangs. Er kann dieser mit seinem Blick folgen während er das Objekt größer zieht oder zusammendrückt.

In früheren Versionen dieser Funktionalität wurden diverse Varianten der Positionierung der Skalierungslinie ausgetestet. In der ersten Version wurde die Linie vom Objektzentrum, durch die Sphäre und bis zur maximalen Größenänderung gezeichnet. Diese Variante ist jedoch irreführend bei der Verkleinerung des Objektes: Der Benutzer kann nur auf einen Teil der Linie blicken und das Objekt in seiner Größe verändern, es ist nicht ersichtlich wieso die Verkleinerung plötzlich stoppt. Um beide Limits zu visualisieren, wurde eine andere Variante getestet: Die Linie wird ausgehend von der Sphäre gezeichnet mit der interagiert wurde, und verläuft auch hier auf der Geraden, welche durch das Objektzentrum und der Position der Sphäre verläuft. Allerdings wird sie bei einer stattfindenden Vergrößerung durch den Benutzer von der Sphäre zum maximalen Punkt der Vergrößerung gezeichnet, bei Verkleinerung von der Sphäre bis zum minimalen Punkt der Verkleinerung. Sie passt sich also dynamisch an, und kann dem Benutzer so stets anzeigen, ob das Objekt gerade vergrößert oder verkleinert wird. Der Nachteil jedoch ist, dass die Linie schnell die Richtung ändert wenn der Benutzer die weiße Sphäre selbst betrachtet oder sein Blick nahe der Sphäre ist. Dies kann schnell ablenkend wirken, deshalb wurde anschließend die oben vorgestellte, statische Variante gewählt, bei der die Linie vom Minimum zum Maximum gezogen wird.

Translation Das Transformations-Tool bietet ebenfalls die Translation von Objekten an. Für die Translation wurde das bisherige Transformations-Tool mit Translationslinien in Form von langen, schmalen, weißen Zylindern erweitert, welche sich auf den verschiedenen Achsen direkt anschließend an den Pfeilen des Tools befinden. Um das Objekt zu bewegen muss der Benutzer auf einen Punkt auf einem der Translationslinien schauen und dabei einmalig die Leertaste drücken. Das Objekt (das heißt das Objektzentrum) bewegt sich dann zu diesem Punkt. Das Objekt kann

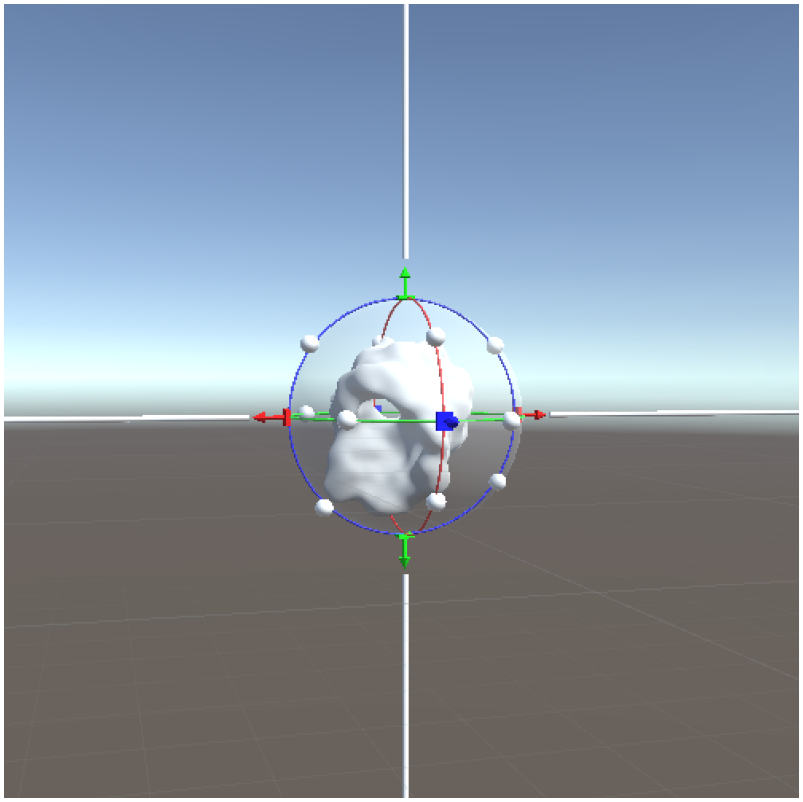

Abbildung 6.8: Die aktivierte Translation: Der Benutzer blickt auf einen Punkt auf den weißen Translationslinien und drückt dabei einmalig die Leertaste. Das Objekt wandert daraufhin an die Position dieses Punktes.

dabei nur auf den Achsen bewegt werden, da dies dem Benutzer die meiste Kontrolle über die Bewegung gibt und nicht zu ungewollten Verschiebungen in weiter Ferne des virtuellen Raumes führt.

Dies ermöglicht es, Objekte sehr schnell im Raum zu bewegen. Der Nachteil dieser Funktion ist jedoch, dass je weiter entfernt sich der vom Benutzer fokussierte Punkt auf der Translationslinie befindet, desto schwerer wird es die Translation selbst zu aktivieren, da die betrachtete Fläche immer kleiner wird. Ebenso ist das Objekt selbst unter Umständen im Weg, der Benutzer müsste sich dann um das Objekt herum bewegen um mit der Translationslinie interagieren zu können. Kleine Positionsänderungen sind ebenfalls problematisch: Soll das Objekt weniger weit auf einer Achse bewegt werden als dessen Radius, ist dies nicht in einem Schritt möglich; der Benutzer muss für exakte Positionierungen das Objekt zuerst woanders platzieren, um den gewünschten Ort auf der Translationslinie überhaupt auswählen zu können (ansonsten wäre das Objekt aufgrund seines Volumens selbst an der Position und nicht eine Translationslinie zur Interaktion).

Die erste Idee für die Umsetzung der Translationsfunktion bestand darin, dass der Benutzer einen der farbigen Pfeile betrachtet und die Leertaste einmal drückt. Anschließend kann er dann mit den Augen das Objekt im Raum verschieben, indem das Objekt mit seinem Blick auf der jeweiligen Achse mitwandert. Dies ist vergleichbar mit der weiter oben vorgestellten Skalierungsfunktion. Während der Implementierung hat sich jedoch herausgestellt, dass dieses Verhalten nicht ganz so

leicht umzusetzen ist: Das Objekt wanderte bei Aktivierung des Translationsprozesses stets sehr schnell vom Benutzer weg und war nicht mehr zurückzuholen. Dies lag vermutlich daran, dass die Szene an sich keinen Hintergrund besitzt, sondern nur einen endlose Horizont. Allgemein sind in keiner Richtung visuellen Grenzen gesetzt. Blickt der Benutzer also nicht auf ein bestimmtes Objekt, sondern in irgendeine Richtung, kann die FOVE nicht beurteilen wo der Blickpunkt sich befindet (das heißt wie nah oder weit entfernt). Somit wandert das Objekt bei dieser Translationsvariante in die endlose Weite. In Abschnitt 9.3 werden weitere Verbesserungsvorschläge zu dieser Variante vorgestellt und diskutiert.

Die nachfolgende Variante der Translationsfunktion sollte die bisherigen Schwierigkeiten umgehen. Hier muss der Nutzer auf einen der farbigen Pfeile blicken und dabei die Leertaste gedrückt halten. Dabei beginnt das Objekt sich mit konstanter Geschwindigkeit in die Richtung, in die der Pfeil zeigt, zu bewegen. Der Benutzer muss nach Aktivierung der Translation nicht weiter auf den Pfeil blicken, sondern kann seinen Blick über das bewegende Objekt schweifen lassen. Lässt der Benutzer die Leertaste los, wird der Translationsvorgang beendet und das Objekt bleibt an der Position, an der die Taste losgelassen wurden.

Dies hat natürlich den Vorteil, dass die Translation selbst nur bei Aktivierung vom Blick des Benutzers abhängig ist. Anschließend kann der Benutzer den Vorgang observieren, ohne dass dies den Prozess beeinflusst. Ungewollte Bewegungen werden somit vermieden. Die konstante Geschwindigkeit führt dazu, dass das Objekt kontrolliert bewegt wird, somit bewegt es sich nicht zu schnell vom Benutzer weg. Verglichen zu der initialen Translationsvariante ist diese Methode allerdings wesentlich langsamer, weitere Distanzen sind langsamer zurückzulegen. Ebenfalls kann es passieren, dass sich der Pfeil, mit welchem interagiert werden soll, vom Benutzer aus auf der anderen Seite des Objektes befindet. Möchte der Benutzer das Objekt in diese Richtung bewegen, muss er sich also zuerst um das Objekt herum bewegen, um den Pfeil zu erreichen und mit ihm interagieren zu können. Weitere Überlegungen zu möglichen Translationskonzepten werden in Abschnitt 9.3 besprochen.

7 Implementierung

In diesem Abschnitt wird auf das Setup für die Implementierung mit dem FOVE VR HMD eingegangen und die nötigen Vorbereitungen für die Entwicklung erläutert. Anschließend wird auf die bei der Anwendung verwendete Architektur eingegangen.

7.1 FOVE Setup

Um mit der Implementierung einer VR Anwendung für das FOVE VR HMD beginnen zu können, muss zuerst das FOVE SDK installiert werden. Dies ermöglicht die Verbindung zwischen der VR Brille und dem Computer. Darüber hinaus wird ein Plugin für die Entwicklung mit der Unity Engine angeboten. Das Plugin muss in das zuvor erstellte Unity Projekt importiert werden, damit das FOVE API innerhalb des Projektes angesprochen werden kann. Zusätzlich müssen bestimmte Objekte in jeder Szene platziert werden, um auf die Daten der VR Brille zugreifen zu können.

7.2 Architektur der Anwendung

Die grundlegende Architektur wurde von einem Beispielprojekt für eine VR Anwendung von Unity3d übernommen und an die Verwendung von Eye Tracking angepasst, um die Verarbeitung der Eingaben aus den verschiedenen Quellen (Controller und Eye Tracking) zu vereinfachen [Gol18]. Somit muss nicht jedes Objekt ständig mögliche an ihn gerichtete Eingaben abfragen, sondern muss nur an andere Klassen melden wenn eine Interaktion mit diesem stattfand.

Die Interaktionen zwischen dem Benutzer und der Anwendung gleicht einem Nachrichtensystem. Anstatt die Eingaben für jedes Objekt getrennt zu definieren, wurden diese in Form von Events in der Klasse *VRInput* zusammengefasst: Jedes Mal, wenn der Benutzer eine physische Taste drückt, werden die dort definierten Events ausgelöst. Parallel dazu werden die Eingaben der VR Brille in der Klasse *VREyeRaycaster* registriert und ebenfalls den dazugehörigen Events zugeordnet. Erst wenn das jeweilige Event auch in der *VREyeRaycaster* Klasse ausgelöst wurde, wird die Aktion ausgeführt. Ein Beispiel hierfür wäre beispielsweise die Selektion eines Buttons des Hauptmenüs: Erst wenn der Benutzer den Button betrachtet und dabei die Interaktionstaste drückt, wird der Button selektiert. Zudem befindet sich auf jedem Interaktionselement ein Skript mit dem Namen *VRInteractiveItem*. Dieses Skript ermöglicht die Interaktion mit dem jeweiligen Objekt: Wird eine Interaktion an diesem Objekt durchgeführt, löst die *VREyeRaycaster* Klasse die zum Event gehörenden Methoden des jeweiligen Objekts aus.

Für die Interaktion sind also drei Bereiche wichtig: der physische Input durch einen Controller oder eine Tastatur, der Eye Gaze Input über die VR Brille und das jeweilige Objekt in der Szene mit dem interagiert wird. So können die Eingaben von den virtuellen Objekten entkoppelt werden.

8 Benutzerstudie

Im Rahmen dieser Arbeit wurde eine Benutzerstudie durchgeführt, um die entwickelten Konzepte zu testen und ihre Benutzerfreundlichkeit zu evaluieren. Die Studie fand in einem kleinen Rahmen statt (12 Teilnehmer), und hat deshalb nur eine eingeschränkte Aussagekraft. Die Daten, welche dafür verwendet wurden, bestehen aus einem Mesh einer molekularen Oberfläche. Diese wurden deshalb gewählt, da es sich um ein 3D Objekt, welches den Teilnehmern völlig unbekannt ist, aber dennoch über eine komplexe und interessante verfügt. Durch die Benutzung eines unbekannten, abstrakten Objekts können sich die Teilnehmer mehr auf die Manipulations-Aufgaben konzentrieren und werden nicht abgelenkt.

8.1 Teilnehmer

Zwölf Teilnehmer nahmen an der Studie teil, davon waren acht männlich und vier weiblich. Sie waren im Alter zwischen 27 und 74. Fünf hatten bereits Erfahrungen mit VR Brillen und einer mit Eye Tracking.

8.2 Aufbau

Die Teilnehmer wurden in zwei Gruppen geteilt: Eine Gruppe musste die Aufgaben mit eingeschaltetem Eye Tracking durchführen, die andere mit ausgeschaltetem Eye Tracking. Die Gruppe mit ausgeschaltetem Eye Tracking musste die Aufgaben stattdessen mithilfe von Kopfbewegungen durchführen, wie es auch bei gewöhnlichen VR Brillen ohne Eye Tracking der Fall ist. Beide Gruppen mussten dieselben Aufgaben durchführen um so vergleichen zu können, wie sich der Einfluss von Eye Tracking auf die Teilnehmer selbst und den Erfolg bei den Aufgaben auswirkt. Es wurden dabei weder Zeit noch Präzision gemessen, während der Beobachtung der Teilnehmer wurden allerdings Notizen zur ungefähren Durchführungsdauer festgehalten. Der Fokus lag auf der Benutzerfreundlichkeit und dem Komfort der verschiedenen Interaktionstechniken.

Das Ziel war es, Menschen mit unterschiedlicher Erfahrung mit VR und Eye Tracking und unterschiedlichen Alters zu finden, um die Wirkung der entwickelten Konzepte auf die verschiedenen Nutzergruppen möglichst neutral zu halten, beziehungsweise auch um die Unterschiede zwischen gewöhnlichen VR Brillen und solchen mit Eye Tracking Technologie zu verdeutlichen. Das heißt die Teilnehmer mit VR Erfahrung wurden bewusst mit der Eye Tracking Technologie konfrontiert um zu sehen, wie gut sie bei der Interaktion mit Elementen mit der Hilfe ihrer Augen als Input zurechtkommen. Hingegen war es schwieriger Teilnehmer zu finden welche Eye Tracking Erfahrung vorweisen konten, weshalb der eine Teilnehmer mit Vorerfahrung in diesem Bereich die Aufgaben

mit eingeschaltetem Eye Tracking durchführte. So war es nicht nur möglich einen Einblick darin zu bekommen wie gut Menschen ohne Eye Tracking Erfahrung mit den Konzepten zurechtkommen, sondern auch solche mit Eye Tracking Erfahrung. Die Teilnehmer

Die Studie bestand aus einem praktischen Teil in welchem die Teilnehmer unterschiedliche Aufgaben innerhalb der entwickelten Anwendung durchführen mussten, und einer dazugehörigen schriftlichen Befragung in Form eines Fragebogens den sie anschließend ausfüllen mussten. Während des praktischen Teils wurden sie beobachtet, und eine Kamera innerhalb des FOVE Setups zeichnete sowohl die Augenbewegungen des Teilnehmers, als auch seine Kopfbewegungen auf (siehe Abschnitt 3.1).

Der praktische Teil bestand aus vier verschiedenen Szenen. In jeder Szene musste der Teilnehmer das Hauptmenü öffnen, ein Untermenü aufrufen, damit interagieren, zum Hauptmenü zurückkehren, ein Objekt für die Objekt-Manipulation laden und anschließend unterschiedliche Manipulations-Aufgaben durchführen. In jeder Szene wurde ein anderes Cursor-Design benutzt bis auf die letzte Szene, in der kein visueller Cursor benutzt wurde. In der ersten Szene wurde der Ring-Cursor ohne Smoothing Funktion verwendet, in der zweiten das Fadenkreuz und die Smoothing Funktion wurde mit einem Wert von 50 eingeschaltet (das heißt der Durchschnitt wurde über die letzten 50 Blickpunkte gebildet, siehe den Absatz zu Smoothing in Abschnitt 6.1). Den Wert von 50 für das Smoothing bei der zweiten Szene wurde so gewählt, dass der Benutzer einen deutlichen Unterschied zu ausgeschaltetem Smoothing feststellen kann, das heißt der Cursor zieht deutlich nach, ist nach vorigen Tests mit anderen Benutzern aber dennoch nicht deutlich zu langsam für die meisten Nutzer. In der dritten Szene wurde der Laserstrahl verwendet, und der Benutzer konnte entscheiden wie stark das Smoothing des Cursors sein sollte. Da es in VR Umgebungen gängig ist einen visuellen Cursor zu haben, sollte mithilfe der vierten Szene, in welcher kein Cursor verwendet wurde, die Notwendigkeit eines Cursors mit und ohne Eye Tracking evaluiert werden.

8.3 Aufgaben

In diesem Abschnitt werden die Aufgaben beschrieben, welche die Teilnehmer verrichten mussten.

Rotation Bei der Rotation mussten die Teilnehmer eine rote Sphäre finden, welche sich irgendwo am Objekt selbst befindet und zu Beginn nicht sichtbar war. Das Objekt sollte so gedreht werden, dass die Sphäre zum Benutzer zeigt. Bei der Freien Rotation mussten sie das Objekt so drehen, dass sich die rote Sphäre in der Dead Zone befindet.

Für die folgenden Rotations-Aufgaben wurde das Transformations-Tool verwendet. Bei der Zwei-Klick-Rotation und der Kontinuierlichen Rotation musste der Teilnehmer versuchen das Objekt so zu drehen, dass die rote Sphäre sich hinter dem blauen Pfeil des Transformations-Tools befindet. Die Ziele der Rotations-Aufgaben waren alle ähnlich gestaltet, um zu sehen wie gut die Teilnehmer mit den verschiedenen Interaktionstechniken zurechtkommen und ihre Benutzerfreundlichkeit zu vergleichen.

Skalierung Für die Skalierungs-Aufgabe wurden zwei zusätzliche Objekte als Referenz in der Szene platziert, eines auf jeder Seite des Interaktionsobjekts. Ein Referenzobjekt war kleiner als das zu manipulierende Objekt, und eines größer. Das zu manipulierende Objekt musste zuerst auf eine ähnliche Größe reduziert werden wie das kleinere der beiden Referenzobjekte, und anschließend so vergrößert werden, dass es eine ähnliche Größe wie das größere der beiden Referenzobjekt aufwies.

Translation Für die Translations-Aufgabe wurden wieder zwei Referenzobjekte im virtuellen Raum platziert: Eines befand sich links oberhalb der Position des zu verschiebenden Objekts, eines rechts unterhalb. Lediglich die Translation auf der X- und Y-Achse wurden getestet, da Translationen auf der Z-Achse nur möglich sind, wenn der Benutzer dazu in der Lage ist sich um das Objekt herum zu bewegen. Sonst kann er nicht sinnvoll mit den zugehörigen Elementen des Transformations-Tools interagieren. Die Anwendung wurde in sitzender Position getestet. Der Benutzer musste das Objekt zuerst auf die Positon des einen, dann auf die Position des anderen Referenzobjekts bewegen. Für diese Aufgabe wurde kein visueller Cursor verwendet.

8.4 Der Fragebogen

Nach dem praktischen Teil mussten die Teilnehmer einen Fragebogen ausfüllen in dem sie zuallererst nach Beeinträchtigungen in ihrer Sehkraft und bisherigen Erfahrungen mit VR Brillen und Eye Tracking befragt wurden. Anschließend wurden sie der Reihe nach zu ihrer Meinung zu den verschiedenen Funktionen innerhalb der Anwendung befragt.

Es gab ein Cursor-Ranking, in dem sie die Cursor nach ihren persönlichen Vorlieben in eine Rangliste bringen mussten. So war es möglich, klare Präferenzen zwischen den einzelnen Cursor-Designs herauszufinden.

Anschließend wurden der Reihe nach Aussagen gegeben, zu denen die Teilnehmer jeweils anhand einer Likert-Skala Stellung nehmen sollten. Die Skala bestand hierbei aus fünf Antwortmöglichkeiten die angekreuzt werden konnten, das heißt es gab ebenfalls eine ”neutralAntwort.

Die Aussagen zum Cursor waren allgemein gehalten, um die Notwendigkeit des selbigen zu erfahren. Hierbei sollte der Teilnehmer dazu Stellung nehmen, ob er einen Cursor hilfreich, die Benutzung mit einem Cursor angenehm und die Benutzung ohne Cursor angenehm fand. Zusätzlich wurde gefragt, ob der Teilnehmer den Cursor mit Smoothing oder ohne besser fand, oder ob er keine Präferenz hatte.

Die Aussagen zu den Menüs und der Objekt-Manipulation ähnelten sich sehr: Stets wurde die Aussage getätigt, dass die Bedienung dem Teilnehmer leicht gefallen ist. Sowohl bei der Rotationsaufgabe ohne Tool, als auch bei der Rotationsaufgabe, der Skalierungsaufgabe und der Translationsaufgabe mit Tool wurde eine zusätzliche Aussage darüber getätigt, dass der Teilnehmer die jeweilige Funktion angenehm fand. Dies wurde so gewählt, da eine Aufgabe einem Teilnehmer leicht fallen kann, obwohl er die Funktionalität nicht angenehm in der Benutzung findet. Dies gibt also zusätzlich Informationen über die Benutzerfreundlichkeit wieder.

Zuletzt wurden drei zwei Textfelder angegeben, in denen der Teilnehmer frei mitteilen konnte was ihm bei der Anwendung gefallen und was ihm nicht gefallen hat. Zudem bot ihm ein weiteres Textfeld Platz für die Mitteilung sonstiger Gedanken zur Anwendung.

8.5 Ergebnisse

In diesem Abschnitt werden die Ergebnisse der Studie vorgestellt. Diese Ergebnisse spiegeln neben den Beobachtungen der Teilnehmer während der Durchführung auch die Inhalte der Fragebögen wieder und repräsentieren somit äußere und innere Reaktionen der Teilnehmer auf die Anwendung und deren Interaktionskonzepte. Zudem werden die Beobachtungen während der Durchführung der Studie interpretiert, um daraus zukünftige Designentscheidungen abzuleiten.

Menüs Bei der ersten Aufgabe in jeder Szene, dem Aufruf des Hauptmenüs, fiel auf, dass die Teilnehmer sehr oft das Objektmenü aufrufen wollten. Aufgrund eines Fehlers war das Objektmenü stets sichtbar, auch wenn das Objekt selbst noch nicht geladen war. Als die Teilnehmer jedoch dazu angewiesen wurden das Hauptmenü zu öffnen, glitt ihr Blick stets zuerst zum Objektmenü und interagierten damit, obwohl das Hauptmenü lediglich mit der Enter-Taste auf der Tastatur aufzurufen war. Es kam zu Verwirrung auf Seiten des Teilnehmers, der Leiter der Studie musste ihn auf das korrekte Verhalten zum Aufruf des Hauptmenüs hinweisen beziehungsweise ihn daran erinnern. Trotz des unbeabsichtigten Fehlers innerhalb der Anwendung, konnten daraus interessante Erkenntnisse gewonnen werden. Eventuell liegt dieses Verhalten der Teilnehmer daran, dass das Hauptmenü keine visuelle Präsenz innerhalb des virtuellen Raumes hat. Der Teilnehmer interagiert stets mit virtuellen Objekten, doch beim Hauptmenü muss er mit der Tastatur interagieren, welche er nicht sehen kann. Der Fokus des Teilnehmers liegt also vermehrt auf virtuellen Objekten. Ebenso stört die Abwesenheit einer visuellen Repräsentation des Hauptmenüs die Immersion, die Aufmerksamkeit des Teilnehmers muss bei jedem Aufruf des Hauptmenüs auf den ëchten"Raum um ihn herum gezogen werden. Unter diesen Aspekten stellt sich nun die Frage, ob es eventuell doch notwendig ist, ein visuelles Element im virtuellen Raum zu haben, wenn Interaktionen damit durchgeführt werden sollen, beziehungsweise ob es notwendig ist, virtuelle Interaktionselemente von solchen wie sie in der wahren Welt existieren (das heißt Tastaturen, Controller, ..) stärker voneinander zu trennen oder diese zusammen gar nicht innerhalb derselben Anwendung zu verwenden. Die Teilnehmer gaben im Fragebogen mehrheitlich an, dass ihnen die Interaktion mit den Menüs leicht gefallen ist und sie angenehm war. Ein Teilnehmer bemängelte die Größe der Buttons im Hauptmenü, diese sollten wesentlich größer sein. Dieser Teilnehmer verwendete Eye Tracking. Da sonst kein anderer Teilnehmer zu der Größe der Buttons Stellung nahm könnte dies auf einen inkorrekten Sitz der Brille und eine gestörte Registrierung der Pupillen des Teilnehmers zurückzuführen sein.

Rotation Die Aufgabe für die *Freie Rotation* war die erste Aufgabe mit der die Teilnehmer konfrontiert wurden. Dies könnte der Grund sein, aus dem diese einige Zeit benötigten um sich an die Funktion zu gewöhnen. Nachdem sie sich an die Auswirkungen ihrer Blickrichtung gewöhnt hatten, waren sie relativ schnell darin das Objekt so zu rotieren wie sie es wollten.

Im Vergleich dazu schien die Verwendung der *Zwei-Klick-Rotation* wesentlich schwieriger für die Teilnehmer zu sein. Es dauerte wesentlich länger sich daran zu gewöhnen wie sich ihre Blickrichtung während des zweiten Tastendrucks auf die Rotationsrichtung auswirkt als es bei der freien Rotation

der Fall war. Zudem konnten die Teilnehmer nur schwer einschätzen, wie sehr sich der kleinere der beiden Winkel zwischen den Visualisierungslinien auf die eigentliche Rotation auswirkt. Daraus folgten viele unbeabsichtigte Rotationen des Objekts. Gerade in Bezug auf die Rotationsrichtung in Verbindung mit der Position der Linien benötigten die Teilnehmer wesentlich mehr Zeit für die Beendigung der Aufgabe. Obwohl ein Lernprozess gut zu erkennen war, blieb es für die Teilnehmer mit eingeschaltetem Eye Tracking eine Herausforderung, kleinere Rotationen durchzuführen, da sie ihre Augen sehr ruhig auf einen Punkt fixieren mussten, um einen sehr geringen Winkel zwischen den Linien zu schaffen bevor die Rotation mithilfe der Leertaste bestätigt wurde. Dazu kam, dass Vor- und Zurückbewegungen der Augen zur Betrachtung des Objekts, der Visualisierungslinien und des aufgespannten Winkels kaum möglich waren bei der Einstellung des korrekten Winkels, da diese Bewegungen die Position der einen Linie veränderte und somit auch die Winkelgröße und die daraus resultierende Rotation. Eine Betrachtung der Situation und eine Einschätzung des Resultats waren also schwer. Die Teilnehmer ohne Eye Tracking waren viel eher dazu in der Lage den Cursor zu kontrollieren und konnten wegen der gesteigerten Kontrolle der positionierbaren Linie und der Unabhängigkeit des Blicks die Interaktion schneller durchführen und die Aufgabe schneller lösen als die andere Gruppe. Da die Teilnehmer ohne Eye Tracking ihren Kopf unabhängig von ihren Augen bewegen konnten und somit die eigentliche Interaktion besser überblicken konnten, fiel es ihnen Leichter die Linie zufriedenstellen zu positionieren.

Die *Kontinuierliche Rotation* war im Allgemeinen einfacher für die Teilnehmer: Die konstante Rotationsgeschwindigkeit und die Freiheit das Objekt betrachten zu können ohne dabei dauerhaft bestimmte Teile des Tools betrachten zu müssen während die Leertaste kontinuierlich gedrückt wird half den Teilnehmern die Kontrolle über die Rotation zu behalten. Mit dieser Rotations-Variante wurde die Aufgabe am schnellsten gelöst, was darauf hinweisen könnte dass dies die beste Lösung ist. Die Teilnehmer ohne Eye Tracking waren ähnlich schnell bei der Durchführung der Aufgabe, mussten jedoch ihre Köpfe wesentlich mehr bewegen während des Prozesses.

Die Freie Rotation kam bei den Teilnehmern durchweg gut bis sehr gut an: Vier von sieben Teilnehmern der Gruppe mit Eye Tracking ist sie leicht gefallen, sogar sechs von sieben fanden sie angenehm. Ähnlich war dies auch bei der Gruppe ohne Eye Tracking: Vier von fünf ist sie leicht gefallen, drei von fünf fanden sie angenehm. Die Rotation mit dem Tool kam bei den Teilnehmern ohne Eye Tracking jedoch besser an: Die Antworten dazu waren durchweg positiv. Bei den Teilnehmern mit Eye Tracking jedoch gingen die Meinungen sehr auseinander: Vier von sieben ist die Rotation mit Tool leicht gefallen, eher angenehm bis angenehm empfanden es vier von sieben. Jeweils zwei Teilnehmern ist die Rotation mit dem Tool eher nicht leicht gefallen und fanden sie eher nicht angenehm. Die benötigte Präzision bei der Interaktion mit den Ringen des Tools war ohne Eye Tracking wesentlich leichter zu erreichen, weshalb die Resonanz bei dieser Gruppe besser ausfiel. Probleme bei der Fokussierung der Ringe führten schnell zu Frustration und sorgten deshalb vermutlich für ein breiter gefächertes Meinungsbild bei den Teilnehmern mit Eye Tracking.

Skalierung Bei der *Skalierung* fiel selbst bei eingeschaltetem Eye Tracking sehr stark auf, dass manche Teilnehmer das Gefühl hatten ihren Kopf während der SKalierung übermäßig bewegen zu müssen, obwohl allein die Augenbewegungen darauf Einfluss nehmen konnten. Selbst nachdem sie darauf hingewiesen wurden behielten sie übermäßige Kopfbewegungen bei. Es fiel ihnen offenbar schwer, nur ihre Augen zu benutzen. Teilnehmer mit VR Erfahrung taten dies umso mehr. Dies kann daran liegen, dass diese es eher gewohnt sind, dass allein ihre Kopfbewegungen für die Positionierung des Interaktionspunkts genutzt werden können, und nicht die Augen. Allgemein wurden in beiden

Gruppen viele Kopfbewegungen während dieser Aufgabe beobachtet. Allein der Teilnehmer mit Eye Tracking Erfahrung bewegte seinen Kopf nur sehr gering. Es scheint, als wäre die Kontrolle mithilfe des Blicks in dieser Situation weniger intuitiv als es bei den anderen Szenarios der Fall war.

Ebenso überraschend war es, dass nahezu jeder Teilnehmer stets dieselbe weiße Sphäre für die Aktivierung des Skalierungsvorgangs wählte, unabhängig von der Distanz zum Referenzobjekt für die jeweilige Skalierungsaufgabe. Da die beiden Referenzobjekte an unterschiedlichen Seiten der Szene positioniert waren, war dies sehr überraschend, denn so mussten sie ihren Kopf wesentlich mehr bewegen wenn sie zwischen den Referenzobjekten und dem zu skalierenden Objekt hin- und herschauen wollten während der einzelnen Skalierungsschritte als wenn sie die nächst mögliche weiße Sphäre gewählt hätten. Dies galt selbst für den Teilnehmer mit Eye Tracking. Dieses Verhalten lässt viele Mutmaßungen offen. Es könnte beispielsweise in vorigen Erfahrungen mit der Interaktion liegen, dass die Teilnehmer mitunter nicht erfolgreich mit Elementen in der Szene interagieren konnten da der Interaktionspunkt nicht auf des Elements ruhte. Somit wählen sie lieber stets dieselbe Sphäre zur Aktivierung der Skalierung aus, da diese bisher erfolgreich war. Manche Teilnehmer waren auch irritiert davon, dass sie die weißen Sphären der Ringe auf der Rückseite des Objekts sehen, mit diesen aber nicht interagieren konnten. Dies fiel vermehrt bei den Sphären auf, kam jedoch auch bei der Interaktion mit den Ringen bei der Rotation vor. Ähnliche Elemente zu sehen, aber nicht mit diesen interagieren zu können, scheint allgemein unlogisch für die Teilnehmer zu sein, auch wenn diese sich auf der Rückseite eines Objekts befinden. In Kapitel 9 werden Überlegungen hierzu diskutiert.

Die Sphären, welche sich weiter oben am Objekt befanden, waren generell schwerer für die Teilnehmer zu erreichen, da sie ihren Kopf weit nach oben drehen mussten. Bei einem weiter entfernten Objekt wäre dies nicht mehr der Fall gewesen, dennoch wird der Benutzer so in seiner Wahl der Interaktionselemente eingeschränkt.

Gerade für die Teilnehmer ohne Eye Tracking war diese Aufgabe sehr anstrengend, da es für jegliche Interaktion und Betrachtung der Referenzobjekte nötig war, immense Kopfbewegungen zu machen. Im Allgemeinen kann man feststellen, dass bei dieser Aufgabe die Verwendung von Eye Tracking wesentlich weniger anstrengend ist für den Benutzer.

Betrachtet man die Auswertung der Fragebögen, ergibt sich ein interessantes Bild: Obwohl die Skalierung für die Teilnehmer ohne Eye Tracking viele Kopfbewegungen benötigt, ist die Skalierung von diesen Teilnehmern durchweg als positiv wahrgenommen worden. Zwei der fünf Teilnehmer dieser Gruppe ist die Skalierung leicht gefallen, drei ist sie eher leichtgefallen. Vier fanden die Skalierung angenehm, einer eher angenehm. Hingegen sind die Antworten bei der Gruppe mit Eye Tracking über die drei mittleren Antwortmöglichkeiten verteilt: Fünf von sieben ist die Skalierung eher leicht gefallen, vier fanden sie eher angenehm. Da die Cursor einen großen Einfluss hatten auf die Durchführung der Aufgaben, lässt sich dieses Ergebnis eventuell mit der Tatsache erklären, dass der bei dieser Aufgabe verwendete Laserstrahl bei den Teilnehmern am schlechtesten abgeschnitten hat. Die Interaktion mit den Sphären wurde durch Schwierigkeiten bei der Positionierung des Laserstrahls erschwert.

Translation Bei der *Translation* traten mit und ohne Eye Tracking diverse Probleme auf. Je weiter entfernt der Fokuspunkt auf den Translationslinien (das heißt den Zylindern) war, desto eher fand keine Translation statt. Dies liegt daran, dass die Zylinder von Collidern umgeben sind,

welche dieselbe Form haben wie die Zylinder, aber deutlich größer sind. Die Interaktion mit den Translationslinien erfolgt über deren Collider, das heißt betrachtet der Benutzer einen Punkt auf diesem Collider und drückt die Enter-Taste, dann verschiebt sich das Objekt auf diesen Punkt auf dem Collider in Achsenrichtung (das heißt das Objekt bewegt sich nicht auf allen Achsen, sondern nur der in welche die Translationslinie zeigt). Ruht der Blick des Benutzers nicht auf dem Collider passiert nichts. Je weiter entfernt sich die Translationslinien und ihre Collider befinden, desto geringer wird die Fläche mit der der Benutzer interagieren kann. Ebenso trug die Perspektive zu einer Verringerung der interagierbaren Fläche bei. Die Teilnehmer mit Eye Tracking empfanden es als schwer einen Punkt auf der Translationslinie zu fixieren, oftmals passierte nichts, obwohl sie davon überzeugt waren die Translationslinie betrachtet zu haben. So kam es zu viel Verwirrung und Frustration. Nahe Teile der Translationslinie waren einfach zu fokussieren, aber die Teilnehmer kamen schnell an ihre Grenzen. Sie begannen oftmals, ihre Augen zusammenzukneifen, um weiter entfernte Punkte auf den Translationslinien besser fokussieren zu können, dies könnte aber ebenfalls zu den Schwierigkeiten beigetragen haben, da die Eye Tracking Kameras der FOVE unterhalb der Augen sitzen und dann die Pupillen und ihre Position und Bewegung nicht mehr einwandfrei verfolgen können. Da in dieser Szene kein Cursor zur Verfügung stand, konnten sie nicht überprüfen ob die Eye Tracking Hardware ihrem Blick korrekt folgt. Manche Teilnehmer begannen, absichtlich neben die Translationslinien zu blicken, da sie einen Fehler vermuteten und diesen auszugleichen versuchten.

Die Teilnehmer ohne Eye Tracking hatten aufgrund des fehlenden Cursors kein Indiz dafür, wo sich der Interaktionspunkt im Sichtfeld genau befindet. Dies machte es enorm schwer mit den schmalen Translationslinien zu interagieren, insbesondere wenn die Fokuspunkte weiter entfernt waren. Da es das Ziel war die Funktionalität zu testen und nicht nur das Fehlen des Cursors wurde nach einer gewissen Zeit wieder der Ring-Cursor aktiviert. Dennoch hatten die Teilnehmer große Probleme, den Ring-Cursor in größerer Entfernung auf den Translationslinien zu platzieren. Keiner der Teilnehmer ohne Tracking war dazu in der Lage, das Objekt auf die exakte Position der Referenzobjekte zu platzieren. Obwohl die Teilnehmer mit Eye Tracking ebenfalls Probleme bei der Positionierung hatten, waren sie dennoch dazu in der Lage das Objekt wesentlich schneller und präziser an die Positionen der Referenzobjekte zu platzieren. Dies war zumindest bei ausgeschaltetem Cursor zu erwarten, da die Teilnehmer ohne Eye Tracking keinerlei Möglichkeit hatten zu validieren, ob sich der Interaktionspunkt auf den Translationslinien befindet.

Aufgrund dieses entwickelten Konzepts zur Translation waren kleine Bewegungen des Objekts nicht möglich. Die Teilnehmer waren lediglich dazu in der Lage, Punkte auf den Interaktionslinien zur Translation auszuwählen (das heißt nur außerhalb des Objekts und des Transformations-Tools). Das bedeutet, dass das Objekt Strecken, welche kleiner waren als der Objektradius, nicht zurücklegen kann. Alle Teilnehmer machten diese Feststellung recht zügig und empfanden dies als frustrierend. Oftmals bewegten sie das Objekt zuerst eine große, grobe Strecke, und wollten dann anschließend kleine Modifikationen an der Position vornehmen. Manche änderten daraufhin ihre Strategie und bewegten zuerst das Objekt wesentlich weiter entfernt vom Zielpunkt, und versuchten dann den Zielpunkt in einem Schritt genau zu treffen. Dies benötigt allerdings oft mehrere Versuche.

Im Zuge dessen fiel ihnen auch auf, dass die Abschnitte der Translationslinien, welche aufgrund der Perspektive von ihnen aus hinter dem Objekt befanden, nicht angewählt werden konnten. Je nachdem wo sich der Benutzer gerade im virtuellen Raum befindet, kann er bestimmte Translationslinien nicht erreichen und das Objekt demnach auch nicht frei in jede Richtung bewegen. Auch dies versuchten

sie wie oben erwähnt dadurch zu umgehen, indem sie das Objekt zuerst in eine andere Richtung bewegten, um dann anschließend den zuvor vom Objekt verdeckten Abschnitt der Translationslinie erreichen zu können.

Die Auswertung der Fragebögen ergab bei den Teilnehmern ohne Eye Tracking ein klares Bild: Aufgrund des fehlenden Cursors bei dieser Aufgabe ist die Translation den Teilnehmern nicht leicht gefallen, und sie war nicht angenehm. Bei den Teilnehmern mit Eye Tracking waren die Ergebnisse eher auf die verschiedenen Antwortmöglichkeiten verteilt: Die Aussagen darüber, ob die Translation leicht gefallen oder angenehm war, komplett widersprechen. Dennoch gab es keine deutliche Tendenz: Drei der sieben Teilnehmer ist die Translation eher nicht leicht gefallen, einem ist sie eher leicht gefallen und dreien ist sie leicht gefallen. Eine ähnliche Aufteilung zeigt sich bei der Aussage, ob die Translation angenehm war: Zwei von sieben antworteten mit ”trifft eher nicht zu”, einer mit ”neutral”, zwei mit ”trifft eher zu”, und zwei mit ”trifft zu”. Man kann erkennen, dass es einen großen Unterschied gibt zwischen den beiden Gruppen. Selbst nach Einschalten eines visuellen Cursors bei der Gruppe ohne Eye Tracking waren sie von der Funktionalität nicht überzeugt, auch wenn sie darauf positiver reagierten. Es scheint, als wäre der Einsatz von Eye Tracking wesentlich effizienter in diesem Kontext als ohne. Dennoch stellt sich die Frage, weshalb genau die Gruppe mit Eye Tracking solch eine Variation an Meinungen aufwies.

Die Wahl des Cursors Die Wahl des Cursor-Designs hatte einen großen Einfluss darauf wie die Teilnehmer mit Elementen in den Szenen interagierten.

Die Teilnehmer mit Eye Tracking waren überraschenderweise ohne Cursor am schnellsten bei ihren Interaktionen. Der Ring-Cursor und das Fadenkreuz hatten eine ähnliche Wirkung auf die Teilnehmer, wobei der Ring-Cursor vermutlich wegen der fehlenden Visualisierung des Interaktionspunktes etwas schlechter abgeschnitten hat. Dies spiegelte auch die Auswertung der Fragebögen wider: Die Teilnehmer mit Eye Tracking bevorzugten mehrheitlich die Abwesenheit eines Cursors, der Ring-Cursor und das Fadenkreuz waren etwa gleichauf, und der Laserstrahl wurde am wenigsten bevorzugt. Dabei fiel jedoch auf, dass diejenigen ohne Korrigierung ihrer Fehlsichtigkeit eher dazu tendierten einen Cursor zu bevorzugen, wobei dies abhängig vom Grad ihrer Fehlsichtigkeit abhing. Die geringe Anzahl derer, die eine Fehlsichtigkeit besaßen und sich in der Gruppe mit aktiviertem Eye Tracking befanden war leider zu klein, als dass sich daraus klare Aussagen ableiten ließen, dennoch gibt es gewisse Tendenzen. Gerade da die verwendete Hardware nicht mit einer Brille für die Korrigierung einer Fehlsichtigkeit kompatibel ist, wäre dies eine Möglichkeit zur Verbesserung des Designs für diejenigen, die auf eine Brille angewiesen sind (siehe Kapitel 9). Das schlechte Abschneiden des Laserstrahls im Vergleich zum Ring-Cursor und dem Fadenkreuz lässt Raum für Spekulationen offen. Dadurch, dass die Linie vom Benutzer ausgeht und nur eine bestimmte, feste Länge aufweist, unabhängig davon welches Objekt gerade fokussiert wird, könnte zu mehr Verwirrung bei der Tiefenwahrnehmung führen als bei den anderen beiden Varianten. Beim Ring-Cursor und dem Fadenkreuz wird nicht die gesamte Distanz des Blicks visualisiert, sondern quasi nur der Fokuspunkt. Obwohl sich der Cursor dabei ebenso nicht genauso weit vom Benutzer entfernt ist wie das Objekt das betrachtet wird, fällt dies hier weniger auf.

Die Ergebnisse bei den Teilnehmern ohne Eye Tracking fielen hingegen sogar genau umgekehrt aus: Wie erwartet gefiel die Abwesenheit des Cursors der großen Mehrheit der Teilnehmer am wenigsten, dicht gefolgt vom Laserstrahl. Am besten schnitt das Fadenkreuz ab, direkt mit einer Stimme weniger der Ring-Cursor. Aufgrund der fehlenden Visualisierung des Interaktionspunktes

beim Ring-Cursor schnitt dieser nicht ganz so gut ab, die Teilnehmer hatten große Schwierigkeiten gerade mit kleineren Elemente zu interagieren. Dies teilten manche ebenfalls in den Textfeldern des Fragebogens mit. Bei der Verrichtung der Aufgaben hatten die Teilnehmer mit Cursor viel Kontrolle über die Position ihres Cursors, selbst mit kleineren oder schmaleren Elementen konnte gut interagiert werden. Sehr kleine und kontrollierte Bewegungen waren möglich, dementsprechend waren diese Teilnehmer sehr schnell bei der Erledigung ihrer Aufgaben wenn besondere Präzision verlangt war.

Smoothing Interessanterweise hat die Smoothing-Funktion des Cursors die Teilnehmer nicht ganz überzeugt: Fünf von sieben präferierten den Cursor ohne Smoothing. Auch während den Beobachtungen wurde schnell klar, dass das Smoothing genau auf den Benutzer eingestellt werden muss, sonst wird der Cursor schnell als zu träge wahrgenommen, die Teilnehmer mussten oft auf den Cursor warten bevor eine Interaktion gestartet werden konnte. Obwohl die Position des nachziehenden Cursors nicht ausschlaggebend für die Interaktion selbst war und es nicht notwendig war auf ihn zu warten, taten es die Teilnehmer dennoch. Keiner der Teilnehmer versuchte mit einem Element der Szene zu interagieren wenn nicht der Cursor darauf war. Die Teilnehmer sind vermutlich von der jahrelangen Benutzung von Computern und dazugehörigen Computermäusen gewohnt, dass eine Interaktion erst möglich wird, wenn der Mauscursor sich an der Position des Interaktionselements befindet. Die Smoothing-Funktion muss sehr vorsichtig verwendet werden, lieber bewegt sich der Cursor etwas zu schnell mit den Augen mit als etwas zu langsam, denn der Cursor selbst ist, wie oben bereits festgestellt wurde, eine Ablenkung für den Benutzer; jegliche auffälligen Bewegungen des Cursors lenken ihn nur noch weiter von seinem eigentlichen Ziel ab.

Zusammenfassend kann man sagen, dass die Kombination aus VR und Eye Tracking bei den Teilnehmern der Studie gut ankam. Die meisten der Teilnehmer hatten keine oder nur sehr wenig Erfahrung mit VR, vor allem im Bereich der Interaktion. Dennoch kamen sie relativ schnell mit den Funktionen und Aufgaben zurecht. Während der Studie gab es einige Teilnehmer der Gruppe mit Eye Tracking, welche im Laufe der Zeit über inkorrektes Tracking ihrer Augen klagten: Der Cursor befand sich nicht mehr dort wo ihr Fokuspunkt war, sondern leicht versetzt davon. Dies rührt vermutlich durch eine Veränderung des Sitzes der VR Brille während des Tragens, da sie etwas schwer in das Gesicht hängt und nach vorne rutschen kann. Ebenso kann bei Benutzern ohne VR Erfahrung die Brille nicht fest genug am Kopf befestigt sein, was ebenfalls zu einem Verrutschen der Brille führen kann. Da die Kalibrierung für das Eye Tracking am Anfang stattfindet, nimmt dies einen erheblichen Einfluss auf die Benutzung. Oftmals musste also der Sitz der Brille während der Studie überprüft und gegebenenfalls korrigiert werden, und stets auch eine erneute Kalibrierung durchgeführt werden. Bei manchen Teilnehmern kam dies sogar mehrmals während ihrer Sitzung vor. Die Brille wurde zudem von manchen Teilnehmern auf Dauer als zu schwer empfunden, Abdrücke der Brille im Gesicht fand man nach den einzelnen Sitzungen bei allen Teilnehmern.

9 Mögliche Verbesserungen und Erweiterungen

Auf der Basis der Benutzerstudie werden in diesem Abschnitt einige mögliche Verbesserungen der getesteten Konzepte, sowie potenzielle Erweiterungen der Anwendung vorgestellt und diskutiert.

9.1 Cursor

Die Wahl des Cursors hatte einen großen Einfluss auf die Benutzerfreundlichkeit der Anwendung. Aufgrund der guten Resonanz bei Abwesenheit eines Cursors wäre es sinnvoll dem Benutzer in den Menü-Einstellungen die Möglichkeit zu geben, das Cursor-Design selbst zu wählen oder den visuellen Cursor ganz auszuschalten. Ebenso wäre es denkbar, das Smoothing als Option anzubieten, mit vom Benutzer selbst einstellbaren Grad des 'Nachziehens' des Cursors.

9.2 Menü

Der Aufruf des Hauptmenüs könnte beispielsweise über einen visuellen Button geschehen, welcher nur dann im Bild erscheint, wenn der Benutzer seinen Blick an den Rand seines Sichtfeldes bewegt. Somit wäre der Button nicht permanent im Sichtfeld des Benutzers, sondern nur dann wenn er das Hauptmenü auch wirklich aufrufen möchte. Erweitern könnte man dies auch, indem bestimmte wichtige, oft verwendete Optionen des Hauptmenüs neben dem Button erscheinen, welcher für den Aufruf des Hauptmenüs zuständig ist.

Die Gestaltung des Hauptmenüs selbst lässt noch viele weitere Optionen offen: Es wäre denkbar, dass das Hauptmenü selbst nicht mithilfe von zweidimensionalen Buttons dargestellt wird, sondern durch die Verwendung von räumlichen Objekten die Dreidimensionalität des virtuellen Raums ausnutzt. Beispielsweise könnten Objekte wie Würfel oder Sphären für das Aufrufen von Untermenüs benutzt werden. Um ein Objekt zu laden könnten kleine Versionen davon, anstatt nur Bilder von den Objekten, in der Objektauswahl platziert werden.

Dasselbe gilt auch für das Objektmenü: Das Design besteht aus zweidimensionalen Buttons. Auch hier wäre es denkbar, die Dreidimensionalität besser zu nutzen und neue Konzepte für die Darstellung zu entwickeln. Die Position des Objektmenüs ist noch sehr statisch, denkbar wäre es, diese so zu gestalten, dass dieses sich dynamisch mit dem Objekt und dem Benutzer mitbewegt, sodass es für den Benutzer stets greifbar bleibt. Gerade wenn der Benutzer dazu in der Lage ist sich im virtuellen Raum zu bewegen wird dies unerlässlich sein.

9.3 Transformations-Tool

Einige Teilnehmer bemängelten die Tatsache, dass die Teile des Transformations-Tools, welche sich auf der Rückseite des Objekts befanden, sichtbar waren, aber nicht für die Interaktion zur Verfügung standen. Um ungewolltes Verhalten bei beispielsweise der Skalierung zu vermeiden ist es ratsam, solche Interaktionselemente wie die weißen Sphären, welche sich auf der Rückseite des Objekts befinden, für den Benutzer nicht sichtbar zu machen.

Rotation Die Klick-Rotation stellte die Teilnehmer mit eingeschaltetem Eye Tracking vor eine Herausforderung. Es fehlte die Präzision bei der Selektion der farbigen Ringe. Ebenso war der Blick des Nutzers für die Rotation selbst notwendig, hinderte aber den Benutzer daran den Winkel der beiden Linien zu betrachten. Es wäre denkbar, die Interaktionsfläche der Ringe zu vergrößern, um die Interaktion zu erleichtern. Zusätzlich könnte ein separater physischer Button hinzugefügt werden, welcher den Blick des Benutzers einfriert - das heißt, dass der Blick des Benutzers in dieser Zeit nicht die Position der zweiten Linie beeinflusst. Somit könnte der Rotationswinkel betrachtet werden.

Skalierung Wenn der Benutzer während des Skalierungsprozesses dieses oder ein anderes Objekt betrachten möchte, verändert dies die aktuelle Größe des zu skalierenden Objekts. Möchte der Benutzer beispielsweise zwei Objekte auf eine ähnliche Größe bringen, ist dies nur in mehreren separaten Schritten möglich. Um dem Benutzer die Möglichkeit zu geben sich umzusehen, könnte man eine Funktion hinzufügen mit welcher der Benutzer den Skalierungsprozess ëinfrieren"könnte, das heißt das Objekt wird nicht weiter skaliert, sondern verbleibt in der Größe zur Zeit des Einfrierens.

Eine Alternative dazu wäre, den Skalierungsprozess an sich vom Blick des Benutzers zu entkoppeln, sodass ausschließlich die Aktivierung mit den Augen durchgeführt wird. Die Skalierung könnte dann mit einem physischen Tastendruck auf der Tastatur gesteuert werden.

Translation Das aktuelle Konzept für die Translation könnte dahingehend modifiziert werden, dass der Fokuspunkt des Benutzers auf die am nächsten liegende Translationslinie projiziert wird. Somit müsste der Benutzer nicht mehr exakt einen Punkt auf den Translationslinien betrachten um die Translation aktivieren zu können.

Alternativ dazu könnte bei gleichzeitigem Blick auf eine der farbigen Pfeile des Tools und Tastendruck zwei Buttons im Sichtfeld des Benutzers erscheinen, mit deren Hilfe der Benutzer das Objekt auf der jeweiligen Achse bewegen könnte. Diese zwei Buttons wären dann für die beiden Richtungen auf der jeweiligen Achse zuständig. Die eigentliche Bewegung des Objekts könnte dabei durch einmaligem Blick auf einen der Buttons und dauerhaft gedrückter Taste erfolgen, oder aber etappenweise mit einzelnem Tastendruck. Bei der Variante mit gedrückt gehaltener Taste könnte der Blick des Benutzers nur für die Aktivierung der Bewegung des Objekts notwendig sein, während die Taste gedrückt wird wäre der Benutzer also in der Lage sich frei umzusehen. Würden separate, visuelle Buttons für die Translation verwendet, wäre der Benutzer vollständig unabhängig davon wo

sich das Objekt im virtuellen Raum befindet, und könnte das Objekt stets in die gewünschte Richtung bewegen. Kleinere Distanzen könnten so ebenfalls zurückgelegt werden. Der Nachteil jedoch ist, dass der Vorgang deutlich mehr Zeit in Anspruch nimmt als bei dem entwickelten Konzept.

10 Zusammenfassung und Ausblick

In dieser Arbeit wurden die typischen Augenbewegungen des Menschen erläutert, sowie daraus resultierende Probleme bei der Entwicklung von Konzepten für die Interaktion mithilfe von Eye Tracking Input vorgestellt. Das FOVE VR HMD mit integriertem Eye Tracking wurde vorgestellt, und zusammen mit der Unity Engine für die Entwicklung eines Prototyps verwendet. Der Prototyp bietet, neben einem allgemeinen Benutzerinterface und einem kachelförmigen Hauptmenü, auch verschiedene Techniken zur Objektmanipulation für die Untersuchung eines dreidimensionalen Objekts in einer virtuellen Umgebung. Die für den Prototypen entwickelten Konzepte wurden mithilfe einer Benutzerstudie evaluiert und auf ihre Benutzerfreundlichkeit getestet. Dafür erledigten die Teilnehmer der Studie an sie gestellte Aufgaben und füllten anschließend einen Fragebogen dazu aus. Diese Studie ergab Einsicht in die Stärken und Schwächen der einzelnen Konzepte und hilft bei der Verbesserung der Anwendung, beispielsweise beim Design von Konzepten für komplexere Aufgaben in VR als die reine Selektion von virtuellen Buttons. Die Konzepte der Rotation kamen bei den Teilnehmern sehr gut an, gerade die Freie Rotation fand viel Anklang. Bei der Translation von Objekten gibt es noch einen großen Handlungsbedarf. Das hierfür entwickelte Konzept zeigte gewisse Schwächen, kleinere Distanzen konnten nicht zurückgelegt werden. Indem andere Ansätze untersucht werden, wie beispielsweise das Hinzufügen von virtuellen Buttons zur Translation des Objektes, könnten solche Probleme überwunden werden. Überraschenderweise zeigte sich, dass ein visueller Cursor für Interaktionen in VR mithilfe von Eye Tracking im Allgemeinen nicht notwendig ist und eher als störend empfunden wird, es sei denn die verwendete Technologie arbeitet nicht präzise genug oder die Interaktionselemente sind zu klein. Allgemein zeigen die Ergebnisse der Studie, dass Eye Tracking als eine Form des Inputs für Anwendungen eine hilfreiche Option darstellt.

Ausblick

Während des Designprozesses stellte sich heraus, dass die Kombination aus Eye Tracking und VR viele Möglichkeiten für die Forschung gibt, es allerdings noch sehr viel Raum für Weiterentwicklungen gibt. Gerade für den Bereich Immersive Analytics haben VR und AR Anwendungen ein großes Potenzial, das weiter erforscht werden muss. Nimmt man Eye Tracking als Eingabeform hinzu, ergeben sich Interaktionsformen, welche wesentlich angenehmer für den Benutzer sind und eine längere Nutzungsdauer des Systems ermöglichen. Die Manipulation von Daten zur Analyse findet in vielen Bereichen statt, und weitere Ansätze zur leichteren Interaktion via Eye Tracking müssen entwickelt werden. Neben des in dieser Arbeit verwendeten FOVE VR HMD existieren noch andere VR Geräte, welche mit Eye Tracking aufgerüstet werden können: Für die HTC Vive Pro gibt es zusätzlich erwerbbare Eye Tracking Hardware, welche selbst das Tragen einer Brille mit Sehstärke ermöglicht. Dennoch ist das FOVE VR HMD im Vergleich dazu die günstigste Kombination aus VR

Brille und Eye Tracking auf dem Markt. Da die für solche Anwendungen benötigte Hardware immer erschwinglicher wird ist abzusehen, dass zukünftig mehr und mehr Menschen in verschiedenen Bereichen beginnen werden mit diesen Geräten in den verschiedensten Bereichen zu arbeiten.

Literaturverzeichnis

[BKDA82] M. B. Friedman, G. Kiliany, M. Dzmura, D. Anderson. „EYETRACKER COMMUNICATION SYSTEM.“ In: *Johns Hopkins APL Technical Digest (Applied Physics Laboratory)* (Juli 1982), S. 250–252 (zitiert auf S. 11).

[Gol18] W. Goldstone. *Get started with VR! Sample pack & Learning Articles*. 14. Dez. 2018. URL: https://blogs.unity3d.com/2015/12/09/get-started-with-vr-sample-pack-learning-articles/ (zitiert auf S. 23, 39).

[I S07] K. T.-I. I. Scott Mackenzie. *Text Entry Systems: Mobility, Accessibility, Universality*. MORGAN KAUFMANN PUBL INC, 11. März 2007. 332 S. ISBN: 0123735912. URL: https://www.ebook.de/de/product/6369482/i_scott_mackenzie_kumiko_tanaka_ishii_text_entry_systems_mobility_accessibility_universality.html (zitiert auf S. 21).

[Jac91] R. J. K. Jacob. „The use of eye movements in human-computer interaction techniques: what you look at is what you get“. In: *ACM Transactions on Information Systems* 9.2 (Apr. 1991), S. 152–169. DOI: 10.1145/123078.128728 (zitiert auf S. 17–19).

[Jac93] R. J. K. Jacob. „Eye movement-based human-computer interaction techniques: toward non-command interfaces“. In: 1993 (zitiert auf S. 11, 18, 21, 22).

[Ohn98] T. Ohno. „Features of eye gaze interface for selection tasks“. In: *Proceedings. 3rd Asia Pacific Computer Human Interaction (Cat. No.98EX110)*. IEEE Comput. Soc, 1998. DOI: 10.1109/apchi.1998.704190 (zitiert auf S. 17, 20).

[PLW13] A. M. Penkar, C. Lutteroth, G. Weber. „Eyes Only: Navigating Hypertext with Gaze“. In: *Human-Computer Interaction – INTERACT 2013*. Springer Berlin Heidelberg, 2013, S. 153–169. DOI: 10.1007/978-3-642-40480-1_10 (zitiert auf S. 19).

[Sha02] B. D. Shaviv. „The design and improvement of an eye-controlled interface“. In: 2002 (zitiert auf S. 20).

[SR95] D. M. Stampe, E. M. Reingold. „Selection By Looking: A Novel Computer Interface And Its Application To Psychological Research“. In: *Studies in Visual Information Processing*. Elsevier, 1995, S. 467–478. DOI: 10.1016/s0926-907x(05)80039-x (zitiert auf S. 21).

[TJ00] V. Tanriverdi, R. J. K. Jacob. „Interacting with eye movements in virtual environments“. In: *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '00*. ACM Press, 2000. DOI: 10.1145/332040.332443 (zitiert auf S. 11).

[WM02] D. J. Ward, D. J. C. MacKay. „Fast hands-free writing by gaze direction“. In: *Nature* 418.6900 (Aug. 2002), S. 838–838. DOI: 10.1038/418838a (zitiert auf S. 21).

[WM87] C. Ware, H. H. Mikaelian. „An evaluation of an eye tracker as a device for computer input2“. In: *Proceedings of the SIGCHI/GI conference on Human factors in computing systems and graphics interface - CHI '87*. ACM Press, 1987. DOI: 10.1145/29933.275627 (zitiert auf S. 11, 18, 20, 21).

Alle URLs wurden zuletzt am 14. 12. 2018 geprüft.

Erklärung

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

Ort, Datum, Unterschrift