Lehrstuhl für Mensch-Maschine-Kommunikation
der
Technischen Universität München

Multimodale Interaktion in Augmented Reality Umgebungen am Beispiel der Spieledomäne

Stefan Reifinger

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzender	Univ.-Prof. Dr.-Ing. Ulf Schlichtmann
Prüfer der Dissertation	1. Univ.-Prof. Dr.-Ing. habil. Gerhard Rigoll
	2. Univ.-Prof. Dr.-Ing. Werner Hemmert

Die Dissertation wurde am 05.06.2008 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 28.10.2008 angenommen.

Vorwort

Die vorliegende Arbeit wurde im Rahmen meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Lehrstuhl für Mensch-Maschine Kommunikation der Technischen Universität München erstellt. Mein besonderer Dank gilt daher meinem Doktorvater *Univ.-Prof. Dr.-Ing. habil. Gerhard Rigoll* für die Möglichkeit, diese Arbeit an seinem Lehrstuhl durchführen zu dürfen. Durch seinen Einsatz erst fand ich das Arbeitsumfeld vor, das mir den erfolgreichen Abschluss dieser Arbeit ermöglichte. Ebenfalls trug er durch seine wissenschaftliche Betreuung maßgeblich zum Gelingen dieser Arbeit bei.

Mein weiterer Dank gilt *Dipl.-Math. Gregor McGlaun*, der durch seinen Einsatz einen wesentlichen Teil zur Umsetzung des Arbeitsumfeldes beigetragen hat.

Besonderer Dank für eine sehr gute Zusammenarbeit am Lehrstuhl und ein inspirierendes Arbeitsklima gebührt meinen Kollegen, von denen ich *Dipl.-Ing. Markus Ablaßmeier*, *Dipl.-Ing. Tony Poitschke* und *Dipl.-Ing. Jürgen Gast* besonders hervorheben möchte, die dem Forschungsalltag durch ihr soziales Engagement stets mehr Leben gaben. Ebenfalls möchte ich dem Kollegen *Dipl.-Ing. Xin Wang* für seine Einführung in seinen Kulturkreis danken.

Für die immer einwandfreie Unterstützung in allen technischen Belangen danke ich den Herren *Dr.-Ing. Claus von Rücker*, *Peter Brand*, *Heiner Hundhammer* und *Erwin Ertl*.

Herzlicher Dank für ihre Beiträge geht auch an die Diplomanden *Felix Hoffmann*, *Manuel Huber*, *Marius Fahlbusch*, *Ruben Alvarez* und *Michael Klenk*, den Studien-/Bachelorarbeitern *Fabian Heinemann*, *Hannes Grüner*, *Tobias Knaup*, *Samira Salman* und *Andreas Rittinger*, sowie den interdisziplinären Projektarbeitern *Elisabeth Wolf*, *Marek Orschweski* und *Philipp Kemmeter*.

Ganz besonderer Dank geht an meine Familie, die mir durch ihre uneingeschränkte Unterstützung erst das Studium ermöglicht haben, ohne das diese Arbeit nicht möglich gewesen wäre.

München, im Juni 2008
Stefan Reifinger

Kurzzusammenfassung

Vergleicht man Computerspiele aus den Anfangszeiten mit denen, die heutzutage den aktuellen Stand der Technik repräsentieren, zeigt sich, dass innerhalb der letzten 30 Jahre eine immense Entwicklung statt gefunden hat. Sowohl die grafische Darstellung als auch die Komplexität der Interaktion hat sich enorm erhöht. Interaktion mit Computerspielen, unterteilt in Eingabe und Ausgabe, beschränkt sich nicht mehr auf die reine Darstellung einfacher Inhalte, die mit simplen Bedienelementen, wie etwa einem Joystick, gesteuert werden können. Interaktion hat sich von einer reinen Ein-/Ausgabe zu einem immersiven Erlebnis gewandelt, mit Möglichkeiten, die sich immer weiter in die bisherigen Interaktionsgewohnheiten des Menschen einfügen.

Daher fokusiert diese Arbeit die Weiterentwicklung aktueller Interaktionsmöglichkeiten hinsichtlich der Eingabe und auch der Ausgabe. Gegenstand der Arbeit ist also die Entwicklung und Untersuchung neuartiger Möglichkeiten der Interaktion, speziell in der Spieledomäne.

Auf der Eingabeseite werden drei verschiedene Ansätze zur Gestenerkennung des Nutzers umgesetzt. Dabei stützt sich ein Ansatz auf die Verwendung von Trackingdaten, die mittels eines Infrarot-Trackingsystems gewonnen werden. Bei zwei Ansätzen erhält man die Daten, die Rückschlüsse auf Gesten des Nutzers ziehen lassen, durch die Verwendung von Neigungs- und Gyrosensoren.

Auf der Ausgabeseite wird die Visualisierung von einer rein virtuellen Darstellung auf die Darstellung mittels Augmented Reality erweitert, die eine nahtlose Einbindung der Spieleumgebung in die reale Umgebung des Nutzers ermöglicht. Ebenfalls wird die akustische Ausgabe von einer Stereodarbietung auf die Verwendung von virtueller Akustik erweitert. Diese Veränderungen der Ausgabe ermöglichen eine Integration des Spieles in die reale Umgebung des Nutzers und resultieren in einer höheren Immersion des Spielers in das Spielgeschehen.

Diese Ausgabemöglichkeiten werden ebenso wie die Eingabemöglichkeiten in einer eigens dafür entwickelten Systemarchitektur integriert, die mittels Netzwerkkommunikation eine modulare und somit performance-orientierte Umsetzung von (Spiele-)Applikationen ermöglicht.

Um die Interaktion nicht rein auf den Austausch zwischen Mensch und Maschine zu begrenzen, wird in dieser Arbeit ebenfalls der Aspekt der Mensch-Mensch Interaktion sowie die Interaktion zwischen der realen und der virtuellen Umgebung des Nutzers berücksichtigt.

Dazu wird ein Videokonferenzsystem entwickelt, das es einer beliebigen Anzahl von Teilnehmern ermöglicht, audiovisuell in Interaktion zu treten, und in die Spieleapplikation eingebettet werden kann.

Weiterhin wird eine Kollisionserkennung basierend auf Algorithmen der Bildverarbeitung entwickelt, die es ermöglicht, reale Objekte in der Umgebung des Nutzers zu erkennen und in die virtuelle Umgebung einzubringen. Auf diese Weise lassen sich Wechselwirkungen zwischen realen und virtuellen Objekten in die Spieleapplikation integrieren.

Zur Veranschaulichung der entwickelten Interaktionsmöglichkeiten werden in dieser Arbeit zwei Spieleapplikationen entworfen. Hierbei werden ein virtuelles Billardspiel (BillARd) und ein virtuelles Bowlingspiel (Bowl-AR-ama) beschrieben. Diese beiden Applikationen vereinen die entstandenen Möglichkeiten der Interaktion dieser Arbeit und dienen zu deren Demonstrationszwecken.

Um die Auswirkungen der Interaktionsmöglichkeiten besser einschätzen zu können, werden in dieser Arbeit die Kernelemente in Nutzerstudien untersucht. Hierbei zeigt sich, dass die Verbindung von akustischer und visueller Erweiterung der realen Umgebung die Immersion des Nutzers erheblich verstärken kann. Ebenfalls wird ersichtlich, dass die Bedienung mittels Gesten den Grad der Eintauchung in die virtuelle Umgebung stark steigert, wobei sich jedoch für jede Interaktionsart spezifische Vor- und Nachteile ergeben, die in dieser Arbeit aufgezeigt werden.

Diese Arbeit stellt also die Grundlage für weitere Entwicklungen multimodaler Interaktion in der Spieledomäne dar.

Inhaltsverzeichnis

KAPITEL 1

Einleitung

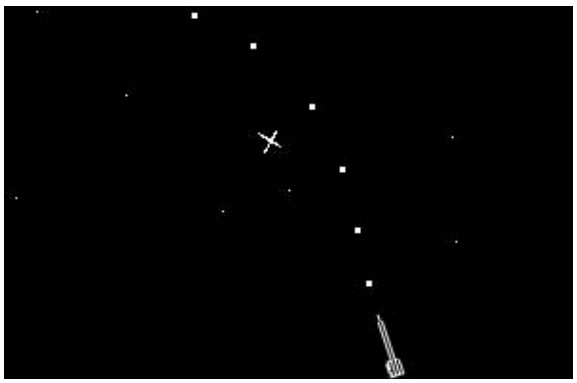
Es ist Dezember im Jahre 1972. Im einem Wohnzimmer sitzt ein Junge mit einem kleinen Kästchen in der Hand vor dem Fernseher, auf dem sich ein weißer Fleck hin und her bewegt. Es ist still im Wohnzimmer, nur das Klacken des Kästchens ist zu hören. „Pong“ nennt sich dieses revolutionäre Spiel für zu Hause, das den Jungen und seinen Freund für Stunden fesselt. Heute sitzt dieser Junge im Wohnzimmer und sieht seinem Sohn zu, wie er am Fernseher mit der Spielekonsole spielt. Auch heute noch fesselt das Spiel für Stunden. Jedoch ist der weiße bewegliche Fleck auf dem Bildschirm einer bunten Vielfalt gewichen. Heute sitzt sein Sohn nicht wie er damals still mit einem kleinen Kästchen in der Hand, sondern er springt vor den Fernseher auf und ab und bewegt dazu seine Hände. Auf die Frage, was er denn spiele, antwortet der Junge, dass er die Fensterscheiben reinigen solle. Aus der Spielekonsole, die einst allein durch einen weißen beweglichen Fleck begeisterte, ist ein multimodales Ereignis geworden.

Die vorliegende Arbeit hat die Zielsetzung, den Anfang für einen weiteren großen Schritt in der Entwicklung der Spiele zu schaffen. Die nächste Generation soll nicht mehr im Wohnzimmer vor dem Fernseher gefesselt sitzen, sondern neue Spiele spielen, die sich nahtlos in die Umgebung der Spieler einfügen. Dazu sollen sich Spiele in die Umgebung des Nutzers eingliedern und dabei einen Zusammenhang zwischen realer und virtueller Umgebung herstellen. Die Interaktion fügt sich hierbei ebenfalls in die reale Umgebung des Nutzers ein, die Ein- und Ausgabe bedient sich dabei Mitteln, die an die natürliche Interaktion des Menschen mit seiner realen Umgebung anlehnen.

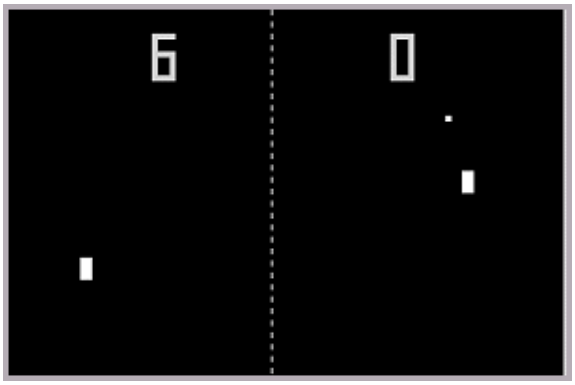
Zu Beginn dieser Arbeit steht ein Überblick über die Entwicklung der Computerspiele, angefangen 1958 bis heute. Dabei liegt der Schwerpunkt auf den Interaktionsmöglichkeiten, die sich im Laufe der Zeit entwickelt haben.

1.1 Die Geschichte der Computerspiele

Das erste Computerspiel der Welt wurde 1958 entwickelt. „Tennis for Two“ bestand aus einem Analogcomputer und einem Oszillographen zur visuellen Darstellung. Zur Steuerung standen den Nutzern zwei kleine Kästchen mit zwei Knöpfen zur Verfügung, mittels derer der Schlag sowie der Schlagwinkel eingestellt werden konnten. 1962 verbreitete sich „Spacewar!“ [75] aufgrund extremer Kosten für Hardware nur an amerikanischen Universitäten. Dabei diente



(a) „Spacewar“ (1962) [75]



(b) „Pong“ (1972) [32]

Abbildung 1.1: Beginn der Computerspiele: „Spacewar“ und „Pong“

das Spiel primär als Demonstrationsprogramm für die Leistungsfähigkeit der Hardware und simulierte ein Gefecht zweier Raumschiffe (vgl. Abb. 1.1(a)).

„Spacewar!“ war ebenfalls für zwei Spieler ausgelegt und bot Interaktionsmöglichkeiten in Form von Knöpfen, um verschiedene Funktionalitäten (beispielsweise die Navigation des Raumschiffes) zu steuern. 1972 wurde von „Atari“ das Computerspiel „Pong“ [32] vorgestellt und der Öffentlichkeit in Form von Spielautomaten, deren Entwicklung aus der preisgünstigen Fernsehtechnologie hervorging, zugänglich gemacht. „Pong“ (abgeleitet aus dem englischen „Ping Pong“) simuliert in einfachster Form ein Tischtennisspiel (vgl. Abb. 1.1(b)). Die Interaktionsmöglichkeit für beide Spieler bestand aus einem Drehknopf, der die Position des Schlägers steuerte.

Aufgrund der rasanten Entwicklung der Computerspiele kamen Ende der 70er Jahre die ersten Spielekonsolen für den Heimanwender auf den Markt. Mit der Einführung von „Space Invaders“ für den „Atari 2600“ erfolgte der Durchbruch der Spielekonsolen. „Space Invaders“ war eines der ersten Computerspiele mit farbiger Darstellung. Diese wurde durch sogenannte Overlay-Folien erzeugt, die vor dem Bildschirm befestigt waren. Ebenfalls erweiterte „Space Invaders“ erstmalig die monomodale Ausgabe auf eine multimodale Ausgabe, da nicht mehr nur eine rein visuelle Ausgabe erfolgte, sondern die akustische Ausgabe integriert wurde. Diese akustische Ausgabe war bereits interaktiv und änderte sich, wenn auch nur rudimentär, angelehnt an das aktuelle Spielgeschehen. Auch die Interaktionsmöglichkeiten für den Nutzer wurden erweitert. Während bei „Pong“ zur Steuerung noch ein Drehknopf verwendet wurde, erfolgte die Steuerung bei „Space Invaders“ durch einen 2-Wege Joystick und einen Knopf. Diese Möglichkeit der Interaktion wurde für „Pacman“ [13] 1980 auf einen 4-Wege Joystick erweitert. Ebenfalls wurde durch „Pacman“ die Wechselwirkung zwischen Computer und Nutzer eingeführt. Gegner, in Form von Spielfiguren, erkennen die Eingaben des Nutzers und reagieren darauf. Weiterhin erfolgte bei „Pacman“ (vgl. Abb. 1.2(a)) die grafische Ausgabe nicht mehr in Schwarz/Weiß, sondern in Farbe.

In den 80er Jahren wurde der Personal Computer eingeführt. Dieser war anfangs technisch jedoch noch nicht in der Lage, als Konkurrenz zu Spielekonsolen aufzutreten. Aufgrund mangelnder Leistung und fehlender Hardware (beispielsweise die Möglichkeit von akustischer Ausgabe) dominierten Spielekonsolen. 1983 führte „Nintendo“ die 8-Bit Konsole „Famicon“ ein. Die Interaktionsmöglichkeiten des Nutzers stiegen stark an. So wurden Eingabegeräte in Form von Pistolen, Handschuhen oder speziellen Fussmatten eingeführt. Mit der Einführung des „C64“ 1985 begann die Entwicklung von Computerspielen auch für den Einsatz auf

(a) „Pacman“ (1980) [13]

(b) „The Eidolon“ (1985) [50]

Abbildung 1.2: Weiterentwicklung der Computerspiele: „Pacman“ und „The Eidolon“

Personal Computern. Die Interaktionsmöglichkeiten waren hier auf visuelle und akustische Ausgabe und mechanische Eingabe in Form von Joysticks, Paddlen, Maus oder einer Pistole beschränkt. Für den „C64“ erschien 1985 der erste Ego-shooter „The Eidolon“ [50] (vgl. Abb. 1.2(b)). 1987 erschien der „Amiga 500“, der ebenfalls als Personal Computer konzipiert war. Gegen Ende der 80er Jahre kamen die ersten mobilen Spielekonsolen, wie etwa der „Game Boy“ von „Nintendo“ auf den Markt.

In den 90er Jahren koexistierten Spielekonsolen und Personal Computer. Beide Technologien entwickelten sich rasant und ermöglichten zunehmend eine realistischere visuelle Darstellung. Ebenfalls wuchs die Nutzung des neuen Mediums „Internet“. Für die Computerspiele bedeutete dies eine beinahe beliebige Erweiterung der Mehrspielerfunktionalität. 1991 startet mit „Neverwinter Nights“ [14] das erste „MMORPG“ (Massively Multiplayer Online Role-Playing Game). 1997 erschien „Ultima Online“ [33], eines der bis dahin größten Mehrspieler-Computerspiele mit bis zu 140.000 Spielern gleichzeitig. Übernommen aus der Luftfahrt wurde die mechanische Ausgabe in Form von „Force Feedback“-Geräten eingeführt. Diese Technologie ermöglichte die Erweiterung der bis dahin auf visuelle und akustische Ausgabe beschränkten Darstellung um die mechanische Komponente. Angepasst an bestimmte Spielsituationen konnten diese Geräte mechanische Bewegungen in Form von Vibrationen, Stößen oder Gegenkraft des Nutzers an den Spieler übertragen. Die visuelle Ausgabe wurde durch Verwendung von 3-D-Brillen nun auch stereo möglich. Spiele mit Sprachsteuerung konnten sich jedoch aufgrund mangelnder Zuverlässigkeit der Spracherkennungssysteme immer noch nicht durchsetzen.

1.2 Motivation

Die visuelle Darstellung von Computerspielen ist in den letzten 50 Jahren stetig verbessert worden. Heutige Computerspiele bestechen mit einer annähernd realistischen Grafik, die durch Verwendung von entsprechender Hardware auch stereoskopisch dargestellt werden kann.

Ebenfalls bedienen sich Computerspiele der Multimodalität. Auf der Eingabeseite kommen mechanische Eingabegeräte, wie etwa Tastatur oder Maus beim Heimcomputer zum Einsatz. Darüber hinaus wurden applikationsspezifische Eingabegeräte entwickelt, wie zum Beispiel ein Lenkrad samt Pedalerie für den Einsatz in Fahrsimulationen.

Auf der Ausgabeseite nimmt die Zahl der Modalitäten ebenfalls zu. Neben der primären visuellen Ausgabe existiert die akustische Ausgabe. Aber auch die haptische Wahrnehmung des Nutzers wird angesprochen, da Eingabegeräte gleichzeit als Ausgabegerät agieren, indem sie

mechanische Bewegungen in Form von Gegenkraft oder Vibrationen gezielt zur Unterstützung der visuellen und akustischen Ausgabe verwenden.

Die Entwicklung einer nahezu exakten Abbildung der Realität durch Simulation, die möglichst alle Sinne des Nutzers anspricht, führt in logischer Konsequenz dazu, dass zukünftige Computerspiele mit der Realität des Nutzers verschmelzen. So soll der Nutzer in die Lage versetzt werden, Computerspiele interaktiv in seiner realen Umgebung einzusetzen. Dabei soll nicht nur die Darstellung mit seiner Umgebung verschmelzen, sondern auch die Eingabe soll sich an den Benutzer anpassen. Die Interaktion des Nutzers ist hierbei also nicht mehr beschränkt auf die reale oder die virtuelle Umgebung, sondern ermöglicht die gleichzeitige Interaktion mit beiden Umgebungen zur selben Zeit und im selben Raum. Virtuelle und reale Umgebung verschmelzen also zu einer gemeinsamen gemischten Umgebung für den Nutzer.

Daher ist es erstrebenswert, die Möglichkeiten der natürlichen Informationsabgabe des Menschen, wie etwa Gesten oder Sprache, als Eingabeformen in Computerspiele zu integrieren, um auch hier eine möglichst hohe Einbindung in die reale Umgebung des Nutzers zu erreichen. Gleichzeitig sollen die Möglichkeiten der Mehrspielerfunktion beibehalten bzw. erweitert werden.

1.3 Vorarbeiten und aktueller Stand der Technik

Augmented Reality (AR) bezeichnet die Anreicherung der realen Umgebung des Menschen mit virtueller Information. Diese Information wird mit der Realität überlagert und dem Menschen präsentiert. Derzeit beschränkt sich AR in den meisten Fällen auf eine monomodale Darstellung in Form einer rein visuellen Anreicherung. In das Blickfeld des Menschen wird dabei virtuelle Information in Form von Objekten oder Texten eingebracht. Als beispielhafte Anwendung sind die Unterstützung des Arztes in der Medizin [45] oder die Unterstützung eines Produktentwicklers im Design [52] zu nennen. Ebenfalls bestehen bereits Ansätze für eine multimodale Darstellung, d.h. virtuelle Informationen werden nicht nur visuell, sondern auch akustisch dargeboten [55]. Bei den genannten Anwendungen steht jedoch nicht die Interaktion von Mensch und AR-System im Vordergrund, vielmehr hat AR hier rein darstellenden Charakter.

Die Interaktion zwischen Mensch und Computer erfolgt im allgemeinen entweder monomodal oder multimodal, d.h. entweder mit nur einer einzigen Möglichkeit der Eingabe oder mit mehreren Eingabemöglichkeiten. Untersuchungen hierzu haben ergeben, dass bei der multimodalen Interaktion die Eingabemöglichkeiten (zum Beispiel in Form von Sprache oder Gestik) auf verschiedene Weise vom Menschen verwendet werden. Im einfachsten Fall werden die Eingabemöglichkeiten alternierend, im kompliziertesten Fall synergetisch verwendet [77]. Erste Untersuchungen speziell in der AR-Domäne haben gezeigt, dass die multimodale Verwendung von Eingabegeräten eine robustere Erkennung der Benutzerintention zulässt als die Verwendung von Einzelerkennern [38]. Für die weitere Entwicklung von AR-Systemen ist es daher notwendig, die Untersuchung der multimodalen Interaktion mit AR-Systemen voranzutreiben und aus den Ergebnissen verbesserte Interaktionsmöglichkeiten für den Menschen abzuleiten.

Erste Schritte, AR in der Spieledomäne zu etablieren, wurden mit der Entwicklung von ARQuake [68] unternommen. Hierbei wurde ein populärer Ego-shooter („Quake“) in die AR portiert. Die Interaktion beschränkt sich hierbei jedoch auf die Navigation in der Spieleumgebung und der Interaktion mittels einer monomodalen Eingabeform (virtuelle Waffe in der Hand). Bis heute folgten einige Umsetzungen einfacher Computerspiele, wie beispielseise „TankWar“ [61]. Weitere Möglichkeiten, Spiele innerhalb einer AR Umgebung umzusetzen,

Abbildung 1.3: Sony's Eyetoy für die Playstation 2 [23]

finden sich beispielsweise mit „Butterfly Effect: An Augmented Reality Puzzle Game“ [62], „MonkeyBridge“ [7] oder „Ninja on a Plane“ [11]. Alle diese Spiele bilden jedoch virtuelle Computerspiele in der AR ab. Simulationen von realen Spielen fanden bisher nur wenig statt, wie etwa am Beispiel von Tennis [28].

Der kommerzielle Einsatz von interaktiver AR beginnt. So hat die „Playstation 2„ von „Sony“ mit seinem „EyeToy“ ein System vorgestellt, das es dem Nutzer erlaubt, ein Computerspiel mittels der Bewegung von Kopf und Armen zu steuern [23].

Dazu wird vor dem Nutzer eine Kamera aufgebaut, die dessen Bewegungen beobachtet. Diese Bewegungen werden dann in das Spiel umgesetzt. So kann der Nutzer in einem Spiel beispielsweise einen virtuellen Kopf mit seinen Händen waschen (vgl. Abb. 1.3) oder mit seinem Kopf einen Fußball in der Luft halten. Der Nutzer wird dabei ebenfalls auf dem Bildschirm dargestellt und so optisch in das Spiel integriert.

Aufgrund dieser Vorarbeiten und der beginnenden kommerziellen Entwicklung ergibt sich ein großer Forschungsbedarf in interaktiver AR, wobei hier insbesondere das Szenario der Computerspiele sehr vielversprechend erscheint.

1.4 Lösungsansatz

In vielen Bereichen der Industrie wird bereits seit Jahren die Technologie der Virtual Reality (VR) (vgl. Kapitel 2.1.2) verwendet. Diese Technik ermöglicht die Visualisierung von künstlich generierten Objekten in Echtzeit und in stereoskopischer Darstellung. Im Designprozess entstandene virtuelle Modelle können so kosten- und zeitgünstig betrachtet und bewertet werden. Die Herstellung realer Prototypen kann somit im frühen Stadium des Designprozesses minimiert werden.

Neben der VR hat sich die AR (vgl. Kapitel 2.1.1) entwickelt. Diese Technologie ist verwandt mit der VR. Sie ermöglicht die grafische Darstellung ähnlich der VR, jedoch mit dem Unterschied, dass virtuelle Modelle nicht losgelöst von der realen Umwelt existieren müssen, sondern mit dieser in Wechselwirkung treten können. Somit wird die Darstellung virtueller Objekte innerhalb des realen Umfeldes des Nutzers ermöglicht. Dabei beschränkt sich die Darstellung nicht auf eine ausschließliche Überlagerung von realer und virtueller Umgebung, sondern die virtuelle steht in direkter Beziehung zur realen Umwelt. Auf diese Weise erscheint

die virtuelle Umgebung dem Nutzer als integrierter Bestandteil seiner realen Umgebung.

Vergleicht man die Anwendungsgebiete der beiden Technologien mit dem Streben der Computerspiele nach immer stärkerer Realitätsnähe, zeigt sich, dass der Einsatz von VR und AR in der Spieledomäne eine bahnbrechende Entwicklung ist. Der Einsatz von VR ist bei aktuellen Computerspielen (zum Beispiel bei sogenannten „Ego-Shootern“) nicht mehr weg zu denken. Lediglich die Integration der virtuellen Spieleumgebung in die reale Umwelt des Nutzers, welche die Verwendung von AR ermöglicht, fehlt hier. Der Nutzer wird also innerhalb seiner realen Umgebung zum „Helden“ des virtuellen Spieles. Aufgrund der Mobilität von AR-Systemen ist der Nutzer flexibel in der Wahl der Spieleumgebung. Durch Koppelung mehrerer Systeme ist ebenfalls die Möglichkeit der gemeinsamen Nutzung des Computerspieles mit Freunden gegeben. Hierbei kann die gemeinsame Verwendung sowohl räumlich gemeinschaftlich als auch getrennt vorgenommen werden. Durch die Integration von natürlichen Eingabeformen, wie Sprache oder Gesten, kann in Kombination mit AR eine Spieleumgebung geschaffen werden, die sich im Idealfall nahtlos in die reale Umgebung des Nutzers eingliedert und somit ein höchstes Mass an Immersion erzeugt.

1.5 Gliederung der Arbeit

Das folgende Kapitel 2 beginnt mit der Einführung der Begriffe des Arbeitstitels, erläutert danach Grundlagen von Koordinatensystemen und Koordinatentransformationen, vermittelt die notwendige Hardware für die Darstellung virtueller Umgebungen und stellt die zugrundeliegende Softwareplattform vor, die in der Arbeit entwickelt und verwendet wurde.

Die Interaktionsmöglichkeiten zwischen Mensch und Maschine werden in den Kapiteln 3 und 4 behandelt. Hier wird in den beiden Kapiteln unterschieden zwischen der Interaktion von der Maschine zum Menschen und vom Menschen zur Maschine. Dabei werden Verfahren vorgestellt, die Augmented und Virtual Reality ebenso wie virtuelle Akustik als Ausgabe einbinden. Für eine Eingabe werden drei Möglichkeiten zur Gestenerkennung präsentiert.

Kapitel 5 beschreibt ein Verfahren, Interaktion zwischen realen und virtuellen Objekten zu ermöglichen, sowie eine Umsetzung eines Videokonferenzsystem für interpersonelle Interaktion.

Anschließend werden zwei Beispielapplikationen vorgestellt, die die zuvor vorgestellten Interaktionsmöglichkeiten in einer Applikation vereinen. *BillARd - Augmented Reality Billard* in Kapitel 6.1 und *Bowl-AR-ama - Augmented Reality Bowling* in Kapitel 6.2.

Nach der Beschreibung dieser Applikationen werden auf dem System durchgeführte Evaluierungen und deren Ergebnisse in Kapitel 7 vorgestellt.

Abgeschlossen wird diese Arbeit dann mit einem Resumee in Kapitel 8, das die Arbeit kurz zusammenfasst, deren Ergebnisse diskutiert und einen Ausblick auf weiterführende Forschungsschwerpunkte gibt.

KAPITEL 2

Thematische Grundlagen

Dieses Kapitel beschäftigt sich mit den Grundlagen, die zum Verständnis des Arbeitstitels notwendig sind. Hierzu zählen die Definitionen von VR, AR sowie deren thematische Abgrenzung gegeneinander. Für die Berechnung von virtuellen Umgebungen sind Koordinatensysteme (KOS) und Koordinatentransformationen unerlässlich und werden daher ebenso kurz vorgestellt wie diesen Systemen zugrunde liegende Hardware-Komponenten. Ebenfalls thematisiert das Kapitel die Grundlagen der multimodalen Interaktion zwischen Mensch und Maschine und der daraus resultierenden Verknüpfung mehrerer Interaktionskanäle. Dabei versucht dieses Kapitel jedoch nicht, technische Details einzelner Technologien zu erläutern, sondern deren fundamentale Funktionsweise aufzuzeigen. Für die Umsetzung der einzelnen Methoden im Detail in dieser Arbeit sei hiermit auf die entsprechenden Kapitel verwiesen.

2.1 Augmented und Virtual Reality

Im folgenden Kapitel werden die Grundzüge der AR und VR erläutert, deren grundsätzlicher Systemaufbau sowie deren Abgrenzung zueinander.

2.1.1 Augmented Reality

AR bescheibt die Überlagerung von realen und virtuellen Informationen. Der Nutzer wird hierbei nicht von der realen Umgebung abgeschnitten, vielmehr wird die virtuelle Umgebung in die reale Umgebung eingebettet. Der Nutzer kann daher nicht nur mit der realen sondern auch mit der virtuellen Umgebung interagieren. Der Wahrnehmungsbereich des Nutzers umfasst hier also sowohl die reale Umgebung als auch die virtuelle Umgebung. Ebenfalls steht die reale Umgebung in direktem Bezug mit der virtuellen Umgebung. Es werden also beide Umgebungen nicht nur überlagert, sondern in der Darstellung inhaltlich richtig kombiniert. Verdeutlichen lässt sich dieser Zusammenhang durch das „Realitäts-Virtualitäts-Kontinuum“ (siehe Abb. 2.1) nach [57].

Das „Realitäts-Virtualitäts-Kontinuum“ zeigt den Zusammenhang zwischen der realen und der virtuellen Umgebung. Der Bereich dazwischen wird als gemischte Realität bezeichnet. Je nach Ausgangslage der Anreicherung spricht man von Erweiterter Realität bzw. Erweiterter Virtualität. Wird einer realen Umgebung virtuelle Information zugefügt, spricht man von Erweiterter Realität.

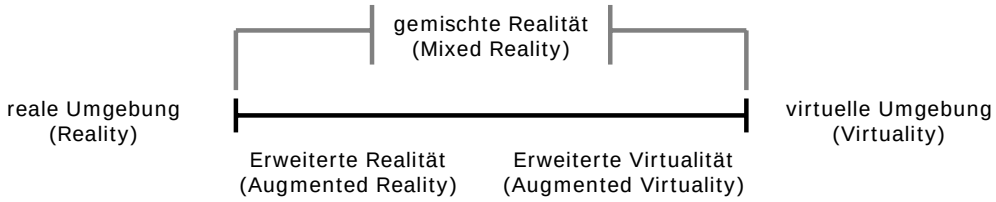


Abbildung 2.1: Realitäts-Virtualitäts-Kontinuum nach [57]

AR muss nach [5] drei Anforderungen erfüllen. Die Darstellung von realer und virtueller Umgebung muss kombiniert sein. Die realen und virtuellen Informationen müssen 3-dimensional zueinander in Bezug stehen. Zudem muss die Darstellung und die Interaktivität in Echtzeit erfolgen. Aus der letzten Anforderung ergibt sich, dass nicht alle Fälle von gemischter Realität zur AR gezählt werden dürfen. Filme, die auf virtuelle Ergänzung der Realität setzen, wie zum Beispiel „Roger Rabbit“, werden folglich nicht als AR gewertet.

Ein AR-System besteht im Allgemeinen aus drei grundlegenden Bestandteilen. Die aktuelle Position und Orientierung des Nutzers im Raum erfasst ein Trackingsystem (vgl. Kap. 2.3.1). Aus diesen Daten berechnet das Visualisierungssystem den aktuellen virtuellen Sichtbereich des Nutzers. Zusammen mit den zugrundeliegenden 3-D-Modellen sowie derer Position und Orientierung innerhalb des Raums wird der Sichtbereich visualisiert (vgl. Kap. 2.3.2). Die Kombination von realem und virtuellem Sichtbereich erfolgt über zwei verschiedene Technologien. Die erste Variante erlaubt dem Nutzer, den realen Sichtbereich direkt wahr zu nehmen. Hierbei erfolgt die Überlagerung mittels eines optischen Mediums, das sowohl die direkte Wahrnehmung der realen Umgebung als auch von virtuellen Informationen erlaubt. Die zweite Variante kombiniert den realen und virtuellen Sichtbereich, ohne dass eine direkte Wahrnehmung der realen Umgebung möglich ist. Hierbei wird der aktuelle, reale Sichtbereich des Nutzers, der über eine Kamera erfasst wird, mit dem virtuellen Sichtbereich überlagert. Das kombinierte Ausgangsbild wird dem Nutzer anschließend auf einem Ausgabemedium dargestellt. Für die Umsetzung in dieser Arbeit wurde Variante zwei gewählt. Dem Nutzer eines AR-Systems stehen neben der Ausgabe optional auch Eingabegeräte zur Verfügung, mittels derer die aktuelle virtuelle Umgebung manipuliert werden kann. Diese Eingaben fließen auch hier in die Visualisierung ein und führen somit zu wahrnehmbaren Veränderungen der Ausgabe (vgl. Abb. 2.2).

Der Einsatz von AR nahm seine Anfänge beim Militär. Soldaten werden bis heute zusätzliche Informationen über ihr Helmvisier eingeblendet [86]. Ebenfalls können Soldaten Navigationsunterstützung erhalten, indem Informationen bezüglich der Routenplanung in unbekanntem Gelände mittels eines Displays angezeigt und mit ihrer Umwelt überlagert werden können (beispielsweise das „BARS“ System [37] oder [48]).

Ein weiteres Anwendungsfeld findet sich in der Medizin, in der Patientendaten während Operationen über den Patienten überlagert werden. So ist es möglich, zuvor aufgenommene Röntgendaten positionsgenau mit der zu operierenden Stelle zu überlagern [25].

Für den privaten Anwender zeigt sich die Verwendung von AR mehr und mehr bei der Übertragung im Fernsehen. Hierbei werden Informationen, die für den Zuschauer relevant sind, zum aktuellen Geschehen hinzugefügt. Dabei ist zu unterscheiden, ob diese Informationen statisch eingeblendet werden oder ob sie passend zur Szene überlagert werden. Erster Fall, beispielsweise die Einblendung von Rundenzeiten bei Motorsportübertragungen zählen per Definition nicht zu AR, da sie mit der Umgebung nicht in Bezug stehen. Zweiter Fall hingegen kann als AR betrachtet werden. Hier wird beispielsweise bei Übertragungen von

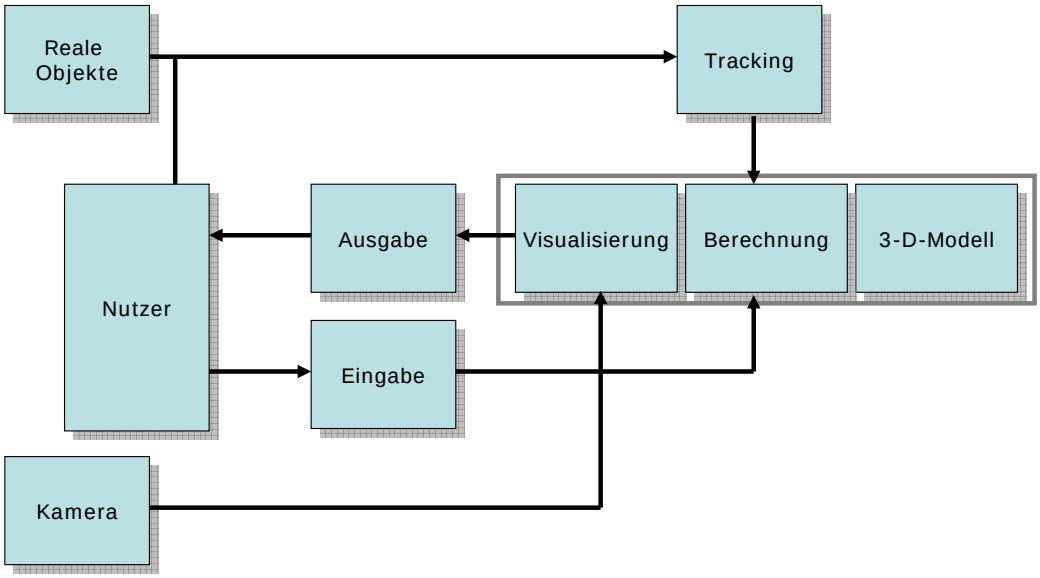


Abbildung 2.2: Schematischer Aufbau eines Augmented Reality Systems

Fußballspielen kontextabhängig Information zum Spielgeschehen hinzugefügt (beispielsweise wird die Tordistanz bei einem Freistoß anhand eines Pfeiles angezeigt [49]).

Auch im privaten Gamingsektor kommt Technologie, ähnlich der AR, mehr und mehr zum Einsatz. So ermöglicht die „Playstation 2“ mittels seiner „Eyetoy“ Technologie die Integration des Nutzers in das aktuelle Spielgeschehen [23].

2.1.2 Virtual Reality

VR beschreibt eine vollständig vom Computer generierte Umgebung, in die ein Nutzer eintauchen kann. Der Nutzer befindet sich zwar in der realen Umgebung, diese wird jedoch vollständig vom Nutzer abgeschirmt, sodass eine Interaktion des Nutzers nur noch mit der virtuellen Umgebung erfolgt. Der Wahrnehmungsbereich des Nutzers wird also auf die rein virtuelle Umgebung beschränkt. Ein VR-System besteht im Allgemeinen, wie ein AR-System, aus drei grundlegenden Bestandteilen. Ein Trackingsystem erfasst die aktuelle Position und Orientierung des Nutzers im Raum relativ zu einem fest definierten WeltKOS. Dem Visualisierungssystem stehen diese Informationen bezüglich des Nutzers zur Verfügung und es berechnet daraus den aktuellen Sichtbereich des Nutzers innerhalb der virtuellen Umgebung. Alle virtuellen Objekte liegen dem Visualisierungssystem als 3-D-Modell vor. Alle Objekte, die sich innerhalb des Sichtbereiches des Nutzers befinden, werden anschließend vom Visualisierungssystem berechnet und auf das Ausgabesystem geleitet. Auf diese Weise wird dem Nutzer der für ihn sichtbare Ausschnitt aus der virtuellen Umgebung zur Verfügung gestellt. Als Ausgabesystem werden im Allgemeinen herkömmliche Monitore für Desktop Anwendungen bis hin zu komplexen Anordnungen von Projektionswänden (sogenannte CAVEs) verwendet. Neben dem Ausgabesystem stehen dem Nutzer eines VR-Systems optional auch Eingabegeräte eines Eingabesystems zur Verfügung, mittels derer die aktuelle virtuelle Umgebung manipuliert werden kann. Diese Eingaben fließen in die Visualisierung ein und führen somit zu wahrnehmbaren Veränderungen der Ausgabe (vgl. Abb. 2.3). Übliche Eingabegeräte sind hierbei taktile Eingabegeräte, wie etwa eine Maus oder ein Datenhandschuh.

VR findet zahlreichen Einsatz in den verschiedensten Bereichen. So wird die virtuelle

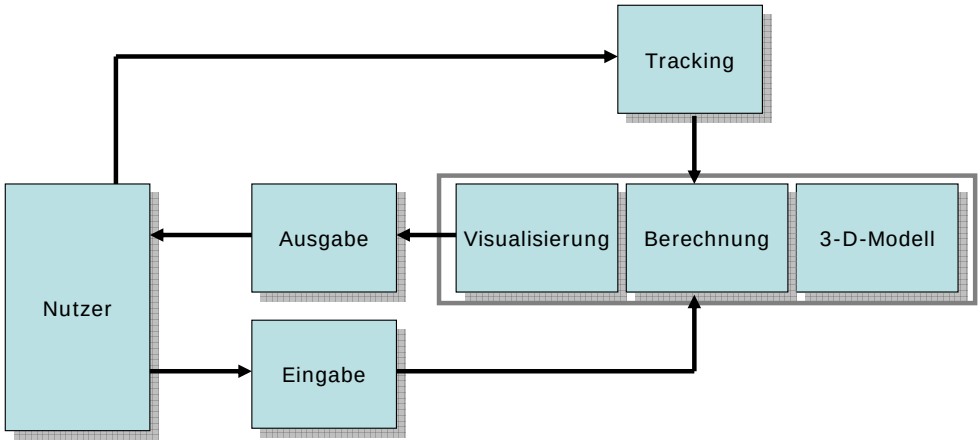


Abbildung 2.3: Schematischer Aufbau eines Virtual Reality Systems

Darstellung im Designprozess neuer Produkte, unter anderem dem Automobil [19], verwendet oder um Crash-tests zu simulieren [44]. Durch die schnelle Veränderbarkeit von virtuellen Modellen lassen sich verschiedene Designvarianten in kurzer Zeit realisieren und darstellen. So wird ein großes Maß an Kosten und Zeit für Produktion von Prototypen eingespart.

2.1.3 Abgrenzung Virtual und Augmented Reality

Betrachtet man also die Definition von VR und AR, ergeben sich deutliche Unterschiede. Während VR den Nutzer vollständig von seiner realen Umwelt trennt und an deren Stelle die virtuelle Umgebung tritt, erweitert die AR die reale Umwelt des Nutzers mit der virtuellen Umgebung. Der Nutzer kann hier im Gegensatz zur VR sowohl mit seinem realen als auch seinem virtuellen Umfeld interagieren. Aufgrund der unterschiedlichen Darstellungen ergeben sich hinsichtlich der Hardware verschiedene Probleme. Die Rechenintensität ist bei der VR höher, als bei der AR, da hier die komplette Umgebung modelliert und visualisiert werden muss. AR dagegen stellt hier nur einzelne Objekte grafisch dar. Diese Darstellung jedoch muss mit der realen Umwelt kombiniert werden. Daraus resultiert eine exakte und positionsgenaue Überlagerung der virtuellen mit der realen Umgebung. Diese Positionierung muss in Echtzeit errechnet und grafisch visualisiert werden, um ein Nachziehen der virtuellen Umgebung in der realen Umwelt zu verhindern.

2.2 Koordinatensysteme und -transformationen

Für die Augmented Reality haben KOS und deren Beziehung zueinander eine sehr große Bedeutung. Um berechnen zu können, welchen Ausschnitt ein Nutzer aktuell durch ein Head-Mounted-Display (HMD) sieht, muss der Zusammenhang zwischen virtuellen Objekten und dem Betrachtungswinkel des Nutzers berechnen werden. Hierfür werden drei KOS notwendig.

Das WeltKOS (X_W, Y_W, Z_W) beschreibt das über allem liegende, globale KOS, in dem alle anderen KOS liegen. Innerhalb dieses KOS befinden sich das ObjektKOS (X_O, Y_O, Z_O) und das KameraKOS (X_K, Y_K, Z_K). Das ObjektKOS dient der Beschreibung und der Abbildung eines oder mehrerer virtueller Objekte. Diese werden in der Regel durch geometrische Angaben, wie etwa Form oder Positionen von Eckpunkten, innerhalb dieses KOS modelliert. Das KameraKOS beschreibt die Position und Orientierung der Kamera, aus deren Sicht die Szene

betrachtet wird. Diese Sicht steht in der Regel stellvertretend für die Blickrichtung des Auges eines Nutzers. Für die Verwendung eines HMDs, das sich der Video-See-Through Technologie bedient, wird die Position der Kamera in diesem KameraKOS angegeben. Diese Position und Orientierung wird durch eine Kalibrierung des HMDs vorgenommen.

Abb. 2.4 verdeutlicht nochmals den Zusammenhang der drei KOS, hier vereinfacht dargestellt für einen zweidimensionalen Fall. Im WeltKOS befindet sich ein Objekt und eine Kamera, die das Objekt betrachtet. Es wird deutlich, dass das Objekt nicht vollständig im Kamerasichtfeld liegt und somit nur teilweise dargestellt wird.

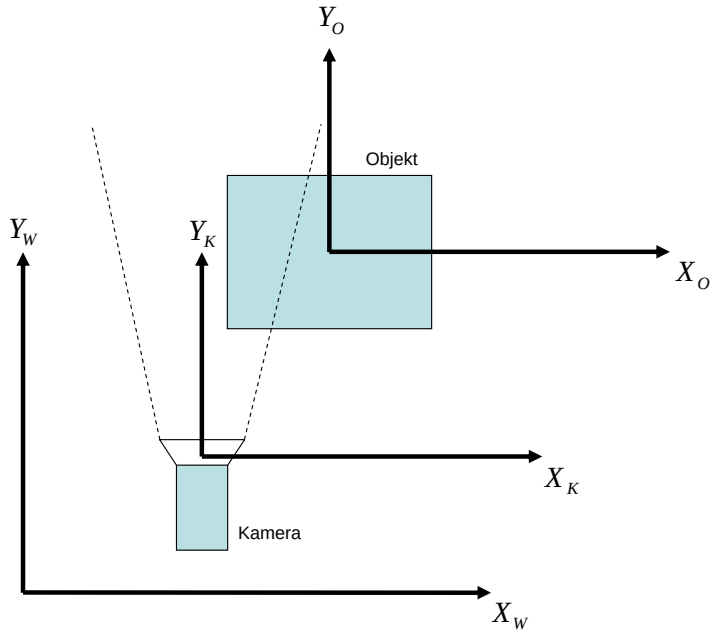


Abbildung 2.4: Zusammenhang der Koordinatensysteme in 2-D

Um nun die Beziehung zwischen diesen KOS zu errechnen, werden Transformationen benötigt. Transformationen beschreiben den Übergang eines AusgangsKOS $(X_1, X_2, X_3, ..., X_N)$ in ein neues KOS $(X'_1, X'_2, X'_3, ..., X'_N)$. Dabei sind drei Basistransformationen gebräuchlich. Diese sind die Translation, die eine Verschiebung eines KOS bezeichnet (vgl. Abb. 2.5), die Rotation, mit der die Drehung eines KOS beschrieben wird (vgl. Abb. 2.6), und die Skalierung, mit der Änderungen der Größe abgebildet werden (vgl. Abb. 2.7). Liegen diesen Basistrans-

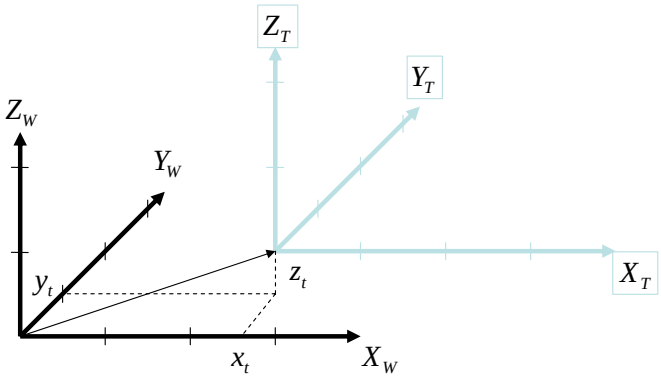


Abbildung 2.5: Translation eines Koordinatensystems

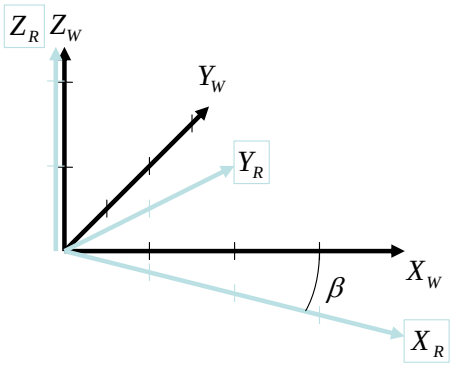


Abbildung 2.6: Rotation (um die Z-Achse) eines Koordinatensystems

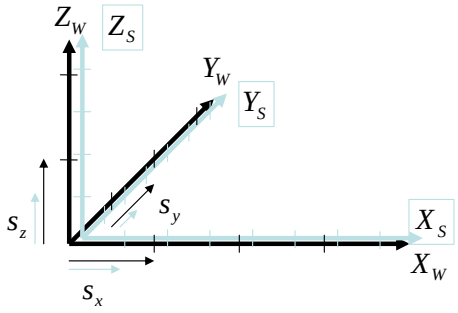


Abbildung 2.7: Skalierung eines Koordinatensystems

formationen kartesische Koordinaten zugrunde, ergibt sich für die Transformationsgleichung der Translation die neue Position $\underline{p}'$ zu

$$\underline{p}' = \begin{pmatrix} p'_x \\ p'_y \\ p'_z \end{pmatrix} = \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix} + \begin{pmatrix} p_x \\ p_y \\ p_z \end{pmatrix}$$

als Vektoraddition aus einem Verschiebungsvektor $\underline{t}$ und dem Vektor der ursprünglichen Position $\underline{p}$. $\underline{t}$ besteht dabei aus den Komponenten t_x, t_y und t_z, die jeweils die Verschiebung entlang der drei Basisachsen bezeichnen.

Die Berechnung der Transformationsgleichung für eine Rotation erfolgt im Allgemeinen basierend auf der Rotation um jeweils eine der drei Basisachsen und ergibt die neue Position als Vektor $\underline{p}'$ für die Drehung η um die X-Achse zu

$$\underline{p}' = \begin{pmatrix} p'_x \\ p'_y \\ p'_z \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\eta & -\sin\eta \\ 0 & \sin\eta & \cos\eta \end{pmatrix} \cdot \begin{pmatrix} p_x \\ p_y \\ p_z \end{pmatrix}$$

Wird dagegen um den Wert θ um die Y-Achse rotiert, ergibt sich die Transformationsgleichung zu

$$\underline{p}' = \begin{pmatrix} p'_x \\ p'_y \\ p'_z \end{pmatrix} = \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix} \cdot \begin{pmatrix} p_x \\ p_y \\ p_z \end{pmatrix}$$

Eine Drehung um die Z-Achse um ϕ ergibt folglich die Transformationsgleichung zu

$$\underline{p}' = \begin{pmatrix} p'_x \\ p'_y \\ p'_z \end{pmatrix} = \begin{pmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} p_x \\ p_y \\ p_z \end{pmatrix}$$

Da die Reihenfolge der Abfolge der einzelnen Drehungen zur Berechnung eine Rolle spielt, ergibt sich eine gesamte Transformationsgleichung für eine Rotation um alle Basisachsen gemäß der Reihenfolge der Ausführung der einzelnen Rotationen um die jeweilige Basisachse. Für eine Rotation beginnend um die X-Achse, gefolgt von einer Drehung um die Y-Achse und einer abschließenden Rotation um die Z-Achse ergibt sich dann die Transformationsgleichung

$$\underline{p}' = \begin{pmatrix} p'_x \\ p'_y \\ p'_z \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\eta & -\sin\eta \\ 0 & \sin\eta & \cos\eta \end{pmatrix} \cdot \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix} \cdot \begin{pmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} p_x \\ p_y \\ p_z \end{pmatrix}$$

$$= \begin{pmatrix} \cos\phi\cos\theta & -\sin\phi\cos\theta & \sin\theta \\ \sin\phi\cos\eta + \cos\phi\sin\theta\sin\eta & \cos\phi\cos\eta - \sin\phi\sin\theta\sin\eta & -\cos\theta\sin\eta \\ \sin\phi\cos\eta - \cos\phi\sin\theta\cos\eta & \cos\phi\sin\eta + \sin\phi\sin\theta\cos\eta & \cos\theta\cos\eta \end{pmatrix} \cdot \begin{pmatrix} p_x \\ p_y \\ p_z \end{pmatrix}$$

Hierbei bezeichnen die Winkel η den Winkel der Drehung um die X-Achse, θ den Winkel der Drehung um die Y-Achse und ϕ den Winkel der Drehung um die Z-Achse. Es ergibt sich also zur Berechnung eine Multiplikation zwischen der resultierenden Rotationsmatrix R und dem Vektor $\underline{p}$, der die Position des KOS beschreibt.

Die Berechnung für die Skalierung ergibt für die neue Position $\underline{p}'$ die Transformationsgleichung

$$\underline{p}' = \begin{pmatrix} p'_x \\ p'_y \\ p'_z \end{pmatrix} = \begin{pmatrix} s_x \cdot p_x \\ s_y \cdot p_y \\ s_z \cdot p_z \end{pmatrix}$$

wobei hier eine Vektormultiplikation zwischen dem Skalierungsvektor $\underline{s}$ mit den Elementen s_x, s_y und s_z und der aktuellen Position $\underline{p}$ vorgenommen wird.

Soll nun also der Zusammenhang zweier KOS berechnet werden, ergibt sich diese aus einer Kombination der drei Basistransformationen als eine Mischung von Vektoradditionen, Matrix-Vektor-Multiplikationen und Vektormultiplikation. Bei komplexeren Zusammenhängen ergeben sich hier jedoch sehr schnell komplizierte Berechnungsvorschriften. Aus diesem Grund werden für die Berechnungen solcher Transformationsketten homogene Koordinaten herangezogen. Diese ermöglichen die Darstellung aller Basistransformationen mithilfe von Matrix-Vektor-Multiplikationen in einer modularen Verkettung. Bei den homogenen Koordinaten werden die kartesischen Koordinaten um die Komponente h erweitert. Diese wird in der Regel mit dem Wert 1 belegt. Daraus ergibt sich $\underline{p}_{homogen}$ zu

$$\underline{p}_{homogen} = \begin{pmatrix} p_x \\ p_y \\ p_z \\ 1 \end{pmatrix}$$

Die daraus resultierende allgemeine Transformationsgleichung in homogenen Koordinaten

$$\underline{p}' = T \cdot \underline{p}$$

beschreibt somit den Zusammenhang zweier KOS. In homogenen Koordinaten ergeben sich die Basistransformationen für die Translation zu

$$\underline{p}' = \begin{pmatrix} p'_x \\ p'_y \\ p'_z \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} p_x \\ p_y \\ p_z \\ 1 \end{pmatrix}$$

wobei hier die Komponenten t_x, t_y und t_z jeweils die Verschiebung entlang der Basisachse beschreibt. Die Rotationsmatrix ergibt sich bei Berücksichtigung der Reihenfolge X-, Y- und Z-Achse zu

$$\underline{p}' = \begin{pmatrix} p'_x \\ p'_y \\ p'_z \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} \cos\phi\cos\theta & -\sin\phi\cos\theta & \sin\theta & 0 \\ \sin\phi\cos\eta + \cos\phi\sin\theta\sin\eta & \cos\phi\cos\eta - \sin\phi\sin\theta\sin\eta & -\cos\theta\sin\eta & 0 \\ \sin\phi\cos\eta - \cos\phi\sin\theta\cos\eta & \cos\phi\sin\eta + \sin\phi\sin\theta\cos\eta & \cos\theta\cos\eta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} p_x \\ p_y \\ p_z \\ 1 \end{pmatrix}$$

wobei hier die Winkel η, θ und ϕ jeweils den Drehwinkel um die Basisachse bezeichnet. Die Skalierung ergibt sich folglich zu

$$\underline{p}' = \begin{pmatrix} p'_x \\ p'_y \\ p'_z \\ 1 \end{pmatrix} = \begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} p_x \\ p_y \\ p_z \\ 1 \end{pmatrix}$$

mit den Skalierungswerten s_x, s_y und s_z für jede Basisachse.

Mit homogenen Koordinaten kann also eine komplexe Transformationskette in kartesischen Koordinaten in eine einfachere Multiplikation von Matrizen überführt werden. Dadurch werden nicht nur Rundungsfehler minimiert, die durch mehrfache verschiedene Rechenoperationen auftreten können, sondern auch effizientere Algorithmen verwendbar, die auf die Berechnung von Matrixmultiplikationen hin optimiert sind. Aufgrund der Assoziativität von Matrizenmultiplikationen können aufeinanderfolgende Matrizen zu einer einzigen Matrix zusammengefasst werden. Dadurch lassen sich komplexe Berechnungen in kürzerer Zeit ausführen. Neben diesem Vorteil ist es auch möglich, verschiedene Schritte innerhalb einer Transformationskette auszutauschen oder Schritte einzufügen. Während bei kartesischen Koordinaten hierfür meist die gesamte Transformationskette neu berechnet werden muss, genügt bei der Verwendung von homogenen Koordinaten das Austauschen, Hinzufügen oder Entfernen einer Matrix.

Homogene Koordinaten werden jedoch nur innerhalb der Software verwendet. Die Modellierung von Objekten oder die Positionierung von KOS oder Objekten erfolgt in kartesischen Koordinaten. Diese werden dann von der Software zur Berechnung der Visualisierung automatisch in eine homogene Darstellung transformiert.

2.3 Überblick über Hardware zur Darstellung virtueller Szenarien

Das folgende Kapitel beschreibt verschiedene Komponenten, die zur Darstellung von virtuellen Umgebungen notwendig sind. Dabei bezieht sich das Kapitel auf die zwei wichtigsten Teile, das Tracking und die Techniken zur Visualisierung.

Tracking ist notwendig, um die Position von Nutzer oder realen Objekten zu ermitteln. In der Regel existiert ein übergeordnetes WeltKOS, in dem die virtuelle Umgebung angeordnet ist. Innerhalb von diesem werden virtuelle Objekte platziert. Sollen diese nun an die reale Umgebung angepasst werden, müssen die Positionen von realen Objekten erfasst werden, um diese in Wechselwirkung mit den virtuellen Objekten treten zu lassen. Weiterhin muss die Position und Blickrichtung des Nutzers im WeltKOS bestimmt werden, damit der in diesem Sichtfeld darzustellende Bereich ermittelt werden kann. Sind nun in diesem Bereich virtuelle Objekte platziert, werden diese entsprechend ihrer Ausrichtung dargestellt. Hierbei werden verschiedene Verfahren zur Darstellung unterschieden. Die Visualisierung kann entweder mittels eines HMDs erfolgen oder über eine Anzeigefläche, wie etwa einen Monitor.

2.3.1 Trackingsysteme

Um Positionen von realen Objekten oder dem Nutzer messen zu können, stehen zwei grundsätzliche Methoden zur Verfügung. Diese unterscheiden sich in der Verwendung von Markern. Markerbasierte Verfahren benötigen zusätzliche Hardware, die es dem Trackingsystem erlaubt, Daten zu erfassen. Diese Marker sind in der Regel physikalische Objekte mit bestimmten Eigenschaften, nach denen das Erkennungssystem des Trackingsystems gezielt suchen kann. Diese Eigenschaften können beispielsweise optische, akustische oder magnetische Eigenschaften sein.

Zu den markerbasierten, optischen Systemen gehört beispielsweise „ARToolkit“ [39], das ein frei verfügbares Framework zur Markererkennung bereitstellt. Diese Marker bestehen aus auf Papier gedruckten, schwarzen Quadraten mit weissen inneren Quadraten. Durch Bildverarbeitung werden die Konturen dieser Quadrate extrahiert und durch ihre projezierte Lage die Anorndung in 3-D relativ zur Kamera gemessen. Diese stellen dann die Grundlage eines KOS, an dem eine virtuelle Umgebung ausgerichtet werden kann. Weiterentwickelt für die Verwendung auf mobilen Geräten wurde „ARToolkit“ zu „ARToolkitPlus“ [89].

Ebenfalls markerbasiertes, optisches Tracking stellt Infrarot (IR)-Tracking dar, das u.a. in [29] verwendet wird. Hier werden Kugeln, die speziell im IR Bereich reflektieren, oder IR Light Emitting Diodes (LED) verwendet. Kameras senden IR Lichtblitze aus, die an den Kugeln reflektiert werden. Diese Reflexionen werden von den Kameras aufgenommen und mittels Triangulation in 3-D-Koordinaten umgerechnet. An geometrischen Anordnungen solcher Kugeln lässt sich ein KOS definieren, welches einer AR Anwendung als Grundlage dienen kann.

Neben den optischen, markerbasierten Methoden, kommen auch Trackingsysteme basierend auf magnetischen Eigenschaften zum Einsatz, wie beispielsweise schon Ende der 70er Jahre [71]. Dabei wird von einem Sender ein magnetisches Feld erzeugt. Innerhalb von diesem bewegen sich Sensoren, die durch die Bewegung im Magnetfeld Ströme messen können. Durch die Messung dieser Ströme kann auf die Bewegung im Raum zurückgeschlossen werden. Zwei der am meisten verbreiteten Systeme sind das „Flock of Birds“ von „Ascension“ [17] und „Fastrack“ von „Polhemus“ [69].

Akustische Trackingsysteme, wie beispielsweise das von „Intersense“ [34] angebotene „IS-900 inertial-ultrasonic motion tracking system“, nutzen den Effekt von unterschiedlichen Laufzeiten von akustischen Signalen. Dazu werden von mehreren, fest im Raum positionierten Sendern abwechselnd Signale im Ultraschall-Bereich ausgesandt, die dann von Empfängern aufgenommen werden. Aus jedem dieser Signale kann die Laufzeit zum entsprechenden Sender berechnet werden. Aus diesen Informationen kann dann auf die Position im Raum, relativ zu den Sendern, geschlossen werden.

Markerlose Trackingsysteme benötigen keine Marker mit definierten Eigenschaften. Markerlose Systeme zeichnen sich dadurch aus, dass sie ohne zusätzliche Hardware am Objekt dessen Position und Orientierung ermitteln können. Dies wird ermöglicht durch die Erkennung der Geometrie des zu messenden Objektes, das zuvor als Modell zur Verfügung steht. Mittels Bildverarbeitung kann aus einem 2-D-Bild, die 3-D-Information bzgl. des Objektes errechnet werden [70]. Eine weitere Möglichkeit besteht im Vergleich des aktuellen Kamerabildes mit zuvor aufgenommenen Referenzbildern [80]. Ebenfalls werden andere Ansätze verfolgt, wie etwa die Hand des Nutzers als Marker zu verwenden und dessen Lage relativ zur Kamera zu ermitteln [47].

2.3.2 Visualisierungstechniken

Um virtuelle Umgebungen darstellen zu könnnen, wird ein System zur grafischen Ausgabe benötigt. Für die Verwendung in AR-Szenarien kommen hier nutzergebundene Systeme oder nutzerungebundene Systeme zum Einsatz. Nutzergebundene Systeme befinden sich direkt am Nutzer und sind mit diesem durch entsprechende Tragevorrichtungen fest verbunden. Nutzerungebundene Systeme dagegen sind nicht fest mit dem Nutzer verbunden, sondern stehen entweder frei im Raum oder können vom Nutzer in die Hand genommen werden.

Zu den nutzergebundenen Systemen gehören HMDs, die im verwendeten Visualisierungsverfahren verschiedene Möglichkeiten anbieten [10]. Unterschieden wird hierbei die Darstellung mittels Video-See-Through (VST) (vgl. Abb. 2.8) und Optical-See-Through (OST) (vgl. Abb. 2.9).

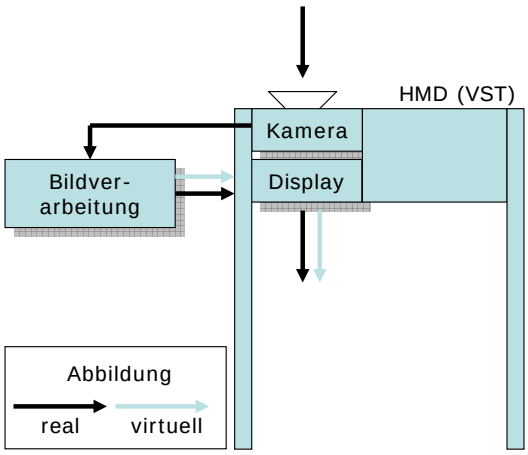


Abbildung 2.8: Schematischer Aufbau eines Head-Mounted-Displays mit Video-See-Through

Bei der VST befindet sich vor dem Auge des Nutzers auf der Innenseite der Brille ein Display, über das grafische Ausgaben erfolgen (vgl. Abb. 2.10 (a)). Dabei wird der Nutzer

komplett von seiner realen Umgebung abgeschnitten. Über eine Kamera, die auf der Vorderseite der Brille angebracht ist, kann jedoch das Bild im Bereich des Blickwinkels vor der Brille erfasst werden und dem Nutzer über das Display angezeigt werden. Auf diese Weise kann er seine Umgebung über das Kamera-Display-System aufnehmen. Diese Methode bietet die Möglichkeit, das aufgenommene Videobild mit zusätzlichen, virtuellen Informationen anzureichern und somit reale und virtuelle Umgebung zu überlagern. Durch Verwendung von einer Kamera und eines Displays getrennt für jedes Auge ist eine Darstellung in stereo möglich. Allerdings gestaltet sich diese Darstellung gerade im Nahbereich des Nutzers schwierig [1], da die Kameras nicht wie die Augen auf einen Punkt fixiert werden können, sondern starr in eine Richtung blicken.

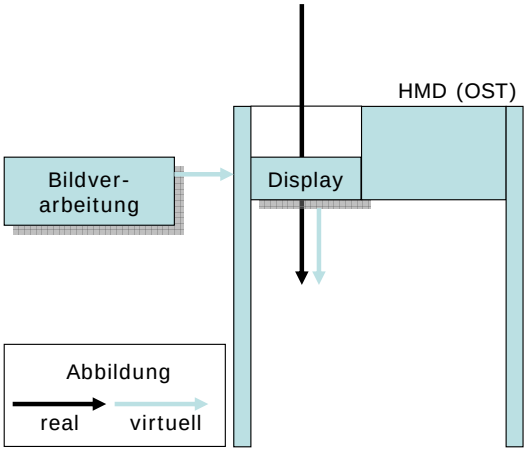
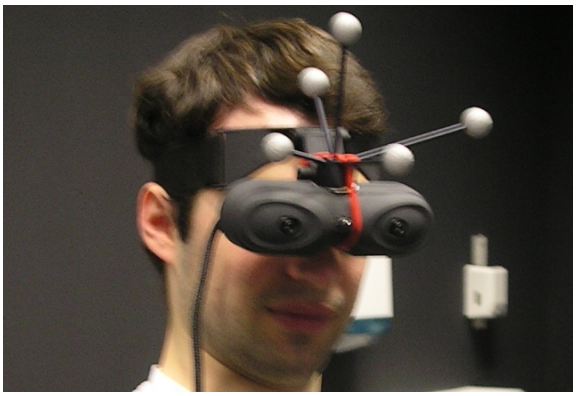


Abbildung 2.9: Schematischer Aufbau eines Head-Mounted-Displays mit Optical-See-Through

Bei der OST wird die reale Umgebung des Nutzers nicht abgeschirmt. Statt einem Kamera-Display-System wird hier ein optisches Element verwendet, das es ermöglicht, die reale Umgebung mit virtuellen Informationen zu überlagern. Dies erfolgt in der Regel durch einen halbtransparenten Spiegel, auf den ein vom System erzeugter, virtueller Inhalt in das Auge des Nutzers gespiegelt werden kann (vgl. Abb. 2.10 (b)).



(a) Darstellung mittels Video-See-Through

(b) Darstellung mittels Optical-See-Through

Abbildung 2.10: Nutzer mit Head-Mounted-Displays

Beide vorgestellten Systeme bieten ihre Vor- und Nachteile. Das VST verursacht durch die Berechnung der gesamten Szene eine Verzögerung zwischen Bewegungen des Nutzers und der

Darstellung der Bewegung im Display. Das heisst, wenn sich der Nutzer in einer Umgebung bewegt, hängt die Darstellung der Umgebung seiner eigenen Bewegung nach. Besonders tritt dieser Effekt bei schnellen Kopfdrehungen auf. Weiterhin ist auch die Rate der dargestellten Frames begrenzt und erzeugt ein ruckelndes Bild der Umgebung. Durch die komplette Berechnung der kombinierten realen und virtuellen Szene werden jedoch reale und virtuelle Umgebung immer in korrektem Zusammenhang dargestellt, d.h. die Registrierung erfolgt hier immer passend. Bei der Verwendung des OST dagegen ist immer ein Versatz zwischen realen und virtuellen Objekten erkennbar, d.h. die Registrierung erfolgt nicht exakt [4]. Dies resultiert aus der getrennten Berechnung der virtuellen Umgebung und der „Darstellung“ der realen Umgebung. Diese wird ohne Verzögerung abgebildet, da hier keine Berechnung erfolgen muss. Durch diese Trennung kann die Darstellung mittels OST mit weniger Rechenlast erfolgen, da hier im Gegensatz zur VST nicht das komplette Videobild mit verarbeitet werden muss. Durch das Aufnehmen der Umgebung beim VST leidet auch die Darstellungsqualität der realen Umgebung, da diese durch die verwendete Hardware, wie etwa Kameraqualität und -auflösung, limitiert ist. Ein weiterer großer Unterschied liegt in der verdeckenden Darstellung virtueller Objekte und derer Farbeingrenzung. Bei der VST können virtuelle Objekte vollkommen deckend vor das Videobild gelegt werden. Virtuelle Objekte können also reale Objekte völlig verdecken, so dass diese nicht mehr sichtbar sind. Die OST dagegen kann dies, bis auf wenige Ausnahmen [42], nicht ermöglichen, da durch eine einfache Überlagerung zweier optischer Strahlenwege keiner vom anderen vollkommen verdeckt werden kann. Vor allem helle, reale Objekte können somit von virtuellen Objekten nicht abgedeckt werden und scheinen immer durch die virtuellen Objekte hindurch. Somit ist es auch nicht möglich, virtuelle Objekte in schwarz darzustellen.

Nutzerungebundene Systeme vereinen Anzeigegeräte, wie beispielsweise einen herkömmlichen Monitor, mobile Endgeräte [65], [88] oder ein Head-Up-Display (vgl. Abb. 2.12). Allen ist hier gemein, dass sie nicht die komplette reale Umgebung des Nutzers erweitern, wie es etwa nutzergebundene Systeme ermöglichen. Vielmehr agieren diese Systeme als Darstellung eines Ausschnittes der kombinierten Umgebung (vgl. Abb. 2.11).

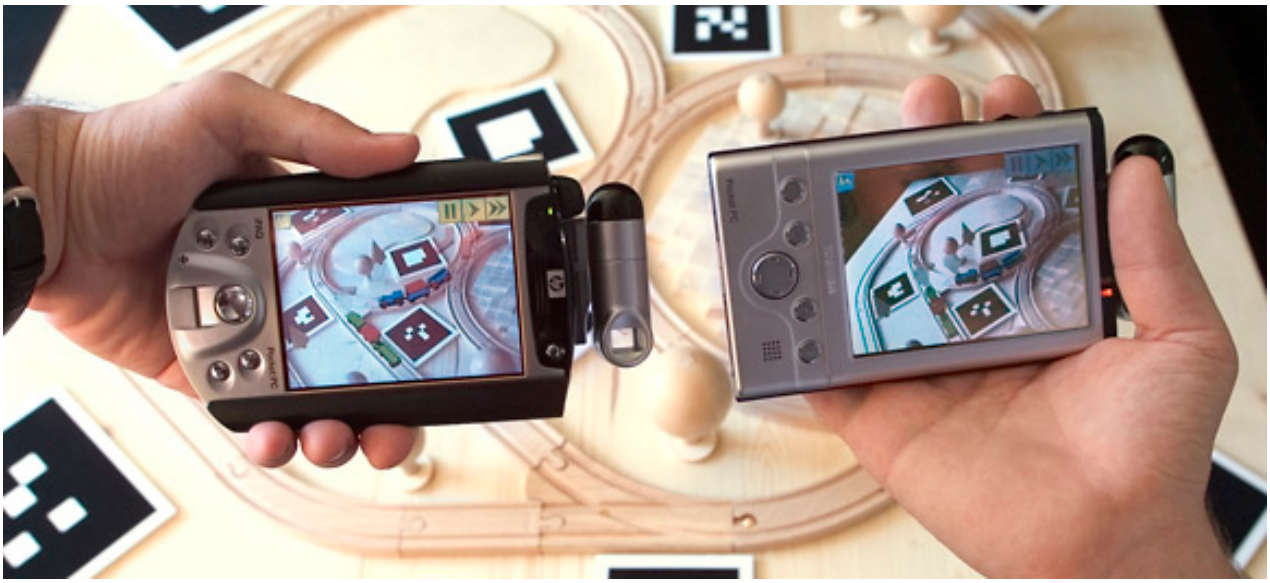


Abbildung 2.11: Darstellung mittels mobiler Endgeräte [88]

Nutzergebundene Systeme erfordern eine Kalibrierung, um den Zusammenhang zwischen

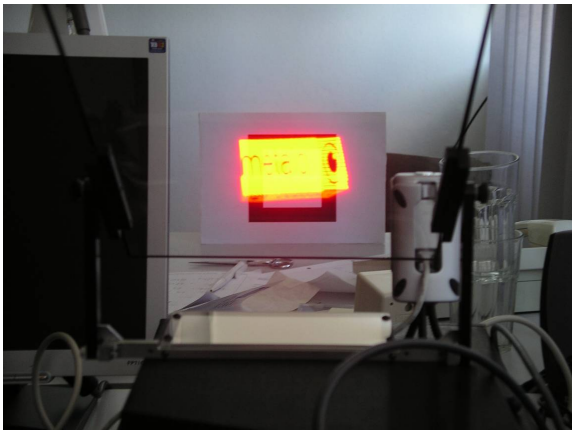


Abbildung 2.12: Darstellung eines roten Quaders mittels eines Head-Up-Displays

dem Blickwinkel des Nutzers und dem Sichtbereich des HMDs herzustellen. Da diese Systeme direkt mit dem Nutzer verbunden sind, muss also eine Bestimmung der Transformation zwischen dem angenommenen KOS des Auges des Nutzers und dem KOS des Displays vorgenommen werden. Dabei muss die Translation und die Rotation zwischen beiden KOS berücksichtigt werden. Für diese Aufgabe existieren bereits Methoden, wie etwa in [39], [85] oder [27] beschrieben.

2.4 Multimodale Interaktion

Der Mensch interagiert mit seiner Umwelt nicht nur in einer monomodalen Art und Weise. Gesprächspartner verwenden meist unbewusst verschiedene redundante Übertragungskanäle, um Informationen auszutauschen. Dabei wertet jeder Gesprächspartner alle angebotenen Kanäle aus und fügt deren Informationen zu einer semantischen Bedeutung zusammen. So reagieren Menschen auf eine „Ja-oder-Nein“-Frage meist nicht nur mit einer sprachlichen Antwort, sondern unterstützen diese auch noch mittels einer Kopfgeste (Nicken oder Kopfschütteln). Auf diese Weise können Umwelteinflüsse, die die Übertragungsqualität eines einzelnen Kanals beeinträchtigen, wie etwa schlechte Lichtverhältnisse oder Umgebungslärm, kompensiert werden.

Die Verarbeitung von Sinneswahrnehmungen ist untergliedert in die Sensorik, Kognition und Motorik (vgl. Abb. 2.13). Die Sensorik des Menschen fasst sämtliche Kanäle zusammen, mit denen der Mensch Informationen seiner Umgebung aufnehmen kann, wie zum Beispiel Sehen oder Hören. Die Kognition verarbeitet und komprimiert diese Informationen. Die Motorik fasst alle Kanäle des Menschen zusammen, mit denen der Mensch Informationen an seine Umwelt abgibt, also zum beispielsweise Äußerungen in Form von Sprache oder Gestik.

Multimodale Interaktion in der Mensch-Maschine-Kommunikation (MMK) ist definiert durch die gleichzeitige Verwendbarkeit von verschiedenen Geräten zur Eingabe in ein System, beispielsweise mittels Gesten oder Sprache, in einer sinnvollen Art und Weise, kombiniert mit einer multimodalen Ausgabe des Systems [64]. Erweitert wird diese Definition von [8] auf ein System, das Informationen über verschiedene Übertragungskanäle in mehreren Ebenen der Abstraktion repräsentiert und verändert. Ein multimodales System ist demnach hauptsächlich dadurch charakterisiert, dass der Nutzer über verschiedene Ein- und Ausgabekanäle mit einem System interagieren kann. Um aus den Eingaben die Intention des Nutzers zu erken-

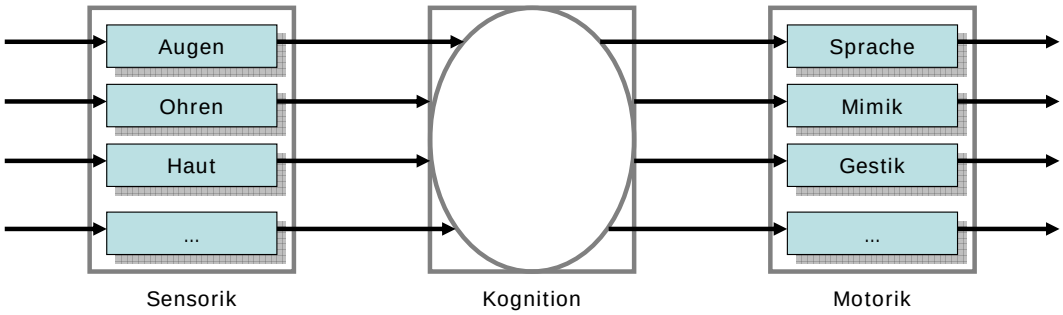


Abbildung 2.13: Informationsverarbeitung des Menschen

nen, verwendet ein solches System Informationen, die von verschiedenen Erkennungssystemen geliefert werden. Aus diesen werden Inhalt und Bedeutung der Eingabe des Nutzers extrahiert.

Im Gegensatz zu monomodaler Interaktion erhöht die multimodale Interaktion die Effizienz, die Robustheit gegenüber Fehlern und das Gefühl einer natürlichen Interaktion, ähnlich wie die zwischenmenschliche Kommunikation. Die Verwendung verschiedener, auch redundanter Eingabemöglichkeiten eröffnet die Kompensation der auszutauschenden Daten. Übertragungskanäle, die aufgrund von Umwelteinflüssen in einer schlechten Übermittlung der Information resultieren, können durch die Verwendung von anderen Kanälen ausgeglichen werden. So kann ein System mit rein bildbasierter Gestenerkennung in Dunkelheit nur schlechte Ergebnisse liefern, die Erkennungsrate eines Spracherkenners dagegen leidet nur bei Störgeräuschen. Eine Kombination gleicht jeweils die schlechten Sichtverhältnisse oder die Umgebungsgeräusche aus und erhöht damit die Rate der Intentionserkennung. Multimodale Interaktion eröffnet außerdem jedem Nutzer die Wahlmöglichkeit der Eingabeart. So kann der Nutzer seine Eingabeart individuell an seine Bedürfnisse anpassen. Der Nutzer kann damit das System mit der Eingabeart bedienen, die er persönlich für die notwendige Eingabe am besten geeignet empfindet. Auch dadurch lässt sich die Erkennungsrate des Systems erhöhen.

Um aus verschiedenen Eingabekanälen eine gesamtheitliche Betrachtung der Informationen zu ermöglichen, müssen die Daten der Eingabekanäle zusammengefasst werden. Dieser Vorgang wird als Fusion bezeichnet. Zur Datenfusion existieren verschiedene Ansätze, die grundlegend in regelbasierte und stochastische Ansätze untergliedert werden. Bei regelbasierten Ansätzen wird die Fusion mittels zuvor definierter Regeln durchgeführt. Dabei werden bestimmte Ereignisse, beispielsweise das Erkennen eines Wortes oder einer Geste, oder Kombinationen von Ereignissen mit festgelegten Aktionen verknüpft. Beispielsweise verknüpft eine Regel das Ereignis „Abspielen“ mit der Startfunktion des CD-Spielers. Bei stochastischen Ansätzen wird der Zeitpunkt der Fusion der Daten zur Unterscheidung herangezogen. Frühe Fusion auf Signalebene wird beispielsweise mittels Hidden-Markov-Modellen (HMMs) modelliert, späte Fusion auf semantischer Ebene mit Neuronalen Netzen oder dynamischen Bayes'schen Netzen.

2.5 Umgesetzte Softwarestruktur

Dieses Kapitel widmet sich der zugrundeliegenden Softwarestruktur. Dabei erläutert es weder Details einzelner Komponenten noch gibt es einen detailierten Blick auf deren softwaretechnische Umsetzung. Vielmehr dient dieses Kapitel dazu, einen globalen Überblick über das gesamte System und die Wechselwirkung zwischen einzelnen Komponenten zu vermitteln.

2.5.1 Ausgangsbasis

Als Grundlage zur Umsetzung der Software in dieser Arbeit standen drei kommerziell verfügbare Produkte als Ausgangsbasis zur Verfügung. Dabei handelt es sich um ein Software-Framework, das grundlegende Funktionalität zur Erstellung von virtuellen Umgebungen bereithält, sowie ein Trackingssystem, das auf IR basiert. Zur Visualisierung kam ein HMD mit stereoskopischer Video-See-Through-Technologie zum Einsatz. Ausgehend von diesen drei Komponenten und den an die Software gestellten Anforderungen, wurde ein Gesamtkonzept für eine Softwarelösung entwickelt.

„Unifeye SDK“ von „Metaio“ [56]
Mit „Unifeye“ existiert ein sehr umfangreiches Framework, mit dem sich viele Basisfunktionen zur Erzeugung von AR umsetzen lassen. Zu diesen Basisfunktionen zählen die Anbindung an das ebenfalls bestehende IR-Trackingsystem, die Berechnung der Blickposition des Nutzers anhand von zuvor gemessenen Abweichungen zwischen Kamera und Auge sowie Funktionen, die die direkte Erstellung und Veränderung von virtuellem Inhalt ermöglichen (vgl. Abb. 2.14).

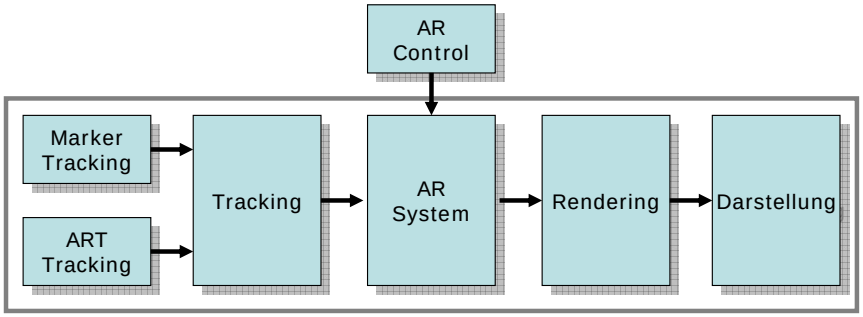


Abbildung 2.14: Schematischer Aufbau des „Unifeye SDK“

„Unifeye“ lässt sich als „ActiveX“ Komponente in eigene Applikationen einbinden. Über eine Schnittstelle lassen sich alle relevanten Befehle und Informationen der aktuellen Augmented Reality Umgebung aufrufen und verändern. Zu diesem Funktionsumfang gehört die Anbindung an ein Trackingsystem (sowohl IR-Tracking der Firma ART (vgl. Kapitel 2.5.1) als auch makerbasiertes Tracking), die Visualisierung der AR-Umgebung abhängig vom gewählten Trackingsystem sowie die Möglichkeit, die AR-Umgebung zu verändern. Die Eingriffsmöglichkeiten auf die AR Umgebung erstrecken sich vom Erstellen und Zerstören von virtuellen Objekten über deren Manipulation hinsichtlich Position und Orientierung im Raum bis hin zu szenenspezifischen Parametern.

Die Darstellung der virtuellen Umgebung erfolgt durch die Angabe von einem KOS einer Kamera relativ zu dem KOS der Welt. Hierbei wird der Sichtbereich durch die positive Z-Achse definiert. Die Richtung dieser Achse innerhalb der Weltkoordinaten entspricht also der Blickrichtung der Kamera, die diese virtuelle Szene betrachtet. Entsprechend dieses Blickwinkels wird die virtuelle Umgebung visualisiert.

Aufgrund seiner Umsetzung ist die Komponente „Unifeye“ jedoch nur in der Lage, lokal von einer Applikation angesprochen zu werden. Diese Beschränkung auf den lokalen Betrieb, bei dem die Darstellung nur auf dem Computer, auf dem die Applikation läuft, erfolgt, limitiert die Möglichkeit der Darstellung. Eine Synchronisation von mehreren, verteilten Computern ist daher nicht möglich. Applikationen im Mehrnutzerbetrieb oder auch der Betrieb eines stereoskopischen HMDs entfallen.

„DTrack“ von „Advanced Realtime Technologies“ (A.R.T.) [2]
Zur Lokalisation von Nutzern oder Objekten innerhalb eines WeltKOS wird ein Trackingsystem benötigt. Das in dieser Arbeit verwendete IR-Trackingsystem „DTrack“ von A.R.T. besteht im Wesentlichen aus den drei Kernkomponenten Target, Kamera (vgl. Abb. 2.16) und Steuerung.

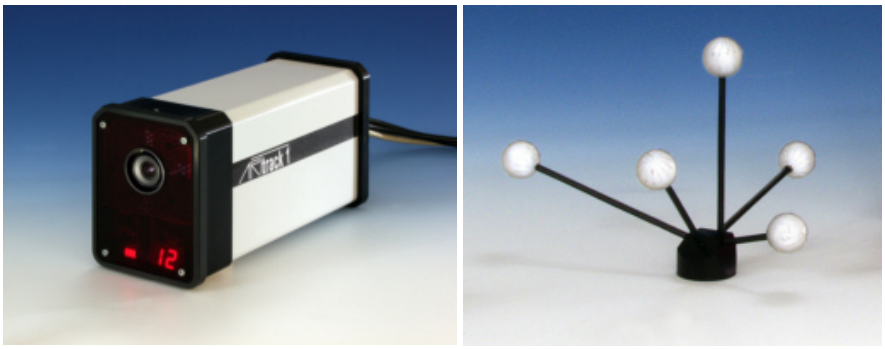


Abbildung 2.15: Komponenten des Trackingsystems: Kamera und Target

Sechs IR-Kameras (vgl. Abb. 2.15) mit integrierten IR LEDs senden synchronisiert von der Steuerungssoftware IR-Blitze mit einer Frequenz von 60 Hz aus. Kugeln, sogenannte Marker, mit besonders stark reflektierender Oberfläche im IR-Bereich, angeordnet in einem Target (vgl. Abb. 2.15), reflektieren diesen Blitz. Die sechs Kameras nehmen jeweils aus deren Blickwinkel die Reflexionen auf und berechnen innerhalb der Bilder jeweils die Position aller Reflexionen. Diese 2-D-Informationen leiten sie an die Steuerung zurück. Die Steuerung erhält also nach jedem IR-Blitz die 2-D-Informationen über die Reflexionspunkte im Raum.

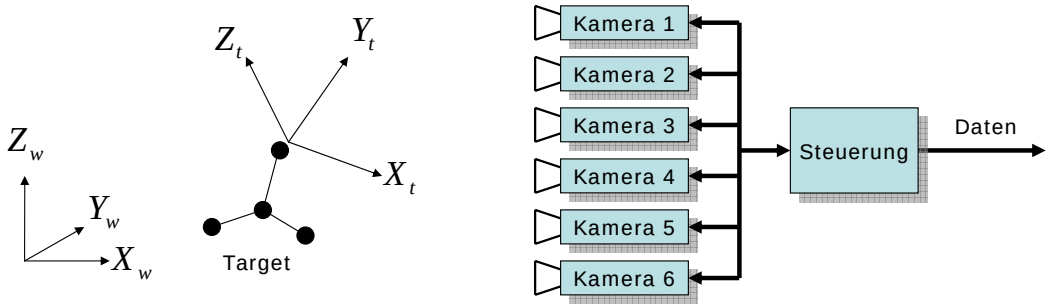


Abbildung 2.16: Schematischer Aufbau des Trackingsystems

Basierend auf einer initialen Kalibrierung wird ein WeltKOS (x_w, y_w, z_w) festgelegt (vgl. Abb. 2.16), welches den Bezugspunkt aller Messungen darstellt. Durch eine weitere Kalibrierung eines Targets wird ein TargetKOS an diesem Target ausgerichtet. Dabei unterscheidet das Trackingsystem Targets anhand ihrer spezifischen Geometrie. Die Informationen bzgl. dieses relativen TargetKOS (x_t, y_t, z_t) werden vom Trackingsystem aufgenommen. Diese Information beinhaltet die Transformation von Welt- und TargetKOS, die sich aus der Rotation und Translation relativ zum WeltKOS zusammensetzt.

Diese Information wird von der Steuerung in ein Daten-Paket mit definiertem Format konvertiert (vgl. Abb. 2.17) und mittels einer Netzwerkverbindung an angeschlossene Computer

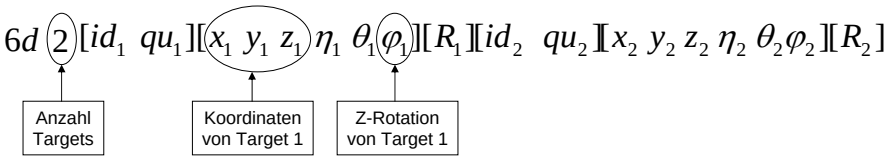


Abbildung 2.17: Versendetes Datenpaket des Trackingsystems

zur weiteren Verarbeitung gesendet. Das Daten-Paket enthält die Anzahl der erkannten Targets, sowie die Translation und Rotation jedes der Targets im Raum. Dabei wird die Rotation sowohl als Drehwinkel um jede der drei Raumachsen als auch als Rotationsmatrix übermittelt. Die Berechnung der aktuellen Transformation erfolgt dabei mit einer Wiederholrate von 60Hz.

Head-Mounted-Display von „Trivisio“ [84]
Um AR in die Umgebung des Nutzers einblenden zu können, stand ein HMD der Firma „Trivisio“ zur Verfügung. Dieses besteht im Wesentlichen aus zwei getrennten Kamera-Display-Systemen. Pro Auge existiert eine Kamera, die mittels einer Verbindung über den „Universal Serial Bus“ (USB) das aktuelle Kamerabild an einen Computer sendet. Auf diesem erfolgt dann die Berechnung des Bildes, das auf dem zugehörigen Display über einen normalen Grafikausgang auf der Innenseite des HMD dargestellt werden soll. Auf diese Weise lässt sich das Sichtfeld des Nutzers für jedes Auge getrennt mit Informationen anreichern und somit ein stereoskopischer Eindruck erweckt.

2.5.2 Problemstellung

An die Softwarestruktur stellen sich diverse Anforderungen. Ausgehend von der Themenstellung soll die Software in der Lage sein, dem Nutzer multimodale Interaktion zu ermöglichen. Das bedeutet, dass eine AR-Applikation sowohl multimodale Eingaben verarbeiten können als auch multimodale Ausgaben produzieren muss. Die Anbindung verschiedener Erkennersysteme muss also aus Performance- und Kompatibilitätsgründen auf mehrere Computer verteilt werden können. Auf diese Weise lässt sich die gesamte Systemlast gezielt auf verschiedene Systeme verteilen. Hierbei sollen verschiedene Erkennersysteme unabhängig ihrer Plattform eingebunden werden können. Aus denselben Gesichtspunkten ergibt sich die Notwendigkeit, die multimodale Ausgabe ebenfalls auf verschiedene Computer aufzuteilen. Hierbei ist ein besonderer Augenmerk auf die Synchronisation der visuellen Ausgabe zu legen, da das verwendete Modul „Unifeye SDK“ (vgl. Kapitel 2.5.1) nur auf den lokalen Betrieb ausgelegt ist. Die Auslegung der Software auf ein verteiltes Gesamtkonzept erfordert dabei also zwingend die Möglichkeit, die aktuelle AR-Umgebung nicht nur lokal, sondern auch verteilt darstellen zu können. Die Visualisierung muss hierbei den engen Anforderungen hinsichtlich Latenz und Synchronisation genügen, sodass gewährleistet ist, dass zu jedem Zeitpunkt auf jedem Computer dieselbe Szene dargestellt wird.

2.5.3 Umsetzung

Die umgesetzte Softwarestruktur basiert auf dem Ansatz, möglichst viele einzelne Komponenten auf spezielle, für diese Komponente optimierte Plattformen zu verteilen. Der gewählte Ansatz der Softwarestruktur ist also ein verteilter Ansatz.

Dabei werden die einzelnen Komponenten auf jeweils eigene Rechner verteilt. Weiterhin ist die Softwarestruktur in drei Schichten untergliedert (vgl. Abb. 2.18).

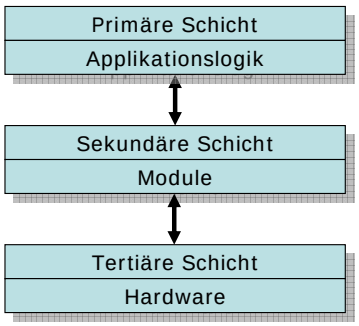


Abbildung 2.18: Schichtenmodell der Softwarestruktur

Die primäre Schicht besteht aus der Applikationslogik, in der alle Informationen und Daten aller angeschlossenen Komponenten aus der sekundären Schicht gesammelt und verarbeitet werden. Die Module der sekundären Schicht erhalten oder senden Daten an die Hardware der tertiären Schicht. Die Umsetzung der sekundären Schicht erfolgt in der Bereitstellung einzelner in sich abgeschlossener Module. Diese Module stellen der übergeordneten primären Schicht Schnittstellen zur Verfügung, mittels derer Daten mit der Hardware der tertiären Schicht ausgetauscht werden können. Die Module aus der sekundären Schicht sind also so aufgebaut, dass sie von jeder beliebigen Applikation verwendet werden können. Des Weiteren muss sich der Entwickler einer Applikation nicht mit der Umsetzung auf Hardwareebene befassen, sondern kann direkt mit den von den bereitgestellten Modulen übermittelten Daten arbeiten. In der tertiären Schicht findet sowohl die direkte Kommunikation zwischen Hardware und Software statt als auch die (Vor-)verarbeitung von Daten. So werden beispielsweise Daten eines Sensors aufgenommen und interpretiert. Die Interpretation wird dann an die sekundäre Schicht übertragen, die sie der primären Schicht zur Verfügung stellt.

Die konkrete Umsetzung dieser Architektur erfolgt auf die Zielplattform Windows mittels der Programmiersprache C#. Die Module der sekundären Schicht wurden als Klassen umgesetzt, die in sich die gesamte Logik zur Kommunikation mit der angeschlossenen Hardware beinhaltet. Aufgrund der Verteilung auf mehrere Rechner besteht ein Modul aus zwei Teilen, die mittels des „User Datagram Protocols“ (UDP) und des „Transmission Control Protocols“ (TCP) miteinander kommunizieren und Daten austauschen (vgl. Abb. 2.19). Im Folgenden werden der Teil, der mit der Applikationslogik kommuniziert, „Client“ und der Teil, der mit der Hardware Daten austauscht, „Server“ genannt.

Die Module sind in einzelne Komponenten untergliedert. Einzelne Komponenten sind beispielsweise die Visualisierung der AR-Umgebung synchron auf verteilten Darstellungsrechnern mittels AR oder VR, die akustische Ausgabe von virtuellen Schallquellen sowie Komponenten, die die Anbindung verschiedener Eingabesysteme, wie etwa einen Sprach- oder Gestenerkenner, ermöglichen. Aufgrund der Umsetzung der Komponenten als einzelne Klassen können die einzelnen Komponenten auch über die primäre Schicht an andere Komponenten der sekundären Schicht übergeben werden. Insofern muss die primäre Schicht nicht alle Daten der sekundären Schicht verwalten, sondern kann gezielt die Informationen einer Komponente an eine andere Komponente übergeben.

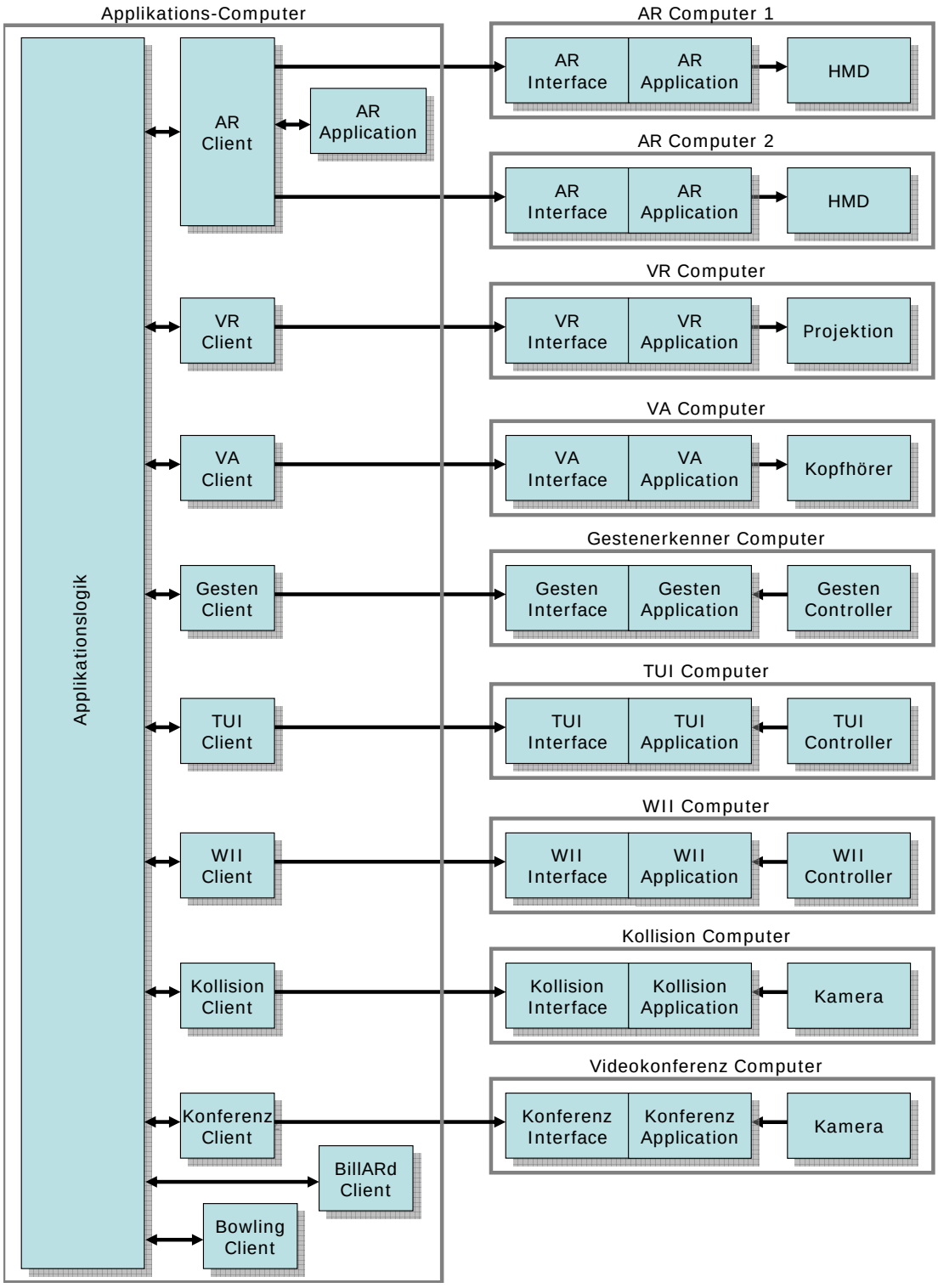


Abbildung 2.19: Schematischer Aufbau der Softwarestruktur

KAPITEL 3

Interaktion der Maschine zum Menschen

Einer der beiden Kanäle der Informationsübertragung ist der Weg von der Maschine zum Menschen. In dieser Interaktion liegt die Möglichkeit des Systems, Ausgaben für den Nutzer bereitzustellen, in denen der aktuelle Zustand des Systems übermittelt werden kann.

Dieses Kapitel beschäftigt sich mit drei Möglichkeiten, Ausgaben des Systems an den Nutzer zu reichen. Unterschieden wird hierbei die visuelle Darstellung mittels AR oder VR und die akustische Übermittlung durch die Verwendung eines Systems zur Erzeugung von Virtueller Akustik (VA) System.

3.1 Augmented Reality zur visuellen Darstellung

Ein wesentlicher Bestandteil dieser Arbeit besteht darin, AR als Ausgabemöglichkeit zur Visualisierung der virtuellen Umgebung nutzen zu können. Daher dient die in diesem Kapitel beschriebene Komponente der grafischen Visualisierung der durch die AR-Applikation erzeugten Umgebung. Grundlage dieser Komponente ist das „Unifeye SDK“ (vgl. Kapitel 2.5.1). Da „Unifeye SDK“ bereits viele Basisfunktionalitäten zur Erzeugung, Visualisierung und Manipulation von virtuellen Objekten enthält, liegt in diesem Kapitel der Schwerpunkt auf der Realisierung eines Ansatzes, der die Darstellung von AR auf mehreren, verteilten Rechnern ermöglicht, da „Unifeye“ an sich nur den lokalen Betrieb unterstützt.

Die Komponente hat also die Aufgabe, die von der Applikation generierte virtuelle Umgebung und derer Manipulationen nicht nur auf einem Rechner lokal darzustellen, sondern diese auf beliebig viele Rechner zu erweitern. Hierbei ist die Synchronität der Szene, also die Beschreibung der aktuellen virtuellen Umgebung, auf den einzelnen Rechnern von besonderer Wichtigkeit, da die asynchrone Darstellung vor allem bei der Verwendung eines HMDs mit Stereo-Technologie zu Einbußen in der Darstellungsqualität führt. Hier würde sich die Darstellung von ein und derselben Szene zu einem bestimmten Zeitpunkt unterscheiden, da sich jeder Rechner in einem unterschiedlichen Systemzustand befinden könnte. Ebenfalls können bei gleichzeitigem Zugriff von mehreren Nutzern die darzustellenden Daten inkonsistent werden, wenn Befehle zu unterschiedlichen Zeitpunkten bei den Nutzern ausgeführt werden, zum Beispiel das Manipulieren von Objekten, die bereits von anderen Nutzern gelöscht wurden.

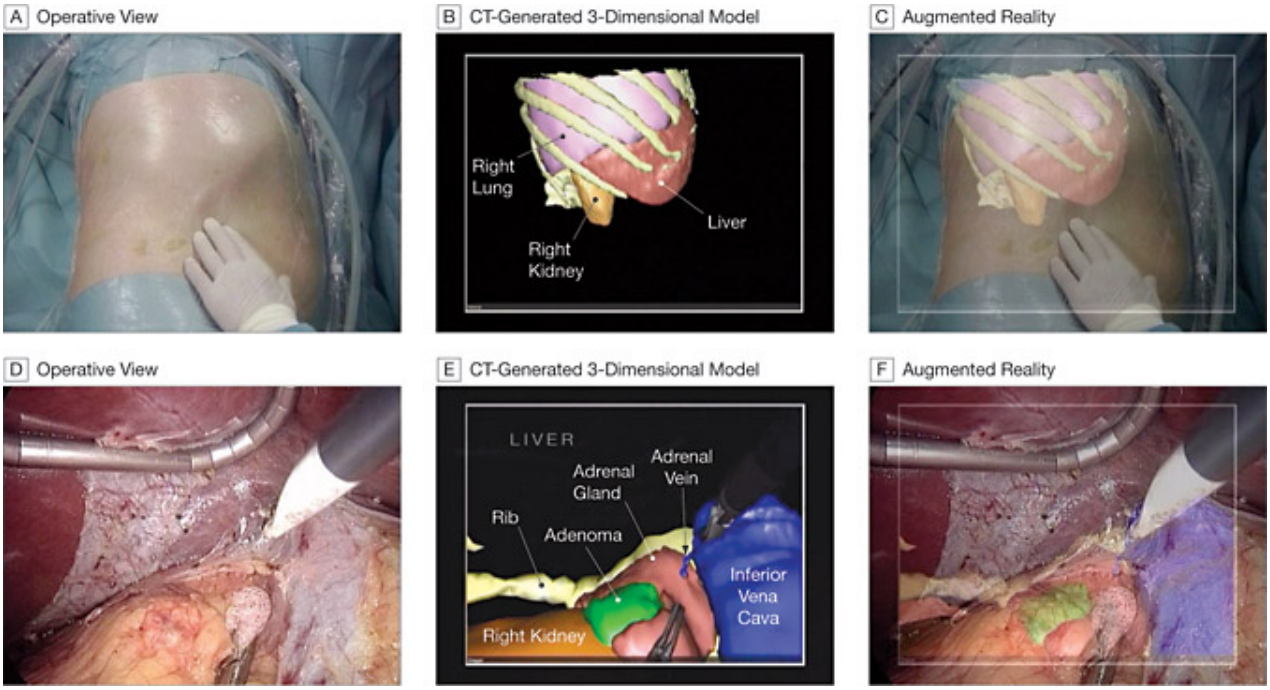


Abbildung 3.1: Überlagerung von virtuellen Patientendaten in einer Operation [54]

3.1.1 Thematisch verwandte Vorarbeiten

AR zur Visualisierung wird in vielen Bereichen eingesetzt. Diese reichen hierbei von militiärischer Verwendung, beispielsweise zur Informationsdarstellung für Soldaten [86], [37] oder [48] über die Unterstützung im Produktdesign bis hin zu medizinischen Applikationen. Abb. 3.1 zeigt „JAMA“, bei dem die Überlagerung von Patientendaten im aktuellen Sichtfeld des Operateurs ermöglicht wird [54].

Ebenfalls wird AR verwendet, um Spiele mit der Realität eines Nutzers zu überlagern. Beispiele für die Spieledomäne sind „AR Quake“ [68] oder „Human Pacman“ [12] (vgl. Abb. 3.2). Hierbei wurden zwei bekannte Spiele für den Computer („Quake“ und „Pacman“) auf ein AR-System umgesetzt, sodass sie mit der realen Umgebung überlagert und in Bezug zueinander gesetzt werden können.

Immer bedeutsamer wird die AR auch im Bereich des Fernsehens, wo zusätzliche Informationen, vor allem im Sport, mit der aktuellen Szene überlagert werden. Diese Technik kommt zum Beispiel bei der Übertragung von Schwimmen, Motorsport oder Fußball zum Einsatz. Hier können beispielsweise Distanzen in Form von Kreisen um Spieler, die Namen oder andere Informationen bzgl. der Schwimmer in die aktuelle Übertragung integriert werden [49] (vgl. Abb. 3.3).

3.1.2 Problemstellung und mögliche Lösungsansätze

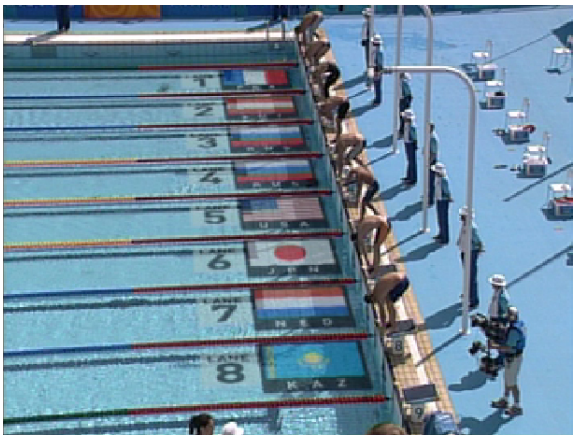
Um AR als Ausgabemedium zur Visualisierung der virtuellen Umgebung des Nutzers verwenden zu können, müssen bestimmte Anforderungen erfüllt werden. Kernpunkt ist die Trennung der Visualisierung von der Applikationslogik, sodass nur eine Komponente die Visualisierung übernimmt, die jedoch von der Applikation unabhängig ist. Auf diese Weise können beliebige Applikationen auf die Visualisierung zugreifen und es ist ebenfalls möglich, die nun getrennte Visualisierung sogar auf einen weiteren Computer auszulagern. Dies ist notwendig, um zum

(a) „AR Quake“ [68]

(b) „Human Pacman“ [12]

Abbildung 3.2: Augmented Reality in der Spieledomäne

(a) Verwendung beim Fussball



(b) Verwendung beim Schwimmen

Abbildung 3.3: Augmented Reality im Fernsehen [49]

einen ein Stereo HMD verwenden zu können, da dieses mittels zweier Computer für jedes Auge separat angesteuert wird. Folglich ist hier für jedes Auge ein eigener Betrachtungswinkel der Szene zu berechnen. Zum anderen können so auch mehrere Nutzer zeitgleich die Szene betrachten, die jeweils einen individuellen Standpunkt in der gemeinsamen Szene haben.

Die Darstellung der Szene muss demnach verteilbar sein und darf nicht mehr lokal beschränkt sein. Die Szene muss daher auf verteilten Rechnern synchron gehalten werden, um eine exakt gleiche Darstellung auf allen angeschlossenen Rechnern sicherzustellen. Dies ist sowohl für die Bereitstellung der Visualisierung für ein HMD notwendig, das sonst für jedes Auge verschiedene Szenenzustände darstellen würde, sowie für den Betrieb mit mehreren Nutzern, deren Ansicht der Szene nicht identisch wäre.

Eine Lösungsmöglichkeit wäre also, die aktuelle Szene nur auf einem einzigen Rechner zu speichern und die Berechnung der Darstellung auf diesem Rechner durchzuführen (vgl. Abb. 3.4). Hierbei müssten dann die generierten Bilder für jeden benötigten verteilten Rechner ebenfalls von diesem einzigen Rechner berechnet und anschließend auf die Ausgabe des Rechners übermittelt werden. Somit würde der gesamte Rechenaufwand für die Darstellung auf allen Rechnern von einem einzigen Rechner getragen werden. Dies würde zu einer deutlichen Einbuße der Gesamtperformance führen. Ziel ist es daher, auf allen verteilten Rechnern

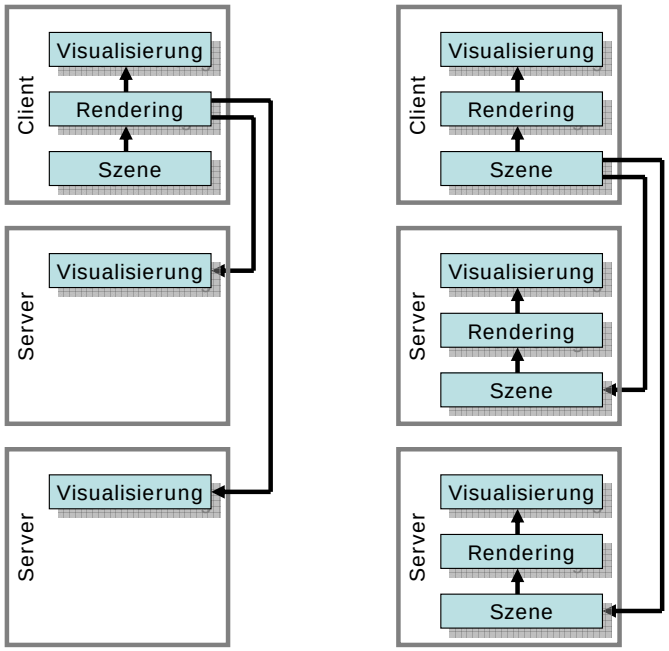


Abbildung 3.4: Methoden zur Synchronisation einer virtuellen Szene

die aktuelle Szene bereitzustellen und jeweils auf diesem Rechner die Darstellung zu berechnen. Dies ermöglicht zwar eine hohe Gesamtperformance, bringt aber die Problematik mit sich, dass jegliche Szenenänderung synchron auf allen verteilten Rechnern vorgenommen werden muss, damit als Grundlage der Darstellung nicht verschiedene Szenen dienen. Es muss hierbei also sichergestellt sein, dass eine neue Änderung der Szene nur erfolgen darf, sobald alle angeschlossenen Computer die letzte Änderung erfolgreich aktualisiert haben. Diese Methode bietet ebenfalls den Vorteil der beliebigen Skalierbarkeit, da die Rechenleistung immer auf die darstellenden Computer verteilt wird und nicht von einem einzelnen Computer getragen werden muss.

Im folgenden Kapitel wird der in dieser Arbeit entwickelte Lösungsansatz vorgestellt.

3.1.3 Umsetzung

Angelehnt an die Softwarestruktur (vgl. Kapitel 2.5.3) ist das Modul in zwei Teile untergliedert. Hierbei wurde ein Client-Server Ansatz umgesetzt, bei dem ein „Unifeye“ im Client die Verwaltung der gesamten Szene übernimmt. Beliebige verbundene Server erhalten sämtliche Änderungen der Szene mittels einer Netzwerkverbindung, welche sie dann an ein lokales „Unifeye“ weiterreichen.

Der Client steht hier der Applikation als Interface zur Verfügung und übernimmt die Kommunikation mit den Servern. Das bedeutet, dass für die Applikation nur der Client angesprochen werden muss. Dieser ist für die Applikation identisch mit dem originalen „Unifeye“. Befehle der Applikation werden vom Client ausgeführt, visualisiert und zeitgleich automatisch an den angeschlossenen Server weiter gesendet. Auf diese Weise erfolgt die gesamte Abwicklung der Synchronisation und Kommunikation innerhalb einer einzelnen Komponente. Diese kann von jeder Applikation angesprochen werden und ist für die Applikation unabhängig in der Anzahl der verbundenen Server. Weiterhin können innerhalb dieser Komponente weitere Funktionalitäten für die Interaktion mit der AR-Umgebung zur Verfügung gestellt werden.

Somit kann der Funktionsumfang des bestehenden „Unifeye“ erweitert werden.

Grundlegend ist für die Umsetzung des Moduls AR die Entwicklung eines definierten Nachrichtenformats für die Erzeugung der zu sendenden Datenpakete. Aufgrund der Tatsache, dass hier Funktionen mit teils optionalen Parametern verschickt werden sollen, kann hier keine einfache Konvertierung in reine Zeichenketten erfolgen. Die Auswertung wäre bei unbekannter Anzahl von Parametern einer Funktion nicht realisierbar. Aus diesem Grund wird auf das „ICE“ Framework zurückgegriffen, dass für die Kommunikation von verteilten Rechenprozessen entwickelt und optimiert wurde [36]. Dieses übersetzt auf dem Server und dem Client definierbare Funktionsaufrufe mit deren Parametern und Rückgabewerten in ein Format, das über Netzwerk übermittelt werden kann. Funktionen, die in diesem Framework definiert wurden, können danach im Programm aufgerufen werden, ohne dass zusätzlich für die Netzwerkübertragung gesorgt werden muss, da dieses das Framework erledigt.

Funktionsaufrufe der Applikation werden innerhalb des Clients direkt an das lokale „Unifeye“ weitergereicht. Gleichzeitig werden diese Funktionsaufrufe mittels „ICE“ in dessen Nachrichtenformat gewandelt und an die verbundenen Server verschickt. Hierbei liegt für jede Funktion eine exakte Beschreibung vor, in der der Name der Funktion und ihre möglichen Parameter und Rückgabewerte hinterlegt sind. Anhand dieser Beschreibung werden Funktionen in einen String kodiert, der per UDP an alle verbundenen Server geschickt wird. Die Beschreibung ermöglicht den Servern anschließend, den übermittelten String wieder in die ursprüngliche Funktion samt seiner Übergabewerte zu dekodieren. Diese Funktion wird dann innerhalb des Servers an dessen lokales „Unifeye“ weitergeleitet und ausgeführt. Einige Funktionsaufrufe des „Unifeye“ liefern Rückgabewerte, wie etwa die Abfrage von Sichtbarkeiten virtueller Objekte. Diese Rückgabewerte werden nur innerhalb des Clients verwendet und abgefragt. Server erhalten von Client keine Funktionen weitergereicht, die Rückgabewerte liefern würden (unidirektionale Übertragung). Server haben also einen rein darstellenden Charakter und dienen nur der Visualisierung. Somit stellt diese Komponente den vollen Funktionsumfang des originalen „Unifeye“ zur Verfügung.

Die Übertragung der Strings, die die Informationen der auszuführenden Funktionen enthalten, erfolgt nicht mittels TCP, sondern über UDP. Aufgrund der Kontrolle des Paketaustausches, die eine sichere Übertragung garantiert, entstehen teils sehr hohe Laufzeiten für Datenpakete. Diese verringern die maximale Anzahl an Befehlen pro Zeit erheblich (in Tests betrug die Rate teilweise unter fünf Befehle je Sekunde). Dynamische Manipulationen, wie etwa Translationen sind damit auf eine bestimmte Anzahl von Schritten pro Zeit begrenzt und führen zu einer ruckartigen Darstellung. Daher erfolgt die Übertragung mittels UDP. Die Übertragungsraten sind hier wesentlich höher (etwa 20 Befehle je Sekunde) und ermöglichen so flüssigere Manipulationen. Gleichzeitig wird aber auf die Sicherheit der Übertragung verzichtet.

Um die Synchronisation zu gewährleisten, wurde eine eigene Funktion erstellt, die als einzige einen Rückgabewert liefert. Sendet also der Client diese Funktion an einen Server, erhält der Client erst eine Antwort des Servers in Form eines Rückgabewertes, sobald dieser sämtliche vorher übertragenen Befehle ausgeführt hat, und er diese spezielle Funktion ausführt. Diese Funktion besteht lediglich aus dem Rücksenden eines einfachen boolschen Wertes (true). Auf diese Weise kann der Client erfahren, ob alle Server die Aktualisierung der Szene gemäß dem letzten Befehl abgeschlossen haben. Dadurch werden alle Server synchronisiert, da der Client den nächsten Befehl erst nach dem Erhalt des boolschen Wertes (true) aller Server versendet und somit auf alle Server wartet.

(a) „Half Life 2“ [16]

(b) „Second Life“ [35]

Abbildung 3.5: Virtual Reality in der Spieledomäne

3.2 Virtual Reality zur visuellen Darstellung

AR und VR sind verwandte Techniken zur Visualisierung von virtuellen Inhalten (vgl. Kapitel 2.1.1 und 2.1.2). Daher soll durch ein Modul eine Verbindung zwischen diesen beiden Visualisierungstechniken zur Verfügung gestellt werden.

Im einfachsten Fall steht AR und VR nur rein kontextuell in einem Zusammenhang. Hierbei besteht die Kombination der beiden Techniken darin, mittels der VR eine virtuelle Umgebung zu schaffen, die in einem inhaltlichen Zusammenhang zur AR steht. Die nächste Kombinationsmöglichkeit kombiniert die AR und VR nun nicht mehr nur inhaltlich, sondern auch szenenabhängig. Mittels VR werden hier also Teile der AR-Szene visualisiert. Dabei kommt eine virtuelle Kamera zum Einsatz, mittels der ein Ausschnitt der AR-Szene als VR-Szene dargestellt wird. Der komplexeste Fall ist die Verschmelzung von AR und VR. Hierbei wird die AR-Szene durch Inhalte der VR Szene ergänzt. Beide Szenen stehen jedoch nicht nur kontextuell, sondern auch physikalisch im Zusammenhang. In dieser Arbeit fand nur die Darstellung von AR mittels VR Einsatz und wird im folgenden Kapitel näher beschrieben.

3.2.1 Thematisch verwandte Vorarbeiten

VR wird schon seit einigen Jahren verwendet. Die Anwendungen reichen hier von der Unterstützung von Designern, die Prototypen großteils virtuell erstellen, über die Flugsimulation zum Training von Piloten, Lokführern etc. bis hin zum Spielebereich, in dem die immer realistischere Visualisierung oberstes Ziel ist. Des Weiteren entwickeln sich auch rein virtuelle Online-Gemeinschaften, wie etwa „Second Life“, die eine rein virtuelle Darstellung seiner Umgebung und seiner Bewohner (in Form von Avataren) bietet. Abb. 3.5 zeigt einen Screenshot des aktuellen Computer-Spieles „Half Life 2“ [16] sowie einen aus „Second Life“ [35].

Ebenfalls steigt mit der technischen Machbarkeit auch die Anzahl rein virtuell erstellter Filme, wie etwa „Toy Story“ oder „Shrek“.

3.2.2 Problemstellung

Bei der Kombination von AR und VR sollen Inhalte der AR innerhalb der VR abgebildet werden. Auf diese Weise können Nutzer, die nicht innerhalb der AR agieren, dennoch Zugang zu dieser erhalten. Hierzu muss sichergestellt werden, dass die zur Visualisierung zugrunde liegenden Daten in beiden Darstellungssystemen gleich sind, um Inkonsistenzen zu vermeiden.

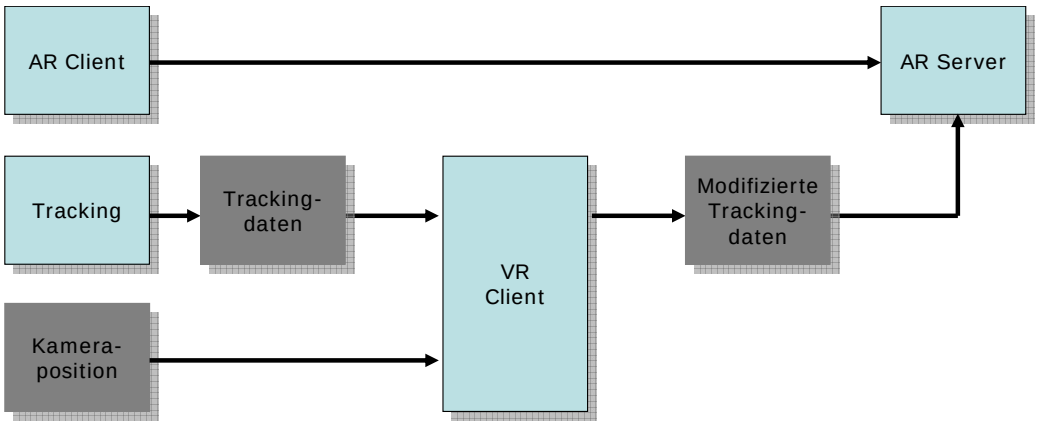


Abbildung 3.6: Umsetzung Virtual Reality

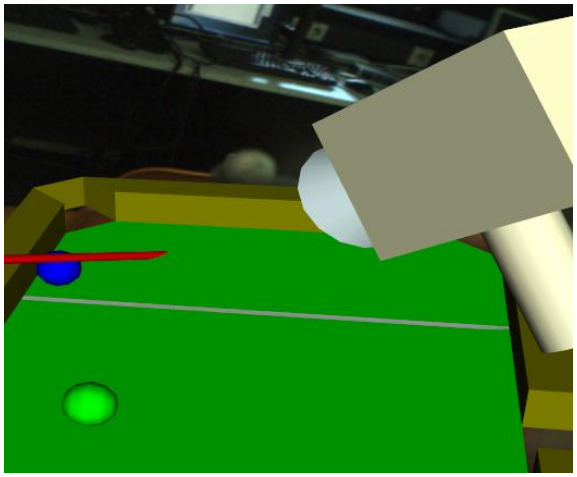
Die AR-Szene soll also aus mehreren Betrachtungswinkeln gleichzeitig dargestellt werden. Auf der einen Seite für den Nutzer, der sich tatsächlich eigenständig innerhalb der Szene befindet und dem diese ausgehend von seiner realen Position und Orientierung in dieser mittels des HMDs visualisiert wird, auf der anderen Seite dem Nutzer, der sich außerhalb der Szene befindet und diese mittels einer Projektion dargestellt bekommt. Diese Darstellung erfordert die Möglichkeit, eine Position und Orientierung innerhalb der Szene festzulegen, von der ausgehend die Visualisierung für den außenstehenden Nutzer erfolgt.

3.2.3 Umsetzung

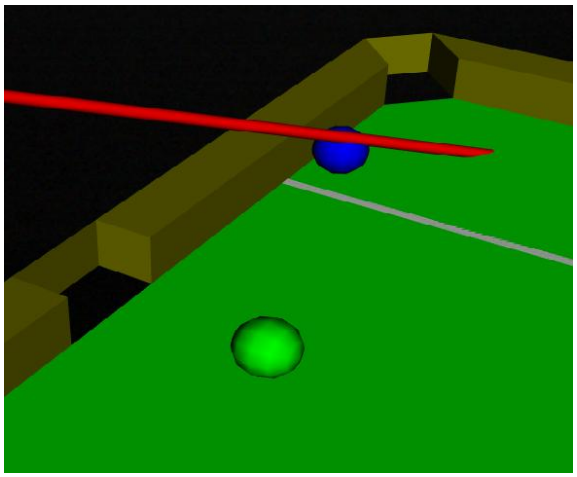
Um die Konsistenz zwischen AR- und VR-Applikation zu gewährleisten, basiert die Visualisierung der VR auf dem „Unifeye“, das die gesamte virtuelle Szene gespeichert hat. Somit können die virtuellen Inhalte beider Visualisierungsarten synchonisiert werden, da sie auf einem gemeinsamen Datensatz basieren. Die beiden Visualisierungstechniken unterscheiden sich nur darin, dass die VR die reale Umgebung in der Visualisierung ausblendet und den Standort und die Blickrichtung des Nutzers frei innerhalb der Szene platzieren kann, da kein Bezug zwischen realer und virtueller Umgebung existiert. Daher basiert das VR-Modul auf einem Server des AR-Moduls (vgl. Kapitel 3.1).

Dieses enthält bereits die Steuerbarkeit des Clients über ein Netzwerk, der die gesamte virtuelle Szene verwaltet. Zur Verwendung als VR-Modul enthält dieses die Modifikation der Darstellung der realen Umgebung sowie eine Möglichkeit, die Position und Orientierung innerhalb der virtuellen Umgebung frei zu wählen. Diese Positionierung kann sowohl als fixe Position in Weltkoordinaten angegeben werden als auch relativ zu einem KOS, das durch ein IR-Target definiert ist (vgl. Abb. 3.6). Je nach Wahl der Positionierungsart, verändert das VR-Modul den eingehenden Datenstrom des Trackingsystems. Bei einer fixen Positionierung innerhalb des WeltKOS werden die Position $\underline{x}_{t,1}$ und Orientierung $R_{t,1}$ (errechnet aus den Drehwinkeln $\eta_{t,1}$, $\theta_{t,1}$ und $\phi_{t,1}$) des ersten IR-Targets $t,1$ durch die fixen Positionierungswerte $\underline{x}_p$ und die feste Rotationsmatrix R_p (errechnet aus den Drehwinkeln η_p, θ_p und ϕ_p) ersetzt.

$$\underline{x}_{t,1} = \begin{pmatrix} x_{t,1} \\ y_{t,1} \\ z_{t,1} \end{pmatrix} = \begin{pmatrix} x_p \\ y_p \\ z_p \end{pmatrix}$$



(a) mittels Augmented Reality



(b) mittels Virtual Reality

Abbildung 3.7: Darstellungsarten einer virtuellen Szene

$$R_{t,1} =$$

$$\begin{pmatrix} \cos\phi_{t,1}\cos\theta_{t,1} & -\sin\phi_{t,1}\cos\theta_{t,1} & \sin\theta_{t,1} \\ \sin\phi_{t,1}\cos\eta_{t,1} + \cos\phi_{t,1}\sin\theta_{t,1}\sin\eta_{t,1} & \cos\phi_{t,1}\cos\eta_{t,1} - \sin\phi_{t,1}\sin\theta_{t,1}\sin\eta_{t,1} & -\cos\theta_{t,1}\sin\eta_{t,1} \\ \sin\phi_{t,1}\cos\eta_{t,1} - \cos\phi_{t,1}\sin\theta_{t,1}\cos\eta_{t,1} & \cos\phi_{t,1}\sin\eta_{t,1} + \sin\phi_{t,1}\sin\theta_{t,1}\cos\eta_{t,1} & \cos\theta_{t,1}\cos\eta_{t,1} \end{pmatrix}$$

$$= R_p =$$

$$\begin{pmatrix} \cos\phi_p\cos\theta_p & -\sin\phi_p\cos\theta_p & \sin\theta_p \\ \sin\phi_p\cos\eta_p + \cos\phi_p\sin\theta_p\sin\eta_p & \cos\phi_p\cos\eta_p - \sin\phi_p\sin\theta_p\sin\eta_p & -\cos\theta_p\sin\eta_p \\ \sin\phi_p\cos\eta_p - \cos\phi_p\sin\theta_p\cos\eta_p & \cos\phi_p\sin\eta_p + \sin\phi_p\sin\theta_p\cos\eta_p & \cos\theta_p\cos\eta_p \end{pmatrix}$$

Bei einer Positionierung relativ zu einem IR-Target werden die Werte des ersten IR-Targets in die entsprechenden Werte des gewählten Targets t, n verändert.

$$\underline{x}_{t,1} = \begin{pmatrix} x_{t,1} \\ y_{t,1} \\ z_{t,1} \end{pmatrix} = \begin{pmatrix} x_{t,n} \\ y_{t,n} \\ z_{t,n} \end{pmatrix}$$

$$R_{t,1} =$$

$$\begin{pmatrix} \cos\phi_{t,1}\cos\theta_{t,1} & -\sin\phi_{t,1}\cos\theta_{t,1} & \sin\theta_{t,1} \\ \sin\phi_{t,1}\cos\eta_{t,1} + \cos\phi_{t,1}\sin\theta_{t,1}\sin\eta_{t,1} & \cos\phi_{t,1}\cos\eta_{t,1} - \sin\phi_{t,1}\sin\theta_{t,1}\sin\eta_{t,1} & -\cos\theta_{t,1}\sin\eta_{t,1} \\ \sin\phi_{t,1}\cos\eta_{t,1} - \cos\phi_{t,1}\sin\theta_{t,1}\cos\eta_{t,1} & \cos\phi_{t,1}\sin\eta_{t,1} + \sin\phi_{t,1}\sin\theta_{t,1}\cos\eta_{t,1} & \cos\theta_{t,1}\cos\eta_{t,1} \end{pmatrix}$$

$$= R_p =$$

$$\begin{pmatrix} \cos\phi_{t,n}\cos\theta_{t,n} & -\sin\phi_{t,n}\cos\theta_{t,n} & \sin\theta_{t,n} \\ \sin\phi_{t,n}\cos\eta_{t,n} + \cos\phi_{t,n}\sin\theta_{t,n}\sin\eta_{t,n} & \cos\phi_{t,n}\cos\eta_{t,n} - \sin\phi_{t,n}\sin\theta_{t,n}\sin\eta_{t,n} & -\cos\theta_{t,n}\sin\eta_{t,n} \\ \sin\phi_{t,n}\cos\eta_{t,n} - \cos\phi_{t,n}\sin\theta_{t,n}\cos\eta_{t,n} & \cos\phi_{t,n}\sin\eta_{t,n} + \sin\phi_{t,n}\sin\theta_{t,n}\cos\eta_{t,n} & \cos\theta_{t,n}\cos\eta_{t,n} \end{pmatrix}$$

Auf diese Weise wird die Position des Betrachters der VR innerhalb der Weltkoordinaten gesetzt und für die Visualisierung herangezogen. Zusätzlich wird an diese Position $\underline{x}_{t,1}$ eine virtuelle Kamera (vgl. Abb. 3.7(a)) in die virtuelle Szene eingefügt, die für Nutzer mit einem

HMD sichtbar ist. Abb. 3.7(a) zeigt einen Ausschnitt einer AR-Szene aus der Sicht des Nutzers durch das HMD. Sichtbar ist ebenfalls die virtuelle Kamera mit der der Sichtbereich für die VR festgelegt wird. Im Gegensatz zur AR wird der Sichtbereich hier nicht durch die tatsächliche Position und Orientierung des Nutzers im Raum festgelegt. Abb. 3.7(b) zeigt dieselbe Szene aus Sicht der virtuellen Kamera. Diese Sicht kann nun mittels des VR-Systems beispielsweise mittels einer Projektion auf einer Leinwand dargestellt werden. Im Gegensatz zur Darstellung innerhalb der AR wird bei der Darstellung mittels der VR die reale Umgebung nicht eingebunden. Es werden hier also ausschließlich virtuelle Inhalte visualisiert.

3.3 Virtuelle Akustik zur akustischen Darstellung

Zur Erweiterung der AR-Umgebung von monomodaler (visueller) Ausgabe wird die akustische Komponente der Ausgabe hinzugefügt. Dadurch wird eine multimodale Ausgabe ermöglicht. Dieses Kapitel beschreibt die grundlegenden Begriffe der VA, erläutert die Problemstellung und geht im Anschluss auf den gewählten Lösungsansatz ein [87].

Binaural Room Impulse Response (BRIR) beschreibt die Filterfunktion, die ein realer Schall von seiner Quelle zum Empfänger durchläuft. Dieser Filter ist abhängig von der Umgebung, in der sich der Schall bewegt. So zählen zu den Parametern des Filters zum Beispiel die Geometrie des Raumes, die Oberfläche von Wänden oder Gegenständen innerhalb des Raumes oder die Beschaffenheit des Körpers des Nutzers. Vernachlässigt man die Parameter, die von der Umgebung des Nutzers hervorgerufen werden, so reduzieren sich die den Filter bestimmenden Parameter auf solche, die durch die Beschaffenheit des Körpers gestellt werden. Dieser reduzierte Filter wird Head Related Impulse Response (HRIR) genannt.

3.3.1 Thematisch verwandte Vorarbeiten

Um VA zu erzeugen, existieren bereits zahlreiche Ansätze. Dabei wird zwischen Ansätzen, die ein Lautsprecherarray (beispielsweise [82] oder [18]) oder einen Kopfhörer verwenden, unterschieden.

Für VA in Kombination mit AR können folgende Beispiele angeführt werden: Das „Listen“-Projekt erweitert die reale Umgebung des Nutzers mit akustischen Informationen [21]. Dabei bietet das System dem Nutzer intuitiven Zugang zu personalisierten und kontextabhängigen akustischen Informationen, während dieser seine Umgebung erkundet. Der Nutzer trägt dabei einen kabellosen Kopfhörer, der von einem Trackingsystem erfasst wird, um die gegenwärtige Position des Nutzers zu erkennen. Auf diesen Informationen basierend werden dem Nutzer entsprechende akustische Signale auf den Kopfhörern dargeboten. Eine Kombination von VA und AR findet man bei [30]. Dabei stützt sich das System auf [31], ein Framework, das es Designern ermöglicht, immersiv akustische Informationen in AR-Szenarien einzubetten und diese über verschiedene Ausgabegeräte, wie etwa Kopfhörer oder ein Surroundsystem, auszugeben.

3.3.2 Problemstellung

Das Integration der VA hat die Zielsetzung, dem Nutzer der AR-Umgebung eine virtuelle Schallquelle im Raum zu simulieren (siehe Abb. 3.8).

Dabei soll die Position der virtuellen Schallquelle innerhalb der AR-Umgebung sowohl statisch sein, also an einer festen Position im Raum positionierbar sein als auch dynamisch

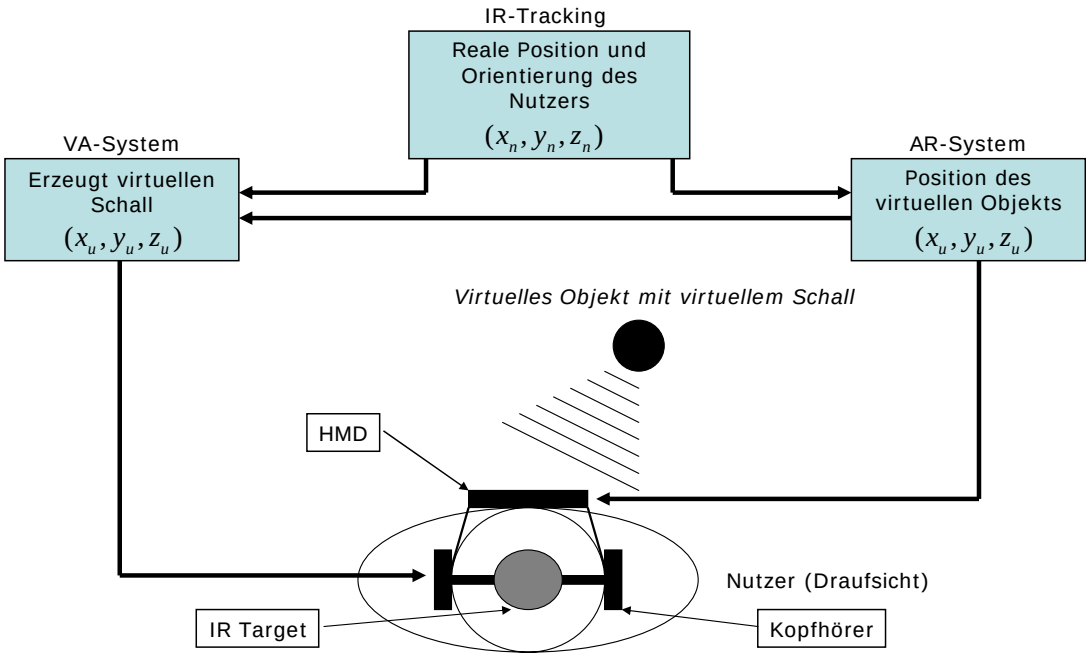


Abbildung 3.8: Schematischer Aufbau der Kombination von Augmented Reality und Virtueller Akustik

entlang einer vorgegebenen Trajektorie bewegt werden können. Gleichzeitig soll die Position von einer Applikation steuerbar sein, um eine Überlagerung von virtuellen Objekten und virtuellen Schallquellen zu erreichen. Um die Ermittlung der Position des Nutzers zu ermöglichen, soll das Modul an das IR-Trackingsystem (vgl. Kapitel 2.5.1) angebunden werden. Weiterhin müssen drei Kriterien erfüllt werden [53]. Die Latenz des Gesamtsystems darf 85 ms nicht übersteigen, um keine hörbaren Verzerrungen zu verursachen. Die Aktualisierungsrate der Trackingdaten muss mindestens 60 Hz betragen und BRIRs bzw. HRIRs müssen entweder gemessen oder interpoliert auf ein Grad genau vorliegen, um der Bildung von hörbaren Artefakten, wie Rauschen oder Knacksen, vorzubeugen.

3.3.3 Umsetzung

Die umgesetzte Lösung der VA basiert auf einem Ansatz, bei dem zur Wiedergabe der virtuellen Schalle ein Kopfhörer verwendet wird. Auf diesem ist ein IR-Target zur Bestimmung der Position und der Orientierung des Nutzers innerhalb des Raumes (siehe Abb. 3.9).

Dem System zur Erzeugung virtueller Schalle stehen zum Datenaustausch drei Interfaces zur Verfügung. Das Netzwerk-Interface dient der Kommunikation sowohl mit dem IR-Trackingsystem als auch mit der externen Steuerung über eine Fernsteuerung auf Softwarebasis. Das Akustik-Interface dient dem Austausch von Eingangs- und Ausgangssignalen. Eingangssignale können dem System entweder offline mittels gespeicherter Audiosignale in Form von Windows *.wav. Dateien oder online mittels des Analogeingangs der Soundkarte zur Verfügung gestellt werden. Das Datenbank-Interface stellt die benötigten Sets von HRIR/-BRIRs zur Verfügung. Zur Berechnung der Richtung aus der der Nutzer den virtuellen Schall wahrnehmen soll, werden die Daten des Trackingsystems mit Informationen zu Position und Orientierung des Nutzers im Raum sowie die Position der virtuellen Schallquelle im Raum herangezogen. Aus diesen wird die Richtung berechnet und die für diese Richtung zugehöri-

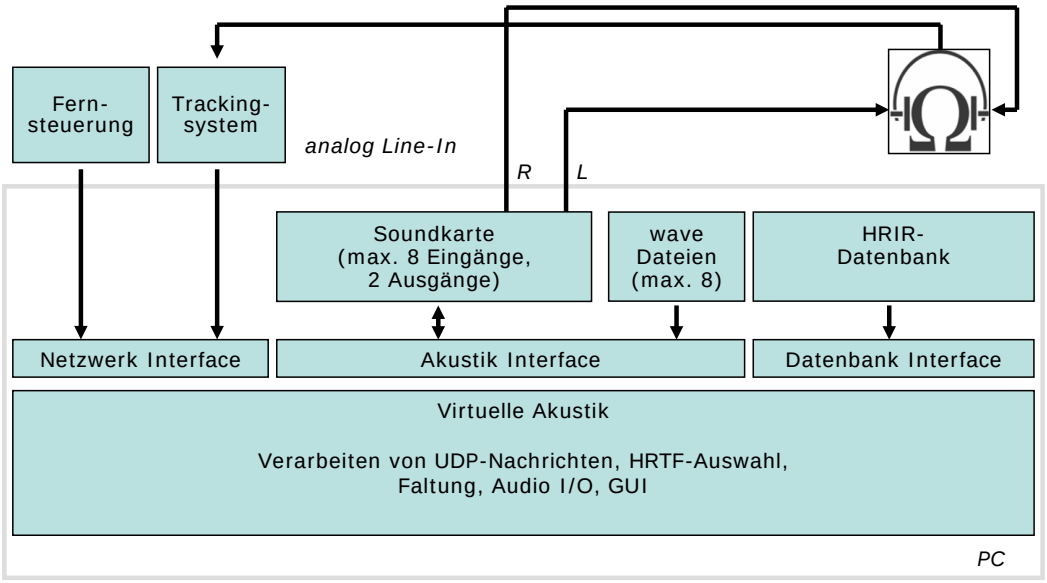


Abbildung 3.9: Umsetzung Virtuelle Akustik

ge in der Datenbank hinterlegte HRIR zur Berechnung des virtuellen Schalls herangezogen. Nach der Berechnung wird das Signal über die Kopfhörer an den Nutzer weitergeleitet und wiedergegeben. Die Berechnung der virtuellen Schalle erfolgt mittels partitionierter Faltung, dem Overlap Save Algorithmus.

Das Ausgangsignal $a(t)$ einer simulierten virtuellen Schallquelle berechnet sich aus der Faltung des Quellsignals $e(t)$ und der Filter-Impuls-Antwort $h(r)$. Das simulierte Signal ergibt sich also für das rechte Ohr zu

$$a_r(t) = e(t) * h_r(t)$$

und entsprechend für das linke Ohr zu

$$a_l(t) = e(t) * h_l(t)$$

Das VA-System ist in der Lage, bis zu acht Schallquellen parallel für das rechte und linke Ohr zu berechnen. Diese Berechnung kann sequentiell erfolgen. Dazu müssen Quellsignal und Impulsantwort des Filters dieselbe Samplingrate aufweisen. Alle Signale sind zeit- und wertdiskret mit n als unabhängige Variable, die die Zeit repräsentiert ($t = n \cdot T_a$). Das Eingangssignal $e_i[n]$ (Kanal i) passiert einen Filter mit begrenzter Impulsantwort des Filters $h_i[n]$. Die maximal mögliche Länge einer Impulsantwort liegt bei 144000 Samples. Bei einer Samplingrate von 48kHz entspricht das in etwa drei Sekunden. Diese Länge reicht aus, um größere Räume zu simulieren, da die Nachhallzeit hier etwa zwei Sekunden beträgt [91]. Das Ausgangssignal $a_i[n]$ eines nicht rekursiven Filters entspricht der Faltung des Eingangsignals, falls dieses endlich ist, mit der Impulsantwort des Filters

$$a_i[n] = e_{finite}[n] * h_i[n]$$

Die schnelle Frequenzfaltung ist ab einer Filterlänge von mehr als 30 Samples effizienter als eine Umsetzung im Zeitbereich [90]. Daher ist der Filter als schnelle Frequenzfaltung mit

$$F\{a_i[n]\} = F\{e_{finite}[n] * h_i[n]\} = E_{finite}[f_d] \cdot H_i[f_d] = A_i[f_d]$$

umgesetzt. Die Transformation erfolgt mittels (inverser) Fast-Fourier-Transformation (FFT). Um ein Eingangssignal mit unbegrenzter Länge verarbeiten zu können, verwendet die Faltung eine segmentweise Filterung des Signals. Das Eingangssignal $e_i[n]$ wird in begrenzte Blöcke $e_{finite}[n]$ mit der Länge l unterteilt. Jeder dieser Blöcke wird mit der Filterimpulsantwort mit der Länge K gefaltet, wobei die für den entsprechenden Winkel zwischen Blickrichtung des Nutzers und der virtuellen Schallquelle benötigte Impulsantwort verwendet wird. Die Berechnung des Winkels erfolgt aus der Position (x_H, y_H) innerhalb der Weltkoordinaten (x_w, y_w) des Kopfes H, die durch ein IR-Target bestimmt wird, und der Position (x_Q, y_Q) der virtuellen Schallquelle Q. Ebenfalls ist der Drehwinkel ϑ_H des IR Targets um die Z-Achse des WeltKOS bekannt. Aus diesen Angaben lässt sich der Winkel

$$\vartheta = sign(y_Q - y_H) \cdot \arccos\left(\frac{x_Q - x_H}{\sqrt{(x_Q - x_H)^2 + (y_Q - y_H)^2}})\right) \cdot \frac{180^o}{\pi} - \vartheta_H + \beta$$

berechnen, falls $(x_Q - x_H)^2 + (y_Q - y_H)^2$ nicht den Wert 0 annimmt. Um das daraus resultierende periodische Signal zu korrigieren, wird der Overlap-Save-Algorithmus verwendet (siehe Abb. 3.10).

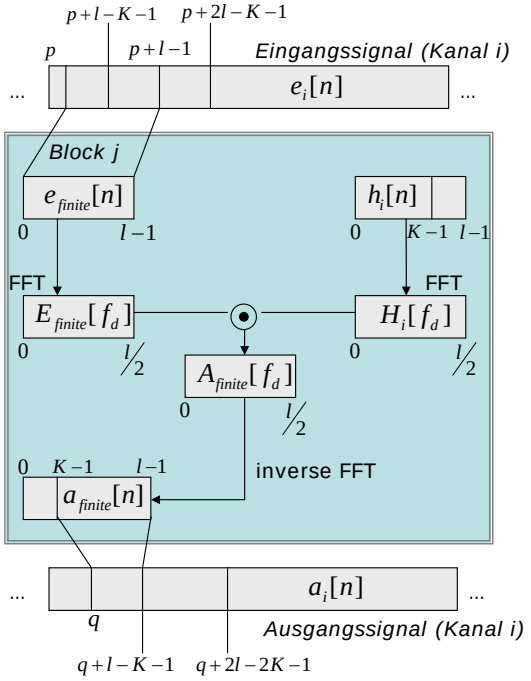


Abbildung 3.10: Overlap-Save Algorithmus

Dieser schneidet aus dem Eingangssignal $e_i[n]$ Blöcke der Länge $e_{finite}[n]$ mit der Länge von l Samples heraus. Bei jedem Iterationsschritt wird der Startpunkt des Eingangssignals um $(l - K)$ im Zeitbereich weitergeschoben. Daraus ergibt sich eine Überlappung der herausgeschnittenen Blöcke. Der hier verwendete Algorithmus verwendet für K den Wert $\frac{l}{2}$, was zu 1024 Samples oder 21,3 ms bei einer Samplingrate von 48 kHz führt. Durch die zyklische Faltung sind die ersten K Samples des Ausgangssignals $a_{finite}[n]$ nicht korrekt und werden verworfen. Aufgrund der verwendeten Filterimpulsantwort mit einer Länge von mehr als 30 ms wird der Algorithmus modifiziert [83]. Um eine dynamische partitionierte Faltung mit Impulsantworten einer Länge von 144000 Samples in Echtzeit umzusetzen, werden

sowohl das Eingangssignal (siehe Abb. 3.11) als auch die Filterimpulsantwort partitioniert (siehe Abb. 3.12).

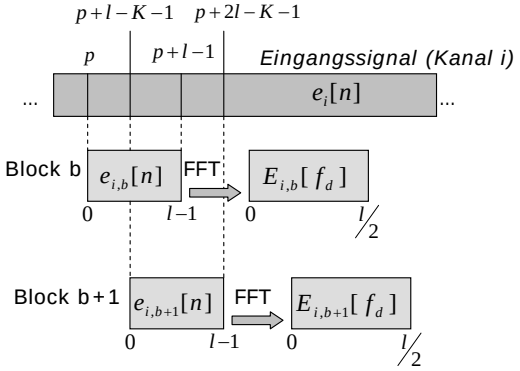


Abbildung 3.11: Partitionierung des Eingangssignals

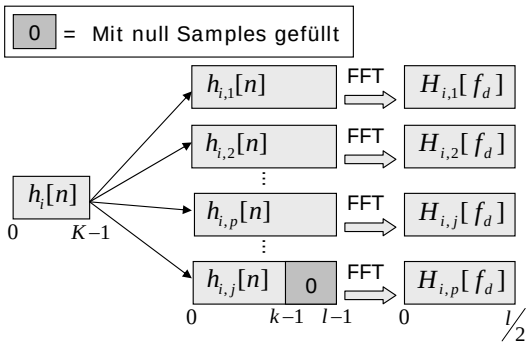


Abbildung 3.12: Partitionierung der Impulsantwort des Filters

Das Eingangssignal wird in überlappende Blöcke mit der Länge l unterteilt. Jeder Block $e_{i,b}[n]$ wird vom Zeitbereich in den Frequenzbereich transformiert

$$F\{e_{i,b}[n]\} = E_{i,b}[f_b)$$

Die Impulsantworten $h_i[n]$ mit der Länge von K Samples werden in p Impulsantworten $h_{i,j}[n]$ mit einer Länge von je k Samples untergliedert. Wenn die Division von $\frac{K}{k}$ keine natürliche Zahl ergibt, wird p aufgerundet und der restliche Block mit Samplen des Wertes null aufgefüllt. Daher enthält die Impulsantwort am Ende Stille, die jedoch die Funktion des Systems nicht beeinträchtigt. Jede der p Impulsantworten $h_{i,j}[n]$ wird mit dem Eingangssignal $e_{i,b}[n]$ gefaltet. Die daraus resultierenden p Ausgangssignale $A_{i,b,j}[f_d]$ im Frequenzbereich werden zu einem Ausgangssignal $A_b[f_d]$ addiert, wobei die jeweilige Verzögerung entsprechend ihres Indexes zu p berücksichtigt wird (siehe Abb. 3.13).

Das Ausgangssignal $A_b[f_d]$ wird mittels inverser FFT vom Frequenzbereich in den Zeitbereich transformiert $a_b[n] = F^{-1}\{A_b[f_d]\}$ und dem Ausgangssignal $a_i[n]$ hinzugefügt (siehe Abb. 3.14).

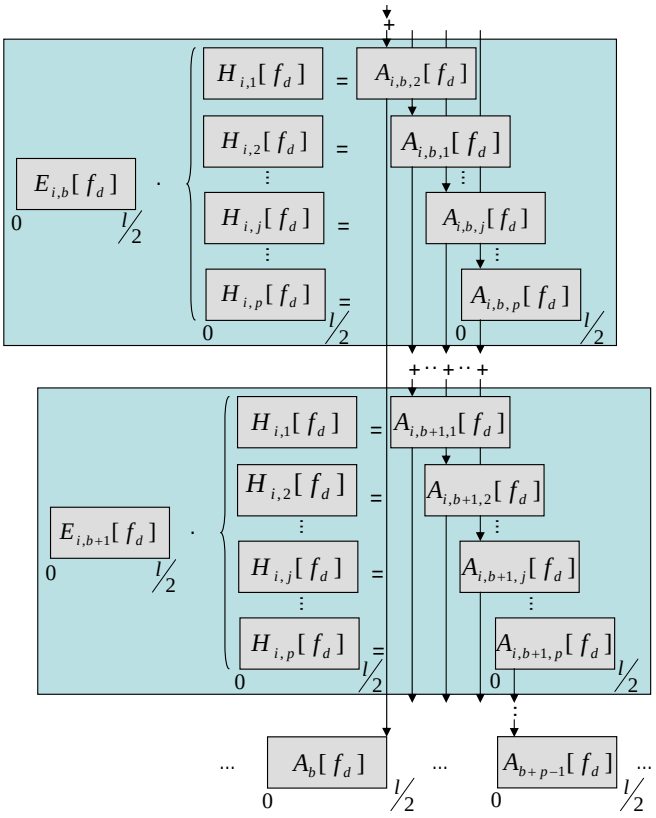


Abbildung 3.13: Faltung des partitionierten Eingangssignals und Impulsantwort des Filters

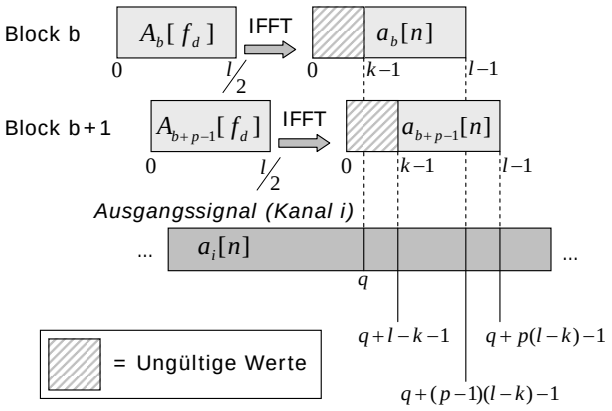


Abbildung 3.14: Erzeugung des Ausgangssignals

KAPITEL 4

Interaktion des Menschen zur Maschine

Damit eine echte Interaktion, also eine Art Dialog zwischen System und Nutzer entstehen kann, ist es notwendig, dass der Nutzer dem System Informationen übermitteln kann. Diese Informationen werden dann vom System ausgewertet und mit definierten Aktionen verknüpft.

Das folgende Kapitel beschreibt drei Möglichkeiten, mithilfe derer der Nutzer Eingaben an das System machen kann. Dabei sind diese Möglichkeiten speziell an den Einsatz bei der Visualisierung mittels AR angepasst.

Die Möglichkeiten sind hier die Eingabe mithilfe von Gesten, sowohl statische als auch dynamische, die Manipulation der Szene mittels eines Tangible User Interfaces (TUIs) sowie die Verwendung eines „WII“ Controllers einer handelsüblichen „Playstation“ von „Sony“. Dieser Controller wird hier als kombiniertes Ein- und Ausgabegerät verwendet. Es ist also in der Lage, sowohl Informationen vom System an den Nutzer als auch umgekehrt zu übermitteln.

4.1 Gestenerkennung durch Infrarottracking

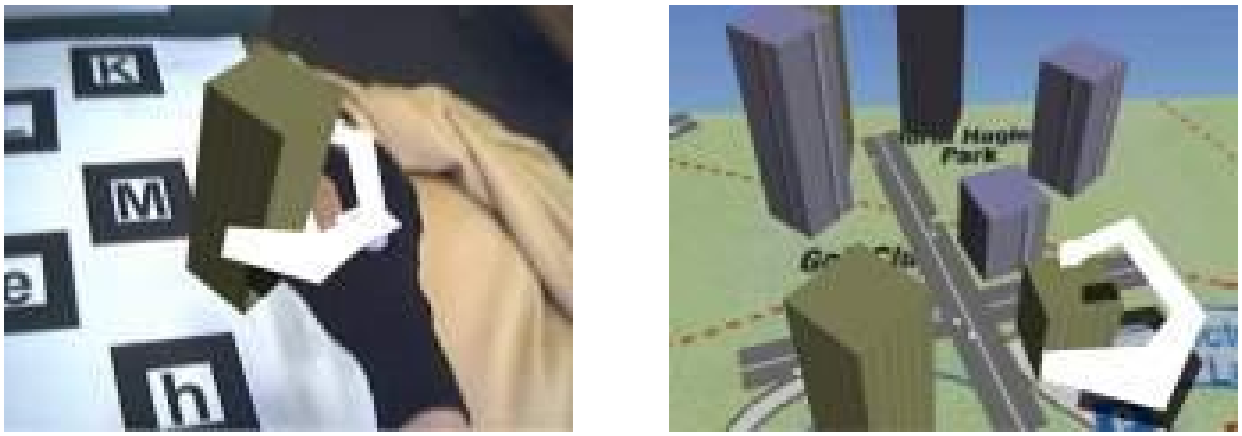
Das folgende Kapitel beschreibt die Integration eines Gestenerkenners, basierend auf einem IR-Trackingsystem (vgl. Kapitel 2.5.1). Dieses Trackingsystem wird parallel zur Berechnung der AR verwendet, das System zur Gestenerkennung bedarf hierbei also keiner zusätzlichen Hardware. Dabei fügt es sich in das bestehende Framework ein, besteht also aus einem Client, der der Applikation Gesten des Nutzers mitteilt, und dem Server, der die gesamte Erkennungsleistung erbringt [74].

4.1.1 Thematisch verwandte Vorarbeiten

Im Bereich der Gestenerkennung wird unterschieden zwischen „instrumentierten“ und „nicht-instrumentierten“ Erkennungsverfahren. „Instrumentiert“ bezeichnet hierbei Verfahren, bei denen die Hand des Nutzers mit zusätzlicher Hardware versehen wird. Hierzu zählen zum Beispiel Verfahren, die sich eines Handschuhs bedienen [46]. Diese jedoch extrahieren aus den Sensordaten des Handschuhs nur statische Gesten, also die aktuelle Fingerstellung der Hand des Nutzers.

Ein anderes Verfahren stellt FingARTips [9] vor. Hierbei werden drei Finger des Nutzers mit kleinen Papiermarkern versehen, die mithilfe einer Kamera und einer angeschlossenen Bildverarbeitung erkannt werden. Anschließend wird die Transformation zwischen Kamera

und Marker berechnet (vgl. Abb. 4.1). Basierend auf diesen Daten können Rückschlüsse auf Fingerstellung und daraus resultierenden Gesten gefolgert werden.

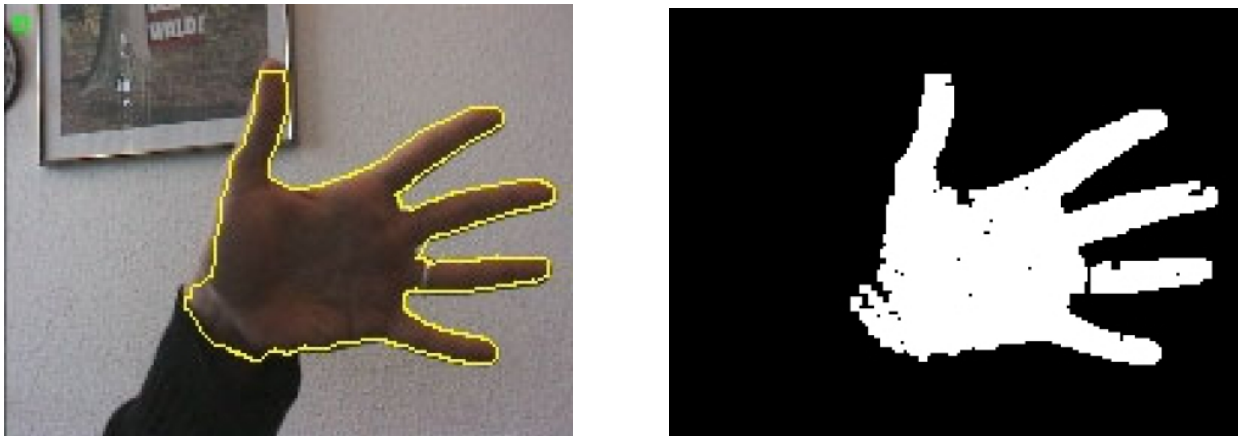


(a) Ansicht 1 (b) Ansicht 2

Abbildung 4.1: Beispiel für instrumentierte Gestenerkennung: FingARTips [9]

„Nicht-instrumentierte“ Ansätze dagegen setzen auf eine reine Erkennung anhand von Bilddaten. Hierbei wird die Hand des Nutzers gefilmt und daraus die Fingerstellung extrahiert. Aus diesen kann anschließend ebenfalls wieder auf Gesten rückgeschlossen werden. [59] verwendet hierbei zwei Kameras, die auf dem HMD des Nutzers befestigt sind, und damit die Finger bzw. Hände des Nutzers beobachten.

[58] verwendet ebenfalls eine Kamera, die die Hand des Nutzers beobachtet, um aus den gewonnenen Bilddaten die Hand zu segmentieren und aus den so erhaltenen Daten auf Gesten des Nutzers zu schließen (vgl. Abb. 4.2).



(a) Farbiges Bild mit Umrandung (b) Extrahierte Hand

Abbildung 4.2: Beispiel für „nicht-instrumentierte“ Gestenerkennung: VTM - Videobasiertes Tracking-Modul [58]

4.1.2 Problemstellung

Das Ziel ist die Umsetzung eines Gestenerkenners mittels IR-Tracking, das auf dem bestehenden Trackingsystem aufsetzt und auch bei parallelem Betrieb der AR-Applikation betriebs-

fähig ist. Der Nutzer soll dann innerhalb einer AR-Applikation in der Lage sein, mit seinen beiden Händen sowohl statische als auch dynamische Gesten zur Interaktion mit der AR-Umgebung zu verwenden. Hierzu sollen beide Hände bzw. deren Finger durch das Trackingsystem erfasst und basierend auf den Trackingdaten eine Erkennung der Gesten ermöglicht werden. Dabei ist es ebenfalls notwendig, dass Gesten unabhängig der Position im Raum erkannt werden. Zudem muss sichergestellt werden, dass durch Verdeckung verursachte fehlende Daten die Gestenerkennung nicht vollständig zum Erliegen bringen.

4.1.3 Umsetzung

Um die Anforderungen an das Trackingsystem möglichst vollständig zu erfüllen, muss eine Interpolation der Positionswerte der getrackten Finger erfolgen, um hier mögliche Lücken im Datenstrom sinnvoll zu schliessen. Unabhängig von der Orientierung und Position des Nutzers im Raum müssen Gesten erkannt werden, wodurch der Einsatz einer Transformation der Positionen erforderlich wird. Diese aufgezeichneten Bewegungen sollen mit zuvor gelernten Bewegungsmustern verglichen werden. Die erkannten Bewegungsmuster sollen dann vom Client ausgegeben werden. Die Bewegungsmuster unterscheiden sich hierbei in statischen und dynamischen Gesten. Bei statische Gesten handelt es sich um zeitinvariante Stellungen der Hand, wie etwa Zeigen oder Greifen. Dynamische Gesten dagegen sind zeitvariante Stellungen der Hand und reichen von einfachem Winken bis hin zu in die Luft gezeichneten Buchstaben. Die Gestenerkennung in der AR erfordert, im Unterschied zur rein dynamischen Gestenerkennung, die Information über die Position des Ortes, an der die Geste ausgeführt wurde. Nur damit lassen sich Objekte, die ein Nutzer greifen möchte, identifizieren. Auf diese Weise wird dem Nutzer ermöglicht, virtuelle Objekte sowie reale Gegenstände zu greifen (wobei bei virtuellen Objekten kein taktiles Feedback übertragen wird) und somit seine Interaktionsgewohnheiten von der Realität in die Virtualität zu übertragen. Damit wird die Absicht der AR, Realität und Virtualität zu verschmelzen, weiter vorangetrieben. Bei der Verwendung eines Gestenerkenners, der keine Ortsinformationen ermittelt, könnte nur eine Interaktion erfolgen, bei der die Realität und die Virtualität des Nutzers nicht gleichzeitig berücksichtigt würden. Ein solches System würde also nur als Trigger mittels Gesten dienen und Funktionen unabhängig vom Ort der Gesten ausführen.

4.1.3.1 Übersicht

Das Gestenerkennungssystem ist in drei Teile untergliedert (vgl. Abb. 4.3). Die Positionsdaten von IR-Targets innerhalb des WeltKOS liefert das IR-Trackingsystemmit einer Auflösung von einem Millimeter. Daher trägt der Nutzer des Gestenerkenners an den Fingern (Daumen und Zeigefinger an beiden Händen) passive oder aktive (vgl. Kapitel 4.1.4) IR-Targets (vgl. Abb. 4.4). Auf diese Weise ist es möglich, die aktuelle Position und Orientierung der vier Finger innerhalb des WeltKOS zu ermitteln.

Diese Positionsdaten werden vom Trackingsystem mittels UDP-Nachrichten an den Server des Gestenerkenners geschickt. Die Daten durchlaufen im Server drei Schritte: In einer Vorverarbeitung werden die Bewegungsdaten segmentiert, wobei die Start- und Endpunkte dynamischer Bewegungen detektiert werden. Anschließend werden die Bewegungen interpoliert, um mögliche verlorene Daten zu kompensieren. Abschließend wird die Bewegung von einer Bewegung in 3-D im Raum in eine 2-D-Ebene transformiert. Die so gewonnenen, segmentierten Bewegungen werden in Teilbewegungen, ähnlich der Phoneme bei der Spracherkennung, aufgeteilt, um sie anschließend klassifizieren zu können. Als Ergebnis übermittelt

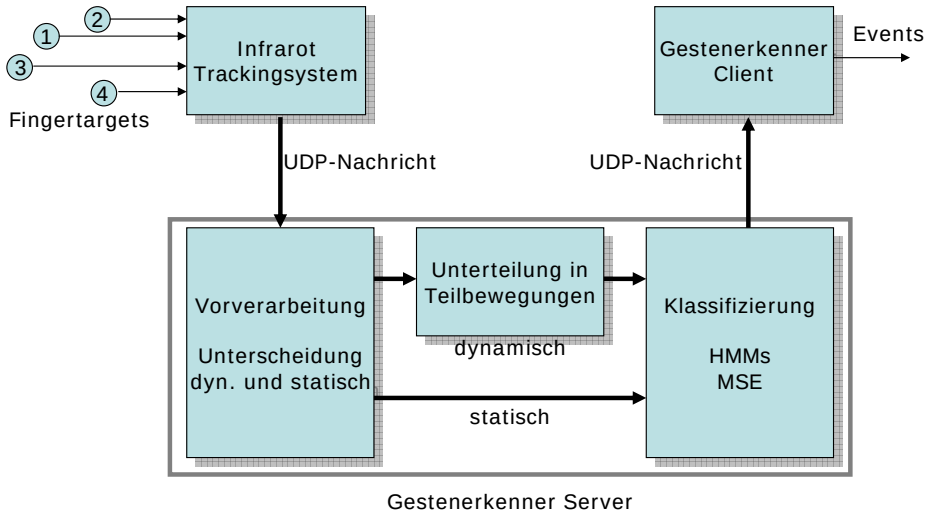
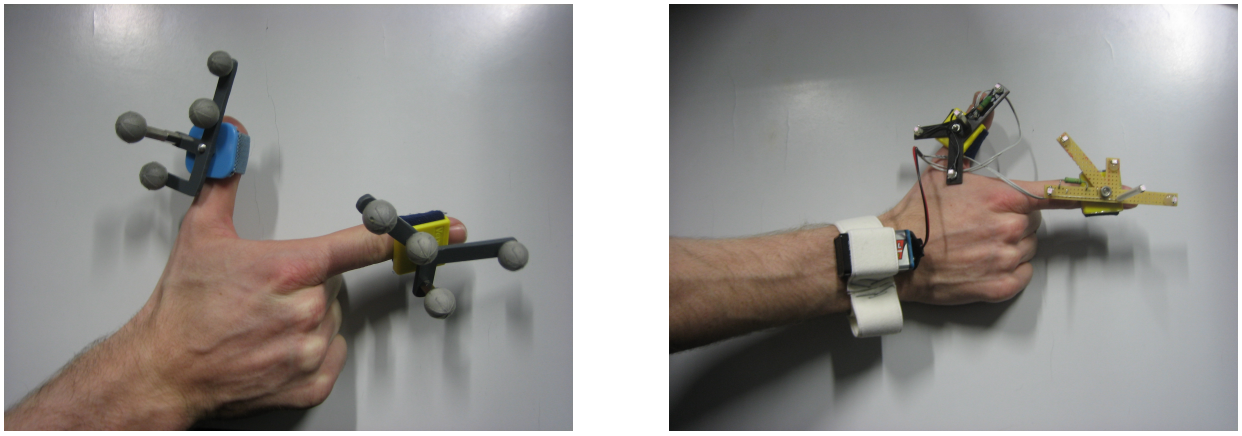


Abbildung 4.3: Schematischer Aufbau des Gestenerkenners

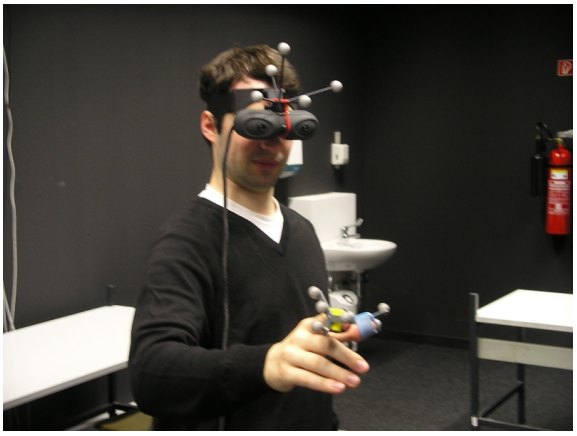


(a) Passive Targets

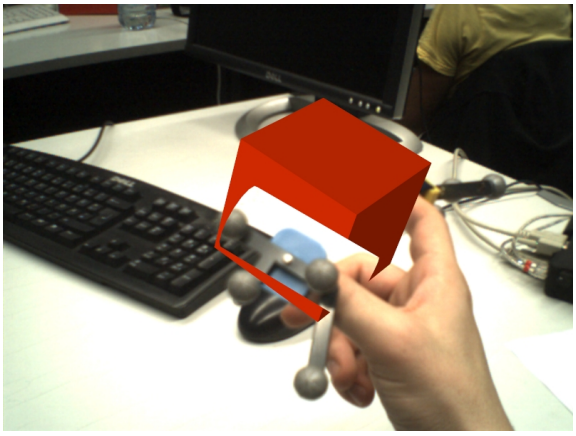
(b) Aktive Targets

Abbildung 4.4: Vergleich aktiver und passiver Fingertargets

der Server die erkannte Geste an den Gestenerkenner-Client. Dieser kann in jede beliebige Applikation eingebunden werden und stellt die Daten des Gestenerkenners in Form von Events zur Verfügung. Bei der Erkennung werden statische und dynamische Gesten unterschiedlich klassifiziert. Statische Gesten werden mittels des Abstands zwischen Zeigefinger und Daumen zugeordnet. In dieser Arbeit wurden nur zwei statische Gesten umgesetzt, das Zeigen und das Greifen. Zusätzlich wurde eine Geste modelliert, die einen Ruhezustand definiert, die also keine Geste des Nutzers repräsentiert. Der Abstand zwischen Zeigefinger und Daumen jeder Hand wird anhand der eingehenden Trackingdaten ermittelt. Unterschreitet er einen Abstand von 2 cm, wird für die betreffende Hand die Geste „Greifen“ klassifiziert (vgl. Abb. 4.5). Ein Abstand über 10 cm dagegen beschreibt die Geste „Zeigen“, da hier der Abstand zwischen den Fingern maximal wird. Der restliche dazwischen liegende Bereich wird als Ruhezustand beschrieben, in dem der Nutzer keine Geste ausführt. Um nun Objekte im Raum greifen zu können, muss eine Umrechnung der Koordinaten des Objektes erfolgen. Ein Objekt, das ergriffen wurde, wird in das KOS des Targets transformiert, um mit der Hand mit geführt werden zu können. Die Position des Objektes ist allerdings im WeltKOS angegeben, soll aber



(a) Aus Sicht eines Beobachters



(b) Aus Sicht des Nutzers

Abbildung 4.5: Greifen eines virtuellen Objektes mittels Gestenerkenner

in Abhängigkeit des Targets angegeben sein. Daher ist es notwendig, die Position des Objektes in Abhängigkeit der Position des Targets der Hand zu berechnen und für die Visualisierung zur Verfügung zu stellen. Die Position $p_{O,T}$ des Objektes im TargetKOS lässt sich wie folgt bestimmen:

$$\underline{p}_{O,T} = R^{-1}(\underline{p}_{O,W} - \underline{p}_{T,W})$$

Hierbei entspricht $\underline{p}_{O,W}$ der Position des Objektes im WeltKOS, $\underline{p}_{T,W}$ der Position des Targets im WeltKOS und R der Rotationsmatrix des Targets im WeltKOS. Ebenfalls muss beim Loslassen des Objektes die Position aus dem TargetKOS wieder zurück in die Position im WeltKOS $\underline{p}_{O,W}$ errechnet werden, damit das Objekt in diesem dargestellt werden kann. Dies lässt sich durch

$$\underline{p}_{O,W} = \underline{p}_{T,W} + R \cdot \underline{p}_{O,T}$$

erreichen. Zur dynamischen Gestenerkennung ist eine Datenverarbeitung in mehreren Schritten notwendig, die im Folgenden erläutert werden.

4.1.3.2 Datenverarbeitung

Um Gesten zu erkennen, müssen die eingehenden Daten des IR-Trackingsystems segmentiert, interpoliert und transformiert werden. Ausgehend von diesen transformierten Daten wird eine Geste in Teilbewegungen unterteilt und mittels eines HMMs klassifiziert.

Segmentierung
Damit einzelne dynamische Gesten erkannt werden können, müssen Gesten als solche im Bewegungsablauf detektiert werden. Es muss also automatisch erkannt werden, ob der Nutzer aktuell eine dynamische Geste an das System abgibt. Aus diesem Grund findet eine Segmentierung zu Beginn der Gestenerkennung statt. Es werden daher nur solche Bewegungsmuster an die Erkennung weitergeleitet, die als Bewegungsmuster einer Geste eingestuft werden. Dazu wird die Bewegungsgeschwindigkeit der Finger im Raum verwendet, die sich einfach anhand der Informationen bzgl. der Position der Finger bestimmen lässt. Hierfür wird der Abstand der letzten Position zur aktuellen gebildet und mittels der Framerate von 60 Hz in eine Geschwindigkeit umgerechnet. Überschreitet die Bewegungsgeschwindigkeit einen eingestellten

Schwellwert (hier bei 20 cm je Sekunde) für eine Dauer von mehr als 350 ms, wird die Aufzeichnung der Positionen gestartet. Wird die Bewegungsgeschwindigkeit für eine Zeitdauer von einer Sekunde wieder unterschritten, wird die Geste als beendet eingestuft und an die Interpolation weitergereicht. Dabei werden die für die Zeitdauer aufgenommenen Positionen herangezogen. Es werden jedoch nicht die absoluten Positionen, sondern die Positionsänderungen übergeben. Dies ist notwendig, weil die Erkennung unabhängig vom Startpunkt der Bewegung erfolgen soll.

Interpolation
Aufgrund von Einflüßen der Umgebung, wie etwa Lichtverhältnisse oder Verdeckungen, können Daten bzgl. der aktuellen Position verloren gehen. So kann es beispielsweise passieren, dass die Positionsdaten eines Fingers über einen bestimmten Zeitraum nicht verfolgt werden können. Diese fehlenden Werte können nicht exakt bestimmt werden, sie müssen also mittels einer Interpolation geschätzt werden. Hierbei wird die „Hermitsche-Spline-Interpolation“ (HSI) benutzt. Diese wird jedoch nicht auf die Positionsänderungen angewandt, sondern auf die Start- und Endposition der zu interpolierenden Bewegungsstrecke. Das ist notwendig, da bei nicht erkannten Targets keine Positionsänderungen mehr erfolgen und somit keine Interpolation statt finden kann.

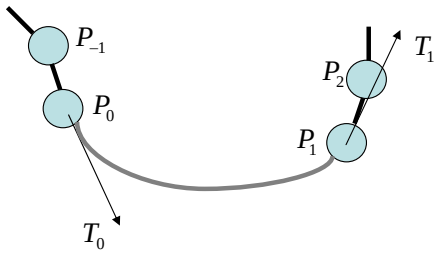


Abbildung 4.6: Hermitsche Spline Interpolation

Die so berechneten Positionen werden nach der Interpolation in Positionsänderungen umgerechnet und den bisher bestehenden Daten hinzugefügt. Für die Interpolation werden die Position P_0 und Tangente T_0 des letzten sichtbaren Punktes sowie die Position P_1 und Tangente T_1 des ersten wieder sichtbaren Punktes benötigt. Alle Positionen zwischen diesen beiden Punkten werden mittels folgender kubischen Grundfunktionen der HSI berechnet:

$$h_1(s) = 2s^3 - 3s^2 + 1$$

$$h_2(s) = -2s^3 + 3s^2$$

$$h_3(s) = s^3 - 2s^2 + s$$

$$h_4(s) = s^3 - s$$

Hierbei stellt der Parameter

$$s = \frac{1}{\text{Anzahl nicht erkannter Punkte}}$$

den Index des fehlenden Punktes dar, der auf den Wertebereich von 0 bis 1 skaliert wurde. Die Anzahl der nicht erkannten Punkte berechnet sich aus den Zeitstempeln der beiden Punkte,

aus deren Differenz die Anzahl der verlorenen Datensätze berechnet werden kann. Der Punkt mit Index s berechnet sich dann anschließend wie folgt:

$$P(s) = h_1(s) \cdot P_0 + h_2(s) \cdot P_1 + h_3(s) \cdot T_0 + h_4(s) \cdot T_1$$

Im Anschluss an diesen Schritt wird der so gewonnene durchgehende Bewegungsablauf an die Transformation weitergereicht. Durch diese Transformation wird die Bewegung von 3-D-Koordinaten in 2-D-Koordinaten projiziert.

Transformation
Um eine Erkennung von Gesten unabhängig von der Orientierung des Nutzers im Raum zu ermöglichen, benötigt die Erkennung eine immer gleichbleibende Beschreibung der Gesten. Diese muss also alle gleichen Gesten, welche jedoch an beliebigen Positionen im Raum ausgeführt werden können, einheitlich darstellen. Nur dann ist gewährleistet, dass Gesten an beliebigen Orten immer als ein und dieselbe Geste klassifiziert werden. Zusätzlich vereinfacht die Anwendung der Transformation die Klassifizierung, da ansonsten ein Training für jeden Ort im Raum für jede Geste durchgeführt werden müsste. Zur Transformation von 3-D-Koordinaten in 2-D-Koordinaten wurden zwei verschiedene Ansätze verfolgt, die auch in der Evaluierung gegenübergestellt und verglichen wurden (vgl. Kapitel 7.1). Beiden Ansätzen ist jedoch gemein, dass sie voraussetzen, dass Gesten immer annähernd orthogonal zur X-Y-Ebene ausgeführt werden. Diese Annahme kann getroffen werden, da der Nutzer Gesten zumeist im Stehen ausführt. Jedoch werden durch diese Einschränkung Gesten, die parallel zur X-Y-Ebene ausgeführt werden nicht erkannt, da sie in beiden Ansätzen auf eine Dimension reduziert werden und somit nicht mehr klassifizierbar sind.

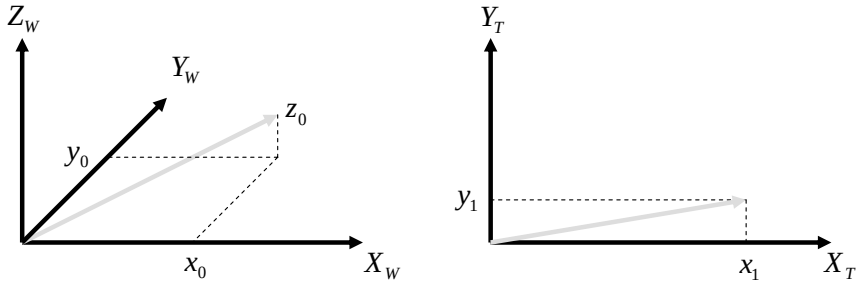


Abbildung 4.7: Längentreue Abbildung als Transformation

Der erste Ansatz bedient sich der Idee der längentreuen Abbildung, die die Änderung der Z-Koordinate in Weltkoordinaten z_0 direkt in die Änderung der Y-Koordinate y_1 überführt (vgl. Abb. 4.7). Die Formel hierzu lautet also:

$$y_1 = z_0$$

Die X-Koordinate der Abbildung berechnet sich aus der Länge der Bewegung in Weltkoordinaten unter Berücksichtigung der X- und Y-Änderung. Die Berechnungsformel hierzu lautet also:

$$x_1 = \sqrt{x_o^2 + y_o^2}$$

Auf diese Weise wird jeweils die Länge der Bewegung in der X-Y-Ebene auf die X-Komponente der Abbildung umgerechnet.

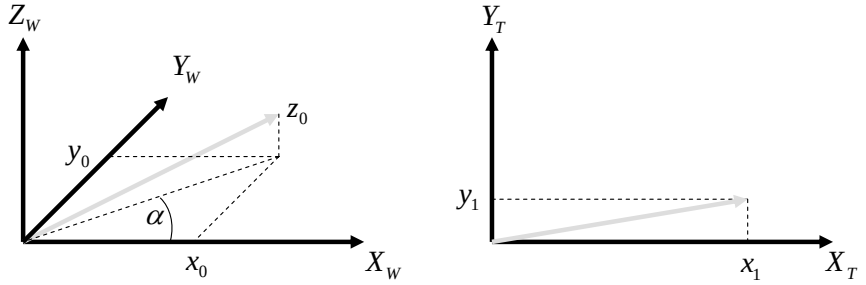


Abbildung 4.8: Rotation der Bewegungsebene als Transformation

Im zweiten Ansatz wird die gesamte Ebene, in der die Bewegung stattgefunden hat um die Z-Achse des WeltKOS gedreht. Anschließend wird die Z-X-Ebene des WeltKOS auf die Y-X-Ebene der Abbildung überführt vgl. Abb. 4.8). Bei der Rotation der Bewegungsebene wird also der Winkel zwischen der X-Achse des WeltKOS mit der Projektion der Bewegung auf die X-Y-Ebene gebildet. Um diesen Winkel wird dann die Bewegung um die Z-Achse des WeltKOS in entgegengesetzter Richtung gedreht. Der Winkel lässt sich hierbei einfach folgendermaßen bestimmen:

$$\alpha = tan^{-1}(\frac{Y_0}{X_0})$$

Im Anschluss daran wird wieder, wie bei Ansatz 1, die X-Z-Ebene des WeltKOS in die Y-X-Ebene der Abbildung überführt. Beide Ansätze resultieren im selben Ergebnis der Abbildung. Jedoch unterscheiden sie die Erkennung der Blickrichtung des Nutzers. Ansatz 1 geht davon aus, dass der Nutzer entlang der positiven Y-Achse des WeltKOS blickt, während er die Geste ausführt. Ebenfalls können durch die Rechenvorschrift nur Bewegungen entlang der positiven X-Achse der Abbildung abgebildet werden. Daher wird bei Ansatz 1 eine Abfrage nach der Richtung eingefügt, die den resultierenden X-Wert in der Abbildung entweder addiert oder subtrahiert. Ansatz 2 dagegen bezieht die Richtungsinformation aus einem initialen Abgleich der Zeigerichtung des Fingers des Nutzers. Ausgehend von diesem Winkel können nun die Orientierung des Nutzers bestimmt und entsprechend die Werte für die Abbildung angepasst werden.

Unterteilung in Teilbewegungen
Angelehnt an die Spracherkennung, die Phoneme zur Klassifikation verwendet, basiert die Erkennung der Gesten in dieser Arbeit auf der Kombination verschiedener Teilbewegungen. Diese Teilbewegungen sind 16 Basisbewegungen, aus denen sämtliche Gesten als Kombination konstruiert werden können, zu denen beispielsweise ein Halbkreis im Uhrzeigersinn oder eine Bewegung senkrecht nach unten zählen (vgl. Abb. 4.9).

So ist es möglich, Gesten auf verschiedene Arten zu modellieren. Eine „0“ kann von verschiedenen Nutzern auf verschiedene Weisen gezeichnet werden (beispielsweise mit oder gegen den Uhrzeigersinn). Durch die Verwendung der Teilbewegungen müssen nicht sämtliche verschiedenen Möglichkeiten vollständig trainiert werden, sondern es reicht die Zusammenfassung verschiedener Kombinationen zu einer einzigen Geste. Diese kann dann später per Client an die Applikation übergeben werden, andernfalls müsste die Zusammenfassung verschiedener Gesten die Applikation vornehmen.

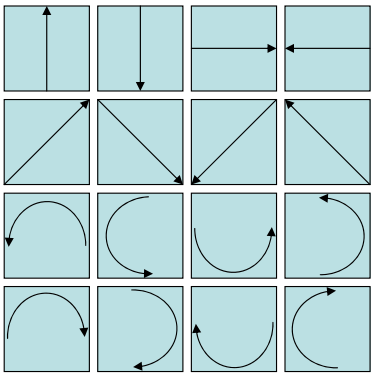


Abbildung 4.9: Teilbewegungen einer Geste

Training
Da der Fokus dieser Arbeit auf der gesamten Umsetzung von multimodaler Interaktion liegt, soll hier nur kurz auf das Training und die Erkennung mittels HMMs unter Verwendung des HTK Toolkits [20] eingegangen werden. Zu Beginn werden die 16 Teilbewegungen trainiert. Dazu wird durch den Nutzer eine Segmentierung vorgenommen, da dieser selbst den Start- und Endzeitpunkt einer Teilbewegung festlegen kann (dies erfolgt mittels eines Tastendrucks). Alle möglichen Teilbewegungen werden in einer Datei mit Namen definiert („models.list“), um diese später zuordnen zu können. In einem „Dictionary“ werden abschließend allen Teilbewegungen aussagekräftige Namen zugeordnet. Diese sind später Grundlage der Kombination zu Gesten. Daran anschließend werden die Gesten mittels einer „Dictionary“, in der der Name der Geste eingetragen ist, sowie deren zugrundeliegende Teilbewegungen definiert. Mit diesen Grundlagen kann nun das HTK Toolkit trainiert und für die Erkennung verwendet werden.

Erkennung
Für die Erkennung werden die Daten des Trackingsystems, wie zuvor beschrieben, segmentiert und vorverarbeitet. Diese Daten werden nun an das HTK Toolkit zur Klassifizierung übergeben. Dieses übergibt daraufhin eine Liste der Klassen sowie deren Wahrscheinlichkeit. Mit einem zuvor festgelegten Schwellwert kann nun bestimmt werden, wie zuverlässig bzw. sicher das Ergebnis der Klassifikation ist. Ist dieser Schwellwert überschritten, übernimmt der Gestenerkenner die zurückgelieferte Klasse und reicht das Ergebnis an den Client mittels einer Netzwerkverbindung weiter.

4.1.3.3 Kommunikation und Datenbereitstellung

Der Client erhält sämtliche Klassifizierungsergebnisse, die über den definierten Schwellwert reichen. Diese Ergebnisse werden über Events an die Applikation übergeben. Hierzu kann jede beliebige Applikation einen Client einbinden und die Events dieses Clients abbonieren. Auf Basis der Gestenerkennung können dann Funktionen durch Events getriggert werden, zum Beispiel kann ein Objekt durch eine bestimmte Geste geladen oder gelöscht werden.

Abb. 4.10 zeigt nochmals einen Nutzer, der mittels AR durch Gesten mit einem virtuellen Objekt interagieren kann. Dabei zeigt die Darstellung auf dem Bildschirm den aktuellen Blickwinkel des Nutzers innerhalb der AR.

(a) Nutzer nähert sich Objekt

(b) und ergreift es

Abbildung 4.10: Interaktion mittels Gesten

4.1.4 Verbesserung durch den Einsatz aktiver LEDs

Passive Marker sind einigen Nachteilen unterworfen. Neben einem hohen Anschaffungspreis ist deren reflektierende Oberfläche sehr empfindlich und nutzt im Verlauf der Zeit durch physikalische Einflüsse, wie etwa Stöße, ab. Außerdem sind passive Targets aufgrund der Baugröße der Kugeln räumlich sehr ausgedehnt, was bei der Verwendung als Target zum Tracking der Finger zu Problemen bei der Interaktion führt. Diese zeigen sich durch das Verhaken zweier Targets von benachbarten Fingern. Passive Marker sind daher auf zwei Stück je Hand begrenzt, um eine Trennung beider Targets zu ermöglichen und um den Tragekomfort für den Nutzer möglichst zu erhalten.

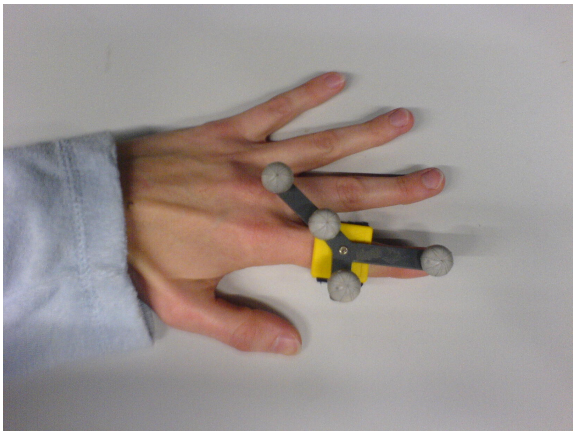
Aus diesen Gründen wurde ein alternativer Ansatz verwendet. Statt passiver Targets, die IR-Strahlung der Kamerablitze reflektieren, kommen aktive Leuchtdioden (Light Emitting Diodes - LEDs) zum Einsatz, die Licht derselben Wellenlänge aussenden. Somit können die bisherigen passiven Kugeln durch aktive LEDs ersetzt werden. Hier findet eine Unterscheidung zwischen einem vollständigen und einem teilweisen Target statt. Ein herkömmliches Target, das die Kugeln mit einer festen Anordnung an LEDs einfach ersetzt, wird als vollständiges Target bezeichnet. Einzelne, an den Fingerspitzen angebrachte LEDs, die anschließend regelbasiert bestimmten Fingern zugeordnet werden müssen, werden als teilweise Targets bezeichnet.

Beiden Ansätzen liegen identische Basisbedingungen zugrunde. LEDs benötigen eine aktive Stromversorgung, so dass deren Einsatz auf eine lange Lebensdauer optimiert werden muss. Weiterhin verursacht die Abstrahlcharakteristik einer LED Probleme bei der Erkennung im Trackingsystem, da eine LED einen auf etwa 120 Grad begrenzten Abstrahlwinkel hat.

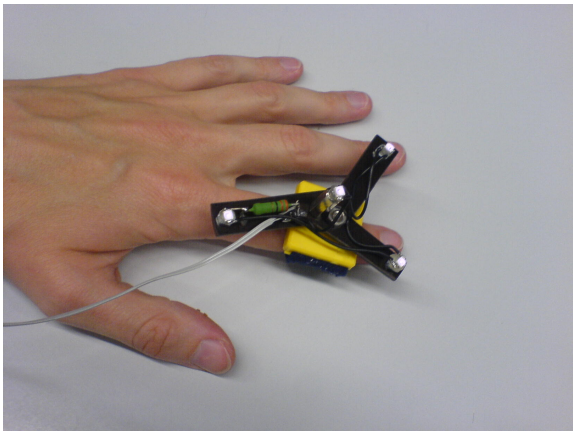
In dieser Arbeit wurden nur vollständige Targets verwendet, deren Umsetzung im Folgenden beschrieben wird.

Die Umsetzung eines vollständigen Targets beinhaltet die Ersetzung aller passiven Marker durch aktive Marker. Um eine gleiche Redundanz bzgl. der Reflexionspunkte zu erreichen, wurden wie auch beim passiven Modell vier Marker verwendet, woaus ein Target erzeugt wird (vgl. Abb. 4.11).

Die Hardware besteht aus einer einfachen Serienschaltung von vier LEDs. Diese benötigen eine Spannungsquelle und einen Vorwiderstand, der die an den LEDs anliegende Spannung begrenzt (vgl. Abb. 4.12).



(a) passiv mit Kugeln



(b) aktiv mit Leuchtdioden

Abbildung 4.11: Vergleich vollständige Targets mit passiven und aktiven Markern

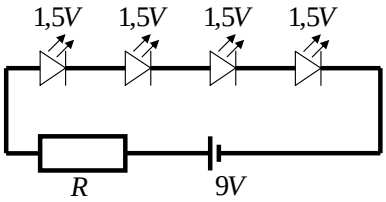


Abbildung 4.12: Serienschaltung des Targets

In der hier verwendeten Schaltung kommt eine 9 V Batterie zum Einsatz ($U_{Batterie} = 9V$) sowie eine LED des Typs „SFH421“, deren Spannung auf maximal 1,5 V begrenzt werden muss ($U_{LED} = 1,5V$). Der Strom durch die LED beträgt hierbei 100 mA ($I_{LED} = 0,1A$). Die daraus am Widerstand abfallende Spannung berechnet sich daher aus

$$U_R = U_{Batterie} - (4 \cdot U_{LED}) = 9V - (4 \cdot 1,5V) = 3V$$

zu 3 V. Aus diesen Angaben lässt sich die Größe des Vorwiderstandes aus

$$R = \frac{U_R}{I_R} = \frac{3V}{0,1A} = 30\Omega$$

zu 30 Ohm berechnen. Aufgrund der Stromstärke von 100 mA muss der verwendete Widerstand an die daraus resultierende Leistung angepasst werden, die sich aus

$$P = U_R \cdot I_R = 3V \cdot 0,1A = 0,3W$$

zu 300 mW berechnet. Daher kommt hier ein Widerstand mit einem halben Watt Leistung zum Einsatz. Da der Abstrahlwinkel der LED nur 120 Grad beträgt, wird hier eine Diffusionsfolie verwendet, die ähnlich wie ein Lampenschirm die abgehende Strahlung in alle Richtungen verteilt. Auf diese Weise wird der Abstrahlwinkel auf etwa 190 Grad erhöht, was die Strahlenintensität im Gegenzug aber vermindert. Die Befestigung der Batterie erfolgt am Handgelenk des Nutzers, um dessen Mobilität durch die Verwendung des aktiven Targets nicht einzuschränken. Hierbei wird die Batterie beispielsweise mit einem elastischen Band oder an einer Uhr als Träger befestigt (vgl. Abb. 4.13). Die Kosten für das beschriebene Target belaufen sich auf etwa fünf Euro.

(a) elastisches Band

(b) Uhr als Träger

Abbildung 4.13: Befestigungsmöglichkeiten der Batterie des aktiven Targets

4.2 Tangible User Interface als Eingabe

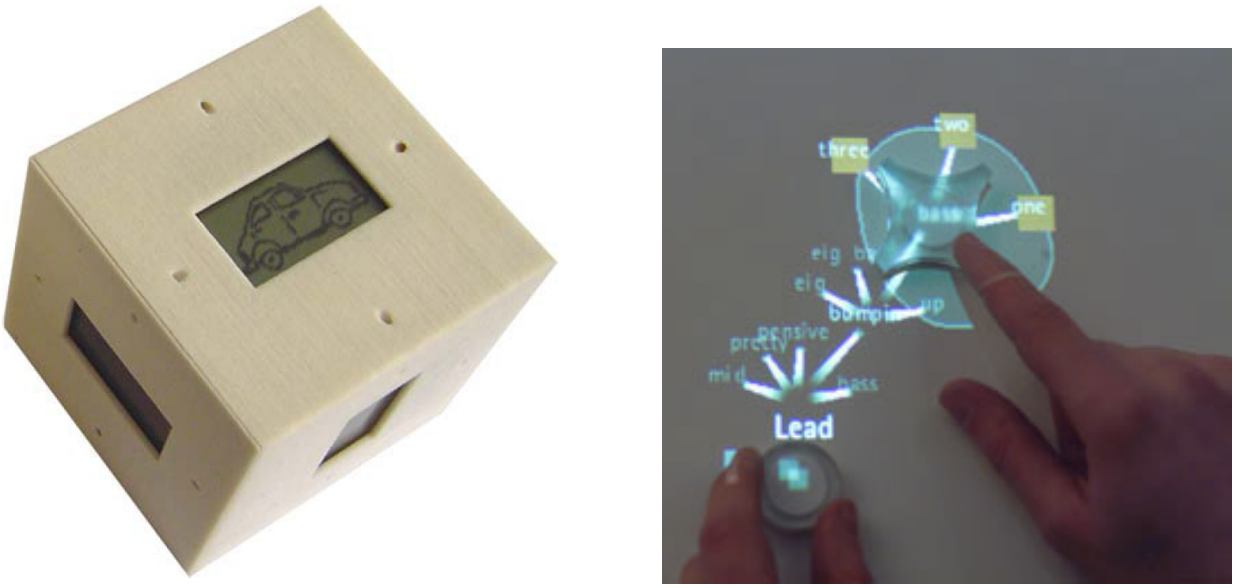
Das folgende Kapitel verfolgt einen weiteren Ansatz, neuartige und auf die 3-D-Umgebung der AR angepasste Eingabemöglichkeiten in das Gesamtkonzept zu integrieren. Hierbei kommt ein TUI zum Einsatz, welches ein Eingabegerät in Form einer realen Hardware darstellt, in das Sensorik zur Messung von Neigungen oder Rotationen integriert ist. Diese Sensoren liefern dabei Messdaten, die auf die Manipulation des Interfaces durch den Nutzer rückschließen lassen. Basierend auf diesen Messdaten können Veränderungen der Szene vorgenommen werden. Der grundlegende Gedanke eines TUIs ist also die direkte Interaktionsmöglichkeit mit der Virtualität mittels eines Interfaces. Dieses ist vom Nutzer greifbar und repräsentiert eine für die Virtualität angepasste Interaktionsmetapher.

4.2.1 Thematisch verwandte Vorarbeiten

TUIs können im Allgemeinen in zwei Gruppen unterteilt werden. Die einen stellen ein in sich geschlossenes Eingabesystem dar, das die gesamte Sensorik zur Eingabe und eventuelle Ausgabemöglichkeiten in sich vereint. Die andere Gruppe besteht aus mehreren Komponenten, wobei das eigentliche Interface nur durch einen physikalischen Körper repräsentiert wird und die Eingabe bzw. Ausgabe mit weiterer Hardware erfolgt.

[43] beschreibt einen Würfel, in dem die Sensorik zur Messung von Lageinformationen untergebracht ist (vgl. Abb. 4.14(a)). Ebenfalls beinhaltet dieser Würfel sechs Displays zur visuellen Ausgabe von Informationen an den Nutzer. Dieser kann mit Hilfe von Gesten, die durch die Sensorik gemessen werden, Eingaben an das System vornehmen. Dieser Würfel

gehört also zur Gruppe der Interfaces, die alle Funktionalitäten in sich vereinen.



(a) „Display Cube“ [43] (b) „Audiopad“ [66]

Abbildung 4.14: Beispiele für Tangible User Interfaces

„Audiopad“ dagegen stellt ein TUI dar, bei dem nur kleine Bausteine das eigentliche Interface abbilden [66]. Diese werden über eine Kamera, die oberhalb eines Tisches angebracht ist, getrackt (vgl. Abb. 4.14(b)). Die so gewonnenen Informationen über diese Bausteine dienen der Eingabe. So können bestimmte Funktionen durch bestimmte Positionen oder Orientierungen auf dem Tisch getriggert werden. Als Ausgabe an den Nutzer wird mittels eines Projektors visuelle Information auf den Tisch abgebildet. Dieses Interface ist also nicht in sich geschlossen.

4.2.2 Problemstellung

Vor allem für 3-D-Anwendungen, wie beispielsweise AR Szenarien, erscheint die Interaktion mit Maus und Tastatur ungeeignet, da beide Eingabegeräte eine feste Unterlage benötigen. Dies führt dazu, dass der Nutzer an seinen Arbeitsplatz gebunden ist und sich in seiner Umgebung nicht frei bewegen kann. Maus und Tastatur sind Eingabegeräte, die nur zwei Dimensionen zur Eingabe verwenden (X-Y-Ebene der Maus). Dies hat zur Folge, dass Manipulationen in AR, deren wesentlicher Bestandteil die Darstellung in 3-D ist, in mehrere Schritte aufgeteilt und in 2-D transformiert werden müssen. So muss eine Translation in 3-D in mindestens zwei Schritten in 2-D umgewandelt werden.

Diese Einschränkungen vermindern den Grad der Immersion der AR erheblich und widersprechen der Idee, die Umgebung des Nutzers virtuell so anzureichern, dass sie sich nahtlos in seine reale Umgebung eingliedert.

Ziel ist es also, ein Eingabegerät zu entwickeln, das 3-D-Manipulationen ermöglicht und sich in die Umgebung des Nutzers integrieren lässt. Dieses Interface ermittelt anhand geeigneter Sensorik die Manipulationen des Nutzers und setzt diese in eine Manipulation der Daten um. Auf diese Weise können virtuelle Objekte mittels realer Objekte manipuliert werden. Da bereits ein Gestenerkenner auf Basis des IR-Trackingsystems realisiert wurde, soll

in diesem Ansatz ein Interface entstehen, das von diesem Trackingverfahren unabhängig ist. Somit werden andere Varianten der Sensorik notwendig.

4.2.3 Umsetzung

Das folgende Kapitel beschäftigt sich mit der Umsetzung eines TUIs. Dieses verwendet einen Beschleunigungssensor sowie ein Gyroskop als Sensoren. Mit deren Messdaten können Applikationen Informationen bzgl. des Zustands des TUIs gewinnen und darauf basierend Funktionen ausführen. Somit lässt sich eine Steuerung mithilfe des TUIs erreichen.

4.2.3.1 Übersicht

Das System besteht in Anlehnung an das Gesamtsystem aus einem Server-Client-Ansatz (vgl. Abb. 4.15). Das TUI beinhaltet die Sensorik zur Messung seines aktuellen Zustands sowie Taster zur Eingabe von Nutzerintentionen. Diese Sensorik, bestehend aus einem Beschleunigungssensor sowie einem Gyroskop, übermittelt ihre Daten mittels der USB-Schnittstelle an den TUI Server, unter anderem die Kalibrierung der Sensorik, die einem kontinuierlichen Drift unterworfen ist. Die vorverarbeiteten Daten werden anschließend umgewandelt, um später der Applikation zur Verfügung stehen zu können. Diese werden der Applikation über den Client angeboten, der seine Informationen über ein Netzwerk vom TUI-Server erhält.

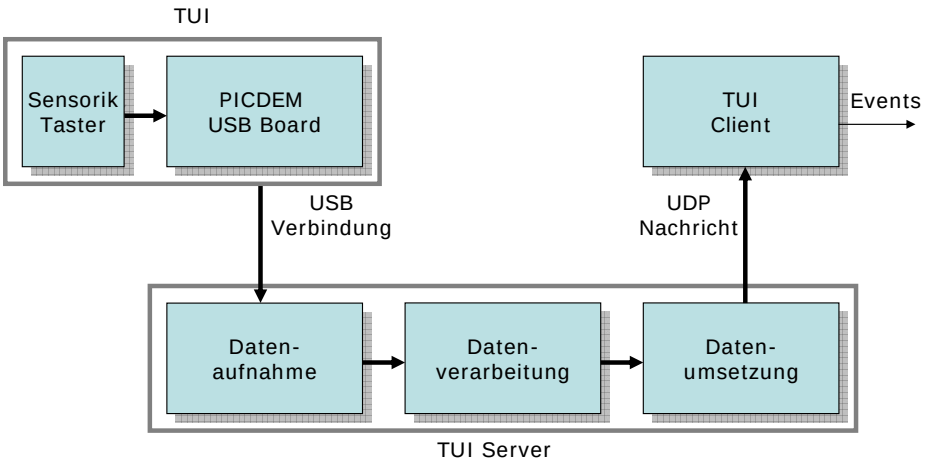


Abbildung 4.15: Schematischer Aufbau des Tangible User Interfaces

4.2.3.2 Hardware

Das TUI besteht aus vier Hauptbestandteilen: das Sensormodul, vier Drucktaster, das USB-Board zur Übertragung an den Server und das Gehäuse, das alle Komponenten vereint.

Sensormodul
Das verwendete Sensormodul ist angelehnt an das für die Luftfahrt verwendete Inertial Measurement Unit. Dieses System, welches aus einem GPS-Sensor, drei Beschleunigungssensoren und zwei oder drei Gyroskopen besteht, ist für die Bestimmung der Position und der Orientierung des Flugzeugs zuständig.

Aufgrund der hohen Kosten eines solchen Systems werden die Sensoren beschränkt. Zum Einsatz kommen in diesem System ein 2-D-Beschleunigungssensor „ADXL320“, der Beschleunigungen in Richtung seiner X- und Y-Achse erfassen kann, und ein Gyroskop „ADXRS300“, das Winkelgeschwindigkeiten um seine Z-Achse messen kann. Abb. 4.16 zeigt das Sensormodul, auf dem mit der Zahl 1 der Beschleunigungssensor und mit der Zahl 2 das Gyroskop gekennzeichnet sind. Weiterhin sind in der Darstellung die Richtungen der Achsen durch Pfeile beschrieben, wobei X die X-Achse, Y die Y-Achse und R_z die Rotation um die Z-Achse bezeichnen. Das Modul ist über die Pins „Ground“ und „5V“ an die 5 V Versorgungsspannung des „PICDEM^{TM} FS USB Boards“ angeschlossen. Mit diesen Sensoren wäre die Messung von fünf Freiheitsgraden möglich, wird hier aber auf drei begrenzt, um den Effekt des Drifts der Neigungssensoren zu eleminieren.

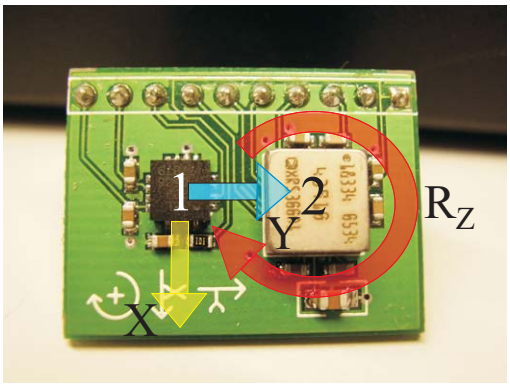


Abbildung 4.16: Sensormodul mit Beschleunigungssensor (1), Gyroskop (2)) und eingezeichneten Messrichtungen

Der „ADXL320“ von „Analog Devices“ ist ein 2-D-Beschleunigungssensor, der sowohl dynamische (z.B. Vibrationen) als auch statische (z.B. Gravitation) Beschleunigungen bis zu 5g messen kann [78]. Die Messung statischer Beschleunigungen bietet die Möglichkeit, den Sensor als Neigungssensor zu verwenden. Die Ausgangssignale der X- und Y-Achse werden als analoge Spannungswerte ausgegeben, die der aktuell wirkenden Beschleunigung entsprechen. Für die digitale Konvertierung wird der 10-Bit A/D Wandler des „PIC 18F2455 Mikrocontrollers“ verwendet. Der Beschleunigungswert, gemessen entlang der X-Achse des Sensormoduls, wird dafür über einen eigenen analogen Eingang des Mikrocontrollers gesendet, während der Wert für die Y-Achse über einen anderen analogen Eingang geschickt wird. Aus den Werten des Beschleunigungssensors kann zum einen die Neigung des Sensors um die entsprechende Achse gemessen werden, zum anderen die Translation entlang der X- und Y-Achse. Die Neigung ergibt sich aus der gemessenen Beschleunigung, wobei sie bei einem Wert von einem g äquivalent zu einer Neigung von 90 Grad ist. Durch zeitliche Integration können anhand der Beschleunigungswerte Entfernungen entlang der Achsen bestimmt werden. Hier müssten jedoch die Sensoren ohne Neigung bewegt werden, da sich sonst die Messung verfälscht. Dies liegt daran, dass Neigung und Bewegung nicht nur durch einen Messwert unterscheidbar sind. Daher werden die zu messenden Größen hier auf die Neigung um die jeweilige Achse beschränkt, die Messung der Translation des TUIs entfällt.

Beim „ADXRS300“ von „Analog Devices“ handelt es sich um ein Gyroskop, der Winkelgeschwindigkeiten um die Z-Achse von bis zu 300°/s messen kann. Die aktuell wirkende Winkelgeschwindigkeit wird als analoger Spannungswert ausgegeben. Die Drehung im Uhrzeigersinn ist als positive Drehrichtung definiert und führt zu einer Steigerung der Spannung.

Auch hier wird der 10-Bit A/D Wandler des „PIC 18F2455 Mikrocontrollers“ verwendet, um die Daten zu konvertieren. Die analogen Ausgangswerte werden über einen Pin an den analogen Eingang des Mikrocontrollers gesendet. Durch Messung der Winkelgeschwindigkeit kann durch zeitliche Integration auf den aktuellen Drehwinkel um die Z-Achse geschlossen werden. Jedoch ist zu beachten, dass Winkel nur bis 90 Grad gemessen werden können, da der Beschleunigungssensor nur absolute Werte liefert. Es ist daher nicht möglich zu differenzieren, ob ein Winkel zwischen 0 und 90 Grad, oder 90 und 180 Grad liegt.

Drucktaster
Neben der Sensorik, die den Status des Tangible User Interfaces misst, stehen dem Nutzer zusätzlich vier Taster zur Verfügung. Diese Taster dienen der Auswahl bestimmter Funktionsmodi und können durch den Nutzer aktiv verändert werden. Die vier Taster sind so angeordnet, dass sie vom Nutzer bequem mit den beiden Daumen bedient werden können, ohne dabei das Interface umgreifen zu müssen. Um verschiedene Funktionen umzusetzen, wurden zwei verschiedene Taster verbaut. Neben zwei Momentkontakttastern, die in Abb. 4.18 mit der Zahl 1 gekennzeichnet sind, befinden sich noch zwei einpolige Ein-/Austaster auf der Oberfläche des Gehäuses, welche mit der Zahl 2 markiert sind. Momentkontakttaster rasten beim Drücken im Gegensatz zu den Ein-/Austastern nicht ein. Das bedeutet, dass das Triggern einer Funktionalität nur für den Zeitraum des aktiven Drückens besteht. Beim Ein-/Austaster dagegen kann der Funktionsmodus durch einmaliges Drücken dauerhaft ausgewählt werden, bis er durch erneutes Drücken wieder deaktiviert wird.

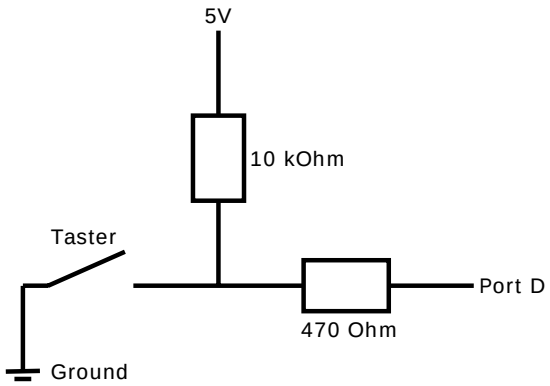


Abbildung 4.17: Pull-Up-Schaltung für die Tasten des Tangible User Interfaces

Die Taster wurden über separate Pins an den analogen Eingang des Boards angeschlossen. Da das Board an diesen Ports über keinen internen Pull-Up-Widerstand verfügt, wurde eine externe Pull-Up-Schaltung verwendet, welche in Abb. 4.17 dargestellt ist.

USB-Board
Um Daten der Sensoren und Taster an den Computer zur weiteren Verarbeitung zu leiten, wurde ein USB-Board verbaut. Ein solches USB-Board bietet die Möglichkeit, Daten angeschlossener Bauelemente direkt über USB an einen Computer weiterzureichen. Dazu kann im Speicher des Boards ein entsprechendes Programm hinterlegt werden, in dem die Übertragung definiert wird. Grundlegende Komponente des Boards ist ein Mikrocontroller des Typs „PIC 18F2455 USB-Mikrocontroller“, welcher USB 2.0 unterstützt und Datenübertragung mit low-speed (1.5 Mbit/s) sowie full-speed (12 MBit/s) erlaubt. Mithilfe mitgelieferter Software kann ein Programm erstellt werden, das im Speicher des Boards abgelegt wird und den

Datenfluss kontrolliert. Anschließend werden die Daten an den Computer übertragen, wo sie weiter verarbeitet werden können.

Gehäuse
Um die Sensorik sowie die Taster zur Interaktion verwenden zu können, müssen alle Komponenten in ein Gehäuse integriert werden, das dem Nutzer als TUI dient. Bei der Auswahl des Gehäuses sind mehrere Faktoren von Bedeutung. Das Interface soll ergonomisch gestaltet sein, damit es vom Nutzer auch länger in der Hand gehalten werden kann. Zudem sollen die Taster so angeordnet sein, dass sie leicht zu erreichen sind, ohne dass ein Umgreifen notwendig ist. Bei der Wahl der Form soll eine Anlehnung an Objekte des Alltags des Nutzers erfolgen, um eine leichte Gewöhnung zu erreichen und damit die Einarbeitungszeit zu verkürzen.

Als Form wurde hier ein Quader gewählt, da dieser neben Kugel, Kegel und Zylinder eine der vier geometrischen Basisobjekte darstellt. Vorteil des Quaders ist die Möglichkeit, alle drei Koordinatenachsen unterscheidbar zu machen. Durch die Wahl von drei verschiedenen Kantenlängen kann der Nutzer ohne visuelle Kontrolle die aktuelle Lage des Quaders bzgl. der drei Achsen feststellen. Auf diese Weise kann der Nutzer blind mit dem Interface interagieren.

Die Größe des Quaders bestimmt sich maßgeblich aus der Größe der zu verbauenden Komponenten. Die Ausmaße der Komponenten betragen 25 mm x 19 mm x 4 mm (Länge x Breite x Höhe) für das Sensormodul und 150 mm x 56 mm x 23 mm für das „PICDEM™ FS USB Board“. Wie aus diesen Maßen ersichtlich wird, wäre es von Vorteil, nur das Sensormodul in das TUI zu integrieren. Das Verbindungskabel zwischen Board und Sensormodul darf jedoch nicht beliebig lang sein, da das Ausgangssignal vom Sensormodul ansonsten stark abgeschwächt wird. Ein kurzes Kabel würde wiederum eine erhebliche Einschränkung bezüglich der Beweglichkeit des Benutzers bedeuten. Daher wurden sowohl das Sensormodul als auch das USB-Board in das Gehäuse integriert. Der reglementierende Faktor ist hierbei nur mehr das USB-Verbindungskabel, dessen Länge deutlich höher sein kann als das Verbindungskabel zwischen Sensorik und USB-Board.

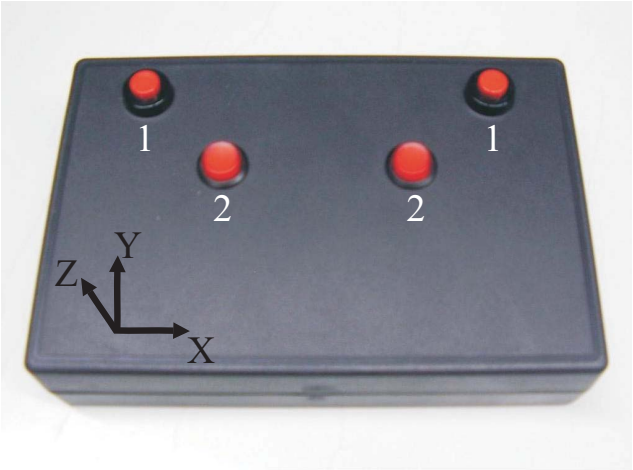


Abbildung 4.18: Design des Tangible User Interfaces

Aus diesen Überlegungen heraus wurde ein quaderförmiges Gehäuse gewählt, dessen Kantenlängen 186 mm x 123 mm x 40 mm betragen. In Abb. 4.18 ist das TUI mit den entsprechenden Koordinatenachsen dargestellt.

4.2.3.3 Datenverarbeitung

Das USB-Board besitzt einen 10 Bit A/D Wandler, mit dem die anliegenden Spannungen der Sensoren von analogen in digitale Werte überführt werden. Die Spannungen werden demnach als Werte zwischen 0 und 1024 dargestellt und an den Computer übertragen.

Betrachtet man allein das gewandelte Eingangssignal eines Sensors, so zeigt sich, dass bei ruhender Lage des Interfaces dennoch eine Änderung der Messwerte auftritt, da die Sensorik niemals völlig eben liegt oder sich völlig in Ruhe befindet. Das Interface liegt bei dieser Messung auf einer ebenen Unterlage und wird nicht bewegt. Der zu erwartende Messwert liegt bei exakt 512, schwankt aber um diesen erwarteten Wert. Auch stellt sich heraus, dass bei einer Auslenkung um 90 Grad in positiver bzw. negativer Richtung die Messwerte ebenfalls von den erwarteten Werten abweichen.

Um aus den Messwerten absolute Winkel zu berechnen, sind sowohl die Nullpunkte als auch die Punkte der maximalen Auslenkung notwendig. Aus diesem Grund muss das Signal an diese Abweichung angepasst, d.h. kalibriert werden.

Im Folgenden werden die Referenz-Nullpunkte mit $refx_{0g}$, $refy_{0g}$, $refg_0$ und die Referenz-Punkte der maximalen Auslenkung mit $refx_{+1g}$, $refx_{-1g}$, $refy_{+1g}$, $refy_{-1g}$ bezeichnet.

Für eine Kalibrierung des Nullpunktes $refg_0$ für das Gyroskop wird das TUI waagrecht auf einer festen Oberfläche aufgelegt. Die Ausgangssignale werden nun über einen Zeitraum von drei Minuten aufgezeichnet und anschließend wird der Mittelwert dieser Daten als Referenzwert verwendet.

Für die Bestimmung der Referenzwerte für eine Neigung um 90 Grad in positiver und negativer X- und Y-Achse $refx_{+1g}$, $refx_{-1g}$, $refy_{+1g}$, $refy_{-1g}$ wird das TUI senkrecht auf eine eben Fläche gestellt. Anschließend werden die Messwerte nacheinander für alle vier Seitenflächen über einen Zeitraum von drei Minuten gemessen. Aus diesen Messwerten wird der Mittelwert aller vier Testreihen bestimmt, welcher jeweils als Referenz für die entsprechende Achse und Richtung verwendet wird. Hierbei ergibt sich für die X-Beschleunigung $+1g$ im Mittel ein Wert von 567,44 (minimal 565, maximal 569), für $-1g$ von 442,25 (minimal 440, maximal 444) und für die Y-Beschleunigung $+1g$ ein Mittelwert von 578,22 (minimal 575, maximal 580) und für $-1g$ 453,5 (minimal 447, maximal 455).

Für die Berechnung der Nullpunkte $refx_{0g}$, $refy_{0g}$ wird der Mittelwert der jeweiligen Achse bzgl. der positiven und negativen maximalen Auslenkung gebildet (Referenzwerte für $-1g$ und $+1g$ der jeweiligen Achse). Es ergibt sich also zu

$$refx_{0g} = \frac{refx_{+1g} + refx_{-1g}}{2}$$

$$refy_{0g} = \frac{refy_{+1g} + refy_{-1g}}{2}$$

Die Messdaten für die Bestimmung des Sensormodul-Nullpunkts mit den berechneten Referenzwerten lassen sich für das Gyroskop im Mittel mit 575,21 (minimal 580, maximal 581), für die X-Beschleunigung mit 504,85 und für die Y-Beschleunigung mit 515,86 als Mittelwert bestimmen. Mithilfe der Referenzwerte kann nun durch

$$winkel_x = \frac{x_{accel} - refx_{0g}}{refx_{+1g} - refx_{0g}}$$

$$winkel_y = \frac{y_{accel} - refy_{0g}}{refy_{+1g} - refy_{0g}}$$

aus den Ausgangsdaten x_{accel} bzw. y_{accel} die Neigung des Beschleunigungssensors berechnet werden. Hierbei bezeichnen x_{accel} bzw. y_{accel} die aktuell gemessenen, digital gewandelten Spannungen der jeweiligen Sensoren.

4.2.3.4 Kommunikation und Datenbereitstellung

Die so ermittelten Werte für die Neigung bzw. Drehung um eine der drei Basisachsen wird anschließend über eine Netzwerkverbindung vom Server zum Client übertragen. Dieser speichert die aktuellen Messwerte intern ab. Zu diesen gehören die drei Winkel sowie der Zustand der vier Taster. Diese Werte sind jederzeit durch eine Applikation, die den Client integriert hat, abrufbar. Zusätzlich sendet der Client Events bei einer Veränderung des aktuellen Zustandes einer der Taster. Somit kann die Applikation durch das abonnieren der Events sofort auf eine Statusänderung einer der Taster reagieren und damit eine Funktion verknüpfen.

4.3 WII-Controller als kombiniertes Ein- und Ausgabegerät

Die in Kapitel 4.1 und 4.2 vorgestellten Ansätze sind als reine Eingabemöglichkeiten, die in Kapitel 3.1, 3.2 und 3.3 gezeigten Methoden dagegen als reine Ausgabemöglichkeiten vorgesehen. Die Ausgabemöglichkeiten sind hierbei auf die visuelle und akustische Ausgabe beschränkt. Ein wichtiger Rückgabekanal stellt jedoch das haptische Feedback dar.

Aus diesem Grund wird im folgenden Kapitel eine Möglichkeit vorgestellt, ein Gerät für eine kombinierte Ein- und Ausgabe zu verwenden. Dieses Gerät ist ein im Handel erhältlicher Game Controller, der sogenannte WII-Controller. Dieser kommuniziert drahtlos und stellt Ein- und Ausgabemöglichkeiten zur Verfügung.

4.3.1 Problemstellung

Um die reale Umgebung des Nutzers zu erweitern, wird im Allgemeinen bei AR auf den visuellen Kanal zurückgegriffen. Gerade die Interaktion mit AR-Szenarien erfordert jedoch nicht nur die Möglichkeit eines visuellen Feedbacks. So erhält ein Nutzer bei der Interaktion mit realen Gegenständen ein haptisches oder taktiles Feedback beim Greifen oder Berühren eines realen Objektes. Dies fehlt bei der Interaktion mit virtuellen Objekten. Aus diesem Grund ist es notwendig, ein Ausgabegerät zu integrieren, das in der Lage ist, ein haptisches Feedback für den Nutzer zu erzeugen. Ebenfalls sind die bisher vorgestellten Ansätze für die Ein- und Ausgabe voneinander getrennt. Es existiert also kein Gerät, das sowohl die Eingabe, als auch die Ausgabe vereint.

Ziel ist es daher, ein Gerät in die Umgebung einzubinden, dass diese Möglichkeiten bereit stellt. Da dieses frei in der realen Umgebung des Nutzers verwendet werden soll, wäre es erstrebenswert, dass die Kommunikation zwischen der Sensorik des Gerätes kabellos erfolgt, um die Bewegungsfreiheit des Nutzers nicht einzuschränken.

4.3.2 Umsetzung

Im folgenden Kapitel wird die Umsetzung des kombinierten Ein- und Ausgabegerätes beschrieben. Hierbei kommt ein WII-Controller von Nintendo [63] zum Einsatz. Dieser Controller bietet bei kabelloser Kommunikation über Bluetooth sowohl Eingabemöglichkeiten in

Form von Tastern als auch Ausgabemöglichkeiten mittels akustischer, visueller und haptischer Elemente. Gerade durch das gezielte Vibrieren bietet sich diese Hardware als Grundlage für das kombinierte Ein- und Ausgabegerät an.

4.3.2.1 Übersicht

Ähnlich zu den bisherigen Ein- und Ausgabegeräten besteht das hier vorgestellte System aus einem Server-Client-Ansatz (vgl. Abb. 4.19). Der WII-Controller ist via Bluetooth mit dem WII-Controller-Server verbunden. Dieser Server besteht aus der „WIImote LIB“ und einer Steuerung von Menü und Menülogik. Über die „WIImote LIB“ können sowohl die über den WII-Controller getätigten Eingaben ausgelesen werden als auch die Ausgabemöglichkeiten des Controllers angesprochen werden. Die Logiksteuerung des Menüs übernimmt hierbei die Ver-

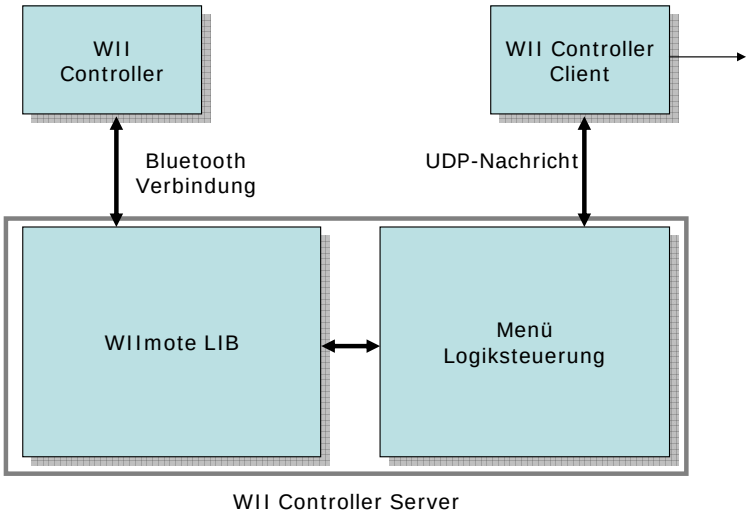


Abbildung 4.19: Schematischer Aufbau des WII-Controllers als Ein- und Ausgabegerät

mittlungstätigkeit zwischen der „WIImote LIB“ und dem WII-Controller-Client. Der Client stellt der Applikation Möglichkeiten bereit, sowohl die Eingaben über den Controller einzulesen als auch die Ausgabe über diesen anzusteuern. Server und Client sind hierbei über eine Netzwerkverbindung gekoppelt und kommunizieren über UDP Nachrichten.

4.3.2.2 Hardware

Grundlage des kombinierten Ein- und Ausgabegerätes ist der WII-Controller von Nintendo [63]. Dieser Controller hat eine Größe von etwa 3x3x15 cm. Über ein im Gerät verbautes Bluetooth Interface kann der Controller mit jedem Computer, der mit einem Bluetooth Interface ausgestattet ist, kommunizieren. Der Controller (vgl. Abb. 4.20) stellt insgesamt sieben Taster und ein Navigationskreuz (bestehend aus vier Tastern) zur Verfügung. Neben diesen enthält der Controller noch eine Kamera an der Spitze, die in der Lage ist, IR Strahlung zu detektieren, sowie Neigungssensoren, um die Lage im Raum bestimmen zu können. Beide werden jedoch für den weiteren Verlauf nicht berücksichtigt. Als Eingabe werden hier nur die Taster verwendet. Als Ausgabe stellt der Controller einen Lautsprecher für akustisches, vier LEDs für visuelles und ein Vibrationselement für ein haptisches Feedback zur Verfügung. Für das hier vorgestellte Gerät kommen nur die visuellen und haptischen Elemente zum Einsatz. Für das kombinierte Ein- und Ausgabegerät stehen also im weiteren Verlauf die Taster als

Abbildung 4.20: WII-Controller von Nintendo [63]

Eingabemöglichkeit, die LEDs und das Vibrationselement als Ausgabemöglichkeit zur Verfügung.

4.3.2.3 Software

Die Software des WII-Controller-Servers bestehen aus den beiden Komponenten „WIImote LIB“ [67] und der Steuerung von Menü und Logik des Menüs. „WIImote LIB“ ist eine auf C# basierende, frei verfügbare Bibliothek, die Basisfunktionen des WII-Controllers zugänglich macht. Die Kommunikation erfolgt durch die Übertragung eines Filestreames zwischen dem Controller und der „WIImote LIB“. Hierbei erzeugt der Controller einen aktuellen Zustandsvektor, der alle Informationen des gegenwärtigen Zustandes des Controllers abbildet. Hierbei sind beispielsweise die Taster als binärer Wert vorhanden, deren Wert je nach Stellung des Tasters mit Null oder Eins belegt ist (vgl. Abb. 4.21). Diesen Filestream empfängt die „WIImote LIB“, wertet ihn aus und erzeugt aus diesen Informationen Events, die von einer Applikation empfangen werden können, oder bietet die Möglichkeit, den aktuellen Zustand einzelner Elemente abzufragen. Für die Ansteuerung wiederum erzeugt die „WIImote LIB“

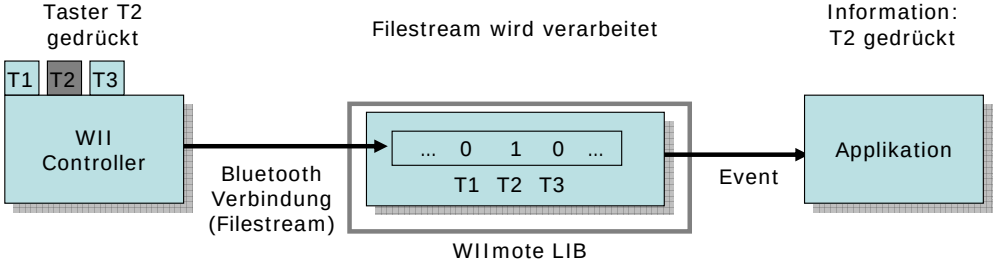


Abbildung 4.21: Exemplarische Umsetzung des Filestreams des WII-Controllers

einen Filestream, der an den Controller übertragen wird. In diesem Filestream können ebenfalls die Werte der Elemente des Controllers gesetzt werden. Der Controller empfängt diese Werte dann und setzt die Werte auf dem Controller um. So kann beispielsweise durch die Angabe des Wertes Null oder Eins eine LED ein- oder ausgeschaltet werden. Auf diese Weise kann die Logiksteuerung Rückmeldungen über den Controller an den Nutzer abgeben und dessen Eingaben mittels des Controllers zur Steuerung von Menüs heranziehen.

Der Aufbau eines solchen Menüs ist hierbei abstrakt umgesetzt worden. Da sich jedes Menü hinsichtlich der Struktur, dem Inhalt und dem Aussehen unterscheiden kann, wurden in der Menüsteuerung nur Basisklassen umgesetzt. Diese ermöglichen eine einfache Adaption an die geforderte Menüstruktur, deren optisches Erscheinungsbild und deren Inhalte.

Die Elemente, aus denen ein Menü entsteht, sind ausgehend von einem Hauptmenü entweder weitere Untermenüs oder Steuerelemente (vgl. Abb. 4.22). Steuerelemente stehen dabei als hierarchisch niedrigstes Element, weil sie kein weiteres Untermenü mehr aufrufen können, sondern direkt eine Aktion auslösen, wenn sie aktiviert oder deaktiviert werden. Untermenüs dagegen können sowohl Steuerelemente als auch wiederum weitere Untermenüs enthalten. Durch diese Unterscheidung können beliebige hierarchische Menüstrukturen aufgebaut werden.

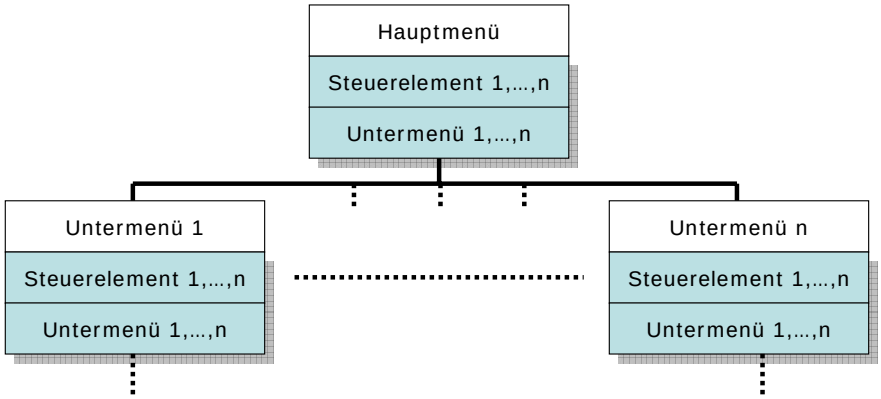


Abbildung 4.22: Hierarchie der Menüstruktur

Das Hauptmenü stellt gleichzeitig auch die logische Steuerung der Menüs zur Verfügung. Eine Applikation kann also über das Hauptmenü neue Untermenüs oder Steuerelemente erzeugen und dem Menü hinzufügen. Diese können mit einem eindeutigen Namen versehen werden, um später eine einfache Zuweisung zu Aktionen zu ermöglichen. Das Hauptmenü übernimmt ebenfalls die Steuerung und Koordination der Eingaben durch den WII-Controller. So sind hier feste Taster des Controllers für bestimmte Funktionen innerhalb eines Menüs vorgesehen. Beispielsweise kann in einem Untermenü durch Betätigen des Navigationskreuzes zwischen den Steuerelementen oder den weiteren Untermenüs navigiert werden oder durch das Betätigen der „Home-Taste“ des Controllers in das Hauptmenü zurückgesprungen werden. Auf diese Weise können die Eingaben über den Controller auf das Menü umgesetzt werden.

Um nun auch eine visuelle Rückmeldung für den Nutzer zu ermöglichen, beinhaltet das Hauptmenü Zugang zur aktuellen AR-Szene und kann virtuelle Objekte erzeugen und manipulieren. Die virtuellen Objekte können dann als grafische Visualisierung von Elementen des Menüs dienen. Über das Hauptmenü können diese für die jeweiligen Untermenüs und Steuerelemente frei definiert werden. Darüber hinaus beinhaltet ein solches Element die Möglichkeit, das ihm zugewiesene, virtuelle Objekt abhängig von seinem aktuellen Systemzustand

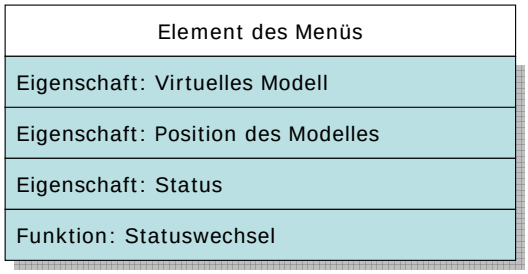


Abbildung 4.23: Aufbau eines Elements im Menü

zu manipulieren. So können ausgewählte Elemente durch eine Veränderung als ausgewähltes Element hervorgehoben werden. Diese Methodik findet sich auch in gängigen GUIs, bei denen beispielsweise gedrückte Knöpfe farblich abgesetzt erscheinen.

Ein solches Element des Menüs besteht also aus drei Eigenschaften und einer Funktion. Die drei Eigenschaften werden beim Erstellen des Elements über das Hauptmenü gesetzt. Ebenfalls wird die Funktion beim Wechsel des Status des Elementes hinterlegt. Innerhalb dieser Funktion können dann sowohl Aktionen, die das Element selbst betreffen, als auch Aktionen, die eine Änderung von Parametern oder ein Triggern von weiteren Funktionen in der Applikation hervorrufen, abgelegt werden. Die Kommunikation läuft dabei mit der Applikation immer über das Hauptmenü. Die Applikation erfährt also immer vom Hauptmenü, welches Element des Menüs aktuell eine Änderung seines Status erfahren hat.

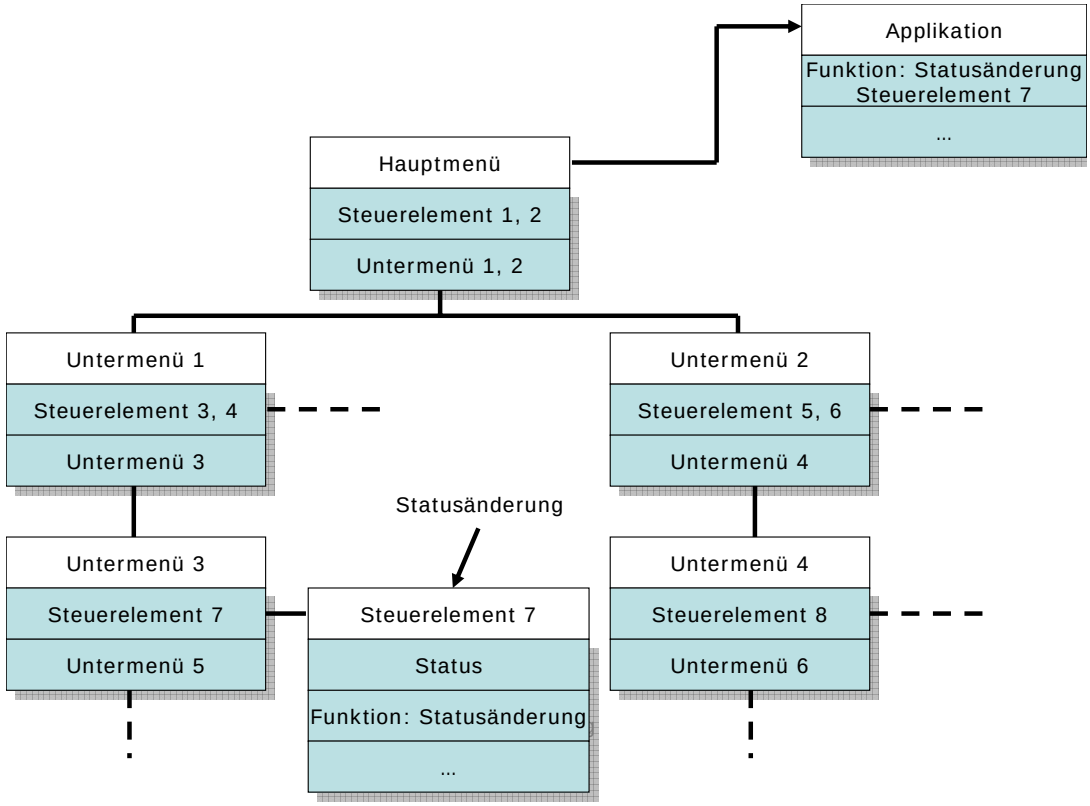


Abbildung 4.24: Exemplarischer Ablauf einer Statusänderung

Abb. 4.24 zeigt hierbei exemplarisch den Ablauf bei der Statusänderung von Steuerelement

7. Dabei wird die Änderung des Status im Steuerelement selbst vorgenommen und die zugehörige Funktion beim Statuswechsel aufgerufen. Hier könnte beispielsweise die Änderung der Farbe als visuelles Feedback vorgenommen werden. Gleichzeitig übermittelt das Hauptmenü die Änderung des Status von Steuerelement 7 an die angebundene Applikation, die wiederum eine Funktion aufruft, die Aktionen bei der Änderung des Status von Steuerelement 7 enthält.

Die Beschreibung eines solchen Menüs ist sehr abstrakt. In Kapitel 6.1.5 wird dieses Menü in die Beispielapplikation „BillARd“ integriert und mit einer grafischen Ausgabe versehen. Dieses Kapitel verdeutlicht die Möglichkeiten, die das hier vorgestellte System bietet.

KAPITEL 5

Alternative Interaktionsformen

Das folgende Kapitel behandelt Interaktionsmöglichkeiten ausserhalb der Mensch-Maschine-Interaktion. Darunter fallen die Interaktion zwischen realer und virtueller Umgebung und deren Objekte, die durch eine bildbasierte Kollisionserkennung umgesetzt wird, sowie die Interaktion zwischen Menschen, die durch ein Videokonferenzsystem ermöglicht wird, das in die AR der Nutzer eingebunden ist.

5.1 Kollisionserkennung zwischen Realität und Virtualität durch Bildverarbeitung

Für eine Interaktion mit virtuellen Objekten muss dem System sowohl die Position der virtuellen Objekte als auch die Position der realen Objekte, die mit diesem virtuellen Objekt interagieren sollen, bekannt sein. Dies kann mittels eines Trackingsystems erfasst werden. Zur Interaktion mit der Hand des Nutzers werden, wie in Kapitel 4.1 beschrieben, IR-Targets an der Hand des Nutzers befestigt, um daraus auf die Position der Hand im Raum rückschließen zu können. Sollen nun andere beliebige reale Objekte zur Interaktion mit virtuellen Objekten herangezogen werden, so müssen alle realen Objekte in ihrer geometrischen Form modelliert werden und mittels eines Trackingsystems verfolgt werden. Reale Objekte müssten also mit IR-Targets versehen werden, um Interaktion zwischen ihnen und virtuellen Objekten zu ermöglichen.

Aus diesem Grund behandelt dieses Kapitel einen Ansatz, der ohne IR-Trackingsystem auskommt, um reale beliebige Objekte in die AR-Szene zu integrieren und so eine Wechselwirkung zwischen ihnen und der AR-Szene zu ermöglichen. Dieser Ansatz wird in das bestehende Framework als Client-Server Ansatz integriert.

5.1.1 Thematisch verwandte Vorarbeiten

Die Vorarbeiten in diesem Themengebiet können in zwei Kategorien unterteilt werden. Zum einen die automatische Modellierung von 3-D-Objekten und deren Integration in die AR-Szene, zum anderen die Kollisionsdetektion zwischen den Objekten.

Im Bereich der automatischen Modellierung von realen Objekten bestehen bereits viele Ansätze, die in aktive und passive Ansätze unterteilt sind. Aktive Ansätze verwenden struk-

turiertes Licht, das auf ein Objekt ausgesendet wird und basierend auf dessen Abbildung auf die 3-D-Informationen des Objektes zurückgerechnet werden kann, wie etwa in [76] oder [51]. Passive Ansätze dagegen errechnen diese Informationen anhand von Aufnahmen mit normalen Kameras und nutzen dabei mehrere Ansichten des Objektes, beispielsweise in [60] oder [3]. Beide Ansätze liefern 3-D-Modelle von realen Objekten, die für die Verwendung in einer virtuellen Umgebung geeignet sind.

Um nur die Kontur eines 3-D-Objektes zu erhalten, bestehen ebenfalls Ansätze, die in den verschiedensten Disziplinen angesiedelt sind. So dient [26] der Konturerkennung von bewegten Objekten oder [72] der Erkennung von Handgesten durch Detektion derer Kontur.

Diese Daten sind Grundlage der Kollisionserkennung in einer virtuellen Umgebung, die bereits viele Lösungsansätze, zum Beispiel [41] oder [22], bereithält.

5.1.2 Problemstellung

Aufgabe der Kollisionserkennung ist es, reale Objekte in Echtzeit in die AR-Szene zu integrieren. Hierbei sollen keine instrumentierten Ansätze (vgl. Kapitel 4.1.1) zum Einsatz kommen, damit Objekte nicht mit zusätzlicher Hardware in Form von IR-Targets versehen werden müssen. Zusätzlich soll mit der Kollisionserkennung das Einbringen beliebiger, zuvor noch nicht modellierter Objekte möglich sein.

Aus diesen Gründen wurde in dieser Arbeit ein bildbasierter Ansatz verfolgt. Dabei werden Objekte mittels bildverarbeitender Ansätze aus ihrem Hintergrund segmentiert und automatisch modelliert. Die Aufgabe bestand also darin, sowohl Hardware als auch Software umzusetzen, die ein solches System ermöglichen. Die Verwendung von nur einer Kamera reduziert die Modellierung von realen Objekten auf deren Konturen, da hierbei keinerlei Tiefeninformationen über die Objekte gewonnen werden können. Ebenfalls beschränkt sich dieser Ansatz auf eine Modellierung, bei der die Texturierung der Objekte nicht beachtet wird. Die aus den Bilddaten gewonnenen Informationen dienen lediglich als Basis, um virtuelle Modelle zu erzeugen. Diese Modelle bilden die Grundlage zur Berechnung von Kollisionen zwischen virtuellen Objekten in der AR-Szene.

5.1.3 Umsetzung

Die Umsetzung der Kollisionserkennung basiert auf dem Ansatz, reale Objekte in einem Kamerabild zu lokalisieren. Dabei filmt eine Kamera eine Oberfläche, auf der reale Objekte eingebracht werden können. Mit Methoden der Bildverarbeitung können reale Objekte vor diesem Hintergrund segmentiert werden. Dazu wird eine Differenzbildanalyse durchgeführt, die ein s/w Bild liefert, auf dem der Hintergrund in schwarz und das segmentierte reale Objekt in weiß abgebildet ist. Am Ende der Bildverarbeitung steht ein Bild, auf dem die Kontur des realen Objektes vom Hintergrund abgegrenzt ist. Aus dieser Kontur lassen sich deren Eckpunkte bestimmen, die das Objekt beschreiben und aus denen sich ein virtuelles Modell des realen Objektes erstellen lässt. Das so erzeugte Modell kann in die AR-Szene eingebracht werden. Dafür wird die Kamera mit dem IR-Trackingsystem erfasst und aufgrund deren Position und Lage im Raum wird auf die Position des realen Objektes im Raum geschlossen. An dieser Position kann nun das virtuelle Modell des Objektes eingeblendet werden, oder aber auch als unsichtbares virtuelles Objekt zur Kollisionsdetektion zwischen virtuellen Objekten herangezogen werden. Auf diese Weise erreicht man eine Wechselwirkung zwischen realen und virtuellen Objekten. Die detaillierte Umsetzung ist im Folgenden beschrieben.

5.1.3.1 Übersicht

Der physikalische Aufbau des Kollisionserkenners besteht aus einem Tisch, einer Kamera, einem Gestänge zur Befestigung der Kamera und einem aktiven IR-Target (vgl. Abb. 5.1).

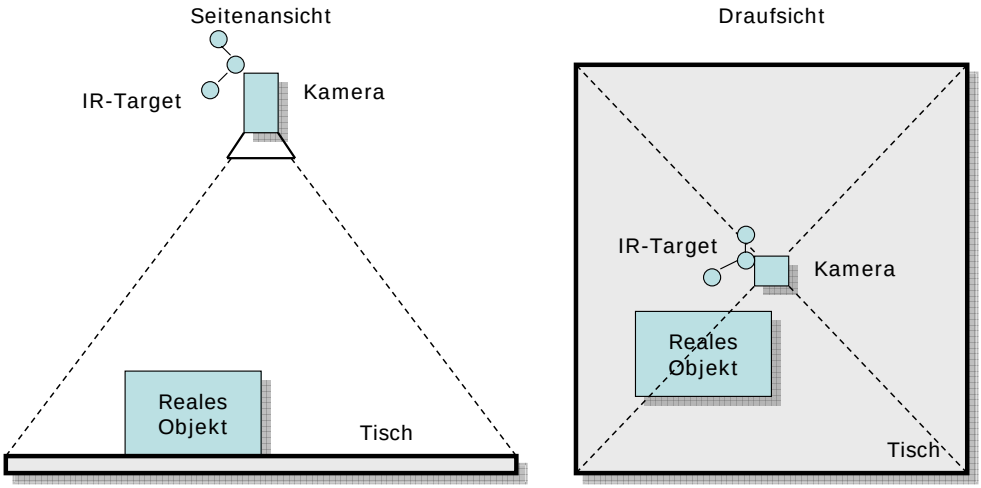


Abbildung 5.1: Ansichten des Aufbaus des Kollisionserkenners

Der Tisch hat eine Oberfläche von etwa einem Meter auf einen Meter und besitzt eine hellgraue Färbung. Als Kamera kommt eine „Quickcam Communicate STX“ von „Logitech“ zum Einsatz. Diese liefert eine Auflösung von 640 x 480 Bildpunkten bei einer Bildwiederholungsrate von 30 Frames je Sekunde. Mithilfe des Gestänges ist sie so auf den Tisch montiert, dass sie etwa einen Meter oberhalb des Tisches senkrecht zur Oberfläche des Tisches nach unten blickt. Die Befestigung ist dabei sehr stabil, um eine gleichbleibende Position relativ zum Tisch zu gewährleisten. Dies ist notwendig, da an der Kamera ein IR-Target befestigt ist, das die Position und Orientierung der Kamera im Raum liefert. Aus diesen Daten wird später die Position der Tischoberfläche und somit die Position des Objektes auf dem Tisch in der Welt berechnet. Da am Tisch Rollen angebracht sind, kann dieser frei im Raum bewegt werden. Durch das Tracking der Kamera kann so an jeder beliebigen Position im Raum ein reales Objekt auf der Tischoberfläche in die AR Szene eingebracht werden.

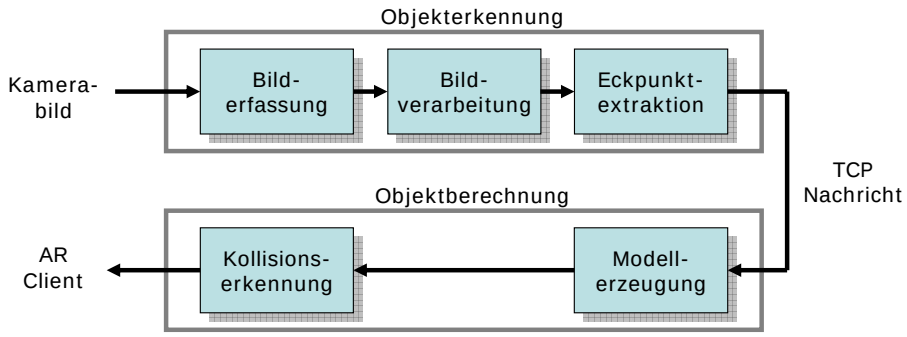


Abbildung 5.2: Schematischer Aufbau der Kollisionserkennung

Die Software besteht aus zwei Teilen (vgl. Abb. 5.2). Die Objekterkennung nimmt das Bild der Kamera auf, verarbeitet es und extrahiert daraus die Eckpunkte des realen Objektes.

Diese Daten werden über eine Netzwerkverbindung an die Objektberechnung weitergeleitet. Daraus wird ein virtuelles Modell der Kontur des Objektes erzeugt, das dann in die AR-Szene eingebracht werden kann. Hierbei findet gleichzeitig eine Kollisionserkennung mit anderen virtuellen Objekten innerhalb der AR-Szene statt.

5.1.3.2 Objekterkennung

Die gesamte Objekterkennung ist wegen der höheren Performance in C++ geschrieben. Weiterhin wurde das bereits bestehende Framework „OpenCV“ (Open Computer Vision) von „Intel“ [15] verwendet, das grundlegende Algorithmen zur Bildverarbeitung bereitstellt und auf deren Performance optimiert ist.

Die Objekterkennung untergliedert sich in das Einlesen des Kamerabildes, die Bildverarbeitung, in der das eingelesene Bild bearbeitet wird, und das Finden von Objekten im gefilterten Bild. Aus diesem werden dann die Eckpunkte der Objekte extrahiert und für die anschließende weitere Verarbeitung in ein eigenes Datenformat konvertiert.

Mit „OpenCV“ kann durch die Verwendung von Herstellertreibern auf die über USB angeschlossene Kamera zugegriffen werden. „OpenCV“ übergibt dieses Kamerabild stets im selben Format für alle unterstützten Kameratypen und gewährleistet somit hohe Flexibilität.

Um nun Objekte im Bild vom Hintergrund (hellgraue Tischoberfläche) zu trennen, muss eine Filterung des Bildes durchgeführt werden. Aufgrund wechselnder Lichtverhältnisse, die in Änderungen des Farbtons resultieren, ist ein Filter, der auf Farbtöne beschränkt ist, nicht ausreichend. Ebenfalls könnten mittels eines solchen Filters keine Schatten oder Reflexionen der Umgebung herausgefiltert werden. Da die Oberfläche des Tisches nur Grautöne mit geringer Farbsättigung aufweist, kommt in dieser Arbeit ein Filter zum Einsatz, der nach Farbsättigung und Helligkeit filtert.

Das eingelesene Kamerabild $\mathcal{B}_{RGB}$ liegt im RGB Farbraum vor und hat die Dimension von $n \times m$, $n, m \in \mathbb{N}$ (vgl. Abb. 5.3 (a)). Aufgrund der Beschreibung im additiven Farbraum ist das Filtern nach Farbsättigung und Helligkeit in diesem Datenformat nicht möglich. Aus diesem Grund wird es in den HSV-Farbraum transformiert. Dieser Farbraum beschreibt Farben mittels des Farbtons H (Hue), der Farbsättigung S (Saturation) und der Dunkelstufe V (Value). Dabei wird der Farbton auf einem Kreis beschrieben, wobei jedem Winkel ein Farbton zugeordnet ist. Sättigung und Dunkelstufe dagegen werden in Prozent angegeben.

Das Bild $\mathcal{B}_{RGB}$ im RGB Farbraum kann dann in den HSV-Farbraum als $\mathcal{B}_{HSV}$ überführt werden. Hierzu werden folgende Hilfsfunktionen benötigt:

$$\min_{RGB}(n1, n2) := \min_{k \in \{0,1,2\}} (\mathcal{B}_{RGB}(n1, n2, k))$$

$$\max_{RGB}(n1, n2) := \max_{k \in \{0,1,2\}} (\mathcal{B}_{RGB}(n1, n2, k))$$

$$\operatorname{diff}_{RGB}(n1, n2) := \max_{RGB}(n1, n2) - \min_{RGB}(n1, n2)$$

Dann lassen sich die Bilder der einzelnen HSV-Kanäle $\mathcal{B}_H$, $\mathcal{B}_S$ sowie $\mathcal{B}_V$ wie folgt definieren:

$$\mathcal{B}_{H_t}(n1, n2) := \begin{cases} nicht\ definiert & \text{falls } \operatorname{diff}_{RGB}(n1, n2) = 0 \\ (\frac{\mathcal{B}_G(n1,n2) - \mathcal{B}_B(n1,n2)}{\operatorname{diff}_{RGB}(n1,n2)}) \cdot 60 & \text{falls } \mathcal{B}_R(n1, n2) = \max_{RGB}(n1, n2) \\ (\frac{\mathcal{B}_B(n1,n2) - \mathcal{B}_R(n1,n2)}{\operatorname{diff}_{RGB}(n1,n2)}) \cdot 60 & \text{falls } \mathcal{B}_G(n1, n2) = \max_{RGB}(n1, n2) \\ (\frac{\mathcal{B}_R(n1,n2) - \mathcal{B}_G(n1,n2)}{\operatorname{diff}_{RGB}(n1,n2)}) \cdot 60 & \text{falls } \mathcal{B}_B(n1, n2) = \max_{RGB}(n1, n2) \end{cases}$$

$$\mathcal{B}_H(n1, n2) := \mathcal{B}_{H_t}(n1, n2) \bmod 360$$

$$\mathcal{B}_S(n1,n2) := \begin{cases} 0 & \text{falls } \max\limits_{RGB}(n1,n2) = 0 \\ 100 \cdot \frac{\text{diff}_{RGB}(n1,n2)}{\max_{RGB}(n1,n2)} & \text{sonst} \end{cases}$$

$$\mathcal{B}_V(n1,n2) := \max_{RGB}(n1,n2) \cdot 100$$

Das Bild $\mathcal{B}_{HSV} : \{1,2,\ldots,n\} \times \{1,2,\ldots,m\} \times \{0,1,2\} \to [0;360[$ lässt sich somit definieren als

$$\mathcal{B}_{HSV}(n1,n2,k) := \begin{cases} \mathcal{B}_H(n1,n2) & \text{falls } k = 0 \\ \mathcal{B}_S(n1,n2) & \text{falls } k = 1 \\ \mathcal{B}_V(n1,n2) & \text{falls } k = 2 \end{cases}$$

Durch Versuche mit dem Systemaufbau konnten die Schwellwerte von V und S zu

$$V_{min} = 27,5\% \text{ und } S_{max} = 27,5\%$$

bestimmt werden. Hierbei bezeichnet V_{min} den minimalen Helligkeitswert und S_{max} die maximale Sättigung. Zum Tisch gehören somit alle Bildpunkte, welche V_{min} nicht unter- und gleichzeitig S_{max} nicht überschreiten. Bildpunkte, für die

$$V < V_{min} \text{ oder } S > S_{max}$$

gilt, gehören folglich nicht zum Hintergrund, sondern werden als Teil eines Objektes betrachtet. Die Schwellwertfunktion

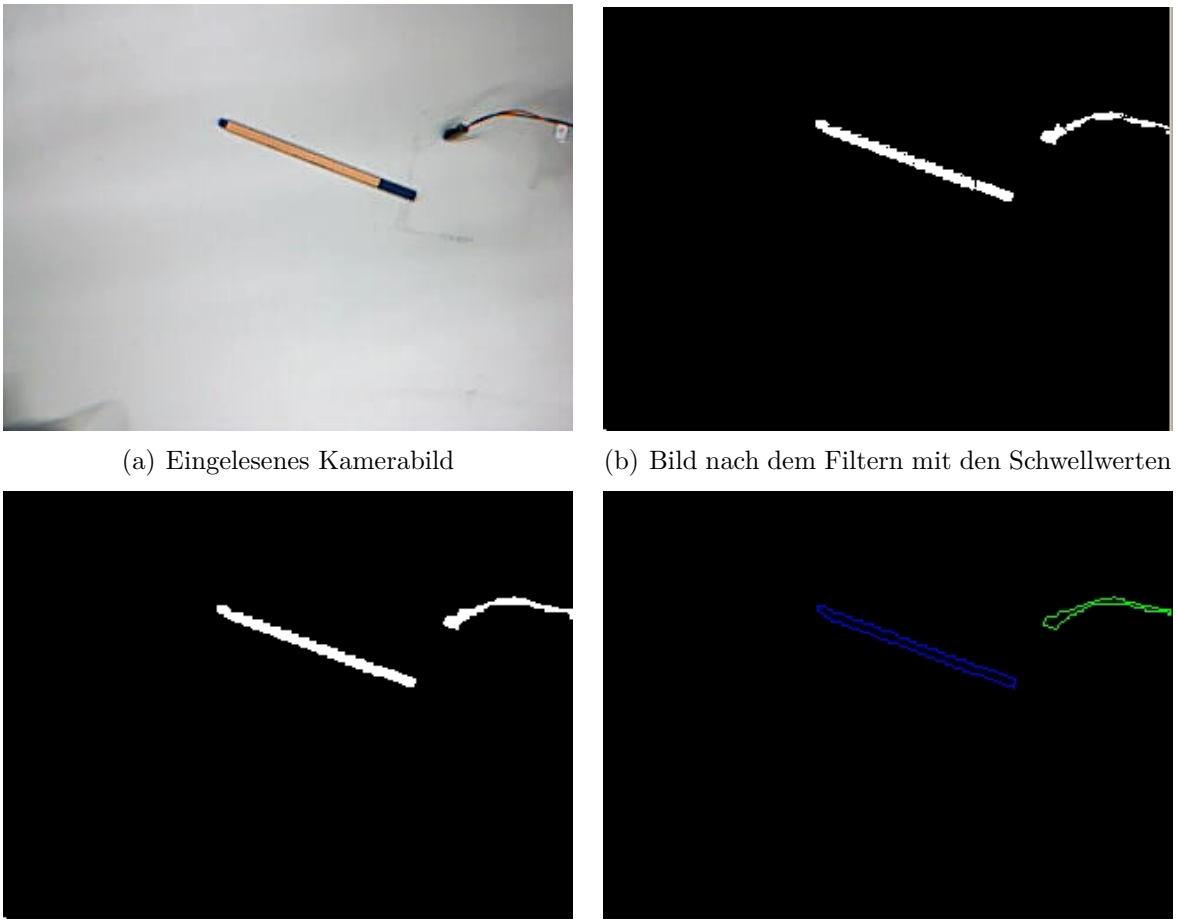
$$s : \mathcal{B}_{HSV} \to \{1,2,\ldots,n\} \times \{1,2,\ldots,m\}$$

welche die Menge aller Punkte zurückliefert, die zur Gruppe der Objekte und somit nicht zum Tisch gehören, ist daher wie folgt definiert:

$$\begin{aligned} s(\mathcal{B}_{HSV}) :=& \{p \in \{1,2,\ldots,n\} \times \{1,2,\ldots,m\} \mid \\ & \mathcal{B}_S(p) > S_{max} \vee \mathcal{B}_V(p) < V_{min}\} \end{aligned}$$

Um auch bei färbenden Beleuchtungen den Tisch als Hintergrund zu erkennen, wurde der Schwellwert für die Sättigung S relativ hoch angesetzt. Untersuchungen im Labor haben gezeigt, dass die Sättigung bei annähernd weißem Licht nicht über 15% steigt und somit die Erkennungsraten bei farbigem Licht erhöhen, ohne dabei die Raten bei weißem Licht zu verringern. Der Schwellwert für die Dunkelstufe V wurde durch Tests so bestimmt, dass der dunkelste Schatten bei heller Beleuchtung als Tisch identifiziert wird. Dadurch können sehr dunkle Objekte (vgl. Abb. 5.3 (a)) dennoch vom Tisch unterschieden werden. Die Versuche zeigten, dass die Wahl der Schwellwerte den Filter sehr robust machten, wenngleich helle oder reflektierende Objekte nicht vom Tisch unterscheidbar sind.

Beim Filtern des Kamerabildes können kleine Lücken oder Löcher in den Objekten entstehen. Durch Lichtreflexionen oder durch dünne, helle Linien oder Punkte in der Struktur eines Objektes können solche Punkte fälschlicherweise dem Hintergrund zugeordnet werden. Dies



(a) Eingelesenes Kamerabild

(b) Bild nach dem Filtern mit den Schwellwerten

(c) Bild nach dem Schließen von Löchern

(d) Konturen von Objekten im Bild

Abbildung 5.3: Schritte der Objekterkennung

kann dazu führen, dass ein einzelnes Objekt (Stift) in mehrere kleinere Objekte (Stift und Kappe) zerfällt (vgl. Abb. 5.3 (b)). Daher müssen diese Objekte wieder zu ganzen Objekten zusammengeführt werden.

Da diese Löcher im Allgemeinen klein sind, kann hier die morphologische Operation „Schließen“ mit einem quadratischen Strukturelement der Größe 5 mal 5 Pixel angewandt werden. Dabei setzt sich die Operation aus den beiden Operationen „Dilatation“ und „Erosion“ zusammen. Bei der Dilatation wird um das Zentrum des Strukturelements herum verglichen, ob mindestens eines der benachbarten Pixel zum Objekt gehört. In diesem Fall wird das Zentrum als Teil des Objektes gewertet. Die Erosion dagegen vergleicht um das Zentrum herum, ob jedes der benachbarten Pixel zum Objekt gehört. In diesem Fall werden nur diese Pixel dem Objekt zugeordnet. Das „Schließen“ ermöglicht damit, kleine Lücken zwischen zwei Pixelgruppen zu schließen, ohne dabei deren Form maßgeblich zu verändern. Die kleine Größe des gewählten Strukturelements gewährleistet, dass nicht zusammenhängende Objekte verschmelzen. Somit werden Objekte, die in kleinere Teile zerfallen sind, wieder zu einem Objekt zusammengeführt (vgl. Abb. 5.3 (c)).

Der somit mehrstufige Filter $F(\mathcal{B}_{RGB})$ lässt sich mathematisch für ein Bild im RGB-Farbraum $\mathcal{B}_{RGB}$ und ein Strukturelement $\mathbf{S}$ wie folgt formulieren. Dabei bezeichnet s die Schwellwertfunktion und $\mathcal{B}_{HSV}(\mathcal{B}_{RGB})$ die Bildfunktion $\mathcal{B}_{HSV}$ in Abhängigkeit von $\mathcal{B}_{RGB}$

$$F(\mathcal{B}_{RGB}) := s(\mathcal{B}_{HSV}(\mathcal{B}_{RGB})) \bullet \mathbf{S}$$

Innerhalb des vorliegenden Bildes können nun verschiedene Objekte gleichzeitig vorhanden sein. Um nun die Objekte zu unterscheiden, müssen die Konturen der einzelnen Objekte ermittelt werden. Dazu ist in „OpenCV“ ein Algorithmus [81] implementiert, der die äußere Kontur einer Pixelgruppe findet und als geordnete Liste von 2-D-Koordinaten im Bild zurückliefert. Dieser Algorithmus kann mehrere Objekte unterscheiden, für die weitere Verarbeitung wird aber nur das jeweils erste Objekt der Liste betrachtet, alle weiteren Objekte werden verworfen. In Abb. 5.3 (d) ist das Ergebnis der Konturensuche sichtbar, die äußeren Konturen von zwei Objekte werden farblich hervorgehoben.

Um nun die Objekte in ein virtuelles Modell umzuwandeln, müssen im letzten Schritt noch die Koordinaten, angegeben in Pixeln, in Millimeter umgerechnet werden. Dazu wird ein konstanter Skalierungsfaktor berechnet, der durch eine initiale Kalibrierung eine vorgegebene Länge in Millimetern in Pixeln abbildet. Nach der Bestimmung des Skalierungsfaktors kann die Größe eines Objektes berechnet werden, wobei die Skalierungsfunktion $f_s : \mathbb{N}^2 \rightarrow \mathbb{R}^2$ wie folgt definiert sei:

$$f_s(x, y) = s \cdot f_(x, y) \qquad s \in \mathbb{R}$$

Des Weiteren sei für eine erkannte Kontur K und einen Punkt $P_i := (x_i, y_i) \in K$ $\bar{P}_i$ definiert als

$$\bar{P}_i := f_s(P_i)$$

Nach der Skalierung aller Punkte der Kontur K geht diese in $\bar{K}$ über:

$$\bar{K} = \bigcup_i \{\bar{P}_i\}$$

Nun liegt das Objekt als Anordnung von Eckpunkten vor, die als 2-D-Koordinaten in Millimetern angegeben sind. Um es über die Netzwerkverbindung zur Objektberechnung zu senden, muss es zuvor noch in ein 1-D-Feld gewandelt werden, das wie folgt definiert ist:

$$[\bar{x}_1, \bar{y}_1, \bar{x}_2, \bar{y}_2, ..., \bar{x}_n, \bar{y}_n] \qquad (\bar{x}_i, \bar{y}_i) \in \bar{P}_i \in \bar{K}$$

Aufgrund der beschriebenen Umsetzung können zwei Sorten von Objekten nicht oder nur falsch erkannt werden. Es handelt sich hierbei um sehr dünne Objekte (vgl. Abb. 5.4) und um solche, die einen Hohlraum in ihrer inneren Struktur aufweisen, zum Beispiel ein Ring (vgl. Abb. 5.5).

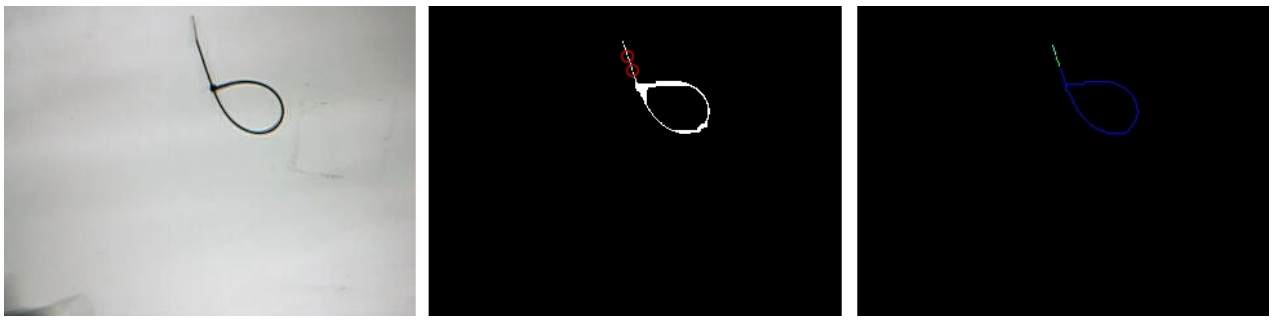


Abbildung 5.4: Dünnes Objekt bei der Objekterkennung

Sehr dünne Objekte werden bei der Filterung aufgrund der begrenzten Auflösung der Webcam oft in mehrere kleine Teilobjekte zerfallen oder gänzlich durch die morphologischen Operationen gelöscht.

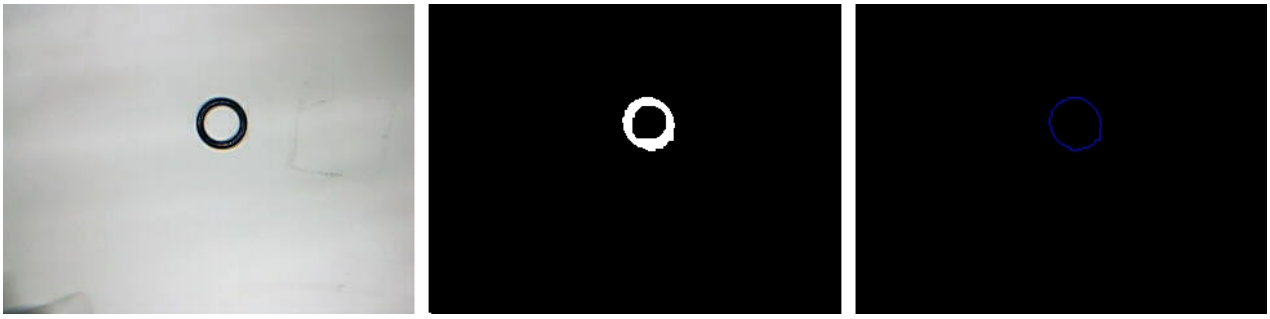


Abbildung 5.5: Ring bei der Objekterkennung

Die Verwendung des Algorithmus [81] bringt mit sich, dass nur äußere Konturen erkannt werden. Der Algorithmus ist nicht in der Lage, innere Konturen von Objekten zu erkennen, wodurch sie ohne ihre inneren Löcher modelliert werden.

5.1.3.3 Objektverarbeitung

Die Objektverarbeitung übernimmt die Modellierung eines virtuellen Objektes aus den berechneten Koordinaten seiner Eckpunkte sowie die Kollisionserkennung zwischen diesem Modell und anderen virtuellen Objekten (vgl. Abb. 5.6(a)), die sich innerhalb der AR-Szene befinden.

Die Koordinaten der Eckpunkte liegen bisher nur als 2-D-Koordinaten vor, da diese aus nur einem Bild extrahiert werden konnten. AR benötigt aber Modelle, die 3-D-Koordinaten aufweisen. Daher müssen alle 2-D-Koordinaten um eine feste Höhe ergänzt werden. Zur Modellierung in AR wird die Beschreibungssprache „Virtual Reality Modelling Language“ (VRML) verwendet. In dieser kann mit einem „IndexedFaceSets“ (IFS) ein Objekt durch seine Eckpunkte definiert werden. Dazu werden in zwei Listen zum einen die Koordinaten der Eckpunkte aufgeführt und zum anderen die daraus zu bildenden Polygone. Die Objektverarbeitung wandelt also die vorliegenden Koordinaten der Eckpunkte von 2-D in 3-D. Dies geschieht, indem bei einem Koordinatensatz die Höhe zum Wert 0 gesetzt wird $(x, y, 0)$ und in einem weiteren Koordinatensatz eine zuvor festgelegte Höhe h hinzugefügt wird und somit aus der 2-D-Kontur (x, y) eine 3-D-Kontur erstellt wird (x, y, h). Somit liegen nun die Koordinaten für die Kontur in der X-Y-Ebene vor sowie parallel dazu um die Höhe h versetzt.

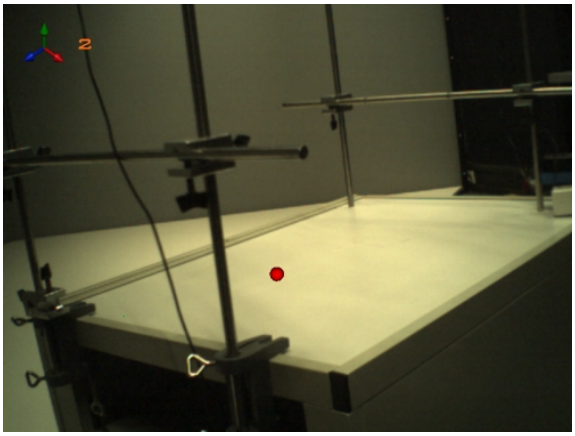
Um diese 3-D-Kontur nun in AR verwenden zu können, wird ein zunächst leeres IFS in VRML erstellt. In diesem befinden sich in den beiden Listen noch keine Elemente. Mithilfe eines Algorithmus werden nun die 3-D-Koordinaten in Echtzeit in die Liste des leeren IFS geschrieben. Anschließend werden aus den Koordinaten Flächen erzeugt und in die zweite Liste geschrieben. Dabei werden zum einen die Flächen der Konturen verbunden, was zu zwei parallelen Konturen führt, zum anderen aber auch die Seitenflächen zwischen den beiden Konturen. Auf diese Weise entsteht ein Modell in 3-D, das die ursprüngliche Kontur des Objektes als Pseudo 3-D-Modell abbildet.

Im weiteren Verlauf werden basierend auf den aus dem aktuellen Kamerabild gewonnenen Eckpunkten des Objektes die beiden Listen des IFSs laufend neu beschrieben. Das virtuelle Modell als IFS passt sich also laufend der aktuellen Kontur des Objektes an.

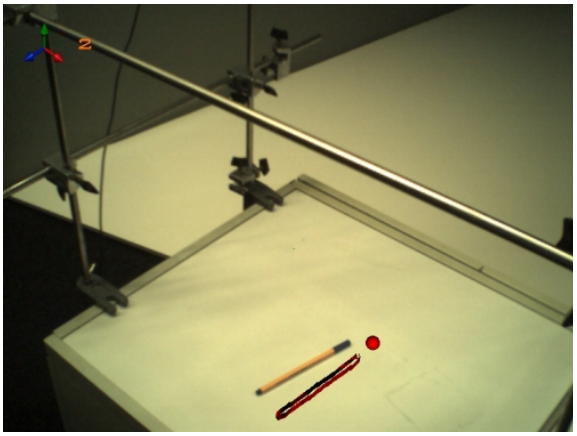
Damit das Objekt nun in der AR-Szene sichtbar wird (vgl. Abb. 5.6(a)), sendet die Objektverarbeitung das Modell an den AR-Client, der die aktuelle AR-Szene verwaltet, sobald das Objekt im Sichtbereich der Kamera erscheint. Um die Performance zu erhöhen, wird nicht in jedem Schritt das vorherige IFS aus der Szene gelöscht und das neue IFS geladen, sondern

es werden zur Laufzeit nur die Einträge der Liste verändert. Auf diese Weise müssen die Objekte nicht laufend gelöscht und neu geladen werden, sondern können ausgehend vom leeren Modell direkt verändert werden. Diese Methodik steigert die Darstellungsrate von 0,5 Frames je Sekunde auf etwa 10 Frames je Sekunde.

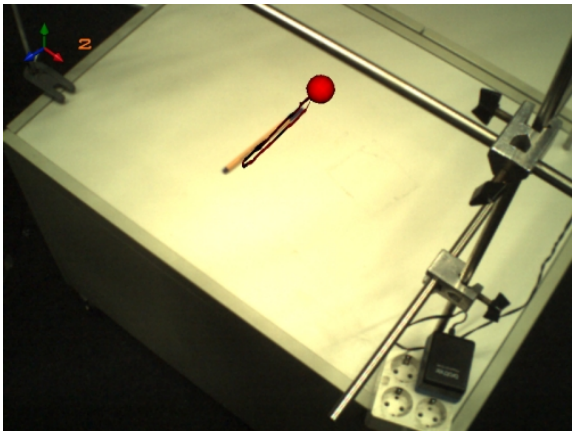
Um eine Überlagerung von realem Objekt und virtuellem Modell zu erreichen, muss die Position in der AR-Szene mit der realen Position innerhalb des WeltKOS übereinstimmen. Andernfalls würde das erzeugte Modell nicht deckungsgleich mit dem realen Objekt erscheinen (vgl. Abb. 5.6(b)). Mittels des IR-Targets an der Kamera wird ein TargetKOS erzeugt, das seinen Ursprung genau auf der Ebene des Tisches hat und dessen X-Y Ebene parallel mit der Oberfläche des Tisches ist. Die Positionierung wird also bereits beim Tracking berücksichtigt und muss nicht separat von der Objektverarbeitung ermittelt werden, da das virtuelle Objekt direkt an das TargetKOS gebunden werden kann und somit auf dem Tisch erscheint.



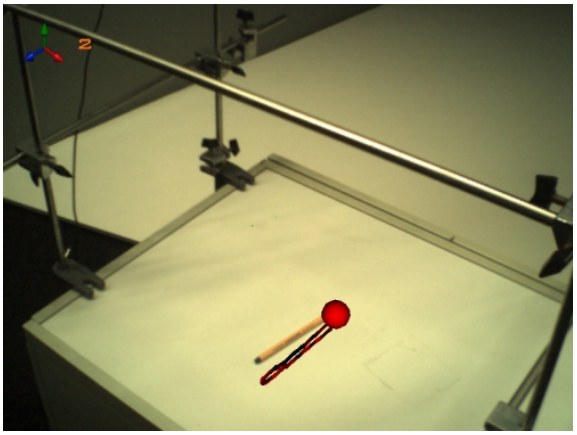
(a) Nur eine virtuelle Kugel ist auf dem Tisch in die AR Szene geladen. Kein realer Stift ist sichtbar und somit auch nicht als virtuelles Modell vorhanden



(b) Ein realer Stift wird in das Sichtfeld der Kamera eingebracht und sofort als virtuelles Modell in die Szene eingefügt



(c) Der reale Stift berührt die virtuelle Kugel und verursacht dadurch eine Größenänderung der virtuellen Kugel



(d) Die Kollision zwischen realem Stift und virtueller Kugel aus einem anderen Blickwinkel

Abbildung 5.6: Modellierung von realen Objekten und deren Kollisionen mit virtuellen Objekten in Augmented Reality

Um nun Kollisionen zwischen virtuellen Objekten innerhalb einer AR-Szene zu erkennen (vgl. Abb 5.6(c) und (d)), müsste jeder einzelne Punkt mit allen anderen Punkten der Szene

hinsichtlich seiner Position im Raum verglichen werden. Aus diesem Grund vereinfacht die Objekterkennung die Detektion von Kollisionen zwischen Objekten. Zum einen werden die Objekte zur Kollisionsdetektion auf den 2-D-Fall reduziert, wodurch die Anzahl der zu prüfenden Punkte reduziert wird. Zum anderen werden die Kollisionsobjekte stark vereinfacht. In dieser Modellierung kommen zur Kollision mit realen Objekten nur virtuelle Kugeln zum Einsatz (dies jedoch könnte zu Lasten der Performance auch abgeschaltet werden).

Da die realen Objekte auf der Oberfläche des Tisches liegen und sich die in dieser Arbeit vorgesehene Verwendung der Kollisionserkennung (vgl. Kapitel 6.1) mit virtuellen Objekten auf dieser Oberfläche beschränkt, kann die Detektion von Kollisionen auf eine Projektion in die Ebene der Tischoberfläche reduziert werden. Diese Einschränkung setzt jedoch voraus, dass reale und virtuelle Objekte in ihrer Höhe beschränkt sind, da sonst Kollisionen fälschlich detektiert werden können, sobald die Projektion eines hohen Punktes sich in der Ebene mit einem niedrigen Punkt schneidet. Die Projektion der virtuellen Kugeln wird vereinfacht mit Kreisen auf der Tischoberfläche vorgenommen (Radius und X-/Y-Position gleichbleibend). Damit ist eine Kollisionserkennung eine einfache 2-D-Abstandsanalyse der Konturpunkte des erkannten Objektes mit den Mittelpunkten der Kugeln. Eine Kollision liegt vor, wenn der errechnete Abstand den entsprechenden Kreisradius unterschreitet (vgl. Abb 5.6(c) und (d)).

5.2 Interpersonelle Interaktion durch ein Videokonferenzsystem

Das in Kapitel 3.1 vorgestellte AR-Framework bietet bereits die Möglichkeit, in einer AR-Szene sowohl lokal als auch räumlich getrennt gemeinsam zu arbeiten. Für die gemeinschaftliche Arbeit (Kollaboration) ist es jedoch erforderlich, andere Nutzer, die räumlich getrennt sind, in die AR-Szene zu integrieren. Dies erleichtert die Kommunikation, die die Kollaboration beispielsweise durch Gesten unterstützt. Ebenfalls erweist es sich als hilfreich, Informationen von Applikationen, die nicht innerhalb der AR verfügbar sind, einzubinden. Hierzu zählt beispielsweise das Grafische User Interface (GUI) eines CAD-Programms. Aus diesem Grund wird das Framework um ein System zur Erzeugung einer Videokonferenz erweitert.

5.2.1 Thematisch verwandte Vorarbeiten

Die Integration einer Videokonferenz in die Virtual Reality beschreibt [73]. Hier werden sowohl virtuelle Objekte als auch die Gesprächspartner per Videokonferenz in eine Szene eingebunden, die allen Teilnehmern der Videokonferenz zugänglich ist. Die Teilnehmer sind hierbei jedoch auf die Darstellung mit einem Monitor gebunden, jegliche Überlagerung von realen und virtuellen Inhalten ist nicht möglich.

Einen anderen Ansatz verfolgt [6]. Das hier entwickelte System stellt sowohl das eigene spiegelbildliche Abbild als auch das Abbild des Gegenübers gemeinsam auf einem Bildschirm dar . Beide Teilnehmer können virtuelle Objekte laden, die an Papiermarker gebunden werden. Manipulationen an den Objekten werden stets bei beiden Teilnehmern synchronisiert. Diese Vorgehensweise erlaubt es zwar, virtuelle Objekte in die tatsächliche Umgebung zu integrieren, fesselt den Benutzer aber immer noch an den Monitor. Das System ist außerdem auf zwei Teilnehmer beschränkt und die Darstellung beider Teilnehmer auf dem Bildschirm kann nicht als ideal erachtet werden, da das eigene Spiegelbild zu Sehen ungewohnt ist und eher irritiert als nützt.

[40] verwendet bei dieser Umsetzung eines Videokonferenzsystems neben einem Monitor auch ein HMD. In diesem System ist der Einsatz des HMDs auf die Verwendung von einem einzelnen Teilnehmer beschränkt. Die anderen Gesprächspartner sitzen hingegen vor einem Monitor und werden von einer Kamera gefilmt. Aus diesen Aufnahmen werden virtuelle Objekte erstellt, die der Träger des HMDs an Papiermarker binden kann. Somit ist es gelungen, die Abbilder der entfernten Teilnehmer in die reale Umgebung zu integrieren. Leider ermöglicht dieses System nur einer Person, seine Gesprächspartner zu sehen. Die Personen vor den Monitoren müssen auf eine Darstellung ihrer Gesprächspartner verzichten. In Weiterentwick-

(a) Teilnehmer mit Head Mounted Display

(b) Sicht des Nutzers mit dem Head Mounted Display

Abbildung 5.7: Videokonferenzsystem nach [40]

lungen dieser Arbeit wurde dem Bestreben nachgegangen, die Darstellung der Personen noch realitätsnäher zu gestalten und die Präsenz im Raum zu steigern. Mögliche Ansätze dazu sind der Übergang von einer 2-D- in eine 3-D-Darstellung der Teilnehmer oder die Elimination des Hintergrunds der Konferenzteilnehmer.

Alle Ansätze verbinden eine bestimmte Kombination von Geräten, sind also auf eine einzelne definierte Systemkonfiguration festgelegt. Daher soll das zu entwickelnde Videokonferenzsystem die Möglichkeit bieten, verschiedene Ansätze zur Verfügung zu stellen, um in einer späteren Evaluierung (vgl. Kapitel 7.5) diese Konfigurationen einander gegenüberstellen zu können, um somit Vor- und Nachteile einzelner Systemaufbauten aufzudecken.

5.2.2 Problemstellung

Die zur Modellierung realer Objekte zugrunde liegende Beschreibungssprache VRML erlaubt die Texturierung von Oberflächen mit zuvor aufgenommenen Media Daten (Offline Daten), wie zum Beispiel Bilder oder Videos. Diese Texturen werden als Materialeigenschaft des Objektes definiert und mit ihm verknüpft. Media Daten, die in Echtzeit generiert werden (Online Daten), können jedoch nicht eingebunden werden. Es fehlt also die Integrationsmöglichkeit von Videostreams. Videokonferenzen werden jedoch immer in Echtzeit gehalten, stellen also Informationen in Form von Online Daten zur Verfügung. Aus diesem Grund soll das Videokonferenzsystem dem AR-System Online Daten in einer Form zur Verfügung stellen, die die Darstellung dieser Media Daten in Echtzeit ermöglicht. Um die Realitätsnähe eines Gesprächspartners zu erhöhen, muss das Modul innerhalb eines Bildes einer angeschlossenen Kamera den Hintergrund vom Teilnehmer segmentieren. Auf diese Weise soll der Hintergrund elimi-

niert werden, sodass nur noch der Teilnehmer der Videokonferenz innerhalb der AR-Szene sichtbar ist.

5.2.3 Umsetzung

VRML bietet die Möglichkeit, Materialeigenschaften von Modellen zu definieren. Zu diesen Eigenschaften zählt unter anderem die Textur einer Oberfläche. Mit VRML lassen sich also Objekte erstellen, deren Oberfläche mit Offline Daten texturiert werden können. „Unifeye“ ist in der Lage, spezifische Eigenschaften von Modellen zur Laufzeit zu ändern, ohne dass dadurch ein erneutes Nachladen des manipulierten Objektes notwendig wird. Aus diesem Grund ist es möglich, die Textur eines Modells zur Laufzeit zu verändern.

Durch eine kontinuierliche Veränderung der Textur mittels des „Unifeye“ kann also der Eindruck von Online Daten erzeugt werden. Bei ausreichender Aktualisierungsrate der Oberflächentextur des Modells (etwa 24 Frames je Sekunde) lässt sich ein scheinbar flüssiger Ablauf umsetzen. Dazu generiert ein Server (vgl. Abb. 5.8) ein Standbild entweder von einer angeschlossenen Kamera oder von einem definierten Ausschnitt des Bildschirms. Dieses Standbild wird anschließend auf einem Fileserver abgelegt, der sowohl vom Server als auch vom Client erreichbar ist. Nach dem Speichern des Bildes wird der Client benachrichtigt, dass ein aktualisiertes Standbild verfügbar ist. Dieser lädt das Bild daraufhin vom Fileserver und aktualisiert die Textur des entsprechenden Modells.

Durch eine Unterscheidung der angeschlossenen Server ist es möglich, mehrere Datenquellen innerhalb einer AR-Szene darzustellen.

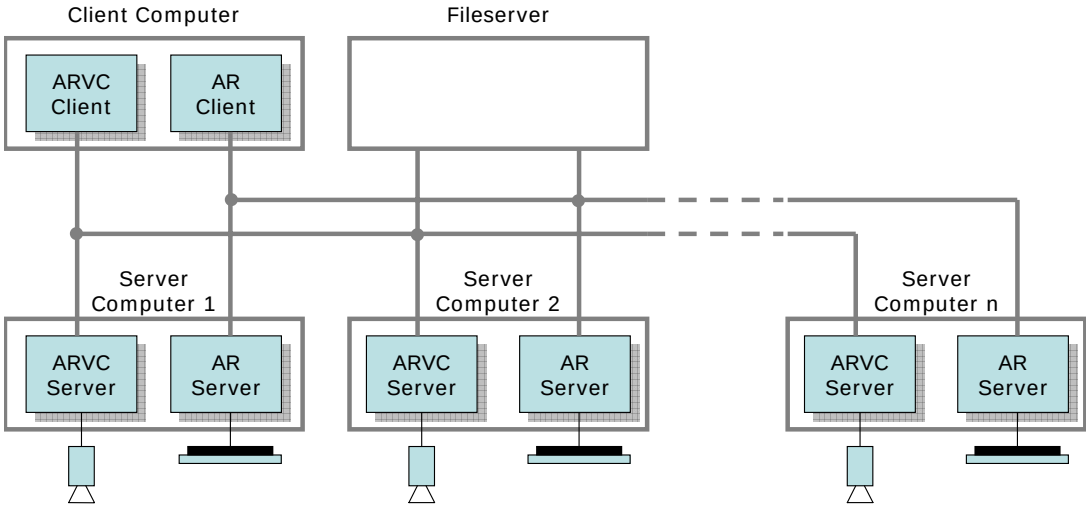


Abbildung 5.8: Schematischer Aufbau des Videokonferenzsystem

5.2.3.1 Übersicht

Um eine einfache Einbindung in verschiedenen AR-Applikationen zu ermöglichen, ist der Client als Klasse umgesetzt worden, der durch die Applikation mithilfe weniger Funktionen bedienbar ist (vgl. Abb. 5.9). Beim Erstellen einer neuen Videokonferenz wird dem Client die aktuelle AR-Szene übergeben. Diese ist für die gesamte Applikation und deren Module gleich, sodass Manipulationen jedes einzelnen Modules bezüglich der AR-Szene innerhalb jedes Modules abgefragt werden können. So kann die Videokonferenz mit anderen Modulen

kombiniert werden, wodurch sie sich zeitgleich innerhalb derselben AR-Szene befindet. Server werden durch einen einfachen Funktionsaufruf durch den Client erstellt. Dabei wird zunächst ein in VRML beschriebenes Modell in Form eines flachen Quaders geladen. Die Materialeigenschaften bezüglich der Größe und Textur dieses Quaders sind fest definiert, sodass sie im Verlauf durch den Client verändert werden können. Beim Erstellen eines neuen Servers wird zusätzlich der Ort des Servers in Form des Zielrechners und des UDP-Ports angegeben. Dadurch ist der Client in der Lage, verschiedene Server zu unterscheiden. Nach dem Erstellen des Quaders und dem Bereitstellen der Kommunikation wird eine Nachricht an den Server gesendet. Diese Nachricht beinhaltet ein definiertes Kommando, das die Bildaufnahme des Servers startet und ihm eine eindeutige ID zuweist. Gleichzeitig beginnt der Client, auf eingehende Nachrichten der angeschlossenen Server zu warten. Diese Nachrichten beinhalten sowohl die ID des Servers, den Pfad des gespeicherten Standbildes als auch die Größe des Bildes in X- und Y-Ausdehnung. Ausgehend von diesen Informationen verändert der Client das Erscheinungsbild des Quaders, der mit der gesendeten ID des Servers korrespondiert.

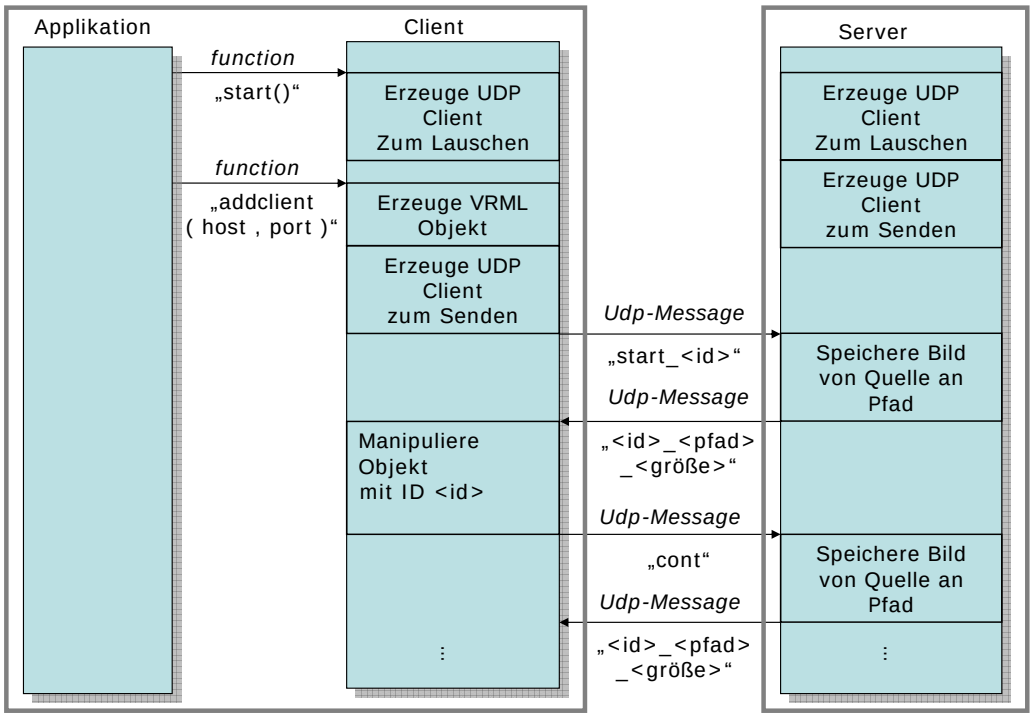


Abbildung 5.9: Kommunikation zwischen Client und Server

5.2.3.2 Kommunikation und Datenbereitstellung

Der Server des Videokonferenzsystems besteht aus zwei Kernkomponenten: die Netzwerkeinheit, die die Kommunikation mit dem Client steuert, die ankommenden Befehle auswertet und ausgehende Nachrichten verschickt, sowie die Aufnahmeeinheit, die die Media Daten, die von den angeschlossenen Quellen geliefert werden, aufnimmt und speichert. Als Quellen dienen hierbei entweder eine angeschlossene Kamera oder eine Screenshot-Funktion, die den Inhalt eines beliebigen Bereichs des Desktops aufnimmt (vgl. Abb. 5.10).

Jeder Server besitzt im Netzwerk eine eindeutige Kennung in Form des Hosts und des Ports, an dem die Netzwerkeinheit auf Nachrichten wartet. Auf diese Weise sind alle angeschlossenen Server vom Client eindeutig zu unterscheiden. Jeder Server wartet bei Start auf

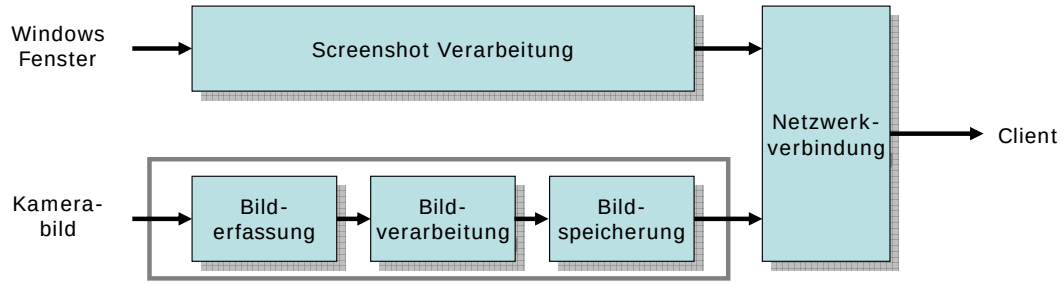


Abbildung 5.10: Schematischer Aufbau des Servers

eine Nachricht des Clients, in der er dem Server zum einen die systemeigene ID übermittelt und damit zum anderen den Aufnahmezyklus des Servers startet. Der Server nimmt dann die aktuell vorliegenden Daten der Quelle, speichert diese als Bild auf den Fileserver und sendet danach eine Nachricht an den Client. Diese Nachricht beinhaltet die ID des Servers, den Pfad des aktuellen Bildes im Netzwerk sowie die Größe des aktuellen Bildes. Der Pfad des Bildes wird so gewählt, dass die Verfügbarkeit für alle angeschlossenen Systemkomponenten sicher gestellt ist. Deshalb werden die Media Daten nicht lokal gespeichert, sondern auf einem Fileserver abgelegt, da die Pfadangabe hier für alle Komponenten identisch ist. Nach dem Versenden der Nachricht wartet der Server auf eine Reaktion des Clients, nach der die nächste Aufnahme erfolgt. Da der Dateizugriff nur sequentiell, nicht aber parallel, erfolgen kann, speichert der Server die Media Daten in einen Ringpuffer, bestehend aus zehn Bildern. Dadurch wird der zeitgleiche Zugriff von Client und Server auf die Media Daten verhindert.

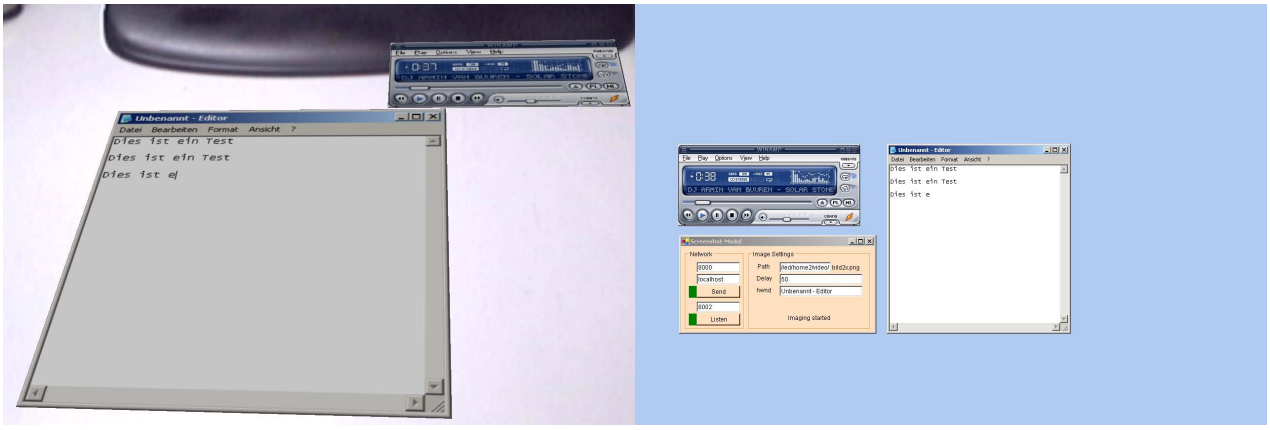


Abbildung 5.11: Darstellung von Media Daten in einer Augmented Reality Szene

Abbildung 5.11 zeigt die Visualisierung von Media Daten des Desktops (rechts) innerhalb einer AR-Szene (links). Für dieses Beispiel kommen zwei getrennte Server zum Einsatz, die durch zwei verschiedene UDP-Ports unterschieden werden. Ein Server bildet das vor ihm liegende Fenster, in diesem Fall den Mediaplayer „Winamp“ ab (ist daher auch nicht sichtbar), der zweite das Fenster des „Windows Editors“. Das Beispiel zeigt ebenfalls die frei wählbare Position, Skalierung und Orientierung innerhalb der AR Szene. Die Fenster stehen also in der AR-Szene nicht im selben Bezug zueinander, wie innerhalb des Desktops. Um den Eindruck der realen Anwesenheit des Gesprächspartners weiter zu erhöhen, werden die Media Daten, die einen Teilnehmer der Videokonferenz zeigen, vom Hintergrund befreit, sodass nur noch der Teilnehmer in die AR-Szene eingeblendet wird.

5.2.3.3 Entfernung des Hintergrundes

Für die Videokonferenz wurde ein simpler Algorithmus implementiert, der die kontinuierlich eintreffenden Bilder mit einem zuvor aufgenommenen Bild des Hintergrunds pixelweise vergleicht und somit ein Differenzbild erstellen kann. Beide Bilder müssen dabei die gleiche Größe, d.h. die gleiche Anzahl horizontaler und vertikaler Pixel, besitzen.

Zuerst wird dazu das Referenzbild in die einzelnen Pixel zerlegt. Diese Pixel werden anschließend in ihre einzelnen Farbkomponenten Rot $R_R(x, y)$, Grün $G_R(x, y)$ und Blau $B_R(x, y)$ aufgespalten. Die einzelnen Farben können dabei Werte im Bereich von 0 bis 255 annehmen. Ebenfalls werden analog für das aktuelle Videobild die Werte Rot $R_V(x, y)$, Grün $G_V(x, y)$ und Blau $B_V(x, y)$ berechnet. Nun werden die jeweiligen Farbwerte der korrespondierenden Pixel beider Bilder voneinander subtrahiert. Daraus ergibt sich für den Differenzwert Rot:

$$\delta_R = R_R(x, y) - R_V(x, y)$$

Analog werden die Differenzwerte für Grün δ_G und Blau δ_B berechnet. Die jeweiligen Differenzwerte werden anschließend quadriert und mit zuvor festgelegten Grenzwerten für jede Farbe verglichen. Diese quadratische Abweichung entscheidet, ob ein Bildpunkt des Videobildes zum Hintergrund gehört oder ob es sich um einen Teil des Vordergrundes handelt. Nur wenn die quadratische Abweichung in allen drei Farbwerten unter dem Grenzwert bleibt, zählt dieser Bildpunkt zum Hintergrund, in allen anderen Fällen handelt es sich um einen Teil des Vordergrundes. Wenn es sich bei einem Pixel um einen Teil des Hintergrunds handelt, wird der Farbwert auf den RGB-Wert $(0, 255, 0)$ gesetzt und dieses Pixel somit für die weitere Verarbeitung markiert. Dieser Algorithmus wird auf alle Bildpunkte der beiden Bilder angewandt.

In einem zweiten Verarbeitungsschritt werden alle Pixel, die zuvor als Hintergrund markiert wurden, durchsichtig gestellt. Somit wird ein Herausfiltern des Vordergrundes erreicht. Um den Erkennungsalgorithmus an die Umgebungsvariablen, wie die Ausleuchtung des Raumes und die Qualität der Kamera, anzupassen, können die Grenzwerte unabhängig voneinander und dynamisch verändert werden.

(a) Schwellwert zu groß

(b) Schwellwert zu klein

Abbildung 5.12: Differenzbilder mit verschiedenen Schwellwerten

Ein Verringern des Grenzwertes führt dazu, dass mehr Werte dem Vordergrund zugeordnet werden. So kann die Erkennung des Vordergrundes bei einer geringen Differenz des Video- und

Referenzbildes verbessert werden, wodurch das Differenzbild allerdings zu größerem Rauschen neigt. Den gegenteiligen Effekt erreicht man, wenn man den Grenzwert erhöht. Das Rauschen wird weniger, dadurch entstehen aber im Abbild der Person Löcher, da dann zu ähnliche Pixel fälschlicherweise dem Hintergrund zugeordnet werden (vgl. Abb 5.12). Durch Variieren der Grenzwerte kann eine situationsbedingte Idealkonfiguration erreicht werden.

Um eine möglichst flexible Systemgestaltung zu erreichen, kann die Differenzbildberechnung während des laufenden Betriebes der Videokonferenz aktiviert bzw. deaktiviert werden, ohne dass dadurch ein Neustart der Konferenz nötig wird. Ebenso ist das System für die spätere Evaluierung in der Lage, verschiedene Systemsetups umzusetzen. Als Ausgabe steht wahlweise ein Monitor oder ein HMD zur Verfügung. Als zugrunde liegendes Trackingsystem kann auf eines zurückgegriffen werden, das entweder auf einem Papiermarker oder einem IR-Target basiert.

KAPITEL 6

Multimodale Interaktion bei zwei exemplarischen Spielen

Die in den vorhergegangenen Kapiteln beschriebenen Bestandteile dienen als Grundlage, um multimodale Interaktion mit einem AR-System zu ermöglichen. Diese Bestandteile werden nun anhand von zwei konkreten Applikationen aus der Spieledomäne umgesetzt, zum einen eine Realisierung von Billard, zum anderen Bowling. Bei beiden Spielen werden die entwickelten Interaktionsmöglichkeiten integriert, um deren Verwendbarkeit und Potential zu demonstrieren.

6.1 BillARd - Augmented Reality Billard

Das in diesem Kapitel vorgestellte Beispiel für die Interaktionsmöglichkeiten aus den vorangegangenen Kapiteln beschreibt die Entwicklung von „BillARd“. Dabei wurde Billard in die AR transferiert. Die Aufgabe dieser Umsetzung liegt in der Demonstration der Interaktionsmöglichkeiten, nicht auf der möglichst realistischen Nachbildung physikalischer Prinzipien. Ebenfalls wurden die gültigen Spielregeln von Billard nicht in die Berechnung des Spielablaufes integriert.

6.1.1 Übersicht

„BillARd“ basiert auf dem bekannten Billard Spiel. Bei der Umsetzung in AR wird dabei ein realer Tisch und ein reales Queue verwendet. Auf dem realen Tisch wird ein virtueller Billardtisch überlagert, auf dem sich virtuelle Kugeln befinden. Durch das reale Queue können diese Kugeln angestoßen werden.

Das entwickelte „BillARd“ besteht aus mehreren Komponenten (vgl. Abb 6.1). Die visuelle Ausgabe des Systems erfolgt über ein HMD. Um ein haptisches Feedback zu erzeugen, kommt der WII-Controller zum Einsatz. Dieser dient auch als Eingabemöglichkeit und stellt als kombiniertes Ein- und Ausgabegerät die Schnittstelle für das Menü zur Steuerung von „BillARd“ dar. Mittels einer Kollisionserkennung können reale Objekte in Echtzeit in die Berechnung des Spieles eingebunden werden. Durch die angeschlossene Gestenerkennung können die virtuellen Kugeln auch mit der Hand gegriffen und versetzt werden.

Das IR-Trackingsystem empfängt die Lage und Position des realen Tisches, des Queues sowie der Finger und des Kopfes des Nutzers. Diese Informationen dienen dem System zur

Berechnung von physikalischen Vorgängen (beispielsweise das Stoßen mit dem Queue) und der Visualisierung mittels des HMDs.

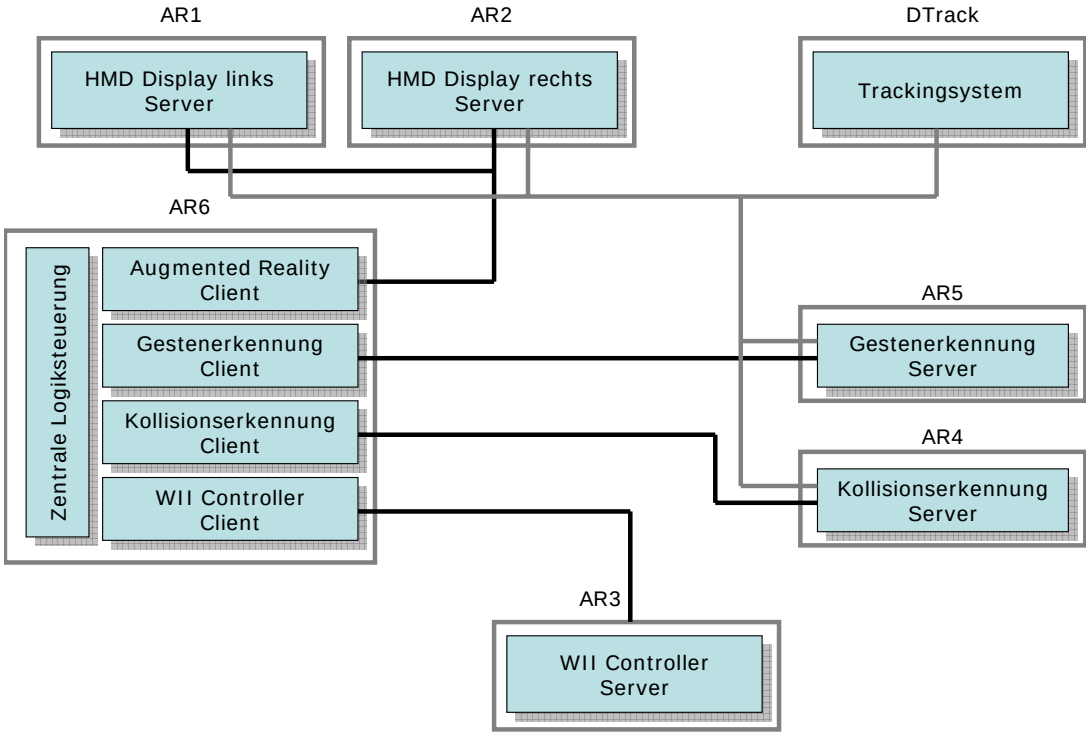


Abbildung 6.1: Schematischer Aufbau von „BillARd“

Die gesamte logische Steuerung übernimmt hier die zentrale Logiksteuerung. Diese verbindet alle einzelnen Clients der Interaktionsmöglichkeiten miteinander und stellt deren Kommunikation untereinander sicher. Diese Clients kommunizieren wiederum mit den zugehörigen Servern über ein Netzwerk. Ebenfalls simuliert die Steuerung die physikalischen Vorgänge des Spieles unter Berücksichtigung der Daten der Clients (beispielsweise die Kollision zwischen virtueller Kugel und realem Gegenstand, der durch die Kollisionserkennung auf dem Tisch detektiert wurde).

6.1.2 Aufbau der verwendeten Hardware

Grundlage von „BillARd“ ist ein Hardwareaufbau, der aus dem Tisch und dem Queue besteht. Der Tisch hat eine Größe von 120x60x60 cm und eine schwarze Oberfläche. Auf diesem ist ein Metallgerüst angebracht, an dem die Kamera der Kollisionsdetektion in einer Höhe von etwa 70cm über dem Tisch befestigt ist (vgl. Abb. 6.2). Deren Position befindet sich genau mittig über dem Tisch und die Blickrichtung ist senkrecht auf die Tischoberfläche ausgerichtet. Am Gerüst ist ein Target befestigt, über das das IR-Trackingsystem die Position des Tisches im Raum bestimmen kann. Dies ist notwendig für die Positionierung des virtuellen Tisches, der auf dem realen Tisch abgebildet wird. Weiterhin muss die Position der Kamera bekannt sein, um die Position und Größe der realen Objekte auf dem Tisch über die Kollisionserkennung exakt auf dem Tisch positionieren zu können.

Das Queue ist fest mit dem WII-Controller verbunden (vgl. Abb. 6.3). Dieser ist hier so ausgerichtet, dass die Tasten des Controllers bei der Benutzung nach oben zeigen. Oben definiert sich hierbei über die Anordnung des Targets, das am Queue angebracht ist. Die

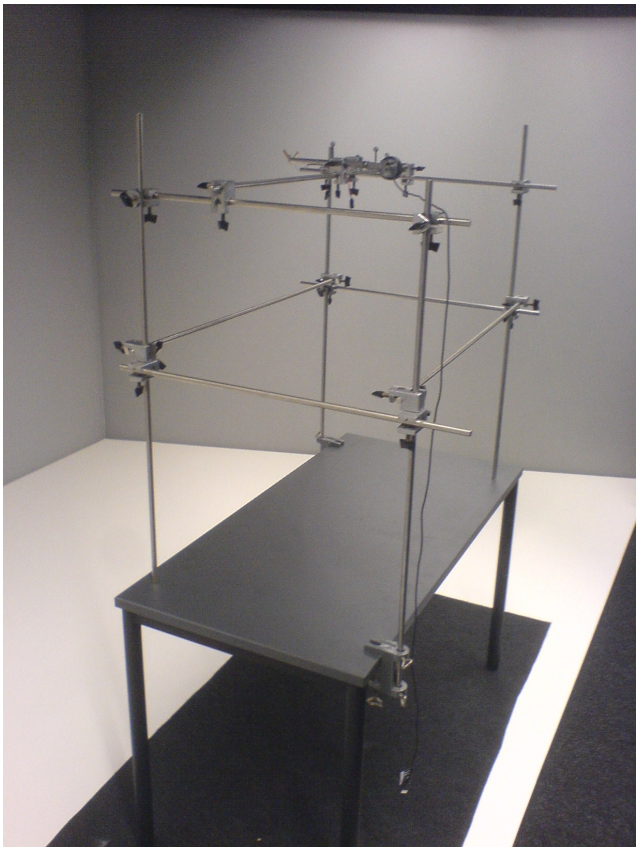


Abbildung 6.2: Tisch und Kamerahalterung von „BillARd“

Kugeln des Targets sind derart angeordnet, dass der Nutzer beim Spielen mit dem Queue nicht in den natürlichen Bewegungsabläufen beim Stoßen mit einem Queue beeinträchtigt wird. Das Target am Queue wird für die Positionierung des Menüs benötigt, aber auch um die aktuelle Position der Spitze des Queues für Stöße mit virtuellen Kugeln zu berechnen.

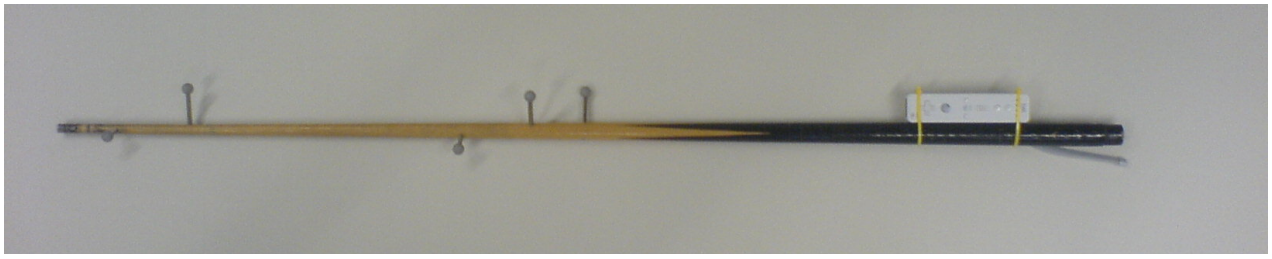


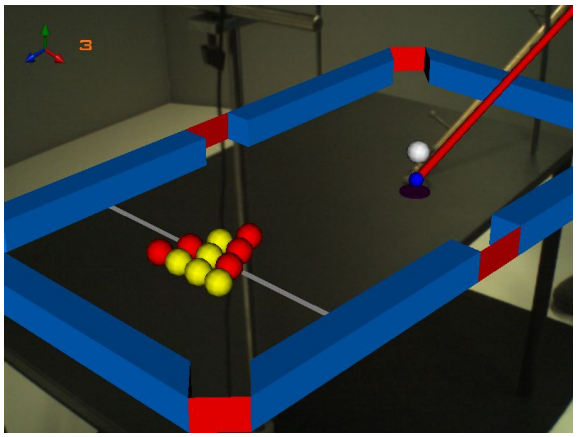
Abbildung 6.3: Queue mit WII-Controller von „BillARd“

Weiterhin befinden sich an Daumen und Zeigefinger der rechten Hand des Nutzers zwei Targets, um der Gestenerkennnung die Position und Lage der Finger im Raum mitteilen zu können.

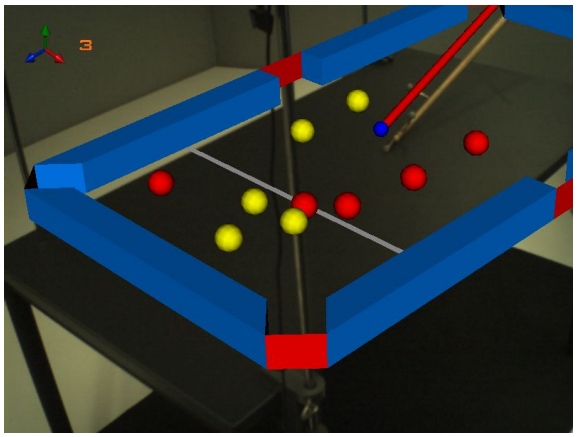
6.1.3 Logiksteuerung

Als zentrale Steuereinheit regelt die Logiksteuerung sämtliche Programmabläufe innerhalb von „BillARd“. Zu diesen Aufgaben zählen die Verbindung der einzelnen Clients und derer Daten

untereinander, ebenso wie die Berechnung der physikalischen Vorgänge. Weiterhin sorgt die Logiksteuerung für die Visualisierung der Szene innerhalb der AR für den Nutzer, aber auch für die Visualisierung der GUIs für das Programm selbst. So ist zwischen der Visualisierung für den Spieler und den Spielleiter zu unterscheiden. Das GUI für den Spielleiter bietet die Möglichkeiten, das Programm zu starten oder zu konfigurieren. Währenddessen stellt die Visualisierung dem Spieler nur die reine Szene in AR zur Verfügung (vgl. Abb. 6.4).



(a) „BillARd“ vor dem Anstoß



(b) „BillARd“ während des Spieles

Abbildung 6.4: „BillARd“ aus Sicht eines Beobachters

Für „BillARd“ wurde für die visuelle Ausgabe das in Kapitel 3.1 beschriebene System zur Visualisierung von AR verwendet. Für ein haptisches Feedback wurde der WII-Controller aus Kapitel 4.3 integriert. Dieser stellt zugleich das für „BillARd“ entwickelte Menü in AR zur Verfügung und dient somit als Eingabegerät. Als zusätzliche Eingabemöglichkeit kommt der Gestenerkenner aus Kapitel 4.1 zum Einsatz. Dieser dient dazu, dem Nutzer das Greifen der Kugeln zu ermöglichen. Die eingebundene Kollisionserkennung aus Kapitel 5.1 ermöglicht es, reale Objekte auf dem Tisch zu platzieren. Diese realen Objekte werden dann automatisch virtuell in Echtzeit modelliert und können mit dem Spielgeschehen in Interaktion treten.

Diese Vorgänge der Interaktion wirken sich auf die zugrundeliegende Physik des Spieles aus. Daher übernimmt die Logiksteuerung auch die komplette Simulation der physikalischen Vorgänge. Hierzu zählen die Roll- und Stoßeigenschaften der Kugeln untereinander, mit der Bande oder mit dem Queue. Ebenfalls werden hier die Kollisionen zwischen den Kugeln und den automatisch modellierten realen Gegenständen berücksichtigt.

Die folgenden Kapitel beschreiben die Entwicklung bzw. Integration der einzelnen Elemente der Logiksteuerung.

6.1.4 Physik

Die Simulation der Physik für „BillARd“ umfasst die Berechnung von Kollisionen von einer Kugel mit dem Queue, also dem Stoßen, mit einer weiteren Kugel, mit der Bande, sowie mit dem realen Objekt, das durch die Kollisionserkennung bereits virtuell abgebildet wurde. Alle möglichen Kollisionspartner werden während des Spieles durchgehend auf eine mögliche Kollision hin überprüft. Wird eine solche detektiert, wird die Simulation der Kollision gestartet. Die einzelnen Kollisionsmöglichkeiten und deren Berechnung werden nun im Folgenden vorgestellt.

Für die Simulation eines Stoßes mit dem Queue benötigt die Physik die Position der Spitze des Queues, die Geschwindigkeit des Queues beim Stoß, sowie die Stoßrichtung. Die beiden letzten Parameter dienen anschließend zur Berechnung der Bewegung der virtuellen, gestoßenen Kugel. Über die Position der Spitze des Queues kann ermittelt werden, wann eine Kollision von Kugel k und der Spitze s des Queues vorliegt.

Die Position der Spitze des Queues kann direkt aus den Daten des IR-Trackingsystems bestimmt werden. Das KOS des Queues ist so platziert, dass dessen Ursprung genau in der Spitze des Queues liegt. Daher ist die Position der Spitze mit dem Ursprung des KOS identisch. Eine Kollision wird dann detektiert, sobald der Abstand $d_{s,k}$ dieser Spitze zu einem Mittelpunkt m_k einer Kugel unter den Radius r_k fällt.

$$d_{s,k} < r_k = \sqrt{(x_s - x_k)^2 + (y_s - y_k)^2 + (z_s - z_k)^2}$$

Die Geschwindigkeit des Queues berechnet sich aus den vorangegangenen Bewegungsschritten, die über das Trackingsystem ermittelt werden können. Ein Bewegungsschritt stellt die Distanz v_s zwischen zwei Positionen der Spitze des Queues zu den Zeitpunkten t und $t-1$ dar.

$$v_s = \sqrt{(x_{s,t} - x_{s,t-1})^2 + (y_{s,t} - y_{s,t-1})^2 + (z_{s,t} - z_{s,t-1})^2}$$

Aus den letzten 5 Bewegungschritten wird dann eine durchschnittliche Geschwindigkeit V_s errechnet.

$$V_s = \sum_{n=0}^{-4} \frac{v_{s,n}}{5}$$

Die Stoßrichtung ergibt sich aus der Differenz zweier fester Punkte im KOS des Queues, die in Weltkoordinaten umgerechnet werden. Dazu wird der Richtungsvektor der umgerechneten Punkte ermittelt und als Richtung für den Stoß angenommen. Da der Ursprung mit der Spitze des Queues übereinstimmt und dieser an der z-Achse des Queues ausgerichtet ist, kann ein Punkt auf der z-Achse des KOS des Queues herangezogen werden. Zusammen mit dem Ursprung des KOS des Queues werden diese beiden Punkte in das KOS des Tisches übergeführt. Dort kann die Richtung aus den beiden Positionen bestimmt werden. Es gilt also für einen Punkt im KOS des Queues $\underline{x}_q$

$$\underline{x}_t = R_t(R_q\underline{x}_q + \underline{p}_q - \underline{p}_t)$$

Wobei $\underline{x}_t$ den Punkt im KOS des Tisches beschreibt. R_t bzw. R_q beschreiben die Rotationsmatrizen zwischen WeltKOS und dem KOS des Tisches bzw. Queues. $\underline{p}_t$ und $\underline{p}_q$ bezeichnen dabei den Ursprung der beiden KOS in Weltkoordinaten.

Die Kollision zwischen zwei Kugeln wird detektiert, sobald der Abstand zweier Kugelmittelpunkte kleiner als die Summe der beiden Radien der Kugeln werden. Für die Berechnung der neuen Richtung und der resultierenden Geschwindigkeit wird ein dezentraler, elastischer Stoß angenommen. Dabei sind die Mittelpunkte der beiden Kugeln $\underline{p}_1$ und $\underline{p}_2$, sowie deren Geschwindigkeiten $\underline{v}_1$ und $\underline{v}_2$, bekannt (vgl. Abb. 6.5).

Ausgehend vom Impulserhaltungssatz werden die beiden Geschwindigkeiten in zwei Anteile zerlegt. Ein Anteil wird in Richtung der Verbindungslinie der beiden Mittelpunkte beim Stoß aufgetragen, der zweite senkrecht dazu. Der Anteil in Richtung der Verbindungslinie wird nach dem Wechselwirkungsprinzip ausgetauscht, der Anteil senkrecht dazu bleibt bei den Kugeln unverändert. Um die neuen Richtungen und Geschwindigkeiten zu berechnen, benötigt man

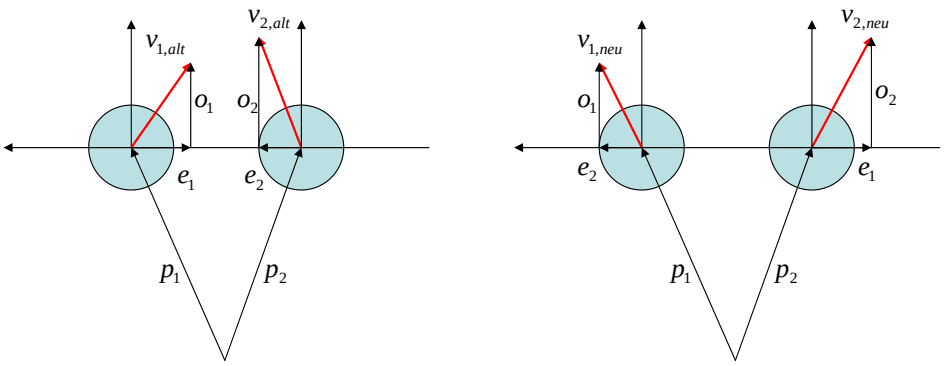


Abbildung 6.5: Berechnung der Geschwindigkeiten bei Kollision zweier Kugeln

zunächst den Richtungsvektor $\underline{e}$ der Verbindungslinie zwischen den Mittelpunkten der Kugeln. Dieser ergibt sich aus den Positionen der Kugeln zu

$$\underline{e} = \underline{p}_1 - \underline{p}_2$$

Um nun daraus die Geschwindigkeitsanteile $\underline{e}_1$ und $\underline{e}_2$ entlang der Verbindungslinie zu bestimmen, erfolgt eine Berechnung des Skalarproduktes mit dem normierten Verbindungsvektor $\underline{e}$. Um nun auch die Geschwindigkeitsanteile $\underline{o}_1$ und $\underline{o}_2$ senkrecht zur Verbindungslinie zu erhalten, werden die X- und Y-Koordinaten vertauscht und ein Vorzeichen gewechselt. Mittels dieser Vektoren kann wieder durch das Skalarprodukt auf die Geschwindigkeitsanteile geschlossen werden. Es ergeben sich also die Geschwindigkeiten $\underline{v}_{1,neu}$ und $\underline{v}_{2,neu}$ der Kugeln nach dem Stoß aus

$$\underline{v}_{1,neu} = \underline{e}_2 + \underline{o}_1 = ((\underline{v}_2 * \underline{e})\underline{e}) + ((\underline{v}_1 * \underline{o})\underline{o})$$

$$\underline{v}_{2,neu} = \underline{e}_1 + \underline{o}_2 = ((\underline{v}_1 * \underline{e})\underline{e}) + ((\underline{v}_2 * \underline{o})\underline{o})$$

Sobald der Abstand zwischen dem Kugelmittelpunkt und der Bande geringer wird, als der Radius der Kugel, wird eine Kollision zwischen Kugel und Bande erkannt. Für die Berechnung der neuen Bewegung der Kugel ist die Geschwindigkeit vor dem Stoß $\underline{v}_{alt}$ bekannt. Diese setzt sich zusammen aus dem Anteil $\underline{e}$, der senkrecht auf die Bande zeigt, und dem Anteil $\underline{o}$, der parallel zur Bande verläuft (vgl. Abb. 6.6). Der Stoß wird hier wie bei der Kollision zwischen zwei Kugeln als elastischer Stoß simuliert. Die Masse der Bande ist dabei wesentlich größer, als die der Kugel.

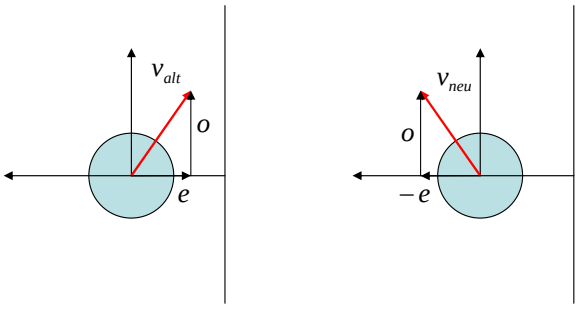


Abbildung 6.6: Berechnung der Geschwindigkeiten bei Kollision mit der Bande

Die neue resultierende Geschwindigkeit $\underline{v_{neu}}$ errechnet sich hierbei aus der Kombination von $\underline{o}$ und $\underline{-e}$

$$\underline{v_{neu}} = \underline{-e} + \underline{o}$$

Ähnlich erfolgt die Berechnung der Kollision zwischen einem modellierten Objekt in Form eines approximierten Zylinders und einer Kugel. Ein reales Objekt wird von der Kollisionserkennung durch einen Zylinder angenähert. Dabei ist der Mittelpunkt und der Radius bekannt. Eine Kollision wird ausgelöst sobald der Abstand zwischen den Mittelpunkten des Zylinders und der Kugel geringer ist, als die Summe der beiden Radien. In diesem Fall wird ausgehend vom Mittelpunkt des Zylinders $\underline{m}$ und dem Kollisionspunkt $\underline{k}$ ein Richtungsvektor $\underline{r}$ berechnet.

$$\underline{r} = \underline{m} - \underline{k}$$

Basierend auf diesem wird im Kollisionspunkt eine Ebene angenommen, die senkrecht auf dem Richtungsvektor steht. Anschließend wird die Kollision zwischen der Ebene und der Kugel analog zur Berechnung einer Kollision zwischen einer Kugel und der Bande durchgeführt. Für die Bestimmung des Zylinders zur Annäherung des realen Objektes sei hiermit auf das nachfolgende Kapitel verwiesen.

6.1.5 Integration bestehender Funktionalitäten

Die Logiksteuerung verbindet neben neuen Funktionen auch die Möglichkeiten von bereits bestehenden Systemen. Diese bestehenden Systeme wurden in die Logiksteuerung integriert, um auch auf diese Funktionen Zugriff zu erhalten. Für „BillARd“ wurde der Gestenerkenner aus Kapitel 4.1, die automatische Kollisionserkennung aus Kapitel 5.1 und der WII-Controller als Ein- und Ausgabegerät mit Menüsteuerung aus Kapitel 4.3 integriert.

Gestenerkenner
Um dem Nutzer zu ermöglichen, virtuelle Kugeln mit der Hand zu ergreifen, wurde ein System zur automatischen Erkennung von Gesten in „BillARd“ integriert. Dabei handelt es sich um den in Kapitel 4.1 vorgestellten Gestenerkenner, der auf IR-Tracking basiert. Für „BillARd“ ist es lediglich erforderlich, die Gesten des Ergreifens und Loslassens zu erkennen. Zusätzlich wird die Position der Geste benötigt. Aus diesen Angaben kann dann eine Berechnung von Greifvorgängen in der AR umgesetzt werden.

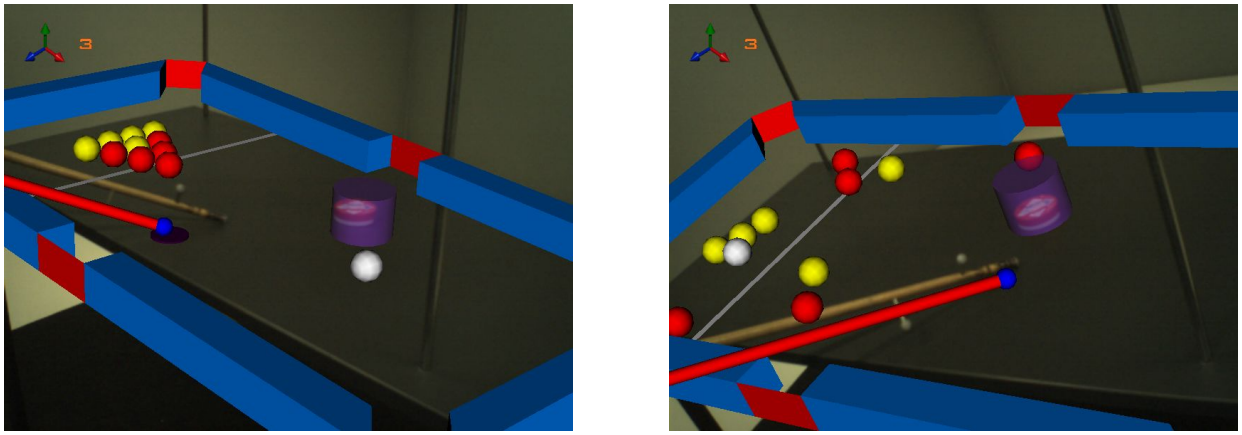
Die Greifgeste wird erkannt, sobald sich Daumen und Zeigefinger unter einen festgelegten Schwellwert annähern. Diese Berechnung erfolgt innerhalb des Gestenerkenners. Ebenfalls übermittelt dieser die Position der Geste. Bei der Integration in „BillARd“ werden diese Informationen herangezogen und mit den Positionen der virtuellen Kugeln verglichen. Befindet sich eine der Kugeln näher als eine definierte Distanz zum Ort der Geste, wird die Berechnung des Greifvorgangs der entsprechenden Kugel initiiert. Der Abstand zwischen Kugel und Finger $A_{K,F}$ berechnet sich dabei aus der Position der Kugel (x_K, y_K, z_K) und der Position des Fingers (x_F, y_F, z_F) nach

$$A_{K,F} = \sqrt{(x_K - x_F)^2 + (y_K - y_F)^2 + (z_K - z_F)^2}$$

Nach dem Greifen einer Kugel wird die Position der Kugel automatisch mit der Position des Fingers gleichgesetzt. Auf diese Weise kann die Kugel, solange sie ergriffen ist, mit der Hand

des Nutzers im Raum platziert werden. Beim Loslassen der Kugel wird diese wieder zurück auf den Tisch gesetzt. Dabei wird die Position der Kugel im KOS des Tisches errechnet und die Höhe auf den Wert Null gesetzt. Dies entspricht einem Fallen der Kugel auf den Tisch, wobei keinerlei weitere Effekte, wie etwa ein beschleunigtes Fallen oder ein Springen auf dem Tisch, in der Berechnung simuliert wird.

Kollisionserkennung
Um eine Wechselwirkung zwischen realen und virtuellen Objekten zu ermöglichen, wurde das System zur automatischen Kollisionserkennung aus Kapitel 5.1 verwendet. Dieses System ist in der Lage, ein Objekt vor einem definierten Hintergrund zu detektieren und dessen Konturen in ein virtuelles Modell überzuführen. Für „BillARd“ wurde hierzu eine Kamera über dem realen Tisch montiert, der die Oberfläche dieses Tisches beobachtet. Auf dieser Oberfläche wird ebenfalls „BillARd“ visualisiert. Ein reales Objekt auf dem Tisch kann dadurch erkannt und in die Berechnung der Laufbahn von virtuellen Kugeln eingebunden werden. Somit können virtuelle Kugeln von realen Objekten abprallen oder abgelenkt werden. Abb. 6.7 zeigt hierbei die Interaktion mit einem realen Objekt vor bzw. nach dem Stoß.



(a) Objekt vor dem Anstoß

(b) Objekt während dem Spiel

Abbildung 6.7: Kollisionserkennung bei „BillARd“

Das bestehende System übergibt der Logiksteuerung einen Array, der die Positionen von erkannten Eckpunkten des realen Objektes enthält. Um die Berechnung der Kollisionen zu vereinfachen und somit die Rechenlast zu senken, werden die realen Objekte als Zylinder angenähert. Um den Mittelpunkt $\underline{s}$ dieses Zylinders zu bestimmen, wird der Schwerpunkt der übermittelten Koordinaten $\underline{p_i}$ errechnet

$$\underline{s} = (s_x, s_y)^T = \frac{1}{N}\sum_{i=1}^{N}\underline{p_i} = \frac{1}{N}(\sum_{i=1}^{N}x_i, \sum_{i=1}^{N}y_i)^T$$

Ausgehend von diesem Schwerpunkt wird der Radius des Zylinders r aus den Punkten berechnet durch

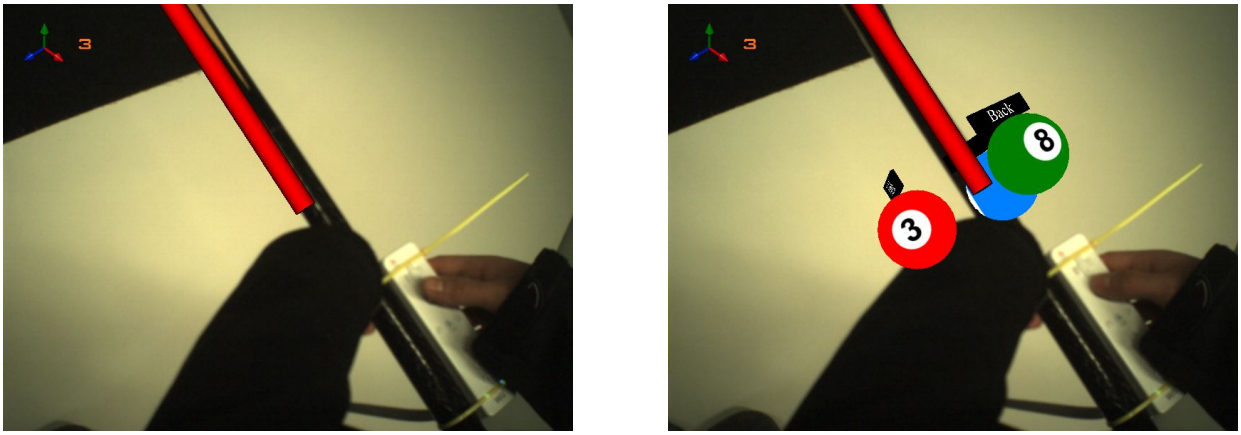
$$r = \frac{1}{N}\sum_{i=1}^{N}\|\underline{s} - \underline{p_i}\| = \frac{1}{N}\sum_{i=1}^{N}\sqrt{(s_x - x_i)^2 + (s_y - y_i)^2}$$

Basierend auf Mittelpunkt und Radius kann nun ein virtueller Zylinder erstellt werden. Dessen Höhe wird mit einem festen Wert angegeben. Dies ist jedoch nur für die Visualisierung

notwendig, da zur Berechnung von Kollisionen nur Mittelpunkt und Radius herangezogen werden. Die Vorgehensweise zur Berechnung wird im vorangegangenen Kapitel 6.1.4 beschrieben.

WII-Controller
Die Integration des WII-Controllers ermöglicht zum einen die Übertragung eines haptischen Feedbacks und zum Anderen die Steuerung von „BillARd“ mittels eines 3-D-Menüs. Das System zur Verwendung des WII-Controllers ist in Kapitel 4.3 beschrieben.

Um neben dem visuellen auch ein haptisches Feedback zu ermöglichen, wird die Möglichkeit zum Vibrieren des WII-Controllers genutzt. Wird eine Kollision des Queues mit einer Kugel erkannt, wird die Vibration des Controllers ausgelöst. Neben dieser Funktionalität dient der Controller als Steuergerät für ein 3-D-Menü, mittels dem „BillARd“ innerhalb der AR kontrolliert werden kann. Dazu wird auf die abstrakte Modellierung von Menüs mittels des Controllers zurückgegriffen. Dieser stellt ein Hauptmenü als Schnittstelle zur Applikation zur Verfügung. Innerhalb dieses Hauptmenüs können Untermenüs oder Steuerelemente definiert werden. Zu jedem dieser Elemente des Menüs können Funktionen hinterlegt werden, die aufgerufen werden, sobald ein Element an- oder abgewählt bzw. aktiviert oder deaktiviert wurde. Angewählt werden Objekte, sobald der Nutzer auf das entsprechende Element navigiert, aktiviert werden diese bei der Auswahl durch den Nutzer, um die dahinterliegende Funktion auszuführen.



(a) ohne eingeblendetes Menü (b) mit eingeblendetem Menü

Abbildung 6.8: Queue mit WII Controller

Um die Möglichkeiten der AR in vollem 3-D-Umfang nutzen zu können, wurde ein Menü entwickelt, das virtuelle Billardkugeln zur Visualisierung verwendet (vgl. Abb. 6.8). Diese Kugeln besitzen zusätzlich ein Label, auf dem der Name des Elementes des Menüs aufgelistet ist. Das umgesetzte Menü besteht zu Demonstrationszwecken aus zwei Hierarchien, in denen jeweils Steuerelemente untergebracht sind. Die Steuerelemente dienen dabei zur Änderung von Spieloptionen, wie etwa der Veränderung des Spielmodus oder physikalischen Eigenschaften, wie etwa dem Rollwiderstand der Kugeln.

Als Kugeln wurden einfache virtuelle Modelle erstellt, die allerdings mit einer internen Logik versehen sind. Das Hauptmenü bietet die Möglichkeit, angewählte Elemente als visuelles Feedback zu manipulieren (beispielsweise eine Veränderung der Farbe). In diesem Menü wird ein angewähltes Element durch die Rotation der virtuellen Kugel dargestellt. Da jedoch eine durchgehende Rotation eine dauerhafte Manipulation der Kugel erforderlich macht, diese vom

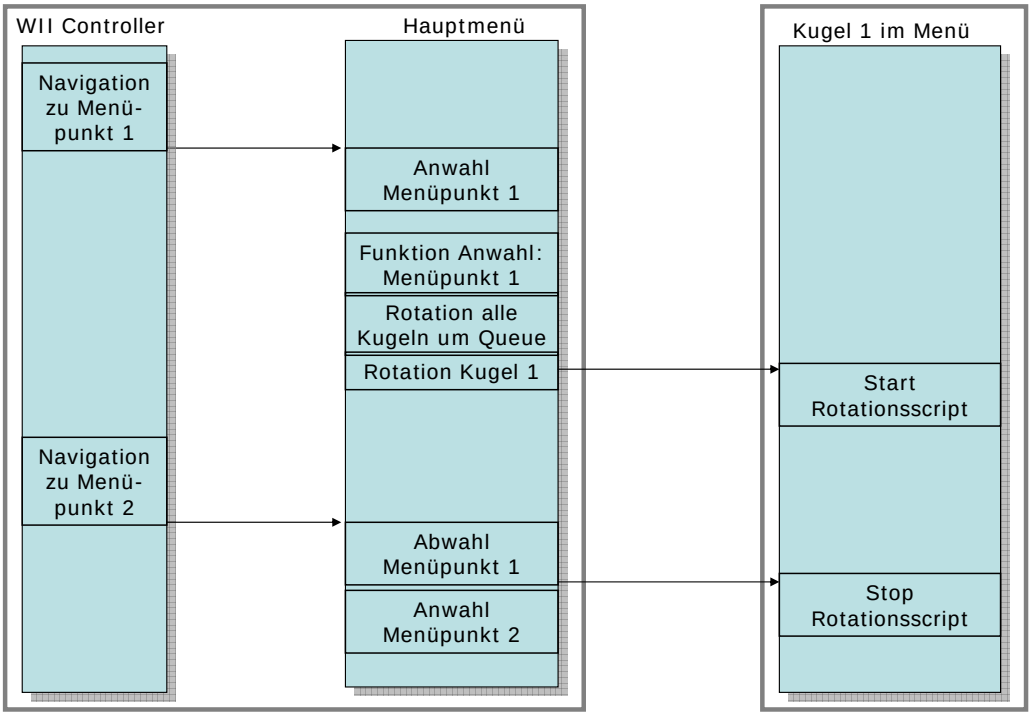


Abbildung 6.9: Ablauf bei der Navigation im Menü

Hauptmenü jedoch nicht unterstützt wird, ist die Steuerung der Logik innerhalb des Modelles untergebracht (vgl. Abb. 6.9). Bei der Anwahl des Elementes durch den Nutzer wird ein Script gestartet, dass innerhalb des Modelles die Lage der Kugel verändert. Dieses Script wird erst bei der Abwahl des Elementes wieder gestoppt. Weiterhin wird das angewählte Element in Richtung des Nutzers ausgerichtet, zeigt also bei der Anwahl nach oben. Auch hierzu wird die Rotation aller Kugeln verändert. Durch diese Rotation entsteht der Eindruck eines um das Queue drehenden Menüs. Bei Aktivierung eines Steuerelementes wird die dahinterliegende Funktion innerhalb der Logiksteuerung aufgerufen und somit eine Einstellung innerhalb von "BillARd“ verändert.

6.2 Bowl-AR-ama - Augmented Reality Bowling

Dieses Kapitel beschreibt „Bowl-AR-ama“, eine Realisierung von Bowling in der AR. Hierbei wird ähnlich wie bei „BillARd“ der Fokus nicht auf die möglichst reale Physik hinter der Simulation gelegt, sondern auf die Integration aller Interaktionsmöglichkeiten.

6.2.1 Übersicht

Das entwickelte „Bowl-AR-ama“ nutzt die Interaktionsmöglichkeiten der AR und VR, die VA für die Interaktion der Maschine zum Menschen, die Gestenerkennung als Interaktion vom Menschen zur Maschine sowie die Videokonferenz zur Interaktion zwischen Menschen. Gesteuert wird das Spiel von einer alles steuernden, zentralen Verwaltungseinheit, der Logiksteuerung. Sämtliche Komponenten (Server) sind auf eigene Computer ausgelagert, um die Performance des Gesamtsystems möglichst hoch zu halten (vgl. Abb. 6.10). Wie in den

entsprechenden Kapiteln beschrieben, werden in die Logiksteuerung die Clients der Komponenten eingebunden, mit deren die Logiksteuerung direkten Zugriff auf Funktionen des Servers erhält.

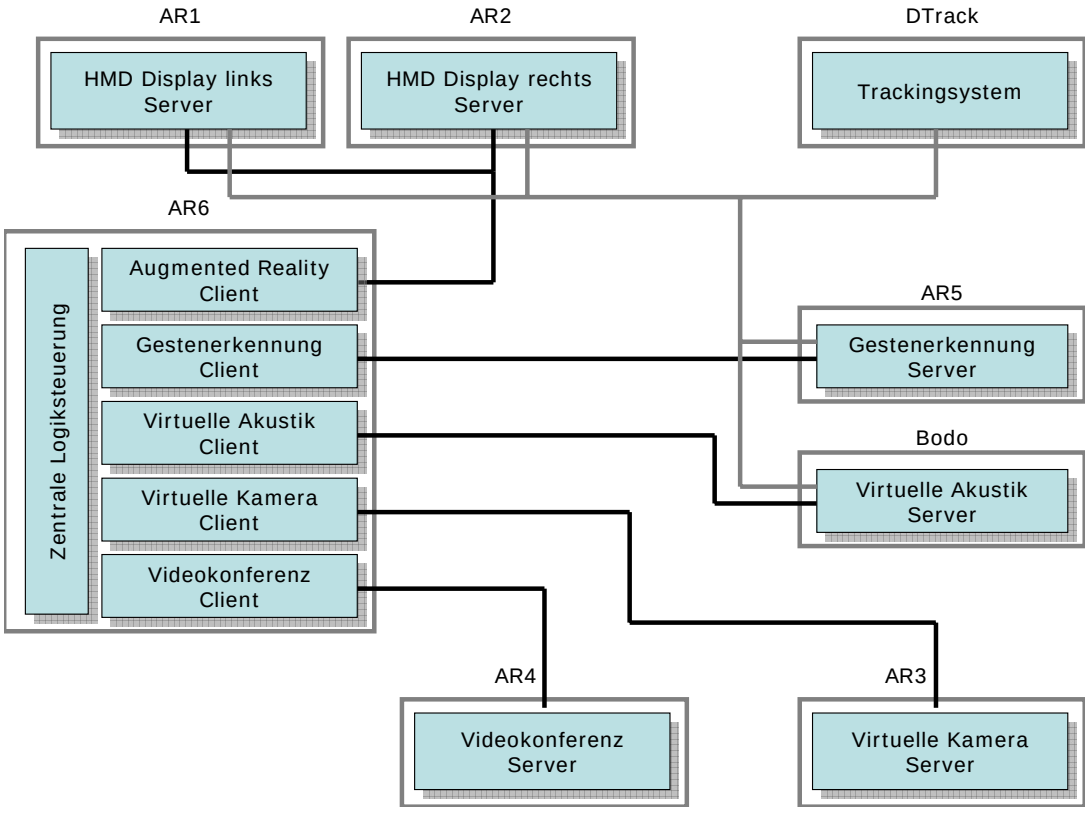


Abbildung 6.10: Schematischer Aufbau von „Bowl-AR-ama“

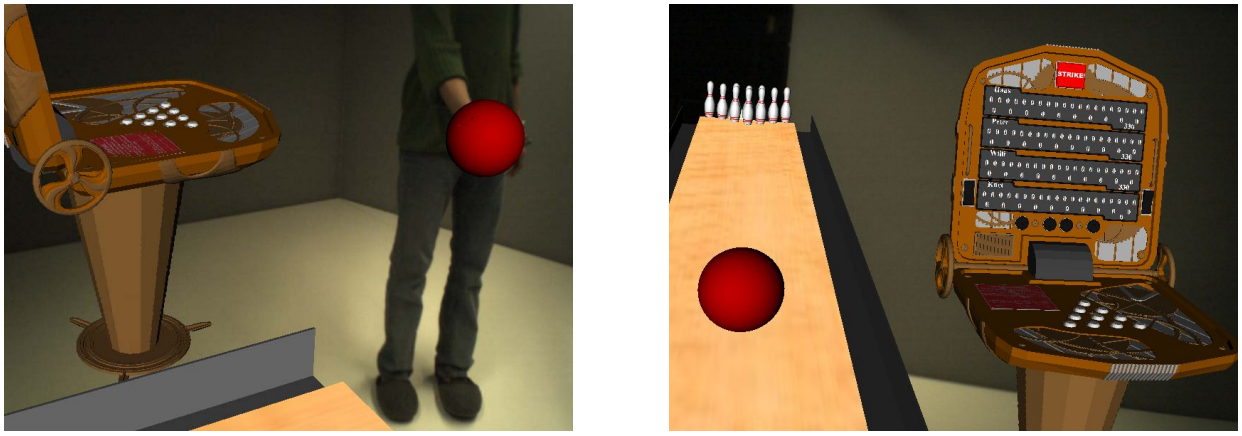
Zusätzlich zu diesen Komponenten versendet das Trackingsystem seine Informationen bzgl. der Targets an alle Komponenten außer der Videokonferenz und der virtuellen Kamera, da diese keine Daten über die erkannten Targets benötigen.

Die Logiksteuerung reagiert auf Informationen des Gestenerkenners, berechnet bei einem Wurf die physikalischen Abläufe (Rollen der Kugel, Umfallen der Pins) und gibt diese visuell über die AR bzw. VR aus. Zusätzlich werden akustische Informationen über die VA ausgegeben. Die Videokonferenz stellt weitere Spielpartner in der AR dar, sofern sie im Blickfeld einer Kamera sind, die für die Aufnahme der Gesprächspartner vorgesehen ist.

Abbildng 6.11 gibt einen kurzen Einblick in das Spiel. Die Abbildung zeigt zum einen die Sicht eines externen Beobachters, der das Spiel mit einem HMD betrachtet, zum anderen die Sicht des Spielers, der sich ebenfalls mittels eines HMDs in das Spielgeschehen versetzt fühlt.

6.2.2 Logiksteuerung

Die Logiksteuerung stellt die zentrale Komponente der Applikation „Bowl-AR-ama“ dar. Diese koordiniert das Zusammenspiel aller einzelnen Komponenten, die zur Interaktion des Nutzers mit dem Spiel notwendig sind. Zusätzlich stellt sie die komplette Steuerung der Abläufe innerhalb des Spieles sowie deren grafische Visualisierung für den Nutzer zur Verfügung. Hierzu zählen die physikalischen Abläufe innerhalb des Spieles und die Umsetzung der Regeln für



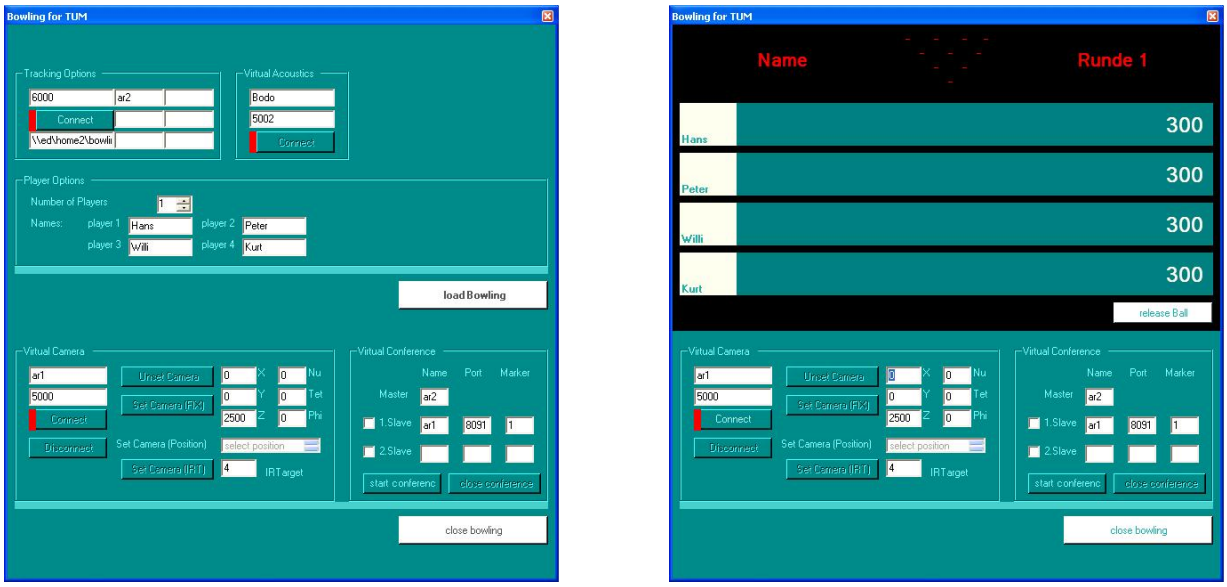
(a) Blick eines Beobachters

(b) Blick aus Nutzersicht

Abbildung 6.11: „Bowl-AR-ama“ im Einsatz

ein Bowling-Spiel. Grafisch visualisiert werden neben dem aktuellen Zustand des Spieles, wie etwa Punkteanzeigen innerhalb der AR, auch die Zustände der integrierten Komponenten zur Interaktion mithilfe eines GUIs am Steuerrechner. Über diese Darstellung wird die Konfiguration des Spieles zur Laufzeit ermöglicht.

Beim Start von „Bowl-AR-ama“ können vom Nutzer die Module, die eingebunden werden sollen, konfiguriert werden. Hierbei zählt unter anderem die Netzwerkverbindung, um eine Kommunikation der Interaktionsmodule mit der Logiksteuerung zu ermöglichen. Die Logiksteuerung stellt somit den Steuerungsrechner des Spiels dar. Abbildung 6.12 zeigt hier die Oberfläche der Logiksteuerung beim Start und während des Spiels.



(a) beim Start

(b) während des Spieles

Abbildung 6.12: Das Graphical User Interface von „Bowl-AR-ama“

Zu den Interaktionsmodulen gehört die Gestenerkennung, mit der die Bowlingkugel vom Nutzer mit der Hand ergriffen und geworfen werden kann. Dazu wurde die in Kapitel 4.1 vorgestellte Gestenerkennung verwendet. Für die akustische Ausgabe von Geräuschen, die

während des Spieles durch die Kugel oder die Kegel erzeugt werden, wurde das System zur Erzeugung von VA aus Kapitel 3.3 herangezogen. Die visuelle Anzeige erfolgt über die Darstellung mittels AR aus Kapitel 3.1 und über die virtuelle Kamera, die in Kapitel 3.2 vorgestellt wurde. Um auch eine Interaktion zwischen Spielern zu ermöglichen, wurde das System zur Erzeugung einer Videokonferenz aus Kapitel 5.2 verwendet.

Im Folgenden werden die einzelnen Komponenten erläutert, die in die Logiksteuerung integriert wurden.

6.2.3 Physik

Um eine möglichst realistische Simulation eines realen Bowling-Spieles zu erreichen, ist die Simulation von physikalischen Einflüssen auf die Szene notwendig. Dafür müssen Informationen bzgl. der Kugel, wie etwa Größe, Gewicht oder Drehimpuls, in die Berechnung einfließen. Ebenfalls sind Faktoren, die die physikalischen Eigenschaften der Bahn, wie etwa Reibung oder Unebenheiten, zu berücksichtigen, da diese die Bewegung der Kugel beeinflussen. Die Bewegung der Kugel muss unter Beeinflussung durch diese Faktoren, berechnet werden. Ebenfalls muss die Kollision zwischen der Kugel und den Kegeln, sowie der Kegel untereinander, berücksichtigt werden. Bei der Kollision entstehen abhängig von Kollisionsort und -geschwindigkeit verschiedene Kräfte, die auf die Kugel oder die Kegel wirken. Diese müssen mit ihren daraus resultierenden Kräften und Momenten für eine Simulation berechnet werden.

Da der Fokus bei „Bowl-AR-ama“ nicht auf der Umsetzung realistischer Physik liegt, sondern auf der Interaktion mit dem Spiel an sich, wurde hier auf eine existierende Physik-Berechnung zurückgegriffen. Hierbei handelt es sich um die „Open Dynamics Engine (ODE)“, die auf Berechnungen von dynamischen Körperbewegungen im Raum spezialisiert ist [79]. Diese eignet sich daher optimal zur Berechnung von Bewegungen der Kugel und Kegel in der AR-Umgebung. „ODE“ ist eine frei verfügbare Bibilothek in C++ und benötigt einen sogenannten Wrapper, der diese Bibilothek für andere Programmiersprachen zugänglich macht. Hierbei wurde auf das „Tao“ Framework [24] zurückgegriffen, da dieses die Unterstützung der hier verwendeten Programmiersprache C# bietet.

„ODE“ stellt für die Berechnung der Physik bereits alle notwendigen Basisfunktionen bereit. Dazu zählen z.B. die Erstellung von dynamischen Körpern unter Berücksichtigung ihrer Eigenschaften, wie etwa Masse, Dichte oder Oberfläche, sowie deren Positionierung im Raum. Weiterhin übernimmt „ODE“ die Kollisionsdetektion von Körpern und deren Verarbeitung unter Berücksichtigung von Reibung und anderen Einflüssen. Dabei lässt sich das Zeitintervall zwischen zwei Simulationsschritten frei wählen und somit die Simulationsgenauigkeit und der Rechenaufwand festlegen.

Die Simulation der Physik beginnt mit der Detektion eines Wurfes der Kugel, also sobald eine gegriffene Kugel losgelassen wird. Hierbei werden zu diesem Zeitpunkt die aktuell existierenden starren und beweglichen Körper, wie etwa Bahn, Kugel oder Kegel, im Kollisionsraum der Welt erzeugt. Diese stellen die virtuellen Objekte der AR-Szene dar und werden möglichst an deren physikalische Eigenschaften angepasst. Um die Berechnung zu erleichtern, werden die Kegel als abgerundete Zylinder approximiert. Nach der Erzeugung und Positionierung der Körper werden diese mit Geschwindigkeiten oder Drehmomenten belegt. Hierbei kommen die Daten des Gestenerkenners bzgl. der Geschwindigkeit und Richtung der Kugel zum Einsatz. Um die Komplexität der Simulation möglichst gering zu halten, wurde auf die Berücksichtigung eines Drehmomentes sowie auf eine geölte, unebene Bowlingbahn verzichtet. Daraus resultiert eine Bewegung, die geradlinig ist und nicht in eine Richtung während des Rollvorganges abgelenkt wird. Der Nutzer kann also seinen Wurf nur mit der Aufsetzposition

und der Richtung des Wurfes variieren. Nach der Übergabe der Parameter an die Simulation beginnt diese mit der Berechnung der vorgegebenen Zeitschritte.

Die Simulation wird hier mit einer Dauer von 120 Sekunden durchgeführt, die Berechnung erfolgt also für einen Simulationszeitraum von 2 Minuten, in denen die Bewegungen der Körper berechnet werden. Die Berechnung erfolgt für Zeitintervalle von jeweils 0,5 Sekunden. Es werden also insgesamt 240 Simulationsschritte berechnet. Hierbei werden die vorhergehenden Informationen in die Berechnung der folgenden Schritte integriert, sodass Kollisionen zwischen Körpern erkannt werden und in die Berechnung mit einfließen. Die Daten aller Körper (Translation und Rotation) werden für jeden Simulationsschritt gespeichert, um diese anschließend zur Visualisierung der Simulation verwenden zu können. Dazu werden die Translation und Rotation jedes Körpers für jeden Simulationsschritt in die Darstellung übernommen, die virtuellen Objekte der AR-Szene also verändert. Durch die fortwährende Manipulation der Objekte entsteht der Eindruck der Bewegung der virtuellen Objekte für den Nutzer.

Um die Anzahl der umgefallenen Kegel zu ermitteln, wird für jeden der letzte Rotationswert untersucht. Ist die Gesamtrotation ungleich null, so ist der Kegel umgefallen und wird beim zweiten Wurf nicht mehr aufgestellt, d.h. er ist unsichtbar und wird bei der Physik-Simulation nicht berücksichtigt. Erst wenn der nächste Spieler an der Reihe ist, werden alle Kegel von neuem aufgestellt und in die Simulation einbezogen.

6.2.4 Integration bestehender Funktionalitäten

Neben der Simulation physikalischer Eigenschaften verwaltet die Logiksteuerung auch die Interaktionsmöglichkeiten zwischen der Applikation und dem Nutzer. Die Integration dieser wird im Folgenden beschrieben:

Gestenerkenner
Zur Detektion von Gesten des Nutzers kommt für „Bowl-AR-ama“, wie bei „BillARd“, der in Kapitel 4.1 vorgestellte Gestenerkenner zum Einsatz.

Für „Bowl-AR-ama werden nur die Gesten „Greifen“ und „Loslassen“ benötigt, da diese die Interaktion des Nutzers mit der Kugel beschreiben. Über den Client des Gestenerkenners, der in die Logiksteuerung eingebunden ist, können Events bezüglich dieser Gesten abgefragt werden. Hierbei wird zwischen den Events „Greifen“ und „Loslassen“ unterschieden, die jeweils mit zwei eigenen CallBackFunktionen verknüpft sind. Wird also eine der beiden Gesten erkannt, führt das Programm automatisch die zugehörige Funktion auf.

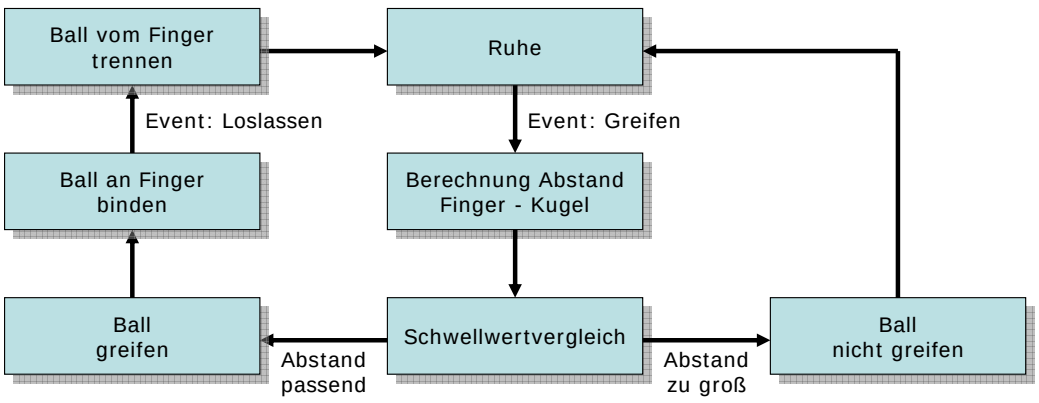


Abbildung 6.13: Behandlung von Gesten in „Bowl-AR-ama“

Abb. 6.13 zeigt, dass sich das System initial in einem Ruhezustand befindet, der nur durch das Event „Greifen“ verlassen werden kann. Bei dieser erkannten Geste wird der Abstand $A_{K,F}$ zwischen der Position $\underline{x}_K$ der Kugel und der Position $\underline{x}_F$ des Fingers des Nutzers berechnet. Ausgehend von einem festgelegten Schwellwert S, der etwas höher als der Radius der Kugel gewählt wurde, ermittelt das System, ob tatsächlich eine Intention des Nutzers vorliegt, die Kugel zu ergreifen. Diese Intention wird dem Nutzer unterstellt, wenn der berechnete Abstand unter dem Schwellwert liegt, die Finger also nahe der Kugel ist.

$$A_{K,F} = \sqrt{(x_K - x_F)^2 + (y_K - y_F)^2 + (z_K - z_F)^2}$$

Ist der Abstand größer als der Schwellwert, wird die Geste verworfen und das System kehrt in den Ruhezustand zurück. Bei einem Abstand, der kleiner als der Schwellwert ist, wird die Kugel aus dem WeltKOS herausgelöst und an das KOS des Daumens gebunden (vgl. Abb. 6.14). Dabei berechnet sich die Position der Kugel $\overline{x}_{K,D}$ im KOS des Daumens aus ihrer Position $\overline{x}_{K,W}$ im WeltKOS, der Position des Daumens $\overline{x}_{D,W}$ im WeltKOS und dessen Rotationsmatrix R zu

$$\underline{x}_{K,D} = R^{-1}(\underline{x}_{K,W} - \underline{x}_{D,W})$$

(a) Blick eines Beobachters

(b) Blick aus Sicht des Nutzers

Abbildung 6.14: Greifen der Kugel mittels Gestenerkennung

So kann der Nutzer die Kugel mit seiner Hand beliebig im Raum verschieben. Dieser Zustand kann nur durch das Event „Loslassen“ verlassen werden. Diese wird ausgelöst, sobald sich der Abstand zwischen den beiden Fingern den Schwellwert wieder übersteigt. Wird also diese Geste erkannt, wird die Kugel vom KOS des Fingers zurück in die Weltkoordinaten transformiert.

$$\underline{x}_{K,W} = \underline{x}_{D,W} + \underline{x}_{K,D} \cdot R$$

Zeitgleich erfolgt die Berechnung der Geschwindigkeit $v_{K,n}$ sowie der Richtung $\underline{d}_{K,n}$ der Kugel beim Loslassen. Die Geschwindigkeit des Daumens $v_{D,n}$ zu einer Position des Daumens $\underline{x}_{D,n}$ berechnet sich aus den Positionsänderungen zwischen der aktuellen Position $\underline{x}_{D,n}$ und der zuvor gemessenen Position $\underline{x}_{D,n-1}$ und deren zeitlichen Abständen $t_{D,n} - t_{D,n-1}$.

$$v_{D,n} = \frac{\sqrt{(x_{D,n} - x_{D,n-1})^2 + (y_{D,n} - y_{D,n-1})^2 + (z_{D,n} - z_{D,n-1})^2}}{t_n - t_{n-1}}$$

Um die Geschwindigkeitsmessung robuster gegen einmalige Fehlberechnungen, verursacht durch Aussetzer der Messreihe des Trackingsystems, zu machen, wird über die letzten fünf Geschwindigkeiten gemittelt. Daraus berechnet sich die aktuelle Geschwindigkeit der Kugel zu

$$v_{K,n} = \sum_{n=0}^{-4} \frac{v_{D,n}}{5}$$

Der Richtungsvektor $\underline{d}_{K,n}$ der Kugel zum Zeitpunkt n errechnet sich aus der Differenz der Positionsvektoren $\underline{x}_{D,n}$ und $\underline{x}_{D,n-1}$ des Daumens zu den Zeitpunkten n und $n-1$.

$$d_{K,n} = \underline{x}_{D,n} - \underline{x}_{D,n-1}$$

Diese dienen dann der Simulation als Parameter.

Virtuelle Akustik
Um die Darstellung von „Bowl-AR-ama“ nicht nur visuell zu ermöglichen, wurde das System zur Erzeugung von VA aus Kapitel 3.3 integriert.

Hierbei wird der Client des VA-Systems in die Logiksteuerung eingebunden. Dieser Client regelt ausgehend von zuvor festzulegenden Eigenschaften die klanglichen Einstellungen des VA-Systems, wie etwa das zu verwendende HRIR Set, sowie die Berechnung und Wiedergabe der Schalle.

Um Schalle abzuspielen, müssen dem System zuvor Informationen bzgl. der zu verwendenden Abtastrate übermittelt werden, welche sich hierbei nach der Rate der abzuspielenden Schalle richtet. Eine falsche Zuweisung würde in einer falschen Wiedergabe der Schalle resultieren. Anschließend erfolgt die Zuweisung der zu verwendenden HRIR-Sets, in denen zuvor gespeicherte Sätze von Impulsantworten geladen sind. Wichtigste Information ist die Angabe der Pfade, die zu den abzuspielenden Schallen gehören und sich lokal auf dem Rechner der VA befinden. Hierbei stehen acht Kanäle zur Verfügung, mit denen acht Schalle gleichzeitig an verschiedenen Positionen abgespielt werden können. Nach diesen Definitionen wartet das VA-System auf Steuerungsbefehle des Clients.

„Bowl-AR-ama“ bedient zwei verschiedene Ereignisse, denen akustische Informationen zugeordnet sind. Hierzu zählt das Auftreffen und Rollen der Kugel auf der Bahn sowie das Kollidieren und Umfallen einer bestimmten Anzahl von Kegeln. Um verschiedene Geräusche der Kollision bereitzustellen, wurden verschiedene Schalle geladen, die jeweils eine bestimmte Anzahl von Kollisionen wiedergeben. Beim Loslassen der Kugel wird das Abspielen des Schalles initiiert, der den Aufprall auf der Bahn und anschließendes Rollen der Kugel abspielt. Beim Aufprall auf die Kegel wird berechnet, wie viele Kegel beim Wurf umgefallen sind, der entsprechende Schall ausgewählt und abgespielt. Dabei wird dem VA-System übermittelt, an welcher Stelle im Raum die jeweiligen Schalle simuliert werden sollen. Hier wurde die Position für die beiden Ereignisse auf den Anfang der Bahn sowie den ersten Kegel gesetzt. Da bei diesem System die Entfernung zwischen Nutzer und Schallquelle nicht berücksichtigt wird, ist die Lautstärke in den Schallen bereits an diesen Umstand angepasst, das Rollen wird also im Verlauf leiser.

Virtuelle Kamera
Um auch Zuschauern die Möglichkeit zu bieten, am Spielgeschehen teilzunehmen, wurde die virtuelle Kamera aus Kapitel 3.2 eingebunden. Diese ermöglicht die Darstellung der AR-Szene

mittels VR, berücksichtigt aber nicht die reale Umgebung, in der die Darstellung der AR eingebettet ist.

Dazu stellt der Client der virtuellen Kamera verschiedene Optionen zur Verfügung, die aktuelle Szene darzustellen. Über die Oberfläche der Logiksteuerung ist es möglich, direkt die Position und Blickrichtung der Kamera festzulegen. Diese Einstellungen bestimmen den Sichtbereich, der anschließend dargestellt wird. Um die Bedienung für den Nutzer zu erleichtern, wurden feste Kamerapositionen vordefiniert (vgl. Abb. 6.15). Hierzu zählen statische, aber auch dynamische Positionen. Statisch sind Positionen, bei denen sich der Blickwinkel im Verlauf nicht ändert, also zum Beispiel an der Stelle des ersten Kegels mit Blickrichtung auf die Bahn oder die Draufsicht auf die Kegel. Dagegen können auch dynamische Positionen gewählt werden, zu denen die Verfolgung der Kugel von der Startposition aus gehört. Hierbei befindet sich die Kamera über dem Anfang der Bahn und blickt nach unten. Die rollende Kugel wird von dieser Position aus verfolgt, es ändert sich lediglich die Neigung der Kamera. Dabei bleibt also die Position $\underline{x}_{Ka}$ der Kamera immer gleich

$$\underline{x}_{Ka} = \begin{pmatrix} x_{Ka} \\ y_{Ka} \\ z_{Ka} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 2000 \end{pmatrix}$$

Da der Anfang der Bahn genau auf die Position $(0, 0, 0)$ gelegt wurde, muss also die Kamera nur in Richtung der Z-Achse um 2000mm verschoben werden. Die Blickrichtung der Kamera ist in Richtung der negativen Z-Achse des WeltKOS definiert. Sobald also die Kugel losrollt, muss die virtuelle Kamera um die X-Achse gedreht werden, da die Bahn entlang der Y-Achse angeordnet ist. Dazu wird die Rotationsmatrix R_{Ka} der Kamera verändert.

$$R_{Ka} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\eta & -\sin\eta \\ 0 & \sin\eta & \cos\eta \end{pmatrix}$$

Der Winkel η berechnet sich hierbei aus der Höhe der Kamera h_{Ka} und der Entfernung x_{Ku} der rollenden Kugel vom Startpunkt aus.

$$\eta = \tan\frac{x_{Ku}}{h_{Ka}}$$

Alternativ kann auch die Kameraposition über der Kugel gehalten werden, sodass sich die Kamera über der Kugel mit bewegt. Dabei wird also die Position der Kamera stets über der Position der Kugel gehalten und ergibt sich daher zu

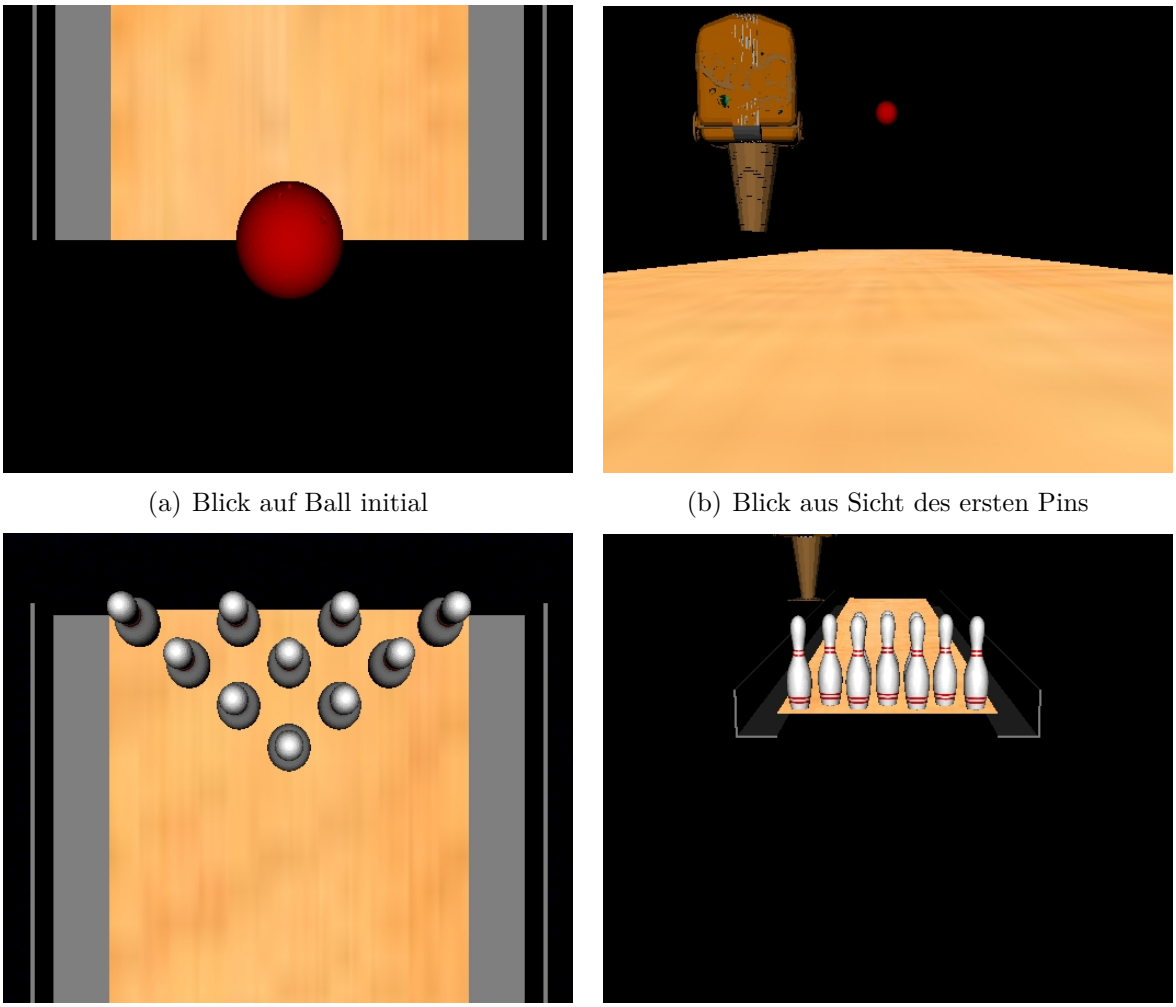
$$\underline{x}_{Ka} = \begin{pmatrix} x_{Ka} \\ y_{Ka} \\ z_{Ka} \end{pmatrix} = \begin{pmatrix} x_{Ku} \\ y_{Ku} \\ 2000 \end{pmatrix}$$

Die Rotation der Kamera entfällt hierbei, da der Blickwinkel immer entlang der negativen Z-Achse verläuft. Auf diese Weise werden Ansichten möglich, die bei einem realen Bowling-Spiel nicht möglich wären. Beispielsweise kann durch die virtuelle Kamera das Geschehen aus Sicht der rollenden Kugel dargestellt werden. Dabei wird die Position der Kamera in den Mittelpunkt der rollenden Kugel gesetzt, in Höhe des halben Radius r_{Ku} der Kugel

$$\underline{x}_{Ka} = \begin{pmatrix} x_{Ka} \\ y_{Ka} \\ z_{Ka} \end{pmatrix} = \begin{pmatrix} x_{Ku} \\ y_{Ku} \\ 0,5 \cdot r_{Ku} \end{pmatrix}$$

Der Blickwinkel beträgt hierbei 90 Grad gedreht um die X-Achse, blickt also parallel zur Bahn entlang der Y-Achse und kann daher vereinfacht werden zu

$$R_{Ka} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}$$



(a) Blick auf Ball initial

(b) Blick aus Sicht des ersten Pins

(c) Pins von oben

(d) Pins von hinten

Abbildung 6.15: Ansichten der virtuellen Kamera in „Bowl-AR-ama“

Bei der Verwendung der virtuellen Kamera wird eine Darstellung einer Kamera in die Szene eingefügt, um den Nutzern der AR den aktuellen Blickwinkel der Zuschauer mitzuteilen.

Videokonferenz

Um auch eine Interaktion zwischen Nutzern an verschiedenen Orten zu ermöglichen, wurde hier das Videokonferenzsystem aus Kapitel 5.2 verwendet. Über die Oberfläche der Logiksteuerung kann hier gewählt werden, welche Teilnehmer in die Szene eingebunden werden sollen. Hierbei ist generell eine Anzahl von maximal zwei Teilnehmern vorgesehen, welche aber beliebig erhöht werden könnte.

Die Teilnehmer können sowohl an festen Positionen als auch an dynamischen Positionen, ähnlich der virtuellen Kamera, im Raum dargestellt werden. Abb. 6.16 zeigt hierbei die

Abbildung 6.16: Videokonferenz in „Bowl-AR-ama“

Darstellung eines Teilnehmers der Videokonferenz, eingebettet in die Umgebung des „Bowl-AR-ama“.

KAPITEL 7

Evaluierung

Das folgende Kapitel beschreibt die Untersuchungen, die für die einzelnen Interaktionsmöglichkeiten durchgeführt wurden. Hierbei werden sowohl der Versuchsaufbau und -ablauf als auch die Ergebnisse beschrieben.

7.1 Gestenerkennung durch Infrarottracking

Das in Kapitel 4.1 vorgestellte System zur Erkennung von Gesten wurde in einer Evaluierung untersucht. Ziel der Untersuchung war hierbei sowohl die objektive Messung der Zuverlässigkeit als auch die subjektive Bewertung dieser Eingabemöglichkeit durch die Nutzer. Bei der Messung der Zuverlässigkeit sollte die Erkennungsleistung der richtig erkannten dynamischen Gesten ermittelt werden. Bei der Bewertung der Interaktionsmöglichkeit durch die Nutzer wurden der Tragekomfort der Targets, die Intuitivität, die Einarbeitungszeit, die Immersion und der Unterhaltungswert untersucht.

Die subjektive Bewertung sollte also zeigen, ob sich das System für Einsteiger zur Interaktion innerhalb von AR eignet, es also intuitiv zu bedienen und schnell erlernbar ist. Weiterhin sollte die Untersuchung zeigen, ob das System auch für eine längere Verwendung geeignet ist, da das System zusätzliche Hardware an der Hand des Nutzers erfordert. Schließlich war auch von Interesse, wie sehr diese Eingabemöglichkeit dem Nutzer das Gefühl vermittelt, sich innerhalb der AR-Szene zu befinden.

Für die Untersuchung wurden beide entwickelten Ansätze (vgl. Kapitel 4.1.3.2) gegenübergestellt. Des Weiteren wurde bei der verwendeten Hardware die Unterscheidung zwischen passiven und aktiven Targets berücksichtigt (vgl. Kapitel 4.1.4).

7.1.1 Versuchsaufbau/-ablauf

Die Evaluierung erfolgte im AR-Labor des Lehrstuhls und war in zwei Abschnitte untergliedert. Im ersten Teil erfolgte die objektive Messung der erkannten Gesten anhand einer festgelegten Zeichenfolge, im zweiten Abschnitt wurde eine Beispielapplikation verwendet, innerhalb derer die Nutzer Gesten zur Interaktion verwenden sollten (vgl. Abb. 7.1).

Für einen Vergleich der zugrunde liegenden Technik und ihrer Auswirkungen auf die Erkennungsleistung wurden für einen Durchgang jeweils zuerst die passiven Targets verwendet

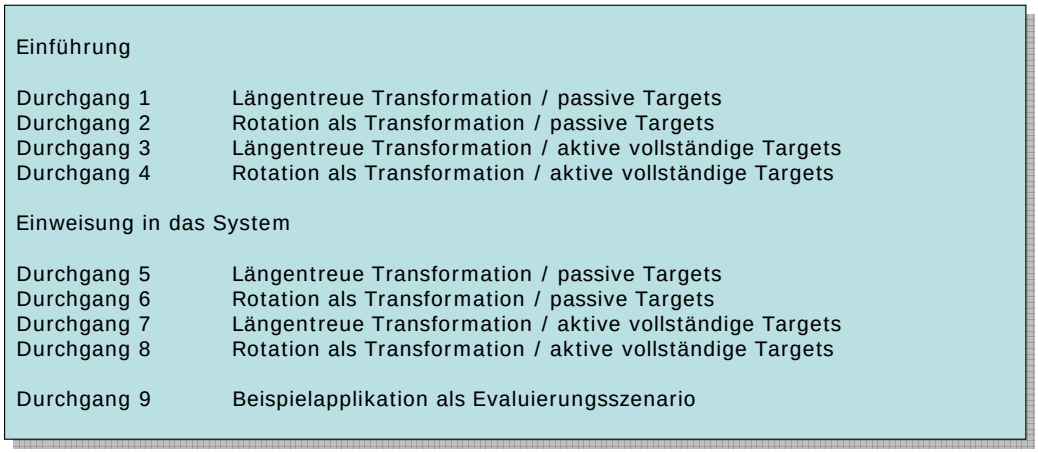


Abbildung 7.1: Versuchsablauf der Evaluierung des Gestenerkenners

und im Anschluss daran die aktiven Targets. Auf diese Weise lassen sich Unterschiede der beiden Verfahren feststellen. Ebenfalls wurden beide Ansätze der Gestenerkennung untersucht, die in Kapitel 4.1.3.2 vorgestellte Rotation und längentreue Abbildung. Zur Untersuchung der Erkennungsrate wurden als Gesten die Buchstaben und Zahlen in der Zeichenfolge

X – O – W – L – K – 1 – 2 – 3 – 4 – 5 – 1 – 3 – X – W – 2 – O – K – 4 – 5 – L – W

vorgegeben, die die Testperson der Reihe nach ausführen sollte. Dabei wurde für jeden Durchlauf protokolliert, wie viele Gesten gar nicht, falsch oder richtig erkannt wurden. Die Messung erfolgte hierbei einmal ohne Einweisung in das System, also ohne spezifische Vorgaben, und einmal mit Einweisung. Dies diente der Feststellung, ob das System von Nutzern erlernt werden muss, d.h. eine Verbesserung der Erkennungsleistung durch ein gezieltes Training der Nutzer möglich ist. Insgesamt bestand der erste Abschnitt aus acht Durchläufen, die später miteinander verglichen werden konnten. Um Aussagen hinsichtlich der Intuitivität, der Einarbeitungszeit und des Tragekomforts bewerten zu können, stand den Versuchspersonen nach jedem der acht Durchläufe ein Fragebogen zur Verfügung, in dem die Eigenschaften „Intuitivität“, „Einarbeitungszeit“ sowie „Komfort“ auf einer Skala von 1 (sehr positiv) bis 6 (sehr negativ) zu bewerten waren.

Im Anschluss an die Messung der Erkennungsrate folgte das Evaluierungsszenario anhand einer Beispielapplikation. Hierbei wurden weder Ansätze noch Targets variiert, sondern fest der Ansatz der längentreuen Abbildung mit den passiven Targets verwendet. Dieses Szenario dient nicht mehr der Erfassung der Erkennungsrate, sondern der Bewertung subjektiver Kriterien durch die Nutzer. Daher liegt der Fokus dieser Applikation auf der Interaktion mit der AR-Umgebung mittels des Gestenerkenners. Dabei stehen dem Nutzer verschiedene dynamische Gesten zur Verfügung (Zeichen „L“, „W“, „K“, „X“, „O“ sowie Zahlen 1 bis 5), mit denen sich zuvor definierte Aktionen steuern lassen:

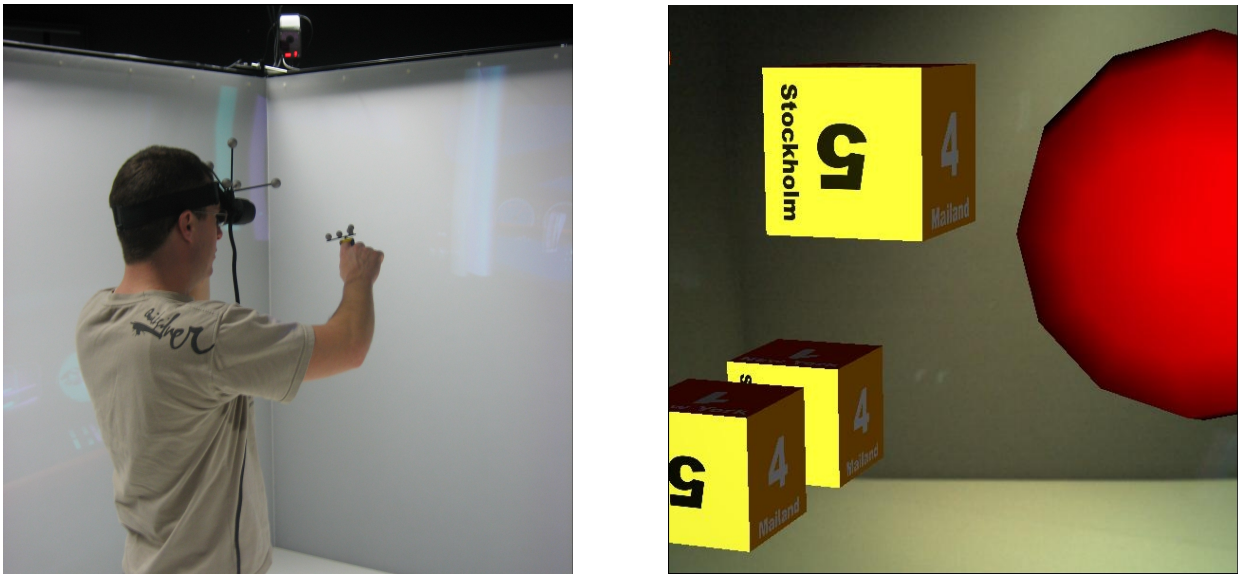
- Das Zeichen „L“ ermöglicht das Laden von neuen Objekten. Im Anschluss daran erwartet das System ein weiteres Symbol, welches den zu ladenden Objekttyp definiert (Zeichen „W“ oder „K“).

- Das Zeichen „W“ lädt, wenn es im Anschluss an das Zeichen „L“ vom Nutzer ausgeführt wird, einen Würfel in den Raum. Diesem wird zur Identifikation eine eindeutige ID

zugewiesen. Die IDs werden von 1 an aufsteigend verteilt und sind aktuell auf fünf Objekte begrenzt.

- Das Zeichen „K“ lädt, analog zu Zeichen „W“, eine Kugel in die aktuelle Umgebung und vergibt eine ID.
- Wird das Zeichen „X“ ausgeführt, wird das Objekt, welches derzeit an die Hand gebunden ist, genau an der Stelle losgelassen, an der die Bewegung des „X“ endet. Falls kein Objekt an die Hand gebunden ist, führt das System keine Aktion aus.
- Auf das Zeichen „O“ reagiert das System, indem das aktuell an die Hand des Nutzers gebundene Objekt losgelassen wird. Von dieser Position der Hand aus durchfliegt das Objekt danach automatisch einen Kreis im Raum. Auch bei diesem Zeichen wird keine Aktion ausgeführt, wenn kein Objekt an die Hand gebunden ist.
- Die Zeichen für die Zahlen „1“ bis „5“ haben zur Folge, dass das System überprüft, ob ein Objekt mit der ID der aktuellen dynamischen Geste geladen ist. Wenn dies der Fall ist, bewegt sich das Objekt mit dieser ID von seiner aktuellen Position zur rechten Hand. Dort angekommen, wird es fest an die Hand gebunden und ist so lange an dieser, bis es wieder mit einer der Zeichen „X“ oder „O“ losgelassen wird. Zu beachten ist, dass immer nur ein Objekt an die Hand gebunden werden kann.

Abbildung 7.2 zeigt eine Versuchsperson bei der Interaktion innerhalb des Evaluierungsszenarios sowie die Darstellung der Szene aus Nutzersicht.



(a) Ansicht extern (b) Ansicht intern

Abbildung 7.2: Ansichten des Evaluierungsszenarios des Gestenerkenners

Ähnlich wie nach dem ersten Abschnitt erfolgt nach dem Evaluierungsszenario eine Befragung der Nutzer mithilfe eines Fragebogens. Hierbei werden die Eigenschaften Immersion, Tragekomfort, Intuitivität, Einarbeitungszeit und Unterhaltungswert auf einer Skala von 1 (sehr positiv) bis 6 (sehr negativ) bewertet. Dadurch können Stärken oder Schwächen des Gestenerkenners unter Verwendung von AR deutlich werden.

7.1.2 Ergebnisse

An der Untersuchung beteiligten sich insgesamt zehn Personen. Davon waren 70% männlich und 30% weiblich. Das Durchschnittsalter der Versuchspersonen betrug 25 Jahre.

Vor der Diskussion der Ergebnisse der Messung und der Fragebögen werden Beobachtungen festgehalten, die während der Versuche mit den Versuchspersonen zu Tage traten. Diese wurden nicht durch die Messung oder Befragung ermittelt und werden daher gesondert behandelt. Dabei konnten diese Beobachtungen in folgende Kategorien eingeordnet werden:

Misstrauen in die Technik
Auffallend bei der Verwendung des Systems ohne Einweisung war das fehlende Vertrauen in das Gestenerkennersystem. Die Testpersonen versuchten, die Gesten so zu zeichnen, wie sie ihrer Meinung nach möglichst ideal erkannt werden könnten. Daraus resultierten nicht natürlich vollzogene Gesten, die aufgrund der Abwandlung nicht korrekt erkannt werden konnten. So wurden Zahlen beispielsweise ähnlich der digitalen Darstellung mit sieben Strichen gezeichnet.

Physikalische Probleme
Eine wichtige Rolle bei der Erkennung der Gesten spielt die Sichtbarkeit der Targets. Es hat sich gezeigt, dass sehr große Personen außerhalb des Sichtbereichs der Kameras gerieten. Damit konnte die Bewegung der Targets, und somit der Finger des Nutzers, nicht mehr aufgezeichnet werden. Speziell bei der Verwendung der aktiven Targets fiel dieser Effekt stark auf, da diese Targets bauartbedingt bereits eine wesentlich engere Abstrahlcharakteristik aufweisen als die passiven, reflektierenden Targets. Die Fehlerrate aufgrund von nicht erkannten Markern stieg aus diesem Grund bei großen Personen auf 50% der falsch erkannten Gesten an. Im Durchschnitt lag dieser Wert nur bei ca. 33%.

Pausendetektion
Große Probleme bereitete auch die Erkennung von Anfang und Ende einer Geste. Im aktuellen System wird diese Segmentierung durch eine Pausendetektion vorgenommen. Zu Beginn und zu Ende einer Geste muss also die Hand des Nutzers einen Augenblick in Ruhe bleiben, um eine Pause erkennen zu können. Viele Testpersonen starteten die Geste jedoch ohne vorherige Pause oder beendeten die Geste mit dem Herablassen des zeichnenden Arms, sodass die Geste nicht korrekt erkannt wurde. Dies liegt daran, dass die Pause erst mit dem ruhendem Arm am Körper erreicht wurde und somit die Geste bis zu diesem Zeitpunkt klassifiziert wurde. Die beschriebenen Probleme lassen sich jedoch durch die gezielte Schulung der Versuchspersonen verringern. Auf diese Weise lassen sich die Erkennungsraten im Vergleich zu ungeübten Nutzern stark verbessern. Im Folgenden werden die Ergebnisse der Messung der Erkennungsraten sowie die Auswertung der Fragebögen vorgestellt.

Messung der Erkennungsrate ohne Einweisung
Im ersten Durchlauf erhielten die Testpersonen keinerlei Einführung in das System und seine Besonderheiten. Abb. 7.3 zeigt, dass keine der Systemaufbauten eine befriedigende Erkennungsleistung der dynamischen Gesten ermöglicht. Die nicht erkannten Gesten reichen von 2,38% bei der längentreuen Abbildung bei Verwendung von LEDs bis hin zu 21,86% bei der Rotation mit passiven Targets. Die höchsten Werte richtig erkannter Gesten erreichten die beiden Ansätze mit LEDs bei Erkennungsraten von etwa 38%. Einzelne Personen erzielten jedoch bereits ohne Einweisung Erkennungsraten zwischen 60% und 90%. Die Erkennungsraten dieser Testpersonen veränderten sich im Verlauf der nachfolgenden Versuche nur geringfügig.

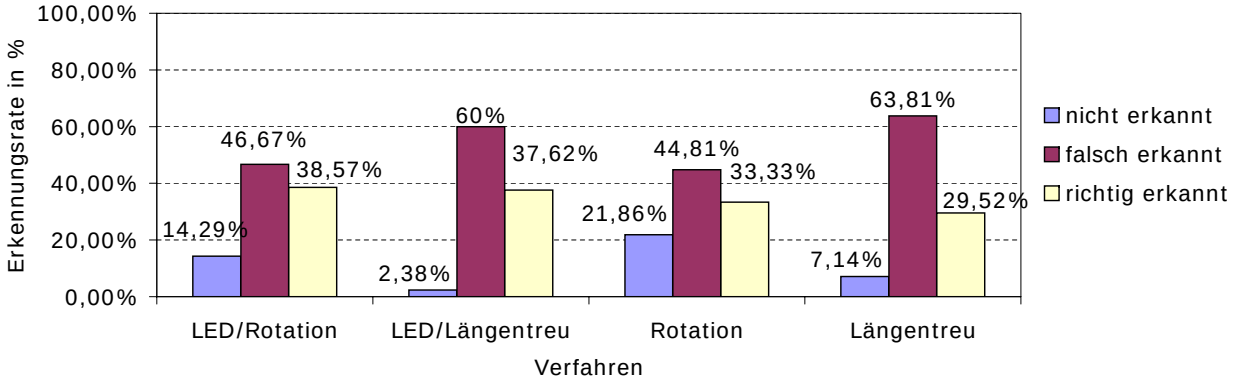


Abbildung 7.3: Erkennungsrate im ersten Test

Ursache für die niedrige Erkennungsrate sind die genannten Probleme, denen sich die Versuchspersonen nicht bewusst waren (Misstrauen, physikalische Probleme und Schwierigkeiten bei der Pausendetektion).

Messung der Erkennungsrate mit Einweisung
Nach dem ersten Durchlauf aller Kombinationen erfolgte eine Einweisung in das System sowie eine individuelle Schulung der Versuchspersonen anhand der Fehler, die beim ersten Durchgang bei dieser zu einer schlechten Erkennungsrate geführt hatten. So konnte sichergestellt werden, dass jede Versuchsperson wichtige Methoden bei der Darstellung von Gesten berücksichtigen konnte.

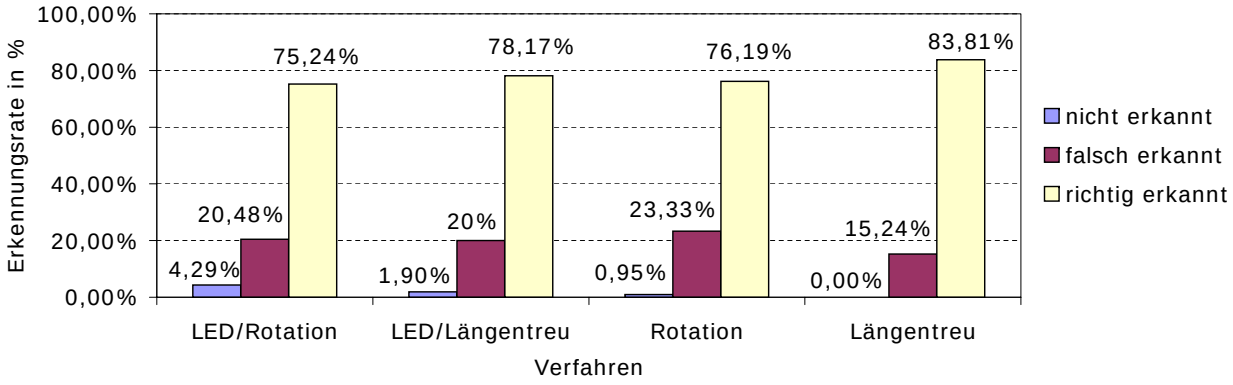


Abbildung 7.4: Erkennungsrate im zweiten Test

Daraus resultierend erkennt man in Abb. 7.4 einen Rückgang der nicht erkannten Gesten auf 0% bei der längentreuen Abbildung mit passiven Targets und einer maximalen Rate von 4,29% bei LEDs, die mittels des Rotationsverfahrens transformiert wurden. Die Erkennungsrate der Gesten stieg bei allen Methoden stark an. Im Durchschnitt erreichte die Erkennungsrate zwischen 75% und 80%. Am besten schnitten hierbei das längentreue Abbildungsverfahren sowie die Verwendung von passiven Targets ab. Es zeigt sich also, dass sowohl die Wahl der Hardware als auch die Wahl der Transformation einen Einfluss auf die Erkennungsleistung des Systems haben.

Der Unterschied zwischen aktiven und passiven Targets lässt sich durch die bauartbedingte Abstrahlcharakteristik erklären. Passive Targets reflektieren das einfallende Licht in

alle Richtungen, können also von mehreren Kameras gleichzeitig erkannt werden. Aktive Targets dagegen strahlen in einen bestimmten Sichtbereich (Begrenzung auf etwa 120 Grad) und können somit nur von einem Teil der Kameras gleichzeitig detektiert werden.

Betrachtet man die Ursachen für die Fehler genauer, so zeigt sich, dass etwa ein Drittel der Fehlerkennungen bei fehlenden Positionsdaten der Targets auftrat. Lediglich bei der Verwendung von passiven Targets mittels des längentreuen Abbildungsverfahren sank dieser Fehler auf etwa 10% ab. Diese hohen Werte verdeutlichen den großen Einfluss der Sichtbarkeit der Targets auf die Erkennungsrate. Auch durch die Interpolation der fehlenden Positionsdaten lässt sich dieser Fehler nur bis zu einem bestimmten Grad senken. Der Verlust eines Targets kann zwar für kurze Abschnitte interpoliert werden, fehlt jedoch ein Target über einen größeren Zeitraum hinweg, kann auch mit der Interpolation keine korrekte Erkennung mehr erreicht werden.

Auswertung der Fragebögen
Nach der Messung und nach dem Evaluierungsszenario wurden mithilfe eines Fragebogens die Eigenschaften des Systems durch die Versuchspersonen bewertet. Nach der Messung der Erkennungsrate handelte es sich hierbei um den Tragekomfort von aktiven und passiven Targets, der Intuitivität der Interaktion mit dem System und der subjektiv gefühlten notwendigen Einarbeitungszeit. Alle Eigenschaften wurden auf einer Skala von 1 (sehr positiv) bis 6 (sehr negativ) bewertet. Nach der Interaktion im Evaluierungsszenario wurden die Testpersonen nochmals hinsichtlich dieser Eigenschaften befragt (ausgenommen der Tragekomfort der aktiven Targets, da hier nur noch passive Verwendung fanden). Zusätzlich wurden jedoch die Eigenschaften Unterhaltungswert und Immersion ergänzt, die hierbei die Stärke des Gefühls, Teil der Szene bei der Interaktion mit dieser Szene zu sein, beschreibt.

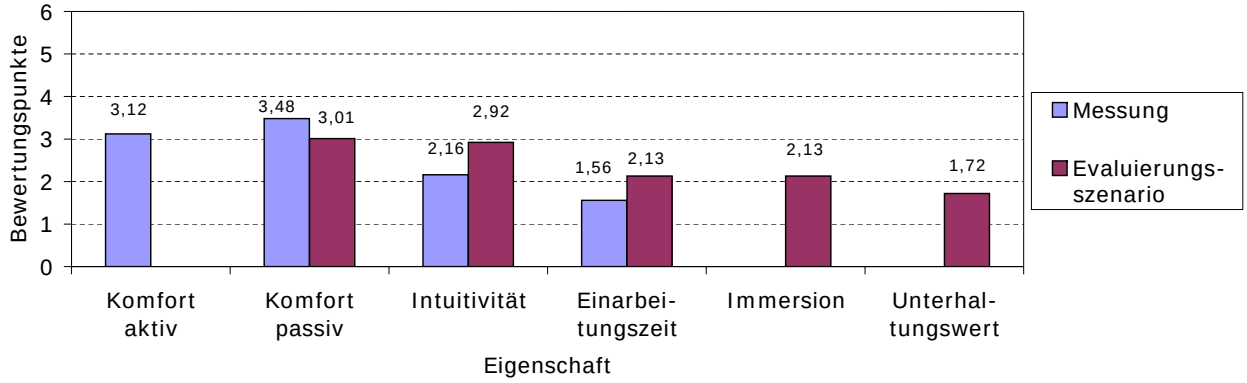


Abbildung 7.5: Ergebnisse der Fragebögen

Betrachtet man den Komfort der Targets (vgl. Abb. 7.5), fällt die durchwegs neutrale Bewertung auf. Sowohl der Tragekomfort der aktiven als auch der passiven Targets wird mit 3,12 bzw. 3,48 bewertet. Die Versuchspersonen gaben an, dass keiner der beiden Typen als komfortabel empfunden wurde. Die aktiven Targets waren den passiven zwar im Bereich der Größe und Masse an den Fingern deutlich überlegen, büßten diesen Vorsprung aber durch die zusätzlich notwendige Batterie und Verkabelung wieder ein. Bei der Verwendung im Evaluierungsszenario fiel der Wert gegenüber der Messung leicht ab, was auf die Kombination von HMD und den Targets zurückzuführen ist. Der Tragekomfort des Displays ist ebenfalls nicht optimal, sodass der Komfort des Targets hierbei im Vergleich etwas besser abschneiden kann und somit als geringfügig komfortabler wahrgenommen wurde.

Die Intuitivität der Interaktion wurde von den Versuchspersonen durchwegs als hoch eingestuft. Bei der Messung erreichte der Wert durchschnittlich 2,16, dieser stieg beim Evaluierungsszenario jedoch auf 2,92 an. Diese Verschlechterung lässt sich direkt mit den Bedienmetaphern in Verbindung bringen. Bei der Messung mussten die Versuchspersonen lediglich vorgegebene Gesten zeichnen. Hierbei traten nur bei Symbolen Probleme auf, bei denen ein mehrfaches Ansetzen notwendig war (beispielsweise „K“). Beim Evaluierungsszenario dagegen mussten bestimmte Operationen mittels passender Verkettungen von Gesten gesteuert werden. So musste ein Objekt zum Laden nach der Geste „L“ noch spezifiziert werden (Typ Würfel „W“ oder Kugel „K“). Ebenfalls fehlte einigen Nutzern die Möglichkeit, die IDs von Objekten in Erfahrung zu bringen und waren somit gezwungen, durch einfaches Testen die ID eines Objektes herauszufinden.

Die Einarbeitungszeit wurde von allen Testpersonen durchgehend als sehr niedrig eingestuft. Die Bewertung erreichte hier bei der Messung einen durchschnittlichen Wert von 1,56, beim Evaluierungsszenario 2,13. Da die Symbole in Form von Zahlen und Buchstaben bereits aus dem Alltag vertraut waren, fiel hier die Zeit zum Erlernen sehr kurz aus. Ähnlich wie bei der Intuitivität litt die Einarbeitungszeit unter der Verkettung von Gesten, um Aktionen im Evaluierungsszenario zu steuern.

Die Immersion des gesamten Systems zur Interaktion wurde als sehr hoch eingestuft. Auch der Unterhaltungswert wurde sehr positiv bewertet. Mit einer Bewertung von 2,13 bzw. 1,72 fiel das Urteil hierbei sehr positiv aus. Die Versuchspersonen interagierten länger, als es für die Untersuchung notwendig war, und begründeten dies mit dem Spaß an dieser neuen Möglichkeit der Interaktion mit einem Computersystem. Ebenfalls war ein deutliches Interesse anhand reger Nachfrage nach weiteren Interaktionsmöglichkeiten festzustellen.

7.1.3 Diskussion der Ergebnisse

Die Untersuchung hat folgende Ergebnisse geliefert: Der längentreue Ansatz zur Transformation der 3-D-Daten in 2-D-Daten hat sich in der Untersuchung als das beste Verfahren herausgestellt. Die Erkennungsraten fallen mit diesem Ansatz gegenüber der Transformation mittels der Rotation um drei bzw. sieben Prozentpunkte höher aus. Ebenfalls zeigte sich, dass die Verwendung von passiven Targets bessere Klassifikationsergebnisse lieferte als die Verwendung von aktiven Targets. Dieser Unterschied beruht jedoch größtenteils auf der engeren Abstrahlcharakteristik der LEDs gegenüber der reflektierenden Eigenschaft der passiven Targets. Betrachtet man nur ideale Fälle, bei denen die Targets also erkannt werden, liegen die aktiven Targets vor den passiven. Generell hat sich gezeigt, dass der Verlust eines Targets die Erkennung stark verringert. In der Evaluierung wurden etwa ein Drittel aller Fehler durch das Verlieren der Targets im Trackingsystem verursacht. Eine Verbesserung der physikalischen Erkennung der Targets würde also eine Steigerung der Erkennungsrate mit sich bringen.

Betrachtet man die subjektive Bewertung der Versuchspersonen, so wird deutlich, dass dieses Verfahren zur Interaktion gut angenommen wurde. Schwachpunkte bilden lediglich der Tragekomfort der Targets sowie die Umsetzung der Steuerung auf Basis der Gesten. Hier sind möglichst intuitive Eingaben erforderlich, um den sonst sehr intuitiven Umgang mit dem System, das sich auch durch die kurze Einarbeitungszeit auszeichnet, beizubehalten. Das System erzeugt bei den Versuchspersonen einen hohen Grad der Immersion, was sich letztlich sehr positiv auf den Unterhaltungswert eines solchen Systems auswirkt.

Das vorgestellte System zur Interaktion mittels Gesten ist also für die Verwendung innerhalb der AR gut geeignet. Jedoch müssen zur Optimierung noch physikalische Probleme in Angriff genommen werden. Hierzu zählt sowohl der Aufbau des Trackingsystems (also die

überlappende Abdeckung des Raumes durch den Einsatz mehrerer Kameras) als auch der Aufbau der verwendeten Targets, deren Bauform einerseits möglichst klein und leicht werden muss, um den Tragekomfort zu erhöhen, andererseits aber auch eine breite Abstrahlcharakteristik besitzen muss, um eine gute Sichtbarkeit im Trackingsystem zu gewährleisten.

7.2 Vergleich passive und aktive Infrarot-Targets

Das in Kapitel 4.1.4 vorgestellte System, aktive LEDs statt passiver Kugeln für das IR-Trackingsystem als Targets zu verwenden, wurde in einer Untersuchung mit dem bisherigen Ansatz verglichen. Dabei waren nur objektive Messwerte ausschlaggebend, die Evaluierung umfasste also nur Messungen zum Vergleich von Eigenschaften passiver und aktiver Targets. Subjektive Bewertungen durch Versuchspersonen fanden hierbei nicht statt.

Die relevanten Eigenschaften waren die maximale Leuchtdauer der aktiven Marker, die maximale Distanz zu einer Kamera, in der der Marker noch erkannt wird, der minimale Abstand zweier Marker, in dem sie noch als eigene Marker detektiert werden können und der Abstrahlwinkel, unter dem sie von einer Kamera noch erfasst werden können.

7.2.1 Versuchsaufbau/-ablauf

Um die maximale Leuchtdauer zu messen, wurde eine LED im Bereich des Trackingsystems platziert und mit einer voll aufgeladenen 9V Batterie in Betrieb genommen. Zeitgleich startete ein Messprogramm die Zeitmessung. Dieses Programm wertete jede Sekunde die Daten des Trackingsystems hinsichtlich der Sichtbarkeit des Markers aus. Sobald der Marker über 30 Sekunden nicht mehr sichtbar war, wurde die Zeitmessung gestoppt und der Wert gespeichert.

Das Messprogramm wurde ebenfalls zur Messung des maximalen Abstandes eines Markers zu einer Kamera verwendet. Hierbei wertete das Programm die Sichtbarkeit des Markers über die Daten des Trackingsystems aus. Wurde der Marker über einen Zeitraum von 3 Sekunden nicht erkannt, berechnete das Programm aus der Position der Kamera und der zuletzt gemessenen Position des Markers den Abstand und speicherte diesen ab. Zur Messung bewegte ein Versuchsleiter jeweils eine aktive LED und eine passive Kugel von der Kamera weg, solange bis der Marker nicht mehr von dieser erkannt wurde. Dabei wurde einmal ein aktiver Marker mit und einmal ohne Diffusionskugel verwendet.

Ebenfalls konnte mit dem Programm der minimale Abstand zweier Marker zueinander bestimmt werden, um noch als zwei getrennte Marker detektiert zu werden. Dazu wurden vor der Kamera zwei gleichartige Marker platziert. Deren Position wurde vom Trackingsystem ausgegeben. Sobald vom Trackingsystem, bei einer Bewegung der Marker aufeinander zu, nur mehr ein einzelner Marker erkannt wurde, speicherte das Programm den letzten Wert der beiden Positionen der Marker und berechnete den Abstand dieser Positionen.

Um den Abstrahlwinkel zu bestimmen, wurden die Marker auf einer Kreisbahn in der Kameraebene um die Kamera bewegt. Dabei wurde die Ausrichtung der LED beibehalten, wodurch sich eine Drehung der LED relativ zur Kamera einstellte. Sobald der Marker nicht mehr erkannt werden konnte, wertete das Messprogramm die Position des Markers aus und berechnete aufgrund dieser den Winkel, unter dem der Marker in der Kamera erschien.

7.2.2 Ergebnisse

In der Untersuchung zeigte sich, dass die maximale Leuchtdauer einer aktiven IR-LED im hier vorliegenden Systemaufbau bei etwa neun Stunden betrug. Dieser Wert ergab sich aus dem Mittelwert zweier getesteter LEDs, die jeweils acht bzw. zehn Stunden aktiv erkannt werden konnten.

Bei der Bestimmung des maximalen Abstandes eines Markers zur Kamera ergab sich für die passiven Marker eine Entfernung von 5,80 Meter. Aktive Marker konnten ohne Verwendung einer Diffusionskugel auf einen Anstand von maximal 6,80 Meter erkannt werden. Wurde jedoch die Diffusionskugel verwendet, sank dieser Wert auf 4,40 Meter.

Als minimaler Abstand zwischen zwei Markern ergab sich, dass passive Marker mindestens einen Abstand von 1,5 cm voneinander haben müssen, um in einem Abstand von drei Metern zur Kamera noch getrennt voneinander erscheinen und somit unterscheidbar bleiben. Dagegen konnte der Mindestabstand bei der Verwendung von aktiven Markern, sowohl mit als auch ohne Diffusionskugeln, auf einen Zentimeter bestimmt werden.

Die gemessenen Abstrahlwinkel der passiven Marker betrugen etwa 300 Grad. Dagegen konnten für aktive Marker mit Diffusionskugeln ein Winkel von 190 Grad ermittelt werden. Dieser sank bei der Verwendung ohne dieser Kugeln auf einen Wert von etwa 120 Grad ab.

7.2.3 Diskussion der Ergebnisse

Vergleicht man nun aktive und passive Targets, die aus den untersuchten Markern aufgebaut sind, kommt man zu folgendem Ergebnis:

Passive Marker sind in ihrer Abstrahlcharakteristik durch ihre geometrische Form mit 300 Grad deutlich winkelunabhängiger als aktive Marker mit 120 bzw. 190 Grad. Ihre Größe von etwa 14mm Kugeldurchmesser führen jedoch zu unhandlichen Targets, die sich vor allem bei Anwendungen negativ auswirken, bei denen ein Target möglichst klein gehalten werden sollte. Hier können aktive Targets aufgrund der kleinen LEDs besser abschneiden. Ebenfalls begünstigt dies der minimale Abstand von 1,5cm bei passiven Markern gegenüber 1cm bei aktiven Markern. Durch die höhere Leuchtstärke der aktiven Marker können diese bis zu 6,80 Meter Entfernung detektiert werden. Passive Marker dagegen werden nur auf eine Distanz von 5,80 erkannt. Jedoch schneiden hier die aktiven Marker mit Diffusionskugeln mit 4,40 Metern nochmals schlechter ab.

Betrachtet man wirtschaftliche Aspekte, so zeigt sich, dass aktive Marker in der Anschaffung deutlich günstiger sind als passive Marker. Ein passives Target mit vier Markern kostet etwa 30 Euro, dagegen ein aktives nur etwa fünf Euro. Durch den physikalischen Verschleiß der Oberfläche durch Stöße oder Kratzer bei passiven Markern müssen diese abhängig von ihrer Nutzung nach etwa einem Jahr ersetzt werden. Die Lebensdauer von LEDs dagegen ist deutlich länger.

Passive Marker sind sehr mobil einsetzbar, da sie keine Stromversorgung benötigen. Dagegen müssen zum Betrieb eines aktiven Targets entweder an eine Batterie gekoppelt sein, oder mittels eines Kabels an eine Stromversorgung angebunden sein. Dies schränkt den Bewegungsfreiraum deutlich ein.

7.3 Tangible User Interface als Eingabe

In einer Untersuchung wurde das in Kapitel 4.2 beschriebene System zur Interaktion mit virtuellen Objekten anhand eines TUIs evaluiert. Dabei stand im Vordergrund, verschiedene Interaktionsarten bei verschiedenen Interaktionsoperationen zu vergleichen. Interaktion mit virtuellen Objekten kann als Selektion und Manipulation oder einer Kombination dieser beiden Arten betrachtet werden. Zur Interaktion stehen verschiedene Eingabesysteme zur Verfügung. Als Basissystem dient eine Maus samt Keyboard. Dieses System wird neben dem in Kapitel 4.1 vorgestellten und in Kapitel 7.1 untersuchten System zur Gestenerkennung verwendet. Daher soll in dieser Evaluierung untersucht werden, welches der drei Systeme für welche Manipulationsart am besten geeignet ist. Als Referenz für alle drei Systeme dient die Interaktion in der realen Umgebung. Die Funktionen von Gestenerkenner und TUI sind in den entsprechenden Kapiteln nachzulesen.

Die Interaktion in der realen Umgebung erfolgt mit der Hand des Nutzers und mit realen Objekten, die für die späteren virtuellen Versuche auch virtuell existieren. Auf diese Weise lässt sich die Interaktion in der virtuellen Umgebung mit der Interaktion in der realen Umgebung direkt vergleichen. Die Interaktion mit Maus und Keyboard bedient sich zuvor definierter Tastenkombinationen, um bestimmte Funktionen zu starten. Diese sind anschließend in der Versuchsbeschreibung, ebenso wie die Funktionsweise aller anderen Interaktionstypen, erklärt.

Die Evaluierung zielte darauf ab, die einzelnen Systeme in Abhängigkeit der Interaktionsart auf bestimmte Eigenschaften zu untersuchen. Hierzu zählt die Immersion, bei der aufgrund der verschiedenen bauartbedingten Interaktionsarten große Bewertungsunterschiede zu erwarten sind. So bindet die Interaktion mittels Maus/Keyboard den Nutzer an ein Eingabegerät ohne direkte Verbindung zu den zu manipulierenden virtuellen Objekten. Die Interaktionsarten werden jeweils eine individuelle mentale Beanspruchung des Nutzers erzeugen. Dies ist bedingt durch das Level der Abstraktion, um Aktionen mit Reaktionen des Systems in Verbindung zu bringen. Ebenso spielt hier eine Rolle, ob die Interaktion vom Nutzer in echter 3-D ausgeführt werden kann, oder ob er, wie etwa bei der Maus, ein 2-D-Eingabegerät nutzen muss. Hierbei ist eine Transformation von 3-D in 2-D durch den Nutzer erforderlich. Neben der mentalen Beanspruchung unterscheiden sich die Systeme auch hinsichtlich der physischen Belastung. Jedes System hat seine individuelle Hardware, dessen Verwendung für den Nutzer jeweils ein bestimmtes Maß an Kraft und Ausdauer erfordert. So kann der Nutzer beispielsweise bei der Interaktion mit der Maus seine Hand auf dem Tisch liegen lassen, während er bei der Gestenerkennung die Hand durchgehend in der Luft halten muss. Aus diesen beiden Eigenschaften ergibt sich die Bewertung in Bezug auf den Komfort, den das System insgesamt bei der Interaktion bietet. Ein wichtiges Kriterium für ein Interaktionssystem stellt die Leistung dar, die ein Nutzer mithilfe des Systems erreichen kann. Hierzu zählen Eigenschaften wie die Intuitivität der Bedienung, die daraus resultierende notwendige Einarbeitungszeit sowie die Schnelligkeit, mit der Manipulationen von virtuellen Objekten durchgeführt werden können. Auch hier sind große Unterschiede zwischen den Systemen zu erwarten.

Die Untersuchung soll also zeigen, welche der Interaktionsmöglichkeiten für welche der Interaktionsarten (Selektion, Manipulation) den besten Zugang liefert. Gemessen werden hierbei Einflussgrößen, die den individuellen Zugang abbilden (Intuitivität, Einarbeitungszeit und Schnelligkeit), Größen, die ergonomische Aspekte betrachten (mentale und physische Beanspruchung sowie Komfort) und der Immersionsgrad, der zeigt, wie stark sich der Nutzer als Teil der virtuellen Szene fühlt.

7.3.1 Versuchsaufbau/-ablauf

Die Evaluierung erfolgte im AR-Labor des Lehrstuhls und war in eine Einführung und vier folgende Abschnitte untergliedert (vgl. Abb. 7.6). Zu Beginn erhielt die Versuchsperson eine kurze Einweisung, um einen Überblick über die Thematik und Technik zu bekommen. Im Anschluss wurde der genaue Ablauf der Evaluierung erläutert, bevor die Funktionsweise der Gesten und des TUIs näher erklärt wurden. Vor den Versuchen erhielt die Testperson eine kurze Einweisung in die verschiedenen Interaktionen mithilfe der Maus, der Gesten und des TUIs. Diese wurde bewusst kurz gehalten, da so die Intuitivität und Einarbeitungszeit der verschiedenen Eingaben besser beobachtet werden konnte. Nach dieser Einführung in die Funktionsweise der einzelnen Interaktionssysteme folgten drei Versuche, in denen jeweils jede der vier Interaktionsarten zur Lösung derselben Aufgabe verwendet werden sollten. Abschließend erfolgte ein Versuch, bei dem zwei Evaluierungsszenarien speziell für das TUI bewertet werden sollten.

Einführung	
Durchgang 1	Versuch 1 / reale Interaktion
Durchgang 2	Versuch 1 / Maus/Keyboard als Interaktionsgerät
Durchgang 3	Versuch 1 / Gestenerkenner als Interaktionsgerät
Durchgang 4	Versuch 1 / TUI als Interaktionsgerät
Durchgang 5	Versuch 2 / reale Interaktion
Durchgang 6	Versuch 2 / Maus/Keyboard als Interaktionsgerät
Durchgang 7	Versuch 2 / Gestenerkenner als Interaktionsgerät
Durchgang 8	Versuch 2 / TUI als Interaktionsgerät
Durchgang 9	Versuch 3 / reale Interaktion
Durchgang 10	Versuch 3 / Maus/Keyboard als Interaktionsgerät
Durchgang 11	Versuch 3 / Gestenerkenner als Interaktionsgerät
Durchgang 12	Versuch 3 / TUI als Interaktionsgerät
Durchgang 13	TUI Beispielapplikation 1 als Evaluierungsszenario
Durchgang 14	TUI Beispielapplikation 2 als Evaluierungsszenario

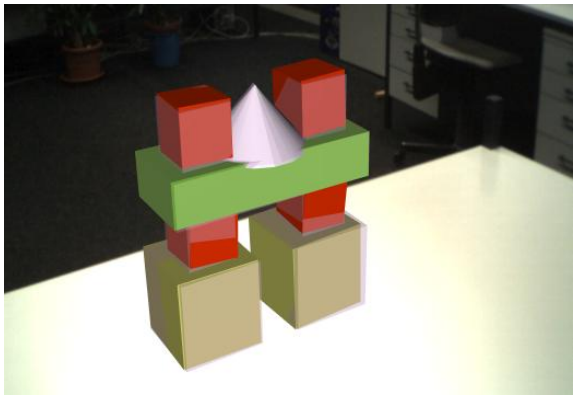
Abbildung 7.6: Versuchsablauf der Evaluierung des Tangible User Interfaces

Um Stärken und Schwächen in Abhängigkeit der Interaktionsart, also der Translation oder Rotation von Objekten, bewerten zu können, wurden die ersten drei Versuche für jede dieser Interaktionsarten durchgeführt. Versuch 1 bediente sich nur der Translation von Objekten, Versuch 2 lediglich der Rotation. In Versuch 3 erfolgte dann eine kombinierte Interaktion, die innerhalb einer Aufgabe mehrere Schritte Translation und Rotation umfasste. Im Folgenden werden die jeweiligen Aufgaben der drei Versuche erläutert.

Versuch 1 (Schriftzug: AR TUI)
Versuch 1 diente der Untersuchung der Manipulationsart Translation. Hierbei sollten Objekte also nur verschoben werden. Dazu existierten Buchstaben sowohl in realer als auch virtueller Umgebung, die zu Beginn in einer für alle Versuchspersonen und alle Interaktionssysteme gleichen Anordnung sortiert waren.

Aufgabe ist es anschließend, die Buchstaben in eine vorgegebene Reihenfolge zu bringen. Die initiale Anordnung ist in Abbildung 7.7 (a) für die virtuelle Umgebung dargestellt.

(a) Versuch 1: Buchstaben

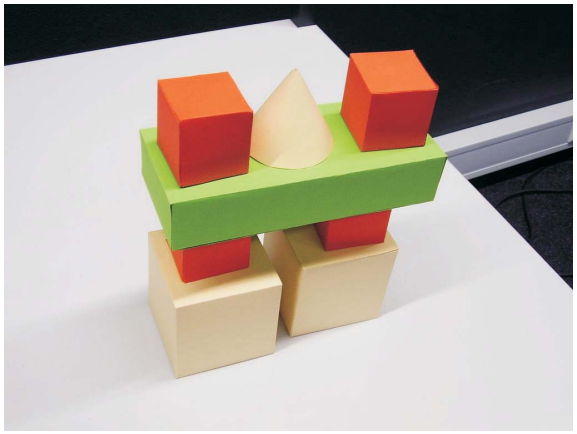


(b) Versuch 3: Baukasten

Abbildung 7.7: Virtuelle Szenarien bei der Evaluierung des Tangible User Interfaces

Versuch 2 (Ziffern-Würfel mit Städtenamen)
Versuch 2 zielte dagegen nur auf die Untersuchung der Interaktionsart Rotation ab. Objekte sollten hierbei also nicht verschoben, sondern nur gedreht werden. Dazu gab es in dieser Aufgabe einen Würfel, auf dessen Seiten Zahlen und Städtenamen notiert waren. Die Startausrichtung des Würfels war für jeden Nutzer sowie für jede Interaktionsart gleich.

(a) Versuch 2: Würfel



(b) Versuch 3: Baukasten

Abbildung 7.8: Reale Szenarien bei der Evaluierung des Tangible User Interfaces

Aufgabe war es, die Städtenamen entsprechend der Zahlen auf den Würfelseiten in aufsteigender Reihenfolge zu nennen. Abbildung 7.8 (a) zeigt hier den Würfel in der realen Umgebung zu Beginn der Aufgabe.

Versuch 3 (Baukasten)
Versuch 3 kombinierte nun die Interaktionsarten der beiden vorhergehenden Versuche 1 und 2. Hierbei sollten Objekte sowohl verschoben als auch gedreht werden. Daher stand in dieser Aufgabe ein Baukasten zur Verfügung, der Bauklötze in bestimmten Formen und Farben bereitstellte. Diese Bauklötze existierten sowohl in der realen (vgl. Abb 7.8 (b)) als auch in der virtuellen Umgebung (vgl. Abb. 7.7 (b)). Die Anordnung der Bauklötze wurde für den Beginn der Aufgabe jeweils gleich gewählt. Ziel der Aufgabe war es, die Bauklötze in die in den Abbildungen sichtbare, vorgegebene Gestalt zu bringen.

Bei allen drei Versuchen wurden objektive und subjektive Daten erhoben. Als objektive Daten wurde die Zeitdauer zur Lösung der jeweiligen Aufgabe gemessen, um eine Aussage über die Schnelligkeit der Interaktionsart treffen zu können. Subjektive Daten wurden mittels Fragebögen nach jedem Versuch erhoben. Dazu bewerteten die Versuchspersonen für jede Interaktionsart und jeden Versuch die Eigenschaften Immersion, physische sowie mentale Beanspruchung, Intuitivität, Komfort, Einarbeitungszeit und Schnelligkeit. Die Bewertungsskala reichte jeweils von 1 (sehr negativ) bis 6 (sehr positiv).

Versuch 4/5 (Beispielapplikationen TUI)
In Versuch 4 und 5 wurde ein Szenario verwendet, das speziell auf das TUI hin entwickelt wurde. Die beiden Szenarien nutzten hier also direkt die Vorteile des entwickelten Systems. Versuch 4 stellte eine virtuelle Landkarte dar, die mithilfe des TUIs gedreht und verschoben werden konnte. Ebenfalls konnte die Größe und Skala der Karte verändert werden. Versuch 5 simulierte ein virtuelles, ferngesteuertes Modellauto, das auf dem Boden des Labors fuhr und mit dem TUI gesteuert werden konnte. Ziel dieser beiden Versuche war nicht mehr der Vergleich von bestimmten Eigenschaften, sondern vielmehr eine Demonstration der Möglichkeiten dieses Interfaces und eine generelle Kritik an diesem System. Auf diese Weise konnten sich die Nutzer frei über Vor- und Nachteile des Systems äußern.

Bedienfunktionalität der Interaktionsarten
Für die Interaktion der einzelnen Systeme erfolgte eine konkrete Umsetzung, mit der bestimmte Funktionen gesteuert werden konnten. Im Folgenden sollen nun diese Funktionalitäten beschrieben werden.

Bei der Interaktion mit realen Objekten in der realen Umgebung erfolgt die Interaktion mit der Hand des Nutzers und den realen Objekten. Translation und Rotation eines Objektes als Reaktion stehen hier also in direktem Bezug zur Aktion des Nutzers.

Bei Verwendung von Maus und Keyboard als Interaktionssystem erfolgt die Selektion eines Objektes durch anklicken dieses Objektes mit der Maus. Eine Selektion ist hier notwendig, um das zu manipulierende Objekt zu kennzeichnen. Da die Maus nur ein 2-D-Eingabegerät ist, müssen bestimmte Tastenkombinationen verwendet werden, um alle drei Achsen zu berücksichtigen. Translation in der X-Y-Ebene erfolgt durch Anklicken des Objektes und gleichzeitiges Drücken der mittleren Maustaste. Eine Verschiebung entlang der Z-Achse wird durch zusätzliches Drücken der „ALT GR“ Taste des Keyboards ermöglicht. Die Rotation eines Objektes erfolgt durch Anklicken mit der linken Maustaste. Die Rotation ist hierbei auf zwei Freiheitsgrade beschränkt, so dass nur Rotationen um die X- und Y-Achse erfolgen.

Der Gestenerkenner benötigt sogenannte Targets an Daumen und Zeigefinger des Nutzers (vgl. Kapitel 4.1). Für die Evaluierung beschränkt sich das System auf die Erkennung der Greifgeste, da dynamische Gesten nicht mit der realen Interaktion oder den anderen Systemen vergleichbar wären. Die Translation und Rotation von virtuellen Objekten erfolgt hierbei also durch das Greifen eines Objektes und anschließendes Verschieben oder Drehen der Hand des Nutzers bei greifenden Fingern. Die Selektion des Objektes erfolgt über das Greifen des Objektes. Für eine realistischere Darstellung wurden die Finger des Nutzers grob durch ein virtuelles Abbild modelliert, das es erlaubt, Verdeckungen von Fingern und virtuellen Objekten korrekt darzustellen.

Das TUI bietet keine direkte Selektionsmöglichkeit, wie etwa Maus oder Gestenerkennung. Daher erfolgt die Selektion mittels der beiden oberen Momentkontakt-Taster, die die Objekte in der Reihenfolge ihrer internen ID auswählt. Um das aktuell sichtbare Objekt kenntlich

zu machen, ändert dieses bei Selektion kurz seine Größe. Mithilfe dieser Taster kann also innerhalb der Objekte vor oder zurück gesprungen werden. Für eine kontinuierliche Translation innerhalb der X-Y-Ebene eines selektierten Objektes muss der rechte Taster gedrückt werden. Dies aktiviert die Translation als Manipulation solange, bis der Taster erneut gedrückt wird. Eine Translation entlang der Z-Achse erfolgt bei zusätzlich gedrücktem rechten Momentkontakt-Taster. Zur Verschiebung muss das TUI um seine eigene X- bzw. Y-Achse geneigt werden. Um eine ungewollte Translation in kleinen Bereichen zu unterbinden, gibt es einen Schwellwert, der überschritten werden muss, um die Translation zu starten (hier 20 Grad). Eine beschleunigte Translation ist durch einen weiteren Schwellwert geregelt, der bei Überschreiten die Translationsgeschwindigkeit um den Faktor 4 erhöht (hier 65 Grad). Die kontinuierliche Rotation eines Objektes um die X- bzw. Y-Achse wird mittels des linken Tasters aktiviert. Das Neigen des Interfaces um seine X- bzw. Y-Achse aktiviert hierbei die Rotation, wobei die Drehgeschwindigkeit direkt proportional zum Neigungswinkel des Interfaces ist. Eine Rotation um die Z-Achse erfolgt durch Drücken des linken oberen Momentkontakt-Tasters. Danach lässt sich durch Drehung um die Z-Achse ein virtuelles Objekt um denselben Winkel drehen.

7.3.2 Ergebnisse

Die Evaluierung wurde mit 15 Versuchspersonen durchgeführt, wobei die Teilnehmer zwischen 23 und 27 Jahre alt waren und das Durchschnittsalter bei etwa 25 Jahren lag. Unter den Probanden befanden sich vier Frauen und elf Männer.

Messung der Ausführungsdauer
Vergleicht man die Messergebnisse der Ausführungsdauer, fällt auf, dass die Interaktion in der realen Umgebung mit Abstand die schnellste darstellt.

Bei der Interaktion, die sich auf die Translation beschränkt (Versuch 1), wurde die Aufgabe im Schnitt innerhalb von 6,7 Sekunden gelöst (vgl. Abb. 7.9).

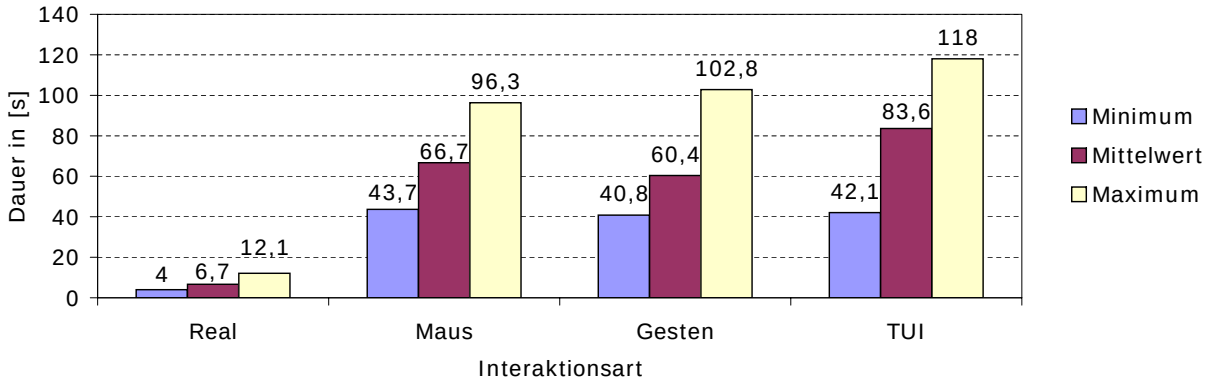


Abbildung 7.9: Zeiten für Versuch 1 (Schriftzug: AR TUI)

Beinahe gleich schnitten hier die Interaktionsarten Maus/Keyboard mit 66,7 Sekunden und Gesten mit 60,4 Sekunden ab. Die Interaktion mit dem TUI ist mit 83,6 Sekunden knapp 30% langsamer als die beiden anderen Methoden. Diese Ergebnisse zeigen, dass keine der virtuellen Interaktionen an die Ausführungszeiten der realen Interaktion heranreicht. Im Schnitt sind diese um den Faktor 10 bis 13 langsamer. Dies liegt darin begründet, dass für die Interaktion mit den realen Objekten beide Hände verwendet werden konnten. Ebenfalls unterscheidet

sich diese Interaktion darin, dass physikalische Einflüsse bei der virtuellen Interaktion nicht berücksichtigt wurden. So können Objekte nicht einfach auf dem Tisch verschoben werden, da keine Einflüsse von Schwerkraft berücksichtigt wurden. Das TUI hat in diesem Versuch den Nachteil, dass mehrere Objekte in der Szene manipuliert werden müssen. Es ist also eine Selektion durch eine Listenauswahl notwendig, die mit dem TUI nur sehr umständlich ausführbar ist, während Maus/Keyboard und Gesten eine direkte Auswahl des Objektes erlauben. Betrachtet man jedoch die minimale Ausführungszeit, so zeigt sich, dass es Nutzer gibt, die mit allen drei virtuellen Interaktionsmethoden ähnlich schnelle Zeiten erreichen konnten.

Vergleicht man die Zeiten, die bei der Rotation (Versuch 2) gemessen wurden, fällt auf, dass die Interaktion mit Maus/Keyboard hier deutlich schlechter abschneidet. In der realen Umgebung erfolgte die Lösung der Aufgabe im Schnitt innerhalb von 8,8 Sekunden (vgl. Abb. 7.10).

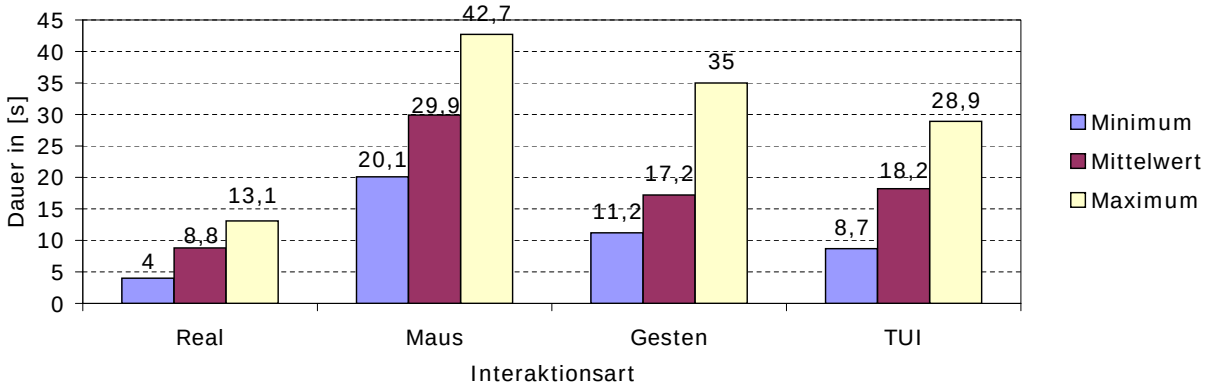


Abbildung 7.10: Zeiten für Versuch 2 (Ziffern-Würfel mit Städtenamen)

Die durchschnittliche Dauer bei Verwendung von Maus/Keyboard liegt hier bei 29,9 Sekunden. Dagegen schneiden Gestenerkennung und TUI beinahe gleich gut mit 17,2 und 18,2 Sekunden mit etwa der halben Zeit ab. Auch hier zeigt sich, dass die reale Interaktion von keiner der virtuellen erreicht werden kann. Jedoch liegt der Faktor mit etwa 2 bis 3 hierbei deutlich niedriger als bei der Translation. Begründen lässt sich diese Verbesserung gegenüber der Translation mit dem TUI mit der Beschränkung auf ein einziges Objekt, das zu rotieren war. Das schlechte Abschneiden von Maus/Keyboard lässt sich hier leicht damit begründen, dass bei der Rotation die Nachteile der Transformation von einer 2-D-Eingabemöglichkeit auf eine 3-D-Manipulation voll zum Tragen kommen. Für die Rotation muss neben der Maustaste noch zusätzlich eine Taste auf der Tastatur gedrückt werden. In diesem Versuch schneiden das TUI und das Gestenerkennungssystem um den Faktor 2 besser ab, da das Objekt direkt manipuliert werden kann. Die Aktion des Nutzers ist also direkt mit der Reaktion des Objektes verknüpft.

Ein Vergleich der Zeiten für die Kombination von Translation und Rotation (Versuch 3) zeigt, dass hier ähnliche Differenzen wie bei Versuch 1 zu messen sind. Die reale Interaktion ist hier nach etwa 9,9 Sekunden beendet (vgl. Abb. 7.11). Die Interaktion mittels Gesten zeigt sich hier mit 93 Sekunden im Mittel als die schnellste Methode. Das TUI benötigt hier die längste Zeit mit 138 Sekunden, die Interaktion mit Maus/Keyboard reiht sich zwischen die beiden anderen mit 111,3 Sekunden ein. In diesem Versuch tritt der Effekt der fehlenden physikalischen Simulation von Kollisionen und Schwerkraft deutlich hervor. In der realen Interaktion können Objekte einfach aufeinander abgestellt werden, wohingegen in der virtuellen Interaktion Objekte durch fehlende Kollisionserkennung ineinander gestellt werden können.

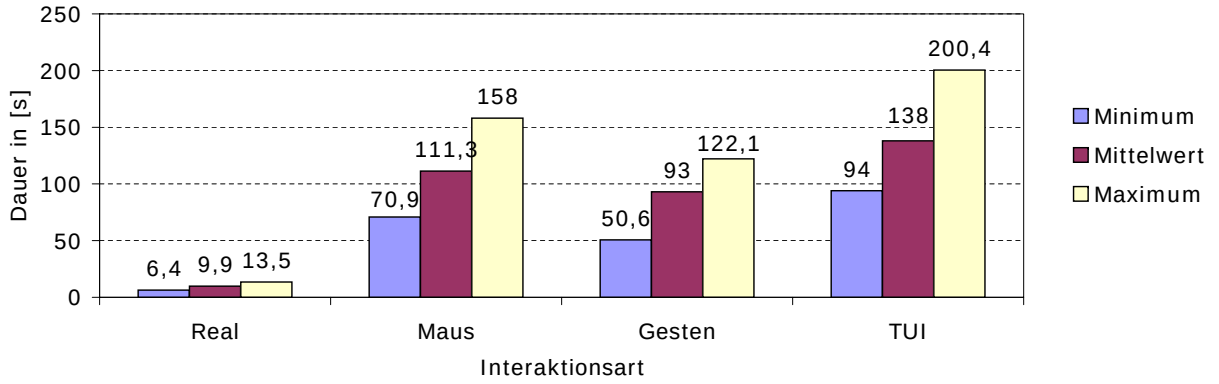


Abbildung 7.11: Zeiten für Versuch 3 (Baukasten)

Das Gestenerkennungssystem schneidet hier am besten ab, da die Selektion eines Objektes direkt möglich ist und die Manipulation direkt erfolgt. Durch die notwendige Selektion verliert das TUI Zeit während des Auswählens eines Objektes aus einer Liste. Dabei gewinnt es aber durch die direkte Manipulationsmöglichkeit. Die Maus/Keyboard-Kombination dagegen ermöglicht eine direkte Selektion durch Anklicken, kann jedoch keine direkte Manipulation zur Verfügung stellen. Die Ergebnisse zeigen jedoch, dass die Selektion hier den größeren Einfluss auf die Ausführungszeiten hat.

Fragebogen
Neben einer objektiven Untersuchung anhand der Ausführungsdauern wurden in der Evaluierung auch subjektive Daten erfasst. Abbildung 7.12 und 7.13 zeigen die Ergebnisse der Auswertung des Fragebogens, wobei jeweils der Durchschnitt der Werte für die jeweilige Eigenschaft angegeben wird.

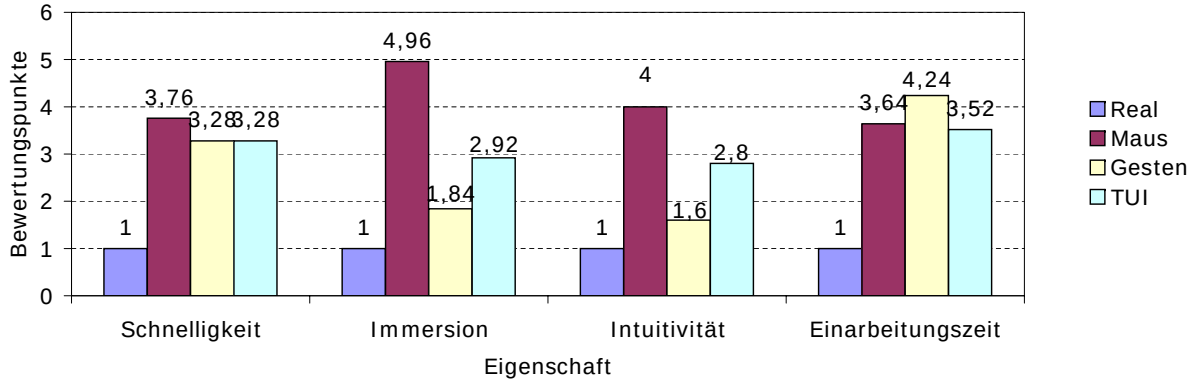


Abbildung 7.12: Auswertung des Fragebogenteils der ersten beiden Versuche I

Die Bewertung der realen Interaktion erreichte bei der Schnelligkeit, der Immersion, der Intuitiviät, dem Komfort sowie der mentalen und physischen Beanspruchung erwartungsgemäß immer den besten Wert. Die Bewertung der subjektiv wahrgenommenen Schnelligkeit (Maus/Keyboard mit 3,76, Gesten und TUI mit je 3,28) unterscheidet sich bei den Interaktionsverfahren kaum. Immersion und Intuitivität sind bei Maus/Keyboard deutlich höher bewertet (4,96 und 4) als bei Gesten (1,84 und 1,6) und TUI (2,92 und 2,8). Dies lässt sich darin begründen, dass keine direkte Verbindung zwischen den Aktionen des Nutzers und den

Reaktionen der Objekte vorhanden waren. Ebenfalls zeigt sich, dass die Interaktion mittels Gesten beinahe den Immersionsgrad und die Intuitivität der realen Interaktion erreicht. Dies ist hier auf die Verbindung von Aktionen und Reaktionen zurückzuführen, da Objekte, wie aus der realen Interaktion gewohnt, mit der eigenen Hand manipuliert werden können. Die Einarbeitungszeit wurde bei allen Methoden ähnlich bewertet (Maus mit 3,64, Gesten mit 4,24 und TUI mit 3,52).

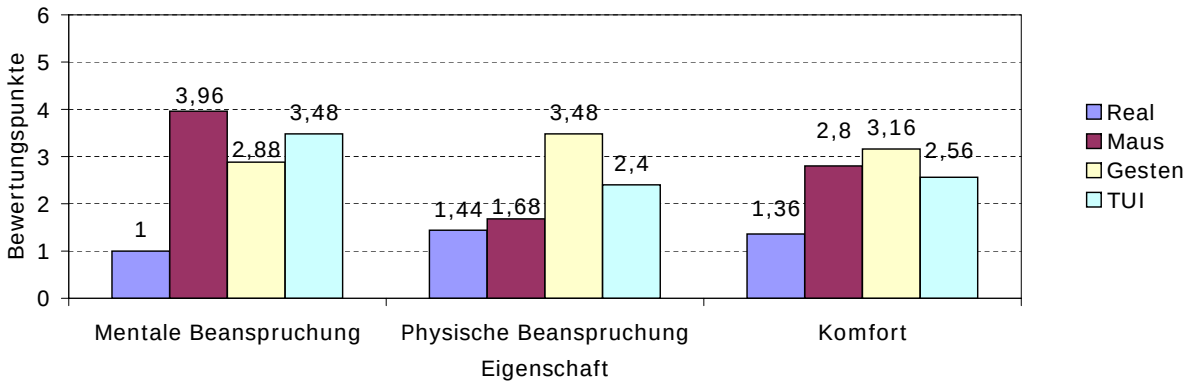


Abbildung 7.13: Auswertung des Fragebogenteils der ersten beiden Versuche II

Die Bewertung der mentalen Beanspruchung war bei der Verwendung von Maus/Keyboard mit 3,96 am höchsten. Die geringste mentale Beanspruchung verzeichnet die Interaktion mittels Gesten mit 2,88, das TUI wurde hier mit 3,48 bewertet. Dagegen ist die physische Beanspruchung bei Maus/Keyboard mit einer Wertung von 1,68 deutlich geringer, als die beiden anderen Methoden (Gesten mit 3,48 und TUI mit 2,4). Die mentale Beanspruchung steigt bei der Verwendung der Maus/Keyboard-Kombination stark durch die Verknüpfung verschiedener Tastenkombinationen an. Ähnlich wie bei der Verwendung des TUI muss der Nutzer sich bestimmte Befehle merken, um die Aufgaben zu lösen. Dagegen erfordert die Verwendung von Gesten eine erhöhte physische Beanspruchung, da der Nutzer zum einen seine Hand immer in der Luft halten muss und er zum anderen die Objekte nicht aus der Entfernung verändern kann. Er muss sich also zu einem Objekt hin bewegen, um eine Manipulation auszuführen, was bei den anderen beiden Methoden auch aus der Ferne geschehen kann. Daher wird diese Methode auch mit dem schlechtesten Wert für Komfort (3,16) bewertet. Die anderen Interaktionsmethoden schneiden etwa gleich ab (Maus mit 2,8 und TUI mit 2,56).

Die Bewertungen des Versuchs 3 zeigen ein ähnliches Bild. Auch hier gewinnt die reale Interaktion in allen Eigenschaften (vgl. Abb. 7.14 und 7.15).

Die subjektive Schnelligkeit wird im Gegensatz zu den ersten beiden Versuchen äquivalent der realen Geschwindigkeit eingeschätzt, nämlich Gesten als schnellste Interaktionsmöglichkeit anschließend die Maus und als langsamste das TUI. Auffallend ist die Bewertung der mentalen Beanspruchung für diesen Versuch. Aufgrund der höheren Komplexität der Anwendung, die sowohl den Einsatz von Rotation als auch von Translation erfordert, liegt die mentale Beanspruchung mit 4,2 beim TUI höher als bei der Maus mit 3,6. Die vielen Tastenkombinationen, die vom Benutzer ausgewählt werden müssen, um die gewollte Manipulation auszuführen, erfordern ein hohes Maß an Konzentration. Die restlichen Eigenschaften Immersion, Intuitivität, physische Beanspruchung und Komfort wurden ähnlich eingeschätzt, wie nach den ersten beiden Versuchen. Sowohl Immersion als auch Intuitivität wurden von den Probanden für die Interaktion mit Hilfe des TUIs besser eingestuft als mit der Maus. Die Gesten erreichten

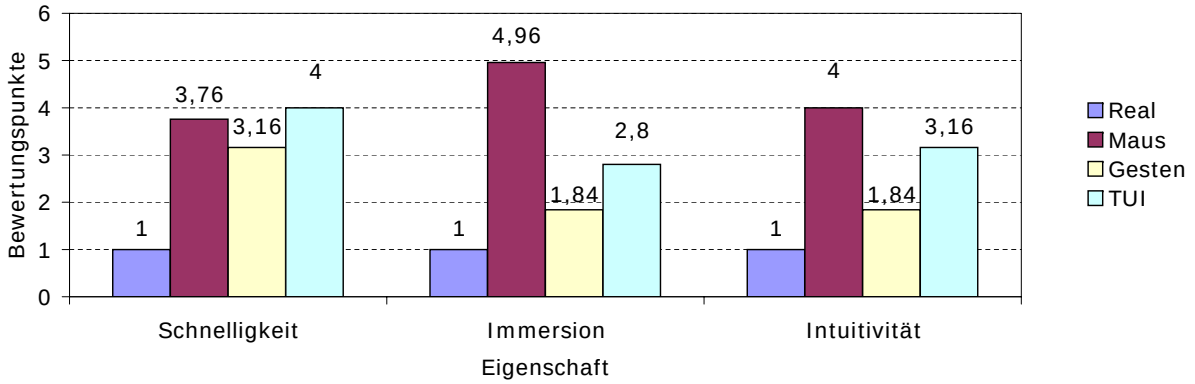


Abbildung 7.14: Auswertung des Fragebogenteils zum dritten Versuch I

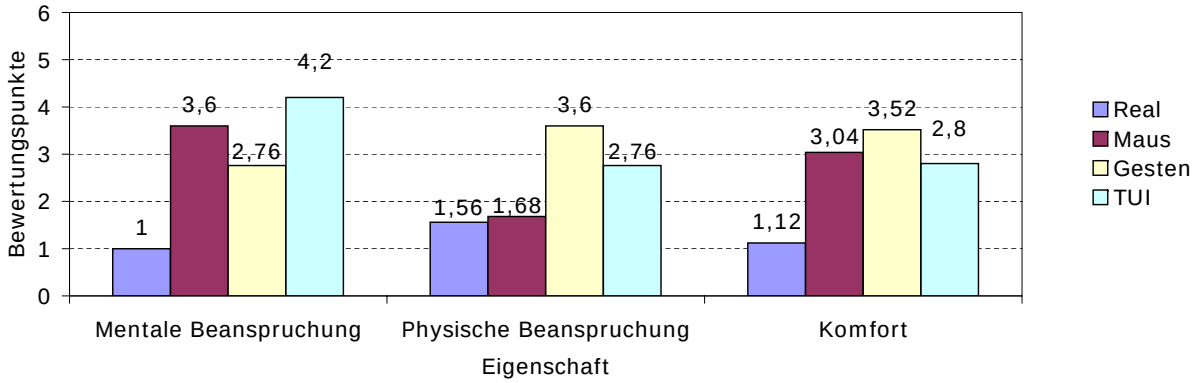


Abbildung 7.15: Auswertung des Fragebogenteils zum dritten Versuch II

wiederum das beste Ergebnis. Im Hinblick auf die physische Beanspruchung und den Komfort ist erneut der Vorteil der Maus und des TUIs gegenüber den Gesten zu erwähnen.

Abbildung 7.16 verdeutlicht abschließend die unterschiedliche Bewertung der Schnelligkeit und mentalen Beanspruchung der Interaktion mithilfe des TUIs nach den jeweiligen Versuchen. Beide Eigenschaften wurden aufgrund der Komplexität des letzten Tests um mehr als einen halben Bewertungspunkt schlechter eingestuft. Ein Vergleich der anderen Eigenschaften nach den jeweiligen Versuchen zeigt keine bzw. nur geringe Unterschiede.

Testen der implementierten Anwendungen
Der Umgang mit der virtuellen Karte fiel den Probanden auf Anhieb sehr leicht, weshalb Navigieren in der Karte mit dem Referenzpunkt keine Probleme darstellte. Es wurde vor allem die intuitive und einfache Steuerung der Karte mithilfe des TUIs hervorgehoben. Kritisiert hingegen wurde die schlechte Auflösung der Karte, die das Lesen der Straßennamen erschwerte. Der Grund dafür lag in der großen Komprimierung der Bilddateien, die zur flüssigen Bewegung der Karten notwendig war. Des Weiteren kritisierten die Anwender die hohen Ladezeiten der Karten beim Hinein- bzw. Herauszoomen, welche auf die trotz Komprimierung noch sehr großen VRML-Dateien zurückzuführen sind.

In der zweiten Anwendung gab es einige Versuchspersonen, die mit der Steuerung des virtuellen Autos nicht von Anfang an zurecht kamen und sich erst daran gewöhnen mussten. Die Probanden hatten Probleme mit der Lenkung des Fahrzeugs, da ihnen unklar war, wie sie das TUI kippen sollten, um eine gewünschte Richtungsänderung des Fahrzeugs zu erreichen.

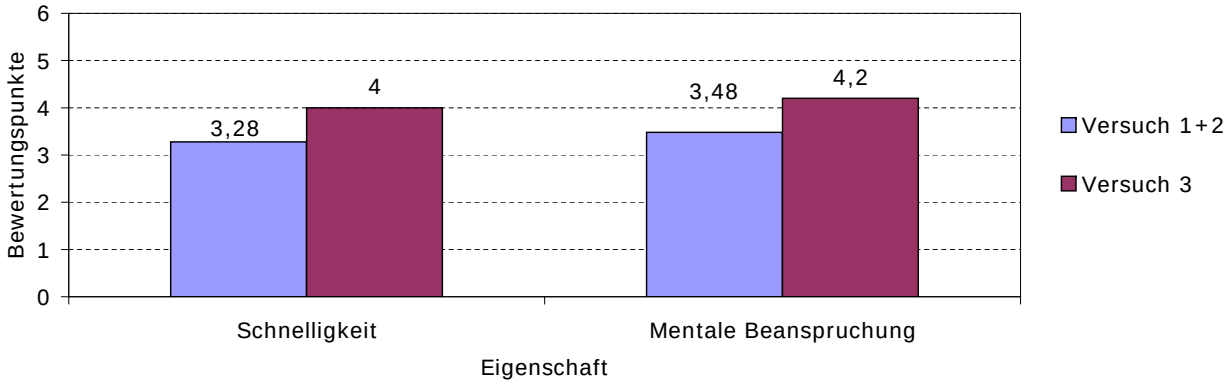


Abbildung 7.16: Vergleich der Bewertung der Schnelligkeit und mentalen Beanspruchung nach Versuch 1 + 2 und Versuch 3

Deshalb wurde vorgeschlagen, die Geschwindigkeit mithilfe der Tasten steuern zu können. Trotz der anfänglichen Schwierigkeiten bei der Steuerung des Autos bereitete die Anwendung den Probanden eine Menge Spaß. Durch die VA wurde die Anwendung als sehr realistisch eingestuft.

7.3.3 Diskussion der Ergebnisse

Die Untersuchung hat gezeigt, dass das TUI im Vergleich zu Maus/Keyboard oder den Gesten vor allem für komplexe Anwendungen mehr Zeit in Anspruch nimmt, da keine direkte Auswahl von Objekten möglich ist. Dennoch bietet dieses Interface bei der Manipulation von virtuellen Objekten in der AR den Vorteil eines höheren Maßes der Immersion gegenüber der Maus/Keyboard-Kombination. Die Interaktion mit der Maus, bei welcher der Anwender an den Arbeitsplatz gebunden ist, vermittelt den Eindruck, wie bei GUIs, auf einem Monitor zu agieren anstatt in einer 3-D-Szene integriert zu sein. Ebenfalls hat sich gezeigt, dass dieses Interface ein höheres Maß an Intuitivität aufweist als eine Kombination von Maus und Keyboard.

Gegenüber der Manipulation virtueller Objekte durch Gesten hat das TUI vor allem Vorteile in Bezug auf physische Belastung und Komfort, da zum Agieren keine großen Bewegungen notwendig sind. So muss der Nutzer bei der Interaktion mittels Gesten an den Ort des virtuellen Objektes laufen und kann dieses Objekt nicht aus der Ferne auswählen. Ebenfalls bietet es diesem System gegenüber den Vorteil, dass der Nutzer keine zusätzliche Hardware tragen muss, die sich negativ auf die Belastung auswirkt. Die Unabhängigkeit eines externen Trackingsystems stellt einen weiteren Vorteil des TUIs dar.

Das TUI bietet also eine gute Alternative zur Interaktion mit Gesten, vor allem für spezifische Anwendungen, die nicht alle sechs möglichen Freiheitsgrade benötigen oder für die eine direkte Manipulation nicht sinnvoll erscheint. Die vier Tasten, die für beliebige Aktionen verwendet werden können, bieten ein hohes Maß an Flexibilität.

7.4 Virtuelle Akustik als Ausgabemedium

Mithilfe einer Untersuchung sollte das in Kapitel 3.3 vorgestellte System zur Erzeugung von VA evaluiert werden. Dabei wurden zwei Schwerpunkte unterschieden: In Teil 1 sollte der

Einfluss auf die Wahrnehmung von AR-Umgebungen gemessen werden, in Teil 2 dagegen der Einfluss von AR auf die Lokalisation von virtuellen Schallquellen.

Im ersten Teil sollte untersucht werden, ob die Verwendung von verschiedenen Arten der akustischen Darstellung (ohne Schall, „Mono-Schall“, „Stereo-Schall“, VA) Einfluss auf bestimmte Eigenschaften der Wahrnehmung der Testpersonen hat. Relevante Eigenschaften waren hierbei die Realitätstreue, d.h. wie realistisch die Darstellung insgesamt wahrgenommen wurde, die Präsenz des virtuellen Objektes, die durch das Objekt verursachte geistige Beanspruchung, Ablenkung, die Spannung, die die Szene vermittelte, und wie stark der Zusammenhang zwischen Objekt und Schall empfunden wurde, d.h. ob ein Objekt zusammen mit virtuellem Schall als eine Einheit wahrgenommen wurde.

Der zweite Teil konzentrierte sich auf den Einfluss der Darstellung virtueller Objekte auf die Wahrnehmung von virtuellen Schallen. Ziel dieses Versuches war es, den Einfluss von zum Schall passenden und nicht passenden virtuell sichtbaren Objekten auf die Lokalisation des Schalls herauszufinden. Der Versuch wurde so ausgelegt, dass hierbei die Lokalisation gestört und auch unterstützt werden konnte. Vor dem Versuch wurde die Vermutung aufgestellt, dass die Versuchspersonen den Schall an der Stelle des passenden Objektes lokalisieren würden, obwohl die Schalle in gewissen Abständen zum Objekt positioniert sind. Das virtuelle Objekt zieht also die „Aufmerksamkeit“ der Schallquelle auf sich.

7.4.1 Versuchsaufbau/-ablauf

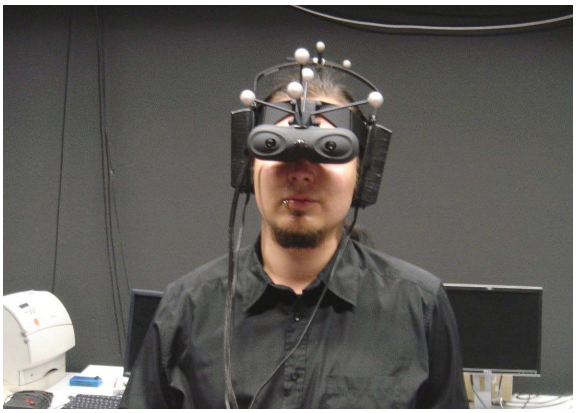
Die Evaluierung wurde im AR-Labor des Lehrstuhls durchgeführt und war in elf Durchgänge unterteilt (vgl. Abb. 7.17)

Einführung	
Durchgang 1	visuell: Hubschrauber / akustisch: kein Schall / keine Wiedergabe
Durchgang 2	visuell: Hubschrauber / akustisch: Hubschrauber / Mono-Wiedergabe
Durchgang 3	visuell: Hubschrauber / akustisch: Hubschrauber / Stereo-Wiedergabe
Durchgang 4	visuell: Hubschrauber / akustisch: Hubschrauber / Virtuelle Akustik
Durchgang 5	visuell: Hubschrauber / akustisch: Bienenschwarm / Mono-Wiedergabe
Durchgang 6	visuell: Hubschrauber / akustisch: Bienenschwarm / Stereo-Wiedergabe
Durchgang 7	visuell: Hubschrauber / akustisch: Bienenschwarm / Virtuelle Akustik
Durchgang 8	visuell: kein Objekt / akustisch: Hubschrauber / Virtuelle Akustik
Durchgang 9	visuell: rote Kugel / akustisch: Hubschrauber / Virtuelle Akustik
Durchgang 10	visuell: Libelle / akustisch: Hubschrauber / Virtuelle Akustik
Durchgang 11	visuell: Hubschrauber / akustisch: Hubschrauber / Virtuelle Akustik

Abbildung 7.17: Versuchsablauf der Evaluierung der Virtuellen Akustik

Um den Einfluss verschiedener akustischer Darbietungen zu bewerten, wurde der Versuchsperson eine virtuelle Szene mit einem Hubschrauber dargestellt, bei dem zu Beginn keine akustische Unterstützung erfolgte. Darauf folgend wurde jede der drei akustischen Methoden mit jeweils einem Schall, der zum Objekt passte (Hubschraubergeräusch), und einem Schall, der nicht dazu passte (Geräusch eines Bienenschwarms), abgespielt. Hierbei wurde die akustische Darbietung in jedem Schritt verbessert, beginnend mit der Mono-Darbietung, bei der der Schall innerhalb des Kopfes wahrgenommen wird (also keine Richtungsinformation berücksichtigt wird), gefolgt von der Stereo-Darbietung (die nur das dem Objekt zugewandte Ohr mit Schall versorgte), bis hin zur VA, die die Richtung des virtuellen Schalls simulierte.

Die Versuchspersonen waren hierzu etwa mittig im Bereich des Trackingsystems platziert und trugen ein HMD zur visuellen Darstellung der virtuellen Szene sowie einen Kopfhörer, der die akustische Darbietung der virtuellen Schalle übernahm. Abb. 7.18 zeigt hier eine Versuchsperson mit HMD und Kopfhörer sowie die Szene, die zur Untersuchung dargestellt wurde.



(a) Versuchsperson für Untersuchung Virtuelle Akustik

(b) Darstellung Hubschrauber für Untersuchung der Virtuellen Akustik

Abbildung 7.18: Ansichten des Evaluierungsszenarios der Virtuellen Akustik

Während des Versuches kreiste der virtuelle Hubschrauber um die Versuchsperson, die sich frei im Raum bewegen konnte. Ebenfalls konnte die Versuchsperson ihren Kopf frei bewegen, um der Flugrichtung des Hubschraubers bei Bedarf folgen zu können.

Nach jedem Durchlauf bewerteten die Versuchspersonen auf einer Skala von 1 (sehr positiv) bis 6 (sehr negativ) die Eigenschaften nach ihrer subjektiven Wahrnehmung.

Für den zweiten Teil wurde derselbe Versuchsaufbau verwendet. Lediglich die Darstellung wurde verändert. So wurden im ersten Durchlauf kein virtuelles Objekt, im zweiten eine rote Kugel, im dritten eine Libelle und im vierten der virtuelle Hubschrauber dargestellt. Akustisch wurde zu jedem der Objekte der Schall eines fliegenden Hubschraubers dargeboten. Dieser wurde jedoch für jedes Objekt nicht nur auf die exakt gleiche Position gelegt, sondern auch mit Abweichungen von 10°, 20° und 40° in beide Richtungen zum Objekt abgespielt. Die Versuchspersonen blickten daraufhin in die Richtung aus der sie den Schall wahrnahmen, um zu messen, welche Abweichung von der realen Position zu der wahrgenommenen Position vorlag. Insgesamt lokalisierte jede Versuchsperson also sieben Positionen pro Objekt.

7.4.2 Ergebnisse

An der Untersuchung nahmen insgesamt 12 Versuchspersonen im Alter von 21 bis 55 Jahren teil. Das durchschnittliche Alter lag bei 28 Jahren.

Teil 1: Bewertung der AR
Abb. 7.19 zeigt die Bewertung der Eigenschaften bei den verschiedenen akustischen Darbietungsarten, wenn der Schalles eines fliegenden Hubschraubers zu hören ist. Dagegen ist in Abbildung 7.20 die Bewertung dieser Eigenschaften bei der Verwendung eines Schalles, der einen fliegenden Bienenschwarm, aufgeführt.

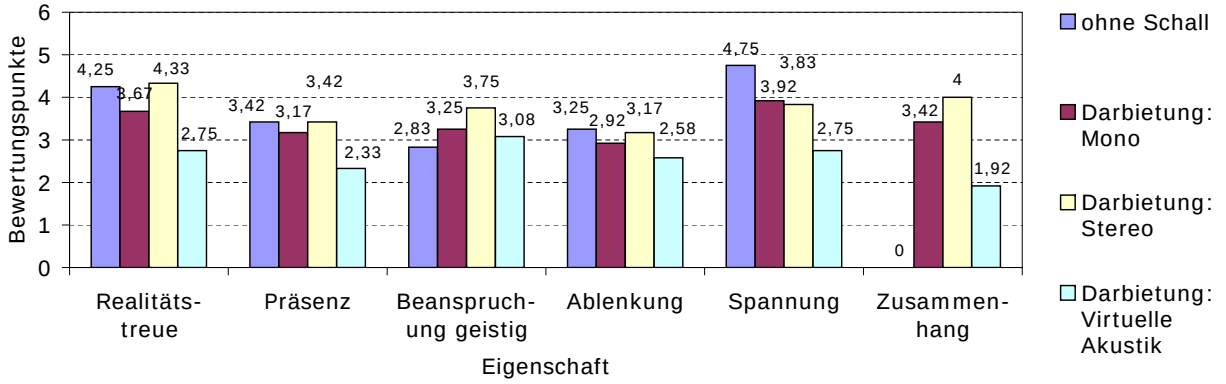


Abbildung 7.19: Bewertung der Szenarien mit Hubschrauberschall

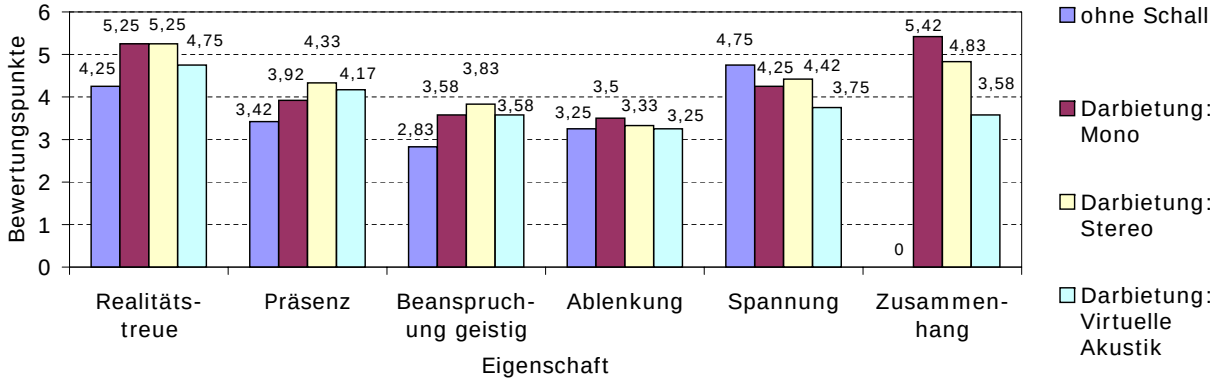


Abbildung 7.20: Bewertung der Szenarien mit Bienenschwarmschall

Es wird deutlich, dass die Darbietung mit dem zum Objekt passenden Hubschraubergräusch besser wahrgenommen wurde als die mit dem nicht passenden Geräusch. Die Realitätstreue wurde hier mit einem Wert von 4,25 bewertet, für den Fall, dass kein Schall abgespielt wurde. Die Realitätstreue bei Verwendung des Hubschrauberschalles reicht von 3,67 für die Mono-Darbietung über 4,33 für die Stereo-Darbietung bis hin zu 2,75 für die VA. Die VA vermittelt also den realitätsnähesten Eindruck der gesamten Szene. Im Vergleich dazu zeigt sich bei der Verwendung des Bienenschwarmschalles, dass auch hier die VA den höchsten Grad an Realitätsnähe erzeugt. Diese liegt jedoch zwischen ein bis zwei Bewertungsstufen unter der Bewertung zuvor.

Die Bewertung der Präsenz des virtuellen Objektes zeigt einen ähnlichen Verlauf. Die Präsenz ohne Schall liegt bei 3,42. Bei Mono-Darbietung beträgt sie 3,17, bei Stereo-Darbietung 3,42 und bei VA 2,33. Die VA erzeugt hier also auch die höchste Präsenz des virtuellen Objektes. Diese liegt für jede Schallart stets niedriger als bei der Verwendung des Bienenschwarmgeräusches. Hier liegen die Bewertungen bei 3,92 im Mono-Betrieb, 4,33 für Stereo-Darbietung und 4,17 für VA. Die Darstellung eines objektfremden Schalles beeinflusst die Präsenz des virtuellen Objektes also deutlich in negativer Form.

Die Bewertung der geistigen Beanspruchung zeigt, dass die Darbietung von Schallen in jedem Fall die Beanspruchung erhöht. So wird diese ohne Schall mit 2,83 bewertet, für die Verwendung von Hubschrauber- bzw. Bienenschwarmschall liegt sie dagegen bei Werten zwischen 3,83 und 3,08. Beiden Schallen jedoch ist gemein, dass die Beanspruchung bei Stereo-

Darbietung die höchste ist (Wert von 3,75 bzw. 3,83).

Die Ablenkung, die durch die gesamte Szene verursacht wird, bleibt durchgehend auf einem annähernd gleichen Niveau. Die Bewertung für den Hubschrauberschall liegt zwischen 3,25 ohne Schall bis zu 2,58 bei Verwendung der VA. Bei einem objektfremden Schall liegt dieser Wert zwischen 3,5 und 3,25.

Betrachtet man die durch die Szene erzeugte Spannung, stellt sich heraus, dass die Verwendung eines zum Objekt passenden Schalls stets eine bessere Bewertung erzielt. Durch die Darbietung eines Schalles (auch objektfremd) steigt die Spannung der Szene immer an. Die VA erzielt hierbei eine Wertung von 2,75 bzw. 3,75 und ist damit deutlich positiver bewertet, als die Verwendung von Mono-Darbietung (3,92 und 4,25) oder Stereo-Darbietung (3,83 und 4,42). Ebenfalls zeigt sich, dass die Wahl, ob Mono- oder Stereo-Darbietung, keinen großen Effekt auf die Spannung hat.

Der Zusammenhang zwischen Schall und Objekt ist bei der VA am höchsten. Selbst mit der Bewertung von 3,58 für den objektfremden Schall liegt diese deutlich vor den beiden anderen Methoden mit Werten von 5,42 bzw. 4,83. Mit 1,92 erreicht hier die VA bei der Darbietung des zum Objekt passenden Schalles einen Höchstwert.

Abb. 7.21 stellt nochmals die Wertungen der Eigenschaften im direkten Vergleich zwischen Hubschrauberschall und Bienenschwarmschall gegenüber, wobei hier auch die Unterscheidung zwischen der Mono-Darbietung und der VA getroffen wird.

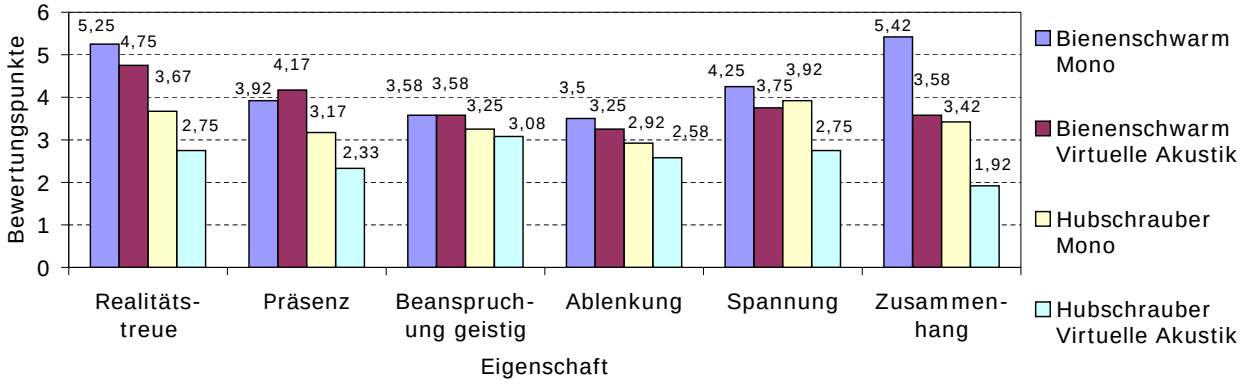


Abbildung 7.21: Vergleich der Szenarien mit unterschiedlichen Schalldarbietungen

Die Bewertung des zum Objekt passenden Schalles liegt also stets niedriger. Mit Ausnahme der Präsenz beim objektfremden Schall erhält die VA immer eine bessere Bewertung als die Stereo-Darbietung des Schalles.

Teil 2: Einfluss AR auf die Wahrnehmung

In Abb. 7.22 sind die Ergebnisse des zweiten Teils der Untersuchung dargestellt. Das Diagramm zeigt den Winkel zwischen Objekt und Schall, den die Versuchspersonen wahrgenommen haben, aufgetragen über den tatsächlichen Abweichungen zwischen dem Objekt und dem Schall.

Die Ergebnisse zeigen, dass die Versuchspersonen den Schall nicht an der Stelle des Objektes lokalisiert haben, wenn dieser nicht auch an der Stelle des Objektes wiedergegeben wurde. Hierbei sind die Mediane in den Bereichen +10° und +20° nur unwesentlich in Richtung der 0° verschoben. In den Bereichen -20° und +20°, bei denen das Objekt noch bei Blick in Richtung des Schalls zu sehen war, sind die Mediane der Lokalisation bei „keinem Objekt“ und dem

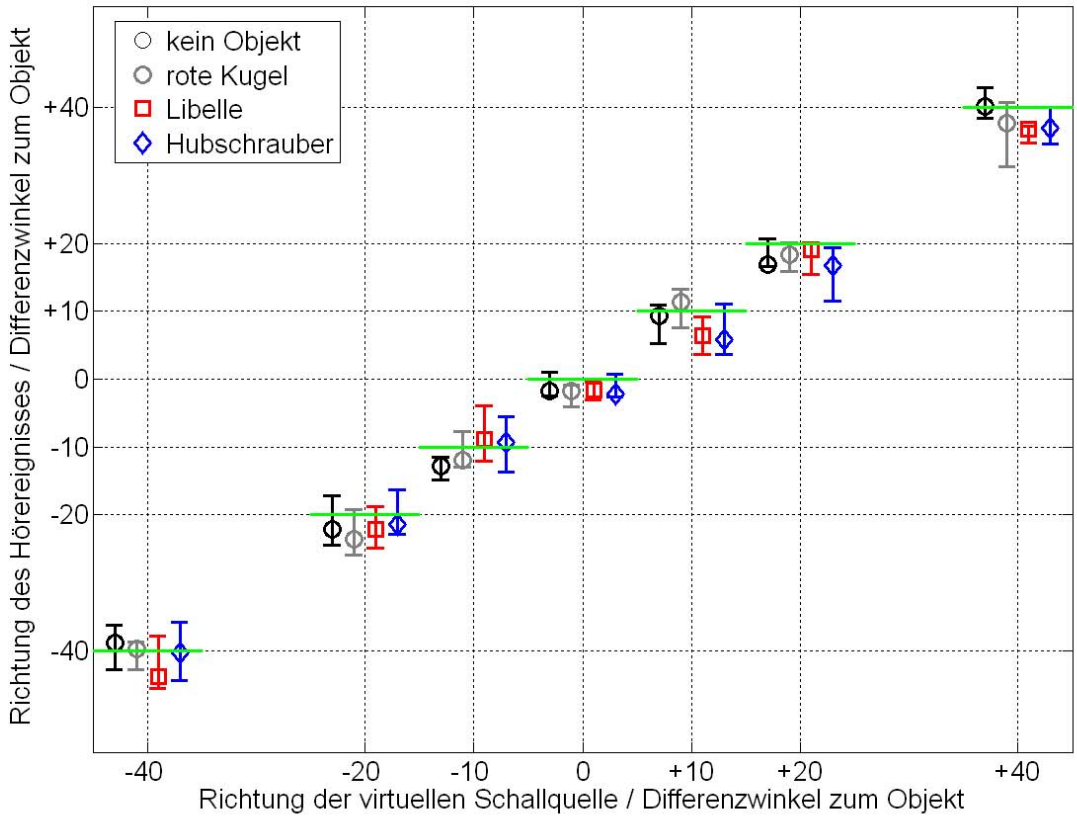


Abbildung 7.22: Darstellung der Ergebnisse der Lokalisation von Schallen mit visuellem Stimulus mit Medianen und Interquartilbereichen

„Hubschrauber“ identisch, sodass hierbei eine Beeinflussung der Lokalisation durch den zum Objekt passenden Schall nicht nachgewiesen werden kann.

Selbst eine minimale Hilfe zur Lokalisation durch die visuellen Objekte lässt sich an den Versuchspersonen nicht nachweisen. Die Interquartilbereiche, wenn Schall und Objekt an derselben Stelle dargestellt werden, sind zwar geringer, jedoch existieren diese auch für die Lokalisation „ohne Objekt“. Dieses Phänomen lässt sich womöglich damit begründen, dass es sich dabei um die Ausgangsblickrichtung und die Ausrichtung der Füße zu Beginn des Versuches handelt.

7.4.3 Diskussion der Ergebnisse

Vergleicht man nun die Ergebnisse der Evaluierung, so stellt man fest, dass die Darstellung der AR durch die Verwendung von Schall verbessert wird. Jedoch muss der verwendete Schall hierbei zu dem virtuellen Objekt passen, das diesen Schall aussendet. Die Untersuchung hat hier klar herausgestellt, dass Schall, der nicht zum Objekt passt, einen negativen Effekt auf die Wahrnehmung des Nutzers hat. Die Realitätstreue und Präsenz der AR-Szene werden in diesem Fall schlechter wahrgenommen als eine Darstellung, die auf Schall gänzlich verzichtet.

Die Verwendung von VA steigert diese Werte der Realitätstreue und Präsenz im Vergleich zu einer reinen Mono- oder Stereo-Darbietung stark. So erhielt in der Untersuchung das Szenario mit VA und Hubschrauberschall in diesen Eigenschaften eine deutlich niedrigere

Bewertung als die beiden anderen Methoden zur Darbietung von Schallen.

Es zeigt sich also, dass die Art der Schallwiedergabe und die Wahl des Schalles einen starken Einfluss auf die Wahrnehmung der Realitätstreue und Präsenz einer AR-Szene hat.

Betrachtet man die Messungen der Lokalisation von Schallquellen in Abhängigkeit von der Darstellung virtueller Objekte, so wird in dieser Untersuchung kein Effekt deutlich. Die Lokalisation von Schall wurde in dieser Evaluierung nicht von der Darstellung eines virtuellen Objektes beeinflusst. Weder Objekte mit passendem Schall noch Objekte mit nicht zum Objekt passendem Schall konnten die Lokalisation der virtuellen Schallquelle in Richtung der Position des virtuellen Objektes beeinflussen.

7.5 Videokonferenzsystem

Das in Kapitel 5.2 entwickelte System zur Integration eines Videokonferenzsystems in eine AR-Szene wurde in einer Evaluierung untersucht. Hierbei sollten verschiedene Anzeigearten für die Videokonferenz verglichen werden. Als grafische Visualisierung stehen ein Monitor und ein HMD zur Verfügung, als zugrunde liegendes Trackingsystem kann ein bildbasiertes Verfahren anhand von Papiermarkern oder IR-Targets verwendet werden. Zusätzlich kann die Darstellung des Gesprächspartners mit Darstellung des Hintergrundes oder mit segmentiertem Gesprächspartner ohne Hintergrund erfolgen. Alle daraus resultierenden Kombinationen wurden auf bestimmte Eigenschaften rein subjektiv durch Versuchspersonen hin bewertet. Zu beurteilen waren Eigenschaften, die die Darstellung selbst betreffen, sowie Eigenschaften, die den Umgang mit dem System und seiner Hardware beschreiben. Zur Darstellung wurden die Bildqualität (wie etwa Auflösung oder Pixelfehler), der Realitätseindruck (also, wie real der Gesprächspartner erscheint) sowie die empfundene Präsenz (also die empfundene Anwesenheit) des Gegenüber untersucht. Zum Umgang mit dem System wurde der Komfort bewertet, wie angenehm also die Kombination aus Hardwarekomponenten ist, sowie das Handling, d.h. wie unproblematisch das System zu bedienen ist. Zusätzlich wurde noch die subjektive Empfindung des Zeitversatzes zwischen Audio- und Videoinformationen abgefragt.

7.5.1 Versuchsaufbau/-ablauf

Die Untersuchung fand im AR-Labor des Lehrstuhles statt, wofür es an die erforderlichen Vorgaben angepasst wurde. So wurde hier die Videokonferenz als unidirektionales System aufgebaut. Das heißt, der Versuchsleiter war selbst der Gesprächspartner der Versuchspersonen, erhielt jedoch kein Videobild der Versuchsteilnehmer und befand sich selbst nicht innerhalb einer AR-Umgebung. In der AR-Umgebung befand sich also lediglich die Testperson, die den Versuchsleiter als Gesprächspartner in seine reale Umgebung virtuell eingeblendet bekam. Um den Versuchsleiter und die Testpersonen innerhalb des Labors räumlich voneinander zu trennen, wurde eine Trennwand in das Labor eingebracht, die den direkten visuellen Kontakt zwischen den beiden Gesprächspartnern unterband. Da das System keine akustischen Signale übertragen konnte, wurde für die Untersuchung der direkte Weg zwischen den Teilnehmern gewählt, die Gesprächspartner kommunizierten somit also nicht über ein technisches Übertragungssystem, sondern durch die Trennwand. Das System war so eingestellt, dass das Gesicht und ein kleiner Teil des Oberkörpers des Versuchsleiters im Videobild sichtbar waren.

Die Untersuchung war in acht Durchgänge unterteilt (vgl. Abb. 7.23), die sich jeweils in der Kombination der Hardware unterschieden. Die ersten vier Durchläufe erfolgten mit

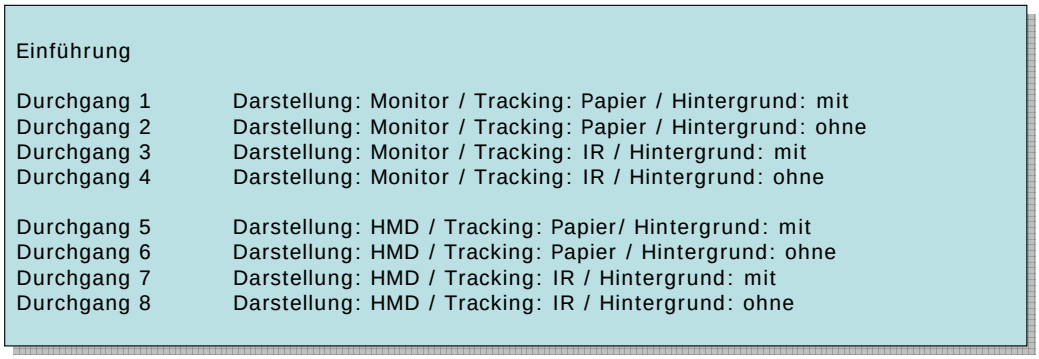


Abbildung 7.23: Versuchsablauf der Evaluierung der Videokonferenz

einem Monitor, die letzten vier mit dem HMD als visuelles Ausgabesystem. Jeweils die ersten beiden dieser Versuche wurden mittels eines Papiermarkers, die letzten beiden mittels eines IR-Targets durchgeführt. Abb. 7.24 zeigt hier den Unterschied im Systemaufbau bei der Verwendung eines Monitors und bei Verwendung des HMDs.

Bei der Darstellung des Gesprächspartners konnte der Hintergrund vom Teilnehmer abgetrennt werden. Auf diese Weise erschien nur noch der Gesprächspartner in der AR-Umgebung. Abb. 7.25 zeigt hier das Videobild mit und ohne Hintergrund bei Verwendung des IR-Targets und bei der Nutzung eines Papiermarkers.

Bei jedem der acht Durchläufe führte der Versuchsleiter einen Dialog mit dem Versuchsteilnehmer über eine Dauer von 3 Minuten. Dabei sollte die Versuchsperson das System erfahren sowie seine Vor- und Nachteile kennenlernen. Der Versuchsleiter besprach hierbei Themen, bei denen Gesten notwendig waren (Verdeutlichung von Größen, etc.), um die Möglichkeiten einer Videokonferenz auszunutzen.

Im Anschluss an jeden Durchlauf bewertete die Versuchsperson das vorgestellte System mithilfe eines Fragebogens hinsichtlich bestimmter Eigenschaften. Hierzu zählten die Eigenschaften Komfort und Handling, die den Umgang mit den System beschrieben, die Qualität des Systems hinsichtlich der Bildqualität und der Audioeigenschaften sowie der wahrgenommene Realitätseindruck und die Präsenz. Dazu diente eine Bewertungsskala mit Werten von 1 (sehr positiv) bis 6 (sehr negativ). Die Eigenschaften Komfort, Handling und Audio mussten bei zwei aufeinander folgenden Versuchen, die sich lediglich durch die Darstellung oder Ausblendung des Hintergrunds unterschieden, nicht bewertet werden, da dies auf sie keinen Einfluss hatte.

In einer Einführung wurde den Versuchsteilnehmern vor den Durchgängen ein Überblick über AR und die nachfolgenden Systemaufbauten gegeben. Ebenfalls erhielten sie hierbei einen Überblick über die zu beurteilenden Eigenschaften und die zugrunde liegende Skala zur Bewertung.

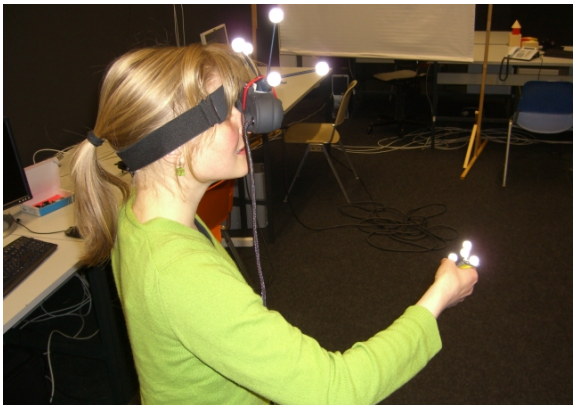
7.5.2 Ergebnisse

Die Versuche wurden mit insgesamt zehn Personen durchgeführt. Das Alter der Versuchsteilnehmer lag zwischen 25 und 51 Jahren. Das Durchschnittsalter betrug 33 Jahre. 50% der Teilnehmer waren Frauen, 50% Männer. 40% gaben an, dass ihnen AR bereits vor diesem Versuch ein Begriff war. Von diesen 40% würden sich 50% als Anfänger auf diesem Gebiet bezeichnen, 50% hatten bereits so viel Umgang mit einem AR-System, dass sie sich selbst

(a) Monitor und Papiermarker

(b) Monitor und IR-Target

(c) HMD und Papiermarker



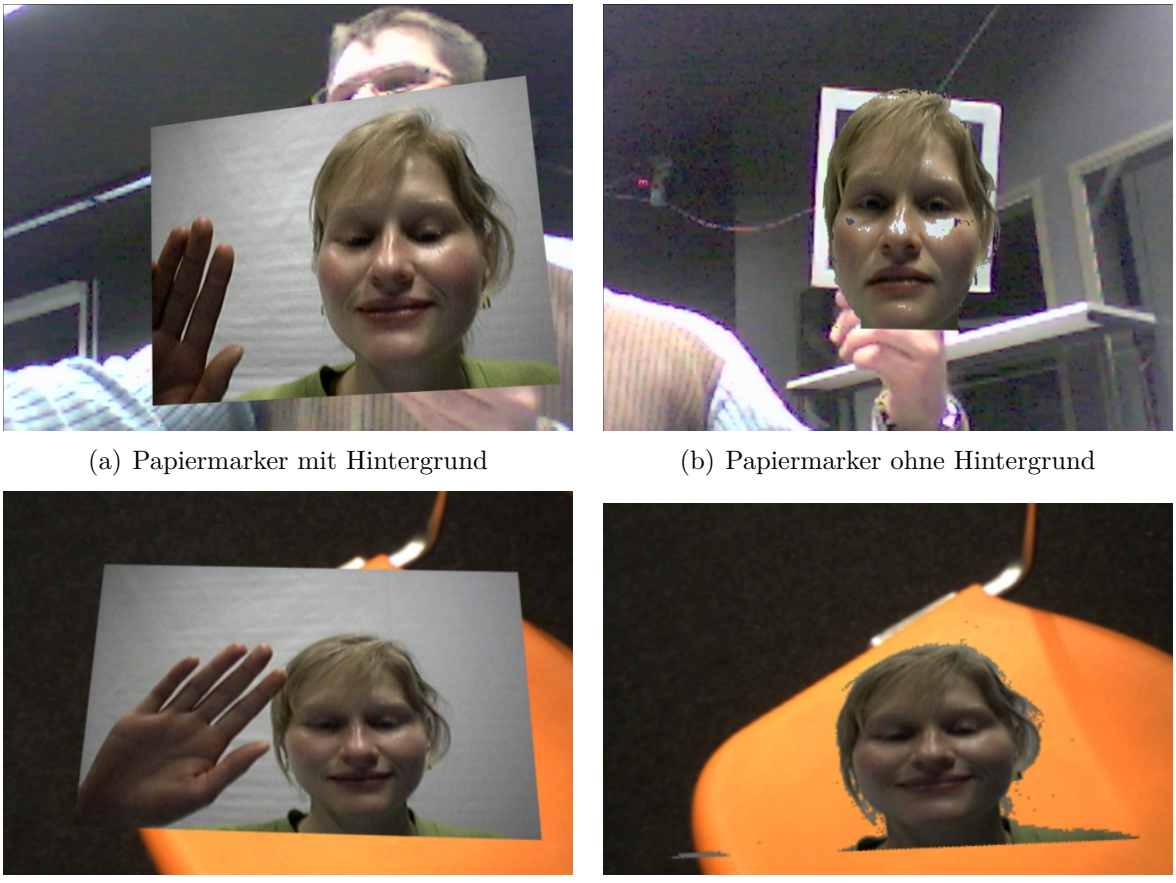
(d) HMD und IR-Target

Abbildung 7.24: Videokonferenz mit unterschiedlichen Systemkombinationen

als Fortgeschrittene einstufen. Kein Teilnehmer schätzte sich selbst als Profi ein. 70% der Versuchsteilnehmer hatten bisher noch nie ein Videokonferenzsystem angewandt und auch an keiner Videokonferenz teilgenommen. 20% gaben an, selten an Videokonferenzen teilzunehmen, und 10% nutzten regelmäßig ein solches System.

Abb. 7.26 und 7.27 stellen die Ergebnisse der Fragebögen dar. Im Folgenden werden nun die Ergebnisse für die einzelnen Versuche mit den acht verschiedenen Systemaufbauten präsentiert.

Versuch 1: Monitor, Papiermarker, mit Hintergrund
Die Versuchspersonen bewerten die Bildqualität des übertragenen Bildes mit 2,5. Die Auflösung wurde von allen als ausreichend empfunden, um auch die Mimik des Gesprächspartners erkennen zu können. Mit einer durchschnittliche Bewertung von 3,6 wurde der Realitätseindruck als nicht optimal empfunden, wofür hauptsächlich die etwas „ruckartigen“ Bewegungsabläufe im Bild und der Papiermarker im Hintergrund verantwortlich waren. Die Versuchspersonen hatten auch nicht den Eindruck, der Gesprächspartner würde sich tatsächlich im Raum befinden, was sich in der Benotung der Präsenz mit 3,7 ausdrückt. Komfort und Handling dieser Kombination wurden mit 3,5 und 4,3 gewertet, woraus sich schließen lässt, dass der Umgang mit diesem System als nicht gut empfunden wurde. Negativ im Umgang mit dem Papiermarker ist den Teilnehmern vor allem aufgefallen, dass dieser stets voll im Kamerabild sein muss, also nicht zufällig von einem Finger der haltenden Hand verdeckt werden darf, und



(a) Papiermarker mit Hintergrund

(b) Papiermarker ohne Hintergrund

(c) IR-Target mit Hintergrund

(d) IR-Target ohne Hintergrund

Abbildung 7.25: Videokonferenz mit und ohne Hintergrund

generell eine wenig robuste Erkennung lieferte. Bei der Tonübertragung gab hauptsächlich die Tatsache, dass das Gesehene dem Gehörten erst mit einem zeitlichen Versatz folgte, den Ausschlag. Dies resultierte aus der Verarbeitung der Bildinformationen, wodurch diese zeitlich nach dem unverarbeiteten, akustischen Signal bei der Versuchsperson ankam. Dass sie den Gesprächspartner nicht aus der Richtung hören konnten, in der sie ihn auch sahen, spielte hingegen nur eine beiläufige Rolle und wurde bei diesem Versuchsaufbau von den meisten gar nicht wahrgenommen.

Versuch 2: Monitor, Papiermarker, ohne Hintergrund
Das Entfernen des Hintergrunds führte zu Löchern im Bild (verursacht durch Lichtverhältnisse, die bestimmte Regionen im Gesprächspartner als Hintergrund erkennen ließen) und zu einer unscharfen Kontur des Gesprächspartners. Diese Defizite zeigen sich in der Bewertung der Bildqualität mit einem Ergebnis von 4. Die Wahrnehmung des Teilnehmers wurde jedoch leicht verbessert. Der Realitätseindruck stieg leicht auf 3,4 und die Präsenz stieg auf 3,1. Hier ist also eine Verbesserung der Wahrnehmung erkennbar, die sich allerdings negativ auf die Qualität des Bildes auswirkte.

Versuch 3: Monitor, IR-Target, mit Hintergrund
Die Qualität des Bildes wurde mit 2,4 ähnlich gut wie in Versuch 1 bewertet. Durch die Verwendung des IR-Targets statt des Papiermarkers konnte der Realitätseindruck im Vergleich

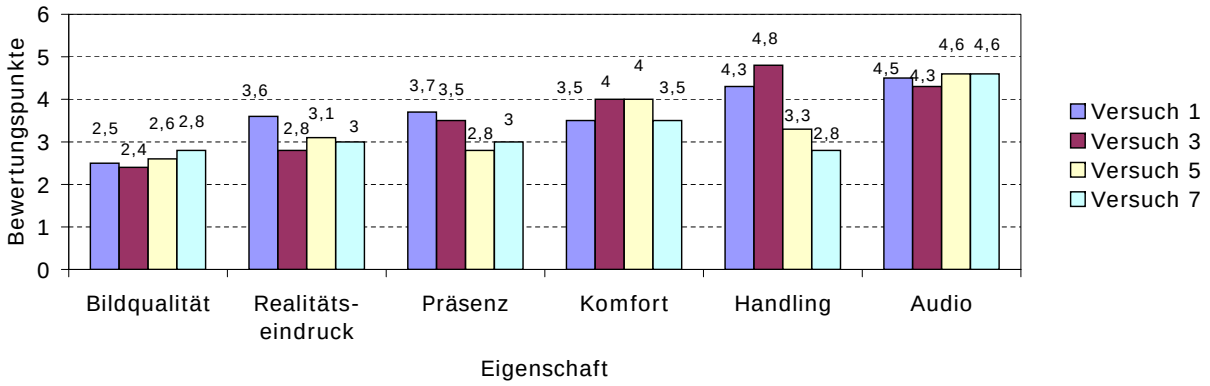


Abbildung 7.26: Ergebnisse Versuche 1, 3, 5 und 7 mit Hintergrund

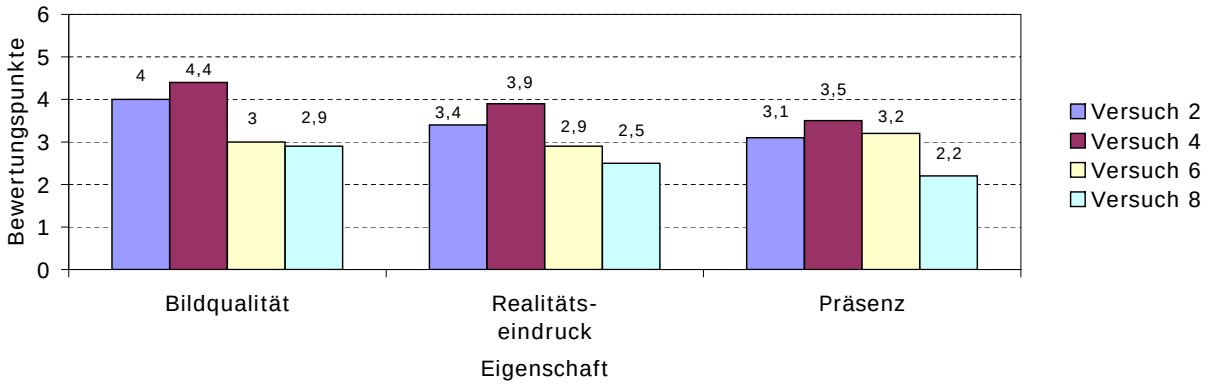


Abbildung 7.27: Ergebnisse Versuche 2, 4, 6 und 8 ohne Hintergrund

zu Versuch 1 mit 2,8 um fast eine ganze Bewertungseinheit gesteigert werden. Auf die Präsenz hingegen hat der Wechsel des Markers jedoch kaum einen Einfluss, was eine ähnliche Bewertung von 3,5 deutlich macht. Dieser spezielle Versuchsaufbau macht es notwendig, dass die Versuchspersonen sowohl das IR-Target als auch die Kamera mit dem sich darauf befindenden Marker so zueinander ausrichten müssen, dass der virtuelle Konferenzteilnehmer auf dem Monitor erscheint. Diese Koordination von zwei Targets gleichzeitig führte bei einigen Versuchsteilnehmern zu größeren Problemen, weshalb der Komfort mit 4 und das Handling mit 4,8 als sehr schlecht eingeschätzt wurden. Die Bewertung der Audioeigenschaften unterscheidet sich mit einem Wert von 4,3 nur unwesentlich von Versuch 1.

Versuch 4: Monitor, IR-Target, ohne Hintergrund
Analog zu Versuch 2 zeigt sich auch hier eine Verschlechterung der Bildqualität auf 4,4. Diese fällt sogar noch gravierender aus als, da nun die Löcher im angezeigten Bild noch deutlicher sichtbar werden und das IR-Target im Gegensatz zum Papiermarker keine neutrale Hintergrundfläche mehr bietet. Dieser Umstand wirkt sich auch auf die Bewertung des Realitätseindrucks aus, der mit 3,9 schlechter beurteilt wurde als in Versuch 2. Ebenfalls sinkt die Präsenz des Gesprächspartners auf 3,5.

Versuch 5: HMD, Papiermarker, mit Hintergrund
Versuch 5 nutzte erstmals das HMD zur visuellen Darstellung. Aufgrund der Bauart sind die in der Brille verwendeten Displays qualitativ minderwertiger als der zuvor verwendete

Monitor. Dies äußert sich beispielsweise in der niedrigeren Auflösung oder der Qualität der Farbwiedergabe. Dennoch wirkte sich diese Darstellung nur wenig auf die Bewertung der Bildqualität aus, die mit 2,6 kaum niedriger ausfällt als die vorhergehenden Versuche 1 und 3. Der Realitätseindruck wird mit 3,1 ähnlich wie in Versuch 3 bewertet, die Präsenz gewinnt hier dagegen deutlich (bewertet mit 2,8). Dies ist darauf zurückzuführen, dass das Sichtfeld mit HMD dem realen Sichtfeld des Nutzers entspricht. Bei Verwendung des Monitors wurde hingegen die Umgebung in Richtung der Versuchsperson dargestellt, da die Kamera in der Nähe des Monitors angebracht war und in Richtung des Benutzers blickte. Durch Verwendung des HMDs war es den Versuchspersonen nun möglich, sich frei im Raum zu bewegen. Diese Bewegungsfreiheit wirkt sich dennoch nicht auf Komfort und Handling aus (Wertung 4 bzw. 3,3), da die Trageeigenschaften des HMDs als äußerst unangenehm empfunden wurden. Das Handling des Papiermarkers wurde von den Versuchsteilnehmern mit 3,3 deutlich besser bewertet als in den vorangegangenen Versuchen. Die Bewertung der Audioeigenschaften mit 4,6 änderte sich auch in diesem Versuch nicht.

Versuch 6: HMD, Papiermarker, ohne Hintergrundbild
Ähnlich der Versuche zuvor ist auch hier eine Verschlechterung der Bildqualität erkennbar, sobald der Hintergrund entfernt wird. Die Bewertung sinkt hier auf 3. Der Realitätseindruck steigt in dieser Kombination auf 2,9, während die Präsenz auf eine Bewertung von 2,9 absinkt.

Versuch 7: HMD, IR-Target, mit Hintergrund
Durch die Verwendung des IR-Targets, statt des Papiermarkers aus Versuch 5, ließ sich kein deutlicher Einfluss des Markers auf die Bildqualität, den Realitätseinduck und die Präsenz nachweisen. Die Werte in diesen Kategorien sind bei beiden Markertypen bei sonst identischem Versuchsaufbau mit Werten von 2,8, 3 und 3 nahezu identisch. Der Komfort des HMDs hat sich bei diesem Versuch mit einem Wert von 3,5 etwas verbessert, was sich durch die langsame Gewöhnung der Testpersonen an das ungewohnte Anzeigegerät erklären lässt. Beim Handling, bewertet mit 2,8, konnte das IR-Target nun aber seine Überlegenheit gegenüber dem Papiermarker deutlich zeigen. Die Audioeigenschaften haben sich, wie erwartet, mit einem Wert von 4,6 nicht verbessert.

Versuch 8: HMD, IR-Target, ohne Hintergrund
Im Gegensatz zu allen vorangegangenen Versuchen hat sich die Bildqualität, bewertet mit 2,9, dieses Mal durch die Entfernung des Hintergrunds nicht wesentlich verschlechtert. Auch der Realitätseindruck konnte mit 2,5 deutlich gesteigert werden. Der Gewinn durch die ausschließliche Darstellung des Gesprächspartners überwog dieses Mal deutlich gegenüber den Einbußen durch ausgefranste Ränder oder Löcher im Bild. Zu diesem Ergebnis passt auch die positive Einschätzung der Präsenz durch die Versuchsteilnehmer. Mit einer Bewertung von 2,2 brachten sie eindeutig zum Ausdruck, dass der Gesprächspartner nun gut in die reale Umgebung integriert war und er tatsächlich anwesend wirkte.

7.5.3 Diskussion der Ergebnisse

Ziel der Untersuchung war es festzustellen, welche der Kombinationen für die Verwendung mit einem Videokonferenzsystem am besten geeignet ist. Außerdem sollte die Evaluierung zeigen, welche Auswirkungen die Wahl bestimmter Eigenschaften auf die Bewertung des Gesamtsystemes hat. Folgende Feststellungen konnten bei der Auswertung der Ergebnisse getroffen werden.

Betrachtet man im Vergleich die Bewertungen von Kombinationen mit und ohne der Darstellung des Hintergrunds, zeigt sich, dass sich die Entfernung des Hintergrunds hinter dem Gesprächspartner in allen Kombinationen negativ auf die empfundene Bildqualität auswirkte. Diese Verschlechterung kann auf die damit verbundenen Fehler (Löcher und unscharfe Kanten des Gesprächspartners) im Bild zurückgeführt werden. Die Bewertung ist hier bei der Verwendung des Monitors deutlich stärker ausgefallen als bei der Verwendung des HMDs. Der Einfluss auf den Realitätseindruck ist dagegen gering. Lediglich bei Versuch 4 (also die Kombination aus Monitor und IR-Target) verschlechtert sich der Realitätseindruck, bei allen anderen Fällen steigt dieser Wert leicht an. Die Präsenz des Gesprächspartners scheint von der Wahl des Hintergrunds nicht abhängig zu sein, da keine eindeutige Tendenz ersichtlich ist. Lediglich bei der Kombination aus HMD und IR-Target ohne Hintergrund zeigt sich eine deutliche Steigerung der Präsenz.

Die Wahl des Targets, mit dem der Gesprächspartner im Raum platziert wird, hat auf die Bildqualität keinen Einfluss. Verwendet man das HMD zur Darstellung, schneidet das IR-Target in Bezug auf den Realitätseindruck leicht besser ab als der Papiermarker. Am Monitor zeigt sich dieser Effekt nur bei der Darstellung des Gegenübers mit Hintergrund. Lediglich bei der Verwendung des IR-Targets im Zusammenspiel mit dem HMD und der Hintergrundentfernung kann dieser Marker seine Überlegenheit in der Kategorie Präsenz zeigen. Diese Konfiguration suggeriert dem Benutzer am meisten, dass der andere Videokonferenzteilnehmer tatsächlich anwesend ist. Dies gelingt bei allen anderen Konfigurationen nur bedingt. Der Papiermarker vermittelt ein höheres Maß an Komfort in Kombination mit dem Monitor, bei Verwendung mit dem HMD dagegen das IR-Target. Betrachtet man das Handling, stellt man fest, dass in Versuch 3 und 4 das IR-Target das schlechteste Ergebnis auf diesem Gebiet im ganzen Versuchsfeld liefert, in Versuch 7 und 8 dagegen das beste. In Bezug auf das Handling lässt sich das IR-Target besser mit dem HMD kombinieren als mit dem Monitor.

Ein Vergleich der Darstellungsart stellt heraus, dass die Bildqualität des HMDs etwas besser bewertet wird als die Darstellung mittels eines Monitors. Die gleiche Tendenz ist auch beim Realitätseindruck erkennbar, wobei das HMD im Durchschnitt etwas besser abschneidet als der Monitor. In der Kategorie Präsenz werden diese Unterschiede in der Bewertung ebenfalls nur bei der Verwendung des Monitors deutlich sichtbar. In Versuch 8 wird mit dem HMD der beste Wert für die Präsenz erreicht. Weder das HMD noch der Monitor kann sich im Bereich Komfort deutlich vom anderen absetzen. Der Gewinn an Bewegungsfreiheit, der durch das HMD erlangt wird, wird durch den schlechten Tragekomfort wieder ausgeglichen. Den Umgang mit den Markern empfanden die Versuchspersonen in Kombination mit dem HMD einfacher als mit dem Monitor.

Zusammenfassend lässt sich also festhalten, dass nur für die Kombination aus HMD und IR-Target eine durchgehende Verbesserung des Systems sichtbar wurde. Bei allen anderen Versuchen glichen sich Steigerungen des Realitätseindruckes und der Präsenz mit der deutlich schlechteren Bildqualität wieder aus. Um dem entgegen zu wirken, muss ein besserer Algorithmus zur Segmentierung des Hintergrunds zum Einsatz kommen. Eine Verbesserung der Bewertungen des IR-Targets im Vergleich zum Papiermarker zeigt sich nur bei der Verwendung eines HMDs. Wird ein Monitor als Ausgabe verwendet, schneiden die beiden Targets ähnlich ab. Der störende Hintergrund des Papiermarkers wird am Monitor kaum wahrgenommen, bei der Entfernung des Hintergrunds tritt er sogar positiv in Erscheinung, da die Löcher im Bild dadurch nicht so hervortreten. Die Verwendung des IR-Targets steigert die Bewertungen erst bei Verwendung in Kombination mit dem HMD. Dieses wäre als Anzeigegerät eindeutig zu bevorzugen, jedoch leidet dieses System am schlechten Tragekomfort des

Gerätes. Die positiven Effekte, die durch die höhere Bewegungsfreiheit und die stärkere Integration der virtuellen Umgebung in die reale entstehen, würden stärker ausfallen, wenn das HMD angenehmer zu Tragen wäre. Als bestes System hat sich die Kombination aus HMD, IR-Target und Hintergrundentfernung herausgestellt. Sie liefert mit Abstand die besten Ergebnisse in den Kategorien Realitätseindruck, Präsenz und Handling. Die Bildqualität ist zwar nicht ideal, mit einer durchschnittlichen Bewertung von 2,9 aber immer noch in einem akzeptablen Bereich. In der gesamten Evaluierung hat die Übertragung der Audiosignale ohne technisches System zu einer Asynchronität zwischen Bild und Ton geführt. Dies resultierte in einer schlechten Bewertung über alle Versuche hinweg. Aus diesem Grund muss der Übertragung der akustischen Daten sowie deren Aufbereitung (beispielsweise mittels der VA aus Kapitel 3.3) große Aufmerksamkeit geschenkt werden.

7.6 BillARd - Augmented Reality Billard

Um das in Kap. 6.1 entwickelte „BillARd“ auf die Eignung als Demonstrator zu testen, wurde auf dem System eine Evaluierung durchgeführt. Vorrangiger Aspekt der Untersuchung lag auf der subjektiven Bewertung des Demonstrators. Die Evaluierung beschränkte sich daher ausschließlich auf die Bewertung subjektiver Parameter durch die Probanden. Diese Parameter waren in drei Gruppen untergliedert. Diese Gruppen behandelten Aspekte der Umsetzung des Hardwaresetups, wie etwa die Positionierung von einzelnen Komponenten oder Markern, und der Darstellung mittels AR, Eigenschaften des umgesetzten „BillARd“ selbst, also beispielsweise die Spielbarkeit, und die Bewertung der umgesetzten Physik der Applikation. Weiterhin wurde auch die Bewertung der Ein- und Ausgabemöglichkeit mittels des WII-Controllers eingebunden, da diese Umsetzung stark an eine konkrete Anwendung gebunden ist (Menüs benötigen Inhalte, hier im Kontext zu „BillARd“).

7.6.1 Versuchsaufbau/-ablauf

Für die Untersuchung wurden die Probanden in Gruppen zu je zwei Personen eingeladen. Zu Beginn erhielten diese eine Einführung in die Thematik der AR, sowie der im Labor als Testumgebung verwendeten Hardware und deren Aufbau. Anschließend wurde den Testpersonen „BillARd“ vorgestellt, sowohl theoretisch, also die Spielsteuerung, als auch praktisch anhand eines kurzen Spielzuges als Demonstration. Die Durchführung der Untersuchung konnte jeweils nur abwechselnd erfolgen, da nur ein HMD für die Evaluierung zur Verfügung stand. Durch die Darstellung der Szene auf dem Monitor konnte jedoch die Person ohne HMD jederzeit dem Spielgeschehen folgen. Der Versuch gliederte sich in drei Teile. Im ersten Teil sollten die Versuchspersonen unter Anleitung des Spielleiters (Versuchsleiters) sich mit allen Möglichkeiten des Systems vertraut machen. Hierbei erhielten sie direkte Unterstützung durch den Spielleiter bei auftretenden Problemen. Im zweiten Teil wurde die bildbasierte Kollisionserkennung hinzugefügt. Dabei wurden verschiedene Gegenstände auf dem realen Tisch platziert, die nun aktiv in das Spielgeschehen eingebunden wurden. Dazu wurden die Probanden aufgefordert, diese Objekte direkt mit in ihren Spielzug einzubinden (beispielsweise durch direktes Anspielen). Im dritten Teil erfolgte ein freies Spielen, das nicht mehr durch den Spielleiter betreut wurde. Jeder Teil der Evaluierung dauerte fünf Minuten. Abschließend sollten die Versuchspersonen einen Fragebogen ausfüllen mit den zu bewerteten Eigenschaften auf einer Skala von 1 (sehr positiv) bis 6 (sehr negativ).

7.6.2 Ergebnisse

An der Untersuchung nahmen insgesamt zehn Personen teil. Diese waren im Alter von 22 bis 58 Jahren, das durchschnittliche Alter betrug 28 Jahre. 50% der Personen hatte bis zum Zeitpunkt der Untersuchung keinerlei Erfahrung mit AR gesammelt. 80% der Personen hatten dagegen bereits Erfahrungen im Bereich Spiele gesammelt, die mittels VR dargestellt werden. Im Folgenden werden die Ergebnisse der Fragebögen zu den jeweiligen Eigenschaften aufgeführt. Dabei wird zwischen der Bewertung des „BillARd“ und der Bewertung der Menüsteuerung mittels des WII-Controllers als Ein- und Ausgabegerät unterschieden.

7.6.2.1 BillARd

Abb. 7.28 zeigt die Ergebnisse der Bewertung von „BillARd“ hinsichtlich der Umsetzung in AR. Dabei zeigt sich, dass der Tragekomfort des HMDs mit einer Wertung von 4 als nicht besonders gut eingestuft wird. Der Realismus der virtuellen Objekte und deren Integration in die reale Umgebung der Versuchspersonen wurde jeweils mit 3 bewertet. Die Bewegungsfreiheit der Personen wurde im Schnitt mit einer Wertung von 2,6 versehen, dabei wurde das Queue mit den Markern mit einer Wertung von 2 deutlich besser bewertet, als der Gerüstaufbau für die Kollisionserkennung mit 2,9.

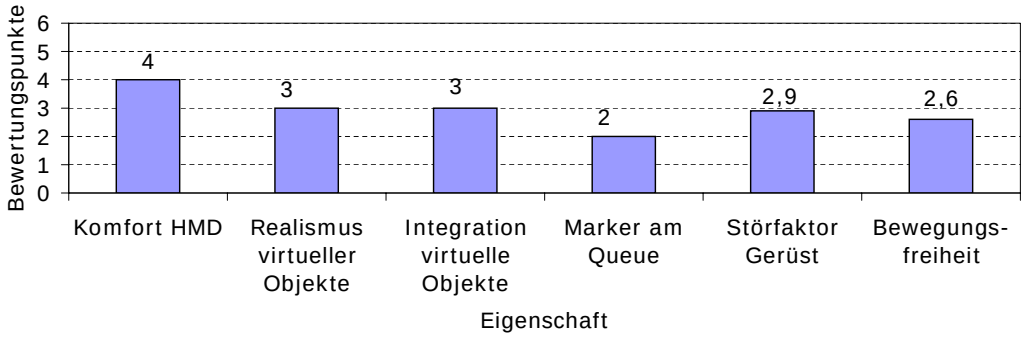


Abbildung 7.28: Ergebnisse der Fragebögen zu Augmented Reality

Die Ergebnisse der Bewertung des Spieles „BillARd“ selbst zeigt Abb. 7.29. Die generelle Spielbarkeit von „BillARd“ wurde im Mittel mit 3,2 bewertet. Die Eignung des Spieles als Demonstrator für AR wurde mit 2,1 als positiv eingestuft. Dabei wurde die Schwierigkeit, virtuelle Kugeln durch das reale Queue anzustoßen mit 2,3 als einfach bewertet. Hilfreich wurde hierbei auch der virtuelle Schatten der Spitze des Queue mit 2,2 benotet, der die aktuelle Position der Spitze über dem Tisch anzeigte. Der am Queue angebrachte WII-Controller als Menüsteuerung und Gerät für das haptische Feedback wurde mit 1,9 als nicht störend empfunden. Die Erkennbarkeit der Optionen, die mittels der Menüsteuerung eingestellt werden können, wurde mit 1,6 als sehr gut bewertet.

In Abb. 7.30 sind die Ergebnisse der Bewertung bzgl. der Umsetzung der Physik dargestellt. Der Realismus der Stöße der Kugeln wurde im Schnitt mit 2,9 als durchschnittlich bewertet. Der Bewegungsablauf wurde dagegen mit 1,6 als sehr flüssig und realistisch bewertet. Das haptische Feedback, das beim Stoßen einer Kugel durch den WII-Controller über das Queue an den Nutzer abgegeben wird, wurde dagegen lediglich mit einem Wert von 3,6 bewertet. Die realen Objekte, die durch die Kollisionserkennung in die AR Umgebung eingebracht wurden, wurden mit 2,4 als realistisch eingestuft. Die Interaktion mit der AR Umgebung,

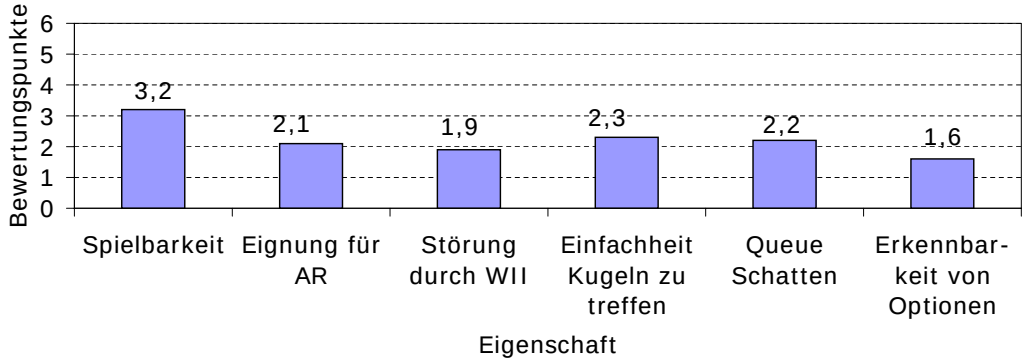


Abbildung 7.29: Ergebnisse der Fragebögen zu Billard

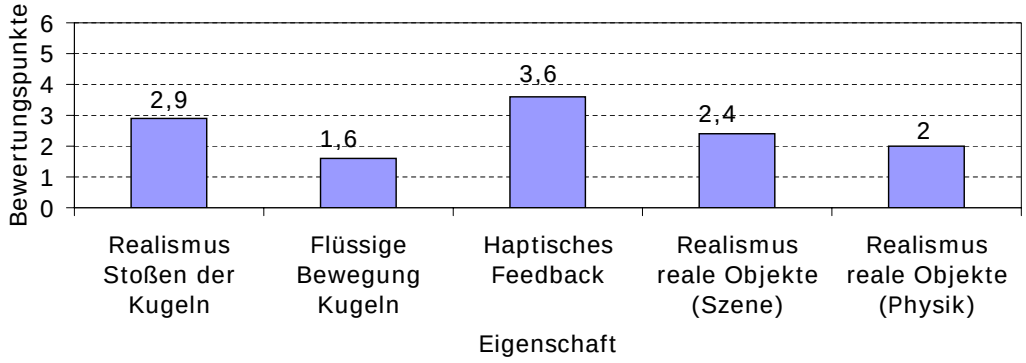


Abbildung 7.30: Ergebnisse der Fragebögen zu Physik

also die Einbindung in die Physik, wurde sogar noch besser mit einer Bewertung von 2,0 eingeschätzt.

7.6.2.2 WII-Controller als Ein- und Ausgabegerät

Da die Bewertung eines Ein- und Ausgabegerätes sehr stark an die Applikation gebunden ist, wurde die Evaluierung des WII-Controllers anhand der Steuerung der Optionen von „BillARd“ durchgeführt. Abb. 7.31 zeigt, dass die Gestaltung des Menüs mit einem Wert von 1,5 als sehr positiv bewertet wurde. Die Möglichkeit, die Menüs mittels des WII-Controllers zu steuern wurde ebenfalls mit einer Wertung von 1,6 als sehr positiv eingestuft. Die Intuitivität des Menüs wurde mit 1,5 als sehr intuitiv bewertet.

7.6.3 Diskussion der Ergebnisse

Die Evaluierung hat gezeigt, dass das umgesetzte „BillARd“ durchaus als Demonstrator für die Möglichkeiten der AR herangezogen werden kann. Die Bewertungen zeigen deutlich, dass das Umfeld, das zur Erzeugung von AR verwendet wird, noch nicht den Anforderungen der Probanden genügt hat. So wurde beispielsweise der Tragekomfort des HMDs als eher schlecht eingestuft. Zum anderen zeigte sich, dass die Darstellungsqualität durch das HMD noch nicht überzeugen konnte. Die zusätzliche Hardware neben dem HMD in Form eines Gerüstes zur Befestigung der Kamera, die der Kollisionserkennung dient, hat ebenfalls eine Verschlechterung der empfundenen Bewegungsfreiheit zur Folge. Insgesamt zeigt sich also, dass sich die Hardware noch nicht genügend in die reale Umgebung des Nutzers eingliedert. Neben diesen

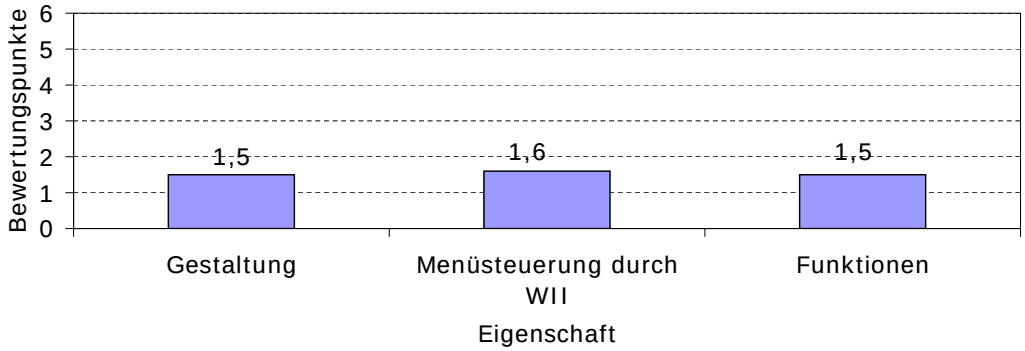


Abbildung 7.31: Ergebnisse der Fragebögen zu WII-Controller

Problemen mit der Hardware taucht auch die Problematik der korrekten Kalibrierung des HMDs auf, das bei ungenauer Kalibrierung einen Versatz zwischen dem rechten und linken Auge aufweist, so dass virtuelle Objekte nicht als einziges stereoskopes Objekt wahrgenommen werden können. Die Umsetzung von Billard in die AR konnte jedoch in großen Teilen die Probanden überzeugen. Zwar wurde die generelle Spielbarkeit nur neutral bewertet, aber diese Wertung kann durchaus auf die Probleme der Hardware zurückgeführt werden. Durchwegs gut wurden die einzelnen Möglichkeiten bewertet, mit der virtuellen Umgebung in Interaktion zu treten. Daher fiel auch die generelle Wertung der Eignung als Demonstrator gut aus. Lediglich die physikalische Simulation des Spieles und der eingebundenen realen Objekte kann nicht überzeugen. So wurde das haptische Feedback mittels des WII-Controllers als schlecht bewertet, was sich auf die mangelnde Intensität zurückführen ließ. Die Integration realer Objekte konnte dagegen ebenfalls nur etwas besser als neutral abschneiden. Da „BillARd“ jedoch als Demonstrator der Interaktionsmöglichkeiten konzipiert wurde, kann man zusammenfassend feststellen, dass sich dieses Spiel für die Darstellung von Interaktionsmöglichkeiten innerhalb der AR gut verwenden lässt.

7.7 Bowl-AR-ama - Augmented Reality Bowling

Das in Kapitel 6.2 vorgestellte Anwendungsszenario „Bowl-AR-ama“ wurde in einer Evaluierung untersucht. Da es sich hierbei um eine Applikation handelt, die erstellt wurde, um die Möglichkeiten der in dieser Arbeit entwickelten Interaktionsmöglichkeiten zu demonstrieren, lag der Fokus dieser Studie nicht auf der Evaluierung der einzelnen Interaktionsmöglichkeiten, sondern auf der Bewertung der Applikation und der Integration der Interaktionsarten selbst.

Im Vordergrund standen also die Bewertung von Eigenschaften, die die Wirkung auf die Versuchsperson beschreiben. Hierzu zählen das Design der Applikation (also die Modellierung der virtuellen Objekte innerhalb von „Bowl-AR-ama“), die Umsetzung von der zugrunde liegenden Physik (bzw. ihrer Simulation) und ihrer Visualisierung (Flüssigkeit der Bewegungen), der daraus resultierende gesamte Eindruck der Realitätstreue und den dadurch hervorgerufenen Grad der Immersion. Weiterhin wurden Eigenschaften bewertet, die den Umgang mit dem Gesamtsystem beschreiben, wie etwa der Komfort (Tragekomfort Hardware) oder der Grad der Freiheit, die eine Bewegung innerhalb von „Bowl-AR-ama“ ermöglicht.

Zusätzlich wurden auch die Interaktionskomponenten hinsichtlich ihrer Integration in “Bowl-AR-ama” abgefragt. Zu diesen zählten die Integration der Videokonferenz aus Kapitel 5.2 (also die Präsenz des Gesprächspartners), die Darstellung der Szene mittels der

virtuellen Kamera aus Kapitel 3.2 (Flüssigkeit der Bewegungen sowie Platzierungsmöglichkeiten) sowie das Handling mit den Targets der Gestenerkennung aus Kapitel 4.1.

Ausgehend von diesen Ergebnissen sollte also sichergestellt werden, ob die entwickelte Applikation „Bowl-AR-ama“ als Demonstration zur Interaktion in AR in der Spieledomäne geeignet ist.

7.7.1 Versuchsaufbau/-ablauf

Zur Durchführung der Testspiele wurden die Personen in kleinen Gruppen in das AR-Labor des Lehrstuhls eingeladen. Hier stand den Personen „Bowl-AR-ama“ zur Verfügung.

Zu Beginn konnte jeder Spieler seinen Namen ins User Interface des Bowlingspiels eintragen. Nachdem das Spiel gestartet wurde, erhielten alle Teilnehmer eine kurze Einführung in die AR, die Funktionsweise des IR-Trackingsystems sowie den Umgang mit dem HMD und den Fingertargets.

Im Anschluss daran folgten eine kurze Einweisung in die Spielregeln von Bowling und der eigentliche Beginn des Spieles. Eine Person war Teil der AR und konnte die Interaktionsmöglichkeiten von „Bowl-AR-ama“ nutzen. Hierbei wurden sämtliche Interaktionsmöglichkeiten verwendet, um eine anschließende Wertung aller Möglichkeiten vornehmen zu können. Zeitgleich konnten die restlichen Teilnehmer, die nicht Teil der AR waren, das Geschehen mithilfe der virtuellen Kamera mitverfolgen und die Einstellung dieser frei wählen. Nach jeder Spielrunde wurde die aktive Person gewechselt, um eine gleichmäßige Verteilung auf aktive und passive Spieler zu erreichen. Einer der passiven Spieler diente dabei als Gesprächspartner der virtuellen Videokonferenz und wurde somit in die Umgebung des aktiven Spielers eingeblendet.

Nach Beendigung des Spiels füllte jeder Teilnehmer einen Fragebogen aus, auf dem die Eigenschaften mittels einer Skala von 1 (sehr positiv) bis 6 (sehr negativ) bewertet werden konnten.

7.7.2 Ergebnisse

Insgesamt wurden zwölf Testpersonen im Alter von 11 bis 27 Jahren eingeladen. Hiervon waren 50% Schüler und 50% Studenten. 42% der Personen gaben an, bereits Erfahrungen mit AR gesammelt zu haben. Dagegen werden VR-Spiele von 49% der Befragten von Zeit zu Zeit und von 33% regelmäßig gespielt. Lediglich eine Person machte die Aussage, noch nie ein Spiel in VR gespielt zu haben.

Im Folgenden werden die Ergebnisse der Befragung zu den einzelnen Eigenschaften aufgeführt. Abb. 7.32 und 7.33 stellen diese Ergebnisse dar.

Als besonders positiv ist den Befragten das Design der virtuellen Objekte sowie die Bewegungsfreiheit im Raum aufgefallen. Dies zeigt die Bewertung mit 1,75 bzw. 2,5. Der Komfort des HMD dagegen wurde mit 4,25 als unkomfortabel empfunden. Auch die Flüssigkeit der Bewegungen der virtuellen Objekte in der AR-Szene wurde mit 3,25 eher negativ beurteilt. Der Grad der Immersion wurde durchschnittlich mit 2,83 bewertet, der Realitätseindruck mit 3.

Die Möglichkeit, die Bowlingkugel mit den Fingern greifen zu können, wurde von den Spielern positiv aufgenommen und als eine intuitive Methode bezeichnet. Der Umgang mit den Targets, also deren Handling, wurde durchschnittlich mit 2,58 bewertet. Allerdings traten immer wieder Probleme mit der Erkennung der Fingertargets auf, was von den Versuchspersonen stark bemängelt wurde. Sobald sich ein Target außerhalb des Sichtbereichs der IR-Kameras

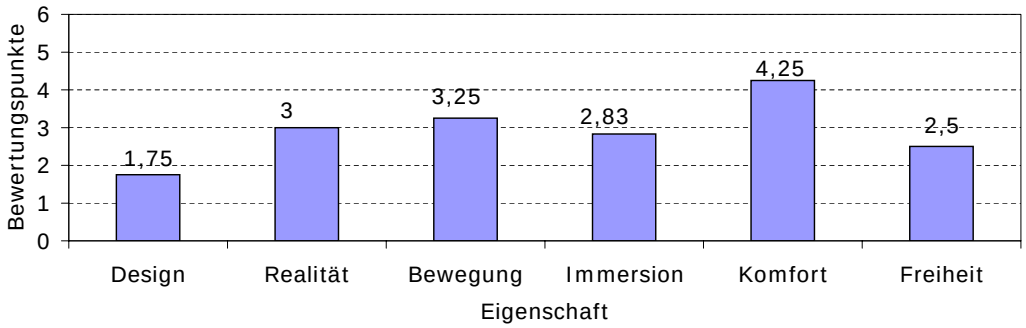


Abbildung 7.32: Ergebnisse der Fragebögen I

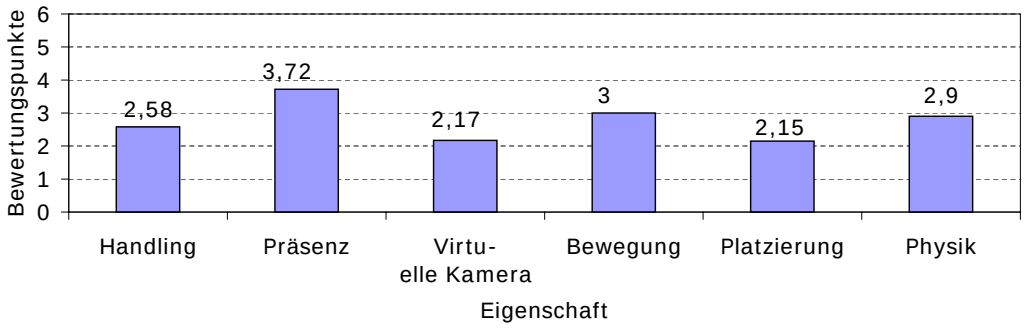


Abbildung 7.33: Ergebnisse der Fragebögen II

befand, konnte dieses nicht identifiziert werden und folglich konnten weder Gesten erkannt werden, noch konnte die Kugel, die an dieses Target gebunden war, angezeigt werden.

Die Testpersonen hatten nicht das Gefühl, dass sich die von der virtuellen Konferenz eingeblendete Person tatsächlich im Raum befand. Die Präsenz des Gesprächspartners wurde hier nur mit einem Wert von 3,72 bewertet. Als Grund hierfür wurde angegeben, dass das virtuelle Abbild des Konferenzteilnehmers zu klein war, um es als realistisch empfinden zu können. Außerdem war die Qualität des Kamerabildes nicht hochwertig genug.

Die virtuelle Kamera, generell bewertet mit 2,17, konnte bei den meisten Personen den Eindruck erwecken, die Geschehnisse der AR live mitverfolgen zu können. Leider war die Wiedergabe der Aufnahmen immer wieder stark ruckartig, weshalb die Flüssigkeit der Bewegungen der Objekte nur mit 3 bewertet wurde. Die verschiedenen Möglichkeiten zur Positionierung der Kamera wurden mit 2,15 bewertet.

Die Realitätstreue der Physik, welche der Simulation der Objektbewegungen zugrunde lag, wurde mit 2,9 bewertet.

Am Ende des Spiels stimmten alle Personen der Aussage zu, dass ihnen das Bowlingspiel Spaß gemacht habe und dass ihnen ein guter Überblick über die Arbeiten im Bereich der AR des Lehrstuhls vermittelt wurde. Ein Teil der Personen war durch das Bowlingspiel überzeugt, dass AR in der Spieledomäne in naher Zukunft eine echte Konkurrenz zu Spielen der VR sein kann.

7.7.3 Diskussion der Ergebnisse

Die Untersuchung hat gezeigt, dass „Bowl-AR-ama“ als Demonstration der Interaktionsmöglichkeiten innerhalb der AR geeignet ist. Die Bewertungen zeigen, dass die Versuchspersonen

sowohl das Spiel selbst als auch deren integrierte Interaktionsmöglichkeiten gut beurteilen.

Mängel im Umgang mit dem System traten durch das IR-Tracking auf, auf dem die Gestenerkennung basiert. Durch Lücken beim Erkennen der Targets an den Fingern der Nutzer traten teilweise Probleme bei der Interaktion auf, die sich in einer schlechteren Bewertung ausdrücken.

Weiterhin konnte die Flüssigkeit der Bewegungen nicht überzeugen. Durch eine große Anzahl von Objekten und Bewegungen entstand eine hohe Auslastung durch die Visualisierung, die zu Verzögerungen in der Berechnung der Darstellung führte. Dies resultierte sowohl für die Darstellung der AR als auch der VR mittels der virtuellen Kamera in einer schlechteren Bewertung.

Ebenfalls litt die Bewertung des Komforts unter den Trageeigenschaften des verwendeten HMDs. Dieses ist für einen Dauerbetrieb ergonomisch nicht geeignet und verursachte eine deutlich schlechtere Wahrnehmung des Komforts für die Nutzer. Dazu trugen auch die verwendeten Targets des Gestenerkenners bei, die nicht für einen Dauerbetrieb für den Nutzer geeignet sind.

KAPITEL 8

Resumee

Abschließend fasst das folgende Kapitel die Ergebnisse, die in dieser Arbeit gewonnen wurden, zusammen und diskutiert diese. Daraufhin folgt ein kurzer Ausblick auf mögliches Verbesserungspotential und weitere Forschungsrichtungen, die sich im Laufe dieser Arbeit herausgestellt haben.

8.1 Zusammenfassung

Die hier vorliegende Arbeit hat die Zielsetzung, multimodale Interaktion mit einem Augmented Reality-System in der Spieledomäne zu ermöglichen. Fokus der Arbeit ist hierbei nicht die Fusion bestehender Eingabemöglichkeiten zu einer Nutzerintention, sondern die Entwicklung geeigneter Interaktionsmöglichkeiten zwischen dem Nutzer und dem System speziell für die Augmented Reality. Die Interaktionsmöglichkeiten bestehen hierbei aus der Interaktion des Systems zum Nutzer, also aus Ausgabemöglichkeiten, sowie der Interaktion des Nutzers zum System, also der Eingabemöglichkeiten. Weiterhin steht auch die Interaktion zwischen realen und virtuellen Objekten im Fokus dieser Arbeit sowie interpersonelle Interaktion in Form eines Videokonferenzsystems.

Die Arbeit beruht auf einer eigens entwickelten Softwareplattform, die alle umgesetzten Funktionalitäten in einem gesamten Paket vereint. Das Konzept dieser Plattform sieht die Verteilung einzelner, modular zusammenstellbarer Komponenten auf verschiedene Rechner vor. Somit kann die Rechenlast auf verschiedene Systeme verteilt werden und spezielle Applikationen können auf dafür optimierten Plattformen ausgeführt werden. Die Kommunikation erfolgt hier mittels Netzwerkverbindungen.

Auf dieser Plattform basieren drei Interaktionsarten, die die Ausgaben des Systems an den Nutzer übertragen. Diese gliedern sich in zwei visuelle und eine akustische Methode. Die visuelle Ausgabe ist sowohl über Augmented, als auch Virtual Reality möglich. Dabei wird in beiden Fällen auf eine bereits kommerziell existierende Software („Unifeye“) zurückgegriffen, die im Rahmen dieser Arbeit um die Verteilbarkeit auf eine beliebig große Anzahl Rechner erweitert wurde. Dabei liegt der Schwerpunkt auf der Umsetzung einer performanten und gleichzeitig synchron ablaufenden Steuerung aller angeschlossener Rechner, die die virtuelle Szene mit Augmented oder Virtual Reality darstellen. Dies wird durch die Verwendung eines speziellen Frameworks gewährleistet, das für verteilte Netze entwickelt und optimiert wurde („ICE“).

Die Umsetzung von virtueller Akustik erfolgt durch binaurale Methoden. Durch die Filterung virtueller Schallquellen lässt sich mittels Head Related Transfer Functions die Position einer solchen Quelle simulieren. Dabei wird die Blickrichtung des Nutzers mithilfe des IR-Trackingsystems verfolgt und für die Berechnung der virtuellen Schallquelle herangezogen. Die so gewonnenen akustischen Signale werden über einen Kopfhörer an den Nutzer abgegeben.

Für Eingaben an das System stehen in dieser Arbeit drei Möglichkeiten zur Verfügung. Um speziell die Vorteile der Augmented Reality zu wahren, ist in das System ein Gestenerkenner integriert, der die Position von Daumen und Zeigefinger des Nutzers über das IR-Tracking verfolgt. Basierend auf diesen Daten können sowohl statische Gesten (wie etwa Greifen und Zeigen) sowie dynamische Gesten (wie etwa Winken) unterschieden werden. Dadurch können Gesten nicht nur als Auslöser für Funktionen genutzt werden, sondern stehen in direktem räumlichen Zusammenhang mit der Szene. So kann ein virtuelles Objekt mit einer Greifgeste direkt verschoben werden. Die Erkennung statischer Gesten erfolgt in dieser Arbeit mittels Euklid'scher Abstandsmessung, die Erkennung dynamischer Gesten basiert auf der Klassifizierung mittels Hidden-Markov-Modellen. Um den Tragekomfort zu erhöhen, stehen passive und aktive IR-Targets zur Verfügung.

Weiterhin besteht die Möglichkeit, Eingaben mit einem Tangible User Interfaces vorzunehmen. Dabei wurden Sensoren, die Neigungswinkel und Winkelgeschwindigkeiten messen können, in ein Gehäuse verbaut. Zusammen mit vier Tastern entstehen so Bedienmöglichkeiten, virtuelle Objekte direkt manipulieren zu können, indem das Interface bewegt oder geneigt wird.

Als ein kombiniertes Ein- und Ausgabegerät besteht im System ein WII Controller, der sowohl Eingaben in Form von Tastern verarbeiten kann, als auch Ausgaben bereit stellt. Dazu zählt ein visuelles und ein haptisches Feedback. Durch die Verwendung dieses Controllers können also haptische Informationen in Form von Vibrationen an den Nutzer übertragen werden. Weiterhin bildet dieser Controller die Basis für ein generisches Menüsystem für 3D-Menüs in Augmented Reality Applikationen. Dieses System bildet abstrakt Menüs ab, die so von beliebigen Anwendungen gestaltet und verwendet werden können.

Um die Interaktionen nicht nur auf die Ein- und Ausgabe zu beschränken, stellt das System auch die Wechselwirkung von realen und virtuellen Objekten zur Verfügung. Hierbei werden mithilfe einer Kamera reale Objekte auf der Oberfläche eines Tisches, über dem die Kamera montiert ist, detektiert, durch Algorithmen der Bildverarbeitung vom Tisch segmentiert und schließlich in ein 3-D-Modell übergeführt. Dieses Modell kann dann in die Augmented Reality-Szene integriert werden, wodurch auch Kollisionen zwischen dem Modell und weiteren virtuellen Objekten möglich sind. Diese Kollisionen können mit Aktionen verknüpft werden, die den Eindruck von Interaktion zwischen dem realen und dem virtuellen Objekt entstehen lässt.

Um die interpersonelle Interaktion zu bedienen, bietet das System eine Videokonferenz an, die sich in die virtuelle Umgebung des Nutzers einfügt. Der Gesprächspartner wird dabei von einer Kamera aufgenommen, optional vom Hintergrund befreit und in die Augmented Reality-Szene eingebunden.

Um die Interaktionsmöglichkeiten anschaulich darstellen zu können, zeigt diese Arbeit zwei Beispiele aus der Spieledomäne. Zum einen ein virtuelles Billardspiel („BillARd“), zum anderen ein virtuelles Bowlingspiel („Bowl-AR-ama“). Diese beiden Beispielszenarien zeigen demonstrativ die Möglichkeiten auf, die durch die zuvor beschriebenen Interaktionsarten bestehen.

Sämtliche grundlegenden Interaktionsarten wurden in Nutzerstudien untersucht. Die Ergebnisse der Evaluierung zeigen eine durchweg positive Bilanz und Akzeptanz durch die Anwender. Das zugrunde liegende Trackingsystem verursachte die größten Probleme, da sich herausstellt, dass ungenaue oder gar fehlende Information über die Position und Orientierung von Nutzer oder Objekten zu großen Einbußen in der Wahrnehmung der Versuchspersonen führt. Ebenfalls zeigt sich, dass die Ansätze für Nutzereingaben in speziellen Szenarien unterschiedlich gut abschneiden.

8.2 Diskussion

Betrachtet man die einzelnen Interaktionsmöglichkeiten genauer, zeigt sich, dass Defizite festzustellen sind.

Die Performance der Darstellung mittels der Augmented Reality ist direkt abhängig von der Anzahl der geladenen Objekte. Eine höhere Anzahl von virtuellen Objekten verringert also die Anzahl der pro Sekunde berechneten Bilder. Ebenfalls ist nur eine begrenzte Anzahl von Manipulationen pro Sekunde möglich. Flüssige Bewegungen können daher nur schwer abgebildet werden. Hier muss ein Kompromiss zwischen Schrittweite der Bewegung und der Bewegungsgeschwindigkeit getroffen werden. Ein weiteres Problem stellt die Notwendigkeit der exakten Kalibrierung dar. Um eine exakte Überlagerung von realen und virtuellen Objekten zu erhalten, muss das System sehr genau kalibriert werden, was jedoch sehr schwierig ist. Da die Darstellung der Virtual Reality auf derselben Grundlage basiert, gilt die Problematik hier ebenso.

Die virtuelle Akustik bereichert zwar die Augmented Reality, jedoch ist diese in dieser Arbeit auf eine reine Simulation der Richtung, aus der ein virtueller Schall wahrgenommen wird, beschränkt. Das umgesetzte System ist weder in der Lage die Elevation noch die Entfernung eines Schalles zu simulieren. Diese kann lediglich durch die Lautstärke der virtuellen Schallquelle nachgeahmt werden.

Die Erkennung von Gesten mithilfe des infraroten Trackingsystems ermöglicht die Steuerung von ortsgebundenen Funktionen innerhalb der Augmented Reality. Dies erlaubt dem Nutzer eine sehr realistische Interaktion mit seiner virtuellen Umgebung, da sie sehr stark an die Interaktion mit der realen Umgebung angelehnt ist. Diese Methode jedoch beinhaltet große Probleme, falls das Trackingsystem nicht einwandfrei läuft. Werden die Targets, die der Nutzer an seinen Fingern tragen muss, nicht erkannt, verliert das System jegliche Interaktionsmöglichkeit. In der Evaluierung zeigte sich, dass der Verlust der Targets sehr häufig auftritt und damit die Zuverlässigkeit der Gestenerkennung stark veringert wird.

Die vom Trackingsystem unabhängige Gestenerkennung mittels des Tangible User Interfaces ermöglicht einen neuen Zugang für den Nutzer, Manipulationen auf virtuelle Objekte anzuwenden. In der vorliegenden Umsetzung jedoch zeigt sich, dass das Interface zwar die Manipulation von Objekten einfach ermöglicht, aber die Selektion, also die Auswahl eines virtuellen Objektes, nur schwer durchführbar ist. Weiterhin ist das Interface aktuell auf drei Freiheitsgrade beschränkt und bildet somit nicht alle sechs möglichen Grade der Translation und Rotation ab.

Die Verwendung des WII Controllers als haptisches Ausgabegerät ist noch nicht ausreichend. Bei diesem Gerät können bislang keine Parameter hinsichtlich der Vibrationsstärke oder Frequenz vorgegeben werden. Aus diesem Grund können keine unterschiedlichen Arten von Feedback vom Nutzer differenziert werden.

Mit der Kollisionserkennung von realen und virtuellen Objekten werden Wechselwirkungen

der realen und virtuellen Umgebung des Nutzers in Echtzeit ermöglicht, ohne dabei zuvor die realen Objekte modellieren oder tracken zu müssen. Beim aktuellen Systemaufbau ist jedoch nur die Erfassung von Konturen möglich, da nur eine einzelne Kamera für die Bilderfassung zum Einsatz kommt. Somit können keine komplexen 3-D-Objekte für eine Interaktion herangezogen werden. Weiterhin ermöglicht das System keine Kollisionserkennung in verschiedenen Ebenen, die Erkennung arbeitet aufgrund der Projektion aller Objekte auf eine einzige Ebene nur in 2-D. Die Erkennung von Kollisionen ist aufgrund der Performance auf ein spezielles Szenario optimiert, indem virtuelle Objekte nur in Form von Kugeln berücksichtigt werden. Kollisionen können also nur zwischen einem realen Objekt und virtuellen Kugeln detektiert werden.

Die Interaktion zwischen voneinander entfernten Nutzern ermöglicht die Integration eines Videokonferenzsystems. Dieses erlaubt die Kollaboration mehrerer Nutzer innerhalb derselben Augmented Reality-Szene und stellt zeitgleich die Kommunikation zwischen den Nutzern her. Die Darstellung ist hierbei jedoch auf 2-D beschränkt und die Qualität der Aufnahme wird durch die Verwendung von Algorithmen der Bildverarbeitung zur Trennung von Hintergrund und Gesprächspartner limitiert. Diese verursachen ein Rauschen im Bild, das sich durch Löcher im Dialogpartner oder Artefakte aus dem Hintergrund zeigen. Weiterhin besteht die Problematik, dass der Gesprächspartner ortsfest vor der Kamera befindlich sein muss, um ihn für die anderen Gesprächsteilnehmer zu erfassen. Möchte der Teilnehmer nun auch innerhalb seiner Augmented Reality-Umgebung interagieren, ist er auf den kleinen Raum vor der Kamera begrenzt und wird mit seinem Head-Mounted-Display dargestellt, das er tragen muss, um seine Szene dargestellt zu bekommen.

8.3 Ausblick

Nimmt man nun die Diskussion der Probleme als Ausgangsbasis, ergeben sich zunächst für die einzelnen Interaktionsmöglichkeiten weiterführende Entwicklungsmöglichkeiten.

Um die Darstellung der Augmented und Virtual Reality zu verbessern, wird eine größere Rechenleistung benötigt. Gleichzeitig könnte auch die Verwendung eines anderen Softwarepakets, als das in dieser Arbeit verwendete, eine Steigerung der Performance mit sich bringen. Die Entwicklung verbesserter Algorithmen zur Kalibrierung des Gesamtsystems kann die Genauigkeit bei der Überlagerung von realen und virtuellen Objekten verbessern.

Die Verbesserung der virtuellen Akustik ist bereits Schwerpunkt der Forschung. Die Simulation der Elevation der virtuellen Schallquelle sowie deren Abstand zum Nutzer könnte die Immersion des Nutzers weiter erhöhen. Ebenfalls könnte die Umsetzung eines Systems, das nicht auf einem Kopfhörer, sondern einem Lautsprecherarray basiert, eine Verbesserung zur Folge haben.

Die Erkennung von Gesten mit dem Trackingsystem kann auf verschiedene Arten weitergeführt werden. Allem voran muss die Verbesserung der Erkennungsrate der Targets stehen. Das Bestreben in dieser Arbeit, alternativ auf aktive Targets auszuweichen, sollte weiterverfolgt werden. Dabei kann durch die Reduktion der einzelnen LEDs auf eine je Finger der Tragekomfort erheblich verbessert werden. Dazu muss jedoch noch ein Weg gefunden werden, einzelne LEDs zuverlässig dem zugehörigen Finger zuzuordnen. Ist dies geschehen, können durch die Verbesserung der Klassifikation durch neue Ansätze die Erkennungsrate und der Wortschatz der dynamischen Gesten erhöht werden.

Das Tangible User Interface könnte durch die Erweiterung der Sensorik auf die Erfassung von sechs Freiheitsgraden profitieren, da damit alle Manipulationen durch dieses Interface

abgedeckt werden könnten. Dabei ist jedoch die genaue Kalibrierung der Sensorik notwendig, die den Messdaten insbesondere bei der Messung von Bewegungen (statt Rotationen) dem Drift der Hardware gegenüberstehen. Um exakte Bewegungsabstände erfassen zu können, muss diese Problematik weiterer Forschungsgegenstand sein.

Um eine Differenzierung von haptischem Feedback zu ermöglichen, können weitere Geräte integriert werden, die in der Lage sind haptisches Feedback zu erzeugen. Hierbei kann in Zukunft der Fokus auf Geräten liegen, sowohl Vibrationen, als auch Stöße erzeugen können. Dies könnte beispielsweise durch ein mechanisches Federsystem ermöglicht werden, das eine gespannte Feder ruckartig entspannt und somit einen Stoß generiert.

Die Interaktion zwischen realer und virtueller Umgebung kann durch den Einsatz von zwei Kameras verbessert werden. Mit einer zweiten Kamera können Tiefeninformationen über die realen Objekte ermittelt werden und somit nicht nur als Kontur in 2-D sondern in 3-D modelliert werden. Damit wäre auch die Kollisionserkennung nicht nur auf eine einzige Ebene beschränkt, sondern könnte in Abhängigkeit der Höhe der Objekte erfolgen. Die Steigerung der Performance durch optimierte Algorithmen kann die Detektion von Kollisionen verbessern, eine Erweiterung auf eine größere Anzahl von Objekten, die nicht mehr als Kugel idealisiert sind, wäre dadurch erreichbar.

Verbesserte Algorithmen der Bildverarbeitung könnten auch die Qualität der Videokonferenz positiv beeinflussen. Eine robustere Trennung zwischen Hintergrund und Teilnehmer führt zu einer erhöhten Realitätstreue der Darstellung. Die Verwendung von zwei Kameras zur Erfassung des Dialogpartners könnte auch hier die Darstellung von 2-D in 3-D ermöglichen. Denkbar wäre auch, Modelle des Gesprächspartners zu erstellen, die dann in Abhängigkeit des aktuellen Zustandes in Echtzeit modelliert würden. Somit könnten die Ortsgebundenheit des Teilnehmers und der störende Einfluss des sichtbaren Head-Mounted-Displays eliminiert werden.

Das umgesetzte System kann durch das Hinzufügen weiterer Interaktionsmöglichkeiten verbessert werden. So kann die Ausgabeseite durch die Erweiterung von visueller und akustischer Darstellung um die taktile Ausgabe ergänzt werden. Damit ist die Szene nicht nur sicht- und hörbar, sondern auch tastbar. Fügt man der Eingabe klassische Elemente, wie etwa die Spracherkennung, hinzu, kann die Nutzerintention durch die Fusion bestehender Interaktionsarten robuster gestaltet werden.

Die vorliegende Arbeit hat also die Grundlage dafür geschaffen, Spiele aus der Virtual Reality in die Augmented Reality zu übertragen. Werden die bestehenden Methoden verbessert und neuartige Möglichkeiten hinzugefügt, kann in Zukunft die Interaktion mit Spielen in der realen Umgebung des Nutzers erfolgen.

Abbildungsverzeichnis

Abkürzungsverzeichnis

AR	Augmented Reality
BRIR	Binaural Room Impulse Response
FFT	Fast-Fourier Transformation
HMD	Head-Mounted-Display
HMM	Hidden-Markov-Modell
HRIR	Head Related Impulse Response
HSI	Hermitsche-Spline-Interpolation
IFS	IndexedFaceSet
IR	Infrarot
KOS	Koordinatensystem
LED	Light Emitting Diode
MMK	Mensch-Maschine-Kommunikation
OST	Optical-See-Through
TCP	Transmission Control Protocol
TUI	Tangible User Interface
UDP	User Datagram Protocol
USB	Universal Serial Bus
VA	Virtuelle Akustik
VR	Virtual Reality
VRML	Virtual Reality Modelling Language
VST	Video-See-Through

Stichwortverzeichnis

Literaturverzeichnis

[1] Dynamic virtual convergence for video see-through head-mounted displays: Maintaining maximum stereo overlap throughout a close-range work space. In *ISAR '01: Proceedings of the IEEE and ACM International Symposium on Augmented Reality (ISAR'01)*, page 137, Washington, DC, USA, 2001. IEEE Computer Society.

[2] A.R.T. advanced realtime tracking GmbH. http://www.ar-tracking.com. Website, Zugriff von, 2006.

[3] Marco Andreetto, Nicola Brusco, and Guido M. Cortelazzo. Automatic 3d modeling of archaeological objects. In *Conference on Computer Vision and Pattern Recognition Workshop - Volume 1*, 2003.

[4] Ronald Azuma and Gary Bishop. Improving static and dynamic registration in an optical see-through hmd. In *SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 197–204, New York, NY, USA, 1994. ACM.

[5] Ronald T. Azuma. A survey of augmented reality. *Presence: Teleoperators and Virtual Environments*, 6(4):355–385, August 1997.

[6] Istvan Barakonyi, Tamer Fahmy, and Dieter Schmalstieg. Remote collaboration using augmented reality videoconferencing. In *GI '04: Proceedings of Graphics Interface 2004*, pages 89–96, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 2004. Canadian Human-Computer Communications Society.

[7] István Barakonyi, Markus Weilguny, Thomas Psik, and Dieter Schmalstieg. Monkeybridge: autonomous agents in augmented reality games. In *ACE '05: Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology*, pages 172–175, New York, NY, USA, 2005. ACM.

[8] C. Benoit, J. Martin, C. Pelachaud, L. Schomaker, and B. Suhm. Audio-visual and multimodal speech systems.

[9] Volkert Buchmann, Stephen Violich, Mark Billinghurst, and Andy Cockburn. Fingartips: gesture based direct manipulation in augmented reality. In *GRAPHITE '04: Proceedings*

of the 2nd international conference on Computer graphics and interactive techniques in Australasia and South East Asia, pages 212–221, New York, NY, USA, 2004. ACM.

[10] O. Cakmakci and J. Rolland. Head-worn displays: a review. *Display Technology, Journal of*, 2(3):199–216, 2006.

[11] Denis Chekhlov, Andrew Gee, Andrew Calway, and Walterio Mayol-Cuevas. Ninja on a plane: Automatic discovery of physical planes for augmented reality using visual slam. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, November 2007.

[12] Adrian David Cheok, Siew Wan Fong, Kok Hwee Goh, Xubo Yang, Wei Liu, and Farzam Farzbiz. Human pacman: a sensing-based mobile entertainment system with ubiquitous computing and tangible interaction. In *NetGames '03: Proceedings of the 2nd workshop on Network and system support for games*, pages 106–117, New York, NY, USA, 2003. ACM.

[13] Adrian David Cheok, Kok Hwee Goh, Wei Liu, Farzam Farbiz, Sze Lee Teo, Hui Siang Teo, Shang Ping Lee, Yu Li, Siew Wan Fong, and Xubo Yang. Human pacman: a mobile wide-area entertainment system based on physical, social, and ubiquitous computing. In *Advances in Computer Entertainment Technology*, pages 360–361. ACM, 2004.

[14] BioWare Corp. http://nwn.bioware.com/. Website, Zugriff von, 2008.

[15] Intel Corp. http://www.intel.com/technology/computing/opencv/index.htm. Website, Zugriff von, 2007.

[16] Valve Corp. http://orange.half-life2.com/hl2.html. Website, Zugriff von, 2007.

[17] Ascension Technology Corporation. http://www.ascension-tech.com/. Website, Zugriff von, 2008.

[18] E. Corteel. Synthesis of directional sources using wave field synthesis, possibilities, and limitations. *EURASIP J. Appl. Signal Process.*, 2007(1):188–188, 2007.

[19] F. Dai and M. Goebel. Virtual prototyping - an approach using virtual reality techniques. In *Proceedings of 14th ASME Int. Computers in Engineering Conference*, 1994.

[20] Cambridge University Engineering Department. http://htk.eng.cam.ac.uk/. Website, Zugriff von, 2007.

[21] Gerhard Eckel. Immersive audio-augmented environments: The listen project. In *IV '01: Proceedings of the Fifth International Conference on Information Visualisation*, pages 571–573, Washington, DC, USA, 2001. IEEE Computer Society.

[22] Jens Eckstein and Elmar Schömer. Dynamic collision detection in virtual reality applications. In V. Skala, editor, *WSCG'99 Conference Proceedings*, 1999.

[23] Sony Computer Entertainment Europe. http://de.playstation.com/games-media/games/detail/item32241/eyetoy:-play-3/. Website, Zugriff von, 2008.

[24] The Tao Framework. http://www.taoframework.com. Website, Zugriff von, 2007.

[25] Henry Fuchs, Mark A. Livingston, Ramesh Raskar, D'nardo Colucci, Kurtis Keller, Andrei State, Jessica R. Crawford, Paul Rademacher, Samuel H. Drake, and Anthony A. Meyer. Augmented reality visualization for laparoscopic surgery. In *MICCAI '98: Proceedings of the First International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 934–943, London, UK, 1998. Springer-Verlag.

[26] V. Garcia, E. Debreuve, and M. Barlaud. A contour tracking algorithm for rotoscopy. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Toulouse, France, May 2006.

[27] Y. Genc, F. Sauer, F. Wenzel, M. Tuceryan, and N. Navab. Optical see-through hmd calibration: A stereo method validated with a video see-through system. In *International Symposium for Augmented Reality*, 2000.

[28] Anders Henrysson, Mark Billinghurst, and Mark Ollila. Face to face collaborative ar on mobile phones. In *ISMAR '05: Proceedings of the Fourth IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 80–89, Washington, DC, USA, 2005. IEEE Computer Society.

[29] Gerrit Hillebrand, Martin Bauer, Kurt Achatz, and Gudrun Klinker. Inverse kinematic infrared optical finger tracking. In *9th International Conference on Humans and Computers (HC 2006)*, September 2006.

[30] Charles E. Hughes, Eileen Smith, Christopher Stapleton, and Darin E. Hughes. Augmenting museum experiences with mixed reality. In *KSCE 2004: Proceedings of Knowledge Sharing and Collaborative Engineering - 2004*, 2004.

[31] Darin E. Hughes, Scott Vogelpohl, and Charles E. Hughes. Designing a system for effective use of immersive audio in mixed reality. In *SIMCHI'05: Proceedings of 2005 International Conference on Human-Computer Interface Advances in Modeling and Simulation*, pages 51–57, 2005.

[32] Atari Inc. http://www.atari.com. Website, Zugriff von, 2008.

[33] Electronic Arts Inc. http://www.uo.com/. Website, Zugriff von, 2008.

[34] InterSense Inc. http://www.intersense.com. Website, Zugriff von, 2008.

[35] Linden Research Inc. http://secondlife.com. Website, Zugriff von, 2007.

[36] ZeroC Inc. http://www.zeroc.com. Website, Zugriff von, 2008.

[37] S. Julier, Y. Baillot, M. Lanzagorta, D. Brown, and L. Rosenblum. Bars: Battlefield augmented reality system. In *NATO Symposium on Information Processing Techniques for Military Systems*, October 2000.

[38] Ed Kaiser, Alex Olwal, David McGee, Hrvoje Benko, Andrea Corradini, Xiaoguang Li, Phil Cohen, and Steven Feiner. Mutual disambiguation of 3d multimodal interaction in augmented and virtual reality. In *ICMI '03: Proceedings of the 5th international conference on Multimodal interfaces*, pages 12–19, New York, NY, USA, 2003. ACM.

[39] H. Kato and M. Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Proceedings of the 2nd International Workshop on Augmented Reality (IWAR 99)*, October.

[40] H. Kato, M. Billinghurst, K. Morinaga, and K. Tachibanaa. The effect of spatial cues in augmented reality video conferencing. In *Proceedings of the 9th International Conference on Human-Computer Interaction*, pages 478–481, New York, NY, USA, 2001. Lawrence Erlbaum Associates.

[41] Yoshifumi Kitamura, Andrew Smith, Haruo Takemura, and Fumio Kishino. A real-time algorithm for accurate collision detection for deformable polyhedral objects. *Presence: Teleoper. Virtual Environ.*, 7(1):36–52, 1998.

[42] Kiyoshi Kiyokawa, Yoshinori Kurata, and Hiroyuki Ohno. An optical see-through display for mutual occlusion with a real-time stereovision system. *Computers and Graphics*, 25(5):765–779, 2001.

[43] Matthias Kranz, Dominik Schmidt, Paul Holleis, and Albrecht Schmidt. A display cube as tangible user interface. In *In Adjunct Proceedings of the Seventh International Conference on Ubiquitous Computing (Demo 22)*, September 2005.

[44] S. Kuschfeldt, M. Schulz, T. Reuding, M. Holzner, and T. Ertl. Advanced visualization of crashworthiness simulations using virtual reality techniques. In *Proceedings of the Conference on High Performance Computing in Automotive Design, Engineering, and Manufacturing, Paris*, Paris, October 7-10 1996. Silicon Graphics Inc. / Cray Research.

[45] Shaun W. Lawson, John R. G. Pretlove, Alison C. Wheeler, and Graham A. Parker. Augmented reality as a tool to aid the telerobotic exploration and characterization of remote environments. *Presence: Teleoperators and Virtual Environments*, 11(4):352–367, 2002.

[46] C. Lee and Y. Xu. Online, interactive learning of gestures for human/robot interfaces. In *Proceedings of IEEE International Conference on Robotics and Automation*, 1996.

[47] T. Lee and T. Höllerer. Handy ar: Markerless inspection of augmented reality objects using fingertip tracking. In *International Symposium on Wearable Computers (IEEE ISWC) 2007*, October 2007.

[48] M. A. Livingston, L. J. Rosenblum, S. J. Julier, D. Brown, Y. Baillot, J. E. Swan II, J. L. Gabbard, and D. Hix. An augmented reality system for military operations in urban terrain. In *Proceedings of Interservice/Industry Training, Simulation, and Education Conference*, 2002.

[49] Orad Hi Tec Systems Ltd. http://www.orad.tv. Website, Zugriff von, 2007.

[50] LucasArts. http://www.lucasarts.com. Website, Zugriff von, 2008.

[51] H. Maas. Robust automatic surface reconstruction with structured light. 29:709–713, 1992.

[52] Blair MacIntyre, Maribeth Gandy, Jay Bolter, Steven Dow, and Brendan Hannigan. Dart: The designer's augmented reality toolkit. In *ISMAR '03: Proceedings of the The 2nd IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 329–330, Washington, DC, USA, 2003. IEEE Computer Society.

[53] Ph. Mackensen. *Auditive Localization. Head movements, an additional cue to localization.* PhD thesis, 2004.

[54] J. Marescaux, F. Rubino, M. Arenas, D. Mutter, , and L. Soler. Augmented-reality-assisted laparoscopic adrenalectomy. *JAMA*, 292(18):2214–2215, 2004.

[55] Frank Melchior, Tobias Laubach, and Diemer de Vries. Authoring and user interaction for the production of wave field synthesis content in an augmented reality system. In *ISMAR '05: Proceedings of the Fourth IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 48–51, Washington, DC, USA, 2005. IEEE Computer Society.

[56] Metaio. http://www.metaio.com. Website, Zugriff von, 2006.

[57] Paul Milgram and Fumio Kishino. A taxonomy of mixed reality visual displays. *IEICE Transactions on Information Systems*, E77-D(12), December 1994.

[58] H. Müller. Vtm - Videobasiertes Tracking-modul, 2003.

[59] Thomas B. Moeslund, Moritz Störring, and Erik Granum. Pointing and command gestures for augmented reality. 2004.

[60] W. Niem. Robust and fast modelling of 3-d natural objects from multiple views. In *SPIE Proceedings - Image and Video Processing II*, pages 388–397, 1994.

[61] Trond Nilsen. Tankwar: Ar games at gencon indy 2005. In *ICAT '05: Proceedings of the 2005 international conference on Augmented tele-existence*, pages 243–244, New York, NY, USA, 2005. ACM.

[62] Marleigh Norton and Blair MacIntyre. Butterfly effect: An augmented reality puzzle game. In *ISMAR '05: Proceedings of the Fourth IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 212–213, Washington, DC, USA, 2005. IEEE Computer Society.

[63] Nintendo of Europe GmbH. http://www.nintendo.de. Website, Zugriff von, 2008.

[64] Sharon Oviatt. Ten myths of multimodal interaction. *Communications of the ACM*, 42(11):74–81, 1999.

[65] Wouter Pasman, Charles Woodward, Mika Hakkarainen, Petri Honkamaa, and Jouko Hyväkkä. Augmented reality with large 3d models on a pda: implementation, performance and use experiences. In *VRCAI '04: Proceedings of the 2004 ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry*, pages 344–351, New York, NY, USA, 2004. ACM.

[66] James Patten. http://www.jamespatten.com/audiopad. Website, Zugriff von, 2007.

[67] Brian Peek. http://blogs.msdn.com/coding4fun/archive/2007/03/14/1879033.aspx. Website, Zugriff von, 2008.

[68] Wayne Piekarski and Bruce Thomas. Arquake: the outdoor augmented reality gaming system. *Communications of the ACM*, 45(1):36–38, 2002.

[69] Polhemus. http://www.polhemus.com/. Website, Zugriff von, 2008.

[70] Muriel Pressigout. Real-time markerless tracking for augmented reality: The virtual visual servoing framework. *IEEE Transactions on Visualization and Computer Graphics*, 12(4):615–628, 2006. Member-Andrew I. Comport and Member-Eric Marchand and Member-Francois Chaumette.

[71] F. Raab, E. Blood, T. Steiner, and R. Jones. Magnetic position and orientation tracking system. September 1979.

[72] Aditya Ramamoorthy, Namrata Vaswani, Santanu Chaudhury, and Subhashis Banerjee. Recognition of dynamic hand gestures. *Pattern Recognition*, 36(9):2069–2081, 2003.

[73] H. Regenbrecht, C. Ott, M. Wagner, T. Lum, P. Kohler, W. Wilke, and E. Mueller. An augmented virtuality approach to 3d videoconferencing. In *ISMAR '03: Proceedings of the The 2nd IEEE and ACM International Symposium on Mixed and Augmented Reality*, page 290, Washington, DC, USA, 2003. IEEE Computer Society.

[74] Stefan Reifinger, Frank Wallhoff, Markus Ablaßmeier, Tony Poitschke, and Gerhard Rigoll. Static and dynamic hand-gesture recognition for augmented reality applications. In *HCI Intern. '07: Proceedings of the 12th Conference of Human-Computer-Interaction*, pages 728–737, Heidelberg, Germany, 2007. Springer-Verlag.

[75] Steve Russell. http://de.wikipedia.org/wiki/spacewar. Website, Zugriff von, 2008.

[76] Robert Sablatnig, Srdan Tosovic, and Martin Kampel. Combining shape from silhouette and shape from structured light for volume estimation of archaeological vessels. In *ICPR '02: Proceedings of the 16 th International Conference on Pattern Recognition (ICPR'02) Volume 1*, pages 364–367, Washington, DC, USA, 2002. IEEE Computer Society.

[77] B. Schuller, M. Ablaßmeier, R. Müller, S. Reifinger, T. Poitschke, and G. Rigoll. Speech communication and multimodal interfaces. In *Advanced Man Machine Interaction*, pages 141–190, Heidelberg, Germany, 2006. Springer-Verlag.

[78] ADXL320: Small and Thin ±5g Accelerometer. http://www.analog.com. Website, Zugriff von, 2006.

[79] Russel Smith. http://www.ode.org. Website, Zugriff von, 2007.

[80] Didier Stricker and Thomas Kettenbach. Real-time and markerless vision-based tracking for outdoor augmented reality applications. In *ISAR '01: Proceedings of the IEEE and ACM International Symposium on Augmented Reality (ISAR'01)*, page 189, Washington, DC, USA, 2001. IEEE Computer Society.

[81] Satoshi Suzuki and Keiichi Abe. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, 30(1):32–46, 1985.

[82] Gunther Theile and Helmut Wittek. Wave field synthesis : A promising spatial audio rendering concept. *Acoustical science and technology*, 25(6):393–399, 2004.

[83] A. Torger and A. Farina. Real-time partitioned convolution for ambiophonics surround sound. pages 195–198, 2001.

[84] Trivisio. http://www.trivisio.com. Website, Zugriff von, 2007.

[85] Mihran Tuceryan, Yakup Genc, and Nassir Navab. Single-point active alignment method (spaam) for optical see-through hmd calibration for augmented reality. *Presence: Teleoper. Virtual Environ.*, 11(3):259–276, 2002.

[86] Ellison C. Urban. The information warrior. pages 493–501, 2000.

[87] Florian Völk, Stefan Kerber, Hugo Fastl, and Stefan Reifinger. Design und Realisierung von virtueller Akustik für ein Augmented-Reality-Labor. In *Tagungsband DAGA 2007*, März 2007.

[88] D. Wagner, T. Pintaric, F. Ledermann, and D. Schmalstieg. Towards massively multi-user augmented reality on handheld devices. In *Proceedings of the Third International Conference on Pervasive Computing (Pervasive 2005)*, pages 208–219, 2005.

[89] Daniel Wagner and Dieter Schmalstieg. Artoolkitplus for pose tracking on mobile devices. In *Proceedings of 12th Computer Vision Winter Workshop (CVWW'07)*, Februar.

[90] U. Zölzer. Digitale Audiosignalverarbeitung. 2005.

[91] M. Zollner and E. Zwicker. Elektroakustik. 1993.