

# Entwicklung eines Natural User Interface mit Hilfe moderner AR und AI Technologien

## MASTERARBEIT

Angewandte Informatik

an der

Fachhochschule Erfurt

von

**Sebastian Rieger**

Abgabedatum 15.11.2017

Bearbeitungszeitraum	24 Wochen
Matrikelnummer	10286908
Betreuer der Masterarbeit	Prof. Rolf Kruse
Zweitbetreuer der Masterarbeit	Prof. Steffen Avemarg

## **Erklärung**

Ich, Sebastian Rieger, versichere hiermit, dass ich die vorliegende Masterarbeit mit dem Thema „Entwicklung eines Natural User Interface mit Hilfe moderner AR und AI Technologien“ selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe.

---

Ort      Datum

---

Unterschrift

# Inhaltsverzeichnis

<b>1 Abstract</b>	<b>5</b>
<b>2 Einleitung</b>	<b>6</b>
<b>3 Natural User Interfaces</b>	<b>7</b>
3.1 Natürliche Bedienung durch Gesten . . . . .	7
3.2 Erstellung eines Use Cases . . . . .	8
3.3 Auswertung des Use Cases . . . . .	9
<b>4 Augmented Reality</b>	<b>10</b>
4.1 Typen von visuellen Ausgabegeräten . . . . .	10
4.1.1 Handheld-Geräte . . . . .	11
4.1.2 Video-See-Through-Displays . . . . .	12
4.2 Auswertung der Möglichkeiten für ein Natural User Interface . . . . .	14
<b>5 Artificial Intelligence</b>	<b>16</b>
5.1 Sprachassistenten . . . . .	16
5.2 Funktionsweise der Sprachassistenten . . . . .	16
5.3 Untersuchung der Sprachassistenten . . . . .	17
5.3.1 Siri . . . . .	17
5.3.2 Alexa . . . . .	18
5.3.3 Cortana . . . . .	18
5.3.4 Google Assistent . . . . .	18
5.3.5 Auswertung der Möglichkeiten . . . . .	19
<b>6 Beschreibung eines Natural User Interface</b>	<b>20</b>
6.1 Einsatzgebiete . . . . .	20
6.1.1 Architektur Designer . . . . .	20
6.1.2 Bauen im Bestand . . . . .	20
6.1.3 Unterstützung von Rettungskräften . . . . .	21
6.1.4 Bestmögliches Beispiel . . . . .	21
6.2 Erstellung eines Architektur Editor als Prototyp . . . . .	22
6.3 Beschreibung der Gesten und Sprachkommandos . . . . .	23
6.3.1 Gesten . . . . .	23
6.3.2 Sprachkommandos . . . . .	25
6.4 Überprüfung der Gesten und Sprachkommandos . . . . .	27

6.4.1	Gesten . . . . .	27
6.4.2	Sprachkommandos . . . . .	28
<b>7</b>	<b>Versuch der Entwicklung eines Natural User Interface anhand der Microsoft HoloLens</b>	<b>29</b>
7.1	Entwicklungsgrundlagen . . . . .	29
7.1.1	Grundlagen . . . . .	29
7.1.2	Gestenentwicklung für die HoloLens . . . . .	30
7.1.3	Sprachkommandos für die HoloLens . . . . .	32
7.2	MixedRealityToolkit . . . . .	32
7.3	Spatial Mapping . . . . .	32
7.4	Das NUI Skript . . . . .	33
7.5	Der SpeechController . . . . .	35
7.6	Der WorldCursor . . . . .	36
7.7	Der UIController . . . . .	37
7.8	Probleme bei der Entwicklung . . . . .	37
7.8.1	Ungenauigkeiten bei der der Meshaufzeichnung . . . . .	37
7.8.2	Gestenlimitierung der HoloLens . . . . .	37
7.8.3	Sprachkommando Limitierung . . . . .	37
7.9	Auswertung des Programm in Hinsicht auf das geplante NUI . . . . .	38
<b>8</b>	<b>Fazit</b>	<b>39</b>
8.1	Augmented Reality . . . . .	39
8.2	Artificial Intelligence . . . . .	39
8.3	Allgemeine Machbarkeit . . . . .	40
<b>9</b>	<b>Zusammenfassung</b>	<b>41</b>
<b>10</b>	<b>Abkürzungsverzeichnis</b>	<b>42</b>

## 1 Abstract

## 2 Einleitung

In den letzten zwanzig Jahren hat sich die Bedienung von Computern grundlegend geändert. Vor nicht all zu vielen Jahren gab es nur die Möglichkeit mit Hilfe von Maus und Tastatur mit einem Computer zu interagieren.

Mit dem Aufkommen von Touch-Screens jedoch änderte sich auch die Benutzung von Computern. Es war nun möglich, direkt mit dem Bildschirm zu interagieren ohne den Umweg über die Maus.

Als dann wenig später die ersten Sprachsteuerungen auf den Markt kamen, änderte sich die Interaktion mit dem Computer erneut. So ist es nun möglich, Computern mittels Sprache Befehle zu erteilen oder Texte zu sprechen, welche automatisch transkribiert werden.

Heute ist es mit manchen Smartphones schon möglich, Nachrichten wie SMS zu schreiben und zu versenden, ohne das Telefon überhaupt in die Hand zu nehmen. Dies ist nur durch neuste Entwicklungen in der *künstliche Intelligenz* (KI) möglich.

Im selben Zeitraum hat sich parallel auch die *Virtuelle Realität* (VR) entwickelt. Sie versucht Menschen mit Hilfe von verschiedenen Brillen in eine virtuelle Welt zu versetzen. Die Entwicklungen solcher VR Systeme wurde vor allem von der Spieleindustrie getrieben, da sie immer neue Versuche unternimmt, Spieler besser in die Welt des Spiels zu versetzen.

Eine Abstufung der VR ist die *Augumented Reality* (AR), welche versucht die analoge, reale Welt um digitale Inhalte zu erweitern. Hierbei sind die Möglichkeiten für den Einsatz von AR fast unbegrenzt. Es ist also quasi möglich, jede menschliche Tätigkeit durch die AR zu unterstützen.

Genau hier setzt der Schwerpunkt der Arbeit an. Im Verlauf soll versucht werden, ein *Natural User Interface* (NUI) unter Zuhilfenahme moderner AR und KI Technologien zu erstellen.

Hierfür werden aktuelle AR und KI Systeme daraufhin untersucht, wie sie im Zusammenspiel ein NUI bilden können, welches allein durch Sprache und Gesten mit einem Computer interagiert.

Es soll eine übliche Tätigkeit mit Hilfe einer AR Technologie in der realen Welt abgebildet werden, welche dann von einem Menschen nur unter Zuhilfenahme von Sprache und Gesten gesteuert und bearbeitet werden kann.

### 3 Natural User Interfaces

Als Natural User Interfaces NUI werden Benutzeroberflächen von Computern und Programmen bezeichnet, welche sich durch Gesten, Sprache, Berühren, Tippen und Wischen bedienen lassen. In der heutigen Zeit begegnen wir ständig solchen NUIs, zum Beispiel beim mehrmals täglichen Griff zum Smartphone oder der Smartwatch. Über einen berührungsempfindlichen Display, werden Apps heute schon über Gesten gesteuert. [DP15]

Diese Steuerung funktioniert zum Teil bei der jüngeren Generation, so genannten „digital natives“, also Menschen welche im digitalen Zeitalter geboren sind, schon automatisch und unterbewusst. Jeder, der schon einmal eine App bediente kennt die Geste zum löschen aus einer Liste! Die Bewegung des Eintrags nach links oder rechts löscht das entsprechende Element getreu nach dem Motto „Aus den Augen aus dem Sinn“. [Dig17]

Diese Art von Gesten ist so intuitiv für Menschen, dass auch „digital immigrants“, also Menschen, die erst im Erwachsenenalter den Umgang mit digitalen Geräten gelernt haben, diese nach einmaliger Erklärung gelernt.

Aber das natürliche Verhalten von uns Menschen geht noch viel weiter. Diese einmal auf einem Smartphone gelernte Geste in einer App wird unterbewusst auch auf andere Applikationen übertragen. Eine App, welche diese einfache und mittlerweile etablierte Geste nicht unterstützt, wird umgehend Kritik ernten, da sie mit altbekanntem bricht und so ein „Natürliches“ arbeiten nicht mehr gegeben ist.

#### 3.1 Natürliche Bedienung durch Gesten

Was jedoch genau macht eine Geste „Natürlich“? Die Beantwortung dieser Frage ist im gleichen Maße trivial wie komplex.

Dies bedeutet, sobald einmal eine „Natürliche Geste“ gefunden wurde, erscheint wie gegeben und wird nicht in Frage gestellt, weil sie ja „Logisch“ ist. Das Komplexe ist jedoch, solche „logischen Gesten“, erst einmal zu finden.

Hierfür müssen viele Aspekte aus den Verhaltenswissenschaften beachtet werden:

- Arbeitswissenschaft
- Psychologie
- Soziologie
- Pädagogik

Ergänzend zu diesen Aspekten gibt es die EN ISO 9241, welche Richtlinien der Mensch-Computer-Interaktion beschreibt. Insbesondere der Teil 110 „Grundsätze der Dialoggestaltung“ kann helfen passende Bedienmöglichkeiten für ein NUI zu finden. [ISO11] Die Grundsätze sind:

- Aufgabenangemessenheit - geeignete Funktionalität, Minimierung unnötiger Interaktionen
- Selbstbeschreibungsfähigkeit - Verständlichkeit durch Hilfen / Rückmeldungen
- Steuerbarkeit - Steuerung des Dialogs durch den Benutzer
- Erwartungskonformität - Konsistenz, Anpassung an das Benutzermodell
- Fehlertoleranz - unerkannte Fehler verhindern nicht das Benutzerziel, erkannte Fehler sind leicht zu korrigieren

- Individualisierbarkeit - Anpassbarkeit an Benutzer und Arbeitskontext
- Lernförderlichkeit - Minimierung der Erlernzeit, Metaphern, Anleitung des Benutzers

### 3.2 Erstellung eines Use Cases

Anhand dieser Grundsätze kann nicht nur ein System, welches mit Tastatur und Maus bedient wird, definiert werden sondern auch ein NUI. Dies wird nun beispielhaft an der Geste für „löschen“ gezeigt, um sicher zu Stellen, dass im späteren Verlauf der Arbeit ein NUI anhand genau dieser Grundsätze definiert werden kann.

**Aufgabenangemessenheit** Das Wischen zum Löschen eines Eintrags ist schnell und einfach zu erlernen. Ebenso kann die Geste schnell wiederholt werden, um möglichst schnell auch mehrere Elemente löschen zu können.

**Selbstbeschreibungsfähigkeit** Die Geste ist zwar einfach und mittlerweile auch bekannt, aber es muss auch sichergestellt sein, dass neue Nutzer die Geste lernen können. Dies kann zum Beispiel durch ein kurzes wackeln beim ersten öffnen angezeigt werden. Dem Nutzer wird durch diese Bewegung suggeriert, dass diese Elemente bewegbar sind.

**Steuerbarkeit** Wischt der Nutzer nun bewusst oder durch Neugier ausgelöst vom Wackeln das Element aus dem Bildschirm, muss immer die Möglichkeit bestehen, das Element wiederherstellen zu können. Dies ist nur konsequent, da eine so einfache Geste auch einmal versehentlich gemacht werden kann.

**Erwartungskonformität** Ist diese Geste einmal etabliert, muss sie konsequent in allen Listen implementiert sein, da es sonst zum Bruch der Erwartung kommt und der Nutzer unnötiger verwirrt wird.

**Fehlertoleranz** Zu Fehlertoleranz gehört das wiederherstellen eines Eintrags wie schon unter Steuerbarkeit beschrieben. Zusätzlich kommt jedoch hinzu, dass die Geste eventuell vom Gerät falsch interpretiert wurde und der Nutzer gar nichts löschen wollte. Auch in solchen Fällen muss eine Wiederherstellung möglich sein.

**Individualisierbarkeit** Das Wischen ist eine tolle Geste! Aber in welche Richtung muss gewischt werden? Nach links oder rechts wischen zum Löschen? Genau hier muss die App schlau reagieren, denn die richtige Antwort ist: Beides muss möglich sein. Bekanntermaßen gibt es Linkshänder und Rechtshänder, wobei das Gerät zumeist mit der dominanten Hand gehalten wird. Für einen Rechtshänder, welcher den Daumen zum wischen links hat, ist das Wischen nach rechts leichter als nach links. Genau umgekehrt ist es bei Linkshändern. Diese wischen eher nach Links.

Dieses Verhalten kann auch selbst nachgestellt werden, egal ob der Nutzer Links- oder Rechtshänder ist. Die Geste wird unterbewusst durch beide Hände in unterschiedliche Richtungen ausgeführt.

**Lernförderlichkeit** Bei der Lernförderlichkeit trifft ganz klar das Sprichwort „Aus den Augen aus dem Sinn, zu. Der Nutzer möchte etwas „weg, haben. Warum es also nicht außerhalb des Displays platzieren? Es wird nicht mehr gesehen und ist somit „weg,. Diese einfache Analogie ist für jeden Nutzer zu verstehen und umzusetzen.“

### **3.3 Auswertung des Use Cases**

Anhand des vorgestellten Use Cases wurde gezeigt, dass es möglich ist ein NUI zu designen. Hierbei müssen natürlich die Gegebenheiten des jeweiligen Gerätes beachtet werden, wie zum Beispiel, dass ein Smartphone in beiden Händen gehalten werden kann. Dadurch kann sich gegebenen Falls der Use Case erweitern oder verändern.

1991 beschrieb Mark Weiser schon, wie sich Computer immer mehr in unseren Alltag integrieren und langsam immer unkenntlicher werden. Die Grenze zwischen realer Umwelt und virtueller Welt verschwimmt heute immer mehr. [Wei91]

Dies hat natürlich auch zur Folge, dass sich die Schnittstelle zwischen Mensch und Computer ständig ändert und weiterentwickelt. Musste man 1997 zu Beginn von Google noch Stichworte mit Maus und Tastatur eingeben um das Internet zu durchsuchen, reicht heute schon ein Sprachbefehl.

Der Computer wertet mit verschiedenen Algorithmen das gesprochene aus und versucht dem Nutzer eine direkte Antwort zu geben, während man 1997 von Google eine Liste mit Webseiten bekam, auf welchen die Schlagworte zu finden waren.

## 4 Augmented Reality

In der heutigen Zeit ist fast jedem der Begriff Augmented Reality geläufig. Jedoch gibt es im wissenschaftlichen Umfeld keine einheitliche Definition. Georg Klein definiert die AR als „Anreicherung der realen Welt um computergenerierte Zusatzobjekte“. [Kle]

Viele Abhandlungen zu diesem Thema beziehen sich auf das „Reality-Virtuality Continuum“, welches von Milgram, Takemura, Utsumi und Kishino beschrieben wird (Abbildung 1). Dieses stellt bildlich dar, wie sich eine AR-Anwendung einordnen lässt. Links ist die reale Umgebung (Real Environment) und rechts die Virtuelle Umgebung (Virtual Environment) zu sehen.

Heutige VR-Anwendungen müssen also ganz rechts eingeordnet werden. Zwischen der realen und der virtuellen Umgebung gibt es jedoch noch weitere Abstufungen. So gibt es die „Augmented Reality“, welche die reale Welt um virtuelle Elemente ergänzt. Sie bezieht sich eher auf die reale Umgebung, weshalb sie weiter links angeordnet ist. Die „Augmented Virtuality“ ist das genaue Gegenteil der AR. Hier wird die virtuelle Welt des Computers um reale Gegenstände ergänzt.

Im wissenschaftlichen Umfeld wird als Oberbegriff für „Augmented Reality“ und „Augmented Virtuality“ oft „Mixed Reality“ oder „Enhanced Reality“ verwendet. [MBRS11]

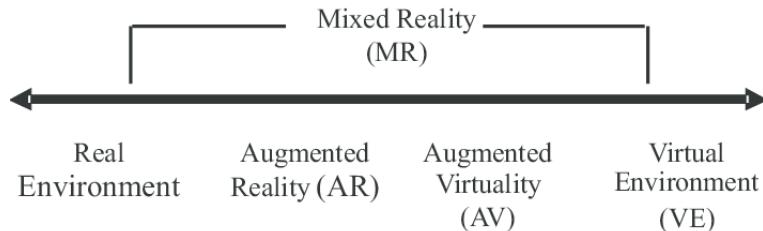


Abbildung 1: Reality-Virtuality Continuum [MTUK]

Der Begriff der *Augmented Virtuality* (AV) ist heute weniger gebräuchlich, da es keine wirklichen Anwendungsfelder gibt, in denen eine virtuelle Welt um reale Gegenstände ergänzt werden muss. Im Umkehrschluss wird heute jedoch die AR immer bedeutender, da bei fast jeder menschlichen Tätigkeit eine Unterstützung durch Computergenerierte Objekte oder Informationen möglich ist.

Im medizinischen Umfeld können Ärzten zum Beispiel Positionen von empfindlichen Gewebe oder zu entfernenden Fremdkörpern bei einer Operation angezeigt werden. Eine andere Einsatzmöglichkeit wäre das unterstützende Anzeigen von Gefahren oder Fluchtwegen für Feuerwehrleute, die sich in einem stark verrauchten Brandobjekt befinden. Computer können durch Infrarot und Wärmebild das Sichtvermögen eines Feuerwehrmannes in einer Gefahrensituation so entscheidend erhöhen, um sich selbst und andere zu retten.

Ein Gerät, welches später für die Erstellung eines NUI dienen kann, muss folgende Kriterien erfüllen:

- Das Gerät muss Freihändig bedienbar sein.
- Das Gerät muss die AR-Inhalte exakt zur realen Welt wiedergeben.
- Das Gerät muss Handgesten und Sprache des Nutzers erkennen können.

### 4.1 Typen von visuellen Ausgabegeräten

Im folgenden Abschnitt werden mögliche visuelle Ausgabegerätetypen genauer beschrieben und auf ihre Einsatzmöglichkeiten untersucht.

#### 4.1.1 Handheld-Geräte

Als Handheld werden in Geräte bezeichnetet, die wie der Name schon sagt von einer Person in der Hand gehalten werden. Im AR-Umfeld sind hier Smartphones oder Tablets gemeint, welche heutzutage durch ihre Dual- oder Quad-Core-Prozessoren und teilweise mehrere Gigabyte RAM in der Lage sind, auch komplexere AR-Anwendungen zur Ausführung zu bringen.

In Smartphones und Tablets nimmt die Kamera das Bild der realen Welt auf und stellt es in Echtzeit auf dem Display dar. Ein Programm (App) ist dann in der Lage, die aufgenommene reale Welt um virtuelle Elemente zu erweitern und diese innerhalb des Kamerabildes auf dem Bildschirm zu platzieren.

Hierbei ergibt sich jedoch ein Problem, da sich der Blickwinkel des Betrachters vom Blickwinkel der Kamera unterscheidet, was zu einer unsauberen Platzierung der Objekte im Bild führt. In Abbildung 2 ist zu sehen, wie sich der Blickwinkel der Kamera und der Blickwinkel des Betrachters unterscheiden.

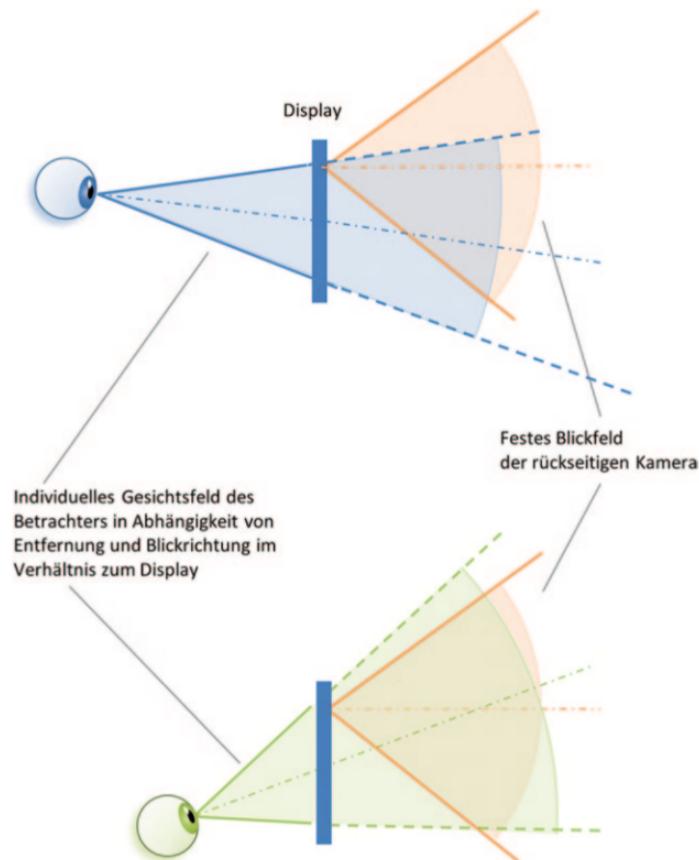


Abbildung 2: Unterschiedliche Blickfelder von Betrachter und Kamera bei Handheld-AR [DBGJ13]

Um den Unterschied zwischen den Blickwinkeln auszugleichen, muss das entsprechende Gerät die beiden Blickwinkel aufeinander abstimmen. Dies kann jedoch kein zur Zeit auf dem Markt befindliches Gerät leisten.

Einen ersten Ansatz hierzu hat die University of California gemacht, indem eine Gruppe Forscher ein Tablet mit einem Kinect Sensor verbunden haben. Mit Hilfe des zusätzlichen Sensors und dem „KinectFusion Algorithmus“ konnten sie eine „Magic Lens“ entwickeln, die beide Blickwinkel in etwa aufeinander abstimmt. [BLT<sup>+</sup>12]

In Abbildung 3 ist einmal der Unterschied zwischen einer Aufnahme mit Magic Lens und ohne schematisch dargestellt.

Im linken Bildabschnitt ist gut zu sehen, dass Bäume und andere Gegenstände im Bildschirm exakt an der Position in der realen Welt liegen. Im Gegensatz dazu sind Bäume und andere Gegenstände im rechten Bildabschnitt merklich verzerrt.

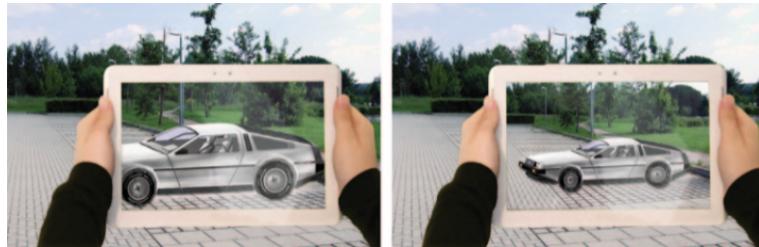


Abbildung 3: Schematische Darstellung des Magic Lens Effekts [DBGJ13]

Ein weiteres Problem ist, das AR-Anwendungen für die optimale Skalierung und Platzierung von virtuellen Objekten im Raum zusätzlich Tiefeninformationen des aktuellen Bildes benötigen, denn nur dann ist auch gewährleistet, dass virtuelle Objekte in der richtigen Größe im realen Raum positioniert werden.

Dieses Problem versucht gerade das Projekt Tango von Google zu lösen. Mit Hilfe eines Lasers und einer Infrarotkamera können Tiefeninformationen zum aktuellen Bild aufgenommen und verarbeitet werden. [Goo17]

Nur mit der Tango Technologie ist es bisher möglich, AR-Objekte richtig skaliert und im richtigen Blickwinkel präzise im Raum zu positionieren. Weitergehende Informationen sind auf den Developer-Seiten<sup>1</sup> von Google zu finden. Bisher unterstützen jedoch nur das „Lenovo Phab2“ und das „Asus ZenFon AR“ diese Technologie, da hier eine zusätzliche Sensoreinheit im Smartphone verbaut sein muss.

Abschließen kann über AR-Anwendungen ausgesagt werden, dass sie grundsätzlich lauffähig auf Handheld Geräten wie Tablets oder Smartphones sind. Die Bedingungen sind heutzutage jedoch noch nicht optimal, wie eben dargelegt wurde.

#### 4.1.2 Video-See-Through-Displays

Video-See-Through Geräte existieren im Vergleich zu AR-Anwendungen schon ziemlich lange. Schon Anfang der 1940er Jahre wurden erste Head-Up-Displays in Kampfflugzeugen eingesetzt, um die Piloten mit zusätzlichen Informationen zu versorgen, welche immer im Blick behalten werden müssen. [Wik17a]

Von Anfang an, wurden die zusätzlichen Informationen auf eine Glasfläche projiziert, um das Sichtfeld nicht unnötig einzuschränken. Es wurde entweder direkt auf das Cockpit Fenster projiziert oder die Piloten hatten kleine durchsichtige Displays in den Helm eingearbeitet, auf die mit Hilfe von Spiegeln projiziert werden konnte.

Mit der Veröffentlichung des iPhones im Jahre 2007 wurde das Thema AR immer interessanter, da mit ihm die technische Grundlage gelegt wurde. 2008 erschien der AR-Browser Wikitude wodurch das Feld immer mehr Fuß fasste und sich nun rasant weiter entwickelt. [Jöc14]

Die durch die immer größerer Verbreitung von AR-Anwendungen wurde auch die Hardware entsprechend weiterentwickelt und angepasst, was zu den ersten AR-Brillen führte.

Durch die Kombination von Handheld-Geräten und Head-Up-Displays entstanden bis 2014 die ersten Video-See-Through-Brillen oder auch AR-Brillen genannt. Die Entwicklung dieser Brillen

---

<sup>1</sup><https://get.google.com/tango/>

steckt noch in den Kinderschuhen, aber dennoch gibt es schon zwei auf dem Markt verfügbare Geräte, die für die spätere Entwicklung eines NUI in Betracht gezogen werden können. Zum einen ist das die Epson Moverio BT-200 und zum anderen die Microsoft HoloLens, welche zur Zeit nur für Entwickler verfügbar ist.

Die HoloLens und die Moverio sind beide Prismen basiert. In Abbildung 4 ist die Funktionsweise von Prismen basierten See-Through-Displays schematisch dargestellt. Ein normales Display ist an einer Seite eines Prismas angebracht. Das Prisma leitet das Licht des Displays weiter, so dass es in das Auge des Betrachters weitergeleitet wird. Weil das Prisma nach vorn durchsichtig ist, wird auch das Umgebungslicht direkt in das Auge des Betrachters geleitet. Innerhalb des Prismas vermischen sich die beiden Bilder, wobei das Displaylicht das Umgebungslicht überlagert. Für den Betrachter sieht es somit aus, als würden die Objekte in der realen Welt zu finden sein.

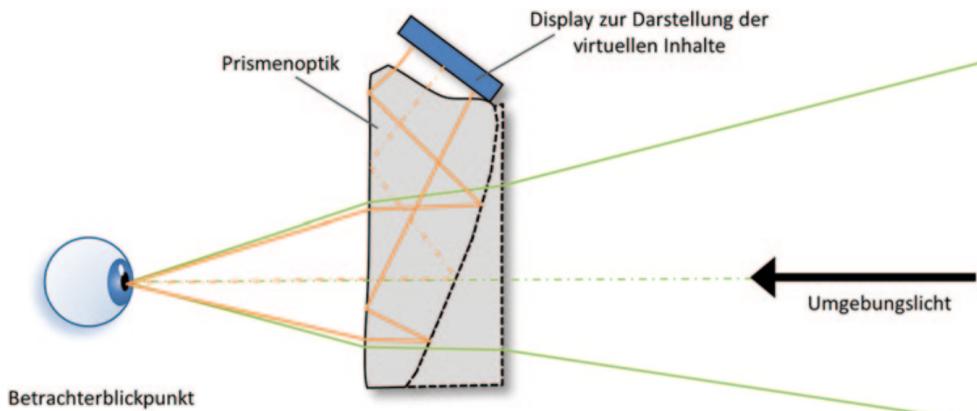


Abbildung 4: Schematische Darstellung des Aufbaus eines See-Through-Displays [DBGJ13]

**Die Microsoft HoloLens** wurde 2015 von Microsoft vorgestellt und ist momentan die wahrscheinlich beste AR-Brille auf dem Markt. Sie verfügt über mehrere Kameras und Tiefensensoren, womit die HoloLens in der Lage ist, sich selbstständig im Raum zu orientieren.

Ausgestattet mit einem Intel Atom x5-Z810 Prozessor und 2 Gigabyte RAM, ist die Brille sehr gut aufgestellt und kann Programme und Berechnungen ohne zusätzliche Hardware ausführen. Als Betriebssystem kommt Windows 10 zum Einsatz. Die Akkulaufzeit wird mit zwei Stunden angegeben, wobei dies wie immer abhängig von der geforderten Leistung ist. [ARV17]

Für Entwickler hat Microsoft ein *Software Development Kit* (SDK) bereitgestellt mit dessen Hilfe es möglich ist Anwendungen für die HoloLens zu entwickeln. [Hol17a]

Über verschiedene Sensoren kann die HoloLens Bewegungen und Gesten des Nutzers wahrnehmen, erkennen und verarbeiten.

Die HoloLens ist standardmäßig in der Lage, grundlegende Gesten zu erkennen, mit denen zum Beispiel ein Tippen, Halten, Rotieren und Zoomen möglich ist. [Hol17a] Auf genauere technische Details zur Programmierung der HoloLens wird im Kapitel ???? genauer eingegangen.

Eine Besonderheit der HoloLens ist natürlich das mit Microsoft ein großer Softwarekonzern hinter der Entwicklung steht und somit ein schnelles voranschreiten der Technik quasi sicher ist.



Abbildung 5: Aufgesetzte HoloLens [Wik17f]

**Die Epson Moverio BT-200** eine weitere AR-Brille ist die sogenannte Epson Moverio BT-200, welche auf Android-Basis arbeitet. Im Gegensatz zur HoloLens, kann die Moverio wie eine normale Brille getragen werden und sieht auch so aus.

Intern sorgt ein nicht genauer benannter 1,2 GHz Dual-Core-Prozessor für die nötige Rechenleistung. Die Brille verfügt über 8 Giabyte Speicher, der sich mittels SD-Karten erweitern lässt.

Ähnlich wie die HoloLens verfügt auch die Moverio über eine Tiefebildkamera, welche es ermöglicht AR-Inhalte exakt zu platzieren. Ein Beschleunigungssensor registriert zusätzlich die Bewegungen des Kopfes des Trägers.

Gesteuert wird die Brille über ein Controller-Kästchen, welche im Lieferumfang enthalten ist. Dieses verfügt zur Steuerung verschiedene Button wie Home oder Zurück, welche auch von Android bekannt sind. Zusätzlich kann das System über einen eingebauten Touchscreen im Controller Gesten des Nutzers erkennen und interpretieren.

Der Akku soll 6 Stunden durchhalten, was ein längeres Vergnügen verspricht, als es bei der HoloLens gegeben ist.

Der „Google Play“-Store ist auf der Brille nicht verfügbar, jedoch portiert Epson einige Anwendungen damit diese auf der Brille lauffähig sind.



Abbildung 6: Epson Moverio mit Controller [VRS16]

## 4.2 Auswertung der Möglichkeiten für ein Natural User Interface

Für die Erstellung eines NUI gibt es frontendseitig zwei Möglichkeiten. Zum einen gibt es die Handheld-Geräte, zum anderen gibt es noch die AR-Brillen.

Als Handheld-Gerät kann heute wie dargelegt fast schon jedes Smartphones oder Tablet verwendet werden. Ohne eine eingebaute Magic-Lens 4.1.1 oder die Google Tango Technologie, ist das präzise arbeiten mit solchen Geräten nicht möglich. Hinzu kommt noch, dass der Benutzer des Interfaces immer mit einer Hand das entsprechende Gerät halten muss, wodurch diese dann nicht mehr zum Arbeiten zur Verfügung steht. In der Tabelle 1 wurden alle Handheld Geräte zusammen betrachtet, da es hier praktisch keine Unterscheide gibt.

Anforderung	Priorisierung	Handheld Geräte	HoloLens	Moverio BT-200
Freihändig bedienbar	sehr hoch	X	✓	X
exakte AR-Inhalte	hoch	X	✓	✓
Handgestenerkennung	sehr hoch	✓	✓	✓
Spracherkennung	sehr hoch	✓	✓	✓

Tabelle 1: Tabellarischer Vergleich der betrachteten Geräte

Das ständige halten eines Smartphones oder Tablets ist auch anstrengend für den Nutzer und seine Armmuskulatur.

Handheld-Geräte sind eine gute und auch einfache Lösung um Filme zu schauen oder Spiele zu spielen. Eine Langzeitnutzung wie in einem 8 stündigen Arbeitstag steht jedoch außer Frage.

Hier kommen die AR-Brillen ins Spiel. Sie ermöglichen eine Anreicherung der realen Welt ohne dass ein Nutzer permanent ein Gerät halten muss, da die Brillen durch die natürliche Kopfform halten.

Die Epson Moverio Brille ist ein erster Ansatz um AR-Brillen einer breiten Masse zur Verfügung zu stellen. Jedoch kann sie durch viele Nachteile nicht mit der HoloLens mithalten. Im Gegensatz zur Moverio kann sich die HoloLens selbstständig im Raum orientieren.

Ein weiterer K.O.-Punkt ist, dass die HoloLens Handgesten des Nutzers direkt erkennt, ohne wie bei der Moverio auf ein zusätzlichen Controller zurück greifen zu müssen.

Im Vergleich zu Handheld-Geräten ist die Moverio nicht besser, da auch ständig ein Gerät zur Steuerung in der Hand des Nutzers verbleiben muss und somit nicht frei für andere Aufgaben ist.

Die Microsoft HoloLens ist bisher die einzige Möglichkeit um ein NUI zu entwickeln, wie es die Arbeit vorsieht. Dies geht auch aus Tabelle 1 hervor. Sie kann völlig freihändig gesteuert werden, ohne das der Nutzer zusätzliche Hardware zum Steuern oder Anzeigen in der Hand halten muss.

Im Verlauf der Arbeit wird durch die klare Überlegenheit der HoloLens diese als einzige weiter betrachtet und für die Entwicklung verwendet. Alle anderen Alternativen konnten nicht überzeugen.

## 5 Artificial Intelligence

*Artificial Intelligence* (AI) oder im deutschen auch *artifizielle Intelligenz* oder *künstliche Intelligenz* bezeichnet in der Informatik ein System, welches eine automatisierte Intelligenz besitzt. Eine eindeutige Definition für AI gibt es nicht, da es schon in der Psychologie an einer genauen Definition mangelt. [Wik17b]

In der heutigen Zeit werden Systeme und Algorithmen, die auf neuronalen Netzen basieren, als annähernd intelligent bezeichnet. Ein Beispiel hierfür ist „Google Translate“, welches zwischen verschiedenen Sprachen mittels neuronalen Netz übersetzt. [Mer16]

Es gibt aber auch intelligente Systeme wie „IBM Watson“, welche nicht auf neuronalen Netzen basieren und annähernd intelligent wirken. Diese Intelligenz ist das Resultat einer sehr großen Wissensbasis und verschiedenen kombinatorischen Algorithmen. Um zu zeigen, wie Leistungsfähig Watson ist, trat er 2011 in der Quizshow „Jeopardy“ an und gewann gegen die weltbesten Spieler. [IBM17]

### 5.1 Sprachassistenten

Im allgemeinen Sprachgebrauch werden als AI die heute üblichen Sprachassistenten wie Siri, Alexa, Cortana oder Google Assistent bezeichnet. Das im Verlauf der Arbeit zu entwickelnde NUI soll auf einen dieser Assistenten aufbauen.

Die Entwicklung einer eigenen Spracherkennung ist im Rahmen dieser Arbeit nicht möglich, da dies sehr viel Zeit und Arbeit in Anspruch nimmt. Selbst Firmen wie Google oder Amazon benötigten mehrere Jahre um ihre Assistent auf das aktuelle Leistungsniveau zu heben.

### 5.2 Funktionsweise der Sprachassistenten

Auch wenn die genauen Implementierungen der Sprachassistenten verschiedenen sind, so haben sie alle eins gemeinsam und zwar ist dies der Weg der Sprachverarbeitung.

Die Funktionsweise von Siri und Co. ist recht einfach gehalten. Der Nutzer löst mit einem Stichwort, „Hey, Siri...“, „Ok, Google...“ und so weiter (Abbildung 7 1.), die Benutzung aus. Der darauf folgende Sprachbefehl wird aufgezeichnet und komprimiert an einen Server weitergeleitet (Abbildung 7 2.), welcher den gesprochenen Text auswertet und in schriftliche Befehle umwandelt (Abbildung 7 3.). Diese Befehle werden dann umgehend zurück an das Endgerät des Nutzers versand (Abbildung 7 4.), welches nun auf die Befehle reagieren kann (Abbildung 7 5.). In Abbildung 7 ist dieses vorgehen noch einmal schematisch Dargestellt. [Ebe17]

?????Wie sieht es besser aus? Verweis auf Abb. oder einzelnen Punkte angeben? oder punkte ganz weglassen? ?????

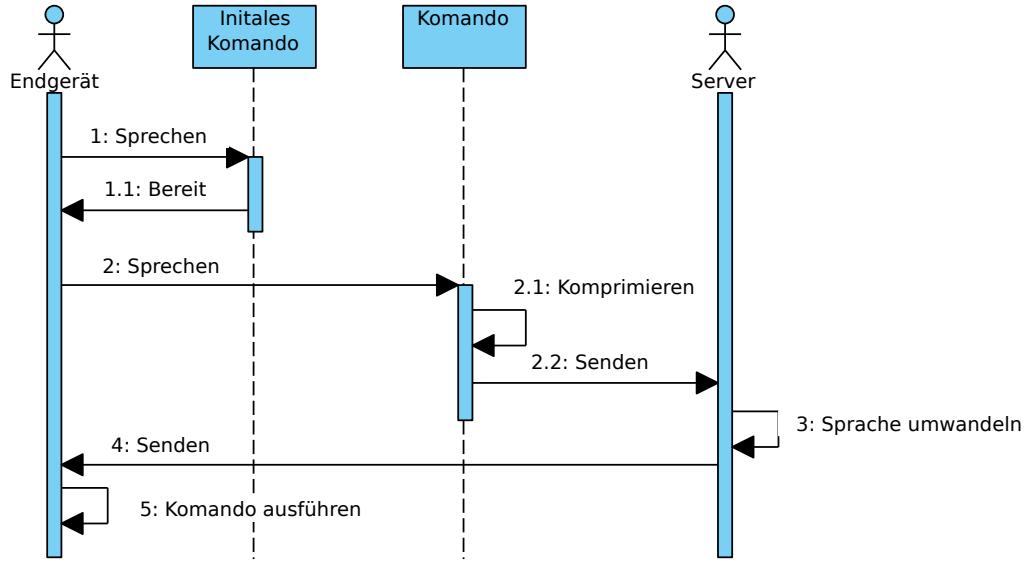


Abbildung 7: Schematische Darstellung der Kommunikation von Sprachassistenten

Hierbei ist zu beachten, dass jegliche Kommunikation zwischen Endgerät und Server bei allen Assistenten über HTTPS gesichert ist.

Innerhalb der Server, welche die Sprache zu Text wandeln, kommen fast immer neuronale Netze zum Einsatz, da nur sie in der Lage sind, schnell und eindeutig die Sprachbefehle zu wandeln. Durch die gehäufte Verwendung mit immer neuen Stimmen und Befehlen lernen die Systeme immer mehr hinzu und können so die Anfragen immer schneller und besser umwandeln. Dies hat zur Folge, dass ein System besser wird, je häufiger es genutzt wird.

In den nun folgenden Abschnitten werden die genannten Assistenten auf ihre Möglichkeiten untersucht.

### 5.3 Untersuchung der Sprachassistenten

Es soll nun untersucht werden, in wie weit die heute gängigen Assistenten genutzt werden können, um eine Sprachsteuerung für ein NUI (siehe Kapitel ?????) umzusetzen.

Hierbei muss der Assistent folgende Kriterien erfüllen:

- Lauffähig auf der Microsoft HoloLens
- Offene Programmierschnittstelle (SDK)
- Möglichkeit eigener Sprachkommandos

#### 5.3.1 Siri

Der wohl älteste und bekannteste Sprachassistent ist „Siri“. Siri ist eine Software zur Spracherkennung und wurde von Apple im Jahre 2010 zusammen mit der Firma Siri Inc. von Apple aufgekauft. Schon ein Jahr später wurde der Assistent im Zuge der Veröffentlichung des iPhone 4s vorgestellt und veröffentlicht.

Siri ist eine proprietäre Software und nur iOS, macOS, watchOS und tvOS einsetzbar, weshalb eine Verwendung auf der HoloLens nicht möglich ist. Zwar verfügt Siri über ein offenes SDK, dieses ist jedoch nur für Apple eigene Betriebssysteme ausgelegt. [Sir17a] [Sir17b]

### 5.3.2 Alexa

Alexa ist der Name des Sprachassistenten, welcher von Amazon entwickelt wird und im November 2014 veröffentlicht wurde. Seither entwickelt Amazon seinen digitalen Assistenten stetig weiter. [Wik17c]

Grundsätzlich funktioniert auch Alexa wie schon im Abschnitt ???? beschrieben. Alexa arbeitet mit sogenannten Skills. Diese ermöglichen es Entwicklern, den Assistent mit neuen Funktionen zu erweitern. Will ein Nutzer diesen Skill verwenden, muss er ihn zunächst innerhalb seines persönlichen Assistenten installieren. Danach beherrscht Alexa die entsprechenden Kommandos des Skills. [Ama17b]

Die eigentlichen Skills, also kleine Programme, welche innerhalb der Amazon Cloud ausgeführt und durch den Assistenten gestartet werden, können entweder in Java oder Node.js geschrieben werden. Um diese später zu veröffentlichen muss der Entwickler über ein Amazon Developer Account verfügen.

Amazon verfolgt zur Zeit eine große Marketing-Offensive welche für verschiedene Produkte der Amazon Echo Serie, welche Alexa integriert haben. Zusätzlich ist es möglich, Alexa auch auf eigenen Geräten verfügbar zu machen. Hierfür wird das *Amazon Voice Service* (AVS) benötigt. Dieses SDK ermöglicht es den Amazon Assistent auch auf Geräten verfügbar zu machen, welche nicht von Amazon hergestellt wurden. [Ama17a]

Mit den Skills und dem AVS-SDK, welches C++ basiert ist, ist es mögliche, alle Geräte die ein Mikrofon besitzen mit dem Amazon Assistenten auszustatten.

Ein weiterer Vorteil von Alexa sind die sogenannten Speechcons. Diese werden speziell von Amazon zur Verfügung gestellt, um eine möglichst natürliche und vor allem richtig betonte Aussprache zu gewährleisten. Speechcons sind kurze Phrasen, welche in der Regel Umgangssprachlich genutzt werden, wie zum Beispiel der Ausspruch „ich glaub mein Schwein pfeift“ oder „Puste-kuchen“. Zusätzlich steht mit „Whispers“ auch ein Modus bereit, welcher Alexa flüstern lässt. [Pak17] [Ale17]

All dies macht Alexa zumindest für den Zuhörer momentan zu dem besten Assistenten auf dem Markt.

### 5.3.3 Cortana

Nutzer von „Windows 10“, einer Xbox oder eines Windows Phone kennen sie bereits, „Cortana“, den digitalen Assistenten, welcher von Microsoft entwickelt wurde. Er wurde im Jahr 2014 veröffentlicht und steht neben den genannten Microsoft Geräten auch für Android und iOS zur Verfügung. Grundsätzlich kann Cortana eine Suche starten, Programme ausführen oder Termine und Notizen verwalten. [Wik17d]

Über Skills können Entwickler ähnlich wie bei Alexa zusätzliche Funktionen erstellen und für Nutzer freigeben. Das von Microsoft bereitgestellte Skills Kit ist momentan noch nicht freigegeben und befindet sich aktuell in einer öffentlichen Beta-Phase.

Skills für Cortana werden ähnlich wie die von Alexa in der Clound ausgeführt und können in .NET oder Node.js geschrieben werden. [Mic17]

### 5.3.4 Google Assistent

Der Google Assistant ist ein Assistent, welcher von Google entwickelt und 2016 als Nachfolger von Google Now erschienen ist. [Wik17e]

Eine Besonderheit dieses Assistenten ist, dass er Fragen auch aus dem Kontext früher gestellter Fragen beantworten kann. So kann ein Nutzer zum Beispiel fragen „In welchem Verein spielt Manuel Neuer?“. Wenn als nächstes die Frage „Wie alt ist er?“ folgt, kann der Google Assistant aus dem Kontext schließen, dass ebenfalls „Manuel Neuer“ als „er“ gemeint ist. [SGS17]

Der Google Assistant ist durch eine Vielzahl von SDKs wie zum Beispiel Android, Apple, C++, Python und weitere unter quasi allen Plattformen lauffähig. [API17]

Mit sogenannten „Agents“ ist es möglich, ähnlich wie mit Alexa Skills, über ein Web-Interface neue Funktionen zu erstellen.

### 5.3.5 Auswertung der Möglichkeiten

Nach dem Überblick über die vier Assistenten Siri, Alexa, Cortana und Google Assistant wird nun ausgewertet, welcher am besten für die Erstellung eines NUI geeignet ist.

Im Kapitel 4 wurde die Microsoft Hololens für die Erstellung eines NUI favorisiert. Auf ihr muss also der Assistent lauffähig sein. Da Apple für Siri nur ein SDK unter iOS bereitstellt ist es nicht möglich, diesen Assistent auf der Hololens zu verwenden. Alexa, Cortana und der Google Assistant können auf der Hololens aufsetzen. Dies funktioniert durch die bereitgestellten SDKs zumindest theoretisch und muss in der Praxisumsetzung der Arbeit noch einmal geprüft werden.

Alle Assistenten verfügen über ein offenes SDK, welches kostenlos verwendet werden kann. Lediglich bei der Ausführung von Befehlen innerhalb der Cloud (Skills) fallen Kosten an. Hierbei rechnen alle Hersteller nach Anzahl der Ausführungen ab und nicht nach Laufzeit oder Rechenleistung.

Siri, Alexa und der Google Assistant haben jeweils die Möglichkeit, eigene Funktionen zu erstellen und diese innerhalb von Programmen zur Ausführung zu bringen. Durch das Web-Interface des Google Assistant können zwar eine Vielzahl von Funktionen erstellt werden, jedoch sind die Möglichkeiten bei Alexa zumindest aktuell um ein vielfaches größer, da hier Java oder Node.js Quellcode zur Ausführung kommt. Siri kann Befehle verarbeiten und Funktionen innerhalb einer installierten App aufrufen. Jedoch ist die Funktionalität hierauf beschränkt. Lediglich Apple kann eigene neue Funktionen für Siri bereitstellen.

Microsoft arbeitet zur Zeit an einem Skill SDK für Cortana, welches ähnlich wie der Amazon Ansatz aussieht. Für Cortana kommt in der Cloud .NET oder Node.js Code zur Ausführung. Aktuell befindet sich dieses SDK in einer öffentlichen Beta-Phase und die Schnittstellen sind noch nicht Final.

In der Tabelle 2 ist dies noch einmal zusammengefasst.

Anforderung	Priorisierung	Siri	Alexa	Cortana	Google Assistant
Lauffähig auf der Microsoft HoloLens	sehr hoch	X	✓	✓	✓
Offene Programmierschnittstelle (SDK)	hoch	✓	✓	✓	✓
Möglichkeit eigener Sprachkommandos	sehr hoch	✓	✓	X	✓

Tabelle 2: Tabellarischer Vergleich der Assistenten

Aufgrund der vorangegangenen Analyse zeigte sich, dass Alexa momentan die besten Möglichkeiten ist um in ein NUI integriert zu werden. Hierfür spricht vor allem das AVS-SDK, welches schnell kompiliert und einsatzbereit ist. Die zweite Wahl wäre der Google Assistant mit seiner Vielzahl von SDKs. Sollten beide Möglichkeiten technisch nicht umsetzbar sein, so muss auf Cortana zurückgegriffen werden. Hier ist eine Lauffähigkeit durch Microsoft garantiert. Jedoch ist das Skill SDK noch nicht final veröffentlicht.

## 6 Beschreibung eines Natural User Interface

Im folgenden Kapitel werden verschiedene mögliche Einsatzgebiete für ein NUI mit aktuellen AR und AI-Technologien beschrieben. Anhand eines Einsatzgebietes wird dann ein NUI genauer beschrieben und designed.

### 6.1 Einsatzgebiete

Um überhaupt ein NUI erstellen und beschreiben zu können, muss ein spezielles Szenario gewählt werden, an das Gesten und Sprachkommandos angepasst werden.

#### 6.1.1 Architektur Designer

Schon heute gibt es unzählige Programme, mit dem Gebäude designt werden können. Eines davon ist „Architekt 3D“, mit dem nicht nur Modelle erstellt werden können, sondern auch Grund- und Auf-risse erzeugt werden können.



Abbildung 8: Darstellung eines Hauses in Architekt 3D [Arc17]

In Abbildung 8 ist die Darstellung eines Hauses zu sehen, welches mit „Architekt 3D“ erstellt wurde. In einer AR-Umgebung mit einem NUI wäre es jetzt zum Beispiel möglich, um das Hologramm des Gebäude herum zu gehen oder es per Sprachbefehl vergrößern oder verkleinern. Es könnte auch möglich sein, zum Beispiel Leitungen einzublenden.

Mit Gesten könnte es ebenso möglich werden, Fenster und Türen zu verschieben oder Objekte zu selektieren.

Aber selbst das Designen von Grund auf ist innerhalb einer AR-Anwendung kein Problem. Über ein Sprachkommando könnten zum Beispiel Wände ausgewählt werden, welche dann über Gesten im Raum gesetzt, verschoben oder gelöscht werden können.

Das System könnte ähnlich wie „Google Blocks“ mit welchem es unter Verwendung der „Oculus Rift“ oder der „HTC Vive“ möglich ist, 3D-Modelle zu erstellen, gestaltet werden. [Blo17]

#### 6.1.2 Bauen im Bestand

Eine weitere Anwendungsmöglichkeit für ein verwandtes System wäre das Bauen im Bestand, also wenn schon bestehende Häuser verändert und modifiziert werden sollen. Mit Hilfe von Sprachbefehlen könnte man bestehende Strom oder Rohrleitungen unter den Wänden anzeigen, um zu sehen, welche baulichen Veränderungen getroffen werden können und welche nicht.

So ist es zum Beispiel möglich, neue Leitungen zu planen ohne alte zu beschädigen oder suchen zu müssen.

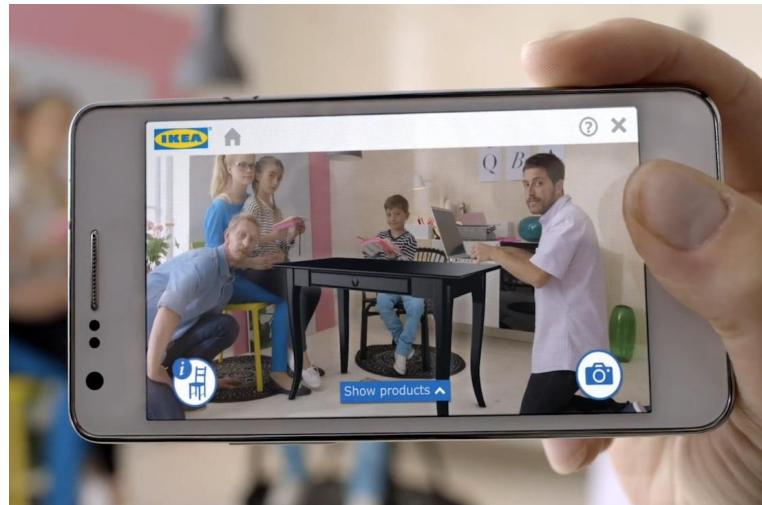


Abbildung 9: Ikea's AR Möbelstücke [Ike13]

Es wäre möglich, zum Beispiel Küchenzeilen in Räumen zu platzieren, um zu zeigen, wie diese ausfallen und ob sie zum Raum passen. Ein Vorreiter in dieser Technik ist die Möbelhauskette Ikea. Seit 2014 können Kunden von Ikea den Katalog mit der Ikea eigenen App scannen und so Möbelstücke auswählen, welche dann wiederum als AR-Objekt in der eigenen Wohnung angesehen werden können. In Abbildung 9 ist die Ikea-App zu sehen, in der Nutzer AR-Möbel in ihrer Wohnung platzieren. [Ike17]

### 6.1.3 Unterstützung von Rettungskräften

Feuerwehrleute rücken täglich aus, um unser aller Leben zu retten, sollten wir einmal in eine Gefahrensituation geraten. In stark verqualmten Räumen suchen Feuerwehrleute zum Beispiel nach Überlebenden. Hilfreich wäre hier zum Beispiel ein Gerät, welches unter schlechter Sicht, den Raum Scannt und Hindernisse, wie Tische, Schränke oder aber weitere Türen anzeigt.

Über Sprachbefehle wäre es zum Beispiel möglich, den Raum zu Scannen und zu Visualisieren. Eine andere Möglichkeit wäre, anhand von Wärmesignaturen nach Überlebenden zu suchen.

Ein solcher Einsatz liegt jedoch noch in weiter Ferne, da hierfür Geräte entwickelt werden welche Räume auch unter schlechter Sicht zuverlässig scannen und zum anderen auch Wärmebilder verarbeiten können.

Momentan sind Aktuelle AR-Geräte noch nicht in der Lage unter solchen Situation zu arbeiten, geschweige den dass sie den harten Einsatz überstehen würden. Denn Wasser, Staub, Hitze und harte Stöße sind noch immer der Feind von solchen Geräten.

### 6.1.4 Bestmögliches Beispiel

Es wurden drei mögliche Beispiele beschrieben, Anhand von denen ein Prototyp entwickelt werden kann, welcher genau aufzeigt, in wie weit ein NUI mit aktuellen AR und AI Technologien erstellt werden kann und welche Stärken und Schwächen es hat.

Der dringendste Bedarf für eine Applikation hat mit Sicherheit die Feuerwehrleute, welche unter Einsatz ihres Lebens arbeiten. Leider gibt es für eine solche Anwendung noch keine Hardware.

Nicht nur, dass aktuelle AR-Brillen heute nur bei guter Sicht arbeiten, sie verfügen auch nicht über die Möglichkeit, Wärmebilder zu erstellen, weshalb dieser Ansatz nicht weiter verfolgt wird.

Das Bauen im Bestand mit AR ist natürlich sehr nützlich. Jedoch ist das Rahmenwerk, welches als Vorarbeit geleistet werden muss, vom Umfang zu groß für diese Arbeit. Es müsste eine Raumerkennung entwickelt werden, welche zuverlässig Türen und Fenster erkennt. Außerdem müssen erst Beispieldaten für Leitungsnetze von Räumen angelegt werden.

Am besten eignet sich der Architektur Editor. Es ist heute relativ einfach möglich, ein grundlegendes Design Programm zu erstellen, welches Objekte erstellen oder verändern kann. Dieses System, kann dann um ein NUI erweitert werden.

## 6.2 Erstellung eines Architektur Editor als Prototyp

Natürlich ist es innerhalb der Arbeit nicht möglich, einen realistischen Architektur Editor zu erstellen, mit dem es möglich wird hochauflösende AR-Modelle von Gebäuden zu designen. Hierauf liegt auch nicht das Augenmerk der Arbeit.

Umgesetzt werden soll ein 3D-Designer, mit welchem es ähnlich wie im Spiel „Minecraft“ möglich ist, verschiedene AR-Blöcke in der realen Welt zu platzieren, um hiermit Gegenstände designen zu können.

Der entstehende prototypische Designer soll dann später nur über Gesten und Sprache gesteuert werden können. Hierfür werden folgende Gesten und Sprachbefehle benötigt.

Gesten:

- Block setzen
- Block entfernen
- Objekt drehen

Sprachbefehle:

- Block Material auswählen

Es ergeben sich für den reinen Design Einsatz also drei Gesten und ein Sprachbefehl. Mit den Gesten ist es möglich, Blöcke zu setzen, zu entfernen und das gesamte Objekt zu drehen. Mit einem Sprachbefehl kann das Material der Blöcke geändert werden.

Aber ein NUI macht nicht nur aus, dass es mit Gesten und Sprache gesteuert werden kann. Es muss sich auch auf den Nutzer einlassen und Fehlertolerant sein. Das heißt im Umkehrschluss, es muss auch möglich sein Gesten durch Sprache zu ersetzen und umgekehrt. Hieraus ergeben sich weitere Gesten und Sprachbefehle:

Gesten:

- Block Material auswählen

Sprachbefehle:

- Block setzen
- Block entfernen
- Objekt drehen

Es ergeben sich also aus vier Bedienmöglichkeiten acht mögliche Bedienszenarien für das reine Designen. Ob alle Möglichkeiten umgesetzt werden, wird sich in der späteren Bedienung herausstellen. Eventuell sind Gesten zum Auswählen des Materials nicht sinnvoll und praktikabel ebenso wie das Setzen und Entfernen von Blöcken über Sprache.

### 6.3 Beschreibung der Gesten und Sprachkommandos

Nachdem alle Steuerungstypen für den Prototyp festgelegt wurden, muss nun eine genaue Beschreibung erfolgen. Im ersten Schritt geht es rein um die grundlegende Erstellung von Gesten und Sprachkommandos, ohne technische Plattformen oder Limitierungen einzubeziehen. Dies geschieht anschließend im Kapitel ?????.

#### 6.3.1 Gesten

Gesten, welche innerhalb eines NUI implementiert werden, müssen selbsterklärend sein. Somit muss eine Geste gefunden werden, welche möglichst in der realen Welt genau das tut, was auch im Computer getan werden soll.

**Setzen und Entfernen** Um leichte Gegenstände zu platzieren, werden diese Gegenstände von Menschen zwischen Daumen und Zeigefinger - größere oder schwerere Gegenstände auch schon mal unter Zuhilfenahme des Mittelfingers - gegriffen und auf der gewünschten Stelle platziert. Siehe Abbildung 10.



Abbildung 10: Platzieren eines Gegenstandes mit der Hand [Che17]

Ohne die entsprechende Schachfigur ergeben sich dann Gesten wie diese. In Abbildung 11 ist die entsprechende Geste mit geschlossener Hand zu sehen, wobei in Abbildung 12 die Hand geöffnet ist. Die Bedeutung der Gesten ist jedoch die selbe.



Abbildung 11: Platzierende Geste Variante 1 [Pla16]



Abbildung 12: Platzierende Geste Variante 2 [Pla15]

Werden die Finger nun geöffnet, wird der Gegenstand abgesetzt und die Position nicht weiter verändert. Der Block ist gesetzt.

Umgekehrt funktioniert die Geste genau so. Werden die Finger innerhalb eines Blocks geschlossen, wird dieser selektiert und kann danach verschoben oder entfernt werden. Menschen entfernen Gegenstände, indem sie diese greifen und dann woanders ablegen. Der Prototyp kann einen Stein, der losgelassen wird und dann in der Luft ist, als gelöscht interpretieren.

**Objekte drehen** Um Objekte auf einer Fläche zu drehen, nutzen Menschen in der Regel den Zeigefinger, um das Objekt auf einer Seite anzustoßen. Dieser Impuls dreht dann das entsprechende Objekt. Für das NUI bedeutet das also, dass ein ausgestreckter Zeigefinger, der in der Luft rotiert, das Objekt rotieren soll. In Abbildung 13 ist diese Geste einmal schematisch dargestellt.

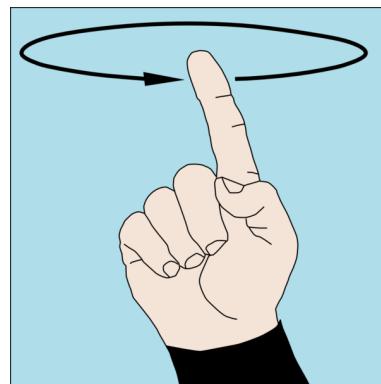


Abbildung 13: Rotieren Variante 1 [Rot13]

Das Objekt kann aber auch noch auf eine andere Art gedreht werden. Menschen können ein Gegenstand auch zwischen Daumen und Zeigefinger einklemmen und es dann durch eine Drehung der Hand drehen.

Im Umkehrschluss bedeutet dies, dass auch diese Geste bekannt sein muss. Das NUI muss entsprechend zwei drehende Finger auch als Kommando zum Drehen auffassen. In Abbildung 14 ist diese Geste dargestellt.

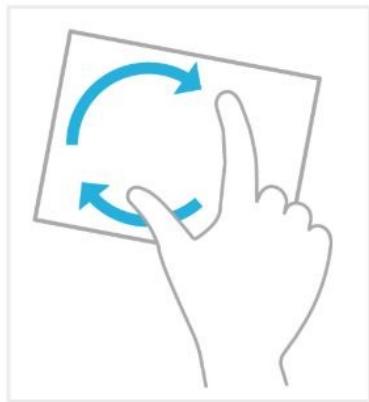


Abbildung 14: Rotieren Variante 2 [Mez]

### 6.3.2 Sprachkommandos

Sprachkommandos sind Sätze, welche von einem NUI als Eingabe verarbeitet werden und Aktionen auslösen. Grammatikalisch muss ein Satz mindestens aus zwei Bestandteilen bestehen, dem Subjekt, also das was verändert wird, und dem Prädikat, das angibt was getan werden soll. Somit müssen auch alle Sprachkommandos mindestens aus zwei Worten bestehen. Der Nutzer muss also angeben, was verändert werden soll und wie es verändert werden soll.

Somit ergeben sich für den konkreten Einsatzzweck folgende grundlegende Befehle.

- Block setzen
- Block entfernen
- Objekt drehen
- Material ändern

Die ersten drei Befehle sind eindeutig. Es ist klar, dass der selektierte Objekt gesetzt, entfernt oder gedreht werden soll. Ist nichts selektiert, kann der Befehl nicht ausgeführt werden. Andernfalls hat die Eingabe des Sprachkommandos eine unmittelbare Folge.

Das Ändern des Materials hingegen ist komplexer. Der Befehl hat eine Variable, nämlich die Art des Materials. Das Material muss im Programm bekannt sein, um es auswählen zu können. Somit ist der Satz „Ändere das Material zu Stein.“ ein eindeutiger Befehl. Das Interface muss jedoch so programmiert werden, dass der Auslöser „Ändere das Material“ ist. „Stein“ muss zusätzlich erkannt werden, denn diese Variable enthält die eigentliche Information.

Ist der Auslöser richtig erkannt worden, so muss das Interface das entsprechende Material wählen. Ist das Material dem Programm nicht bekannt, so muss es dies an den Nutzer kommunizieren.

Sprache ist jedoch nicht so eindeutig wie Gesten. Wörter können zum Beispiel mit Dialekt ausgesprochen werden. Wodurch die Erkennung um einiges schwieriger wird. Diese Zuordnung übernehmen jedoch Frameworks. Programmierer müssen sich heute nicht mehr darum kümmern dass ein Wort unterschiedlich ausgesprochen werden kann.

Alle heutigen Sprachassistenten Nutzen für diese Erkennung lernende Algorithmen, welche durch die Nutzung ständig verbessert werden.

Worum sich Programmierer aber immer noch kümmern müssen, sind synonyme Wörter, welche verwendet werden können. Von einem NUI wird erwartet, das es Synonyme erkennt und dennoch den richtigen Befehl ausführt.

Das Wort „setzen“ hat laut Duden folgende Synonyme:

- platzieren
- postieren
- hintun
- absitzen
- abhocken
- eingraben
- einpflanzen
- einsetzen
- stecken
- sedimentieren
- wetten
- tippen

Die Wörter „wetten“ und „tippen“ beziehen sich eindeutig auf das „setzen“ einer Wette und sind somit keine echten Alternativen für ein NUI. Ebenso verhält es sich mit eingraben, einpflanzen, einsetzen, stecken und sedimentieren. Diese Wörter beziehen sich eindeutig auf das „setzen“ einer Pflanze. „Absitzen“ und „abhocken“ stammen aus der schweizerischen Mundart, sind aber auch in Deutschland gebräuchlich. Sie beziehen sich auf das hinsetzen einer Person und sich somit auch nicht als Alternativen für ein NUI vergesehen. „Platzieren“, „postieren“ und „hintun“ hingegen sind eindeutige Alternativen für das angedachte NUI. Es ist durchaus aus im Bereich des möglichen, dass ein Nutzer „Block platzieren“ oder „Block postieren“ an Stelle von „Block setzen“ sagt. Das Interface muss also bei den Befehlen:

- Block setzen
- Block postieren
- Block platzieren
- Bock hintun

immer einen Block setzen.

Wenn man das Schema weiter verfolgt, so ergeben sich auf für den Befehl „Block entfernen“ Synonyme Befehle:

- Block entfernen
- Block löschen
- Block tilgen
- Bock wegtun

Für „Objekt drehen“ kommen noch einmal folgende Befehle hinzu:

- Objekt drehen

- Objekt rotieren

Aus den ehemals vier grundlegenden Befehlen (siehe Abschnitt 6.2) sind schon 10 geworden, nur damit sich das Interface „Natürlich“ bedienen lässt.

Sprache erlaubt es aber auch Wörter zu trennen, so kann zum Beispiel der Befehl „Block hintun“ auch „Tue den Block hin“ lauten, ohne das sich an der Bedeutung des Befehls etwas ändert. Um dies umzusetzen gibt es technisch gesehen verschiedene Möglichkeiten.

Eine ist die so genannte *Speech Recognition Grammar Specification* (SRGS) des *World Wide Web Consortium* (W3C), welche Sprachen grammatisch beschreiben. Hier können in *Extensible Markup Language* (XML) konformer weise, Wortschemen eingegeben werden, um einer Maschine zu verstehen zu geben, wie ein Mensch, einen Befehl aussprechen würde.

Hierbei werden Regeln aufgeführt, welche dann verschieden verkettet werden können, um Sätze zu bilden. Über ein solches XML können unter anderem auch verschiedene Sprachen und Dialekte abgebildet werden.

## 6.4 Überprüfung der Gesten und Sprachkommandos

Im folgenden werden die beschriebenen Gesten und Sprachkommandos anhand der ISO 9241 Teil 110 überprüft.

### 6.4.1 Gesten

Grundsätze	Setzen / Entfernen	Drehen
Aufgaben- angemessenheit	Das bewegen der Hand, sowie das interagieren auf der entsprechenden Stelle, wo sich der Block befindet, sind einfach auszuführen und der Aufgabe angemessen, da der Block um den geht virtuell berührt werden soll.	Die Bewegung der Finger ist eine einfache Geste, welche Objekte drehen sollen. Sie ist einfach gehalten und den Bewegungen von Smartphones nachempfunden.
Selbstbeschreibungs- fähigkeit	Die Geste ist selbst beschreibend, wenn der Block auf der Position der Finger projiziert wird. Durch das Drücken wird entsprechend der Block gesetzt oder entfernt.	Das Drehen der Finger ist nicht ganz Selbstbeschreibend, jedoch gleicht das adaptive NUI, welche die Geste interpretiert dies wieder aus, so dass entsprechend einer Drehbewegung erkannt werden kann.
Steuerbarkeit	Wird ein Block versehentlich gesetzt, so kann dies auch wieder rückgängig gemacht werden, da ein Block zu jeder Zeit wieder entfernt werden kann.	Die Drehung kann ebenso wie das Setzen und Entfernen rückgängig gemacht werden, indem eine Drehung in die entgegengesetzte Richtung gemacht wird.
Erwartungskonformität	Die Geste ist Erwartungskonform zur realität gehalten und somit Erwartungskonform in jeder Hinsicht, da keine andere Handlung ausgeführt wird.	Das Drehen ist ebenso Erwartungskonform, da beim drehen keine andere Handlung ausgeführt wird.

Fehlertoleranz	Durch die Wiederholung der Geste auf der selben Stelle kann ein versehentlich gelöschter oder gesetzter Block wieder gesetzt oder entfernt werden.	Das Drehen in die Gegenrichtung bewirkt, dass die Wirkung der ersten Drehung aufgehoben wird.
Individualisierbarkeit	Durch das NUI wird eine Ungenauigkeit der Nutzer abgefangen. Jedoch ist die Geste so eindeutig, dass sie nicht weiter Individualisierbar ist.	Für das Drehen gibt es zwei verschiedene Möglichkeiten wie diese Geste umgesetzt werden kann. Über das NUI werden auch hier Ungenauigkeiten von Seiten der Nutzer ausgeglichen.
Lernförderlichkeit	Da die Geste zum Setzen und entfernen aus der realen Welt stammen, sind sie eindeutig und ohne Probleme zu verstehen.	Das Drehen mittels einen oder zwei Finger ist grundsätzlich auch aus der Natur gegriffen. Da die Gesten auch auf Smartphones verwendet werden, sind auch diese schon bekannt.

#### 6.4.2 Sprachkommandos

Sprache ist zwar nicht immer eindeutig, jedoch wird sie dies im Kontext betrachtet. Befehle, welche also zu einem NUI gesprochen werden, werden entweder Verstanden und ausgeführt oder nicht. Weshalb eine Überprüfung anhand der ISO 9241 Teil 110 nicht sinnvoll ist.

Durch die Verwendung von synonymen Wörtern (siehe Abschnitt 6.3.2) wird das Verständnis von Sprachbefehlen durch ein NUI um einiges erweitert, was eine bessere Verwendbarkeit nach sich zieht.

## 7 Versuch der Entwicklung eines Natural User Interface anhand der Microsoft HoloLens

Im Kapitel 4 wurde untersucht, welches Gerät sich am besten zur Entwicklung eines NUI eignet, um AR Inhalte darzustellen und zu bearbeiten. Hierbei ergaben die Untersuchungen, dass die HoloLens das momentan beste Gerät auf dem Markt ist.



Abbildung 15: Darstellung von AR-Inhalten der HoloLens [Hol17b]

Um Sprachkommandos innerhalb dieses NUI umzusetzen, wurden verschiedene Sprachassistenten miteinander verglichen (siehe Kapitel 5). Am besten eignet sich hierfür Amazon Alexa oder der Google Assistant.

Das im Kapitel 6 beschriebene NUI wird nun auf der Microsoft HoloLens umgesetzt. Hierbei wird auf Einschränkungen und Möglichkeiten eingegangen.

### 7.1 Entwicklungsgrundlagen

Auf der HoloLens läuft ein angepasstes Windows 10, durch welches es möglich wird, jedes Programm auszuführen, welches auf der *Universal Windows Platform* (UWP) basiert. Im Windows Store gibt es zur Zeit schon einige Anwendungen speziell für die HoloLens.

Im Abschnitt 6.2 wurde ein Prototyp spezifiziert, welcher im Ansatz dem „Minecraft Editor“ entspricht. Dieser soll mit Hilfe der Gesten- und Sprach-Erkennung der Microsoft HoloLens um ein NUI erweitert werden.

Für die Entwicklung von Apps für die HoloLens wurde die Spieleengine Unity speziell erweitert und unterstützt in der Version 2017.1 die Entwicklung von Anwendungen speziell für die HoloLens.

Um Anwendungen zu Debuggen und auf die HoloLens zu übertragen, wird ein C# Projekt von Unity erstellt, welche auf die AR-Brille übertragen wird. [Hol17b]

#### 7.1.1 Grundlagen

Der Prototyp (im folgenden WorldBuilder) genannt, hat folgenden Grundaufbau.

Eine Unity-Szene bildet den physischen Raum virtuel in der nach Szene und zeigt diesen als Mesh an. Innerhalb dieses Raumes ist es dann möglich gelbe Blöcke zu setzen.

Das NUI soll nun die Steuerung dieser Szene wie in Abschnitt 6.3 übernehmen.

In Abbildung 16 ist die Unity Szene in Aktion zu sehen. Im Hintergrund befindet sich der physische Raum, welcher in die Szene integriert wird. Abgebildet wird dieser Raum innerhalb des WorldBuilder als Mesh, um zu zeigen, wie der Raum von der HoloLens erkannt wurde. Dank des erkannten Raumes, können die Blöcke direkt innerhalb des Raumes auf dem Boden oder anderen Oberflächen positioniert werden.

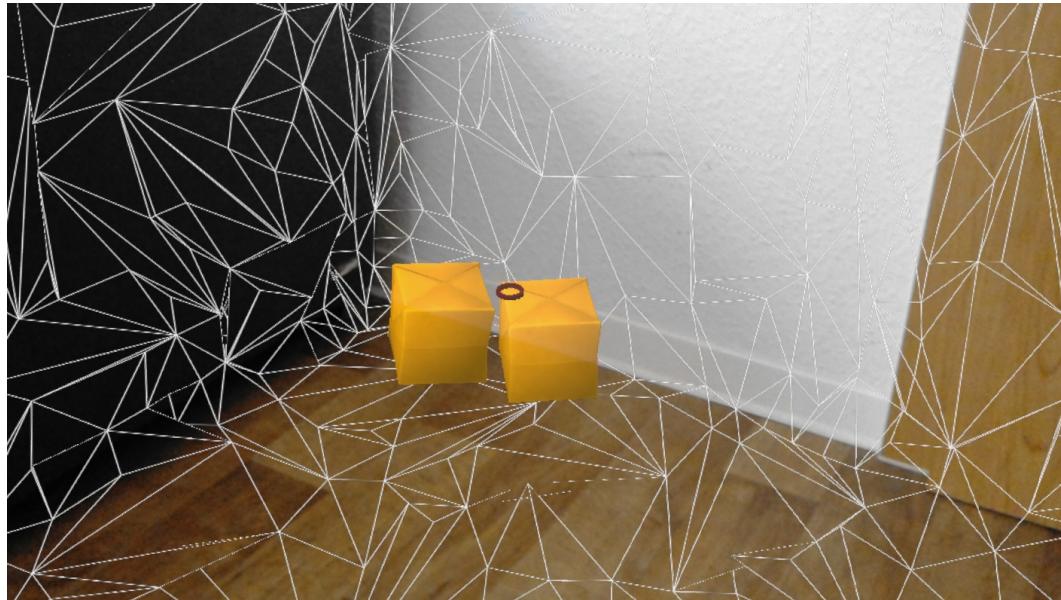


Abbildung 16: WorldBuilder mit Raum-Mesh und Blöcken

### 7.1.2 Gestenentwicklung für die HoloLens

Die HoloLens unterstützt ab Werk zwei Gesten. Zum einen die Select-Geste und zum anderen die Home-Geste. In Abbildung 17 wird Select in der „Ready“ Stellung gezeigt. Somit ist der Zeigefinger nach oben gerichtet und die anderen Finger bilden zusammen mit dem Daumen einen Tunnel.

Wird der Zeigefinger nun auf den Daumen bewegt, ist es als würde der Nutzer einen „Klick“ mit der Maustaste tätigen. Die Position der Hand ist dabei egal. Selektiert wird immer die Schaltfläche, auf der der Cursor ruht, welcher sich immer in der Mitte des Sichtfeldes befindet und der sich mit den Kopfbewegungen des Nutzers mitbewegt.



Abbildung 17: Die Select-Geste der HoloLens [Ges17b]

Die zweite Geste ist die Home-Geste, welche von der HoloLens selbst als „Bloom“ bezeichnet wird. In Abbildung 18 ist diese Geste dargestellt. Die Handfläche zeigt nach oben, wobei die Finger sich alle über der Handfläche berühren. Wird die Hand dann geöffnet, so erkennt die HoloLens die Bloom-Geste und öffnet das Home-Menü der HoloLens, beziehungsweise schließt es, wenn es bereits geöffnet ist.



Abbildung 18: Die Bloom-Geste der HoloLens [Ges17b]

Die Select-Geste hat zusätzlich noch zwei Erweiterungen. Wenn der Finger der Select-Geste auf den Daumen bewegt und dort gehalten wird, interpretiert die Brille dies als „Hold“. Hold ist gleichbedeutend mit einem langen Klick, über das zum Beispiel ein Kontextmenü angezeigt werden kann.

Wird die Hand während der Hold-Geste nach oben, unten, links oder rechts bewegt, kann dies als Scrollen oder Zoomen interpretiert werden, wie es in Abbildung 19 zu sehen ist.



Abbildung 19: Die Zoom-Geste oder Scroll-Geste der HoloLens [Ges17b]

Die HoloLens unterscheidet also drei Gesten, Select, Bloom und Hold. Um die Lernkurve gering zu halten und Nutzer nicht zu überfordern, wurden laut Microsoft die Gesten auf diese drei beschränkt. Das hinzufügen eigener Gesten ist also nicht möglich. [Ges17a]

**Neudefinition der Gesten** für die HoloLens. Die im Kapitel 6 beschriebenen Gesten können technisch nicht umgesetzt werden, wodurch die Gesten neu definiert werden müssen um den technischen Anforderungen und den eines NUI gerecht zu werden.

**Das Setzen eines Blocks** sollte, um den natürlich Fluss der HoloLens gerecht zu werden, über die Select-Geste umgesetzt werden. Hierbei sollte der Cursor immer einen Block anzeigen, damit der Nutzer sieht wie er abgesetzt aussieht.

**Das Löschen eines Blocks** wiederum kann entsprechend über eine Hold-Geste umgesetzt werden, welche beim auslösen den entsprechenden Block löscht.

**Drehen von Objekten** oder besser gesagt von einzelnen Blöcken ist nicht sinnvoll. Das Drehen von verbundenen Blöcken jedoch kann über die Zoom-Geste nach links oder rechts erreicht werden. Hierbei muss natürlich auf die Physik geachtet werden, denn Objekte können nicht durch reale Gegenstände bewegt werden.

### 7.1.3 Sprachkommandos für die HoloLens

Die HoloLens verfügt wie mittlerweile jedes Windows 10 getriebene Gerät über eine Cortana-Instanz, welche den Nutzer unterstützt. Der Funktionsumfang ist gleich mit einem standard Windows 10. Innerhalb eines Programms kann Cortana jedoch nicht verwendet werden.

Technisch ist die Verwendung von Amazon Alexa oder dem Google Assistant innerhalb der Unity-Skripte zwar möglich, jedoch nicht zwingend notwendig, da Unity über die UWP-API verfügt, besitzt es auch über die darin enthaltene SpeechRecognition-API. [Voi17]

Über die Speech-API können direkt, innerhalb der Unity-Skripte, Schlüsselwörter angegeben werden. Werden sie erkannt, können ihnen zugewiesene Routinen ausgeführt werden.

Dies funktioniert zusätzlich über eine SRGS-Grammatik, welche schon im Abschnitt 6.3.2 beschrieben wurde.

Es entfällt bei der Verwendung der HoloLens also ein Sprachassistent wie Alexa oder der Google Assistant und die damit verbundene Entwicklung von Skills.

## 7.2 MixedRealityToolkit

Das MixedRealityToolkit wird von Microsoft entwickelt und ist eine Sammlung von Skripten und Komponenten, welche Programmierer bei der Entwicklung von Programmen für die HoloLens und Windows Mixed Reality headsets unterstützen.

Viele Grundlegende Elemente, wie zum Beispiel Inputs, UI Elemente oder andere Helper sind hier schon fertig implementiert. Was wiederum Zeit beim entwickeln spart. [Mix17]

Der Quellcode ist ebenso wie viele Beispiele auf Github<sup>2</sup> zu finden.

## 7.3 Spatial Mapping

Mit Hilfe des „Spatial Mapping“ ist es möglich, den physischen Raum innerhalb der Unity Szene einzubauen. Hierbei gibt es zwei Möglichkeiten der Implementierung. Zum einen gibt es die Möglichkeit die fertige Skript-Implementierung des MixedRealityToolkit zu verwenden. Zum anderen können auch die standard Unity-Komponenten „Spatial Mapping Renderer“ und „Spartial Mapping Collider“ (siehe Abbildung 20) zu verwenden.

Im WordBuilder wurde das Spatial Mapping über Unity-Komponenten umgesetzt, da hier die Implementierung einfacher und übersichtlicher innerhalb der Szene umgesetzt werden kann wie in Abbildung 20 zu sehen ist.

Über den „Spatial Mapping Renderer“ ist es möglich, den physischen Raum, welcher von der HoloLens erkannt wird innerhalb der Szene zu rendern. Hierbei kann natürlich ein beliebiges „Material“ welches vorher definiert wurde eingesetzt werden.

Der „Spatial Mapping Collider“ bringt den erkannten Raum als Collider in die Szene. Über ihn ist somit möglich virtuelle Objekte physikalisch korrekt mit dem Raum interagieren zu lassen. Blöcke, welche in die Luft gesetzt werden, fallen somit auf den Boden des Raumes und bleiben liegen.

---

<sup>2</sup><https://github.com/Microsoft/MixedRealityToolkit-Unity>

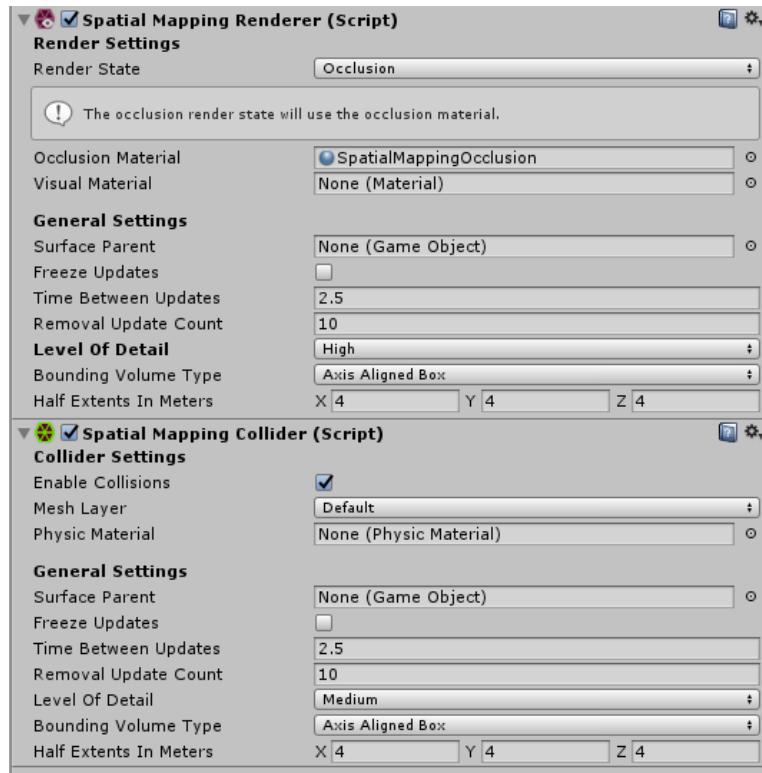


Abbildung 20: Spatial Mapping in Unity

Das Rendern des physischen Raumes innerhalb der Szene ist nicht vorgesehen. Jedoch wurde es zu Entwicklungszwecken implementiert. Über Sprachbefehle wie „Show Room“ oder „Hide Mesh“ kann der erkannte Raum ein oder ausgeblendet werden. Die Befehle sind konform des NUI implementiert und können auf unterschiedliche weise ausgesprochen werden. In Abbildung 16 ist das eingebündete Mesh zu sehen.

## 7.4 Das NUI Skript

Die Start Methode, ist in Unity-Skripten der Konstruktor, welche einmal zum Start der Szene aufgerufen wird.

Im ersten Teil des NUI-Skripts 1 (Zeilen 3-13) werden alle vom Skript benötigten Komponenten geladen. In Zeile 13 werden über einen Methodenaufruf alle „Keywords“ generiert, auf welche das NUI reagieren soll. Daraufhin, wird der „KeywordRecognizer“ mit den erstellten Keywords erstellt und gestartet (Zeilen 17-21).

Als nächster Schritt, wird der „GestureRecognizer“ erstellt und definiert, was bei einer erkannten Tap-Geste geschehen soll (Zeile 24-25). Wird ein Tap erkannt, wird ein Raycast von der Kameramitte ausgeführt. Trifft der Raycast ein Objekt so wird ein neuer Würfel gesetzt oder der getroffene Würfel entfernt, je nachdem ob der Nutzer gerade im Setzen- oder Entfernenmodus ist.

```

1 void Start()
2 {
3     SpeechController = this.GetComponent<SpeechController>();
4
5     SpeechController.Welcome();
6
7     UIController = this.GetComponent<UIController>();
8
9     WorldCursor = this.GetComponent<WorldCursor>();

```

```

10     CurrentCube = YellowCube;
11
12     CreateKeyWords();
13
14
15     // Tell the KeywordRecognizer about our keywords.
16     keywordRecognizer = new KeywordRecognizer(keywords.Keys.ToArray());
17
18     // Register a callback for the KeywordRecognizer and start recognizing!
19     keywordRecognizer.OnPhraseRecognized += KeywordRecognizer_OnPhraseRecognized
20         ;
21     keywordRecognizer.Start();
22
23
24     _recognizer = new GestureRecognizer();
25     _recognizer.TappedEvent += (source, tapCount, ray) =>
26     {
27         var headPosition = Camera.main.transform.position;
28         var gazeDirection = Camera.main.transform.forward;
29
30         RaycastHit hitInfo;
31         if (Physics.Raycast(headPosition, gazeDirection, out hitInfo))
32         {
33             if (Remove)
34             {
35                 if (hitInfo.transform.gameObject.tag == "cube")
36                 {
37                     Destroy(hitInfo.transform.gameObject);
38                     SpeechController.CubeRemoved();
39                 }
40             }
41             else
42             {
43                 Instantiate(CurrentCube, hitInfo.point, Quaternion.identity);
44                 SpeechController.CubeSet();
45             }
46         }
47     };
48
49     _recognizer.StartCapturingGestures();
50 }

```

Listing 1: Start Methode des NUI-Skripts

Im weiteren Verlauf des Skripts sind unzählige Keywords definiert und was passieren soll, wenn der KeywordRecognizer diesen Befehl erkennt. Im Listing 2 ist dies einmal beispielhaft dargestellt. Es wird dem „Dictionary“ „keywords“ ein Schlüssel-Wert-Paar hinzugefügt. Dies ist zum einen der Befehl als String und die Definition, welche ausgeführt wird wenn das Keyword erkannt wird. Im Beispiel wird die Methode „SelectYellowCube“ ausgeführt.

```

1 keywords.Add("SelectYellowCube", () =>
2 {
3     SelectYellowCube();
4 });

```

Listing 2: Keyword Erstellung

Wählt der Nutzer einen Würfeltyp aus, wird das entsprechende Prefab geladen und dem Cursor übergeben. Zusätzlich wird der UIController auf den Würfeltyp umgestellt und der SpeechController, welcher Sprachausgaben macht, informiert (Listing 3).

```

1 private void SelectYellowCube()
2 {
3     CurrentCube = YellowCube;
4
5     WorldCursor.SetCurrentCube(CurrentCube);

```

```

6     UIController.CubeColor(Color.yellow);
7     UIController.RemoveMode(false);
8
9     SpeechController.CubeColorChanged();
10 }

```

Listing 3: Auswählen eines Würfels

## 7.5 Der SpeechController

Die Hauptaufgabe des „SpeechController“ ist das heranführen des Nutzers an das Programm. Wird der WorldBuilder gestartet begrüßt der SpeechController den Nutzer und gibt eine kurze Einführung in die Bedienung.

Die Text zu Sprache Funktion ist Bestandteil des MixedRealityToolkit, wobei vier Stimmen zur Auswahl stehen. Im Listing 4 ist die erste Initialisierung zu sehen. Es wird in Zeile 4 eine Stimme ausgewählt. Danach kann ein Text übergeben werden, welcher durch Sprachsynthese eine Audioausgabe generiert.

```

1 public void Welcome()
2 {
3     textToSpeech = this.GetComponent<TextToSpeech>();
4     textToSpeech.Voice = TextToSpeechVoice.Zira;
5     textToSpeech.StartSpeaking("Welcome to the WorldBuilder. You can build what
6         ever you like out of cubes only with your voice and gestures.");
}

```

Listing 4: Testausgabe

„TextToSpeech“ kann zur Zeit nur in englisch genutzt werden. Zwar kann auch ein anderssprachiger Satz eingegeben werden, hierbei ist aber die Ausgabe schlecht, da das Framework auf englischer Sprachbetonung basiert.

Damit sich die Sprachausgabe nicht überlagert muss vor der Verarbeitung des zweiten Textes geprüft werden, ob die vorherige Ausgabe beendet ist. Dies geschieht über zwei Methoden, wie im Listing 5 beispielhaft zu sehen ist.

Die obere Methode ruft die untere als Co-Routine auf. Hierbei wird der Aufruf solange wiederholt bis die Bedingung in Zeile 8 erfüllt ist. Also bis kein Text mehr in der Schlange ist und die Sprachausgabe beendet wurde. Ist das dann der Fall, wird einmalig der Eingabetext in Zeile 11 Synthetisiert.

Innerhalb des SpeechController-Skripts bedinden sich viele solcher Methodenpaare, welche von anderen Skripten aufgerufen werden können.

```

1 public void CubeSet()
2 {
3     StartCoroutine(CubeSetSpeak());
4 }
5
6 private IEnumerator CubeSetSpeak()
7 {
8     yield return new WaitWhile(() => textToSpeech.SpeechTextInQueue() || 
9             textToSpeech.IsSpeaking());
10    if (firstBlockSet)
11    {
12        textToSpeech.StartSpeaking("Great! You can remove cubes by saying Remove
13            Cubes!");
14        firstBlockSet = false;
15    }
}

```

Listing 5: Testausgabe mit Überlagerungssperre

## 7.6 Der WorldCursor

Der WorldCursor zeigt die aktuelle Position des Cursors an. Hierfür macht der Cursor in der Update-Methode, welche einmal pro Frame durchlaufen wird, einen Raycast.

Der Raycast hat drei mögliche Funktionen, welche er abdecken muss. Trifft er kein Objekt, so ist der Raum nicht vollständig geladen und es kann kein Würfel platziert werden. In diesem Fall, wird die Nachricht „Loading Room! Please Wait ...“, mittig im Sichtfeld des Nutzers, angezeigt.

Trifft der Raycast einen Punkt, so wird ein Würfel an der entsprechenden Position angezeigt. Hierbei sieht der Würfel genau aus, wie der, welcher bei einer Tap-Geste platziert wird. Ist jedoch der Remove-Modus aktiviert, in dem Würfel entfernt werden können, wird an Stelle des Würfels ein kreisförmiger Cursor angezeigt, welcher bei einer Tap-Geste den anvisierten Würfel löscht.

Im Listing 6 wird in Zeile 8 der Raycast durchgeführt. Trifft dieser, wir abhängig vom aktuellen Zustand, entweder der Remove-Cursor oder der Würfel-Cursor angezeigt. In den Zeilen 12-19 wird der Remove-Cursor eingeblendet und entsprechend gedreht, so das dieser immer auf der Oberfläche des anvisierten Objektes ist.

Ist der Remove-Modus deaktiviert, so wird in den Zeilen 23-29 der Würfel-Cursor aktiviert und zum richtigen Punkt bewegt.

Trifft der Raycast nicht, so wird in den Zeilen 34-37 der Text der Nachricht, welche angibt das der Raum geladen wird, angezeigt.

```
1 void Update()
2 {
3     var headPosition = Camera.main.transform.position;
4     var gazeDirection = Camera.main.transform.forward;
5
6     RaycastHit hitInfo;
7
8     if (Physics.Raycast(headPosition, gazeDirection, out hitInfo))
9     {
10         if (Remove)
11         {
12             CurrentCubeMeshRenderer.enabled = false;
13
14             CursorMeshRenderer.enabled = true;
15
16             Cursor.transform.position = hitInfo.point;
17
18             Cursor.transform.rotation = Quaternion.FromToRotation(Vector3.up,
19                           hitInfo.normal);
20
21             SpeechController.RemoveCubes();
22         }
23         else
24         {
25             loadingText.text = "";
26
27             CurrentCubeMeshRenderer.enabled = true;
28             CursorMeshRenderer.enabled = false;
29             CurrentCube.transform.position = hitInfo.point;
30
31             SpeechController.CubeCursor();
32         }
33     }
34     else
35     {
36         loadingText.text = "Loading Room! \nPlease Wait ...";
37
38         CurrentCubeMeshRenderer.enabled = false;
39         CursorMeshRenderer.enabled = false;
39 }
```

Listing 6: Anzeigen des WorldCursor

## 7.7 Der UIController

Der UIController ist eine Unterstützung der Anzeige, in welchem Modus der Nutzer sich zur Zeit befindet.

Es wird im oberen linken Bildschirmbereich ein Quadrat in der aktuellen Farbe der Würfel angezeigt. Außerdem ist das Quadrat im Remove-Modus durchgestrichen.

Die Anzeige dient lediglich zur Unterstützung des Nutzers, hat aber keine funktionale Bedeutung. Der Code besteht aus Standard-Unity-Prefabs und der Quellcode des Skriptes ist trivial, weshalb hier auf ein Programmierbeispiel verzichtet wird.

## 7.8 Probleme bei der Entwicklung

In den folgenden Abschnitten werden festgestellte Probleme bei der Entwicklung und mögliche Lösungen beschrieben.

### 7.8.1 Ungenauigkeiten bei der der Meshaufzeichnung

Die HoloLens scannt während des gesammten Betriebs unablässig die Umgebung und passt das Mesh immer wieder an. Hierdurch verändert sich der Untergund der Blöcke, welcher durch das Mesh abgebildet wird, was dazu führt, dass die Blöcke innerhalb der Szene zittern und sich gegebenenfalls bewegen, wenn das Mesh aktualisiert wird.

Eine Lösung des Problems wäre zum Beispiel die Physik der Blöcke abzuschalten, damit diese nicht auf die Bewegung des Bodens reagieren.

### 7.8.2 Gestenlimitierung der HoloLens

Im Abschnitt 7.1.2 wurde beschrieben, dass die HoloLens nur die drei Gesten Select, Bloom und Hold beherrscht. Diese Limitierung ist nicht technischer Natur, sondern die Beschränkung wurde eingeführt, um Nutzer nicht zu überfordern.

Bei der Entwicklung eines NUI ist diese Beschränkung jedoch eher hinderlich. Zum Vergleich können Geräte wie zum Beispiel die „Leap Motion“ viel mehr Gesten erkennen und verarbeiten. Die Leap Motion verfügt über eine Unity Integration, durch welche es Grundsätzlich möglich wäre, Gesten zu erkennen und diese innerhalb der HoloLens Szene zu verarbeiten. [Lea17]

### 7.8.3 Sprachkommando Limitierung

Die HoloLens verfügt zum einen über Cortana, einen Assistenten, welcher sich aktuell nicht von Programmierern erweitern lässt. Aus diesem Grund kann er auch nicht in Programme integriert werden.

Wie in Abschnitt 7.1.3 schon beschrieben, ist es möglich, über die in Unity integrierte UWP Sprachkommandos zu implementieren. Hierbei gibt es aber auch eine Limitierung. Aktuell unterstützt Unity nur englischsprachige Kommandos.

Da es sich bei der HoloLens aktuell um ein Entwickler-Gerät handelt, welches nicht auf dem freien Markt gibt, ist diese Limitierung aber nicht dramatisch. Dass Interface muss lediglich mit

englischer Sprache bedient werden. Die Umsetzung anderer Sprache ist aber abzusehen, da auch der Assistent Cortana schon in der deutschen Sprache verfügbar ist.

## 7.9 Auswertung des Programm in Hinsicht auf das geplante NUI

Im Kapitel 3 wurde ein NUI grundlegend beschrieben. Hierbei wurde erklärt, welche Dinge zu beachtet sind und wie eine Umsetzung geplant werden kann.

Der Abschnitt 6.2 wurde hierauf aufbauend ein konkretes NUI am Beispiel eines prototypischen Architektur Editors beschrieben. Es wurde genaustens Definiert, wie Sprachbefehle und Gesten umgesetzt werden müssen um ein möglichst „Natürliches“ Interface zu erstellen, welches einfach zu erlernen und zu bedienen ist.

Im Abschnitt 7.1.2 wurde jedoch gezeigt, dass es nicht möglich ist zusätzliche Gesten für die HoloLens zu entwickeln. Woraufhin die Definition der Gesten angepasst werden musste.

Auch wenn die Entwicklung der Gesten nicht wie im geplanten Maße umgesetzt werden konnte, so konnten jedoch die erdachten Sprachkommandos umgesetzt werden. Da die HoloLens momentan noch nicht auf den freien Markt erhältlich ist, ist die aktuelle Spracherkennung lediglich auf die englische Sprache beschränkt. Um zu zeigen, dass die Entwicklung eines NUI möglich ist, ist diese Einschränkung jedoch nicht von Bedeutung.

Mit der reinen HoloLens ist es also nicht möglich, eine AR-Anwendung zu entwickeln die mit frei erdachten Gesten arbeitet.

## 8 Fazit

Im folgenden Kapitel wird ein Fazit darüber gezogen, wie gut es heute schon möglich ist ein NUI nur mit aktuellen AR und AI Technologien zu erstellen. Hierbei wird auf die Ergebnisse und Erkenntnisse aus der Arbeit eingegangen.

### 8.1 Augmented Reality

Ein NUI unter der Verwendung moderner AR und AI Technologien ist heute schon möglich. Es gibt verschiedene Typen von AR-Fähigen Geräten schon heute auf den Markt. Ihnen allen ist eins Gemein. Sie versuchen durch unterschiedliche Ansätze virtuelle Objekte in die reale Welt einfließen zu lassen.

Während Handheld-Geräte relativ trivial versuchen ein Kamerabild auf einen Bildschirm zu erweitern, so ist der Ansatz von Holo-Brillen wesentlich komplexer, in dem sie 3D-Hologramme im Raum platzieren.

Modernste Brillen wie die HoloLens können nicht nur Hologramme anzeigen, sie können sich auch selbst im Raum orientieren was sie wiederum befähigt ein Hologramm stabil im Raum darzustellen. Dies bedeutet, dass ein Nutzer um das Hologramm herum gehen kann, ohne das es seine Position relativ zum Raum ändert.

Für diese Art der Hologramm-Darstellung gibt es in der realen Welt praktisch schier endlose Einsatzmöglichkeiten, weshalb die weitere Entwicklung solcher Geräte auch in Zukunft interessant bleibt. Der nächste Große technische Schritt ist wohl mit der neuen HoloLens von Microsoft im Jahre 2019 zu erwarten.

Da Hologramme nicht angefasst werden können, ist es unerlässlich eine Bedienmöglichkeit in Form eines NUI zu finden. Hierbei können Hologramme allein über Sprache und Gesten gesteuert werden.

Wie in der Arbeit gezeigt (Abschnitt 7.1.2) ist es mit der HoloLens aktuell nur begrenzt möglich ein solches Interface zu erstellen. Dies liegt an der technischen Beschränkung der HoloLens, welche nur die beiden Grundgesten „Tap“ und „Bloom“ kennt.

Abhilfe könnte die nächste Generation bieten, welche Voraussichtlich eine bessere Gestenunterstützung bieten soll. Um aktuell mehr Gesten erkennen zu können, müsste ein zusätzliches Gerät wie zum Beispiel die „Leap Motion“ eingebunden werden. Denn sie kann frei Gesten von Nutzern erkennen. Diese ist aber im eigentlichen Sinne nicht portabel, was wiederum die Bewegungsfreiheit der Nutzer einschränkt.

Durch die grundlegenden Gesten und etwas Erfindergeist ist es aber schon heute möglich annähern natürliche Gesten umzusetzen.

### 8.2 Artificial Intelligence

Sprachassistenten gibt es auf den Markt schon seit einiger Zeit und der neuste Trend scheint zu sein, möglichst viele Geräte mit ihnen zu versehen um in naher Zukunft wirklich alles und jedes mit Sprache steuern zu können.

Die beiden Hauptkonkurrenten Amazon Alexa und Google Assistant sind sich nahezu ebenbürtig und unterscheiden sich nur in kleinen Dingen. So kann der Google Assistant einzelne Stimme unterscheiden oder Alexa viele Dinge besser und umgangssprachlicher aussprechen. Die Unterschiede sind damit aktuell so gering, dass es fast egal ist, welchen Assistenten ein Programmierer verwendet.

Microsoft mit Cortana und Apple mit Siri sind hier weit zurück und müssen dringend an Land gewinnen um nicht am Markt unterzugehen. So ist es mit beiden zur Zeit nicht möglich, eigene Anwendungen mit ihnen zu erweitern.

Für eine mehr oder minder triviale Umsetzung von Sprachbefehlen, wie sie im WorldBuilder zu Einsatz kommen ist auch ein triviales Framework zur Spracherkennung wie das innerhalb des MixedRealityToolkits vollkommen ausreichend.

Für einen einfachen Sprachbefehl wird also noch kein Assistent benötigt. Geht es somit nur um eine Reaktion die auf einen Befehl erfolgen soll, ist so ein Framework vollkommen ausreichend. Erst wenn die Befehle Kontextabhängig werden muss ein Assistent zur Verwendung kommen, denn nur er kann über ein Neuronales Netz und eine riesige Wissensdatenbank solche Arten von Befehlen sinnvoll verarbeiten und ausführen.

Die alleinige Spracherkennung und die Verarbeitung von Befehlen ist also schon heute gegeben und kann ohne Probleme Einzug in ein NUI finden. Lediglich der Anwendungsfall entscheidet ob ein einfaches Framework zur Spracherkennung ausreicht oder ob eine AI eingesetzt werden muss. Durch offene APIs ist aber auch dies heute kein Problem mehr. Es kann Quasi jede Firma ihr Produkt um Alexa oder den Google Assistant erweitern.

### 8.3 Allgemeine Machbarkeit

Die Erstellung eines NUI ist eine sehr komplexe Angelegenheit. Entwickler müssen sich nicht nur sorgen um die technische Umsetzung machen, sondern auch um das menschliche Verhalten.

Es muss immer versucht werden, Gesten und Sprache so natürlich wie möglich zu gestalten um ein flüssiges arbeiten zu ermöglichen. Hierbei muss vor allem auch die menschliche Mechanik beachtet werden. Wie würde eine Geste mit einem realen Gegenstand aussehen und wie kann diese auf ein Hologramm übertragen werden.

Dies ist aber noch nicht mal die wichtigste Frage. Denn ein anderer Gesichtspunkt sind Ermüdungserscheinungen, welche bei häufiger Wiederholung auftreten und so nicht nur zu Muskelschmerzen sondern auch zu Problemen mit den Sehnen oder dem allgemeinen Bewegungsapparat des Menschen führen können.

Die ständige „Tap“-Geste der HoloLens zum Beispiel empfinden viele Menschen auf auf Dauer sehr anstrengend und ein dauernder Einsatz über einen ganzen Arbeitstag ist wohl nicht zu empfehlen.

An diesem Punkt müssen Programmierer und Softwarearchitekten quasi zu Ärzten werden um eine Möglichkeit zu finden die Belastung dauerhaft zu reduzieren. Dies würde durch verschiedene Gesten welche die selbe Bedeutung haben gehen.

Grundlegend kann aber gesagt werden, dass es schon heute möglich ist ein NUI mit aktuellen AR und AI Technologien umzusetzen. Auch wenn es aktuell noch Einschränkungen gibt, so werden diese in absehbarer Zeit aufgehoben. Ein Zusammenspiel von Technologien ist ohne Probleme möglich, da sowohl AR als auch AI Technologien schon heute über Größe und Vielfältige APIs verfügen. Dies bezieht sich nicht nur auf die möglichen Programmiersprache, sondern auch auf den Umfang der möglichen Steuerung.

## 9 Zusammenfassung

## 10 Abkürzungsverzeichnis

**KI** *künstliche Intelligenz*

**VR** *Virtuelle Realität*

**AR** *Augmented Reality*

**NUI** *Natural User Interface*

**AV** *Augmented Virtuality*

**SDK** *Software Development Kit*

**AI** *Artificial Intelligence*

**AVS** *Amazon Voice Service*

**SRGS** *Speech Recognition Grammar Specification*

**W3C** *World Wide Web Consortium*

**XML** *Extensible Markup Language*

**UWP** *Universal Windows Plattform*

## Abbildungsverzeichnis

1	Reality-Virtuality Continuum [MTUK] . . . . .	10
2	Unterschiedliche Blickfelder von Betrachter und Kamera bei Handheld-AR [DBGJ13]	11
3	Schematische Darstellung des Magic Lens Effekts [DBGJ13] . . . . .	12
4	Schematische Darstellung des Aufbaus eines See-Through-Displays [DBGJ13] . .	13
5	Aufgesetzte HoloLens [Wik17f] . . . . .	13
6	Epson Moverio mit Controller [VRS16] . . . . .	14
7	Schematische Darstellung der Kommunikation von Sprachassistenten . . . . .	17
8	Darstellung eines Hauses in Architekt 3D [Arc17] . . . . .	20
9	Ikea's AR Möbelstücke [Ike13] . . . . .	21
10	Platzieren eines Gegenstandes mit der Hand [Che17] . . . . .	23
11	Platzierende Geste Variante 1 [Pla16] . . . . .	24
12	Platzierende Geste Variante 2 [Pla15] . . . . .	24
13	Rotieren Variante 1 [Rot13] . . . . .	24
14	Rotieren Variante 2 [Mez] . . . . .	25
15	Darstellung von AR-Inhalten der HoloLens [Hol17b] . . . . .	29
16	WorldBuilder mit Raum-Mesh und Blöcken . . . . .	30
17	Die Select-Geste der HoloLens [Ges17b] . . . . .	30
18	Die Bloom-Geste der HoloLens [Ges17b] . . . . .	31
19	Die Zoom-Geste oder Scroll-Geste der HoloLens [Ges17b] . . . . .	31
20	Spatial Mapping in Unity . . . . .	33

## **Tabellenverzeichnis**

1	Tabellarischer Vergleich der betrachteten Geräte . . . . .	14
2	Tabellarischer Vergleich der Assistenten . . . . .	19

## Literatur

- [Ale17] Speechcon Reference. Technical report, Amazon Inc., Oktober 2017.
- [Ama17a] Alexa Voice Service . Technical report, Amazon Inc., Oktober 2017.
- [Ama17b] Alexa Skills Kit. Technical report, Amazon Inc., Oktober 2017.
- [API17] Build natural and rich conversational experiences. Technical report, Google Inc., Oktober 2017.
- [Arc17] Architekt 3D 2017 X9 Platinum . <http://www.avanquest.com/Deutschland/software-online/architekt-3d-2017-x9-platinum--504078>, Oktober 2017.
- [ARV17] Microsoft HoloLens. Technical report, ARVRZone, Mai 2017.
- [Blo17] Blocks. <https://vr.google.com/blocks/>, Oktober 2017.
- [BLT<sup>+</sup>12] Domagoj Baricevic, Cha Lee, Matthew Turk, Tobias Höllerer, and Doug A. Bowman. A Hand-Held AR Magic Lens with User-Perspective Rendering. Technical report, University of California, 2012.
- [Che17] Chess in hand PNG image image with transparent background. <http://pngimg.com/download/8416>, Oktober 2017.
- [DBGJ13] Ralf Dörner, Wolfgang Broll, Paul Grimm, and Bernhard Jung. *Virtual und Agmented Reality*. Springer Vieweg, 2013.
- [Dig17] Digital Native. [https://de.wikipedia.org/wiki/Digital\\_Native](https://de.wikipedia.org/wiki/Digital_Native), Juli 2017.
- [DP15] Raimund Dachselt and Bernhard Preim. *Interaktive Systeme*. Springer-Verlag, 2015.
- [Ebe17] Christoph Ebert. Grundlagen der Spracherkennung – so funktionieren Alexa, Cortana, Siri & Co. <https://entwickler.de/online/mobile/grundlagen-spracherkennung-alexas-cortana-siri-579769393.html>, Februar 2017.
- [Ges17a] Gesture design. Technical report, Microsoft Inc., Oktober 2017.
- [Ges17b] Gestures. Technical report, Microsoft Inc., Oktober 2017.
- [Goo17] Google Tang. Technical report, Google, 2017.
- [Hol17a] Hololens. Technical report, Microsoft, Mai 2017.

[Hol17b] Microsoft HoloLens. <https://www.microsoft.com/de-de/hololens>, Oktober 2017.

[IBM17] Watson. [https://www.ibm.com/watson/?cm\\_mc\\_uid=91647690093115002014743&cm\\_mc\\_sid\\_50200000=1500201474](https://www.ibm.com/watson/?cm_mc_uid=91647690093115002014743&cm_mc_sid_50200000=1500201474), Jul 2017.

[Ike13] Place IKEA furniture in your home with augmented reality. <https://www.youtube.com/watch?v=vDNzTasuYEw>, Juni 2013.

[Ike17] IKEA erweitert Katalog mit Augmented Reality. <http://crossretail.de/ikea-erweitert-katalog-mit-augmented-reality/>, Oktober 2017.

[ISO11] Ergonomische Anforderungen für Bürotätigkeiten mit Bildschirmgeräten. Technical report, International Organization for Standardization, Juli 2011.

[Jöc14] Marja-Liisa Jöckel. Augmented Reality: Definition, Anwendung, Apps. *entwickler.de*, August 2014.

[Kle] Georg Klein. *Visual Tracking for Augmented Reality*. PhD thesis.

[Lea17] Leap Motion. <https://www.leapmotion.com/>, Oktober 2017.

[MBRS11] Anett Mehler-Bicher, Michael Reiß, and Lothar Steiger. *Augmented Reality Theorie und Praxis*. Oldenbourg Verlag, 2011.

[Mer16] Johannes Merkert. Google: Translate-KI übersetzt dank selbst erlernter Sprache. <https://www.heise.de/newsticker/meldung/Google-Translate-KI-uebersetzt-dank-selbst-erlernter-Sprache-3502351.html>, November 2016.

[Mez] Jorge Mezcua. Diving signs you need to know . <http://www.fordivers.com/en/blog/2013/09/12/señales-de-buceo-que-tienes-que-conocer/>.

[Mic17] Create intelligent, personalized experiences for users with the Cortana Skills Kit . Technical report, Microsoft Inc., Oktober 2017.

[Mix17] MixedRealityToolkit. Technical report, Microsoft Inc., Oktober 2017.

[MTUK] Paul Milgram, Haruo Takemura, Akira Utsumi, and Fumio Kishino. Augmented Reality: A class of displays on the reality-virtuality continuum.

[Pak17] Ingo Pakalski. Amazon lässt Alexa natürlicher klingen. *Golem*, Main 2017.

[Pla15] iStock. <http://www.istockphoto.com/de/foto/hand-geste-der-heben-etwas-gm467874314>

März 2015.

[Pla16] iStock. [http://www.istockphoto.com/de/fotos/pinchng-hand-gesture?](http://www.istockphoto.com/de/fotos/pinchng-hand-gesture?excludenudity=true&sort=mostpopular&mediatype=photography&phrase=pinchng%20hand%20gesture)  
excludenudity=true&sort=mostpopular&mediatype=photography&phrase=  
pinching%20hand%20gesture, Februar 2016.

[Rot13] Windows 8 Touch Gestures. <http://www.intowindows.com/windows-8-touch-gestures/>, Juni 2013.

[SGS17] Markus Schmidt, Anna Gerdt, and Patrick Skoruppa. Google Assistant: Neue Funktionen. *Computer Bild*, Mai 2017.

[Sir17a] Creating an Intents App Extension. Technical report, Apple Inc., September 2017.

[Sir17b] Creating an Intents App Extension. Technical report, Apple Inc., September 2017.

[Voi17] Voice input in Unity. Technical report, Microsoft Inc., Oktober 2017.

[VRS16] Epson Moverio BT-200. <https://www.virtual-reality-shop.co.uk/epson-moverio-bt-200/>, Mai 2016.

[Wei91] Mark Weiser. The Computer for the 21 st Century. *Scientific American*, page 12, September 1991.

[Wik17a] Head-Up-Display. <https://de.wikipedia.org/wiki/Head-up-Display>, Mai 2017.

[Wik17b] Künstliche Intelligenz. [https://de.wikipedia.org/wiki/K%C3%BCnstliche\\_Intelligenz](https://de.wikipedia.org/wiki/K%C3%BCnstliche_Intelligenz), Jul 2017.

[Wik17c] Amazon Alexa. [https://en.wikipedia.org/wiki/Amazon\\_Alexa](https://en.wikipedia.org/wiki/Amazon_Alexa), Oktober 2017.

[Wik17d] Cortana (Software). [https://de.wikipedia.org/wiki/Cortana\\_\(Software\)](https://de.wikipedia.org/wiki/Cortana_(Software)), September 2017.

[Wik17e] Google Assistant. [https://de.wikipedia.org/wiki/Google\\_Assistant](https://de.wikipedia.org/wiki/Google_Assistant), August 2017.

[Wik17f] Microsoft HoloLens. [https://de.wikipedia.org/wiki/Microsoft\\_HoloLens](https://de.wikipedia.org/wiki/Microsoft_HoloLens), Mai 2017.