UNIVERSITÄT KOBLENZ · LANDAU
Fachbereich 4: Informatik

BRICKMAKERS

Augmented Reality Anwendung für Windows Mixed Reality unter Verwendung der HoloLens zur Vermarktung von Werbeflächen

Bachelorarbeit

zur Erlangung des Grades Bachelor of Science (B.Sc.)
im Studiengang Computervisualistik

vorgelegt von

Sören Schröder

Erstgutachter:	Prof. Dr.-Ing. Stefan Müller (Institut für Computervisualistik, AG Computergraphik)
Zweitgutachter:	M.Sc. Thomas Kaspers (BRICKMAKERS GmbH, Entwicklung)

Koblenz, im September 2017

Erklärung

Ich versichere, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen, als die angegebenen Quellen und Hilfsmittel benutzt habe.

	Ja	Nein
Mit der Einstellung der Arbeit in die Bibliothek bin ich einverstanden.	☑	☐

Koblenz, 27.09.2017 .....

(Ort, Datum) (Unterschrift)

Institut für Computervisualistik
AG Computergraphik
Prof. Dr. Stefan Müller
Postfach 20 16 02
56 016 Koblenz
Tel.: 0261-287-2727
Fax: 0261-287-2735
E-Mail: stefanm@uni-koblenz.de

UNIVERSITÄT KOBLENZ · LANDAU

Fachbereich 4: Informatik

Aufgabenstellung für die Bachelorarbeit
Sören Schröder (Matr.-Nr. 212 200 024)

Thema: **Augmented Reality Anwendung für Windows Mixed Realtiy unter Verwendung der HoloLens zur Vermarktung von Werbeflächen.**

Augmented Reality Lösungen rücken vermehrt in den Fokus der Öffentlichkeit. Zurzeit geschieht dies noch größtenteils durch Anwendung auf Smartphone und Tablet. Allerdings existieren auch zahlreiche neue Entwicklungen im Bereich der Daten- bzw. AR-Brillen. Diese bieten einen wesentlich höheren Grad an Immersion. Gleichzeitig schränken sie den Nutzer dabei weniger in seinen Aktivitäten ein. Einer der dabei herausstechenden Entwicklungen ist die HoloLens von Microsoft. Die HoloLens ist eine AR-Brille, welche über mehrere Stunden völlig autark genutzt werden kann und so große Mobilität und unterschiedlichste Einsatzszenarien ermöglicht. Dabei bietet sie die Möglichkeit, ein vergleichsweise großes Sichtfeld mit virtuellen Informationen anzureichern. Anhand eines praktischen Anwendungsfalls sollen die aktuellen Möglichkeiten und Grenzen der HoloLens und der damit verbunden Windows Mixed Reality Plattform erarbeitet werden. Dabei stellt sich als Herausforderungen das Entwickeln einer Anwendung, unter Ausnutzung der Möglichkeiten der Windows Mixed Reality Plattform und den Hardwarekomponenten der HoloLens. Gleichzeitig aber auch der Umgang mit denen, aus der Wahl von Hard- und Software resultierenden Einschränkungen.

Ziel dieser Bachelorarbeit ist es, die Grenzen und Möglichkeiten der HoloLens an Hand eines praktischen Szenarios zu untersuchen. Die Anwendung soll mit Hilfe der HoloLens Informationen zu Werbeflächen als holographisches Element präsentieren. Hierbei soll es dem Nutzer ermöglicht werden, sich zu Werbeflächen, die sich in seinem Blickfeld befinden, Informationen einblenden zu lassen. Optional ist ein Anwendungsfall gedacht, in dem sich der Nutzer seine Werbung virtuell auf dem Werbeträger anzeigen lassen kann, um diese vor dem Druck in der realen Umgebung testen zu können.

Schwerpunkte dieser Arbeit sind:

1. Analyse und Recherche von ähnlichen Anwendungen
2. Einarbeitung in Windows Mixed Reality Entwicklung
3. Konzeption eines eigenen Ansatzes
4. Implementierung
5. Bewertung
6. Dokumentation der Ergebnisse

Koblenz, den 27.3.2017

- Sören Schröder -

- Prof. Dr. Stefan Müller-

Danksagungen

An dieser Stelle möchte ich mich bei einigen Personen bedanken, ohne die diese Bachelorarbeit nicht zustande gekommen wäre. Beginnen möchte ich mit allen Kollegen bei Brickmakers, die mich bei Fragen stets unterstützt haben. Insbesondere bei Timo Ziegler, der mir die Möglichkeit gegeben hat, die Arbeit bei Brickmakers zu schreiben und die notwendige Hard- und Software zur Entwicklung der Anwendung bereitgestellt hat. Des Weiteren bei Thomas Kaspers für das Übernehmen der Aufgabe des Zweitgutachters und das Korrigieren dieser Ausarbeitung. Daran anschließend geht der Dank an Stefanie Kugler, für die sprachlichen Anmerkungen. Außerdem bedanke ich mich bei der Firma awk AUSSENWERBUNG GmbH, für die Zurverfügungstellung der Daten zu den Werbeträgern.

Ein weiteres Dankeschön geht an Prof. Dr. Stefan Müller, für die Möglichkeit meine Bachelorarbeit am Institut für Computervisualistik schreiben zu können und für die Betreuung während dieser. Des Weiteren möchte ich mich bei allen Kommilitonen, die sich der *minus42* verbunden fühlen, für die gemeinsame Studienzeit bedanken. Es war mir eine Freude mit euch gemeinsam zu studieren und ich wünsche euch alles erdenklich Gute für eure weiteren Wege.

Außerdem möchte ich mich bei meiner Familie bedanken. Insbesondere bei meinen Eltern, welche mir durch ihre Unterstützung während der letzten Jahre, dieses Studium ermöglicht haben. Zu guter Letzt bedanke ich mich bei meiner Freundin für ihre Unterstützung und Hilfe, besonders in den letzten Wochen dieser Arbeit.

Zusammenfassung

Diese Bachelorarbeit befasst sich mit der Entwicklung einer Anwendung für die HoloLens von Microsoft. Die Anwendung dient der Vermarktung von Werbeflächen der Firma awk AUSSSENWERBUNG GmbH. Anhand der Entwicklung wird die Frage beantwortet, welches die Möglichkeiten und Grenzen der HoloLens in Verbindung mit der Mixed Reality Platform sind. Dabei wird auch auf die Probleme eingegangen, welche bei der Entwicklung einer Anwendung für eine neue Technologie, wie die HoloLens auftreten. Neben der neuen Technologie, ergeben sich auch durch den Einsatzort der Anwendung weitere Herausforderungen. Diverse Anwendungsbeispiele und Präsentationen lassen vermuten, dass die HoloLens primär für Anwendungen innerhalb von Räumen ausgelegt ist. Die zu entwickelnde Anwendung ist dagegen für die Verwendung außerhalb geschlossener Räume konzipiert. Bei der Entwicklung konnten Erkenntnisse über diese neue Technologie gewonnen werden. Zum einen wurde deutlich, dass sowohl die HoloLens, als auch die Entwicklungsumgebung noch an einigen Stellen verbessert werden können. Zum anderen, dass die HoloLens nicht für den Einsatz im Freien geeignet ist. Trotz der Schwierigkeiten bei der Entwicklung konnten auch die vielen Möglichkeiten der HoloLens beleuchtet werden.

Abstract

This bachelor thesis deals with the development of an application for the Microsoft HoloLens. The application is used for the marketing of advertising spaces that belongs to the company awk AUSSENWERBUNG GmbH. On basis of the development, the question is answered which are the possibilities and limitations of the HoloLens and the Mixed Reality Platform. Problems are also addressed, that come along with the development of an application for such a new technology, like the HoloLens is. Beside the new technologies, some challenges come also from the applications operational locations. Several application examples and presentations suggest, that the HoloLens is primarily designed for indoor usage. Instead the developed application is for outdoor use only. During the development, several insights can be gained about this new technology. On the one hand it becomes clear, that the HoloLens and also the development environment aren't completely matured yet. On the other hand, that the HoloLens isn't an outdoor device at all. Despite the difficulties during the development, there occur many possibilities that are associated with this new technology.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Glossar

Application Programming Interface Programmteil, der die Anbindung einer Software an eine weitere ermöglicht. vii, 13

Augmented Reality Das Erweitern der Realität durch virtuelle Elemente. vii, 1, 4

Augmented Virtuality Das erweitern einer virtuellen Szene um Elemente aus der Realität. vii, 4, 6

Bluetooth Low Energy Im vergleich zu vorherigem Bluetooth, stromsparende Technik zur Kommunikation über kurze Strecken (10m). vii, 19

Direct3D API von Microsoft zur Programmierung von 3D-Grafiken, Teil von DirectX. vii, 12, 13

DirectX Sammlung von API für den Zugriff auf Video- und Audiohardware, die häufig für Computerspiele verwendet wird. 14, 22, 44

Entity Framework OR-Mapper, für die Verwendung in .NET Programmen. vii, 15, 37

Extensible Application Markup Language Auf XML basierende Beschreibungssprache zur Definition der visuellen Darstellungen von UWP Anwendungen. viii, 13

Eye-Tracking Technologie, welche die Position der Pupillen und somit die Blickrichtung des Nutzers ermittelt. 11

Gaze Eine Methode der Eingabe mit der HoloLens, die einen Cursor durch Drehen des Kopfes, ähnlich wie mit einer Maus, im Raum bewegt. 11, 13, 18, 20, 23, 33, 34

Global Positioning System System, mit dem Geräte ihre Position, mittels Trilateration von Satellitensignalen, bestimmen können. iii, vii

Head-Mounted Display Ein Anzeigegerät, welches am Kopf des Benutzers befestigt wird und über ein augennahes Display oder direkte Projektion auf die Netzhaut, am Computer erzeugte Bilder darstellt. vii, 1, 7

Head-Up-Display An der Kamera verankerte Anzeigeform, die häufig in Computerspielen zum Einsatz kommt, um Informationen anzuzeigen. vii, 23

HoloBillboards Im Rahmen dieser Arbeit entwickelte Anwendung zur Vermarktung von Werbeflächen. iii, vii, 17, 20, 22–25, 27, 28, 33, 36, 41, 44–49, 51

Holographic Processing Unit Speziell für die HoloLens entwickelter Prozessor u. a. zur Verarbeitung von großen Mengen an Sensordaten. vii, 9

HoloLens Eine Brille von Microsoft, die das Anzeigen von virtuellen Objekten im realen Raum ermöglicht und dabei die physikalischen Gegebenheiten beachtet. i, iii, iv, 1–3, 7–16, 18–25, 27–31, 33, 36–51

Inertial Measurement Unit Bauteil, welches verschiedene Sensoren, wie Beschleunigungssensoren, Lagesensoren, Magnetometer, etc. vereint. vii, 8

Language-Integrated Query Vereinheitlichtes Modell zur Abfrage von Daten aus verschiedenen Typen von Datenquellen. vii, 16, 38

Mixed Reality Beschreibt den Bereich zwischen Realität und Virtualität im Reality-Virtuality Continuum. vii, 4, 7, 15

Mixed-Reality-Portal Portal, welches zukünftig zur Nutzung von AR- und MR-Brillen unter Windows gedacht ist. 7

OpenGL Eine API zu Entwicklung von 2D- und 3D-Anwendungen. 14

Reality-Virtuality Continuum Eine Definition von Milgram et. al [Mil+95] zur Klassifizierung von Technologien, bei denen Realität und Virtualität vermischt werden. iii, viii, 2, 4

Shell Software, welche das Bindeglied zwischen Benutzer und Computer darstellt. 12–14, 23, 33

Unity Plattformübergreifende Spiele-Engine von Unity Technologies. iii, 14, 22, 25–32, 35–37, 39, 40, 44, 46, 48

Universal Windows Platform Eine mit Windows 10 eingeführte Plattform um Anwendungen für verschiedene Familien von Geräten anhand von einheitlichen APIs zu entwickeln. viii, 10, 14, 16, 36

Virtual Reality Technologie, die den Nutzer in eine virtuelle Umgebung versetzt. viii, 1, 4, 5

Akronyme

API Application Programming Interface. 13–16, 21, 22, 26, 35–39, 42–44, *Glossar:* Application Programming Interface

AR Augmented Reality. 1, 4–7, 49, *Glossar:* Augmented Reality

AV Augmented Virtuality. 4, 6, 7, *Glossar:* Augmented Virtuality

BLE Bluetooth Low Energy. 19, 20, 36, *Glossar:* Bluetooth Low Energy

CPU Central Processing Unit. 8, 33

CRUD Create, Read, Update and Delete. 16

D3D Direct3D. 12–14, *Glossar:* Direct3D

DLL Dynamic Linked Library. 37, 40

EF Entity Framework. 15, 16, 37, 38, *Glossar:* Entity Framework

GPS Global Positioning System. iii, 5, 19–21, 41, 42, 45, 47, *Glossar:* Global Positioning System

HB HoloBillboards. *Glossar:* HoloBillboards

HMD Head-Mounted Display. 1, 6, 7, *Glossar:* Head-Mounted Display

HPU Holographic Processing Unit. 9, 10, *Glossar:* Holographic Processing Unit

HUD Head-Up-Display. 23, 24, *Glossar:* Head-Up-Display

IMU Inertial Measurement Unit. 8–10, 22, 42, 46, 50, *Glossar:* Inertial Measurement Unit

LCoS Liquid-Cristal-on-Silicon. 8

LINQ Language-Integrated Query. 16, 38, *Glossar:* Language-Integrated Query

MR Mixed Reality. 4, 7, 15, *Glossar:* Mixed Reality

OR-Mapper Object-Relational-Mapper. 15

REST Representational State Transfer. 15

RVC Reality-Virtuality Continuum. iii, 2, 4–7, *Glossar:* Reality-Virtuality Continuum

SDK Software Development Kit. 32

UI User Interface. 12–14

UWP Universal Windows Platform. 10, 13, 16, 25–27, 29, 30, 36, 37, 39, *Glossar:* Universal Windows Platform

VR Virtual Reality. 1, 4–8, 49, *Glossar:* Virtual Reality

XAML Extensible Application Markup Language. 13, *Glossar:* Extensible Application Markup Language

1 Einleitung

Die Verfügbarkeit von Geräten, welche die Nutzung von Augmented Reality (AR) und Virtual Reality (VR) ermöglichen, hat in den letzten Jahren stark zugenommen. Zwei bekannte Produkte im Bereich VR sind die Oculus Rift und das Google Cardboard, wobei beide sehr unterschiedliche Ansätze verfolgen. Bei der Oculus Rift, welche 2013 veröffentlicht wurde, handelt es sich um ein Head-Mounted Display (HMD) welches an den Computer angeschlossen wird und primär auf Desktop-Computerspiele ausgelegt ist. Das Google Cardboard (veröffentlicht 2014) geht hier einen anderen Weg. Bei diesem handelt es sich um eine Kombination aus einer App für Smartphones und einem kostengünstigem HMD, welches im Wesentlichen eine Halterung für das Smartphone ist. In diese wird das Smartphone eingespannt und über die Cardboard App wird die Nutzung von VR-Inhalten ermöglicht. Allgemein haben VR-Systeme gemein, dass der Nutzer visuell von der Außenwelt abgeschirmt wird.

Einen anderen Ansatz wählt Microsoft mit der 2016 vorgestellten HoloLens [Micc]. Die HoloLens schirmt den Nutzer nicht ab, sondern ermöglicht ihm weiterhin das Wahrnehmen der Umgebung mittels transparenter Displays. Im Gegensatz zu bspw. simplen AR-Anwendungen auf dem Smartphone, wie bspw. *Pokémon Go*, in welchen die virtuellen Inhalte über das Kamerabild geblendet werden, geht die HoloLens hier einen Schritt weiter. Sie hat mittels diverse Sensoren und Techniken die Möglichkeit, die von ihr angezeigten virtuellen Elemente, mit dem realen Raum interagieren zu lassen. Dadurch entsteht eine völlig neue Form des Vermischens von Realität und virtuellen Inhalten. Die HoloLens stellt somit in gewisser Weise das erste freiverkäufliche Gerät einer neuen Klasse von Computern dar. Im Zuge dieser Arbeit, wird die HoloLens anhand eines praktischen Beispiels, in Bezug auf ihre Möglichkeiten und Grenzen untersucht.

1.1 Motivation

In vielen Bereichen kann von den neuen Möglichkeiten einer Technologie wie der HoloLens profitiert werden. In der Arbeitswelt gibt es unzählige Beispiele, in denen durch eine Erweiterung der Realität profitiert werden kann. Denkbar sind hier bspw. Systeme, die Menschen bei ihrer täglichen Arbeit unterstützen und so eine Verbesserung der Sicherheit oder höhere Effizienz zu ermöglichen. Hierzu können durch die dreidimensionalen Hologramme Daten visualisiert werden, die es dem Nutzer ermöglichen, diese auf eine völlig neue Art und Weise wahrzunehmen.

Aber auch Unterhaltung in Form von Spielen ist ein Anwendungsfall für Geräte wie die HoloLens. Zum einen ermöglicht sie dem Spieler, sich frei in der Welt des Spieles zu bewegen. Gleichzeitig kann er dabei durch die Verschmelzung von Virtuellem und Realem weiterhin mit der realen

Welt interagieren. Durch eine Verbindung mehrerer Nutzer sind außerdem völlig neue Mehrspielererlebnisse möglich. Hierbei können die Mitspieler das Spiel aus jeweils unterschiedlichen Perspektiven wahrnehmen und dabei auf natürliche Weise, durch Sprache und Gesten, untereinander kommunizieren.

Dies sind nur zwei Beispiele, die zeigen, welches Potential diese Technologie besitzt. Ein Potential, welches auch in anderen Branchen genutzt werden kann. Die Firma awk AUSSENWERBUNG GmbH aus Koblenz ist einer der führenden Anbieter von Außenwerbeträgern. Das Kerngeschäft der awk ist die Vermietung von 18/1-Großflächen. Hier kann, durch das Anreichern der Realität mit den Metadaten der großformatigen Werbeflächen, ein interessanter Usecase für die HoloLens geschaffen werden. Anhand von diesem kann untersucht werden, in welchem Maße die HoloLens für die Vermarktung von Werbeträgern im Außenbereich verwendbar ist.

1.2 Aufgabenstellung

Im Rahmen dieser Bachelorarbeit wird die Entwicklung einer Anwendung für die HoloLens beschrieben. Die HoloLens ist ein Gerät der *Mixed Reality Platform* von Microsoft. Die Arbeit befasst sich dabei mit der Frage, welche Grenzen und Möglichkeiten die HoloLens bietet. Diese soll anhand der Erfahrungen, welche bei der Entwicklung der Anwendung gesammelt wurden, beantwortet werden.

Die zu entwickelnde Anwendung soll zur Vermarktung von Werbeträgern, im speziellen Großflächen, der awk AUSSENWERBUNG GmbH dienen. Werbetreibende und somit potentielle Kunden der awk sollen mit der Anwendung die Möglichkeit bekommen, sich Informationen zu den einzelnen Werbestandorten anzeigen zu lassen. Hierbei soll die Realität der Nutzer, bei Verwendung der Applikation, durch die Einblendung der Informationen erweitert werden.

1.3 Struktur der Arbeit

Die Unterteilung der Arbeit erfolgt in sechs Abschnitte, wobei mit dem ersten Abschnitt, mit der Motivation und der Aufgabenstellung, in die Arbeit eingeleitet wird. Die folgenden Abschnitte zwei bis fünf bilden den Hauptteil der Arbeit. Dieser beginnt mit den Grundlagen, welche im weiteren Verlauf der Arbeit von Belang sind. Hier wird neben den verwendeten Technologien auch auf das Konzept des Reality-Virtuality Continuum eingegangen. Die beiden darauffolgenden Abschnitte befassen sich mit der Konzeption und Implementierung der Anwendung. In der Konzeption werden die Anforderungen anhand des Anwendungsszenarios aufgestellt. Zu diesen werden in den weiteren Abschnitten Lösungsansätze diskutiert und die Struktur der Anwendung wird vorgestellt. Die Implementierung

der, in der Struktur vorgestellten Komponenten wird im vierten Abschnitt beschrieben. Der nächste Abschnitt befasst sich mit der Auswertung. Hierbei wird zum einen die Anwendung anhand der Anforderungen evaluiert, zum anderen wird die Forschungsfrage anhand der bei der Entwicklung gesammelten Erfahrungen beantwortet. Der abschließende Abschnitt liefert ein Fazit zu dieser Arbeit und bietet einen Ausblick auf eine mögliche Weiterentwicklung der Anwendung und auf die potentielle Zukunft der HoloLens in Kombination mit der Mixed Reality Platform.

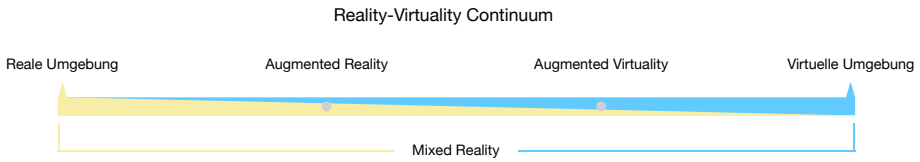


Abbildung 1: Reality-Virtuality Continuum nach Milgram et al. (1994)

2 Grundlagen

In den folgenden Abschnitten werden die Grundlagen beschrieben, welche für die darauffolgenden Teile benötigt werden. Hierbei werden die theoretischen Grundlagen zu häufig verwendeten Begriffe wie Augmented Reality, Virtual Reality, und Mixed Reality gelegt. Außerdem wird die in der Arbeit verwendete Hardware und Software beschrieben.

2.1 Reality-Virtuality Continuum

Das Reality-Virtuality Continuum (RVC) wurde von Milgram et. al [Mil+95] im Jahr 1994 zur Klassifizierung von Technologien, welche die Realität mit virtuellen Komponenten verschmelzen lassen, erdacht. Wie in Abbildung 1 zu sehen, kann das RVC als eine Gerade betrachtet werden. An deren einem Ende wird die reale Umgebung markiert und an dessen gegenüberliegendem die virtuelle Umgebung. Zwischen diesen beiden Extrempunkten können Konzepte wie Augmented Reality und Augmented Virtuality platziert werden. Die Position ist dabei davon abhängig, ob ein entsprechendes Gerät oder Programm die Realität mit virtuellen Elementen anreichert oder reale Elemente in eine virtuelle Umgebung eingebunden werden.

2.1.1 Augmented Reality

Als Augmented Reality (AR) versteht man im Allgemeinen das Erweitern der wahrgenommenen Realität durch virtuelle Objekte. Hier ist Realität als die vom Nutzer wahrgenommen Realität zu verstehen, da im engeren Sinne Realität als *alles was existiert* definiert werden kann. Daraus würde folgen, dass diese auch nicht darüber hinaus erweitert werden kann [HFN11].

Eines von einigen in den letzten Jahren erschienen AR-Geräte, ist die Google Glass. Bei der Glass handelt es sich um eine Brille, die es dem Nutzer ermöglicht, sich in der oberen, rechten Ecke seines Blickfeldes Informationen anzeigen zu lassen. Hierbei kann es sich bspw. um Navigationsinformationen beim Fahrradfahren handeln (siehe Abbildung 2). Diese können während der Fahrt abgelesen werden, ohne, dass der Nutzer seine Aufmerksamkeit von der Straße ablenken muss. Trotz eines ausbleibenden kommerziellen Erfolgs, hatte diese nach ihrer Vorstellung, vor allem aus

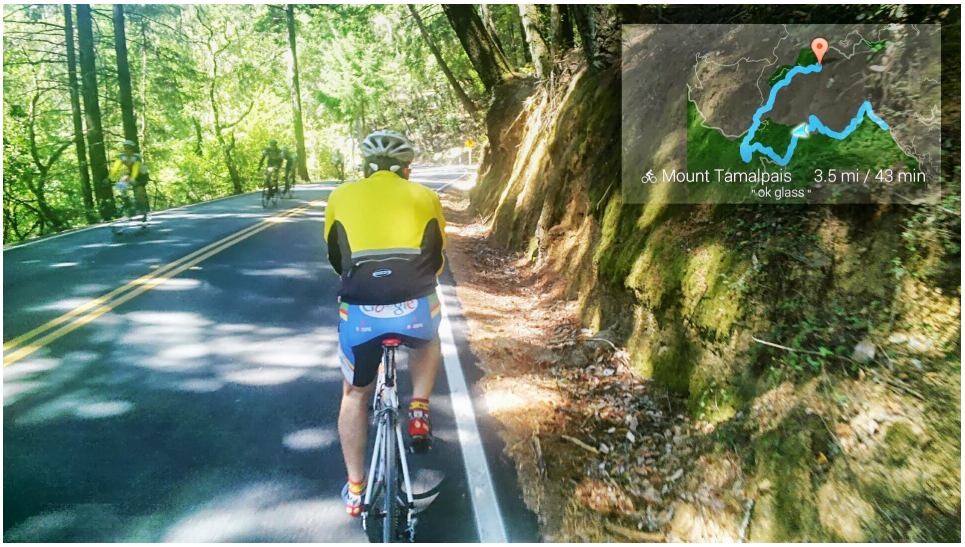


Abbildung 2: Google Glass Navigation [Goo].

Gründen des Datenschutzes, ein breites mediales Echo hervorgerufen. Die mitunter berechtigten Bedenken bezogen sich auf die nach vorne gerichtete Kamera, durch welche dem Nutzer das unauffällige Fotografieren möglich gewesen wäre.

Eine Anwendung, die AR in der breiten Massen bekannt gemacht hat, ist das Smartphone-Spiel *Pokémon Go*. In diesem geht es darum den Charakter über das Spielfeld zu bewegen, um die als *Pokémon* bezeichneten virtuellen Monster zu fangen. Das Spiel praktiziert den Ansatz von AR in zweierlei Hinsicht. Zum einen besteht das Spielfeld aus Kartendaten der realen Welt. Um die Spielfigur zu bewegen, muss der Nutzer seinen Standort in der Realität verändern, was mittels GPS in der Anwendung abgebildet wird. Zum anderen hat der Spieler die Möglichkeit, wenn er die Position eines der Monster erreicht hat, sich dieses auf seinem Smartphone zusammen mit der realen Umgebung anzeigen zu lassen. Die reale Welt wird also mit der virtuellen überblendet, wodurch das Erlebnis für den Nutzer einen höheren Grad an Immersion erreicht [Cop17].

Im RVC ist AR auf der linken Seite, neben der realen Umgebung anzusiedeln (siehe Abbildung 1). Das ist nachvollziehbar, da, wie in den Beispielen zuvor beschrieben, bei AR ein Großteil der Realität zu sehen ist und dieser mit zusätzlichen, virtuellen Inhalten angereichert wird.

2.1.2 Virtual Reality

Ein weiteres Konzept stellt Virtual Reality (VR) dar. Bei diesem wird nicht wie bei AR, die reale Umgebung mit Informationen angereichert, sondern der Nutzer wird in eine dreidimensionale Simulation versetzt. Hierbei wird

für den Nutzer durch Verwendung von Computergrafik, eine realistisch wirkende Welt synthetisiert. Diese virtuelle Realität reagiert dabei auf Eingaben durch den Nutzer [BC03]. Dabei kann es sich um Eingaben mit einem Gamepad, Tastatur und Maus oder andere Controller handeln.

In den neueren Generationen von VR-Headsets, wie der Oculus Rift [Ocu17] wird häufig auch ein Lagesensor verwendet um die Rotationen des Kopfes zu erfassen. Dies bietet dem Nutzer die Möglichkeit, eine Szene auf natürliche Art und Weise zu erkunden. Darüber hinaus wird auch die Position des Kopfes im Raum erfasst [Des+14]. Mittels dieser Technik kann sich der Nutzer, insofern der reale Raum dies zulässt, durch den virtuellen Raum bewegen und Elemente in diesem aus verschieden Perspektiven und Distanzen betrachten. Das Verarbeiten der Eingaben und die daraus resultierten Manipulationen in der Szene werden dabei von einem Computer in Echtzeit verarbeitet. Dies hat zur Folge, dass die Simulation vom Nutzer als fesselnd empfunden wird. Beide Faktoren führen dazu, dass der Nutzer eine erhöhte Immersion der Szene wahrnimmt [BC03].

Wie bereits im vorherigen Absatz erwähnt, existieren zurzeit einige VR-Lösungen in Form von HMDs. Die Oculus Rift hat in diesem Bereich eine gewisse Pionierrolle inne, da nach ihrem Erfolg auf Kickstarter [Kic12] viele weitere Unternehmen mit der Entwicklung von VR-Headsets begonnen haben.

Ein weiteres Konzept zum Erzeugen einer virtuellen Realität ist der CAVE, welcher erstmals im *Electronic Visualization Laboratory* der Universität von Illinois in Chicago entwickelt wurde. Dabei handelt es sich um einen begehbaren Würfel, bei dem auf jede Fläche ein Bild projiziert wird [CSD93]. Der Nutzer trägt dabei eine aktive 3D Brille und erlebt so eine begehbare virtuelle Realität [BC03].

Auf der Achse des RVC kann VR sehr weit rechts, kurz vor der *virtuellen Umgebung* platziert werden. VR ist aber keine reine virtuelle Umgebung, da bspw. die Eingaben des Nutzers, welche in der realen Welt vorgenommen werden, einen Einfluss auf die virtuelle Realität haben [Cop17]. Somit erfolgt auch hier eine Vermischung von Virtuellem mit Elementen aus der Realität.

2.1.3 Augmented Virtuality

Augmented Virtuality (AV) kann als Gegenstück zu AR gesehen werden. Wie der Name vermuten lässt, wird hier eine virtuelle Szene um Elemente aus der Realität erweitert. Auch wenn dieser Begriff im Alltag nicht so präsent ist, wie AR und VR, begegnet man AV doch relativ häufig. Ein Beispiel dafür ist der Wetterbericht im Fernsehen [Cop17]. Bei diesem besteht die Szene zum Großteil aus virtuellen, am Computer erstellten Wetterkarten. Diese Virtualität wird durch die reale Aufnahme des Meteorologen erweitert. Das Verhältnis zwischen Virtuellem und Realem ist hier für die Unter-

scheidung zwischen AR und AV entscheidend.

Ein weiteres Beispiel für AV findet sich im Mixed-Reality-Portal von Microsoft. Hierbei erlebt der Nutzer, genauso wie bei VR, mit Hilfe eines Headsets eine virtuelle Simulation. Das Mixed-Reality-Portal bietet die Möglichkeit, durch das Abgehen des Raumes die zur freien Bewegung zur Verfügung stehende Fläche zu definieren. Wenn sich der Nutzer anschließend bei aufgesetztem Headset durch den Raum bewegt, wird diese Bewegung auf die virtuelle Szene übertragen. Kommt er nun an den Rand des zuvor definierten Bereiches, wird ihm dies über eine Wand aus Punkten im virtuellen Raum angezeigt. Somit wird im Mixed-Reality-Portal die Simulation um die Information über die Begebenheiten des realen Raums erweitert.

2.1.4 Mixed Reality

Mixed Reality (MR) ist ein weiterer Begriff aus dem RVC. In diesem bezeichnet er den Bereich zwischen der realen Umgebung und der virtuellen Umgebung. Somit sind die zuvor beschriebenen Ansätze AR, VR und AV Bestandteile der Mixed Reality. Des Weiteren ist zu erkennen, dass der Übergang zwischen diesen fließend ist und durch ein ab- bzw. zunehmen der Realität bzw. Virtualität gekennzeichnet ist [Dör+16]. Wobei hier bei einer überwiegend, aber nicht ausschließlich, realen Umgebung von AR gesprochen wird. Ab dem Punkt an dem mehr Virtualität als Realität vorhanden ist, wird von AV gesprochen, bis zu dem Punkt, wo nahezu ausschließlich die virtuelle Realität vorhanden ist und somit von VR die Rede ist.

2.2 HoloLens

Die HoloLens von Microsoft ist der erste autarke Computer, welcher es dem Nutzer ermöglicht, sich virtuelle Objekte im realen Raum anzeigen zu lassen und mit ihnen zu interagieren [Micc]. Im Kontext der HoloLens taucht häufig der Begriff der *Hologramme* auf. Microsoft bezeichnet damit die im Raum platzierten virtuellen Objekte. Hierbei handelt es sich aber nicht im wissenschaftlich, physikalischen Sinne um Hologramme [Gut16], da die zugrundeliegende Technologie zur Erzeugung der virtuellen Elemente eine andere als bei der Holographie ist.

2.2.1 Hardware

Bei der HoloLens handelt es sich um einen Head-Mounted Display. Anders als bei VR-Geräten ist es aber möglich, durch die transparenten Displays der Brille hindurchzusehen und somit nicht von der Realität abgeschnitten zu sein. Ein Großteil der in der HoloLens verbauten Hardware

Central Processing Unit (CPU)Modell	Intel® Atom™ x5-Z8100P CPU .04 GHz
Unterstützter Befehlssatz	64-Bit x64 [Tay16b]
RAM	2GB
GPU Model (HPU) **midrule Dedizierter Videospeicher**	HoloLens Graphic 114 MB
Geteilter Systemspeicher	980 MB
Speicher Modell	Toshiba 064G32
Speicher Größe	64 GB
Betriebssystem	Windows 10
Kernel Architektur	32-Bit x86
Batteriekapazität	16500 mWh

Tabelle 1: Hardwarespezifikationen der HoloLens

ist ähnlich der in anderen tragbaren Computern, wie zum Beispiel Smartphones oder Tablets, was anhand der Spezifikationen in Tabelle 1 zu sehen ist. Die Daten wurden unter Verwendung der Software *AIDA64* [Fin] auf der HoloLens ausgelesen und durch weitere Informationen ergänzt. Auf die weiteren Bauteile der HoloLens, die maßgeblich an der Darstellung der virtuellen Objekte im realen Raum beteiligt sind, wird in den folgenden Absätzen eingegangen.

Optik Die *Optik* (siehe Abbildung 3) ist für die physikalische Darstellung der virtuellen Objekte zuständig. Sie besteht aus zwei transparenten Linsen, welche über eine Brücke verbunden sind. In dieser befindet sich für jede Linse jeweils eine *Light-Engine*. Dabei handelt es sich um Projektoren mit einem Liquid-Cristal-on-Silicon (LCoS) Display. Das von den Projektoren abgegebene Licht wird durch ein optisches Gitter gebrochen, sodass es innerhalb der Linse, welche als Lichtwellenleiter fungiert, zu einer Totalreflexion kommt. Dadurch wird das Licht zu einem Bereich geleitet, in welchem es wiederum so umgelenkt wird, dass es zum Austrittsbereich gelangt. In diesem wird der Winkel der Reflexion durch erneute Brechung verringert, sodass es keine weitere Totalreflexion gibt, und das Licht in Richtung des Auges austreten kann [Gut16]. Das Sichtfeld, welches von der Optik der HoloLens ermöglicht wird, ist im Vergleich zu vielen VR-Headsets, mit 30° mal 17,5° relativ gering [Doc15].

Inertiale Messeinheit Die Inertial Measurement Unit (IMU) befindet sich ebenfalls in der Brücke zwischen den beiden Linsen (siehe Abbildung 3), oberhalb der Projektoren. Die IMU beinhaltet drei Sensoren. Ein Beschleu-

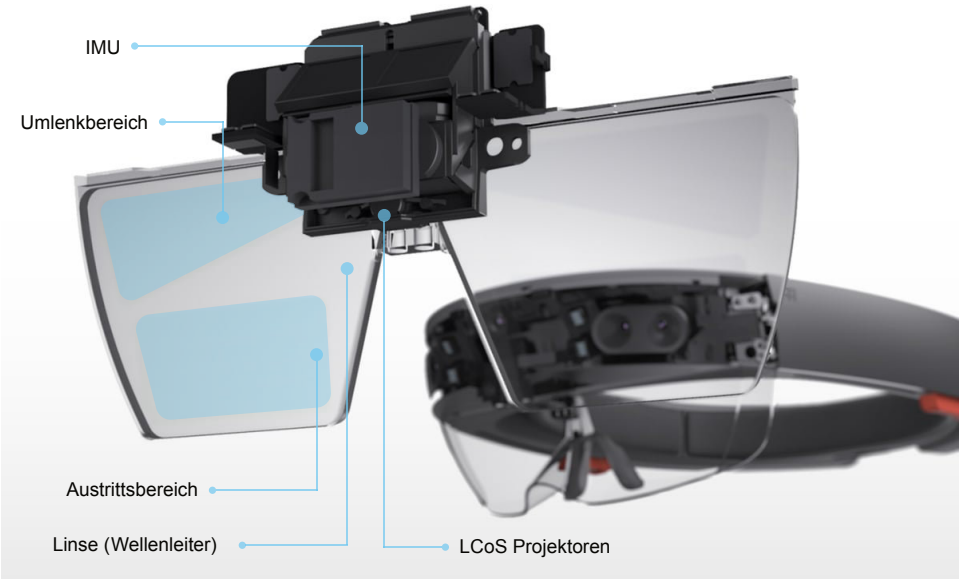


Abbildung 3: Optik der HoloLens [Micb] [Col16]

nigungsmesser, ein Gyroskop und ein Magnetometer. Die Daten der IMU werden, zusammen mit den Daten der Kameras, von der HoloLens dazu verwendet, die Position und Bewegung des Nutzers im Raum zu bestimmen [Tay16a].

Sensorleiste Wie in Abbildung 4 zu erkennen, besitzt die HoloLens eine Sensorleiste, bestehend aus sechs Kameras und einem Umgebungslichtsensor. Die jeweils zwei äußeren, seitlich ausgerichteten Kameras werden vom Hersteller als *environmental understanding cameras* bezeichnet und ermitteln die Position der HoloLens im Raum. In Verbindung mit der IMU kann somit die Pose (Position und Orientierung) des Systems bestimmt werden [Cue17].

In der Mitte der Sensorleiste, im oberen Segment, befindet sich die Tiefenkamera. Diese kann die Position von Objekten im Raum anhand des von ihnen reflektierten Infrarotlichts bestimmen. Das Infrarotlicht kommt dabei von ebenfalls in der Sensorleiste befindlichen Laser-Projektoren [Tay16b].

Des Weiteren sind unterhalb der Tiefenkamera ein Umgebungslichtsensor, für adaptive Helligkeitsanpassungen und eine 2 MP Foto-/HD Video-Kamera zum Aufnehmen von Bildern und Videos aus Benutzersicht, welche deshalb auch als Webcam bezeichnet wird, zu finden [Col16].

Holographic Processing Unit Bei der Holographic Processing Unit (HPU) handelt es sich um einen von Microsoft entwickelten Prozessor. Dieser dient dazu, die große Menge an Sensordaten, welche durch die Tiefenkamera,

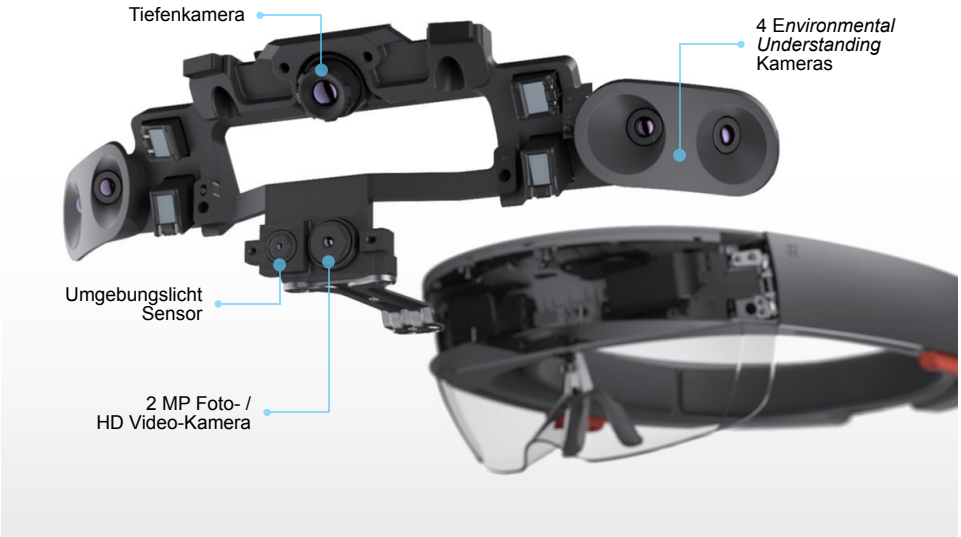


Abbildung 4: Sensorleiste der HoloLens [Mica] [Col16]

die Umgebungskameras und die IMU bereitgestellt werden, zu verarbeiten und an das SoC (System on a Chip) weiterzuleiten. Bei der HoloLens handelt es sich um ein Intel Cherry Tail SoC [Ang16]. Für diese Aufgabe befinden sich in der HPU 24 Digital-Signal-Prozessor-Kerne (DSP), welche speziell auf die Verarbeitung der großen Menge an Messdaten ausgelegt sind. Außerdem errechnet die HPU die virtuellen Objekte, die über die Optik in der realen Welt angezeigt werden [Kol16].

Lautsprecher Die beiden roten Elemente, an den Seiten der HoloLens (siehe Abbildung 3 und 4) sind die Lautsprecher. Beim Tragen der HoloLens befinden sich diese oberhalb der Ohren des Nutzers. Durch eine Phasenverschiebung, des vom Lautsprecher ausgegeben Tones, hört es sich für den Nutzer an, als wäre eine Schallquelle an einem bestimmten Ort im Raum. Diese Technologie wird auch als *Spatial Sound* bezeichnet. Somit hört es sich für den Nutzer an, als komme ein Geräusch, bspw. beim Erscheinen eines Objektes, von einer bestimmten Position im Raum [Tay16b]. Daraus ergibt sich eine erhöhte Immersion für den Nutzer.

2.2.2 Software

Betriebssystem Als Betriebssystem ist Windows 10 in der 32-Bit Version installiert. Zwei Punkte sind daran bemerkenswert. Zum einen bietet die HoloLens durch die, mit Windows 10 einhergehende Unterstützung der Universal Windows Platform (UWP) die Möglichkeit, sofern vom Entwickler vorgesehen, alle für diese entwickelten Apps zu verwenden [Tay16c].

(a) Ausgangsposition der Hand **(b)** Air-Tap ausführen

Abbildung 5: Air-Tap Geste

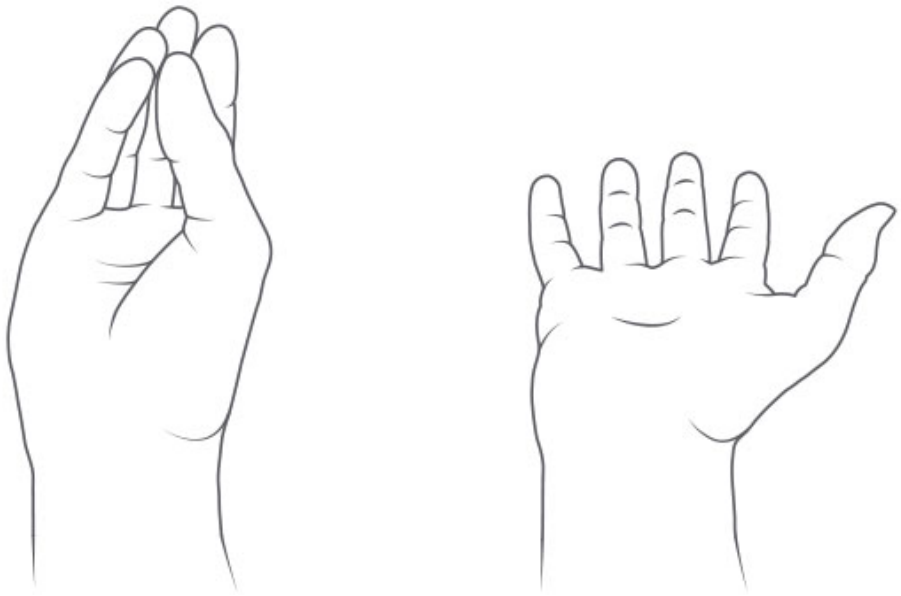
Hierbei handelt es sich um die Apps, welche über den Windows App Store installiert werden können.

Microsoft verwendet für die HoloLens die 32-Bit Ausführung ihres Betriebssystems, obwohl der Prozessor 64-Bit kompatibel ist. Hierbei wird sich die Eigenschaft von 32-Bit Systemen zu Nutze gemacht, dass die Instruktionen bei diesen nur die Hälfte an Speicher belegen, verglichen mit einem 64-Bit Systems. Da die HoloLens in Summe weniger als 4 GB Speicher adressieren muss und somit in diesem Punkt nicht auf einen 64-Bit Befehlssatz angewiesen ist, wird die 32-Bit Version von Windows 10 verwendet [Tay16b].

Bedienung Die Steuerung der HoloLens durch den Benutzer ergibt sich aus drei grundlegenden Komponenten. Dabei handelt es sich um die Blickrichtung, von Microsoft als Gaze bezeichnet, das Erkennen von Gesten und die Spracherkennung.

Die Komponente, welche der Nutzer als erstes, in der Anwendung zum Erlernen der Steuerung, beigebracht bekommt, ist der Gaze. Der Gaze kann dabei als die Blickrichtung des Nutzers beschrieben werden. Er ist ein Vektor, welcher seinen Ursprung an der Position des Nutzers bzw. der HoloLens hat. Seine Richtung entspricht dabei der Blickrichtung. Dabei muss beachtet werden, dass nicht die eigentliche Blickrichtung des Nutzers anhand von Eye-Tracking erfasst wird, sondern lediglich die Richtung, in die der Nutzer seinen Kopf dreht.

In Verbindung mit dem Gaze ermöglicht die Erkennung von verschiedenen Gesten dem Nutzer die Interaktion mit den virtuellen Objekten. Der Gaze wird hier verwendet um ein Element anzuvisieren, indem darauf geschaut wird. Zum Auswählen des anvisierten Elements wird daraufhin die *Air-Tap* Geste, welche das Äquivalent der HoloLens zu einem Mausklick



(a) Ausgangsposition der Hand **(b)** Bloom ausführen

Abbildung 6: Bloom Geste

am Computer ist, verwendet. Hierbei hält der Nutzer seine Hand wie auf Abbildung 5a zu sehen ist, was als *Ready State* bezeichnet wird. Zum Ausführen der Geste tippt er mit dem Zeigefinger nach unten auf den Daumen (siehe Abbildung 5b), um anschließend wieder in die Ausgangsposition zurück zu kehren. Diese Geste kann erweitert werden, indem der Zeigefinger nach dem Tippen nicht in die Ausgangsposition zurückgeführt wird. Die Hand kann so, in der *geschlossenen* Geste bewegt werden. Dies kann bspw. zum Navigieren in User Interfaces (UIs), dem Scrollen von Seiten oder zur Manipulation (rotieren, skalieren, bewegen) von Objekten im Raum verwendet werden. Alternativ zum Air-Tap kann auch der beigelegt Klicker der HoloLens verwendet werden.

Eine weitere Geste ist die *Bloom*-Geste. Hierbei hält der Nutzer, wie auf Abbildung 6a zu sehen, die Hand waagerecht, mit der Handfläche nach oben vor sich. Dabei berühren sich alle Finger, ausgestreckt in einem Punkt über der Handfläche. Von dieser Ausgangsposition wird die Geste durch das abspreizen aller Finger (siehe Abbildung 6b) ausgeführt. Wird die Bloom innerhalb der Shell ausgeführt, wird das Startmenü ein- bzw. ausgeblendet. Wird diese innerhalb einer Direct3D (D3D) -Anwendung ausgeführt, verlässt der Nutzer diese und kehrt in die Shell zurück. Die Geste ist mit dem Drücken der Windows Taste am Desktop vergleichbar.

Als Alternative zu manchen Gesten kann auch die Sprache zur Steuerung von Anwendungen verwendet werden. Dies kann entweder in Kom-

bination mit dem Gaze geschehen, wie zum Beispiel bei Bedienelementen wie Buttons im App Store, bei denen das Wort *Select* einen Air-Tap ersetzten kann. Alternativ gibt es auch Anwendungen welche globale Kommandos verarbeiten und so auf bestimmte Schlüsselwörter mit entsprechendem Verhalten reagieren können. Die Sprachsteuerung ist zum jetzige Zeitpunkt allerdings nur auf Englisch verfügbar. Dies wird vermutlich spätestens in einer Version der HoloLens für den Endkunden auch für andere Sprachen möglich sein.

Shell Als Shell wird ein Programm beschrieben, welches die Verbindung zwischen dem Nutzer und dem Computer herstellt. Dies können bspw. kommandobasierte Terminals oder grafische Benutzeroberflächen sein. Die Shell der HoloLens wird seit der Umbenennung von Windows Holographic in Windows Mixed Reality [Sur17], analog zum Namen der Plattform als *Mixed Reality Shell* bezeichnet.

Das Startmenü bildet das primäre visuelle Element zur Interaktion des Nutzers mit dem System. Es kann über die Bloom-Geste ein- und ausgeblendet werden. Aus dem Startmenü heraus können Anwendungen im realen Raum platziert werden. Für 2D-Anwendungen wird ein entsprechendes Fenster im Raum platziert, für 3D-Anwendungen eine entsprechende Verknüpfung in Form eines Fensters oder dreidimensionalen Objektes.

Außerdem gibt es noch eine weitere Art von Anwendung, welche es ermöglicht, 3D Objekte innerhalb der Shell zu platzieren. Diese werden wie Anwendungen im Raum platziert, erfüllen dabei allerdings einen rein dekorativen Zweck. Bisher ist *Holograms* von Microsoft die einzige Anwendung, die diese Möglichkeit bietet, da es zurzeit keine entsprechende Application Programming Interface (API) der Shell gibt, um dies Drittentwicklern zu ermöglichen.

Anwendungen Auf der HoloLens kann grundlegend zwischen drei Typen von Anwendungen unterschieden werden. Die ersten beiden Typen können dabei von Drittentwicklern implementiert werden.

Zum einen können dies 2D UWP Anwendungen sein, wie sie bspw. auf einem Windows 10 Desktop PC vorzufinden sind. Diese können entweder mit Extensible Application Markup Language (XAML) zur Beschreibung des UI und C#, VB, oder C++ für die Programmierung der Logik oder mit HTML und JavaScript entwickelt werden [Ken+17]. Diese Anwendungen werden als 2D-Fenster in der Shell der HoloLens dargestellt und können an der Stelle, an der die Anwendung vom Nutzer im Raum platziert wurde, ausgeführt werden, ohne dass dieser die Umgebung der Shell verlassen muss.

Der zweite Typ von Anwendung sind Direct3D (D3D) Anwendungen.

Diese können entweder mit der Unity Engine unter Verwendung von C# Skripten implementiert werden oder mittels der DirectX API unter Verwendung von C++. Beim Ausführen wird die Shell ausgeblendet und es wird ausschließlich die D3D Anwendung angezeigt.

2.3 Unity

Unity ist eine plattformübergreifende Spiele-Engine des Herstellers Unity Technologies. Diese ermöglicht das Entwickeln von Spielen und grafischen 2D/3D-Anwendungen auf einem höheren Level der Abstraktion als dies mit APIs wie DirectX oder OpenGL möglich ist. Die grafische Oberfläche der Engine, welche zum Erstellen der Anwendungen verwendet wird, ist der Unity Editor.

Eine Anwendung kann aus mehreren Szenen bestehen. Eine Szene besteht aus beliebig vielen GameObjects. GameObjects können dabei 3D-Modelle, Bestandteile des UI oder auch leere Objekte sein. Letztere werden häufig dazu verwendet, um mehrere Kind-Objekte, in der hierarchischen Übersicht der Szene, in einem, ggf. leeren, Eltern-Objekt zu gruppieren. Dabei ist zu beachten, dass ein Kind-Objekt immer in Relation zum Eltern-Objekt transformiert wird.

Einem GameObject können beliebig viele Komponenten zugewiesen werden. Diese Komponenten tragen dazu bei, die Eigenschaften der Objekte zu verändern. Hierbei kann es sich bspw. um Materialien, Texturen oder Kollisionsboxen handeln. Die Komponente zur Transformation existiert für jedes Objekt und kann nicht entfernt werden. Skripte stellen eine weitere Art von Komponenten dar. Sie werden den entsprechenden GameObjects, zur Programmierung der Anwendung, zugewiesen. Diese Skripte beinhalten in der Regel eine Klasse, welche die von Unity bereitgestellt `MonoBehavior` erweitert und das Verhalten des Objekts implementiert.

Unity bietet die Möglichkeit, komplexere Kombinationen von mehreren GameObjects und ihren Komponenten als Vorlage zu speichern. Diese Vorlagen werden als *Prefabs* bezeichnet und können entweder im Editor oder programmatisch in der Szene platziert werden.

Im Gegensatz zu vielen anderen Spiele-Engines gibt es bei Unity keine programmierbare Gameloop, welche alle Elemente eines Spiels steuert. Stattdessen können alle Klassen, welche `MonoBehavior` erweitern, die Methode `Update` implementieren, welche von der Engine bei jedem Frame aufgerufen wird.

Durch eine Zusammenarbeit mit Microsoft wurde in der Version 5.5 die explizite Unterstützung der HoloLens zu Unity hinzugefügt [Uni]. Zuvor war schon die Entwicklung von Anwendungen, wie sie auch auf der HoloLens laufen, möglich. Die explizite Unterstützung erfolgt aber erst in dieser Version.

2.4 MixedRealityToolkit-Unity

Das MixedRealityToolkit-Unity [Mic16b] (zuvor HoloToolkit-Unity) ist eine Sammlung von Skripten und Komponenten zur Entwicklung von Anwendungen für die HoloLens. Das Toolkit ist außerdem auf die Entwicklung von Anwendungen für Mixed Reality (MR)-Headsets ausgelegt. Als MR-Headsets bezeichnet Microsoft die nicht transparenten Headsets, welche die Windows Mixed Reality Platform nutzen. Letzteres findet in dieser Arbeit keine weitere Beachtung, da der Fokus auf Anwendungen für die HoloLens liegt. Die Änderung des Namens des Toolkits zeigt, dass Microsoft versucht eine einheitliche Softwarelösung für alle Geräte, die im Spektrum der Mixed Reality angesiedelt sind, zu etablieren.

2.5 ASP.NET Web API

ASP.NET Web API [Micd] ist ein Framework von Microsoft, welches zum Entwickeln von Representational State Transfer (REST) APIs verwendet werden kann. Bei REST handelt es sich um einen Architekturstil nach dem APIs entwickelt werden können. Dieser erfordert die Einhaltung einer Reihe von Prinzipien, welche sich am Verhalten des Internet orientieren [SK17].

Bei Web API handelt es sich um eines, von mehreren Modulen des ASP.NET-Frameworks. Das ASP.NET-Framework basiert wiederum auf dem .NET-Framework, welches ebenfalls von Microsoft entwickelt wird. Zur optimalen Nutzung von .NET stellt Microsoft seit 2002 die Sprache C# zur Verfügung. Die Sprache wurde 2003 von der ISO standardisiert und wird bis heute weiterentwickelt. Sie ist die von Microsoft empfohlene Sprache zur Entwicklung von .NET-Anwendungen. Darüber hinaus können alle Sprachen, die CLS (Common Language Specification) Standard einhalten, zur .NET-Entwicklung verwendet werden [Bal11].

2.6 Entity Framework

Beim Entity Framework (EF) handelt es sich um einen Object-Relational-Mapper (OR-Mapper). Dieser überträgt Daten, welche sich in einer objektorientierten Anwendung in Objekten befinden, in eine relationale Datenbank und umgekehrt. Es bildet somit die Schicht zwischen einer .NET-Anwendung und den persistenten Daten, bspw. in einer Datenbank [Ady+07].

Das EF ermöglicht es dem Entwickler mit den Daten auf einem höheren Abstraktionslevel zu arbeiten. Dies wird ermöglicht, indem EF die Anbindung an die Daten über ein objektorientiertes Interface zur Verfügung stellt [CMA07]. Das Framework liefert bei einem Zugriff auf die Daten so keinen Eintrag aus der Datenbank zurück, welcher noch auf ein Objekt übertragen werden muss, sondern ermöglicht das direkte Arbeiten mit einem Objekt. Umgekehrt muss beim Speichern nur das Objekt gespeichert

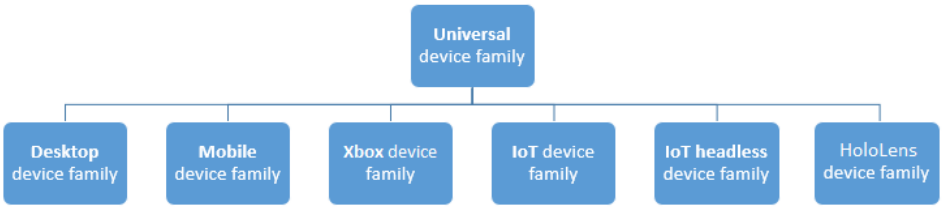


Abbildung 7: Gerätefamilien der Universal Windows Plattform [Whi+17]

werden, worauf das EF dieses in Datenbankeinträge übersetzt [Ler10]. Somit können alle Create, Read, Update and Delete (CRUD) Operationen direkt auf den Objekten der verwendeten Sprache ausgeführt werden.

Zum Erstellen der Datenbank bietet EF die Möglichkeit der Code-First-Migration. Hierbei erzeugt das EF das Datenbankschema anhand des Datenmodells, bzw. aktualisiert dieses in der weiteren Entwicklung, den Änderungen am Datenmodell entsprechend.

Darüber hinaus ermöglicht das EF die Nutzung von Language-Integrated Querys (LINQs). Hierbei handelt es sich um Ausdrücke in der verwendeten Sprache, welche beim Kompilieren in bspw. SQL Ausdrücke übersetzt werden um CRUD Operationen auszuführen [MBB06].

2.7 Universal Windows Platform API

Microsoft führte mit der Veröffentlichung von Windows 10 die Universal Windows Platform ein. Damit wurde eine gemeinsame Anwendungsplattform für alle Geräte, auf denen Windows 10 läuft, zur Verfügung gestellt. UWP arbeitet dabei mit dem Konzept der *Gerätefamilien*. Dabei beschreibt eine Gerätefamilie die APIs, welche von allen Geräten, die zu dieser Familie gehören, unterstützt werden. Jede Familie kann diese APIs an eine untergeordnete Gerätefamilie vererben.

Wie in Abbildung 7 zu sehen ist, stellt die HoloLens Gerätefamilie eine untergeordnete Gerätefamilie der universellen Gerätefamilie dar. Somit erbt die HoloLens Gerätefamilie die von der universellen Gerätefamilie spezifizierten APIs und erweitert dies zusätzlich um die, für die eigene Familie spezifischen APIs.

Der Entwickler hat somit die Möglichkeit eine UWP-Anwendung für eine möglichst große Anzahl an Geräten zu entwickeln, indem er nur die APIs der universellen Gerätefamilie nutzt. Alternativ kann seine Entwicklung auch nur auf eine spezielle Gerätefamilie, wie bspw. die HoloLens Gerätefamilie, abzielen. Dann kann er die geerbten und die eigenen APIs der Familie verwenden [Whi+17].

Abbildung 8: Großfläche (Plakatwand) der awk [awk17a]

3 Konzeption der Anwendung

In den folgenden Abschnitten wird die Konzeption der Anwendung erläutert. Dabei wird auch das Anwendungsszenario beschrieben, auf Grund dessen sich der Funktionsumfang der Anwendung ergibt. Für die, aus dem Funktionsumfang resultierenden Probleme und Herausforderungen, werden im Anschluss Lösungsansätze präsentiert, bewertet und eine begründete Auswahl getroffen. Im Weiteren wird die, sich daraus ergebende Struktur der Anwendung anhand eines Komponentendiagramm erörtert.

3.1 Anwendungsszenario

Die zu entwickelnde Anwendung, welche den Namen HoloBillboards trägt, ist zur Vermarktung von großflächigen Plakatwänden (siehe Abbildung 8) der Firma awk AUSSENWERBUNG GmbH [awk17b] konzipiert. Dabei ist es ihre primäre Aufgabe, dem Nutzer zusätzliche Informationen zu den Plakatflächen anzuzeigen, während er sich diese vor Ort anschaut. Die Zielgruppe für die Anwendung sind Werbetreibende und somit potentielle Kunden der awk, bei der Begutachtung möglicher Werbestandorte. Die Anwendung bietet den Nutzern dabei die Möglichkeit, sich zusätzliche Informationen zu den Standorten anzeigen zu lassen. Angezeigt werden sollen Daten, die für den Werbetreibenden potentiell von Interesse sind und ihn in seinen Entscheidungen bzgl. der Miete unterstützen. Dies können bspw. der Preis und andere Kosten, die Verfügbarkeit in einem bestimmen Zeitraum oder die durchschnittliche Anzahl der Personenkontakte pro Tag sein.

3.2 Anforderungen

An dem zuvor beschriebenen Anwendungsszenario werden die folgenden Anforderungen an die Anwendung abgeleitet:

- Die Anwendung muss die Plakatwand, zu der Informationen angefordert werden, identifizieren.
- Der Benutzer hat die Möglichkeit Informationen zu einer Plakatwand anzufordern.
- Der Nutzer muss wissen, für welche Plakatwand er Informationen angezeigt bekommt.
- Die Informationen müssen dem Benutzer angezeigt werden.
- Der Benutzer hat die Möglichkeit die Informationen auszublenden.
- Für den Benutzer muss jederzeit erkennbar sein, wo die Informationen angezeigt werden.
- Die angezeigten Informationen müssen nach Änderungen seitens des Anbieters beim nächsten Anzeigen aktualisiert sein.

3.3 Lösungsansätze

In den folgenden Absätzen werden Lösungsansätze für die zuvor aufgelisteten Anforderungen aufgezeigt und miteinander verglichen. Hierbei wird auch auf einige, die Anwendung betreffende Einschränkungen auf Seiten der HoloLens eingegangen und beschrieben, wie mit diesen umgegangen werden kann. Außerdem wird dargelegt und begründet, welche Ansätze für die Entwicklung der Anwendung ausgewählt wurden.

3.3.1 Nutzerinteraktion

Damit der Nutzer die Möglichkeit hat die Informationen zu einer Plakatwand anzufordern, muss er mit der Anwendung interagieren können. Dazu können die schon beschriebenen Eingabemethoden der HoloLens (Gaze, Gesten und Sprache) verwendet werden. Um dem Nutzer das Anvisieren von Objekten mittels Gaze zu ermöglichen, muss dieser visualisiert werden. Dazu kann ein Cursor verwendet werden. Dieser wird im Raum an der Stelle platziert, an der der Gaze auf ein virtuelles Objekt oder auf ein, von der Tiefenkamera erkanntes reales Objekt, trifft. Sollte der Gaze an einer Stelle auf kein Objekt treffen, da an dieser Stelle bspw. noch keine Erkennung des Raum stattgefunden hat, wird der Cursor in einer festen, zuvor definierten Entfernung angezeigt.

Dem Nutzer ist es somit möglich, durch einen Air-Tap auf einer Plakatwand zu signalisieren, dass er zu dieser Information angezeigt bekommen möchte. Eine Alternative zu Gesten bildet die Steuerung mittels Spracherkennung, welche zum Zeitpunkt der Arbeit (2017) ausschließlich auf Englisch möglich ist. Gerade, wenn Anwendungen komplexer werden, ist die Steuerung durch Sprache nötig, da die Anzahl der zur Verfügung stehenden Gesten begrenzt ist. Microsoft empfiehlt keine weiteren Gesten zu definieren, um dem Nutzer über alle Anwendungen hinweg ein einheitliches Nutzungserlebnis zu ermöglichen [Mic17b].

Da sich die Bedienung mittels Gesten und Sprache nicht gegenseitig ausschließen wurden für die Anwendung beide Methoden kombiniert. Der Nutzer kann so, nachdem er den Cursor auf eine Plakatwand bewegt hat, entweder eine Air-Tap Geste ausführen oder mit den Befehlen *open* oder *show details* die Informationen anfordern. Analog dazu erhält der Nutzer die Möglichkeit, über eine Schaltfläche, welche auch mittels Air-Tap bedient werden kann oder den Sprachbefehle *close* und *hide details*, diese wieder auszublenden. Die Unterstützung von beiden Ansätzen zur Bedienung ermöglicht es dem Nutzer frei zu wählen, welche Variante der Bedienung er bevorzugt.

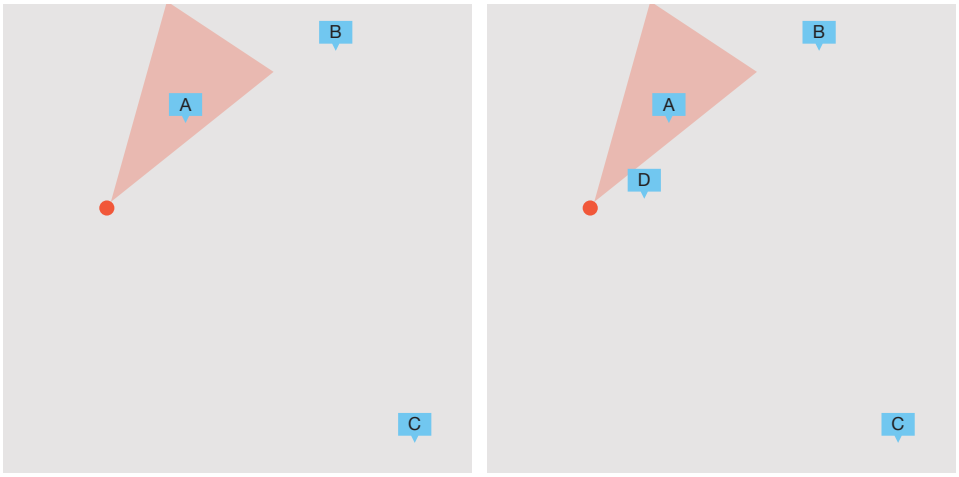
3.3.2 Plakatwand Identifizieren

Nachdem der Benutzer die Informationen zu einer Plakatwand angefordert hat, muss das Programm ermitteln, um welchen Werbeträger es sich handelt. Die Plakatwände verfügen über keinen eindeutigen Marker, in Form eines QR-Codes oder Ähnlichem, welcher bei einer Entfernung von mehreren Metern mit der eingebauten Kamera lesbar ist.

Ein weiterer Ansatz zur Bestimmung der Plakatwand ist es, den Standort der Plakatwand mit dem des Benutzers in Relation zu setzen. Dabei müssen jeweils die Standorte von Benutzer und Plakatwand bekannt sein. Die Standorte der Werbeträger ist durch die Daten der awk vorhanden. Da die HoloLens allerdings nicht über ein eingebautes GPS-Modul verfügt, muss die Position des Nutzers über andere Wege ermittelt werden.

Eine mögliche Lösung dafür ist, ein Smartphone zu verwenden, welches die Information über den Standort an die HoloLens übermittelt. Möglich ist dies mittels Bluetooth Low Energy (BLE) Advertising Paketen (vgl. [Osl17]). Ein Vorteil dieser Methode ist es, dass die Geräte nicht, wie bei konventionellen Bluetooth-Verbindungen, miteinander gekoppelt werden müssen. Das Smartphone sendet für jede Änderung seiner aktuellen Position ein Datenpaket, welches mit einem bestimmen *Company Identifier* versehen ist. Auf Seiten der HoloLens kann über die Bluetooth Schnittstelle auf den Empfang eines Pakets mit diesem Company Identifier gewartet werden.

Bei diesem Vorgehen ist es möglich, dass durch Dritte Pakete mit gleichem Company Identifier versendet werden und diese von der Anwen-



(a) Günstige Situation **(b)** Ungünstige Situation

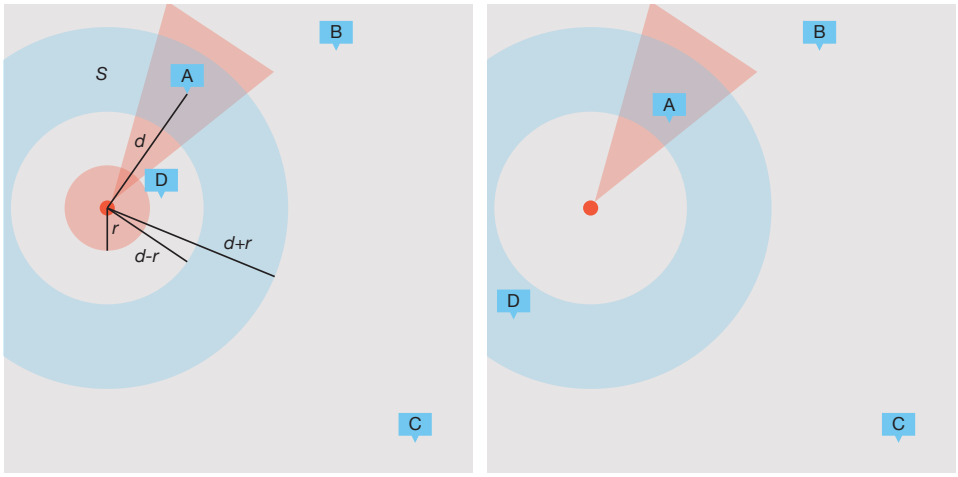
Abbildung 9: Identifizierung mittels GPS

dung auf der HoloLens angenommen werden. Da hier aber keine sensiblen Daten übertragen werden und das Stören der Anwendung ein eher uninteressantes Ziel darstellt, kann ein solcher Angriff als unwahrscheinlich betrachtet werden. Bei HoloBillboards wurde der zuletzt beschriebene Ansatz verwendet, da BLE wegen des geringen Energieverbrauchs eine vielversprechende Lösung für den mobilen Einsatz darstellt. Der einfache Austausch der Daten zwischen den Gräten ermöglicht dabei eine, im Gegensatz zu anderen Methoden der Datenübertragung zwischen den Geräten, vergleichsweise komfortable Nutzung.

Die einfachste Möglichkeit mittels der GPS-Daten eine Plakatwand zu identifizieren ist es, die Plakatwand mit dem geringsten Abstand zum Nutzer auszuwählen. In Abbildung 9a ist eine exemplarische Situation dargestellt. Die drei Rechtecke mit den Buchstaben A, B und C stellen Plakatwände dar, der rote Punkt steht für den Nutzer. Das rote Dreieck symbolisiert sein Blickfeld in Richtung Plakatwand A. In einer Konstellation wie dieser, in welcher die anvisierte Plakatwand auch den geringsten Abstand zum Nutzer hat, kann dieses mittels GPS identifiziert werden.

Denkbar ist auch eine Situation wie in Abbildung 9b zu sehen ist. Es ist erkennbar, dass der Nutzer, mit dem genannten Ansatz, hier Daten zu Plakatwand D bekommen würde, da ihm diese am nächsten ist. Der Gaze liegt dagegen aber auf Plakatwand A, wodurch der Nutzer Informationen zu dieser erwarten würde. Ein weiterer Faktor, der zu falschen Informationen führen kann, ist die Ungenauigkeit des GPS. Da dieses in der Regel nur auf wenige Meter genau ist, kann dies gerade bei einer hohen Dichte an Plakatwänden dazu führen, dass eine Werbefläche falsch identifiziert wird.

Gerade in Innenstädten gibt es viele Orte, an denen mehrere Plakat-



(a) Günstige Situation **(b)** Ungünstige Situation

Abbildung 10: Identifizierung mittels GPS und Abstand

wände auf engem Raum stehen, was zu einer Häufung von nicht korrekten Identifizierungen führen kann. Plakatwand D kann im oben aufgeführten Beispiel ausgeschlossen werden, wenn bekannt ist, in welchem Abstand sich der Nutzer zur zu identifizierenden Plakatwand befindet. Ist diese Information vorhanden, kann ein Suchbereich definiert werden, in dem sich die Plakatwand befindet. Ein entsprechender Suchradius ist als blauer Ring in Abbildung 10a zu sehen. Dieser ergibt sich aus der Entfernung vom Nutzer zur Plakatwand und der Ungenauigkeit des GPS. Letztere wird in Form eines Radius angegeben, bei dem eine Wahrscheinlichkeit von 68% besteht, dass sich das Gerät innerhalb des Kreises um die angegebene Position befindet [Goo17]. Die Entfernung ist in Abbildung 10a als Distanz d eingezeichnet, die Genauigkeit als Radius r. Der aus den beiden Werten resultierende Suchbereich S ist der Bereich zwischen den zwei Kreisen um den Standort. Der innere Kreis entspricht dabei dem Radius $d - r$, der äußere Kreis dem Radius $d + r$. Um Ungenauigkeiten bei den angegebenen Standorten der Plakatwände und Messfehlern des GPS entgegenzuwirken, muss der Suchbereich um einen bestimmten Faktor vergrößert werden. Wie in der Abbildung 10a zu sehen ist, wird Plakatwand D durch dieses Verfahren ausgeschlossen und A korrekt identifiziert.

Zur Implementierung dieses Ansatzes ist es Notwendig den Abstand vom Nutzer zur Plakatwand zu kenn. Wie in den Grundlagen beschrieben, verfügt die HoloLens über eine Tiefenkamera. Auf deren Tiefenwerte kann allerdings nicht direkt zugegriffen werden. Sie sind aber indirekt über die Spatial Mapping API zugänglich. Als Spatial Mapping bezeichnet Microsoft die virtuelle Repräsentation des realen Raums in der Umgebung der HoloLens [Mic17h]. Diese Repräsentation wird unter anderem mittels der

Tiefenkamera generiert. Auch an diesem Punkt wird wieder deutlich, dass die HoloLens für Innen konzipiert ist, da das Spatial Mapping auf einen Bereich zwischen 0,85m und 3,1m begrenzt ist. Da somit nicht garantiert werden kann, dass eine Repräsentation der Plakatwand durch das Spatial Mapping existiert, kann dieses nicht verwendet werden.

Ein weiterer Sensor, auf welchen über die API zugegriffen werden kann, ist die Fotokamera. Anhand eines Bildes aus Nutzerperspektive kann eine Detektion von Werbeflächen erfolgen. Hierzu kann, wie von [Kri12] beschrieben, eine Kombination mehrerer Bildverarbeitungsverfahren verwendet werden um die Umrisse einer Plakatwand zu bestimmen. Anhand des ermittelten Parallelogramms, der Information über die Größe der Plakatwände und die Eigenschaften der Kamera, kann die Entfernung zu dieser berechnet werden. Dieser Ansatz konnte u. a. aus zeitlichen Gründen nicht im Rahmen dieser Bachelorarbeit implementiert werden. Es wird aber im Weiteren die grundlegende Machbarkeit von Bildverarbeitung auf der HoloLens überprüft.

Wie Abbildung 10b zu erkennen ist, kann auch die Einschränkung auf einen ringförmigen Suchbereich immer noch zu uneindeutigen Ergebnissen führen. In diesem Beispiel würde sowohl A als auch D in Frage kommen. Die Identifizierung kann, durch die Einbeziehung der Blickrichtung des Nutzers, weiter verbessert werden. Die dafür benötigte Orientierung der HoloLens, welche über eine API zugegriffen werden kann, ist für die Lösung dieses Problem allerdings nicht verwendbar. Dies ist darin begründet, dass sich die Position an der initialen Position der HoloLens, beim Start der Anwendung, orientiert [Mic17a] und nicht an einer definierten geographischen Achse. Das Ermitteln einer entsprechenden geographischen Ausrichtung wäre über das in der IMU verbauten Magnetometer möglich. Allerdings gibt es auch hier, wie bei der Tiefenkamera, keine Möglichkeit, auf diese zuzugreifen.

3.3.3 Anzeige der Informationen

Um die ermittelten Informationen darstellen zu können, bedarf es einer Komponente, welche die Anzeige dieser vornimmt. Wie zu Beginn der Arbeit geschildert, existieren grundlegend zwei verschiedene Ansätze um Anwendungen für die HoloLens zu entwickeln. Zum einen können diese mittels DirectX auf API-Ebene entwickelt werden. Einen Ansatz mit einem höheren Level an Abstraktion bietet dagegen die Entwicklung mittels einer Spiel-Engine. Hier arbeitet Microsoft stark mit der Firma Unity Technologies zusammen um die Entwicklung von HoloLens Anwendung mit der Unity Engine zu ermöglichen. Da zum einen viele Dokumentationen, Tutorials und Artikel auf die Entwicklung mit Unity abzielen und es als schnellster Weg in der HoloLens-Entwicklung gilt [Mic17j], ist Unity die Plattform, welche für HoloBillboards ausgewählt wurde.

Unabhängig von der Technologie, die zum Anzeigen der Informationen verwendet wird, gibt es verschiedene Ansätze darüber, wie diese angezeigt werden. Die schon existierenden Anwendungen zeigen, dass es wichtig ist, dem Nutzer eine Orientierungshilfe zu geben. Ohne solche Hilfen kann es vorkommen, dass ein Nutzer nicht weiß, in welche Richtung er blicken soll bzw. an welcher Stelle im Raum die Anwendung gerade seine Aufmerksamkeit verlangt.

Der einfachste Ansatz ist hierbei der des Head-Up-Display (HUD), wie es in Videospielen häufig zur Anzeige von Informationen verwendet wird. Dabei würden im Fall von HoloBillboards die Informationen der Plakatwände an einer festen Position im Display angezeigt. Durch diesen Ansatz kann der Benutzer die Informationen nicht aus dem Auge verlieren. Die Nachteile dieser Technologie sind dabei zum Beispiel, dass immer derselbe Bereich vom Display verdeckt wird. Ein weiteres Problem mit fixierten Anzeigen auf der HoloLens besteht darin, dass sich Benutzer bei Verwendung dieser häufig unwohl fühlen. Dies zeigt sich darin, dass die Anwender das Bedürfnis bekommen, die Anzeige *abzuschütteln* [Mic17c]. Aus diesen Gründen ist ein HUD, zum Anzeigen der Informationen, ungeeignet.

Ein weiterer Ansatz ist, die Informationen auf einem virtuellen Objekt bspw. einer Fläche, welche sich an einer festen Position im Raum befindet, anzuzeigen. Diese Fläche verhält sich ähnlich wie ein Fenster in der Shell, welches im Raum platziert wurde. Es würde seine Position relativ zum Raum beibehalten, wenn der Nutzer sich durch diesen bewegt. Der Nutzer kann das Objekt dabei verlieren und ohne Unterstützung kann es für ihn schwierig sein es wiederzufinden. Um ihm ein Wiederfinden der Anzeige zu erleichtern, können Indikatoren verwendet werden, wie sie bspw. in den Tutorials der Mixed Reality Academy [Mic17e] und im Spiel Robo Raid [Mic17f] verwendet werden. In beiden Anwendungen handelt es sich um einen oder mehrere Pfeile, je nachdem, auf wie viele Objekte in diesem Moment hingewiesen werden soll, welche in die Richtung dieser zeigen. Die Pfeile werden in den genannten Beispielen in der Nähe des Gaze platziert und um diesen rotiert, sodass der Nutzer intuitiv den Gaze in die Richtung des Pfeils bewegen kann um das Objekt zu finden.

Ein weiteres Problem einer festen Position ist es, entscheiden zu müssen an welchem Ort im Raum das Objekt platziert wird. Die Anzeige der Informationen kann dabei bspw. in der Nähe des Nutzers oder neben der Plakatwand erfolgen. Beide Positionen haben Vor- und Nachteile. In der Nähe der Plakatwand, kann die Anzeige für den Nutzer zu weit entfernt sein um die Informationen lesen zu können. Dem kann durch eine Skalierung der Anzeige, in Abhängigkeit von der Entfernung zwischen Nutzer und Objekt, entgegengewirkt werden.

Sowohl zur korrekten Platzierung im Raum, als auch zur Skalierung benötigt man an diesem Punkt die Distanz zur Plakatwand, welche, wie zuvor schon beschrieben, nicht zur Verfügung steht. Bei einer Platzierung

in der Nähe des Nutzers, kann dieser seinen Standort nicht verlassen ohne die Informationen aus dem Auge zu verlieren, wodurch die Mobilität der HoloLens verloren geht.

Eine Lösung, welche eine Kombination der beiden zuvor genannten Ansätzen darstellt, wird in HoloBillboards verwendet. In der HoloLens-Entwicklung wird dies als *Tagalong* bezeichnet. Damit wird beschrieben, dass das entsprechende Objekt, in diesem Fall das Fenster mit den Informationen, zwar als Objekt im 3D-Raum platziert wird, dabei aber mit der Position des Nutzers verankert ist. Dabei hat es für den Nutzer den Eindruck, dass ihm das Objekt folgt, ohne ein unangenehmes Gefühl wie beim HUD zu erzeugen. Das Objekt befindet sich dabei ungefähr auf Augenhöhe des Nutzers in einem Abstand von einem bis zwei Meter, was vom Nutzer als angenehmer Abstand zum Kopf wahrgenommen wird [Mic17c]. Bewegt sich der Nutzer nun durch den Raum, folgt ihm das Objekt am Rand seines Blickfeldes. Dreht der Nutzer seinen Kopf, passt das Fenster seine Position an, sobald es aus dem Blickfeld des Nutzers verschwindet. Dies geschieht allerdings mit einer leichten Verzögerung, was die Bewegung natürlicher wirken lässt [Mic17c]. Da es sich bei dem Fenster um ein Objekt handelt, bei dem nur die frontale Ansicht von Interesse ist, richtet sich dieses immer mit der Front in Richtung des Benutzers aus. Somit sind die Informationen für den Nutzer jederzeit ersichtlich und wieder auffindbar ohne, dass er an eine feste Position im Raum gebunden ist oder ein zu großen Bereich des Sichtfeldes verdeckt wird.

3.3.4 Aktualisieren der Informationen

Eine weitere Anforderung an die Anwendung besteht darin, dass die Daten der Plakatwände, auch nach einer Änderung auf Seiten der awk, in der Anwendung weiterhin aktuell sein müssen. Grundlegend gibt es zwei in Frage kommende Methoden, die Daten für die Nutzung in der Anwendung zur Verfügung zu stellen. Bei der ersten Methode werden die Daten zusammen mit der Anwendung ausgeliefert. Bei der zweiten, werden die Daten von einem Server bereitgestellt und von der Anwendung, sobald diese benötigt werden, abgerufen.

Im Vergleich zur Server Lösung ist die lokale Variante um ein vielfaches schneller beim Abrufen der Daten. Der Grund dafür ist, dass von der Anwendung nicht zuerst eine Verbindung zum Server aufgebaut werden muss um die Daten anschließend über das Internet zu übertragen. Da die Abfrage der Daten nicht in hohem Maße zeitkritisch ist, kann dieser Punkt vernachlässigt werden.

Einen wichtigeren Punkt stellt hier die Aktualität der Daten dar. Die Daten der Plakatwände weisen eine hohe Volatilität auf. Hier kann es zu täglichen Anpassungen an Preis und Verfügbarkeit kommen. Aus diesem Grund kommt für HoloBillboards nur die Verwendung einer Client-Server-

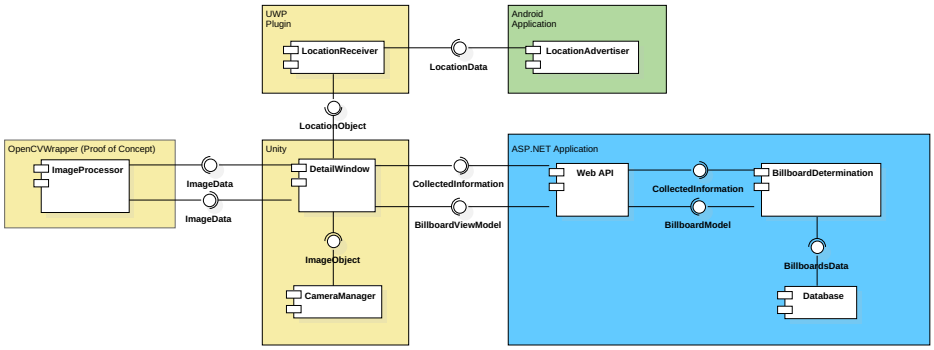


Abbildung 11: Komponentendiagramm HoloBillboards

Architektur in Frage. Das Aktualisieren der Anwendung würde zu lange dauern, da hierbei eine neue Version der Anwendung erstellt und verbreitet werden muss.

Die Daten liegen bei dem gewählten Ansatz in einer zentralen Datenbank, welche zu jederzeit aktualisiert werden kann. In diese können Änderungen jederzeit von Seiten der awk eingespielt werden und sind somit direkt in der Anwendung verfügbar.

3.4 Struktur der Anwendung

Anhand der zuvor beschriebenen Lösungsansätze ergeben sich verschiedene Komponenten aus denen die Anwendung besteht. Die jeweiligen Komponenten sind in Abbildung 11 in logische Bereiche der Anwendung unterteilt. Jeder dieser Bereiche ist mit einer von drei Farbe hinterlegt, da HoloBillboards aus insgesamt drei Anwendungen besteht. Der Name HoloBillboards bezieht sich im Weiteren auf das Gesamtsystem der Anwendungen. Die Anwendung die auf der HoloLens läuft, stellt dabei den Kern des Systems dar. Alle Bereiche, die im Komponentendiagramm gelb hinterlegt sind, gehören zur in Unity entwickelten HoloLens Anwendung. Der grüne Bereich repräsentiert die Android Anwendung und der blaue die ASP.NET Anwendung.

Die Android Anwendung ist ein autarker Bestandteil des Systems. Wenn die Anwendung gestartet wird, sendet sie bei jeder Positionsänderung ein Paket. Diese Pakete werden vom *LocationManager* angenommen und der aktuelle Standort gespeichert. Der LocationManager ist Bestandteil eines UWP Plugins für die Unity Anwendung, da nur so auf die Bluetooth-Funktionalität zugegriffen werden kann.

Das zentrale Element innerhalb der Unity Anwendung ist das *DetailWindow*. Wenn der Nutzer eine Eingabe vornimmt, wird diese vom DetailWindow verarbeitet, da es zum jetzigen Zeitpunkt das einzige Element in

der Anwendung ist, mit dem interagiert werden kann. Wenn der Nutzer einen Air-Tap ausführt, wird der aktuelle Standort aus dem UWP Plugin geladen und die Informationen werden dann an die Web API weitergegeben. Diese gibt die Daten an die *BillboardDetermination*-Komponente weiter, welche anhand der Position die passende Plakatwand aus der Datenbank an die Web API zurückgibt. Diese sendet die Informationen in einem *ViewModel* an die Unity Anwendung zurück, wo diese vom DetailWindow angezeigt werden.

CameraManager und *ImageProcessor* sind Bestandteile einer Machbarkeitsstudie bezüglich der Nutzung der Bibliothek zur Bildverarbeitung *OpenCV*. Bei einem Air-Tap wird durch den CameraManager ein Bild mit der Frontkamera aufgenommen und mit entsprechenden Metadaten als Objekt weitergegeben. Die Bilddaten werden an den *ImageProcessor* weitergeleitet. In diesem werden die Daten zur Demonstration in eine Datenstruktur aus der OpenCV Bibliothek überführt und anschließend an die Unity Anwendung zurückgegeben.

4 Implementierung der Anwendung

In diesem Abschnitt wird auf die Projektstruktur und die Implementierung der Anwendung eingegangen. Es wird beschrieben, wie die standardmäßige Struktur eines HoloLens Projektes aussieht. Um eine verbesserte Übersicht über die zum System gehörenden Projekte zu erhalten, wird eine Alternative Projektstruktur vorgestellt, welche bei der Entwicklung von HoloBillboards verwendet wurde.

In den weiteren Abschnitten wird auf die Implementierung der einzelnen Komponenten eingegangen. Hierbei werden die zentralen Funktionen dieser erklärt. Es wird auch auf einige der, bei der Entwicklung aufgetretenen Probleme eingegangen und erklärt, wie mit diesen umgegangen wurde bzw. wie diese gelöst werden konnten. Außerdem werden die Zusammenhänge zwischen den einzelnen Komponenten erklärt.

4.1 Projektstruktur

Das Erstellen einer Anwendung für die HoloLens mit Unity beginnt in der Regel mit dem Anlegen eines neuen Projektes in Unity. Das Projekt erhält den Namen *HoloBillboardsUnity* und wird in einem gleichnamigen Ordner erzeugt. Alle für das Projekt relevanten Daten befinden sich in diesem Ordner. In Unity wird eine Szene angelegt, in welcher Elemente zur Anwendung hinzugefügt und wieder entfernt werden können. C# Skripte, welche sich im *Assets*-Ordner befinden, können aus Unity heraus in Visual Studio geöffnet und editiert werden. Visual Studio muss hierfür zuvor als Standardeditor ausgewählt sein. Unity erzeugt beim ersten Öffnen eines Skriptes eine Projektmappe, in welche alle in der Anwendung vorhanden Skripte referenziert werden. Wird im Unity Editor ein Skript erstellt, so wird dieses automatisch zum Projekt hinzugefügt. Somit sind jederzeit, in beiden Programmen dieselben Dateien vorhanden.

Um ein Unity Projekt auf der HoloLens auszuführen, muss dieses zuvor vom Unity Editor zu einer UWP Anwendung kompiliert werden. Dazu muss vor dem Build-Prozess im Menü (siehe Abbildung 12) die Universal Windows Plattform ausgewählt werden. Anschließend müssen die entsprechenden Einstellungen ausgewählt werden. Je nach Situation können einige der Einstellungen abweichen. Für die meisten Situationen werden allerdings diese Einstellungen benötigten. Vor dem Prozess muss die zu verwendende Szene ausgewählt werden. Da bei HoloBillboards nur eine Szene existiert, reicht es, die geöffnete Szene hinzuzufügen. Abschließend muss ein Ordner zur Ausgabe der Anwendung angegeben werden. Hierzu wird ein Ordner mit dem Namen *App* innerhalb des Projektordners erstellt und ausgewählt. Bei späteren Build-Prozessen kann dieser wiederverwendet werden.

Die Ausgabe des Build-Prozesses ist wiederum eine Visual Studio Pro-

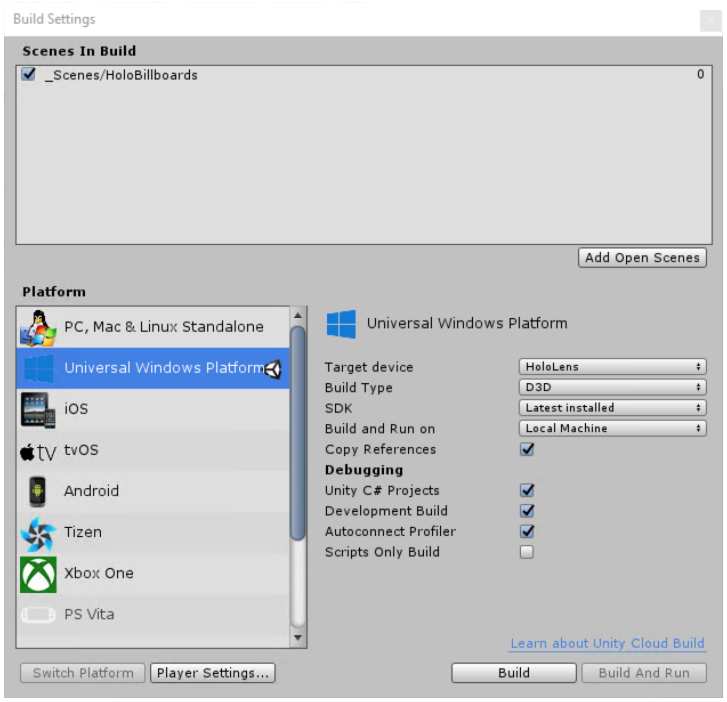


Abbildung 12: Einstellungen des Build-Prozesses in Unity

jektmappe. Darin befindet sich das Projekt, welches auf der HoloLens ausgeführt werden kann. Dieses trägt analog zum Unity Projekt die Bezeichnung *HoloBillboardsUnity*. Zu diesem Zeitpunkt in der Entwicklung existieren zwei Visual Studio Projektmappen, welche beide zur Entwicklung genutzt werden. Teilweise kann ein häufiges Wechseln zwischen den beiden Projektmappen notwendig sein. Dies ist nicht nur für den Entwickler unpraktisch, auch benötigt jede zusätzliche Instanz von Visual Studio Ressourcen auf dem Computer. Durch die Implementierung weiterer Komponenten würde sich die Anzahl der Projektmappen weiter erhöhen. Auch wenn diese nicht immer geöffnet sind, erhöht dieser Zustand den Aufwand der Verwaltung des Gesamtprojektes. Bei HoloBillboards wird dieses Problem gelöst, in dem eine eigene Projektmappe in Visual Studio erstellt wird, in welche die einzelnen Projekte aus den verschiedenen Projektmappen eingebunden werden. Dies ermöglicht den Zugriff auf alle benötigten Projekte innerhalb einer Instanz von Visual Studio. Sollte es notwendig sein, können die ursprünglichen Projektmappen separat geöffnet werden, da diese durch die neue Projektmappe nicht beeinflusst werden.

Die im weiteren Verlauf des Kapitels beschriebenen Komponenten liegen mit dem Unity Projekt in einem gemeinsamen Ordner. Außerdem befindet sich in diesem die HoloBillboards-Projektmappe, welche die einzelnen Projekte enthält. Dies ist nicht zwingend nötig, ermöglicht aber eine

gemeinsame Versionierung und Codeverwaltung aller Projekte mit *Git*.

4.2 Testen der Anwendung

Im Laufe der Entwicklung einer Anwendung muss diese viele Male gestartet und überprüft werden um die Auswirkung von Änderungen an Design und Code zu validieren. Durch den Unity Editor und Visual Studio stehen dem Entwickler vier verschiedene Möglichkeiten zur Verfügung, mit denen er seine Anwendung ausführen kann. Diese reichen vom Ausführen der Unity Anwendung im Editor bis hin zum Bereitstellen und Ausführen der Anwendung auf der HoloLens.

4.2.1 Emulator im Editor

Die einfachste Möglichkeit die Anwendung auszuführen und zu testen, ist sie im Editor der Unity Engine zu starten. Unity ermöglicht dies mit der *Holographic Emulation* Funktionalität. Hierfür wird im gleichnamigen Fenster in Unity der *Emulation Mode* auf *Simulate in Editor* gestellt. Gesteuert wird die Anwendung mittels eines Gamepads, wobei die Bewegungen im Raum und des Kopfes durch die zwei Analog-Sticks simuliert werden. Die Gesten können durch das Drücken von bestimmten Knöpfen simuliert werden.

Diese Möglichkeit hat den Vorteil, dass die Anwendung ohne das Übersetzen in eine UWP-Anwendung und das Bereitstellen auf der HoloLens ausgeführt werden kann. Dies ist besonders geeignet um kleine optische Anpassungen oder marginale Änderungen im Code zu überprüfen. Allerdings stößt diese Methode auch schnell an ihre Grenzen. So ist die Simulation mit dem Gamepad zwar intuitiv, allerdings bekommt der Entwickler so kein Gefühl dafür, wie sich die Anwendung auf der HoloLens bedienen lässt. Auch der visuelle Eindruck ist nicht mit dem der HoloLens vergleichbar, da Objekte auf der HoloLens in vielen Aspekten anders aussehen als auf dem Bildschirm. Für kleine Änderungen ist diese aber ausreichend. Außerdem kann plattformspezifischer Code nicht im Emulator ausgeführt werden. Auch kommt es häufiger dazu, dass eine Anwendung, die im Editor problemlos läuft, auf der HoloLens aus verschiedensten Gründen nicht lauffähig ist. Somit ist diese Methode eine gute Ergänzung, aber alleine nicht ausreichend zur Entwicklung einer HoloLens Anwendung.

4.2.2 Holographic Remoting Player

Eine weitere Möglichkeit zum Testen der Anwendung besteht in der Nutzung des *Holographic Remoting Player*. Dieser wird auf der HoloLens ausgeführt und kann aus dem App Store heruntergeladen werden [Mic16a]. Wenn die Anwendung auf der HoloLens ausgeführt wird, kann im Unity Editor eine Verbindung zu dieser hergestellt werden. Hierzu muss in Unity,

im Fenster Holographic Emulation der Emulator Mode auf die Option *Remote to Device* gestellt und eine Verbindung zur HoloLens hergestellt werden. Wird die Anwendung anschließend in Unity ausgeführt, erfolgt die Ausgabe des Bildes zusätzlich auf der HoloLens. Die Anwendung lässt sich dabei über die Eingabemethoden der HoloLens steuern, wird aber nach wie vor auf dem Computer, auf welchem der Unity Editor läuft, ausgeführt. Der Datenaustausch zwischen der HoloLens und dem Unity Editor erfolgt dabei über ein lokales Netzwerk, mit dem beide Geräte verbunden sein müssen. Durch die Ausführung im Unity Editor ist es auch bei dieser Methode nicht möglich, plattformspezifischen Code auszuführen.

Durch die Übertragung der Anwendung auf die HoloLens hat diese Methode den Vorteil gegenüber des Editors, dass der Entwickler die Dimension der Objekte in Relation zur realen Welt sehen kann. Die Bedienung der Anwendung ist zwar mittels den Eingabemethoden der HoloLens möglich, eignet sich aber nicht zur Validierung dieser. Das liegt an der Verzögerung, mit der die Eingaben verarbeitet werden und somit zu einer Veränderung in der Szene führen. Die Verzögerung ist durch die Übertragung der Daten über das lokale Netzwerk bedingt. Die Startzeit der Anwendung ist dabei, nach einmaligem Starten der Holographic Remoting App und dem Herstellen der Verbindung zur HoloLens, ähnlich schnell wie bei der Ausführung im Editor. Dadurch eignet sich die Methode zur visuellen Überprüfung der Anwendung.

4.2.3 HoloLens Emulator

Visual Studio stellt dem Entwickler eine weitere Möglichkeit zum Testen der Anwendung zur Verfügung. Das von Unity erstellte UWP Projekt kann mit Visual Studio in einem, auf Microsofts Virtualisierungs-Technologie Hyper-V basierenden Emulator ausgeführt werden. Die Steuerung erfolgt dabei über Maus und Tastatur. Im Vergleich zum Editor, kann in diesem Emulator auch plattformspezifischer Code ausgeführt werden. Darüber hinaus, ist dieser nicht nur in der Lage die Anwendung auszuführen, es ist ihm auch möglich das Betriebssystem der HoloLens inklusive der Shell zu emulieren. Dies bietet dem Entwickler zum Beispiel die Möglichkeit die Verknüpfung der Anwendung zu testen. In Bezug auf das Testen von Darstellung und Bedienung verhält es sich mit dem Emulator analog zum Testen der Anwendung im Emulator des Unity Editor.

4.2.4 Bereitstellen auf HoloLens

Um eine Anwendung in vollem Umfang zu testen, muss diese auf der HoloLens bereitgestellt werden. Dies kann in Visual Studio über zwei Wege erfolgen. Entweder wird die HoloLens mit einem USB-Kabel an den Computer angeschlossen oder die Anwendung wird über das lokale Netz-

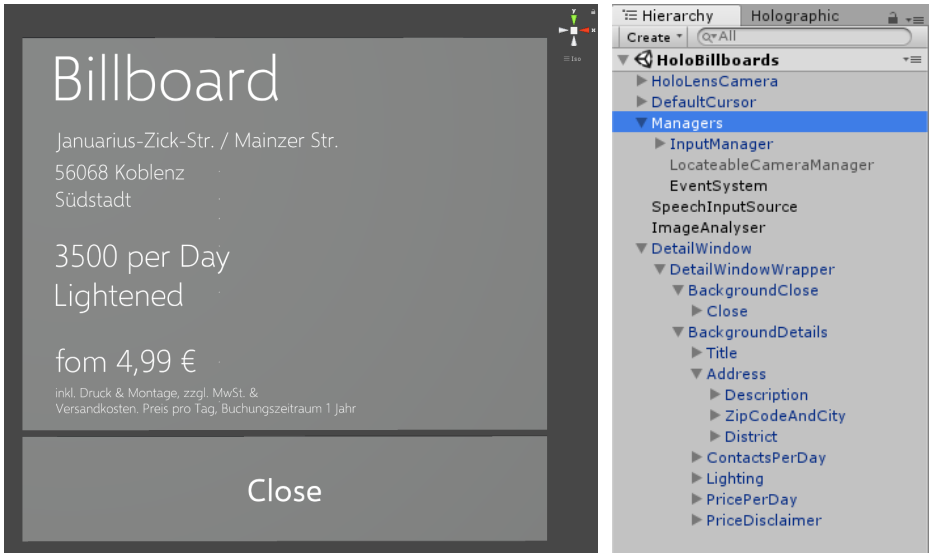
werk übertragen. Der direkte Anschluss an den Computer hat den Vorteil, dass die Bereitstellung wesentlich schneller als über das WiFi der HoloLens möglich ist. Mit dem mitgelieferten Kabel kann sich der Entwickler nur sehr eingeschränkt bewegen. Über WiFi müssen dagegen etwas längere Zeiten zum Übertragen der Anwendung in Kauf genommen werden. Dafür kann der Entwickler die App ohne Einschränkung testen. Bei den Einstellungen zum Ausführen der Anwendung ist zu beachten, dass die Anwendung, wenn die Konfiguration *Debug* ausgewählt wird, nur mit max. 30 Bildern pro Sekunde ausgeführt wird. Für die optimale Bildrate von 60 Bildern pro Sekunde müssen die Einstellungen *Release* oder *Master* ausgewählt werden.

4.3 Unity Anwendung

Bevor die eigentliche Entwicklung einer HoloLens Anwendung beginnen kann, müssen einige Einstellungen an der Szene und im Unity Projekt vorgenommen werden. Dazu wird das MixedRealityToolkit-Unity in das Projekt importiert, welches als *Asset Package* aus einem *Github Repository* [Mic16b] heruntergeladen werden kann. Außerdem müssen die bei der Erstellung des Projektes in der Szene platzierten Objekte entfernt werden, um anschließend das *HoloLensCamera* Prefab in der Szene zu platzieren. Bei diesem Prefab handelt es sich um eine Kamera, deren Position, Clear Flag und Clipping Planes für die HoloLens angepasst wurden. Die Kamera entspricht dabei der Perspektive, aus welcher die virtuellen Objekte in der Anwendung gerendert werden. Die Position der Kamera wird beim Ausführen der Anwendung durch die Sensordaten der HoloLens entsprechend der Bewegungen des Nutzers angepasst.

Das Mixed Reality Toolkit bietet viele nützliche Skripte, Prefabs und Shader. Zum Zeitpunkt der Entwicklung ist die Dokumentation teilweise lückenhaft bzw. nicht immer ausreichend detailliert. Zur Verwendung und Funktionsweise vieler Elemente gibt es keine Erklärung, weshalb nicht zu allen verwendeten Elementen in dieser Ausarbeitung eine detaillierte Beschreibung erfolgen kann. Auch konnte durch die fehlende oder teilweise nur sehr kurze Beschreibung der einzelnen Komponenten nicht immer sichergestellt werden, dass die verwendete Lösung der bestmöglichen entspricht.

Des Weiteren müssen zwei Einstellungen des Projekts angepasst werden. Zum einen muss die Qualität des Projekts angepasst werden. Hierzu muss in den Qualitätseinstellungen die Auswahl von *Fantastic* auf *Fastes* gesetzt werden, um Einbrüche der Framerate zu verhindern. Dies ist notwendig, da die HoloLens eine verhältnismäßig geringe Grafikrechenleistung besitzt. Zum anderen muss im Abschnitt *Rendering* der Einstellungen des Spielers die Unterstützung für Virtual Reality ausgewählt werden und *Windows Holographic* (in zukünftigen Versionen *Windows Mixed Reality*) als



(a) Detailfenster **(b)** Hirarchie der Szene

Abbildung 13: Screenshots aus dem Unity Editor

Software Development Kit (SDK) ausgewählt werden.

4.3.1 Aufbau des UI

Zur Anzeige der Informationen wird ein Fenster (siehe Abbildung 13a) erstellt. Zur Erstellung dieses Detailfensters wird der Szene (siehe Abbildung 13b) in Unity das `DetailWindow` GameObject hinzugefügt. Diesem wird das gleichnamige Skript hinzugefügt. In dieser Klasse sind Methoden zum Abrufen und Anzeigen der Informationen implementiert. Das Material für den Hintergrund des Fensters verwendet den *Fast Configurable* Shader. Hierbei handelt es sich um einen schnellen Shader aus dem Mixed Reality Toolkit. Die Informationen werden in Textfeldern, deren Inhalte dynamisch verändert werden können, innerhalb des Fensters dargestellt. Des Weiteren wird ein Button zum Schließen des Detailfensters hinzugefügt. Führt der Anwender einen Air-Tap auf diesem aus, wird der `DetailWindowWrapper` (siehe Abbildung 13b) deaktiviert. Durch die Deaktivierung eines GameObject in Unity wird dieses, zusammen mit allen Kind-Elementen ausgeblendet. Das `DetailWindow` Skript läuft weiter, da das `DetailWindow` weiterhin aktiv ist.

Zur Erstellung der Textfelder wird das `UITextPrefab` aus dem Mixed Reality Toolkit verwendet. Das Prefab ermöglicht eine konsistente Nutzung von Schriftgrößen innerhalb der Anwendung. Dieselbe Schriftgröße würde bei unterschiedlichen Skalierungen der Objekte zu unterschiedlich großen Schriften führen. Der Skalierungsfaktor ist so gewählt, dass Schrift-

größen aus Design Programmen übernommen werden können [Mic17i]. In HoloBillboards besitzen die Textfelder ein skaliertes Eltern-Element. Um die zuvor beschriebe Eigenschaft zu erhalten, müssen die Prefabs zuerst ohne Eltern-Element in der Szene platziert werden. Im nächsten Schritt können diese dann zugewiesen werden. Nur so bleibt die Skalierung erhalten.

Dem Detailfenster werden weitere Skripte des Mixed Reality Toolkit hinzugefügt. Das `Tagalong`-Skript, fügt dem Detailfenster dabei die gleichnamige Eigenschaft hinzu, wodurch dieses mit der Position des Nutzers verankert wird und ihm bei seinen Bewegungen folgt. Mittels der vom Skript zur Verfügung gestellten Parameter kann das Verhalten in Bezug auf Abstand zum Nutzer und Bewegungsgeschwindigkeit, angepasst werde. Außerdem wird das `Billboard`-Skript, ebenfalls aus dem Mixed Reality Toolkit stammend, hinzugefügt. Dieses sorgt dafür, dass das Detailfenster immer zum Anwender ausgerichtet ist.

Der `DetailWindowWrapper` wird während der Initialisierung der `DetailWindow` Klasse, welche zum Start der Anwendung erfolgt, deaktiviert. Er wird wieder aktiviert, wenn der Nutzer Informationen zu einer Plakatwand anfordert. Eine Alternative hierzu wäre, das Detailfenster als Prefab zu definieren und bei jeder Anfrage ein neues Objekt zu initialisieren. Dies hätte zum Vorteil, dass das Detailfenster, solange es nicht angezeigt wird, keinen Speicherplatz im Arbeitsspeicher belegt. Dies ist aber bei den wenigen Daten und der entsprechend geringen Speichergröße zu vernachlässigen. Dahingegen ist das Initialisieren eines Objektes wesentlich aufwändiger, als das Reaktiveren. Gerade auf einer schwächeren, mobilen CPU, wie in der HoloLens verbaut, sollte dies vermieden werden. Daraus ergibt sich allerdings die Einschränkung, dass nur eine Instanz des Detailfensters existieren kann und somit auch nur Informationen zu einer Plakatwand angezeigt werden. Für den in HoloBillboards gewählten Ansatz, stellt dies jedoch kein Problem dar.

4.3.2 Benutzerinteraktion

Um dem Nutzer die Interaktion mit der Anwendung zu ermöglichen, benötigt dieser einen Cursor. Dazu wird das `DefaultCursor` Prefab der Szene hinzugefügt. Dabei handelt es sich um einen Cursor, der die Position des Gaze anzeigt. Dieser ist dabei in Erscheinung und Verhalten dem der HoloLens-Shell identisch. Der Cursor ist dabei im Ausgangszustand ein weißer Punkt. Wird die Bereit-Geste (siehe Abbildung 5a) erkannt, wird der Punkt zu einem Kreis transformiert. Wird ein Air-Tap erkannt, wird der Kreis kurzzeitig wieder zum Punkt und kehrt nach Beenden der Geste in die Kreisform zurück.

Die Anwendung muss des Weiteren in der Lage sein, Gesten des Benutzers zu erkennen und zu verarbeiten. Dazu wird ein leeres GameObject,

welches die Bezeichnung *Managers* trägt, erstellt. Diese kann als eine Art Ordner betrachtet werden um weitere Elemente in der Szenenhierarchie zu gruppieren. Innerhalb des Managers Objekt wird das `InputManager` Prefab aus dem Mixed Reality Toolkit hinzugefügt. Der InputManager verarbeitet die Eingaben des Nutzers [Eri16]. Über ein Event System benachrichtigt der `InputManager` die entsprechenden Objekte über die Eingaben des Nutzers. Klassen, die an Objekte angehängt wurden, haben die Möglichkeit auf diese Events zu reagieren, indem sie entsprechende Interfaces implementieren. Um einen Air-Tap zu erkennen muss bspw. das `IInputClickHandler` Interface implementiert werden.

Als dritte Methode, zur Interaktion mit der Anwendung, wird die Spracherkennung integriert. Hierzu wird ein weiteres leeres Objekt mit der Bezeichnung *SpeechManager* hinzugefügt, welches ebenfalls als Kind-Elemente des Managers Objekt in der Szene platziert wird. Dem SpeechManager wird das, ebenfalls aus dem Mixed Reality Toolkit stammende Skript `SpeechInputSource` hinzugefügt. Für dieses Skript können in den Eigenschaften des entsprechenden GameObject Schlüsselwörter eingetragen werden, auf welche die Spracherkennung reagieren soll. Optional kann für jedes Schlüsselwort auch eine Taste auf der Tastatur ausgewählt werden, durch deren Drücken die Erkennung eines Schlüsselwortes simuliert werden kann. Letzteres ist nur beim Testen der Anwendung im Editor möglich. Durch die Implementierung des `ISpeechHandler` Interface, haben die Objekte die Möglichkeit, auf die Erkennung von Schlüsselwörtern zu reagieren.

Die `DetailWindow` Klasse implementiert beide zuvor genannten Interfaces. Durch die Einbindung des `IInputClickHandlers` Interface muss die Klasse die Methode `OnInputClicked` implementieren. Diese wird aufgerufen, wenn ein Air-Tap erkannt wird, während der Gaze auf dem entsprechenden Objekt liegt. Hierbei muss beachtet werden, dass das Detailfenster zu diesem Zeitpunkt deaktiviert ist und der Nutzer auf die reale Plakatwand zielt. Hierzu bietet der `InputManger` die Möglichkeit, ein GameObject als Fallback für eine Eingabe zu registrieren. Wird das Detailfenster als solches registriert, wird bei jeder registrierten Eingabe, die nicht von einem anderen Objekt behandelt wird, die `OnInputClicked` Methode des Detailfensters ausgeführt. Durch den Aufruf dieser Methode wird mit der Aktivierungsprozedur für das Detailfenster begonnen. Innerhalb dieser Prozedur werden die folgenden Aufgaben erledigt:

- Das Abfragen des Standortes in der `GpsReceiver` Klasse.
- Die Aufnahme des aktuellen Bildes der Webcam mittels der `LocateableCameraManager` Klasse [Mic17d].
- Das Weiterleiten des aufgenommenen Bildes an die `ImageAnlyser` Klasse.

- Das Senden der Informationen an die Web API.
- Das Verarbeiten der Daten aus der Antwort der Web API.
- Das Übertragen der empfangenen Daten in die entsprechenden Textfelder.
- Die Aktivierung des `DetailWindowWrapper`.

Analog zur Verwendung des `IInputClickHandler` wird durch die Verwendung des `ISpeechHandler` die Methode `OnSpeechKeywordRecognized` implementiert, in welcher, je nach erkanntem Schlüsselwort, entsprechende Methoden aufgerufen werden.

4.3.3 Abrufen der Informationen

Bevor das Detailfenster aktiviert wird, müssen die anzuzeigenden Informationen von der Web API abgefragt werden. Im ersten Schritt müssen dafür die Standortdaten ermittelt werden. Ein Objekt der Klasse `GpsReceiver`, welche für den Empfang und die Speicherung dieser Standortdaten zuständig ist, wird beim Initialisieren der Klasse `DetailWindow` instanziiert. Von diesem können die zuletzt empfangenen Standortdaten abgerufen werden. Da nur die Klasse `DetailWindow` auf diese Information zugreift, ist es nicht notwendig, die Klasse `GpsReceiver` über ein GameObject öffentlich zugänglich zu machen.

Die erhaltenen Informationen zum Standort werden im Anschluss an die Web API übermittelt. Dazu wird das Unity Plugin Http Client [Cla17] verwendet. Möglich ist auch die Verwendung, der in Unity integrierten Technik zur Kommunikation mit einer Web API. Http Client bietet hier aber eine einfachere Möglichkeit den Aufruf durchzuführen. Im ersten Schritt wird ein `Uri` Objekt erzeugt, welches die URL der Web API und die Standortdaten enthält. Im Zweiten wird das Objekt zusammen mit einer Callback Methode, welche nach Erhalten der Antwort ausgeführt wird, an den Http Client übergeben und die Anfrage an die API gesendet.

In der Callback Methode wird nach erfolgreicher Prüfung des Http Status Code das erhaltene Ergebnis, welches im JSON-Format vorliegt, in ein `BillboardViewModel` Objekt geparst. Dieses enthält nach dem Parsen die Informationen zur Plakatwand. Für das Übertragen der Informationen in die Textfelder muss auf diese zugegriffen werden können. Dies kann in Unity ermöglicht werden, indem die `DetailWindow` Klasse für jedes zu befüllende Textfeld eine öffentliche Variable vom Typ `Text` implementiert. Diese können im Unity Editor mit den Textfeldern verknüpft werden. Somit kann im Skript über die Variablen auf die Inhalte der Textfelder zugegriffen werden.

4.4 Android Anwendung

Die Aufgabe der Android Anwendung ist es, den Standort des Smartphones an die, auf der HoloLens laufende Anwendung zu übermitteln. Dies geschieht, wie von Oslebo [Osl17] beschrieben, mittels BLE Advertising Paketen. Zur Entwicklung der Anwendung wird die Android Studio IDE verwendet. Beim Erstellen der Anwendung wird von Android Studio ein Grundgerüst für die Anwendung angelegt. Die Anwendung wird dabei in der Klasse `MainActivity` implementiert.

Nach dem Start der Anwendung wird ein `LocationManager` erstellt, welchem ein `LocationListener` Objekt übergeben wird. Das `LocationListener` Objekt implementiert eine Callback Methode, die durch den `LocationManager` bei einer Standortänderung ausgeführt wird. Die Callback Methode erhält als Parameter den aktuellen Standort und führt die Schritte zum Versenden der Daten aus. Hierzu werden die Standortdaten in ein Byte-Array übertragen und gleichzeitig auf dem Bildschirm angezeigt. Das Byte-Array und der Company Identifier, welcher als ID zur Erkennung des Paketes verwendet wird, werden zusammen mit der Konfiguration der Übertragung, an den `BluetoothLeAdvertiser` übergeben, welcher das Paket versendet.

4.5 UWP Plugin

Um die von der Android Anwendung gesendeten Daten auf der HoloLens zu empfangen, wird eine Komponente benötigt, die diese Funktionalität mittels der Bluetooth API ermöglicht (vgl. [Osl17]). Der Zugriff auf die Bluetooth Funktionalität ist über die API der Universal Windows Platform möglich. Solange die Anwendung im Editor ausgeführt wird, kann allerdings nicht auf die plattformspezifische UWP API zugegriffen werden. Unity bietet hier zwei Möglichkeiten diese trotzdem zu nutzen, sobald die Anwendung auf der HoloLens oder dem HoloLens Emulator ausgeführt wird. Die erste Möglichkeit ist die des plattformabhängigen Kompilierens. Hierzu werden im Code *Compiler-Flags* gesetzt. Mittels dieser Flags können bestimmte Abschnitte, in Abhängigkeit der Zielplattform kompiliert bzw. übersprungen werden. So kommt es, bei im Editor fehlenden APIs nicht zu Fehlern. Da dies aber zu unübersichtlichem Code führt, wird in HoloBillboards der Ansatz gewählt, bei dem der Code, welcher die UWP API nutzt, in eine separate Bibliothek ausgelagert wird. Diese wird im Weiteren als *HoloBillboardsUWPAPI* bezeichnet. Um die Anwendung auch im Editor ausführen zu können, wird eine weitere Bibliothek erstellt, welche die gleichen Klassen und Methoden enthalten muss wie die HoloBillboardsUWPAPI-Bibliothek besitzt.

Beide Bibliotheken werden in der HoloBillboards-Projektmappe angelegt. Dabei ist das UWP Plugin eine Universal Windows Bibliothek und

das Editor Plugin eine .NET-Framework Bibliothek mit der .NET Version 3.5. Es muss sichergestellt werden, dass beide Projekte dieselbe Versionsnummer tragen und die exportierten Dynamic Linked Library (DLL) Dateien gleich benannt sind [Uni17]. Beides kann in den Einstellungen der jeweiligen Projekte konfiguriert werden. Damit die erstellten Dateien direkt von Unity erkannt und verwendet werden können, muss als Ausgabeordner der Plugin Ordner des HoloBillboardsUnity-Projekt gewählt werden. Da beide Plugins denselben Namen tragen müssen, wird das Editor Plugin nach Konvention in den Plugin Ordner gelegt und das UWP Plugin in einen Unterordner.

Unity selber erkennt die Bibliotheken sobald diese im Plugin Ordner erstellt werden. Für die korrekte Zuordnung müssen diese noch konfiguriert werden. Hierzu wird in den Eigenschaften des Editor Plugins die Editor Plattform ausgewählt. Analog dazu wird für das UWP Plugin der Eintrag *WSAPlayer* gewählt. Außerdem muss bei den Einstellungen des UWP Plugins ein Platzhalter ausgewählt werden. Hier wird das Editor Plugin ausgewählt. Durch diese Konfiguration wird beim Ausführen der Anwendung im Editor das Editor Plugin geladen und bei der Ausführung auf der HoloLens das UWP Plugin.

Das Plugin selber besteht aus zwei Klassen. Der `GpsDataPacket` Klasse, welche die empfangenen Byte-Arrays konvertiert und als Attribute speichert und der `GpsReceiver` Klasse. Diese empfängt die Daten von der Android Anwendung und stellt die aktuelle Position für andere Klassen zur Verfügung. Bei der Instanziierung des `GpsReceiver` Objekts wird im Konstruktor der Klasse ein `BluetoothLEAdvertisingWatcher` Objekt erzeugt. Diesem wird, sowohl der zuvor auch in der Android Anwendung verwendete Company Identifier, als auch eine Callback Methode übergeben. Diese wird ausgeführt, sobald ein Advertising Paket mit entsprechendem Identifier empfangen wurde. In der Callback Methode wird anhand der empfangenen Daten ein `GpsDataPacket` Objekt erzeugt, welches als aktuelle Position in der Klasse gesetzt wird.

4.6 ASP.NET Anwendung

Die in der HoloLens Anwendung verwendete Web API wird von einer ASP.NET Anwendung zur Verfügung gestellt. Diese nimmt die Anfragen auf einer bestimmten URL an und ermittelt anhand der Parameter die Informationen zur benötigten Plakatwand. Die Daten werden mittels des Entity Framework aus der Datenbank abgefragt und in der Anwendung verarbeitet. Das Ergebnis wird anschließend, als HTTP-Response an die HoloLens Anwendung zurückgeben.

Die Anwendung wird als weiteres Projekt mit der Bezeichnung *HoloBillboardsAPI* zur Projektmappe hinzugefügt. Zur Verwendung des EF, muss dieses über den Nuget-Paketmanager dem Projekt hinzugefügt wer-

den. Um für die Anwendung einen entsprechenden API Endpunkt zu erstellen, wird der `BillboardController` erstellt. Ein Controller ist eine Klasse, welche die Klasse `ApiController` erweitert und eigene Methoden zum Entgegennehmen von HTTP Anfragen implementiert. Der Klassenname `BillboardControler` ergibt sich aus der Konvention den Controller nach der Entität, auf welche zugegriffen wird, hier `Billboard`, zu benennen.

Der `BillboardController` besitzt eine Methode zum Annehmen der Anfrage. Dabei werden die in der URL mitgegebenen Parameter vom Web API Framework automatisch zu Parametern im Methodenaufruf. Somit sind die Daten direkt zur Weiterverwendung verfügbar und müssen nicht aus der URL geparst werden. Das EF stellt für den Zugriff auf die Daten in der Datenbank einen `DataContext` zur Verfügung. Über dieses Objekt kann auf alle `Billboard` Einträge in der Datenbank zugegriffen werden. Mittels eines Language-Integrated Query werden diese nach der Distanz zum aktuellen Standort des Benutzers sortiert. Anschließend wird der Eintrag mit der geringsten Distanz ausgewählt.

Es ist üblich, für jede Entität zwei Datenmodelle zu erstellen. Eines, in diesem Beispiel die Klasse `Billboard`, welches eine Repräsentation der Einträge in der Datenbank darstellt. Ein weiteres, hier das `BillboardViewModel`, welches verwendet wird um die Informationen nach außen zur Verfügung zu stellen. Hierbei wird versucht, nur so viele Informationen wie nötig nach außen verfügbar zu machen. Das erhöht zum einen die Sicherheit, da ein potentieller Angreifer weniger Informationen über das System erhalten kann. Zum anderen wird durch eine Reduzierung der Informationen auch die zu übertragende Datenmenge geringer, was gerade bei großen Entitäten sinnvoll ist. Das `Billboard` Objekt muss somit in ein `BillboardViewModel` übertragen werden. Dazu werden die benötigten Werte manuell in ein neues View Model kopiert. Das View Model wird dann zusammen mit dem Entsprechenden HTTP Status Code 200 zurückgegeben. Dabei wird das View Model von der Web API automatisch in einen JSON String serialisiert. Sollte keine Plakatwand gefunden worden sein, wird dies als `NotFound` Ergebnis von der API zurückgegeben.

4.7 Bibliothek für Datenmodelle

Häufig müssen bei Client-Server Infrastrukturen auf beiden Seiten die gleichen Datenmodelle verwendet werden. Im vorhandenen System trifft dies auf das `BillboardViewModel` zu. Dieses wird sowohl serverseitig in der Web API verwendet, als auch auf Seiten der HoloLens Anwendung, wo das Resultat wieder zu einem Objekt geparst werden muss. Hierbei muss dafür gesorgt werden, dass auf Seiten der HoloLens Anwendung strukturell dasselbe Objekt geparst wird, wie auf der Seite des Servers serialisiert wurde. Dabei ist es von Vorteil, wenn auf beiden Seiten dieselbe Klasse verwendet

wird. Diese müssen dann nur an einem Ort verwaltet werden und unterschiedliche Versionen der Klasse können vermieden werden. Die Klassen benötigen allerdings auf Seiten der HoloLens für die Kompatibilität mit Unity leichte Anpassungen. Diese Anpassungen werden serverseitig nicht benötigt, weshalb diese hier vermieden werden können und die Größe der Klasse in der Serveranwendung somit reduziert werden kann. Dies ist bei der jetzigen Größe der Anwendung kein entscheidender Faktor, das System wird aber modularer, was zu einer besseren Wartbarkeit führt.

Zur Implementierung wird ein Projekt mit dem Namen *HoloBillboards-ModelLibrary* angelegt. Bei der Anwendung handelt es sich um eine .NET 3.5 Bibliothek. In dieser wird das `HoloBillboardsViewModel` mit seinen Eigenschaften implementiert. Dieses kann somit zum einen von der ASP.NET Applikation durch den Import der Bibliothek direkt verwendet werden, zum anderen von der Unity Anwendung zur Vererbung an ein eigenes View Model verwendet werden. Die Unity Anwendung muss das View Model zusätzlich als serialisierbar deklarieren und eine entsprechende Methode zur Serialisierung verwenden um aus dem JSON der API ein Objekt zu erzeugen.

4.8 OpenCV Einbindung

OpenCV ist eine der bekanntesten Bibliotheken für Aufgaben im Bereich des maschinellen Sehens. Dieser Bereich ist gerade für Anwendungen auf der HoloLens interessant, da es bei der Interaktion mit der realen Umgebung häufig zu Situationen kommt, in denen Informationen über die Umgebung von Interesse sind. Ein Beispiel hierfür ist die Erkennung von Plakatwänden in Bildern. Aus diesem Grund wird in der Anwendung die Möglichkeit der Nutzung von OpenCV geprüft.

Es existieren einige Plugins für Unity, welche eine Unterstützung von OpenCV in Unity ermöglichen. Das Plugin *OpenCV for Unity* zeigt darüber hinaus die Kompatibilität zur HoloLens in einem Beispielprogramm [Eno17]. Dieses ist aber weder kostenfrei noch ist der Quellcode frei verfügbar. Daher wurde von der Verwendung dieses Plugins abgesehen.

Eine Alternative hierzu bietet der Ansatz einen Wrapper für die OpenCV Bibliothek zu implementieren, welcher wiederum von der Unity Anwendung importiert werden kann (vgl. OpenCVAndUnity [Was]). Der Ansatz basiert dabei auf der Möglichkeit, Bibliotheken, welche in C++ geschrieben wurden, als Plugins in Unity einzubinden. Der hierbei zu beachtende Unterschied zwischen C# und C++ Bibliothek ist, dass es sich bei C++ um nicht verwalteten Code handelt. Bei nicht verwaltetem Code muss sich der Entwickler u. a. um die Verwaltung von Pointer und die *Garbage Collection* kümmern. Dahingegen muss sich der Entwickler bei verwaltetem Code, wozu die C# Skripte der Unity Anwendung und der Code der UWP API gehören, darüber, in der Regel keine Gedanken machen. Werden

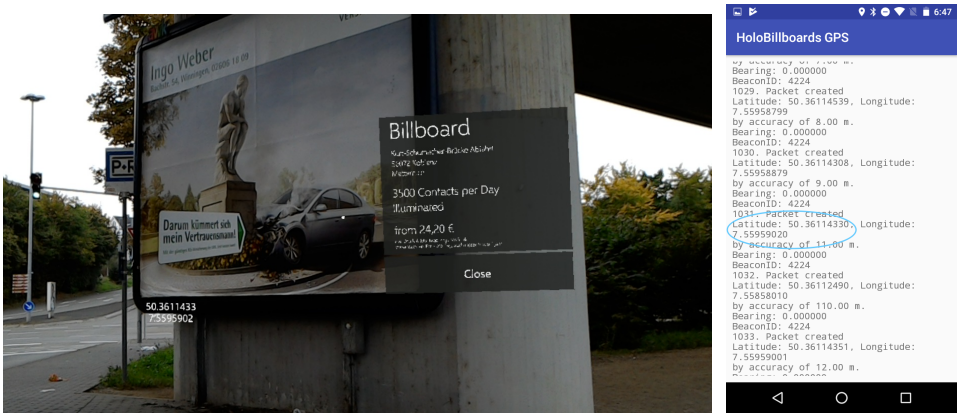
Plugins mit nicht verwaltetem Code verwendet, muss besonders auf die Übergabe der Daten zwischen den beiden Arten von Code geachtet werden.

Der Ansatz besteht dabei aus zwei Komponenten. Zum einen die `ImageAnalyser` Klasse, welche in C# geschrieben ist und einem entsprechenden GameObject zugewiesen ist. Von dieser wird die Wrapper Bibliothek `ImageProcessor` eingebunden, welche die zweite Komponente bildet. In dieser kann dann die OpenCV Bibliothek verwendet. Da es sich beim `ImageProcessor` um eine C++ Bibliothek handelt, wird das C++ Interface von OpenCV verwendet.

Der Download von OpenCV umfasst eine kompilierte Version der Bibliothek als DLL-Datei. Diese ist ausschließlich als 64-Bit Version verfügbar und somit nicht kompatibel zur HoloLens. Allerdings wird diese benötigt um die Anwendung im Simulator des Unity Editor zu testen, da es sich bei diesem um ein Programm mit 64-Bit-Architektur handelt. Die DLL-Datei wird in den Plugin Ordner gelegt und in Unity als 64-Bit Plugin konfiguriert. Dies kann in Verbindung mit dem `ImageProcessor` im Simulator des Unity Editors erfolgreich getestet werden.

Für die Verwendung von OpenCV auf der HoloLens selber wird eine 32-Bit Version der Bibliothek benötigt. Diese kann mit dem Quellcode erzeugt werden. Dies wurde mit den Versionen 3.2.0 und 3.3.0 von OpenCV versucht. In beiden Fällen wurden entsprechende Bibliotheken mit, den Daten des Headers zufolge, 32-Bit-Architektur erzeugt. Mit beiden Versionen kann keine, auf der HoloLens lauffähige Bibliothek erzeugt werden. Die DLL-Dateien können beim Start der Anwendung nicht geladen werden, mit dem Hinweis, dass diese eine unbekannte Architektur aufweisen.

Der zuvor beschriebene Ansatz kann somit nur als bedingt funktionsfähig bezeichnet werden. Die Verwendung im Simulator des Unity Editor stellt hierbei kein Problem dar. Dies kann durch die Implementierung des beschriebenen Ansatzes gezeigt werden. Auf der HoloLens ist dies, aus den beschriebenen Gründen, nicht möglich. Somit kann zum jetzigen Zeitpunkt kein Weg aufgezeigt werden, mit dem die OpenCV Bibliothek ohne den Kauf eines Plugins in eine HoloLens Anwendung eingebunden werden kann.



(a) Sicht durch die HoloLens **(b)** GPS Log

Abbildung 14: Anwendung in Benutzung an der Abfahrt der Kurt-Schumacher-Brücke in Koblenz

5 Auswertung

In den folgenden Abschnitten erfolgt die Evaluation der Anwendung HoloBillboards. Diese erfolgt zum einen anhand der in der Konzeption erhobenen Anforderungen. Hierbei wird betrachtet, ob und wie diese erfüllt werden. Für den Fall, dass eine Anforderung nicht oder nur zum Teil erfüllt wird, werden die hierfür verantwortlichen Probleme genannt. Zum anderen wird die Forschungsfrage aus der Aufgabenstellung beantwortet. Hierbei werden auf der einen Seite die Möglichkeiten erörtert, welche sich mit einer neuen Technologie wie der HoloLens ergeben. Auf der anderen Seite wird auch aufgezeigt, welche Defizite sowohl bei der Hardware als auch bei der Software, in Bezug auf die Anwendung und auch im Allgemeinen, existieren. Auch wird hierbei darauf eingegangen, wie gut sich zum jetzigen Zeitpunkt Anwendungen für die HoloLens entwickeln lassen.

5.1 Anforderungserfüllung

Plakatwandidentifizierung Die in der Auflistung an erster Stelle stehende Anforderung verlangt, dass die Anwendung in der Lage ist die Plakatwand zu bestimmen, zu welcher der Nutzer Informationen anfordert. Diese Anforderung kann die Anwendung nur in Teilen erfüllen. Hierbei ist es stark von der gegebenen Situation abhängig, ob die Plakatwand korrekt bestimmt wird. Die Bestimmung der Plakatwände geschieht mittels GPS. In der HoloLens Anwendung (siehe Abbildung 14a) sind die GPS-Daten zu sehen, welche von der Android Anwendung gemessen und übertragen werden (siehe Abbildung 14b). Das Einblenden der GPS-Koordinaten geschieht hier zu Demonstrations- und Testzwecken.

In einigen Konstellationen kann eine korrekte Bestimmung der Plakatwand nur durch GPS jedoch nicht sichergestellt werden. Eine korrekte Identifizierung ist nur in dem Fall möglich, in dem alle anderen Plakatwände eine größere Entfernung zum Nutzer aufweisen. Das in diesem Aspekt nicht den Anforderungen genügende Ergebnis ist auf mehrere Punkte zurückzuführen. Ein Aspekt hierbei ist der eingeschränkte Zugriff auf die Sensoren der HoloLens. Mit einem Zugriff auf die Daten der IMU kann das Ergebnis verbessert werden, da der Anwendung so die Himmelsrichtung, in welche der Nutzer schaut, bekannt ist. Außerdem fehlte es für eine bessere Erkennung an der Information über die Entfernung zwischen Plakatwand und Benutzer.

Informationsanforderung Dem Nutzer wird von der Anwendung die Möglichkeit gegeben Informationen über eine bestimmte Plakatwand anzufordern. Hierzu wurden die entsprechenden Schnittstellen zur Nutzerinteraktion implementiert. Der Nutzer kann die Informationen über die entsprechenden Sprachbefehle oder einen Air-Tap anfordern. Die Spracheingabe ist dabei auch in lauten Umgebungen möglich. Diese wurde in der Nähe einer stark befahrenen Straße getestet. Es war trotz der erhöhten Umgebungslautstärke nicht notwendig lauter zu Sprechen um eine Erkennung der Sprache durch die HoloLens zu ermöglichen. Sowohl das Anfordern der Informationen an der Web API, als auch das Empfangen und Verarbeiten der Antwort funktioniert, eine bestehende Internetverbindung vorausgesetzt.

Informationszugehörigkeit Durch das verwendete Bedienungskonzept der Anwendung wird die Anforderung erfüllt, dass der Nutzer weiß, zu welcher Plakatwand er Informationen angezeigt bekommt. Der Nutzer erhält, bei korrekter Identifizierung, die Daten zu der Plakatwand, welche er beim Ausführen des Air-Taps im Fokus hat. Allerdings kann der Nutzer die Plakatwand aus dem Fokus verlieren was bei diesem ggf. dazu führen kann, dass er nicht mehr weiß, auf welche Plakatwand sich die Informationen beziehen. Dies kann durch einen virtuellen Marker an der Plakatwand verhindert werden, welcher dem Nutzer verdeutlicht auf welche Werbefläche sich die Informationen beziehen. Für diese Lösung muss die Position der Plakatwand im Raum bekannt sein. Dies ist auch hier, aus den zuvor genannten Einschränkungen der Tiefenkamera, nicht sicher zu stellen. Auch an diesem Punkt würde die Anwendung von der Information über die Entfernung zur Plakatwand profitieren.

Informationsanzeige Des Weiteren gibt es zwei Punkte welche Anforderungen an die Anzeige der Informationen stellen. Die erste fordert, dass die Informationen dem Nutzer angezeigt werden. Wenn ein Nutzer Infor-

Abbildung 15: Anzeige von Informationen zur Plakatwand an der Bushaltestelle Winninger Straße in Koblenz

mationen durch einen Air-Tap anfordert und diese von der Web API zurückgeliefert werden, folgt darauf die Aktivierung des Detailfensters und die Anzeige der Daten. Die Daten werden in die definierten Textfelder des Fensters eingetragen und dieses wird, wie in Abbildung 15 zu sehen ist, vor dem Nutzer angezeigt.

Bei der Anzeige von Inhalten auf der HoloLens im Freien, wird sehr schnell deutlich, dass diese nicht dafür ausgelegt ist. Je nach Hintergrund ist es selbst ohne direkte Sonneneinstrahlung schwer bis unmöglich die virtuellen Elemente zu erkennen. Dieses Problem kann zurzeit nicht auf Seiten der Anwendung gelöst werden, da es sich hier um die Helligkeitsgrenze der Projektoren der HoloLens handelt. Eine Möglichkeit diesem Problem entgegenzuwirken ist das Verwenden eines Sonnenlichtfilters, welcher auf die Front der HoloLens geklebt wird. Als Sonnenfilter kann eine Folie verwendet werden, die einen bestimmten Prozentsatz des Sonnenlichts blockiert [Gay16]. Durch den somit abgedunkelten Hintergrund wirkten das Bild der HoloLens im Verhältnis heller und die virtuellen Objekte können besser wahrgenommen werden.

Die zweite Anforderung trägt dafür Sorge, dass der Nutzer durchgehend Kenntnis darüber haben soll, wo sich das Detailfenster befindet. Dies wird mit dem Tagalong-Verhalten sichergestellt. Dadurch kann das Fenster nicht aus dem Sichtfeld des Nutzers verschwinden. Durch die verzögerte Bewegung des Detailfensters fühlt sich der Nutzer von diesem nicht gestört. Außerdem sorgt das Billboard-Verhalten dafür, dass das Detailfenster immer in Richtung des Nutzers ausgerichtet ist.

Detailfenster ausblenden Die Informationen, die der Nutzer angezeigt bekommt, müssen von diesem auch wieder ausgeblendet werden können. Diese Anforderung wird durch einen entsprechenden Button ermöglicht. Dieser kann vom Nutzer mittels Air-Tap betätigt werden, um das Detailfenster zu deaktivieren.

Datenaktualität Der letzte Punkt der Anforderungsliste stellt sicher, dass die Anwendung immer die aktuellen Daten anzeigt. Dies wird in HoloBillboards durch die Client-Server-Architektur der Anwendung realisiert. Die Informationen zu den Plakatwänden können in der Datenbank, auf welche der Server zugreift, jederzeit geändert werden. Wird in der Anwendung nun eine Plakatwand abgerufen, werden die Daten aus der Datenbank geholt und sind dementsprechend jederzeit aktuell. Hiermit geht die Voraussetzung einher, dass die HoloLens eine Verbindung zum Internet herstellen kann.

5.2 Forschungsfrage

Die zentralen Fragen, die die Entwicklung der Anwendung begleitet haben, waren die nach den Möglichkeiten, die die HoloLens mit der WindowsMixedReality Plattform bietet und welche Grenzen dieser Technologie zum jetzigen Zeitpunkt gesetzt sind. Zuerst werden die Möglichkeiten aufgezeigt, welche sich aus dieser Technologie ergeben. Im zweiten Teil des Abschnittes wird auf die Grenzen der Hardware und Plattform eingegangen.

Möglichkeiten Anhand von HoloBillboards lassen sich einige, wenn auch bei weitem nicht alle Möglichkeiten der HoloLens aufzeigen. Grundlegend lässt sich sagen, dass Microsoft bemüht ist, Entwicklern einen leichten Einstieg in die Entwicklung von Anwendungen für die Mixed Reality Plattform zu bieten. Dies zeigt sich z. B. in der Kooperation mit Unity. Durch die Abstraktion der Unity Engine kann der Entwickler die wesentlich komplexere Programmierung mit der DirectX API vermeiden. Darüber hinaus werden dem Programmierer einige grundlegende Arbeiten, wie das Konfigurieren der Kamera, das Erstellen des Cursors und anderer Basiselemente durch das Mixed Reality Toolkit erspart. Zusammen mit den Tutorials wird hier ein sehr einfacher Einstieg in die Entwicklung ermöglicht. Der einfache Einstieg ermöglicht die Umsetzung vieler verschiedener Ideen, was zu einer Bereicherung der Plattform führt. Wie wichtig eine große Entwicklergemeinschaft für eine Plattform ist, kann an den verschiedenen Plattformen für Smartphones betrachtet werden.

Durch die Entwicklung einer Anwendung für die HoloLens ist es möglich einen ersten Einblick in eine neue Geräteklasse zu erhalten. Diese neue

Klasse von Geräten unterscheidet sich auf vielen Ebenen zu konventionellen Computern. Eine davon ist die Bedienung, welche nicht mehr wie heute oft üblich über Maus, Tastatur, Touchscreen oder Gamepad erfolgt. Gleiches gilt für das Erstellen von grafischen Benutzerschnittstellen oder den Umgang mit der Tonausgabe in Anwendungen. Die HoloLens bietet als erstes Gerät dieser Klasse die Möglichkeit, sich mit den neuen Konzepten vertraut zu machen und sich an der Entwicklung der Mixed Reality Plattform zu beteiligen.

Eine weitere Ebenen ist die Art und Weise wie Inhalte dem Nutzer präsentiert werden und welchen Grad an Immersion dieser bei der Nutzung der Anwendung erfährt. Dieser Grad ist bei HoloBillboards, im Vergleich zu anderen Anwendung für die HoloLens, noch relativ gering. Trotzdem ermöglicht die HoloLens der Anwendung eine, bisher nicht möglich gewesene Art der Präsentation von Informationen. Das Fenster mit den Informationen schwebt vor dem Nutzer im Raum und bewegt sich mit ihm durch diesen. Mit den Techniken des Spatial Mappings und Spatial Sound existieren weitere Möglichkeiten die Immersion von Anwendungen zu erhöhen. Hierbei spielt die Interaktion der virtuellen Objekte mit dem realen Raum eine wichtige Rolle. Durch diese erfährt der Nutzer nicht nur eine visuelle Überblendung, sondern bekommt das Gefühl, eines fließenden Übergangs zwischen Realität und Virtualität.

Durch die Entwicklung von HoloBillboards konnte gezeigt werden, dass viele Möglichkeiten existieren, die HoloLens zu erweitern. Diese kann über die Bluetooth-Schnittstelle mit vielen anderen Geräten kommunizieren. Dabei kann es sich, wie in der Anwendung um das Bereitstellen von GPS-Daten handeln um der HoloLens die Möglichkeit der Standortbestimmung zu liefern. Drüber hinaus sind andere Erweiterungen der HoloLens denkbar, da diese mit Bluetooth und WiFi die zwei wichtigsten Wege zur Übertragung von Daten zwischen Geräten unterstützt. Durch letzteres hat die HoloLens außerdem die Möglichkeit, eine Verbindung zum Internet herzustellen. In HoloBillboards wird diese verwendet um Informationen vom Server abzurufen. In Verbindung mit der Thematik des Cloud-Computing, dem Auslagern von aufwändigen Rechenoperationen auf Server, welche über das Internet erreichbar sind, bieten sich Möglichkeiten, aufwändige Berechnungen und große Datenmengen auszulagern.

Grenzen Der zweite Aspekt der Forschungsfrage beschäftigt sich mit den Grenzen der HoloLens. Diese beziehen sich zum einen Teil auf die verbaute bzw. fehlende Hardware. Zum anderen auf die softwareseitigen Einschränkungen der Plattform.

Eine der Grenzen betrifft die Geschwindigkeit in der die Entwicklung von Anwendungen für die HoloLens möglich ist. Hier hat die Entwicklung von HoloBillboards gezeigt, dass diese nur selten die Erwartungen erfüllt.

Ein Grund dafür sind immer wieder auftretende Probleme, die vermutlich aus dem Zusammenspiel der verschiedenen Komponenten resultieren, welche bei der Entwicklung verwendet wurden. Der Unity Editor wird dabei durch das MixedRealityToolkit-Unity erweitert und der Quellcode wird in Visual Studio editiert. Somit sind mindestens drei unterschiedliche Produkte an der Entwicklung einer Anwendung beteiligt. Dazu kommen der Emulator bzw. die HoloLens selber auf welcher die Anwendung ausgeführt wird. Bei der Entwicklung kommt es häufig zu Problemen im Ablauf, welcher die Unity Anwendung auf der HoloLens bereitstellt. Viele dieser Probleme konnten durch erneutes Kompilieren oder einen Neustart sowohl der HoloLens als auch des verwendeten Computers behoben werden. Dieses zeitintensive beheben von temporär auftretenden Problemen erschwert eine genaue zeitliche Planung. Bei HoloBillboards wird die Komplexität durch das Vorhandensein von Plugins und der Android Anwendung weiter erhöht, wodurch auch die Fehleranfälligkeit des Systems erhöht wird.

Eine weitere Einschränkung ist die Verwendung der 32-Bit-Architektur. Diese hat bei der Entwicklung dazu geführt, dass es nicht mit einem vertretbaren Aufwand möglich ist, die OpenCV Bibliothek einzubinden. Mit dem Fehlen dieser Bibliothek gehen der HoloLens viele Möglichkeiten verloren. Sollte diese in einem Projekt doch benötigt werden, muss der Entwickler auf das kostenpflichtiges Plugin aus dem Unity Store zurückgreifen. Dies ist gerade für nicht kommerzielle Anwendungen nicht immer möglich und schränkt hier das kreative Potential der Entwickler ein.

Der eingeschränkte Zugriff auf die Sensoren der HoloLens ist ein weiteres Problem, welches bei der Entwicklung aufgetreten ist. Der fehlende Zugriff auf die IMU hat die Verbesserung des Ergebnisses in starkem Maße eingeschränkt. Auch wenn die HoloLens primär für die Anwendung im Inneren konzipiert ist, gibt es auch hier einige Anwendungen, bei denen ein Zugriff auf das Magnetometer, das Gyroskop oder den Beschleunigungssensor nützlich ist.

Es ist außerdem nicht möglich Objekte wie das Detailfenster am äußeren Sichtbereich des Nutzers zu platzieren, sodass er diese nur noch im Augenwinkel wahrnimmt. Dies liegt am eingeschränkten Sichtfeld der HoloLens. Ein Objekt, welches sich am Rand der Linse der HoloLens befindet, ist in Relation zum Sichtfeld des Benutzers noch sehr weit in der Mitte. Deshalb muss bei der Entwicklung darauf geachtet werden, dass das Sichtfeld der HoloLens nicht mit dem des Nutzers übereinstimmt. Außerdem wird durch die Kanten der Anzeigen das Gefühl der Immersion verringert, da in einem großen Teil des Sichtfeldes des Anwenders keine virtuellen Objekte angezeigt werden.

Die größte Grenze, welche bei der Entwicklung der Anwendung für die HoloLens aufgetreten ist, ist die des Einsatzbereiches. HoloBillboards ist eine Anwendung die ausschließlich für den Einsatz im Freien entwickelt wurde. Die HoloLens dagegen ist ein Gerät, welches primär für den

Einsatz innen konzipiert ist. Dies wird an vielen Punkten deutlich, beginnend mit dem Fehlen eines Moduls zur Übertragung von Daten über Mobilfunk. Dadurch wird vorausgesetzt, dass, bei Verwendung einer Anwendung wie HoloBillboards, bei der eine Internetverbindung erforderlich ist, ein mobiler Wifi Zugangspunkt mitgeführt wird. Auch das Fehlen eines GPS-Moduls wird bei der Entwicklung der Anwendung deutlich. Dieses kann zwar über Bluetooth angebunden werden, erfordert aber auch ein weiteres Gerät, welches der Nutzer mit sich führen muss.

Ein weiterer Aspekt ist die Akkulaufzeit der HoloLens. Diese wird zwar vom Hersteller bei *normaler* Nutzung mit 5,5 Stunden angegeben. Diese wird zum einen durch die Verwendung von WiFi und Bluetooth und zum anderen durch die benötigte Helligkeit der Projektoren verringert. Ein weiteres Problem der Projektoren besteht darin, dass je nach Helligkeit der Umgebung und Farbe des Hintergrundes, selbst die höchste Helligkeitsstufe, nicht für ein erkennbares Bild ausreicht. Diese Problematik kann an sonnenreichen Tagen dazu führen, dass die Nutzung der HoloLens nicht möglich ist. Die Projektoren sind somit in zweierlei Hinsicht eine Schwachstelle, wenn es um den Einsatz im Freien geht.

6 Fazit und Ausblick

Ziel dieser Ausarbeitung, in deren Rahmen eine Anwendung für die HoloLens entwickelt wurde, war es die Grenzen und Möglichkeiten der HoloLens aufzuzeigen. Die Anwendung wurde zur Vermarktung von Plakatwänden der awk AUSSENWERBUNG GmbH entwickelt. Aufgabe der Anwendung ist es, dem Nutzer Informationen zu diesen Großflächen anzuzeigen. Anhand der Entwicklung der Anwendung konnten viele Erkenntnisse über die neue Plattform gesammelt werden, welche bei der Weiterentwicklung der Anwendung und der Entwicklung neuer Anwendungen nützlich sein können.

6.1 Fazit

Durch die Entwicklung einer Anwendung für die HoloLens konnten viele Erkenntnisse über diese neue Plattform gewonnen werden. Es konnte gezeigt werden, dass mittels der verfügbaren Werkzeuge wie Unity und dem Mixed Reality Toolkit ein schneller Einstieg in die Entwicklung für die Mixed Reality Plattform möglich ist. Allerdings wurden durch die Vielzahl der beteiligten Programme die Probleme deutlich, welche die Entwicklung in einigen Aspekten erschweren. Auch wurde gezeigt, dass die HoloLens in manchen Punkten, in denen sie nicht die nötigen Voraussetzungen bietet, erweitert werden kann. Es wurde gezeigt, dass die Bibliothek OpenCV, zum jetzigen Zeitpunkt nicht ohne größeren Aufwand oder ein kostenpflichtiges Plugin auf der HoloLens genutzt werden kann. Außerdem kann das Fehlen von Sensoren und Module teilweise durch Datenübertragung von dritten Geräten über die vorhandenen Schnittstellen wie WiFi und Bluetooth ausgeglichen werden. Darüber hinaus wurde gezeigt, dass mit der HoloLens neue Konzepte der Bedienung, der Anzeige von Informationen und der Interaktion mit dem Nutzer benötigt werden. Diese gilt es weiter zu erforschen und zu verbessern. Im Gesamten wird deutlich, dass die HoloLens gerade für den Einsatz außerhalb von Räumen noch einiges an Weiterentwicklung benötigt. Es gibt sowohl auf der Ebene der Hardware als auch der Software noch einige Punkte, welche dafür verbessert werden müssen. Trotzdem ist die HoloLens, dafür, dass es sich bei ihr um ein Gerät für Entwickler handelt, eine schon sehr weit fortgeschrittene und ausgereifte Technologie, in der ein großes Potential steckt.

Am Beispiel von HoloBillboards wird abschließend deutlich, dass die HoloLens in ihrer jetzigen Form nicht für Anwendungen im Freien geeignet ist. Zwar können viele Probleme und Einschränkungen, welche den Einsatz im Freien erschweren, umgangen werden. Letztendlich sind die Funktionen der HoloLens aber auf geschlossene Räume ausgelegt. Deshalb kann, nach den Erfahrungen aus der Entwicklung von HoloBillboards, nur davon abgeraten werden, Anwendungen für die HoloLens zu entwickeln,

deren Nutzung ausschließlich im Freien möglich ist. Auch wenn dies mit diversen Einschränkungen möglich ist, ist die Nutzung des gesamten Potentials der HoloLens, in ihrer jetzigen Form nur in geschlossenen Räumen möglich. Somit kann abschließend festgehalten werden, dass die HoloLens nicht für die Vermarktung von Werbeträgern im Außenbereich geeignet ist.

6.2 Ausblick

HoloLens und Mixed Reality Plattform Die HoloLens und die Mixed Reality Plattform bedürfen noch einiger Verbesserungen, bis diese für den Endkunden bereit sind. Trotzdem wird das Potential dieser Technologie deutlich. Microsoft hat bereits die nächste Version der HoloLens angekündigt, zu welcher allerdings noch kein Datum für die Veröffentlichung bekannt ist. Es wurde aber bekannt gegeben, dass in der HoloLens ein AI-Chip zur Implementierung von *Deep Nural Networks* verbaut sein wird [Mic17g]. Dieser kann möglicherweise für eine natürliche Spracherkennung oder zur semantischen Analyse von Bildern verwendet werden.

Die HoloLens ist möglicherweise ein ähnlich richtungsweisendes Gerät für den Bereich Mixed Reality, wie das iPhone vor 10 Jahren für den Smartphone-Markt. Auch hier wird das volle Potential der Hardware erst sichtbar, wenn viele Ideen in Form von Anwendungen auf der Plattform umgesetzt werden. Dies wurde von Microsoft erkannt und es wird ein möglichst leichter Einstieg in die Entwicklung ermöglicht. Durch die Erweiterung der Mixed Reality Plattform um Geräte mit undurchsichtigen Display wird der Weg, den Microsoft eingeschlagen hat deutlich. Es wurde mit der Mixed Reality Platform eine gemeinsame Plattform für verschiedenen Arten von Mixed Reality Geräten geschaffen. Hierbei steht die Anwendung im Mittelpunkt und nicht das jeweils verwendete Gerät. Mit dieser Vision versucht Microsoft die strikte Trennung zwischen AR und VR aufzulösen und die Bedeutung der Plattform und der auf ihr laufenden Anwendungen in den Vordergrund zu rücken. Für die Zukunft kann dies bedeuten, dass die selbe Anwendung zusammen mit mehreren Menschen genutzt werde kann unabhängig davon, welches Mixed Reality Gerät sie dazu verwenden.

HoloBillboards Die Anwendung ist in ihrer jetzigen Form nur bedingt einsetzbar. Abgesehen von den zuvor erwähnten Punkten, welche eine Nutzung der HoloLens im Freien erschweren, müssen auch auf Seiten der Anwendung noch weitere Verbesserungen implementiert werden.

Die Implementierung der beschriebenen Ansätze zur Erkennung von Plakatwänden auf Bildern ist hierzu ein erster Schritt, wodurch der Abstand vom Nutzer zu diesen ermittelt werden kann. Durch diese Informationen können gleich mehrere Verbesserungen in der Anwendung erzielt werden. Der wichtigste Punkt ist hierbei eine bessere Identifizierung

der Plakatwände zu erreichen. Durch den Abstand kann der Suchbereich bedeutend verringert werden, was zu besseren Ergebnissen führt. Dieses könnte darüber hinaus verbessert werden, wenn die Himmelsrichtung, in die der Nutzer schaut, bekannt ist. Da dem System nur die Rotation in Relation zur initialen Position der HoloLens bekannt ist und ein Zugriff auf die IMU nicht absehbar ist, wäre ein Workaround notwenig. Der Nutzer könnte sich nach dem Start der Anwendung, mittels des Kompasses auf dem Smartphone in eine vordefinierte Himmelsrichtung drehen und dies der Anwendung bestätigen. Somit wäre eine Kalibrierung der Rotation des Nutzers erfolgt und es könnte damit auf die aktuelle Blickrichtung geschlossen werden, was den Suchbereich erheblich eingrenzen würde.

Unter der Voraussetzung es wird eine Möglichkeit zur Bildverarbeitung auf der HoloLens gefunden, könnte mit einer kontinuierlichen Analyse der Bilder der Kamera eine automatische Erkennung von Plakatwänden erfolgen. Am Standort der erkannten Plakatwand kann so anhand der Blickrichtung des Benutzers und die Entfernung zur Plakatwand ein Indikator angezeigt werden. Dieser kann als Button, zur Anzeige der Informationen verwendet werden.

Die Anzeige der Informationen könnte ebenfalls einige Verbesserungen erhalten. So ist es möglich dem Nutzer die Entscheidung zu Überlassen ob er das Detailfenster an einem fixen Ort platzieren oder es als Tagalong mit sich führen möchte. Auch die Anzeige von mehreren Detailfenstern ist eine Möglichkeit. Zur besseren Orientierung kann eine Karte implementiert werden, auf der sowohl der Standort des Benutzers als auch die Standorte der Plakatwände zu sehen sind.

Als eine weitere Funktion welche im Zuge der Erkennung von Plakatwänden realisiert werden kann, ist das Überblenden einer Plakatwand mit einem eigenen Motiv. Hierzu kann der Anwender sein eigenes Motiv über eine zu definierende Schnittstelle in das System laden. Dieses würde dann die Möglichkeit bieten, dass sich der Nutzer sein Plakat virtuell auf die Plakatwand projizieren lässt um diese vor einem potentiellen Druck in der realen Umgebung zu sehen. Durch die Verankerung im Raum hat der Nutzer die Möglichkeit, das Plakat von verschiedenen Positionen und Blickwinkeln zu betrachten.

Eine Form der Interaktion mit dem Nutzer, welche bisher keinen Einzug Anwendung erhalten hat, ist die Wiedergabe von Tönen. Diese kann dazu verwendet werden dem Nutzer ein akustisches Feedback zu geben, wenn dieser mit dem Air-Tap eine Aktion ausgelöst hat oder ein Schlüsselwort erkannt wurde. Außerdem kann das Erkennen einer Plakatwand, zusammen mit der Einblendung eines Indikators, akustisch signalisiert werden.

Die vielen Verbesserungsmöglichkeiten zeigen, welches Potential in der HoloLens steck. Zwar sind zurzeit für viele dieser Verbesserungen noch relativ umständliche Lösungen nötig. Eventuell wird eine der nächsten Ge-

nerationen der HoloLens besser für den Einsatz im Freien geeignet sein und die vorgeschlagenen Verbesserungen für HoloBillboards könnten somit umgesetzt werden.

Literatur

[Ady+07] Atul Adya et al. „Anatomy of the ADO.NET entity framework“. In: *the 2007 ACM SIGMOD international conference*. New York, New York, USA: ACM Press, 2007, S. 877–888. ISBN: 9781595936868. DOI: 10.1145/1247480.1247580.

[Ang16] Chris Angelini. *Microsoft HoloLens: HPU Architecture, Detailed*. Aug. 2016. URL: http://www.tomshardware.com/news/microsoft-hololens-hpu-architecure-28nm,32586.html (besucht am 28. 08. 2017).

[awk17a] awk. *Großfläche Standort Straße*. 2017. URL: http://www.awk.de/produkte/grossflaechen-strasse.html (besucht am 21. 09. 2017).

[awk17b] awk. *Home*. German. 2017. URL: http://www.awk.de/home.html (besucht am 05. 09. 2017).

[Bal11] Heide Balzert. *Basiswissen Web-Programmierung*. XHTML, CSS, JavaScript, XML, PHP, JSP, ASP.NET, Ajax. W3L-Verlag, 2011. ISBN: 9783868340334.

[BC03] G C Burdea und P Coiffet. *Virtual Reality Technology*. Academic Search Complete. Wiley, 2003. ISBN: 9780471360896.

[Cla17] Clayton Industires. *Http Client*. 2017. URL: https://www.claytoninds.com/#httpclient (besucht am 14. 09. 2017).

[CMA07] Pablo Castro, Sergey Melnik und Atul Adya. „ADO.NET entity framework - raising the level of abstraction in data programming.“ In: *SIGMOD Conference* (2007), S. 1070–1072. DOI: 10.1145/1247480.1247609.

[Col16] Seth Colaner. *What's Inside Microsoft's HoloLens And How It Works*. Aug. 2016. URL: http://www.tomshardware.com/news/microsoft-hololens-components-hpu-28nm,32546.html (besucht am 27. 08. 2017).

[Cop17] Adrien Coppens. „Merging real and virtual worlds: An analysis of the state of the art and practical evaluation of Microsoft Hololens“. In: (Juni 2017). arXiv: 1706.08096 [abs/1706.08096].

[CSD93] Carolina Cruz-Neira, Daniel J Sandin und Thomas A DeFanti. „Surround-screen projection-based virtual reality“. In: *the 20th annual conference*. New York, New York, USA: ACM Press, 1993, S. 135–142. ISBN: 0897916018. DOI: 10.1145/166117.166134.

[Cue17] Eduardo Cuervo. „BEYOND REALITY: Head-Mounted Displays for Mobile Systems Researchers“. English. In: *GetMobile: Mobile Computing and Communications* 21.2 (Aug. 2017), S. 9–15. DOI: 10.1145/3131214.3131218.

[Des+14] Parth Rajesh Desai et al. „A Review Paper on Oculus Rift-A Virtual Reality Headset“. In: (Aug. 2014). arXiv: 1408.1173 [abs/1408.1173].

[Doc15] Doc-Ok. *On the road for VR: Microsoft HoloLens at Build 2015, San Francisco*. Mai 2015. URL: http://doc-ok.org/?p=1223 (besucht am 21. 09. 2017).

[Dör+16] Ralf Dörner et al. „Virtual Reality und Augmented Reality (VR/AR) - Auf dem Weg von der Nische zum Massenmarkt.“ In: *Informatik Spektrum* (2016).

[Eno17] EnoxSoftware. *HoloLens with OpenCV for Unity Example*. Aug. 2017. URL: https://github.com/EnoxSoftware/HoloLensWithOpenCVForUnityEx (besucht am 25. 09. 2017).

[Eri16] Liv Erickson. *Unity 5.5 and the HoloToolkit – Changes in Input when Developing HoloLens Apps*. Dez. 2016. URL: https://livierickson.com/blog/unity-5-5-and-the-holotoolkit-changes-in-input-when-developing-hololens-apps/ (besucht am 13. 09. 2017).

[Fin] FinalWire Ltd. *Downloads | AIDA64*. URL: https://www.aida64.com/downloads (besucht am 28. 08. 2017).

[Gay16] Nathan Gaydhani. *Project: Using the Microsoft HoloLens outdoors with a visor.* Dez. 2016. URL: http://digitalreality.guru/2016/12/project-using-the-microsoft-hololens-outdoors-with-a-visor/ (besucht am 21. 09. 2017).

[Goo] Google. *Google Glass Navigation*. URL: https://support.google.com/glass/answer/3079688?hl=en (besucht am 06. 09. 2017).

[Goo17] Google. *Location | Android Developers*. 2017. URL: https://developer.android.com/reference/android/location/Location.html#getAccuracy() (besucht am 11. 09. 2017).

[Gut16] Karl Guttag. *AR/MR Combiners Part 2 – Hololens*. Okt. 2016. URL: http://www.kguttag.com/2016/10/27/armr-combiners-part-2-hololens/ (besucht am 24. 08. 2017).

[HFN11] Olivier Hugues, Philippe Fuchs und Olivier Nannipieri. „New Augmented Reality Taxonomy - Technologies and Features of Augmented Environment.“ In: *Handbook of Augmented Reality* Chapter 2 (2011), S. 47–63. DOI: 10.1007/978-1-4614-0064-6_2.

[Ken+17] John Kennedy et al. *What's a Universal Windows Platform (UWP) app?* März 2017. URL: https://docs.microsoft.com/en-us/windows/uwp/get-started/whats-a-uwp (besucht am 01. 09. 2017).

[Kic12] Kickstarter. *Oculus Rift: Step Into the Game*. 2012. URL: https://www.kickstarter.com/projects/1523379957/oculus-rift-step-into-the-game (besucht am 23. 08. 2017).

[Kol16] Panagiotis Kolokythas. *Hololens: Microsoft lüftet HPU-Geheimnis*. Aug. 2016. URL: https://www.pcwelt.de/news/Hololens-Microsoft-lueftet-HPU-Geheimnis-10028103.html (besucht am 28. 08. 2017).

[Kri12] Madhav Krishna. *Billboard Detection*. Juni 2012. URL: http://tech.adstruc.com/post/28306434540/billboard-detection (besucht am 11. 09. 2017).

[Ler10] Julia Lerman. „Introducing the ADO.NET Entity Framework". In: *Programming Entity Framework*. O'Reilly, 2010, S. 1–18. ISBN: 978-0596520281.

[MBB06] Erik Meijer, Brian Beckman und Gavin Bierman. „LINQ: reconciling object, relations and XML in the .NET framework". In: *SIGMOD 2006* (Juni 2006), S. 706–706. DOI: 10.1145/1142473.1142552.

[Mica] Microsoft. *HoloLens Sensorbar*. URL: https://www.microsoft.com/en-us/hololens/hardware (besucht am 24. 08. 2017).

[Micb] Microsoft. *HoloLensOptics*. URL: https://www.microsoft.com/en-us/hololens/hardware (besucht am 24. 08. 2017).

[Micc] Microsoft. *Microsoft HoloLens*. English. URL: https://www.microsoft.com/en-us/hololens (besucht am 22. 08. 2017).

[Micd] Microsoft. *Web API*. URL: https://www.asp.net/web-api (besucht am 30. 08. 2017).

[Mic16a] Microsoft. *Buy Holographic Remoting Player - Microsoft Store*. 2016. URL: https://www.microsoft.com/en-us/store/p/holographic-remoting-player/9nblggh4sv40 (besucht am 21. 09. 2017).

[Mic16b] Microsoft. *MixedRealityToolkit-Unity*. Nov. 2016. URL: https://github.com/Microsoft/MixedRealityToolkit-Unity (besucht am 30. 08. 2017).

[Mic17a] Microsoft. *Coordinate systems*. 2017. URL: https://developer.microsoft.com/en-us/windows/mixed-reality/coordinate_systems (besucht am 11. 09. 2017).

[Mic17b] Microsoft. *Gesture design*. 2017. URL: https://developer.microsoft.com/en-us/windows/mixed-reality/gesture_design (besucht am 07. 09. 2017).

[Mic17c] Microsoft. *Hologram*. 2017. URL: https://developer.microsoft.com/de-de/windows/mixed-reality/hologram (besucht am 05. 09. 2017).

[Mic17d] Microsoft. *Locatable camera in Unity*. 2017. URL: https://developer.microsoft.com/en-us/windows/mixed-reality/locatable_camera_in_unity (besucht am 21. 09. 2017).

[Mic17e] Microsoft. *Mixed Reality Academy*. 2017. URL: https://developer.microsoft.com/en-us/windows/mixed-reality/academy (besucht am 11. 09. 2017).

[Mic17f] Microsoft. *RoboRaid*. 2017. URL: https://www.microsoft.com/de-de/hololens/apps/roboraid (besucht am 11. 09. 2017).

[Mic17g] Microsoft. *Second version of HoloLens HPU will incorporate AI coprocessor for implementing DNNs - Microsoft Research*. Juli 2017. URL: https://www.microsoft.com/en-us/research/blog/second-version-hololens-hpu-will-incorporate-ai-coprocessor-implementing-dnns/ (besucht am 20. 09. 2017).

[Mic17h] Microsoft. *Spatial mapping*. 2017. URL: https://developer.microsoft.com/en-us/windows/mixed-reality/spatial_mapping (besucht am 08. 09. 2017).

[Mic17i] Microsoft. *Text in Unity*. 2017. URL: https://developer.microsoft.com/en-us/windows/mixed-reality/text_in_unity (besucht am 21. 09. 2017).

[Mic17j] Microsoft. *Unity development overview*. 2017. URL: https://developer.microsoft.com/en-us/windows/mixed-reality/unity_development_overview (besucht am 11. 09. 2017).

[Mil+95] Paul Milgram et al. „Augmented reality: a class of displays on the reality-virtuality continuum“. In: *Photonics for Industrial Applications*. Hrsg. von Hari Das. SPIE, Dez. 1995, S. 282–292. DOI: 10.1117/12.197321.

[Ocu17] Oculus VR. *Oculus*. 2017. URL: https://www.oculus.com/ (besucht am 22. 08. 2017).

[Osl17] Damian Oslebo. *GPS on the Microsoft Hololens*. Jan. 2017. URL: https://www.gamedev.net/articles/programming/general-and-gameplay-programming/gps-on-the-microsoft-hololens-r4497/ (besucht am 08. 09. 2017).

[SK17] Dirk Srocke und Florian Karlstetter. *Was ist eine REST API?* Juni 2017. URL: http://www.cloudcomputing-insider.de/was-ist-eine-rest-api-a-611116/ (besucht am 26. 09. 2017).

[Sur17] Surur. *Microsoft renames Windows Holographic to Windows Mixed Reality*. März 2017. URL: https://mspoweruser.com/microsoft-renames-windows-holographic-windows-mixed-reality/ (besucht am 05. 09. 2017).

[Tay16a] Allen G Taylor. „Chapter 1: What Is the Microsoft HoloLens?“ In: *Develop Microsoft HoloLens Apps Now*. Berkeley, CA: Apress, 2016, S. 3–8. ISBN: 978-1-4842-2201-0.

[Tay16b] Allen G Taylor. „Chapter 13: HoloLens Hardware“. In: *Develop Microsoft HoloLens Apps Now*. Berkeley, CA: Apress, 2016, S. 153–159. ISBN: 978-1-4842-2201-0.

[Tay16c] Allen G Taylor. „Chapter 9: Developing with Unity and Visual Studio“. In: *Develop Microsoft HoloLens Apps Now*. Berkeley, CA: Apress, 2016, S. 75–90. ISBN: 978-1-4842-2201-0.

[Uni] Unity Technologies. *Unity - Microsoft Windows - HoloLens*. URL: https://unity3d.com/de/partners/microsoft/hololens (besucht am 30. 08. 2017).

[Uni17] Unity Technologies. *Unity - Manual: Universal Windows Platform: Plugins on .NET Scripting Backend*. Mai 2017. URL: https://docs.unity3d.com/Manual/windowsstore-plugins.html (besucht am 21. 09. 2017).

[Was] Emil Wasilewski. *OpenCVAndUnity*. URL: https://github.com/limered/OpenCVAndUnity (besucht am 18. 09. 2017).

[Whi+17] Tyler Whitney et al. *Intro to the Universal Windows Platform*. Feb. 2017. URL: https://docs.microsoft.com/en-us/windows/uwp/get-started/universal-application-platform-guide (besucht am 01. 09. 2017).