IB-RM-OP-2016-201

Entwicklung eines intuitiven Bedienkonzepts für Assistenzroboter unter Verwendung von Augmented Reality und Gestensteuerung

Tamara von Sawitzky

DLR Deutsches Zentrum für Luft- und Raumfahrt

BACHELORARBEIT

ENTWICKLUNG EINES INTUITIVEN BEDIENKONZEPTS FÜR ASSISTENZROBOTER UNTER VERWENDUNG VON AUGMENTED REALITY UND GESTENSTEUERUNG

Freigabe:	Der Bearbeiter:	Unterschriften
	Tamara von Sawitzky	TvSawitzky
	Betreuer:	
	Christian Nißler	C. Nißler
	Der Institutsdirektor	
	Dr. Alin Albu-Schäffer	A. Albu-Schäffer

Dieser Bericht enthält 70 Seiten, 32 Abbildungen und 4 Tabellen

Ort: Oberpfaffenhofen	Datum: 14.09.2016	Bearbeiter: Tamara von Sawitzky	Zeichen:

Deutsches Zentrum für Luft- und Raumfahrt

Technische Hochschule Ingolstadt

Entwicklung eines intuitiven Bedienkonzepts für Assistenzroboter unter Verwendung von Augmented Reality und Gestensteuerung

Bachelorarbeit

zur Erlangung des akademischen Grades
Bachelor of Engineering

Technische Hochschule Ingolstadt
Fakultät Elektrotechnik und Informatik
Studiengang Mechatronik

Verfasser:	Tamara von Sawitzky
Betreuer am DLR:	Dipl.-Ing. Christian Nißler Dipl.-Ing. Jörn Vogel Dipl.-Ing. Roman Weitschat
Erstprüfer:	Prof. Dr. rer. nat. Thomas Grauschopf
Zweitprüfer:	Prof. Dr.-Ing. Ulrich Schmidt
Ausgabedatum:	25.07.2016
Abgabedatum:	20.09.2016

Eidesstaatliche Erklärung

Ich erkläre hiermit, dass ich diese Arbeit selbstständig verfasst, noch nicht anderweitig für Prüfungszwecke vorgelegt, keine anderen als die angegebenen Quellen oder Hilfsmittel benutzt sowie wörtliche und sinngemäße Zitate als solche gekennzeichnet habe.

Ingolstadt, den 20.09.2016 _____

Tamara von Sawitzky

Inhaltsverzeichnis

1 Einleitung

In der Industrie werden Roboter zur Entlastung der Arbeiter bei der Verrichtung von monotoner und schwerer körperlicher Arbeit verwendet. Diese Roboter befinden sich hinter einer technischen oder organisatorischen Schutzvorrichtung, welche den Zutritt zu den Industrierobotern verhindert, um schwere körperliche Verletzungen der Arbeiter vorzubeugen. Es wird jedoch angestrebt, eine Zusammenarbeit zwischen Roboter und Mensch zu ermöglichen [DIN EN ISO 10218-1:2012-01]. Hierfür muss eine sichere Zusammenarbeit zwischen Mensch und Maschine gewährleistet sein. Um dies zu erreichen, weicht die bisherige physikalische Schutzvorrichtung einer Schutzbarriere, die über Sensoren verwirklicht wird.

Für jeden Anwendungsfall einer Mensch-Roboter-Interaktion ist eines der Hauptprobleme die Kommunikation zwischen Mensch und Maschine. Hierzu zählt zum einen, wie der Mensch einem Roboter Befehle mitteilen kann, zum anderen, auf welche Weise der Mensch eine Antwort von einem Roboter erhält. In aktuellen Anwendungen sind meist periphere Geräte im Einsatz, wie zum Beispiel Tastaturen, Computermäuse und Joysticks, die eine Interaktion zwischen Mensch und Maschine ermöglichen.

Es gibt jedoch immer mehr Anwendungsfälle, bei denen eine freihändige Interaktion mit einem Roboter wünschenswert oder auch unumgänglich ist. Dazu zählen Anwendungsszenarien, in denen ein Mensch keine Hand frei hat, um herkömmliche Eingabegeräte zu bedienen und somit eine neue Form der Kommunikation benötigt. Darunter fallen beispielsweise Arbeiter, die mit einer Aufgabe beschäftigt sind, und somit ihre Hände nicht für andere Aufgaben frei haben, z.B. im Industriebereich. Menschen mit Einschränkungen der oberen Extremitäten können alltägliche Aufgaben, wie beispielsweise das Greifen eines Wasserglases, nicht mehr bewältigen und können durch Assistenzroboter unterstützt werden.

Dabei stellt sich jedoch die Frage, wie diese Menschen mit einem Roboter kommunizieren können, wenn bisherige Systeme, wie Joystick und Tastaturen, nicht verwendet werden können oder sollen.

1.1 Problemstellung

In dieser Arbeit soll eine freihändige Interaktion mit einem Roboter umgesetzt werden. Mit dieser soll eine berührungslose Mensch-Maschine-Interaktion für Personen im Industrieumfeld ermöglicht werden, sowie im Medizinbereich Personen mit Einschränkungen der oberen Extremitäten eine Möglichkeit gegeben werden, einen im Alltag unterstützenden Assistenzroboter zu bedienen.

Für die Umsetzung soll eine Datenbrille verwendet werden. Mithilfe von Augmented Reality und Gestensteuerung sollen verschiedene Objekte ausgewählt und anschließend von einem Roboterarm aufgenommen werden. Die Objekte sollen hierauf dem Träger der Brille gereicht werden.

Die Verwendung von Augmented Reality erlaubt eine kamerabasierte Erkennung und Tracking von Objekten. Die Gestenerkennung soll über die in der Datenbrille integrierte Sensorik erfolgen, mit welcher Kopfgesten detektiert werden können. Dieser Ansatz soll den Einsatz von Händen für die Mensch-Roboter-Interaktion verzichtbar machen, und eine freihändige Interaktion mit einem Roboter ermöglichen.

Abbildung 1.1 zeigt einen Anwendungsfall, bei dem mit einer Datenbrille Kisten mit Kleinteilen ausgewählt werden können, die ein Arbeiter für ein Bauteil benötigt, welches er in einer Kooperation mit einem Roboter fertigt.

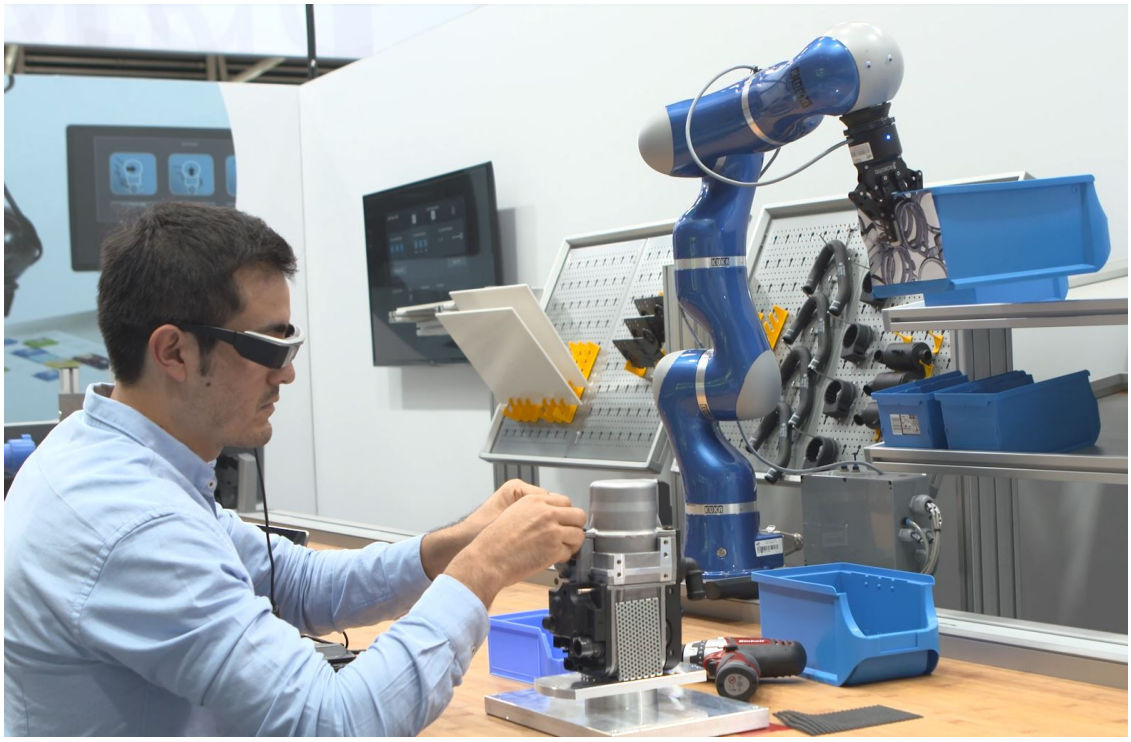

Abbildung 1.1: Anwendungsszenario Industrie, Quelle: DLR

1.2 Struktur der Arbeit

Zu Beginn der Arbeit wird in Kapitel 2 auf die Grundlagen in den Bereichen *Augmented Reality* und *Mensch-Maschine-Interaktion* eingegangen. In Kapitel 3 sind die Anforderungen an die Anwendung definiert. Kapitel 4 befasst sich mit der Umsetzung der Anwendung. Das Hervorheben von Objekten mittels Augmented Reality ist in Abschnitt 4.1 beschrieben. Abschnitt 4.2 befasst sich mit der Erkennung von Gesten, über die Objekte aus- und abgewählt

werden können. Die Realisierung der Kommunikation mit einem Roboter ist in Abschnitt 4.3 ausgeführt. Abschnitt 4.4 führt die einzelnen Komponenten zu einer Gesamtanwendung zusammen. Kapitel 5 befasst sich mit der Auswertung der in Kapitel 3 gestellten Anforderungen. Kapitel 6 schließt die Arbeit mit einer Zusammenfassung und einem Ausblick auf zukünftige Chancen und Probleme von Anwendungen mit Datenbrillen ab.

2 Grundlagen

In diesem Kapitel sind die für diese Arbeit notwendigen Grundlagen zur Augmented Reality und zur Mensch-Maschine-Interaktion beschrieben. Augmented Reality wird in dieser Arbeit für die Hervorhebung von Objekten verwendet. Das Unterkapitel zur Mensch-Maschine-Interaktion stellt aktuell verwendete Methoden zur Kommunikation zwischen Mensch und Maschine dar.

2.1 Augmented Reality

In ihrer einfachsten Form ist Augmented Reality (AR) – zu Deutsch *„Erweiterte Realität“* – die Kunst der Überlagerung von Computergrafiken über einen Livevideostream der realen Welt [Madden, 2011; Zhou et al., 2008].

Laut Azuma [1997] sind AR-Verfahren so definiert, dass sie reale und virtuelle Darstellungen kombinieren, in Echtzeit interaktiv sind und die virtuellen Darstellungen mit der realen Umgebung registrieren.

Diese Definition lässt sich nach Madden [2011] noch weiter auslegen: Augmented Reality ist das Tracken von Objekten in Echtzeit, das Erkennen von Bildern und Objekten, sowie das Anzeigen von Kontext oder anderen Daten in Echtzeit.

Es existieren jedoch unterschiedliche Definitionen von AR. Ist die Liveübertragung eines Sportevents auch dem Bereich AR zuzuordnen, wenn Spielerstatistiken während des Spiels eingeblendet werden? Oder sind Anzeigen über Batteriestatus und die Fotoanzahl einer Kamera schon AR? In beiden Fällen werden zusätzliche Informationen eingeblendet. Es ist umstritten, ob dies valide Beispiele von Augmented Reality sind, da oft Echtzeittracking als zwingende Bedingung für AR angesehen wird. [Madden, 2011]

2.1.1 Verbreitete Verfahren

Madden [2011] beschreibt einige allgemeingültigen AR-Verfahren.

Zum einen gibt es die **gravimetrische AR**, die typischerweise als *Browser* bezeichnet wird. Diese Art von AR stützt sich auf das Gravimeter eines Geräts, mit dem die Position und Orientierung des Nutzers berechnet wird. Hier werden Informationen über Objekte auf dem Kamerabild eingeblendet, jedoch sind diese abhängig vom Ort und nicht von dem aktuellen Bild der Kamera. Bekannte Beispiele für Browser sind *junaio*, *Layar* und *Wikitude*.

Ein weiterer Ansatz sind **Referenzmarker**, mittels derer Objekte der realen Welt getrackt werden können. Die Marker treten meist in Form einer Kombination aus schwarz und weiß kodierten Quadraten auf, siehe Abbildung 2.1. Sobald ein Marker erkannt wird, wird eine Aktion ausgeführt, die zum Beispiel ein 3D-Objekt mit einer Kontur überlagert.

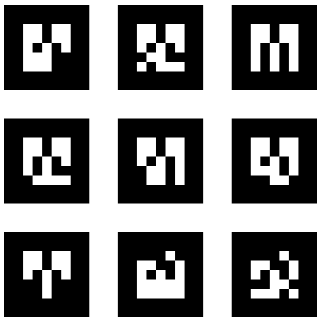

Abbildung 2.1: *AprilTags* als Beispiel für Marker [Olson, 2011]

Marker können als eine natürliche Weiterentwicklung der **Barcodes** gesehen werden. Wenn ein Barcode mithilfe eines Geräts, beispielsweise einem Smartphone, eingescannt wird, werden dazugehörige Informationen auf diesem Gerät ausgegeben.

Weit bekannt sind **Quick Response (QR)** Codes, welche verwendet werden, um mithilfe eines QR-Scanners direkt auf URLs weiterzuleiten. Im Vergleich zu Markern, bei denen jeder einer bestimmten Anwendung zugeordnet ist, enthalten QR-Codes die Daten bereits in ihrem Muster. Die Codes folgen einem ISO Standard, um von verschiedensten QR-Scannern entschlüsselt werden zu können. Wenn ein QR-Code auf eine veränderte URL zeigen soll, muss dafür ein neuer generiert werden.

Bei der **markerlosen AR** wird AR entweder ohne Tracking, oder aber mit Tracking, ohne einen spezifischen Marker genutzt. Gravimetrische Browser sind ein Beispiel für markerlose AR, da Orte von Interesse und andere Daten im Kamerabild angezeigt werden. Viele bezeichnen gravimetrische Browser nicht als AR, da die Daten nur über dem Kamerabild eingeblendet, aber nicht an diesem ausgerichtet sind.

Markerloses Tracking wiederum trackt Objekte, ohne spezielle Marker zu verwenden. Ein bekanntes Beispiel dafür ist das Face Tracking.

2.1.2 Abgrenzung von Virtual Reality

Um virtuelle Realität (VR) zu erfahren, sind spezielle Geräte erforderlich. Dagegen benötigt AR nur eine Möglichkeit die aktuelle Umgebung aufzunehmen, und die Mittel die Computerwelt in Form von eingeblendeten Objekten darzustellen. Während unter VR die reale Umwelt ausgeschaltet wird, um in eine virtuelle, neu geschaffene Umgebung einzutauchen, reichert

AR die reale Welt um graphische Objekte an und ergänzt somit nur die vorhandene Realität. Außerdem ermöglicht sie dem Nutzer, mit den virtuellen Bildern und den realen Objekten nahtlos zu interagieren. [Fischer und Klein, 2009; Madden, 2011]

Eine weitere Abgrenzung kann durch das von Milgram et al. [1995] entwickelte „Realitäts-Virtualitäts-Kontinuum“ vorgenommen werden (Abbildung 2.2).

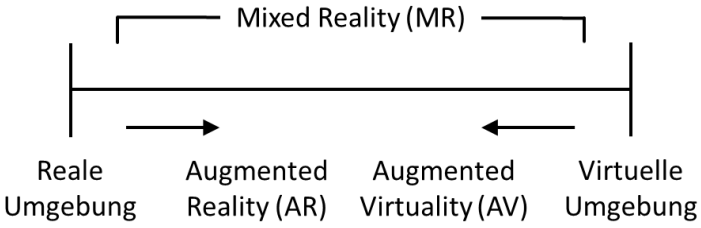

Abbildung 2.2: Realitäts-Virtualitäts-Kontinuum nach Milgram et al. [1995]

Der linke Bereich des Kontinuums, die *reale Umgebung*, umfasst alle realen Objekte, die von einer Person betrachtetet werden. Der rechte Bereich, die *virtuelle Umgebung*, definiert nur virtuelle Objekte, beispielsweise einer Computerspiel-Simulation. Der Bereich zwischen den Extrempunkten des Kontinuums wird als *Mixed Reality (MR)* bezeichnet. Darin werden virtuelle und reale Objekt in beliebiger Weise miteinander kombiniert. Bei der *Augmented Reality* überwiegt der reale Anteil, wobei bei der weniger verbreiteten *Augmented Virtuality (AV)* der virtuelle Anteil dominiert.

2.1.3 Trackingverfahren

Zhou et al. [2008] analysierten die Entwicklung der AR-Forschung im Hinblick auf Trackingtrends über die Jahre 1998 bis 2008. Ausgehend von dieser Analyse wird auf zukünftige Forschungsbereiche, sowie auf die Begrenzungen der AR eingegangen.

Die Veröffentlichung befasst sich spezifisch mit Tracking, Interaktion und Darstellung, da diese laut Zhou et al. [2008] die Bereiche der AR-Forschung mit der stärksten Entwicklung darstellen.

2.1.3.1 Sensorbasierte Trackingverfahren

Diese Trackingverfahren basieren auf magnetischen, akustischen, Beschleunigungs-, gyroskopischen, optischen und/oder mechanischen Sensoren. Ein ausführlicher Überblick über die verschiedenen Ansätze ist in Rolland et al. [2001] zu finden. Es wurde ebenfalls die Kombination von verschiedenen Sensoren erforscht, um ein robusteres Tracking zu ermöglichen.

2.1.3.2 Computer Vision-basierte Trackingverfahren

Sensorbasierte Trackingverfahren verhalten sich analog zu offenen Regelkreisen, bei denen die Ausgaben fehlerbehaftet sind. Computer Vision-basierte Verfahren jedoch verwenden Methoden der Bildverarbeitung, um die Lage der Kamera relativ zu realen Objekten zu berechnen. Somit verhalten sie sich analog zu geschlossenen Regelkreisen, welche den auftretenden Fehler dynamisch korrigieren. [Bajura und Neumann, 1995]

Die Computer Vision-Trackingverfahren können in *merkmalbasiert* und *modellbasiert* unterteilt werden [Pressigout und Marchand, 2006].

Merkmalbasierte Verfahren

Merkmalbasierten Verfahren liegt das Finden von Korrespondenzen zwischen 3D-Weltkoordinaten und 2D-Bildmerkmalen zugrunde. Aus diesen Informationen lässt sich die Lage der Kamera berechnen.

In den früheren Veröffentlichungen wurden oft künstliche Marker, wie zum Beispiel *April-Tags*, zur Bestimmung der Kameraposition verwendet. Am bekanntesten ist die *ARToolKit* Bibliothek [Kato und Billinghurst, 1999], welche auch heute noch häufig verwendet wird. Es wurden auch effiziente Methoden zum Finden von Linien [Stricker et al., 1999] und zum Minimieren von Fehlern beim markerbasierten Tracking [Comport et al., 2003] erforscht. Quadratische Marker stellen das dominante Verfahren dieser Zeit dar. Der zentrale Ansatz zur Kombination von Mustererkennung und Lageberechnung wurde bereits 1998 von Rekimoto publiziert. Andere Forscher wiederum verwendeten ringförmige Referenzpunkte [Cho et al., 1998], kreisförmige Marker [Vogt et al., 2002] oder 2D-Barcode Referenzpunkte [Naimark und Foxlin, 2002].

Die Kameraposition kann auch mithilfe von natürlichen Merkmalen anstelle von Markern berechnet werden. Unter den Begriff natürliche Merkmale fallen etwa Punkte, Linien, Ecken oder Konturen. Park et al. [1998] zeigen, wie natürliche Merkmale über künstliche Merkmale hinaus für das Tracking verwendet werden können. Sobald die Kameraposition aus künstlichen Merkmalen berechnet wurde, erfasst das System dynamisch zusätzliche natürliche Merkmale, welche zum regelmäßigen Aktualisieren der Position genutzt werden. Auf diese Weise ist es möglich, ein robusteres Tracking zu erreichen, auch wenn sich die ursprünglichen Referenzpunkte nicht mehr im Sichtfeld der Kamera befinden. Der Bereich der natürlichen Merkmale war der aktivste der Computer Vision-Tracking-Forschung im Zeitraum von 1998 bis 2008. [Zhou et al., 2008]

Modellbasierte Verfahren

Modelle, zum Beispiel CAD-Modelle oder 2D-Vorlagen, eines Objekts werden bei modellbasierten Verfahren verwendet. Das erste modellbasierte Verfahren, welches auf dem *International Symposium on Mixed and Augmented Reality (ISMAR)* präsentiert wurde, adaptiert einen Ansatz der Robotik, um die Kameraposition aus einer Vielfalt von Modellmerkmalen (Linien, Kreise, Zylinder und Sphären) zu berechnen [Comport et al., 2003]. Basierend auf den Linien und Ecken eines Modells konstruieren die Tracker ihr Modell. Ecken sind die Merkmale, die am häufigsten verwendet werden, da sie rechnerisch effizient und sehr robust gegenüber Änderungen in der Belichtung sind.

Über den Zeitraum von 1998 bis 2008 ist eine Entwicklung von markerbasierten Methoden zu natürlichen Merkmalen und modellbasierten Verfahren auszumachen.

2.1.3.3 Hybride Trackingverfahren

Als hybride Trackingverfahren bezeichnet man die Kombination von unterschiedlichen Trackingverfahren. Bei der Verwendung eines einzelnen Verfahrens, welches je nach Situation gute bis nicht verwertbare Ergebnisse liefern kann, können kurzzeitige Ausfälle des Tracking auftreten. Diese Ausfälle zu kompensieren ist das Ziel von hybriden Verfahren. [Dörner et al., 2014]

2.1.4 Displays

Ein wichtiger Punkt bei der Verwendung von Augmented Reality ist die Hardware, über welche die Anzeige der Augmentierungen erfolgt. Je nach Anwendungsfall können sogenannte Head-Mounted-Displays oder auch mobile Endgeräte, wie zum Beispiel Smartphones, verwendet werden.

2.1.4.1 Durchsicht-Head-Mounted-Displays

Durchsicht-Head-Mounted-Displays (HMDs) ermöglichen dem Nutzer die reale Welt zu erfassen. Durch optische oder virtuelle Technologien können virtuelle Objekte eingeblendet werden. Es gibt zwei Hauptkategorien: *optical see-through* (OST) und *video see through* (VST) HMDs.

OST-Displays ermöglichen das Sehen der Umgebung mit den „eigenen Augen“, in welche Graphiken in das Blickfeld des Nutzers eingeblendet werden. Bei VST-Displays wird dem Nutzer ein Livestream der Umgebung angezeigt, der mit graphischen Objekten überlagert wird.

Ein Vorteil dieser Displays ist zum einen die Konsistenz zwischen realer und synthetischer Sicht, zum anderen die Möglichkeit Bildverarbeitungsmethoden anzuwenden. Unter anderem sind Verdeckungsprobleme aufgrund verschiedener Bildverarbeitungsmethoden ein handhabbares Problem [Kiyokawa, 2008].

2.1.4.2 Projektionsbasierte Displays

Projektionsbasierte Displays werden verwendet, um zum Beispiel graphische Informationen direkt auf realen Objekten anzuzeigen. Somit eignen sich diese Displays für Applikationen, bei denen kein Nutzer ein Displaymedium tragen muss.

2.1.4.3 Mobile Endgeräte

Mobile Endgeräte sind eine Alternative zu HMDs. Es gibt eine Vielzahl dieser Geräte, die für AR geeignet sind. Tablet PCs, Handys, Smartphones und PDAs sind einige davon [Zhou et al., 2008]. Aufgrund ihrer Mobilität ermöglichen diese Geräte, AR-Anwendungen außerhalb von kontrollierten Umgebungen zu verwenden [Nóbrega et al., 2015].

2.1.5 Problemstellung: Registrierung, Tracking und Überlagerung

Grundlegend muss jedes AR-System folgende Probleme lösen: die Registrierung und das Tracking, sowie die Überlagerung mit virtuellen Objekten.

In der Registrierungsphase müssen die Positionen bestimmt werden, an denen zusätzliche Informationen angezeigt werden sollen. Dafür werden GPS (bei gravimetrischer AR) oder andere Merkmale (zum Beispiel *Natural Features*), wie in Abschnitt 2.1.3 beschrieben, verwendet. Nach der Registrierung müssen die 2D- oder 3D-Überlagerungen gerendert und richtig positioniert werden. Die Interaktion ist von der Art der Registrierung, sowie des verwendeten Geräts abhängig. [Nóbrega et al., 2015]

2.1.5.1 Registrierung und Tracking

Referenzmarker und QR-Codes besitzen sehr markante Eigenschaften, wie zum Beispiel ein binäres Farbsystem, einfache geometrische Formen und eine bekannte physikalische Größe. Vorteilhaft ist, dass Marker gut und stabil zu tracken sind. [Nóbrega et al., 2015]

In letzter Zeit werden *ARToolKit*-ähnliche Marker von Bildmarkern abgelöst [Tillon et al., 2011; Uchiyama und Marchand, 2011]. *Metaio* und *Vuforia* sind zwei Softwarebibliotheken, die beispielsweise für Smartphones verwendet werden können.

Aktuell entwickeln sich die AR-Systeme weg von Markern, hin zur Bildanalyse und Detektion von visuellen Elementen [Nóbrega et al., 2015].

2.1.5.2 Überlagerung mit virtuellen Objekten

Über einen Algorithmus zur Lageschätzung kann die Position und Orientierung des Markers und auch die Skalierung erfasst werden, woraufhin das virtuelle Objekt positioniert und skaliert werden kann [Myojin et al., 2012; Sukan et al., 2012].

Unter der Verwendung von natürlichen Merkmalen kann, ähnlich den Markern, unter anderem die Rotation und Translation des Objekts berechnet werden. Das virtuelle Objekt wird dann dementsprechend ausgerichtet.

Ziel ist es, die virtuellen Objekte so genau wie möglich in die reale Umgebung einzubetten.

2.2 Mensch-Maschine-Interaktion

Es gibt eine Vielzahl von Möglichkeiten für einen Menschen, eine Maschine zu bedienen. Dieser Bereich ist der Mensch-Maschine-Interaktion zuzuordnen. *Human-Machine-Interaction* (HMI) ist ein interdisziplinäres Feld, welches sich mit der Theorie, dem Design, der Implementierung und der Auswertung der Möglichkeiten, wie Menschen Geräte verwenden und mit ihnen interagieren, beschäftigt.

Die typischen Formen der Interaktion erfolgen über Eingabegeräte wie Maus, Tastatur, Bedienpanels und Joysticks [Schenk und Rigoll, 2010]. Seit dem Aufkommen von Smartphones und Tablets ist auch die Interaktion mittels Touchpad weit verbreitet.

Eine weitere, neuere Möglichkeit der HMI sind die sogenannten natürlichen Benutzerschnittstellen. Abschnitt 2.2.1 gibt einen Überblick über diese Art der Interaktion mit einer Maschine. Abschnitt 2.2.2 geht auf die Problemstellung bei der Gestenerkennung ein. Abschnitt 2.2.3 befasst sich mit den Herausforderungen von gestenbasierten HMIs.

2.2.1 Natural User Interfaces

Natural User Interfaces (NUI), zu Deutsch natürliche Benutzerschnittstellen, stellen natürliche Formen der Interaktion dar. Zu diesen zählen Berührung, Sprache oder Gesten. NUIs haben ein großes Potenzial, unsere Interaktion zu erweitern, besonders in den Fällen, bei denen die traditionellen Methoden ungeeignet oder umständlich sind. Sie bringen neue Probleme und neue Herausforderungen, sowie ein Potenzial für große Fehler und Verwirrung mit sich. Zugleich bergen sie große Vorteile und Potenziale. [Norman, 2010]

2.2.1.1 Berührungssteuerung

Spätestens seit dem Aufkommen von Smartphones und Tablets sind Touchpads eine gängige Art der Mensch-Maschine-Interaktion. Die Berührungssteuerung ermöglicht ein einfaches Bedienen der Inhalte über Bewegungen der Finger auf der Touchpadoberfläche.

2.2.1.2 Sprachsteuerung

Apple's *Siri* oder *Microsoft*'s *Cortana* [Jesdanun] können über Sprachsteuerung im Internet nach Suchbegriffen suchen, Kontakte anrufen, das aktuelle Wetter wiedergeben oder auch verschiedenste angeschlossene Geräte ansteuern. Sprachsteuerung wird bereits in Fahrzeugen verwendet. Die Fahrer können beispielsweise Kontakte aus ihrem Telefonbuch anrufen, oder das im Auto integrierte Navigationssystem bedienen.

Um mit Sprache Aktionen zu steuern, werden Worte aufgenommen und auf hinterlegte Sprachmuster hin abgeglichen. Wenn eine ausreichende Ähnlichkeit gefunden wurde, wird eine Aktion ausgeführt. [Schenk und Rigoll, 2010]

2.2.1.3 Gestensteuerung

Gesten können entweder über optische Sensoren (Computer Vision) oder gerätebasiert (tragbare Geräte) aufgenommen werden. Ziel ist es, die menschlichen Gesten mithilfe mathematischer Algorithmen zu interpretieren. Durch Gestensteuerung wird dem Menschen ermöglicht, sich, ohne physikalischen Kontakt zu einer Maschine, mit dieser zu verbinden und zu interagieren. Gesten können von jeglicher Körperbewegung verursacht werden, üblicherweise werden jedoch Hand- und Kopfgesten verwendet. [Khairnar et al., 2015]

Finger- und Handgesten

Finger- und Handgesten finden Verwendung in der Ansteuerung von Handprothesen, oder in der Gestensteuerung im Automobil. Mit Sensoren bestückte Armbänder, wie unter anderem das *Myoband* [Nuwer, 2013] (ein EMG-basiertes Armband) und taktile Armbänder [Kõiva et al., 2015], ermöglichen das Erkennen von Gesten, um beispielsweise Prothesen oder auch Computer zu bedienen.

Die Abnahme der relevanten Muskelsignale kann beispielsweise über Oberflächen-Elektromyographie (sEMG) [Jiang et al., 2012] oder Kraftmyographie (FMG) [Wininger et al., 2008] erfolgen. Gestenerkennung mit sEMG und FMG zählen zu den gerätebasierten Verfahren. Finger- und Handgesten können auch über optische Verfahren erkannt werden. Nissler et al. [2016] haben mit ihrer neuen Methode, der optischen Myographie (OMG), bewiesen,

dass es möglich ist, den menschlichen Vorderarm visuell zu untersuchen und daraus die Fingerbewegung zu rekonstruieren.

Kopf- und Augengesten

Rechy-Ramirez et al. [2012] verwenden einen Ansatz zur freihändigen Steuerung durch Kopfgesten. Die Kopfbewegung wird hierbei über ein in einem Headset integrierten Gyroskop erkannt. Der Ansatz von Jia et al. [2007] erschließt die Kopfbewegung aus der Position der Nase. Es ist ebenfalls möglich über Eye-Tracking und Augenblinzeln Maschinen zu steuern [Gajwani und Chhabria, 2010].

Über die Detektion der Blickrichtung ist es möglich, nur durch Hinsehen auf einem großen Display mit einzelnen Bereichen zu interagieren [Riener und Sippl, 2014].

2.2.2 Problemstellung: Erkennen, Tracken und Zuordnen von Gesten

Um Gestensteuerung zu nutzen, müssen grundsätzlich drei Probleme gelöst werden: das Erfassen, Erkennen und Extrahieren der relevanten Merkmale (eng. *Feature Detection*), das Verfolgen dieser Merkmale (eng. *Tracking*), sowie die Zuordnung einer Bedeutung zu den Posen und Trajektorien (engl. *Classification*) [Preim und Dachselt, 2015].

2.2.2.1 Erfassen von Merkmalen

In dieser Phase müssen über Sensoren Gesten, besser gesagt deren Bewegungsablauf, aufgenommen werden. Zu diesen Sensoren zählen beispielsweise Datenhandschuhe, Remote-Controller oder auch Tiefensensoren, mittels derer die Position und Bewegung im Raum erfasst werden kann. Die Erfassung von Bewegungen kann in *gerätebasierte* und *optische* Verfahren unterteilt werden. [Preim und Dachselt, 2015]

Gerätebasierte Verfahren

Bei gerätebasierten Verfahren werden die Sensoren meist am Körper getragen, so zum Beispiel Datenhandschuhe oder das *Myoband*. Dadurch lassen sich Handpositionen oder auch Fingerpositionen bestimmen.

Vorteil hierbei ist, dass meist eine hohe Präzision erreichbar ist. Nachteilig ist, dass die Geräte vor der Interaktion angelegt werden müssen und im Fall von Biosignalen eingelernt werden müssen.

Optische Verfahren

Kameras oder andere optische Sensoren können ebenfalls zum Erfassen der Gesten eingesetzt werden. Mittels Bildverarbeitung können die Gesten erkannt werden.

Dieses Verfahren hat den Vorteil, dass die Bewegung nicht eingeschränkt ist, und dass das System sofort genutzt werden kann. Zeitliche und räumliche Auflösung, sowie die Genauigkeit der eingesetzten mathematischen Algorithmen können die Präzision der Erkennung mindern.

2.2.2.2 Verfolgen von Merkmalen

Nach der Erfassung der Merkmale müssen diese über die Zeit getrackt werden. Die gefundenen Merkmale werden, bei optischen Verfahren, in jedem folgenden Bild im Bereich der vorherigen Position gesucht. Beim Verfolgen von Gesten spielen oft bekannte Modelle der zu trackenden Körperteile und das Wissen um deren mögliche Bewegungen eine wichtige Rolle. [Preim und Dachselt, 2015]

2.2.2.3 Zuordnung einer Bedeutung

Um einer Geste eine bestimmte Pose oder Bewegung zuzuordnen, müssen zunächst *Trainingsdaten* erhoben werden, mittels derer die verschiedenen möglichen Klassen trainiert werden können. Dieser Vorgang wird als Klassifizierung bezeichnet. Im Anwendungsfall werden dann die aufgenommenen Gesten mit den verschiedenen Klassen verglichen, um die Aufnahmen zuzuordnen. Bei der Klassifizierung werden oft statische Modelle und maschinelles Lernen verwendet, um die Trainingsdaten und die aufgenommenen Daten abzugleichen. [Preim und Dachselt, 2015]

2.2.3 Herausforderungen von gestenbasierten Natural User Interfaces

Die wesentliche Herausforderung ist die Umsetzung einer zuverlässigen Gestenerkennung. Die Kontrolle von Maschinen mittels Gestensteuerung muss, wie jede andere Form der Interaktion, den Grundregeln des Interaktionsdesigns folgen. Dabei muss ein klares Konzept für die Interaktion mit der Maschine bestehen und die verschiedenen möglichen Ausdrücke müssen definiert sein [Norman, 2010].

Die Gesten können mit Sensoren aufgezeichnet und über mathematische Algorithmen ausgewertet werden. Hier ist zu beachten, dass dieselbe Geste in einem anderen Kontext etwas anderes bedeuten kann. Dies ist dem Menschen bewusst, eine Maschine kann dies jedoch

schwer herausfinden [Wexelblat, 1997].

Latenzzeiten in der Gestenerkennung sollen möglichst gering sein. Der Grund dafür ist, dass der natürliche Interaktionsfluss nicht unterbrochen werden soll. Vor allem bei sicherheitskritischen Systemen muss darauf geachtet werden, dass die Gesten zu 100 % erkannt werden und dass sehr kurze Latenzzeiten eingehalten werden können. [Preim und Dachselt, 2015]

Komplexe Systeme können nicht durch einen ausschließlich gestenbasierten Ansatz umgesetzt werden. Um spezifischere und genauere Befehle erteilen zu können, muss ein hybrides System verwendet werden. Das heißt, die Gesten werden mit Datenhandschuhen, gesprochenen Kommandos oder auch peripheren Eingabegeräten gekoppelt. [Norman, 2010]

2.2.4 Vor- und Nachteile von Natural User Interfaces

Die Gestensteuerung kann unsere derzeitige HMI-Interaktion erweitern, sobald diese Technik weiter entwickelt ist [Norman, 2010]. Da Gesten täglich auch bei der Mensch-Mensch-Interaktion verwendet werden, sind uns diese sehr vertraut. Um Maschinen bedienen zu können, müssen die verwendeten Gesten wohl durchdacht sein. [Preim und Dachselt, 2015]

Ein weiterer Vorteil ist die erreichbare Schnelligkeit. Bei gutem Design der Gesten und deren Erkennung können beispielsweise Befehle schneller ausgelöst werden als mit einer Computermaus. [Preim und Dachselt, 2015]

Allerdings werden in verschiedenen Ländern Gesten anders interpretiert. In Indien wird unter anderem Nicken als Verneinung und Kopfschütteln als Bejahung aufgefasst. Wenn ein System ausschließlich über Gesten bedient werden soll, ist es für den Nutzer schwer selbst herauszufinden, wie er vorzugehen hat. Zur Unterstützung können aber Tutorials und Hilfsmenüs erstellt werden, mithilfe derer der Nutzer die Gesten einüben und zuordnen kann. [Norman, 2010]

Ein weiterer Nachteil ist, dass, so lange die Gestenerkennung aktiv ist, jede aufgenommene Geste eine Aktion auslöst. Dies kann zu Problemen führen, wenn gleichzeitig mit anderen Menschen interagiert wird. Aus diesem Grund muss die Gestenerkennung gesperrt und freigegeben werden können. [Preim und Dachselt, 2015]

3 Anforderungsdefinition

In diesem Unterkapitel werden die Anforderungen an die im Laufe dieser Arbeit implementierte Anwendung gestellt. Die Anforderungen sind in funktionale, nicht-funktionale, sowie Schnittstellenanforderungen gegliedert.

3.1 Funktionale Anforderungen

In diesem Abschnitt wird auf die Funktion der Anwendung eingegangen. Insgesamt werden drei Anforderungen an die Funktion gestellt:

[F1] – Es sollen Objekte hervorgehoben werden.

[F2] – Die hervorgehobenen Objekte sollen mittels Gestensteuerung ausgewählt werden.

[F3] – Es soll eine Kommunikation mit einem Roboter aufgebaut werden, um Befehle und Antwort zu übertragen.

Die funktionalen Anforderungen werden in den folgenden Abschnitten weiter ausgeführt.

3.1.1 Hervorheben von Objekten

In der Bildverarbeitung gibt es einige Möglichkeiten Objekte hervorzuheben. Je nach Anwendungsszenario kann die Kontur eines Objekts farblich hinterlegt oder auch eine Geometrie um das erkannte Objekt gezeichnet werden [Szeliski, 2010].

In der in dieser Arbeit beschriebenen Anwendung sollen keine Konturen oder andere geometrische Formen hervorgehoben werden. Dem Nutzer der Anwendung soll stattdessen ein 3D-Modell eines erkannten Objekts angezeigt werden. Der Vorgang des Einblendens von 3D-Objekten wird als Augmentierung bezeichnet. Eingeblendete 3D-Modelle stellen einen Teil der Möglichkeiten dar, die reale Welt zu erweitern.

3.1.2 Auswahl von Objekten

Die Anwendung soll unter Verwendung einer Datenbrille von *Epson* (vgl. Abschnitt 3.3.1) umgesetzt werden. Die Datenbrille ist mit Sensoren ausgestattet. Dazu zählen unter anderem

Beschleunigungssensoren und ein Gyroskop. Aus diesem Grund bietet sich die Auswahl von Objekten mittels einer Kopfgestensteuerung an.

Ziel ist es, ein Aus- und Abwählen von Objekten durch Nicken sowie Kopfschütteln zu erreichen. In den meisten Ländern der Welt sind Nicken und Kopfschütteln die verwendeten Gesten, um zuzustimmen oder zu verneinen. Viele Ansätze zum Detektieren der Gesten beruhen auf Videoaufnahmen, welche die Bewegung des Kopfes oder der Augen nachverfolgen. Die Erkennung der Gesten soll in dieser Arbeit nur mittels Sensordaten der Brille erfolgen.

3.1.3 Kommunikation mit einem Roboter

Um eine Interaktion mit einem Roboter zu ermöglichen, müssen diesem Informationen bezüglich der auszuführenden Aufgabe übermittelt werden. Die *Epson Moverio BT-200* bietet drei Kommunikationsstandards zur Übertragung von Informationen: Bluetooth, USB und WLAN.

Desweiteren soll eine Anbindung an die am DLR Oberpfaffenhofen im Institut für Robotik und Mechatronik verwendete Software „*Links and Nodes*“ (Abschnitt 3.3.2) umgesetzt werden. Dies hat den Grund, dass damit die Kommunikation mit allen an diesem Institut angebundenen Robotern erfolgen kann. Dadurch ist es notwendig, eine Verbindung zu einem PC herzustellen, auf welchem die Steuerungssoftware des Roboters läuft.

3.2 Nicht-Funktionale Anforderungen

Neben der Funktionalität der Anwendung müssen auch Aspekte der Benutzerfreundlichkeit betrachtet werden. Hierzu zählen der Einfluss der Latenzzeit und des Jitters des Systems auf die Anwendung, sowie die Interaktion mit der Anwendung.

[N1] – Es wird eine geringe Latenzzeit des Systems angestrebt.

[N2] – Der zeitliche Jitter in der Latenz soll gering gehalten werden.

[N3] – Der räumliche Jitter, über welchen die Positionsgenauigkeit angegeben wird, soll minimal sein.

[N4] – Die Interaktion mit der Brille soll intuitiv erfolgen.

In den folgenden Ausführungen wird auf die Latenz, den zeitlichen und räumlichen Jitter, sowie auf die Interaktion eingegangen.

3.2.1 Latenzzeit bei der Objektverfolgung und -hervorhebung

In Bezug auf Virtual und Augmented Reality ist die Latenzzeit ein fundamentales Kriterium, da die Latenzzeit einen großen Einfluss auf die Wahrnehmung der virtuellen Welt/Objekte hat. Eine geringe Latenz ermöglicht eine für den Betrachter angenehme Wahrnehmung der virtuellen Elemente. Um diese Wahrnehmung zu ermöglichen, müssen sich die virtuellen Objekte über die Zeit möglichst genau an der gewünschten Position in der realen Welt befinden. Ist dies nicht der Fall, so tritt in einer virtuellen Welt ein Wackeln des virtuellen Objekts, auch räumlicher Jitter genannt, auf, in einer augmentierten Welt stimmt die Position des Objekts offensichtlich nicht mehr mit der gewünschten Position überein. [Abrash, 2012]

3.2.1.1 Latenz

Die Latenz beruht auf einer Zeitverzögerung von Elementen der virtuellen Welt im Vergleich zur Realität. Jede Aktion, welche ein virtuelles Element und dessen Ausgabe betrifft, benötigt eine gewisse Zeit zur Ausführung. Die Gesamtverzögerung des Systems setzt sich aus einer Vielzahl von Ursachen zusammen. Hierzu gehören beispielsweise die Positionsberechnung, die Berechnung des Objekts, welches auf dieser Position angezeigt werden soll, das Rendering der Szene, die Zeit zum Anzeigen des Objekts, etc. [Craig, 2013]

Es ist wünschenswert, dass die Latenz des Systems noch unterhalb der visuellen Wahrnehmungsschwelle des Menschen liegt. Dies hat den Grund, dass dann die Aktualisierung der Position und Lage des Objekts nicht mehr wahrgenommen werden kann. Im Idealfall sollen Bilder mit einer Geschwindigkeit ausgegeben werden, die keine Einzelbildabfolge erkennen lässt. In der Praxis ist eine Umsetzung schwer möglich, da schon ein einfacher Perspektivenwechsel die Komplexität der Berechnungen erhöht, und somit zu einer größeren Latenzzeit führt. [Dörner et al., 2014]

3.2.1.2 Latenz bei Virtual Reality- und Augmented Reality-Systemen

Bei Virtual Reality-Systemen ist es möglich, verschiedene Aktivitäten miteinander zu synchronisieren, wobei auch hier reale Aspekte, wie beispielsweise Kopfbewegungen, nicht synchronisierbar sind. Es ergibt sich eine Gesamtlatenz, die großteils von nicht synchronisierbaren Events oder der größten Latenzzeit der einzelnen Komponenten abhängig ist. Kritischer wird es jedoch bei Augmented Reality-Systemen. Da es in der Realität keine Latenzzeiten gibt können die verschiedenen Aktivitäten nicht synchronisiert werden (die Latenz zwischen Auge und Hirnverarbeitung wird hier als zu gering angenommen). Es tritt der Effekt auf, dass die virtuelle Welt die reale Welt sozusagen zeitlich versetzt aufholt. In diesem Fall ist es nicht möglich, die Latenz ganz zu eliminieren. [Craig, 2013]

Brooks (1999) und Ellis et al. (1997) empfehlen eine Latenz von 50 ms für HMDs. Abrash meint, dass eine Latenz von 20 ms für Virtual Reality und besonders für Augmented Reality bereits zu viel ist, wobei in der Forschung von einem Schwellwert von 15 ms oder auch 7 ms die Rede ist.

3.2.1.3 Latenzzeitmessungen

Um die Latenzzeiten von VR-Systemen zu messen, gibt es bereits standardisierte Verfahren, mit welchen die Latenzzeiten für Trackingsysteme oder auch die Ende-zu-Ende Latenz bestimmt werden kann [Dörner et al., 2014].

In der Literatur gibt es jedoch noch keine standardisierten Latenzmessungen für Augmented Reality Systeme. Sielhorst et al. [2007] veröffentlichten eine Latenzmessung für AR-Video-Durchsicht-Geräte. Das Gerät wird hierbei auf einen Bildschirm gerichtet, welcher das Kamerabild des Geräts anzeigt. Sobald die Kamera beginnt, das Video aufzunehmen, entsteht ein Tunneleffekt. Um die Latenz zu berechnen, wird die Zeit zum Auftreten des ersten Tunnelfenster und der ersten Aufnahme des Startbildes verglichen. Dies wird rekursiv auf neue Tunnelfenster angewandt.

3.2.1.4 Erwartete Ergebnisse der Latenzmessung

In Abschnitt 5.2.1.1 ist eine durchgeführte Latenzzeitmessung beschrieben. Die in Abschnitt 3.3.1 beschriebene Hardware, die *Epson Moverio BT-200*, besitzt eine geringe Bildauflösung von 540×480 Pixel. Die Kamera selbst hat eine Framerate von maximal 30 Hz und folglich eine systembedingte Latenz von $33.\overline{33}$ ms. Der Erwartungswert für die Latenz wird auf einen Bereich zwischen 50 ms und 250 ms und somit auf 20 Hz bis 4 Hz festgesetzt.

3.2.1.5 Jitter

Räumlicher Jitter wird in einem System durch das Rauschen in den aufgenommenen Positionsdaten, beispielsweise eines Markers, verursacht. Mit diesem Jitter wird unter anderem das Wackeln von Augmentierungen beschrieben. In Bezug auf die Latenz wird ebenfalls von Jitter gesprochen, jedoch bezieht sich das in diesem Fall auf Abweichungen der Latenzzeit. [Liang et al., 1991; Teather et al., 2009]

Ein Versuch zur Messung des Jitters ist in Abschnitt 5.2.1.3 beschrieben.

3.2.2 Interaktion

Die Anwendung soll so umgesetzt werden, dass nicht über normalerweise verwendete Eingabegeräte mit einem Assistenzroboter interagiert wird. Die Kommunikation mit einem Assistenzroboter soll für Menschen mit Handamputationen möglich sein oder aber auch für Arbeiter, die beispielsweise an einer Montagelinie arbeiten und keine freie Hand für die Bedienung eines Interfaces zur Verfügung haben. Es soll eine berührungslose Interaktion mit der Anwendung ermöglicht werden. Somit sind andere Arten der Eingabe erforderlich, wie zum Beispiel Kopfgesten und die Blickrichtung.

Ein Ziel für die Umsetzung der Interaktion ist ebenfalls, dass diese intuitiv erfolgen kann und keine Einlernphase benötigt.

3.3 Schnittstellenanforderungen

Für die Umsetzung der Anwendung soll die Datenbrille *Epson Moverio BT-200* verwendet werden. Des Weiteren soll eine Anbindung zum Roboterkontrollsystem „*Links and Nodes*“ über einen Computer erfolgen.

[S1] – *Epson Moverio BT-200*

[S2] – Anbindung an einen Computer

Die Brille, sowie die Software „*Links and Nodes*“ werden nachfolgend beschrieben.

3.3.1 Epson Moverio BT-200

Diese Brille gehört zu der Kategorie Optical-See-Through-Smart-Glasses, ist also eine Durchsicht-Brille. Sie kann HD- und 3D-Inhalte anzeigen und wird auch für Augmented Reality-Anwendungen verwendet. Die Brille besitzt unter anderem ein Gyroskop, GPS, sowie eine Vielzahl weiterer Sensoren. Unter Verwendung dieser Sensoren kann die Bewegung des Nutzers nachverfolgt werden. Diese Informationen können in Anwendungen integriert werden. Um eine intuitive Bedienung zu ermöglichen, wird via Kabel ein Multi-Touch-Trackpad angeschlossen. Das verwendete Betriebssystem ist Android mit der Version 4.0.4. Das Gerät besitzt einen Prozessor mit 1.2 GHz und 1 GB RAM. [Epson, a]

Abbildung 3.1: Epson Moverio BT-200 [Epson, a]

Die eingebaute Kamera hat eine Auflösung von 540×480 Pixel und eine maximal mögliche Framerate von 30 Hz. Das LCD-Display hat eine Auflösung von 960×540 Pixel und eine Bildwiederholrate von 60 Hz. Das Sichtfeld ist uneingeschränkt, jedoch deckt das LCD-Display nur etwa 23° in der Diagonalen ab. Die Displaygröße ist abhängig vom Projektionsabstand. Bei 5 m Abstand hat das Display eine Größe von 80 ″, bei 20 m 320 ″. [Epson, b]

3.3.2 Computeranbindung

Der Roboter soll über das Roboterkontrollsystem, welches am DLR in Oberpfaffen am Institut für Robotik und Mechatronik verwendet wird, angesteuert werden.

„*Links and Nodes*“ ist eine institutseigene Software, über die die Ansteuerung und Kommunikation mit Robotern abgewickelt wird [Schmidt, 2013]. Um diese Software zu verwenden, muss eine Verbindung zu einem Computer aufgebaut werden.

4 Umsetzung

Die Umsetzung zum Erreichen des Ziels der Arbeit gliedert sich in vier Teile. Dazu gehören das Hervorheben eines Objekts unter Verwendung von Augmented Reality (**F1**) in Abschnitt 4.1, die zu implementierende Gestensteuerung (**F2**), mit welcher die Objekte aus- und auch abgewählt werden können, in Abschnitt 4.2, sowie die Kommunikation zwischen Brille und Roboter (**F3**) in Abschnitt 4.3. Abschnitt 4.4 befasst sich mit der Zusammensetzung dieser Komponenten zu einer Gesamtanwendung.

4.1 Hervorheben eines Objekts mittels Augmented Reality

In diesem Abschnitt wird das ausgewählte Framework, sowie die Umsetzung der Augmentierung beschrieben.

4.1.1 Verwendetes Framework

Als Framework wurde *Vuforia* ausgewählt. Dieses bringt den großen Vorteil mit sich, die Hervorhebung nicht nur auf einem Videostream anzuzeigen, sondern auch die Augmentierungen für Durchsichtgeräte direkt im Sichtfeld des Trägers zu überlagern. Dieses Framework bietet die Möglichkeit, Objekte, Bilder, Zylinder, Text, Boxen und Frame Markers (ähnlich zu QR-Codes) zu erkennen [Vuforia, c]. Die virtuellen Elemente werden anhand von Markern ausgerichtet.

Es gibt zwei Möglichkeiten Augmentierungen anzuzeigen: die eine ist, die Augmentierung direkt in das aufgenommen Kamerabild zu integrieren, die andere, die realen Objekte mit Augmentierungen zu überlagern, ohne den Kamerastream einzublenden. Die erste Variante kann mit sogenannten Videodurchsicht-, die letztere mit Durchsichtgeräten umgesetzt werden.

Bei beiden Varianten wird zunächst über die integrierte Kamera die Umgebung aufgenommen. In jedem Einzelbild wird dann nach einem oder mehreren Markern gesucht. Wenn ein Marker erkannt wurde, wird die Position und Lage des einzublendenden virtuellen Objekts berechnet. Bei den Videodurchsichtgeräten wird daraufhin die Augmentierung direkt in das aktuelle Bild eingefügt. In diesem Schritt jedoch unterscheiden sich Videodurchsicht- und Durchsichtgeräte. Bei Durchsichtgeräten findet die Überlagerung nicht auf dem Videostream statt, sondern die Augmentierungen werden im Bereich des Sichtfeldes des Trägers der Brille auf den realen

Markern angezeigt. Der augmentierbare Bereich des Sichtfeld ist hierbei auf den Bereich eingeschränkt, in dem das eingeblendete LCD-Display sichtbar ist, siehe Abbildung 4.1.

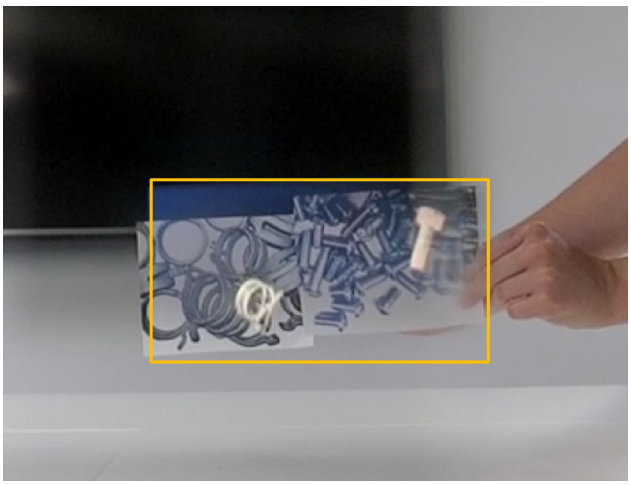

Abbildung 4.1: Sichtfeld des Trägers der Brille und der durch das LCD-Display abgedeckte augmentierbare Bereich (orange)

In der Abbildung ist der Bereich des LCD-Displays durch einen orangen Rahmen hervorgehoben. Mittels einer Kamerakalibrierung wird aus der Position und Lage des Markers im Videostream die Position und Lage der Marker in dem eingeschränkten Sichtfeld berechnet. Die virtuellen Elemente werden dann an dieser Position und Lage auf einem transparenten Hintergrund eingefügt.

Die Anzeige ist bei Durchsichtgeräten jedoch auf den Bereich des LCD-Displays beschränkt. Das Durchsichtgerät ermöglicht bei Verwendung eines HMDs ein angenehmeres Erlebnis der Augmented Reality, da die Augmentierungen sozusagen direkt auf die reale Welt überlagert werden. Es ist hier ebenfalls möglich, den aufgenommenen Videostream einzublenden, jedoch sieht man seine Umgebung doppelt, einmal die reale Umgebung und einmal die aufgenommene Umgebung, die auf dem LCD-Display angezeigt wird.

Im Falle von *Vuforia* werden Kalibrierungsdaten mittels eines speziellen Kalibrierungsprogramms erhoben [Vuforia, b]. Die daraus erhaltenen Informationen werden für die Berechnungen der 3D-Sicht verwendet.

4.1.2 Objekterkennung und -tracking

Wie in Abschnitt 2.1.3 beschrieben, werden bei Computer Vision-basierten Verfahren oft Marker oder natürliche Merkmale (Englisch: *„natural features“*) für die Objekterkennung sowie das Objekttracking verwendet. Um Objekte hervorheben zu können, müssen die Marker oder Merkmale dem System zuerst bekannt gemacht werden.

In dieser Anwendung werden keine Marker verwendet, sondern aus Markern extrahierte natürliche Merkmale. Aus den im Kamerabild gefunden Merkmalen können Punkte und Ebenen bestimmt werden, anhand welcher die virtuellen Objekte positioniert werden. Das Tracking mit natürlichen Merkmalen ist eine sehr rechenaufwändige Operation. [Tönnis, 2010]

Da *Vuforia* ein kommerziell erhältliches Software Development Kit (SDK) ist, sind die für die Objekterkennung und das -tracking verwendeten Methoden nicht zugänglich. Um eigene Marker zu verwenden, müssen diese auf dem *Vuforia Developer Portal* im sogenannten *Target-Manager* hochgeladen werden. Im *Target-Manager* werden die Merkmale extrahiert. Die Trackbarkeit des Markers wird auf einer Skala von 0 bis 5 bewertet, wobei 0 für nicht trackbar und 5 für sehr gut trackbar steht. Die hochgeladenen Marker können als Datenbank heruntergeladen und in die Anwendung eingebunden werden.

Insgesamt können maximal 100 Marker innerhalb einer Applikation verwendet werden, jedoch ist die Maximalanzahl der simultan trackbaren Marker hardwareabhängig. Meistens ist es möglich, zwischen fünf und zehn Marker gleichzeitig zu tracken. [Vuforia, a]

Abbildung 4.2 zeigt den Standardmarker von Vuforia für das Tracken mit natürlichen Merkmalen und die daraus extrahierten Merkmale.

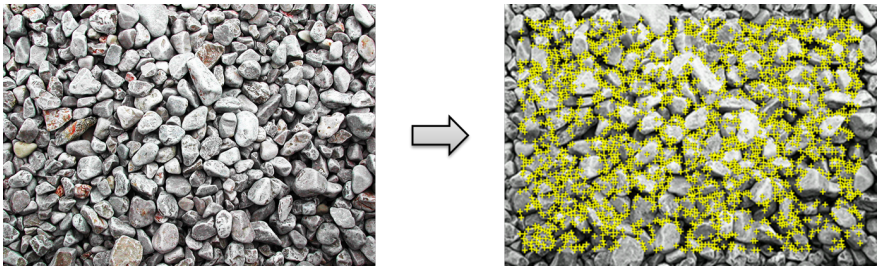

Abbildung 4.2: Marker und extrahierte Merkmale

4.1.3 Augmentierung

Sobald ein Marker im Sichtfeld des Trägers der Brille, genauer gesagt dem Bereich des LCD-Displays, auftaucht, soll ein Objekt auf diesem angezeigt werden.

Jedes Bild, welches von der Kamera der *Epson Moverio BT-200* aufgenommen wird, wird auf die dem System bekannten Marker überprüft. Sobald alle Marker überprüft wurden, werden für die erkannten Marker mittels OpenGL 3D-Modelle geladen. Daraufhin werden die Modelle, welche an den Markern ausgerichtet werden, im Sichtfeld angezeigt.

Die in der Datenbank enthaltenen Marker können über einen ID-String identifiziert werden. Dies ermöglicht die Zuweisung unterschiedlicher 3D-Modelle.

4.2 Gestensteuerung zum Aus- und Abwählen von Objekten

Für die Umsetzung des Aus- und Abwählen von Objekten mittels Gestensteuerung werden für diese Anwendung „Nicken“ und „Kopfschütteln“ verwendet.

4.2.1 Nicken und Kopfschütteln

Nicken und Kopfschütteln können als Bewegung auf einer Kreisbahn um eine Achse aufgefasst werden. Die Bewegung um die jeweiligen anderen beiden Achsen wird dabei als minimal angenommen. Wie aus Abbildung 4.3 ersichtlich, ist das Nicken eine rotatorische Bewegung um die x-Achse, das Kopfschütteln eine rotatorische Bewegung um die y-Achse.

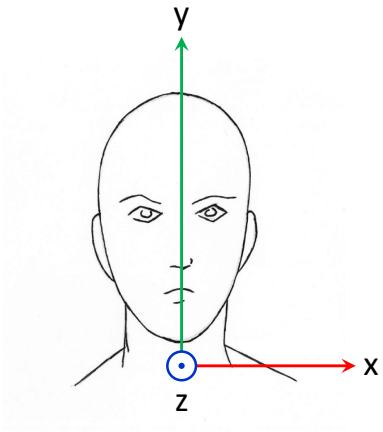

Abbildung 4.3: Verwendetes Koordinatensystem

4.2.2 Gestenerkennung

Für die Gestenerkennung wird das in der Brille integrierte Gyroskop verwendet, welches die Winkelgeschwindigkeit misst. Da es sich bei Nicken und Kopfschütteln um eine rotatorische Bewegung handelt, kann das Modell aus Abbildung 4.3 auf die Brille übertragen werden. Die Achsenbezeichnungen stimmen hierbei überein.

Abbildung 4.4 zeigt aufgezeichnete Datensätze der rotatorischen Geschwindigkeit für das Nicken und Kopfschütteln um alle drei Achsen. Die Sensordaten werden alle 200 ms abgetastet. In den orange hervorgehobenen Bereichen wurden Gesten aufgenommen. Es ist ersichtlich, dass für das Nicken die Geschwindigkeit um die x-Achse, sowie für das Kopfschütteln die Geschwindigkeit um die y-Achse signifikant für die Erkennung der jeweiligen Geste ist.

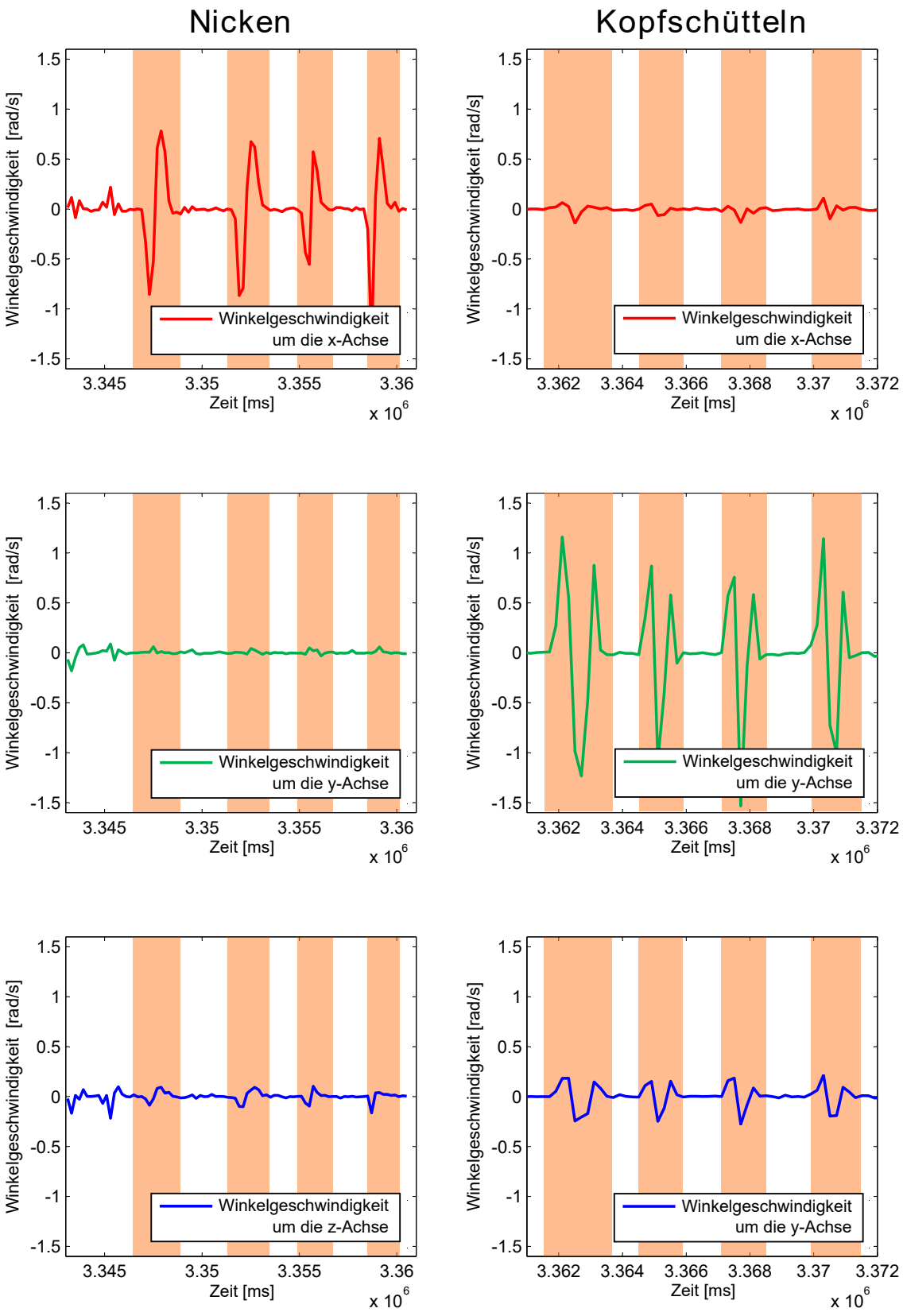

Abbildung 4.4: Aufgenommene Gyroskop-Sensordaten um die x-, y- und z-Achse für die Gesten Nicken und Kopfschütteln

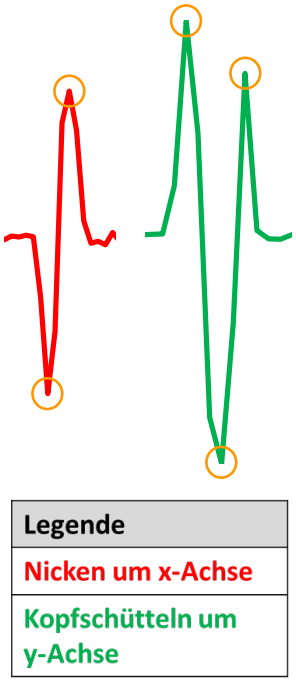

Abbildung 4.5:
Muster der Gesten

Die Bewegungsabfolge eines Nickens ist eine Abwärtsbewegung gefolgt von einer Aufwärtsbewegung. Beim Kopfschütteln findet eine Bewegung von links nach rechts und wieder nach links oder andersherum statt. Dieses Muster ist in den aufgezeichneten Daten aus Abbildung 4.4 für das Nicken um die x-Achse und für das Kopfschütteln um die y-Achse zu finden.

Wie aus Abbildung 4.5 zu entnehmen, können beim Nicken zwei alternierende Extremwerte und beim Kopfschütteln drei alternierende Extremwerte ausgemacht werden. Beim Vorgang des Nickens wird zuerst eine negative Geschwindigkeit um die x-Achse, gefolgt von einer positiven Geschwindigkeit um die x-Achse gemessen. Während des Kopfschüttelns wird um die y-Achse zuerst eine negative, dann eine positive und darauf eine negative Geschwindigkeit gemessen. Beim Kopfschütteln von rechts nach links und wieder nach rechts ist das Vorzeichen der Geschwindigkeiten umgekehrt.

Um die Gesten anhand der aufgenommen Daten zu erkennen, werden die jeweils aktuellsten 20 aufgenommenen Sensordaten, somit die letzten 4 s, der spezifischen Achse ausgewertet.

Die Gestenerkennung wird am Beispiel des Nickens beschrieben: Nacheinander werden alle Datenpunkt überprüft, ob diese kleiner gleich einem negativen Schwellwert sind. Sobald ein Datenpunkt dieses Kriterium erfüllt, werden die folgenden Datenpunkte auf das Überschreiten des positiven Schwellwertes überprüft. Sobald beide Bedingungen erfüllt sind, wird dem System das Erkennen einer Nickbewegung mitgeteilt.

Die Detektion des Kopfschüttelns folgt dem gleichen Schema mit insgesamt drei anstatt zwei Schwellwertabfragen.

4.3 Kommunikation mit einem Assistenzroboter

Dieser Abschnitt befasst sich mit der Kommunikation zwischen der Datenbrille und dem verwendeten Leichtbauroboter (LBR) von KUKA. Das hierfür verwendete Kommunikationsmedium zwischen Brille und LBR ist ein Computer. Um eine plattformübergreifende Kommunikation zu ermöglichen, wurde das Kommunikationsskript in Python verfasst.

4.3.1 Kommunikationsstandards

Für die Kommunikation stehen drei Standards zur Verfügung: Bluetooth, USB und WLAN.

Bluetooth ermöglicht eine kabellose Verbindung, jedoch verbraucht diese Art der Verbindung mehr Energie als die normale Nutzung eines Geräts. Es gibt bereits eine Bluetooth-Variante, die einen geringeren Akkuverbrauch hat. Diese Variante ist jedoch erst ab Androidversion 4.3 verwendbar [developer.android.com, b], das für die Datenbrille aktuellste Betriebssystem ist jedoch Android 4.0.4. Ein Update auf eine höhere Version ist zurzeit nicht verfügbar. Die Reichweite beträgt in etwa 10 m. Die Datenübertragungsrate liegt im Bereich von 3 Mbit/s.

USB erlaubt das gleichzeitige Kommunizieren mit einem angeschlossenen Gerät und das Laden eines der angeschlossenen Geräte. Der Bewegungsradius ist auf die Länge des Kabels beschränkt. Die Datenübertragungsrate bei USB 2.0 beträgt etwa 480 Mbit/s, bei USB 3.0 etwa 5 Gbit/s.

Bei WLAN-Verbindungen können leicht Störungen beim Datenempfang auftreten. Der Akkuverbrauch ist höher als bei einer Bluetoothverbindung. Die *Epson Moverio* verwendet den Standard IEEE 802.11b/g/n. Hier sind Datenübertragungsraten von 11 bis 150 Mbit/s möglich.

Die Wahl fiel aus zwei Gründen auf eine USB-Anbindung. Zum einen fällt die geringe Akkulaufzeit der Brille stark ins Gewicht. Diese beträgt maximal sechs Stunden. Wenn die Brille in einem Industriekontext verwendet wird, reicht die Akkulaufzeit bei einem Achtstundentag nicht aus. Da die Anwendung ebenfalls Ressourcen benötigt, um verschiedenste Berechnungen durchzuführen, ist von einer geringeren Akkulaufzeit auszugehen. Somit ist eine kontinuierliche Verbindung der Brille zu einer Stromquelle notwendig.

Der weitere Grund für die Umsetzung mittels USB war die geplante Einbindung der Anwendung in einen Messeaufbau des DLR. Da auf einer großen Messe das WLAN nicht immer zuverlässig verfügbar ist und auch die Akkulaufzeit der Brille nicht für einen ganzen Messetag ausreichend ist, war eine USB-Anbindung die einzige Möglichkeit.

4.3.2 Kommunikation über USB

Um mittels eines Android-Geräts über USB zu kommunizieren, wird das sogenannte *Android Open Accessory (AOA)*-Protokoll verwendet. Dieses Protokoll ermöglicht zwei Modi: den Host- und den Accessory-Modus. [developer.android.com, a]

Es wird der Accessory-Modus verwendet, da dieser ein zur Kommunikation zeitgleiches Laden des Android-Geräts ermöglicht.

4.3.3 Aufbau einer Verbindung zwischen Brille und Computer

Für den Aufbau einer Datenverbindung zwischen Brille und Computer muss das AOA-Kommunikationsprotokoll implementiert werden, siehe Abbildung 4.6.

Hierfür müssen zuerst die über USB angeschlossenen Geräte erkannt werden. Ein Gerät kann anhand seiner *Vendor ID* und seiner *Product ID* identifiziert werden.

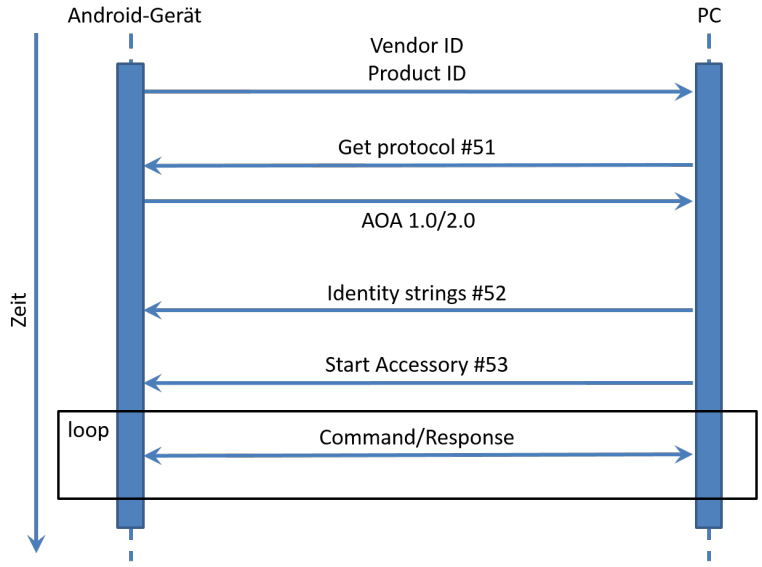

Abbildung 4.6: Sequenzdiagramm Android Accessory Protokoll (nach source.android.com)

Als nächster Schritt wird das Protokoll angefordert, über welches die Version für die Accessory-Modus Unterstützung abgefragt wird. Wenn die richtige AOA-Version zurückgegeben wird, werden Strings zur Identifikation des *Accessory Development Kits (ADK)* an das Gerät übertragen. Diese Strings beinhalten Identifikatoren für: Hersteller, Modellname, Beschreibung, Version, URL und die Seriennummer. Die Strings müssen mit den in der spezifischen Androidanwendung hinterlegten Strings übereinstimmen. Daraufhin wird der Accessory-Modus gestartet, woraufhin sich das Android-Gerät nochmals auf dem Bus, diesmal im Accesory-Modus, zu erkennen gibt.

Sobald sich das Gerät im Accessory Modus befindet, können zwischen der Applikation und dem Computer Daten ausgetauscht werden. Die Kommunikation zwischen Brille und Computer beschränkt sich für die implementierte Anwendung auf das Senden der Objekt-ID des ausgewählten Markers an den Computer und das Senden eines Rücksetz-Flags an die Brille, um in den Anfangszustand zurückzukehren.

4.3.4 Anbindung an Links and Nodes

Über einen Computer wird die Kommunikation zwischen Datenbrille und Roboter umgesetzt.

Sobald die Brille dem Computer die Auswahl eines Objekts übermittelt, werden mit Hilfe des Kommunikationsskripts Variablen der Software „*Links and Nodes*“ gesetzt. Ermittelt der LN-Manager eine Änderung des Werts dieser Variablen, so wird eine spezifisch hinterlegte Aktion ausgeführt, beispielsweise das Greifen eines Objekts.

Nachdem der Roboter die zugewiesene Aktion ausgeführt hat, gibt dieser Rückmeldung an den Computer. Dieser leitet das Signal an die Brille weiter, und das System kehrt in den Anfangszustand zurück.

4.4 Gesamtanwendung

Dieses Kapitel befasst sich mit der Zusammensetzung der in den vorangegangenen Abschnitten (4.1 bis 4.3) beschriebenen Komponenten zu einer einzigen Anwendung.

Zu Beginn wird auf die intuitive Bedienung der Applikation eingegangen, gefolgt von der Auswahl eines Objekts. Der allgemeine Programmablauf der Anwendung schließt dieses Unterkapitel ab.

4.4.1 Bedienung

Eine intuitive Bedienung erleichtert dem Nutzer den Umgang mit Maschinen. Die in dieser Arbeit entwickelte Applikation stützt sich hinsichtlich ihrer Bedienung auf einen visuellen Eingang (Kamera), einen sensorischen Eingang (Gyroskop), sowie einen visuellen Ausgang (LCD-Display).

In den meisten Ländern der Welt wird Nicken zur Bejahung, sowie Kopfschütteln als Verneinung verwendet. Um ein Objekt auszuwählen wird von der Anwendung ein Nicken als Bestätigung verstanden, ein Kopfschütteln als Verneinung, wodurch ein Objekt wieder abgewählt wird.

Die Auswahl eines Objekts findet über einen dreiphasigen Ablauf statt.

In der ersten Phase, der **Auswahlphase**, wird dem Nutzer für jedes trackbare Objekt ein 3D-Modell in Form dieses spezifischen Objekts angezeigt. Somit kann gewährleistet werden, dass der Träger der Brille intuitiv versteht, welche Hervorhebung sich auf welches Objekt bezieht.

Sobald ein Objekt durch ein Kopfnicken ausgewählt wurde, tritt die zweite Phase, die **Bestätigungsphase** ein. Hier wird vom Nutzer nochmals eine Bestätigung eingeholt, ob auch wirklich das richtige Objekt ausgewählt wurde. Versinnbildlicht wird dies durch ein 3D-Modell eines Fragezeichens. Die nicht ausgewählten Objekte werden in dieser Phase nicht mehr hervorgehoben. Die Auswahl eines Objektes erfolgt über Nicken. Dabei muss sich das Objekt in einem bestimmten Bereich des Sichtfeldes befinden. Dies ist in Abschnitt 4.4.2 beschrieben.

Wenn der Nutzer in der Auswahlphase das falsche Objekt ausgewählt hat, kann er dieses durch ein einfaches Kopfschütteln wieder abwählen und kehrt zur ersten Phase zurück. Ist jedoch das richtige Objekt durch das Fragezeichen hervorgehoben, so wird durch ein weiteres Nicken die dritte Phase eingeleitet.

Sobald die dritte Phase, die **Ausführungsphase**, beginnt, wird dem Roboter ein Befehl, beispielsweise zum Greifen und Reichen des gewählten Objekts, übermittelt. Die Augmentierung zeigt nun ein Häkchen, welches dem Nutzer vermittelt, dass dieses Objekt ausgewählt ist.

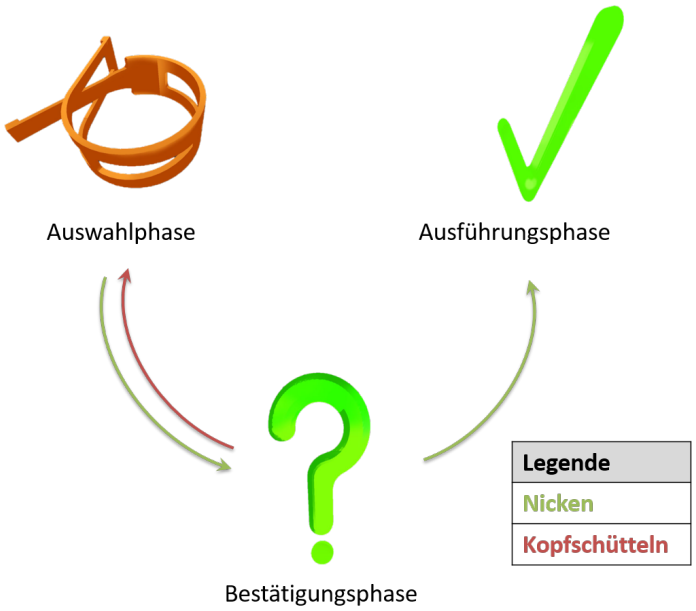

Abbildung 4.7: Visueller Phasenablauf

Abbildung 4.7 zeigt den visuellen Phasenablauf für die Auswahl eines Objekts (Markers). In diesem Beispiel wird das Objekt durch das 3D-Modell einer Schelle hervorgehoben. Sobald das Objekt über Nicken ausgewählt wird, wird anstelle der Schelle ein Fragezeichen angezeigt. Möchte der Nutzer die für dieses Objekt spezifiziert Aktion von dem Roboter ausführen lassen, erfolgt die Bestätigung durch ein weiteres Nicken und ein Haken erscheint. Andernfalls kann durch ein Kopfschütteln zurück zur Auswahlphase gewechselt werden.

4.4.2 Auswahl eines Objekts

Ein Objekt kann nur dann ausgewählt werden, wenn sich das System in der Auswahlphase befindet. Wie bereits in Abschnitt 4.1.2 erwähnt, können mehrere Marker simultan getrackt werden. Nun stellt sich die Frage, welcher Marker bei einem Kopfnicken ausgewählt wird.

Die Umsetzung der Auswahl eines Objekts wird anhand von 2 Markern erklärt.

Abbildung 4.8 stellt das LCD-Display dar, welches vor dem Träger projiziert wird. Der Bildbereich ist in x- und y-Richtung jeweils von -1 bis 1 skaliert. Für die Auswahl eines Objekts wird in der Anwendung nur ein eingeschränkter Bereich des Displays verwendet (orange hervorgehoben).

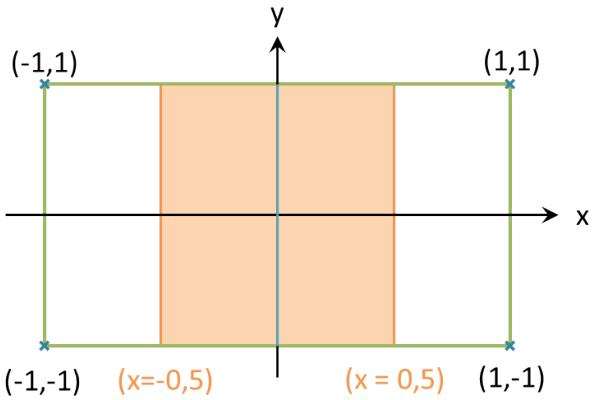

Abbildung 4.8: Bereich, in welchem ein Objekt ausgewählt werden kann

Wie aus dem Beispiel in Abbildung 4.9 ersichtlich, werden zwei Marker von der Anwendung erkannt und die spezifischen Objekte im Sichtfeld des Trägers der Brille eingeblendet. Der grüne Rahmen zeichnet den Umriss des in der Anwendung transparent gehaltenen LCD-Displays nach. Der Aufteilung aus Abbildung 4.8 folgend ist der Bereich, in welchem Objekte ausgewählt werden können, orange hervorgehoben, sowie die vertikale Mittellinie in blau angezeichnet.

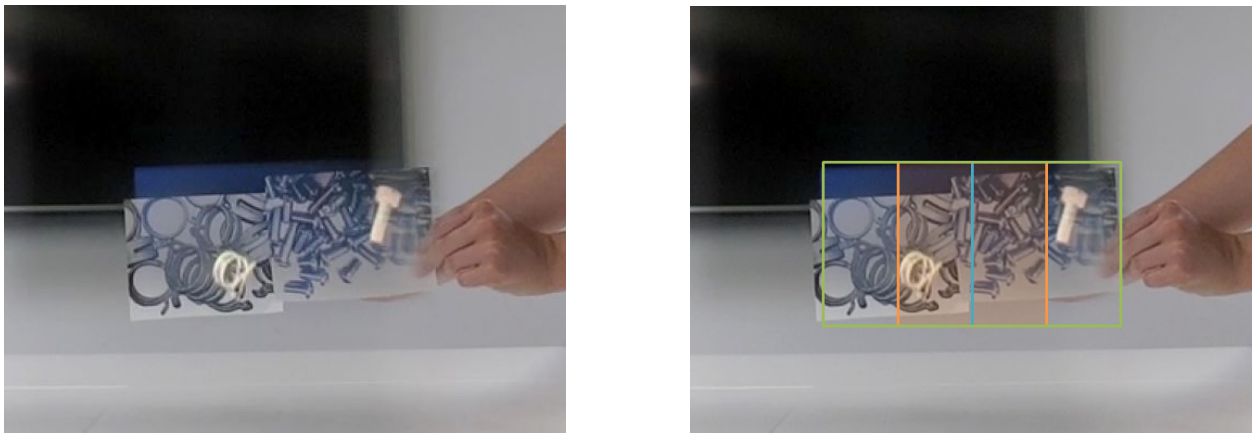

Abbildung 4.9: Sicht des Nutzers (links) und Verbildlichung des Auswahlbereichs (rechts)

Objekte können nur dann ausgewählt werden, wenn sich der zentrale Punkt der Augmentierung im orangen Bereich befindet und somit $|x| \leq 0.5$ erfüllt wird. Alle Marker, die diese Bedingung erfüllen, können ausgewählt werden.

Sobald eine Augmentierung dieses Kriterium erfüllt, kann das damit verbundene Objekt ausgewählt werden. In Abbildung 4.9 trifft dies für die Schelle zu. Wenn mehrere Marker in diesem Bereich detektiert werden, muss ein weiteres Kriterium erfüllt sein. Es wird das Objekt mit dem kleinsten euklidischen x-Abstand zum Mittelpunkt ausgewählt.

4.4.3 Programmablauf

Abbildung 4.10 stellt den Programmablauf der Anwendung dar. Ausgangspunkt ist die Auswahlphase, in der die Augmentierungen der möglichen auswählbaren Objekte angezeigt werden. In diesem Beispiel sind dies zwei Boxen, welche Schrauben und Schellen beinhalten. Wenn der Nutzer nun nickt, um eine Box auszuwählen, wird wie in Abschnitt 4.4.2 beschrieben vorgegangen und das sich zentral vor dem Nutzer befindliche Objekt ausgewählt. In diesem Beispiel wurde die Box mit den Schellen selektiert. Wenn der Nutzer das richtige Objekt ausgewählt hat, kann dieses durch ein weiteres Nicken bestätigt werden. Die Augmentierung verändert sich in diesem Schritt von einem Fragezeichen zu einem Häkchen.

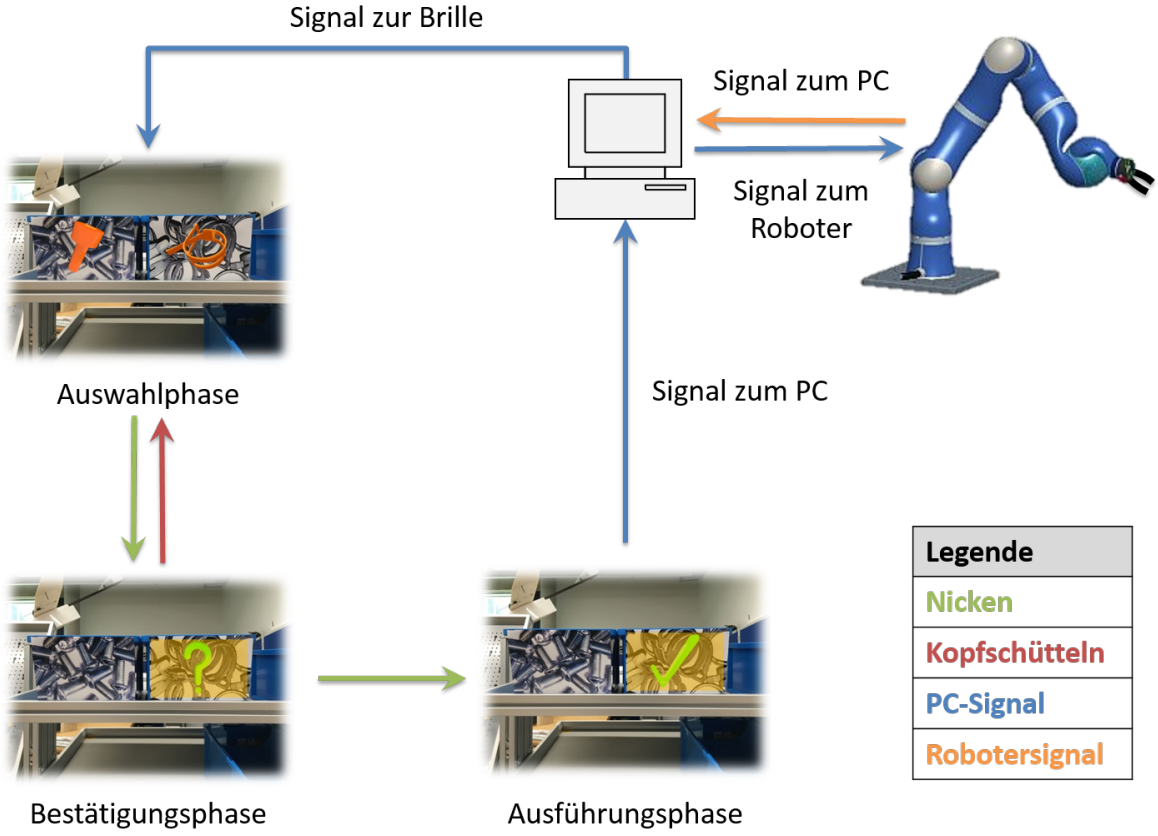

Abbildung 4.10: Programmablauf

Sobald die Ausführungsphase erreicht wird, sendet die Android-Anwendung über USB ein Signal an den PC. Der Datensatz enthält die Identifikationsnummer des ausgewählten Objekts. Daraufhin wird über das Pythonskript die zugehörige Variable in „*Links and Nodes*“ gesetzt. Nun führt der Roboter eine in „*Links and Nodes*“ hinterlegte Aktion aus. Nach Ausführung dieser Aktion gibt der Roboter ein Signal an den LN-Manager zurück. Das Pythonskript leitet dieses Signal an die Brille weiter. Dies bewirkt einen Sprung zurück zur Auswahlphase und auf den Markern wird wieder die spezifische Augmentierung angezeigt. Nun kann ein neues Objekt selektiert werden.

5 Evaluierung der Anforderungen

Dieses Kapitel befasst sich mit der Evaluierung der in Kapitel 3 an die Anwendung gestellten Anforderungen.

5.1 Funktionale Anforderungen

In diesem Abschnitt wird die Funktion der Anwendung anhand der Anforderungen bewertet. Es wurden insgesamt drei Anforderungen an die Funktionalität gestellt: das Hervorheben von Objekten durch Augmented Reality (**F1**), das Aus- und Abwählen von Objekten mittels Gestensteuerung (**F2**), sowie die Umsetzung einer Kommunikation zwischen Brille und Roboter (**F3**).

5.1.1 Hervorheben von Objekten

Abbildung 5.1 zeigt ein Beispiel für das Hervorheben von Objekten. Der orange Rahmen deutet den Bereich des LCD-Displays der Brille an.

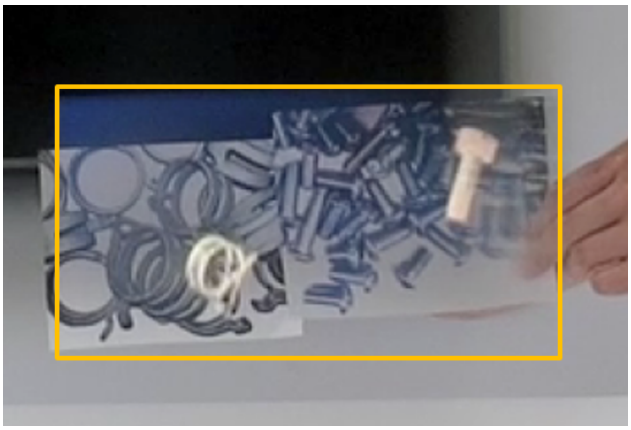

Abbildung 5.1: Beispiel für augmentierte Objekte (Visualisierung)

Die Augmentierungen werden auf den Markern angezeigt. Sie werden jedoch nicht genau im Zentrum des Markers eingefügt, sondern weichen etwas von dieser Position ab. Dies liegt daran, dass die Kamera nur ein monoskopisches Bild liefert, aber eine 3D-Augmentierung ausgegeben werden soll. Der 3D-Effekt muss für jeden Nutzer neu kalibriert werden. Diese Kalibrierung liefert jedoch ungenauere Ergebnisse als die Kalibrierung von Stereosystemen. Dies führt zu einer falschen Positionierung der Augmentierung.

Bei einem geringen Abstand zum Marker überlagern die Bilder für das rechte und das linke Auge noch nicht. Erst ab einem ausreichenden Abstand (abhängig von der Kalibrierung) wird der 3D-Effekt sichtbar.

Aus dem aufgenommenen Kamerabild müssen die Position und Lage des Markers erkannt werden. Hierbei kann es zu Fehldetektionen kommen, sodass die Augmentierung falsch positioniert und ausgerichtet wird. Es sind ebenfalls Sprünge in der Augmentierung auszumachen. Die Ursache dafür sind erhöhte Latenzzeiten. Genaueres kann Abschnitt 5.2.1.1 entnommen werden.

5.1.2 Empirische Schwellwerte für die Gestenerkennung

Um Nicken und Kopfschütteln robust detektieren zu können, wird ein Schwellwert zur Erkennung der jeweiligen Geste benötigt. Wie in Abschnitt 4.2.2 ausgeführt, können Nicken und Kopfschütteln mittels Gyroskop erkannt werden. Hierfür werden die aufgenommenen Daten auf die gestenspezifischen Muster (siehe Abbildung 5.2) hin überprüft. Für jede Geste wird ein Schwellwert benötigt. Für die Erkennung des Nickens wird beispielsweise zuerst das Unterschreiten des negativen Schwellwerts abgefragt und daraufhin das Überschreiten des positiven Schwellwerts. Die Detektion des Kopfschüttelns erfolgt über drei Schwellwertabfragen (siehe Abschnitt 4.2.2). Der Betrag beider Schwellwerte ist gleich.

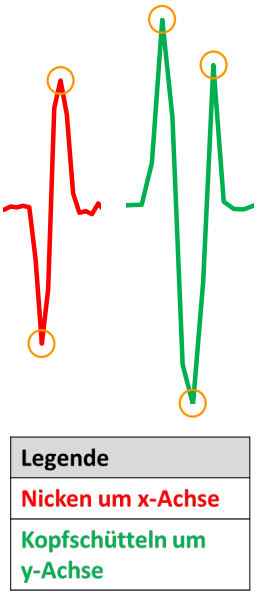

Abbildung 5.2: Muster der Gesten

Zur Ermittlung der Schwellwerte wurden Bewegungsdaten von zehn Testpersonen mittels Gyroskop erhoben. Insgesamt wurden pro Testperson die Gyroskopsensordaten für 10-mal Nicken, sowie 10-mal Kopfschütteln aufgenommen.

Die aufgenommenen Daten werden auf beide Gesten überprüft. Die Ergebnisse sind in Abbildung 5.3 dargestellt.

Die oberen beiden Graphen zeigen die erfolgreiche Detektion der Gesten, die unteren die Überprüfung der Datensätze auf eine Falschdetektion. Insgesamt werden acht Schwellwerte von 0 bis 0.8 rad/s betrachtet. An der y-Achse ist die Anzahl der absolut erkannten Gesten angezeichnet. Der Graph links oben zeigt die Überprüfung der Datensätze des Nickens auf das Nickmuster. Rechts oben ist die Überprüfung der Datensätze zum Kopfschütteln auf das Kopfschüttelmuster dargestellt. Die Graphen darunter zeigen die Ergebnisse der Falschdetektion, in welcher die aufgenommenen Datensätze auf das jeweilig andere Muster geprüft werden.

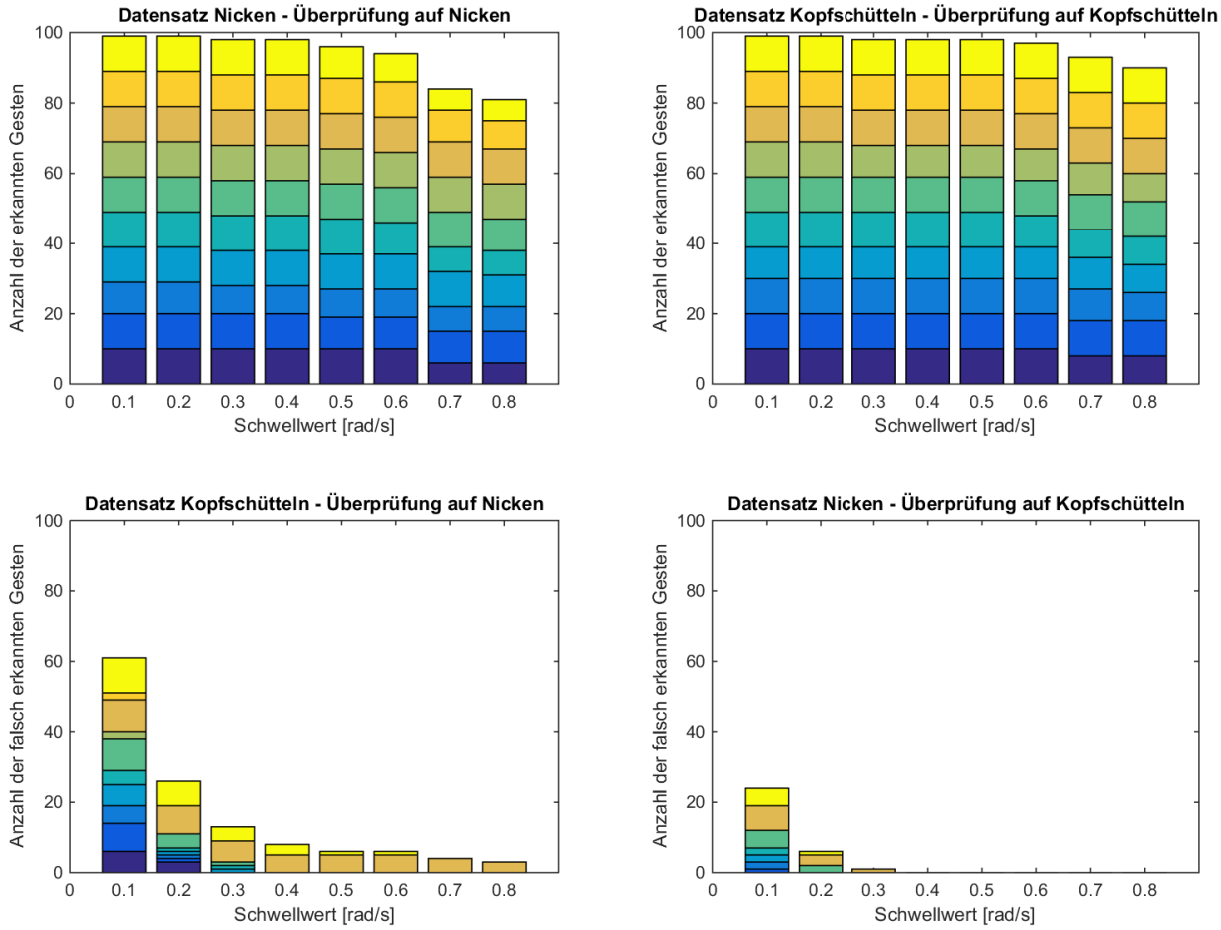

Abbildung 5.3: Überprüfen der aufgenommenen Datensätze auf Nicken und Kopfschütteln, sowie Überprüfung auf Falschdetektion

Wenn die Datensätze auf die jeweilig andere Geste überprüft werden, fällt auf, dass beim Kopfschütteln bis einschließlich zu dem Schwellwert von 0.3 rad/s die Gesten oft falsch erkannt werden. Ab 0.4 rad/s wurde nur bei zwei Datensätzen die Gesten teilweise falsch detektiert. Es ist auch ersichtlich, dass bei der braun hervorgehobenen Testperson der Großteil der Datensätze des Kopfschüttelns fälschlicherweise als Nicken detektiert wurde, obwohl die meisten Datensätze als Kopfschütteln erkannt wurden. Dies liegt daran, dass diese spezifische Testperson ihren Kopf bei Ausführung des Kopfschüttelns ebenfalls stark um die x- und z-Achse bewegt hat.

Bei der Überprüfung der Datensätze zum Nicken auf das Kopfschütteln wurden bis einschließlich 0.3 rad/s einige der Gesten falsch detektiert.

Um ein robustes Erkennen der Gesten zu realisieren, muss ein Schwellwert gewählt werden, der eine hohe Erfolgsrate hat und bei dem nur zu einem geringen Anteil Fehldetektionen auftreten.

Für die Erkennung des Nickens wurde somit ein Schwellwert von 0.5 rad/s festgesetzt, da bei diesem 98 % der Gesten aus den aufgenommenen Datensätzen für das Nicken richtig detektiert wurden. Die Falschdetektion kann vernachlässigt werden, da zum einen bei 0.5 rad/s nur 10 % der aufgenommenen Datensätze für das Kopfschütteln bei nur zwei Testpersonen zu einer Fehldetektion führte. Zum anderen kann angenommen werden, dass die Personen bei der Verwendung der Applikation bewusst eine der beiden Gesten ausführen.

Für die Erkennung des Kopfschüttelns wurde ein Schwellwert von 0.4 rad/s ausgewählt, da in den aufgenommenen Datensätzen ab 0.4 rad/s keine Fehldetektion mehr aufgetreten ist. Die Erfolgsrate für die Erkennung des Nickens liegt bei den aufgenommenen Daten bei 98 %.

Die Kombination aus einer beschränkten Länge an vorherigen Datenpunkten und der Schwellwert verhindern, dass normale Kopfbewegungen fälschlicherweise als Gesten detektiert werden. Es werden jeweils die letzten 20 Datenpunkte der Messreihe (4 s) auf die Gesten untersucht. Sobald eine Geste detektiert wurde, wird die Liste der Datenpunkte von Neuem befüllt und die Datenpunkte der bereits erkannten Geste nicht weiter berücksichtigt. Wenn sich der Nutzer von links nach rechts oder von oben nach unten umsieht, wird angenommen, dass der Richtungswechsel nicht innerhalb der 4 s stattfindet und dass, wenn doch, der Betrag der Winkelgeschwindigkeit der Bewegung unter dem gesetzten Schwellwert liegt.

5.1.3 Kommunikation mit einem Roboter

Die Kommunikation mit einem Roboter erfolgt über einen PC und die Software „*Links and Nodes*“ (LN) (siehe Abschnitt 3.1.3). Über den LN-Manager wird der angeschlossene Roboter angesteuert. Abbildung 5.4 zeigt eine Beispielanwendung für die Industrie.

Die grünen Pfeile zeigen den Interaktionsfluss einer Versuchsperson mit der Brille an. Durch Nicken werden die einzelnen Auswahlstufen durchlaufen. Sobald die Ausführungsphase (3D-Modell: Häkchen) erreicht wird, folgt der Roboter einem geplanten Bewegungsablauf. Der Bewegungsablauf ist in „*Links and Nodes*“ für das spezifisch ausgewählte Objekt hinterlegt. Es ist möglich, über LN einen anderen Pfad und somit eine andere Aufgabe zuzuweisen.

Abbildung 5.4 zeigt die auf der Messe vorgeführte Anwendung.

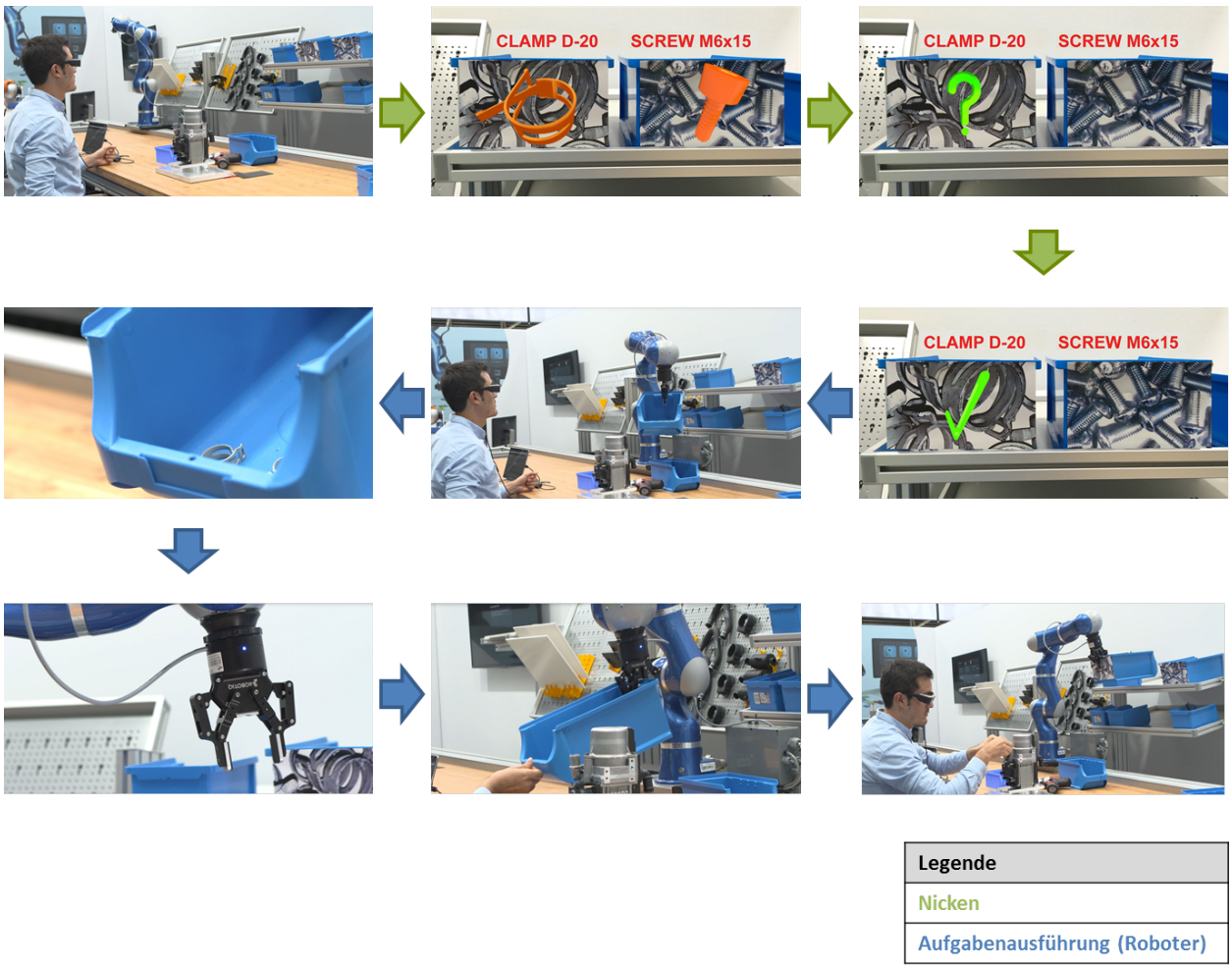

Abbildung 5.4: Beispielszenario für eine Anwendung in der Industrie

5.2 Nicht-Funktionale Anforderungen

In diesem Abschnitt werden die Messungen zu Latenz (**N1**), zum zeitlichen (**N2**) und räumlichen Jitter (**N3**) des Systems, beziehungsweise der Anwendung und eine Benutzerstudie zur Interaktion mit der Anwendung (**N4**) ausgewertet.

5.2.1 Messungen zu Latenz und Jitter

Die zu Latenz und Jitter des AR-Systems durchgeführten Messungen sind in diesem Abschnitt beschrieben. Wie bereits in Abschnitt 3.2.1 erwähnt, liegt die Ursache für Latenz an der benötigten Berechnungs- und Renderingzeit, welche zum Erzeugen der Bilder benötigt wird und an der Verzögerung in den aufgenommenen Orientierungsdaten. Der räumliche Jitter entsteht durch Rauschen in den aufgenommenen Positionsdaten, während der zeitliche Jitter sich auf das Rauschen in der Latenzzeit bezieht. [Liang et al., 1991]

5.2.1.1 Latenzzeitmessung

Für die Latenzzeitmessung wird ein Experiment ähnlich des für VR-Systeme oft verwendete Pendelexperiment genutzt. Es wird jedoch nur eine sinusähnliche Bewegung in x-Richtung ausgeführt. Der Versuchsaufbau ist an Liang et al. [1991] orientiert.

Versuchsbeschreibung

Das zentrale Element dieses Versuchs ist ein DIN A4 großer Marker. Als Marker wird der Standardmarker von *Vuforia* (siehe Abbildung 4.2) verwendet. Mittels der Kamera *CASIO EXILIM EX-F1* wird ein Video mit einer Framerate von 600 FPS aufgezeichnet, welches den Marker und das auf der Brille angezeigt LCD-Display erfasst. Es wird nur die angezeigte Bildfrequenz für ein Auge aufgenommen. Um eine gute Sicht auf das LCD-Display zu erhalten, muss sich die Kamera direkt hinter dem Brillenglas befinden (siehe Abbildung 5.5). Dies stellt kein Problem für die Aufnahme des Markers dar, da man durch die Brille hindurchsehen kann.

Die Anwendung ist so implementiert, dass die Hervorhebung, sprich das eingefügte 3D-Modell, zentral auf dem Marker platziert wird. Um die Auswertung der Latenzmessung zu erleichtern, wird der Mittelpunkt des Markers durch eine rote Markierung gekennzeichnet und als 3D-Modell eine Kugel verwendet. Die Abbildungen 5.5 und 5.6 zeigen den Versuchsaufbau.

Abbildung 5.5: Aufbau der Latenzzeitmessung

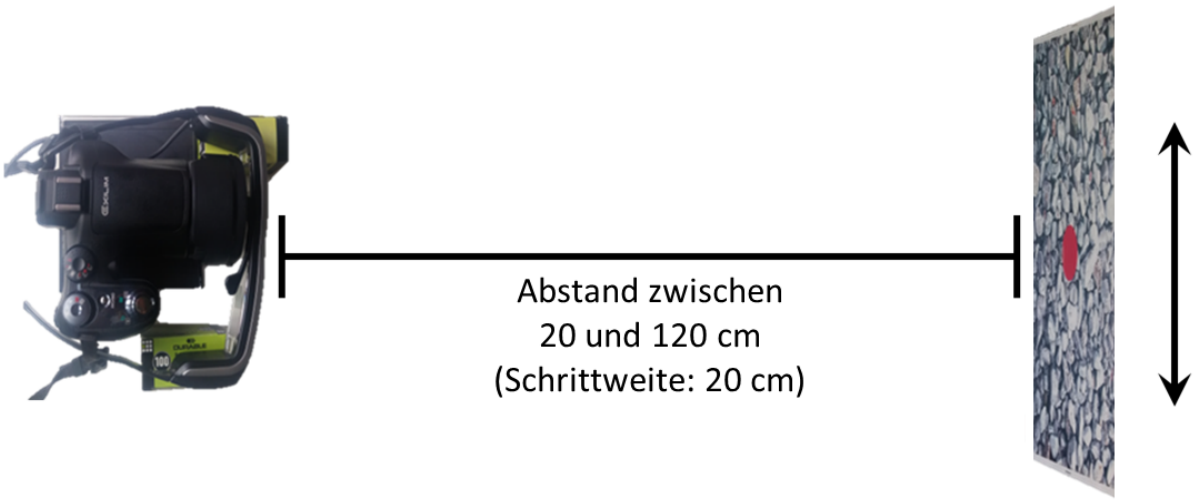

Abbildung 5.6: Aufnahme der Bewegung eines Markers in verschiedenen Abständen

Um die Latenzzeit des Systems zu berechnen, wird das aufgenommene Video Bild für Bild durchgesehen. Es wird für jedes gefundene Extremum der Markerposition, sowie der Position der Augmentierung in x-Richtung der Zeitpunkt notiert. Dazu werden die Zeitstempel der Videoaufnahme verwendet. Die Latenz v berechnet sich somit allgemein aus $l = t_\mathrm{A} - t_\mathrm{P}$ mit der Latenzzeit l, der Zeit der höchsten Position der Augmentierung t_A und der Zeit der höchsten Postion des Pendels t_P.

Durchführung des Experiments

Der Versuch wurde in einem geschlossenen Raum mit konstanten Lichtverhältnissen durchgeführt.

Angesetzt waren zehn Versuche mit unterschiedlichen Abständen zwischen Brille und Marker. Die verschiedenen Abstände dienen dazu, den Einfluss der geringen Kameraauflösung der *Epson Moverio BT-200* auf die Latenzzeit zu erkennen. Es wurden Abstände von 20 cm

bis 2 m, in 20 cm-Schritten, zwischen Brille und Marker betrachtet. Pro Versuch werden etwa fünf Bewegungen in x-Richtung ausgewertet. Die Bewegung besteht aus einer manuellen Verschiebung des Markers in der Horizontalen von einem Ausgangspunkt nach rechts und wieder nach links zurück zum Ausgangspunkt. Aufgrund des kleinen Sichtfeldes war es nicht möglich, eine kontrollierte Pendelbewegung zu simulieren.

Es waren nur sechs Versuchsdurchführungen möglich, da der Marker ab einem Abstand von etwas über 1.2 m nicht mehr detektiert wurde.

Versuchsauswertung

Zur Auswertung wurde ein Tracking-Tool von Open Source Physics (OSP) [Open Source Physics] verwendet. Dieses Tool ermöglicht das autonome tracken von Templates. Dafür wird ein zu verfolgender Bildbereich markiert. Über die Länge des Videos werden die Positionsdaten des Templates während des Template Matchings abgespeichert. Diese Daten wurden anschließend für die Auswertung verwendet. Die visualisierten Positionsdaten sind im Anhang unter Abschnitt A.1.2 zu finden.

Anhand des Graphen in Abbildung 5.7 wird die Auswahl der Referenzpunkte zur Berechnung der Latenzzeit beschrieben.

Da keine direkte Pendelbewegung imitiert werden konnte, kann in diesem Fall nicht direkt nach den auftretenden lokalen Minima und Maxima gesucht werden, um aus diesen die Latenzzeiten zu berechnen. Beim Wechsel der Bewegungsrichtung wird meistens für eine kurze Zeit an einer Stelle verharrt. In diesem Bereich sind die Pixelwerte quasi konstant. Die Zeitpunkte des Erreichens dieser „Endpositionen“ stellen die für die Berechnung notwendigen Referenzwerte dar.

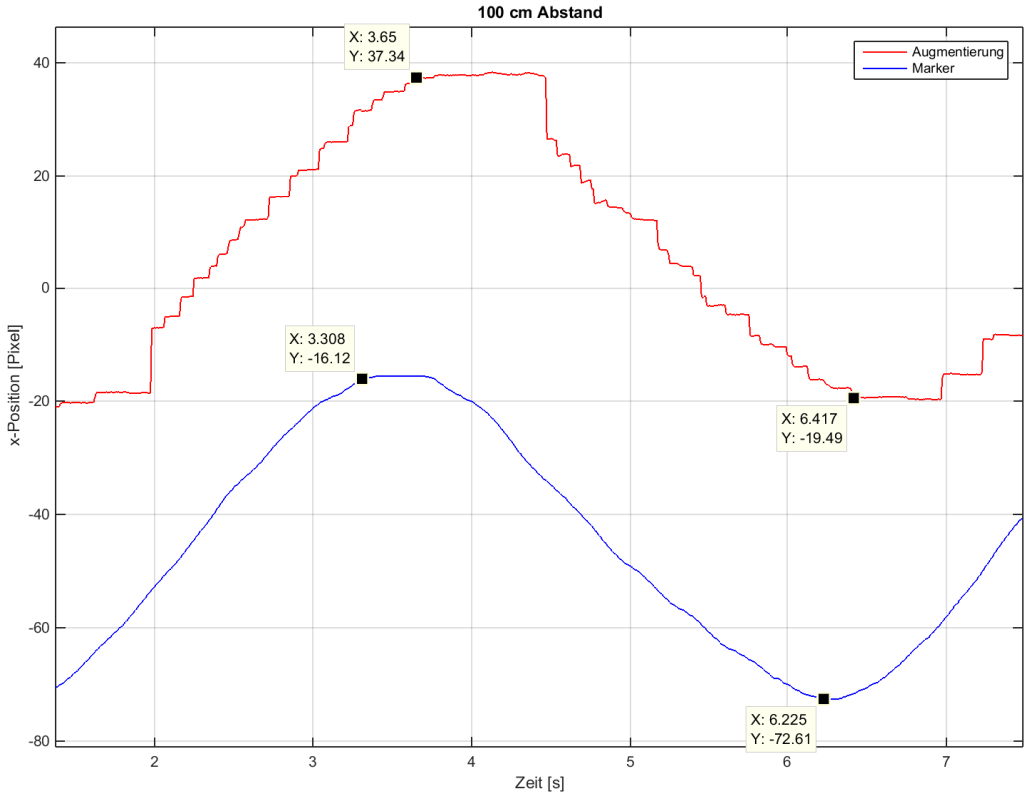

Abbildung 5.7: Beispiel zur Erfassung der Referenzdaten zur Berechnung der Latenzzeit (subpixelgenau)

Abbildung 5.8 visualisiert den Median der gemessenen Latenzzeiten für die durchgeführten Messungen:

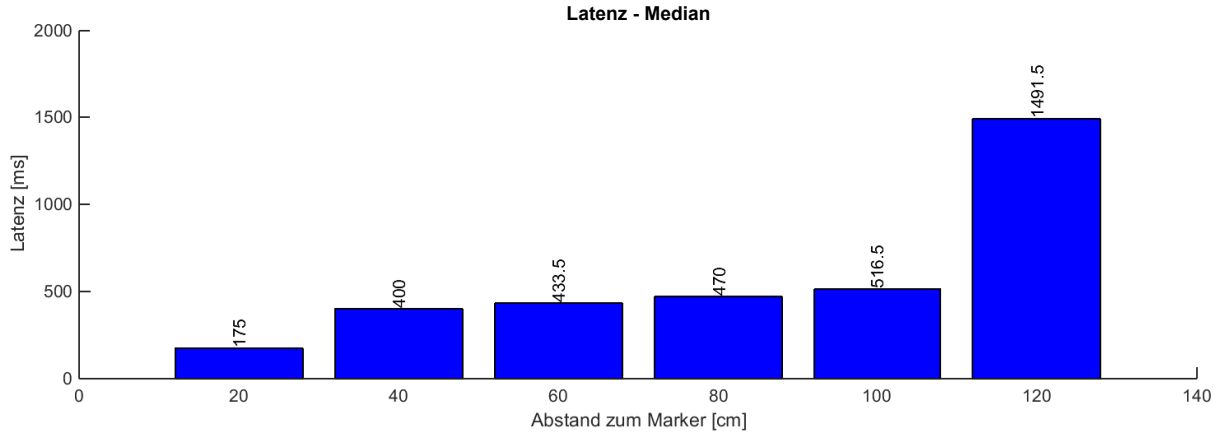

Abbildung 5.8: Gemessene Latenzzeiten für verschiedene Abstände

Es treten hohe Latenzzeiten im Bereich von 175 ms (≈ 5.7 Hz) bis 1491.5 ms (≈ 0.67 Hz) auf. Je größer der Abstand zwischen Brille und Marker wird, desto größer wird die gemessene Latenzzeit. Für die Abstände zwischen 140 und 200 cm liegen keine Messwerte vor, da der Marker bei diesen Abständen nicht mehr erkannt wurde.

Die hohen Latenzzeiten können auf verschiedene Ursachen zurückgeführt werden. Diese Ursachen können in Teilschritte, die eine konstante Berechnungszeit haben, und Teilschritte, die in der Berechnungsdauer variieren, unterteilt werden.

Für die Messung wurde immer dasselbe 3D-Modell verwendet. Somit kann davon ausgegangen werden, dass die Zeit zur Berechnung des Modells über alle Messungen konstant ist. Eine weitere Konstante ist die Umrechnung vom Kamerabild auf den Bereich, der auf dem LCD-Display sichtbar ist. Hier wird auf eine bereits berechnete Projektionsmatrix zugegriffen. Die Berechnung der Lage des Markers aufgrund der gefundenen Merkmale im Bild wird ebenfalls als konstant betrachtet.

Für die Varianz in der Latenzzeit ist wahrscheinlich der Detektions- und Trackingalgorithmus verantwortlich. Da die Algorithmen nicht bekannt sind, kann hier keine weitere Aussage getroffen werden. Ebenfalls spielt die schlechte Kameraauflösung von 480×540 Pixel eine Rolle, welche die Suche nach den natürlichen Merkmalen erschwert, da bei größer werdenden Abständen der Marker immer stärker verpixelt/verschwimmt.

5.2.1.2 Messung des zeitlichen Jitters

Um den Jitter in der Latenzzeit zu berechnen, werden die Daten verwendet, mittels derer die Latenzzeiten aus Abschnitt 5.2.1.1 berechnet wurden (Anhang: Tabelle A.1.2).

Der zeitliche Jitter ist eine andere Bezeichnung für das Rauschen in der Latenzzeit. Abbildung 5.9 zeigt die Standardabweichung der Latenzzeiten:

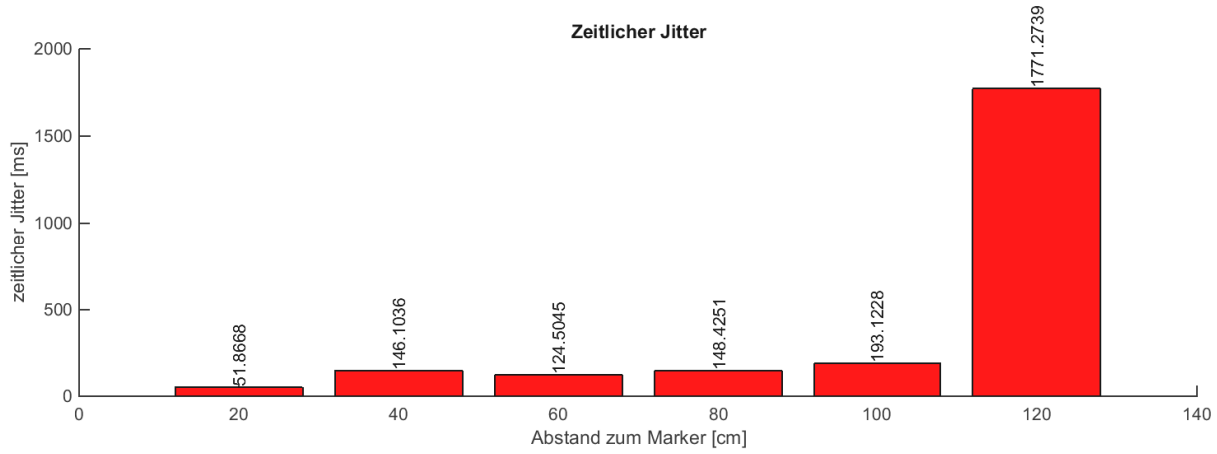

Abbildung 5.9: Zeitlicher Jitter

Auch hier sind große Schwankungen zu erkennen. Unter Annahme der Aussage über die konstanten und variablen Komponenten der Latenzzeit (Abschnitt 5.2.1.1) kann davon ausgegangen werden, dass der zeitliche Jitter allein von der Berechnungsdauer der Algorithmen abhängt. Stark auffällig ist der zeitliche Jitter von 1.77 s bei einem Abstand von 1.20 m. Hier ist der Einfluss des gering aufgelösten Kamerabildes zu sehen. Aufgrund der stark verschwommenen Aufnahme des Markers dauert es wahrscheinlich sehr lange, bis der Algorithmus eine ausreichende Anzahl an natürlichen Merkmalen findet, um die Lage des Markers zu berechnen.

5.2.1.3 Messung des räumlichen Jitters

Für die Messung des räumlichen Jitters (Rauschen in den Positionsdaten) wird, wie bei der Latenzeitzmessung, auf einen Ansatz von Liang et al. [1991] zurückgegriffen.

Versuchsbeschreibung

Zur Aufnahme der Positionsdaten wird ein stationäres System verwendet. Ein Marker wird dazu direkt im Sichtfeld (Bereich des LCD-Displays) platziert. Der Aufbau ist in Abbildung 5.10 dargestellt.

Abbildung 5.10: Messaufbau der Jittermessung

Über einen festgesetzten Zeitraum sollen die Positionsdaten in x-, y- und z-Richtung der Augmentierung abgespeichert werden.

Durchführung des Experiments

Der Versuch wurde in einem geschlossenen Raum mit konstanten Lichtverhältnissen durchgeführt. Als Marker wurde ebenfalls der Standardmarker von *Vuforia* (siehe Abbildung 4.2) in der Größe DIN A4 verwendet.

Es wurden insgesamt zehn Versuche mit unterschiedlichen Abstände zwischen Brille und Marker durchgeführt. Die Abstände liegen zwischen 20 und 200 cm mit einer Schrittweite von 20 cm. Pro Versuch wurden Daten im Zeitbereich einer Minute aufgenommen.

Versuchsauswertung

Abbildung 5.11 zeigt die subpixelgenauen Standardabweichungen der Position der Augmentierung in x-, y- und z-Richtung (Tabelle A.3). Es werden jeweils die Standardabweichungen für die Positionsdaten des linken und des rechten Bildes dargestellt.

Man kann den räumlichen Jitter in diesem Fall als nicht existent annehmen, da alle Standardabweichungen kleiner als 1 Pixel sind.

Bis 1.2 m Abstand wird der Standardmarker erkannt und kann erfolgreich getrackt werden. Über einem Abstand von 1.2 m wird der Marker nicht mehr erkannt. Es ist jedoch möglich, den Marker unter 1.2 m Abstand zu detektieren und daraufhin den Abstand zu erhöhen. Dabei kann der Marker noch nachverfolgt werden und die Augmentierung wird eingeblendet. Die durch diese Bewegung aufgenommenen Datenpunkte wurden nicht für die Berechnung der Standardabweichung verwendet.

Bei Bewegung des Markers bei Abständen über 1.2 m verliert der Trackingalgorithmus sehr schnell den Marker. Aus diesem Grund konnten die Latenzzeitmessungen aus Abschnitt 5.2.1.1 nur bis zu einem Abstand von 1.2 m durchgeführt werden.

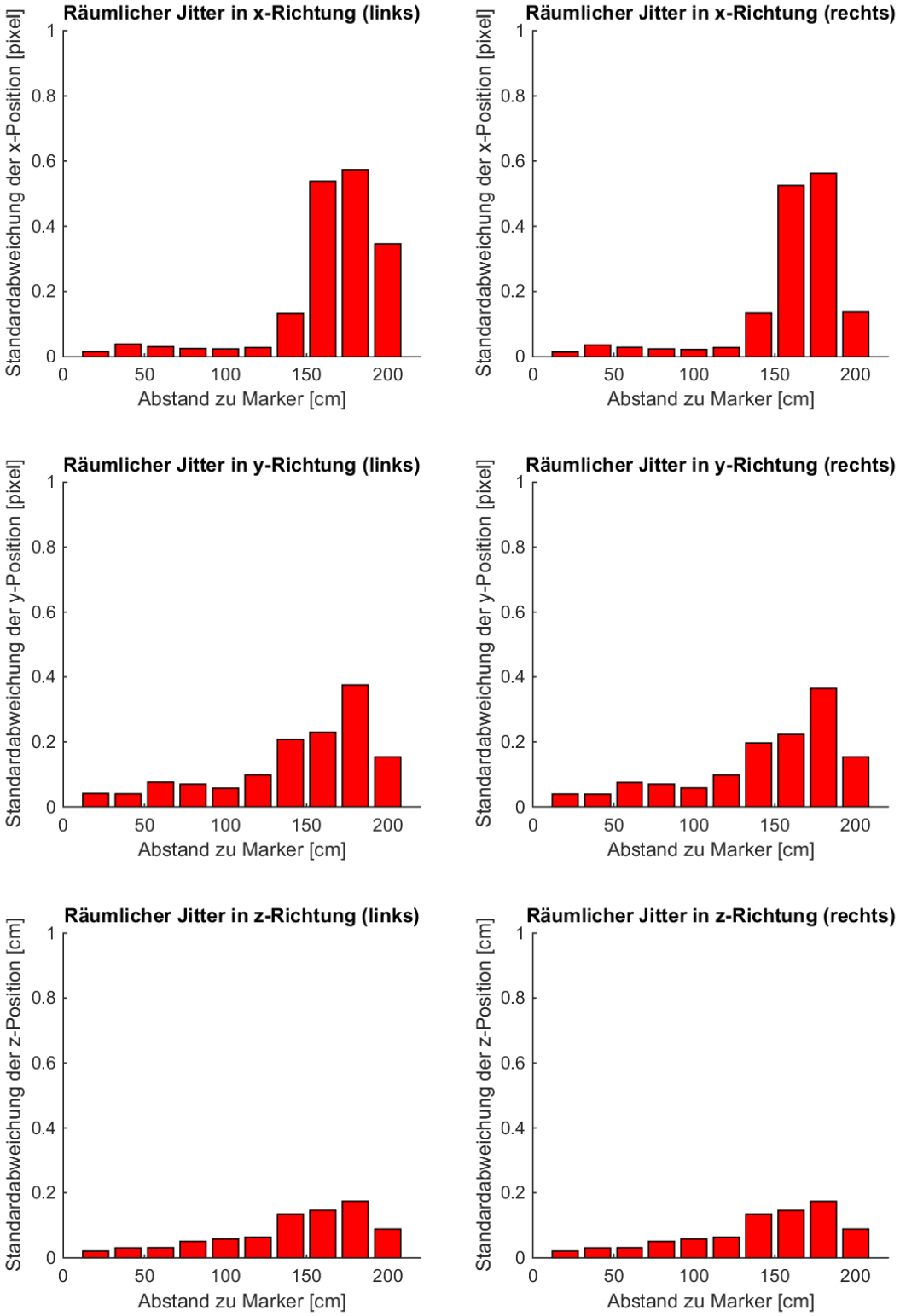

Abbildung 5.11: Räumlicher Jitter in x-, y- und z-Richtung für Abstände von 20 cm bis 200 cm zwischen Brille und Marker

5.2.2 Benutzerstudie zur Interaktion

Um die Intuitivität der Interaktion und das Zusammenspiel der einzelnen Komponenten bewerten zu können, wurden zehn Testpersonen befragt (Fragebogen: A.4).

Zuerst wurde für jede Testperson die 3D-Sicht mittels des Standard- *Vuforia*-Markers kalibriert. Die Testpersonen wurden gebeten, die Kalibrierung nicht in die Bewertung mit einfließen zu lassen. Daraufhin hatten die Probanden Zeit, die 3D-Darstellung der Augmentierung anhand zweier Marker zu beurteilen, sowie die Phasen durch Nicken und Kopfschütteln zu durchlaufen.

Für die Benutzerstudie wurde nur der Phasendurchlauf ohne Roboter betrachtet, da nur die Interaktion mit der Brille betrachtet werden sollte und nicht die Kommunikation mit dem Roboter. Aus diesem Grund kann von der Ausführungsphase über ein Nicken direkt in die Auswahlphase übergegangen werden.

Abbildung 5.12: Visueller Phasenablauf (Benutzerstudie)

In dieser Auswertung wird nicht jeder Punkt des Fragebogens abgehandelt. Es wird vorwiegend auf die gewonnenen Erkenntnisse bezüglich der Datenbrille, der Intuitivität, den eingeblendeten Augmentierungen und der Erkennung der Gestensteuerung eingegangen.

Datenbrille

Ein großes Problem in Bezug auf die Datenbrille war die Tatsache, dass die Brille auf dem Kopf keinen sicheren Halt hat. Besonders bei Brillenträgern war zu bemerken, dass diese sehr darauf bedacht waren, den Kopf nicht zu schnell zu bewegen, damit die Brille ihre Position beibehält. Aus diesem Grund war für diese Personengruppe das Durchlaufen der drei Phasen nicht zufriedenstellend, da sie ihren Hauptfokus auf die Brille legen mussten.

Es wurde ebenfalls angemerkt, dass es für die Augen sehr anstrengend ist, auf das LCD-Display zu fokussieren. Zwei Probanden sprachen von einem leichten Schwindelgefühl. Auch der kleine Sichtbereich der Brille wurde bemängelt.

Intuitivität

Um ein Objekt hervorzuheben, muss sich der Marker in dem vom LCD-Display abgedeckten Sichtfeld befinden. Diese Einschränkung verringert die Intuitivität der Anwendung, da die Testpersonen sich auf diesen Bereich fokussieren müssen, obwohl sie den Marker sehen.

Die Auswahl der Objekte, sowie die 3D-Modelle der Augmentierungen wurden als sehr intuitiv bewertet.

Augmentierung

Bei der Anzeige der Augmentierungen waren Sprünge in der Anzeige bemerkbar, jedoch wurden diese als nicht störend wahrgenommen. Der 3D-Effekt wurde bemängelt, da die Bilder für das rechte und das linke Auge nicht übereinander lagen, sondern zweimal das gleiche Objekt nebeneinander angezeigt wurde. Die Ursache für diesen Effekt ist die aufgrund der Monoskopie der Kamera durchgeführte Kalibrierung.

Die Marker wurden bis zu einem Abstand von 1.2 m erfolgreich detektiert, über 1.2 m nicht mehr. Wenn der Marker jedoch schon detektiert wurde und im Bild blieb, war das Tracking auch über 1.2 m erfolgreich, bei schnelleren Bewegungen ging der Marker jedoch schneller verloren.

Gestenerkennung

Aufgrund des häufigen Verrutschens der Brille kam es zu Falschdetektionen. Die Probanden führten die Gesten vorsichtig aus, um die Brille nicht zu verlieren.

6 Zusammenfassung und Ausblick

In diesem Kapitel werden nochmals die Ergebnisse der Evaluierung zusammengefasst, sowie ein Fazit gezogen (Abschnitt 6.1). Abschnitt 6.2 greift abschließend die weiteren Chancen und offenen Probleme für Anwendungen mit Datenbrillen auf.

6.1 Zusammenfassung

Die im Rahmen dieser Bachelorarbeit zu lösende Aufgabe war es, eine freihändige Interaktion mit einem Roboter zu realisieren. Diese Interaktion soll beispielsweise Anwendung in der Industrie für eine Mensch-Roboter-Kollaboration finden oder auch Menschen mit Einschränkungen der oberen Extremitäten bei der Interaktion mit einem Hilfsroboter unterstützen. Die Umsetzung erfolgt unter Verwendung von Augmented Reality und Gestensteuerung. Es wird die Datenbrille *Epson Moverio BT-200* verwendet. Die implementierte Anwendung ist als Proof-of-Concept gedacht.

Es werden verschiedene Marker zur Objekterkennung verwendet, für die ein spezifischer, geplanter Pfad oder auch eine Befehlsabfolge für den verwendeten Roboter hinterlegt ist. Im Kamerabild wird jedoch nicht direkt der Marker detektiert, sondern die zuvor extrahierten natürlichen Merkmale. Zur Auswahl dieser Marker wurde eine Gestensteuerung implementiert, die „Nicken“ als „Auswählen“ und „Kopfschütteln“ als „Abwählen“ interpretiert. Die Hervorhebung dieser Marker durchläuft insgesamt drei Phasen. In der ersten Phase, der Auswahlphase, werden 3D-Modelle angezeigt, welche auf den Inhalt der Aufgabe des Roboters schließen lassen. Die Modelle werden dem Nutzer in dem Sichtfeld der Brille, zentriert über den Markern, angezeigt. Durch Nicken kann ein Marker ausgewählt werden. Der Nutzer erreicht damit die Bestätigungsphase, in der ein Fragezeichen eingeblendet wird. Ist sich der Nutzer sicher, dass die ausgewählte Aufgabe ausgeführt werden soll, dann kann dieser die Aufgabe über erneutes Nicken bestätigen und ein Haken erscheint (Ausführungsphase). Wenn doch eine andere Aufgabe gewünscht ist, kann die Aufgabe durch Kopfschütteln wieder abgewählt werden.

Sobald der Nutzer die Aufgabe bestätigt hat, wird dieses Signal über USB an einen Computer weitergeleitet, über den ein Roboter gesteuert werden kann. Der Computer gibt dem Roboter nun den Befehl, die hinterlegte Aufgabe auszuführen. Nach Beenden der Aufgabe sind wieder alle Aufgaben verfügbar und die Augmentierungen der ersten Phase werden angezeigt.

Die Aufgabenstellung, die Entwicklung eines Bedienkonzepts für eine freihändige Interaktion mit einem Roboter, konnte erfolgreich unter Verwendung von Augmented Reality und Gestensteuerung umgesetzt werden. Diese Applikation kann auf die verschiedensten Bereiche angewandt werden, für welche das freihändige Auswählen eines Objekts und/oder das Bestätigen einer Aktion notwendig ist. Jedoch gibt es noch sehr viel Raum für die Entwicklung von Augmented Reality-Anwendungen.

Die Auswertungen der an die Anwendung gestellten Anforderungen (Kapitel 3) in Kapitel 5 zeigt deutlich, dass bei den Augmentierungen starke, für den Menschen merkbare Latenzzeiten auftreten. Dies führt zu einer sprunghaften und zeitlich verzögerten Anzeige der 3D-Modelle (siehe Abschnitt 5.2.1.1). Ein weiteres Manko ist das schmale Sichtfeld, welches nur einen kleinen Bereich zur Detektion der Marker und dem Hervorheben dieser ermöglicht. Die geringe Größe des Sichtfeldes hat einen starken Einfluss auf die Interaktivität der Anwendung, da das Sichtfeld direkt auf die Marker ausgerichtet werden muss, obwohl der Nutzer den Marker sieht (Abschnitt 5.2.2). Ein großer Nachteil bei dieser Anwendung ist ebenfalls der geringe Aktionsradius der Markerdetektion. Der Aktionsbereich ist auf 1.2 m Entfernung beschränkt und damit für den Einsatz in einer Industrieumgebung zu klein. Im Anwendungsfall für die Unterstützung von Menschen mit Einschränkungen der oberen Extremitäten ist dieser Radius für viele Aktivitäten jedoch vermutlich ausreichend.

6.2 Ausblick

Das Einbinden von computergestützter Erweiterung der Realität in Bezug auf die Industrie [Fraunhofer, 2016] und andere Branchen birgt viele Potentiale. Bis diese Technologie jedoch vermehrt zum Einsatz kommen kann, müssen noch einige Probleme gelöst werden.

Zu den Problemen zählen zum einen die bisher hohen Latenzzeiten von AR-Brillen. Durch das Aufrüsten der Hardware können die Latenzzeiten weiter minimiert werden. Eine höhere Framerate der Aufnahme, sowie eine höhere Rechenleistung tragen ebenfalls zu einer geringeren Latenzzeit bei.

Ein weiterer Punkt ist das kleine, durch die Datenbrille abgedeckte, augmentierbare Sichtfeld. Bei anderen Vertretern der AR-Brillen, wie *Google Glass* und *HoloLens*, ist dieses Sichtfeld ebenfalls recht klein. Für industrielle Anforderungen wird dieses wahrscheinlich nicht zufriedenstellend sein.

Eine wichtige Anforderung an Augmented Reality Systeme ist die Datensicherheit im Unternehmen. Damit Augmented Reality Anwendungen richtig funktionieren können, müssen Daten verarbeitet und gespeichert werden, meistens in Form von Videostreams. Somit wird das gesamte Arbeitsfeld des Angestellten, welcher eine solche Datenbrille trägt, mitgefilmt.

Aufgrund der großen Datenmenge ist die dauerhafte Speicherung der Daten unwahrscheinlich. Es handelt sich jedoch um sensible Daten und die Mitarbeiter könnten sich überwacht fühlen. Somit ist von einer geringen Akzeptanz von Datenbrillen am Arbeitsplatz auszugehen.

Für eine weitere Verbreitung und Entwicklung von AR-Systemen sind „die weiter voranschreitende Vernetzung, die Entwicklung der Displaytechnologie und die Qualität der Trackingsysteme“ [Tönnis, 2010, S. 165] ausschlaggebend.

Der Einsatz von Datenbrillen und Augmented Reality bietet unter anderem die Vorteile, Arbeitsschritte einzublenden, um mit visuellem Feedback beispielsweise fehlerhafte Platzierungen zu vermeiden und die Fehlerquoten zu reduzieren [Schart, 2015a]. Zum Beispiel können Baupläne auf Bauteile überlagert werden und bereits mit vorgeschriebenem Drehmoment festgezogene Schrauben grün hervorgehoben werden.

Auch in anderen Branchen, wie Medizin und Innenarchitektur findet AR Anwendung. In der Medizin werden unter anderem bereits AR-unterstützt minimal-invasive Operationen und OP-Planungen durchgeführt. Chirurgen werden durch Überblendungen von graphischen Scans und Hilfestellungen unterstützt. Innenarchitekten können ihren Kunden Eindrücke von geplanten Projekten übermitteln, indem sie beispielsweise Einblendungen des Hauses den Kunden direkt auf das reale Grundstück projiziert zeigen können. [Schart, 2015b]

Das Zukunftsprojekt Industrie 4.0/Produktion der Zukunft soll eine intelligente Fabrik [StMWi, 2015] ermöglichen. Industrie 4.0 eröffnet viele neue Möglichkeiten und stellt gleichzeitig neue Anforderungen an die Mensch-Maschine-Interaktion. Die vielfältigen Nutzungspotentiale durch die computergestützte Erweiterung der Realität können im Industriesektor entlang des gesamten Produktlebenszyklus eingesetzt werden. [Fraunhofer, 2016]

Unter anderem sind die Bereiche visuelle Planung, zum Beispiel zur Überprüfung, ob ein neuer Motor in den vorhandenen Motorraum passt, oder auch die Qualitätssicherung, beispielsweise zur Prüfung von Schweißpunkten, aufzuführen.

In den nächsten 5 bis 10 Jahren wird sich diese Technologie laut Prognosen, wie beispielsweise dem Gartner Hype Cycle für neuartige Technologien [Burton und Walker, 2015], sehr stark weiterentwickeln und ebenfalls in der Industrie verwendet werden.

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

Abrash, M. Latency the sine qua non of AR and VR. *Ramblings in Valve Time*, 2012.

Azuma, R. T. A survey of augmented reality. *Presence: Teleoperators and virtual environments*, 6(4):355–385, 1997.

Bajura, M. and Neumann, U. Dynamic registration correction in video-based augmented reality systems. *Computer Graphics and Applications, IEEE*, 15(5):52–60, 1995.

Burton, B. and Walker, M. Hype cycle for emerging technologies, 2015, 2015.

Cho, Y., Lee, J., and Neumann, U. A multi-ring color fiducial system and an intensity-invariant detection method for scalable fiducial-tracking augmented reality. In *In IWAR*. Citeseer, 1998.

Comport, A. I., Marchand, É., and Chaumette, F. A real-time tracker for markerless augmented reality. In *Proceedings of the 2nd IEEE/ACM International Symposium on Mixed and Augmented Reality*, page 36. IEEE Computer Society, 2003.

Craig, A. B. *Understanding augmented reality: Concepts and applications*. Newnes, 2013.

developer.android.com. USB Host and Accessory. `https://developer.android.com/guide/topics/connectivity/usb/index.html` . Abgerufen am 25.06.2016, a.

developer.android.com. Bluetooth Low Energy. `developer.android.com/guide/topics/connectivity/bluetooth-le.html` . Abgerufen am 15.06.2016, b.

DIN EN ISO 10218-1:2012-01. Industrieroboter - Sicherheitsanforderungen - Teil 1: Roboter, 01 2012.

Dörner, R., Broll, W., Grimm, P., and Jung, B. *Virtual und augmented reality (VR/AR): Grundlagen und Methoden der Virtuellen und Augmentierten Realität*. Springer-Verlag, 2014.

Epson. *Epson Moverio Bt-200 - Brochure*. `http://www.epson.com/alf_upload/pdfs/brochure_moverio.pdf` . Abgerufen am 04.07.2016, a.

Epson. *Epson Moverio Bt-200 Technical Information for Application Developer*. `https://tech.moverio.epson.com/en/life/bt-200/pdf/bt200_tiw1405ce.pdf` . Abgerufen am 04.06.2016, b.

Fischer, J. and Klein, G. *Visual tracking for augmented reality: Edge-based tracking techniques for ar applications*. VDM Publishing, 2009.

Fraunhofer. Augmented Reality im industriellen Einsatz. *Fraunhofer-Einrichtung für Ent-*

wurfstechnik Mechatronik, 2016.

Gajwani, P. S. and Chhabria, S. A. Eye motion tracking for wheelchair control. *International Journal of Information Technology*, 2(2):185–187, 2010.

Jesdanun, A. Apple has Siri, and now Microsoft has 'Halo'-inspired Cortana. `http://www.dailynews.com/technology/20140402/apple-has-siri-and-now-microsoft-has-halo-inspired-cortana` . Abgerufen am 25.08.2016.

Jia, P., Hu, H. H., Lu, T., and Yuan, K. Head gesture recognition for hands-free control of an intelligent wheelchair. *Industrial Robot: An International Journal*, 34(1):60–68, 2007.

Jiang, N., Dosen, S., Müller, K.-R., and Farina, D. Myoelectric control of artificial limbs—is there a need to change focus. *IEEE Signal Process. Mag*, 29(5):152–150, 2012.

Kato, H. and Billinghurst, M. Marker tracking and HMD calibration for a video-based augmented reality conferencing system. In *Augmented Reality, 1999.(IWAR'99) Proceedings. 2nd IEEE and ACM International Workshop on*, pages 85–94. IEEE, 1999.

Khairnar, P. P., Wanjara, A. G., Bhosale, R., and Kamble, S. Human machine interface. *Multidisciplinary Journal of Research in Engineering and Technology*, pages 31–35, 2015.

Kiyokawa, K. An introduction to head mounted displays for augmented reality. *Emerging Technologies of Augmented Reality (Ed. Haller, Thomas and Billinghurst)*, 2008.

Kõiva, R., Riedenklau, E., Viegas, C., and Castellini, C. Shape conformable high spatial resolution tactile bracelet for detecting hand and wrist activity. In *2015 IEEE International Conference on Rehabilitation Robotics (ICORR)*, pages 157–162. IEEE, 2015.

Liang, J., Shaw, C., and Green, M. On temporal-spatial realism in the virtual reality environment. In *Proceedings of the 4th annual ACM symposium on User interface software and technology*, pages 19–25. ACM, 1991.

Madden, L. *Professional Augmented Reality Browsers for Smartphones*. John Wiley & Sons, 2011.

Milgram, P., Takemura, H., Utsumi, A., and Kishino, F. Augmented reality: A class of displays on the reality-virtuality continuum. In *Photonics for industrial applications*, pages 282–292. International Society for Optics and Photonics, 1995.

Myojin, S., Sato, A., and Shimada, N. Augmented reality card game based on user-specific information control. In *Proceedings of the 20th ACM international conference on Multimedia*, pages 1193–1196. ACM, 2012.

Naimark, L. and Foxlin, E. Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker. In *Proceedings of the 1st International Symposium on Mixed and Augmented Reality*, page 27. IEEE Computer Society, 2002.

Nissler, C., Mouriki, N., and Castellini, C. Optical myography: Detecting finger movements by looking at the forearm. *Frontiers in neurorobotics*, 10, 2016.

Nóbrega, R., Cabral, D., Jacucci, G., and Coelho, A. NARI: Natural Augmented Reality Interface. In *Proceedings of the International Conference on Computer Graphics Theory and Applications, GRAPP*, pages 504–510, 2015.

Norman, D. A. Natural user interfaces are not natural. *interactions*, 17(3):6–10, 2010.

Nuwer, R. Armband adds a twitch to gesture control. *New Scientist*, 217(2906):21, 2013.

Olson, E. Apriltag: A robust and flexible visual fiducial system. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 3400–3407. IEEE, 2011.

Open Source Physics. Tracker Video Analysis and Modeling Tool. `www.opensourcephysics.org/items/detail.cfm?ID=7365` . Abgerufen am 20.06.2016.

Park, J., You, S., and Neumann, U. Natural feature tracking for extendible robust augmented realities. In *Proc. Int. Workshop on Augmented Reality*, 1998.

Preim, B. and Dachselt, R. *Interaktive Systeme: Band 2: User Interface Engineering, 3D-Interaktion, Natural User Interfaces*. Springer-Verlag, 2015.

Pressigout, M. and Marchand, E. Hybrid tracking algorithms for planar and non-planar structures subject to illumination changes. In *Proceedings of the 5th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 52–55. IEEE Computer Society, 2006.

Rechy-Ramirez, E. J., Hu, H., and McDonald-Maier, K. Head movements based control of an intelligent wheelchair in an indoor environment. In *Robotics and Biomimetics (ROBIO), 2012 IEEE International Conference on*, pages 1464–1469. IEEE, 2012.

Rekimoto, J. Matrix: A realtime object identification and registration method for augmented reality. In *Computer Human Interaction, 1998. Proceedings. 3rd Asia Pacific*, pages 63–68. IEEE, 1998.

Riener, A. and Sippl, A. Head-pose-based attention recognition on large public displays. *IEEE computer graphics and applications*, 34(1):32–41, 2014.

Rolland, J. P., Davis, L., and Baillot, Y. A survey of tracking technology for virtual environments. *Fundamentals of wearable computers and augmented reality*, 1:67–112, 2001.

Schart, D. Augmented Reality in der Industrie. `https://www.wearear.de/augmented-reality-in-der-industrie/` . Abgerufen am 10.08.2016, 2015a.

Schart, N., Dirk und Tschanz. *Praxishandbuch - Augmented Reality*. UVK, 2015b.

Schenk, J. and Rigoll, G. *Mensch-Maschine-Kommunikation: Grundlagen von sprach-und*

bildbasierten Benutzerschnittstellen. Springer-Verlag, 2010.

Schmidt, F. Links and nodes manual. 2013.

Sielhorst, T., Sa, W., Khamene, A., Sauer, F., and Navab, N. Measurement of absolute latency for video see through augmented reality. In *Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 1–4. IEEE Computer Society, 2007.

source.android.com. Android Open Accessory Protocol 1.0. `https://source.android.com/devices/accessories/aoa.html` . Abgerufen am 25.06.2016.

StMWi. Zukunftsstrategie BAYERN DIGITAL. *Bayerisches Staatsministerium für Wirtschaft und Medien, Energie und Technologie*, Juli 2015.

Stricker, D., Klinker, G., and Reiners, D. A fast and robust line-based optical tracker for augmented reality applications. In *Proceedings of the international workshop on Augmented reality: placing artificial objects in real scenes: placing artificial objects in real scenes*, pages 129–145. AK Peters, Ltd., 1999.

Sukan, M., Feiner, S., and Energin, S. Poster: Manipulating virtual objects in hand-held augmented reality using stored snapshots. In *3D User Interfaces (3DUI), 2012 IEEE Symposium on*, pages 165–166. IEEE, 2012.

Szeliski, R. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

Teather, R. J., Pavlovych, A., Stuerzlinger, W., and MacKenzie, S. I. Effects of tracking technology, latency, and spatial jitter on object movement. In *3D User Interfaces, IEEE Symposium*, pages 43–50. IEEE, 2009.

Tillon, A. B., Marchal, I., and Houlier, P. Mobile augmented reality in the museum: Can a lace-like technology take you closer to works of art? In *Mixed and Augmented Reality-Arts, Media, and Humanities (ISMAR-AMH), 2011 IEEE International Symposium On*, pages 41–47. IEEE, 2011.

Tönnis, M. *Augmented Reality: Einblicke in die erweiterte Realität*. Springer-Verlag, 2010.

Uchiyama, H. and Marchand, E. Toward augmenting everything: Detecting and tracking geometrical features on planar objects. In *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, pages 17–25. IEEE, 2011.

Vogt, S., Khamene, A., Sauer, F., and Niemann, H. Single camera tracking of marker clusters: Multiparameter cluster optimization and experimental verification. In *Proceedings of the 1st International Symposium on Mixed and Augmented Reality*, page 127. IEEE Computer Society, 2002.

Vuforia. *How many targets can I track with Vuforia.* `https://developer.vuforia.com/forum/faq/how-many-targets-can-i-track-vuforia` . Abgerufen am 23.06.2016, a.

Vuforia. Vuforia Calibration Assistant. `developer.vuforia.com/library/articles/Training/Vuforia-Calibration-app` . Abgerufen am 10.06.2016, b.

Vuforia. What your app can see. `http://www.vuforia.com/Features` . Abgerufen am 10.06.2016, c.

Wexelblat, A. Research challenges in gesture: Open issues and unsolved problems. In *International Gesture Workshop*, pages 1–11. Springer, 1997.

Wininger, M., Kim, N.-H., and Craelius, W. Pressure signature of forearm as predictor of grip force. *Journal of rehabilitation research and development*, 45(6):883, 2008.

Zhou, F., Duh, H. B.-L., and Billinghurst, M. Trends in augmented reality tracking, interaction and display: A review of ten years of ISMAR. In *Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*, pages 193–202. IEEE Computer Society, 2008.

A Anhang

A.1 Latenzzeiten

A.1.1 Median der Latenzzeit

Abstand in [m]	0.2	0.4	0.6	0.8	1.0
Latenzzeit in [ms]	175	400	433.5	470	516.5

Abstand in [m]	1.2	1.4	1.6	1.8	2.0
Latenzzeit in [ms]	1491.5	–	–	–	–

Tabelle A.1: Übersicht des Median der gemessenen Latenzzeiten in Abhängigkeit von dem Abstand zwischen Brille und Marker

A.1.2 Latenzzeiten der Abstände 20 cm bis 120 cm

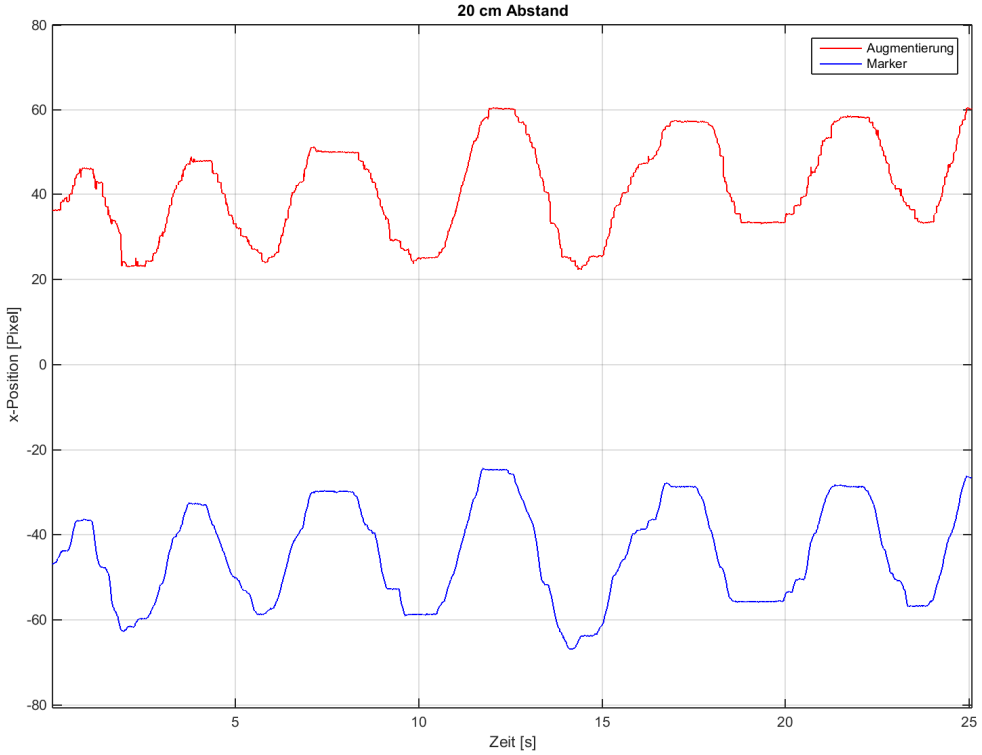

Abbildung A.1: 20 cm Abstand

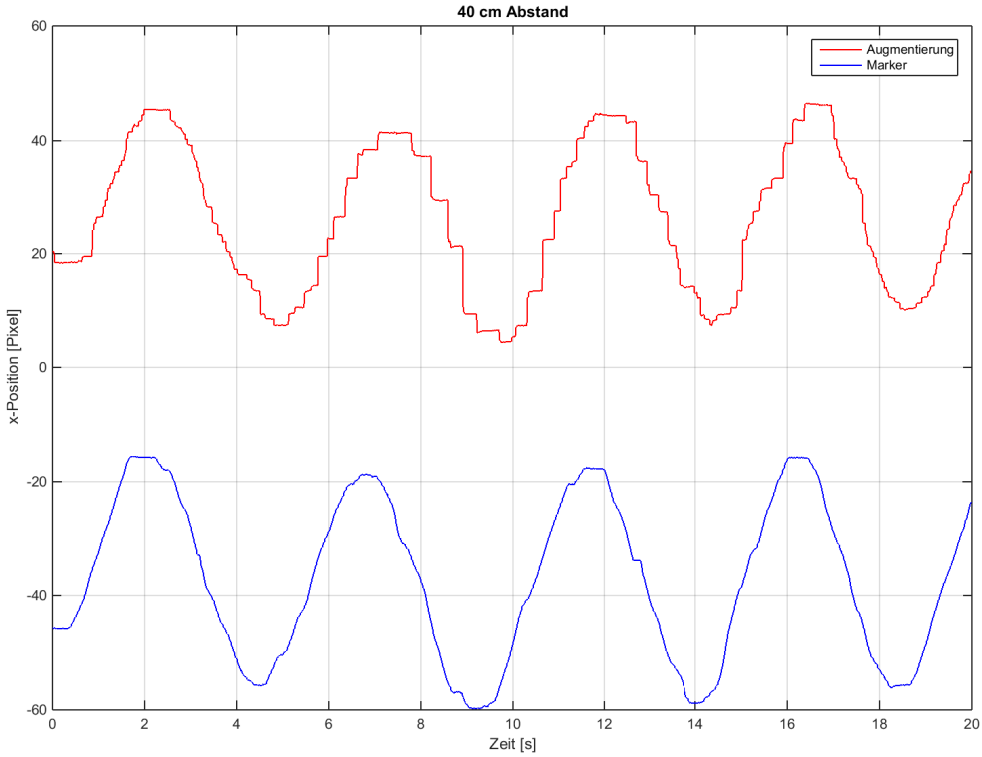

Abbildung A.2: 40 cm Abstand

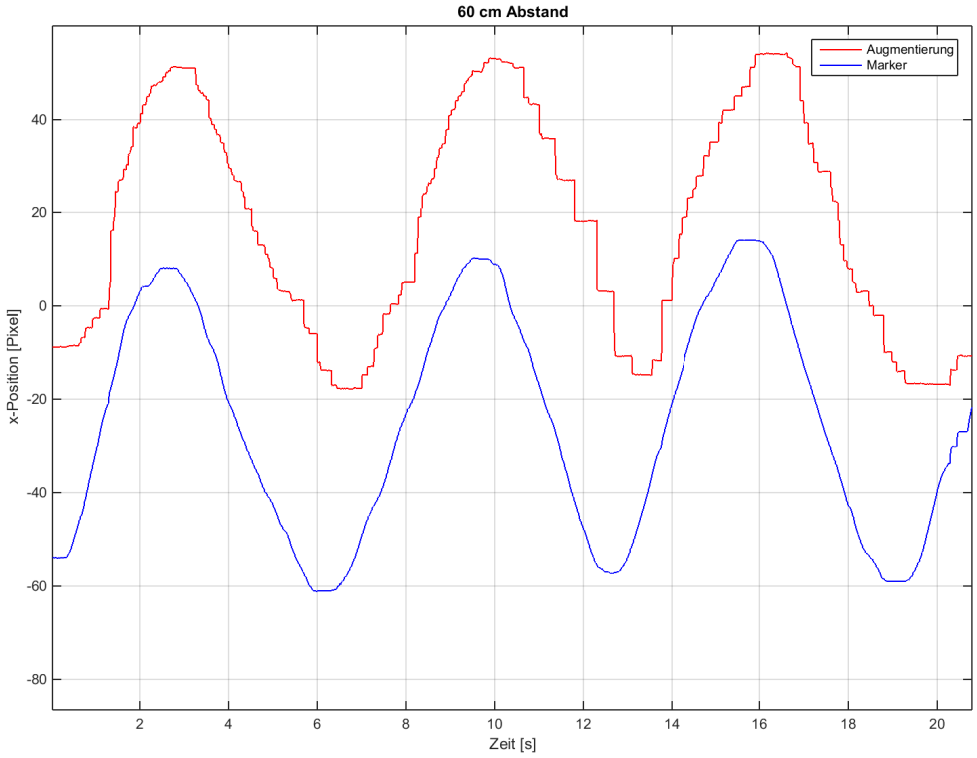

Abbildung A.3: 60 cm Abstand

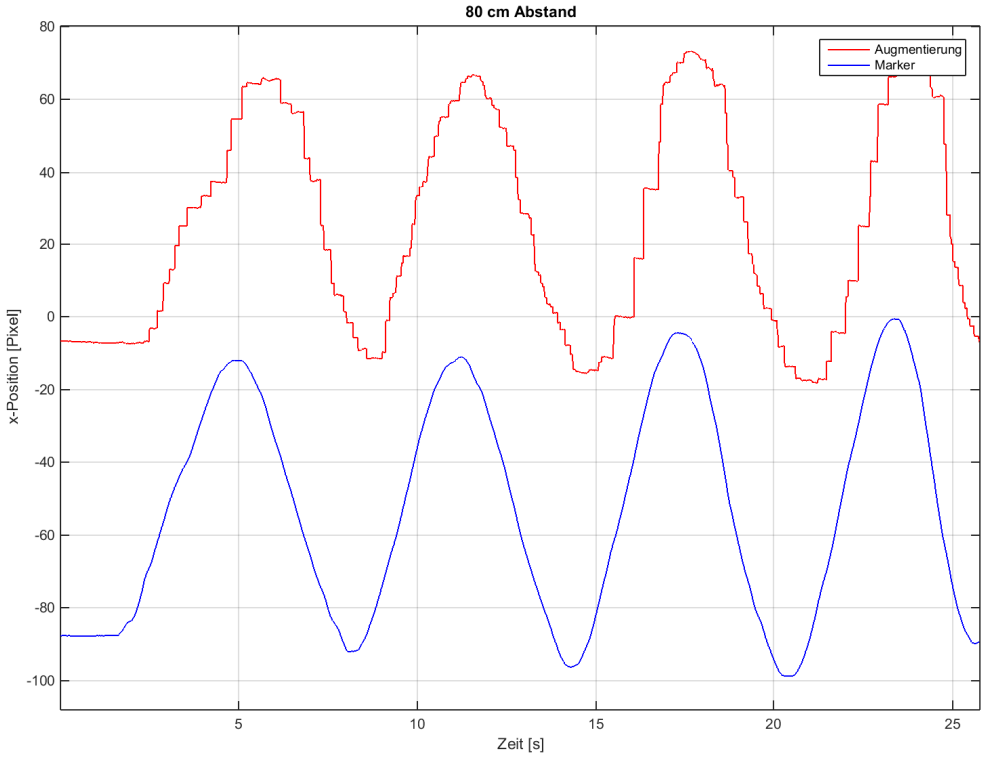

Abbildung A.4: 40 cm Abstand

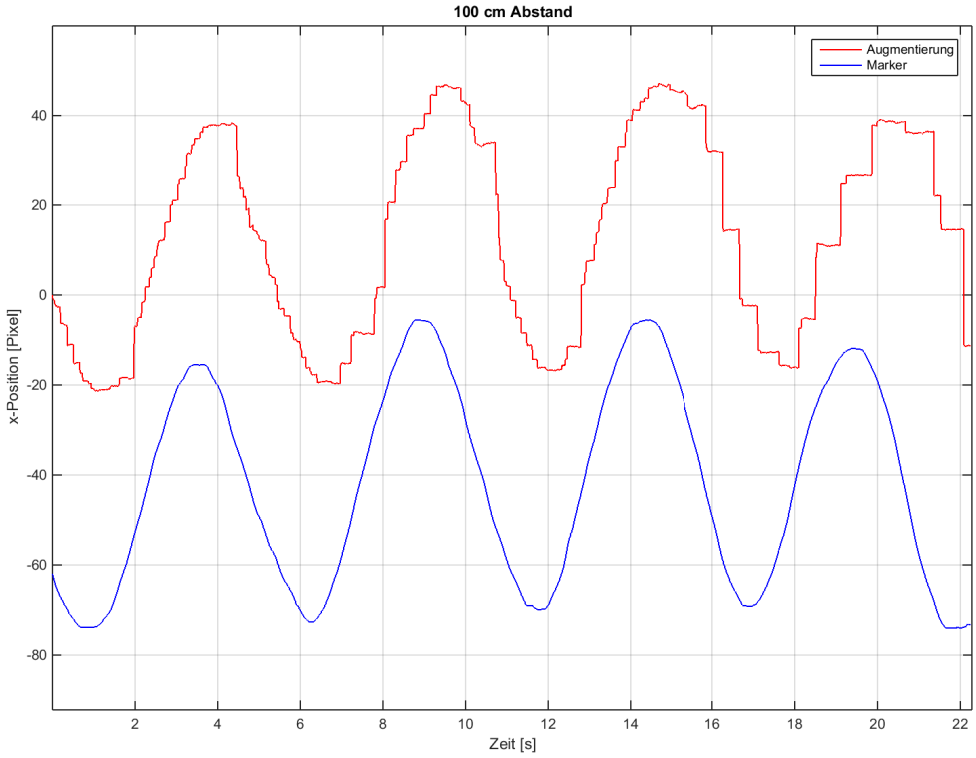

Abbildung A.5: 100 cm Abstand

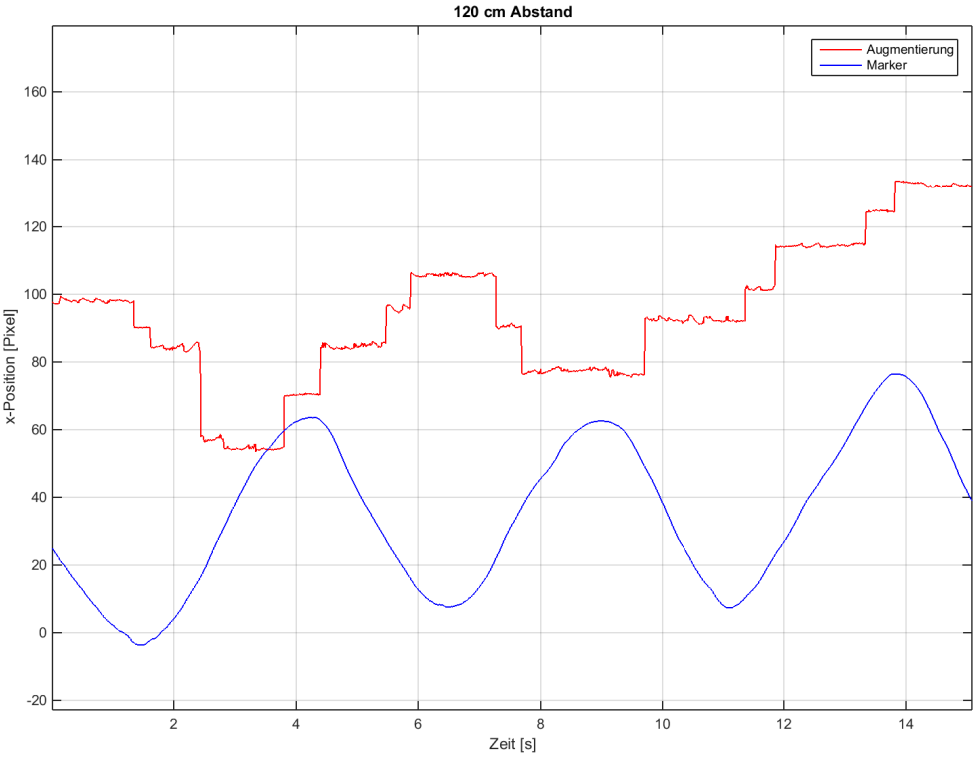

Abbildung A.6: 120 cm Abstand

A.2 Zeitlicher Jitter

	Abstand [m]				
	0.2	**0.4**	**0.6**	**0.8**	**1.0**
Zeitlicher Jitter [ms]	51.8668	146.1036	124.5045	148.4251	193.1228

	1.2	**1.4**	**1.6**	**1.8**	**2.0**
Zeitlicher Jitter [ms]	1771.2739	–	–	–	–

Tabelle A.2: Übersicht des Median des gemessenen zeitlichen Jitters in Abhängigkeit von dem Abstand zwischen Brille und Marker

A.3 Räumlicher Jitter

	Abstand [m]				
	0.2	**0.4**	**0.6**	**0.8**	**1.0**
Jitter in **x**-Richtung [Pixel]	0.0154	0.0388	0.0308	0.0255	0.0239
Jitter in **y**-Richtung [Pixel]	0.0412	0.0402	0.0762	0.0701	0.0575
Jitter in **z**-Richtung [cm]	0.0208	0.0306	0.0314	0.0508	0.0581

	1.2	**1.4**	**1.6**	**1.8**	**2.0**
Jitter in **x**-Richtung [Pixel]	0.0284	0.1332	0.5383	0.5736	0.3461
Jitter in **y**-Richtung [Pixel]	0.0984	0.2071	0.2297	0.3754	0.1541
Jitter in **z**-Richtung [cm]	0.0636	0.1349	0.1465	0.1743	0.0885

Tabelle A.3: Übersicht des Median des gemessenen räumlichen Jitters in Abhängigkeit von dem Abstand zwischen Brille und Marker (linkes Bild)

	Abstand [m]				
	0.2	**0.4**	**0.6**	**0.8**	**1.0**
Jitter in **x**-Richtung [Pixel]	0.0144	0.0363	0.0291	0.0240	0.0224
Jitter in **y**-Richtung [Pixel]	0.0395	0.0390	0.0751	0.0701	0.0581
Jitter in **z**-Richtung [cm]	0.0208	0.0306	0.0314	0.0508	0.0581

	1.2	**1.4**	**1.6**	**1.8**	**2.0**
Jitter in **x**-Richtung [Pixel]	0.0284	0.1343	0.5251	0.5620	0.1370
Jitter in **y**-Richtung [Pixel]	0.0981	0.1966	0.2231	0.3650	0.1541
Jitter in **z**-Richtung [cm]	0.0636	0.1348	0.1463	0.1740	0.0885

Tabelle A.4: Übersicht des Median des gemessenen räumlichen Jitters in Abhängigkeit von dem Abstand zwischen Brille und Marker (rechtes Bild)

A.4 Benutzerstudie zur Interaktion

User Study – Augmented Reality and Gesture Control

Subject:	Date:

Mental Demand How mentally demanding was the task?

☐ ☐ ☐ ☐ ☐

Very low Very high

Physical Demand How physically demanding was the task?

☐ ☐ ☐ ☐ ☐

Very low Very high

Performance How successful were you in accomplishing what you were asked to do?

☐ ☐ ☐ ☐ ☐

Perfect Failure

Effort How hard did you have to work to accomplish your level of performance?

☐ ☐ ☐ ☐ ☐

Very low Very high

Frustration How insecure, discouraged, irritated, stressed and annoyed were you?

☐ ☐ ☐ ☐ ☐

Very low Very high

1 of 3

User Study – Augmented Reality and Gesture Control

Augmentation	How was the impression of the augmentation?			
☐	☐	☐	☐	☐
Very good				Very bad
Gesture Detection	How well were the gestures detected?			
☐	☐	☐	☐	☐
Very good				Very bad
Gesture Lag	How long was the delay between the detection of a gesture and the change of the 3D-object?			
☐	☐	☐	☐	☐
Short				Long
Augmentation Lag	How was the lag of the augmentation? (Smoothness)			
☐	☐	☐	☐	☐
Very low				Very high
Marker Distance	How well was the marker detected below 1 m distance?			
☐	☐	☐	☐	☐
Very good				Very bad
Marker Distance	How well was the marker detected above 1 m distance?			
☐	☐	☐	☐	☐
Very good				Very bad

User Study – Augmented Reality and Gesture Control

Comfort How comfortable are the glasses?

☐	☐	☐	☐	☐
Very comfortable				Not comfortable

Intuitivity How intuitive was the handling of the task?

☐	☐	☐	☐	☐
Very intuitive				Not intuitive

Comments: