MASTERARBEIT

Entwicklung einer multifunktionalen Virtual-Reality-Plattform für virtuelle Meetings und KI-gestützte Interaktionen mit NPCs

zur Erlangung des akademischen Grades Master of Science (M.Sc.)

vorgelegt am 14. Juli 2025

Ehsan Haghshenas

Erstprüfer: Prof. Dr. Eike Langbehn

Zweitprüfer: Simon Dewert

HOCHSCHULE FÜR ANGEWANDTE
WISSENSCHAFTEN HAMBURG
Department Medientechnik
Finkenau 35
22081 Hamburg

Zusammenfassung

Das vorliegende Projekt mit dem Titel „Entwicklung einer multifunktionalen Virtual-Reality-Plattform für virtuelle Meetings und KI-basierte Interaktionen mit NPCs“ befasst sich mit der Konzeption und Implementierung eines neuartigen Systems zur virtuellen Kommunikation im Kontext immersiver VR-Umgebungen. Ziel dieser Plattform ist es, die Qualität virtueller Besprechungen zu verbessern, das Gefühl der Präsenz zu stärken und den Nutzer*innen eine intelligente sowie interaktive Erfahrung zu ermöglichen.

Im Rahmen dieses Projekts betreten die Anwender*innen mithilfe von Virtual-Reality-Headsets wie der Oculus Quest eine dreidimensionale Umgebung, in der sie sich gemeinsam in Gruppen treffen, miteinander sprechen, Nachrichten austauschen und über interaktive Avatare kommunizieren können. Ein zentrales Merkmal des Systems ist die Integration eines intelligenten virtuellen Charakters (NPC), der als KI-gestützter Assistent innerhalb der Sitzung fungiert. Dieser NPC übernimmt dabei die Rolle eines Moderators oder Beobachters, der mithilfe künstlicher Intelligenz in der Lage ist, sowohl fachliche als auch allgemeine Fragen der Teilnehmenden zu beantworten und so die Interaktivität der Besprechung aufrechtzuerhalten.

Zur Entwicklung dieser Plattform kamen verschiedene Technologien zum Einsatz, darunter **Unity**, **XR Interaction Toolkit**, **NetCode**, **Relay**, **Lobby**, **Vivox** für Sprachkommunikation sowie **Convai AI** zur Generierung intelligenter Verhaltensweisen für NPCs. Der Fokus des Projekts liegt auf der Entwicklung einer technischen Lösung, die sowohl zuverlässig als auch aus Nutzerperspektive attraktiv und effektiv ist.

Der primäre Antrieb zur Realisierung dieses Projekts liegt in den Herausforderungen großer Unternehmen – etwa aus der Industrie- oder Automobilbranche – im Zusammenhang mit der Organisation physischer Besprechungen. In solchen Unternehmen, insbesondere in Bereichen mit mehreren parallel arbeitenden Teams, treten häufig Überschneidungen bei der Nutzung und Buchung von Konferenzräumen auf, was die rechtzeitige Durchführung interaktiver Meetings erheblich erschwert. Die Problematik verschärft sich, wenn Teams während der Besprechung mit physischen Modellen oder spezifischen Werkzeugen interagieren müssen. In solchen Szenarien bietet die vorgeschlagene Virtual-Reality-Plattform eine interaktive und ortsunabhängige Alternative zu Präsenzsitzungen und schafft eine praktische, effektive und jederzeit zugängliche Lösung für alle Teammitglieder.

Im weiteren Verlauf dieser Arbeit werden die einzelnen Phasen der Konzeption, Entwicklung, auftretenden Herausforderungen und deren Lösungsansätze sowie die technische Struktur und die Ergebnisse der durchgeführten Usability-Tests der Plattform detailliert beschrieben.

Abstract

This thesis, titled "Development of a Multifunctional Virtual Reality Platform for Virtual Meetings and AI-Based Interactions with NPCs", explores the design and implementation of an innovative system for virtual communication in immersive VR environments. The platform aims to enhance the quality of online meetings, increase the sense of presence, and provide users with an intelligent and interactive experience.

In this project, users utilize VR headsets such as the Oculus Quest to enter a 3D environment where they can gather in groups, speak, send messages, and interact through avatars. A key feature of the system is the integration of a smart virtual character (NPC) who acts as an AI-powered assistant during the meeting. This NPC plays the role of a moderator or observer and is capable of answering both technical and general questions to maintain the interactive nature of the session.

The platform is developed using technologies such as **Unity**, **XR Interaction Toolkit**, **NetCode**, **Relay**, **Lobby**, **Vivox** for voice communication, and **Convai AI** for generating intelligent NPC behaviors. The main focus lies in creating a technically reliable and user-friendly solution that is both effective and engaging.

The motivation behind this project originates from the challenges faced by large corporations—particularly in industrial or automotive sectors—regarding the organization of in-person meetings. In such environments, especially where multiple teams operate simultaneously, conflicts in booking and using conference rooms often lead to delays and ineffective sessions. This becomes even more critical when direct interaction with models or physical tools is required. The proposed VR platform offers an interactive and location-independent alternative to physical meetings, enabling practical and accessible collaboration for all team members.

The subsequent chapters of this thesis provide a detailed discussion of the design process, development phases, encountered challenges, technical architecture, and the results of user testing conducted on the platform.

Inhaltsverzeichnis

Abbildungsverzeichnis

Danksagung

Die vorliegende Masterarbeit entstand nicht allein durch meine Bemühungen, sondern auch dank wertvoller Unterstützung verschiedener Personen, die mich während dieser spannenden und herausfordernden Reise begleitet haben.

Zunächst möchte ich mich herzlich bei meinem Erstprüfer, Herrn Prof. Dr. Eike Langbehn, bedanken. Seine konstruktiven und wertvollen Anregungen halfen mir dabei, meine anfänglich noch rohe Idee in eine ausgereifte und durchdachte Form zu bringen. In zahlreichen Gesprächen und regelmäßigen Treffen konnte ich von seinem Fachwissen profitieren, was maßgeblich zur erfolgreichen Entwicklung dieser Software beigetragen hat.

Mein besonderer Dank gilt ebenso meinem Chef, Herrn Jörn Kutzer, bei Union Investment Real Estate GmbH Hamburg. Er gab mir im Februar 2023 die Möglichkeit, als Werkstudent im IT-Bereich dieses großen Konzerns praktische Erfahrungen zu sammeln. Diese Tätigkeit war nicht nur eine Inspiration für meine Masterarbeit, sondern half mir auch dabei, ein tieferes Verständnis für die Bedürfnisse solcher Unternehmen zu entwickeln.

Von ganzem Herzen danke ich meinen Eltern, die mich trotz der Entfernung stets emotional unterstützt haben, sowie meiner Schwester Mahsa, die mir während des gesamten Masterstudiums große Motivation und wertvolle Unterstützung gegeben hat. Ebenso möchte ich dem Partner meiner Schwester, Julian, meinen Dank aussprechen, dessen hilfreiche Ratschläge und konstruktive Vorschläge entscheidend zum Erfolg meines Projekts beigetragen haben.

Zuletzt danke ich herzlich meinem guten Kommilitonen Tobias Korzyzkowske für den freundschaftlichen Austausch und die gegenseitige Unterstützung während unseres gemeinsamen Studiums.

Ohne euch alle wäre diese Arbeit in ihrer jetzigen Form nicht möglich gewesen.

1 Einleitung

Nach der COVID-19-Pandemie hat die Nutzung videobasierter virtueller Meetings – etwa über Plattformen wie Zoom oder Microsoft Teams – in Arbeitsumgebungen erheblich zugenommen. Trotz Vorteilen wie zeitlicher Flexibilität und der Reduzierung von Reisekosten, sind diese Formate mit Nachteilen verbunden: eine geringere Beteiligung der Teilnehmenden, Unterbrechungen im natürlichen Gesprächsfluss und Ermüdungserscheinungen unter den Teilnehmer*innen. Diese Herausforderungen verdeutlichen den Bedarf an interaktiveren und immersiveren Lösungen. Virtual Reality (VR) hat sich als Technologie etabliert, die eine intensivere virtuelle Präsenz und natürlichere Interaktionen ermöglicht und damit das Interesse zahlreicher Organisationen und Forschungseinrichtungen geweckt hat. Studien zeigen, dass VR-basierte Meetings die Interaktion fördern und das Gefühl der sozialen Präsenz verstärken können.

Vor diesem Hintergrund wurde im Rahmen dieser Arbeit eine multifunktionale VR-Plattform entwickelt, deren Ziel die qualitative Verbesserung virtueller Meetings ist. Das Alleinstellungsmerkmal der Plattform liegt in der Integration intelligenter NPCs (non-player characters), die mithilfe künstlicher Intelligenz mit den Nutzer*innen interagieren. Die Kombination aus VR-Technologie und intelligenten virtuellen Charakteren stellt einen innovativen Ansatz dar, um die Nutzererfahrung zu bereichern und eine dynamische sowie effektive Umgebung für virtuelle Zusammenarbeit zu schaffen.

Bedeutung von Virtual Reality in industriellen und beruflichen Kontexten

Virtual Reality hat sich in den letzten Jahren über den Bereich Gaming hinaus zunehmend in professionellen und industriellen Anwendungsfeldern etabliert. Berichten zufolge kann der Einsatz von VR in Schulung und Kompetenzentwicklung bis zum Jahr 2030 einen wirtschaftlichen Mehrwert von rund 294 Milliarden US-Dollar generieren. Zahlreiche Branchen nutzen VR bereits für Sicherheitstrainings und Wartungssimulationen, was zur Effizienzsteigerung betrieblicher Abläufe beiträgt.[1] Große Technologieunternehmen wie Apple und Meta investieren Milliardenbeträge in die Entwicklung von VR-Headsets und gehen davon aus, dass diese Technologie bald weitreichend in Arbeits- und Lernumgebungen integriert sein wird.

Herausforderungen traditioneller virtueller Meetings

Obwohl virtuelle Meetings eine praktische Lösung für ortsunabhängige Kommunikation darstellen, sind sie mit signifikanten Einschränkungen im Bereich der zwischenmenschlichen Interaktion verbunden. Der Begriff „Zoom-Fatigue“ hat Einzug in die Geschäftswelt gehalten und beschreibt die Ermüdung, Anspannung und Erschöpfung, die durch die wiederholte Teilnahme an Videokonferenzen entsteht – verursacht durch den Mangel an natürlicher Interaktion sowie dem psychischen Druck, kontinuierlich in die Kamera zu blicken. Diese Form

der Belastung kann zu reduzierter Beteiligung, einer Verschlechterung der Gesprächsqualität und insgesamt geringerer Produktivität führen. Da das Arbeiten im Homeoffice zunehmend zur Norm wird, ist die Verbesserung der Interaktivität virtueller Meetings zu einer Notwendigkeit geworden. Studien belegen, dass Meetings in VR-Umgebungen das Gefühl von Nähe und Verbundenheit unter Kolleg*innen deutlich stärken können – mit einer gemeldeten Verbesserung des „social presence"-Gefühls um bis zu 58 % – und somit die Qualität der Kommunikation erheblich steigern.[2] Dies verdeutlicht das Potenzial von VR als kompensierende Lösung für die Schwächen herkömmlicher Videokonferenzen.

Abbildung 1: Meetings im Unternehmens-Metaversum (Quelle: PwC Switzerland, 2023)

Einsatz intelligenter NPCs in der Mensch-Maschine-Interaktion

Die jüngsten Fortschritte im Bereich der künstlichen Intelligenz haben das Maß an Realismus und Interaktivität von NPCs in virtuellen Umgebungen deutlich erhöht. In VR-basierten Lern- und Interaktionsszenarien können diese intelligenten Charaktere Rollen wie Kolleginnen, Kundinnen oder virtuelle Mentorinnen übernehmen und realitätsnahe Simulationen ermöglichen. Durch ihre Fähigkeit, menschenähnliche Verhaltensweisen darzustellen und mithilfe von Natural Language Processing (NLP) echte, bidirektionale Dialoge zu führen, schaffen sie authentische Interaktionserlebnisse. In einem Trainingsszenario könnte beispielsweise ein NPC als unzufriedene Kundin oder ein unzufriedener Kunde auftreten und Mitarbeitende dazu bringen, ihre Fähigkeiten im Kundenumgang zu trainieren – oder in einem virtuellen Meeting als Assistent fungieren und gezielt Informationen bereitstellen.

Integration von Unity, XR, KI und Echtzeit-Netzwerktechnologien

Die Realisierung einer multifunktionalen VR-Plattform erfordert die Integration verschiedener fortschrittlicher Technologien, um ein ganzheitliches, intelligentes und kollaboratives Nutzererlebnis zu ermöglichen. Die Game-Engine Unity in Verbindung mit XR-Frameworks

(Extended Reality) bietet die technische Grundlage zur Entwicklung interaktiver 3D-Umgebungen und ermöglicht die Implementierung immersiver VR-Erlebnisse auf einer Vielzahl von Hardwareplattformen – von autarken Geräten wie Meta Quest bis hin zu leistungsstärkeren Systemen.

Für die Unterstützung von Mehrpersonensitzungen in Echtzeit ist jedoch auch eine stabile, schnelle Netzwerkarchitektur unabdingbar, da ein akzeptables Maß an synchroner, virtueller Präsenz nur bei minimaler Latenz gewährleistet werden kann.[3] Schließlich erlaubt die Einbindung KI-basierter Algorithmen die Integration intelligenter virtueller Charaktere, die sich dynamisch an das Verhalten der Nutzer*innen anpassen – und so alle essenziellen Komponenten für ein lebendiges, interaktives VR-Umfeld vereinen. Die kontinuierliche technologische Weiterentwicklung sowie die zunehmende Überwindung technischer Hürden in den letzten Jahren verdeutlichen, dass diese Kombination von Technologien den Weg für die nächste Generation virtueller Kollaborationsplattformen ebnen kann – eine breite Akzeptanz solcher Lösungen in der Industrie erscheint zunehmend realistisch.

1.1 Motivation

Im Rahmen dieser Forschung basiert die persönliche Motivation des Autors auf eigenen beruflichen Erfahrungen in großen Unternehmen, in denen die Reservierung von Besprechungsräumen regelmäßig eine Herausforderung darstellte. Häufig musste erheblicher Zeitaufwand betrieben werden, um einen verfügbaren Raum zu finden, wobei es zudem oft zu Überschneidungen bei Raumreservierungen kam – was zu Unzufriedenheit und Zeitverlust führte. Laut Berichten verbringen Mitarbeitende täglich bis zu 30 Minuten mit der Suche nach einem freien Konferenzraum. Darüber hinaus kann ineffizientes Management von Besprechungsräumen Unternehmen jährlich bis zu 37 Milliarden US-Dollar durch Zeit- und Ressourcenverschwendung kosten. Etwa 20 % der Konferenzraumreservierungen enden zudem in sogenannten „Ghost Bookings“, bei denen die reservierten Räume ungenutzt bleiben und wertvolle Kapazitäten verloren gehen.[4]

Diese persönlichen Erfahrungen sowie die genannten Statistiken verdeutlichen die Defizite traditioneller Präsenzbesprechungen und stärken die Motivation des Autors, im Rahmen dieser Arbeit einen innovativen Lösungsansatz in Form einer virtuellen Plattform zu verfolgen.

Auch aus technischer und industrieller Sicht ist der Bedarf an der Entwicklung einer solchen Plattform offensichtlich. In den vergangenen Jahren – insbesondere seit der COVID-19-Pandemie – hat die Tendenz zu Remote-Arbeit und virtuellen Meetings stark zugenommen. Zwar sind videobasierte Konferenzsysteme wie Zoom und Microsoft Teams weit verbreitet, jedoch zeigen sie deutliche Einschränkungen – vor allem bei längeren Meetings oder wenn eine intensive zwischenmenschliche Interaktion erforderlich ist. Der Grund hierfür liegt in der begrenzten Fähigkeit solcher Systeme, ein Gefühl physischer Präsenz oder sozialer

Verbundenheit zu vermitteln. Die eingeschränkte Übertragung nonverbaler Signale sowie die fehlende Natürlichkeit der Interaktion beeinträchtigen die Qualität der Teilnahme und des Austauschs. Diese Defizite werden besonders dann spürbar, wenn reichhaltige menschliche Interaktionen notwendig sind, und unterstreichen den Bedarf an einer Weiterentwicklung bestehender Technologien.

Virtual Reality (VR) bietet als vielversprechender Lösungsansatz das Potenzial, diese Herausforderungen zu überwinden. Interaktive 3D-Umgebungen ermöglichen es Teilnehmenden, sich wie in einem gemeinsamen physischen Raum zu fühlen und natürlicher miteinander zu interagieren. Eine aktuelle Studie, in der Mitarbeitende mehrerer Unternehmen ihre Meetings in der VR-Umgebung Horizon Workrooms (ein Produkt von Meta) abhielten, zeigt, dass VR-Meetings die Interaktion, das Gefühl der sozialen Präsenz und den natürlichen Gesprächsfluss deutlich verbessern. Die Teilnehmenden berichteten, dass VR durch die Übertragung nonverbaler Signale und eine intuitivere Gesprächsstruktur eine Erfahrung bietet, die sehr nahe an Präsenzsitzungen herankommt.[5]

Die Entwicklung einer VR-Plattform für Meetings kann somit die kommunikativen Schwächen herkömmlicher Werkzeuge beheben und die Qualität der Zusammenarbeit im Remote-Setting verbessern.

Neben der Durchführung realer Besprechungen zwischen menschlichen Teilnehmerinnen erweitert die Integration intelligenter virtueller Charaktere (NPCs – Non-Player Characters) die Funktionalität der Plattform erheblich. In Schulungs- und Trainingsszenarien bietet VR die Möglichkeit, realitätsnahe Situationen in einer sicheren und kontrollierten Umgebung zu simulieren. In vielen dieser Anwendungen treten virtuelle Menschen auf – entweder in Form von Avataren, die von realen Personen gesteuert werden, oder als autonome Agenten, die computergesteuert agieren. Mehrere Studien belegen, dass Menschen dazu neigen, auf virtuelle Charaktere sozial zu reagieren, als hätten sie es mit echten Menschen zu tun. Daher können intelligente NPCs zur Gestaltung interaktiver Lernszenarien eingesetzt werden. In einer Untersuchung etwa wurde ein KI-gestützter NPC als Lernassistent in einer VR-Umgebung genutzt, um Bedenken hinsichtlich der Teilnahmebereitschaft und Lernangst der Nutzerinnen in virtuellen Bildungssettings zu adressieren.[6]

Die Präsenz solcher intelligenten Agenten in der vorgeschlagenen VR-Plattform unterstreicht ihren multifunktionalen Charakter: Neben der Durchführung von Arbeitsbesprechungen eignet sich die Plattform auch als Umgebung zur Schulung, zum Training sozialer Kompetenzen und zur realitätsnahen Simulation von Interaktionen mit KI-basierten Systemen.

1.2 Umsetzungsszenario

Im Rahmen dieses Umsetzungsszenarios findet ein virtuelles Meeting in einer Virtual-Reality-Umgebung statt, an dem mehrere Teilnehmende aus unterschiedlichen geografischen Regionen

beteiligt sind. Der/die Projektleiter*in ist bereits mit der Sitzung verbunden, übernimmt die Moderation, während die übrigen Teammitglieder mittels ihrer VR-Headsets in den virtuellen Raum eintreten. Jede Person wird dabei durch einen individuellen Avatar im virtuellen Konferenzraum repräsentiert, und alle sitzen gemeinsam um einen virtuellen Konferenztisch. Die gestaltete Umgebung ähnelt einem realen Konferenzsaal, ausgestattet mit Sitzplätzen für alle Teilnehmenden sowie virtuellen Hilfsmitteln zur Unterstützung der Interaktionen während der Besprechung.

Die Kommunikation zwischen den Teilnehmenden erfolgt sowohl über Sprachübertragung als auch über Textnachrichten. Für jede teilnehmende Person ist ein virtuelles Tablet auf dem Konferenztisch vorgesehen, über das Textnachrichten eingegeben werden können. Diese Nachrichten werden in Echtzeit an alle anderen übertragen und erscheinen unmittelbar auf den Tablets der übrigen Teilnehmenden.

Im Konferenzraum befinden sich außerdem zwei große nebeneinander angebrachte Bildschirme. Diese dienen insbesondere der Interaktion mit einem nicht-spielbaren Charakter (NPC), welcher als virtueller Assistent fungiert. Sobald eine teilnehmende Person dem NPC eine Frage stellt, wird der entsprechende Text auf dem ersten Bildschirm angezeigt, sodass alle Anwesenden die gestellte Frage mitverfolgen können. Anschließend recherchiert der NPC, basierend auf öffentlich zugänglichen Informationen aus dem Web, eine geeignete Antwort und präsentiert diese sowohl als Text auf dem zweiten Bildschirm als auch akustisch mittels synthetischer Stimme im virtuellen Raum.

Der NPC wird ausschließlich dann aktiv, wenn er direkt von einer Person angesprochen und ihm eine konkrete Frage gestellt wird. Beispielsweise kann eine Teilnehmerin eine allgemeine oder fachliche Frage äußern – woraufhin der NPC aktiviert wird und eine passende Antwort generiert. Es ist dabei wichtig zu betonen, dass der NPC keinen Zugriff auf unternehmensinterne Daten oder Dokumente hat; seine Antworten basieren ausschließlich auf öffentlich verfügbaren Webinformationen.

Am Ende der Sitzung oder beim Verlassen eines Teilnehmenden wird der entsprechende Avatar aus der virtuellen Umgebung entfernt. Dieser Prozess erfolgt automatisch und unmittelbar in dem Moment, in dem sich eine Person freiwillig vom System abmeldet (Abmeldung). Nur die Avatare der noch anwesenden Personen verbleiben im virtuellen Konferenzraum. Abschließend beendet der/die Projektleiter*in die Sitzung offiziell, und alle verbleibenden Teilnehmenden loggen sich aus dem System aus.

1.3 Projektziel

Das Hauptziel dieses Projekts besteht in der Konzeption und Entwicklung einer mehrbenutzerfähigen Virtual-Reality-Plattform, die interaktive virtuelle Meetings ermöglicht. Die Plattform soll es räumlich getrennten Nutzer*innen erlauben, sich in einer gemeinsamen

dreidimensionalen Umgebung zu versammeln, miteinander zu interagieren und zudem auf natürliche Weise mit intelligenten, nicht-spielbaren Charakteren (NPCs) zu kommunizieren.

Zur Erreichung dieses Ziels wurden folgende Teilziele definiert:

- **Entwurf und Implementierung** einer VR-Mehrbenutzerumgebung für virtuelle Meetings, die die gleichzeitige Teilnahme mehrerer räumlich verteilter Personen ermöglicht.
- **Integration eines Echtzeit-Kommunikationssystems** für Sprach- und Textübertragung zwischen den Nutzer*innen, um natürliche Interaktionen während der Besprechung zu simulieren.
- **Implementierung und Einbindung intelligenter NPCs** in die virtuelle Umgebung, um eine dynamische Interaktion und natürliche Gespräche mit den Nutzer*innen zu ermöglichen.
- **Evaluierung der Funktionsfähigkeit und Systemstabilität** der Plattform in Testszenarien, um die Erfüllung funktionaler und qualitativer Projektanforderungen sicherzustellen.

1.4 Struktur und Methodik

Die Umsetzung dieses Projekts lässt sich in vier übergeordnete Phasen gliedern: die Vorrecherche, die Analyse technischer Konzepte, die Entwurfs- und Implementierungsphase sowie die Test- und Evaluationsphase. In diesem Abschnitt werden zunächst diese Phasen kurz erläutert, bevor der strukturelle Aufbau der Masterarbeit beschrieben wird, welcher den genannten Ablauf widerspiegelt.

In der Phase der Vorrecherche wurde eine umfassende Untersuchung vorhandener Literatur und relevanter Beispiele durchgeführt, um ein fundiertes Verständnis für den Bereich mehrbenutzerfähiger Virtual-Reality-Plattformen zu gewinnen. Dabei wurden Fachliteratur und bestehende Projekte gesichtet, zentrale Konzepte zu VR-Technologien, Mehrnutzersystemen und Mensch-Computer-Interaktion identifiziert sowie der Stand aktueller Technologien und Lösungsansätze bewertet. Diese Analyse diente der präzisen Problemdefinition und Anforderungsspezifikation und legte das wissenschaftlich-technische Fundament für die weiteren Schritte.

In der Phase der technischen Konzeptanalyse wurden verschiedene Optionen zur Umsetzung der funktionalen Anforderungen des Projekts im Detail untersucht. Hierbei erfolgte eine systematische Bewertung existierender Technologien und Frameworks in den Bereichen Echtzeit-Netzwerkkommunikation, Mehrbenutzersysteme, text- und sprachbasierte Kommunikation in virtuellen Umgebungen sowie die Integration künstlicher Intelligenz zur

Steuerung virtueller Charaktere (NPCs). Jedes dieser Werkzeuge wurde hinsichtlich Leistungsfähigkeit, Einschränkungen und Projektkompatibilität analysiert. Auf Grundlage dieser Bewertungen wurden die geeignetsten Technologien und Tools für die Systemarchitektur ausgewählt. Das Ergebnis war ein Set technischer Ansätze, das die Basis für Entwurf und Implementierung des Systems bildete.

In der Entwurfs- und Implementierungsphase wurde auf Basis der vorherigen Erkenntnisse das System konzipiert und die Umsetzung einzelner Komponenten eingeleitet. Zunächst wurde eine Gesamtarchitektur entworfen, in der die Hauptmodule und Subsysteme – wie Nutzer- und Sitzungsmanagement, Netzwerkkommunikation, VR-Interaktion sowie intelligente Agenten – definiert und ihre Wechselwirkungen beschrieben wurden. Ziel des Designs war es, sämtliche zuvor identifizierten Anforderungen zu erfüllen, darunter die Durchführung von VR-Meetings, gleichzeitige Mehrnutzerfähigkeit, sprachliche und textuelle Kommunikation sowie Interaktionen mit intelligenten NPCs. Anschließend wurde das System mithilfe der ausgewählten Technologien implementiert. Die Entwicklung erfolgte iterativ und schrittweise, wobei jede neue Komponente nach ihrer Fertigstellung getestet und auf ihre Integration in das Gesamtsystem überprüft wurde. Am Ende dieser Phase lag ein funktionsfähiger Prototyp der Plattform vor, in dem sämtliche Zielmerkmale realisiert waren.

In der Test- und Evaluationsphase wurde die entwickelte Plattform einer Reihe von Tests unterzogen, um den Grad der Zielerreichung zu überprüfen. Zunächst wurden Funktionstests durchgeführt, um die einzelnen Systemfunktionen zu validieren – darunter die gleichzeitige Verbindung mehrerer Nutzer*innen, die Qualität der Sprach- und Textkommunikation, das Reaktionsverhalten der intelligenten NPCs sowie die Systemstabilität unter realen Bedingungen. Darüber hinaus wurden Leistungskennzahlen erhoben, um eine detaillierte technische Bewertung zu ermöglichen. Ergänzend dazu wurde die Plattform in einer Pilotanwendung getestet, bei der qualitative Rückmeldungen zur Nutzererfahrung und zum Immersionsgrad erhoben wurden. Die Ergebnisse dieser Tests und Rückmeldungen wurden analysiert, um die Wirksamkeit des Systems hinsichtlich der gesetzten Projektziele zu bewerten. Die Auswertung zeigte, inwieweit das entwickelte System den Erwartungen gerecht wurde und in welchen Bereichen Verbesserungsbedarf besteht.

2. Hintergrund und verwandte Studien

Die Untersuchung bestehender Literatur und verwandter Studien bildet eine grundlegende Komponente jeder wissenschaftlichen Arbeit. Diese Phase ermöglicht es der*dem Forschenden nicht nur, ein tiefgreifendes Verständnis des thematischen Kontexts und seiner historischen Entwicklung zu erlangen, sondern auch Forschungslücken zu identifizieren sowie Stärken und Schwächen früherer Ansätze zu analysieren. In diesem Kapitel wird der Fokus auf Multi-User-VR-Plattformen gelegt. Es erfolgt ein strukturierter Überblick über Schlüsselkonzepte, bestehende Projekte, etablierte Anwendungsfelder und aktuelle Forschungstrends. Ziel dieses Abschnitts ist es, eine theoretische und analytische Grundlage für die präzise Problemdefinition, die Auswahl geeigneter Technologien und die zielgerichtete Systementwicklung bereitzustellen.

2.1 Einführung in das Forschungsfeld

Multi-User-VR-Plattformen zählen zu den fortschrittlichsten Formen der Mensch-Computer-Interaktion der letzten Dekade. Diese Systeme ermöglichen die Echtzeitinteraktion mehrerer Nutzer*innen innerhalb gemeinsamer digitaler Räume und finden zunehmend Anwendung in den Bereichen Ausbildung, industrielle Zusammenarbeit, Verhaltensforschung und kollektive Unterhaltung. Ihre besondere Stärke liegt in der Erzeugung eines ausgeprägten „sozialen Präsenzgefühls“ sowie der Integration intelligenter Interaktionswerkzeuge, wodurch sie sich deutlich von herkömmlichen virtuellen Umgebungen abgrenzen. Eine fundierte Analyse dieses Feldes erfordert die Untersuchung grundlegender Konzepte, praktischer Anwendungsbeispiele und aufkommender Trends im Bereich der Multi-User-VR-Technologien.[7]

2.1.1 Definition von Multi-User-VR-Plattformen

In dieser Technologie interagieren mehrere Nutzerinnen gleichzeitig über Headsets und Interaktionsgeräte innerhalb eines gemeinsamen virtuellen Raums. Diese Plattformen beinhalten in der Regel Hardwarekomponenten wie VR-Headsets und Bewegungssensoren sowie netzwerkbasierte Softwarelösungen, die Positions- und Bewegungsdaten der einzelnen Nutzerinnen in Echtzeit austauschen. Ein Beispiel hierfür ist das Projekt **MuVR**, das eine leichtgewichtige, einfach zu konfigurierende Multi-User-VR-Plattform vorstellt. Diese Lösung nutzt kostengünstige, tragbare Hardware, die direkt am Körper getragen wird und den schnellen und automatisierten Systemstart ohne komplexe Infrastruktur ermöglicht.[8]

Abbildung 2: Das vom Benutzer getragene MuVR-System (links), das MuVR-System mit Anmerkungen (rechts) (Quelle: illusioneering lab, 2014)

2.1.2 Industrielle, pädagogische und forschungsbezogene Anwendungen

Ein zentrales industrielles Einsatzgebiet von Multi-User-VR liegt im simulationsbasierten Training unter risikobehafteten Bedingungen. So wurde im Projekt **EPICSAVE** eine immersive, dreidimensionale Multi-User-Umgebung für das Teamtraining von Notfallmedizinischem Personal entwickelt. In dieser Umgebung nehmen zwei Nutzer*innen gleichzeitig über VR-Headsets an Trainingsszenarien teil und führen mithilfe von Echtzeitbewegungstracking Aktivitäten wie das virtuelle Übergeben von Ausrüstung durch.[9]

Abbildung 3: Zwei Schulungsteilnehmer, ausgestattet mit Virtual-Reality-Brillen und Eingabegeräten (links), der virtuelle Patient in der virtuellen Umgebung (rechts) (Quelle: JMIR Publications, 2020)

Erste Studien zeigen, dass solche VR-basierten Simulationen Entscheidungsfähigkeiten und Teamkoordination in kritischen Situationen signifikant verbessern können.

Im Bildungsbereich erfreuen sich Multi-User-VR-Plattformen insbesondere im kollaborativen Lernen wachsender Beliebtheit. Solche virtuellen Umgebungen stellen simulierte gemeinsame

Räume dar, in denen verteilte Lernende miteinander kommunizieren und an Gruppenaktivitäten teilnehmen können – ohne physisch anwesend zu sein. Diese Eigenschaften machen VR zu einem effektiven Werkzeug für Fernunterricht und kooperatives Lernen.[10] Schülerinnen und Schüler bzw. Studierende können somit in einem geteilten virtuellen Klassenraum interagieren. Darüber hinaus werden Multi-User-VR-Plattformen auch in der wissenschaftlichen Forschung eingesetzt, z. B. von Sozialpsychologinnen und Kognitionswissenschaftlerinnen zur Simulation und Analyse von Face-to-Face-Interaktionen im Labor.

2.1.3 Historische Entwicklung und aktuelle Fortschritte

In den letzten Jahren hat sich Virtual-Reality-Technologie rasant weiterentwickelt. Schätzungen zufolge werden bis zum Jahr 2024 weltweit über 171 Millionen Menschen VR-Technologie nutzen.[11] Parallel dazu hat sich die Nutzung multiuserfähiger VR-Anwendungen als aufkommender Trend etabliert. Solche Anwendungen schaffen gemeinsame virtuelle Räume und ermöglichen es mehreren Nutzer*innen, über individuelle 3D-Avatare miteinander zu interagieren – was zu einer neuen Tiefe an Immersion und sozialer Interaktion führt. Mit dem Aufkommen des Begriffs „Metaverse“ und erheblichen Investitionen großer Technologieunternehmen (einschließlich der Umbenennung von Facebook in „Meta“) haben sich die Einsatzmöglichkeiten dieser Plattformen erheblich erweitert. Die Entwicklung verläuft dynamisch und weist auf ein zunehmendes industrielles und gesellschaftliches Interesse hin.

2.2 Überblick über verwandte Projekte und Systeme

Angesichts der zunehmenden Verbreitung von Virtual-Reality-Anwendungen in den Bereichen Remote-Zusammenarbeit und virtuelle Interaktion ist die Analyse bestehender erfolgreicher Projekte und relevanter Plattformen unerlässlich. Ziel dieses Abschnitts ist es, aktuelle Beispiele systematisch zu untersuchen, ihre Stärken und Schwächen herauszuarbeiten und sie anschließend mit der im Rahmen dieser Arbeit entwickelten Plattform zu vergleichen. Die Betrachtung von Systemen wie Horizon Workrooms, Spatial und VRChat ermöglicht ein besseres Verständnis für die Positionierung und die innovativen Ansätze des vorliegenden Projekts.

2.2.1 Analyse global etablierter Beispiele (z. B. Horizon Workrooms, Spatial, VRChat)

Horizon Workrooms wurde von Meta (ehemals Facebook) entwickelt und als offizielle Plattform für virtuelle Meetings konzipiert. Das System umfasst personalisierte VR-Avatare, Whiteboards, die Möglichkeit zur Bildschirmfreigabe sowie die Integration mit

Kalenderfunktionen.[12] Die dreidimensionale Raumdarstellung verstärkt das Gefühl der gleichzeitigen Anwesenheit. Allerdings zeigen Analysen, dass einige Funktionen – wie die getrackte Tastatur oder das Whiteboard-Tool – entfernt wurden und die Nutzung auf das Meta-Ökosystem und Quest-Headsets beschränkt ist.[13]

Abbildung 4: Ein Gesamtüberblick über „Horizon Workrooms“ (Quelle: XR Today, 2023)

Spatial ist eine Plattform mit einem kreativen und künstlerischen Fokus, die sich auf die Präsentation und Interaktion mit 3D-Modellen und digitalen Inhalten konzentriert. Die Anwendung ist geräteunabhängig zugänglich, unter anderem über Webbrowser, und bietet eine benutzerfreundliche Oberfläche, realitätsnahe Avatare sowie Unterstützung für das Hochladen von 3D-Inhalten.[14]

Abbildung 5: Eines der Werbebilder von Spatial (Bild von Spatial) (Quelle: Skarredghost, 2020)

VRChat ist eine weit verbreitete soziale Plattform, die Nutzer*innen ermöglicht, eigene Avatare zu erstellen, virtuelle Welten zu gestalten und frei miteinander zu interagieren.[15] Die Anwendung ist stark auf soziale und unterhaltende Erlebnisse ausgerichtet und daher für formale oder unternehmensbezogene Zwecke weniger geeignet. Dennoch zeichnet sie sich durch eine große Community, umfangreiche Lernressourcen und ein hohes Maß an kreativer Freiheit aus.

Abbildung 6: Spieler im VRChat (Quelle: Wikipedia, 2025)

2.2.2 Vergleich mit dem im Projekt entwickelten System

Die in dieser Arbeit entwickelte Plattform verfolgt einen praxisorientierten Ansatz mit klarem Fokus auf industrielle und bildungsbezogene Anwendungsfälle. Im Gegensatz zu Horizon Workrooms ist sie nicht an ein spezifisches Ökosystem wie Meta gebunden. Ähnlich wie Spatial bietet sie ein flexibel gestaltetes Umfeld, legt jedoch den Schwerpunkt auf funktionale Anforderungen aus dem industriellen Bereich. Im Unterschied zu VRChat basiert das System auf einer formalen Struktur mit klar definierten Komponenten und Rahmenbedingungen.

Ein besonderes Alleinstellungsmerkmal stellt zudem die Integration KI-gesteuerter NPCs dar, die in der Lage sind, auf Benutzeranfragen dynamisch zu reagieren – eine Funktion, die bei den analysierten Plattformen nur in Ansätzen oder gar nicht vorhanden ist.

2.3 Wissenschaftliche Studien und Forschungsarbeiten

In diesem Abschnitt werden wissenschaftliche Publikationen und universitäre Forschungsarbeiten untersucht, die sich auf drei zentrale Themenbereiche dieses Projekts beziehen: Interaktion in Virtual-Reality-Umgebungen, virtuelle Meetings sowie der Einsatz von NPCs. Ziel ist es, durch eine strukturierte Literaturauswertung die relevanten Herausforderungen und häufigen Forschungsschwerpunkte herauszuarbeiten, um eine fundierte Orientierung für die Umsetzung des vorliegenden Projekts zu gewinnen.

2.3.1 Überblick über akademische Arbeiten zu VR-Interaktion, virtuellen Meetings und NPC-Einsatz

Eine umfassende Studie zum nonverbalen Verhalten in Virtual-Reality-Umgebungen zeigte, dass Bewegungs-Synchronisation zwischen Nutzer*innen selbst bei abstrakten Avatarformen (z. B. Würfeln) erkennbar ist. Die gemessene Synchronität bei realen Zwei-Personen-Interaktionen übertraf dabei deutlich die zufällig generierten Kontrollbedingungen („Pseudosynchronie“) und belegt somit die Förderung sozialer Präsenz durch synchronisierte Bewegungen in VR.[16] Die Ergebnisse unterstreichen die Bedeutung von präzisem Motion-Design und nonverbalem Feedback in Multi-User-VR-Umgebungen.

Eine weitere Studie untersuchte die Erwartungen von Nutzer*innen an das Verhalten von NPCs. Sie stellte fest, dass angemessene Reaktionen und realistische nonverbale Rückmeldungen – ausgelöst durch räumliche oder verhaltensbezogene Trigger – die Glaubwürdigkeit der Interaktion erheblich steigern. Insbesondere wurde das Verhalten der NPCs von Teilnehmenden als zielgerichtet und sinnhaft im Kontext der VR-Interaktion wahrgenommen.[17] Diese Erkenntnisse verdeutlichen die Wichtigkeit sorgfältigen Designs virtueller Charaktere in Multi-User-VR-Systemen.

2.3.2 Herausforderungen in der Literatur

Ein zentrales Problem besteht in der mangelnden Abstimmung zwischen dem nonverbalen Verhalten der NPCs und den Bewegungen menschlicher Nutzer*innen. In vielen Fällen reagieren NPCs nicht schnell oder natürlich genug auf körpersprachliche Signale, was zu einem Rückgang des Präsenzgefühls führt. Diese Herausforderung erfordert fortschrittlichere Bewegungserkennungs-Algorithmen sowie Echtzeit-Reaktionsmechanismen.

Ein weiteres wiederkehrendes Thema in der Literatur ist die sogenannte „Cybersickness“, also Übelkeit und visuelle Ermüdung bei der Nutzung von VR. Eine systematische Übersichtsarbeit wies auf Schwächen gängiger Bewertungsinstrumente wie dem SSQ oder VRSQ hin. Ein neueres Instrument, der „Cybersickness Questionnaire (CSQ)“, zeigte dabei eine verbesserte

psychometrische Validität.[18] Darüber hinaus wurde festgestellt, dass individuelle Faktoren – etwa frühere VR-Erfahrung oder physiologische Reaktionen wie Pupillendurchmesser – einen signifikanten Einfluss auf das Auftreten von Übelkeit haben.[19] Diese Ergebnisse belegen die Notwendigkeit qualitativer Nutzerbewertungen bei der Gestaltung von VR-Meetings.

2.3.3 Häufige Forschungsschwerpunkte

Viele Studien konzentrieren sich primär auf **nonverbale Synchronisation zwischen Nutzer*innen in VR**, die als Schlüsselindikator für soziale Präsenz gilt.[20] Daraus ergibt sich, dass ein natürliches Bewegungsdesign von Avataren essenziell für den Erhalt sozialer Interaktion ist.

Im Bereich NPC-Forschung liegt der Schwerpunkt auf der Entwicklung realistischer Reaktionen, geeigneter Interaktions-Trigger und glaubwürdiger Verhaltensmuster. Die Studien fokussieren sich vor allem auf didaktische Anwendungen und kooperative Szenarien, in denen NPCs als Entscheidungshilfen und Lernbegleiter eine zentrale Rolle spielen.[21]

Für die Analyse der Nutzererfahrung werden mittlerweile Instrumente wie der **CSQ** eingesetzt, die eine gleichzeitige Erfassung physischer und emotionaler Aspekte wie Übelkeit, Ermüdung und deren Auswirkungen auf die Leistung ermöglichen.[22] Dies ist entscheidend für die praxisorientierte Validierung von Multi-User-VR-Plattformen.

2.4 Analyse der Forschungslücke (Research Gap)

In diesem Abschnitt werden auf Basis kognitionswissenschaftlicher Studien zentrale Lücken und Bedarfe identifiziert, die das vorliegende Projekt adressiert. Dabei werden die wesentlichen Unterschiede zu bestehenden Projekten sowie die Einordnung dieser Forschung in das Gesamtspektrum der wissenschaftlichen Arbeiten im Bereich Multi-User-VR analysiert.

2.4.1 Identifikation von Lücken und Bedarfen, auf die das aktuelle Projekt abzielt

Aktuelle Forschungsarbeiten im Bereich Virtual Reality legen ihren Schwerpunkt häufig auf soziale oder bildungsbezogene Interaktionen. Viele dieser Studien berichten jedoch von Defiziten hinsichtlich angemessenem nonverbalem Feedback sowie ineffektiven Reaktionen von NPCs.[23] Das vorliegende Projekt begegnet dieser Problematik durch den gezielten Entwurf intelligenter NPCs, die in der Lage sind, sprachlich und schriftlich auf Nutzerfragen zu reagieren. Im Mittelpunkt steht dabei die Qualität des intelligenten Verhaltens – ein Aspekt, der in bisherigen Systemen oft vernachlässigt wurde.

Darüber hinaus konzentriert sich die bestehende Literatur überwiegend auf den Einsatz von Multi-User-VR-Plattformen im Bildungsbereich. Im Gegensatz dazu sind empirische Daten zu industriellen Anwendungen und kollaborativer technischer Zusammenarbeit bislang nur begrenzt vorhanden.[24] Das hier vorgestellte System richtet sich gezielt auf industrielle Kontexte und unterstützt technische Besprechungen und ingenieurwissenschaftliche Koordination in VR-Umgebungen – ein bislang unterrepräsentiertes Feld.

2.4.2 Abgrenzung zu bestehenden Projekten hinsichtlich Zielsetzung und Umsetzung

Bekannte Plattformen wie Horizon Workrooms oder Spatial fokussieren sich primär auf allgemeine Interaktionen und Seminarformate. Die Integration selbstreaktiver Charaktere ist dort bislang nicht zentral vorgesehen.[25] Im Gegensatz dazu erlaubt das vorliegende System nicht nur sprachliche und schriftliche Kommunikation im industriellen Umfeld, sondern integriert auch NPCs, die gezielt auf Anfragen reagieren – jedoch nur dann aktiv werden, wenn sie direkt angesprochen werden. Dieses „zielgerichtete, szenarienbasierte Interaktionsmodell“ stellt ein zentrales Unterscheidungsmerkmal gegenüber generischen VR-Plattformen dar.

Ein weiteres Alleinstellungsmerkmal ist die Priorisierung industrieller Anwendungsfälle in Kombination mit reaktiven, KI-gesteuerten NPCs. Während VRChat vor allem für offene, soziale VR-Erfahrungen geeignet ist, bietet das vorliegende System einen strukturierten Rahmen für professionelle Teamarbeit in technischen Kontexten.

2.4.3 Positionierung des Projekts im Forschungsfeld

Während sich aktuelle Studien stark auf soziale Präsenz und die Reduktion von Übelkeit (Cybersickness) konzentrieren, fehlt bislang ein Fokus auf das Design reaktiver NPCs in industriellen Multi-User-VR-Umgebungen. Dieses Projekt schlägt eine Brücke zwischen strukturierten industriellen Interaktionen und offenen VR-Erfahrungen, indem es Multi-User-VR mit intelligenter, kontextabhängiger KI kombiniert.

Darüber hinaus nutzt das Projekt bestehende technische VR-Lösungen als Basis, erweitert sie jedoch durch intelligente NPC-Integration – und positioniert sich damit an der Schnittstelle zwischen experimentellen, forschungsorientierten Szenarien und industriell nutzbaren Systemen. Daraus ergibt sich das Potenzial, ein neues Modell für technische Meetings in der virtuellen Realität zu etablieren – ein bisher nicht umfassend untersuchter Anwendungsfall.

3. Grundlegende Konzepte und projektbezogene Technologien

Ziel dieses Kapitels ist es, die grundlegenden Konzepte sowie die zentralen Technologien vorzustellen, die im Rahmen dieses Projekts eingesetzt werden. Es werden Begriffe wie Virtual Reality, virtuelle Meetings, VR-Headsets, intelligente Nicht-Spieler-Charaktere (NPCs) und Multi-User-Entwicklung mit Unity behandelt. Ein fundiertes Verständnis dieser Konzepte ist essenziell, um den Aufbau des Projekts sowie das Zusammenspiel seiner Komponenten besser nachvollziehen zu können. Diese Technologien bilden das Fundament für das Design und die Implementierung der Plattform, und ihre genaue Kenntnis unterstützt die Analyse und Bewertung der einzelnen Projektphasen maßgeblich.

3.1 Begriffe und theoretische Grundlagen

Virtual Reality, Multi-User-Interaktionen und intelligente virtuelle Charaktere (NPCs) sind die drei zentralen Grundpfeiler dieses Projekts. Virtual Reality bezeichnet eine rechnergestützte, immersive Umgebung, in der sich die Nutzerinnen mithilfe spezieller Headsets in Echtzeit bewegen und interagieren können. Multi-User-Interaktion beschreibt die Fähigkeit mehrerer Nutzerinnen, sich gleichzeitig in einem gemeinsamen virtuellen Raum aufzuhalten und miteinander zu kommunizieren – typischerweise über Sprach- oder Textkanäle. Nicht-Spieler-Charaktere (NPCs) sind virtuelle Figuren, die nicht von echten Personen gesteuert werden, sondern mittels Konversations-KI dynamische und natürliche Dialoge mit den Nutzer*innen führen können.

3.1.1 Virtual Reality und interaktive Erfahrung

Virtual Reality (VR) ist eine computergenerierte, simulierte Umgebung, in der Nutzerinnen mithilfe spezieller Headsets mental und sensorisch eintauchen können. Diese Technologie erzeugt ein sogenanntes „Präsenzgefühl“ – ein subjektives Empfinden, sich real in einer virtuellen Umgebung zu befinden. Das Maß an Präsenz hängt stark vom Grad der kognitiven Einbindung der Nutzerinnen ab: Je intensiver ihre Konzentration und geistige Teilnahme, desto stärker das Gefühl der Anwesenheit und desto effektiver die Interaktion.

Die gesamte Nutzererfahrung (User Experience, UX) in VR umfasst verschiedene Dimensionen wie Immersion, Präsenz, kognitive Involvierung sowie allgemeinen Komfort und Zufriedenheit. Immersion ist hierbei vor allem an technische und sensorische Merkmale wie visuelle und auditive Qualität gebunden und bildet die Grundlage für das Erleben von Präsenz.

Präsenz und kognitive Involvierung führen zu intensiveren und befriedigenderen Interaktionen, wodurch eine insgesamt positive und effektive User Experience entsteht.[26]

3.1.2 Multi-User-Interaktionen und Echtzeitsynchronisation

Multi-User-Interaktionen in Virtual-Reality-Umgebungen ermöglichen die gleichzeitige Präsenz mehrerer Nutzerinnen in einem gemeinsamen virtuellen Raum. Diese Art der Interaktion erfordert die Echtzeitsynchronisation von Daten wie der Position der Avatare, Sprachübertragung, Textnachrichten sowie der Umgebungsinteraktionen. Die Bedeutung dieser Synchronisation liegt insbesondere in der Aufrechterhaltung des sozialen Präsenzgefühls und der Verbesserung der Qualität gemeinsamer Kollaboration. Eine Studie in Frontiers in Virtual Reality zeigt, dass die Klarheit und Glaubwürdigkeit der gleichzeitigen Interaktionen eine zentrale Rolle bei der Förderung von geteilter Wahrnehmung und Co-Präsenz spielen. Der Verlust signifikanter Synchronisation – beispielsweise durch Verzögerungen bei Avatar-Bewegungen oder Sprachübertragung – kann hingegen das Vertrauen und die Zufriedenheit der Nutzerinnen erheblich beeinträchtigen.[27]

Zur effizienten Umsetzung solcher Interaktionen nutzen Systeme Architekturen wie Netcode for GameObjects, welche die Zustände und Datenpakete innerhalb akzeptabler Latenzgrenzen übertragen. Die Gewährleistung flüssiger, konsistenter Abläufe in der Interaktion ist eine zentrale Voraussetzung für ein positives Multi-User-Erlebnis in VR-Umgebungen.

3.1.3 Nicht-Spieler-Charaktere (NPCs) und konversationelle Künstliche Intelligenz

Nicht-Spieler-Charaktere (Non-Player Characters, NPCs) sind virtuelle Figuren in VR-Umgebungen, die nicht durch reale Nutzerinnen gesteuert werden. Sie übernehmen Aufgaben wie Interaktion, Orientierungshilfe oder die Simulation menschlichen Verhaltens in der virtuellen Welt. In modernen Anwendungen dienen sie als vermittelnde Schnittstelle zwischen dem System und den Nutzerinnen, können die Nutzererfahrung personalisieren, narrative Elemente vorantreiben oder relevante Informationen bereitstellen.

Durch den technologischen Fortschritt in den Bereichen Maschinelles Lernen und Natural Language Processing (NLP) haben sich NPCs von einfachen, skriptgesteuerten Akteuren zu flexiblen, interaktiven Dialogpartnern entwickelt. Sie sind heute in der Lage, natürliche Gespräche zu führen und intelligent auf Nutzeranfragen zu reagieren, was ihre Einsatzmöglichkeiten in immersiven VR- und AR-Szenarien erheblich erweitert.[28]

3.2 Verwendete Technologien für die Systemimplementierung

Dieses Projekt wurde mit der Spiele-Engine Unity entwickelt, die umfassende Unterstützung für die Erstellung von Virtual-Reality-Erfahrungen bietet. Zur Realisierung natürlicher Interaktionen der Nutzerinnen mit virtuellen Objekten wurde das Paket XR Interaction Toolkit eingesetzt, das hochentwickelte Funktionen zur Einbindung von taktilen und steuerungsbasierten Interaktionen in VR-Umgebungen bereitstellt. Die Mehrspieler-Funktionalität und Sitzungsverwaltung wurden durch die Verwendung von Netcode for GameObjects sowie den Cloud-Diensten Unity Lobby und Unity Relay realisiert. Die Echtzeit-Sprachkommunikation zwischen den Nutzerinnen wird über die Vivox-Bibliothek abgewickelt, während für die Implementierung dialogfähiger virtueller Charaktere die Plattform Convai genutzt wurde.

3.2.1 Unity-Spiel-Engine und VR-Entwicklungswerkzeuge

Unity ist eine leistungsstarke und vielseitige Spiele- und Simulationsengine für die Entwicklung von Virtual-Reality-Anwendungen. Die Engine basiert auf einer modularen XR-Architektur, die sogenannte „Subsysteme“ nutzt. Dadurch kann ein einmal entwickeltes Programm auf verschiedenen VR-Geräten wiederverwendet werden. Unity stellt durch das System zur XR Plug-in-Verwaltung sowie durch unterstützende Pakete eine integrierte Entwicklungsumgebung für VR bereit. Zu den wichtigsten Tools gehören: XR Interaction Toolkit, XR Core Utilities, das Input System und vorgefertigte VR-Projektvorlagen – diese decken einen Großteil der funktionalen Anforderungen typischer VR-Projekte ab.

Darüber hinaus unterstützt Unity eine optimierte Grafikdarstellung, etwa durch die Verwendung der Universal Render Pipeline (URP), und stellt fortschrittliche Grafiktools bereit. Diese ermöglichen die Erstellung komplexer 3D-Umgebungen mit hoher Bildrate auf modernen VR-Headsets. Unity ist kompatibel mit zahlreichen VR-Systemen wie Meta Quest, SteamVR, VisionOS sowie mit branchenüblichen Standards wie OpenXR.

OpenXR ist ein offener und lizenzfreier Standard, der von der Khronos Group entwickelt wurde und einheitliche Programmierschnittstellen (APIs) für Augmented- und Virtual-Reality-Anwendungen bietet. Vor der Einführung von OpenXR mussten Entwickler*innen für jede VR-Plattform eigene Implementierungen vornehmen. OpenXR hingegen ermöglicht durch eine einheitliche API eine erhebliche Reduzierung von Entwicklungszeit und -aufwand. Laut Khronos kann mit OpenXR eine Anwendung einmal entwickelt und auf allen OpenXR-kompatiblen Geräten ausgeführt werden, wobei plattformspezifische Funktionen durch Erweiterungen genutzt werden können.

In Unity ist ein offizielles OpenXR-Plug-in enthalten. Durch die Aktivierung in den Projekteinstellungen können Entwickler*innen ihre Anwendungen direkt auf unterstützten Headsets wie Meta Quest, SteamVR, HoloLens u.a. ausführen. Dieses Plug-in erleichtert laut Spezifikation die plattformübergreifende VR-Entwicklung erheblich und garantiert eine breite Kompatibilität mit möglichst geringem Anpassungsaufwand.

XR Interaction Toolkit ist ein komponentenbasiertes High-Level-Entwicklungspaket in Unity zur Implementierung typischer Interaktionen in VR-/AR-Anwendungen. Es stellt ein gebrauchsfertiges Framework bereit, das 3D-Interaktionen und Benutzeroberflächen über Unitys Ereignissysteme nutzbar macht. Mit dem Toolkit lassen sich grundlegende VR-Interaktionen wie Objektgreifen, Menübedienung und Umgebungsinteraktion schnell und effizient umsetzen.

3.2.2 Netzwerkdienste für Mehrspieler-Funktionalität in Unity (Netcode for GameObjects, Lobby, Relay)

Das Paket **Netcode for GameObjects** (NGO) ist ein offizielles Unity-Framework, das Netzwerkfunktionen direkt in den typischen GameObject– und MonoBehaviour-basierenden Arbeitsablauf integriert. Es enthält zentrale Komponenten wie den NetworkManager, der als zentrales Konfigurations- und Steuerungselement für die Netzwerkverbindungen dient. Dieser verwaltet sowohl die Client-Server-Kommunikation als auch die Synchronisation von Daten zwischen allen beteiligten Clients.

Mit NGO können Spielobjekte netzwerkweit automatisch instanziiert („spawned“) und deren Zustände über sogenannte Network Variables zwischen Clients und Server synchronisiert werden. Zudem ermöglichen Remote Procedure Calls (RPCs) die gezielte Auslösung von Funktionen auf entfernten Instanzen. NGO bietet somit die notwendige Infrastruktur zur Implementierung von Multiplayer-Architekturen wie Host-Client oder dedizierte Server-Modelle in Unity.[29]

Der Dienst **Unity Lobby** unterstützt die Organisation und Verwaltung von Spielsitzungen und die Vermittlung zwischen Spieler*innen. Zu seinen Hauptfunktionen gehören:

- Durchsuchen aktiver Spielsessions und Beitritt zu diesen
- Erstellung privater Lobbys und Teilen eines Join-Codes für direkte Einladungen
- Verwendung der Quick Join-Funktion für den sofortigen Beitritt zu verfügbaren Sitzungen
- Hosting von Lobbys auf Servern zur zentralen Zugriffsverwaltung

Darüber hinaus ermöglicht der Lobby-Dienst eine persistente Verwaltung von Sitzungen, inklusive Mechanismen zum erneuten Beitritt (Rejoin) oder zur Host Migration bei plötzlicher

Trennung. Zusammengefasst übernimmt der Lobby-Service die Sitzungskontrolle und vereinfacht die Verbindungskoordination für Multiplayer-Anwendungen erheblich.[30]

Der Dienst **Unity Relay** dient als Vermittlungsinstanz zwischen Clients und ermöglicht eine stabile und sichere Verbindung über ein zentrales Relay-Server-System, ohne dass NAT-Konfigurationen oder eigene Serverinfrastruktur erforderlich sind. Der Host einer Spielsitzung erzeugt über Relay eine sogenannte Allocation, aus der ein Join-Code generiert wird. Die Clients verbinden sich über diesen Code mit dem Relay-Server, ohne dass IP-Adressen direkt offengelegt werden müssen.

Die Hauptvorteile von Relay sind:

- Sichere Verbindungen über einen Vermittlungsserver mittels Join-Codes
- Wegfall eigener Serverinfrastruktur oder Drittanbieterlösungen zur Netzwerkkoordination
- NAT-Traversal und Verbindung von Teilnehmer*innen in getrennten Netzwerken durch die Funktion eines Relay-Servers als Proxy

Zusammenfassend ergibt sich aus der Kombination von Netcode for GameObjects, Unity Lobby und Unity Relay ein leistungsfähiges Framework zur einfachen Umsetzung vollständiger Multiplayer-Anwendungen in Unity. NGO bildet das technische Fundament zur Daten- und Ereignissynchronisation, Lobby verwaltet Sitzungen und Teilnehmer, während Relay netzwerktechnische Hürden überwindet und die Verbindungsstabilität sicherstellt.

3.2.3 Sprach- und Textkommunikationssystem

Zur Realisierung der Kommunikationsfunktionen zwischen den Nutzerinnen kommen in diesem Projekt zwei unterschiedliche Ansätze zum Einsatz. Für die Sprachkommunikation wird auf die Plattform **Vivox** zurückgegriffen, die eine leistungsfähige und latenzarme Lösung für Echtzeit-Sprachübertragung in Mehrbenutzerumgebungen bereitstellt. Nach dem Betreten eines virtuellen Lobbys werden die Nutzerinnen automatisch einem Sprachkanal zugewiesen, dessen Kennung mit dem Namen des jeweiligen Lobbys übereinstimmt. Diese Zuordnung gewährleistet eine sichere und effiziente Kommunikation zwischen den Teilnehmenden innerhalb derselben Sitzung – ohne die Notwendigkeit einer komplexen Netzwerk-Infrastruktur oder der Weitergabe von IP-Adressen.[31]

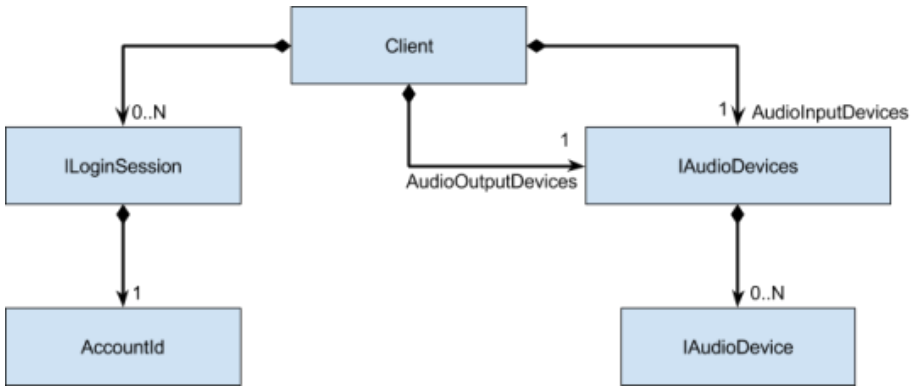

Abbildung 7: Vivox-Klassenmodell-Diagramm (Quelle: Vivox Unity, 15.1.25)

Im Gegensatz dazu basiert das Textnachrichtensystem in diesem Projekt nicht auf Vivox, sondern wurde speziell unter Verwendung der Netcode for GameObjects-Architektur implementiert. Dabei werden Textnachrichten mithilfe von ServerRpc- und ClientRpc-Methoden zwischen dem Server und den Clients übertragen und in einer NetworkList gespeichert. Die Nachrichten werden anschließend in der benutzerdefinierten Benutzeroberfläche (UI) des Projekts angezeigt. Über interaktive Tablets im virtuellen Besprechungsraum können die Nutzer*innen Textnachrichten in Echtzeit empfangen und senden. Dieses System erlaubt eine direkte, asynchrone schriftliche Kommunikation als Ergänzung zur Sprachübertragung.

3.2.4 Konversationale Künstliche Intelligenz und NPC-Plattform (Convai AI)

Convai ist eine cloudbasierte Plattform zur Entwicklung intelligenter Nicht-Spieler-Charaktere (NPCs), die auf konversationaler künstlicher Intelligenz (KI) basieren. Durch den Einsatz von Technologien zur natürlichen Sprachverarbeitung (Natural Language Processing, NLP) sowie multimodaler Verarbeitung (visuell, auditiv, textbasiert und raumbezogen) analysiert Convai die Eingaben der Nutzer*innen und generiert daraufhin Ausgaben in Form von Sprache, Text, Animationen oder interaktiven Verhaltensweisen. Die API von Convai wurde so konzipiert, dass sie die sensorischen Ein- und Ausgaben menschlicher Kommunikation simuliert: visuelle, auditive und textuelle Reize werden verarbeitet und führen zu situationsabhängigen Reaktionen wie gesprochenen Antworten, animierten Gesten oder komplexen Aktionen.

Zusätzlich bietet Convai eine nahtlose Integration mit Spielentwicklungsumgebungen. Das dedizierte Unity SDK stellt sämtliche Werkzeuge zur Verfügung, um konversationelle KI rasch in Unity-Projekte zu integrieren und die Entwicklung natürlicher, intelligenter NPC-Interaktionen zu erleichtern. Die Plattform unterstützt darüber hinaus weitere Engines wie

Unreal Engine und 3JS und bietet umfangreiche Dokumentationen und APIs zur plattformübergreifenden Nutzung.[32]

Abbildung 8: Die Übersicht über NPC-Charaktere mit verschiedenen Rollen (Quelle: Convai)

Funktionen des Unity SDK

- **Demobeispiele**: Das SDK enthält Demonstrationsszenen und Scripts. Zum Beispiel kann die vordefinierte Figur Amelia in einer Beispielszene getestet werden. Wird ihre eindeutige Character ID konfiguriert, beginnt die Konversation automatisch, sobald sich derdie Nutzerin dem NPC nähert.

Struktur des Frage-Antwort-Dialogs

- **Dialogeinleitung**: Die Kommunikation mit einem NPC beginnt typischerweise durch eine Benutzeraktion – etwa durch Annäherung und Aktivieren einer Schaltfläche. Der NPC ist dann bereit, Fragen zu empfangen.
- **Spracheingabe und -verarbeitung**: Gesprochene oder geschriebene Nutzereingaben werden mithilfe von Spracherkennung in Text umgewandelt und an die Server von Convai übermittelt. Dort erfolgt die Analyse durch große Sprachmodelle (LLMs), unter Berücksichtigung des individuellen Wissensstands und Narrativs des NPC.
- **Antwortgenerierung**: Die ermittelten Antworten kombinieren LLM-Logik mit vordefinierten Verhaltensmodellen, um möglichst natürliche und intelligente Reaktionen zu erzeugen. NPCs können zudem komplexe Benutzerbefehle in konkrete Aktionen innerhalb der Spielwelt übersetzen.
- **Antwortausgabe**: Die Antwort wird sowohl in gesprochener Form (mittels Text-to-Speech) als auch als eingeblendeter Text im Spiel dargestellt. Die Kommunikation

erfolgt in Echtzeit über die cloudbasierten APIs von Convai und ist auf minimale Latenzzeiten optimiert.

Verbindung zur API[33]

- **API-Key einrichten**: Nach der Registrierung bei Convai wird ein individueller API-Schlüssel generiert. Dieser wird in Unity über das Menü Convai > API Key Setup eingefügt und als Resources-Objekt gespeichert.
- **Charakterkonfiguration**: Jeder NPC erhält eine eindeutige Character ID, die in Unity mit der zugehörigen Komponente verknüpft wird. Damit wird die Verbindung zwischen Spielobjekt und Cloud-Charakter auf dem Convai-Server hergestellt.
- **Echtzeit-Kommunikation**: Alle Nutzereingaben (Text oder Sprache) werden in Echtzeit an die Server von Convai übertragen. Die Server verarbeiten die Anfrage und senden umgehend eine Antwort zurück. Die Kommunikation erfolgt über gängige Webprotokolle (HTTP/WebSocket) und wurde für Stabilität in Multiplayer-Umgebungen optimiert.

Vorteile im VR-Kontext

- **Umgebungsbewusstsein**: Convai-NPCs können den Kontext der Umgebung (z. B. Anwesenheit von Nutzer*innen) erkennen und ihre Antworten entsprechend anpassen.
- **Inhaltliche Anpassungsfähigkeit**: Entwickler*innen können den Wissensstand und das Narrativ eines NPC individuell gestalten – z. B. als Lehrperson oder virtuellen Guide. Damit sind vielfältige Einsatzbereiche (Bildung, Unterhaltung, Service usw.) möglich.
- **Ressourceneffizienz**: Da NPCs im vorliegenden Projekt ausschließlich auf Fragen reagieren und keine Gespräche initiieren, werden Rechenressourcen gezielt für relevante Interaktionen eingesetzt. Die Cloud-Infrastruktur von Convai nutzt moderne Systeme wie NVIDIA Riva und Accelerated Compute, um die Interaktion flüssig und reaktionsschnell zu gestalten – ein entscheidender Faktor für immersive VR-Erlebnisse.

3.2.5 Hardware- und Plattformen der Virtual Reality (Oculus Quest)

Die eigenständigen Virtual-Reality-Headsets der Meta-Quest-Reihe (früher Oculus Quest) vereinen die Eigenschaften kabelgebundener und mobiler Headsets. Diese All-in-One-Geräte bieten durch ein integriertes Inside-Out-Tracking sechs Freiheitsgrade (6DOF) für Kopf und Controller.[34] Dadurch können Nutzer sich ohne externe Sensoren frei in einem virtuellen Raum bewegen und interagieren. Die integrierte Oculus-Insight-Technologie verwendet vier Weitwinkelkameras sowie Algorithmen der Computer Vision, um die exakte Position von Kopf und Controllern zu erfassen. Zudem schützt das Guardian-System Nutzer durch Abgrenzung eines sicheren Bewegungsbereichs im realen Raum vor Kollisionen mit physischen Objekten.

Die Quest-Headsets sind mit Oculus-Touch-Controllern ausgestattet, die natürliche Handbewegungen in virtuelle Aktionen umsetzen und so das Gefühl der Präsenz der eigenen Hände in der VR stärken. Die internen Displays nutzen hochauflösende LCD-Panels – z. B. bietet das Quest 2 eine Auflösung von 1832 × 1920 Pixel pro Auge, während das ursprüngliche Quest mit 1440 × 1600 Pixel arbeitete. Die grafische Verarbeitung wurde anfangs vom mobilen Qualcomm Snapdragon 835 übernommen, später durch den Snapdragon XR2 (erste und zweite Generation) ersetzt. Diese leistungsfähige Hardware in Kombination mit ausreichend RAM erlaubt es, anspruchsvolle VR-Anwendungen unabhängig vom PC auszuführen.

Unterschiede zwischen Quest 2 und Quest 3

Die zweite und dritte Generation der Quest-Serie unterscheiden sich in mehreren zentralen Punkten:

- **Prozessor und Arbeitsspeicher:** Quest 2 verwendet den Snapdragon XR2 Gen 1 und 6 GB LPDDR4X-RAM, während Quest 3 mit dem XR2 Gen 2 und 8 GB LPDDR5 ausgestattet ist. Die Grafikleistung der neuen Generation ist etwa doppelt so hoch.
- **Display:** Quest 2 bietet 1832 × 1920 Pixel pro Auge bei einer Bildwiederholrate von bis zu 120 Hz, Quest 3 erreicht 2064 × 2208 Pixel pro Auge und unterstützt ebenfalls bis zu 120 Hz – insgesamt rund 30 % mehr Pixel.[35]
- **Mixed Reality (MR):** Quest 3 verfügt über zwei farbige Frontkameras, die ein hochqualitatives Passthrough des realen Umfelds ermöglichen, im Gegensatz zur monochromen Ansicht des Quest 2.[36]
- **Weitere Merkmale:** Quest 3 enthält einen Tiefensensor für automatisches Raum-Scanning und neue Controller ohne Tracking-Ringe mit geringerem Stromverbrauch. Das Gewicht liegt bei ca. 515 g (Quest 3) gegenüber 503 g (Quest 2).

Diese Unterschiede machen die Quest 3 zur leistungsstärkeren Wahl in Bezug auf Bildqualität und Mixed-Reality-Funktionen, während das Quest 2 eine kostengünstige Einstiegsoption bleibt.

Unity-Kompatibilität

Beide Headsets werden von der Unity-Engine umfassend unterstützt. Mit dem XR Plug-in Management können Entwickler das Oculus-Plug-in aktivieren und Anwendungen direkt auf Quest 2 und Quest 3 ausführen. Darüber hinaus stellt Meta spezielle SDKs für Unity zur Verfügung, die den Zugriff auf Eingabedaten der Headsets und Controller erleichtern. Dies ermöglicht die plattformübergreifende Entwicklung immersiver 3D-Umgebungen mit minimalem Anpassungsaufwand.

Rolle in der VR-Interaktion

Durch präzises Tracking von Kopf und Controllern ermöglichen Quest-Headsets eine natürliche Interaktion mit der virtuellen Umgebung. Nutzer können sich durch Kopfdrehungen umsehen

und mit den Touch-Controllern virtuelle Objekte berühren und bewegen. Diese multimodale Interaktion verstärkt das Gefühl der Präsenz im virtuellen Raum. Die ergonomische Gestaltung der Controller und die verbesserte Akkulaufzeit tragen zur Optimierung der Benutzererfahrung bei.

In diesem Projekt sind jedoch Funktionen wie Blick- und Hand-Tracking deaktiviert, sodass sämtliche Interaktionen über die Controller erfolgen. Insgesamt bieten die Oculus-Quest-Geräte durch ihre Sicherheitsfunktionen, Benutzerfreundlichkeit und breite Softwarekompatibilität eine solide Plattform für die Entwicklung und Analyse von VR-Anwendungen.

4. Implementierung

4.1 Gesamtarchitektur des Systems und Projektstruktur

Bevor auf die technischen Details der Implementierung eingegangen wird, ist ein umfassendes Verständnis der allgemeinen Systemstruktur und der technischen Architektur des Projekts unerlässlich. Dieser Abschnitt dient als Grundlage für die Darstellung der weiteren Teile des Implementierungskapitels und vermittelt einen klaren Überblick über die Hauptkomponenten des Projekts sowie deren Zusammenspiel.

Angesichts des modularen Charakters dieses Projekts wurde die Systemarchitektur so entworfen, dass die Entwicklung, das Testen und die Erweiterung der einzelnen Module unabhängig voneinander möglich sind.

4.1.1 Vorstellung der Hauptkomponenten des Systems

Die entwickelte Plattform besteht aus modularen, klar definierten Komponenten, die jeweils eine spezifische Funktion innerhalb der Multiuser-VR-Erfahrung erfüllen. Diese Komponenten umfassen:

- **Benutzermanagement- und Verbindungssystem:** Basierend auf einer Kombination aus Netcode for GameObjects sowie den Unity-Diensten Relay und Lobby, um sichere Sitzungen zu erstellen, zu hosten und Teilnehmer zu verbinden.
- **VR-Interaktionsmodul:** Ermöglicht die direkte Interaktion der Benutzer mit der 3D-Umgebung mithilfe des XR Interaction Toolkit.
- **Kommunikations-Subsystem:** Echtzeit-Sprachkommunikation über Vivox sowie ein eigens entwickeltes textbasiertes Nachrichtensystem auf Basis von Netcode for GameObjects.
- **NPC-System (Non-Player Characters):** Integration des Convai-Dienstes zur dialogbasierten Interaktion mit virtuellen Charakteren, die Benutzereingaben intelligent beantworten.
- **Szenen- und Objektmanagement-Modul:** Zuständig für das Laden von Umgebungen, digitale Tablets, Anzeigen und interaktive Objekte.
- **UI-Infrastruktur:** Zuständig für die Anzeige von Panels, Benachrichtigungen, schwebenden Beschriftungen und anderen In-Game-GUI-Elementen.

Innerhalb dieser Architektur ist jede Komponente so konzipiert, dass sie unabhängig entwickelt und getestet werden kann. Gleichzeitig wird eine reibungslose Echtzeitsynchronisation zwischen den Komponenten gewährleistet. So wird beispielsweise beim Beitritt eines Benutzers zu einem virtuellen Raum automatisch eine neue Sitzung über das Lobby-Modul erstellt, das entsprechende XR Rig des Benutzers aktiviert, sein Avatar in der Szene registriert und bei Bedarf Sprachkommunikation sowie UI-Elemente initialisiert.

4.1.2 Übersichtliche Systemarchitektur

Im Folgenden wird die technische Architektur des Projekts beschrieben. Die Architektur des entwickelten Projekts basiert auf einem klassischen Client-Server-Modell, wobei die Interaktion und Datenflüsse zwischen den Modulen klar nachvollziehbar gestaltet sind.

In dieser Architektur betreten die Clients die virtuelle Umgebung über das XR Rig des Unity-Clients. Die Verbindung zum Netzwerk erfolgt über das Netcode-Framework in Kombination mit den Unity-Diensten Relay und Lobby. Für die Echtzeit-Sprachkommunikation wird die Vivox-Plattform eingesetzt, während der Textnachrichtenversand über ein eigens entwickeltes Nachrichtensystem auf Basis von Netcode abgewickelt wird.

Intelligente Antworten auf Benutzeranfragen werden durch die Conversational-AI-Plattform Convai generiert. Sämtliche Ereignisse, Interaktionen und Echtzeitsynchronisierungen werden über das Netcode-Framework sowie dessen unterstützende Subsysteme auf Client- und Serverseite koordiniert und gesteuert.

4.2 Konfiguration der Entwicklungsumgebung und Zielplattform

In diesem Abschnitt wird der Prozess der Einrichtung der Entwicklungsumgebung sowie der erforderlichen Konfigurationen für den reibungslosen Betrieb des Projekts im Virtual-Reality-Kontext beschrieben. Die Implementierung basiert auf der Spiele-Engine Unity und nutzt spezialisierte Werkzeuge wie das XR Interaction Toolkit, Netcode for GameObjects sowie die Dienste Vivox und Convai. Ein besonderer Fokus liegt auf der Kompatibilität mit OpenXR-basierten VR-Headsets wie dem Oculus Quest 2 und Quest 3. Eine korrekte Konfiguration dieser Komponenten ist entscheidend für die erfolgreiche Entwicklung eines mehrbenutzerfähigen und interaktiven Systems.

4.2.1 Unity-Einstellungen und erforderliche Plugins

Der erste Schritt zur Entwicklung des Projekts besteht in der Installation einer kompatiblen Unity-Version. Dieses Projekt wurde mit der stabilen LTS-Version Unity 2022.3.51f1

umgesetzt, welche vollständige Unterstützung für OpenXR, Netcode und netzwerkbezogene Unity-Dienste bietet.

Anschließend müssen über den Unity Package Manager die folgenden Pakete dem Projekt hinzugefügt werden:

- com.unity.netcode.gameobjects
- com.unity.transport
- com.unity.services.lobby
- com.unity.services.relay
- com.unity.xr.interaction.toolkit
- com.unity.xr.openxr
- com.unity.services.vivox

Die Installation erfolgt über das Menü: Window > Package Manager > Add package by name

Nach der Paketinstallation ist die Build-Plattform auf Android umzustellen, um das Projekt für Oculus Quest-Geräte bereitzustellen. Anschließend müssen im Menü Project Settings unter dem Reiter XR Plugin Management die Optionen XR Plug-in Management sowie OpenXR aktiviert werden.

4.2.2 Konfiguration von OpenXR und dem XR Interaction Toolkit

In diesem Schritt wird die Ausführungsumgebung für Virtual Reality konfiguriert. Zur Verwaltung der Benutzerinteraktionen mit Objekten und Benutzeroberflächen wird das XR Interaction Toolkit verwendet. Zunächst wird ein XR Rig in die Hauptszene eingefügt. Dieses Rig steuert die Position und Rotation des Benutzers in der virtuellen Umgebung und beinhaltet linke und rechte Controller.

Für jeden Controller werden Input Actions definiert, um Aktionen wie Tastendruck und Interaktion mit Benutzeroberflächen zu steuern. Zur Interaktion mit Objekten werden Komponenten wie der XR Ray Interactor verwendet.

Im Abschnitt OpenXR-Einstellungen wird das Oculus Touch Controller Profile als Interaction Profile ausgewählt. Darüber hinaus werden unter Feature Groups nur die erforderlichen Funktionen wie Meta Quest Support aktiviert.

4.2.3 Zielplattform-Einstellungen (Oculus Quest)

Zur Ausführung dieses Projekts auf eigenständigen Virtual-Reality-Headsets wie dem Oculus Quest 2 und 3 werden spezielle Einstellungen unter Build and Run vorgenommen. Im Bereich Project Settings > Player wird die Plattform Android ausgewählt und Parameter wie das Scripting Backend auf IL2CPP gesetzt.

Im Build Settings-Fenster wird die Option Development Build für Testzwecke deaktiviert und die Compression Method auf LZ4 eingestellt, um die endgültige Dateigröße zu reduzieren. Der Projekt-Build wird über ein USB-Kabel mithilfe von ADB auf das Headset übertragen und installiert.

4.2.4 Erstellung eines ersten 3D-Prototyps (Unternehmensumgebung)

Zu Beginn der Entwicklungsphase wurde eine realitätsnahe Arbeitsumgebung in das Projekt integriert, um eine interaktive Testbasis für die Entwicklung und Erprobung der Funktionen bereitzustellen. Diese Umgebung, die ein simuliertes Unternehmensbüro darstellt, spielt eine wichtige Rolle bei der Bewertung des UI-Designs, der Mehrbenutzerinteraktionen sowie der Positionierung von Avataren und NPCs.

Hierfür wurde ein hochwertiges, vorgefertigtes 3D-Asset aus dem Unity Asset Store verwendet. Die Umgebung mit dem Titel "Unity Japan Office" wird kostenlos und offiziell von Unity zur Verfügung gestellt und basiert auf dem echten Unity-Büro in Japan. Sie ist unter folgendem Link verfügbar:

https://assetstore.unity.com/packages/3d/environments/unityjapanoffice-152800

Diese Szene überzeugt durch professionelles Design, realistische Beleuchtung, detaillierte Innenarchitektur und authentische Büroausstattung. Nach Auswahl und Import über den Asset Store werden alle Bestandteile in das Projekt geladen und in einer separaten Szene platziert. Anschließend wurden Elemente wie der Standort des Konferenztisches, Wanddisplays und interaktive Tablets projektspezifisch angepasst und positioniert.

Dank des realistischen Designs dieser Umgebung konnten Funktionen wie Sprachkommunikation, Interaktion mit Benutzeroberflächen, NPC-Ausgaben sowie die Positionsprüfung von Teilnehmern in virtuellen Sitzungen besonders praxisnah und effektiv getestet werden.

4.3 Netzwerkimplementierung (Netcode for GameObjects)

Einer der zentralen technischen Aspekte dieses Projekts ist die Implementierung der Kommunikationsinfrastruktur zwischen mehreren Nutzern in einer gemeinsamen virtuellen Umgebung. Um eine stabile, sichere und latenzarme Echtzeitverbindung zu ermöglichen, wird das von Unity offiziell entwickelte Framework Netcode for GameObjects (NGO) eingesetzt, das sich nahtlos mit weiteren Unity-Diensten wie Relay und Lobby integrieren lässt.

Dieses Framework bietet umfassende Funktionen zur Synchronisation von Benutzerzuständen, Objekten und Interaktionen in Echtzeit. Dank der Unterstützung eines Client-Server-Architekturmodells wird die Entwicklung von Mehrbenutzersystemen in Virtual-Reality-Umgebungen erheblich erleichtert.

Die zugrunde liegende verteilte Netzwerkarchitektur erlaubt es Benutzern aus verschiedenen Regionen der Welt, gleichzeitig innerhalb eines gemeinsamen virtuellen Raums zu interagieren. In diesem Kapitel werden die Installation und Konfiguration von Netcode, Relay und Lobby, die Kommunikationsarchitektur, das Avatar-Management, die Synchronisation von Objekten sowie die Messung von Netzwerklatenzen ausführlich behandelt.

Eine präzise Implementierung dieser Netzwerkschicht stellt sicher, dass alle netzwerkbezogenen Interaktionen reibungslos ablaufen und die Mehrbenutzererfahrung in der VR-Umgebung konsistent und störungsfrei bleibt.

4.3.1 Installation und Einrichtung von Netcode for GameObjects

Der erste Schritt zur Implementierung einer Mehrbenutzerfunktionalität in Unity besteht in der Installation des Frameworks Netcode for GameObjects (NGO), das von Unity entwickelt wurde. Dieses Framework basiert auf einem Client-Server-Modell und ermöglicht die Synchronisation von Objekten, Benutzerzuständen und Echtzeitinteraktionen in Multiplayer- und VR-Projekten.

Die Installation erfolgt über den Unity Package Manager. Dort wird über die Option „Add package by name“ der folgende Paketname eingegeben:

com.unity.netcode.gameobjects

Nach der Installation muss ein neues GameObject zur Szene hinzugefügt werden, das für das Netzwerkmanagement zuständig ist. Dieses Objekt wird mit der Komponente NetworkManager versehen. Zu den Grundeinstellungen dieses Komponenten zählen die Zuweisung eines Player Prefabs, die Auswahl des verwendeten Transports sowie allgemeine Netzwerkkonfigurationen. Als Transport wird standardmäßig Unity Transport verwendet, das im Paket com.unity.transport enthalten ist und vollständig mit NGO integriert wurde.

Im nächsten Schritt wird das Player Prefab definiert, das beim Verbindungsaufbau eines Clients automatisch instanziiert wird. Dieses Prefab muss die Komponente NetworkObject enthalten sowie eigene Skripte, die von NetworkBehaviour erben.

Beispielhafter Netzwerkmanager für VR-Multiplayer-Projekte:

```
1. using Unity.Netcode;
2. // Manages network configurations and player prefab settings.
3. public class NetworkManagerVRMultiplayer : NetworkManager
4. {
5.     // Player prefab for network instantiation
6.     private GameObject playerPrefab;
7.
8.     // Ensure the application keeps running in the background
9.     private bool runInBackground = true;
10.     private int tickRate = 30; // Network tick rate for data synchronization
11.
12.     // Called when the script instance is being loaded
13.     private void Awake()
14.     {
15.         Application.runInBackground = runInBackground;
16.     }
17.
18.     public void CustomStartHost()
19.     {
20.         StartHost();
21.         Debug.Log("Host started successfully.");
22.     }
23.
24.     public void CustomStartClient()
25.     {
26.         StartClient();
27.         Debug.Log("Client connected successfully.");
28.     }
29.
30.     public void SetPlayerPrefab(GameObject newPlayerPrefab)
31.     {
32.         playerPrefab = newPlayerPrefab;
33.         NetworkConfig.PlayerPrefab = playerPrefab;
34.         Debug.Log("Player prefab has been updated.");
35.     }
36.
37.     public void UpdateTickRate(int newTickRate)
38.     {
39.         if (newTickRate < 0)
40.         {
41.             Debug.LogError("Tick rate cannot be negative.");
42.             return;
43.         }
```

```
44.         tickRate = newTickRate;
45.         NetworkConfig.TickRate = (uint)tickRate;
46.         Debug.Log($"Network tick rate updated to {tickRate}.");
47.     }
48. }
```

Codeblock 1: Zentrales Skript zur Steuerung des NetworkManagers, inklusive Startfunktionen, Prefab-Zuweisung und Tickrate-Konfiguration

Im Inspector-Fenster des NetworkManager muss als Transporttyp UnityTransport ausgewählt werden. Voraussetzung dafür ist die Installation des Pakets com.unity.transport, das in den neuesten Unity-Versionen direkt mit NGO kompatibel ist. Die Konfiguration des Transports kann dort ebenfalls vorgenommen werden.

Ein besonders wichtiger Punkt ist, dass alle GameObjects, die zwischen den Nutzern synchronisiert werden sollen, zwingend die Komponente NetworkObject enthalten und als Prefab im Projekt definiert sein müssen. Diese Objekte müssen außerdem durch den NetworkManager instanziiert werden, um korrekt im Netzwerk verteilt zu werden.

Für einen ersten Funktionstest kann eine einfache Benutzeroberfläche (UI) erstellt werden, über die das Gerät als Host, Client oder Server gestartet werden kann. Auf diese Weise lässt sich überprüfen, ob die Verbindung erfolgreich aufgebaut und das Player Prefab korrekt geladen wird.

Diese Grundkonfiguration bildet die Basis der Netzwerkarchitektur des Projekts und wird in den folgenden Abschnitten mit weiterführenden Komponenten wie **Relay** und **Lobby** erweitert.

4.3.2 Installation und Konfiguration von Relay und Lobby zur Raumverwaltung und Client-Verbindung

Nach der erfolgreichen Einrichtung von Netcode for GameObjects (NGO) besteht der nächste Schritt in der Implementierung einer stabilen Mehrbenutzerinfrastruktur mithilfe der Dienste Unity Relay und Unity Lobby. Diese Dienste sind Teil der Unity Gaming Services (UGS) und ermöglichen es, virtuelle Räume zu erstellen, Benutzer zu verwalten sowie sichere Verbindungen zwischen Clients und Host aufzubauen – ohne die Notwendigkeit von Portweiterleitungen oder statischen IP-Adressen.

Registrierung des Projekts und Aktivierung der Dienste

Um diese Funktionen zu nutzen, muss das Unity-Projekt zunächst im UGS-Dashboard registriert werden. Anschließend werden die folgenden Dienste im Dashboard aktiviert:

- **Authentication**
- **Relay**
- **Lobby**

Im nächsten Schritt werden die dazugehörigen Pakete über den Unity Package Manager installiert:

com.unity.services.authentication

com.unity.services.lobby

com.unity.services.relay

Authentifizierung per Anonymous Login

Zur Vereinfachung wird eine anonyme Authentifizierung verwendet, um Benutzer temporär im System zu registrieren. Dies geschieht über die folgende Methode:

```
1. public virtual async Task<bool> Authenticate()
2.     {
3.         // Check if UGS has not been initialized yet, and initialize.
4.         if (UnityServices.State == ServicesInitializationState.Uninitialized)
5.         {
6.             var options = new InitializationOptions();
7.             string playerId = "Player";
8.
9.             // Check for command line args in builds
10.             if (!Application.isEditor && m_UseCommandLineArgs)
11.             {
12.                 playerId += GetPlayerIDArg();
13.             }
14.
15.             options.SetProfile(playerId);
16.             Utils.Log($"{k_DebugPrepend}Signing in with profile {playerId}");
17.
18.             // Initialize UGS using any options defined
19.             await UnityServices.InitializeAsync(options);
20.         }
21.
22.         // If not already signed on then do so.
23.         if (!AuthenticationService.Instance.IsAuthorized)
24.         {
25.             // Signing in anonymously for simplicity sake.
26.             await AuthenticationService.Instance.SignInAnonymouslyAsync();
27.         }
28.
29.         // Cache PlayerId.
30.         XRINetworkGameManager.AuthenicationId = AuthenticationService.Instance.PlayerId;
31.         return UnityServices.State == ServicesInitializationState.Initialized;
```

```
32.     }
```

Codeblock 2: Authentifizierungsfunktion. Diese verwendet eine einfache Authentifizierung und anonyme Anmeldung

Erstellung von Lobby und Relay Allocation durch den Host

Der Host erstellt anschließend eine Relay-Allocation und eine Lobby, die mit dieser Allocation verknüpft wird. Dies geschieht über eine Methode, die unter anderem einen Join-Code generiert, Optionen für die Lobby definiert und die Netzwerkparameter auf das Transport-System überträgt:

```
1. public async Task<Lobby> CreateLobby(string roomName = null, bool isPrivate = false, int playerCount = XRINetworkGameManager.maxPlayers)
2.     {
3.         try
4.         {
5.             m_Status.Value = "Creating Relay";
6.             // Creates a new Allocation based on the defined max players above
7.             var alloc = await RelayService.Instance.CreateAllocationAsync(XRINetworkGameManager.maxPlayers);
8.
9.             m_Status.Value = "Creating Join Code";
10.             // Get a join code based on the Allocation
11.             var joinCode = await RelayService.Instance.GetJoinCodeAsync(alloc.AllocationId);
12.
13.             // Creates Lobby Options Dictionary for other clients to find and join
14.             var options = new CreateLobbyOptions
15.             {
16.                 // Set the Data to be used for lobby filtering
17.                 Data = new Dictionary<string, DataObject>
18.                 {
19.                     {
20.                         // Set Join Code Key
21.                         k_JoinCodeKeyIdentifier, new DataObject(DataObject.VisibilityOptions.Public, joinCode)
22.                     },
23.                     {
24.                         // Set Region Key
25.                         k_RegionKeyIdentifier, new DataObject(DataObject.VisibilityOptions.Public, alloc.Region)
26.                     },
27.                     {
28.                         // Set Build ID Key
29.                         k_BuildIdKeyIdentifier, new DataObject(DataObject.VisibilityOptions.Public, Application.version, DataObject.IndexOptions.S1)
30.                     },
31.                     {
32.                         // Set Scene Key
```

```
33.                         k_SceneKeyIdentifier, new DataObject(DataObject.VisibilityOptions.Public,
SceneManager.GetActiveScene().name, DataObject.IndexOptions.S2)
34.                     },
35.                     {
36.                         // Set Editor Key
37.                         k_EditorKeyIdentifier, new DataObject(DataObject.VisibilityOptions.Public,
hideEditorFromLobby.ToString(), DataObject.IndexOptions.S3)
38.                     },
39.                 },
40.                 IsPrivate = isPrivate,
41.             };
42.
43.             m_Status.Value = "Creating Lobby";
44.             // Creates the Lobby with the specified max players and lobby options. Currently just naming
"General Lobby"
45.             string lobbyName = string.IsNullOrEmpty(roomName) ?
$"{XRINetworkGameManager.LocalPlayerName.Value}'s Room" : $"{roomName}";
46.
47.             // This joinCode is then registered in the form of a room through the Lobby service.
48.             // RATE LIMIT: 2 request per 6 seconds
49.             var lobby = await LobbyService.Instance.CreateLobbyAsync(lobbyName, playerCount, options);
50.             Utils.Log($"{k_DebugPrepend}Created Lobby with Join Code: {joinCode}, Region:
{alloc.Region}, Build ID: {Application.version}, Scene: {SceneManager.GetActiveScene().name}, Editor:
{hideEditorFromLobby}");
51.
52.             // Stop the heartbeat routine if one exists, and starts a new one. This keeps the lobby active for
visibility
53.             if (m_HeartBeatRoutine != null) StopCoroutine(m_HeartBeatRoutine);
54.             m_HeartBeatRoutine = StartCoroutine(LobbyHeartbeatCoroutine(lobby.Id));
55.
56.             //Populate the transport data with the relay info for the host (IP, port, etc...)
57.             m_Transport.SetHostRelayData(alloc.RelayServer.IpV4, (ushort)alloc.RelayServer.Port,
alloc.AllocationIdBytes, alloc.Key, alloc.ConnectionData);
58.             ConnectedToLobby(lobby);
59.
60.             return lobby;
61.         }
62.         catch (Exception e)
63.         {
64.             string failureMessage = "Failed to Create Lobby. Please try again.";
65.             Utils.Log($"{k_DebugPrepend}{failureMessage}\n\n{e}", 1);
66.             // Debug.LogWarning($"[XRMPT] {failureMessage}\n\n{e}");
67.             OnLobbyFailed?.Invoke(failureMessage);
68.             return null;
69.         }
70.     }
```

Codeblock 3: Mit dieser Funktion wird versucht, eine Lobby zu erstellen und eine vernetzte Sitzung durchzuführen

Verbindung des Clients über Relay

Sobald die Lobby erstellt ist, können Clients mithilfe des Join-Codes oder über Filtersysteme dem Raum beitreten. Eine spezifische Methode liest die Relay-Daten aus dem Lobby-Eintrag und überträgt diese an das Transportmodul:

```
1. async Task SetupRelay(Lobby lobby)
2.         {
3.             m_Status.Value = "Connecting To Relay";
4.             // Get the Join Allocation for the lobby based on the key
5.             var alloc = await
RelayService.Instance.JoinAllocationAsync(lobby.Data[k_JoinCodeKeyIdentifier].Value);
6.             // Set the transport client data (IP, port, etc..)
7.             m_Transport.SetClientRelayData
8.             (
9.                 alloc.RelayServer.IpV4, (ushort)alloc.RelayServer.Port,
10.                 alloc.AllocationIdBytes, alloc.Key, alloc.ConnectionData, alloc.HostConnectionData
11.             );
12.             return;
```

Codeblock 4: Beitritt eines Kunden zu einer Sitzung über Relay unter Verwendung von Informationen in der Lobby

Die Kombination aus Relay, Lobby und NGO stellt das Kernstück der Netzwerkarchitektur dieses Projekts dar. Diese Architektur gewährleistet eine sichere, flexible und skalierbare Echtzeitkommunikation, ohne dass technische Barrieren wie Routerkonfigurationen oder statische IPs beachtet werden müssen. Damit wird ein realitätsnahes, interaktives VR-Multiplayer-Erlebnis ermöglicht – mit minimalem Setup-Aufwand für Entwickler und Benutzer.

4.3.3 Client-Server-Kommunikationsarchitektur

Die in diesem Projekt verwendete Kommunikationsarchitektur basiert auf dem klassischen Client-Server-Modell. In diesem Modell übernimmt ein Gerät die Rolle des Hosts, welcher gleichzeitig als Server und Client fungiert. Weitere Benutzer (Clients) verbinden sich mit diesem Host. Dieses Architekturmodell ermöglicht eine zentrale Verwaltung des Objektzustands, eine konsistente Datensynchronisation sowie die Validierung von Ereignissen.

Das Framework Netcode for GameObjects (NGO) bildet das technologische Rückgrat dieser Architektur und gewährleistet eine stabile, skalierbare und echtzeitfähige Kommunikation zwischen den Teilnehmern.

In der aktuellen Version des Projekts wird die Entscheidung, ob das System als Host gestartet oder als Client verbunden werden soll, automatisch in der Methode ConnectedToLobby()

getroffen. Diese Methode prüft zunächst, ob der aktuelle Benutzer der Eigentümer (Host) der Lobby ist. Falls ja, wird die Host-Instanz mithilfe von NetworkManager.Singleton.StartHost() gestartet. Andernfalls wird durch StartClient() eine Verbindung zum existierenden Host aufgebaut.

Die Entscheidung basiert auf dem Vergleich der im Lobby-Datensatz gespeicherten Host-ID mit der aktuellen Authentifizierungs-ID des Benutzers.

Speicherung von Sitzungsinformationen und Ereignisverarbeitung

Zusätzlich zur Initialisierung der Netzwerkverbindung ruft die Methode ConnectedToLobby() relevante Informationen der aktiven Sitzung ab, wie etwa:

- den Namen der Lobby,
- den Beitrittscode (Join Code),
- die zugewiesene geografische Region.

Diese Informationen werden lokal gespeichert und für Diagnose- und Verwaltungszwecke verwendet. Im Erfolgsfall wird im Konsolenfenster eine Bestätigung angezeigt, die den erfolgreichen Verbindungsaufbau dokumentiert. Andernfalls wird eine entsprechende Fehlermeldung ausgegeben, um den Benutzer über das Scheitern der Verbindung zu informieren.

Schließlich wird die Methode SubscribeToLobbyEvents() aufgerufen, um Listener für Ereignisse wie Spielerbeitritte oder -austritte zu registrieren. Dadurch kann das System auf dynamische Veränderungen in der Lobby proaktiv reagieren und beispielsweise Benutzeroberflächen oder Netzwerkzustände entsprechend anpassen.

```
1. protected virtual bool ConnectedToLobby()
2.     {
3.         bool connected;
4.         if (autoConnectOnLobbyJoin)
5.         {
6.             ConnectedRoomRegion =
m_LobbyManager.connectedLobby.Data[LobbyManager.k_RegionKeyIdentifier].Value;
7.             ConnectedRoomCode = m_LobbyManager.connectedLobby.LobbyCode;
8.             ConnectedRoomName.Value = m_LobbyManager.connectedLobby.Name;
9.
10.             if (m_LobbyManager.connectedLobby.HostId == AuthenicationId)
11.             {
12.                 connected = NetworkManager.Singleton.StartHost();
13.             }
14.             else
15.             {
16.                 connected = NetworkManager.Singleton.StartClient();
17.             }
18.         }
```

```
19.         else
20.         {
21.             connected = true;
22.             //Players are connected to the lobby, but have not started a Networked Game session.
23.         }
24.         if (connected)
25.         {
26.             Utils.Log($"{k_DebugPrepend}Connected to game session. Lobby:
{m_LobbyManager.connectedLobby.Name}.");
27.             m_ConnectionState.Value = ConnectionState.Connected;
28.             SubscribeToLobbyEvents();
29.         }
30.         else
31.         {
32.             Utils.Log($"{k_DebugPrepend}Failed to connect to lobby
{m_LobbyManager.connectedLobby.Name}.");
33.             m_LobbyManager.OnLobbyFailed?.Invoke($"Failed to connect to lobby
{m_LobbyManager.connectedLobby.Name}.");
34.         }
35.         return connected;
36.     }
```

Codeblock 5: Prüfen von den Verbindungsstatus zur Lobby und Entscheidungen von Host-ID automatisch, ob der Host oder Client starte

4.3.4 Hinzufügen und Darstellung von Avataren in der gemeinsamen Umgebung

Zur Darstellung der Nutzerpräsenz in der virtuellen Umgebung wird in diesem Projekt ein minimalistisches Avatar-System verwendet, das aus drei zentralen Komponenten besteht: Head, LeftController und RightController. Im Gegensatz zu fortgeschritteneren VR-Systemen wird hier bewusst auf physisch modellierte Hände oder Ganzkörper-Avatare verzichtet. Stattdessen übernehmen virtuelle Controller die Funktion der Nutzerhände.

Dieses leichte und ressourcenschonende Design ist sowohl auf die Hardware-Eigenschaften der Meta Quest Headsets abgestimmt als auch darauf ausgelegt, die Netzwerkleistung zu optimieren und eine höhere Bildrate (Frame Rate) in Mehrbenutzerszenarien zu gewährleisten.

Dynamische Instanziierung und Steuerung

Jeder Avatar wird beim Beitritt eines Spielers zur Sitzung dynamisch instanziiert und durch die Klasse XRINetworkPlayer verwaltet. In der Methode OnNetworkSpawn() dieser Klasse wird zunächst überprüft, ob der betreffende Avatar dem lokalen Benutzer gehört (Local Player).

- Falls dies zutrifft, wird der zugehörige XR Rig aktiviert, sodass der Nutzer die volle Kontrolle über Kamerabewegungen und Eingaben erhält.

- Wenn es sich hingegen um einen fremden Avatar handelt, werden lediglich die Komponenten Head, LeftController und RightController aktiviert und deren Positionen sowie Rotationen werden netzwerkseitig synchronisiert.

Codeausschnitt: OnNetworkSpawn-Methode:

```
1. public override void OnNetworkSpawn()
2. {
3.     base.OnNetworkSpawn();
4.     if (IsOwner)
5.     {
6.         // Set Local Player.
7.         LocalPlayer = this;
8.         XRINetworkGameManager.Instance.LocalPlayerConnected(NetworkObject.OwnerClientId);
9.
10.        // Get Origin and set head.
11.        m_XROrigin = FindFirstObjectByType<XROrigin>();
12.        if (m_XROrigin != null)
13.        {
14.            m_HeadOrigin = m_XROrigin.Camera.transform;
15.        }
16.        else
17.        {
18.            Utils.Log("No XR Rig Available", 1);
19.        }
20.
21.        SetupLocalPlayer();
22.    }
23.    CompleteSetup();
24. }
```

Codeblock 6: Die Methode OnNetworkSpawn() ist für den Start des Avatars verantwortlich

Die Synchronisation der Positionen der Avatar-Komponenten erfolgt in der Methode LateUpdate(). In dieser Methode werden die aktuellen Positionen und Rotationen des Kopfs und der Controller — basierend auf den Daten des XR Origin — den entsprechenden Avatar-Objekten zugewiesen und anschließend über das Netcode for GameObjects (NGO) an die anderen Teilnehmer im Netzwerk übermittelt. Auf diese Weise sind die Bewegungen und Haltungen der Avatare in Echtzeit für alle Nutzer der gemeinsamen VR-Sitzung sichtbar. Die genaue Funktionsweise dieser Aktualisierung ist in der Datei XRINetworkPlayer.cs einsehbar.

Die Verbindung zwischen jedem Avatar und dem Netzwerksystem wird durch die Komponenten NetworkObject, NetworkTransform sowie durch Eigentumsinformationen (Client Authority) gesteuert. Durch die entsprechenden Einstellungen im Avatar-Prefab wird sichergestellt, dass jeder Nutzer ausschließlich seinen eigenen Avatar steuern kann, während die übrigen Avatare lediglich zur Anzeige dienen. Diese Implementierung wird durch die Kombination mit den Relay- und Lobby-Diensten von Unity stabil und zuverlässig über verschiedene Netzwerkverbindungen hinweg aufrechterhalten.

4.3.5 Synchronisation von Zuständen und Objekten zwischen den Nutzern

In einem mehrbenutzerfähigen Virtual-Reality-System ist die Echtzeit-Synchronisation der Positionen und Rotationen von Objekten und Avatar-Komponenten zwischen allen Teilnehmenden eine der wichtigsten technischen Anforderungen. In diesem Projekt erfolgt diese Synchronisation mithilfe der Komponenten des Netcode for GameObjects (NGO), insbesondere durch NetworkTransform und ClientNetworkTransform. Diese Komponenten erfassen die Veränderungen von Position, Rotation und Skalierung auf dem besitzenden Client und übermitteln sie an die übrigen Clients im Netzwerk.

Für die Avatar-Bestandteile der Nutzer – d. h. Kopf, linker und rechter Controller – wird ClientNetworkTransform verwendet, um deren Bewegungen mit minimaler Latenz und maximaler Präzision zu synchronisieren. Dadurch wird beispielsweise sichergestellt, dass Kopf- und Handbewegungen eines Spielers in der realen Welt in Echtzeit für alle anderen sichtbar sind. Die technische Umsetzung basiert auf der internen Struktur des NGO und verwendet Methoden wie SetPositionAndRotation(), welche im LateUpdate()-Zyklus implementiert sind:

```
1. // XRINetworkPlayer.cs
2. protected virtual void LateUpdate()
3. {
4.     if (!IsOwner) return;
5.
6.     // Set transforms to be replicated with ClientNetworkTransforms
7.     leftController.SetPositionAndRotation(m_LeftControllerOrigin.position, m_LeftControllerOrigin.rotation);
8.     rightController.SetPositionAndRotation(m_RightControllerOrigin.position,
m_RightControllerOrigin.rotation);
9.     head.SetPositionAndRotation(m_HeadOrigin.position, m_HeadOrigin.rotation);
10. }
```

Codeblock 7: Synchronisation der Avatar-Komponenten (linker, rechter Controller und Kopf) in der LateUpdate()-Methode. Nur der Besitzer des Avatars darf Bewegungsdaten übertragen

Für andere interaktive Objekte – z. B. Werkzeuge, Bildschirme oder UI-Elemente – wird NetworkObject verwendet. So wird etwa der Status eines virtuellen Buttons, der im gemeinsamen Raum gedrückt wird, als Netzwerkvariable definiert, wobei nur der Server berechtigt ist, Änderungen vorzunehmen. Diese Vorgehensweise verhindert gleichzeitige Konflikte bei mehreren Nutzern und gewährleistet, dass der endgültige Status immer mit dem Server-Referenzzustand synchronisiert ist.

Ein typisches Problem bei der Zustands-Synchronisation ist das Datenkonflikt-Management, wenn mehrere Nutzer gleichzeitig mit einem Objekt interagieren. In diesem Projekt wurde durch die eindeutige Definition des Besitzes (Ownership) jedes Objekts sowie durch

Verwendung der NGO-Methoden wie IsOwner und IsServer sichergestellt, dass nur autorisierte Nutzer ein Objekt steuern oder aktualisieren können. Diese Struktur bewahrt die Datenkonsistenz und verhindert unerwartetes Verhalten bei gleichzeitigen Mehrnutzer-Interaktionen.

4.3.6 Verwaltung von Latenz und Netzwerkfehlern

In mehrbenutzerfähigen Virtual-Reality-Umgebungen ist es entscheidend, eine flüssige und unterbrechungsfreie Nutzererfahrung zu gewährleisten. Dies setzt eine effektive Reduzierung von Latenzzeiten sowie ein robustes Management von Netzwerkfehlern voraus. In diesem Projekt wurden verschiedene Maßnahmen ergriffen, um Verzögerungen zu minimieren und deren Auswirkungen zu kontrollieren. Dazu zählen unter anderem:

- die Begrenzung der Netzwerk-Tickrate (Tick Rate),
- die Verwendung von Client-seitiger Interpolation mit vorhersagbarer Verzögerung,
- sowie das kontinuierliche Monitoring der Verbindungsqualität.

Ein zentrales Instrument zur Latenzreduktion ist das Lag-Compensation-System von Netcode for GameObjects (NGO). Dieses erlaubt es dem Client, bestimmte Aktionen – wie das Interagieren mit Objekten oder das Betreten spezieller Bereiche – bereits vor der Bestätigung durch den Server durchzuführen. Die finale Validierung erfolgt serverseitig, wobei fehlerhafte Zustände automatisch korrigiert werden.

Im Bereich der Fehlerbehandlung werden Probleme wie Verbindungsabbrüche, Server-Unerreichbarkeit oder Störungen im Relay-Service durch Verbindungsprüfungen in den Schichten Lobby und NetworkManager identifiziert. Beispielsweise wird bei einem Verbindungsfehler eine Fehlermeldung angezeigt, und ein automatischer Wiederverbindungsprozess (Reconnection) wird initiiert. Zusätzlich wird die Methode ConnectionApprovalCallback des NGO verwendet, um eingehende Verbindungen auf Authentizität zu prüfen. Dies verhindert den Zugriff unautorisierter Nutzer auf die virtuelle Sitzung. Diese Sicherheitsschicht stärkt nicht nur die Stabilität des Systems, sondern ermöglicht auch eine dynamische Kontrolle über das Netzwerkverhalten.

Für die Laufzeitüberwachung der Netzwerkleistung wurde ein Monitoring-System implementiert, das unter anderem folgende Metriken erfasst:

- Durchschnittliche, minimale und maximale Latenz (Ping-Zeiten),
- Identifikation von Jitter (Netzwerkfluktuationen),
- Analyse der Auswirkungen der Spieleranzahl auf die Netzwerkperformance,
- Vergleich der Netzwerkleistung vor und nach Optimierungsmaßnahmen.

Diese Werte werden in CSV-Dateien gespeichert, die automatisch im Ordner Assets des Projekts generiert werden. Nach Abschluss der Tests werden die erfassten CSV-Dateien analysiert, um Einblicke in das Verhalten und die Stabilität des Netzwerks zu gewinnen.

Dieses Monitoring-System bietet eine präzise Analyse der Netzwerkgesundheit und unterstützt das Entwicklerteam dabei, fundierte Entscheidungen in Echtzeit zu treffen. Ein solches Design ist insbesondere in industriellen Kontexten unerlässlich für die Gewährleistung einer stabilen Multi-User-Erfahrung.

4.4 Kommunikationssysteme des Projekts (Sprachübertragung mit Vivox / Textnachrichten mit eigener Struktur)

Für die Umsetzung von Kommunikationsfunktionen im mehrbenutzerfähigen Virtual-Reality-Projekt wurden zwei unterschiedliche Systeme implementiert: Ein Echtzeit-Sprachchat auf Basis der Vivox-Plattform sowie ein leichtgewichtiges, intern entwickeltes System für den Austausch von Textnachrichten – unabhängig von externen Diensten.

Sprachkommunikation mit Vivox

Für die Integration der Sprachkommunikation wird das offizielle Vivox-Paket verwendet, das im Rahmen der Unity Gaming Services (UGS) angeboten wird. Dieser Dienst ermöglicht die Erstellung von Sprachkanälen. Im vorliegenden Projekt wird jeder Benutzer beim Betreten der Lobby automatisch mit einem Sprachkanal verbunden, dessen Name mit der Lobby-ID übereinstimmt. Diese Struktur gewährleistet eine direkte Korrelation zwischen Lobby-System und Sprachkanal.

Die Verwaltung von Benutzerzugriffen, Kanalteilnahmen und Berechtigungen erfolgt über interne Methoden wie ParticipantAddedToChannel und ParticipantRemovedFromChannel. Darüber hinaus wurden über das Interface IVivoxService Kontrollfunktionen wie MuteInputDevice() und UnmuteInputDevice() zur Steuerung des Mikrofoneingangs implementiert. Mögliche Verbindungsprobleme werden über Events wie ConnectionRecovering und ConnectionRecovered erkannt und dem Nutzer über die UI-Schicht mitgeteilt.

Textkommunikation über eigens entwickelte Lösung

Im Gegensatz zur Sprachkommunikation basiert das System zur Übertragung von Textnachrichten nicht auf Vivox. Stattdessen wurde ein schnelles, stabiles und ressourcenschonendes Messaging-System implementiert, das vollständig auf der Grundlage von Netcode for GameObjects (NGO) arbeitet.

Die Übertragung erfolgt mittels ServerRpc und ClientRpc-Aufrufen. Textnachrichten werden zunächst vom Server empfangen und anschließend über eine NetworkList an alle Clients

verteilt. Die Nachrichten erscheinen dabei in einer eigens entwickelten grafischen Benutzeroberfläche, die in Echtzeit für alle Teilnehmer synchronisiert wird.

Dieses Design ermöglicht Textkommunikation ohne externe Server und integriert sich nahtlos in die bestehende Netzwerkarchitektur des Projekts. Dadurch wird ein hohes Maß an Stabilität, Sicherheit und Performance gewährleistet – besonders in sicherheitsrelevanten oder unternehmensspezifischen Anwendungen.

4.4.1 Konfiguration von Vivox für den Sprachchat

Für die Implementierung eines Echtzeit-Sprachchats in diesem Projekt wurde Vivox eingesetzt. Im Folgenden werden die Schritte zur Installation, Inbetriebnahme und Konfiguration dieser Technologie im Kontext des Projekts erläutert:

Installation des Pakets und Aktivierung des Dienstes

Zunächst wurde das Paket com.unity.services.vivox über den Unity Package Manager hinzugefügt und der Vivox-Dienst im Services-Panel aktiviert. Im Anschluss daran muss das Projekt mit dem Unity Dashboard verbunden werden, sodass die Authentifizierungsdaten (Project ID und Token) automatisch vom Paket übernommen werden können. Dieser Schritt ist entscheidend, um sicherzustellen, dass das SDK korrekt mit dem Vivox-Service kommuniziert.

Ein einfaches Authentifizierungssystem wurde entwickelt, um Benutzer bei Vivox anzumelden. Dieser Prozess beginnt mit dem Aufruf der Methode VivoxService.Instance.InitializeAsync() und wird durch LoginAsync() abgeschlossen, um den Benutzer erfolgreich im System anzumelden:

```
1. try
2. {
3.     await VivoxService.Instance.InitializeAsync();
4.     m_ConnectionStatus.Value = "Voice Service Initialized";
5.     VivoxService.Instance.LoggedIn += LocalUserLoggedIn;
6.     BindToParticipantEvents();
7. }
```

Codeblock 8: Initialisierung des Vivox-Dienstes und Anmeldung des lokalen Benutzers für den Sprachchat

Zur Gewährleistung der strukturellen Konsistenz innerhalb des Projekts wird die Lobby-ID als Name des Sprachkanals verwendet.

In den Projekt-Skripten ist die Anbindung an Vivox modular über das Interface IVivoxService realisiert. Es verwaltet wichtige Ereignisse wie LoggedIn, ChannelJoined und

ParticipantAddedToChannel und ermöglicht so dynamische UI-Aktualisierungen oder Steuerungen wie das Ein-/Ausschalten des Mikrofons.

Die Klasse VoiceChatManager fungiert als zentrale Verwaltungseinheit zwischen dem Hauptsystem und Vivox. Sie übernimmt Aufgaben wie:

- Verwaltung des Benutzerstatus im Sprachkanal,
- Steuerung der Audio-Wiedergabe und
- Kanalbeitritte oder -verlassen.

Zu den wichtigsten Funktionen zählen:

- JoinPositionalChannelAsync() zum Beitreten eines Kanals und
- LeaveAllChannelsAsync() zum Verlassen aller Kanäle.

Ein wesentlicher Aspekt der Architektur ist die Kopplung der Sprachkanäle mit den Lobby-IDs. Dies garantiert die strukturelle Übereinstimmung zwischen dem Netcode for GameObjects-System und Vivox.

Zur feingranularen Steuerung der Audioübertragung stehen die Methoden MuteInputDevice() und UnmuteInputDevice() über das Interface IVivoxService zur Verfügung. Diese werden durch den VoiceChatManager aufgerufen und ermöglichen es den Nutzern, den Sprachfluss individuell zu kontrollieren.

Durch die Integration von Vivox konnte somit ein stabiles, skalierbares und mit der Mehrbenutzerstruktur kompatibles positionales Sprachkommunikationssystem realisiert werden.

4.4.2 Implementierung der Textnachrichtenübertragung zwischen Benutzer:innen

In diesem Projekt wurde zur Übermittlung von Textnachrichten zwischen Benutzer:innen in einer gemeinsamen virtuellen Umgebung ein dediziertes Modul namens NetworkMessageBoardForTablets entwickelt. Dieses Modul basiert auf den Funktionen von Netcode for GameObjects und ermöglicht in einem vernetzten Umfeld die Echtzeit-Übertragung, Anzeige und Verwaltung von Textnachrichten. Angesichts der Virtual-Reality-Natur des Projekts und der Nutzung virtueller Tablets wurde dieses Nachrichtensystem speziell für die Anzeige von Texten auf Oberflächen wie virtuellen Tablet-Displays entworfen.

Das Herzstück der Funktionalität bildet eine Datenstruktur auf Basis von NetworkList<FixedString512Bytes>, die Nachrichten sicher und synchronisiert unter den Clients speichert. Beim Senden einer Nachricht gibt der/die Benutzer:in den Text über eine

virtuelle Tastatur ein und ruft die Methode SubmitTextLocal() auf, um die Nachricht an den Server zu übermitteln. Diese Methode fügt zunächst den Benutzernamen des lokalen Spielers der Nachricht voran, formatiert den Text als FixedString512Bytes und leitet ihn dann an SubmitMessageServerRpc() weiter, um die Nachricht serverseitig zu verarbeiten und zur Nachrichtenliste hinzuzufügen. Anschließend wird die Nachricht über SubmitMessageClientRpc() an alle verbundenen Clients gesendet, sodass sie im jeweiligen UI angezeigt werden kann:

```
1. public void SubmitTextLocal()
2. {
3.    string text = GlobalNonNativeKeyboard.instance.keyboard.text;
4.    string textToSend = $"<b>{XRINetworkPlayer.LocalPlayer.playerName}</b>:<br><br>{text}";
5.
6.    if (textToSend.Length > m_MaxCharacterCount)
7.    {
8.       textToSend = textToSend.Substring(0, m_MaxCharacterCount);
9.    }
10.
11.    FixedString512Bytes newText = new FixedString512Bytes(textToSend);
12.    SubmitMessageServerRpc(newText);
13. }
```

Codeblock 9: Lokale Verarbeitung und Übergabe einer formatierten Textnachricht zur serverseitigen Weiterleitung im vernetzten Nachrichtensystem

Da das Projekt auf einem zentralisierten Eigentumsmodell in Netcode basiert, darf ausschließlich der Server Änderungen an der NetworkList vornehmen. Entsprechend wird die Nachricht erst nach Prüfung und Verarbeitung durch den Server in die Liste aufgenommen. Für jede Nachricht wird ein UI-Objekt auf Basis eines vordefinierten Prefabs erzeugt und im angegebenen Viewport dargestellt. Die Methode CreateText() ist verantwortlich für die Darstellung der empfangenen Nachrichten im UI einschließlich Zeitstempel:

```
1. void CreateText(string text)
2. {
3.    Instantiate(m_MessagePrefab, m_ContentViewport)
4.       .GetComponent<MessageText>()
5.       .SetMessage(text, DateTime.Now.ToString("h:mm tt"));
6.
7.    if (m_ContentViewport.childCount > m_MaxMessageCount)
8.    {
9.       Destroy(m_ContentViewport.GetChild(0).gameObject);
10.    }
11. }
```

Codeblock 10: Methode zur UI-seitigen Darstellung empfangener Nachrichten inklusive Zeitstempel und automatischem Nachrichtenlimit

Dieses Nachrichtensystem ermöglicht nicht nur eine nicht-verbale Kommunikation in der Multiuser-Umgebung, sondern ist auch strukturell nahtlos mit den Netzwerkkomponenten und dem UI-System von Unity integriert. Außerdem wurde es so konzipiert, dass bei einem Verbindungsabbruch zum Server alle vorhandenen Nachrichten gelöscht werden und bei einer erneuten Verbindung nicht wiederhergestellt werden – dies wird über die Methode ConnectedToNetwork() gesteuert.

Diese einfache, aber effektive Struktur bietet eine schnelle und zuverlässige Textkommunikation für verschiedene Szenarien virtueller Meetings – ganz ohne komplexe Infrastruktur oder externe Dienste.

4.4.3 Vergleichstabelle der Kommunikationsfunktionen

Die folgende Tabelle bietet einen analytischen Vergleich zwischen den beiden Hauptkommunikationsfunktionen des Projekts: dem sprachbasierten Voice-Chat über Vivox und dem intern entwickelten System für den Textnachrichtenaustausch. Der Vergleich erfolgt anhand von Kriterien wie Kommunikationsprotokoll, Datentyp, Zeitverhalten, Darstellung im UI sowie Abhängigkeit von externen Servern.

Diese Gegenüberstellung erleichtert das Verständnis für das Design der Kommunikationsarchitektur des Projekts und zeigt die funktionalen Unterschiede beider ergänzenden Kommunikationswege auf. Sie liefert zudem eine rationale Grundlage für die gezielte Auswahl in verschiedenen Anwendungsszenarien.

Nr.	Merkmal	Textnachrichten (intern)	Sprachkommunikation (Vivox)
1	Kommunikationsprotokoll	Netcode for GameObjects	Proprietäres Protokoll (Vivox)
2	Datentyp	Text	Audio
3	Zeitverhalten	Echtzeit (über RPCs)	Echtzeit (über Vivox)
4	Darstellung im UI	Anzeige auf Tablets & Monitoren	Indikatoren über Mikrofonstatus im UI
5	Abhängigkeit von externen Servern	Keine (vollständig intern)	Ja (Vivox-Server)

Tabelle 1: Die Tabelle stellt die wichtigsten Unterschiede zwischen dem internen Textnachrichtensystem und der Sprachkommunikation über Vivox heraus.

4.5 NPC-Modul und Künstliche Intelligenz (Convai)

In diesem Projekt wird die Plattform Convai eingesetzt, um eine natürliche und intelligente Interaktion mit nicht-spielbaren Charakteren (NPCs) zu ermöglichen. Diese Plattform basiert auf dialogorientierter künstlicher Intelligenz und erlaubt es, Charaktere zu entwickeln, die in

der Lage sind, in Echtzeit auf die Sprachbefehle der Nutzer zu reagieren. Die Integration von Convai in Unity umfasst die Installation des offiziellen Plugins sowie die Konfiguration von API-Schlüsseln direkt am Charakterobjekt.[37] Nach der Aktivierung des Spracherkennungsmoduls innerhalb der VR-Umgebung wird die Spracheingabe des Nutzers erfasst und über eine REST-API an die Server von Convai übermittelt. Dort wird die Sprachnachricht analysiert, in Text umgewandelt und die entsprechende Antwort generiert und zurückgesendet.

Im aktuellen Projekt beginnt die Interaktion mit einem NPC nur dann, wenn sich der Nutzer innerhalb einer bestimmten Entfernung zum Charakter befindet. Sobald das Gespräch initiiert wird, erzeugt das Sprachmodell des NPCs auf Basis der erkannten Bedeutung und Konversation ein kontextgerechtes Antwortverhalten. Diese Eigenschaften ermöglichen es, dass die Interaktion zwischen Nutzer und NPC in der virtuellen Umgebung als natürlich, flüssig und menschlich wahrgenommen wird.

4.5.1 Logik und Verhalten des NPC

Der in diesem Projekt eingesetzte nicht-spielbare Charakter (NPC) wurde mithilfe der Plattform Convai entworfen und fungiert als interaktiver Agent innerhalb der Virtual-Reality-Umgebung. Ziel des NPCs ist es, einen menschlichen Nutzer zu simulieren, der in der Lage ist, natürliche Sprache zu verstehen und in Form eines Echtzeit-Dialogs zu antworten. Diese Fähigkeiten sind so konzipiert, dass ein kontextbezogener und natürlicher Dialog zwischen Nutzer und NPC ermöglicht wird. Das dialogorientierte Verhalten des Charakters wird durch eine Cloud-basierte künstliche Intelligenz gesteuert, wodurch eine flüssige Interaktion in der VR-Umgebung ohne lokale Rechenbelastung realisierbar ist.[38]

Auf funktionaler Ebene ist der NPC in der Lage, den Beginn eines Gesprächs zu erkennen, den gesprochenen Inhalt des Nutzers zu analysieren und daraufhin eine semantisch passende und flüssige Antwort in Audioform zu generieren. Dieser Prozess umfasst die Übertragung der aufgezeichneten Sprache an die Convai-Server, die textuelle Verarbeitung mittels maschinellen Lernens sowie die Rückgabe einer generierten Antwort, die vom NPC hörbar wiedergegeben wird. Eine der wesentlichen Stärken des Systems liegt in der Fähigkeit zur Kontextverarbeitung und der Aufrechterhaltung eines konsistenten Dialogs über mehrere Interaktionen hinweg. Dies führt zu einer menschlicheren und überzeugenderen Nutzererfahrung.

Was das Verhaltensdesign betrifft, so kann der NPC – abhängig vom Gesprächsinhalt und dem definierten Kontext – einfache sprachliche oder gestische Reaktionen zeigen. In der aktuellen Projektumsetzung beschränkt sich das Verhalten des NPCs auf verbale Antworten. Funktionen wie Bewegung oder physische Interaktion mit Objekten wurden nicht aktiviert. Die auf Convai basierende Architektur bietet jedoch die Grundlage für eine spätere Erweiterung um komplexere Verhaltensweisen. Diese Struktur erlaubt es den Entwickler:innen, ohne explizite Regeldefinitionen für jede einzelne Interaktion, auf die sprachlogische Intelligenz der Modelle

zurückzugreifen. Dadurch kann das Verhalten des NPC dynamisch und mit minimalem Programmieraufwand realisiert werden.

4.5.2 Integration von Convai mit den eingebetteten Monitoren im Konferenzraum

Zur Verbesserung der Nutzererfahrung und zur Förderung der Transparenz von Interaktionen in der virtuellen Realität wurden in diesem Projekt zwei Monitore im virtuellen Konferenzraum direkt mit dem Kommunikationssystem des intelligenten Agenten (NPC) verbunden. Diese Monitore dienen als visuelle Schnittstelle zwischen den Nutzern und dem NPC. Einer der Monitore (rechts) zeigt die von den Nutzern eingegebenen Fragen an, während der andere Monitor (links) die Antworten des NPC darstellt. Dieses zweigeteilte Design ermöglicht es allen anwesenden Nutzern, den Verlauf des Dialogs zwischen dem Nutzer und dem intelligenten Agenten in Echtzeit nachzuvollziehen und sorgt so für ein interaktives und transparentes Erlebnis.

Zur Umsetzung dieses Mechanismus wurden zwei separate, dedizierte Skripte verwendet: Das Skript NetworkMessageBoardForPlayer ist dafür verantwortlich, alle über die Texteingabe eingegebenen Fragen der Nutzer mittels ServerRpc an den Server zu senden und anschließend über ClientRpc an alle Clients zu verteilen, sodass diese auf dem rechten Monitor angezeigt werden. Das Skript NetworkMessageBoardForNPC übernimmt entsprechend die Darstellung der Antworten des intelligenten NPC (Convai) auf dem linken Monitor. In beiden Skripten wird zur Speicherung und Synchronisation der Nachrichten das Objekt NetworkList<FixedString512Bytes> verwendet. Dies stellt sicher, dass alle Clients stets eine konsistente und synchronisierte Version der übermittelten Nachrichten erhalten.

Die empfangenen Nachrichten werden schließlich mithilfe von benutzerdefinierten UI-Elementen wie MessageText auf den Monitoren angezeigt – mit einer ansprechenden Formatierung inklusive Sendername und Zeitstempel. Zur Begrenzung der Anzeige wurden zusätzlich Einschränkungen in Form von m_MaxMessageCount und m_MaxCharacterCount implementiert. Diese Umsetzung macht die Interaktion mit dem NPC nicht nur für alle sichtbar, sondern steigert auch signifikant die Qualität der Nutzererfahrung sowie das Gefühl von Präsenz in einer realitätsnahen Besprechung.

4.6 Benutzeroberfläche und Design

In diesem Projekt spielt die Benutzeroberfläche (User Interface – UI) sowie das interaktive Design eine zentrale Rolle bei der Verbesserung des Nutzererlebnisses in der virtuellen Realität. Da sich die Nutzer in einer dreidimensionalen, mehrbenutzerfähigen Umgebung bewegen – mit Funktionen wie der Interaktion mit NPCs, dem Anzeigen von Nachrichten und der Nutzung

integrierter Steuerungsmenüs – muss das UI sowohl visuell ansprechend als auch funktional gestaltet sein, um Verwirrung zu vermeiden und sich nahtlos in die virtuelle Realität einzufügen.

Zu diesem Zweck kommt eine breite Palette an interaktiven Elementen zum Einsatz, darunter schwebende Menüs, virtuelle Tablets, Informationsmonitore und Controller-basierte Bedienelemente. Diese Komponenten wurden nicht nur mithilfe des XR Interaction Toolkits in Unity entworfen und verwaltet, sondern auch mit Netcode for GameObjects, dem Audiosystem (Vivox) und dem Konversationsmotor des NPCs (Convai) integriert. Daher umfasst das UI-Design nicht nur das visuelle Erscheinungsbild der grafischen Elemente, sondern auch deren Interaktionslogik, Echtzeitverhalten und Kompatibilität mit Oculus-Controllern.

Im weiteren Verlauf dieses Kapitels werden die wichtigsten UI-Komponenten – einschließlich des Designs der Hauptmenüs, der Steuerungstablets, der Antwortmonitore und der Interaktionen mit der Umgebung sowie den NPCs – schrittweise erläutert. Ziel dieses Abschnitts ist es, einen umfassenden Überblick über die Gestaltung und Implementierung der Benutzeroberfläche in diesem Projekt zu geben, wobei der Fokus auf Einfachheit, Effizienz und einem positiven Nutzererlebnis in der VR-Umgebung liegt.

4.6.1 Interaktion mit dem NPC

Bei der Gestaltung der Interaktionen zwischen Nutzer und nicht-spielbarem Charakter (NPC) in diesem Projekt standen Nutzerzentrierung und Immersion in der virtuellen Realität (VR) im Vordergrund. Das Interaktionssystem wurde so entworfen, dass das Gesprächserlebnis mit dem NPC für die Nutzer möglichst natürlich und intuitiv wirkt.

Die Interaktion beginnt damit, dass sich der Nutzer im Sichtfeld des NPC befindet und sich ihm nähert. Sobald der Nutzer in den effektiven Interaktionsradius eintritt, dreht sich der virtuelle Charakter automatisch zur Position des Nutzers und richtet seinen Blick direkt auf ihn aus. Diese Funktion basiert auf den über das VR-Headset erfassten Tracking-Daten zu Position und Kopfrotation und simuliert somit ein Gespräch auf Augenhöhe wie in der realen Welt.

Abbildung 9: Darstellung eines KI-gesteuerten NPC-Charakters im virtuellen Meetingraum, bereit zur Interaktion mit den Nutzern über das Convai-Konversationssystem

Das Gespräch wird über den rechten Controller des Headsets eingeleitet. Hält der Nutzer die A-Taste gedrückt, beginnt das System mit der Tonaufnahme. Solange die Taste gedrückt bleibt, wird die Spracheingabe aufgezeichnet. Sobald die Taste losgelassen wird, wird die gesammelte Audiodatei zur Verarbeitung an den KI-basierten Dienst gesendet, der unmittelbar eine textliche und gesprochene Antwort zurückliefert. Die Antwort des NPC wird akustisch wiedergegeben, während die Frage des Nutzers sowie die Antwort des NPC gleichzeitig in einer schwebenden Box neben dem Charakter angezeigt werden. Diese Box bleibt stets im Sichtfeld des Nutzers und verstärkt das visuelle Interaktionserlebnis.

In einem weiteren Schritt werden alle Fragen und Antworten über das Netzwerksynchronisierungssystem an andere Nutzer weitergeleitet. Die Texte erscheinen auf zwei Bildschirmen im virtuellen Konferenzraum: Einer zeigt die Fragen der Nutzer, der andere die vom NPC generierten Antworten. Dieses Design ermöglicht es allen anwesenden Nutzern, die Gespräche mitzuverfolgen und fördert den Gruppenzusammenhalt im Meeting.

Neben dem direkten Dialog können Nutzer auch individuelle Einstellungen am NPC vornehmen. Mit einem Druck auf die X-Taste des linken Controllers wird ein schwebendes Einstellungsfenster (Floating Dialog Box) geöffnet, das verschiedene Optionen im Bereich Audio und Benutzeroberfläche bietet.

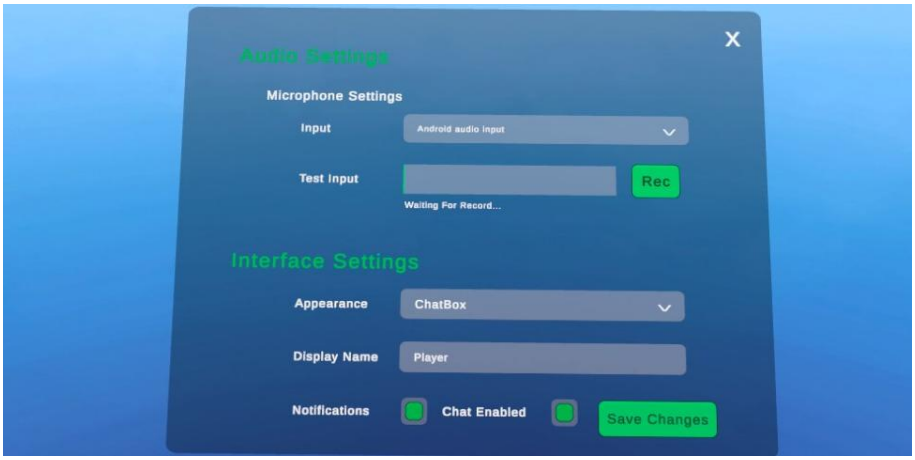

Abbildung 10: Einstellungsmenü für Mikrofoneingang und Chat-Oberfläche mit Funktionen zum Audiotest und zur Anpassung der Chat-Interaktion

Im Abschnitt Audio Settings können Nutzer das Eingabegerät auswählen und dessen Funktionalität testen. Im Abschnitt Interface Settings lässt sich unter anderem die Darstellungsform der Nachrichten (Appearance), der Anzeigename (Display Name) sowie die Aktivierung bzw. Deaktivierung von Textbenachrichtigungen konfigurieren. Zudem steht ein Speichern-Button zur Verfügung, um Änderungen sofort wirksam werden zu lassen.

Diese Interaktionsstruktur ist nicht nur benutzerfreundlich und intuitiv gestaltet, sondern auch technisch vollständig mit der Netzwerkinfrastruktur des Projekts kompatibel und unterstützt nahtlos die Architektur von Netcode for GameObjects sowie die Funktionen des Meta Quest 2.

4.6.2 Interaktion mit Benutzeroberflächen und In-Game-Menüs

Im Rahmen dieses Virtual-Reality-Projekts wurde die Benutzeroberfläche (UI) so gestaltet, dass die Nutzer einen einfachen und intuitiven Zugang zur Plattform sowie einen reibungslosen Ablauf von der Anmeldung bis zur Teilnahme an virtuellen Meetings erleben. Die Benutzeroberflächen sind als schwebende und interaktive Menüs konzipiert und werden über den linken Controller des VR-Headsets aufgerufen. Der Interaktionsprozess beginnt mit der Eingabe eines Anzeigenamens und endet mit dem Beitritt zum Konferenzraum sowie der Konfiguration individueller Einstellungen.

Im ersten Schritt muss jeder Nutzer – unabhängig davon, ob er als Server oder Client auftritt – seinen Display Name festlegen und diesen im Eingabefeld neben der Konferenztür bestätigen.

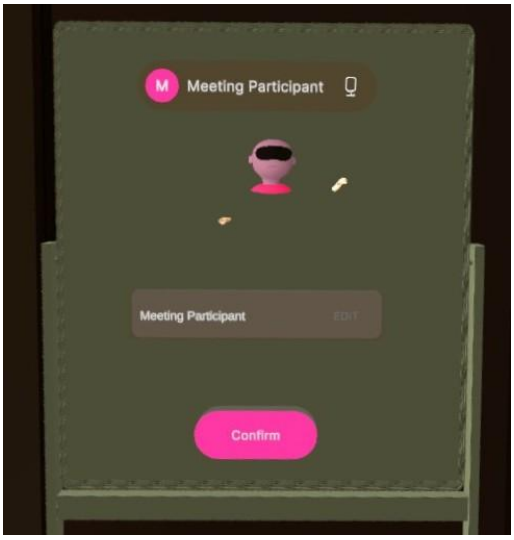

Abbildung 11: Benutzeroberfläche zur Eingabe eines persönlichen Anzeigenamens vor dem Betreten eines Meeting-Raums

Dieser Name dient als visuelle Identität während der gesamten Sitzung und ist für alle anderen Teilnehmenden sichtbar. Nach der Eingabe des Namens teilt sich der Ablauf je nach Rolle des Nutzers auf.

Ein Nutzer mit der Rolle „Server“ hat zwei Möglichkeiten, einen neuen Raum zu erstellen: Durch Klicken auf die Schaltfläche Quick Join wird automatisch ein öffentlicher Raum mit einem Standardnamen und einem eindeutigen Code erstellt. Alternativ kann der Nutzer über die Option New Room benutzerdefinierte Parameter (z. B. Privatsphäre, maximale Teilnehmerzahl) einstellen und den Raum durch Create erstellen.

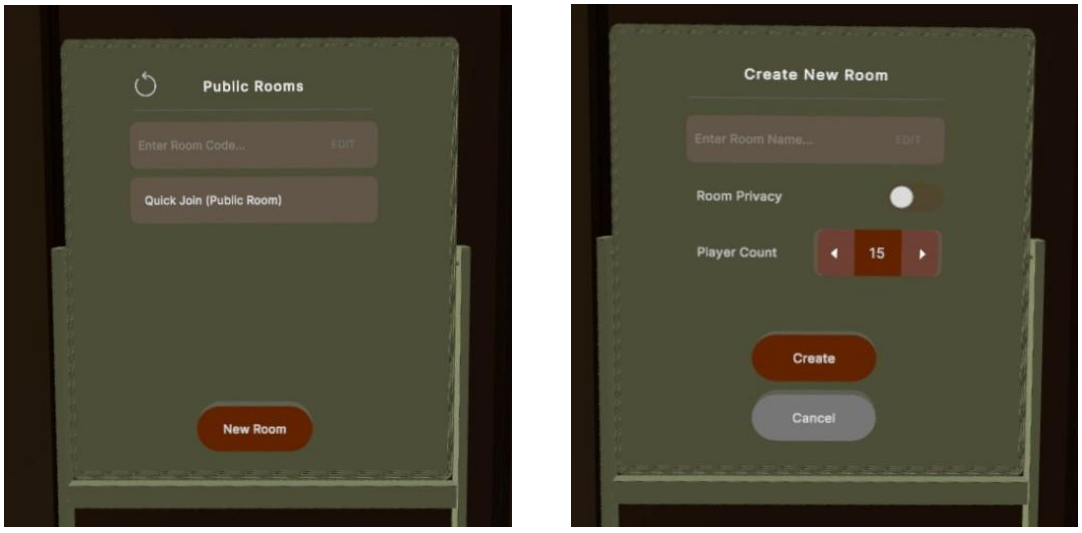

Abbildung 12: Menü zur schnellen Teilnahme an öffentlichen Räumen oder zum Beitritt über einen Raumcode (links), Eingabemaske zur Erstellung eines neuen Raums mit Optionen für Raumname, Privatsphäre und maximale Teilnehmerzahl (rechts)

Die erstellten Räume erhalten einen sechsstelligen eindeutigen Code, der an zwei Stellen angezeigt wird: Erstens auf dem Panel an der Innenwand des Konferenzraums.

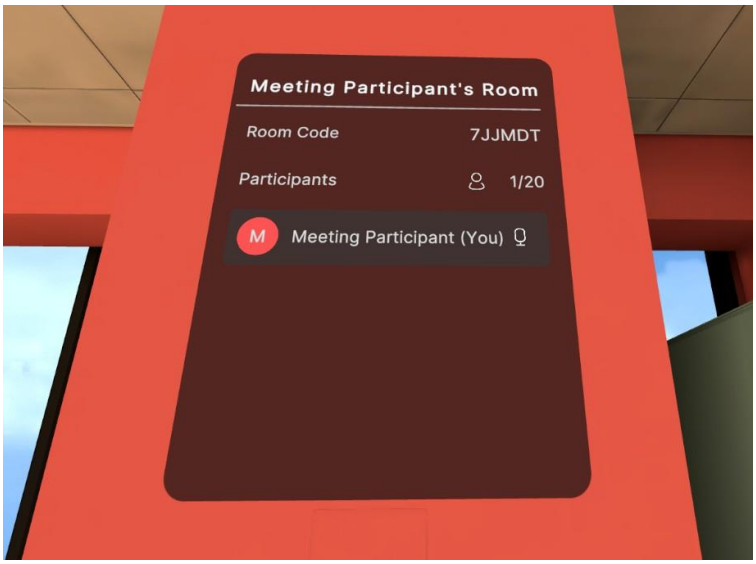

Abbildung 13: Integriertes Informationspanel innerhalb der VR-Umgebung mit Raumcode und Teilnehmerliste

Zweitens im ersten Tab des schwebenden In-Game-Menüs, das über die Menütaste auf dem linken Controller geöffnet wird.

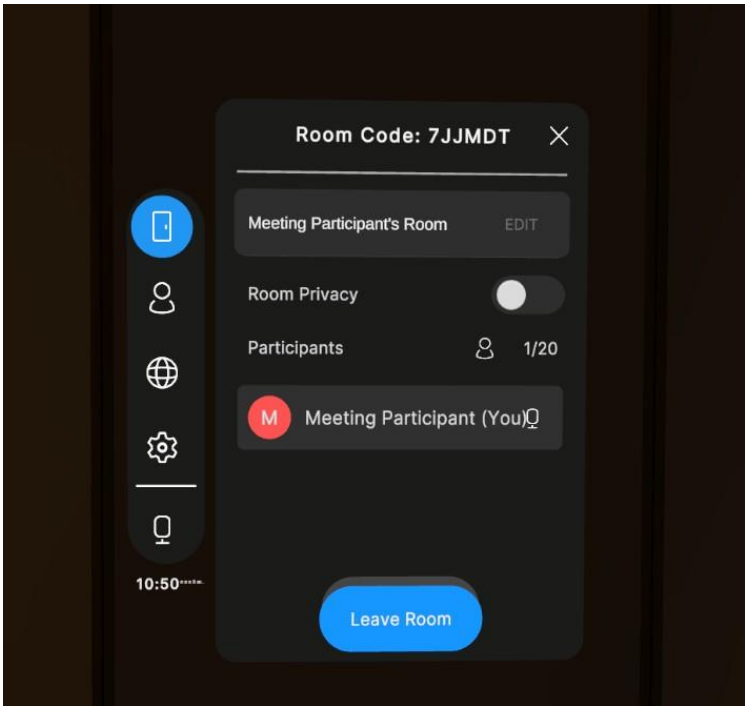

Abbildung 14: Anzeige der aktuellen Raumdaten inklusive Raumcode, Name, Raumtyp und Teilnehmerzahl, mit der Möglichkeit, einige davon zu ändern

Dieser Raumcode wird jedoch nicht automatisch an die Clients übermittelt und muss extern (z. B. per E-Mail, unternehmensinternem Messenger oder anderen Kommunikationsmitteln) weitergegeben werden.

Auch für Client-Nutzer ist der Einstieg ähnlich. Nach Eingabe des Namens erscheint ein Panel mit einer Liste verfügbarer öffentlicher Räume. Ist ein Raum öffentlich, kann er direkt aus der Liste ausgewählt und betreten werden.

Abbildung 15: VR-basierte Benutzeroberfläche zur Auswahl, Anlage oder Teilnahme (per Code) an aktiven öffentlichen Meetings

Ist der Raum privat, erscheint er nicht in der Liste, und der Nutzer muss den Code manuell eingeben, um dem Raum beizutreten – vorausgesetzt, er hat diesen vorher vom Host erhalten.

Nach dem Beitritt zum Raum steht allen Teilnehmenden das schwebende Seitenmenü zur Verfügung, das über den linken Controller aufgerufen wird. Dieses Menü umfasst mehrere Tabs mit unterschiedlichen Funktionen:

- Der erste Tab zeigt den Raumcode und bietet die Möglichkeit, den Raum zu verlassen (Abbildung 14),
- der zweite Tab erlaubt die Konfiguration des Avatars, wie z. B. Farbe oder Anzeigename,

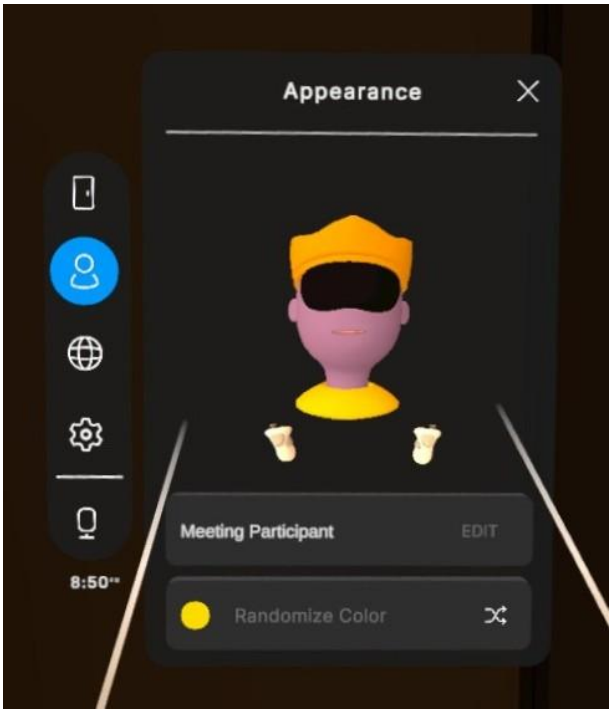

Abbildung 16: Möglichkeit zur Individualisierung des eigenen Avatars durch Auswahl von Farbe und Anzeigename

- der dritte Tab dient der Raumsuche oder -erstellung,

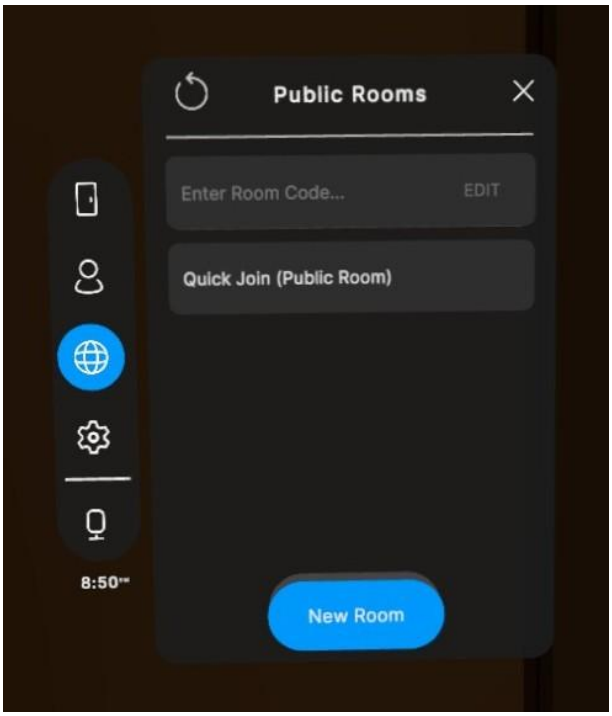

Abbildung 17: Menü zur Anzeige verfügbarer öffentlicher Räume sowie Option zur Erstellung eines neuen Raums

- der vierte Tab umfasst Audio- und Bewegungseinstellungen sowie eine Option zum Beenden der Anwendung,

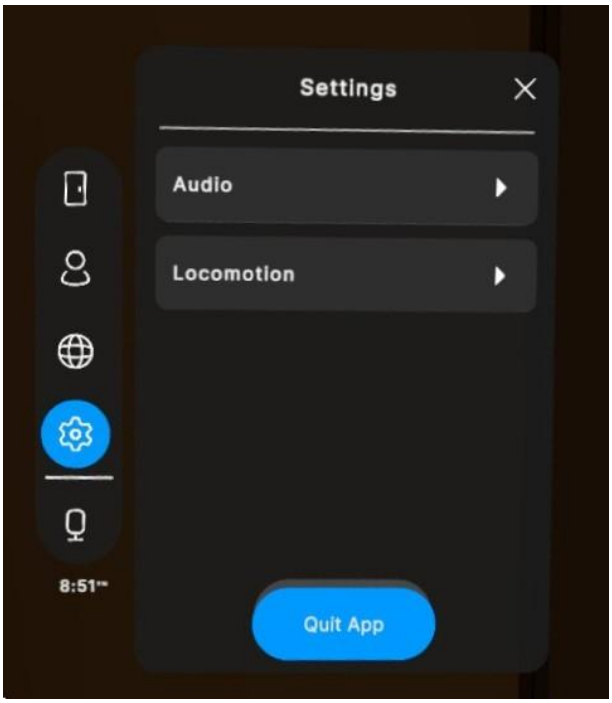

Abbildung 18: Einstellungsmenü zur Konfiguration von Audio- und Bewegungsoptionen für ein optimiertes VR-Erlebnis

- und das Mikrofon-Symbol am unteren Rand erlaubt die Stummschaltung bzw. Aktivierung des Mikrofons.

Ein wichtiger Aspekt der Architektur ist die Option zur rollenbasierten Anpassung im Falle einer kommerziellen Nutzung. In zukünftigen Versionen der Anwendung könnten für Client-Versionen bestimmte Schaltflächen – etwa New Room oder Quick Join – deaktiviert oder vollständig aus der UI entfernt werden. Diese Maßnahme würde der rollenbasierten Zugriffskontrolle in Unternehmen dienen und die Unterscheidung zwischen Gastgebern, Gästen, Moderatoren etc. erleichtern.

Diese hierarchisch strukturierte und interaktive UI-Architektur bietet zusammen mit den flexiblen Konfigurationsmöglichkeiten ein benutzerfreundliches, sicheres und immersives VR-Erlebnis. Gleichzeitig entspricht das Design den gängigen UX-Prinzipien für Virtual Reality und unterstützt realistische Multiuser-Szenarien.

4.7 Test-, Monitoring- und Datenerfassungsmodul

In einem mehrbenutzerfähigen Virtual-Reality-System ist die genaue Überwachung des Netzwerkstatus und des Nutzerverhaltens von entscheidender Bedeutung, da Faktoren wie Verbindungsstabilität, Latenz, Jitter und Datenübertragungsraten die Benutzererfahrung direkt beeinflussen können. Zu diesem Zweck wurde im Rahmen dieses Projekts ein spezielles Modul für Performance-Tests, Netzwerk-Monitoring und die Erfassung quantitativer Daten zur Laufzeit entworfen und implementiert.

Dieses Modul besteht aus mehreren komplementären Komponenten, die gleichzeitig auf Server- und Client-Seite aktiv sind und präzise Informationen über den Zustand der Netzwerkverbindungen und das Verhalten der Benutzer sammeln.

Das Monitoring-Subsystem zeichnet kontinuierlich Kennwerte wie Latenz zwischen Client und Server, verwendeter Transporttyp, Jitter, Zeitunterschiede zwischen Netzwerk-Ticks und die Anzahl verbundener Benutzer auf und speichert diese Informationen im CSV-Format. Diese Dateien dienen als Grundlage für quantitative Auswertungen nach Nutzertests oder zur Analyse der Systemstabilität unter unterschiedlichen Bedingungen.

Das Modul besteht aus drei Hauptkomponenten:

- NetworkStatsReporter für die Datensammlung,
- CustomNetworkLogger zur Ereignisverarbeitung und Datenverwaltung,
- NetworkLatencyCSVLogger zur Speicherung der Informationen in CSV-Dateien.

4.7.1 Funktionale Tests und Evaluation

Der funktionale Testabschnitt des Systems wurde entwickelt, um die korrekte Ausführung der Logik, Interaktionen und Systemreaktionen auf Benutzereingaben zu prüfen. Dabei wird das Verhalten der Clients in unterschiedlichen Szenarien wie dem Beitritt zu einem Raum, dem Aufbau einer Sprachverbindung, der Interaktion mit NPCs, dem Versenden von Nachrichten sowie dem Verlassen des Netzwerks evaluiert. Diese Tests wurden sowohl manuell durchgeführt als auch über vordefinierte Ereignisse im Code dokumentiert.

Jedes Mal, wenn ein neuer Client dem Netzwerk beitritt, fordert der Server mittels der Methode OnClientConnected automatisch grundlegende Informationen vom Client an. Diese umfassen unter anderem die Benutzer-ID, den Namen, das Gerät, die lokale Zeit, den aktuellen Netzwerk-Tick und den Verbindungsstatus. Die gesammelten Informationen werden in einer NetworkStats-Struktur gebündelt und an den Server gesendet.

Zusätzlich werden diese Daten in regelmäßigen Abständen – typischerweise jede Sekunde – sowohl auf Server- als auch auf Client-Seite aktualisiert und über das Netzwerk verteilt. Auf

Serverseite werden sie im Cache clientStatsCache gespeichert und gleichzeitig in eine CSV-Datei geschrieben.

Dieses Verfahren ermöglicht es dem Entwickler, bei Problemen wie Verbindungsabbrüchen, erhöhter Latenz oder Netzwerkinstabilität die genaue Ursache zu identifizieren und gezielt zu analysieren. Darüber hinaus können die gespeicherten Daten auch zur Bewertung der Nutzererfahrung, zum Vergleich verschiedener Testläufe oder zur Erstellung analytischer Dashboards verwendet werden.

4.7.2 Technische Leistungstests, Monitoring und quantitative Datenerfassung

Zur präzisen Bewertung der Netzwerkleistung und der Kommunikationsqualität zwischen Clients und Server wurde in diesem Projekt ein umfassendes Test- und Monitoring-System konzipiert und implementiert. Dieses System besteht aus drei zentralen Komponenten: dem zentralen Logging-Modul (CustomNetworkLogger), dem Echtzeitanalysemodul (NetworkStatsReporter) und dem Persistenzmodul zur CSV-Datenspeicherung (NetworkLatencyCSVLogger). Diese Struktur ermöglicht es dem Entwickler, jederzeit auf analysierbare statistische Daten über Latenzen, Paketverluste, Verbindungsstatus und Netzwerkstabilität zuzugreifen.

a) Skript CustomNetworkLogger.cs: Das zentrale Skript CustomNetworkLogger übernimmt die Sammlung, Analyse und Verwaltung der historischen Netzwerkdaten. In regelmäßigen Intervallen werden Kennwerte wie aktuelle Latenz, durchschnittliche Latenz, Jitter (Standardabweichung der Latenz), Netzwerk-Tick-Differenz, lokale Zeit, Serverzeit, Verbindungsstatus, Anzahl verbundener Clients und genutzter Transporttyp erfasst. Diese Daten werden in der Struktur NetworkStatsRecord gespeichert und in einer konfigurierbaren Liste abgelegt.

Eine besonders relevante Funktion dieses Skripts ist das Batch Logging, bei dem die Daten nicht sofort, sondern gesammelt (z. B. alle 60 Sekunden) in eine CSV-Datei geschrieben werden. Dadurch wird die I/O-Belastung reduziert und die Effizienz erhöht.

Zudem wird ein HashSet verwendet, um doppelte Ereignisse wie Verbindungsaufbau oder -abbruch zu verhindern – durch temporäres Speichern eindeutiger Ereignis-IDs.

Eine weitere fortgeschrittene Funktion ist der Einsatz von Reflection, um nicht-öffentliche Eigenschaften wie IsApproved der Klasse NetworkClient auszulesen. Dies erlaubt eine tiefere Analyse des Verbindungsstatus einzelner Clients. Außerdem unterstützt das Skript die getrennte Erfassung von Statistiken für jeden Client im Server-Modus.

b) Skript NetworkStatsReporter.cs: Das Skript NetworkStatsReporter ist ein zentrales Element im Monitoring-Prozess und dient der kontinuierlichen Sammlung von Netzwerkstatistiken auf Client-Seite. Diese beinhalten u. a. Client-ID, Bildrate (FPS), Benutzername, Latenz, Jitter, Tick-Differenz zum Server sowie die durchschnittliche Round-Trip-Time (RTT).

Sobald das Netzwerk gestartet ist, wird bei Client-Geräten ein Routineprozess mit festen Intervallen (reportingInterval) ausgeführt, der die lokalen Messdaten erfasst und mittels ServerRpc an den Server überträgt. Dort werden die empfangenen Daten in einem Dictionary zwischengespeichert und gleichzeitig in Form einzelner CSV-Einträge gespeichert.

Die Datenstruktur basiert auf der NetworkStats-Klasse, welche das Interface INetworkSerializable implementiert und somit eine sichere und effiziente Datenübertragung gewährleistet.

Die Kommunikation erfolgt über folgende Logik: Nach der Erhebung aller relevanten Messwerte (lokale Zeit, FPS, Systemdaten, Name, Netzwerkmetriken) wird ein NetworkStats-Objekt erstellt und per Methode SubmitClientStatsServerRpc() an den Server gesendet. Der Server aktualisiert daraufhin die Datenbank und speichert sie für spätere Analysen dauerhaft in einer CSV-Datei. Zusätzlich wird bei Verbindung eines neuen Clients sofort ein Datenanforderungs-RPC ausgelöst.

Dieses Design ermöglicht eine präzise Bewertung der Verbindungsqualität jedes einzelnen Nutzers. Im Fall von hohen Latenzen oder starkem Jitter können durch die erzeugten Logs potenzielle Netzwerk- oder Hardwareprobleme gezielt identifiziert und analysiert werden.

c) Skript NetworkLatencyCSVLogger.cs: Zur dauerhaften Speicherung der Netzwerk-Statistiken wird das Skript NetworkLatencyCSVLogger verwendet. Es schreibt strukturierte CSV-Dateien mit Spalten für Ereigniszeitpunkt, Client-ID, Server-/Client-Rolle, Ereignistyp (z. B. Verbindungsaufbau, -abbruch), lokale Zeit, Serverzeit, Latenz, RTT, Jitter und verwendeter Transporttyp.

Eine zusätzliche Funktion dieses Skripts ist das Filtern irrelevanter oder fehlerhafter Einträge. Die Methode ShouldSkipRecord() prüft, ob ein Datensatz gültige und aussagekräftige Inhalte enthält. Falls nicht, wird er von der Speicherung ausgeschlossen, was zur Datenqualität und Übersichtlichkeit beiträgt.

Dieses Netzwerk-Logging- und Monitoringsystem bietet ein leistungsstarkes Werkzeug zur Analyse des Systemverhaltens unter realen Bedingungen. Es unterstützt den Entwickler während der finalen Testphase dabei, potenzielle Schwachstellen wie steigende Latenzzeiten, instabile Verbindungen oder fehlende Netzwerkstabilität zu identifizieren und gezielte Optimierungen vorzunehmen. Solch eine Infrastruktur ist für produktionsreife Mehrbenutzersysteme nicht nur essenziell, sondern auch aus wissenschaftlicher Perspektive von großem Wert.

5. Datenbewertung

Der Abschnitt zur Datenbewertung widmet sich der Analyse, Interpretation und Reflexion der während der Projektumsetzung erhobenen Daten. Ziel ist es, ein klares Bild über die Systemleistung unter realen Bedingungen zu vermitteln und die im Vorfeld definierten Forschungsfragen zu beantworten. Die Auswertungen basieren sowohl auf quantitativen als auch auf qualitativen Daten und dienen der Identifikation von Stärken und Schwächen des entwickelten Systems.

5.1 Einleitung und Bewertungsrahmen

In diesem Abschnitt der Arbeit wird das allgemeine Bewertungsframework vorgestellt. Um eine valide und präzise Systembewertung zu gewährleisten, kamen zwei unterschiedliche Datentypen zum Einsatz: quantitative Daten und qualitative Daten.

Die quantitativen Daten wurden direkt im Unity-Umfeld mittels C#-Skripten erfasst und durch statistische Verfahren wie den t-Test ausgewertet. In diese Auswertung flossen unter anderem Parameter wie Netzwerklatenz, Einfluss der Nutzerzahl auf die Systemperformance sowie Versandzeiten von Nachrichten ein.

Die qualitativen Daten hingegen wurden über eine Online-Umfrage mittels Google Forms erhoben. Ziel war es, Einblicke in die Nutzererfahrungen hinsichtlich Benutzerfreundlichkeit, Interaktionsqualität und allgemeine Zufriedenheit mit dem System zu gewinnen. Die qualitative Analyse erfolgt separat und dient der Bewertung der User Experience (UX).

Das gesamte Bewertungsmodell orientiert sich an drei definierten Forschungsfragen und zielt darauf ab, eine umfassende, valide und praxisrelevante Analyse zur Verbesserung der Gesamtqualität des Projekts zu liefern.

5.2 Quantitative Datenanalyse

In diesem Abschnitt werden die im Verlauf der Projektimplementierung gesammelten quantitativen Daten systematisch ausgewertet. Das Hauptziel besteht darin, eine objektive und datengestützte Bewertung der Systemleistung in Bezug auf Aspekte wie Netzwerklatenz, Jitter, Nutzereffekte und Verbindungsstabilität durchzuführen. Die Analyse erfolgt unter Einsatz statistischer Methoden, um die Aussagekraft der Ergebnisse zu erhöhen.

5.2.1 Methodik der quantitativen Datenanalyse

Zur präzisen Auswertung der im Projekt erhobenen quantitativen Daten wurde in der Unity-Umgebung ein spezielles Analysetool mit dem Namen NetworkDataAnalyzer.cs entwickelt. Die Hauptaufgabe dieses Skripts besteht darin, die in CSV-Dateien (insbesondere LatencyHistory.csv) gespeicherten Messdaten auszulesen und zu analysieren. Diese Dateien enthalten essenzielle Informationen wie die Round-Trip Time (RTT), die Kommunikationslatenz sowie die Höhe der Jitter-Werte zwischen Client und Server.

Das Skript verarbeitet mehrere CSV-Dateien, die unter verschiedenen Testbedingungen – etwa mit variierender Nutzeranzahl und unterschiedlichen netzwerkbezogenen Aktivitäten – generiert wurden. Der Analyseprozess umfasst die Berechnung zentraler statistischer Kennwerte für RTT, darunter Mittelwert (Mean), Median, Minimum, Maximum und Standardabweichung (Standard Deviation). Darüber hinaus identifiziert das Skript durch detaillierte Untersuchung der Jitter-Werte Phasen mit instabiler Netzwerkverbindung und ermöglicht den Vergleich der Netzwerkleistung unter verschiedenen Bedingungen.

Ein weiteres zentrales Merkmal des Skripts ist die Erkennung und Protokollierung von Verbindungs- und Trennungsereignissen der Nutzer. Diese Informationen ermöglichen präzise Analysen zur Stabilität der Verbindung und helfen, zeitliche Muster oder Konstellationen zu identifizieren, die zu signifikanten Leistungseinbrüchen im Netzwerk geführt haben.

Die durch das Skript generierten Analyseergebnisse dienen als Grundlage für die Beantwortung der Forschungsfragen sowie zur fundierten Bewertung der technischen Leistungsfähigkeit der Netzwerkinfrastruktur innerhalb des VR-Systems.

5.2.2 Statistische Analyse

Netzwerklatenz (Latency): Die umfassende Datenanalyse zeigt, dass die durchschnittliche Netzwerklatenz über alle sechs Test-Sitzungen hinweg bei 122 Millisekunden lag. In der ersten Sitzung wurde ein Mittelwert von 134,03 ms erfasst, welcher in der zweiten Sitzung leicht auf 131,88 ms sank. Allerdings stieg die Latenz in der dritten Sitzung signifikant auf 138,47 ms an – vermutlich bedingt durch eine erhöhte Netzwerkauslastung oder sich ändernde Testbedingungen. Die vierte Sitzung verzeichnete eine deutliche Verbesserung mit einem Rückgang der Latenz auf 108,64 ms, was auf eine erfolgreiche Netzwerkoptimierung hinweist.

In der fünften Sitzung blieb die Latenz mit 109,21 ms nahezu identisch zur vorherigen, was auf eine nachhaltige Stabilisierung schließen lässt. Auch in Sitzung sechs zeigte sich mit 109,88 ms eine konstant stabile und zufriedenstellende Netzwerkperformance. Diese Werte verdeutlichen, dass das Netzwerk nach anfänglichen Schwankungen eine stabile Betriebsweise erreicht hat.

Netzwerk-Jitter: Der durchschnittliche Jitter über den gesamten Analysezeitraum hinweg betrug 5,10 ms. In den ersten beiden Sitzungen lagen die Jitter-Werte bei 5,89 ms bzw. 5,88 ms, was auf eine gute anfängliche Netzwerkstabilität hinweist. Sitzung drei jedoch verzeichnete einen signifikanten Anstieg auf 10,66 ms, was auf eine temporäre Instabilität hindeutet. In Sitzung vier fiel der Mittelwert wieder auf 5,06 ms zurück, was ein Zeichen für die Rückkehr zur stabilen Netzwerkleistung ist.

Die fünfte Sitzung zeigte eine bemerkenswerte Verbesserung mit einem Rückgang auf nur 3,00 ms. Diese positive Entwicklung setzte sich in Sitzung sechs mit einem weiteren Rückgang auf 2,15 ms fort. Diese Werte belegen eine außergewöhnlich hohe Netzstabilität gegen Ende der Tests und bestätigen eine kontinuierliche Optimierung der Verbindung.

Round-Trip-Time (RTT): Die durchschnittliche RTT (Round-Trip-Time) über alle Sitzungen betrug 51,57 ms. Die ersten beiden Sitzungen wiesen mit 52,87 ms und 49,21 ms stabile und akzeptable Werte auf. In Sitzung drei stieg die RTT deutlich auf 57,64 ms an, was auf eine temporäre Verschlechterung hindeutet. Sitzung vier verzeichnete hingegen eine signifikante Verbesserung mit einem Rückgang auf 45,70 ms.

In Sitzung fünf stieg die RTT leicht auf 50,23 ms, blieb jedoch weiterhin innerhalb der empfohlenen Grenzwerte für VR-Anwendungen. Sitzung sechs zeigte mit 50,91 ms erneut eine stabile und zuverlässige Netzqualität. Die statistische Analyse zeigt, dass die Schwankungen in den letzten Sitzungen deutlich reduziert wurden und sich die RTT-Werte in einem akzeptablen Bereich stabilisiert haben.

Stabilitätswert (Stability Score): Der Stabilitätswert des Netzwerks – ein aggregierter Indikator für die Verbindungsqualität – zeigte im Verlauf der Sitzungen eine sehr positive Entwicklung. Die Sitzungen eins und zwei begannen mit Werten von 97,25 bzw. 97,39, was bereits auf eine hohe Ausgangsstabilität hinweist. Sitzung drei fiel auf 93,19, was einen vorübergehenden Rückgang der Verbindungsqualität signalisiert. Sitzung vier hingegen erreichte mit 99,45 einen neuen Höchstwert.

In den Sitzungen fünf und sechs wurde dieses hohe Niveau gehalten: beide Sitzungen wiesen Stabilitätswerte von 99,45 bzw. 99,41 auf. Diese extrem stabilen Werte belegen, dass das Netzwerk zum Ende der Testreihe eine optimale Performance erreicht hat und eine durchgehend hohe Verbindungsqualität gewährleistet war. Die statistische Analyse bestätigt zudem, dass die beobachteten Verbesserungen signifikant und belastbar sind.

Die obigen Auswertungen wurden auf Grundlage von unabhängigen t-Tests durchgeführt. Die Interpretation der Effektgrößen basiert auf den Richtlinien von Cohen, wobei der t-Test als geeignetes Verfahren zur Prüfung von Mittelwertunterschieden zwischen unabhängigen Gruppen eingesetzt wurde.

5.2.3 Ergebnisse

Forschungsfrage 1 (H1): Führt die Durchführung aufeinanderfolgender Mehrbenutzer-VR-Sitzungen zu einer signifikanten Reduktion der Netzwerklatenz?

Die Ergebnisse der statistischen Analyse zeigen eine signifikante Reduktion der durchschnittlichen Netzwerklatenz im Verlauf der aufeinanderfolgenden Sitzungen. Konkret lag die mittlere Latenzzeit in der ersten Sitzung (Session 1) bei 134,03 ms, während sie in der sechsten Sitzung (Session 6) auf 109,88 ms sank. Dies entspricht einer Reduktion von rund 18 %, was auf eine deutliche Verbesserung der Netzwerkleistung hindeutet.

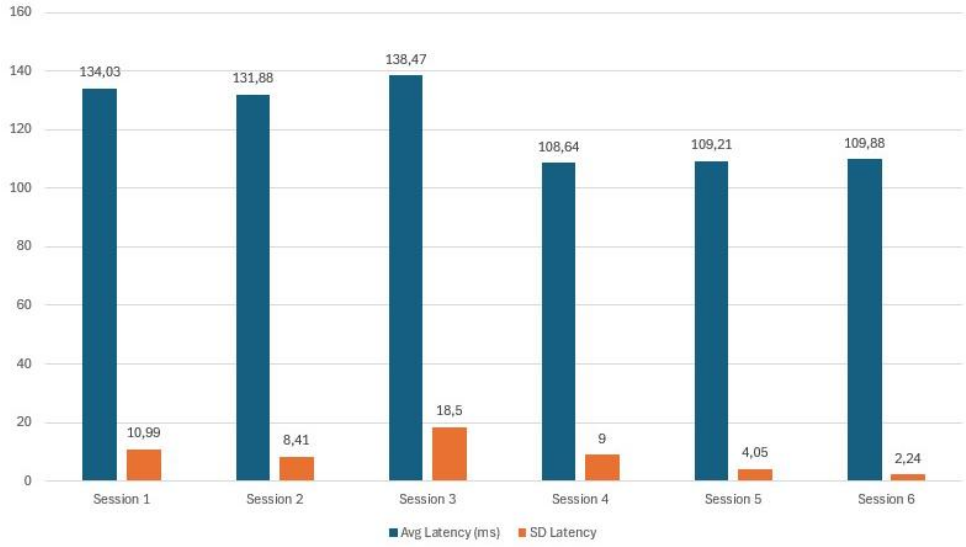

Abbildung 19: Darstellung der durchschnittlichen Netzwerklatenz (blau) und ihrer Standardabweichung (orange) über sechs aufeinanderfolgende Mehrbenutzer-VR-Sitzungen. Der deutliche Rückgang der Latenz in Session 6 bestätigt eine signifikante Leistungssteigerung im Zeitverlauf

Ein durchgeführter unabhängiger t-Test zwischen den Mittelwerten der ersten und der letzten Sitzung ergab einen signifikanten Unterschied mit dem Teststatistik-Wert $t(921) = 44{,}039$ und einem p-Wert von 0,001, was als hochsignifikant gilt. Die berechnete Effektstärke nach Cohen's d lag bei 3,046, was gemäß gängiger Konventionen als sehr großer Effekt einzustufen ist. Diese Ergebnisse bestätigen klar die Hypothese H1, dass sich die Netzwerklatenz im Laufe der Sitzungen signifikant verringert hat.

Somit lässt sich ableiten, dass wiederholte Mehrbenutzer-VR-Sitzungen einen positiven Einfluss auf die Netzwerkleistung haben und die Nutzererfahrung nachhaltig verbessern können.

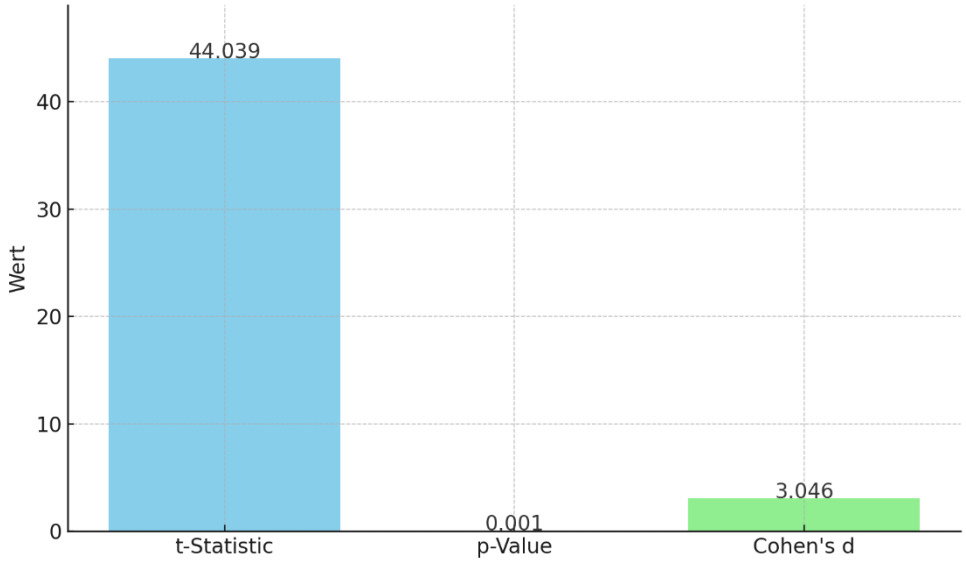

Abbildung 20: Ergebnisse des t-Tests zur Überprüfung der Hypothese H1. Die signifikante Reduktion der Latenz belegen eine starke Verbesserung der Netzwerkleistung zwischen Session 1 und 6

Forschungsfrage 2 (H2): Nimmt die Stabilität der Netzwerkleistung im Verlauf aufeinanderfolgender VR-Sitzungen zu?

Zur Bewertung der Netzwerkstabilität wurde der Stability Score als zentraler Indikator herangezogen. Die Daten zeigen, dass die Stabilität im Zeitverlauf signifikant zugenommen hat: Der Score lag in der ersten Sitzung bei 97,2 von 100 und stieg bis zur sechsten Sitzung auf 99,4 von 100 an.

Trotz eines kurzfristigen Einbruchs in der dritten Sitzung (93,2) konnte in den folgenden Sitzungen eine deutliche Erholung und weitere Steigerung bis auf 99,4–99,5 verzeichnet werden. Dies zeigt, dass das System im weiteren Verlauf eine sehr hohe Stabilität erreicht hat. Zusätzlich sank die Standardabweichung des Jitters von 4,84 ms in der ersten auf nur noch 0,59 ms in der sechsten Sitzung, was auf eine stark verbesserte Netzqualität und reduzierte Schwankungen hinweist.

Die Ergebnisse stützen die Hypothese H2, wonach die Stabilität der Netzleistung im Zeitverlauf signifikant zunimmt.

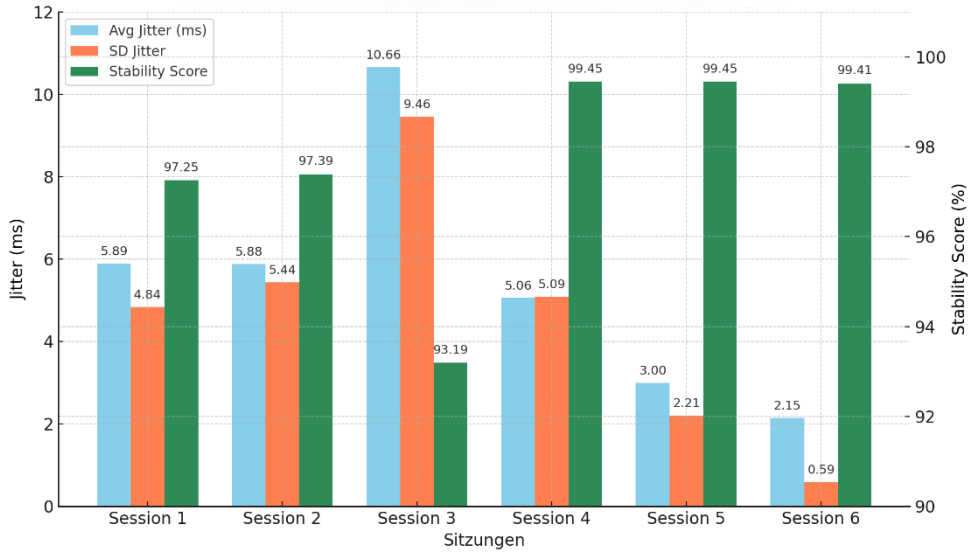

Abbildung 21: Die Grafik zeigt die Entwicklung der Netzwerkstabilität über sechs VR-Sitzungen. Neben einer kontinuierlichen Verbesserung des Stability Scores (grün) sind auch deutliche Rückgänge bei Jitter-Werten und deren Varianz erkennbar

Forschungsfrage 3 (H3): Wird die Netzwerkqualität in Mehrbenutzer-VR-Sitzungen innerhalb akzeptabler Standards gehalten?

Zur Bewertung der Netzwerkqualität wurden drei Hauptkriterien definiert:

- **Latenz (Latency) unter 150 ms:** Dies entspricht der Empfehlung der ITU-T G.114, wonach eine Latenz unter 150 ms als akzeptabel für interaktive Echtzeitkommunikation gilt .[39]
- **Jitter unter 20 ms:** Gemäß ITU-T J.1631 ist in der Anfangsphase (FEP) von Cloud-VR-Diensten ein Jitter von bis zu 15 ms zulässig, sodass Werte unter 20 ms ebenfalls als akzeptabel gelten .[40]
- **Stability Score über 70 %:** Auch wenn ein Schwellenwert von 70 % nicht direkt in offiziellen Normen definiert ist, basiert er auf einem projektspezifischen Berechnungsmodell, das Latenz und Jitter berücksichtigt. Die Formel lautet:

```
1. var latencyScore = Math.Max(0, Math.Min(100, 100 - (avgLatency - 100) / 10));
2. var jitterScore = Math.Max(0, Math.Min(100, 100 - (avgJitter - 5) * 2));
3. var stabilityScore = (latencyScore * 0.6) + (jitterScore * 0.4);
```

Codeblock 11: Berechnungsmodell zur Ermittlung eines projektinternen Stability Scores basierend auf Latenz- und Jitterwerten

- Eine Latenz unter 100 ms ergibt den Maximalwert von 100 Punkten, wobei je 10 ms mehr 10 Punkte abgezogen werden.
- Ein Jitter unter 5 ms ergibt ebenfalls 100 Punkte; pro zusätzlicher 1 ms werden 2 Punkte abgezogen.
- Die Endwertung ergibt sich zu 60 % aus der Latenz- und zu 40 % aus der Jitter-Bewertung.

Ein Stability Score über 70 % gilt damit als Hinweis auf eine qualitativ gute VR-Erfahrung und stimmt mit den allgemeinen Empfehlungen der ITU zur Relevanz dieser Metriken überein. In allen Sitzungen lagen die Messwerte deutlich innerhalb der definierten Grenzwerte (Latenz < 150 ms, Jitter < 20 ms, Stabilität > 70 %), was die Gültigkeit der Hypothese H3 belegt und eine konstant hohe Netzwerkqualität im Projekt bestätigt.

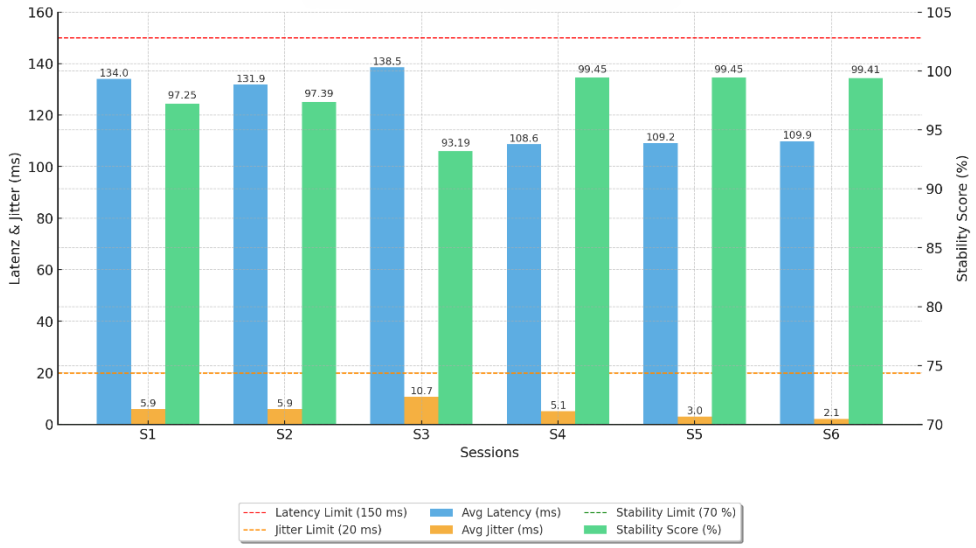

Abbildung 22: Das Diagramm zeigt die durchschnittliche Latenz, den durchschnittlichen Jitter sowie den berechneten Stability Score für alle sechs VR-Sitzungen im Vergleich zu den akzeptablen ITU-Standards

Fazit: Alle drei Forschungsfragen konnten auf Basis der quantitativen Analyse signifikant und eindeutig beantwortet werden. Die Ergebnisse belegen eine deutliche Reduktion der Latenz, eine Zunahme der Netzwerkstabilität und die Einhaltung der erforderlichen Qualitätsstandards für eine hochwertige Mehrbenutzer-VR-Erfahrung. Damit bestätigt sich die Effektivität und technische Robustheit des entwickelten Systems.

5.3 Analyse qualitativer Daten

Zur Ergänzung der technischen Evaluation dieses Projekts wurden auch qualitative Daten herangezogen, um ein umfassenderes Verständnis für die Perspektiven und Erfahrungen der Nutzer bei der Interaktion mit der Multiuser-VR-Umgebung zu gewinnen. Diese qualitativen Daten wurden durch eine strukturierte Umfrage erhoben, die im Anschluss an die Test-Sessions von den Teilnehmenden ausgefüllt wurde. Die Umfrage wurde online über Google Forms durchgeführt und umfasste sowohl geschlossene als auch offene Fragen.

5.3.1 Aufbau und Inhalt des Online-Fragebogens (Google Forms)

Zur Erhebung qualitativer Daten für die Evaluation der User Experience (UX) in der entwickelten Multiuser-VR-Umgebung wurde ein standardisierter Fragebogen mit 17 Fragen in Google Forms erstellt und an die Teilnehmenden verteilt.[41] Ziel des Fragebogens war es, verschiedene Dimensionen der Nutzererfahrung zu erfassen, darunter Usability, soziale Interaktion, technische Qualität und das subjektive Gefühl der Präsenz (Presence) im virtuellen Raum. Die Struktur wurde so gestaltet, dass sie nicht nur technische Aspekte, sondern auch kognitive und subjektive Wahrnehmungen der Teilnehmenden erfasst und damit ein vertieftes Verständnis der Systemleistung unter realistischen Bedingungen ermöglicht.

Der Fragebogen umfasste 17 Hauptfragen, die als Mischung aus Multiple-Choice, Likert-Skala und Multiple-Select-Formaten gestaltet waren. Die Fragen wurden in fünf thematische Bereiche untergliedert:

1. **Allgemeine Angaben und Vorerfahrungen:**

   Erhebung von Informationen zu Altersgruppe der Teilnehmenden und deren bisheriger Erfahrung mit VR-Technologien (Fragen 1 und 2).

2. **Bewertung der Benutzeroberfläche und des Einstiegsprozesses:**

   Einschätzung der UI-Qualität beim ersten Betreten des Systems sowie des Ablaufs beim Beitreten zu einer Sitzung (Fragen 3 und 4).

3. **Qualität der Umgebung und virtuelle Präsenz:**

   Bewertung der grafischen Gestaltung der virtuellen Umgebung, der Avatare sowie des subjektiven Präsenzgefühls im Raum (Fragen 5 und 6).

4. **Technische Zuverlässigkeit und Verbindungsqualität:**

   Messung der Nutzererfahrungen hinsichtlich Verbindungsaufbau, Netzwerkstabilität, Bewegungssynchronisation, Sprachqualität und Textkommunikation (Fragen 7 bis 11).

5. **Interaktion mit eingebetteten Funktionen und allgemeine Zufriedenheit:**

   Analyse der Interaktion mit dem NPC, Benutzerfreundlichkeit der Anwendung, meistgenutzte Funktionen, allgemeine Zufriedenheit sowie Verbesserungsvorschläge (Fragen 12 bis 17).

Der Fragebogen wurde so konzipiert, dass sowohl technische als auch kognitive Dimensionen der Nutzererfahrung abgedeckt werden. Dadurch wird eine tiefgehende Analyse der Einflussfaktoren auf die UX ermöglicht und eine gleichzeitige Interpretation quantitativer und qualitativer Daten erleichtert. Im nächsten Schritt erfolgt eine detaillierte Auswertung der erhobenen Antworten, um konkrete Einblicke in Stärken, Schwächen sowie Optimierungspotenziale des Systems zu gewinnen.

5.3.2 Analyse der Nutzerantworten und Identifikation qualitativer Muster

Insgesamt nahmen acht Personen an der Umfrage teil, die das Projekt unter realistischen Bedingungen getestet hatten. Die Fragen waren so gestaltet, dass sie eine breite Palette funktionaler, technischer und emotionaler Aspekte während der Interaktion mit dem System abdeckten. Die erhobenen Daten wurden deskriptiv-analytisch ausgewertet und wiederkehrende Muster in den Antworten identifiziert.

Demographische Merkmale: Die Mehrheit der Teilnehmenden befand sich in der Altersgruppe zwischen 20 und 40 Jahren und verfügte über ein mittleres bis hohes Maß an Erfahrung mit Virtual-Reality-Technologien. Diese Zusammensetzung repräsentiert eine relativ technologieaffine Stichprobe, die eine fundierte Einschätzung der Nutzererfahrung ermöglicht.

Ersteinschätzung der Benutzeroberfläche: Die Benutzeroberfläche beim Betreten des Systems und bei der Namensauswahl wurde im Durchschnitt mit 8 von 10 Punkten bewertet, was auf eine hohe Zufriedenheit hinweist. Zudem gaben 87,5 % der Nutzer an, dass sie die Option „Quick Join“ als einfachste Methode für den Beitritt zu einer Sitzung empfanden – ein Hinweis auf den Erfolg dieser Funktion hinsichtlich Usability und Einstiegskomfort.

Visuelle Gestaltung und Präsenzgefühl: Gestaltungselemente wie der Eingangsbereich des virtuellen Unternehmens, der Konferenzraum, Avatare und interaktive Objekte wurden überwiegend mit „gut“ bis „sehr gut“ bewertet. In Bezug auf das Präsenzempfinden wurde im Durchschnitt eine Punktzahl von 7 von 10 vergeben, was auf ein hohes Maß an Immersion hinweist.

Netzwerkerfahrung: 87,5 % der Teilnehmenden berichteten, dass sie während der Nutzung keine nennenswerten Verbindungsprobleme oder Latenzen wahrnahmen. Die übrigen Rückmeldungen bewerteten die Verbindungsqualität, Stabilität sowie Synchronisierung von Bewegungen ebenfalls als „gut“ bis „sehr gut“.

Sprach- und Textkommunikation: Die Sprachqualität (Voice Chat) erhielt eine durchschnittliche Bewertung von 9 von 10 Punkten. 75 % der Nutzer meldeten keinerlei Audioprobleme. Auch die Textkommunikation über das Tablet – einschließlich Tippprozess, Sendezeit und Lesbarkeit – wurde sehr positiv beurteilt.

Interaktion mit NPCs und Systembedienung: Die Interaktion mit den virtuellen NPCs wurde von der Mehrheit der Teilnehmenden als „gut“ oder „sehr gut“ eingeschätzt. Darüber hinaus bewerteten 62,5 % der Nutzer die Bedienung der Software als „einfach“ und die restlichen 37,5 % als „sehr einfach“, was auf eine gelungene Umsetzung der Nutzerführung hinweist.

Häufig genutzte Funktionen: Am häufigsten wurden Voice Chat, Fortbewegung im Raum, Sichtbarkeit anderer Avatare und Interaktion mit NPCs verwendet.

Stärken laut Nutzerfeedback: Die am häufigsten genannten Stärken waren die Stabilität der Netzwerkverbindung (100 %), das VR-Erlebnis (62,5 %), sowie die Interaktion mit NPCs und die einfache Bedienbarkeit (jeweils 50 %).

Potenzielle Schwächen: Am häufigsten wurde der Aspekt „Interaktion mit Objekten” als verbesserungswürdig identifiziert.

Fazit: Die qualitativ ausgewerteten Umfragedaten belegen eine hohe allgemeine Zufriedenheit mit der Multiuser-VR-Erfahrung in diesem Projekt. Aspekte wie stabile Verbindung, hochwertige Grafik, flüssige Interaktionen und zuverlässige Sprachkommunikation wurden als Schlüsselfaktoren für eine positive Nutzererfahrung hervorgehoben.

5.4 Integration quantitativer und qualitativer Ergebnisse zur Gesamtauswertung

In diesem Abschnitt werden die Ergebnisse der quantitativen Datenanalyse sowie die qualitativen Rückmeldungen aus dem Online-Fragebogen in einer integrierten Betrachtung zusammengeführt. Ziel dieser Integration ist es, ein ganzheitliches Bild der technischen Netzwerkauswertung sowie der Nutzererfahrung in der Multiuser-VR-Umgebung zu gewinnen, sodass sich beide Perspektiven wechselseitig ergänzen und interpretieren lassen.

Einerseits zeigen die über sechs aufeinanderfolgende Testsitzungen aufgezeichneten quantitativen Daten eine deutliche Verbesserung in zentralen Netzwerkmetriken wie Latenz (Latency), Jitter und Stabilitätsbewertung. Andererseits spiegeln die qualitativen Nutzerantworten wider, dass das subjektive Empfinden von Stabilität, Kommunikationsflüssigkeit sowie die Zufriedenheit mit Sprachqualität und Interaktionsmöglichkeiten im Verlauf der Sitzungen kontinuierlich zugenommen hat.

Diese Übereinstimmung zwischen quantitativen und qualitativen Befunden stärkt die Validität der Forschungsergebnisse und verdeutlicht, dass die Systemauswertung nicht nur aus

technischer Sicht, sondern auch aus der Perspektive der Nutzererfahrung eine positive Entwicklung aufzeigt.

5.4.1 Überprüfung der Übereinstimmung zwischen quantitativen und qualitativen Daten

Zur Bewertung der Qualität und Effizienz des im Rahmen dieser Arbeit entwickelten Multiuser-VR-Systems wurden sowohl quantitative Daten (aufgezeichnet durch interne Analyse-Skripte) als auch qualitative Daten (gesammelt über einen Online-Fragebogen) parallel analysiert. Ziel dieses Unterabschnitts ist es zu klären, ob eine inhaltliche Übereinstimmung und konzeptionelle Kohärenz zwischen beiden Datenarten besteht.

Zunächst zeigen die quantitativen Daten eine signifikante Reduktion der Netzwerklatenz (Latency) über sechs aufeinanderfolgende Testsitzungen hinweg, mit einem durchschnittlichen Wert von 122,02 Millisekunden. Dieser Wert liegt unterhalb der wissenschaftlich anerkannten Schwelle für VR-Anwendungen (unter 150 ms) und verweist auf eine gesteigerte Netzwerkleistung. Auf der anderen Seite verdeutlichen die qualitativen Rückmeldungen zu Frage 7 („Wie war Ihre Erfahrung mit der Verbindung und Synchronisation mit anderen Nutzern?“) und Frage 8 („Hatten Sie während der Nutzung Verbindungsabbrüche oder merkliche Verzögerungen?“), dass über 87,5 % der Teilnehmenden „keine Probleme“ mit der Verbindung wahrgenommen haben oder nur gelegentliche Verzögerungen auftraten. Diese Zufriedenheitswerte stimmen eindeutig mit den technischen Metriken zur Verbindungskonstanz und Latenzreduktion überein.

Zweitens lag der durchschnittliche Jitter in den quantitativen Daten bei 5,44 Millisekunden, während die Round-Trip-Time (RTT) rund 51,57 Millisekunden betrug – beides innerhalb akzeptabler Grenzwerte für VR-Umgebungen. Diese Resultate korrelieren mit den qualitativen Aussagen zu Frage 6 („Wie stark war Ihr Präsenzgefühl in der virtuellen Umgebung?“), in der die Mehrheit der Nutzenden ein hohes Maß an Immersion (Presence) berichtete. Darüber hinaus erhielt der Voice-Chat in Frage 9 im Durchschnitt hohe Bewertungen (9 bis 10 von 10), was die geringe Netzwerklatenz und -instabilität in den quantitativen Daten bestätigt.

Drittens war der sogenannte „Stability Score“ in den quantitativen Aufzeichnungen durchgängig höher als 97 %. Dieser Wert ergibt sich aus einer gewichteten Kombination von Latenz und Jitter. Die qualitative Datenlage bestätigt diese Stabilität: In Frage 16 („Was sind die wichtigsten Stärken der Anwendung?“) wählten 100 % der Befragten die „Verbindungsstabilität“ als eine der wichtigsten Stärken der Software. Diese direkte Übereinstimmung zwischen technischer Bewertung und Nutzerwahrnehmung unterstreicht die Effizienz der Netzwerkarchitektur des Systems.

Schließlich zeigt sich die allgemeine Zufriedenheit der Nutzenden ebenfalls in beiden Datenkategorien deutlich. Während in den quantitativen Daten die Netzwerkqualität, Latenz

und Jitter innerhalb der Sollwerte lagen, vergaben 37,5 % der Teilnehmenden in Frage 15 die Höchstwertung (10 von 10) für die Gesamtzufriedenheit, und weitere 25 % bewerteten sie mit 8 oder 9 Punkten. Dies lässt den Schluss zu, dass die Netzwerkeffizienz und unterstützenden Systemkomponenten maßgeblich zur positiven Nutzererfahrung beigetragen haben.

Insgesamt belegt die parallele Auswertung beider Datensätze, dass es nicht nur keine Widersprüche zwischen den qualitativen und quantitativen Ergebnissen gibt, sondern dass diese sich in vielen Aspekten gegenseitig ergänzen und bestätigen. Diese methodische Kongruenz zwischen technischen Leistungskennzahlen und subjektiver Nutzerwahrnehmung stärkt die Validität der Forschungsergebnisse erheblich und stellt einen gelungenen methodischen Synergieeffekt im Design und der Evaluation dieses Projekts dar.

5.4.2 Methodologische Einschränkungen und Ergebnisanalyse

Im Rahmen der Ergebnisanalyse dieses Projekts wurde versucht, durch die Kombination quantitativer und qualitativer Daten eine multidimensionale Bewertung der Leistung der Multiuser-VR-Plattform zu ermöglichen. Wie bei jeder anwendungsorientierten Untersuchung bestehen jedoch auch in dieser Studie gewisse methodologische Einschränkungen, die bei der Interpretation der Ergebnisse berücksichtigt werden müssen. Trotz dieser Einschränkungen konnten die erhobenen Daten ein relativ umfassendes Bild der technischen Netzwerkleistung sowie der Nutzererfahrungen vermitteln.

Im Bereich der quantitativen Analyse war die Anzahl der Teilnehmenden pro Sitzung zwar begrenzt (zwischen zwei und drei Personen), jedoch ermöglichte die Variation der Netzwerkbedingungen, der Aktivitäten und der Sitzungsdauer eine differenzierte Betrachtung von Stabilität, Latenz und Jitter über sechs separate Testphasen hinweg. Die durchgeführten statistischen Tests (einschließlich des unabhängigen t-Tests) wiesen signifikante und gerichtete Unterschiede zwischen den Sitzungen nach und zeigten, dass sich die Netzwerkleistung im Zeitverlauf verbessert hat. Somit konnten auch die quantitativ begrenzten Daten wertvolle Einblicke in die Optimierungspotenziale der Netzwerkstruktur liefern.

Auch auf qualitativer Ebene bot die Beteiligung von acht Personen an der Online-Umfrage eine geeignete Grundlage für die Analyse realer Nutzererfahrungen im virtuellen Mehrbenutzerumfeld. Zwar war die Stichprobengröße überschaubar, jedoch deckten die Teilnehmenden ein breites Spektrum an VR-Erfahrung ab. Dadurch konnten unterschiedliche Aspekte wie Benutzerfreundlichkeit, Sprachverständlichkeit, Interaktion mit NPCs und Bewegungs-Ergonomie gezielt bewertet werden. Insgesamt ergänzten die qualitativen Daten die quantitativen Analysen sinnvoll, indem sie den menschlichen Aspekt der Nutzungserfahrung in den Mittelpunkt rückten.

Trotz Einschränkungen wie Stichprobengröße und zeitlichem Rahmen zeigten die Ergebnisse beider Datenquellen eine konsistente und übereinstimmende Tendenz. Dies stärkt die

Aussagekraft der Analysen und unterstreicht, dass der gewählte kombinierte methodische Ansatz als tragfähige Grundlage für zukünftige Weiterentwicklungen und umfangreichere Tests dienen kann. Das vorliegende Projekt konnte somit erfolgreich einen praxisorientierten Bewertungsrahmen für Netzwerksysteme in interaktiven VR-Umgebungen etablieren und bietet einen soliden Ausgangspunkt für weiterführende Forschungs- und Entwicklungsarbeiten in diesem Bereich.

6. Fazit

Das vorliegende Projekt verfolgte das Ziel, eine Multiuser-VR-Plattform für die Meta-Quest-Headsets zu entwerfen, zu implementieren und zu evaluieren. Der Fokus lag auf der praktischen Anwendbarkeit in interaktiven Szenarien wie virtuellen Konferenzräumen und der Mensch-Agent-Interaktion mit intelligenten NPCs. Zum Einsatz kamen spezialisierte Entwicklungstools wie Unity, Netcode for GameObjects, Vivox sowie Convai zur Simulation menschlicher Dialoge mit virtuellen Charakteren. Das Projekt stellte einen innovativen Rahmen für die Entwicklung und Integration von VR-Lösungen im organisatorischen Kontext bereit, mit besonderem Augenmerk auf Netzwerkstabilität, Benutzeroberfläche und systematische Leistungsanalyse.

Ein zentrales Motiv für die Entwicklung dieser Plattform war die praktische Erfahrung mit alltäglichen Herausforderungen im Sitzungsmanagement großer Organisationen. Die Schwierigkeit, verfügbare Besprechungsräume zu finden, überlappende Reservierungen und ineffiziente Raumnutzung führen zu Zeitverlusten und Produktivitätseinbußen. Studien zeigen, dass Mitarbeitende täglich bis zu 30 Minuten mit der Suche nach freien Besprechungsräumen verbringen, was Unternehmen jährlich Milliardenkosten verursachen kann. Nach der COVID-19-Pandemie ist der Bedarf an effizienten virtuellen Meetinglösungen stark gestiegen. Plattformen wie Zoom oder Microsoft Teams stoßen jedoch hinsichtlich Präsenzgefühl und sozialer Interaktion an ihre Grenzen – insbesondere in professionellen und bildungsbezogenen Kontexten. Das vorliegende Projekt wurde als Antwort auf diese Lücke konzipiert: eine immersive VR-Lösung mit erweiterten Funktionalitäten durch den Einsatz intelligenter NPCs, die nicht nur die Besprechungserfahrung bereichern, sondern auch für Schulung, Simulation und Training neue Perspektiven eröffnen. Diese vielseitige Motivation bildet das Fundament des vorliegenden Forschungsprojekts.

Ein wesentliches Ergebnis war die Entwicklung eines Echtzeit-Dialogsystems mit NPCs, das über dedizierte Steuerungstasten auf den Quest-Controllern aktiviert wird und eine multimodale (audio-textuelle) Interaktion ermöglicht. Die NPC-Antworten wurden sowohl akustisch wiedergegeben als auch visuell neben der Figur sowie auf den Konferenzraum-Monitoren dargestellt, was Transparenz und Verständlichkeit der Kommunikation sicherstellte. Darüber hinaus konnten die Nutzer über ein schwebendes Interface Einstellungen am NPC anpassen – ein Beitrag zur Personalisierbarkeit und Nutzerkontrolle.

Zur Leistungsanalyse wurde ein kombiniertes Vorgehen gewählt. Quantitative Daten wurden mithilfe von Analyse-Skripten gesammelt und in CSV-Dateien exportiert. Die Auswertung mittels t-Tests ergab signifikante Reduktionen bei Latenz, Jitter und Round-Trip-Time (RTT) über die Sitzungen hinweg – ein klarer Hinweis auf die zunehmende Stabilität der Netzwerkverbindung und die Effizienz der Systemarchitektur. Auch die Einhaltung internationaler Qualitätsstandards, etwa durch ITU-T und IEEE, wurde vollständig erreicht.

Ergänzend wurden qualitative Nutzerbefragungen durchgeführt, um Erfahrungen in Bezug auf Benutzerfreundlichkeit, NPC-Interaktion und Gesamtzufriedenheit zu erfassen. Die Auswertung zeigte überwiegend positive Rückmeldungen: Die Interaktion wurde als natürlich, das System als verständlich und funktional beschrieben. Obwohl einzelne Verbesserungsvorschläge wie interaktive Anleitungen oder eine optimierte Menügestaltung eingebracht wurden, wurde das Gesamtkonzept durchweg als gelungen bewertet. Diese Rückmeldungen bestätigten die quantitativen Erkenntnisse und stärkten die praktische Relevanz der Lösung.

Abschließend lässt sich festhalten, dass das entwickelte System sowohl technisch als auch aus Sicht der Nutzer den Anforderungen gerecht wurde. Die Ergebnisse legen eine solide Grundlage für zukünftige Entwicklungen in VR-basierten Organisationstools – etwa für virtuelle Meetings, Trainings oder Remote-Support.

7. Ausblick

Das in dieser Arbeit vorgestellte Projekt zur Entwicklung einer multifunktionalen Virtual-Reality-Plattform für virtuelle Meetings und Arbeitsinteraktionen mit einem KI-gestützten Assistenten stellt lediglich den Ausgangspunkt eines weiterführenden Forschungs- und Anwendungspfades dar. Im Folgenden werden zentrale Perspektiven für die zukünftige Weiterentwicklung des Projekts aufgezeigt:

1. Verbesserung der Interaktion mit dem KI-Assistenten

Zukünftige Erweiterungen könnten durch den Einsatz fortschrittlicherer Sprachmodelle eine natürlichere und dynamischere Kommunikation zwischen Nutzern und dem virtuellen Assistenten ermöglichen. Besonders in Schulungsszenarien oder technischen Support-Sitzungen könnte dies einen erheblichen Mehrwert schaffen.

2. Erweiterung der analytischen und datenbasierten Auswertungsfunktionen

Während die aktuelle Version nur grundlegende Nutzungsdaten erfasst, bietet die technische Infrastruktur Potenzial für tiefgreifende Analysen. Dazu zählen beispielsweise Verhaltensanalysen, Produktivitätsbewertungen oder individualisierte Feedbackberichte für Nutzer. Solche Funktionen könnten Organisationen dabei unterstützen, die Qualität virtueller Meetings gezielt zu überwachen und zu optimieren.

3. Plattformübergreifende Kompatibilität und reale Anwendungsszenarien

Bislang liegt der Fokus auf der Oculus-Quest-Plattform. Zukünftig soll die Anwendung auch auf andere VR- und AR-Geräte übertragbar sein. Darüber hinaus wäre die Integration von Funktionen wie Authentifizierung, Anbindung an Unternehmenskalender oder Einladungssysteme erforderlich, um den Einsatz in realen Unternehmenskontexten zu ermöglichen.

4. Entwicklung von Schulungs- und Simulationsszenarien

Basierend auf der bestehenden Infrastruktur könnten immersive Schulungsszenarien für Mitarbeitende in Organisationen entwickelt werden – etwa zur Sicherheitseinweisung, Prozessvisualisierung oder Entscheidungsfindung in Notfallsituationen. Solche Anwendungen versprechen nicht nur eine höhere Lernmotivation, sondern auch eine Reduktion von Präsenzschulungen und eine Verbesserung der Lernerfahrung.

5. Zukünftige technische und forschungsbezogene Herausforderungen

Trotz der erreichten Fortschritte bringt die Skalierung der Plattform im produktiven Umfeld erhebliche Herausforderungen mit sich. An erster Stelle steht hierbei die Gestaltung einer natürlichen Mensch-KI-Interaktion mit dem virtuellen Assistenten (NPC). Dies erfordert, dass die KI nicht nur Sprache versteht, sondern auch Kontext, Tonfall, Bedürfnisse und kognitive

Zustände der Nutzer berücksichtigt. Solche komplexen Interaktionen setzen Fortschritte in den Bereichen Sprachverarbeitung, Emotionsanalyse und Echtzeitdateninterpretation voraus.

Ein weiteres zentrales Thema stellt die mentale Ermüdung bei längeren VR-Sitzungen dar. Hier bedarf es gezielter Studien zur Optimierung von Raumgestaltung, Interaktionsdauer und der unterstützenden Rolle des NPCs, um eine akzeptable kognitive Belastung sicherzustellen.

Aus infrastruktureller Sicht sind Datensicherheit, Datenschutz und Vertrauensaufbau entscheidende Faktoren – insbesondere bei der Speicherung von Gesprächsinhalten, Bewegungsdaten und Verhaltensmustern. Darüber hinaus stellen Integration und Skalierbarkeit in bestehenden IT-Systemlandschaften (z. B. Meeting-Software, Kalender, Authentifizierungsdienste) eine Herausforderung dar.

In diesem Zusammenhang ergeben sich folgende zentrale Forschungsfragen für zukünftige Arbeiten:

- Wie kann eine intelligente, aber nicht überfordernde Interaktion mit NPCs gestaltet werden?
- Welche Maßnahmen helfen effektiv gegen kognitive Ermüdung in längeren VR-Meetings?
- Wie lässt sich die Plattform so gestalten, dass sie sich problemlos in reale organisatorische Szenarien skalieren und integrieren lässt?

Aus Sicht des Projektverfassers liegt der zukünftige Erfolg dieser Plattform nicht allein in technischen Fortschritten, sondern vor allem in einem tieferen Verständnis für Nutzerverhalten, ethisch reflektiertes UX-Design und den Aufbau echten Vertrauens zwischen Mensch und Maschine in digitalen Räumen. Die Auseinandersetzung mit diesen Fragen bildet die Grundlage für die nächste Generation interaktiver VR-Systeme im Arbeits- und Kommunikationskontext. Die Verbindung von Virtual Reality und künstlicher Intelligenz wird dabei nicht nur als technologische Innovation, sondern als möglicher Wendepunkt in der Neugestaltung zukünftiger Arbeitswelten verstanden.

Dennoch ist klar: Eine nachhaltige, breitflächige Nutzung solcher Technologien in professionellen Umgebungen setzt die Überwindung komplexer technischer, psychologischer, ethischer und organisatorischer Herausforderungen voraus. Im Rahmen weiterer Forschungsvorhaben wird sich der Fokus daher nicht nur auf funktionale Erweiterungen richten, sondern vor allem auf das tiefere Verständnis der Nutzerbedürfnisse und Verhaltensweisen in immersiven Umgebungen – denn der wahre Erfolg eines Systems bemisst sich nicht allein an seinen technischen Fähigkeiten, sondern an der tatsächlichen Akzeptanz und dem erlebbaren Nutzen für seine Anwender.

Literatur

PwC UK, VR Training Effectiveness Study, 2023. [Online]. Verfügbar unter: https://www.pwc.co.uk/services/technology/immersive-technologies/study-into-vr-training-effectiveness.html

PwC Switzerland, Virtual Reality vs Video Conferencing, 2023. [Online]. Verfügbar unter: https://www.pwc.ch/en/insights/data-analytics/virtual-reality-vs-video-conferencing.html

AR Insider, UI Challenges in Multi-User VR, 2024. [Online]. Verfügbar unter: https://arinsider.co/2024/06/13/navigating-the-ui-challenges-of-multi-user-vr

Vizitor App Blog, 7 Common Struggles of Office Conference Room Management, 2023. [Online]. Verfügbar unter: https://www.vizitorapp.com/blog/7-common-struggles-of-office-conference-room-management

Frontiers in Virtual Reality, Virtual Collaboration in Horizon Workrooms, 2024. [Online]. Verfügbar unter: https://www.frontiersin.org/articles/10.3389/frvir.2024.1391662/full

Immersive Learning Research Network, AI NPC as Educational Assistants in VR, 2021. [Online]. Verfügbar unter: https://publications.immersivelrn.org/index.php/practitioner/article/view/59

T. Weissker, Group Navigation in Multi-User Virtual Reality, Dissertation, Bauhaus-Universität Weimar, 2021. [Online]. Verfügbar unter: https://tim-weissker.de/preprints/2021-dissertation-group-navigation.pdf

J. Thomas, R. Bashyal, S. Goldstein und E. Suma, „MuVR: A Multi-User Virtual Reality Platform“, Proceedings of IEEE Virtual Reality 2014, 2014. [Online]. Verfügbar unter: https://illusioneering.cs.umn.edu/papers/thomas-vr2014.pdf

D. Lerner, S. Mohr, J. Schild, M. Göring und T. Luiz, „An Immersive Multi-User Virtual Reality for Emergency Simulation Training: Usability Study“, JMIR Serious Games, vol. 8, no. 3, e18822, 2020. [Online]. Verfügbar unter: https://games.jmir.org/2020/3/e18822/

N. van der Meer, V. van der Werf, W.-P. Brinkman und M. Specht, „Virtual Reality and Collaborative Learning: A Systematic Literature Review“, Frontiers in Virtual Reality, vol. 4, 1159905, 2023. [Online]. Verfügbar unter: https://doi.org/10.3389/frvir.2023.1159905

Z. Su et al., „Remote Keylogging Attacks in Multi-User VR Applications“, arXiv, 2024. [Online]. Verfügbar unter: https://arxiv.org/abs/2405.14036

N. Ross, „Introducing 'Horizon Workrooms': Remote Collaboration Reimagined“, Meta Blog, 2023. [Online]. Verfügbar unter: https://www.meta.com/blog/workrooms/

„Meta Removes Horizon Workrooms Collaboration Features“, XR Today, 2023. [Online]. Verfügbar unter: https://www.xrtoday.com/mixed-reality/meta-removes-horizon-workrooms-collaboration-features/

Skarredghost, „Spatial hands-on review: the most surprising XR meeting app“, 2020. [Online]. Verfügbar unter: https://skarredghost.com/2020/05/13/spatial-xr-meetings-review/

VRChat Inc., „VRChat Overview“, Wikipedia, zuletzt geändert am 7. Juli 2025. [Online]. Verfügbar unter: https://en.wikipedia.org/wiki/VRChat

Y. Sun, O. Shaikh und A. Stevenson Won, „Nonverbal synchrony in virtual reality“, PLOS ONE, 2019. [Online]. Verfügbar unter: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0221803

R. Xiao und F. Liu, „User Expectations for NPC Interactions and Nonverbal Behaviour in VR“, Proceedings of ACM CHI PLAY, 2024. [Online]. Verfügbar unter: https://dl.acm.org/doi/10.1145/3677098

P. Kourtesis, J. Linnell, R. Amir, F. Argelaguet und S. E. MacPherson, „Cybersickness in Virtual Reality Questionnaire (CSQ-VR): A Validation and Comparison against SSQ and VRSQ“, Virtual Worlds, vol. 2, no. 1, pp. 16–35, 2023. [Online]. Verfügbar unter: https://doi.org/10.3390/virtualworlds2010002

N. Biswas, „Are you feeling sick? – A systematic literature review of cybersickness in virtual reality“, ACM Digital Library, 2024. [Online]. Verfügbar unter: https://dl.acm.org/doi/full/10.1145/3670008

A. Mayer, I. Kilinc und K. Sprügel, „Towards Research Gaps in Collaborative Virtual Reality Environments for Education: A Literature Review“, ResearchGate, 2023. [Online]. Verfügbar unter: https://www.researchgate.net/publication/371706710_Towards_Research_Gaps_in_Collaborative_Virtual_Reality_Environments_for_Education_A_Literature_Review

S. van Gent et al., „An experimental comparison of participants' experience in face-to-face, video, and virtual reality meetings“, Frontiers in Virtual Reality, 2024. [Online]. Verfügbar unter: https://www.frontiersin.org/articles/10.3389/frvir.2024.1463189/full

N. Vij, „The Ultimate User Experience (UX) Guide“, Net Solutions, 2023. [Online]. Verfügbar unter: https://www.netsolutions.com/insights/ultimate-user-experience-guide/

S. Lee, T. Chen und A. Kumar, „Investigating co-presence and collaboration dynamics in realtime virtual reality user interactions: the CoCoVR study,“ Frontiers in Virtual Reality, 2024. [Online]. Verfügbar unter: https://www.frontiersin.org/articles/10.3389/frvir.2024.1478481/full

Y. Nong, H.-T. Zhang und J.-Q. Sun, „Natural Language Processing for Immersive Game Interactions: Improving NLP Models for More Natural Conversations with AI-driven NPCs in AR/VR Games,“ Engineering Open Access, vol. 3, no. 2, pp. 01–09, 2025. [Online]. Verfügbar unter: https://www.opastpublishers.com/open-access-articles/natural-language-processing-for-immersive-game-interactions-improving-nlp-models-for-more-natural-conversations-with-aidriven-npcs-8833.html

Unity Technologies. Netcode for GameObjects Manual (Version 1.8.1), 2024. Verfügbar unter: https://docs.unity3d.com/Packages/com.unity.netcode.gameobjects@1.8/manual/index.html

Unity Technologies. Unity Lobby Service Documentation, 2025. Verfügbar unter: https://docs.unity.com/ugs/manual/lobby/manual/unity-lobby-service

Unity Technologies. Unity Relay Documentation, 2025. Verfügbar unter: https://docs.unity.com/ugs/en-us/manual/relay/manual/introduction

Unity Technologies. Vivox Unity SDK Documentation: Voice and Text Chat for Unity, 2023. Verfügbar unter: https://docs.unity.com/ugs/manual/vivox-unity/manual/Unity/Unity#vivox-unity-sdk-documentation

Convai – Conversational AI Platform. Verfügbar unter: https://convai.com/

Convai. Setting Up Unity Plugin. Verfügbar unter: https://docs.convai.com/api-docs/plugins-and-integrations/unity-plugin/setting-up-unity-plugin

Oculus VR. „Introducing Oculus Quest, Our First 6DOF All-in-One VR System," Meta Horizon Developer Blog, 2018. [Online]. Verfügbar: https://developers.meta.com/horizon/blog/introducing-oculus-quest-our-first-6dof-all-in-one-vr-system/

Meta Platforms. „Introducing Oculus Quest 2, the Next Generation of All-in-One VR," Meta Newsroom, 2020. [Online]. Verfügbar: https://about.fb.com/news/2020/09/introducing-oculus-quest-2-the-next-generation-of-all-in-one-vr/

A. Robertson, „The Meta Quest 3 is sharper, more powerful, and still trying to make mixed reality happen," The Verge, 27.09.2023. [Online]. Verfügbar: https://www.theverge.com/2023/9/27/23890731/meta-quest-3-headset-hands-on-mixed-reality-connect

Convai Technologies, „Unity Plugin Integration Guide". [Online]. Verfügbar unter: https://docs.convai.com/api-docs/plugins-and-integrations/unity-plugin

ITU-T, „G.114: One-way transmission time", International Telecommunication Union, Geneva, Mar. 2003. [Online]. Verfügbar unter: https://www.itu.int/rec/T-REC-G.114

ITU-T, „J.1631: Performance Requirements for Cloud VR Services", International Telecommunication Union, Geneva, Nov. 2021. [Online]. Verfügbar unter: https://www.itu.int/rec/T-REC-J.1631

Wissenschaftliche Studie zur VR-basierten Meeting-Technologie, Google Forms, [Online]. Verfügbar unter: https://forms.gle/zwf37CLxByumjdk67

Vivox Unity SDK documentation - Class model diagram 15.1.25. Verfügbar unter: https://t.ly/D6E5W

8. Anhang

Der Anhang dieser Arbeit befindet sich auf dem der Printausgabe beigelegten Datenträger und umfasst:

- Das vollständige Unity-Projekt
- Videomaterial zur Erprobung durch die Nutzer
- Die Rohdaten der quantitativen Datensammlung sowie deren Analyse
- Die zur Datenerhebung und -analyse entwickelten C#-Skripte
- Die erstellte Online-Umfrage
- Die Gesamtübersicht aller gegebenen Antworten auf die Umfrage
- Sowie alle in dieser Arbeit verwendeten Diagramme und Visualisierungen

Eigenständigkeitserklärung

Ich erkläre hiermit ehrenwörtlich, dass ich die vorliegende Masterarbeit mit dem Titel

Entwicklung einer multifunktionalen Virtual-Reality-Plattform
für virtuelle Meetings und KI-gestützte Interaktionen mit NPCs

eigenständig und ausschließlich unter Verwendung der angegebenen Quellen verfasst habe. Alle übernommenen Inhalte aus der Literatur, dem Internet oder anderen Medien habe ich vollständig und korrekt kenntlich gemacht.

Hamburg, 14. Juli 2025 Ehsan Haghshenas