**Multimodale Telepräsenz mit dem humanoiden Roboter NAO und VR-Brille**

**Studienarbeit**

des Studienganges Angewandte Informatik

an der Dualen Hochschule Baden-Württemberg, Karlsruhe

Ersteller: Fabian Dogendorf, Isabella Schmidt

Abgabedatum: Abgabedatum einfügen

Abgabeort: Duale Hochschule Baden-Württemberg, Karlsruhe

Matrikelnummern: 8053885, 7927456

Kursbezeichnung DHBW: TINF16B4

Betreuer: Prof. Dr. Hans-Jörg Haubner

# Sperrvermerk

Die vorliegende Studienarbeit ist mit einem Sperrvermerk versehen und wird ausschließlich zu Prüfungszwecken des Studienganges Angewandte Informatik der Dualen Hochschule Baden-Württemberg Karlsruhe vorgelegt. Jede Einsichtnahme und Veröffentlichung, auch von Teilen der Arbeit, bedarf der vorherigen Zustimmung der Siemens AG.

# Eidesstattliche Versicherung

Hiermit erklären wir an Eides statt, dass wir die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe. Inhaltlich und wörtlich aus den Quellen entnommene Stellen habe ich als solche kenntlich gemacht.

Inhaltsverzeichnis

[Sperrvermerk 2](#_Toc6153509)

[Eidesstattliche Versicherung 3](#_Toc6153510)

[Abbildungsverzeichnis 6](#_Toc6153511)

[Tabellenverzeichnis 6](#_Toc6153512)

[1. Einführung 7](#_Toc6153513)

[1.1. Aufgabenstellung und Zielsetzung 7](#_Toc6153514)

[1.2. Vorgehensweise 8](#_Toc6153515)

[2. Grundlagen 9](#_Toc6153516)

[2.1. Theoretische Grundlagen 9](#_Toc6153517)

[2.1.1. Telepräsenz 9](#_Toc6153518)

[2.1.2. Robotik/Kinematik 12](#_Toc6153519)

[2.1.3. Virtual Reality 13](#_Toc6153520)

[2.2. Technische Grundlagen 15](#_Toc6153521)

[2.2.1. NAO-Roboter 15](#_Toc6153522)

[2.2.2. Choreographe Suite 17](#_Toc6153523)

[2.2.3. HTC Vive Pro 19](#_Toc6153524)

[2.2.4. Unity 20](#_Toc6153525)

[2.2.4.1. Struktur der Benutzeroberfläche 20](#_Toc6153526)

[2.2.4.2. Grundlegende Building-Blöcke 22](#_Toc6153527)

[2.2.4.3. SteamVR-Plugin 23](#_Toc6153528)

[2.2.5. Maya 24](#_Toc6153529)

[3. Entwurf Kurze Einleitung? Sonst 3 Überschriften nacheinander 26](#_Toc6153530)

[3.1. Theoretische Vorgehensweise 26](#_Toc6153531)

[3.1.1. Anforderungsanalyse 26](#_Toc6153532)

[3.1.2. Geplantes Vorgehen 27](#_Toc6153533)

[3.2. Konzeption 28](#_Toc6153534)

[3.2.1. Aufbau der Anwendung 28](#_Toc6153535)

[3.2.2. Entwurf der virtuellen Umgebung 30](#_Toc6153536)

[4. Implementierung 33](#_Toc6153537)

[4.1. Erstellung der virtuellen Umgebung 33](#_Toc6153538)

[4.1.1. Design der 3D-Objekte 33](#_Toc6153539)

[4.1.2. Aufbau der virtuellen Umgebung 36](#_Toc6153540)

[4.1.2.1. Aufbau der erstellten Szenen 36](#_Toc6153541)

[4.1.2.2. Ordnerstruktur 40](#_Toc6153542)

[4.2. Kommunikation mit dem NAO-Roboter 42](#_Toc6153543)

[4.3. Steuerung des NAO-Roboters 42](#_Toc6153544)

[4.4. Umsetzung der Telepräsenz 42](#_Toc6153545)

[5. Ergebnis 43](#_Toc6153546)

[6. Zusammenfassung und Ausblick 44](#_Toc6153547)

[6.1. Persönliches Feedback 44](#_Toc6153548)

[6.2. Erweiterungsmöglichkeiten 44](#_Toc6153549)

[Quellen 45](#_Toc6153550)

# Abkürzungsverzeichnis

* SDK: Software Development Kit
* API: Application Programming Interface

# Abbildungsverzeichnis

[Abbildung 1: Teleroboter 10](#_Toc6153551)

[Abbildung 2: Gegenübersellung VR, AR und MR 14](#_Toc6153552)

[Abbildung 3: NAO-Roboter 15](#_Toc6153553)

[Abbildung 4: HTC VIVE Pro 19](#_Toc6153554)

[Abbildung 5: Benutzeroberfläche Unity 21](#_Toc6153555)

[Abbildung 6: Architektur-Entwurf 28](#_Toc6153556)

[Abbildung 7: Entwurf Head-Display 30](https://d.docs.live.net/a6b23e99b75a8a99/Documents/Studienarbeit_Entwurf.docx#_Toc6153557)

[Abbildung 8: Entwurf Labor 31](https://d.docs.live.net/a6b23e99b75a8a99/Documents/Studienarbeit_Entwurf.docx#_Toc6153558)

[Abbildung 9: Entwurf Willkommensbildschirm 32](https://d.docs.live.net/a6b23e99b75a8a99/Documents/Studienarbeit_Entwurf.docx#_Toc6153559)

[Abbildung 10: Design-Prozess in Maya 35](#_Toc6153560)

[Abbildung 11: Cubemap Skybox 38](#_Toc6153561)

[Abbildung 12: Ordnerstruktur Unity-Projekt 40](#_Toc6153562)

# Tabellenverzeichnis

[Tabelle 1: Building-Blöcke in Unity 25](#_Toc6333202)

[Tabelle 2: Ordnerstruktur Unity-Projekt 43](#_Toc6333203)

[Tabelle 3: Kernmodule des NAOqi-Frameworks 45](#_Toc6333204)

# Einführung

## Aufgabenstellung und Zielsetzung

Im Rahmen dieser Studienarbeit wurde eine Anwendung konzipiert und implementiert, die es einem menschlichen Benutzer ermöglicht, einerseits einen entfernten Roboter fernzusteuern und andererseits in der entfernten Umgebung visuell präsent zu sein. Für die Umsetzung der Studienarbeit standen ein NAO-Roboter der Firma Aldebaran sowie die Virtual-Reality-Brille HTC Vive Pro zur Verfügung. Mit Hilfe dieser Hardware wurde im Verlauf der Studienarbeit eine virtuelle Umgebung konzipiert und erstellt. Der Operator der Virtual Reality Brille kann sich in dieser Umgebung frei bewegen. Relevant war hierbei insbesondere die Umsetzung einer multimodalen Telepräsenz. Die Übertragung von Befehlen und Sensordaten sollte in beide Richtungen erfolgen können: Sowohl vom Roboter zum Anwender als auch umgekehrt. Der Telepräsenzroboter überträgt hierbei die Wahrnehmung seiner Sensoren an den Anwender und die übermittelten Daten werden dem Benutzer in der virtuellen Umgebung aufbereitet angezeigt. Über die Virtual-Reality-Brille wird es dem Benutzer ermöglicht, in die entfernte Umgebung einzutauchen und den Effekt der Immersion zu spüren. Über die Controller der HTC Vive Pro besteht für ihn zusätzlich die Möglichkeit, die Aktionen des NAO-Roboters zu manipulieren und diesen direkt aus der entfernten Umgebung zu steuern.

Dieses Dokument dient dazu, dem Leser einen Überblick über das Thema dieser Studienarbeit zu verschaffen und die Projektdurchführung zu dokumentieren. Dadurch soll ermöglicht werden, dass andere Projekte von dem gewonnenen Wissen profitieren.

## Vorgehensweise

Diese Studienarbeit dokumentiert die Konzeption und Umsetzung eines Programms zur multimodalen Telekomunikation zwischen einem menschlichen Benutzer und einem entfernten Nao-Roboter. Der Benutzer begibt sich dabei über eine Virtual Reality Brille in eine virtuelle Umgebung. Diese mit Unity erstellte Spielwelt stellt Interaktions- und Kommunikationsmöglichkeiten für den Roboter bereit.

Zunächst werden in Kapitel 2 (Grundlagen) die theoretischen und technischen Grundlagen erläutert. Dieses Kapitel enthält neben theoretischen Grundprinzipien auch die verwendete Software. In Kapitel 3 (Entwurf) wird der Entwurf der Softwarelösung vorgestellt. Dieser beinhaltet die theoretische Vorgehensweise und die Konzeption der entstandenen Anwendung. Dabei werden auch einige Designentwürfe des virtuellen Raumes vorgestellt. Anschließend wird in Kapitel 4 (Implementierung) die Implementierung und Umsetzung der Anwendung dokumentiert. Dokumentiert ist sowohl der Prozess der Softwareentwicklung als auch das Erstellen der virtuellen Umgebung. Die Kommunikation zwischen Roboter und Anwendung wird ebenfalls erläutert. In Kapitel 5 (Ergebnis) wird das Ergebnis der Studienarbeit vorgestellt. Das abschließende Kapitel 6 (Zusammenfassung und Ausblick) enthält einen Ausblick und ein persönliches Feedback.

* AB HIER IST EIN VERGLEICH MIT KAPITEL 3 NÖTIG!

Zur Einarbeit in die Thematik werden zunächst Informationen über die vorhandene Hardware und die Ansteuerung dieser gesammelt. Anschließend werden erste Programmierversuche mit dem NAO über die Software *Choreographe* gestartet und eine virtuelle Testumgebung in der Game-Engine *Unity* erstellt. Durch Tests mit der Virtual Reality Brille, der HTC Vive Pro, wird die korrekte Funktionsweise der Hardware verifiziert. Nachdem erste Erfahrungen in der Ansteuerung des NAO-Roboters, sowie der Erstellung von virtuellen Umgebungen mit Hilfe der Anwendung Unity gemacht wurden, beginnt die Projektkonzeption. In der Konzeptionsphase werden Vorstellungen zum Design der virtuellen Umgebung festgehalten, verglichen und diskutiert. Des Weiteren werden Überlegungen zum Verbindungsaufbau und Übertragen der Daten zum Roboter gemacht.

Die Entwicklung der virtuellen Umgebung und die Ansteuerung des NAO-Roboters wurden anschließend in drei Implementierungsphasen umgesetzt. Nachdem Grundkenntnisse über die Ansteuerung des NAO-Roboters sowie die Erstellung virtueller Umgebungen gesammelt wurden, konnte die Phase der Konzeption einer virtuellen Umgebung eingeleitet werden. In dieser Phase lag der Fokus auf der Erstellung einer virtuellen Umgebung mit Hilfe der Software Unity für die HTC Vive Pro. Außerdem sollte über vorhandene Schnittstellen eine Verbindung der virtuellen Umgebung mit dem NAO-Roboter ermöglicht werden. Die erste Implementierungsphase sollte als Grundlage für die Ermöglichung der Telepräsenz dienen.

Die nächste Implementierungsphase hatte den Schwerpunkt, die Telepräsenz umzusetzen. Hierbei sollten das Kamera- sowie Tonsignal des NAO-Roboters an den Anwender über die VR-Brille übertragen werden. Innerhalb der virtuellen Umgebung wird ein Videostream des NAO-Roboters angezeigt, so dass der Operator nachverfolgen kann, welche Aktionen der Roboter ausführt. Über die virtuelle Umgebung kann der Operator dem Roboter Anweisungen geben, wie beispielsweise Winken oder Hinsetzen geben, die der NAO umsetzen soll. In der finalen Implementierungsphase wurde die Steuerung des Roboters implementiert. Innerhalb der virtuellen Umgebung soll es möglich sein, den Roboter über die Datenhandschuhe der VR-Brille zu steuern. Durch das Abfangen der Signale der Controller kann der NAO-Roboter direkt von dem Anwender gesteuert werden.

# Grundlagen

## Theoretische Grundlagen

### Telepräsenz

Die Technologien der Telepräsenz im Bereich der Robotik ermöglichen es einem menschlichen Benutzer in einer virtuell erzeugten, entfernten Umgebung visuell präsent zu sein. Der Anwender kann durch den Einsatz von Technologien der Virtual Reality die künstliche Wirklichkeit realistisch erleben und in diese vollständig eintauchen. Das Besondere der Technologie ist, dass der Roboter und Mensch eins werden, indem die Sensor-Signale des Roboters an den Operator übertragen werden. Über ein Head-Mounted-Display können die erzeugten Video-Signale in einer 3D-Umgebung visualisiert werden. Die Kopf- und Armpositionen des Benutzers in der digitalen Umgebung können direkt an den Roboter mit Hilfe einer VR-Brille sowie Datenhandschuhen übertragen werden.

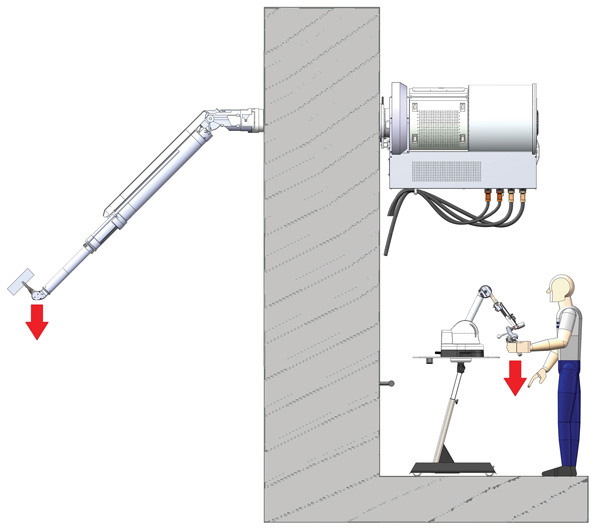


Abbildung 1: Teleroboter [[1]](#footnote-2)

Eine wichtige Komponente der Technologie der Telepräsenz ist das sogenannte Telemanipulations-System. Hierunter versteht man Werkzeuge, mit denen es einem Menschen ermöglicht wird, Gegenstände zu manipulieren, ohne sie mit eigenen Händen anzufassen. Anwendung finden diese Systeme vor allem in Laboren, in denen mit Gefahrenstoffen gearbeitet wird und die somit ein Sicherheitsrisiko für den Menschen darstellen. Die Technik der Telerobotik ermöglicht, dass Bewegungskommandos komfortabel von einem ausgebildeten Anwender ausgeführt werden können, ohne dass dieser sich den mit der Aufgabe verbundenen Gefahren stellen muss. Der Operator hat eine sichere Entfernung und Abschirmung zum gefährlichen Arbeitsplatz und kann den Roboter über einen Videobildschirm beobachten und steuern. Oft werden die Telemanipulatoren dem Menschen direkt nachempfunden, damit eine intuitive Steuerung möglich ist. [[2]](#footnote-3) Ein Beispiel hierfür ist ein Roboterarm, wie in Abbildung xx.xx dargestellt ist. Vorteil dieser Systeme ist es, dass der Mensch zwar Aufgaben durch einen autonomen Roboter durchführen lassen kann, jedoch weiterhin stets die Kontrolle über das System hat. Falls unerwartete Ereignisse eintreten, kann er jederzeit direkt eingreifen und nach eigenem Ermessen eine Entscheidung über die nächste Aktion des Teleroboters treffen.[[3]](#footnote-4)

Im Rahmen dieser Studienarbeit soll der Spezialfall der multimodalen Telepräsenz umgesetzt werden. Darunter versteht man die Kommunikation zwischen Mensch und Roboter, die über verschiedene Kanäle durchgeführt wird. Es werden somit nicht nur Signale vom Menschen zum Roboter gesendet, um den Roboter zu steuern, sondern zusätzlich Signale vom Roboter an den Operator, wie beispielsweise Bild- oder Tonmaterial. Die Umsetzung einer multimodalen Telepräsenz hat den Vorteil, dass durch die beidseitige Übertragung der Sensor-Daten die Steuerung des Roboters direkt nachverfolgt werden kann.

### Robotik/Kinematik

Arten erklären -> humanoider Roboter besonders erwähnen

### Virtual Reality

Unter dem Begriff Virtual Realityversteht man die Darstellung einer digital erstellten künstlichen Wirklichkeit. Eingesetzt wird die Technik sowohl in der Entertainment-Branche als auch in den Medien oder der Medizin. Der Trend prägt inzwischen zunehmend unseren Alltag und die Einsatzgebiete der Technologie werden immer vielseitiger. Beispielsweise kann heutzutage ein Flug in 360 Grad durch den Einsatz von Virtual-Reality-Hardware simuliert werden. Interessant ist außerdem der Einsatz dieser Technologie für die Schulung von Operationstechniken der Medizin über die virtuelle Realität geschult werden. [[4]](#footnote-5)

Mit der Erstellung einer virtuellen Umgebung wird es uns ermöglicht, in eine Realität computergenerierter Bilder und Sounds einzutauchen. Übertragen werden die Daten meist über Head-Mounted-Displays, die sogenannten Virtual Reality-Brillen.[[5]](#footnote-6) Das in der VR-Brille eingebaute Display stellt künstlich erzeugte Bilder dar, die den Anwender vollständig umgeben. Die VR-Brillen verfügen außerdem über Sensoren, welche die Lage und die Position des Anwenders bestimmen können und somit das freie Bewegen in der Umgebung ermöglichen.[[6]](#footnote-7) Oft gibt es neben der Bildübertragung zusätzlich Datenhandschuhe, durch welche der Anwender sich in der virtuellen Umgebung frei bewegen und mit den vorhandenen Objekten interagieren kann.[[7]](#footnote-8) Durch das Ermöglichen der virtuellen Interaktion wird der Effekt der Immersion erzeugt. Dieser beschreibt das vollständige Eintauchen in die virtuelle Realität und somit das Wahrnehmen der computergenerierten Wirklichkeit als reale Welt. Die Grenzen zwischen realer und virtueller Umgebung verschwimmen und die Realität sowie die Wahrnehmung der eigenen Person treten in den Hintergrund.[[8]](#footnote-9)

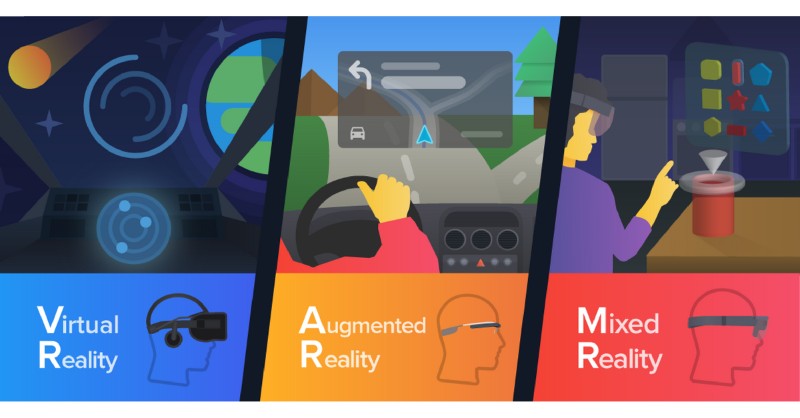


Abbildung 2: Gegenüberstellung VR, AR und MR [[9]](#footnote-10)

Das Konzept der künstlich erzeugten Realität kann in verschiedene Ausprägungen auftreten: Virtual Reality, Augmented Reality und Mixed Reality. Abgegrenzt werden kann die Technologie der zuvor erklärten Virtual Reality von der verwandten Augmented Reality. Diese kennzeichnet sich durch die Erweiterung der realen Welt durch digitale Objekte in Echtzeit. Im Gegensatz zur virtuellen Umgebung befindet sich der Anwender in der Realität, die durch virtuelle Informationen erweitert wird, bei der Virtual Reality hingegen wird die ganze Umgebung virtuell dargestellt.[[10]](#footnote-11) Eine weitere Ausprägungsmöglichkeit ist die sogenannte Mixed Reality. Hierunter versteht man eine Zusammenführung der realen Welt mit einer virtuellen Realität. Dabei verschwimmen die Grenzen zwischen synthetischen und natürlichen Elementen und werden nicht wie bei der Augmented Reality lediglich zusätzlich eingeblendet. [[11]](#footnote-12) Diese Studienarbeit beschäftigt sich lediglich mit der Umsetzung der Augmented Reality. Die Vorgehensweise zur Erstellung der virtuellen Umgebung wird in Kapitel xx.xx beschrieben.

## Technische Grundlagen

### NAO-Roboter

NAO ist ein programmierbarer humanoider Roboter, der von der Firma Aldebaran Robotics entwickelt wurde und im Jahr 2006 zum ersten Mal an den Markt kam. Eingesetzt wird er vor allem zu Bildungs- sowie Forschungszwecken. Dies liegt insbesondere an der benutzerfreundlichen Entwicklungsumgebung des NAO-Roboters, welche die Programmierung des Roboters über Bausteine auch für Programmieranfänger ermöglicht.

Der Roboter ist ein mittelgroßer Roboter mit einer Größe von circa 60 cm. Der Roboter hat sich seit seiner Entwicklung stets weiterentwickelt, derzeit ist bereits die sechste Version auf dem Markt. Der Roboter hat sechs wichtige Charakteristika:

* 25 Freiheitsgrade: Der Roboter verfügt über eine hohe Bewegungsfreiheit, da er aus 25 Gelenken besteht, die es ihm ermöglichen, sich in seiner Umgebung zu bewegen und anzupassen.
* Touch-Sensoren: Der NAO-Roboter ist mit 7 Touch-Sensoren ausgestattet. Diese befinden sich am Kopf, an den Händen und den Füßen. Der Roboter kann seine Umgebung somit wahrnehmen und sich im Raum lokalisieren.
* 4 Mikrophone und Lautsprecher: Die Mikrophone und Lautsprecher ermöglichen die direkte Interaktion mit Menschen.
* Spracherkennung: Der Roboter verfügt über Spracherkennungssysteme, die 20 verschiedene Sprachen unterstützen.
* 2D-Kameras: NAO ist mit 2 Kameras ausgestattet, die Umrisse, Objekte und Menschen erkennen können.
* Voll programmierbare Plattform: Der Roboter ist frei programmierbar in den Sprachen C# und Python. [[12]](#footnote-13)



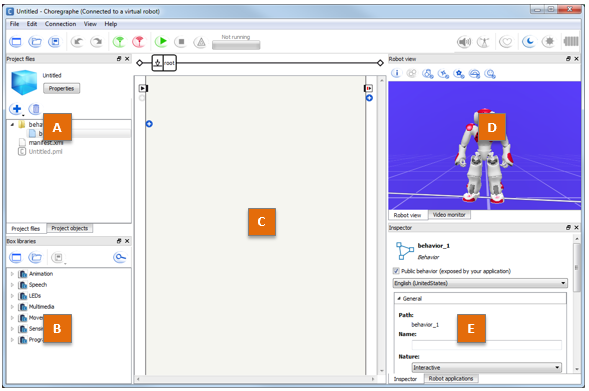
Abbildung 3: NAO-Roboter [[13]](#footnote-14)

In den Roboter ist das Betriebssystem NAOqi integriert und kontrolliert das Verhalten des Roboters. Das Betriebssystem wird bei allen Robotern der Firma Aldebaran eingesetzt und bietet eine zuverlässige und plattformunabhängige Robotikumgebung. Der Roboter kann seinen ganzen Körper bewegen und auf Ereignisse in seiner Umgebung reagieren. Sein Bewegungsmodul beruht auf der inversen Gesamtkinematik, bei welcher verschiedene Parameter wie beispielsweise das Gleichgewicht, bei der Bewegung der Gelenke berücksichtigt werden. Über zwei Kameras kann der Roboter seine Umgebung erkennen und mit Hilfe von Algorithmen zur Ortung und Erkennung von Gesichtern die Umgebung analysieren und mit ihr interagieren.[[14]](#footnote-15)

### Choreographe Suite

Die Choreographe Suite ist eine Desktop-Anwendung, die zur Steuerung des Nao-Roboters verwendet werden kann. Über die Software kann nach Angabe der zugehörigen IP-Adresse des Nao-Roboters, der Roboter direkt angebunden werden und sein Verhalten kontrolliert und gesteuert werden. Es können individuelle Applikationen erstellt werden, die beispielsweise Dialoge oder Verhalten enthalten, ohne selbst Code zu schreiben.   
Die Anwendung verfügt über die folgenden Funktionen:

* Erstellen von Animationen, Verhalten und Dialogen
* Testen der erstellten Funktionen auf einem simulierten oder realen Roboter
* Überwachung und Kontrolle eines Nao-Roboters
* Anreicherung der vordefinierten Verhaltensskripte mit selbst implementiertem Python-Code. [[15]](#footnote-16)



Die Abbildung xx.xx zeigt die Struktur des Software-Tools. Die Benutzeroberfläche lässt sich in verschiedene Bereiche gliedern. Das Project Files Panel (1) zeigt alle vom Anwender konfigurierten Einstellungen des angelegten Projektes sowie alle gespeicherten Dateien innerhalb des Projekts. Das Flow Diagram Panel (2) dient zur Übersicht aller verfügbaren Box-Bibliotheken. Der Anwender kann diese vordefinierten Skripte per Drag-and-Drop in das Flow Diagram Panel (3) ziehen. Die Box-Bibliotheken stellen wiederverwendbare Komponenten dar, die das Verhalten des Nao-Roboters steuern können. Viele Standard-Funktionen, wie beispielsweise Reden oder Gehen, sind bereits vordefiniert und können sofort verwendet werden. Sollte der Anwender spezifischere Aktionen brauchen, kann er selbst eine Box-Bibliothek anlegen und das Verhalten über ein Python-Skript definieren. In dem Panel Flow Diagram kann das gewünschte Verhalten des Nao-Roboters definiert werden. Die Box-Bibliotheken können in einem Flussdiagramm miteinander verknüpft werden und an den Roboter gesendet werden. Der Nao führt dann in der definierten Reihenfolge die Aktionen aus. Das Verhalten des Roboters kann ebenso in dem Robot View Panel (4) direkt nachvollzogen werden. In dem Inspector Panel (5) werden Eigenschaften des ausgewählten Objekts angezeigt, die durch den Anwender konfiguriert werden können.

BILD BEISPIEL FLUSSDIAGRAMM

### HTC Vive Pro

Die HTC Vive Pro ist ein Virtual-Reality-Headset des für Smartphone bekannten Herstellers HTC, das in Kollaboration mit dem Softwareunternehmen Valve entwickelt wurde. Vorgestellt wurde der Vorgänger HTC Vive erstmalig im Jahr 2015 bei dem Mobile World Congress, die Markteinführung erfolgte im nachfolgenden Jahr. Das neue Modell HTC Vive Pro, welches bei der Durchführung dieser Studienarbeit verwendet wurde, kam drei Jahre später mit verbesserter Technologie auf den Markt.



Abbildung 4: HTC VIVE Pro[[16]](#footnote-17)

Ausgestattet ist die HTC Vive Pro mit einer High-Fidelity-Grafik, womit realistisch Grafiken virtueller Umgebung dargestellt werden können. Das eingebaute Dual-OLED-System stellt pro Auge 1440 x 1600 Pixel dar. Über die 32 eingebauten Headset-Sensoren wird eine 360 Grad Bewegungsverfolgung ermöglicht, sodass der Anwender ringsum von den virtuell erstellten Bildern umgeben ist. Die virtuelle räumliche Ebene kann bis zu einer Größe von 5x5 Metern angezeigt werden. Um das Erlebnis abzurunden, sind zusätzlich zu dem Bildschirm Kopfhörer in den Kopfbügel der neuen Version des Headsets integriert. [[17]](#footnote-18)

### Unity

Die Software Unityist eine Game-Engine, die „Spieleersteller mit dem notwendigen Satz von Funktionen versorgt, um schnell und effizient Spiele erstellen zu können“ [[18]](#footnote-19). Game-Engines bieten das Gerüst für die Entwicklung von 2D- sowie 3D-Spielen. Die Anwendung Unity bietet Ressourcen, wie zum Beispiel Grafiken oder Audio-Dateien, um grafische Oberflächen zu entwerfen. Zusätzlich zu den von Unity angebotenen Ressourcen, können 3D-Modelle von Maya oder Photoshop oder anderer Anwendungen importiert werden. Weiterhin können beispielsweise Animationen, Beleuchtungen und Sound-Effekte zu den Szenen hinzugefügt werden, um diese beliebig nach den Anforderungen des Entwicklers anzupassen. Ein wichtiges Feature des Software-Tools ist das Scripting. Scripts ermöglichen es, dem Spiele-Entwickler die Logik von den Komponenten des Spiels zu definieren. Unity vereinfacht die Entwicklung, indem es vordefinierte Scripts anbietet, die beispielsweise das Umschauen des Operators in dem virtuellen Raum durch eine VR-Brille ermöglicht. [[19]](#footnote-20)

### Struktur der Benutzeroberfläche

Die Abbildung xx.xx zeigt die Standardoberfläche von Unity. Die Benutzeroberfläche ist untergliedert in verschiedene Bereiche, welche das Erstellen einer Umgebung erleichtern.

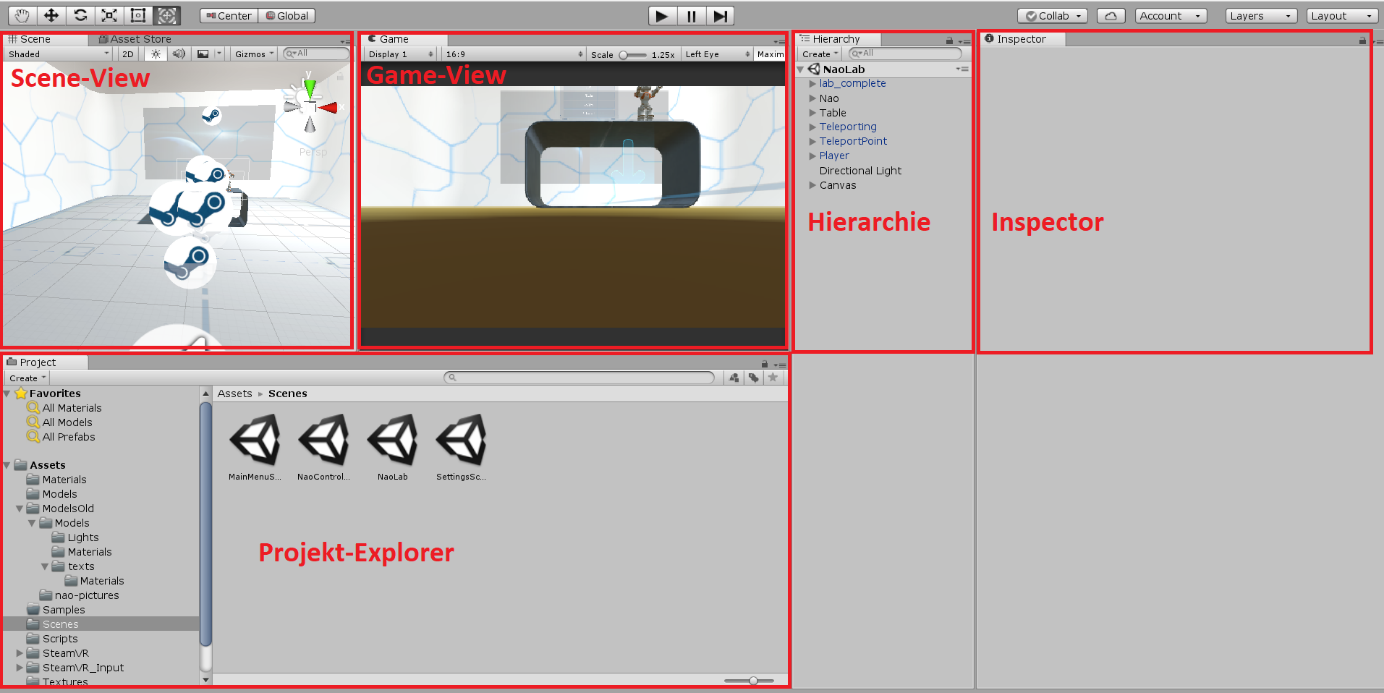


Abbildung 5: Benutzeroberfläche Unity

Mit Hilfe der Toolbar kann der Anwender verschiedene Aktionen in der Szene ausführen, beispielsweise kann das Bild geschwenkt oder skaliert werden. Die Scene-View stellt die designte Szene dar. Durch das Betätigen des Play-Knopfes in der Toolbar wird das Spiel gestartet und in der Game-View angezeigt. Die Game-View ist der sogenannte Spielmodus, mit welchem das konzipierte Spiel getestet werden kann. Hier kann sowohl ein 2D-Spiel direkt über die Anwendung oder ein 3D-Spiel nach dem Verbinden einer VR-Brille getestet werden. Der Projekt-Explorer stellt eine Übersicht der Projektdateien dar und ermöglicht es, neue Dateien wie beispielsweise ein mit Maya erstelltes 3D-Modell per Drag-and-Drop einzufügen. Die Hierarchie hingegen zeigt alle Spielobjekte der dargestellten Szene an und ermöglicht es, neue Game-Objekte hinzuzufügen. In dem Inspector werden die Eigenschaften des ausgewählten Hierarchie-Objektes angezeigt, so dass diese nach den Wünschen des Anwenders angepasst werden können. [[20]](#footnote-21)

### Grundlegende Building-Blöcke

Unity hat drei verschiedene grundlegende Building-Blöcke, die zum Aufbau von Spieleumgebungen verwendet werden können. Die nachfolgende Tabelle erklärt die verschiedenen Arten von Blöcken.

|  |  |
| --- | --- |
| GameObjects | GameObjects sind fundamentale Objekte in Unity. Sie umfassen alle Arten von Objekten in dem erstellten Spiel, wie beispielsweise Lichtquellen, Requisiten oder Spezialeffekte. Sie fungieren dabei lediglich als Container für Komponenten und verfügen noch über keine Funktionalität. Beispiel: Point Light |
| Komponenten | Komponenten können zu Spiel-Objekten hinzugefügt werden. Sie ermöglichen es, dass das Verhalten der verknüpften GameObjects gesteuert werden kann.  Beispiel: Eigenschaften des Point Lights |
| Variablen | Variablen sind Eigenschaften, die von den Entwicklern von Spieleumgebungen frei nach ihren Bedürfnissen angepasst werden können. Sie werden im Inspector-Fenster angezeigt und können dort editiert werden.  Beispiel: Farbe, Position, Intensität des Point Lights |

Tabelle 1: Building-Blöcke in Unity [[21]](#footnote-22)

### SteamVR-Plugin

Unity verfügt außerdem über die Möglichkeit, Plugins zu installieren. Zur Erstellung virtueller Umgebungen ist das Virtual-Reality-System SteamVR notwendig. Das Plugin wurde von dem Softwareunternehmen Valve entwickelt und erlaubt es Entwicklern, die Game-Engine Unity mit dem System SteamVR zu koppeln. Das Plugin ermöglicht es, virtuelle Umgebungen unter Verwendung von VR-Hardware zu erleben. Die Hauptfeatures von SteamVR sind das Laden von Modellen der VR-Controllern, Input-Handling der Controller und die Abschätzung der Bewegung der Hände des Operators bei der Benutzung der Controller.[[22]](#footnote-23)

### Maya

Maya ist ein Softwareprodukt der Firma Alias, welche am 10. Januar 2006 von Autodesk übernommen wurde. Maya wird für die 3D-Visualisierung und Animation von der Film- und Fernsehindustrie sowie von Computerspiel Herstellern eingesetzt. Das Erstellen von 3D-Modellen wird auch in der Industrie, Architektur und Forschung eingesetzt. Maya zählt zu den bekanntesten Softwareprodukten aus dem Bereich 3D-Modellierung, Computeranimation und Rendering.

Die Software bietet unter anderem folgenden Funktionsumfang:

* Erweiterung von Maya durch die Interne Steuerungssprache MEL (Maya Embedded Language). MEL ist eine Skriptsprache und ermöglicht neben der Automatisierung bestimmter Aufgaben und der Umgestaltung des Editors weitere Anpassungen. Durch MEL wird die gesamte GUI gesteuert, eine Anpassung dieser über die C++ API ist nicht möglich. Beide überschneiden sich nicht, sondern ergänzen sich gegenseitig in ihrem Funktionsumfang.
* Neben MEL wird seit Version 8.5 auch die Programmiersprache Python unterstützt. Maya liefert hierzu einen eigenen Interpreter mit. Durch die Skriptsprache Python können Plugins und Skripte zur Funktionserweiterung von Maya entwickelt werden. Durch die Verwendung von Python ist es gestattet, weitere Bibliotheken einzubinden und dadurch auf weitere Funktionen zuzugreifen. Dadurch kann Python als Schnittstelle zu anderen operativen Systemen genutzt werden.
* Der modulare Aufbau von Maya ermöglicht die Integrität und freie Wahl vielzähliger Funktionen. Durch das Modul Maya Fur lassen sich zum Beispiel realistisch aussehendes Fell, Haarflächen oder Gras darstellen. Die Simulation von Flüssigkeiten oder Gasen kann über das Modul Maya Fluids bewerkstelligt werden. Mit Maya Fluids lässt sich allerdings kein Wasser simulieren. Ein weiteres von Modul ist z.B. Maya Cloth, mit dem Kleidungsstücke und Stoff simuliert werden kann.
* Das Rendering in Maya kann durch die Wahl verschiedener, implementierter Renderer angepasst werden. Der native Renderer von Maya heißt Maya Software und ist für die Berechnung aller Objekte zuständig. Maya liefert damit einen qualitativ hochwertigen Renderer, der vergleichsweise langsamer arbeitet und nicht immer physikalisch korrekt ist. Des Weiteren gibt es einen hardwarebasierten Renderer: Maya Hardware. Er ermöglich die Einbindung der 3D-Grafikkarte in den Rendering Prozess. Die Berechnung erheblich schneller, wird jedoch von Hardware­limitierungen eingeschränkt, wodurch z.B. die Texturgröße begrenz wird. Der aus einem deutschen Entwicklerstudio stammende Renderer Mental Ray wird ebenfalls unterstützt. Dieser ermöglicht die annähernd physikalische Darstellung von Beleuchtung, Tiefen- und Bewegungsunschärfe und die Verwendung von Ray Tracing. Ein weiterer Renderer unterstützt Vektor Rendering, der meistens für das Erstellen von Webanimationen mit Flash genutzt wird. Seit 2017 ist der Renderer Arnold in Maya integriert und liefert bessere Ergebnisse für Ray Tracing, Fur-Rendering sowie Bewegungsunschärfe und Volumenrendering.

Es gibt einige bekannte Filme, die mithilfe von Maya erstellt wurden. Diese sind unter anderem Findet Nemo, Shrek. Maya wird auch zur Berechnung von Fantasy Figuren in realen Filmaufnahmen verwendet, wie z.B. das Geschöpf Gollum bei Herr der Ringe.

# Entwurf

Dieses Kapitel dokumentiert die Vorgehensweise für die Entwicklung der Lösung und verschiedene Konzepte. Zu den Konzepten gehören unter anderem die Softwarearchitektur und Designentwürfe des virtuellen Raums.

## Theoretische Vorgehensweise

### Anforderungsanalyse

Wie in der Einführung (siehe 1.1) dargestellt, ist das Ziel die Schaffung einer multimodalen Telepräsenz zwischen einem menschlichen Benutzer und einem entfernten Roboter. Dabei soll eine Virtuelle Umgebung geschaffen werden, in welcher der Benutzer einerseits Steuerungseingaben an den Roboter senden und andererseits Bild- und Toninformationen vom entfernten Roboter erhält. Über die eine Virtual Reality Brille soll der Benutzer in die Umgebung eintauchen können und darüber interagieren können.

Die Virtuelle Umgebung soll folgende Features bereitstellen:

* Die Verbindung zu einem beliebigen Nao aufbauen
* Der Nao soll über den Virtuellen Raum angesteuert werden können, um Aktionen wie Hinsetzen, Aufstehen oder Laufen zu ermöglichen.
* Das Kamerasignal des Nao soll im Virtuellen Raum angezeigt werden. Dadurch kann der Benutzer durch die Umgebung navigieren, in der sich der Nao befindet.
* Das Tonsignal des Nao soll in den Virtuellen Raum übertragen werden
* Die Sprachausgabe auf dem Nao soll über das Eingeben von einfachen Sätzen im Virtuellen Raum ermöglicht werden.

### Geplantes Vorgehen

Die Anforderungen werden mit dem Team besprochen, um Unklarheiten frühzeitig zu beseitigen. Für die Entwicklung wird ein Projektplan entworfen und wichtige Meilensteile festgelegt.

Für den frühzeitigen Entwurf der Anwendung werden Designentwürfe (Mockups) entwickelt, um die unterschiedlichen Vorstellungen im Team besser visualisieren zu können. Diese können einfacher diskutiert werden und sollen Missverständnissen vorbeugen. Anhand der Mockups wird ein Modell für den Virtuellen Raum konzipiert, der die Steuerung für den Roboter enthält. (Telepräsenz?) Die gewählte Architektur soll die Anwendung in Komponenten untergliedern, so dass diese unabhängig voneinander verändert und angepasst werden können. Gleichzeitig ermöglicht die Architektur eine bessere Übersicht der Anwendung. Die zum Planungszeitpunkt feststehenden Funktionen werden in Komponenten eingeteilt. Darüber hinaus werden Schnittstellen zwischen den Komponenten festgelegt, so dass diese durch Erweiterungen oder Rückschläge nachträglich verändert werden können. Die Aufgaben aus dem Projektplan werden in kleinere Aufgaben zerlegt, so dass diese den Komponenten zugeordnet werden können. Jeder Entwicklungsstand wird in ein Git-Repository hochgeladen, so dass der Projektfortschritt bei jedem Teammitglied immer auf dem aktuellen Stand ist. Dies soll die Parallelentwicklung fördern und dem Datenverlust vorbeugen. Durch regelmäßige Teammeetings soll über den aktuellen Entwicklungsfortschritt kommuniziert werden. Jedes Teammitglied sollte über die aktuellen Entwicklungen informiert werden. Durch die Treffen können auch Ratschläge und Verbesserungen durch das Team eingeholt werden. Dokumentation?

## Konzeption

### Aufbau der Anwendung

Die Anwendung wurde nach dem 5-Schichtenmodell (siehe Kapitel x.x) konzipiert. In der nachfolgenden Abbildung ist der Architektur-Entwurf zu sehen.

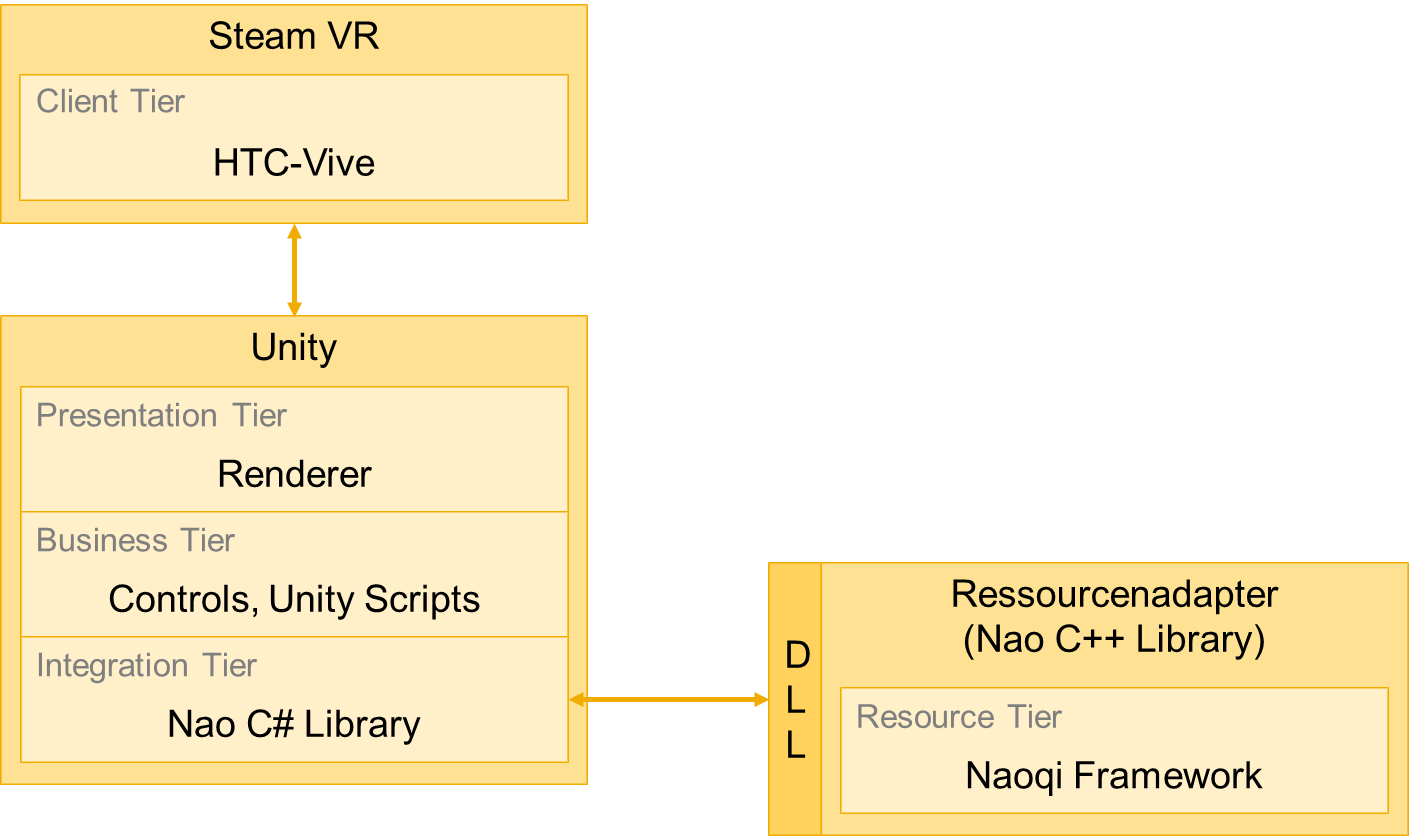


Abbildung 6: Architektur-Entwurf

Resource Tier

Die Ressourceschicht (Resource Tier) ist über einen Ressourcenadapter für das Naoqi Framework integriert. Dieser Adapter realisiert eine DLL-Schnittstelle, über welche Aktionen auf dem Nao gestartet werden können. Diese sind u.a. Laufen, Gehen oder Sprechen. Über den Adapter wird auch auf die Kamera und das Mikrofon des Nao zugegriffen.

Integration Tier

Die Integrationsschicht (Integration Tier) ist für die Kommunikation mit dem Ressourcenadapter zuständig. Dabei soll die Einbindung des Frameworks in Unity sichergestellt werden. Funktionen in dieser Schicht rufen Routinen in der DLL auf und fungieren als Erweiterung von Unity.

Business Tier

Die Businesslogik (Business Tier) übernimmt die Datenlogik und Steuerung in Unity. Speziell entwickelte Skripte in dieser Schicht ermöglichen die Interaktion des Benutzers mit der in der Integrationsschicht implementierten Erweiterung von Unity.

Presentation Tier

Die Präsentationsschicht (Presentation Tier) wird über Unity realisiert und enthält sowohl sämtliche Objekte zur Generierung des Virtuellen Raumes als auch Elemente für die Grafische Benutzeroberfläche (GUI).

Client Tier

Die Clientschicht (Client Tier) übernimmt die Darstellung der in der Präsentationschicht generierten Oberfläche und wird über die HTC-Vive Brille realisiert. Die Verbindung übernimmt das Steam-VR Framework.

### Entwurf der virtuellen Umgebung

Für das Design des Virtuellen Raumes wurden verschiedene Entwürfe entwickelt. Diese werden im folgenden Abschnitt genauer vorgestellt.

**Head-Display**

Bei diesem Entwurf wird der virtuelle Raum so dargestellt, als befinde sich der Benutzer im Kopf des Nao und würde durch seine Augen sehen. Links unten ist ein Abbild des Nao, das Informationen über die einzelnen Gelenkpositionen darstellen soll. Der Benutzer soll dadurch erkennen können, in welcher Lage sich der Nao befindet und welche Aktionen aktuell ausgeführt werden. Rechts oben ist ein Menü für die Steuerung zu sehen. Durch dieses Menü soll der Nao bewegt werden können.

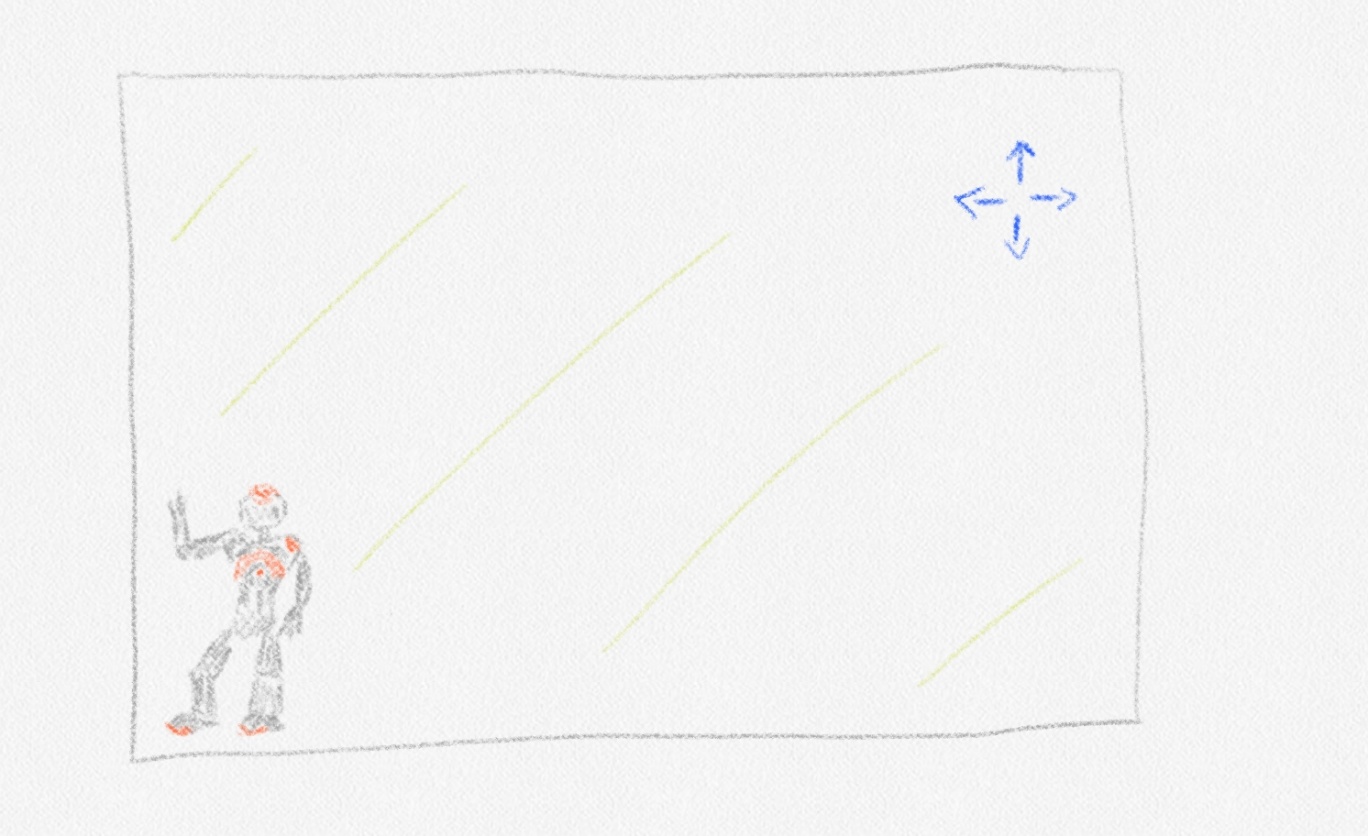


Abbildung 7: Entwurf Head-Display

**Nao-Labor**

Bei diesem Entwurf wurde ein Labor entwickelt, in welchem sich der Benutzer aufhält, während er den Nao steuert. In der Mitte des Raums befindet sich ein Tisch. Über ein Menü auf dem Tisch soll der Benutzer Steuerungseingaben an den Nao senden können, um diesen zu bewegen. Neben dem Menü steht ein Modell des Nao, welches ein gegenwärtiges Abbild des Roboters darstellt. Dieses soll dem Benutzer ermöglichen, die aktuell ausgeführte Aktion und Lage des Roboters zu erkennen. An der Wand hinter dem Tisch befindet sich ein Bildschirm, auf dem das aktuelle Kamerabild des Nao-Roboters dargestellt werden. Der Benutzer kann direkt sehen, was der Nao aufzeichnet. Im virtuellen Raum kann sich der Benutzer durch Teleportation über die Steuerung der HTC Vive bewegen.

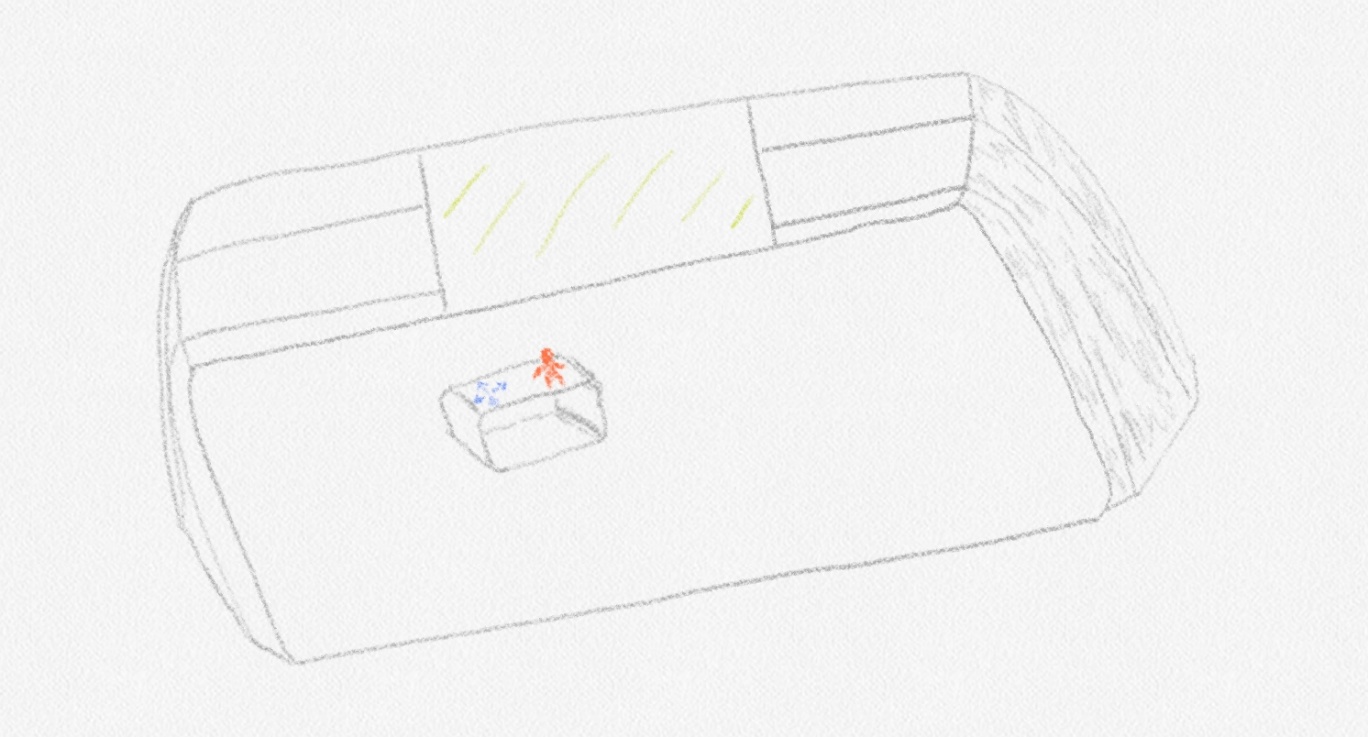


Abbildung 8: Entwurf Labor

**Willkommensbildschirm (Hauptmenü)**

In diesem Entwurf ist ein Hauptmenü auf einer einzelnen Plattform zu sehen. Der virtuelle Raum soll einen Ort zwischen den Welten darstellen, an welchen der Benutzer gebracht wird, wenn er die Anwendung startet. Zu diesem Zeitpunkt wurde noch keine Verbindung mit einem Nao-Roboter hergestellt. Das Menü soll dem Benutzer einen übersichtlicheren Einstieg ermöglichen, da Verbindungsparameter wie z.B. die IP-Adresse eingegeben werden können. Dadurch kann sich der Benutzer mit verschiedenen Nao-Robotern verbinden.

Über einen Menüeintrag wird eine Verbindung zum Nao erstellt und der Benutzer wird in einen weiteren virtuellen Raum für die Fernsteuerung des Roboters teleportiert.

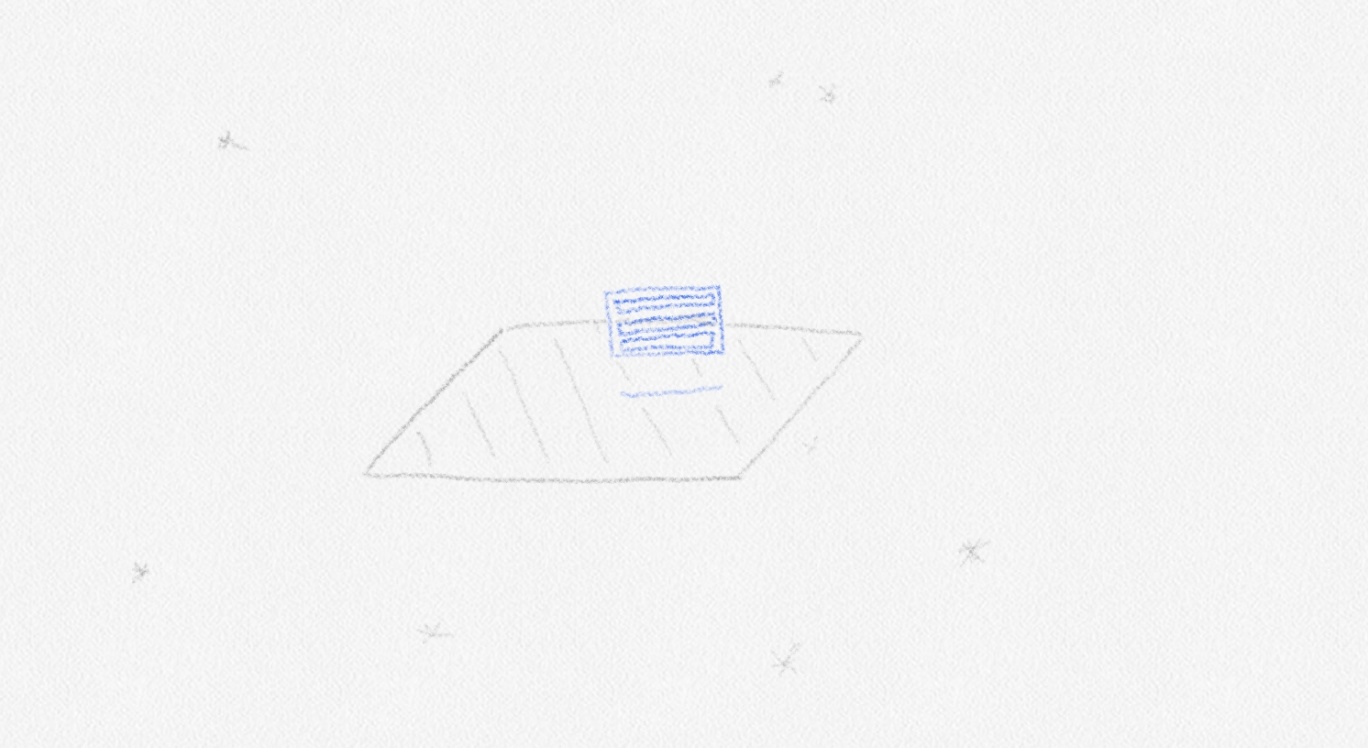


Abbildung 9: Entwurf Willkommensbildschirm

# Implementierung

## Erstellung der virtuellen Umgebung

### Design der 3D-Objekte

Für die virtuellen Räume wurden dreidimensionale Objekte entworfen. Die Räume wurden mithilfe dieser Objekte zusammengesetzt. Dabei wurde gezielt ein futuristisches Design gewählt und die Objekte mit Texturen eingefärbt. Die erstellten Texturen sind fortlaufend und können ohne Unterbrechung nebeneinandergelegt werden. Texturen, die diese Eigenschaft aufweisen, werden auch *seamless* oder *tileable* genannt. Die Texturen sind durch verschiedene Science-Fiction Filme wie zum Beispiel Starwars, Star Trek beziehungsweise Computerspiele wie ELEX inspiriert.

**Labor**

Das Labor wurde aus mehreren Objekten zusammengesetzt. Diese lassen sich konkret in Wand-, Boden- und Deckenstücke, sowie Dekorobjekte unterteilen.

* Zu den Wandstücken gehören gerade Wände und Wände mit 90°-Ecken. Diese können durch Rotation einen viereckigen Raum abbilden. Die blauen Linien auf der Textur (siehe Abbildung xx.xx), gehen an den Rändern nahtlos ineinander über.
* Zu den Bodenstücken gehören drei Objekte: Mittel-, Rand- und Eckstücke. Die Randstücke besitzen einen durchgehen Streifen, der die Randstellen markiert. Dieser grenzt an den unteren Teil der Wandstücke und soll den Übergang hervorheben.
* Das Deckenstück besteht aus einem einzigen Objekt und besitzt keine spezielle Textur. Das Element ist daher komplett weiß. Wie bei den Wänden wurden die Seitenkanten nach innen gebogen, um die entstehenden Ecken abzurunden.
* Die Dekorobjekte bestehen aus einem abgerundeten Tisch, einem Menü und einem Modell des Nao-Roboters. Bis auf das Menü wurden alle Objekte aus bestehenden Objekten zusammengesetzt. Der Tisch ist ein Standardobjekt aus SteamVR. Der Roboter stammt von der Herstellerfirma des Nao, Aldebaran.

Die Objekte für die Wände, den Boden und die Decke wurden mit Maya erstellt. Das Ausgangsobjekt war für alle Objekte ein elementares Quadrat. Im Designprozess wurden weitere Punkte eingeführt und Kanten geschnitten. Durch Rotationen und Transformationen wurden die Objekte in ihre Form gebracht. Zuletzt wurde eine Textur aufgetragen und auf die Punkte des dreidimensionalen Körpers gelegt. Dieser Prozess nennt sich Texture-Mapping. Im Folgenden ist der gesamte Designprozess in Maya für das Wand-Eckstück beschrieben und wird in Abbildung 10: Design-Prozess in Maya dargestellt.

1. Mithilfe der Polygonfunktion wird eine quadratische Fläche erstellt. Anschließend wird die Größe angepasst und das Objekt zu einer Wand rotiert.
2. In das Objekt werden Mittelpunkte der oberen und unteren Kante eingefügt. Das Objekt wird in der Mitte „geschnitten“. Es entstehen zwei zusammenhängende, rechteckige Hälften.
3. Eine Hälfte des Objektes wird um 90° gedreht, so dass eine Ecke entsteht. Dafür werden die Eckpunkte an die richtige Stelle verschoben.
4. Es folgen weitere Anpassungen, so dass das Eckstück an eine gerade Wand gesetzt werden kann. Um die Übergänge zur Decke und zum Boden etwas abzurunden, werden die oberen und unteren Kanten zur Innenseite gebogen. Des Weiteren wird die Innenkante durch das Einfügen von zwei weiteren Kanten abgerundet. Es entstehen zwei abgerundete Segmente in der Mitte.
5. Die Wandtextur wird auf das Objekt gelegt und an die Punkte des Körpers geheftet. Hierbei muss beachtet werden, dass die Regelmäßigkeit der Textur erhalten bleibt, da die Wandstücke beim Aneinanderheften keine Unterbrechungen aufweisen sollen.
6. Abschließender Test, ob das Eckstück auf ein gerades Wandstück passt. Zu sehen ist ebenfalls, dass der Boden an das Wandstück passt.

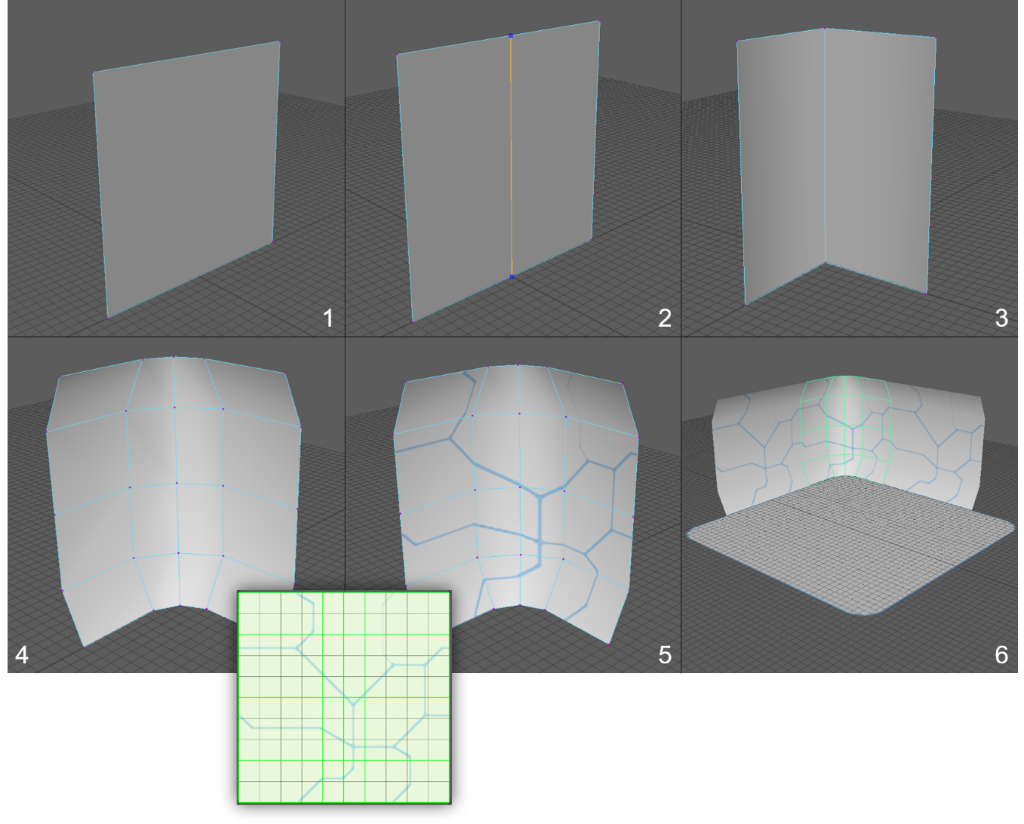


Abbildung 10: Design-Prozess in Maya

### Aufbau der virtuellen Umgebung

Um es dem Anwender zu ermöglichen, in die Welt des Roboters einzutauchen, wurde im Rahmen dieser Studienarbeit eine virtuelle Umgebung unter Verwendung der Game-Engine Unity erstellt. Die virtuelle Umgebung besteht aus zwei Szenen, welche in dem Kapitel Konzeption grundlegend beschrieben werden. Wie in Kapitel xx.xx erläutert, muss das Plugin SteamVR in die Anwendung integriert werden, damit statt zweidimensionaler Umgebungen auch dreidimensionale virtuelle Umgebungen dargestellt werden können. Dadurch kann die Anwendung über Unity gebaut und gestartet werden und direkt über eine VR-Brille visualisiert werden. Das Plugin verfügt über eine Render-Komponente, welche das Rendern des Signals der VR-Kamera übernimmt. Eine zweite wichtige Komponente für das Erstellen virtueller Umgebungen in Unity ist dasCameraRig*.* Sie übernimmt die Steuerung des Headsets und der Controller der HTC Vive. Über Unity kann die Position, an der sich der Operator zum Start der Anwendung befindet, definiert werden. Der Anwender kann seinen Kopf bewegen und die Position wird über das Signal der HTC Vive Pro erfasst und das Bild entsprechend der Position und Blickrichtung gerendert.

### Aufbau der erstellten Szenen

In den beiden erstellten Szenen hat der Anwender die Möglichkeit, sich frei im Raum zu teleportieren. Dies wurde realisiert über die Einbindung des von Unity zur Verfügung gestellten Teleport-Prefabs. Um sich anschließend mit Hilfe des Prefabs frei in der Umgebung bewegen zu können, müssen zusätzlich Teleport-Punkte in der Szene definiert werden. Diese definieren für das System, an welche Stellen des Raumes der Anwender sich teleportieren kann und bieten somit den Entwicklern die Möglichkeit, die Anwendung passend zu den Anforderungen zu gestalten. Weiterhin ist es für das Spielerlebnis von großer Bedeutung, dass der Operator durch die Controller Aktionen ausführen kann und somit auch die Möglichkeit der Teleportation nutzen kann. Der Input der VR-Controller muss in der Unity-Anwendung getrackt werden können, damit die Position der Hände wahrheitsgetreu in der virtuellen Umgebung angezeigt und die Benutzereingabe über die Controller erkannt und umgesetzt werden kann. Sobald die Controller von dem SteamVR-Plugin erkannt werden, werden virtuelle Versionen der Controller in Unity erstellt und zu den bereits vorhandenen Controllern in CameraRig gemapped.

Willkommensbildschirm (Hauptmenü)

Sobald der Operator die Anwendung startet, befindet er sich in der ersten Szene. Der Anwender steht auf einer Plattform im Weltall, welches als eine Art Zwischenwelt fungiert. Zu diesem Zeitpunkt ist er zwar noch nicht mit dem NAO-Roboter verbunden, jedoch bereits rundum von virtuell erstellten Bildern umgeben. In der nachfolgenden Abbildung sieht man den Aufbau des Willkommensbildschirms. In der Mitte dieser Szene befindet sich ein Menü, durch das es dem Anwender ermöglicht wird durch die virtuelle Umgebung zu navigieren. Außerdem kann der Benutzer bevor er die Kontrolle des NAO-Roboters in dem Lab übernimmt, Voreinstellungen wie beispielsweise die Angabe der IP-Adresse des Roboters treffen. Der Willkommensbildschirm hat die Funktion, den Anwender eine Einweisung über die Anwendung zu geben und verschiedene Konfigurationsmöglichkeiten anzubieten.

BILD WILLKOMMENSBILDSCHIRM

Realisiert wurde die Umgebung der ersten Szene durch das Verwenden einer Skybox von Unity. Skyboxen sind Umgebungs-Beleuchtungs-Effekte, die über die gesamte Peripherie der Szene gelegt werden können. Somit kann der Eindruck erweckt werden, dass der Anwender sich in der virtuellen Umgebung befindet. Beim Bewegen des Head-Mounted-Displays werden die Bilder gerendert und entsprechend der Position und Blickrichtung angezeigt. [[23]](#footnote-24) Die Panoramasicht wird bei einer Skybox in die sechs Richtungen der Achsen oben, unten, links, rechts, vorwärts und rückwärts geteilt. Die Bilder sollen nahtlos ineinander übergehen und dem Anwender eine kontinuierliche Umgebung anzeigen, die aus jeder Perspektive dargestellt werden kann. Durch Verwenden dieses Effekts, kann Realismus in die Szene mit minimaler Belastung der Grafikhardware hinzugefügt werden. [[24]](#footnote-25)

Es gibt verschiedene Möglichkeiten, Skyboxen zu erstellen. Beispielsweise kann ein walzenförmiges Panorama benutzt werden, das jedoch lediglich die horizontalen Achsen darstellt und somit nicht für unseren Einsatzzweck geeignet war. Die beste Möglichkeit war für uns der Einsatz einer Cubemap. Die Abbildung xx.xx zeigt den Aufbau der erstellten Cubemap.

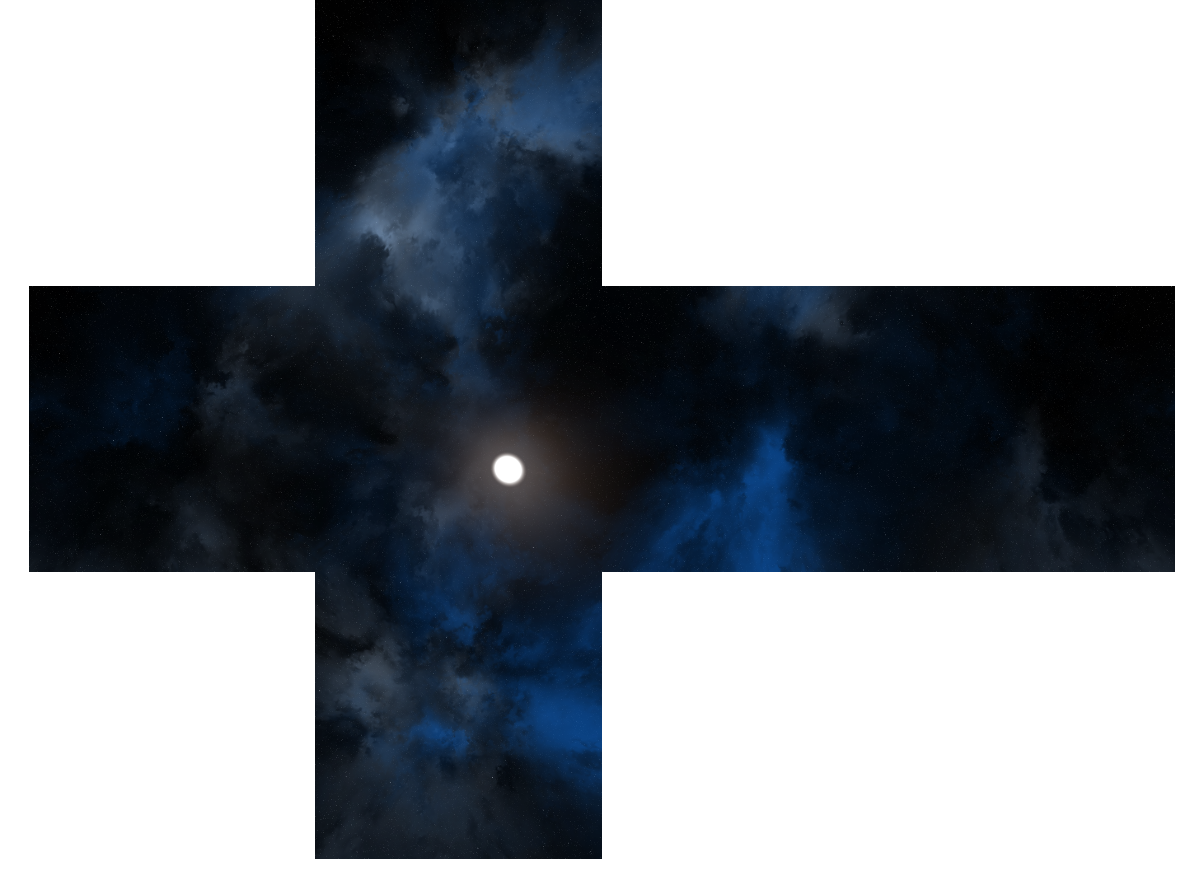


Abbildung 11: Cubemap Skybox

Eine Cubemap besteht aus sechs Bildern, die als Seiten eines Würfels angeordnet werden. Unity rendert diese Bilder während der Laufzeit als Skybox. Die Cubemap ist eine sehr gute Variante einer Skybox, da sie einen hohen Detailgrad hat und sowohl die horizontale als auch die vertikale Ebene vollständig darstellt. [[25]](#footnote-26)

Nao-Lab

Nachdem der Anwender die IP-Adresse des NAO-Roboters in der Willkommensszene der konzipierten Umgebung definiert hat, gelangt er in die zweite Szene. Diese stellt ein Labor dar und dient als Basis der Fernsteuerung des Roboters. Der Anwender ist nun mit dem Roboter verbunden und kann ihn als Teleroboter verwenden und ihn aus der entfernten Umgebung steuern. In der erstellten Szene ist es nicht nur möglich über einen Kanal den Roboter zu steuern, vielmehr wird eine multimodale Kommunikation zwischen Mensch und Roboter ermöglicht. Über ein Menü in der Mitte des Raumes kann der Operator bestimmen, welche Aktionen der Teleroboter ausführen soll. Der Aufbau des Menüs ist äquivalent zu dem Menü der Willkommensszene. Da dem Anwender direkt angezeigt werden soll, was der Roboter macht, wird das Kamerabild des Roboters übertragen und als Display in einer Wand dargestellt.

BILD NAO-LAB

Das Hauptmenü verfügt über verschiedene Auswahlmöglichkeiten (siehe Abbildung xx.xx):

* Posture: Nach Betätigen dieses Buttons, wird das Posture-Untermenü anstelle des Hauptmenüs angezeigt. Über dieses Menü können verschiedene Posen durch den Anwender ausgewählt werden. Somit kann das Verhalten des Roboters gesteuert werden und die Aktion eingeleitet werden (Stehen, Liegen, Sitzen).
* Talk: Der zweite Menüpunkt ist „Talk“. Hierbei sind bereits Standard-Sätze im Skript hinterlegt, die zufällig ausgesucht und vom Nao nachgesprochen werden.
* Walk: Zu dem Menüpunkt „Walk“ sind wiederum verschiedene Möglichkeiten in einem Untermenü hinterlegt. Der Anwender kann auswählen, in welche Richtung der Roboter sich bewegen soll.
* Wave: Der Roboter winkt dem Anwender zu, wenn diese Option ausgewählt wird.

BILD NAO-Hauptmenü

### Ordnerstruktur

Die nachfolgende Abbildung stellt die Ordnerstruktur des erstellten Unity-Projektes dar. Die übersichtliche Struktur ermöglicht ein effizientes Vorgehen und dient der besseren Orientierung. Die Bausteine der Anwendung konnten dadurch in verschiedene Kategorien strukturiert werden.

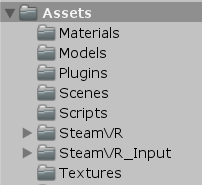


Abbildung 12: Ordnerstruktur Unity-Projekt

Die nachfolgende Tabelle erklärt die Funktionalität der angelegten Ordner.

|  |  |
| --- | --- |
| **Materials** | In diesem Unterordner sind alle erforderlichen Materialien gespeichert. Materialien können beispielsweise auf Farben oder Texturen gelegt werden. |
| **Models** | Der Ordner enthält die mit Maya erstellten 3D-Objekte. |
| **Plugins** | Hier sind alle erforderlichen Plugins gespeichert. Zum Beispiel befindet sich hier die erstellte Bibliothek, die die Kommunikation mit dem NAO-Roboter ermöglicht. |
| **Scenes** | In dem Ordner Scenes sind alle erstellten Szenen der Anwendung gespeichert. |
| **Scripts** | Dieser Ordner enthält alle implementierten Skripte. |
| **SteamVR** | Der Ordner wird automatisch von dem Plugin SteamVR erstellt. Die dort enthaltenen Objekte sind für das Erstellen dreidimensionaler Umgebungen erforderlich. |
| **SteamVR\_Input** | Dieser Ordner wird automatisch von SteamVR hinzugefügt, damit die Steuerung der Controller getrackt werden kann. |
| **Textures** | Texturen können beispielsweise auf Materialien gelegt werden. |

Tabelle 2: Ordnerstruktur Unity-Projekt

## Kommunikation mit dem NAO-Roboter

Der wichtigste Aspekt dieser Studienarbeit war die Ermöglichung der Kommunikation mit einem NAO-Roboter. Die Kommunikation wurde umgesetzt über die Implementierung einer C++-Projekt. Über diese Klasse findet die Anbindung des Roboters sowie die Steuerung statt. Damit auf die Funktionen des NAOqi-Frameworks zugegriffen werden kann, muss die NAOqi-Software Development Kit in das Projekt eingebunden werden. Dieses Kapitel beschreibt die implementierten Funktionalitäten des C++-Projektes. Das C++-Projekt dient als Grundlage für die Anbindung des Roboters in der virtuellen Umgebung. Die Klasse wurde als Klassenbibliothek (.dll) in Unity eingebunden und stellt so die implementierten Funktionalitäten zur Verfügung. Im nächsten Kapitel wird beschrieben, wie auf die implementierten Methoden in Unity zugegriffen werden kann.

Das NAOqi-Framework beinhaltet verschiedene Kernmodule, auf die bei der Implementierung einer Anwendung zur Steuerung des Roboters zugegriffen werden kann. Nachfolgend eine Übersicht über die Programmierschnittstellen der NAOqi, auf welche in dem C++-Projekt zur Umsetzung der Anforderungen der Anwendung zugegriffen wurde:

|  |  |
| --- | --- |
| **NAOqi Module** | **Beschreibung** |
| ALRobotPosture | Das Modul enthält diverse Methoden, um den Roboter vordefinierte Posen einnehmen zu lassen. Dafür bestimmt der Roboter zunächst seine eigene Pose und wendet anschließend die definierte Pose des Anwenders an. Jede Pose ist spezifiziert durch eine eindeutige Gelenk- und Sensorkonfiguration.  Der Aufruf erfolgt über folgende Methode: [*ALRobotPostureProxy::goToPosture*](http://doc.aldebaran.com/2-5/naoqi/motion/alrobotposture-api.html#ALRobotPostureProxy::goToPosture__ssC.floatC)  Als Parameter kann die gewünschte Pose des Roboters übergeben werden („Crouch“, „LyingBack“, „StandInit“, etc.).[[26]](#footnote-27) |
| ALMotion | Das Modul ALMotion definiert Methoden, die die Steuerung der Bewegungen des Roboters vereinfachen. Beispielsweise kann die Position der Gelenke, die Steuerung von Bewegungen oder die Vermeidung von Kollisionen reguliert werden. [[27]](#footnote-28) |
| ALVideoDevice | Dieses Modul ist für die Kameraübertragung des Roboters zuständig. Eingesetzt kann das Modul beispielsweise für die Gesichtserkennung. Bei der Erstellung einer Proxy, um Zugriff auf die Methoden zu bekommen, können verschiedene Parameter definiert werden (Ausflösung, Farbraum, etc.).  Wichtige Methoden:   * *ALVideoDeviceProxy::subscribeCamera*:  Registrieren eines Videomoduls zu ALVideoDevice. Die Rohdaten des Bildes werden in dem definierten Format gespeichert. * *ALVideoDeviceProxy::getImageRemote:* Abrufen des letzten Bildes des Roboters. * *ALVideoDeviceProxy::releaseImage:* Freigabe des Bild-Puffers, welcher vom subscribeCamera blockiert wurde.[[28]](#footnote-29) * *ALVideoDeviceProxy::unsubscribe:*   Abmeldung des angemeldeten Video-Moduls von ALVideoDevice.[[29]](#footnote-30) |
| ALTextToSpeech | Das Modul ermöglicht es dem Roboter, zu sprechen. Der Benutzer kann Strings definieren, die als Kommandos an den NAO geschickt werden. Der Roboter spricht anschließend die Wörter nach. [[30]](#footnote-31)  In der C++-Klasse wurde folgende Methode eingesetzt:  *ALTextToSpeechProxy::say*  Über den Methodenaufruf kann ein String spezifiziert werden, der von dem Roboter gesagt werden soll. |

Tabelle 3: Kernmodule des NAOqi-Frameworks

Über die in der Tabelle dargestellten NAOqi-Module erfolgt in der implementierten C++-Klasse die Steuerung des Roboters. Die Anbindung zu dem Roboter wird jeweils ermöglicht, durch die Erstellung eines Proxys des jeweiligen Moduls. Dabei müssen lediglich die IP-Adresse und die Portnummer des zur Verfügung stehenden NAO-Roboters angegeben werden. Durch diese Angabe kann der Roboter eindeutig identifiziert werden und an das Programm angebunden werden.

Methoden der C++-Klasse erklären (Codebeispiel)

## Steuerung des NAO-Roboters

C# wrapper

Auf Choreographe eingehen – Box-Bibliotheken dienten lediglich als Grundlage für die Programmierung (Python-Script -> C#-Script)

## Umsetzung der Telepräsenz

Unity-spezifisch

# Ergebnis

# Zusammenfassung und Ausblick

## Persönliches Feedback

## Erweiterungsmöglichkeiten

# Literaturverzeichnis

1. Siehe https://www.neimagazine.com/features/featurehot-cell-robot-4483658/featurehot-cell-robot-4483658-462777.html [Letzter Zugriff: 12.01.2019] [↑](#footnote-ref-2)
2. Vgl. Bruder, Jan „Praktische Realisierung einer haptischen Telerobotik-Steuerung für eine interaktive Nutzung“ [2009] [↑](#footnote-ref-3)
3. Vgl. https://www.dlr.de/desktopdefault.aspx/tabid-3228/5011\_read-26483/5011\_page-2/ [↑](#footnote-ref-4)
4. Vgl. https://www.zukunftsinstitut.de/artikel/virtual-reality-die-erschaffung-neuer-welten/ [↑](#footnote-ref-5)
5. Vgl. https://wirtschaftslexikon.gabler.de/definition/virtuelle-realitaet-54243 [↑](#footnote-ref-6)
6. Vgl. https://www.zukunftsinstitut.de/artikel/virtual-reality-die-erschaffung-neuer-welten/ [↑](#footnote-ref-7)
7. Vgl. https://wirtschaftslexikon.gabler.de/definition/virtuelle-realitaet-54243 [↑](#footnote-ref-8)
8. Vgl. http://www.inztitut.de/blog/glossar/immersion/ [↑](#footnote-ref-9)
9. Siehe http://arvr3dmodelling.com/virtual-reality-vs-augmented-reality-vs-mixed-reality/ [↑](#footnote-ref-10)
10. Vgl. https://www.virtual-reality-magazin.de/themen/augmented-reality-vr [↑](#footnote-ref-11)
11. Vgl. https://www.elektronik-kompendium.de/sites/com/2210231.htm [↑](#footnote-ref-12)
12. Vgl. https://www.softbankrobotics.com/emea/en/nao [Letzter Zugriff: 14.04.2019] [↑](#footnote-ref-13)
13. Siehe https://www.generationrobots.com/de/401617-humanoider-roboter-nao-evolution-rot.html [↑](#footnote-ref-14)
14. Vgl. Seo, Kisung „Using NAO – Introduction to interactive humanoid robots“ [↑](#footnote-ref-15)
15. Vgl. ALDEBARAN DOCUMENTATION http://doc.aldebaran.com/2-4/software/choregraphe/choregraphe\_overview.html [Letzter Zugriff: 11.04.2019] [↑](#footnote-ref-16)
16. Siehe https://www.vive.com/de/product/vive-pro/ [↑](#footnote-ref-17)
17. Vgl. https://www.vive.com/de/product/vive-pro-starter-kit/?gclid=Cj0KCQjwyoHlBRCNARIsAFjKJ6Cu3-S1YzX5OfBx3iI5psCH8nb8suJPLiMYi\_mo6je9G4fVAgkNCWcaAvXvEALw\_wcB [↑](#footnote-ref-18)
18. Siehe https://unity3d.com/de/what-is-a-game-engine [↑](#footnote-ref-19)
19. Vgl. https://unity3d.com/de/what-is-a-game-engine [↑](#footnote-ref-20)
20. Vgl. https://msdn.microsoft.com/de-de/magazine/dn759441.aspx [Letzter Zugriff: 12.03.2019] [↑](#footnote-ref-21)
21. Vgl. https://unity3d.com/de/programming-in-unity [Letzter Zugriff: 06.04.2019] [↑](#footnote-ref-22)
22. Vgl. https://assetstore.unity.com/packages/tools/integration/steamvr-plugin-32647   
     [Letzter Zugriff: 06.04.2019] [↑](#footnote-ref-23)
23. Vgl. https://medium.com/aol-alpha/how-to-design-vr-skyboxes-d460e9eb5a75 [Letzter Zugriff: 31.03.2019] [↑](#footnote-ref-24)
24. Vgl. https://unity3d.com/de/learn/tutorials/topics/graphics/using-skyboxes [Letzter Zugriff: 31.03.2019] [↑](#footnote-ref-25)
25. Vgl. https://medium.com/aol-alpha/how-to-design-vr-skyboxes-d460e9eb5a75 [Letzter Zugriff: 31.03.2019] [↑](#footnote-ref-26)
26. Vgl. http://doc.aldebaran.com/2-5/naoqi/motion/alrobotposture.html#alrobotposture   
     [Letzter Zugriff: 16.04.2019] [↑](#footnote-ref-27)
27. Vgl. http://doc.aldebaran.com/2-5/naoqi/motion/almotion.html#almotion [Letzter Zugriff: 16.04.2019] [↑](#footnote-ref-28)
28. Vgl. http://doc.aldebaran.com/2-5/naoqi/vision/alvideodevice-api.html#alvideodevice-api   
     [Letzter Zugriff: 16.04.2019] [↑](#footnote-ref-29)
29. Vgl. http://doc.aldebaran.com/2-5/naoqi/vision/alvideodevice-api.html#alvideodevice-api   
     [Letzter Zugriff: 16.04.2019] [↑](#footnote-ref-30)
30. Vgl. http://doc.aldebaran.com/2-5/naoqi/audio/altexttospeech.html#altexttospeech   
     [Letzter Zugriff: 16.04.2019] [↑](#footnote-ref-31)