

# Fernunterstützung und Zusammenarbeit mit 3D Punktwolken

Masterarbeit  
von

**Kai Westerkamp**

An der Fakultät für Informatik  
Institut für Anthropomatik und Robotik  
Fraunhofer IOSB (IAD)

Erstgutachter:	Prof. Dr.-Ing. Rainer Stiefelhagen
Zweitgutachter:	Prof. Dr.-Ing. habil. Jürgen Beyerer
Betreuender Mitarbeiter:	M.Sc. Adrian Hoppe
Zweiter betreuender Mitarbeiter:	M.Sc. Sebastian Maier

Bearbeitungszeit: 01.06.2017 – 30.11.2017



# Zusammenfassung

Mit der wachsenden Komplexität vieler Systeme wird es immer schwieriger, diese zu warten und zu reparieren. Immer häufiger ist es nötig, auf die Unterstützung eines spezialisierten Experten zurückzugreifen. Diese Experten sind jedoch meist nicht vor Ort und müssen entweder anreisen oder Fernunterstützung bieten. Videostreams, Anrufe oder Bilder können hierbei verwendet werden, sind aber nicht immer ausreichend.

Die zunehmende Verbreitung von kostengünstigen immersiven Virtual Reality (VR) und Augmented Reality (AR) Systemen bietet die Möglichkeit, alternative Systeme zur Fernunterstützung zu entwickeln. VR ermöglicht das verlustfreie Anzeigen von 3D-Inhalten und führt so zu deren besserem Verständnis. AR ermöglicht es, die reale Welt mit Projektionen zu erweitern. Diese Projektionen eignen sich zum Beispiel dazu, Hilfestellungen bei einer Montage oder Reparatur zu visualisieren.

In dieser Masterarbeit wird ein System vorgestellt, bei dem ein lokaler Benutzer in einer AR-Umgebung mit einem Hardwareproblem Unterstützung von einem entfernten Experten in einer VR-Umgebung bekommt. Hierzu wird zunächst ein System vorgestellt, mit dem der lokale Nutzer einen 3D-Punktwolken-Scan des Objekts anfertigen kann. Diese Daten werden beim Experten in einer VR-Umgebung visualisiert und der Experte kann sich das Objekt anschauen. Anschließend wird ein System vorgestellt, bei dem der Experte in VR und der lokale Nutzer in AR zusammen arbeiten können. Der Experte kann mit einer Zeigegeste in Form eines Beams auf das gescannte Objekt zeigen und der lokale Nutzer bekommt diesen Beam am echten Objekt in seine Welt projiziert. Im Referenzszenario standen dem Experten und dem lokalen Nutzer ein Videostream zur Kommunikation zur Verfügung.

Die Nutzerstudie zeigt, dass die Visualisierung der Punktwolke im Durchschnitt ein schnelleres Verständnis der Daten ermöglicht, obwohl die Probanden die Interaktion mit einer Punktwolke schwieriger finden als den Umgang mit 2D-Bildern. Auch bei der Kommunikation zwischen dem Experten und dem lokalen Nutzer war mit dem vorgestellten System eine Verbesserung zu erkennen. Die Kommunikation war weniger fehleranfällig und schneller als bei dem Videoszenario. Das vorgestellte VR/AR-Szenario weist außerdem eine bessere Nutzerzufriedenheit auf. Insbesondere der lokale Nutzer mit der VR-Brille bewertet die eigene Leistung besser und gibt dabei an, weniger frustriert zu sein.





# Inhaltsverzeichnis

<b>1. Einleitung</b>	<b>1</b>
1.1. Motivation und Ziel der Arbeit . . . . .	1
1.2. Aufbau der Arbeit . . . . .	2
<b>2. Grundlagen</b>	<b>3</b>
2.1. Virtuelle Realität . . . . .	3
2.2. Augmented Reality (AR) . . . . .	4
2.3. 3D-Punktwolken-Scan . . . . .	5
2.4. AR-Fernunterstützung . . . . .	6
2.5. Verwendete Technologien . . . . .	6
2.5.1. HTC Vive und Lighthouse Tracking . . . . .	6
2.5.2. HoloLens . . . . .	7
2.5.3. Unreal Engine 4 . . . . .	8
2.5.4. Unity . . . . .	8
2.5.5. Kinect . . . . .	8
<b>3. 3D Aufnahmen mit Punktwolken</b>	<b>11</b>
3.1. Frames aufnehmen und bereinigen . . . . .	12
3.1.1. Aufnahme und Glättung . . . . .	12
3.2. Zusammenfügen von Frames . . . . .	13
3.2.1. Kalibrierung Kinect zu Vive . . . . .	13
3.3. Ergebnisse . . . . .	15
<b>4. Speichern der Punktwolke mit 3D Tiles</b>	<b>17</b>
4.1. 3D Tiles . . . . .	17
4.1.1. glTF . . . . .	17
4.1.2. Tileset und Tiles . . . . .	18
4.2. Implementierung der 3D Tiles . . . . .	20
<b>5. Visualisierung</b>	<b>21</b>
5.1. UE4 Rendering System . . . . .	21
5.2. 3D Tiles laden und vorbereiten . . . . .	21
5.3. Rendering . . . . .	22
5.4. Ergebnisse . . . . .	23
<b>6. HoloLens</b>	<b>25</b>
6.1. Unreal Engine 4 und HoloLens . . . . .	25
6.2. HoloLens Implementierung . . . . .	25
6.3. Kalibrierung . . . . .	25
6.3.1. Kalibrierungsfehler . . . . .	26
6.4. Ergebnisse . . . . .	27

<b>7. Evaluation</b>	<b>29</b>
7.1. Versuchsaufbau . . . . .	29
7.2. Versuchsablauf . . . . .	30
7.3. Statistische Verfahren . . . . .	31
7.4. Probanden und Teams . . . . .	32
7.5. Ergebnisse . . . . .	33
7.5.1. Vorbereitung des Experten . . . . .	33
7.5.2. Kommunikation des gesuchten Steins . . . . .	33
7.5.3. Fehleranzahl . . . . .	37
7.5.3.1. Timings . . . . .	37
7.5.4. NASA TLX und UEQ . . . . .	39
7.5.5. Unabhängigkeit . . . . .	43
7.5.6. Zusammenfassung . . . . .	43
<b>8. Fazit und Ausblick</b>	<b>45</b>
A. 3D Tile JSON . . . . .	47
B. Evaluations Fragebögen . . . . .	48
B.1. Anfangsfragebogen . . . . .	48
B.2. Fragebogen nach jedem Szenario . . . . .	49
B.3. Abschlussfragebogen . . . . .	53
C. Tabellen . . . . .	54
D. Alternative Plots . . . . .	60
E. Datenträger . . . . .	63
<b>Literaturverzeichnis</b>	<b>65</b>

# 1. Einleitung

## 1.1. Motivation und Ziel der Arbeit

Mit der wachsenden Komplexität vieler Systeme wird es immer schwieriger, diese zu warten und zu reparieren. In vielen Fällen verfügt ein lokaler Benutzer oder Technikern nicht mehr über das nötige Fachwissen, um ein Problem mit einer Hardware zu lösen und zu beheben. Es muss ein spezialisierter Experte hinzugezogen werden, der bei der Lösung des Problems helfen muss. Experten sind meist nicht vor Ort und müssen entweder anreisen oder remote hinzu geschaltet werden. Eine Anreise ist zeitaufwändig, teuer und in vielen Fällen deshalb unpraktikabel. Das Zuschalten eines Experten ist zum Beispiel durch eine Videokonferenz oder telefonisch mit versendeten Fotos möglich. Jedoch ist der Experte hierbei an die Perspektive aus dem Stream und den Fotos gebunden. Als Kommunikationsmittel stehen hierbei häufig nur die Sprache und Einzeichnungen in die Bilder zur Verfügung.

In den letzten Jahren wurden erschwingliche Systeme für immersive Virtual Reality (VR) vorgestellt. Head Mounted Displays (HMDs) wie die HTC Vive und Oculus Rift bieten erschwingliche und platzsparende Lösungen, einen Benutzer in hochqualitative virtuelle Realität eintauchen zu lassen. Mit der HoloLens hat Microsoft eine Augmented Reality (AR) Brille vorgestellt, mit der es möglich ist, interaktive 3D-Projektionen in der direkten Umgebung darzustellen.

Die Anwendungsfelder dieser Technologien sind vielfältig. VR kann zum Beispiel eingesetzt werden, um 3D-Daten zu visualisieren und zu analysieren. In der Automobilindustrie kann so ein virtueller Prototyp kostengünstig untersucht werden. Auch die Telepräsenz ist ein mögliches Einsatzgebiet von VR-Technologien. In einer VR-Umgebung können mehrere Nutzer aus verschiedenen, entfernten Orten in einer gemeinsamen Welt zusammengebracht werden. AR-Technologien bieten die Möglichkeit, zusätzliche Informationen in die echte Welt zu projizieren. Es können klassische Inhalte wie Webseiten, Videos und Bilder überall platziert und abgespielt werden. So können Informationen wie Anleitungen und Beschriftungen von Maschinen direkt an der richtigen Stelle eingeblendet werden. Ein weiterer Anwendungsfall für AR-Technologien ist das Einblenden von 3D-Hologrammen, die die Echtwelt ergänzen. So kann zum Beispiel ein Bereich durch eine Markierung hervorgehoben werden oder ein Pfeil direkt auf interessante Stellen weisen.

Ziel dieser Arbeit ist, das Zusammenarbeiten eines lokalen Nutzers mit einer AR-Brille und eines entfernten Experten in VR zu ermöglichen. Der lokale Nutzer hat ein Hardwareproblem, bei dem Expertenwissen benötigt wird. Der Experte soll in einer VR-Umgebung

einen 3D-Scan der Hardware sehen und daran seine Analyse vornehmen. Anschließend soll der Experte mit dem lokalen Benutzer interagieren. An der virtuellen Repräsentation kann der Experte Hilfestellungen geben, die dem lokalen Benutzer mit Hilfe von Projektionen in die Echtwelt visualisiert werden.

In dieser Arbeit wird ein Lösungsansatz für das Aufnehmen einer 3D Punktwolke zur Visualisierung in VR vorgestellt. Anschließend wird ein Lösungsansatz für die Zusammenarbeit des Experten und des lokalen Benutzers präsentiert und evaluiert. In einem Szenario wird untersucht, ob die Zusammenarbeit in VR und AR einen Vorteil gegenüber einem Videoanruf bietet. Hierzu stehen dem Experten die gescannte Punktwolke und eine Zeigegeste in Form eines Laserstrahls zur Verfügung. Zunächst wird evaluiert, ob die 3D Repräsentation mit einer Punktwolke Vorteile oder Nachteile gegenüber 2D-Bildern bietet. Hierzu bekommt der Experte eine Aufgabe, bei der er mit der Punktwolke, bzw. den Bildern etwas in dem Objekt erkennen muss. Des Weiteren wird evaluiert, ob das vorgestellte Interaktionskonzept zwischen dem Experten und dem lokalen Nutzer Vorteile oder Nachteile bietet.

## **1.2. Aufbau der Arbeit**

Zunächst werden im Kapitel 2 theoretische und technisch Grundlagen betrachtet, die für die Entwicklung des Systems benötigt werden. In Kapitel 3 wird eine Aufnahmetechnik von 3D-Punktwolken von Objekten vorgestellt. In Kapitel 4 stellen wir ein Datenformat zur effizienten und kompakten Speicherung der Punktwolken vor und in Kapitel 5 wird über die Visualisierung dieser Punktwolken in einer VR-Umgebung gesprochen. Das Kapitel 6 befasst sich mit der Synchronisation der VR-Umgebung mit der AR-Umgebung und Echtwelt. In Kapitel 7 wird der Aufbau und die Durchführung der Nutzerstudie beschrieben. Anschließend erfolgt die Evaluation der Ergebnisse. Kapitel enthält eine Zusammenfassung der Ergebnisse der Arbeit und einen Ausblick auf zukünftige Erweiterungen und Verbesserungen.

## 2. Grundlagen

### 2.1. Virtuelle Realität

Geprägt wurde der Begriff Virtual Reality (VR) von dem Autor Bamien Broderik in seinem 1982 erschienenen Science-Fiction-Roman *The Judas Mandala* [Bro82]. Eine einheitliche Definition des Begriffs gibt es nicht. Milgram et al. [MTUK95] definieren eine VR-Umgebung als eine Umgebung, in der ein Nutzer komplett in eine künstliche Welt abtaucht. Diese künstliche Welt kann versuchen, die Realität zu imitieren, aber kann auch die Grenzen der Realität überschreiten. Physikalische Gesetze wie Zeit, Gravitation und Materialeigenschaften können verändert und manipuliert werden. Brooks [Bro99] definiert eine VR-Erfahrung als eine, in der der Benutzer effektiv in eine reagierende virtuelle Welt eintaucht. Das impliziert eine dynamische Steuerung des Sichtfeldes durch den Nutzer.

Heim [Hei00] definiert Virtual Reality über die drei “I”s: Interaktion, Immersion und Informationsintensität. Die Immersion beschreibt, wie stark ein Nutzer in eine virtuelle Welt eintaucht. Immersion wird laut Heim durch ein Gerät, das alle Sinne ausreichend abschirmt, erreicht, sodass die Person sich fühlt, als wäre sie an einen andern Ort transportiert worden. Interaktion beschreibt nach Heim die Fähigkeit eines Computers, die Perspektive der Szene so schnell anzupassen wie ein Nutzer seine physikalische Position und Perspektive anpassen kann. Interaktion ist darüber hinaus die Fähigkeit, Nutzereingaben zu erkennen und die virtuelle Welt entsprechend in Echtzeit zu ändern [NB]. Die Informationsintensität ist der Begriff, dass eine virtuelle Welt spezielle Qualitäten, wie Telepräsenz und künstliche Intelligenz, mit sich bringen kann. Durch konstantes updaten der Informationen wird die Immersion und Interaktion des Systems unterstützt. [Hei00].

VR ermöglicht es, Inhalte in 3D darzustellen. Dies kann zu einem besseren Verständnis der Daten durch Wahrnehmungsphänomene wie Kopfbewegungsparallaxe, kinetische Tiefenwirkung und Stereopsis führen [Min95]. So kann ein VR-System zum Beispiel zum Training von Personen, die Blick-Inspektionen an Flugzeugen durchführen sollen, genutzt werden. Durch die VR-Umgebung kann auf ein echtes Flugzeug verzichtet werden. Dieses System wurde von Vora et al. untersucht [VNG<sup>+</sup>02]. Auch Design und Produktionskosten können durch VR gespart werden. Zum Beispiel kann VR als Werkzeug zur Überprüfung eines Produktionsprozesses verwendet werden [dSZ99]. Hierbei werden Prototypen in VR visualisiert und müssen nicht mehr gefertigt werden.

Als Schlüsseltechnologien für VR nennt Brooks [Bro99] optische, akustische und haptische Displays, die den Nutzer in die virtuelle Welt eintauchen lassen und die echte Welt ausblenden. Hierfür wird ein grafisches Rendering System benötigt und ein Tracking-System,

das die aktuelle Position und Orientierung des Kopfes und der Extremitäten des Nutzers bestimmt. Zusätzlich wird ein System benötigt, das die virtuelle Welt aufbaut und simuliert. Brooks nennt zwei Displaytechnologien, die sich für die Darstellung von virtuellen Welten eignen. Head Mounted Displays (HMD) sind Systeme, bei denen sich Bildschirme zur Darstellung der VR-Welt direkt vor den Augen des Betrachters befinden und auf dem Kopf getragen werden. CAVE-artige Systeme haben Projektionsflächen um den Nutzer herum, auf denen die virtuelle Welt dargestellt wird. Bei aktuellen kommerziellen VR-Systemen werden fast ausschließlich HMDs verwendet. Diese sind mobiler und günstiger. CAVE-artige Systeme sind häufig bei Flug- und Fahrsimulatoren im Einsatz.

Aktuelle kommerzielle HMDs lassen sich in zwei Gruppen einteilen. Die günstigen Systeme sind meistens Halterungen, in die ein aktuelles Smartphone eingelegt werden kann (Google Daydream [day], Samsung Gear VR [sam], etc.). Auf dem Display wird nebeneinander für jedes Auge ein Bild dargestellt und mit speziellen Linsen so verzerrt, dass ein VR-Eindruck entsteht. Durch Sensoren in den Smartphones kann die Kopfbewegung des Nutzers registriert und der entsprechende Ausschnitt der VR-Welt gezeigt werden. Interaktionsmöglichkeiten sind der längere Blickkontakt auf ein Objekt oder extra dafür vorgesehene Knöpfe an den Halterungen. Bei Samsung Gear VR kann zusätzlich ein Controller benutzt werden. Die zweite Gruppe der HMDs sind leistungsstarke Systeme. Hierbei ist das Display fest in dem Headset montiert. Durch bessere Trackingsysteme wird meist zusätzlich ermöglicht, dass sich der Nutzer nicht nur umschaun kann, sondern auch dass die Position des Kopfes getrackt wird. Damit steht der Benutzer nicht mehr fest an einem Punkt, sondern kann sich frei im Raum bewegen und diese Bewegung wird in die virtuelle Welt übertragen. Das Simulieren der Welt übernimmt hierbei ein stationärer, leistungsstarker Computer oder eine Konsole, die per Kabel mit dem HMD verbunden sind. Die Oculus Rift [occ] und die Playstation VR [Pla] nutzen für das Tracking ein Kamerasystem. Hierzu werden mindesten zwei Kameras genutzt, um optische Marker an den HMDS zu registrieren. Aus den Aufnahmen der Kameras lässt sich die Kopfposition und -orientierung errechnen. Oculus und Playstation VR unterstützen das Benutzen von optisch getrackten Controllern. Diese ermöglichen das Tracken der Hände, in denen sie gehalten werden, und bieten zusätzliche Tasten zur Interaktion mit der virtuellen Welt. Die HTC Vive[viv] nutzt das dafür entwickelte Lighthouse Tracking System. Durch zwei passive, Laser emittierende, fest montierte Basisstationen können die Geräte ihre Position und Rotation im Raum errechnen. Es wird ebenfalls ein getrackter Controller pro Hand verwendet. Die Vive und das Lighthouse Tracking System werden im Kapitel 2.5.1 genauer vorgestellt. Ein Nachteil von HMDs ist der Tragekomfort: Die Displays sind schwer und durch die Abschirmung der Außenwelt wird es unter den Brillen schnell sehr warm.

Ein generelles Problem mit VR-Technologien ist die Motion Sickness. Die Symptome ähneln einer Seekrankheit, wie Schwindel und Übelkeit. Effektstärke und Auftreten sind hierbei stark von dem jeweiligen Nutzer und der VR-Welt abhängig. Deshalb sollten einige Richtlinien beim Entwickeln eines VR-Systems eingehalten werden [vrB]. Richtlinien sind zum Beispiel das Aufrechterhalten einer hohen Framerate (min. 60 FPS), keine Kameras die vom virtuellen Körper losgelöst werden und keine unnatürlichen Kamerabewegungen, wie ein Schütteln der Kamera bei Explosionen oder beim jedem Schritt des virtuellen Laufens.

## 2.2. Augmented Reality (AR)

Nach Milgram et al. [MTUK95] sind Augmented Reality (erweiterte Realität) und Virtual Reality unterschiedliche Enden des des Reality-Virtuality-Kontinuums. Auf der einen Seite des Kontinuums ist eine Umgebung, die nur aus echten Objekten besteht. Diese Welt kann entweder direkt von einer Person betrachtet werden oder durch ein Fenster in die



Abbildung 2.1.: Vereinfachte Darstellung des Reality-Virtuality Kontinuums nach Milgram et al. [MTUK95]

Welt, wie ein Videodisplay. Auf der anderen Seite ist eine virtuelle Umgebung, die nur aus virtuellen Objekten besteht. Den dazwischenliegenden Bereich stellt einen fließenden Übergang dar, der Mixed Reality genannt wird. In einer Mixed Reality sind sowohl echte als auch virtuelle Objekte auf einem gemeinsamen Display zu erkennen. Augmented Reality ist die Integration von digitalen Informationen in die Umgebung des Nutzers in Echtzeit [aug].

Displays für Augmented Reality lassen sich in zwei Kategorien unterteilen. Bei See-Trough-AR-Displays kann der Betrachter durch den Bildschirm hindurchschauen und damit wird die bestmögliche Visualisierung der Echtwelt erreicht. Mit den Bildschirmen lässt sich diese Realität erweitern. Dieses Prinzip verwendet die Microsoft HoloLens. Der Nutzer trägt ein HMD, durch das er hindurchsehen kann und das die Welt durch Hologramme erweitert.

Monitorbasierte AR-Bildschirme basieren auf dem “Fenster zu der Welt” Prinzip. Hierbei werden computergenerierte Bilder in einen Videostream integriert und so die Realität erweitert. Diese Technologie wird zum Beispiel bei dem Spiel Pokémon GO verwendet [Pok]. Die App nimmt mit der Frontkamera eines Smartphones die Umgebung auf und projiziert ein Pokémon, das es zu fangen gilt, in diese Augmented Reality.

## 2.3. 3D-Punktwolken-Scan

Aus einem Tiefenbild wie das der Kinect können Punktwolken aus einer Perspektive berechnet werden. Für größere Aufnahmen werden aber häufig mehrere Punktwolken aufgenommen, die anschließend vereint werden müssen. Diesen Vorgang nennt man Registration. Hierbei muss für die verschiedenen Ansichten der getrennt aufgenommenen Punktwolken eine relative Position und Orientierung gefunden werden, sodass überlappende Teile perfekt übereinander liegen. Einer der beliebtesten Algorithmen für die Registration ist Iterative Closest Point (ICP) [BM92]. ICP versucht die optimale Transformation zwischen zwei Datensätzen zu finden, indem eine Fehlermetrik zwischen den Punktwolken minimiert wird. Hierbei wird versucht für jeden Punkt aus der einen Punktwolke der jeweils nächste Punkt aus der anderen Punktwolke bestimmt. Dabei wird die Annahme getroffen, dass jeder Punkt einen zugehörigen Punkt in der anderen Punktwolke hat. Da Punktwolken sich häufig nur teilweise überlappen, wurden viele Versuche unternommen den Bereich einzuschränken. Eine Möglichkeit ist die Feature-Erkennung [SLW02, RBMB08], bei der zuerst ein Bereich ausgewählt wird, der in beiden Punktwolken identisch ist. Eine Alternative ist das Einbeziehen weiterer Informationen zur Regression. So können Farben [JK97] und Normalen [BL06] das Ergebnis verbessern. Jedoch sind diese Prozesse zeit- und rechenaufwändig. Für die Kinect gibt es kostenlose Tools, die versuchen, mit verschiedenen Methoden einen 3D-Scan anzufertigen. Das Programm 3D-Scan der Microsoft Corporation [3DS] funktioniert, in dem man die Kinect langsam um das Objekt herum bewegt. Es wird dabei

versucht die Kamerabewegung zu errechnen und nach abgeschlossener Aufnahme wird das 3D-Modell berechnet. Bewegt man sich zu schnell, so bricht der Scan ab und man muss erneut beginnen.

## 2.4. AR-Fernunterstützung

In der Literatur gibt es Ansätze, wie eine lokaler Nutzer, auch Techniker genannt, von einem entfernten Experten (Subject Matter Expert, SME) unterstützt werden kann. Ein Ansatz hierbei ist, dass der lokale Nutzer einen Videostream aufnimmt. Dieser wird dann vom Experten live editiert und wieder beim lokalen Nutzer angezeigt [Kuz92]. Eine Erweiterung dieses Prinzips ist von Bauer et al. [BKS99] vorgestellt worden. Hierbei wird in einem AR-HMD des lokalen Nutzers der 2D-Mauszeiger des Experten eingeblendet. Die Position des Mauszeigers ist aber schnell veraltet, wenn sich der lokale Nutzer bewegt. Jedoch sieht der Experte hierbei nur ein 2D-Abbild der Realität, das für komplexere 3D-Aufgaben nicht immer ausreicht. Chastine et al. [CNZHB08] ermöglichen es dem Experten, einen dreidimensionalen Pfeil in der lokalen Ansicht zu manipulieren. Jedoch ist das Ausrichten des Pfeils schwierig und zeitintensiv. Botteccia et al. [BCJ10] erlauben dem Experten, vorgefertigte 3D-Animationen im Sichtfeld des lokalen Nutzers zu platzieren. Dieses Vorgehen dient dazu, dem lokalen Nutzer zu demonstrieren, wie eine Aufgabe zu lösen ist, jedoch ist dieses Vorgehen wegen den vorgefertigten Animationen nicht sehr flexibel.

Tachia et al. [TAH12] nutzen statische Tiefensensoren, um dynamisch die Umgebung des Experten und des lokalen Nutzers aufzunehmen. Die 3D-Szene des lokalen Nutzers und die Hände des Experten werden zusammengefügt und beiden Seiten präsentiert. Das erlaubt dem Experten mit Handgesten Hilfestellungen zu geben.

Kurata et al. [KSK<sup>+</sup>04] haben ein System entwickelt, bei dem der lokale Nutzer eine Kamera und einen Laserpointer auf der Schulter trägt. Der Experte sieht die Umgebung durch die Kamera und kann mit dem Laserpointer einen Point of Interest in der echten Welt markieren. Lanir et al. [LSCG13] haben dieses Konzept erweitert. Auf einem beweglichen Roboterarm wurde eine Kamera und ein portabler Beamer befestigt. Der Experte kann den Roboterarm steuern und so seine Sicht anpassen und mit dem Beamer 2D-Einzeichnungen in die echte Welt projizieren.

Oda et al. [OES<sup>+</sup>15] haben ein System vorgestellt, bei dem der Experte virtuelle Replikate manipuliert. Die Objekte beim lokalen Nutzer werden durch ein optisches Trackingsystem lokalisiert. In einer VR-Umgebung werden für einen Experten zugehörige virtuelle Proxys visualisiert. Diese virtuellen Proxys sind vorgefertigte 3D-Repräsentationen der Objekte. Der Experte erklärt dem lokalen Nutzer, wie er die Objekte zusammenzubauen hat. Hierfür stehen dem Experten virtuelle Replikate zur Verfügung. Der Experte kann ein Objekt virtuell kopieren und an die richtige Position setzen. Der lokale Benutzer bekommt diese virtuelle Kopie relativ zu den Originalen in seiner AR-Umgebung visualisiert.

## 2.5. Verwendete Technologien

In diesem Unterkapitel wird die verwendete Hardware und Software vorgestellt.

### 2.5.1. HTC Vive und Lighthouse Tracking

Die HTC Vive ist im Kern ein Head-Mounted Display (HMD), das von HTC in Kooperation mit Valve seit April 2016 produziert und verkauft wird. Das System besteht aus dem HMD, zwei Controllern und zwei Basisstationen. Für den Betrieb wird zusätzlich ein leistungsstarker Computer benötigt, der das Simulieren und Rendern der virtuellen Welt



übernimmt. Das Headset wird hierfür mit Kabeln (USB, HDMI und Strom) an den Rechner angeschlossen. Die Controller kommunizieren kabellos mit dem Headset, das die Daten an den Computer weiterleitet.

Die Vive verwendet das von Valve entwickelte Lighthouse Tracking. Hierfür emittieren zwei passive Basisstationen infrarote Laserstrahlen. Diese werden zu einer Laserwand aufgefächert und mit Motoren in regelmäßigen Abständen waagrecht und senkrecht über den Raum geschwenkt. An den Controllern und dem HMD sind Sensoren angebracht, die es ermöglichen, die zeitliche Differenzen zwischen den Lichtblitzen an den verschiedenen Sensoren wahrzunehmen. Aus diesen Zeitunterschieden lässt sich die aktuell Position und Rotation der Controller und des HMDs bestimmen. Dieses Verfahren ermöglicht es, mit zwei Basisstationen beliebig viele Gegenstände zu tracken, jedoch werden aktuell nur ein HMD und zwei Controller an einem Computer unterstützt. Außerdem ist es mit der aktuellen Version nicht möglich, mehr als zwei Basisstationen gleichzeitig zu betreiben. Bei zwei Basisstationen ist der maximale Abstand eingeschränkt. Zwischen den Lichtblitzen wird die Position der Geräte mit den ebenfalls verbauten Gyrosensor und Beschleunigungsmesser approximiert.

Seit dem 27. März 2017 ist der Vive Tracker als Erweiterung des Systems für Entwickler erhältlich. Dieser Tracker dient als Erweiterung zum Tracken beliebiger Geräte mit dem Lighthouse Tracking System. Er ist kompakter als ein Controller, hat aber keine Eingabetasten und eine ebene Auflagefläche mit einer 1/4 Zoll Schraube zur Befestigung. Für die Kommunikation mit dem Computer wird ein weiterer kabelloser USB-Adapter verwendet. Theoretisch können an einem System beliebig viele zusätzliche Tracker angeschlossen werden. Praktisch liegt die Obergrenze bei 16 getrackten Geräten (1 HMD, 2 Controller, 13 Tracker).

Eine weitere Ergänzung für die HTC Vive ist der TPCAST Wireless Adapter. Hierbei wird das Kabel zwischen dem Headset und dem HMD durch eine kabellose Alternative ausgetauscht. Die USB-Daten werden per WLAN übertragen, das Bild mit einer speziell dafür entwickelten hochfrequenten Funkverbindung und für die Stromversorgung wird eine handelsübliche Powerbank verwendet.

Als Bibliothek zur Kommunikation mit der HTC Vive wird OpenVR verwendet. Die OpenVR-API von Valve erlaubt es, auf VR-Hardware von verschiedenen Herstellern zuzugreifen.

Die Genauigkeit des Lighthouse Trackings wurde in einigen Arbeiten untersucht. Das HMD zittert in Ruhelage mit einer Basisstation um 0,3mm und mit zwei Basisstationen erhöht sich diese Ungenauigkeit auf 2,1mm [lig]. Bei eigenen Messungen wurden teilweise größere Ungenauigkeiten festgestellt. In der Abb. 2.2 ist eine Messung mit einer maximalen Abweichung von fast 5mm zu sehen. Die Präzision des Vive-Trackings wurde auch untersucht. [lig] spricht von einem durchschnittlichen Fehler von 1,5 -1,9 mm beim wiederholten Positionieren an Punkten entlang eines Maßstabs.

### 2.5.2. HoloLens

Die Microsoft HoloLens [Holo] ist eine Augmented Reality Brille, die es erlaubt, interaktive Projektionen in der echten Welt darzustellen. Die Brille funktioniert ohne zusätzliche Hardware wie einen Smartphone oder einem zusätzlichen Computer. Der Nutzer schaut auf jedem Auge durch einen transparenten Bildschirm, auf dem 3D-Projektionen eingeblendet werden. Die Steuerung erfolgt durch Kopfbewegungen, Gesten und Sprachsteuerung. Die HoloLens verfügt über ein Inside-Out-Tracking zur Positionsbestimmung in Räumen. Als Sensoren werden ein Beschleunigungssensor, ein Gyroskop, ein Magnetometer, eine Tiefenkamera und vier "environment understanding" Kameras [Holb] verwendet. Die HoloLens

Position		Rotation	
Max Deviation	4.82844421640	Max Deviation	0.08846931904
	038		55437
Standard Deviation	1.14171888541		
	363		

Abbildung 2.2.: Messung des Zitterns des HMD in Ruhelage mit dem Jitter Tester [Jit].  
Position ist in mm, Rotation in Grad angegeben

baut sich mit Hilfe dieser Sensoren ein grobes Mesh der Umgebung auf, das es regelmäßig updatet und verfeinert.

### 2.5.3. Unreal Engine 4

Die Unreal Engine 4 [UE4b] ist eine der meistgenutzten Spiele-Engines. Die Engine wird von Epic Games entwickelt und kann bis zu einem Jahresumsatz von 3000 US-Dollar kostenlos genutzt werden. Unreal unterstützt viele verschiedene Plattformen wie Konsolen, Smartphones, PCs und Virtual Reality. Das Framework der Unreal Engine besteht unter anderem aus einer Grafik-Engine, einer Physik-Engine und einem mitgelieferten Editor mit vielen Werkzeugen zur Spieleentwicklung. Die Implementierung eines Spiels kann entweder in C++ oder mit Hilfe des Blueprint Visual Scriptings der Unreal Engine 4 erfolgen. Diese Blueprints sind eine von Epic Games entwickelte grafische Benutzeroberfläche, die es ermöglicht, objektorientiert zu arbeiten. Die Programmierung erfolgt dabei durch einen Graphen, bei dem verschiedene Knoten (Methoden) per Drag and Drop verbunden werden. Diese Blueprints werden automatisiert in C++-Code übersetzt. C++ und Blueprints können in einem Projekt kombiniert und parallel verwendet werden. Auch die Rendering-Engine bzw. das Unreal Material System lässt sich im Stil der Blueprints anpassen. Die dabei entstehende Graphen werden anschließend zu passenden Grafik-Shadern übersetzt. Eine direkte Anpassung der Shader ist nicht möglich. In der Implementierung wurden Blueprints und C++ verwendet. Viele Funktionen sind einfach durch die visuelle Skriptsprache erreichbar und umsetzbar. Reicht die Funktionalität der Blueprints nicht aus, so wurden diese Funktionen in C++ implementiert.

### 2.5.4. Unity

Unity [uni] ist wie die Unreal Engine eine der meistgenutzten Spieleplattformen. Sie verfügt über einen ähnlichen Funktionsumfang und die Programmierung erfolgt in C#. Ein wichtiger Unterschied ist, dass Unity die Entwicklung für die HoloLens und die Universal Windows Plattform (UWP) unterstützt. Deshalb wurde für die Entwicklung der HoloLens-Anwendung Unity verwendet.

### 2.5.5. Kinect

Die Kinect [Kinb] ist eine Hardware zur Steuerung der Videospielkonsolen Xbox360 und Xbox One. Spieler können damit anstelle des herkömmlichen Controllers alleine durch Körperbewegungen und Sprache die Software bedienen. Die erste Generation der Kinect wurde im November 2010 veröffentlicht und Anfang 2013 die zweite Generation. Als Sensoren sind eine Farbkamera, eine Tiefenkamera und ein Mikrofonarray verbaut. Zu der Kinect wurde 2012 ein nichtkommerzielles Software Development Kit veröffentlicht. Das SDK verfügte

unter anderem über Treiber für Windows und bietet die Möglichkeit in C++, C# oder Visual Basic Applications auf die Kinect zuzugreifen. Unter anderem können ungefilterte Sensordaten abgegriffen werden, aber auch das erkannte Skelett von Personen, die sich im Sichtfeld befinden. Außerdem bietet das SDK die Möglichkeit, ein Tiefenbild in eine Punktwolke umzuwandeln. Die Kinect wurde in vielen Forschungsprojekten verwendet. Es ist ein einheitlicher, günstiger Sensor, der Tiefendaten aufnimmt. Außerdem werden viele benötigten Funktionen direkt im SDK mitgeliefert. In dieser Arbeit wurde eine Kinect 2 für das Aufnehmen einzelner Punktwolken verwendet.



### 3. 3D Aufnahmen mit Punktwolken

Damit ein Experte assistierend zugeschaltet werden kann, benötigt dieser Informationen zu dem problematischen Objekt. Diese Daten sollen dem Experten in einer VR Umgebung präsentiert werden. Deshalb sollte die Aufnahme ein 3D Scan sein. Ziel der Arbeit war es, eine Lösung für das Aufnehmen der benötigten 3D Daten zu finden, die einfach bedienbar ist und ohne aufwendige Nachbearbeitung oder Berechnungen auskommt. Die Entscheidung fiel auf die Aufnahme und Visualisierung von 3D Punktwolken.

Als Sensor wurde die Kinect ausgewählt. Diese ist günstig, portabel und wird in der Forschung häufig als 3D Sensor eingesetzt. Die Kinect als Sensor bietet die Möglichkeit, aus einzelnen Aufnahmen, bestehend aus einem Farbbild und einem Tiefenbild, eine Punktwolke aus der Perspektive der Kinect zu errechnen. Eine Aufnahme (ein Frame) beinhaltet aber nur die Informationen, die aus der Perspektive der Kamera sichtbar sind. Das beinhaltet zum einen die Flächen der nächsten Oberfläche. Dahinterliegende Geometrie wird verdeckt und ist aus einer Perspektive nicht sichtbar (siehe 3.1 (a)). Außerdem sind an Kanten meist nicht genügend Informationen in der Aufnahme enthalten, damit seitliche Flächen richtig in Punkte konvertiert werden können. In vielen Fällen führt das zu falschen und nicht existierenden Flächen der errechneten Punktwolke (siehe 3.1 (b)). Diese Informationen aus einer Aufnahme reichen für die Visualisierung in VR für den Experten nicht aus. Das Objekt sollte von allen Seiten gescannt werden, damit der Experte frei entscheiden kann, von welcher Seite er das Objekt bzw. die Punktwolke betrachtet.

Um eine Punktwolke zu erhalten, die das gesamte Objekt abdeckt, werden mehrere Aufnahmen aus unterschiedlichen Perspektiven gemacht. Diese Aufnahmen müssen aber anschließend richtig zu einer großen Punktwolke zusammengefügt werden. Eine Möglichkeit, 2 Aufnahmen zusammenzufügen, ist über die Transformation zwischen den Kamerapositionen. Mit dieser Transformation lässt sich eine Punktwolke aus einem Frame in das Koordinatensystem einer anderen Aufnahme transformieren. Um diese Transformation zu errechnen, gibt es einige Ansätze, die aber meist rechenaufwändig und fehleranfällig sind. In dieser Arbeit wurde die Kinect mit dem Lighthouse Tracking der HTC Vive verbunden. Das Trackingsystem liefert eine globale Position des Kinect Sensors und ermöglicht damit eine einfache Berechnung der relativen Transformation zwischen 2 Kameraperspektiven. Durch ein globales Tracking zur Aufnahmezeit entfallen nachträgliche Berechnungen, um aus den einzelnen Aufnahmen die Kamerapositionen zu errechnen. Das spart Zeit und Rechenleistung und bietet somit eine einfache und schnelle Möglichkeit eine 3D Aufnahme von einem Objekt zu erstellen.

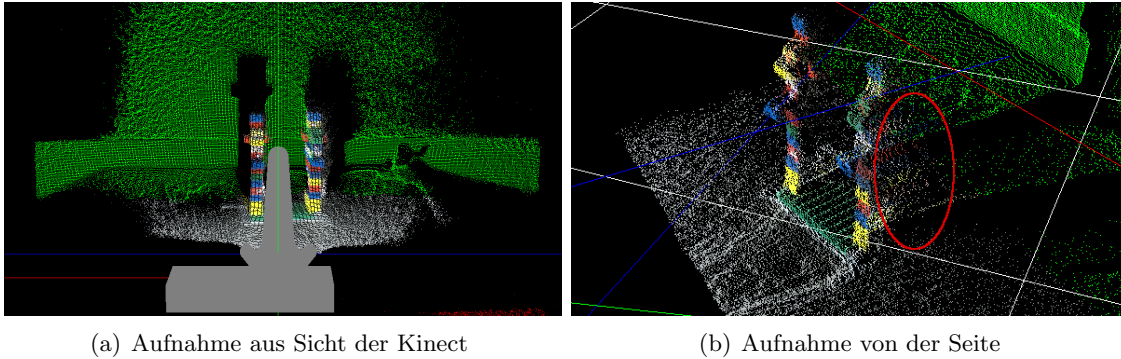


Abbildung 3.1.: Aufnahme der Kinect aus verschiedenen Perspektiven. In Bild (a) ist zu erkennen, wie die Türme den Hintergrund verdecken. In Bild 3.1(b) sind falsche Punkte zu sehen, die durch die Rekonstruktion aus einem 2D Bild entstehen.

### 3.1. Frames aufnehmen und bereinigen

Im ersten Schritt wird ein Frame mit der Kinect aufgenommen. Die Rohdaten der Kinect sind verrauscht und deshalb wird das Tiefenbild geglättet. Anschließend wird das Tiefen- und Farbbild in 3D Punkte umgewandelt und unerwünschte Punkte verworfen.

#### 3.1.1. Aufnahme und Glättung

Zum Aufnehmen einer Punktwolke aus einem Frame wird das Kinect SDK verwendet. Sowohl das Tiefenbild als auch das Farbbild kann man aus der API erhalten. Anschließend wird das Tiefenbild geglättet, um glattere Oberflächen in der Punktwolke zu erhalten. Hierfür braucht man einen Filter, der zwar die Flächen glättet, aber gleichzeitig die Objektkanten erhält. Ein bilateraler Filter erzielt den gewünschten Effekt ist aber relativ rechenaufwändig. Im Paper [MCS14] wird hierfür ein Filter vorgestellt, der auch in dieser Ausarbeitung verwendet wurde. Hierbei wird zunächst das Bild mit einem Gauß-Filter geglättet. Dieser ist nicht kantenerhaltend. Deshalb wird das geglättete Bild anschließend mit dem Original verglichen. Bei zu starker Abweichung vom Original wird der Wert des geglätteten Pixels auf das Original zurückgesetzt.

Nach der Glättung des Tiefenbildes wird dieses in eine Punktwolke umgewandelt. Hierfür wurde ebenfalls das Microsoft Kinect SDK verwendet, das alle benötigten Methoden bereitstellt.

Nach der Umwandlung werden noch weitere Punkte verworfen: Zunächst werden alle Punkte ohne zuordnungsfähige Farbe verworfen, um eine schönere Aufnahme zu erhalten. Der Farbsensor ist nicht an der gleichen Stelle der Kinect und deshalb kann es vorkommen, dass die Tiefenkamera Geometrie aufnimmt, die aus Sicht der Farbkamera verdeckt ist. Außerdem hat der Tiefeinsensor eine andere Auflösung und ein anderes Seitenverhältnis als die Farbkamera, sodass es am oberen und unteren Rand zu nicht farbigen Punkten kommt. Auch alle Punkte, die zu nah oder zu weit vom Sensor entfernt sind, werden nicht weiter betrachtet. Je weiter das Objekt entfernt ist, desto ungenauer werden die Aufnahmen. Im folgenden wurde ein Mindestabstand von 30cm und ein Maximalabstand von 90cm verwendet.

Als letztes werden alle Flächen, deren Oberflächennormale zu weit von dem Kameravektor abweicht verworfen (siehe Abb 3.1 b). Diese Flächen entstehen durch die Umwandlung des 2D Tiefenbildes in eine 3D Punktwolke. Die benötigten Informationen fehlen an dieser Stelle und Punkte werden auf die Fläche zwischen Oberflächenobjekt und Hintergrund gesetzt.

Diese Ebene stimmt nicht mit der wirklichen Oberfläche überein. Diese falschen Punkte müssen entfernt werden. Hierfür wird die Oberflächennormale verwendet. Die Normale wird aus dem Tiefenbild geschätzt.

$$\begin{aligned} dzX Axis &= depthImageAt[x + 1, y] - depthImageAt[x - 1, y] \\ dzY Axis &= depthImageAt[x, y + 1] - depthImageAt[x, y - 1] \\ Normal[x, y] &= Normalize(-dzX Axis, -dzY Axis, 1.0) \end{aligned} \quad (3.1)$$

Mit dem Skalarprodukt lässt sich der Winkel zwischen dem Kameravektor  $(0, 0, 1)$  und der Normalen ausrechnen. Ein maximaler Winkel von  $65^\circ$  hat in den Tests ein gutes Ergebnis geliefert.

## 3.2. Zusammenfügen von Frames

Für das Zusammenfügen der einzelnen Frames werden die Punktwolke und 3 Transformationen benötigt. Die Frames der Kinect und die daraus resultierenden Punktwolken haben ihren Ursprung im Tiefensensor. Die Transformation *transformControllerToKinect* zwischen dem Koordinatensystem des Controllers und der Kinect wurde bestimmt und die globale Transformation des Controllers *transformController* ist in der OpenVR API abfragbar. Die Transformation der lokalen Punktwolke in ein globale ist mit diesen beiden Transformationen möglich:

$$globalPosition = transformController * transformControllerToKinect * localPosition \quad (3.2)$$

### 3.2.1. Kalibrierung Kinect zu Vive

Eine Transformation ist die zwischen dem Koordinatensystem der Kinect und dem des Vive Controllers.

Ist zum Beispiel die Transformation entlang der X-Achse der Kinect verschoben, so verstärkt sich der Fehler, wenn man das Objekt von der anderen Seite, also um  $180^\circ$  gedreht aufnimmt (siehe Abb. 3.2). Der Fehler im lokalen Koordinatensystem wird in das globale transformiert und ist in dem Fall dann in genau entgegengesetzter Richtung.

In der offiziellen Dokumentation der Kinect ist beschrieben, dass der Ursprung von Punktwolken im Tiefensensor liegt (siehe [Kina]). Leider fehlt die exakte Positionsangabe im Gehäuse. Im Bild 3.3 aus dem chinesischen Microsoft Forum ist eine von Benutzern vermessene schematische Darstellung der Kinect abgebildet. Der Tiefensensor liegt hinter der kleineren runden Öffnung. Die exakte Position ließ sich nicht ermitteln, ohne die Kinect zu zerlegen, da auch Fertigungsungenauigkeiten die genaue Position beeinflussen können. Für die Implementation wurde angenommen, dass er sich mittig hinter der Öffnung befindet. Eine digitale Kalibrierung gestaltet sich schwierig, da das Lighthouse Tracking des Controllers zusätzlich einige Ungenauigkeiten mit sich bringt. Der Ursprung des Controllers lässt sich aus den Modellen von Steam VR auslesen (siehe Abb.3.4 (c)) Dieser liegt geschickt für VR Anwendungen, da die Z-Achse in der Hand liegt. Aber für das Tracking von Objekten liegt der Ursprung ungeschickt, da er im Inneren des Controllers liegt und es keine ebene Auflagefläche parallel zu den Achsen gibt. Bei der Implementation stand noch kein Vive Tracker zur Verfügung. Die Tracker sind zusätzliche Sensoren, die für das Tracken von Objekten mit dem Lighthouse System entwickelt wurden. Bei diesen ist der Ursprung in der Schraube und parallel zur Auflagefläche (siehe Abb. 3.5). Für die Arbeit wurde der Controller so nah wie möglich an dem Tiefensensor, also direkt darüber angebracht (siehe Abb. 3.4). Als Hilfe wurde ein 3D gedruckter Zylinder verwendet, der in den Ring des Controllers passt.

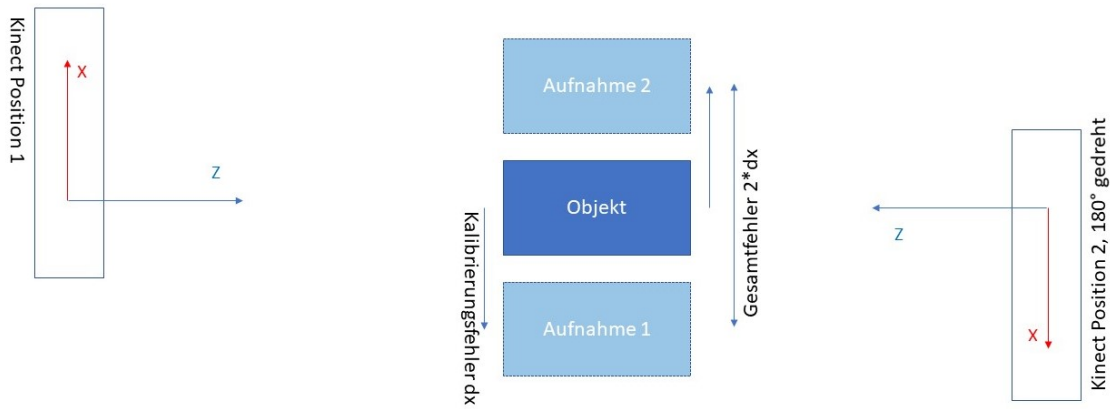


Abbildung 3.2.: Effekt einer falschen Kalibrierung  $dx$  auf die endgültige Punktwolke. Aufnahme 1 und 2 sind von lokalen Koordinaten in Welt Koordinaten transformiert



Abbildung 3.3.: Abmessungen der Kinect. Der Tiefensensor liegt in der kleinen runden Öffnung. Quelle:[Kinc]



Abbildung 3.4.: Befestigung des Controllers an der Kinect. Die Mitte des Controllers ist direkt über dem Tiefensensor. In Bild 3.4(a) ist die 3D gedruckte Halterung zu sehen. Der Controller wird auf den Zylinder gesteckt. In 3.4(c) ist die virtuelle Repräsentation. Koordinatenkreuze zeigen den jeweiligen Ursprung des Geräts (X-, Y- und Z-Achse in rot, grün und blau)





Abbildung 3.5.: Ursprung des Vive Trackers aus der offiziellen Dokumentation. Er ist in der Befestigungsschraube und parallel zu der Auflagefläche.

### 3.3. Ergebnisse

Mit dem vorgestellten Verfahren lässt sich einfach und schnell eine Punktwolke erstellen. Die Ungenauigkeiten des Trackings und Fehler in der Kalibrierung führen aber zu sichtbaren Fehlern in der endgültigen Punktwolke (siehe Abb. 3.6). Zwischen 2 Aufnahmen und den daraus resultierenden Punktwolken ist ein Versatz bis zu 2–3 cm sichtbar. In einer 3D-Umgebung, insbesondere in VR, ist das eine zu große Ungenauigkeit. Durch die Ungenauigkeiten des Trackings verändert sich die Genauigkeit und damit der Versatz der Punktwolken ständig. Vergleicht man mit einem 2m Meterstab die reale Distanz mit der relativen Distanz in VR, so erhält man in VR eine Länge von 1,98 bis 2 m. Die Distanz ist hierbei abhängig von der Orientierung zu den Basisstationen und der aktuellen Kalibrierung des Lighthouse Tracking Systems. Dieses Problem erschwert es, die Kalibrierung zwischen Controller und Kinect zu überprüfen.

Im Gegensatz zur Translation war die Messung der Rotation sehr genau und erzeugte keine sichtbaren Fehler beim Zusammenfügen der unterschiedlichen Punktwolken.

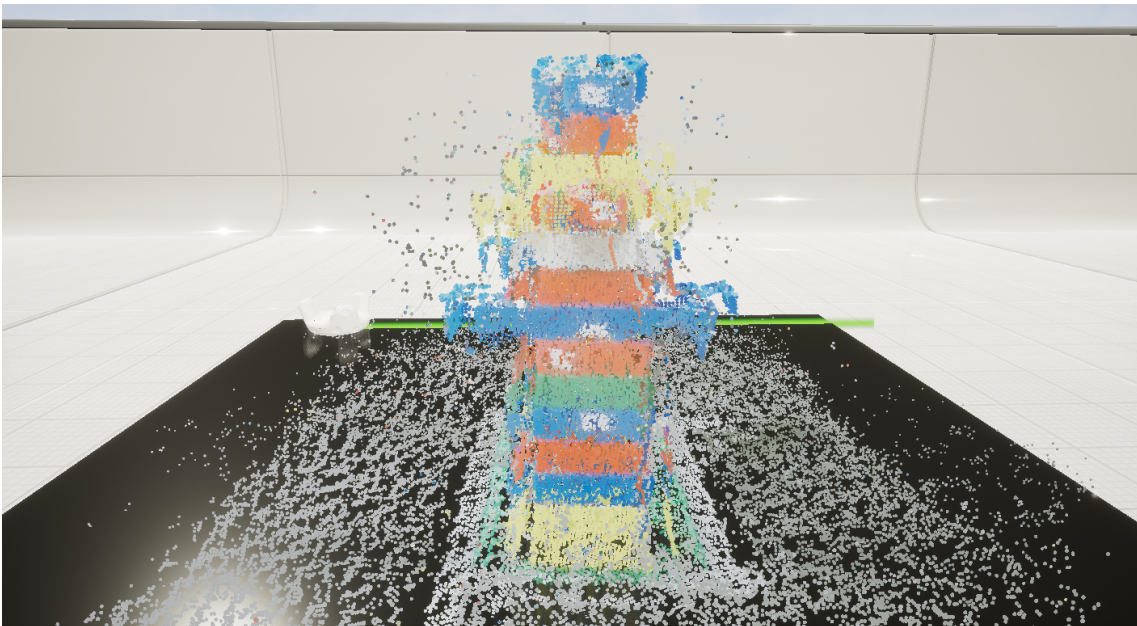


Abbildung 3.6.: Eine Punktwolkenaufnahme mit der vorgestellten Technik. Hierbei wurden 12 Aufnahmen aus verschiedenen Richtungen zu einer Punktwolke zusammengefügt. An den Kanten sind Verschiebungen zu erkennen.

## 4. Speichern der Punktwolke mit 3D Tiles

In diesem Kapitel wird ein grober Überblick über die Struktur und die Komponenten des GL-Transmission Formats (glTF) und der 3D Tiles gegeben. Diese wurden verwendet, um die Punktwolken zu speichern.

### 4.1. 3D Tiles

3D Tiles [3DT] ist eine neue, offene Spezifikation für das Streamen von massiven, heterogenen, geospatialen 3D Datensätzen. Die 3D Tiles können genutzt werden, um Gelände, Gebäude, Bäume und Punktwolken zu streamen und bieten Features wie zum Beispiel Level of Detail (LOD). Für die Arbeit wurde erwartet, dass insbesondere LOD notwendig werden könnte. Es stellte sich allerdings heraus, dass die Datenmengen, die für die Arbeit benötigt wurden, auch ohne LOD bewältigt werden können. Deswegen wurde auf dieses Feature verzichtet.

#### 4.1.1. glTF

Das GL Transmission Format (glTF [GLT]) ist ein Format zum effizienten Übertragen von 3D Szenen für OpenGL-APIs wie WebGL. OpenGL ES und OpenGL. glTF dient als effizientes, einheitliches und erweiterbares Format zur Übertragung und Laden von komplexen 3D Daten. Dieses wird in den 3D Tiles verwendet, um komplexe Geometrie wie Gebäude zu übertragen. In dieser Arbeit kamen aber nur Punktwolken zum Einsatz. Im Vergleich zu aktuellen Standards wie COLADA ist glTF optimiert, schnell übertragen und kann schnell in eine Applikation geladen werden. In einer JSON formatierten Datei (.gltf) wird eine komplette Szene samt Szenengraf, Materialien und deren zugehörigen Shadern, Kamerapositionen, Animationen und Skinning Informationen übertragen. Dabei kann auf externe Dateien verwiesen werden. Diese sind zum Beispiel Binärdaten oder Bilder, die für das einfache und effiziente Übertragen von Geometrie, Texturen oder den nötigen GLSL Shadern genutzt werden.

Eine .gltf-Datei ist JSON formatiert und bildet den Kern jedes glTF Modells. In ihr werden alle grundlegenden Informationen wie zum Beispiel die Baumstruktur des Szenengrafen und die Materialien gespeichert (siehe Abb. 4.1). Eine Szene bildet hierbei den Startpunkt für die zu rendernde Geometrie. Szenen bestehen aus Knoten (Nodes), die beliebig viele Knoten als Kinder haben können. Jeder Knoten kann eine Transformation im lokalen Raum definieren, bestehend aus einer Translation, einer Rotation und einer Skalierung.



Abbildung 4.1.: Struktur einer glTF Szene.

Die Knoten können ein Mesh und damit die eigentliche Geometrie referenzieren. Diese Geometrie wird in Buffern als Binärdaten gespeichert. Auf einen Buffer wird mit einem Accessor und einer Bufferview zugegriffen. In diesen ist spezifiziert, in welchem Format die Daten vorliegen (z.B. ein Array aus 2D-Vektoren (VEC2) aus UNSIGNED SHORT). Alle Datenformate entsprechen Formaten, die in OpenGL vorliegen, sodass die Daten ohne Konvertierung in OpenGL Vertex Array Objekts (VAO), bzw. Vertex Buffer Objekts (VBO) umgewandelt werden können.

Jedes Mesh kann auf ein Material referenzieren. Materialien bestehen aus Materialparametern, Texturen und Techniken. Techniken bestehen hauptsächlich aus GLSL Shader Programmen, das ebenfalls im glTF mitgeliefert wird. Außerdem wird spezifiziert, wie die VAO und VBO des Meshes bei dem Rendervorgang an den Shader gebunden werden müssen.

Ein weiteres Feature von glTF Dateien ist die Möglichkeit, Animationen und Skinning Informationen zu übertragen.

Buffer sind die eigentlichen Daten in einem binären Block. Diese können entweder als externe Datei (.bin) oder als BASE64 codierter String in der JSON Datei angefügt werden. Die Hauptaufgabe der Buffer ist es, große Mengen an Daten wie die Geometrie effizient zu übertragen.

#### 4.1.2. Tilesset und Tiles

Als Basis der 3D Tiles wird ein in JSON beschriebenes Tilesset verwendet, das auf die eigentlichen Daten in Tiles verweist. Das Tilesset hat eine baumartige Struktur aus Tiles und deren Metadaten. Jedes Tile hat hierbei ein 3D Volumen, das den geografischen Bereich beschreibt, und einen geometrischen Fehler des Tiles zur Echtwelt. Außerdem können Kinder



Abbildung 4.2.: Ein Tile mit 4 Kindern. Die 4 Kinder fügen die Gebäude hinzu und liegen im Volumen des Elterntiles. Als Datenstruktur liegt ein nicht uniformer Quadtree vor.

und deren Transformationen zu dem Elterntile angegeben werden. Alle Kinder liegen hierbei in dem Volumen des Elternknotens und können mit verschiedenen Datenstrukturen, wie k-d-Bäumen, Quadrees oder ähnlichem die Region genauer spezifizieren (siehe Bild 4.2. Hierbei können die Kinder das Elterntile ersetzen (replace, z.B. ein genaueres Mesh) oder das bestehende Tile ergänzen (refine, zusätzliche Gebäude oder Details). Die eigentlichen Daten der Tiles sind durch eine URL verlinkt und können dynamisch nachgeladen werden.

Tiles können in unterschiedlichen Formaten sein, zum Beispiel:

**Batched3D Model** 3D Daten, die im GL Transmission Format (glTF 4.1.1) übertragen werden. Zusätzlich können pro Modell Metadaten zum Visualisieren enthalten sein.

**Instanced3D Model** Tileformat für Instancing. Die Geometrie wird als glTF übertragen und zusätzlich eine Liste aus Positionen an denen die Objekte instanziiert werden sollen. Das kann zum Beispiel für Bäume genutzt werden.

**Point Cloud** Format, um Punktwolken zu übertragen. Das Tileformat enthält einen kleinen Header mit allgemeinen Metadaten, einer Beschreibung in JSON sowie einen Binärteil. Der JSON-Teil beschreibt, welche Art von Daten im Binärteil stehen. Insbesondere wird hier angegeben, ob Positionen und Farben enthalten sind und wie diese kodiert sind. Die eigentlichen Daten werden als Binärdaten übertragen und können so ohne weitere Verarbeitung direkt in den (Grafik-)Speicher geladen werden.

**Composite** Tileformat zum gleichzeitigen Übertragen mehrerer einzelner Tileformate in einem. Es lässt sich zum Beispiel ein Batched3D-Modell für Gebäude mit einem Instanced3D Modell für Bäume verbinden und als ein Tile übertragen.





Abbildung 4.3.: Objekt für die Aufnahme. Hinten rechts ist der Tracker fest platziert, der als Ursprung für die 3D Punktwolken verwendet wird.

## 4.2. Implementierung der 3D Tiles

Für das Speichern der Punktwolke wurde auf die Implementierung eines Level-of-Detail-Verfahrens verzichtet. In der Praxis hat sich gezeigt, dass die Punktwolken klein genug sind, um sie immer vollständig zu rendern. Sollte man größere Punktwolken, z.B. von einem ganzen Raum erstellen, könnte LOD Performancevorteile beim Visualisieren bringen. Das verwendete Tileset ist statisch und sehr einfach gehalten (siehe Anhang A) Es beinhaltet ein Tile, das auf die Punktwolke referenziert. Es ist nicht transformiert und hat als Boundigvolume eine statische 5m große Kugel.

Die eigentlichen Daten werden in einem Point Cloud Tile abgespeichert. Die Positionsdaten der einzelnen Punkte werden als Array aus float abgespeichert. Dabei bilden 3 floats immer die x,y,und z Koordinaten eines Punktes. Zusätzlich wird einen Array an Farbdaten gespeichert. Pro Punkt wird jeweils ein Byte pro RGB gespeichert.

Um das Kalibriern zwischen der Echtwelt und einer virtuellern Repräsentation zu vereinfachen, wurde beim Aufnehmen ein Vive Tracker in der Welt platziert und als Ursprung verwendet (siehe Abbildung 4.2). Alle Punkte wurden vor dem Schreiben der Datei mit der folgenden Formel in das lokale Koordinatensystem des Trackers transformiert und können beim Visualisieren erneut an dem Tracker orientiert werden.

$$exportPosition = TrackerPosition^{-1} * globalPosition \quad (4.1)$$

## 5. Visualisierung

In diesem Kapitel wird die Visualisierung der Punktwolke in der Unreal Engine 4 [UE4b] beschrieben. Die Unreal Engine ist eine mächtige Game Engine, die unter anderem Support für verschiedene Virtual Reality Systeme bietet. Durch die Verwendung einer Game Engine muss kein eigener performanter Renderer geschrieben werden, der den Anforderungen für Virtual Reality entspricht.

### 5.1. UE4 Rendering System

Die Unreal Engine 4 verwendet ein Render System, das auf DirectX aufgebaut ist. Das gesamte Rendering ist abstrahiert und aus der Engine heraus hat man keinen direkten Zugriff auf die Grafikkarte und die Shader. Das Unreal Engine Materialsystem ist der vorgesehene Weg, um Shader zu implementieren. Im Stil der UE4 Blueprints lassen sich damit grafisch Materialien definieren, die von der Engine in zugehörige Shader umgewandelt werden.

### 5.2. 3D Tiles laden und vorbereiten

Die Punktwolken können als Array direkt geladen werden, jedoch lässt die Unreal Engine nicht zu, diese auch direkt als 1D Buffer auf die Grafikkarte zu laden und zu verwenden. Als Alternative wurde der eindimensionale Array an Punkten in eine quadratische 2D Textur umgewandelt. Jeder Pixel der Textur besteht aus 4 Werten, jeweils einen für die 3 Farbkanäle Rot, Grün, Blau und  $\alpha$  (Transparenz). Ein Punkt im 3D Raum besteht aus 3 Werten für die Achsen X,Y und Z. Um die Daten auf die Grafikkarte zu laden, werden die Positionen in den 3 Farbkanälen codiert. Farbwerte sind im Wertebereich  $[0 - 1]$  und werden in unterschiedlichen Datenformaten gespeichert. Für die Implementierung wurde das HDR Datenformat der Unreal Engine verwendet. Dabei wird jeder Farbkanal in einer 16 Bit Integer codiert und ergibt damit pro Farbkanal  $2^{16}$  diskrete Farbwerte. Da wir die Koordinaten in der Farbtextur speichern, ist der 3D Raum der Punktwolke ebenfalls in  $2^{16}$  diskrete Schritte pro Achse unterteilt.

Um die Daten in eine Textur zu überführen werden diese vorbereitet. Zunächst wird das Minimum und die Größe entlang jeder Achse bestimmt. Mit diesen Daten lässt sich die

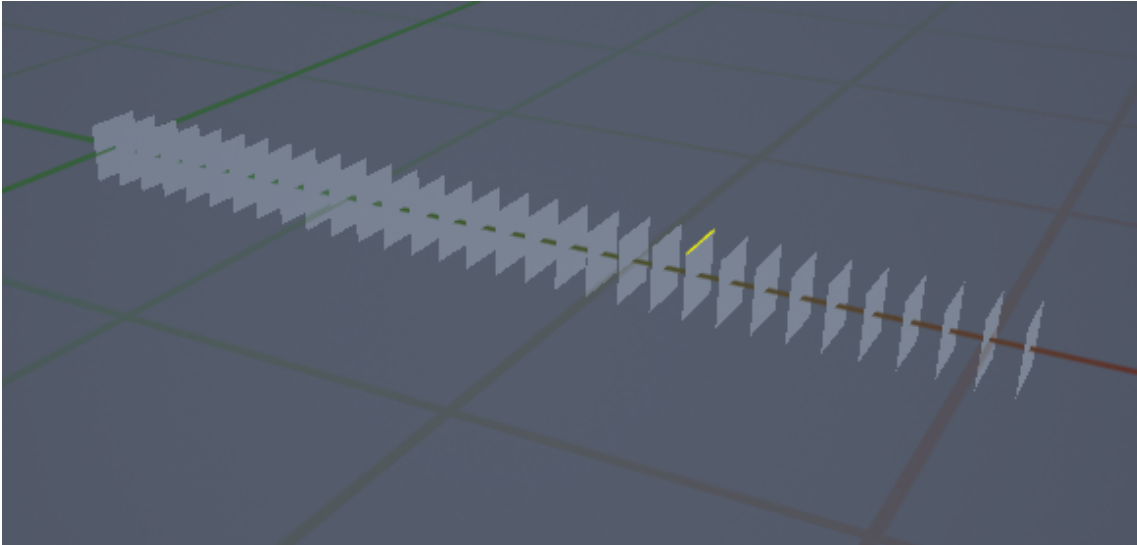


Abbildung 5.1.: Ein Quadchain mit 32 Quads

gesamte Punktwolke in den Einheitswürfel transformieren.

$$\begin{aligned}
 \forall p \in \text{Punktwolke} \\
 \min &= \text{Minimum}(p) \\
 \max &= \text{Maximum}(p) \\
 \text{size} &= \max - \min \\
 p_{\text{Einheitswrfel}} &= \frac{p - \min}{\text{size}}
 \end{aligned} \tag{5.1}$$

Die so codierten Punkte lassen sich in einer Textur abspeichern und im Shader wird diese Transformation umgekehrt.

Das Aufbereiten der Farbtextur ist einfach. Die Farben sind im 3D Tile als Byte große Integer gespeichert und können direkt in eine Unreal Textur konvertiert werden.

Die erzeugten Texturen sind immer quadratisch aber die Anzahl der Punkte ist nicht immer gleich. Durch das Verwerfen von falschen Punkten und das Zusammenfügen von mehreren Aufnahmen kann die Größe der Punktwolke variieren. Für die Implementierung wurde immer die quadratische Textur bestimmt, die genug Platz für alle Punkte bietet und die nicht verwendeten Punkte haben in der Farbtextur einen  $\alpha$  Wert von 0 sonst 1.

### 5.3. Rendering

In der Unreal Engine ist das Rendern von Punktwolken nicht vorgesehen. Als Basis für den Renderingprozess wurde deshalb eine Quadchain verwendet (Siehe Abbildung 5.1) Diese Quadchain besteht aus  $2^{20}$  einzelnen Quadraten entlang der Z Achse. Jedes Quad ist dabei an einer ganzzahligen Z Position und die Ecken sind bei 1 und -1. Durch diese Anordnung hat jedes Quadrat einen Index, die Z Position. Diese Indizierung wird verwendet, um jedes Quadrate einem Punkt zuzuordnen und das Quadrat anschließend an die gewünschte Position zu transformieren.

Beim eigentlichen Rendervorgang kann im Shader, also in UE4 Material, jedes Quads an eine Position eines Punktes geschoben werden. Hierbei wird die X-Position des Quads als



Index in die Punktwolke verwendet. Aus der Größe der quadratischen Textur und dem Index können die Texturkoordinaten (uv) in der 2D Textur berechnet werden.

Mit den Texturkoordinaten kann die Position und die Farbe des Punktes ausgelesen werden. Bevor die Position verwendet werden kann, muss diese aus dem Einheitswürfel in die wirklichen Koordinaten zurück transformiert werden. Die in der Vorbereitung errechnete Größe und das Minimum können verwendet werden, um die Gleichung 5.1 umzukehren. Anschließend fehlt nur noch die Translation und Rotation der Punktwolke in der Welt. Auch hier lässt die Unreal Engine nicht zu, dass direkt eine Matrix als Parameter für das Material übergeben werden kann, aber durch eine Position und eine Rotation lässt sich die Transformation im Material ausrechnen. Die berechnete Endposition wird als WorldPositionOffset an die Grafikkarte weitergegeben, die damit den Punkt an die richtige Weltposition setzt.

Für die Sichtbarkeit wird der Quad zu dem Betrachter gedreht und in der Größe skaliert. Zusätzlich wird das Quad noch durch eine runde Maske zu einem kleinen Kreis umgewandelt und mit der entsprechenden Farbe texturiert.

Die Quadchain hat eine feste Länge und die Punktwolke hat eine veränderbare Größe. Deshalb werden größere Punktwolken in 2 aufgeteilt. Bei kleineren Punktwolken werden die nicht verwendeten Quadrate verworfen.

## 5.4. Ergebnisse

Die resultierenden Punktwolken funktionieren für den Anwendungsfall und die in dieser Arbeit verwendeten relativ kleinen Punktwolken sind in VR visualisierbar. Außer dem Deaktivieren von Schattenberechnungen wurden keine Performance Optimierungen vorgenommen. Die Beispielwerte wurden mit einem i7 6700 und eine GTX 1070 aufgenommen. Eine genaue Analyse ist nicht erfolgt. Eine Punktwolke mit 1596685 einzelnen Punkten erreicht ca. 60fps (siehe Bild 5.2. Hierbei ist zu bedenken, dass die ganze Quadchain von  $2^{20}$  Punkten gerendert wird und überflüssige Geometrie erst im Shader/Material verworfen wird. Bei 4 einzelnen Punktwolken mit 443175 Punkten ( 110369, 115991, 110377, 106438; 4 Instanzen der Quadchain) erreicht dieses Vorgehen ca. 44 fps. Für eine große Punktwolke mit 1797690 Punkten gesplittet in 2 Instanzen sind es ebenfalls 44 fps.



(a) Gesamte Punktwolke



(b) Nahaufnahme

Abbildung 5.2.: Punktwolke mit 106438 Punkten. Die Textur im Hintergrund ist die zugehörige Positionstextur, Rechts oben ist FPS und die Renderzeit in ms zu sehen

## 6. HoloLens

Als Interaktion zwischen dem Experten in einer VR Umgebung und dem lokalen User wird die Microsoft Augmented Reality Brille HoloLens verwendet. Die HoloLens bietet die Möglichkeit, Hologramme in die echte Welt zu projizieren. Als Interaktionsmöglichkeit zwischen VR und AR wurde eine einfache ausgewählt. In VR wurde an einen Controller ein Laserbeam mit fester Länge befestigt. Bei synchronisierten Welten zwischen VR und AR soll dieser Strahl auch in der HoloLens an dem Controller hängen.

### 6.1. Unreal Engine 4 und HoloLens

Es gibt keinen offiziellen Support der HoloLens für die Unreal Engine 4. Aktuell wird das Einwickeln für die neue AR Plattform nur in Unity unterstützt. Microsoft hat auf Github einen Fork der Unreal Engine `UWP` in dem Support für die Universal Windows Plattform (UWP) enthalten ist. UWP bietet eine allgemeine Plattform für Apps, die auf allen Windows basierten Systemen funktionieren. In diesem Repository befindet sich auch ein Development Branch, der die HoloLens Unterstützung in Unreal integriert. Für die Implementierung wurde dieser Development Branch getestet. Nach dem Bauen der aktuellen Version wurde ein HoloLens Template [UE4a] in die Engine geladen. Jedoch gab es Probleme, das Projekt zu packen, sodass es auf die HoloLens geladen werden kann. Deshalb wurde für die AR Implementierung Unity verwendet.

### 6.2. HoloLens Implementierung

Die HoloLens Applikation in Unity ist einfach gehalten und die notwendige Logik wurde in der VR Applikation in Unreal umgesetzt. Zunächst setzt man in der HoloLens einen Weltanker in den Raum. Ein Weltanker dient als besonders wichtiger Punkt und das System soll diesen Punkt immer tracken und konstant halten. Die Unreal Engine sendet jedem Frame einen JSON formatierten String per UDP an die HoloLens in dem Start- und Endpunkt des aktuellen Lasers enthalten sind. Diese werden anschließend in der HoloLens relativ zum Weltanker visualisiert.

### 6.3. Kalibrierung

Die Kalibrierung erfolgt anhand des Weltankers und eines Vive Trackers. Der Weltanker orientiert sich an dem erkannten Boden, sodass die y Achse senkrecht nach oben zeigt. Um



Abbildung 6.1.: Mixed Reality Capture der HoloLens. Zu sehen ist das Koordinatenkreuz und der Vive Tracker. Am Tracker ist die 3D gedruckte Hilfe zu sehen.

die beiden Welten zu synchronisieren, wird ein Vive Tracker so platziert, dass die beiden Koordinatensysteme übereinstimmen (siehe Abb. 6.1) Hierfür wurde an dem Tracker eine 3D gedruckte Hilfe angebracht, um die aktuelle Rotation besser erkennen zu können. Durch einen Tastendruck auf einen Controller wird die aktuelle Rotation und Position des Trackers in der Unreal Engine gespeichert. Mit dieser Transformation kann der Strahl der VR Umgebung in das lokale Koordinatensystem des Weltankers transformiert werden.

### 6.3.1. Kalibrierungsfehler

Diese Art der Kalibrierung ist einfach, aber mit der aktuellen Hardware an mehreren Stellen fehleranfällig. Das erste Problem ist die menschliche Ungenauigkeit. Der Tracker muss exakt an die richtige Position mit der richtigen Rotation gelegt werden. Dabei ist das Hologramm direkt über dem Tracker, verdeckt diesen und erschwert damit das exakte Positionieren. Insbesondere die Rotation ist bei der Kalibrierung ein Problem. Eine kleiner Rotationsfehler wirkt sich weiter entfernt vom Ursprung stark auf die Kalibrierung aus. Zum Beispiel bringt ein Fehler von  $1^\circ$  bei 2m Distanz eine Verschiebung von 3,4cm ( $distance(rotate(1^\circ, (2, 10))(2, 0))$ ).

Zu den menschlichen Fehlern kommen Ungenauigkeiten vom Vive Tracking und dem HoloLens Tracking. Der HoloLens Ursprung verschiebt sich manchmal leicht, wenn er nicht direkt angeschaut wird. Dies ist bedingt durch das Inside-out Tracking der HoloLens. Durch erneutes Ansehen der Ursprungsumgebung wird der Ursprung meistens wiedererkannt und wird zurück auf die ursprüngliche Position gesetzt. Aber es konnten sehr vereinzelt dauerhafte Verschiebungen beobachtet werden. Im Extremfall waren ca. 10cm nach unten zu beobachten. Außerdem gibt es Ungenauigkeiten im Vive Tracking. Problematisch sind hierbei insbesondere die Längenunterschiede zwischen Echtwelt und virtueller Welt. In dem verwendeten Setup wurde ein 2m Zollstock mit einem Vive Controller vermessen. Dabei war die in VR gemessene Distanz 1,98m. Dieser Fehler wirkt sich direkt auf das Zusammenspiel der Vive und der HoloLens aus. Wird der Controller 2m vom Ursprung weg bewegt,

dann bewegt sich der Beam nur um 1,98 in VR und damit entfernt sich auch der Beam in der AR Visualisierung von seiner eigentlichen Position. In der Implementierung wurde versucht, das auszugleichen, indem alle Werte, die per UDP versendet werden, mit dem Faktor 1,025 skaliert wurden. Der Wert wurde experimentell bestimmt.

## 6.4. Ergebnisse

Die Kalibrierung funktioniert, ist aber nicht schnell und fehlerfrei umsetzbar. Bis die Kalibrierung vollständig stimmt, muss der Prozess teilweise ein paar Mal wiederholt und getestet werden. Nahe an dem Weltanker erreicht das umgesetzte Tool den gewünschten Effekt, den Laser an den Vive Controller zu hängen. Entfernt man sich von diesem Punkt, wird das Tracking immer schlechter und die Fehler werden sichtbarer. Für die Evaluation wurde die VR Umgebung verschoben, um getrennte Orte zu simulieren. Damit der Fehler möglichst klein bleibt und keinen großen Einfluss auf die Evaluation hat, wurde die Verschiebung so klein wie möglich gehalten (1,5m).



## 7. Evaluation

Das grundlegende Szenario, das evaluiert wird, spielt zwischen einem lokalen Nutzer (häufig auch Techniker) und einem Experten der remote zugeschaltet werden soll. Der lokale Nutzer hat ein Hardwareproblem und der Experte das Fachwissen, um das Problem zu lösen. Der allgemeine Ablauf bei so einem Szenario ist, dass der lokale Nutzer zunächst Daten aufnimmt und dem Experten zur Vorbereitung sendet. Anschließend lösen die beiden gemeinsam das Problem. Im VR Szenario kann der lokale Nutzer eine Punktwolke aufnehmen und diese dem Experten senden. Dieser kann sich die Punktwolke in einer VR Umgebung anschauen und mit seinem Controller und dem daran befestigten Laser auf die Punktwolke zeigen. Der lokale Nutzer bekommt in seiner AR Brille den Laser an der zugehörigen realen Position visualisiert. So kann der Experte auf die Punktwolkenrepräsentation des Objekts zeigen und der lokale Nutzer sieht diese Zeigegeste am echten Objekt.

Als Referenzszenario, das Video Szenario, wurde ein Videostream gewählt. Als Vorbereitung sendet der lokale Nutzer Bilder an den Experten. Für die Zusammenarbeit erhält der lokale Nutzer ein Handy mit dem er das Objekt filmen kann. Dieser Videostream kann sich der Experte anschauen und diesen als zusätzliches Kommunikationsmittel zur Sprache benutzen, um das Problem zu lösen

### 7.1. Versuchsaufbau

Das Hardwareproblem wurde in der Evaluation mit Duplosteinen simuliert. Aus den Steinen wurden insgesamt 2 Turmpaare aus 2 relativ ähnlichen Türmen gebaut (Siehe Abb. 7.1). In den Türmen wurden verschiedene Farben benutzt, sodass jeder Turm insgesamt 13 farbige Ebenen hat. Jede Farbe wurde mit einer eindeutigen kleinen Beschriftung versehen, bestehend aus einem Buchstaben und einer Zahl. Diese Beschriftung kann dazu verwendet werden, um die Korrektheit bei einem Durchlauf des Experiments zu überprüfen. Des weiteren wurde darauf geachtet, dass in jedem Turmpaar eine Farbsequenz von 4 aneinander grenzenden Farben eindeutig ist. Hierbei sollten sich die Türme aber möglichst ähnlich sein, um die Aufgabe zu erschweren. Auf einem fahrbaren Tisch wurden für beide Turmpaare Markierungen angebracht, damit diese immer an der gleichen Position auf dem Tisch stehen.

Für das VR Szenario wurde zusätzlich ein Vive Tracker auf dem Tisch der Türme platziert. Damit ist es möglich, das Objekt auch in der virtuellen Welt zu tracken und richtig



(a) Turmpaar 1

(b) Turmpaar 2

Abbildung 7.1.: Die beiden Duplotürme, die in der Evaluation verwendet wurden. Die Markierung auf dem Tisch hilft bei der exakten Positionierung.

zu positionieren. Hierfür wurde zunächst beim Aufnehmen der Punktwolke diese in das lokale Koordinatensystem des Trackers transformiert, und beim Visualisieren die aktuelle Transformation des Trackers hinzugefügt. Damit bei der Durchführung die echte Welt nicht komplett mit der virtuellen synchronisiert ist, wird die virtuelle Welt um einen konstanten Vektor verschoben. So wird eine räumliche Trennung erzwungen, sodass der Experte nicht vor Ort ist. Der Experte ist zwar in einer komplett neuen virtuellen Welt und kann den lokalen Nutzer nicht sehen, aber ohne Verschiebung sieht der lokale Nutzer den Experten mit der VR Brille. Zeigt dieser auf einen der Türme könnte der lokale Nutzer aus der Handbewegung Rückschlüsse ziehen. Es wäre somit kein Szenario bei dem Fernunterstützung untersucht wird. Diese Verschiebung wird vor dem Senden der Daten an die HoloLens wieder heraus gerechnet. In der Evaluation wurde dafür eine Verschiebung um 1,5m entlang der negativen X-Achse des Trackers gewählt. Diese Distanz wurde möglichst klein gehalten, um Trackingungenauigkeiten nicht zu verstärken.

Für das Videoszenario wurde die Handy-App IP Webcam genutzt. Diese stellt den Videostream einer Handycamera als Webstream zur Verfügung. Für den Experten wurde für das Referenzszenario das Unreal Projekt angepasst. Der Experte sitzt am Computer und der Videostream wird in der Unreal Engine visualisiert. Damit kann die gleiche Darstellung für die Aufgaben und die gleiche Zeitmessung verwendet werden.

Für die Evaluation wurde auf das Aufnehmen von Punktwolken durch die Probanden verzichtet. Die Methode, die Kinect mit dem Lighthouse Tracking zu verbinden, liefert zu ungenaue Wolken. Deshalb wurden nur statische Punktwolken verwendet, die vorher aufgenommen wurden und per Hand nachbearbeitet wurden. Diese Limitierung führt dazu, dass die Türme nicht umgebaut werden können und nur statisch betrachtet wurden. Für das Videoszenario wurden im Voraus Bilder aufgenommen, die dem Experten bei der Vorbereitung zur Verfügung stehen.

## 7.2. Versuchsablauf

Als erstes muss der Experte das nötige Vorwissen erhalten. In einem echten Szenario hat der Experte bereits alles benötigte Wissen im Vorfeld erlangt. Bei der Evaluation müssen aber alle Probanden in beiden Szenarien das gleiche Wissen erhalten. Das Vorwissen wurde durch die eindeutigen Farbsequenzen simuliert.

Ein Test in beiden Szenarien besteht aus 15 Durchläufen der gleichen Aufgabenstellung.



Bei einem Durchlauf erhält der Experte eine Aufgabe. Diese Aufgabe ist durch eine Farbsequenz von oben nach unten von 4 aufeinanderfolgenden Farben gegeben. Am Anfang oder Ende der Farbsequenz ist ein zusätzlicher gesuchter Stein markiert. Beispiel: Aufgabe aus dem Turmpaar 1: *XXX, Blau, Rot, Grün, Blau*. Im ersten Schritt sucht der Experte damit den mit XXX markierten Stein. Im VR Szenario kann er dieses direkt in der Punktwolke, die der lokale Nutzer vorher aufgenommen hat. Für das Video Szenario stehen dem Experten dafür 6 Bilder aus verschiedenen Perspektiven zur Vorbereitung zur Verfügung.

Nachdem der Experte den gesuchten Stein und dessen Farbe gefunden hat, soll er diesen dem lokalen Nutzer beschreiben. In beiden Szenarien dürfen die beiden Probanden miteinander sprechen, aber keinesfalls die ursprüngliche Farbsequenz verraten. Erlaubt ist damit unter anderem die Farbe des gesuchten Steins oder auch die Position im Turm mitzuteilen. Im VR Szenario wird dem Experten zusätzlich der Beam angeschaltet, mit dem er auf die entsprechende Stelle zeigen kann. Im Video Szenario wird der Videostream angeschaltet, sodass dieser als Interaktionsmöglichkeit zur Verfügung steht. Nachdem der lokale Nutzer den gesuchten Stein erkannt hat, liest dieser die Beschriftung vor.

Der Experte bestimmt selbst, wann er zum nächsten Aufgabenteil voranschreitet. Nachdem er aus dem Vorbereitungsdaten den Stein erkannt hat, drückt er eine Taste (Controller Trigger/ Enter) und bekommt damit Zugriff auf den Laserbeam bzw. den Videostream. Wurde das Label vorgelesen, kann er erneut mit der gleichen Taste zur nächsten Aufgabe gelangen. Bei jedem Tastendruck wird die aktuelle Uhrzeit gespeichert. Damit können die Zeiten errechnet werden.

Der Gesamttablauf der Evaluation erfolge immer im gleichen Ablauf. Vor dem Test wurden der allgemeine Fragebogen ausgefüllt, das grundlegende Szenario erklärt und eine Beispielaufgabe an einem separaten Turm erklärt. Anschließend wurden die beiden Szenarien evaluiert. Ein Team startet hierbei entweder mit dem Video Szenario oder dem VR Szenario und einem Turmpaar. Beim 2. Szenario wurden dann Rollen getauscht und das andere Turmpaar genutzt, um die Ergebnisse nicht zu verfälschen. Außerdem wurde darauf geachtet, dass beide Turmpaare in VR und Video verwendet wurden. Vor dem VR Szenario wurde kurz die Kalibrierung überprüft. Der Experte sollte auf die Turmspitzen zeigen und der lokale Nutzer sollte kurz überprüfen, ob der Laserstrahl auch in der HoloLens den Turm an der gleichen Stelle schneidet. War die Versatz zu groß, wurde neu kalibriert. Nach jedem Szenario wurden die Fragebögen zu dem Test ausgefüllt. Hierbei wurden eigene Fragen, NASA-TLX und der User Experience Questionnaire (UEQ) verwendet. Abschließend gab es noch einen weiteren Fragebogen mit einer allgemeinen Frage und freien Kommentaren. Es gab 15 Aufgaben pro Versuch. Um anfängliche Probleme und den Lerneffekt in der Evaluation nicht mit einzubeziehen, wurden bei jedem Testlauf die ersten fünf Aufgaben verworfen. Das heißt, es wurden nur die letzten zehn Durchläufe der jeweiligen Szenarien ausgewertet und auf Fehler und Zeitunterschiede untersucht.

### 7.3. Statistische Verfahren

Für die statistische Auswertung der Nutzerstudie sind einige Verfahren notwendig. Die Ergebnisse bestehen aus den Antworten der Fragebögen und den automatisierten Zeitmessungen. Die Fragebögen sind meist ordinal skalierte Daten auf einer Skala von -3 bis 3 bei dem jeder Wert nur einmal vorkommen darf. Zusätzlich war es den Probanden möglich, Freitextanmerkungen zu machen. Diese werden bei Relevanz an der entsprechenden Stelle erwähnt.

Für die ordinal skalierten Daten wird der Median und das 1. und 3. Quartil verwendet. Diese werden in Box-Whisker Plots dargestellt. Hierbei ist die Kennzeichnung wie folgt:

**Minimum und Maximum** Whisker

### 1.&3. Quartil Die Box

**Median** gepunktete Linie

**Mittelwert** Raute

Für metrisch skalierte Daten wird als Maß das arithmetische Mittel, im folgenden Mittelwert genannt, verwendet. Zusätzlich wird in den Schaubildern die Standardabweichung angegeben. Die zugehörigen Tabellen sind im Anhang C abgedruckt.

Um Zusammenhänge in den Daten zu finden und zu analysieren, wurden Signifikanzanalysen der Daten durchgeführt. Hierfür wird eine Gegenhypothese, auch Nullhypothese genannt, aufgestellt, welche ausdrückt, dass kein Zusammenhang besteht. Eine berechnete Testgröße (p-Wert) gibt Aufschluss darüber, wie wahrscheinlich die Nullhypothese zutrifft. Liegt der p-Wert unter einem Signifikanzniveau, kann die Nullhypothese verworfen werden. Für das Signifikanzniveau wird meist ein Wert von 5% verwendet [LM] Für die Evaluation wurde deshalb ebenfalls 5% gewählt.

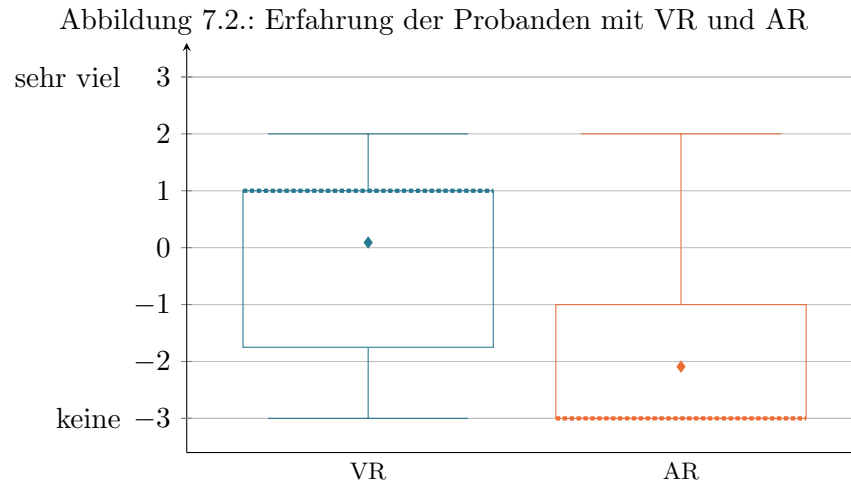
In dieser Arbeit wurde der t-Test für die Signifikanzanalyse verwendet. Dieser bietet Verfahren für abhängige (gepaarte) und unabhängige Stichproben. Abhängige Stichproben liegen vor, wenn Messwiederholung vorliegt, zum Beispiel, falls die Messwerte von der gleichen Person stammen. Auch bei natürlichen Paaren liegt eine Abhängigkeit vor, d.h. die Messwerte stammen von unterschiedlichen Personen, die zusammengehören (Ehemann und Ehefrau, oder Versuchspartnern).

In dieser Evaluation liegt eine Abhängigkeit zwischen dem Experten und dem lokalen Benutzer vor. Unabhängige Stichproben sind zum Beispiel der Vergleich der beiden Experten zwischen dem VR und Video Szenario. Die Rollen werden zwischen den Tests gewechselt und damit nimmt jeder Teilnehmer jede Rolle nur einmal an. Damit ist der Vergleich der Experten in unterschiedlichen Szenarien unabhängig. In dieser Arbeit wird die Nullhypothese verwendet, dass sich die 2 Stichproben nicht unterscheiden. Das ist keine gerichtete Hypothese, deshalb wurde der zweiseitige t-Test verwendet.

## 7.4. Probanden und Teams

Insgesamt wurde die Evaluation mit 13 Teams mit jeweils 2 Personen durchgeführt. Bei den Versuchen 2 und 5 gab es technische Probleme mit dem VR /AR Setup (große tracking Ungenauigkeiten und Abbrüche). Deshalb wurden diese Testläufe komplett aus den Daten gestrichen. Für die folgende Evaluation werden nur die Daten der verbleibenden 11 Teams mit insgesamt 22 Personen betrachtet. An der Studie nahmen 19 männlich und 3 weibliche Probanden teil. Für die Evaluation wird nur die männliche Sprachform verwendet. Alle Aussagen beziehen sich aber auf beiden Geschlechter. In der Altersgruppe bis 20 Jahren waren 2 Teilnehmer, von 20 bis 30 Jahren 17 Teilnehmer und in der Altersgruppe von 30 bis 40 Jahren drei Probanden. 6 der Probanden nutzten eine Sehhilfe und 2 gaben an, an einer Rot-Grün Schwäche zu leiden. Aus den freien Antworten geht hervor, dass eine Rot-Grün Schwäche in dem Versuchsaufbau keine Einschränkungen mit sich brachte. Die rote und grüne Farbe der Duplosteine sei kräftig und unterschiedlich genug, sodass diese trotzdem erkennbar waren.

Jeder Proband wurde nach Erfahrung zu Virtual Reality und Augmented Reality und den dabei verwendeten Systemen gefragt (siehe Abb. 7.2). Dabei ist zu erkennen, dass viele Personen schon ausgeprägte Erfahrungen mit VR gesammelt haben. Der Median der Befragten lag bei 1 wobei -3 für keine Erfahrung und 3 für sehr viel Erfahrung steht. Verwendete Systeme waren hierbei Vive (13), Playstation VR (6), Oculus Rift (6) und verschiedenen Systeme, die ein Handy nutzen (8; Cardboard, Daydream, Gear VR, etc.). Augmented Reality ist hingegen noch nicht stark verbreitet. 14 der 22 Probanden gaben



an, noch keine AR Erfahrung zu haben (Median -3). Die restlichen gaben Erfahrungen mit der HoloLens und mit Pokemon Go an.

## 7.5. Ergebnisse

### 7.5.1. Vorbereitung des Experten

Zunächst betrachten wir die Vorbereitung des Experten. In beiden Szenarien bekam der Experte eine Farbsequenz und sollte den gesuchten Stein in den Turmpaaren erkennen. Im VR Szenario hatte er dazu die Punktwolke gesehen und im Video Szenario standen dem Experte 6 Bilder aus unterschiedlichen Perspektiven zur Verfügung. Im Fragebogen wurde die Experten gefragt, wie einfach es sei, den Stein zu finden (Plot 7.3). Hierbei gaben die Experten im Video Szenario an, dass das Auffinden des Steins sehr einfach sei (Median 3), wobei der VR Experte dieses als signifikant schwerer einstufte (Median 1). Diese Einstufung ging zusätzlich auch aus den freien Kommentaren hervor. 6 der Experten fanden die Punktwolke "schwer sichtbar", "pixelig" und "ungenau" und haben vermutlich deshalb das Finden als schwieriger eingestuft. Kommentare zu den Bildern gab es nicht. Während des Versuchs wurde auch die Zeit für die Vorbereitung gemessen. In der Abbildung 7.4 sind die Zeiten als Mittelwert und Standardabweichung eingezeichnet. Bei 4 unterschiedlichen Gruppen war der Video Experte schneller als der VR Experte. In Gruppe 1 war der Video Experte signifikant schneller. In den restlichen 7 Gruppen war der VR Experte schneller. Signifikante Unterschiede gab es aber nur bei den Gruppen 10 und 13. Im Durchschnitt über alle Versuche hat der VR Experte 9,9 Sekunden und der Video Experte 11 Sekunden gebraucht. Damit ist das Steinfinden im VR Szenario im Durchschnitt 1,1s schneller, aber es konnte keine statistische Signifikanz festgestellt werden. Bemerkenswert ist, dass 5 von 6 Probanden in der Expertenrolle, die die Punktwolke kritisiert haben, im VR Szenario schneller waren als ihr Partner als Experte im Video Szenario. Ein Aspekt auf dem in diesem Test nicht eingegangen wurde, sind die persönlichen Qualifikationen der jeweiligen Experten. Experten mit besserem räumlichen Vorstellungsvermögen oder einer besseren Strategie zum Finden der Steine könnten das Ergebnis beeinflussen.

### 7.5.2. Kommunikation des gesuchten Steins

Die nächste Aufgabe des Experten war es, den identifizierten Stein an den lokalen Nutzer zu kommunizieren. Im VR Szenario stand hierfür der Laserstrahl zur Verfügung, und im Video Szenario ein Livestream.

Abbildung 7.3.: Wie einfach war es, den beschriebenen Stein zu finden?



Abbildung 7.4.: Vorbereitungszeit des Experten

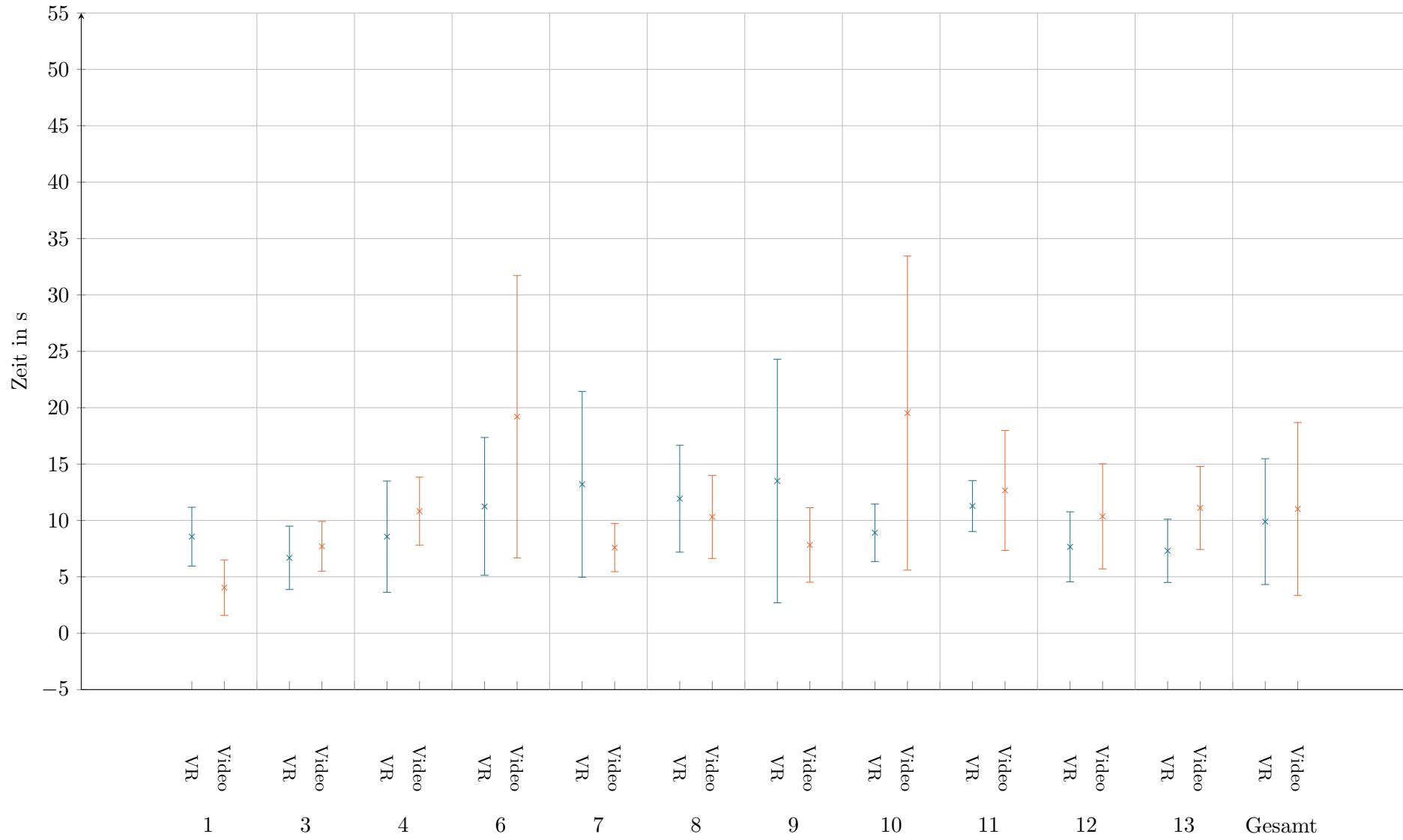
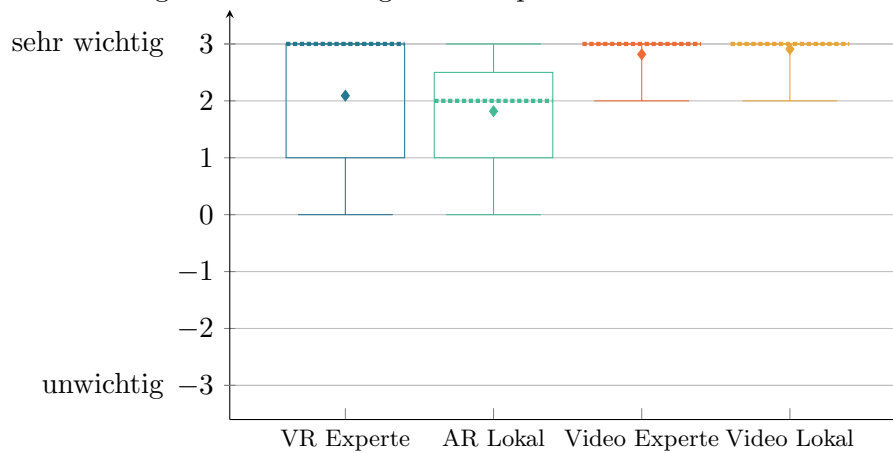


Abbildung 7.5.: Wie wichtig war die sprachliche Kommunikation?



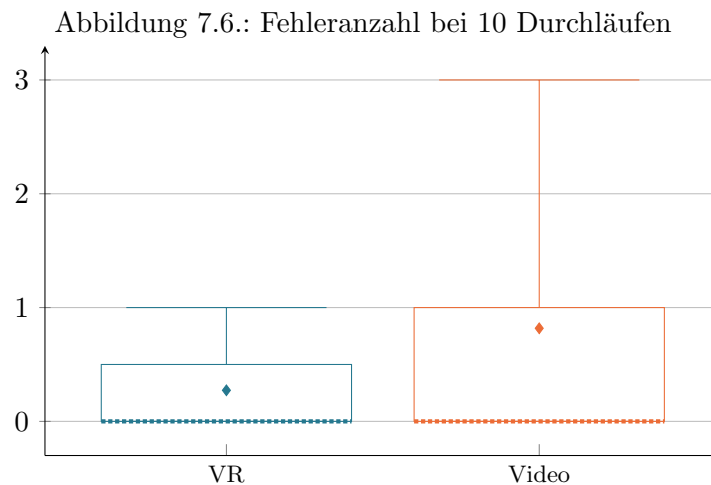
Der Videostream wurde teilweise als verwirrend wahrgenommen und dann von dem Experten fast vollständig ignoriert. Die meisten Teams haben sich bei den ersten Versuchsdurchläufen direkt oder indirekt auf eine gemeinsame Bezeichnung der Türme geeinigt. Häufig wurde links und rechts verwendet, aber auch unterscheidende Merkmale in den Türmen, wie der Turm mit der blauen Spitze oder der Turm mit der Deutschlandflagge. Links und rechts funktionieren insbesondere dann sehr gut, wenn der Nutzer nach jedem vorgelesenen Label an seine Ausgangsposition zurückkehrt. Ansonsten ist diese Bezeichnung teilweise verwirrend für den Experten, da dieser jedes Mal erkennen muss, wie der lokale Nutzer am Objekt steht. Ändert sich die Ansicht des lokalen Nutzers ständig, können sich auch die Bezeichner für jeweiligen Türme ändern. Außerdem wackelt die Kamera stark, wenn sich der lokale Nutzer bewegt, zum Beispiel, um die Beschriftung vorzulesen. Deshalb haben einige Teams nach dem initialen Einigen über die Bezeichnungen der Türme, den Videostream nicht für die Lösung der Aufgabenstellung verwendet. Eine viel stärkere Nutzung der sprachlichen Kommunikation konnte dabei aber nicht beobachtet werden.

Bei anderen Teams wurde der Videostream zum direkten Feedback benutzt. Der lokale Nutzer hat zum Beispiel mit einem Finger auf einen Turm oder Stein gezeigt und nachgefragt, ob er das richtig verstanden hat. Diese Information kann der Experte im Video sehen und die Kontrollfrage direkt beantworten.

Bei funktionierendem Tracking hat der Beam gute Ergebnisse geliefert. Einige Teams konnten durch Zeigen und zusätzliches Sagen der Farbe den Stein eindeutig beschreiben. Damit ist der Beam zumindest ein gute Grundorientierung für den lokalen Nutzer. Ein großes Problem mit dem tracking waren kleine konstante Verschiebungen in eine globale Richtung (Kalibrierfehler, bzw. Längenuntreue). Bei kleineren Verschiebungen wurde dieses als störend empfunden, aber wenn bekannt ist, wie die Verschiebung ist, dann kann diese im Kopf ausgeglichen werden.

Ein weiteres Problem, das in den freien Texten genannt wurde, waren zitterige Hände. Diese kleinen Bewegungen die teilweise durch das Tracking verstärkt werden, werden mitübertragen und können dem lokalen Nutzer das Erkennen erschweren.

In beiden Szenarien war es erlaubt, frei zu reden und zu kommunizieren. Es war lediglich verboten, die Aufgabenstellung des Experten zu sagen. Im anschließenden Fragebogen wurde gefragt, wie wichtig die sprachliche Kommunikation für die jeweilige Person gewesen ist 7.5. Zwischen den lokalen Nutzern ist in AR und Video hierbei ein signifikanter Unterschied feststellbar. Die Person in AR findet die Sprache zwar wichtig (Median 2), jedoch



unwichtiger als der lokale Nutzer im Videoszenario. Zwischen den Experten ließ sich kein signifikanter Unterschied feststellen, jedoch ist der Durchschnitt im VR Szenario geringer.

Außerdem wurden auch die lokalen Nutzer gefragt, wie einfach es war, den beschriebenen Stein zu finden (siehe Abbildung 7.3). Zwischen den lokalen Nutzern ist kein Unterschied feststellbar. Das heißt, die sprachliche Kommunikation war unwichtiger, aber das eigentliche Auffinden wurde in beiden Szenarien gleich leicht bewertet.

### 7.5.3. Fehleranzahl

Wie in der Statistik 7.6 zu sehen ist, wurden kaum Fehler in den letzten 10 Durchläufen gemacht. Im VR Szenario waren 8 von 11 Teams fehlerfrei, im Video Szenario 6 von 11. In den nicht gewerteten ersten 5 Trainingsdurchläufen waren mehr Fehler zu finden. Im Video Szenario wurde 2 mal 3 Fehler gemacht, was auf eine größere Fehleranfälligkeit hinweist.

In beiden Szenarien sind allgemeine Fehler bei dem Versuch häufig beim Experten passiert. Zum Beispiel wurde der letzte Stein der Sequenz beschreiben und nicht der eigentlich gesuchte. Sprachliche Ungenauigkeiten haben auch zu Fehlern geführt. “Der zweite schwarze Stein” und “der zweite Stein, der Schwarze” klingen ähnlich, beschreiben aber unterschiedliche Steine.

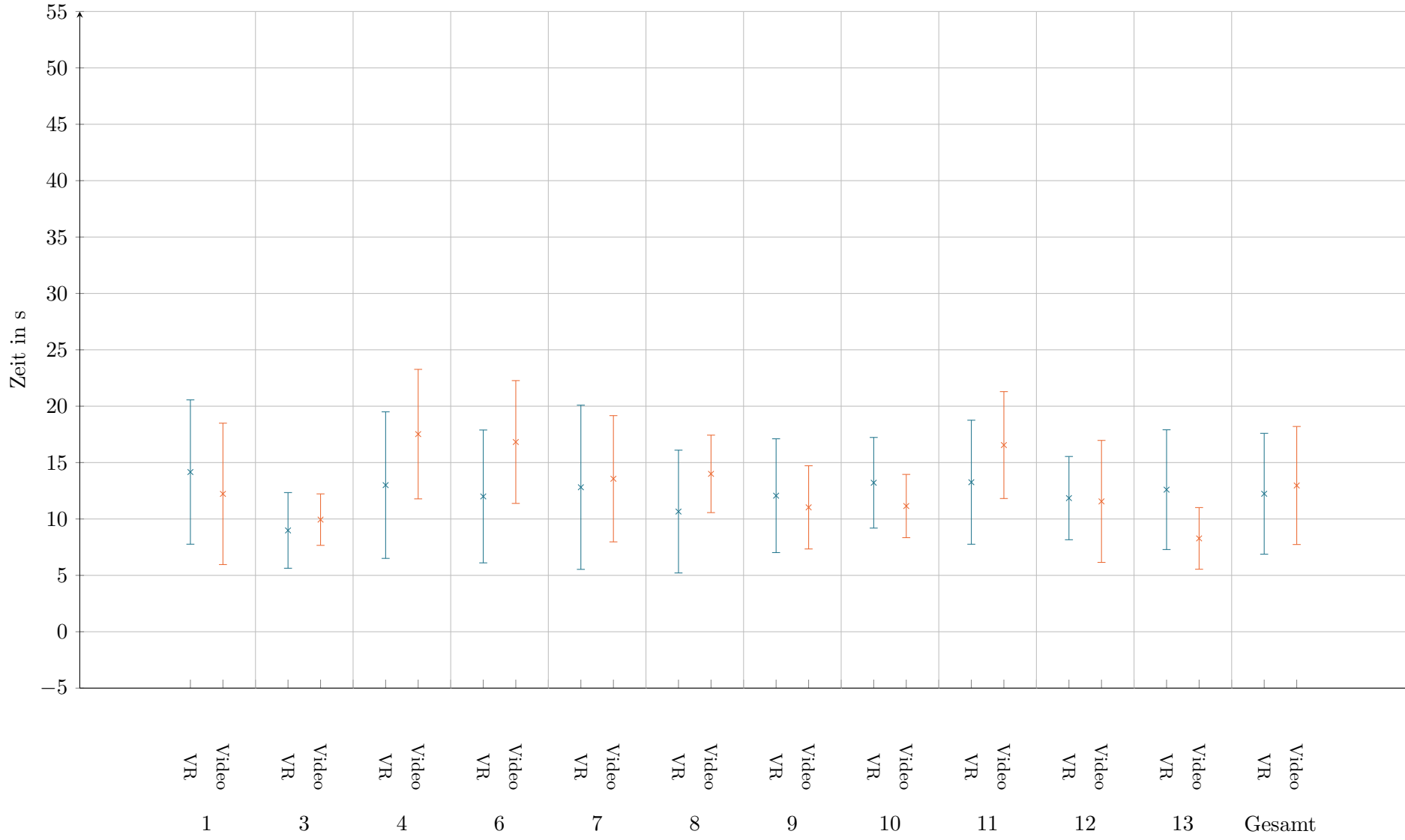
Eine zusätzliche Fehlerquelle im VR Szenario ist ein globaler Versatz des Strahls. Ist zum Beispiel der Strahl um 1 bis 2 Steine verschoben, dann zeigt dieser in der AR Umgebung andere Steine an, als der Experte in seiner VR Umgebung sieht.

Im Video Szenario war das Verwechseln von rechts und links ein häufiger Fehler. Teilweise wurden die Steine richtig beschreiben, aber dann die Beschriftung auf dem anderen Turm vorgelesen. Der zweite Fall, der zu einer links rechts Verwechslung geführt hat war, dass der Experte eine andere Ansicht in den Bildern gewählt hat, als der lokale Nutzer in seinem Videostream. Passt der Experte nicht auf, dann beschreibt er rechts und links anhand des Bildes, aber der lokale Nutzer steht auf der falschen Seite.

#### 7.5.3.1. Timings

Während der Kommunikation zwischen dem Experten und dem lokalen Nutzer wurde ebenfalls die Zeit gestoppt. Die Messung beginnt, sobald der Experte anfängt zu kommunizieren und endet sobald die Beschriftung vorgelesen wurde. In Abbildung 7.7 sind die Zeiten abgebildet. Wie bei der Vorbereitung gab es 6 Gruppen, die in VR schneller waren

Abbildung 7.7.: Kommunikationszeit





und 5, die in dem Video Szenario schneller waren. Signifikant schneller war nur Gruppe 13 im Videoszenario. Insgesamt war das VR Szenario mit 12s Durchschnittszeit 700ms schneller. Ein wichtiger Faktor, warum bei dieser Messung keine Unterschiede festgestellt werden können, ist, dass der lokale Nutzer, nachdem er den Stein erkannt hat, noch die zugehörige Beschriftung finden musste. Dieser Vorgang hat je nach Positionierung der Beschriftung länger gedauert als die eigentliche Kommunikation.

#### 7.5.4. NASA TLX und UEQ

In der Evaluierung wurden 2 standardisierte Tests verwendet: NASA-TLX und der User Experience Questionnaire (UEQ).

Beim NASA TLX [Har06] werden 6 Kategorien auf einer Skala von 0 bis 100 in 5er Schritten bewertet. Auf die anschließende Gewichtung der Kategorien wurde verzichtet. Die Ergebnisse wurden als Box-Whisker Plot 7.9 dargestellt.

Bei der geistigen Anforderung ist zu erkennen dass der lokale Nutzer im Durchschnitt leicht geringere Werte angegeben hat als der Experte. Dieses lässt sich durch die Farbsequenz, die der Experte erkennen musste, erklären. Der AR Benutzer hat im Durchschnitt die niedrigsten Werte angegeben. Bei der körperlichen Anforderung hat der Video Experte signifikant geringere Werte angegeben als der VR Experte. Dies ist nicht verwunderlich, da der Video Experte am PC sitzt, während der VR Experte steht und sich frei in der virtuellen Welt bewegen konnte. Zwischen den lokalen Benutzern ist kein signifikanter Unterschied erkennbar. Bei der zeitlichen Anforderung gibt es keine signifikanten Unterschiede, jedoch bewertet der Benutzer in AR leicht besser. Beim Vergleich der selbst eingeschätzten Leistung gibt es einen signifikanten Unterschied zwischen dem VR Experten und AR Benutzer. Der AR Benutzer gibt hier die beste Leistung an. Bei der Anstrengung gab es keine signifikanten Unterschiede, aber auch hier hat der AR Benutzer leicht besser bewertet. Signifikante Unterschiede gab es aber bei der Frustration. Diese wurde auch beim AR Benutzer am geringsten bewertet. Dieser Trend ist auch im Gesamtscore zu erkennen. Hier ist der Durchschnitt beim lokalen AR Benutzer am geringsten aber nicht signifikant. Bei Betrachtung der gesamten Werte hat der AR Benutzer die höchste Leistung angegeben aber dabei die geringste Frustration gehabt.

Der User Experience Questionnaire (UEQ [UEQ]) ist dazu gedacht, die Nutzererfahrung zu messen. Der Nutzer beantwortet hierzu 26 Gegensatzpaare von Eigenschaften auf einer Skala von -3 bis 3. Die 26 Paare werden dann 6 Skalen zugeordnet und darüber ein Mittelwert gebildet. In der Abbildung 7.10 sind die Mittelwerte und die Standardabweichung der einzelnen Rollen zu sehen. Hier gibt es einige signifikante Unterschiede zwischen den Szenarien zu erkennen. Vergleicht man die Experten miteinander und auch die lokalen Nutzer miteinander so sind die gleichen signifikanten Unterschiede zu erkennen. Die Attraktivität, Stimulation und Originalität ist bei beiden Vergleichen im VR Szenario wesentlich besser. Bei Stimulation und Originalität wurde das Video Szenario sehr schlecht bewertet. Bei der Durchschaubarkeit, Effizienz und Steuerbarkeit sind leicht besserer Bewertungen im VR Szenario zu erkennen, jedoch sind diese nicht signifikant.

Zusammenfassend wird das VR Szenario von den Benutzern besser bewertet. Das System wird generell als attraktiver und origineller bewertet als ein herkömmlicher Videostream. Gleichzeitig gibt der lokale Nutzer in einer AR Umgebung eine besser Leistung und eine geringere Frustration an, obwohl die Kalibrierungs- und Trackingprobleme hauptsächlich den AR Benutzer betreffen. Bei dem Vergleich der Experten ist abgesehen von der körperlichen Anforderung nur bei der Nutzererfahrung ein Unterschied zu erkennen. Diese wird in VR besser bewertet.

Abbildung 7.8.: War es im VR/AR Szenario von Vorteil, von der Perspektive und Bewegung der anderen Person unabhängiger zu sein und nicht an die Ansicht aus des Videos gebunden zu sein?



Abbildung 7.9.: NASA-TLX, Die Skala geht von gering (0) bis hoch (100). Bei Leistung von gut (0) bis schlecht (100)

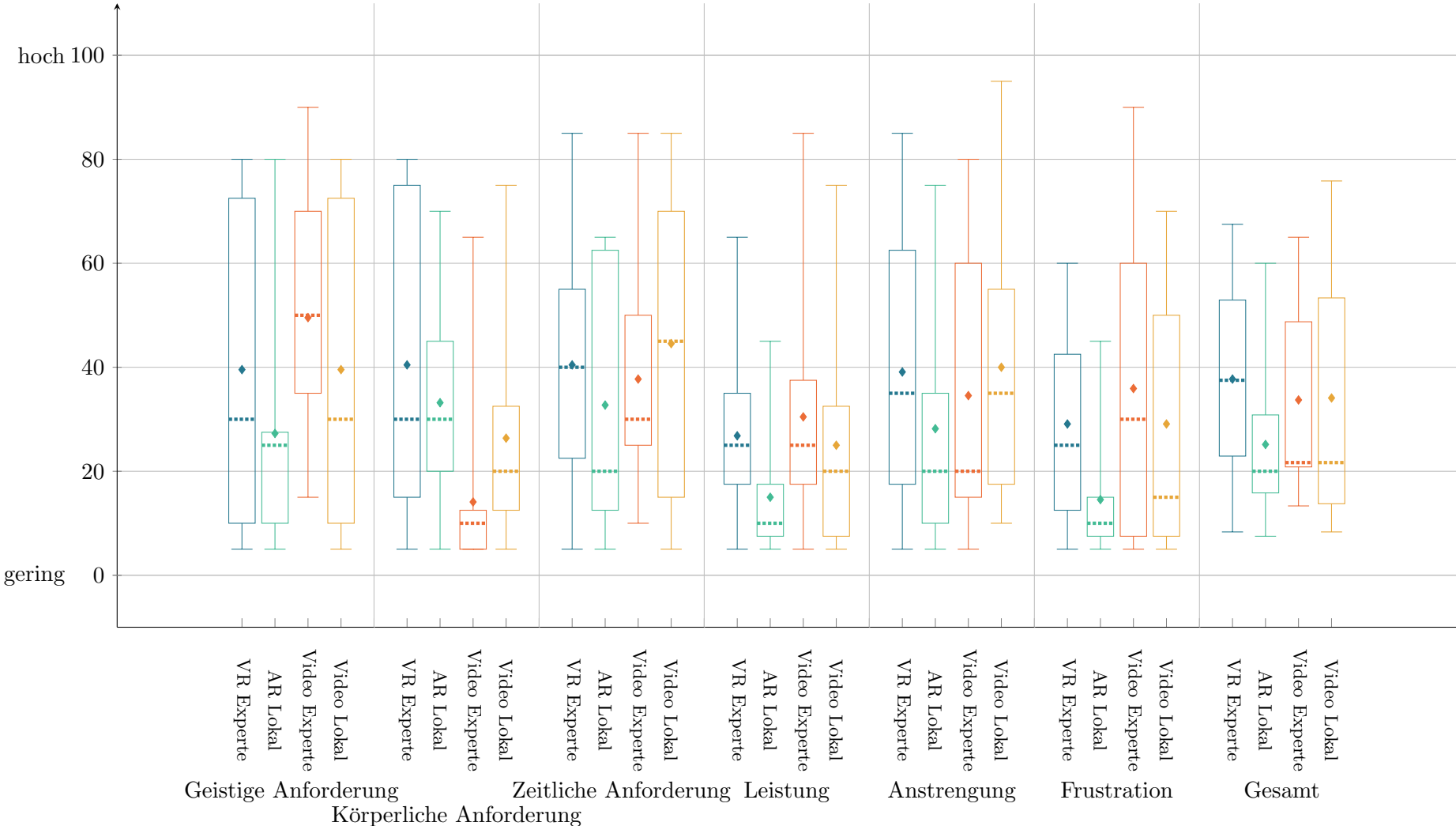
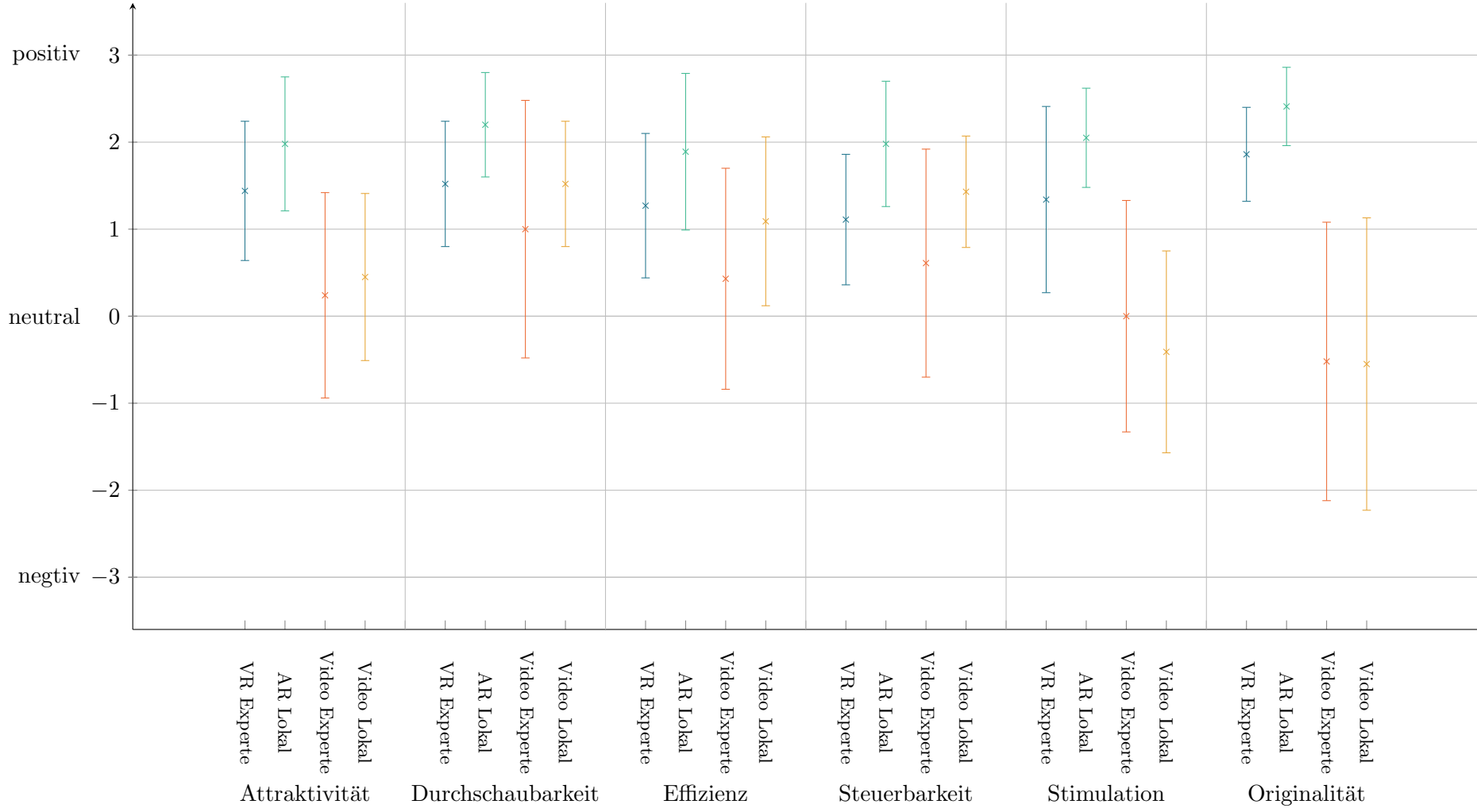


Abbildung 7.10.: Ergebnisse des User Experience Questionnaire



### 7.5.5. Unabhängigkeit

Nachdem die Teams beide Szenarien durchgeführt haben, wurde eine abschließende Frage gestellt. Es wurde gefragt, ob es in dem VR Szenario von Vorteil war, von der andern Person unabhängig zu sein. Im Videostream ist der Experte an die Perspektive und die Bewegungen der Kamera, also den lokalen Nutzer selber gebunden.

In dem Plot 7.8 kann man erkennen, dass die meisten Nutzer diese Unabhängigkeit als Vorteil gesehen haben. Die Kamera bewegt sich und zeigt nicht immer den richtigen Bildausschnitt.

Aber es gibt auch Bewertungen, die diese Unabhängigkeit als sehr schlecht eingestuft haben. Dieses hat 2 in den Bemerkungen genannte Gründe. Zum einen wird der Strahl in der HoloLens nicht mit der Geometrie geschnitten und endet damit nicht am ersten Schnittpunkt, sondern geht durch das Objekt hindurch. Stehen Experte und lokaler Nutzer auf unterschiedlichen Seiten des Objekts und der Strahl geht schräg durch das Modell, so zeigt er auf der Austrittsseite einen falschen Schnittpunkt. Dieser kann zu Verwirrungen führen. Ein zweiter Grund ist, dass wir die gleiche Perspektive gewöhnt sind. Wenn wir entlang des Strahls schauen, ist es einfacher zu erkennen, wo er aktuell schneidet. Steht man senkrecht dazu, ist es schwerer zu erkennen, wo die genauen Schnittpunkte sind. Eine neue Positionierung des lokalen Nutzers löst dieses Problem.

### 7.5.6. Zusammenfassung

Betrachtet man die Evaluation als Ganzes dann spricht diese für die Weiterentwicklung und Erforschung der Zusammenarbeit mit VR und AR Technologien. Das vorgestellte System erreicht trotz technischer Probleme eine leicht bessere Bewertung in fast allen untersuchten Punkten. Die Vorbereitung des Experten in VR anhand einer Punktwolke wurde als schwieriger bewertet, liefert aber im Durchschnitt eine leicht bessere Vorbereitungszeit. Bei der Kommunikation konnte ebenfalls eine leicht bessere Kommunikationszeit gemessen werden, und die benutzte sprachliche Kommunikation wurde weniger als im Video Szenario. Im VR Szenario wurden weniger Fehler gemacht und die Nutzer haben das System besser bewertet.



## 8. Fazit und Ausblick

In der heutigen hochtechnisierten Zeit wird immer häufiger das Fachwissen eines Experten benötigt. Experten sind aber häufig nicht vor Ort und müssten erst anreisen. In dieser Arbeit wurde eine Möglichkeit vorgestellt, um das Zusammenarbeiten eines lokalen Benutzers mit einem nicht anwesenden Experten an einem Objekt zu realisieren. Verwendet wurde dabei eine Vive, die Kinect und die HoloLens. Zunächst müssen hierfür alle benötigten Informationen aufgenommen und an den Experten übermittelt werden. Es wurde zunächst eine Methode vorgestellt, mit einer Kinect und dem Lighthouse Tracking der Vive einen Punktwolken-Scan des Objektes anzufertigen. Der Experte und der lokale Nutzer können dann gemeinsam mit einer Zeigegeste an demselben Objekt arbeiten. Der Experte sieht die 3D Punktwolke und den Beam, wobei der lokale Nutzer den Beam in einer AR Umgebung am echten Objekt eingeblendet bekommt.

Dieses Verfahren und die noch relativ neue Hardware bringt einige Probleme mit, wie zum Beispiel Ungenauigkeiten im Tracking und die Synchronisation der verschiedenen Systeme zueinander. Für den jeweiligen Anwendungszweck sind die Vive und die HoloLens gut geeignet und die Fehler im Tracking fallen kaum auf, jedoch werden diese in Verbindung mit anderen Systemen sichtbar. Die Nutzerstudie zeigt trotz der Probleme mit der Hardware einige Verbesserungen gegenüber dem Videostream Referenzszenario. Das VR/AR System von den Nutzern als attraktiver bewertet und zeigt geringe Frustration und eine bessere selbst eingeschätzte Leistung beim lokalen Benutzer. Außerdem waren die Benutzer im VR Szenario durchschnittlich schneller und es wurden weniger Fehler gemacht.

Das vorgestellte Aufnehmen der Punktwolke mit Kinect und Lighthouse Tracking bietet eine gute und schnelle Grundlage für einen 3D Scan, ist aber zum unbearbeiteten Verwenden noch zu ungenau. Durch Verbesserung der Kalibrierung des Controllers zu Kinect und das Tracking des Controllers könnte dieses Verfahren eine ideale Grundlage für schnelle 3D Scans bieten. Bei der Kalibrierung zueinander könnte man einen anderen Sensor verwenden, bei dem die Position im Gehäuse besser dokumentiert ist. Außerdem könnte ein Vive Tracker anstelle des Controllers bessere Ergebnisse bringen, da der Ursprung des Trackers genau spezifiziert ist und für den Anwendungsfall des Objekt Trackings gedacht ist. In praktischen Anwendungen sollten die Punktwolkenaufnahmen automatisch nachbearbeitet werden. Die einzelnen Teile der Punktwolke haben vergleichsweise kleine Verschiebungen und besitzen häufig gemeinsame Flächen. An diesen könnte im Nachhinein die exakte Punktwolke berechnet werden.

Ein weiterer Kritikpunkt aus der Evaluation sind die zu ungenauen Punktwolken und das

normale Nutzer es nicht gewöhnt sind mit Punktwolken zu arbeiten. Generiert man ein Mesh aus der Wolke, dann könnte dieses eine verbesserte Darstellung ergeben.

Die Bisherige Implementierung erlaubt nur das Aufnehmen und Anzeigen von statischen Punktwolken. Verändert der lokale Nutzer den betrachteten Bereich, dann erhält der Experte darüber keine Infos. Ist die Aufnahmetechnik performant genug, könnte man dem lokalen Nutzer regelmäßig neue Aufnahmen machen lassen. Alternativ positioniert man mehrere Kinects aus verschiedenen Richtungen im Raum, sodass die Area of Interest von allen benötigten Richtungen gescannt wird. Diese Daten könnten live in die VR Umgebung gestreamt und visualisiert werden. Mit einer live Punktwolke kann das Objekt in der Evaluation auch verändert werden. Problematisch könnte die Verdeckung durch den lokalen Nutzer sein.

Die Kalibrierung von Vive zu HoloLens und die Ungenauigkeiten im Tracking bringen weitere Möglichkeiten das System zu verbessern. Die einfache Lösung wäre, die HoloLens selber mit einem Tracker zu versehen und man erhält einen dauerhaft sich selber aktualisieren gemeinsamen Punkt. Damit eliminiert man die Ungenauigkeit aus dem HoloLens Tracking. Das Problem, dass das Lighthouse Tracking nicht längengetreu ist, wird dabei nicht gelöst. Eine weitere Möglichkeit wäre es, Vive Tracker mit Markern zu versehen. Diese können dann mit der HoloLens erkannt werden und in dem Koordinatensystem der HoloLens platziert werden. Damit umgeht man den menschlichen Ungenauigkeiten bei der Kalibrierung und wenn man mehrere Tracker mit Markern verwendet, erhält man die Längen in der HoloLens und der Vive. Mit dieser Information kann man die exakte Längenverzerrung ausrechnen und beseitigen.

Das verwendete Evaluationszenario war relativ einfach, sehr abhängig vom Tracking, der Kalibrierung und den jeweiligen Probanden. Bei weiteren Untersuchungen könnten komplexere Szenarien evaluiert werden, um die Vorteile der VR Umgebung klarer herauszuarbeiten. Um Ungenauigkeiten im Tracking auszugleichen, könnte man größere Objekte verwenden als Duplos. Eine Möglichkeit wäre es, einen ca. 1m großen Würfel auf jeder Seite mit farbigen Schachbrettmustern zu versehen. Bei großen Feldern sollten Trackingungenauigkeiten ein kleineres Problem darstellen und in einem vergleichbaren Videostream wäre die Orientierung noch schwieriger. Jedoch muss sich bei dem Szenario überlegt werden, wie das Vorwissen dem Experten vermittelt wird. In VR ist das relativ einfach möglich durch Highlights oder eingeblendete Pfeile, im Referenzszenario ist dieses schwieriger.

Beim Zusammenspiel im VR und AR könnten mehr Interaktionstechniken evaluiert werden. Bisher hat der lokale Nutzer keine Möglichkeit, dem Experten etwas zu zeigen. Die einfachste Lösung hierfür wäre, dem lokalen Nutzer selbst einen Controller zu geben mit dem er auf das Objekt zeigen kann. Alternativ könnte auf die Klick-Geste der HoloLens zurückgegriffen werden. Klickt der lokale Nutzer in der Welt etwas an, so wird die Linie zwischen Kopf und der Klickposition an den Experten übertragen und dort visualisiert.

Eine weitere hilfreiche Ergänzung könnte das Visualisieren von Avataren in VR und AR sein. Allein die Visualisierung der aktuellen Kopfposition (Headsets) des Partners könnte darüber Aufschluss geben, was dieser gerade betrachtet.

Eine weitere Interaktionsmöglichkeit wäre das Platzieren von 3D Objekten bzw. Hologrammen in der Welt. Der Experte hätte damit z.B. die Möglichkeit, Referenzobjekte direkt darzustellen. Bei animierten Hologrammen könnten so direkt Montageschritte visualisiert werden.



## A. 3D Tile JSON

In diesem Anhang finden Sie das verwendete Tileset der 3D Tiles

```
1 {
2   "asset": {
3     "version": "0.0"
4   },
5   "geometricError": 0,
6   "root": {
7     "boundingVolume": {
8       "sphere": [
9         0,
10        0,
11        0,
12        5
13      ]
14    },
15    "geometricError": 0,
16    "refine": "add",
17    "children": [
18      {
19        "transform": [
20          1.0, 0.0, 0.0, 0.0,
21          0.0, 1.0, 0.0, 0.0,
22          0.0, 0.0, 1.0, 0.0,
23          0.0, 0.0, 0.0, 1.0
24        ],
25        "boundingVolume": {
26          "sphere": [
27            0,
28            0,
29            0,
30            5
31          ]
32        },
33        "geometricError": 0,
34        "content": {
35          "url": "example.pnts"
36        }
37      }
38    ]
39  }
40 }
```

## B. Evaluations Fragebögen

### B.1. Anfangsfragebogen

**Anfangsfragebogen**

**ID:** \_\_\_\_\_ **Datum/Uhrzeit:** \_\_\_\_\_

**Allgemeine Angaben:**

Geschlecht	<input type="radio"/> männlich	<input type="radio"/> weiblich
Alter	<input type="radio"/> <20	<input type="radio"/> 20-30 <input type="radio"/> 30-40 <input type="radio"/> >40
Brillenträger	<input type="radio"/> Ja	<input type="radio"/> Nein
Rot-Grün-Sehschwäche	<input type="radio"/> Ja	<input type="radio"/> Nein
Farbenblind	<input type="radio"/> Ja	<input type="radio"/> Nein

**Erfahrung mit Virtual Reality (VR)**  
Haben Sie bereits Erfahrungen mit Virtual Reality gesammelt?

Ich habe ...	--- -- - -/+ + ++ +++	Ich habe ...
noch keine Erfahrungen gesammelt	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>	sehr viele Erfahrungen gesammelt

Falls Sie schon Erfahrungen mit VR gesammelt haben:  
Welche Hardware wurde von Ihnen genutzt?

**Erfahrung mit Augmented Reality (VR)**  
Haben Sie bereits Erfahrungen mit Augmented Reality gesammelt?

Ich habe ...	--- -- - -/+ + ++ +++	Ich habe ...
noch keine Erfahrungen gesammelt	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>	sehr viele Erfahrungen gesammelt

Falls Sie schon Erfahrungen mit AR gesammelt haben:  
Welche Hardware wurde von Ihnen genutzt?

## B.2. Fragebogen nach jedem Szenario

ID:

Datum/Uhrzeit:

Scenario	<input type="radio"/> VR/AR	<input type="radio"/> Video Stream
Rolle	<input type="radio"/> Experte	<input type="radio"/> Techniker

Wie einfach hat die Zusammenarbeit in dem Szenario geklappt

Die Zusammenarbeit war ...	--- -- - -/+ + ++ +++	Die Zusammenarbeit war ...
sehr schwierig	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>	sehr einfach

Wie einfach war es den beschriebenen Stein zu finden

Das Finden war ...	--- -- - -/+ + ++ +++	Das Finden war ...
sehr schwierig	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>	sehr einfach

Wie sehr waren sie von der sprachlichen Kommunikation abhängig

Die Kommunikation war ...	--- -- - -/+ + ++ +++	Die Kommunikation war ...
unwichtig	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>	sehr wichtig

Freie Kommentare

_____
_____
_____
_____
_____
_____

BITTE WENDEN

## Fragebogen - Teil 1

Klicken Sie in jeder Skale auf den Punkt, der Ihre Erfahrung im Hinblick auf die Aufgabe am besten verdeutlicht.

<b>Geistige Anforderung</b>										
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
										Hoch
<b>Körperliche Anforderung</b>										
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
										Hoch
<b>Zeitliche Anforderung</b>										
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
										Hoch
<b>Leistung</b>										
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
										Schlecht
<b>Anstrengung</b>										
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
										Hoch
<b>Frustration</b>										
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
										Hoch

Wie viel geistige Anforderung war bei der Informationsaufnahme und bei der Informationsverarbeitung erforderlich (z.B. Denken, Entscheiden, Rechnen, Erinnern, Hinsehen, Suchen ...)? War die Aufgabe leicht oder anspruchsvoll, einfach oder komplex, erfordert sie hohe Genauigkeit oder ist sie fehler tolerant?

Wie viel körperliche Aktivität war erforderlich (z.B. ziehen, drücken, drehen, steuern, aktivieren ...)? War die Aufgabe leicht oder schwer, einfach oder anstrengend, erholsam oder mühselig?

Wie viel Zeitdruck empfanden Sie hinsichtlich der Häufigkeit oder dem Takt mit dem die Aufgaben oder Aufgabenelemente auftraten? War die Aufgabe langsam und geruhsam oder schnell und hektisch?

Wie erfolgreich haben Sie Ihrer Meinung nach die vom Versuchsleiter (oder Ihnen selbst) gesetzten Ziele erreicht? Wie zufrieden waren Sie mit Ihrer Leistung bei der Verfolgung dieser Ziele?

Wie hart mussten Sie arbeiten, um Ihren Grad an Aufgabenerfüllung zu erreichen?

Wie unsicher, entmutigt, irritiert, gestresst und verärgert (versus sicher, bestätigt, zufrieden, entspannt und zufrieden mit sich selbst) fühlten Sie sich während der Aufgabe?

**Bitte geben Sie Ihre Beurteilung ab.**

Um das Produkt zu bewerten, füllen Sie bitte den nachfolgenden Fragebogen aus. Er besteht aus Gegensatzpaaren von Eigenschaften, die das Produkt haben kann. Abstufungen zwischen den Gegensätzen sind durch Kreise dargestellt. Durch Ankreuzen eines dieser Kreise können Sie Ihre Zustimmung zu einem Begriff äußern.

Beispiel:

attraktiv	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	unattraktiv
-----------	-----------------------	----------------------------------	-----------------------	-----------------------	-----------------------	-----------------------	-----------------------	-------------

Mit dieser Beurteilung sagen Sie aus, dass Sie das Produkt eher attraktiv als unattraktiv einschätzen.

Entscheiden Sie möglichst spontan. Es ist wichtig, dass Sie nicht lange über die Begriffe nachdenken, damit Ihre unmittelbare Einschätzung zum Tragen kommt.

Bitte kreuzen Sie immer eine Antwort an, auch wenn Sie bei der Einschätzung zu einem Begriffspaar unsicher sind oder finden, dass es nicht so gut zum Produkt passt.

Es gibt keine „richtige“ oder „falsche“ Antwort. Ihre persönliche Meinung zählt!

Bitte geben Sie nun Ihre Einschätzung des Produkts ab. Kreuzen Sie bitte nur einen Kreis pro Zeile an.

	1	2	3	4	5	6	7		
unerfreulich	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	erfreulich	1
unverständlich	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	verständlich	2
kreativ	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	phantasielos	3
leicht zu lernen	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	schwer zu lernen	4
wertvoll	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	minderwertig	5
langweilig	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	spannend	6
uninteressant	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	interessant	7
unberechenbar	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	voraussagbar	8
schnell	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	langsam	9
originell	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	konventionell	10
behindernd	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	unterstützend	11
gut	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	schlecht	12
kompliziert	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	einfach	13
abstoßend	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	anziehend	14
herkömmlich	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	neuartig	15
unangenehm	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	angenehm	16
sicher	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	unsicher	17
aktivierend	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	einschläfernd	18
erwartungskonform	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	nicht erwartungskonform	19
ineffizient	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	effizient	20
übersichtlich	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	verwirrend	21
unpragmatisch	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	pragmatisch	22
aufgeräumt	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	überladen	23
attraktiv	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	unattraktiv	24
sympathisch	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	unsympathisch	25
konservativ	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	innovativ	26

### B.3. Abschlussfragebogen

#### Abschlussfragebogen

War es im VR/AR Szenario von Vorteil von der Perspektive und Bewegung der anderen Person unabhängiger zu sein?

Die Unabhängigkeit war ...	---	--	-	-/+	+	++	+++	Die Unabhängigkeit war ...
Nicht von Vorteil	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Sehr vorteilhaft

#### Freie Kommentare

<hr/>
<hr/>
<hr/>
<hr/>
<hr/>
<hr/>

## C. Tabellen

p-Team ist der p-Wert aus dem t-Test in einem Team. Ex wird dabei der jeweilige Experte mit dem zugehörigen Lokalen Nutzer verglichen.

p-Rolle vergleicht gleiche Rollen. D.h. Es wird der Experte mit dem Experte aus dem anderen Szenario verglichen.

Abbildung C.1.: Erfahrung mit den Technologien

Technologie	Minimum	Q1	Median	Mittelwert	Q3	Maximum
VR	-3	-1.75	1	0.091	1	2
AR	-3	-3	-3	-2.091	-1	2

Abbildung C.2.: Fehler Anzahl bei den Versuchen

Szenario	Minimum	Q1	Median	Mittelwert	Q3	Maximum
VR	0	0	0	0.273	0.5	1
Video	0	0	0	0.818	1	3

Abbildung C.3.: Wie einfach war es den beschriebenen Stein zu finden?

Rolle	Minimum	Q1	Median	Mittelwert	Q3	Maximum	p-Team	p-Rolle
VR Experte	-1	0	1	0.727	2	2	-	0.691
AR Lokal	0	1.5	2	2.000	3	3	0.031	0.856
Video Experte	2	3	3	2.818	3	3	-	-
Video Lokal	-1	1	2	1.909	3	3	0.207	-

Abbildung C.4.: Wie einfach hat die Zusammenarbeit in dem Szenario geklappt?

Rolle	Minimum	Q1	Median	Mittelwert	Q3	Maximum	p-Team	p-Rolle
VR Experte	-1	2	2	2.000	2.5	3	0.640	0.866
AR Lokal	1	2	2	2.182	3	3	-	0.600
Video Experte	-1	1.5	2	1.909	3	3	1.000	-
Video Lokal	-1	2	2	1.909	3	3	-	-



Abbildung C.5.: Wie sehr waren sie von der sprachlichen Kommunikation abhängig?

Rolle	Minimum	Q1	Median	Mittelwert	Q3	Maximum	p-Team	p-Rolle
VR Experte	0	1	3	2.091	3	3	0.539	0.068
AR Lokal	0	1	2	1.818	2.5	3	-	0.004
Video Experte	2	3	3	2.818	3	3	0.557	-
Video Lokal	2	3	3	2.909	3	3	-	-

Abbildung C.6.: Ergebnisse des NASA-TLX

Subscales	Rolle	Minimum	Q1	Median	Mittelwert	Q3	Maximum	p-Team	p-Rolle
Geistige Anforderung	VR Experte	5	10	30	39.545	72.5	80	0.054	0.866
Geistige Anforderung	AR Lokal	5	10	25	27.273	27.5	80	-	0.324
Geistige Anforderung	Video Experte	15	35	50	49.545	70	90	0.417	-
Geistige Anforderung	Video Lokal	5	10	30	39.545	72.5	80	-	-
Körperliche Anforderun	VR Experte	5	15	30	40.455	75	80	0.327	0.025
Körperliche Anforderun	AR Lokal	5	20	30	33.182	45	70	-	0.481
Körperliche Anforderun	Video Experte	5	5	10	14.091	12.5	65	0.174	-
Körperliche Anforderun	Video Lokal	5	12.5	20	26.364	32.5	75	-	-
Zeitliche Anforderung	VR Experte	5	22.5	40	40.455	55	85	0.529	0.793
Zeitliche Anforderung	AR Lokal	5	12.5	20	32.727	62.5	65	-	0.334
Zeitliche Anforderung	Video Experte	10	25	30	37.727	50	85	0.529	-
Zeitliche Anforderung	Video Lokal	5	15	45	44.545	70	85	-	-
Leistung	VR Experte	5	17.5	25	26.818	35	65	0.039	0.672
Leistung	AR Lokal	5	7.5	10	15.000	17.5	45	-	0.232
Leistung	Video Experte	5	17.5	25	30.455	37.5	85	0.581	-
Leistung	Video Lokal	5	7.5	20	25.000	32.5	75	-	-
Anstrengung	VR Experte	5	17.5	35	39.091	62.5	85	0.266	0.311
Anstrengung	AR Lokal	5	10	20	28.182	35	75	-	0.311
Anstrengung	Video Experte	5	15	20	34.545	60	80	0.654	-
Anstrengung	Video Lokal	10	17.5	35	40.000	55	95	-	-
Frustration	VR Experte	5	12.5	25	29.091	42.5	60	0.020	0.550
Frustration	AR Lokal	5	7.5	10	14.545	15	45	-	0.120
Frustration	Video Experte	5	7.5	30	35.909	60	90	0.591	-
Frustration	Video Lokal	5	7.5	15	29.091	50	70	-	-
Gesamt	VR Experte	8.333	22.917	37.500	37.727	52.917	67.500	0.067	0.627
Gesamt	AR Lokal	7.500	15.833	20.000	25.152	30.833	60.000	-	0.311
Gesamt	Video Experte	13.333	20.833	21.667	33.712	48.750	65.000	0.967	-
Gesamt	Video Lokal	8.333	13.750	21.667	34.091	53.333	75.833	-	-

Abbildung C.7.: Ergebnisse des UEQ

Scale	Rolle	Mean	STD	N	Confidence	p-Team	p-Rolle
Attraktivität	VR Experte	1.44	0.80	11.00	0.47	0.18	0.01
Attraktivität	AR Lokal	1.98	0.77	11.00	0.45	-	0.00
Attraktivität	Video Experte	0.24	1.18	11.00	0.70	0.63	-
Attraktivität	Video Lokal	0.45	0.96	11.00	0.57	-	-
Durchschaubarkeit	VR Experte	1.52	0.72	11.00	0.43	0.04	0.31
Durchschaubarkeit	AR Lokal	2.20	0.60	11.00	0.35	-	0.45
Durchschaubarkeit	Video Experte	1.00	1.48	11.00	0.88	0.07	-
Durchschaubarkeit	Video Lokal	1.52	0.72	11.00	0.43	-	-
Effizienz	VR Experte	1.27	0.83	11.00	0.49	0.12	0.08
Effizienz	AR Lokal	1.89	0.90	11.00	0.53	-	0.06
Effizienz	Video Experte	0.43	1.27	11.00	0.75	0.11	-
Effizienz	Video Lokal	1.09	0.97	11.00	0.57	-	-
Steuerbarkeit	VR Experte	1.11	0.75	11.00	0.45	0.02	0.29
Steuerbarkeit	AR Lokal	1.98	0.72	11.00	0.43	-	0.08
Steuerbarkeit	Video Experte	0.61	1.31	11.00	0.77	0.02	-
Steuerbarkeit	Video Lokal	1.43	0.64	11.00	0.38	-	-
Stimulation	VR Experte	1.34	1.07	11.00	0.63	0.03	0.02
Stimulation	AR Lokal	2.05	0.57	11.00	0.34	-	0.00
Stimulation	Video Experte	0.00	1.33	11.00	0.78	0.49	-
Stimulation	Video Lokal	-0.41	1.16	11.00	0.68	-	-
Originalität	VR Experte	1.86	0.54	11.00	0.32	0.02	0.00
Originalität	AR Lokal	2.41	0.45	11.00	0.27	-	0.00
Originalität	Video Experte	-0.52	1.60	11.00	0.95	0.97	-
Originalität	Video Lokal	-0.55	1.68	11.00	0.99	-	-

Abbildung C.8.: War es im VR/AR Szenario von Vorteil von der Perspektive und Bewegung der anderen Person unabhängiger zu sein?

Minimum	Q1	Median	Mittelwert	Q3	Maximum
-3	1	2	1.591	3	3

Abbildung C.9.: Zeiten bei der Vorbereitung in Sekunden.

Gruppe	Szenario	Minimum	Q1	Median	Mittelwert	Q3	Maximum	Standartabweichung	p-Team
1	VR	4.647	7.2785	8.1745	8.5634	10.1935	13.091	2.607087485	-
1	Video	0.812	2.119	3.591	4.0442	6.4855	7.208	2.457542937	0.000866203
3	VR	4.332	4.556	5.3425	6.6891	9.1155	11.47	2.810251371	-
3	Video	4.668	5.914	7.836	7.7071	9.33575	11.299	2.204962003	0.38003371
4	VR	4.69	5.28775	7.226	8.5639	9.1025	21.244	4.931788079	-
4	Video	6.959	9.242	10.0775	10.8225	10.98675	17.163	3.021423662	0.235960886
6	VR	4.634	6.5075	9.5045	11.2449	15.41675	21.613	6.112222026	-
6	Video	6.157	12.236	16.6605	19.1987	23.24475	49.982	12.52046995	0.094131498
7	VR	4.522	9.83975	11.53	13.2071	14.0455	34.973	8.237014109	-
7	Video	5.376	6.31275	6.602	7.5931	9.1185	11.864	2.134926616	0.062970298
8	VR	5.662	8.575	10.314	11.9318	16.54475	18.563	4.738403688	-
8	Video	7.081	8.631	9.2835	10.3164	10.3525	20.119	3.68294448	0.406514676
9	VR	4.736	7.28	9.5105	13.5017	16.37975	40.222	10.79993668	-
9	Video	2.864	4.926	8.3935	7.8274	10.561	11.764	3.301271412	0.141259382
10	VR	6.287	7.4315	8.041	8.9101	9.19525	14.642	2.547110583	-
10	Video	7.132	9.0945	15.692	19.5247	23.97475	45.191	13.92244623	0.040148078
11	VR	7.584	9.675	11.878	11.2807	12.6965	14.711	2.259088855	-
11	Video	7.614	8.99775	9.968	12.6659	15.21725	24.077	5.320344276	0.463001145
12	VR	4.143	6.44375	6.9975	7.662	8.33025	14.731	3.103307017	-
12	Video	4.775	6.48225	9.873	10.3623	13.85375	18.792	4.659484307	0.147111184
13	VR	3.416	5.7925	6.5775	7.3094	8.92075	11.681	2.804479243	-
13	Video	6.503	8.46725	10.459	11.1113	12.68775	19.016	3.678445023	0.018824728
Gesamt	VR	3.416	6.544	8.482	9.896736364	11.67025	40.222	5.580897622	-
Gesamt	Video	0.812	6.8465	9.3055	11.01578182	12.106	49.982	7.670711416	0.217452736

Abbildung C.10.: Zeiten bei der Kommunikation in Sekunden.

Gruppe	Szenario	Minimum	Q1	Median	Mittelwert	Q3	Maximum	Standartabweichung	p-Team
1	VR	4.647	10.5585	13.961	14.1607	16.31075	28.973	6.398871255	-
1	Video	1.461	8.79925	11.2665	12.2253	14.40925	21.814	6.270279881	0.503214913
3	VR	4.545	7.51375	7.93	8.9842	10.415	16.81	3.35447942	-
3	Video	5.187	8.85825	10.1075	9.942	11.026	13.248	2.281522591	0.466230029
4	VR	6.991	9.152	10.7435	13.0017	14.04825	27.164	6.498309063	-
4	Video	10.059	12.919	17.43	17.5238	21.974	25.422	5.743086797	0.116767182
6	VR	6.97	7.91475	11.2635	11.9977	12.401	26.929	5.890580599	-
6	Video	8.2	13.4695	16.9845	16.8248	19.7855	26.037	5.444902444	0.073267611
7	VR	4.522	9.40125	10.626	12.8108	14.30275	31.14	7.280163609	-
7	Video	5.706	9.70325	14.5715	13.566	15.4515	24.172	5.597075089	0.797961079
8	VR	6.824	7.2485	8.627	10.6568	11.15375	24.539	5.444113922	-
8	Video	7.856	12.249	13.909	13.9998	16.4155	18.974	3.435753509	0.12109889
9	VR	7.494	8.88475	10.09	12.0637	14.49375	23.614	5.046441498	-
9	Video	5.85	9.15575	10.9085	11.0296	11.74875	18.23	3.688578383	0.607830407
10	VR	7.336	11.94225	12.369	13.2102	14.6785	22.227	4.0195095	-
10	Video	8.17	8.7535	10.1115	11.1464	13.164	15.43	2.804042685	0.201543374
11	VR	8.588	9.1975	10.5985	13.2585	15.6425	22.809	5.499219808	-
11	Video	10.327	13.7345	15.7525	16.5527	18.88075	25.449	4.739832628	0.168838506
12	VR	7.538	9.075	11.9725	11.849	13.643	19.412	3.691417403	-
12	Video	3.742	7.82675	12.0995	11.5575	15.949	18.917	5.408318351	0.889815926
13	VR	3.416	10.20175	12.3745	12.5982	14.81675	23.008	5.310189446	-
13	Video	5.409	5.92375	7.7985	8.2756	9.8715	12.871	2.733325252	0.038846578
Gesamt	VR	3.416	8.64675	11.0115	12.23559091	14.731	31.14	5.358344542	-
Gesamt	Video	1.461	8.98825	12.4885	12.96759091	15.9685	26.037	5.233443425	0.306500858

## D. Alternative Plots

Abbildung D.11.: Vorbereitungszeit des Experten als Boxplot

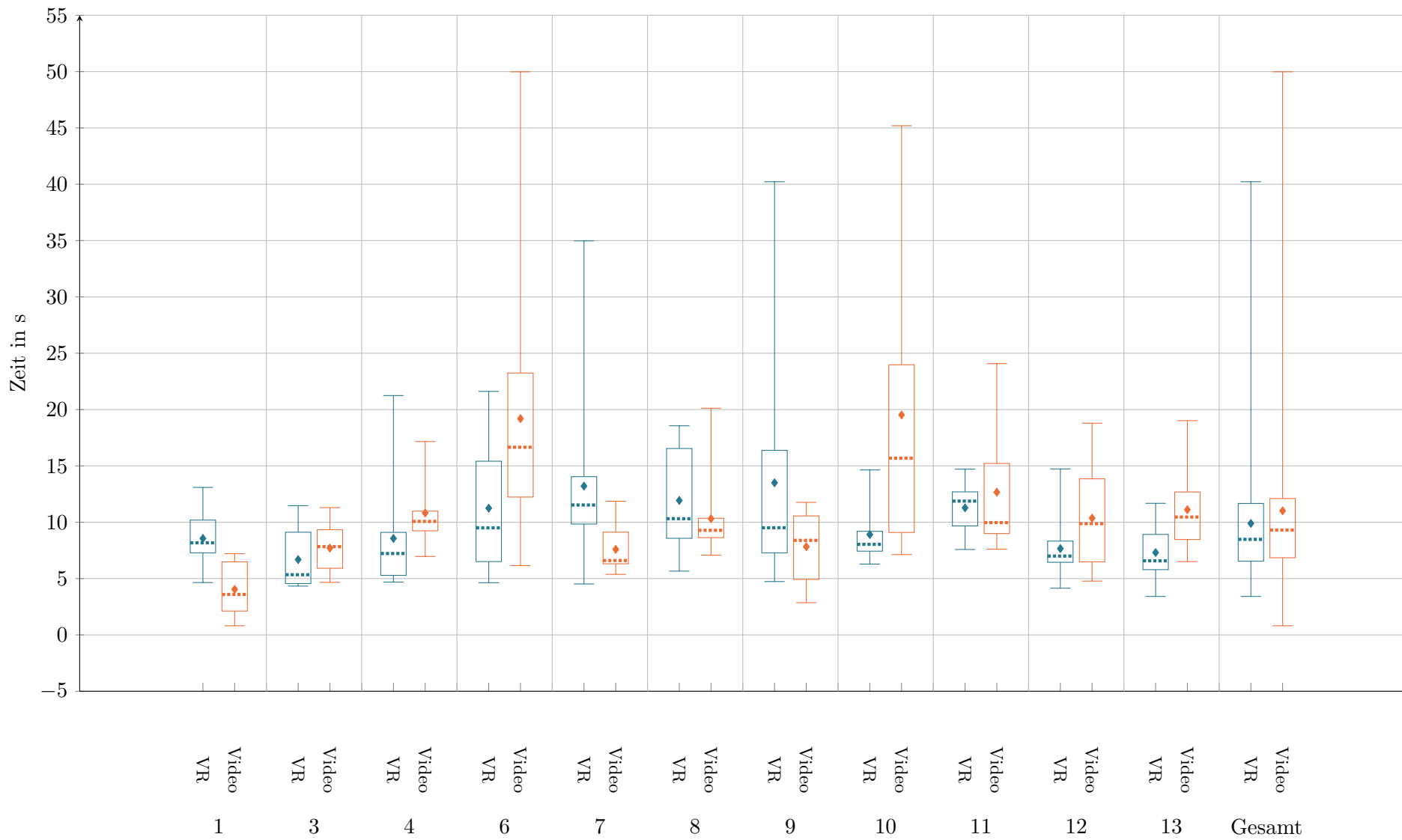
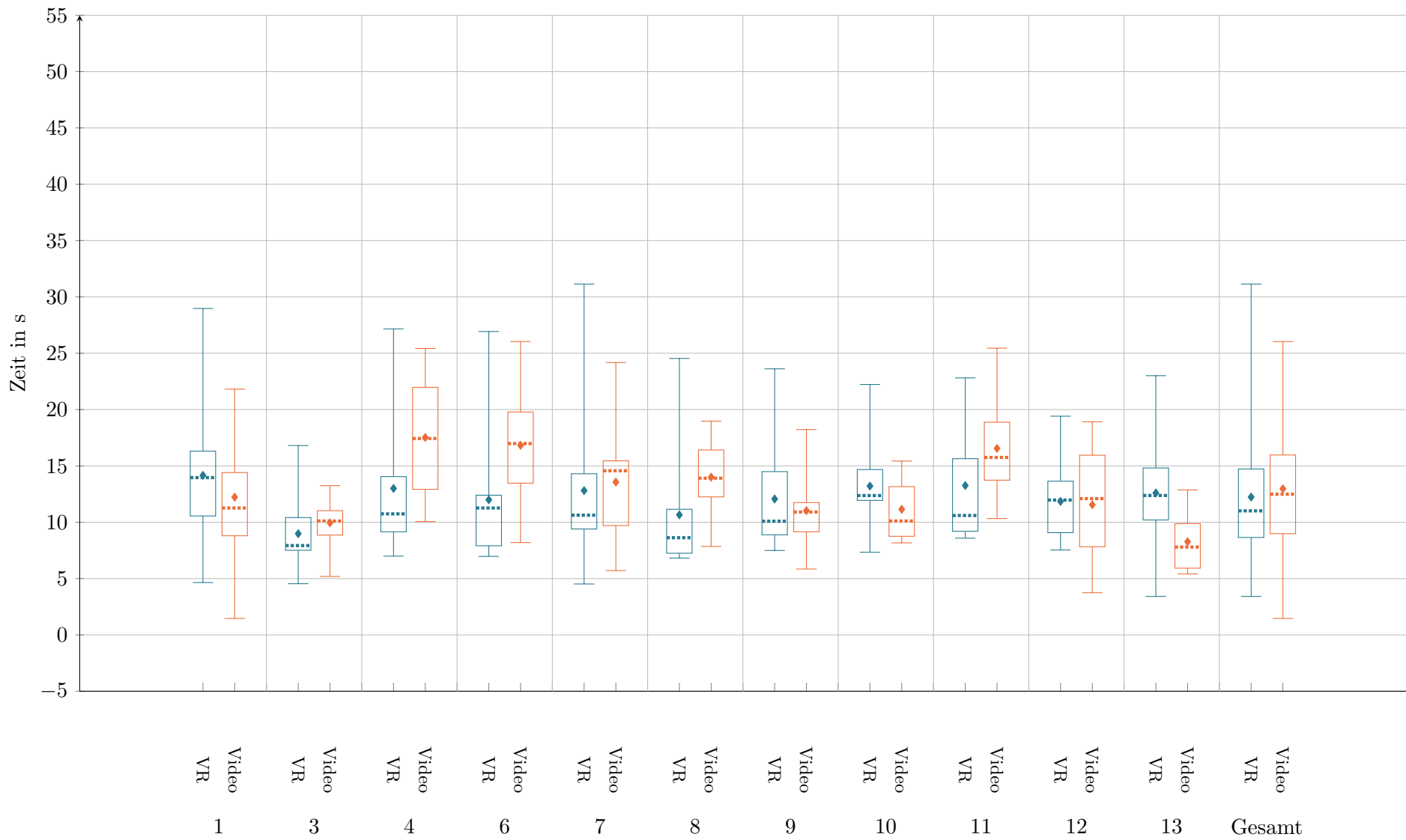


Abbildung D.12.: Kommunikationszeit





## **E. Datenträger**

Auf dem beigefügten Datenträger befinden sich:

- Masterarbeit: PDF und Latex Quellcode
- Die entwickelte Unreal Engine Anwendung
- Die entwickelte Software zum Aufnehmen der Punktwolke
- Evaluation: Fragebögen und Auswertung



# Literaturverzeichnis

- [3DS] *3d scan von der microsoft corporation.* <https://www.microsoft.com/de-de/store/p/3d-scan/9nblggh68pmc>. Accessed: 2017-11-26.
- [3DT] *3d tiles spezifikation.* <https://github.com/AnalyticalGraphicsInc/3d-tiles>. Accessed: 2017-09-13.
- [aug] *augmented realitys.* <http://whatis.techtarget.com/definition/augmented-reality-AR>. Accessed: 2017-11-26.
- [BCJ10] Sébastien Bottecchia, Jean Marc Cieutat, and Jean Pierre Jessel: *T.a.c: Augmented reality system for collaborative tele-assistance in the field of maintenance through internet.* In *Proceedings of the 1st Augmented Human International Conference, AH '10*, pages 14:1–14:7, New York, NY, USA, 2010. ACM, ISBN 978-1-60558-825-4. <http://doi.acm.org/10.1145/1785455.1785469>.
- [BKS99] M. Bauer, G. Kortuem, and Z. Segall: *"where are you pointing at?" a study of remote collaboration in a wearable videoconference system.* In *Digest of Papers. Third International Symposium on Wearable Computers*, pages 151–158, Oct 1999.
- [BL06] K. H. Bae and D. D. Lichti: *Automated registration of unorganized point clouds from terrestrial laser scanners.* pages 222–227, 2006.
- [BM92] P. J. Besl and N. D. McKay: *A method for registration of 3-d shapes.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 14(2):239–256, Feb 1992, ISSN 0162-8828.
- [Bro82] D. Broderick: *The Judas Mandala.* Pocket science fiction. Pocket Books, 1982, ISBN 9780671450328. <https://books.google.de/books?id=f2ZHAAAACAAJ>.
- [Bro99] F. P. Brooks: *What's real about virtual reality?* IEEE Computer Graphics and Applications, 19(6):16–27, Nov 1999, ISSN 0272-1716.
- [CNZHB08] J. Chastine, K. Nagel, Ying Zhu, and M. Hudachek-Buswell: *Studies on the effectiveness of virtual pointers in collaborative augmented reality.* In *Proceedings of the 2008 IEEE Symposium on 3D User Interfaces, 3DUI '08*, pages 117–124, Washington, DC, USA, 2008. IEEE Computer Society, ISBN 978-1-4244-2047-6. <http://dx.doi.org/10.1109/3DUI.2008.4476601>.
- [day] *Google daydream.* <https://vr.google.com/>. Accessed: 2017-11-26.
- [dSZ99] Antonino Gomes de Sá and Gabriel Zachmann: *Virtual reality as a tool for verification of assembly and maintenance processes.* Computers & Graphics, 23(3):389 – 403, 1999, ISSN 0097-8493. <http://www.sciencedirect.com/science/article/pii/S0097849399000473>.

- [GLT] *Gltf spezifikation*. <https://github.com/KhronosGroup/gltf>. Accessed: 2017-09-13.
- [Har06] Sandra G. Hart: *Nasa-task load index (nasa-tlx); 20 years later*. Proceedings of the Human Factors and Ergonomics Society Annual Meeting, 50(9):904–908, 2006. <https://doi.org/10.1177/154193120605000909>.
- [Hei00] M. Heim: *Virtual Realism*. Oxford University Press, 2000, ISBN 9780195350098. <https://books.google.de/books?id=4nyKP2-KlXAC>.
- [Hola] *Hololens*. <https://www.microsoft.com/de-de/hololens>. Accessed: 2017-11-26.
- [Holb] *What’s inside microsoft’s hololens and how it works*. <http://www.tomshardware.com/news/microsoft-hololens-components-hpu-28nm,32546.html>. Accessed: 2017-11-26.
- [Jit] *Jittertester*. [https://www.reddit.com/r/Vive/comments/4n4y6x/test\\_your\\_vive\\_jittershaking\\_before\\_its\\_too\\_late/](https://www.reddit.com/r/Vive/comments/4n4y6x/test_your_vive_jittershaking_before_its_too_late/). Accessed: 2017-11-26.
- [JK97] A. E. Johnson and Sing Bing Kang: *Registration and integration of textured 3-d data*. In *Proceedings. International Conference on Recent Advances in 3-D Digital Imaging and Modeling (Cat. No.97TB100134)*, pages 234–241, May 1997.
- [Kina] *Kinect dokumentation koordinatensysteme*. <https://msdn.microsoft.com/de-de/library/dn785530.aspx>. Accessed: 2017-11-02.
- [Kinb] *Kinect sdk*. <https://developer.microsoft.com/de-de/windows/kinect>. Accessed: 2017-11-26.
- [Kinc] *Kinect tiefensensor position*. <https://social.msdn.microsoft.com/Forums/sqlserver/ja-JP/05a6d2b3-9096-4236-b77a-691c5f047066/kinect-for-windows-v2-?forum=windowsgeneraldevelopmentissuesja>. Accessed: 2017-11-02.
- [KSK<sup>+</sup>04] T. Kurata, N. Sakata, M. Kouroggi, H. Kuzuoka, and M. Billinghurst: *Remote collaboration using a shoulder-worn active camera/laser*. In *Eighth International Symposium on Wearable Computers*, volume 1, pages 62–69, Oct 2004.
- [Kuz92] Hideaki Kuzuoka: *Spatial workspace collaboration: A sharedview video support system for remote collaboration capability*. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI ’92, pages 533–540, New York, NY, USA, 1992. ACM, ISBN 0-89791-513-5. <http://doi.acm.org/10.1145/142750.142980>.
- [lig] *Lighthouse tracking examined*. <http://doc-ok.org/?p=1478>. Accessed: 2017-11-02.
- [LM] Wolfgang Ludwig-Mayerhofer: *Signifikanztests (so) kurz (wie möglich) erklärt*. [https://www.uni-siegen.de/phil/sozialwissenschaften/soziologie/mitarbeiter/ludwig-mayerhofer/statistik/statistik\\_downloads/signifikanztests.pdf](https://www.uni-siegen.de/phil/sozialwissenschaften/soziologie/mitarbeiter/ludwig-mayerhofer/statistik/statistik_downloads/signifikanztests.pdf).
- [LSCG13] Joel Lanir, Ran Stone, Benjamin Cohen, and Pavel Gurevich: *Ownership and control of point of view in remote assistance*. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI ’13, pages 2243–2252, New York, NY, USA, 2013. ACM, ISBN 978-1-4503-1899-0. <http://doi.acm.org/10.1145/2470654.2481309>.

- [MCS14] Manuel Martin, Florian van de Camp, and Rainer Stiefelhagen: *Real time head model creation and head pose estimation on consumer depth cameras*. In *Proceedings of the 2014 2Nd International Conference on 3D Vision - Volume 01*, 3DV '14, pages 641–648, Washington, DC, USA, 2014. IEEE Computer Society, ISBN 978-1-4799-7000-1. <http://dx.doi.org/10.1109/3DV.2014.54>.
- [Min95] Mark Mine: *Virtual environment interaction techniques*. Technical report, UNC Chapel Hill CS Dept, 1995.
- [MTUK95] Paul Milgram, Haruo Takemura, Akira Utsumi, and Fumio Kishino: *Augmented reality: a class of displays on the reality-virtuality continuum*, 1995. <http://dx.doi.org/10.1117/12.197321>.
- [NB] Instr Gökhan Nalbant and Instr Barbaros Bostan.
- [occ] *Oculus rift*. <https://www.oculus.com/rift/?> Accessed: 2017-11-26.
- [OES<sup>+</sup>15] Ohan Oda, Carmine Elvezio, Mengü Sukan, Steven Feiner, and Barbara Tversky: *Virtual replicas for remote assistance in virtual and augmented reality*. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, UIST '15, pages 405–415, New York, NY, USA, 2015. ACM, ISBN 978-1-4503-3779-3. <http://doi.acm.org/10.1145/2807442.2807497>.
- [Pla] *Playstation vr*. <https://www.playstation.com/de-de/explore/playstation-vr/>. Accessed: 2017-11-26.
- [Pok] *Pokemon go*. <https://www.pokemongo.com/de-de/>. Accessed: 2017-11-26.
- [RBMB08] R. B. Rusu, N. Blodow, Z. C. Marton, and M. Beetz: *Aligning point cloud views using persistent feature histograms*. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3384–3391, Sept 2008.
- [sam] *Samsung gear vr*. <http://www.samsung.com/de/wearables/gear-vr-r323/>. Accessed: 2017-11-26.
- [SLW02] G. C. Sharp, S. W. Lee, and D. K. Wehe: *Icp registration using invariant features*. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):90–102, Jan 2002, ISSN 0162-8828.
- [TAH12] Franco Tecchia, Leila Alem, and Weidong Huang: *3d helping hands: A gesture based mr system for remote collaboration*. In *Proceedings of the 11th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry*, VRCAI '12, pages 323–328, New York, NY, USA, 2012. ACM, ISBN 978-1-4503-1825-9. <http://doi.acm.org/10.1145/2407516.2407590>.
- [UE4a] *Unreal engien hololens template*. <https://github.com/ProteusVR/Hololens><https://www.youtube.com/watch?v=KxvAm2qNJ0Q&feature=youtu.be>. Accessed: 2017-11-02.
- [UE4b] *Unreal engine 4*. <https://www.unrealengine.com/en-US/what-is-unreal-engine-4>. Accessed: 2017-11-02.
- [UEQ] *User experience questionnaire (ueq)*. <http://www.ueq-online.org/>. Accessed: 2017-11-26.
- [uni] *Unity*. <https://unity3d.com/de>. Accessed: 2017-11-26.

- [viv] *Htc vive*. <https://www.vive.com/de/>. Accessed: 2017-11-26.
- [VNG<sup>+</sup>02] Jeenal Vora, Santosh Nair, Anand K Gramopadhye, Andrew T Duchowski, Brian J Melloy, and Barbara Kanki: *Using virtual reality technology for aircraft visual inspection training: presence and comparison studies*. *Applied Ergonomics*, 33(6):559 – 570, 2002, ISSN 0003-6870. <http://www.sciencedirect.com/science/article/pii/S000368700200039X>.
- [vrB] *Virtual reality best practices*. <https://docs.unrealengine.com/latest/INT/Platforms/VR/ContentSetup/#vrandsimulationsickness>. Accessed: 2017-11-26.

---

ch versichere wahrheitsgemäß, die Arbeit selbstständig angefertigt, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderungen entnommen wurde.

**Karlsruhe, 30.11.2017**

.....  
(Kai Westerkamp)