HAW HAMBURG

MASTERTHESIS
Niklas Gerwens

Eine Hardware-Abstraktionsschicht für Interaktionen in VR-Applikationen

FAKULTÄT TECHNIK UND INFORMATIK
Department Informatik

Faculty of Computer Science and Engineering
Department Computer Science

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN HAMBURG
Hamburg University of Applied Sciences

Niklas Gerwens

Eine Hardware-Abstraktionsschicht für Interaktionen in VR-Applikationen

Masterarbeit eingereicht im Rahmen der Masterprüfung
im Studiengang *Master of Science Informatik*
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Kai von Luck
Zweitgutachter: Dr. Susanne Draheim

Eingereicht am: 08. Juli 2022

Niklas Gerwens

Thema der Arbeit

Eine Hardware-Abstraktionsschicht für Interaktionen in VR-Applikationen

Stichworte

Virtuelle Realität, Hardware-Abstraktionsschicht, Mensch-Computer-Interaktion, Interaktionsdesign, VR-Eingabegeräte, VR-Ausgabegeräte, Hardware Abstraktion

Kurzzusammenfassung

In dieser Masterarbeit wird eine Hardware-Abstraktionsschicht für VR-Hardware entworfen. Das Ziel ist es Interaktionsdesigner bei der Gestaltung von interaktiven VR-Anwendungen zu unterstützen, indem die Integration von Eingabe- und Ausgabegeräten in das VR-System erleichtert wird. Dafür werden abstrakte Schnittstellen und Modelle für verschiedene Hardwaretypen eingeführt, die Interaktionsdesigner an Stelle von den diversen Schnittstellen und Datenformaten konkreter Hardware benutzen können. Die Übersetzung der abstrakten Modelle in die hardwarespezifischen Modelle erfolgt durch eine Zuordnung der entsprechenden Methoden und Parameter. Der Interaktionsdesigner kann so unabhängig von der Hardware arbeiten und einfacher Eingabe- und Ausgabegeräte austauschen. Weiterhin können neue Geräte ohne Änderungen im Design eingebunden werden. Es wird nur eine Zuordnung zur entsprechenden abstrakten Schnittstelle benötigt. Die Hardware-Abstraktionsschicht wird für eine Evaluation in ein Beispielprojekt integriert, um die Erfüllung der Anforderungen zu prüfen.

Niklas Gerwens

Title of Thesis

A hardware abstraction layer for interactions in VR-applications

Keywords

virtual reality, hardware abstraction layer, human-computer interaction, interaction design, VR input devices, VR output devices, hardware abstraction

Abstract

In this master thesis, a hardware abstraction layer for VR hardware is designed. The objective is to support interaction designers in designing interactive VR applications by facilitating the integration of input and output devices into the VR system. For this purpose, abstract interfaces and models for different hardware types are introduced which interaction designers can use instead of the various interfaces and data formats of specific hardware. The abstract models are translated into the hardware-specific ones by mapping the corresponding methods and parameters. The interaction designer can thus work independently of the hardware and more easily exchange input and output devices. Furthermore, new devices can be integrated without changes in the design. Only a mapping to the corresponding abstract interface is required. The hardware abstraction layer is integrated into an example project for an evaluation to check the fulfillment of the requirements.

Inhaltsverzeichnis

1 Einleitung **1**

1.1 Motivation . . . 1
1.2 Zielsetzung . . . 2
1.3 Aufbau der Arbeit . . . 3

2 Analyse **4**

2.1 Zielsetzung . . . 4
2.2 VR-System . . . 5
2.3 Human-Computer Interaction . . . 7
2.4 Input . . . 8
2.4.1 Interaktionsdesign . . . 8
2.4.2 Eingabegeräte . . . 10
2.4.3 Zusätzliche Artefakte . . . 11
2.4.4 Hand-Tracking . . . 12
2.4.5 Hand-unabhängige Eingabe . . . 13
2.5 Feedback . . . 13
2.5.1 Visuelles Feedback . . . 14
2.5.2 Auditives Feedback . . . 17
2.5.3 Haptisches Feedback . . . 18
2.6 Rahmenbedingungen . . . 20
2.7 Anforderungen . . . 22

3 Design **23**

3.1 Aufbau der Hardware-Abstraktionsschicht . . . 23
3.2 Szenarien . . . 25
3.2.1 Szenario: Dachluke öffnen . . . 25
3.2.2 Szenario: Fahrstuhl . . . 27
3.2.3 Erweiterte Anforderungen an das Design . . . 28

3.3 Hardware-Abstraktionsschicht 29
3.3.1 Übersicht 29
3.3.2 Zusätzliche Artefakte 30
3.3.3 Hand-Tracking 32
3.3.4 Hand-unabhängige Eingabe 34
3.3.5 Feedback Komponente 35
3.3.6 Fazit 36
3.4 Schnittstelle zum Interaktionsframework 37
3.5 Zusammenfassung 38

4 Evaluation **40**
4.1 Übersicht 40
4.2 Auswertung der Anforderungen 40
4.3 Technischer Ausblick 45

5 Zusammenfassung und Ausblick **46**

Literaturverzeichnis **48**

A Anhang **52**

Selbstständigkeitserklärung **54**

1 Einleitung

1.1 Motivation

Der Einsatz von Virtual Reality (VR) in verschiedenen Anwendungsbereichen hat sich in den letzten Jahren weiter etabliert und verbreitet. Gründe für diese Entwicklung sind vor allem die kostengünstige Verfügbarkeit von VR-Hardware wie Head-Mounted Displays (HMDs) und leistungsstarke Grafikkarten durch allgemeine Fortschritte in der Kommunikations- und Informationstechnologie sowie spezifisches Investment großer Unternehmen. Die Anwendungsbereiche sind hinsichtlich des fachlichen Rahmens und der konkreten Zielstellung vielfältig, verfolgen aber grundsätzlich die Idee von dem hohen Immersionsfaktor einer VR-Anwendung zu profitieren. Neben der immersiven Darstellung der virtuellen Welt, trägt dabei besonders die Interaktionsgestaltung mit der virtuellen Welt zur Immersion eines Nutzers bei. Ein Nutzer möchte eine virtuelle Welt nicht nur wahrnehmen, sondern auch durch Interaktionen beeinflussen [8].
Für das immersive Interaktionsdesign einer VR-Anwendung werden Eingabe- und Ausgabegeräte genutzt, welche in der Regel speziell entwickelt wurden, um Interaktionen mit der virtuellen Welt und entsprechendes Feedback möglichst der Realität entsprechend abzubilden. VR-Hardware unterscheidet sich aufgrund dieser Prämisse stark von den aus 2D-Umgebungen bekannten Geräten, wie Maus, Tastatur und Monitor. Während die Nutzer dank der natürlichen Gestaltung der Anwendung, Erfahrungen und Wissen aus der realen Welt transferieren können, müssen die Entwickler ihren Blickwinkel auf das Design von Ein- und Ausgabe neu ausrichten [21].
Die Wahl für den Anwendungskontext geeigneter Geräte spielt dabei eine wesentliche Rolle, um die Ziele der jeweiligen Anwendungen zu realisieren. Forschungen zu diesem Thema sind selten und können aufgrund der variierenden Anforderungen und Schwerpunkte der Kontexte kaum verallgemeinert werden. So haben Anwendungen für medizinische oder industrielle Trainingsszenarien ganz unterschiedliche Ansprüche an das Design von Interaktionen und die Präsentation der virtuellen Welt als Anwendungen zur Demonstration von Produkten oder zum Entertainment der Nutzer. Es bleibt der Ansatz

Untersuchungen für die konkrete Anwendung individuell durchzuführen. Dieser Vorgang wird jedoch durch fehlende Standards der Datenformate und Schnittstellen aktueller Geräte erschwert, die größtenteils prototypisch entwickelt werden. Entwickler sind daher häufig gezwungen sich am Anfang des Designprozesses für bestimmte Hardware zu entscheiden und gezielt für diese zu entwickeln. Ein späterer Wechsel der Hardware würde eine aufwändige Neugestaltung des gesamten Designprozesses voraussetzen.
Auch die Evaluation unterschiedlicher Geräte, die sich jeweils für bestimmte Interaktionen der Anwendung besser eignen könnten, wird durch die diverse Natur der Schnittstellen sehr kostspielig. Entwickler neigen deswegen dazu, ein Design mit einem generischen Hardwaresetup zu verwenden, auch wenn möglicherweise bessere Alternativen für anwendungsspezifische Setups existieren. Aus dieser Problemstellung ergibt sich die Zielsetzung dieser Arbeit.

1.2 Zielsetzung

Um den technischen Einschränkungen bei der Wahl von Hardware bei VR-Anwendungen entgegenzuwirken, wird in dieser Arbeit eine Hardware-Abstraktionsschicht für Interaktionen in VR-Applikationen entworfen. Das Ziel ist, die Integration von Eingabe- und Ausgabegeräten in VR-Anwendungen zu erleichtern und Interaktionsdesignern eine Gestaltung unabhängig von konkreten Hardwareeigenschaften zu ermöglichen. Das Prinzip einer Abstraktionsschicht ist es, eine abstrakte Schnittstelle zur Hardware zu schaffen. Jegliche Kommunikation mit der Hardware läuft über die Schnittstelle, welche zwischen den abstrakten Anfragen und den hardwarespezifischen Formaten übersetzt. Da der Designer nur die abstrakte Schnittstelle kennen muss, verliert er die Bindung an die konkrete Hardware. Während die Hardware-Abstraktionsschicht grundlegend Designer bei Untersuchungen zur Wahl geeigneter Hardware unterstützt, sind solche für den jeweiligen Anwendungskontext vom Designer durchzuführen und nicht Teil dieser Arbeit.

1.3 Aufbau der Arbeit

Diese Arbeit gliedert sich in fünf Kapitel. Nach diesem Abschnitt, welcher die Einleitung beendet folgt in Kapitel 2 eine Analyse der Grundlagen. Dabei wird zunächst der Aufbau eines VR-Systems und der Forschungsbereich der Human-Computer Interaction erläutert, da diese Themen zusammen den Rahmen der Abstraktionsschicht bilden. Anschließend folgt eine Übersicht über VR-Hardware durch eine Klassifizierung von Eingabe- und Ausgabegeräten. Die Analyse wird mit einer Vorstellung der Rahmenbedingungen und der Anforderungen an das Design abgeschlossen.
In Kapitel 3 wird das Design der Abstraktionsschicht erörtert. Hierfür werden nach einer kurzen Übersicht zwei Szenarien beschrieben, um die geforderten Leistungen der Abstraktionsschicht weiter zu spezifizieren. Es folgt eine Erklärung der einzelnen Komponenten und eine abschließende Zusammenfassung.
Die Evaluation des Designs erfolgt in Kapitel 4. Kern der Evaluation ist eine Auswertung der definierten Anforderungen durch die Analyse einer Integration der Abstraktionsschicht in ein Beispielprojekt. Zusätzlich wird ein technischer Ausblick auf mögliche Verbesserungen und Erweiterungen des Designs gegeben. Das Kapitel 5 fasst abschließend die Ergebnisse dieser Arbeit zusammen und vermittelt einen Ausblick auf zukünftige Anwendungsgebiete der Abstraktionsschicht.

2 Analyse

2.1 Zielsetzung

Das Ziel dieser Arbeit ist die Unterstützung von Entwicklern beim Design interaktiver VR-Szenarien. Dem Entwickler soll die Option zur Verfügung gestellt werden, Hardware für bestimmte Szenarien und Interaktionen ohne viel Aufwand zu integrieren und auszutauschen. Der technische Schwerpunkt liegt dabei auf der Optimierung der Integration von Eingabe- und Ausgabegeräten durch vereinheitlichte Strukturen und abstrahierte Schnittstellen in Form einer Hardware-Abstraktionsschicht.
Der Gedanke, abstrakte Schnittstellen für Computersysteme zu gestalten, ist nicht neu. Für klassische Desktop-Systeme existieren schon seit Jahrzehnten Möglichkeiten, unabhängig von konkreter Hardware zu arbeiten. Sowohl Eingabegeräte, wie Mäuse, Tastaturen und Joysticks als auch Ausgabegeräte, wie Monitore, Lautsprecher und Kopfhörer, sind Plug-and-Play für den Nutzer und können über das Betriebssystem und die Entwicklungsumgebung von Anwendungsentwicklern ohne spezifisches Wissen angesprochen werden. Dies ist hauptsächlich durch Standards bei Datenformaten und -übertragung sowie Hardware-Abstraktionsschichten auf Betriebssystem-, Treiber- oder Anwendungsebene möglich. Solch erprobte Vorgehensweisen und Mechanismen sind für den VR-Kontext nicht vorhanden und eine Gestaltung dieser, wird durch die Diversität der rapiden Entwicklung von Hardware-Prototypen für VR erschwert.
Erste Ansätze zur Abstraktion von VR-Hardware finden sich in Plug-in Frameworks wie Unity XR [28] und Steam VR [29]. Diese bieten Entwicklern abstrakte Schnittstellen für weit verbreitete Hardware, indem sie für die Geräte benötigte Software bündeln und Zuordnungen zwischen abstrakten und konkreten Hardware-Eigenschaften aufbauen. Dabei werden jedoch nur bestimmte Eingabe- und Ausgabegeräte ausgewählter Hersteller unterstützt und beide Frameworks sind nur begrenzt offen für Modifikationen.
Die Abstraktionsschicht dieser Arbeit soll ein breiteres Spektrum an Hardwaretypen bereitstellen, um Entwicklern auf der Suche nach der passenden Hardware eine größere

Vielfalt zu bieten. Dies umfasst auch die Möglichkeit, die Abstraktionsschicht jederzeit um Neuentwicklungen von Hardware Prototypen zu erweitern. Dabei spielen viele Aspekte beim Design einer solchen Abstraktionsschicht eine Rolle. So gilt es einerseits natürlich die Eigenschaften der Hardware bei der Gestaltung der Abstraktionsschicht zu beachten, andererseits sollen aber auch der Einsatzkontext, übergeordnete Systeme und etablierte Designprinzipien berücksichtigt werden. Für ein besseres Verständnis der Designentscheidungen bietet es sich deswegen an, zunächst die Bereiche zu betrachten, die den Rahmen der Abstraktionsschicht bilden, bevor die Eigenschaften von Eingabe- und Ausgabegeräten im Detail erörtert werden.

2.2 VR-System

Die Abstraktionsschicht soll Interaktionsdesigner bei der Umsetzung von interaktiven VR-Anwendungen unterstützen. Dabei wird für die Entwicklung und Ausführung einer VR-Anwendung ein VR-System als Basis benötigt. Ein VR-System ist ein spezielles Computersystem, welches drei Aufgaben besitzt. Es erfasst Informationen und Interaktionen des Nutzers als Eingabe, es simuliert eine virtuelle Welt und es erzeugt Reize für den Nutzer als Ausgabe. Für die Eingabe und Ausgabe des Systems werden unterschiedliche Geräte genutzt, um diverse Interaktionen des Nutzers zu registrieren und ein Spektrum an Feedback liefern zu können. Dabei wird das jeweilige Feedback durch ein Verhaltensmodell bestimmt, welches als Teil der Simulation sowohl Reaktionen auf die Eingaben des Nutzers als auch auf Änderungen in der virtuellen Welt, die von dem Nutzer unabhängig sind, generiert. Zusätzliche Komponenten ,wie Datenbanken zur persistenten Datenspeicherung oder Netzwerkschnittstellen für Mehrbenutzeranwendungen, bilden optionale Ergänzungen, sollte das System diese aufgrund bestimmter Anforderungen des Anwendungskontexts benötigen [8]. Eine graphische Darstellung der Komponenten eines VR-Systems und ihres Zusammenspiels findet sich in Abbildung 2.1.

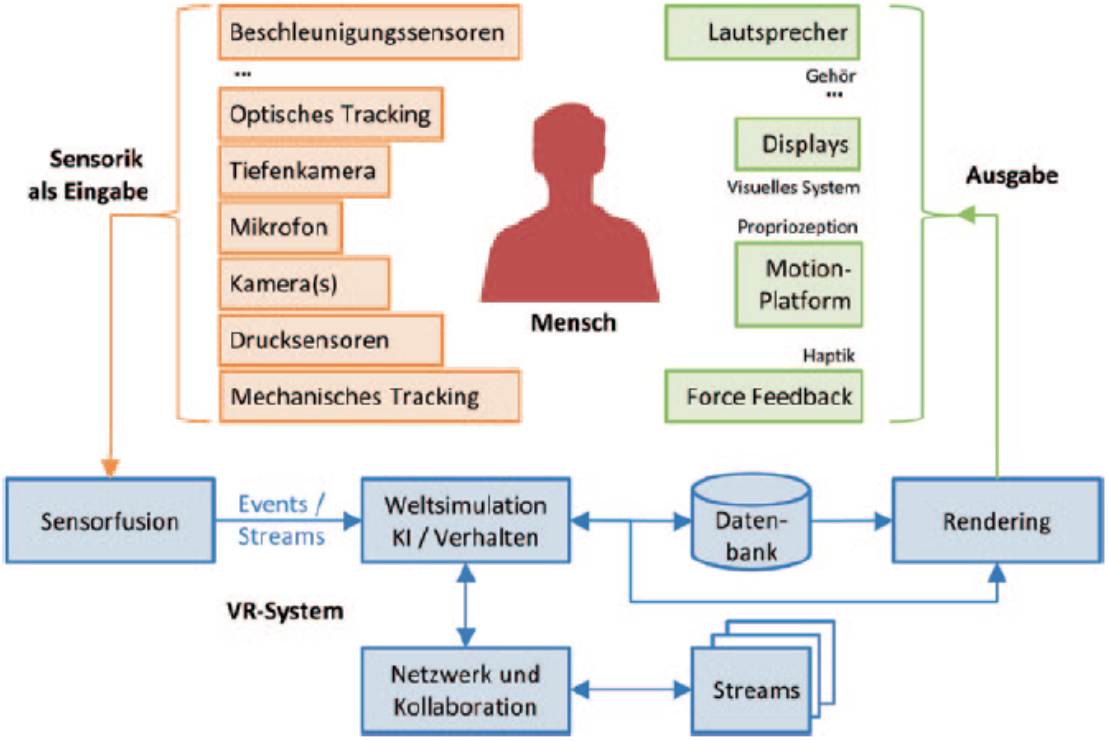


Abbildung 2.1: Aufbau eines VR-Systems [8]

2.3 Human-Computer Interaction

Abstrakt betrachtet ist die Interaktion von einem Nutzer mit der virtuellen Welt eine Kommunikation zwischen einem Menschen und dem Computer, auf dem die virtuelle Umgebung simuliert wird. Ein- und Ausgaben des Nutzers lassen sich demnach als eine Mensch-Computer-Interaktion (MCI, engl. Human-Computer Interaction, HCI) klassifizieren. In dem Forschungsbereich der MCI finden sich Forschungen, die sich mit dem Design, der Evaluierung und der Realisierung interaktiver, computerbasierter Systeme beschäftigen. Der Fokus liegt hierbei auf der benutzergerechten Gestaltung von Schnittstellen basierend auf Erkenntnissen der Informatik und anderer Gebiete, wie der Arbeitswissenschaft, der Psychologie und des Designs [13]. Dabei dient häufig das Prinzip der Usability (deut. Gebrauchstauglichkeit) als Leitgedanke beim Design von MCI-Schnittstellen. Usability beschreibt das Ausmaß, in dem eine Schnittstelle durch bestimmte Nutzer in einem definierten Kontext genutzt werden kann, um festgelegte Ziele effektiv, effizient und zufriedenstellend zu erreichen [6].
Für 2D-Systeme haben sich in den letzten Jahrzehnten hauptsächlich graphische Benutzerschnittstellen z. B. WIMP-Schnittstellen (Windows, Icons, Menus, Pointing) als ein Paradigma der MCI etabliert. Die Integration von WIMP-Schnittstellen in eine virtuelle 3D-Umgebung gestaltet sich jedoch sehr aufwändig, da einzelne 3D-Aktionen zum Teil auf mehrere 2D-Aktionen abgebildet werden müssen. Auch der Nutzer wird stärker belastet, weil er zusätzliche Schritte und die Zuordnung von 2D-Aktionen zu dem entsprechenden Teil der 3D-Aktion lernen muss. Dennoch kann es sinnvoll sein, klassische Benutzerschnittstellen als Ausgangspunkt für das Design von Interaktionen in VR zu verwenden, da Nutzer in der Regel schon bedeutende Kompetenzen mit klassischen Benutzungsschnittstellen erworben haben [9].
Im Gegensatz zu 2D-Systemen gibt es für VR-Systeme noch keine etablierten Vorgehensweisen und Richtlinien für das Design von benutzergerechten Schnittstellen. Deswegen ist man darauf angewiesen, Prototypen zur Erfüllung bestimmter Aufgaben und Anforderungen des Anwendungskontexts zu entwerfen und anschließend auf ihre Eignung hin zu überprüfen. Der Mangel an Standards zeigt sich dabei nicht nur bei den allgemeinen Designprinzipien, sondern spiegelt sich auch in der Hardware zur Ein- und Ausgabe wider. So befindet sich aktuell ein weites Spektrum an Hardwaretypen für VR mit diversen Datenformaten, Modellen und Schnittstellen im Umlauf [1]. Im Folgenden wird dieses Spektrum nach Ein- und Ausgabe getrennt aufgeschlüsselt und kategorisiert.

2.4 Input

Eingaben an ein Computersystem zeichnen sich durch die, vom Interaktionsdesigner gewählte, Interaktionstechnik und die Eigenschaften der genutzten Hardware aus. Dabei werden Technik und Hardware meist im Zusammenspiel betrachtet, da sich bestimmte Kombinationen besser eignen oder sogar voraussetzen. Eine Hardware-Abstraktionsschicht befindet sich zwischen den Geräten zur Eingabe und dem Verhaltensmodell der Simulation, in dem häufig auch die Interaktionstechnik verankert ist. Während also die Eigenschaften der zu abstrahierenden Hardware im Fokus der Analyse stehen, sind für die Abstraktionsschicht als Schnittstelle und Teil des Gesamtprozesses ebenfalls die existierenden Techniken und Designprinzipien für das Interaktionsdesign in VR relevant.

2.4.1 Interaktionsdesign

Grundsätzlich kann ein Nutzer mit einer virtuellen Welt auf jede erdenkliche Art und Weise interagieren. Dabei kann die Natur der Interaktion an die Abläufe und Gegebenheiten der realen Welt gebunden sein oder die Grenzen der physischen Welt überschreiten. Eine möglichst realistische Umsetzung erhöht die Immersion des Nutzers und ermöglicht bereits vorhandenes Wissen über die Interaktion aus der realen in die virtuelle Welt zu übertragen. Eine eher unnatürliche Gestaltung hingegen kann komplexe oder zeitaufwändige Aufgaben erleichtern und durch neue Möglichkeiten mit der Welt zu interagieren das Interesse des Nutzers wecken. Die Entscheidung, in welchem Maße eine realistische Abbildung von Interaktionen gewünscht ist, hängt hauptsächlich vom Anwendungskontext und den Zielen ab, welche die Entwickler mit der Anwendung verfolgen. Dazu nimmt die Verfügbarkeit von Ressourcen wie Hardware, physischer Raum und monetäre Mittel Einfluss auf die Entscheidung. Hierbei gilt in der Regel, dass ein natürliches Design mehr bzw. im Falle der Hardware spezifischere Ressourcen erfordert [21].
Die Entwicklung der Hardware-Abstraktionsschicht dieser Arbeit steht im Kontext einer VR-Schulung für Industrietechniker, in der vor allem Interaktionsabläufe und eine erste Orientierung vermittelt werden sollen, ohne die Techniker zur realen Industrieanlage transportieren zu müssen. Im Vordergrund steht der Lerntransfer des im virtuellen Raum angeeigneten Wissens in die reale Welt. Dafür soll der Zugang zu diesem Wissen für die Techniker erleichtert und die Kosten reduziert werden. Das Ziel des Interaktionsdesigners in VR ist demnach nicht, in jedem Fall eine perfekte Immersion oder Abbildung der Realität zu erreichen; je nach Anforderungen des Kontextes kann es ausreichen, natürliche

Interaktionen zu approximieren oder sogar sinnvoll sein, die Vorteile einer weniger natürlichen Umsetzung auszunutzen [4].
Abseits dieser allgemeinen Einordnung nach Grad der Natürlichkeit im Designprozess, lassen sich Interaktionen in VR auch anhand ihrer Aufgabe in der virtuellen Welt unterscheiden. Dörner et al. führen dazu vier übergeordneten Grundaufgaben ein: Selektion, Manipulation, Navigation und Systemsteuerung [8]. Unter Selektion finden sich alle Interaktionen, die ein Nutzer ausführt, um Objekte oder Teilobjekte der virtuellen Welt auszuwählen. Eine anschließende Interaktion, um den Zustand des ausgewählten Objektes zu verändern, fällt in den Aufgabenbereich der Manipulation. Da eine Manipulation eine vorher erfolgte Selektion voraussetzt und eine Selektion ohne eine folgende Manipulation sehr selten auftritt, werden die beiden Aufgaben beim Design von Interaktionen in VR grundsätzlich zusammen betrachtet. Damit der Nutzer eine Menge an Interaktionen ausführen kann, ist es häufig notwendig, dass er sich in dem 3D-Raum der Anwendung orientiert und bewegt. Hierfür gedachte Interaktionen fallen unter Navigation. Interagiert der Nutzer hingegen nicht mit der virtuellen Welt direkt, sondern mit dem übergeordneten System, spricht man von Interaktionen zur Systemsteuerung. Beispiele dafür sind das Wechseln der Szene oder das Ändern von Einstellungen.
Für ein besseres Verständnis wie eine Interaktion in VR auf verschiedene Arten umgesetzt werden kann und wie ein unterschiedliches Interaktionsdesign die Erfahrung der Nutzer beeinflusst, wird in dem Projekt „Toolchain für 3D-Objektmanipulation in VR-Trainingsumgebungen“ [10] ein simples Testszenario aufgebaut. Ziel des Szenarios ist das Öffnen einer Schranktür. Für die Selektion und Manipulation der Tür werden drei Ansätze mit entsprechender Hardware implementiert:

- Controller
- Hand (Optisches-Tracking)
- Sprachsteuerung

Beim ersten Ansatz hält der Nutzer einen Controller in der physischen Hand, dessen Position und Rotation über ein Tracking-System erfasst und durch ein 3D-Modell in der Szene dargestellt wird. Bewegt der Nutzer den Controller in die Nähe der virtuellen Tür, kann er diese durch einen Knopfdruck selektieren. Anschließend folgt die Tür innerhalb ihrer physikalischen Restriktionen der Position des Controllers, bis der Knopf losgelassen oder der Controller zu weit von der Tür entfernt wird.

Der zweite Ansatz nutzt eine optische Kamera, um die Hände des Nutzers zu tracken. Die Position und Rotation verschiedener Punkte der Hand werden einem 3D-Handmodell zugeordnet und kontinuierlich aktualisiert. Die Selektion der Tür erfordert auch hier die Unterschreitung einer maximalen Distanz zur Tür und ein Trigger-Event, wobei dieses nicht durch einen Knopfdruck, sondern durch das Schließen der Hand erfolgt. Die Manipulation der Tür verhält sich analog zum ersten Ansatz und endet bei Überschreitung der Distanzgrenze zwischen Hand und Tür oder wenn die Hand wieder geöffnet wird.
Während die ersten zwei Ansätze relativ ähnlich umgesetzt werden, verfolgt die Sprachsteuerung einen ganz anderen Ansatz. Über eine Wörterbuch-ähnliche Struktur werden Sprachbefehle den Zuständen von Objekten zugeordnet. Im Testszenario kann der Nutzer durch ein Mikrofon mit den Befehlen „Tür auf“ und „Tür zu“, die Tür direkt in die entsprechenden Zustände versetzen.
Alle drei Ansätze ermöglichen dem Nutzer mit der virtuellen Welt zu interagieren und den Zustand eines Objektes zu manipulieren. Unter dem Aspekt der Natürlichkeit verhalten sie sich dabei jedoch sehr unterschiedlich. Die Tür mit der Hand zu öffnen, kennt der Nutzer aus der Realität. Er benötigt keine zusätzliche Hardware, die er in der Hand halten oder deren Bedienung er erlernen muss. Dafür muss jedoch zusätzlich zur Position und Rotation der Hand auch die Gestik des Nutzers getrackt und ausgewertet werden. Wenn der Fokus der Anwendung nicht auf einem realistischen Ablauf zum Öffnen einer Tür beruht, kann durch die Sprachsteuerung der Vorgang erleichtert und durch Unabhängigkeit von spezifischer Hardware die Kosten reduziert werden. Die Sprachsteuerung erfordert in einem komplexeren Szenario jedoch eine Komponente zur Selektion des gewünschten Objektes, da es durchaus vorkommen kann, dass mehrere Objekte ein ähnliches Verhalten mit denselben Sprachbefehlen aufweisen.

2.4.2 Eingabegeräte

Das breite Spektrum an Eingabegeräten für VR ermöglicht vielfältige Optionen zur Gestaltung von Interaktionen mit virtuellen Umgebungen. Dies lässt sich bereits im Testszenario aus Kapitel 2.4.1 erkennen, in dem drei konkrete Eingabegeräte genutzt werden, um dieselbe Interaktion auf unterschiedliche Weise mit jeweiligen Vor- und Nachteilen umzusetzen. Dabei ist jedoch auch deutlich geworden, wie unterschiedlich einzelne Eingabegeräte funktionieren und in das Interaktionsdesign eingebunden werden. Trotz dieser Heterogenität der Hardware lassen sich aktuelle Eingabegeräte grob den Kategorien Zusätzliche Artefakte, Hand-Tracking sowie Hand-unabhängig zuordnen [17]. Die Kate-

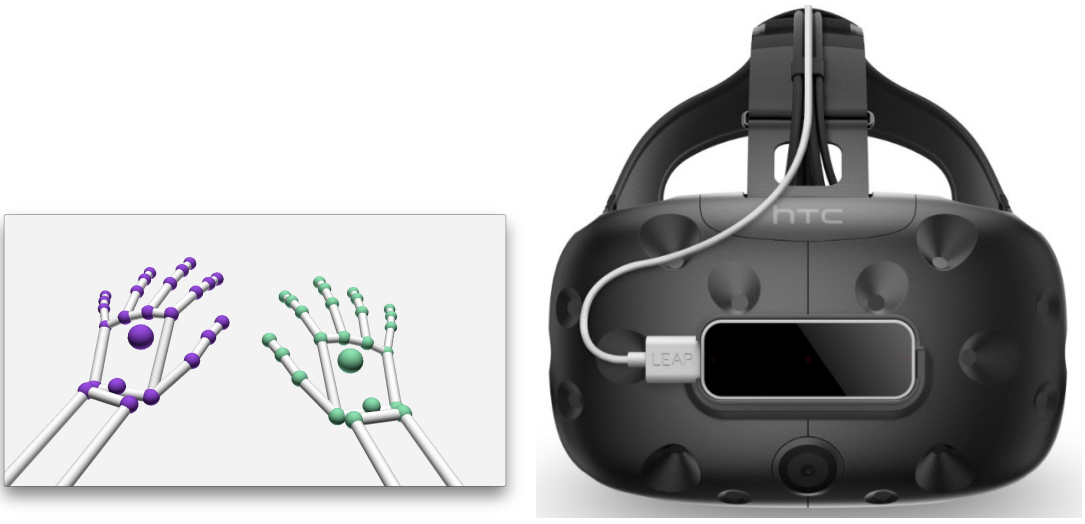
gorien bilden einen Ansatz, um Eingabegeräte anhand ihrer Eigenschaften wie erfasster Datenstrukturen und Datenschnittstellen sowie der allgemeinen Art und Weise der Bedienung durch den Nutzer zu klassifizieren.

2.4.3 Zusätzliche Artefakte

Unter die Kategorie Zusätzliche Artefakte fallen alle Eingabegeräte, welche von dem Anwender in der Hand gehalten werden. Die Grundlage für die Eingabe bildet meistens die Bestimmung von Position und Rotation des Artefakts durch eingebaute Sensoren oder ein externes Tracking-System. Aus den Positions- und Rotationsdaten lässt sich eine Abbildung des Artefakts in der virtuellen Welt zur Interaktion mit Objekten generieren. Weiterhin besitzen einige Artefakte zusätzliche Mechanismen und Sensoren, um weitere Eingaben des Anwenders zu erfassen. Häufig vorhanden sind Knöpfe, Joysticks, Touchpads oder Sensoren zur Erfassung von Berührung, Druck und Beschleunigung. Die so gewonnenen Eingabeparameter bieten dem Interaktionsdesigner mehr Möglichkeiten beim Design von Interaktionen zur Selektion und Manipulation von Objekten. Auch findet sich bei einigen Artefakten die Funktion haptisches Feedback zu vermitteln. Dies wird in Kapitel 2.5.3 weiter erörtert.
Zusätzliche Artefakte können demnach eine breite Masse an Daten erfassen, die als Eingabe interpretiert werden können. Die verschiedenen Eingabegeräte besitzen dabei meist vom Hersteller vorgegebene Datenmodelle und Schnittstellen für den Zugriff auf die erfassten Daten. Während diese Modelle und Schnittstellen innerhalb der Kategorie grundsätzlich sehr ähnlich aufgebaut sind, gibt es jedoch keinen übergreifenden Standard. Zur Integration unterschiedlicher Artefakte muss der Interaktionsdesigner eine Lösung für jedes Eingabegerät entwickeln, welches eingebunden werden soll.
Im Zuge des Interaktionsdesigns erhält der Designer von den Artefakten eine Menge an Daten, aber es existiert keine direkte Zuordnung von diesen zu konkreten Selektions- oder Manipulationsinteraktionen. Diese Zuordnung muss der Designer in dem Verhaltensmodell festlegen. Hierbei werden für Probleme wie die Objektmanipulation außerhalb der Reichweite oder fehlende Genauigkeit der Hardware häufig zusätzliche Interaktionstechniken benötigt [17].

2.4.4 Hand-Tracking

Eingabegeräte der Hand-Tracking-Kategorie nutzen meist optische Tracking-Verfahren, um Position und Rotation der Hand zu bestimmen. Dabei verfolgt das Gerät mit einer internen Kombination aus Sensoren und Algorithmen definierte Punkte der Hände. Aus diesen Punkten wird anschließend ein virtuelles Hand-Modell rekonstruiert. Zusätzlich erfolgt häufig eine Optimierung des Hand-Modells durch die Berechnung von nicht aktiv verfolgten Handelementen. Dies verbessert einerseits die Darstellung in der virtuellen Umgebung und bietet andererseits dem Interaktionsdesigner mehr Eingabewerte, die er referenzieren kann.
Die Sensorik für Hand-Tracking befindet sich in den meisten Fällen am Körper des Anwenders, da die Genauigkeit optischer Sensoren stark von der Distanz abhängig ist. Einige Hersteller wie z. B. Leap Motion, liefern dafür Halterungen, um das Produkt am HMD zu befestigen. Andere verbauen die Sensoren direkt im HMD. In beiden Fällen muss der Interaktionsdesigner beachten, dass er nur Informationen der Eingabegeräte erhält, wenn der Anwender seine Hände im Blickfeld hat [24]. In Abbildung 2.2 ist die Installation der Leap Motion an einem HMD und das Hand-Modell der Leap Motion als ein Beispiel für die Eingabegeräte der Hand-Tracking-Kategorie dargestellt.



(a) Leap Hand-Modell [31] (b) Leap Motion HMD Befestigung [19]

Abbildung 2.2: Die Leap Motion als Beispiel für die Hand-Tracking Kategorie

Im Vergleich zu Geräten der Zusätzlichen-Artefakt-Kategorie erlaubt Hand-Tracking-Hardware allgemein ein natürlicheres Interaktionsdesign. Viele Interaktionen aus der realen Welt, die der Anwender mit seinen Händen ausführt, können direkt abgebildet werden. Der Anwender kann so sein Wissen leichter transferieren, da er die Interaktion bereits aus der Realität kennt und nicht die Bedienung eines Artefakts erlernen muss. Schwieriger gestaltet es sich bei der Übersetzung der Daten, in über die sechs Freiheitsgrade hinausgehende Interaktionen. Hierfür wird eine Implementation einer Gestenerkennung oder anderer aufbauender Ansätze benötigt [17].

2.4.5 Hand-unabhängige Eingabe

Die Kategorie Hand-unabhängig umfasst alle Eingabegeräte, die weder in der Hand gehalten werden noch sich mit der Erfassung der Hände beschäftigen. Grundsätzlich lassen sich also alle Geräte der Kategorie zuordnen, die nicht in die beiden anderen Kategorien passen. Ein explizites Merkmal oder ein ähnliches Datenmodell, welches die Geräte miteinander teilen, wird nicht vorausgesetzt. Deswegen ist die Diversität innerhalb der Hand-unabhängig-Kategorie am größten. Beispiele für die Kategorie, welche in der Praxis regelmäßig Anwendung finden, sind Sprachsteuerung und Blickverfolgung [17].
Während die Eingabewerte der Geräte sich stark unterscheiden, existieren grundlegend ähnliche Bedingungen für die Integration in das Interaktionsdesign. So besitzen die Geräte keine Verortung im virtuellen Raum durch Tracking und einer entsprechenden Abbildung des Eingabegeräts. Die Selektion von Objekten kann demnach nicht durch die Distanz zwischen Abbildung und Objekt erfolgen, sondern muss durch Zuordnungen der Eingabewerte auf bestimmte Objekte oder durch Kombination mit anderen Ansätzen umgesetzt werden. Ebenfalls muss bedacht werden, dass es nicht immer möglich ist, Metaphern aus der realen Welt zu transferieren. Der Anwender muss die Interaktion mit dem entsprechenden Eingabegerät erst erlernen, wodurch abhängig von der Komplexität des Designs ein erhöhter Lernaufwand unter Verlust von Natürlichkeit entsteht [8].

2.5 Feedback

Um die virtuelle Umgebung des VR-Systems für den Nutzer wahrnehmbar zu machen, werden Ausgabegeräte genutzt. Dabei ist das Ziel, dem Nutzer ein möglichst immersives Erlebnis der virtuellen Welt zu liefern. Der Nutzer soll sich präsent in der virtuellen Welt

fühlen und sich wie in der realen Welt verhalten. Dazu werden über die Ausgabegeräte künstlich generierte Sinneseindrücke vermittelt, um die Wahrnehmung des Nutzers zu beeinflussen. Die Sinne, die ein Mensch zur Interpretation einer virtuellen Welt am meisten beansprucht, sind der visuelle, der akustische sowie der haptische Sinn, welches sich auch in der typischen Verwendung von Ausgabegeräten für VR-Systeme widerspiegelt. Weitere Sinne wie der olfaktorische (Riechen) oder der gustatorische Sinn (Schmecken) werden selten virtuell stimuliert und entsprechende Ausgabegeräte existieren meist nur in darauf spezialisierten Forschungslaboren [8].
Der grundlegende Einsatz von Ausgabegeräten für visuelles, auditives und haptisches Feedback ist bereits aus traditionellen Desktop-Umgebungen bekannt. Der Wechsel von 2D- auf 3D-Umgebungen, mit dem Nutzer als aktiver Faktor in der virtuellen Welt, bringt jedoch neue Anforderungen an die Hardware mit sich. Es wird die Möglichkeit benötigt, die 3D-Welt so zu präsentieren, dass der Nutzer bei Orientierung, Navigation und Eingabe mit 3D-Eingabegeräten in der virtuellen Welt durch entsprechendes Feedback unterstützt wird. Diese Zielstellung hat zu einer Entwicklung neuer, für 3D-Umgebungen optimierter Typen an Ausgabegeräten geführt. In den nächsten drei Kapiteln werden die Eigenschaften dieser Typen für die jeweiligen Feedback Kategorien vorgestellt [15].

2.5.1 Visuelles Feedback

Bei der visuellen Ausgabe soll der Nutzer durch Feedback des VR-Systems eine visuelle Darstellung der virtuellen Welt erhalten, die möglichst der Art und Weise entspricht, wie er die reale Welt wahrnimmt. Hierfür werden verschiedene Arten von Displays als Ausgabegeräte genutzt, um das zugrundeliegende 3D-Modell der virtuellen Welt in ein Bild zu überführen und dem Nutzer zu präsentieren. Der Begriff Display wird hier als Überbegriff verstanden, welcher für alle visuellen Ausgabegeräte verwendet wird. Für die Bilderzeugung wird eine virtuelle Kamera genutzt. Diese stellt genau die Objekte dar, die sich im Sichtvolumen der Kamera befinden. Dabei wird das Sichtvolumen durch einstellbare Kameraeigenschaften wie z. B. Öffnungswinkel, Brennweite sowie Position und Ausrichtung der Kamera in der virtuellen Welt definiert.
Die Anforderungen an ein Display für das visuelle Feedback umfassen eine Darstellung mit möglichst guter Auflösung und hohem Kontrast sowie die Abdeckung eines möglichst großen Sichtbereichs. Dabei lassen sich Displays in die Kategorien Monitore, Projektionssysteme und Head-Mounted Displays (HMDs) unterscheiden, welche grundsätzlich die Anforderungen unterschiedlich gut erfüllen. Klassische Monitore, wie man sie aus

Desktop-Umgebungen kennt, sind hell und besitzen gute Kontrastverhältnisse, sind aber dafür eingeschränkt in der Displaygröße. Bei Projektionssystemen ist die Displaygröße flexibel, aber der Kontrast ist schlechter und die Darstellung ist entweder dunkler oder bei Fokus auf hoher Lichtleistung des Projektors sehr teuer. Zusätzlich entstehen je nach Projektionsart weitere Probleme. Bei Projektionen aus dem Anwendungsbereich kann der Nutzer abhängig von seiner Position im Raum die Strahlen des Projektors blockieren. Dies kann durch eine Projektion ausgehend von der Rückseite der Präsentationsfläche vermieden werden. Eine solche Rückprojektion benötigt dafür jedoch mehr Platz für den Aufbau des VR-Systems. Allgemein sind Monitore und Projektionssysteme fast ausschließlich für stationäre Installationen geeignet. Anders sieht es bei HMDs aus, die aufgrund ihrer Mobilität aber auch anderer Eigenschaften viele Vorteile für den Gebrauch als visuelles Ausgabegerät eines VR-Systems bieten und als in der Praxis weit verbreitetes Standardgerät folgend weiter beschrieben werden sollen [8].
HMDs sind mobile Visualisierungsgeräte in Form eines Helmes oder einer Datenbrille, die der Nutzer am Kopf trägt. Dabei ist die Grundidee von HMDs nicht neu. Erste Prototypen von HMDs existierten bereits in den 1960er Jahren [26]. Der technologische Fortschritt und der VR-Trend der letzten Jahre haben jedoch die Entwicklung neuer HMDs ermöglicht, die kostengünstig produziert und so auch vermehrt außerhalb von Forschungslaboren eingesetzt werden können. Neben einem miniaturisierten Display zur Erzeugung eines Bildes, welches über eine Optik dem Nutzer vergrößert dargestellt wird, besitzen HMDs eine Elektronik-Komponente und eine Computerschnittstelle. Die Elektronik-Komponente übernimmt die Ansteuerung von im HMD integrierten Sensoren und dem Display. Dabei unterstützen Sensoren zur Bestimmung von Position und Rotation des Kopfes oder zur Blickverfolgung nicht nur das HMD bei der Ausgabe des Bildausschnitts der virtuellen Kamera, sondern können auch zur Eingabe genutzt werden. Die Schnittstelle ermöglicht die Übertragung der Video- und Sensordaten zwischen HMD und dem Rechnersystem, an dem das HMD angeschlossen ist [8].
In Anbetracht der Anforderungen an visuelle Ausgabegeräte überzeugen HMDs vor allem durch die Stereoskopie-Fähigkeit und das große Sichtfeld. Weiterhin bietet die Möglichkeit, visuelle Einflüsse der Realität vollständig auszublenden, ein erhöhtes Level an Immersion. Dafür leiden Auflösung und Kontrast unter der Einschränkung, alle Teile möglichst kompakt zu verpacken. Auch gilt es, den Tragekomfort und das Auftreten von Cybersickness bei langer Benutzung von HMDs zu beachten [3].
Während in den meisten Fällen für aktuelle VR-Anwendungen HMDs als visuelle Ausgabegeräte genutzt werden, gibt es auch einige Systeme, die Gebrauch von speziellen Displaysystemen machen. Displaysysteme bestehen aus einer Menge an Einzeldisplays,

die kombiniert werden, um ein größeres Sichtfeld abzudecken. Typische Formen sind dabei L-Shapes, Curved-Screens oder ein CAVE-System. CAVE-Systeme zeichnen sich durch die Abdeckung aller Blickrichtungen aus und erreichen dadurch ähnlich wie HMDs die visuelle Abschottung von Nutzern zur realen Welt. In Abbildung 2.3 ist der beispielhafte Aufbau eines CAVE-Systems dargestellt.

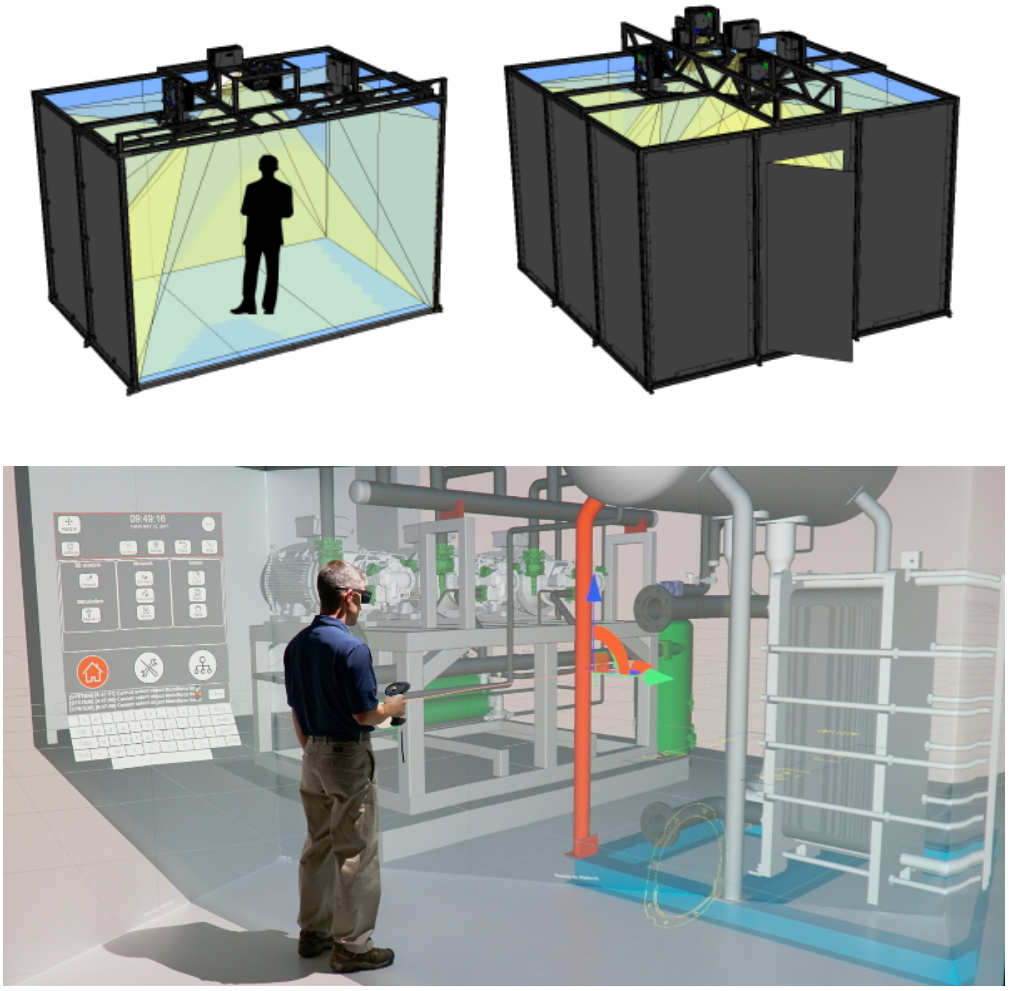


Abbildung 2.3: Aufbau eines CAVE-Systems [30]

Der Einsatz von Monitoren und Projektionssystemen bietet hohe Kontrast- und Auflösungswerte. Dabei können durch eine Kachelung vieler kleiner Displays die hohen Kosten von leistungsstarken Großgeräten umgangen werden. Für die Umsetzung von Displaysystemen werden jedoch zusätzliche Softwarelösungen für die geometrische Kalibration und

die Aufrechterhaltung von Helligkeits- und Farbuniformität zwischen den Einzeldisplays benötigt. Abhängig von der konkreten Anzahl an Displays und den genutzten Algorithmen steigt die Komplexität und die erforderte Leistung des Systems stark [8].
Der alleinige Einsatz von regulären Monitoren und Projektionssystemen für die visuelle Darstellung von 3D-Welten hat aufgrund der verfügbaren Alternativen stark abgenommen, kann aber als zusätzliches Instrument für bestimmte Szenarien sinnvoll sein. So können bei virtuellen Lernumgebungen die Ausbilder über einen Monitor verfolgen, was die Person in der VR-Welt macht oder Zuschauer auf einer Messe an der Demo einer VR-Anwendung teilhaben. Erweitert man dieses Setup um Möglichkeiten zur Eingabe über das zusätzliche Display, schafft man die Grundlage für asynchrones Gameplay [18].

2.5.2 Auditives Feedback

Auch wenn der visuelle Sinn die sicherlich bedeutsamste Informationsquelle bei der Wahrnehmung von virtuellen Welten ist, spielt auch der auditive Sinn eine immer wichtiger werdende Rolle. Ein simples Audiosystem mit klassischen Ausgabegeräten wie Kopfhörern oder Lautsprechern reicht, um den Nutzer bei der zeitlichen Zuordnung von Ereignissen zu unterstützen. Ein Beispiel wäre die Übermittlung eines akustischen Signals bei der Selektion von Objekten, damit der Nutzer weiß, dass die Selektion erfolgreich war und er mit der Manipulation beginnen kann.
Auditives Feedback kann dem Nutzer ebenfalls bei der räumlichen Orientierung innerhalb der virtuellen Welt helfen. Hierfür sind jedoch meist komplexere Audioinstallation wie Mehrkanalaudiosysteme notwendig, um entsprechende Orientierungshilfen geben zu können. Mehrkanalaudiosysteme setzen sich aus mindestens einem Hauptlautsprecher als Basis Audioquelle und mehreren Zusatzlautsprechern zur Unterstützung räumlicher Effekte zusammen. Dabei ist die räumliche Wahrnehmung meist nur innerhalb eines kleinen Bereichs (Sweet Spots) gut möglich. Eine gleichwertige Leistung im gesamtem Anwendungsbereich ist nicht gegeben und die Grenzen des Systems bei einer Anwendung mit viel Nutzerbewegung schnell erreicht [8].
Anders verhält es sich bei Systemen, die das Prinzip der Wellenfeldsynthese umsetzen. Das Grundprinzip der Wellenfeldsynthese ist, ein aufgenommenes Wellenfeld einer realen Begebenheit als synthetisches Wellenfeld wiederzugeben. Das Wellenfeld wird hierfür durch eine große Menge von Lautsprechern erzeugt, die um die Wiedergabefläche herum angeordnet werden. Audioquellen können innerhalb dieser Fläche beliebig positioniert werden. Dafür wird ein zentraler Rechner benötigt, der die Wiedergabe und die Positio-

nierung der Klänge über die Lautsprecher steuert [23]. Der durch eine Wellenfeldsynthese generierte Raumklang ist dabei innerhalb der gesamten Wiedergabefläche gleich potent und besitzt keine Sweet Spots wie herkömmliche Mehrkanalaudiosysteme [20].

2.5.3 Haptisches Feedback

Die audiovisuelle Darstellung einer virtuellen Welt durch für VR optimierte Ausgabegeräte bietet bereits ein immersives Erlebnis für Nutzer, lässt sich aber durch haptisches Feedback weiter ergänzen. Das Ziel der haptischen Ausgabe ist es, die virtuelle Welt für den Nutzer fühlbar zu machen. Dabei spielen verschiedene Sinne der haptischen Wahrnehmung eine Rolle. Neben der aktiven (haptischen) und passiven (taktilen) Wahrnehmung von Berührungen sind auch die Temperatur- und Schmerzwahrnehmung sowie die kinästhetische Wahrnehmung und die Propriozeption, Bestandteile der haptischen Wahrnehmung. Propriozeption und Kinästhesie basieren auf der Wahrnehmung von Reizen aus dem Körperinneren (Tiefensensibilität). Dabei vermittelt die Propriozeption einerseits Informationen über die Position des Körpers im Raum sowie die Stellung der Gelenke und des Kopfes (Lagesinn) und andererseits gibt sie Informationen über den Spannungszustand von Muskeln und Sehnen (Kraftsinn). Die Kinästhesie hingegen bietet eine allgemeine Wahrnehmung von Bewegungsempfindung und spezieller auch das Erkennen der Bewegungsrichtung [11].
Die Wahrnehmung von Berührung, Temperatur und Schmerz sind Teil der Oberflächensensibilität und erfassen Reize von außerhalb, über in der Haut liegende Rezeptoren. Diese Rezeptoren lassen sich in Mechanorezeptoren (Druck, Berührung und Vibration), Thermorezeptoren (Temperatur) und Nozizeptoren (Schmerz) unterteilen. Haptisches Feedback lässt sich durch die künstliche Stimulation der Rezeptoren mit entsprechenden Ausgabegeräten realisieren [12]. Wie im VR-Umfeld üblich existiert eine Vielzahl unterschiedlicher Prototypen für diesen Zweck. Neben einer allgemeinen Einteilung nach dem haptischen Sinn, der durch das Gerät beeinflusst wird, lassen sich die Ausgabegeräte auch nach der Position im Bezug auf den Nutzer unterscheiden. Angelehnt an die Aufteilung in der Arbeit „Enhancing Audiovisual Experience with Haptic Feedback: A Survey on HAV“ [5] werden folgend die Kategorien Wearable Devices, Handheld Devices, Haptic Seats und Non Physical Contact Devices vorgestellt. Dabei werden zum Teil verschiedene Typen an Ausgabegeräten für die Kategorien eingeführt, um die grundlegende Idee der Kategorie zu verdeutlichen. Eine vollständige Übersicht über alle aktuellen Ansätze ist aufgrund der Diversität sehr aufwändig und nicht Teil dieser Arbeit.

Wearable Devices

Wearable Devices sind Ausgabegeräte, die ein Nutzer trägt bzw. am Körper des Nutzers befestigt werden. Mit Wearable Devices können die Rezeptoren über größere Flächen oder sogar am gesamten Körper stimuliert werden. Die Anbringung der Geräte am Körper des Nutzers lassen die Hände frei und blockieren somit nicht die Option zur Eingabe über Zusätzliche Artefakte oder Hand-Tracking Eingabegeräte. Dafür ist die Stärke des haptischen Feedbacks häufig begrenzt, da z. B. die Motoren für mechanische Signale wie Vibration bei entsprechender Stärke schnell zu schwer zum Tragen werden [5].
Beispiele für Wearable Devices sind unter anderem Force Feedback Gloves und Electrical Muscle Stimulation (EMS). Force Feedback Gloves sind Handschuhe, die mehrere Aktoren besitzen, um durch Vibrationen haptisches Feedback zu vermitteln [32]. EMS hingegen nutzt am Körper befestigte Elektroden für elektrische Impulse zur Stimulation von Muskelkontraktionen [16]. Ebenfalls existieren Ansätze zur Beeinflussung der Temperaturwahrnehmung. Ragorin et al. benutzen in ihrer Arbeit [22] eine Konstruktion, bei der ein Nutzer Kühlelemente am Körper trägt, die über eine Schnittstelle von der VR-Umgebung aus gesteuert werden können.

Handheld Devices

Unter Handheld Devices werden die haptischen Ausgabegeräte zusammengefasst, die ein Nutzer in der Hand hält. Auch hier ist die Stärke des Feedbacks durch das Gewicht der Geräte eingeschränkt. Weit verbreitet in dieser Kategorie sind Controller und Zusätze für Controller [25]. Diese kombinieren den Zusätzliche Artefakte Ansatz zur Eingabe mit der Möglichkeit, haptische Rückmeldungen zu geben. So wird weiterhin die Eingabe über die Hände des Nutzers ermöglicht und eine gewisse physische Nähe zwischen den Positionen von Eingabe und darauf basierender Ausgabe von Feedback geschaffen. Die grundlegende Funktion von Controllern als Ausgabegerät für haptisches Feedback ist dem Nutzer dabei bereits aus traditionellen Systemen bekannt [5].

Haptic Seats

Haptic Seats sind modifizierte Stühle, die einem Nutzer taktiles Feedback bieten. Dabei besitzen die Stühle meist Aktoren für Vibration an unterschiedlichen Stellen und bei einigen Modellen die Option, den Stuhl auf mehreren Achsen zu bewegen. So lassen

sich nicht nur einzelne Rezeptoren der Nutzer durch Vibration oder Druck stimulieren, sondern durch eine Bewegung des Stuhls auch die Bewegungswahrnehmung des Nutzers im Allgemeinen. Haptic Seats benötigen für dieses Spektrum an Feedback leistungsstarke Aktoren, was sich wiederum in den hohen Kosten bemerkbar macht. Bekannt sind Haptic Seats vor allem aus Freizeitparks oder komplexen Simulationssystemen und finden sich im VR-Kontext häufig auch nur in spezifischen Trainingsszenarien [5].

Non Physical Contact Devices

Die letzte Kategorie beinhaltet Ausgabegeräte, die sich nicht im direkten Kontakt mit dem Nutzer befinden. Aufgrund der Distanz ist es in den meisten Fällen nicht möglich, bestimmte Rezeptoren des Nutzers zu beeinflussen, sondern nur die allgemeine haptische Wahrnehmung. Durch Ventilatoren oder Heizkörper kann die Temperaturwahrnehmung des Nutzers manipuliert werden [7]. Hierfür können handelsübliche Geräte genutzt werden. Einzig eine Schnittstelle zur Steuerung über die virtuelle Umgebung wird benötigt. Bei Nutzung spezieller Hardware ist es auch möglich, fokussierten Druck über Distanz zu übertragen. Dies kann z. B. mit Ausgabegeräten erfolgen, welche Gebrauch von Ultraschall [14] oder Luftdruck [27] machen.

2.6 Rahmenbedingungen

Neben den theoretischen Grundlagen und dem aktuellen Stand der VR-Hardware sind weitere Faktoren aus dem Umfeld der Arbeit für das Verständnis der später vorgestellten Designentscheidungen wichtig. Grundlegend befindet sich diese Arbeit im Rahmen eines Projekts zur Entwicklung einer VR-Schulung für Wartungstechniker einer Offshore-Windkraftanlage. Das Projekt wird in der Laborumgebung des Creative Space for Technical Innovations (CSTI) an der HAW Hamburg in Zusammenarbeit mit einem Industriepartner realisiert. Dabei ist die Hardware-Abstraktionsschicht nicht Teil der Schulung, sondern ein eigenständiges Projekt, um weiterführende Möglichkeiten in dem Kontext zu erforschen. Das Design der Hardware-Abstraktionsschicht ist dennoch eng verbunden mit dem der VR-Schulung, da alle Referenzszenarien aus der VR-Schulung entstammen und die Komponenten der Abstraktionsschicht zur Evaluation in bestimmte Abläufe der Schulung integriert werden. Für diese Integration ist es notwendig, dass die Abstraktionsschicht einerseits in der selben Entwicklungsumgebung entworfen wird und andererseits

eine Kompatibilität mit dem Interaktionsframework aufweist, welches in der Schulung für das Design von Interaktionen genutzt wird.
Die Abstraktionsschicht wird deswegen in Unity entwickelt. Als eine der populärsten Entwicklungsumgebungen für VR-Anwendungen, bietet Unity bereits Support und Schnittstellen für diverse Eingabe- und Ausgabegeräte. Auch lassen sich Erkenntnisse aus anderen VR-Projekten transferieren, die im CSTI größtenteils in Unity entwickelt werden.
Das Interaktionsframework der VR-Schulung wurde von Jonathan Becker im Zuge seiner Masterarbeit „Entwicklung eines Interaktionsframeworks zur Erstellung interaktiver Virtual Reality Szenarien“ entworfen [2]. Dabei war die parallele Entwicklung der Arbeiten und die allgemeine Zusammenarbeit im gleichen Projektkontext ausschlaggebend für das Design der Schnittstelle zwischen Abstraktionsschicht und Interaktionsframework. In beiden Arbeiten finden sich deswegen häufig Referenzen auf die jeweils andere Arbeit. Der Begriff Interaktionsframework ohne weitere Spezifikation verweist folgend immer auf die Arbeit von Jonathan Becker.
Im Ansatz verfolgt das Interaktionsframework dasselbe Ziel wie die Abstraktionsschicht. Interaktionsdesigner sollen bei der Gestaltung interaktiver VR-Anwendungen unterstützt werden. Dabei setzt das Interaktionsframework jedoch an einer anderen Stelle im VR-System an. Es steht nicht die Integration von Eingabe- und Ausgabegeräten im Mittelpunkt, sondern der Aufbau und die Definition des Verhaltensmodells. Durch eine Vielzahl an Skripten können verschiedene Objekte mit einem grundlegenden Verhalten versehen und anhand von Parametern weiter spezifiziert werden. Ein Eventsystem bietet die Möglichkeit, nutzerunabhängiges Verhalten auszulösen und bildet die Grundlage für das Story Modul, welches die Abbildung von Zusammenhängen zwischen einzelnen Interaktionen ermöglicht. Zusätzlich werden GUIs und Fehlerfeedback Optionen angeboten, um den Designer die Bedienung des Frameworks zu erleichtern.

2.7 Anforderungen

Aus der Analyse ergeben sich einige Anforderungen an die Hardware-Abstraktionsschicht, die im Design berücksichtigt werden müssen. Grundsätzlich sollen Entwickler von VR-Anwendungen durch die Abstraktionsschicht in der Lage sein, unabhängig von der konkreten Hardware, Interaktionen zu gestalten. Dabei binden Entwickler nicht direkt die Abstraktionsschicht in das Design von ihren Interaktionen ein, sondern nutzen die Komponenten dieser indirekt über das Interaktionsframework. Die Schnittstelle zwischen dem Interaktionsframework und der Abstraktionsschicht bildet folglich einen wichtigen Teil des Designs, in dem die Ansprüche des Frameworks beachtet werden müssen.
Das Framework wurde im Rahmen einer VR-Schulung für Industrietechniker entwickelt, unterstützt aber grundlegend die Entwicklung interaktiver VR-Szenarien in diversen Anwendungskontexten. Diese haben wiederum unterschiedliche Ansprüche an die Hardware für Ein- und Ausgaben des virtuellen Systems. Die Hardware-Abstraktionsschicht muss deswegen ein breites Spektrum an Geräten und Gerätetypen zur Verfügung stellen. Dieses Spektrum muss leicht erweiterbar sein, um die rapide Entwicklung von Hardware für VR-Anwendungen aufzufangen und somit die Abstraktionsschicht auch für zukünftige Projekte auf dem aktuellen Stand der Technik zu halten.
Weitere Anforderungen ergeben sich aus dem Mangel an bewährten Vorgehensweisen in Bezug auf den Einsatz von Hardware in VR. So existieren kaum etablierte Zuordnungen davon, welche Hardware für einzelne Interaktionstypen am Besten geeignet ist. Dazu kommt, dass abhängig vom Kontext und Ziel der Anwendung die Zuordnung unterschiedlich ausfallen kann. Um einen Entwickler bei dieser Entscheidung zu unterstützen, sollte ihm die Option verschiedene Hardwaretypen auszuprobieren ermöglicht werden. Demnach muss die Abstraktionsschicht die Möglichkeit bieten, die Hardware für bestimmte Interaktionstypen des Frameworks auszutauschen.
Auch bei den Formaten zur Datenvermittlung bereitet der Mangel an Standards Probleme. Die Hersteller von VR-Hardware benutzen unterschiedliche Datenformate und -modelle. Um die oben genannten Anforderungen zu erfüllen, muss die Abstraktionsschicht von den hardwarespezifischen Formaten und Zugriffsanfragen abstrahieren und ein vereinheitlichtes Format zur Datenvermittlung bereitstellen.

3 Design

In diesem Kapitel wird das Design der Hardware-Abstraktionsschicht zur Unterstützung von VR-Entwicklern vorgestellt. Zunächst wird eine allgemeine Übersicht über den Aufbau gegeben und eine Vertiefung der Anforderungen durch die Präsentation von zwei Beispielszenarien erörtert. Anschließend werden die Komponenten zur Abstraktion der Eingabe- und Ausgabegeräte sowie die Schnittstelle zum Interaktionsframework erläutert. Den Abschluss bildet eine Zusammenfassung des in dieser Arbeit vorgestellten Designs.

3.1 Aufbau der Hardware-Abstraktionsschicht

Aus den Ergebnissen der Analyse lässt sich eine erste Idee des Aufbaus der Abstraktionsschicht und was diese leisten muss herleiten, welche in Abbildung 3.1 dargestellt ist. In diesem Diagramm werden die Komponenten der Abstraktionsschicht und ihre Beziehungen untereinander abgebildet. Des Weiteren wird die Kommunikation mit den externen Eingabe- und Ausgabegeräten sowie dem in Abschnitt 2.6 vorgestellten Interaktionsframework beschrieben.
Grundsätzlich interagiert ein Nutzer mit der virtuellen Welt, indem er Eingabegeräte nutzt. Diese Eingabedaten müssen zunächst für die Anwendung interpretierbar gemacht werden. Dies geschieht meist durch, von dem Hersteller der Hardware bereitgestellte Sofware Development Kits (SDKs), welche auf der ausführenden Maschine installiert oder durch Plug-Ins direkt in die Entwicklungsumgebung eingebunden werden. Da das Interaktionsframework und die Testumgebung in Unity entwickelt werden, ist auch die Abstraktionsschicht für Unity ausgelegt und dadurch die Verfügbarkeit von Schnittstellen zwischen Hardware und Unity relevant.
Wie in Kapitel 2.4 erläutert, weisen die über Plug-Ins in die Entwicklungsumgebung integrierten Eingabegeräte sehr unterschiedliche Merkmale und Eingabedatenstrukturen auf. Diese Problemstellung zu lösen, ist eines der Ziele der Abstraktionsschicht. Um Entwicklern von VR-Anwendungen die Arbeit mit solch diverser Hardware zu erleichtern,

soll die Abstraktionsschicht eine Abstraktionsebene zwischen Hardware und Anwendung schaffen. Soweit möglich soll der Entwickler unabhängig von den später verwendeten Eingabegeräten der Endnutzer bleiben.
Hierfür bietet es sich an, die Eingabegeräte zu kategorisieren und die Eingabedaten innerhalb der Kategorien zu vereinheitlichen. Dies geschieht in der Input Komponente, welche Komponenten für die in Kapitel 2.4 vorgestellten Kategorien enthält. Die so abstrahierten Eingabedaten müssen anschließend dem Interaktionsframework zur Verfügung gestellt werden. Durch die gemeinsame Entwicklung von Interaktionsframework und Abstraktionsschicht zeichnen sich beide Systeme bei der Unterstützung von VR-Entwicklern in ihrer Zusammenarbeit aus. Deswegen spielt die Schnittstelle zwischen Interaktionsframework und Abstraktionsschicht eine entscheidende Rolle. Der konkrete Aufbau der Schnittstelle ist Thema von Kapitel 3.4.
Intern bietet das Interaktionsframework Unterstützung für die abstrahierten Eingabekategorien und verarbeitet die Eingabe. Dabei werden Vorbedingungen geprüft, Zustände von Objekten der virtuellen Welt geändert und Events ausgelöst. Eine detaillierte Beschreibung des Interaktionsframework und wie es benutzt wird findet sich in der zugehörigen Arbeit [2]. Anschließend wird vom Interaktionsframework Feedback generiert, welches dem Nutzer als Reaktion auf die Eingabe präsentiert werden soll.
Für die Feedback Komponente der Abstraktionsschicht gelten dieselben Anforderungen wie für die Eingabe Komponente. Die Nutzer der Abstraktionsschicht sollen beim Design ihrer Anwendung keine Vorkenntnisse über die Ausgabegeräte der Endnutzer benötigen. Analog zur Eingabe Komponente kommuniziert das Interaktionsframework dafür über eine abstrakte Schnittstelle mit der Abstraktionsschicht und wählt eine Feedback Kategorie, wobei die Kategorisierung auf Kapitel 2.5 beruht. Anschließend muss der Impuls innerhalb der Kategorie in eine valide Datenstruktur für die vorhandene Hardware übersetzt werden. Die entsprechende Datenstruktur richtet sich nach der Unity Integration des Hardware SDKs. Demnach ist auch hier eine Unity Integration des Ausgabegeräts erforderlich. Die Feedback Komponente wird in Kapitel 3.3.5 ausführlicher erläutert.
Bevor das Design der einzelnen Komponenten behandelt wird, sollten die für die Designentscheidungen relevanten Anforderungen an diese weiter spezifiziert werden. Dafür bietet es sich an eine beispielhafte VR-Anwendung zu analysieren. Im Folgenden werden deswegen zwei Szenarien beschrieben und anschließend erläutert, welche Anforderungen sich für die Abstraktionsschicht aus diesen ableiten lassen.

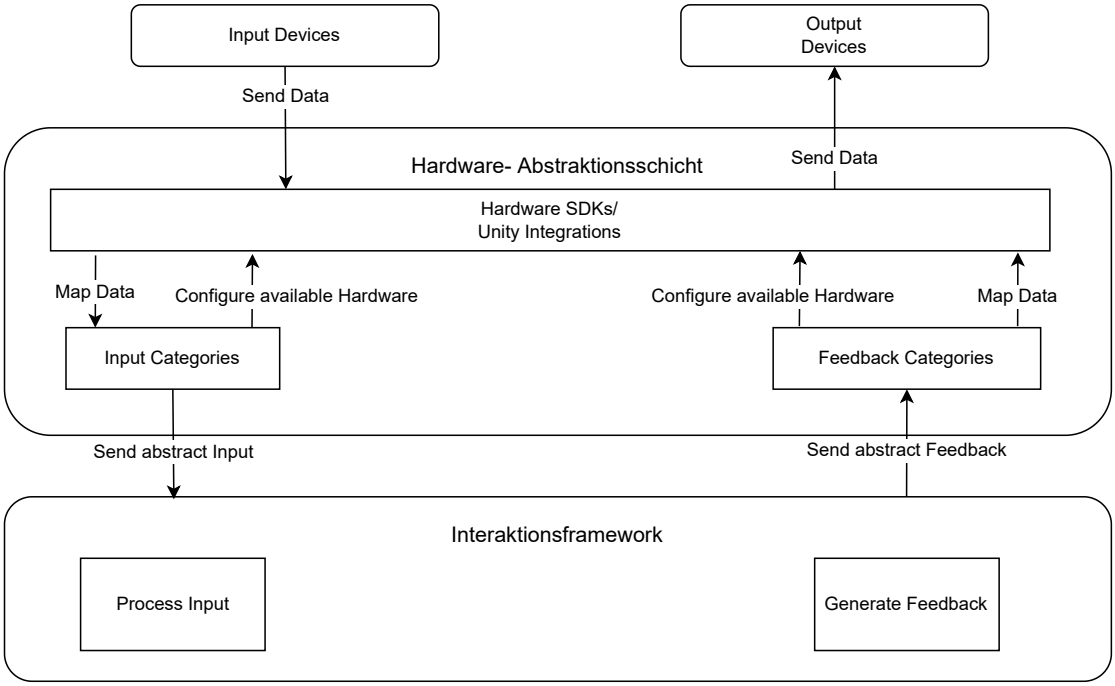


Abbildung 3.1: Komponenten Übersicht der Hardware-Abstraktionsschicht

3.2 Szenarien

Die folgend vorgestellten Szenarien befinden sich beide im Kontext einer VR-Schulung für Wartungstechniker einer Offshore-Windkraftanlage, die parallel zu dieser Arbeit entwickelt wurde. Die Szenarien sind Teil des regulären Ablaufs der Schulung und wurden als Beispiele gewählt, weil sie ein breites Spektrum an Interaktionen beinhalten und von unterschiedlichen Feedback Optionen profitieren können. Dabei dienen die Szenarien als repräsentative Beispiele für Interaktionsabläufe in technischen Anlagen.

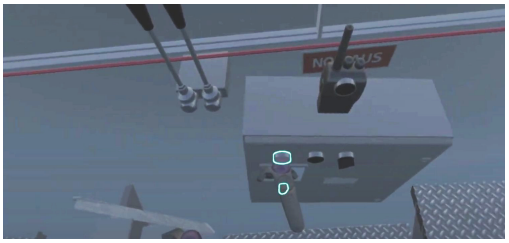
3.2.1 Szenario: Dachluke öffnen

Inhalt des ersten Szenarios ist der Interaktionsablauf, welcher zum Öffnen der Dachluke der Windkraftanlage ausgeführt wird. Hierbei werden grob betrachtet fünf Interaktionsschritte benötigt:

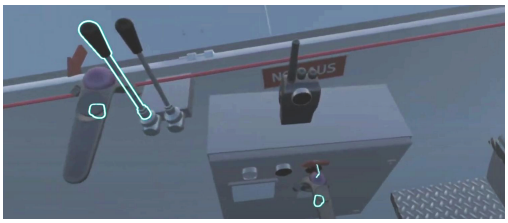
1. Hydraulikschalter umlegen
2. Startknopf drücken
3. Drehschalter umlegen und halten
4. Hebel für die rechte Dachklappe ziehen
5. Hebel für die linke Dachklappe ziehen

Konkret aktiviert ein Wartungstechniker zuerst den Schalter für die Hydraulik. Dieser befindet sich auf Bodenhöhe unter einer Treppe in der Gondel und wird von seiner Startposition aus um 180° rotiert. Als nächstes navigiert der Techniker zum Steuerungsterminal der Dachluken. Dort schaltet er den Hydraulikkompressor durch das Drücken des Startknopfes ein. Folgend werden die beiden Dachklappen nacheinander geöffnet, indem der Drehschalter um 90° nach rechts gedreht wird und synchron dazu der jeweilige Hebel für die rechte oder linke Dachklappe gezogen wird. Dabei öffnen sich die Dachklappen, bis sie ihren maximalen Öffnungsgrad erreicht haben, solange Schalter und Hebel gehalten werden. Eine graphische Übersicht des Ablaufs findet sich in Abbildung 3.2.

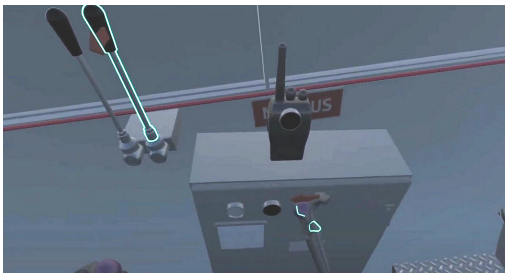
(a) Umstellen der Hydraulik



(b) Start des Hydraulikkompressors



(c) Öffnen der rechten Dachluke



(d) Öffnen der linken Dachluke

Abbildung 3.2: Öffnen der Dachluken

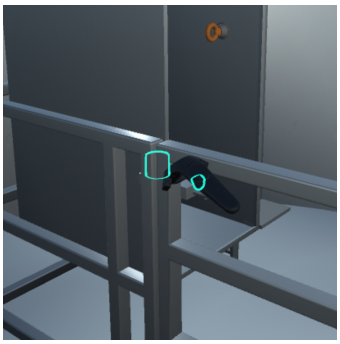
3.2.2 Szenario: Fahrstuhl

Das zweite Szenario beschäftigt sich mit dem Zugang zum Fahrstuhl, welcher verschiedene Ebenen in der Windkraftanlage miteinander verbindet. Der Ablauf des Szenarios teilt sich in die Schritte:

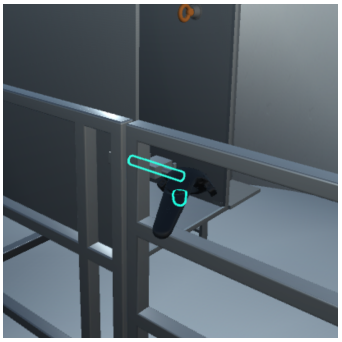
1. Rolltor hochschieben
2. Schlüssel entnehmen
3. Schloss aufschließen
4. Bolzen lösen
5. Tor öffnen

(a) Rolltor hochschieben

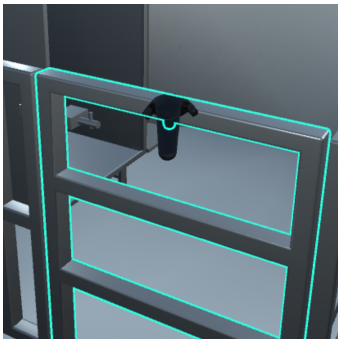
(b) Schlüssel entnehmen



(c) Schloss aufschließen



(d) Bolzen lösen



(e) Tor öffnen

Abbildung 3.3: Zugang zum Fahrstuhl

Um Zugang zum Fahrstuhl zu erhalten, muss der Techniker zunächst über das Tor greifen und das Rolltor hochschieben. Im Fahrstuhl befindet sich der Schlüssel, welcher zum

Entsichern des Bolzens benötigt wird. Ist der Bolzen gelöst, kann das Tor geöffnet werden und der Techniker kann den Fahrstuhl betreten. Aus Sicherheitsgründen muss der Fahrstuhl anschließend wieder gesichert werden, indem die Schritte in umgekehrter Folge durchgeführt werden. Erst dann lässt sich die Steuerung des Fahrstuhls bedienen. Der Ablauf in die Einzelschritte unterteilt ist in Abbildung 3.3 dargestellt.

3.2.3 Erweiterte Anforderungen an das Design

Im Kontext einer Schulung für technische Anlagen rücken andere Anforderungen an den Aufbau der virtuellen Welt und das zugrunde liegende System in den Vordergrund. Das Streben nach einer perfekten Abbildung der Realität und einer entsprechenden Präsentation durch Ausgabegeräte wie das Ultimate Display [26] ist weniger priorisiert, als die Steigerung des Lerntransfers von Wissen aus der virtuellen Umgebung in die reale Welt. Dabei spielt die Art und Weise, wie die Schulungsteilnehmer mit der virtuellen Welt interagieren und Feedback von dieser erhalten, eine wichtige Rolle. Für die Problemstellung, welche Hardware grundsätzlich gut geeignet ist, um den Bediener technischer Anlagen die Gewöhnung an VR zu erleichtern und gelerntes zu transferieren, gibt es jedoch keine etablierte Lösung. Des Weiteren erschweren die Diversität aktueller Eingabe- und Ausgabegeräte ohne standardisierte Systemschnittstellen und die Vielfalt an Interaktionen in einer technischen Schulung dem Entwickler die Entscheidung.
Das Ziel dieser Arbeit ist es, den Entwickler aus dieser Situation zu befreien, indem ihm ein Repertoire an Hardware Optionen zur Verfügung gestellt wird, damit er selbst ohne großen Mehraufwand prüfen kann, welche Hardware für seine Zwecke gut geeignet ist. Dabei soll er sein Interaktionsdesign möglichst unabhängig von der konkreten Hardware gestalten können. Demnach muss ein abstrakter Zugriff auf das angebotene Repertoire möglich sein. Dies umfasst sowohl Eingabegeräte zur Interaktion mit der virtuellen Welt als auch Ausgabegeräte zur Vermittlung von Feedback. Des Weiteren sollten für bessere Vergleichsoptionen nicht nur Geräte einer Kategorie eingebunden werden und eine möglichst dynamische Möglichkeit zwischen Hardwaretypen zu wechseln auch für einzelne Interaktionen angeboten werden. Die Relevanz dieser Option wird nochmals deutlich, wenn man die Interaktionen der Beispielszenarien genauer betrachtet. Diese sind unterschiedlich komplex und benötigen abhängig davon spezifischere Eingaben sowie Möglichkeiten Feedback zu geben. Außerdem sind einige Interaktionen im Kontext der Schulung und des zu vermittelnden Lerninhalts weniger bedeutend. Der Entwickler könnte so eine Interaktion z. B. das Öffnen des Tors auch mit Hardware umsetzen, die

ein weniger natürlichen Ablauf liefert, aber dafür einfacher oder schneller erstellt und umgesetzt wird. Während er parallel dazu verschiedene Hardware für die Bedienung der Hydraulik einbindet, um zu untersuchen, inwiefern diese den Wissenstransfer unterstützt.

3.3 Hardware-Abstraktionsschicht

3.3.1 Übersicht

Die Hardware-Abstraktionsschicht lässt sich allgemein in die Input Komponente für Eingaben und die Feedback Komponente für Ausgaben unterteilen. Aufgabe der Input Komponente ist es, die Eingabedaten unterschiedlicher Eingabegeräte zu verarbeiten und in einem vereinheitlichten Format in Unity zur Verfügung zu stellen. Die Struktur der Eingabedaten kann dabei sehr stark variieren. Während einige Eingabegeräte nur binäre Signale senden, bestimmen andere Position und Rotation verschiedener Punkte und schaffen ein eigenes 3D-Modell. Eine generelle Abstraktion aller Eingabegeräte gestaltet sich demnach eher schwierig. Differenziert man die Eingabegeräte anhand der Kategorien, welche in Kapitel 2.4.2 vorgestellt werden, lassen sich jedoch strukturelle Gemeinsamkeiten innerhalb der Typen feststellen. Die Kategorien „Zusätzliche Artefakte“, „Hand-Tracking“ und „Hand-unabhängige“ lassen sich durch übergeordnete Klassen abstrahieren.
Durch die Einkapselung in abstrakte Klassen benötigt ein Entwickler kein Wissen über konkrete Hardware Eigenschaften bei der Gestaltung von Interaktionen mit der virtuellen Welt. Er kann das Design auf der Klasse für die gewünschten Eingabegeräte aufbauen. Dabei ist die Entscheidung, welche Kategorie er nutzen will, keine zusätzliche Einschränkung oder Mehraufwand im Designprozess, da auch bei dem Interaktionsframework abhängig von der Eingabestruktur andere Komponenten und Einstellungen zur Umsetzung der Interaktion gewählt werden müssen. Eine vollständige Unabhängigkeit von der konkreten Hardware wird zwar so nicht erreicht, das benötigte Wissen über das System wird jedoch auf die abstrakten Klassen reduziert. Der Aufbau der abstrakten Klassen und die Übersetzung von Eingabedaten in die vereinheitlichten Formate, wird in den folgenden Abschnitten für jede Kategorie beschrieben.
Die Feedback Komponente verfolgt denselben Ansatz wie die Input Komponente, durch abstrakte Klassen von den konkreten Ausgabegeräten zu abstrahieren. In Kapitel 2.5 werden Ausgabegeräte für VR in die wesentlichen Wahrnehmungskategorien der visuellen, auditiven und haptischen Wahrnehmung unterteilt. Im Projektrahmen der Abstrak-

tionsschicht wird das visuelle und auditive Feedback durch das Interaktionsframework übernommen. Dieses unterstützt das, für die meisten VR-Anwendungen genutzte Setup mit HMD und Kopfhörer als Ausgabegeräte für die audiovisuelle Darstellung der virtuellen Welt. Dabei enthält das Interaktionsframework bereits Mechanismen zur Abstraktion entsprechender Geräte. Nicht dem Standard entsprechende Ausgabegeräte wie CAVE- und Wellenfeldsynthesesysteme sind nicht im Interaktionsframework enthalten, lassen sich aber aufgrund ihrer Komplexität grundsätzlich nur schwer abstrahieren. In dem Abschnitt zur Feedback Komponente wird deswegen hauptsächlich auf Ausgabegeräte zur Manipulation der haptischen Wahrnehmung Bezug genommen.

3.3.2 Zusätzliche Artefakte

Die Kategorie Zusätzliche Artefakte umfasst alle in der physischen Hand getragenen Eingabegeräte, die über Tracking verfolgt und durch ein Modell in der virtuellen Welt dargestellt werden. Dabei unterscheiden sich die Eingabegeräte durch die Parameter, die bestimmt werden. So ist die Anzahl der Freiheitsgrade (eng. Degrees of Freedom, DoF) ein Kriterium. Wenn nur die drei Rotationsachsen ermittelt werden, spricht man von 3DoF. Erweitert man 3DoF-Tracking um die drei Translationsachsen landet man bei 6DoF. Aktuelle Tracking-Systeme unterstützen fast immer 6DoF, da Interaktionsframeworks bei Interaktionen mit Objekten der virtuellen Welt häufig auf der Position und Rotation der Eingabegeräte aufbauen. Dabei benutzen die unterschiedlichen Tracking-Systeme nicht zwangsläufig das gleiche Koordinatensystem. Dies ist für diese Arbeit jedoch nicht relevant, da die Abstraktionsschicht auf den Unity Integrationen aufbaut, welche die Daten in das Unity interne Koordinatensystem transformieren.
Unabhängig von der Anzahl Freiheitsgrade bieten einige Artefakte zusätzliche Möglichkeiten der Eingabe. Ein Beispiel hierfür sind Controller, welche viele VR-Hardware Hersteller als Eingabegeräte dieser Kategorie verkaufen. Controller zeichnen sich durch eine variierende Anzahl an Knöpfen aus, wobei durch das Design häufig ein Knopf hervorgehoben wird. Dieser Knopf wird als Trigger bezeichnet und ist für die Standardeingabe ausgelegt. Neben den Knöpfen verfügen einige Controller auch über Sensoren, welche über die Hardware Schnittstelle ausgelesen werden können. So können Entwickler auch auf Daten wie Beschleunigung, Berührung oder Druck zugreifen.

Die abstrakte Klasse berücksichtigt all diese möglichen Daten, damit der Nutzer der Abstraktionsschicht nicht im Vorfeld eingeschränkt wird. Sie bietet Methoden, um die Position und Rotation des Artefakts zu erfragen. Auch die Zustände von Knöpfen und Sensoren unterschiedlicher Art werden zur Verfügung gestellt. Des Weiteren bietet sie die Option, verschiedene Events zu abonnieren, welche unter bestimmten Bedingungen der Knöpfe und Sensoren auftreten. Dem Nutzer wird so die volle Bandbreite an Eingabedaten von zusätzlichen Artefakten zur Gestaltung von Interaktion in der virtuellen Welt zur Verfügung gestellt. Es kann jedoch nicht sichergestellt werden, dass die konkrete Hardware des Endnutzers alle Eingabedaten liefern kann. Deswegen wirft die Abstraktionsschicht bei Anfrage eines nicht unterstützten Datentyps eine Exception, welche der Interaktionsdesigner auffangen und behandeln muss.
Eine Übersicht der Eingabedaten, welche durch die abstrakten Klasse verfügbar gemacht werden, findet sich in Abbildung A.1. Dabei basiert das so geschaffene abstrakte Modell der Zusätzlichen Artefakt Kategorie auf einer Analyse konkreter Eingabegeräte, die zum Zeitpunkt dieser Arbeit bevorzugt genutzt werden, um mit virtuellen Welten zu interagieren. Das Modell beinhaltet einerseits grundlegende Eingabedaten wie Position, Rotation, Geschwindigkeit, Griffstärke sowie Beschleunigung des Artefakts, welche durch Tracking und Sensoren bestimmt werden. Andererseits werden auch die Knöpfe von Controllern abgebildet. Dabei werden binäre Zustände, ob die Knöpfe berührt oder gedrückt werden, sowie eindimensionale Achsenwerte wie der Druck auf den Trigger Knopf und zweidimensionale Achsenwerte von Touchpads oder Joysticks berücksichtigt.
Zugriff auf die Eingabedaten erhält der Nutzer der Abstraktionsschicht durch Get-Methoden. Diese sind nach Rückgabetyp aufgebaut, wobei die spezifische Methode für den gesuchten Wert durch die Mitgabe des zugehörigen Bezeichners ausgewählt wird. Hierfür ist für jedes von der Abstraktionsschicht unterstützte Eingabegerät, eine entsprechende Zuordnung notwendig. Konkret müssen die Bezeichner der abstrakten Klasse den Eingabedaten der Hardware und die Get-Methoden der abstrakten Klasse den zugehörigen Methoden der Unity Integration der Hardware zugeordnet werden. Zur Laufzeit der Anwendung kann dann durch eine separate Klasse, welche auf die Präsenz eines Artefakts der unterstützten Integrationen prüft, dynamisch die passende Zuordnung geladen werden.
Zusätzlich zu den Get-Methoden bietet die abstrakte Klasse auch Events und Hilfsmethoden an, welche in Abbildung A.2 dargestellt sind. Für die Events wird das Unity Eventsystem genutzt. In jedem Frame wird überprüft, ob sich der Zustand des Artefakts im Vergleich zum letztem Frame verändert hat. Abhängig vom Typ des Eingabewerts und der Zustandsänderung wird folgend ein entsprechendes Event ausgelöst. Dieses beinhaltet für alle Werte ein Event bei jeglicher Änderung, sowie Events bei Berührung, Druck

und Loslassen eines Knopfes. Der Nutzer spart sich durch das Abonnieren der Events die unzähligen Aufrufe von Get-Methoden und Vergleichen der Werte an verschiedenen Stellen des Interaktionsdesigns. Ein ähnliches Ziel erfüllen die Hilfsmethoden. Durch die Erweiterung des Modells um generelle Eigenschaften des Artefakts, wird dem Nutzer eine Möglichkeit gegeben in seinem Design frühzeitig Optionen für mögliche Unterschiede in der konkreten Hardware zu schaffen. So wird die Notwendigkeit reduziert Ausnahmen im Nachhinein auffangen zu müssen. Hierfür müssen bei der Einbindung von Hardware in die Abstraktionsschicht die entsprechenden Flags der Eigenschaften gesetzt werden.

3.3.3 Hand-Tracking

Unter der Hand-Tracking Kategorie werden alle Eingabegeräte zusammengefasst, welche die physischen Hände der Nutzer in der realen Welt erfassen und dem VR-System als Eingabedaten zur Verfügung stellen. Dabei werden hauptsächlich optische Sensoren genutzt, um die Hände zu bestimmen. Alternativ arbeiten einige Eingabegeräte mit einer Kombination aus optischer Verfolgung und komplexer Drucksensorik, um ein Modell der Hand als Eingabe zu generieren. Auch in dieser Kategorie ist eine Voraussetzung für die Einbindung der Eingabegeräte in die Abstraktionsschicht eine vorhandene Schnittstelle zwischen der Hardware und Unity, welche die Eingabedaten in Unity verfügbar macht.
Während die Eingabegeräte zur Bestimmung der Hände durchaus unterschiedliche Sensoren benutzen, erstellen sie dennoch alle ein 3D-Modell der Hand als Eingabedatenstruktur. Diese Modelle bestehen aus einer Menge an 3D-Positionsdaten von bestimmten Punkten der Hand, die verfolgt werden. Dabei unterscheiden sich die Modelle aktuell verbreiteter Eingabegeräte meist nur geringfügig in der Anzahl der bestimmten Punkte und in der Art und Weise, wie der Nutzer Zugriff aus diese erhält. Ähnlich wie bei den Zusätzlichen Artefakten bietet es sich an ein abstraktes Modell mit vereinheitlichten Zugriffsmethoden zu schaffen, um den Entwickler beim Interaktionsdesign von der konkreten Hardware unabhängig zu machen.
Das abstrakte Hand-Modell besteht aus 25 Punkten und basiert auf den Modellen unterschiedlicher Eingabegeräte. Dabei sind 20 der Punkte aktiv verfolgte Positionsdaten einzelner Schlüsselpunkte der Hand unterteilt in drei Punkte für den Daumen, vier Punkte für die anderen Finger und ein Punkt für die Position des Handgelenks. Die übrigen fünf Punkte repräsentieren die Fingerspitzen, welche nicht aktiv bestimmt, sondern als Hilfspunkte berechnet werden. Während die Hilfspunkte nicht zur Abbildung der Hand

in der virtuellen Welt genutzt werden, können diese jedoch bei der Gestaltung von Interaktionen hilfreich sein. Eine vollständige Liste der Punkte mit ihren Bezeichnern und den dazugehörigen Knochen der Hand findet sich in Abbildung A.3.
Auf dem abstrakten Modell basierend bietet die abstrakte Klasse für Hand-Tracking Zugriff auf die Daten des Modells. Durch den Aufruf einer Get-Methode mit dem Bezeichner des gewünschten Punktes erhält der Nutzer die zugehörigen Positionsdaten. Analog zu der Klasse für Zusätzliche Artefakte ist auch hier Voraussetzung, dass für jede unterstützte Hardware eine Zuordnung zwischen den Punkten des konkreten und des abstrakten Modells erstellt wurde. Des Weiteren erfolgt auch hier die Behandlung von Fehlern, wenn ein Punkt des abstrakten Modells nicht in der Zuordnung berücksichtigt werden konnte, auf Seiten des Interaktionsdesigners.
Anders als bei den Zusätzlichen Artefakten gestaltet sich jedoch die Eingabe von Trigger Events zur Selektion und Manipulation. Während einfache Interaktionen noch durch Kollisionsbestimmung zwischen Interaktionsobjekt und Punkten des abstrakten Hand-Modells realisiert werden können, fehlen für komplexere Interaktionen die Events zur Auswahl und der gewünschten Manipulation des Objekts. So könnte der Knopf aus dem Szenario 3.2.1 durch ein Kollisionsevent aktiviert werden, welches ausgelöst wird, wenn einer der Hilfspunkte für die Fingerspitzen auf den Kollisionsradius des Knopfes trifft. Die Selektion, welcher Hebel zur Öffnung der Dachluken wann gegriffen wird, benötigt jedoch zusätzliche Eingabedaten. Auch Interaktionen mit dem System wie das Öffnen des Menüs gestaltet sich schwierig, ohne einen expliziten Knopf dafür zu besitzen. Für solche Eingaben im Zusammenhang mit Hand-Tracking werden häufig Frameworks zur Gestenerkennung genutzt. Diese bieten meist eine vorbestimme Menge an Gesten und entsprechende Events bei der Erkennung dieser an. Einige ermöglichen sogar die Definition von eigenen Gesten, um Entwicklern ein breiteres Spektrum an Eingabemöglichkeiten zur Verfügung zu stellen. Die Unity Integrationen der von der Abstraktionsschicht unterstützten Hand-Tracking Eingabegeräte besitzen alle ein Modul zur Gestenerkennung. Dabei arbeiten diese Module meist sehr hardwareseitig und funktionieren nur mit den konkreten Hand-Modellen. Die abstrakte Klasse agiert deswegen wieder als Schnittstelle zwischen Interaktionsdesigner und der Hardware. Der Nutzer abonniert über die abstrakte Klasse dem Event einer gewünschten Geste. Die Abstraktionsschicht löst diese Anfrage zur Laufzeit über eine Zuordnung von abstrakter Geste zu konkreter Implementation der Hardware auf. In unseren Szenarien könnte so der Anwender den Hebel mit einer Hand-geschlossen Geste greifen oder das Menü mit einer Pinch-Geste öffnen.

3.3.4 Hand-unabhängige Eingabe

Die Abstraktion von Hand-unabhängigen Eingabegeräten gestaltet sich schwieriger, da die Diversität der einzelnen Geräte innerhalb der Kategorie am größten ist. Es existieren keine ähnlichen Modelle oder Datenstrukturen wie bei den anderen Kategorien auf denen man aufbauen kann und die Geräte erfordern grundlegend unterschiedliche Ansätze im Interaktionsdesign. Eine allgemeine Klasse zur Abstraktion der gesamten Kategorie ist folglich nicht realisierbar. Bestimmte Unterkategorien über abstrakte Klassen verfügbar zu machen, ist jedoch möglich. Als Beispiel dafür wird in dieser Arbeit eine abstrakte Schnittstelle zur Eingabe über Sprachbefehle präsentiert.
Anders als bei den Zusätzlichen Artefakten und dem Hand-Tracking wird bei der Komponente zur Sprachsteuerung keine Abstraktion der konkreten Hardware benötigt, da Unity intern eine Mikrofon Schnittstelle mit abstrakten Zugriffsmethoden anbietet. Dabei erhält Unity Zugriff auf alle vorhandenen Aufnahmegeräte über das Betriebssystem der Anwendung. Jedoch reicht die Aufnahme von Sprache noch nicht, um gezielt mit der virtuellen Welt zu interagieren. Hierfür wird eine Spracherkennungskomponente benötigt, welche die Eingabe filtert und die Möglichkeit zur Verfügung stellt, bestimmten Wörtern Events zuzuordnen. Zur Bereitstellung dieser Funktion existieren einige Services mit entsprechender Unity Integration. Diese Services sind grundsätzlich unabhängig von der konkreten Hardware. Deswegen würde es in der Theorie genügen nur einen Service in die Abstraktionsschicht zu integrieren und dem Interaktionsdesigner eine direkte Schnittstelle zu präsentieren. Da sich die Services jedoch in Qualität, Kosten und Datenschutzrichtlinien unterscheiden ist die Wahl des Services trotzdem sowohl für Entwickler als auch die Endnutzer relevant. Eine entsprechende Auswahl oder der Wechsel des Services soll durch die Abstraktionsschicht ermöglicht werden ohne Einfluss auf das Interaktionsdesign des Entwicklers zu nehmen.
Allgemein unterscheiden sich die Services in ihrer Funktionsweise kaum. Der Designer gibt ein Wort oder eine Folge von Wörtern an und die Spracherkennungssoftware wirft ein Event wenn das Wort oder die Wörter gesprochen werden. Diesen Events kann anschließend mit Angabe einer auszuführenden Methode abonniert werden. Die abstrakte Spracherkennungsklasse baut auf diesem Ablauf auf. Über eine Methode der Klasse registriert der Nutzer den Sprachbefehl und die zugehörige Methode, welche das Feedback der virtuellen Welt auf die Interaktion darstellt. Für jeden unterstützten Service wird dafür eine Zuordnung der Methoden zur Übersetzung der Anfrage benötigt. Der allgemeine Ablauf einer Spracheingabe über die Abstraktionsschicht ist in Abbildung 3.4 dargestellt.

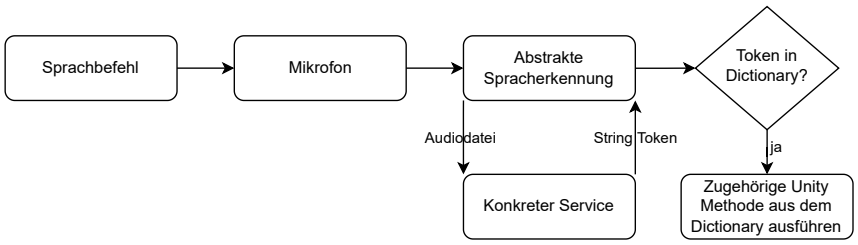


Abbildung 3.4: Ablauf der Spracherkennung

3.3.5 Feedback Komponente

Während sich die vorherigen Abschnitte hauptsächlich mit der Abstraktion von Eingabegeräten zur Interaktion mit virtuellen Welten beschäftigt haben, soll in diesem Abschnitt die andere Richtung, also die Übermittlung von Feedback über Ausgabegeräte thematisiert werden. Nachdem ein Nutzer mit einer virtuellen Welt interagiert, verarbeitet das VR-System die Eingabe und verändert die Zustände von Objekten. Das Ergebnis dieses Prozesses wird anschließend dem Nutzer vermittelt. Auch in VR bildet die Manipulation von optischer und auditiver Wahrnehmung die Grundlage für das Feedback an den Anwender. Dabei wird der Ablauf von Erstellung bis zur Vermittlung von auditivem und visuellem Feedback bereits durch das Interaktionsframework und eine Unity Schnittstelle für Ausgabegeräte dieser Kategorien vollständig abgedeckt. Eine zusätzliche Abstraktionsschicht für diese Feedback Kategorien wird deswegen nicht benötigt. Interessanter gestaltet sich die Umsetzung von Feedback anderer Arten, da für die Einbindung dieser in VR noch keine etablierten Vorgehensweisen oder Schnittstellen existieren.
Die abseits von visuellem oder auditiven Feedback am meisten verbreitete Kategorie ist das haptische Feedback. Über Ausgabegeräte werden Signale an die Rezeptoren des Nutzers gesendet, um die Wahrnehmung von Berührungen, Temperatur und Schmerz zu beeinflussen. Dabei ist für die Popularität dieser Kategorie die Tatsache entscheidend, dass moderne Eingabegeräte der Zusätzliche Artefakt Kategorie häufig auch Funktionen für haptisches Feedback bereitstellen. Die abstrakte Klasse für Zusätzliche Artefakte besitzt zur Unterstützung des Entwicklers deswegen eine Methode, um die Verfügbarkeit von haptischen Feedback Optionen zu überprüfen. Neben den Eingabegeräten mit eingebauter Haptik gibt es jedoch auch spezielle Hardware für haptisches Feedback. Die in 2.5.3 vorgestellten Kategorien Wearable Devices, Handheld Devices, Haptic Seats und

Non Physical Contact Devices bieten eine Vielzahl an haptischen Ausgabegeräten mit unterschiedlicher Anwendung und diversen Ausgabeformaten.
Damit der Entwickler im Designprozess unabhängig von der konkreten Hardware bleibt, wird auch für die Ausgabegeräte eine abstrakte Klasse benötigt. Diese bietet allgemeine Methoden für haptisches Feedback an, welche über Zuordnungen zur Laufzeit in entsprechende Signale unterstützter Hardware übersetzt werden. Dabei enthält die abstrakte Klasse Methoden sowohl für die unterschiedlichen Sinne der haptischen Wahrnehmung als auch für die zielgerichtete Manipulation von Körperteilen. Auch die Stärke des Signals kann mit einem Parameter bei Methoden der abstrakten Klasse angegeben werden. Konkrete Hardware lässt sich durch Zuordnungen der Methoden unter Berücksichtigung der Parameter einbinden. Hierfür wird es tendenziell notwendig, die Hardware selbst über Plug-Ins in Unity verfügbar zu machen, da die Geräte häufig nicht direkt für VR entwickelt wurden und im Gegensatz zu den Eingabegeräten selten bereits eine Schnittstelle zu Unity besitzen. Sollte zur Laufzeit der Anwendung beim Aufruf einer Methode der abstrakten Klasse kein unterstütztes Gerät vorhanden sein, wird keine Fehlermeldung geworfen, die Anwendung ignoriert einfach die Anfrage und läuft ohne Reaktion weiter. Während die Abstraktionsschicht aktuell nur haptisches Feedback unterstützt, ist es auf die gleiche Art und Weise natürlich auch möglich weitere Kategorien einzubinden.

3.3.6 Fazit

In diesem Kapitel wird der Aufbau einer Abstraktionsschicht beschrieben, mit der Entwickler von VR-Anwendungen Interaktion unabhängig von der Hardware des Nutzers gestalten können. Dazu werden die Eingabegeräte in die Kategorien Zusätzliche Artefakte, Hand-Tracking und Sprachsteuerung unterteilt und für jede Kategorie eine abstrakte Klasse definiert. Über die abstrakten Klassen erhalten die Interaktionsdesigner Zugriff auf gewünschte Informationen der Hardware ohne die spezifischen Eigenschaften und Syntax der Hardware berücksichtigen zu müssen. Dabei werden abstrakte Modelle definiert und Zuordnungen für unterstützte Hardware benötigt. Des Weiteren wird die Nutzung der Abstraktionsschicht für Ausgabegeräte durch die Integration einer Schnittstelle für haptisches Feedback aufgezeigt.
Neben der gewonnen Unabhängigkeit von Hardware im Entwicklungsprozess bietet die Abstraktionsschicht weitere Vorteile. So können Entwickler durch die dynamische Erkennung der Hardware und das Laden der entsprechenden Zuordnung auch leichter verschiedene Geräte zur Nutzung innerhalb einer Anwendung anbieten oder Tests zur Wahl

geeigneter Hardware durchführen. Dabei können weitere Geräte der jeweiligen Kategorien jederzeit eingebunden werden, indem eine Zuordnung vorgenommen wird. Auch existieren bereits einige Hilfsmethoden innerhalb der abstrakten Klassen, welche den Entwickler mit zusätzlichen Informationen versorgen oder bei der Fehlerbehandlung unterstützen.

3.4 Schnittstelle zum Interaktionsframework

Durch die Hardware-Abstraktionsschicht ist es Entwicklern von VR-Anwendungen möglich, unterschiedliche Arten von Hardware für Input und Feedback einzubinden, ohne spezifische Hardware Details zu kennen. Dabei steht die Vermittlung der Eingabe- und Ausgabedaten an die entsprechende Hardware im Vordergrund. Methoden und Schemata wie das VR-System die Daten in dem Verhaltensmodell verarbeitet und generiert, sind jedoch nicht Teil der Abstraktionsschicht. Um die Implementation eines komplexeren Szenarios in Bezug auf diese Aspekte zu vereinfachen, kann ein Interaktionsframework genutzt werden. Ein solches Interaktionsframework wurde parallel zu dieser Arbeit von Jonathan Becker entwickelt. Dabei wurden das Framework und die Abstraktionsschicht im Design aufeinander abgestimmt und können problemlos zusammen genutzt werden, um eine interaktive VR-Anwendung zu schaffen.
Nutzt ein Designer die Kombination aus Interaktionsframework und Abstraktionsschicht, muss er für das Design einer konkreten Interaktion zunächst das Skript aus dem Interaktionsframework wählen, welches das gewünschte Verhalten des Interaktionsobjekts beinhaltet. Anschließend modifiziert er die Parameter des Skripts, um das Verhalten auf die spezifische Situation anzupassen. Ein Teil dieser Spezifikation ist die UI-Schnittstelle für die Hardware-Abstraktionsschicht. Über diese Schnittstelle können die Methoden der abstrakten Klassen für Ein- und Ausgabe eingebunden werden. So kann der Entwickler als Bedingung für den Beginn der Interaktion z. B. das Drücken des primären Knopfes der abstrakten Controller Klasse wählen und bei erfolgreichem Abschluss der Interaktion ein haptisches Feedback Signal über die abstrakte Feedback Klasse senden. Dabei gilt zu beachten, dass weder das Interaktionsframework noch die Abstraktionsschicht Tools bereitstellen, um die Absenz kompatibler Hardware zur Laufzeit zu erkennen. Tritt dieser Fall ein, bleiben alle Anfragen an die Hardware-Abstraktionsschicht wirkungslos. Grundsätzlich kann zur Vermeidung dieser Situation eine Mehrfachzuordnung vorgenommen werden. Hierbei werden verschiedene abstrakte Klassen zur Eingabe und Ausgabe beim Design der Interaktion berücksichtigt. Dies bietet auch dem Nutzer der Anwendung Vorteile, da er eine größere Auswahl an Hardware nutzen kann.

Neben dem Feedback, welches als Reaktion einer Interaktion ausgelöst wird, generiert das Interaktionsframework zusätzlich von der Eingabe unabhängiges Feedback. Dieses basiert nicht auf der Selektion oder Manipulation eines bestimmten Objekts, sondern wird von allgemeinen oder Szenario übergreifenden Skripten ausgelöst. Hierfür nutzt das Interaktionsframework auch die Feedback Komponente der Abstraktionsschicht. Möchte der Interaktionsdesigner die Eingabegeräte für den Nutzer der Anwendung visualisieren, kann er über die Hardware-Abstraktionsschicht auf die SDKs der Hardware zugreifen. Diese enthalten in der Regel entsprechende Modelle für Unity und Skripte zur Aktualisierung dieser. Eine Übersicht über die Schnittstelle zwischen Abstraktionsschicht und Interaktionsframework ist in Abbildung 3.5 dargestellt.

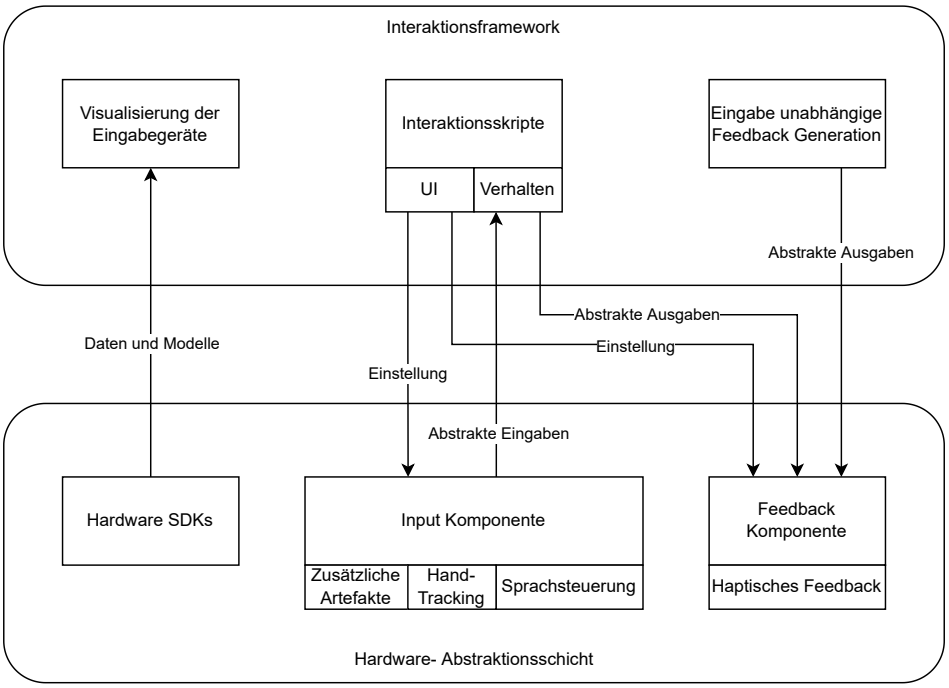


Abbildung 3.5: Schnittstelle zwischen Abstraktionsschicht und Interaktionsframework

3.5 Zusammenfassung

In diesem Kapitel wird eine Hardware-Abstraktionsschicht entworfen, um die Gestaltung von Interaktionen in VR-Applikationen zu erleichtern. Dabei werden nach einer Übersicht über den Aufbau der Abstraktionsschicht zunächst zwei komplexere Interaktionsszenarien beschrieben, um die Anforderungen an eine solche Abstraktionsschicht zu spezifizieren.

Auf Grundlage dieser Anforderungen wird der Aufbau der Abstraktionsschicht vorgestellt. Diese teilt sich in die vier Komponenten Zusätzliche Artefakte, Hand-Tracking, Sprachsteuerung und Feedback. Für jede dieser Komponenten ein abstraktes Modell und eine abstrakte Klasse entworfen. Abschließend wird das Zusammenspiel zwischen der Hardware-Abstraktionsschicht und dem Interaktionsframework erläutert, welches parallel zur Abstraktionsschicht in demselben Projektrahmen entwickelt wurde.
Durch die Komponenten der Abstraktionsschicht wird dem Entwickler ermöglicht, unabhängig von konkreter Hardware zu arbeiten. Anfragen an Eingabe- und Ausgabegeräte werden an die abstrakte Schnittstelle der gewünschten Kategorie gerichtet und dort von der entsprechenden Klasse übersetzt. Weiterhin wird er durch das Interaktionsframework und die Schnittstelle zwischen diesem und der Abstraktionsschicht beim Design unterstützt. Dabei gilt zu klären, ob die einzelnen Anforderungen berücksichtigt werden. Deswegen wird im folgenden Kapitel evaluiert, ob die Anforderungen aus dem Analysekapitel und der Erörterung der Beispielszenarien im Design erfüllt werden.

4 Evaluation

4.1 Übersicht

In diesem Kapitel wird das Design der Hardware-Abstraktionsschicht hinsichtlich der in den Kapiteln 2.7 und 3.2.3 formulierten Anforderungen evaluiert. Für die Evaluation wurde die Abstraktionsschicht in das Projekt zur Schulung von Wartungstechnikern eingebunden, in dessen Kontext auch das Interaktionsframework von Jonathan Becker entwickelt wurde. Die Leistungen der Abstraktionsschicht wurden dem Designer der Schulung über die Schnittstelle zum Interaktionsframework zur Verfügung gestellt. Dabei wurden verschiedene Funktionalitäten in die in Kapitel 3.2 beschriebenen Szenarien der Schulung integriert. Anhand dieser Integration und einer technischen Analyse wird im Folgenden gezeigt, dass die Anforderungen durch das Design der Abstraktionsschicht erfüllt werden. Abschließend wird ein Ausblick auf mögliche Erweiterungen der Abstraktionsschicht gegeben, welche in zukünftigen Projekten umgesetzt werden könnten.

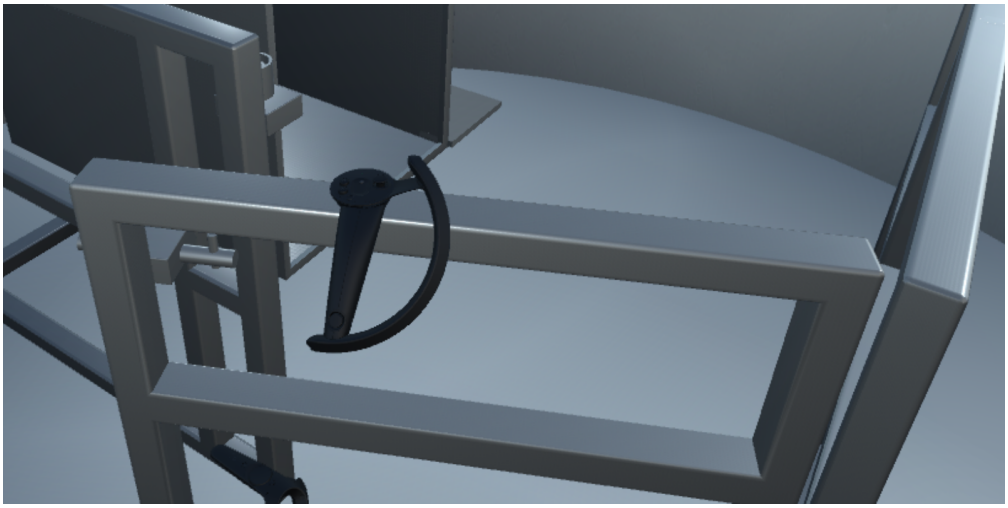
4.2 Auswertung der Anforderungen

In Kapitel 3.2 werden zwei Szenarien aus der Schulung für Wartungstechniker einer Offshore-Windkraftanlage dargestellt. Inhalt des ersten Szenarios ist der Ablauf, welcher ausgeführt wird, um die Dachklappen zu öffnen. Im zweiten Szenario hingegen steht die Bedienung des Fahrstuhls der Anlage im Mittelpunkt. Dabei wurde in der Analyse der beiden Szenarien deutlich, dass es vom Vorteil ist wenn verschiedene Typen an Geräten zur Ein- und Ausgabe bereitgestellt werden und diese generell für Interaktionstypen oder sogar einzelne Interaktionen frei eingestellt und ausgetauscht werden können. Das in dieser Arbeit vorgestellte Design einer Hardware-Abstraktionsschicht soll Interaktionsdesignern diese Optionen zur Verfügung stellen. Dabei soll für den Interaktionsdesigner sowohl der Aufwand für die Integration von Hardware als auch die Bindung an die konkrete Hardware möglichst gering gehalten werden.

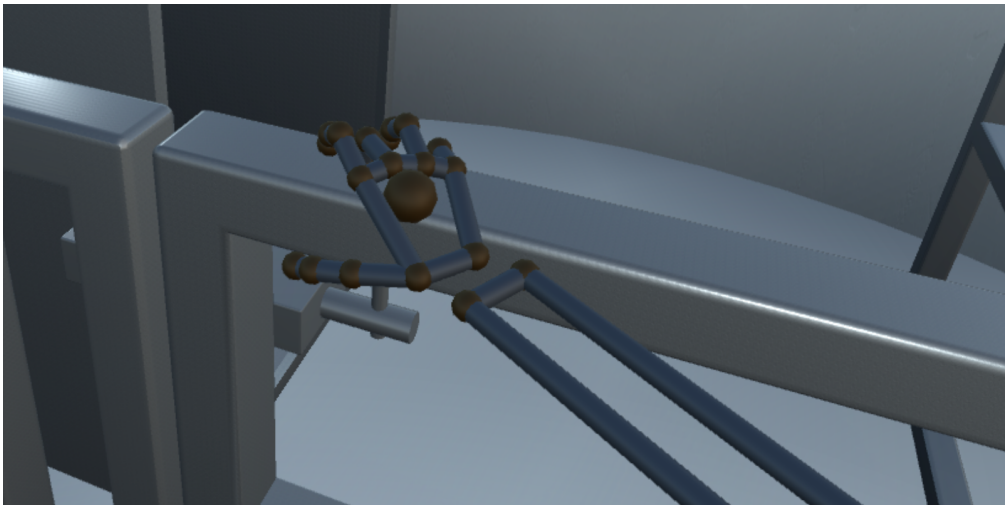
Um die Erfüllung der Anforderungen zu überprüfen, wurde die Abstraktionsschicht in das Projekt der Beispielszenarien eingebunden. Anschließend wurden die Interaktionen der Szenarien erneut gestaltet. Dabei fokussiert diese Auswertung den Einfluss der Abstraktionsschicht auf den Designprozess unter Berücksichtigung der oben genannten Ziele. Eine Untersuchung, welche Hardware sich am Besten für bestimmte Interaktionen eignet, ist nicht Teil dieser Arbeit. Diese Fragestellung ist sehr kontextabhängig und muss von den Interaktionsdesignern für die jeweiligen Szenarien entschieden werden. Die Abstraktionsschicht stellt dafür die technischen Möglichkeiten bereit und unterstützt so die Designer bei der Beantwortung der Fragestellung.
Während die einzelnen Interaktionen der Beispielszenarien innerhalb der Anwendung unterschiedliche Skripte aus dem Interaktionsframework nutzen und verschiedene semantische Ziele erfüllen, lassen sie sich auf einer abstrakten Ebene auf das Greifen von Objekten und das Drücken von Knöpfen als jeweilige Grundinteraktion verallgemeinern. Dabei sind abhängig von der vorhandenen Hardware verschiedene Umsetzungen für die Eingabe dieser Grundinteraktionen sowie der Übermittlung von entsprechendem Feedback als Reaktion des Systems auf diese möglich.
Im ursprünglichen Designprozess der Interaktionen platziert der Entwickler das Skript mit dem gewünschtem Verhalten aus dem Interaktionsframework auf das zugehörige Objekt und modifiziert anhand von Parametern das Verhalten. Dabei ist das Eingabegerät der HTC Vive Standard Controller, dessen Werte über das, vom Hersteller angebotene, Framework direkt abgerufen werden. Dies ist für alle Interaktionen festgelegt und in den Interaktionsskripten verankert. Sowohl das Greifen von Objekten als auch das Drücken der Knöpfe in den Beispielszenarien erfolgt durch den Trigger Knopf des Controllers zur Selektion des Objekts und einer Bewegung des Objekts innerhalb des, durch die Interaktionsskripte definierten, Verhaltens basierend auf der Positions- und Rotationsdaten des Controllers. Feedback erfolgt visuell und auditiv über das Interaktionsframework und die eingebauten Schnittstellen von Unity. Die visuelle Darstellung des Controllers erfolgt durch das Unity Plugin des Herstellers und ist in Abbildung 3.3 erkennbar. Eine Schnittstelle für andere Eingabe- und Ausgabegeräte ist nicht vorhanden.
Anders verhält es sich bei der Integration der Hardware-Abstraktionsschicht in die Szenarien. Über die Skripte des Interaktionsframeworks können nun die Typen von Eingabe- und Ausgabegeräten für Interaktionsobjekte der Szenarien spezifiziert werden. Dabei legt der Interaktionsdesigner für die Eingabe über das Skript fest, welche abstrakten Methoden oder Events des Hardwaretyps für Selektion und Manipulation des konkreten Interaktionsobjekts genutzt werden. Eine Mehrfachbelegung mit verschiedenen Typen für einzelne Objekte ist möglich. Der Interaktionsdesigner arbeitet über das Interakti-

onsframework nur mit den abstrakten Modellen und Methoden der Abstraktionsschicht. Er muss sich weder mit den internen Übersetzungen der Modelle und Methoden in die Strukturen der konkreten Hardware noch mit der dynamischen Auswahl und Darstellung der vorhandenen Hardware beschäftigen. Dies gilt analog auch für das Hinzufügen von Ausgabegeräten verschiedener Typen zur Übermittlung von Feedback des Systems.
Im Detail lässt sich die Anwendung der Abstraktionsschicht anhand einer Interaktion der Beispielszenarien darstellen, welche eine der identifizierten Grundinteraktionen repräsentiert. So ist das Öffnen des Tors im Fahrstuhl Szenario ein Beispiel für eine Interaktion, in der ein Objekt der virtuellen Welt gegriffen und anschließend manipuliert wird. Eine Visualisierung wie die Interaktion im alten Design des Szenarios aussieht, findet sich in Abbildung 3.3e. Das Verhalten der Tür wird dabei über das IOJ Hebel Skript des Interaktionsframeworks definiert. Im Unterschied zur ursprünglichen Version kann der Designer über das Skript nun jedoch die volle Bandbreite abstrakter Eingabe- und Ausgabetypen der Abstraktionsschicht für Selektion und Manipulation nutzen. Die Selektion des Tors kann mit Hilfe eines zusätzlichen Artefakts über das OnPress Event für einen beliebigen Knopf erfolgen oder durch die Hand-geschlossen Geste des abstrakten Hand-Modells. Dabei verwenden die Skripte des Interaktionsframeworks die abstrakten Get-Methoden beider Hardwaretypen, um die Position von Artefakt oder Hand zu erfragen. Diese werden innerhalb der Skripte einerseits benötigt, um zu überprüfen ob die Distanz zum Tor unter dem vom Designer festgelegten maximal Wert liegt und bilden andererseits die Grundlage zur internen Berechnung des Öffnungsgrades, wenn das Tor erfolgreich gegriffen wurde. Die Selektion und die damit einhergehende Manipulation des Tors endet, wenn die Distanz überschritten wird, ein ButtonRelease Event für den gewählten Knopf des zusätzlichen Artefakt Modells auftritt oder ein Hand-geöffnet Event des Hand-Modells registriert wird. In Abbildung 4.1 ist die Umsetzung der Beispielinteraktion über zwei entsprechende Eingabegeräte der Kategorien dargestellt.
Über die Sprachsteuerungskomponente kann das Tor zusätzlich auch noch mit Sprachbefehlen manipuliert werden. Hierfür wird das Objekt mit einem Skript der Abstraktionsschicht versehen, welches die Zuordnung von Wörtern zu Methoden in der Unity GUI ermöglicht. Die Befehle „Tor öffnen“ und „Tor schließen“ werden entsprechenden States des Skripts zugeordnet. Ein State ist hierbei, ein in den Skripten des Interaktionsframeworks definierter Zustand, den ein Objekt mit dem Verhalten des Skripts annehmen kann. States verfügen außerdem über Events, die ausgelöst werden, wenn das Objekt in den entsprechende State wechselt. Dies ist nützlich für die Ausgabe von Feedback. So kann über die Feedbackkomponente das haptische Feedback an den geschlossenen Zustand des Tors gekoppelt werden, um die virtuellen Restriktionen des Tores zu vermitteln.

Der Designer hat in diesem Test die Interaktion mit einem Objekt des Szenarios durch drei Typen von Eingabegeräten und einem Typ an Feedback umgesetzt. Dabei wurde gemäß der Anforderungen weder Wissen über konkrete Hardwareeigenschaften genutzt noch die Möglichkeiten der Interaktion mit anderen Objekten des Szenarios eingeschränkt. Der Designer musste dafür das Interaktionsobjekt nur mit einem Skript ergänzen sowie Zuordnungen zwischen Events zur Ein- und Ausgabe und dem gewünschten Verhalten des Objekts, repräsentiert durch Methoden und Events des Interaktionsskripts, vornehmen. Das Szenario wurde nach der Integration der Abstraktionsschicht durchlaufen, indem für jeden der integrierten Hardwaretypen ein unterstütztes Gerät angeschlossen wurde. Die Interaktion mit dem Tor war für alle Hardwaretypen erfolgreich.
Während die Abstraktionsschicht für alle Hardwaretypen bereits mindestens ein Gerät unterstützt, ergibt sich aus der Analyse der allgemeinen Hardware Entwicklung für VR-Anwendungen zusätzlich eine weitere Anforderung für die Hardware-Abstraktionsschicht. So muss diese aufgrund der rapiden Entwicklung neuer Eingabe- und Ausgabegeräte leicht erweiterbar sein, um den Interaktionsdesignern die volle Bandbreite an aktueller Hardware und Hardwaretypen bereitzustellen. Die Einbindung von neuer Hardware für die vorhandenen Hardwaretypen gestaltet sich, durch das Design der Abstraktionsschicht, sehr einfach. Es muss nur eine Zuordnung zwischen dem konkreten Modell der Hardware und dem abstrakten Modell des Typs vorgenommen und die hardwarespezifische Unity Integration in die Sammlung ergänzt werden. Für die Definition von neuen Hardwaretypen bietet sich eine Orientierung an den bestehenden abstrakten Modellen an, da die Designer so schneller den Umgang erlernen und die Erweiterung der Schnittstelle zum Interaktionsframework erleichtert wird.



(a) Zusätzliche Artefakte



(b) Hand-Tracking

Abbildung 4.1: Beispielinteraktion durch Eingabegeräte der Abstraktionsschicht

4.3 Technischer Ausblick

Während die Abstraktionsschicht, wie im vorherigen Kapitel beschrieben, Designer bei der Gestaltung von Interaktionen in VR bereits vielseitig unterstützt, hat sich durch die Integration in das Projekt der Beispielszenarien auch gezeigt, dass die aktuelle technische Umsetzung an einigen Stellen noch ausgebaut werden kann. So besitzt die Abstraktionsschicht ein Spektrum an Hardwaretypen zur Eingabe, aber bietet bis jetzt nur haptisches Feedback als Ausgabetyp an. Dabei gilt zu beachten, dass visuelles und auditives Feedback über die Pipeline von Interaktionsframework und Unity zur Verfügung gestellt wird. Das Interaktionsframework benutzt für die visuelle Darstellung der virtuellen Welt dazu das SteamVR Plug-In für Unity, welches einige HMDs als Ausgabegeräte unterstützt und so schon einen gewissen Grad an Abstraktion bereitstellt.
Um die Zuständigkeiten zwischen Interaktionsframework und Abstraktionsschicht klarer zu definieren, könnte die Schnittstelle zu visuellen Ausgabegeräten jedoch auch hier in eine entsprechende Komponente der Abstraktionsschicht verlagert werden. Die Komponente könnte durch ein Design analog zum Rest der Abstraktionsschicht folgend auch für die Integration von Geräten genutzt werden, welche von SteamVR nicht standardmäßig angeboten werden. Ähnlich verhält es sich beim auditiven Feedback. Dieses wird aktuell im Interaktionsframework generiert und an die Unity interne Schnittstelle für Audiogeräte weitergeleitet. Hierbei würde eine zusätzliche Komponente nicht wirklich weiter abstrahieren, sondern ausschließlich der Entkopplung dienen.
Weitere Feedback Typen zur Manipulation anderer Sinne existieren meist nur in sehr spezifischen Szenarien und besitzen selten eine Schnittstelle für Unity. Eine allgemeine Möglichkeit Geräte außerhalb der Unity-Umgebung über Betriebssystem- oder Netzwerkschnittstellen einzubinden, wäre denkbar, würde aber größere Adaptionen der Abstraktionsschicht oder sogar die Entwicklung einer externen Schnittstelle voraussetzen.
Neben diesen Erweiterungen zum Ausbau des Angebots der Abstraktionsschicht und einer eindeutigeren Rollenverteilung könnte ebenfalls die Usability verbessert werden. So wäre einerseits eine graphische Oberfläche zur Erstellung von Zuordnungen zwischen konkreter Hardware und abstrakten Modellen nützlich, damit der Prozess für den Interaktionsdesigner vereinfacht wird und keine Änderungen auf Code Ebene vorgenommen werden müssen. Andererseits folgt das Einbinden der unterschiedlichen Komponenten der Abstraktionsschicht keinem einheitlichen Schema und Einstellungsoptionen sind noch auf verschiedene Skripte verteilt. Hier könnte eine zentrale Oberfläche für Unity Abhilfe schaffen und den Umgang mit der Abstraktionsschicht erleichtern.

5 Zusammenfassung und Ausblick

In dieser Arbeit wurde eine Hardware-Abstraktionsschicht für VR-Anwendungen entworfen. Das grundlegende Ziel der Abstraktionsschicht ist, Interaktionsdesigner bei der Wahl geeigneter Eingabe- und Ausgabegeräte zur Gestaltung von Interaktionen in VR-Systemen zu unterstützen. Dabei wurden zuerst die Grundlagen, aktuelle Gerätetypen und die Rahmenbedingungen der Arbeit analysiert, um anschließend die Anforderungen an das Design zu spezifizieren. Die Abstraktionsschicht soll den Interaktionsdesigner ein breites Spektrum an Hardwaretypen für die Ein- und Ausgabe bereitstellen und eine Hardware unabhängige Schnittstelle bieten, um einen leichten Wechsel der konkreten Hardware und den Gebrauch verschiedener Typen an Geräten in einer VR-Anwendung zu ermöglichen. Weiterhin soll die Abstraktionsschicht erweiterbar sein, damit auch zukünftige Hardware-Neuentwicklungen integriert werden können.
Zur Erfüllung der Anforderungen wurden im Design der Abstraktionsschicht Komponenten für die verschiedenen Hardwaretypen entwickelt. Diese basieren auf der Idee, ein abstraktes Modell für die jeweiligen Typen zu definieren und die so vereinheitlichte Datenstruktur über generelle Methoden und Events anzubieten. Konkrete Geräte lassen sich mit einer Zuordnung zwischen ihrem Datenmodell und dem jeweiligen abstrakten Modell einbinden. Im Rahmen der Evaluation wurden die Komponenten in ein Beispielprojekt integriert und getestet. Dabei konnte nachgewiesen werden, dass das aktuelle Design die Anforderungen erfüllt, jedoch technisch noch ausgebaut werden kann.
Die Abstraktionsschicht verfolgt den Ansatz, die Integration von Hardware in VR-Systeme zu erleichtern. Dabei wurden die technische Funktionalität und ein möglicher Einsatz der Abstraktionsschicht innerhalb dieser Arbeit durch die nachträgliche Einbindung in ein bestehendes Projekt demonstriert. Aufbauend auf diesem Beispiel kann in zukünftigen Arbeiten untersucht werden, inwieweit der grundlegende Einsatz der Abstraktionsschicht zur Unterstützung von Interaktionsdesignern sinnvoll ist. Ziel ist es das Potential und die Grenzen der Abstraktionsschicht durch weiterführende Experimente zu erforschen.
Dabei haben Aspekte wie der Anwendungskontext oder der Integrationszeitpunkt im Designprozess Einfluss auf den Erfolg der Abstraktionsschicht bei der Unterstützung

von Interaktionsdesignern. Auch der Kenntnisstand der Interaktionsdesigner spielt eine wichtige Rolle für den Nutzen der Abstraktionsschicht, da diese zwar viele Abläufe vereinfacht, aber sowohl für die Integration in ein Projekt, als auch das Hinzufügen von Hardware immer noch einen gewissen Wissensstand über Hardware voraussetzt. Untersuchungen dieser Aspekte können als Ansatzpunkte für folgende Arbeiten zur Bewertung der Hardware-Abstraktionsschicht dienen.

Literaturverzeichnis

[1] Anthes, Christoph ; García-Hernández, Rubén J. ; Wiedemann, Markus ; Kranzlmüller, Dieter: State of the art of virtual reality technology. In: *2016 IEEE Aerospace Conference*, 2016, S. 1–19

[2] Becker, Jonathan: *Entwicklung eines Interaktionsframeworks zur Erstellung interaktiver Virtual Reality Szenarien.* Hamburg, Germany, HAW Hamburg, Diplomarbeit, Juli 2022. – URL https://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/master/becker.pdf

[3] Caserman, Polona ; Garcia-Agundez, Augusto ; Zerban, Alvar G. ; Göbel, Stefan: Cybersickness in current-generation virtual reality head-mounted displays: systematic review and outlook. In: *Virtual Real.* 25 (2021), S. 1153–1170

[4] Cummings, James ; Bailenson, Jeremy: How Immersive Is Enough? A Meta-Analysis of the Effect of Immersive Technology on User Presence. In: *Media Psychology* 19 (2015), 05, S. 1–38

[5] Danieau, Fabien ; Lecuyer, Anatole ; Guillotel, Philippe ; Fleureau, Julien ; Mollet, Nicolas ; Christie, Marc: Enhancing Audiovisual Experience with Haptic Feedback: A Survey on HAV. In: *IEEE Transactions on Haptics* 6 (2013), Nr. 2, S. 193–205

[6] DIN EN ISO 9241-11: *DIN EN ISO 9241-11:2018-11, Ergonomie der Mensch-System-Interaktion - Teil 11: Gebrauchstauglichkeit: Begriffe und Konzepte (ISO 9241-11:2018); Deutsche Fassung EN ISO 9241-11:2018.* 11 2018. – URL https://doi.org/10.31030%2F2757945

[7] Dionisio, J.: Virtual hell: a trip through the flames. In: *IEEE Computer Graphics and Applications* 17 (1997), Nr. 3, S. 11–14

[8] Dörner, R. ; Broll, W. ; Grimm, P. ; Jung, B.: *Virtual und Augmented Reality (VR/AR)*. Berlin : Springer Vieweg, 2013 (eXamen.press). – ISBN 978-3-642-28902-6

[9] Earnshaw, Rae ; Guedj, Richard ; Dam, Andries V. ; Vince, John: *Frontiers of Human-Centered Computing, Online Communities and Virtual Environments*. Berlin Heidelberg : Springer Science AND Business Media, 2012. – ISBN 978-1-447-10259-5

[10] Gerwens, Niklas ; Becker, Jonathan: *Toolchain für 3D-Objektmanipulation in VR-Trainingsumgebungen*. 2021. – URL https://users.informatik.haw-hamburg.de/~ubicomp/projekte/master2021-proj/beckerGP.pdf. – Zugriff: 01.07.2022

[11] Grunwald, Martin: *Begriffsbestimmungen Zwischen Psychologie Und Physiologie*. S. 1–14. In: *Der bewegte Sinn: Grundlagen und Anwendungen zur haptischen Wahrnehmung*, Birkhäuser Basel, 01 2001. – URL https://doi.org/10.1007/978-3-0348-8302-3_1. – ISBN 978-3-7643-6516-5

[12] Halata, Zdenek ; Baumann, Klaus I.: *Anatomy of receptors*. S. 85–92. In: Grunwald, Martin (Hrsg.): *Human Haptic Perception: Basics and Applications*, Birkhäuser Basel, 11 2008. – URL https://doi.org/10.1007/978-3-7643-7612-3_6. – ISBN 978-3-7643-7611-6

[13] Heinecke, Andreas M.: *Mensch-Computer-Interaktion*. Springer Berlin, 2012. – URL https://doi.org/10.1007/978-3-642-13507-1

[14] Hoshi, Takayuki ; Takahashi, Masafumi ; Iwamoto, Takayuki ; Shinoda, Hiroyuki: Noncontact Tactile Display Based on Radiation Pressure of Airborne Ultrasound. In: *IEEE Transactions on Haptics* 3 (2010), Nr. 3, S. 155–165

[15] Jerald, J.: *The VR Book: Human-Centered Design for Virtual Reality*. Association for Computing Machinery and Morgan & Claypool Publishers, 2015 (ACM Books). – URL https://books.google.de/books?id=ZEBiDwAAQBAJ. – ISBN 9781970001136

[16] Lopes, Pedro ; You, Sijing ; Ion, Alexandra ; Baudisch, Patrick: Adding Force Feedback to Mixed Reality Experiences and Games Using Electrical Muscle Stimulation. In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. New York, NY, USA : Association for Computing Machinery, 2018 (CHI

'18), S. 1–13. – URL https://doi.org/10.1145/3173574.3174020. – ISBN 9781450356206

[17] Mendes, D. ; Caputo, F. M. ; Giachetti, A. ; Ferreira, A. ; Jorge, J.: A Survey on 3D Virtual Object Manipulation: From the Desktop to Immersive Virtual Environments. In: *Computer Graphics Forum* 38 (2019), Nr. 1, S. 21–45. – URL https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13390

[18] Meyer, Uli ; Becker, Jonathan ; Mueller, Thomas ; Jeworutzki, André ; Draheim, Susanne ; Luck, Kai von: Asymmetrical Game Design Approaches Solve Didactic Problems in VR Engineer Trainings. In: *2021 7th International Conference of the Immersive Learning Research Network (iLRN)*, 2021, S. 1–5

[19] Motion, Leap: *HTC Vive FAQ: What You Need to Know About Leap Motion + SteamVR*. https://blog.leapmotion.com/leap-motion-htc-vive-faq/. – Zugriff: 09.06.2022

[20] Nogalski, Malte ; Fohl, Wolfgang: Acoustic redirected walking with auditory cues by means of wave field synthesis. In: *2016 IEEE Virtual Reality (VR)*, 2016, S. 245–246

[21] Ortega, F.R. ; Abyarjoo, Fatemeh ; Barreto, Armando ; Rishe, N. ; Adjouadi, Malek: *Interaction design for 3D user interfaces: The world of modern input devices for research, applications, and game development*. 01 2016. – 1–728 S

[22] Ragozin, Kirill ; Zheng, Dingding ; Chernyshov, George ; Hynds, Danny: Sophroneo: Fear Not. A VR Horror Game with Thermal Feedback and Physiological Signal Loop. In: *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*. New York, NY, USA : Association for Computing Machinery, 2020 (CHI EA '20), S. 1–6. – URL https://doi.org/10.1145/3334480.3381659. – ISBN 9781450368193

[23] Roginska, Agnieszka ; Geluso, Paul: *Immersive Sound - The Art and Science of Binaural and Multi-Channel Audio*. 2017. – ISBN 978-1-317-48010-5

[24] Schneider, Daniel ; Otte, Alexander ; Kublin, Axel S. ; Martschenko, Alexander ; Kristensson, Per O. ; Ofek, Eyal ; Pahud, Michel ; Grubert, Jens: Accuracy of Commodity Finger Tracking Systems for Virtual Reality Head-Mounted Displays. In: *2020 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, 2020, S. 804–805

[25] Sinclair, Mike ; Ofek, Eyal ; Holz, Christian ; Choi, Inrak ; Whitmire, Eric ; Strasnick, Evan ; Benko, Hrvoje: Three Haptic Shape-Feedback Controllers for Virtual Reality. In: *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, 2018, S. 777–778

[26] Sutherland, Ivan E.: The Ultimate Display. In: *Proceedings of the IFIP Congress*, 1965, S. 506–508

[27] Suzuki, Y. ; Kobayashi, M.: Air jet driven force feedback in virtual reality. In: *IEEE Computer Graphics and Applications* 25 (2005), Nr. 1, S. 44–47

[28] Unity: *Unity XR plug-in Framework*. – URL https://docs.unity3d.com/Manual/XR.html. – Zugriff: 04.07.2022

[29] Valve: *SteamVR Unity Plugin*. – URL https://valvesoftware.github.io/steamvr_unity_plugin/. – Zugriff: 04.07.2022

[30] Visbox: *Visbox CAVE Systeme*. – URL https://visbox.com. – Zugriff: 06.07.2022

[31] Vuo: *VUO: Leap Motion hand skeleton points*. https://vuo.org/component/vuo.leap. – Zugriff: 09.06.2022

[32] Wang, Dangxiao ; Song, Meng ; Naqash, Afzal ; Zheng, Yukai ; Xu, Weiliang ; Zhang, Yuru: Toward Whole-Hand Kinesthetic Feedback: A Survey of Force Feedback Gloves. In: *IEEE Transactions on Haptics* 12 (2019), Nr. 2, S. 189–204

A Anhang

Bezeichner	Typ	Beschreibung
artefactPosition	Vector3	Die Position des Artefakts
artefactRotation	Vector3	Die Rotation des Artefakts
artefactVelocity	Vector3	Die Geschwindigkeit des Artefakts
artefactAcceleration	Vector3	Die Beschleunigung des Artefakts
grip	Axis (0.0 to 1.0)	Die Griffstärke des Nutzers
gripButton	bool	Binärer Wert ob das Artefakt gegriffen wird
menuButton	bool	Binärer Wert ob der Menü Knopf gedrückt wird
primaryButton	bool	Binärer Wert ob der primäre Knopf gedrückt wird
primaryTouch	bool	Binärer Wert ob der primäre Knopf berührt wird
secondaryButton	bool	Binärer Wert ob der sekundäre Knopf gedrückt wird
secondaryTouch	bool	Binärer Wert ob der sekundäre Knopf berührt wird
trigger	Axis (0.0 to 1.0)	Die Druckstärke auf den Trigger Knopf
triggerButton	bool	Binärer Wert ob der Trigger Knopf gedrückt wird
triggerTouch	bool	Binärer Wert ob der Trigger Knopf berührt wird
primary2DAxis	Vector2 Axis (-1.0 to 1.0)	X- und Y-Achsenwerte des primären Touchpads oder Joysticks
primary2DAxisButton	bool	Binärer Wert ob das primäre Touchpad/Joystick gedrückt wird
primary2DAxisTouch	bool	Binärer Wert ob das primäre Touchpad/Joystick berührt wird
secondary2DAxis	Vector2 Axis (-1.0 to 1.0)	X- und Y-Achsenwerte des sekundären Touchpads oder Joysticks
secondary2DAxisButton	bool	Binärer Wert ob das sekundäre Touchpad/Joystick gedrückt wird
secondary2DAxisTouch	bool	Binärer Wert ob das sekundäre Touchpad/Joystick berührt wird

Abbildung A.1: Abstraktes Modell für Zusätzliche Artefakte - Eingabedaten

Bezeichner	Typ	Beschreibung
isTracked	bool	Binärer Wert ob Position und/oder Rotation des Artefakts bestimmt werden
isController	bool	Binärer Wert ob das Artefakt die Eigenschaften eines Controllers besitzt
hasTouch	bool	Binärer Wert ob das Artefakt Berührungserkennung unterstützt
OnChange	UnityEvent	Event ausgelöst wenn der Wert sich im Vergleich zum letzten Frame geändert hat
OnPress	UnityEvent	Event ausgelöst wenn der Status des Knopfs sich auf gedrückt ändert
OnRelease	UnityEvent	Event ausgelöst wenn der Status des Knopfs sich auf nicht gedrückt ändert
OnTouch	UnityEvent	Event ausgelöst wenn der Status des Knopfs sich auf berührt ändert
Left	bool	Binärer Wert ob das Artefakt in der linken Hand gehalten wird
Right	bool	Binärer Wert ob das Artefakt in der rechten Hand gehalten wird

Abbildung A.2: Abstraktes Modell für Zusätzliche Artefakte - Events und Hilfsmethoden

Index	Bezeichner	Joint
0	Wrist	Wrist
1	Thumb0	Thumb Metacarpal Bone
2	Thumb1	Thumb Proximal Bone
3	Thumb2	Thumb Distal Bone
4	Thumb3	Thumb Tip
5	Index0	Index Metacarpal Bone
6	Index1	Index Proximal Bone
7	Index2	Index Intermediate Bone
8	Index3	Index Distal Bone
9	Index4	Index Tip
10	Middle0	Middle Metacarpal Bone
11	Middle1	Middle Proximal Bone
12	Middle2	Middle Intermediate Bone
13	Middle3	Middle Distal Bone
14	Middle4	Middle Tip
15	Ring0	Ring Metacarpal Bone
16	Ring1	Ring Proximal Bone
17	Ring2	Ring Intermediate Bone
18	Ring3	Ring Distal Bone
19	Ring4	Ring Tip
20	Pinky0	Pinky Metacarpal Bone
21	Pinky1	Pinky Proximal Bone
22	Pinky2	Pinky Intermediate Bone
23	Pinky3	Pinky Distal Bone
24	Pinky4	Pinky Tip

Abbildung A.3: Abstraktes Modell für Hand-Tracking - Joints

Erklärung zur selbstständigen Bearbeitung einer Abschlussarbeit

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

Ort	Datum	Unterschrift im Original