Evaluation aktueller, markerloser Augmented Reality-Technologien für mobile Endgeräte mit beispielhafter Implementierung

Bachelorthesis an der Hochschule Offenburg

vorgelegt von Benjamin Adolph, Matrikelnummer 180129
im Studiengang Medien und Informationswesen
an der Fakultät Medien und Informationswesen
zur Erlangung des akademischen Grades eines Bachelor of Science

Betreuer:
Hochschule Offenburg - Prof. Dr. Dipl.-Ing. Roland Riempp
SYNDIKAT 7 GmbH - Dr.-Ing. Pascal Giessler

18. August 2019

Eidesstattliche Erklärung

Hiermit versichere ich, dass die vorliegende Arbeit mit dem Thema

Evaluation aktueller, markerloser Augmented Reality-Technologien für mobile Endgeräte mit beispielhafter Implementierung

von mir selbstständig und ohne unerlaubte fremde Hilfe angefertigt worden ist. Sämtliche Stellen der Arbeit, die im Wortlaut oder dem Sinn nach anderen gedruckten oder im Internet verfügbaren Werken entnommen sind, habe ich durch genaue Quellenangaben kenntlich gemacht. Die Arbeit lag in gleicher oder ähnlicher Fassung noch keiner Prüfungsbehörde vor und wurde bisher nicht veröffentlicht.

Offenburg, 18. August 2019

_____

Benjamin Adolph

Danksagung

An dieser Stelle möchte ich mich bei Prof. Dr. Dipl.-Ing. (FH) Roland Riempp und Dr.-Ing. Pascal Giessler für die Betreuung und tatkräftige Unterstützung bedanken. Ihre konstruktive Kritik und hilfreichen Anregungen haben maßgeblich dazu beigetragen, dass die Bachelorarbeit in dieser Form vorliegt.

Die beiden Geschäftsführer der SYNDIKAT 7 GmbH, Mario Beiser und Timo Fink, unterstützten mich in jeder denkbaren Form. Beispielsweise stellten sie den Kontakt zur Five-Konzept GmbH & Co. KG her und ermöglichten mir dadurch die Umsetzung einer praxisnahen Beispielapplikation.

Auch meine Arbeitskollegen Anja Schwendemann, Ronja Lindinger, Pascal Burkhardt und Aljoscha Weber standen mir in meiner Zeit bei SYNDIKAT 7 immer mit Rat und Tat zur Seite.

Abschließend möchte ich meinen Eltern danken, die mir mit viel Geduld und Hilfsbereitschaft zur Seite standen. Die zahlreichen interessanten Ideen, Verbesserungsvorschläge und das Korrekturlesen haben mir bei der Erstellung meiner Thesis sehr weitergeholfen.

Offenburg, 18. August 2019

Abstract

Diese Bachelorarbeit befasst sich mit der Evaluation von aktuellen, markerlosen Augmented Reality-Technologien für mobile Endgeräte. Zu Beginn werden in einem theoretischen Teil die wichtigsten Begriffe und Technologien definiert und eingeordnet.

Das nächste Kapitel gibt einen Überblick über die Möglichkeiten zur Erstellung von AR-Applikationen. Der aktuelle Stand bei markerlosen AR-Technologien wird anhand einer vorab definierten Zielsetzung analysiert und bewertet.

Nach der Bewertung folgt eine genauere Betrachtung der am besten geeigneten Technologie. Die Untersuchung der Software von verschiedenen Herstellern findet nach ausgewählten Kriterien statt. Besonderes Augenmerk liegt dabei auf dem Funktionsumfang und der plattformübergreifenden Kompatibilität. Evaluierung und Test der drei besten Softwarepakete anhand einer minimalistischen praktischen Umsetzung runden die Bewertung ab.

Im letzten Kapitel soll eine Beispielapplikation für die Five-Konzept GmbH & Co. KG erstellt werden. Mit einem der besten drei Softwarepakete erfolgt die Implementierung einer kleinen, betriebssystemübergreifenden AR-Applikation zur markerlosen Platzierung von 3D-Objekten in der Umgebung des Nutzers.

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

“AR is going to take a while, because there are some really hard technology challenges there. But it will happen, it will happen in a big way, and we will wonder when it does, how we ever lived without it. Like we wonder how we lived without our phone today.” - Tim Cook, 2016 [1]

Dieses Zitat von Apple CEO Tim Cook beschreibt leider auch im Jahr 2019 noch gut den Stand der Dinge im Bereich Augmented Reality (AR). Auch wenn AR schon im Jahr 1965 von Ivan Sutherland im Essay „The Ultimate Display“ das erste Mal beschrieben wurde, ist die Entwicklung in den darauffolgenden Jahren jedoch eher langsam vorangeschritten. Erst in den letzten 2-3 Jahren hat sich ein Markt für AR-Applikationen gebildet. Dadurch ist AR in Form von Applikationen wie Pokémon Go, Ikea Place oder Snapchat schon im Alltag einiger Menschen angekommen, die breite Masse nutzt die Technologie aber noch nicht. Laut einer Prognose der International Data Corporation (IDC) ändert sich dies aber in den nächsten Jahren drastisch. Bei den Einnahmen im Bereich Augmented Reality wird von IDC eine Steigrung von 0,2 Milliarden US-Dollar im Jahr 2016 auf 48,7 Milliarden US-Dollar im Jahr 2021 prognostiziert. Ob sich diese Vorhersage bewahrheitet, bleibt abzuwarten. Die vielfältigen Einsatzbereiche von Spielen bis hin zur Gesundheitsversorgung können diesen großen Sprung aber durchaus möglich machen. [2], [3]

Unterstützt wird diese Entwicklung von der immer besser werdenden Leistung der Smartphones und Augmented-Reality-Brillen. Außerdem treiben große Konzerne wie Google, Apple oder Microsoft die Entwicklung von AR maßgeblich voran. Insbesondere im Bereich der markerlosen AR-Technologien ist der Fortschritt enorm. Bei Smartphones und Augmented-Reality-Brillen ist diese Entwicklung sehr interessant, da sie gegenüber markerbasierten Technologien ganz neue Anwendungsfälle ermöglichen. Aktuell taucht auf dem Markt deshalb regelmäßig neue Software auf. Auch die bestehenden Produkte werden stetig weiterentwickelt. Durch diese Schnelllebigkeit kann man als Entwickler leicht den Überblick über das aktuelle Angebot verlieren. [4], [5]

1.2 Zielsetzung

Um der oben genannten Entwicklung entgegen zu wirken, ist das übergeordnete Ziel dieser Arbeit die Evaluation von aktuellen, markerlosen AR-Technologien. Es soll die beste Möglichkeit zur Erstellung einer betriebssystemübergreifenden AR-Applikation gefunden und anhand eines MVPs veranschaulicht werden.

Um dieses Ziel zu erreichen, müssen folgende Fragen beantwortet werden:

- Welche Technologien ermöglichen aktuell die Erstellung von markerlosen AR-Anwendungen für mobile Endgeräte?
- Welche Software mit dieser Technologie bietet das beste Leistungspaket?

Da es eine sehr große Auswahl an Software gibt, dienen die beiden nachfolgenden Kriterien zur Vorselektion:

- Tracking von natürlichen Merkmalen (NFT) und simultane Lokalisierung und Kartierung (SLAM)
- Die Software soll mit einer Code-Basis auf unterschiedlichen Betriebssystemen lauffähig sein.

1.3 Aufbau der Bachelorarbeit

In der Einleitung wurden Motivation und Zielsetzung der Thesis beschrieben. Anschließend werden Informationen zur Geschichte und den Anwendungsfeldern von Augmented Reality zur Verfügung gestellt.

Kapitel 3 umfasst die theoretischen Grundlagen zu Webtechnologien und Augmented Reality. Der Abschnitt Webtechnologien beschäftigt sich mit den Definitionen zu JavaScript Frameworks und Mobile Application Development Frameworks. Zu jeder Webtechnologie werden außerdem die bekanntesten drei Frameworks vorgestellt. Der nächste Abschnitt erklärt die wichtigsten Begriffe im Bereich Augmented Reality. Im Anschluss werden mit Tracking, Registrierung und Darstellung die wichtigsten Punkte der Funktionsweise einer AR-Applikation erläutert. Abschließend gibt es eine kurze Zusammenfassung über die Interaktionsmöglichkeiten mit einer AR-Applikation.

Kapitel 4 ermöglicht dem Leser einen umfassenden Überblick über den aktuellen Stand bei markerlosen AR-Technologien für mobile Endgeräte. Native, webbasierte und betriebssystemübergreifende Technologien werden theoretisch, die beiden letzteren auch praktisch untersucht. Der Test von verschiedenen Beispielanwendungen soll dabei helfen, die theoretisch aufgestellte Bewertung zu unterstützen. Native Technologien werden nur der Vollständigkeit halber erwähnt, da sie die Basis für einige der verwendeten Softwares bilden.

Eine genauere Betrachtung der am besten geeigneten Technologie findet in Kapitel 5 statt. Die Verwendung des Business Readiness Rating Models (BRR) ermöglicht die einheitliche Bewertung der Produkte von verschiedenen Softwareherstellern. Die Gewichtung der Kategorien richtet sich dabei nach ihrer Wichtigkeit für die festgelegte Zielsetzung. Die Evaluation bietet einen detailreichen Überblick über Funk-

tion, Entwicklungsplattformen, Ausgabesysteme, Dokumentation, Support, Community und Benutzerfreundlichkeit. Jede Kategorie ist in verschiedene Metriken unterteilt, aus denen sich die Bewertung für die Kategorie zusammensetzt. Abschließend werden die Bewertungen des Produkts miteinander verrechnet. Das Ergebnis bildet den Business Readiness Rating Score der jeweiligen Software.

Anhand des BRR-Scores werden die drei besten Softwares bestimmt. Die praktische Umsetzung eines minimalistischen Anwendungsfalls mit allen drei Varianten findet in Kapitel 6 statt. Die Testkriterien für die Software sind Installation, Implementierung und Anwendung der fertigen Applikation.

In Kapitel 7 wird neben dem Lösungsansatz sowohl die Konzeption als auch die Umsetzung der Beispielapplikation für die Five-Konzept GmbH & Co. KG beschrieben. Das Ergebnis ist eine betriebssystemübergreifende Applikation, welche auf iOS und Android genutzt werden kann. Sie ermöglicht dem Nutzer die markerlose Platzierung von 3D-Objekten in seiner Umgebung.

In Kapitel 8 wird das Fazit gezogen und ein Ausblick auf die weitere Entwicklung von Augmented Reality gegeben.

2 Geschichte

In der Literatur wird meist das „Sensorama“ von Morton Heilig aus dem Jahr 1962 als erste Form von Augmented Reality genannt. Der Filmemacher kombinierte 3D-Filme mit Stereo-Sound, Vibrationen und verschiedenen Aromen in einem Simulator, um den Nutzer in eine virtuelle Welt zu entführen. [6], [7]

Erst ab dem 20. Jahrhundert ist durch die gesteigerte Rechenleistung von Computern das Generieren und Abspielen von dreidimensionalen Computergrafiken in Echtzeit möglich. Zum ersten Mal konnten so realitätsferne, virtuelle Objekte und Welten erstellt werden. Die ersten Versuche, computergenerierte Bilder der Realität zu überlagern, fanden in den 1960er Jahren statt. Ivan Sutherland kann als einer der Pioniere in Augmented und Virtual Reality bezeichnet werden. [8], [9]

Im Essay „The Ultimate Display“ aus dem Jahr 1965 beschreibt Ivan Sutherland mit dem folgenden Zitat das erste Mal Augmented Reality, ohne es so zu nennen. Der Begriff wird erst später geprägt.

„The user of one of today’s visual displays can easily make solid objects transparent - he can see through matter!“ [10]

Abb. 1 zeigt das erste Head-Mounted-Display (HDM) aus dem Jahr 1968. Die Konstruktion ermöglicht die Bewegungsverfolgung des Kopfs und das Einblenden von Informationen ins Blickfeld des Benutzers.

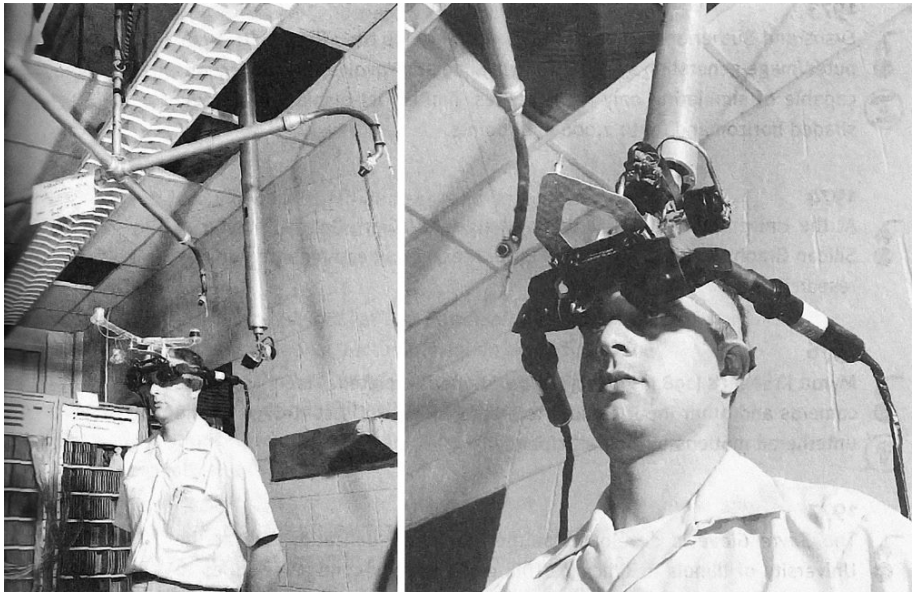

Abb. 1: *Erstes HDM von Ivan Sutherland*

Der Begriff Augmented Reality taucht das erste Mal im Jahr 1992 in einer Arbeit von Thomas P. Caudell und David W. Mizell auf. Im gleichen Jahr entwickelt Louis Rosenberg in den USAF Armstrong Labs das erste immersive Augmented Reality-System zur Unterstützung von Benutzern bei realen und physischen Aufgaben. [11]

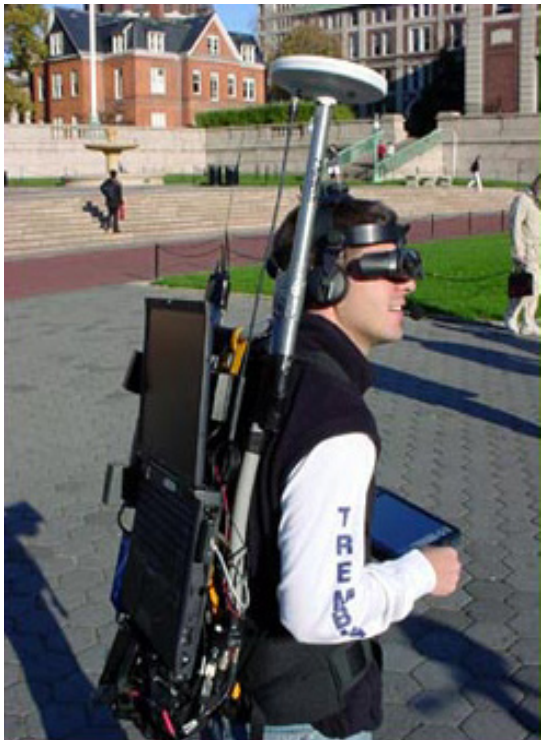

Abb. 2: *Outdoor-AR-System im Einsatz*

1994 definiert Paul Milgram das Reality-Virtuality-Kontinuum, dargestellt in Abb. 9. Drei Jahre später stellt Ronald T. Azuma die allgemein anerkannte Definition von AR auf, siehe Abschnitt 3.2.1. [12], [13]

Abb. 2 zeigt das erste Outdoor-AR-System, welches 1997 von Steve Feiner an der Columbia University vorgestellt wurde. Es benutzt ein HDM, das Global Positioning System (GPS) zur Positionsbestimmung und eine Orientierungsverfolgung. Das Gerät ermöglicht dem Nutzer eine AR-Campus-Tour an der Columbia University. [9]

Im Jahr 1999 wurde ARToolKit von Hirokazu Kato entwickelt. Das Projekt ist Open-Source und hat es vielen Entwicklern das erste Mal ermöglicht, computergenerierte Inhalte mit der realen Welt zu kombinieren. [8]

2008 stellte Wikitude den ersten AR-Browser für das Smartphone zur Verfügung. [7]

Google stellte 2012 das Konzept für die Augmented-Reality-Brille Google Glass vor. Das Projekt wurde jedoch 2015 aufgrund der schlechten Annahme wegen Datenschutzbedenken der Nutzer eingestellt. [7]

2015 kündigt Google eine AR-Plattform für Mobiltelefone namens Projekt Tango an. Auch Microsoft steigt in Form der Hololens in das AR-Geschäft ein. 2017 präsentiert Apple mit ARKit ein Software Development Kit (SDK) für iPhone und iPad. Google zieht daraufhin nach und ersetzte Tango durch das SDK ARCore. Es besitzt nahezu die gleichen Fähigkeiten wie das Äquivalent von Apple.[7]

Heutzutage wird Augmented Reality sehr vielfältig genutzt. Wichtige Einsatzfelder sind das Militär, Fernsehen, Architektur, Medizin, Navigation und Tourismus, Design, Spiele, Werbung, Lehre und Archäologie. Durch die schnelle Entwicklung werden in den nächsten Jahren weitere Bereiche hinzukommen. [7], [9], [14], [15]

3 Theoretische Grundlagen

3.1 Webtechnologien

3.1.1 JavaScript Frameworks

Der Begriff Framework wird auf techterms.com folgendermaßen definiert:

„A framework, or software framework, is a platform for developing software applications. It provides a foundation on which software developers can build programs for a specific platform. For example, a framework may include predefined classes and functions that can be used to process input, manage hardware devices, and interact with system software. This streamlines the development process since programmers don't need to reinvent the wheel each time they develop a new application. A framework is similar to an application programming interface (API), though technically a framework includes an API. As the name suggests, a framework serves as a foundation for programming, while an API provides access to the elements supported by the framework. A framework may also include code libraries, a compiler, and other programs used in the software development process." [16]

Zusammengefasst bieten JavaScript Frameworks dem Entwickler ein Grundgerüst für die gängigsten Funktionen in JavaScript. Die Entwicklung von grundlegenden Funktionalitäten, wie z. B. für die Verwaltung von Events oder Modulen ist somit nicht nötig. Dadurch steht die eigentliche Umsetzung des Projekts im Vordergrund. Im Folgenden sollen JavaScript Frameworks für die Erstellung von Nutzeroberflächen vorgestellt werden. Aktuell sind in diesem Bereich React, Angular und Vue am populärsten. [17], [18]

React

Auf der offiziellen Webseite „https://reactjs.org/" wird React als JavaScript-Bibliothek zum Erstellen von Nutzeroberflächen definiert. Facebook entwickelt React hauptsächlich zur eigenen Verwendung, stellt es aber unter der MIT-Lizenz frei für die Community zur Verfügung. Die Vorteile des Frameworks sind ein einfaches Management und eine gute Skalierbarkeit von komplexen Anwendungen. [19]

Die wichtigste Funktionalität bei React ist das Virtual Document Object Model (VDOM). Es ist eine Kopie des tatsächlichen Document Object Models (DOM). Das DOM ist die Spezifikation der Schnittstelle zwischen HTML und JavaScript. Die Darstellung aller HTML-Dokumente erfolgt in einer Baumstruktur. Die Knoten in diesem Baum stehen für einzelne HTML-Objekte, wie z. B. eine Überschrift. Das VDOM ist auf ein Minimum an Daten beschränkt, dadurch ist die Durchführung von Änderungen schnell möglich.

Sobald React dieses verändert hat, vergleicht ein Algorithmus beide Versionen. Die Bündelung aller Änderungen im VDOM ermöglicht es, diese in einem Schritt im DOM anzupassen (siehe Abb. 3). [20]

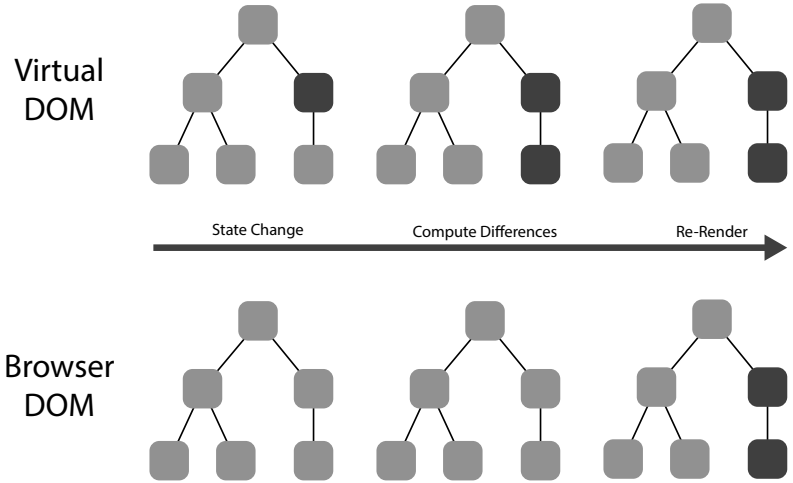

Abb. 3: *Virtual DOM*

Eine weitere Besonderheit gegenüber normalen Webapplikationen ist, dass sich die Benutzeroberfläche von React aus einzelnen Komponenten zusammensetzt. In der offiziellen Dokumentation von React werden Komponenten wie folgt beschrieben:

„Components let you split the UI into independent, reusable pieces, and think about each piece in isolation." [21]

Komponenten sind JavaScript-Klassen oder -Funktionen. Optional kann die Übergabe weiterer Parameter, wie z. B. ***props*** erfolgen. Diese Komponenten geben ein React-Element zurück, welche das Aussehen von einem Teil der Nutzeroberfläche bestimmt. Komponenten können mit unterschiedlichen Inhalten befüllt und flexibel wiederverwendet werden. Das sorgt dafür, dass React-Applikationen beliebig skalierbar und so auch für sehr große Projekte geeignet sind. [20], [21]

Weiterhin unterscheidet man zwischen Komponenten mit Zustand (***stateful***) und ohne Zustand (***stateless***). Die Zustände verwaltet die Komponente selbst. Für mehrere Zustände können unterschiedliche Ansichten erstellt werden. Ändert sich der Zustand einer Komponente, aktualisiert React die entsprechende Ansicht automatisch. Dadurch ist React sehr gut für dynamische Inhalte geeignet. [20], [22]

React wird in JSX geschrieben. JSX ist eine Syntaxerweiterung für JavaScript, welche sich durch einen HTML-ähnlichen Syntax auszeichnet. Diese wird beim sogenannten Build der Applikation wieder in JavaScript übersetzt. Alternativ ist es möglich, die Applikation direkt in JavaScript zu entwickeln. [20], [23]

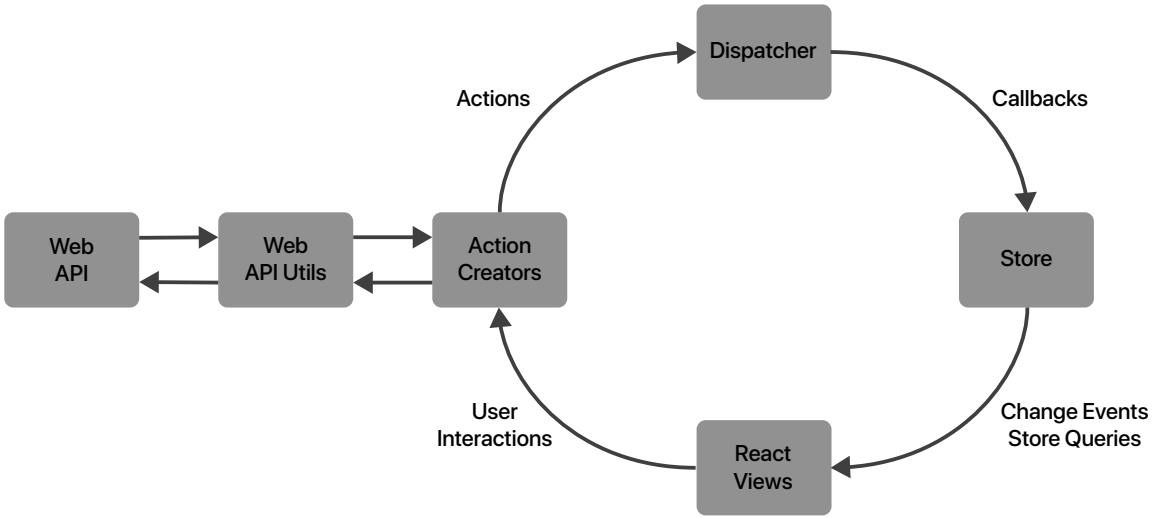

Abb. 4: *FLUX Architektur*

Wichtig ist, dass React keine Model-View-Controller Architektur hat. Facebook benutzt für React die eigens entwickelte FLUX Architektur (siehe Abb. 4). [24]

Nach einer Aktion vom Nutzer erfolgt der Aufruf der Action Creators aus der React View. Die Aktionen werden anschließend an den Dispatcher weitergegeben. Der Dispatcher kann immer nur eine Aktion verarbeiten. Diese verteilt er an sogenannte Stores, welche die Applikationslogik verwalten. Bekommt ein Store eine Aktion zugewiesen, kann diese den Store verändern. Eine Veränderung im Store bewirkt die Aktualisierung der React Views. [24]

Der wesentliche Unterschied zur MVC Architektur ist die Kommunikation zwischen Komponenten. Beim der MVC Architektur ist diese bidirektional, bei der FLUX Architektur unidirektional. Der größte Vorteil hierbei ist, dass die Anwendung so viel leichter auf Fehler zu untersuchen ist. [20], [24]

Vue

Vue ist ein progressives JavaScript-Framework. Progressiv bedeutet hierbei, dass Vue auch nur in bestimmte Teile einer Applikation eingebaut werden kann. Wie React nutzt auch Vue ein VDOM und unterteilt die Nutzeroberfläche in einzelne Komponenten. Zur Implementierung der Komponenten wird entweder JSX oder HTML, CSS und JavaScript genutzt. Der Vorteil bei letzterem ist, dass sich Webentwickler nicht erst in den JSX-Syntax einarbeiten müssen. Sie können direkt mit dem Erstellen einer Applikation beginnen. [25], [26]

Vue basiert auf dem sogenannten Model-View-View-Model (MVVM) und verwendet eine bidirektionale Kommunikation. Dabei verwaltet das Model die Daten, während die View für die Darstellung zuständig ist. Die dazwischenliegende Vue-Instanz ist das View-Model. Wie in Abb. 5 zu sehen, bildet sie die Brücke zwischen Model und View. Werden Daten in der View geändert, sorgt es für eine Synchronisierung mit dem Model. [25], [26]

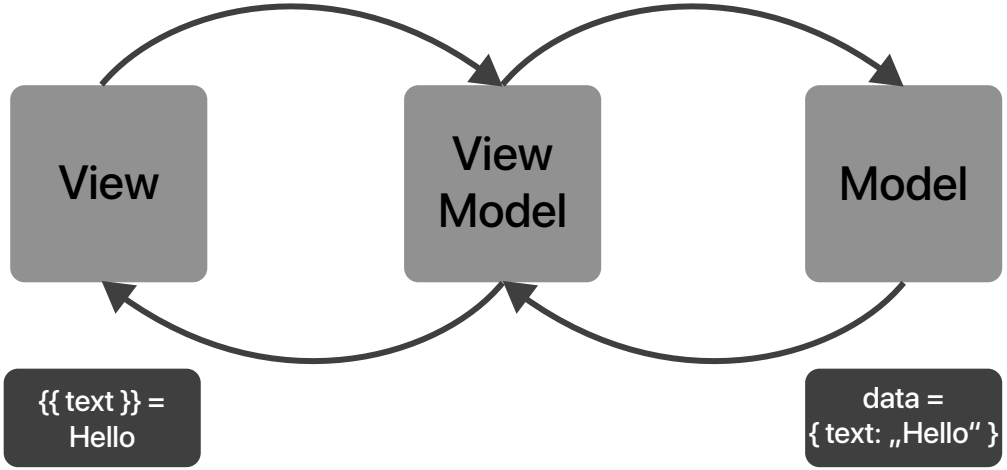

Abb. 5: *Model-View-ViewModel*

Angular

Angular ist eine Plattform und ein TypeScript-basiertes Framework. TypeScript-basiert bedeutet hierbei, dass Angular selbst in TypeScript geschrieben ist. Der Anwender nutzt HTML und TypeScript, um eine Applikation zu erstellen. Angular bietet einen umfassenden Katalog an Funktionalitäten, sodass normalerweise keine externen Skripte benötigt werden. Falls nötig, ist der Import von TypeScript-Bibliotheken möglich. Im Folgenden liegt der Fokus auf der Erstellung von Nutzeroberflächen. Eine Angular-Applikation besteht aus Komponenten, welche sich flexibel wiederverwenden und ändern lassen. Im Gegensatz zu Vue und React verwendet Angular kein VDOM, sondern überträgt die Änderungen direkt in das normale DOM. [27], [28]

Obwohl Angular viele Ansätze der MVC Architektur übernimmt, nutzt es eine komponentenbasierte Struktur (siehe Abb. 6). So bestehen Angular Komponenten aus einem Template (View) und einer Klasse/Komponente (View-Model). Der Entwickler kann je nach Anforderung entscheiden, ob er mit einem unidirektionalen oder bidirektionalen Datenfluss arbeiten will. [27], [28]

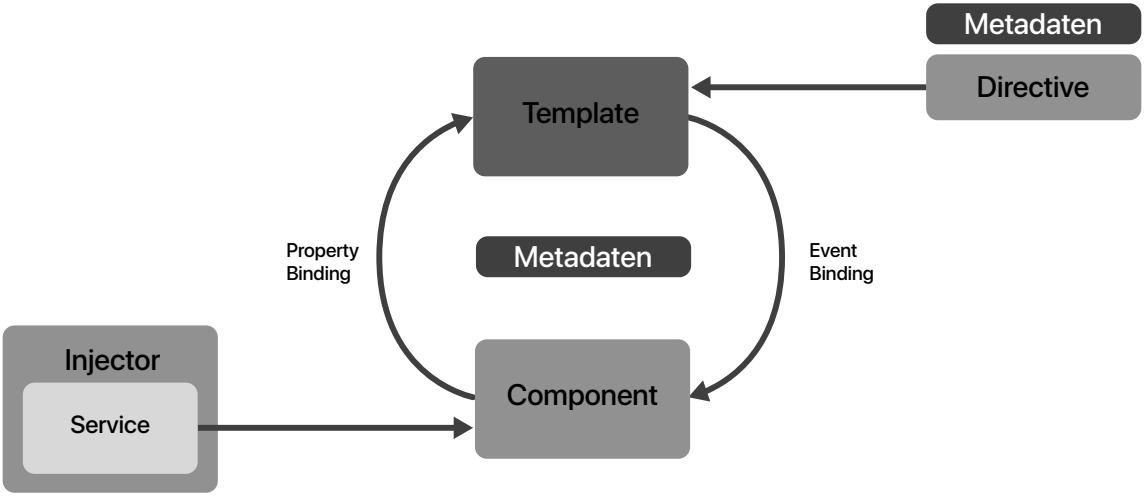

Abb. 6: *Angular Architektur*

Wie Abb. 7 zeigt, ist die Architektur von Angular zwischen React und Vue zu verorten. Während React nur unidirektional und Vue nur bidirektional kommunizieren kann, ist für den Entwickler bei Angular die Nutzung beider Arten möglich.

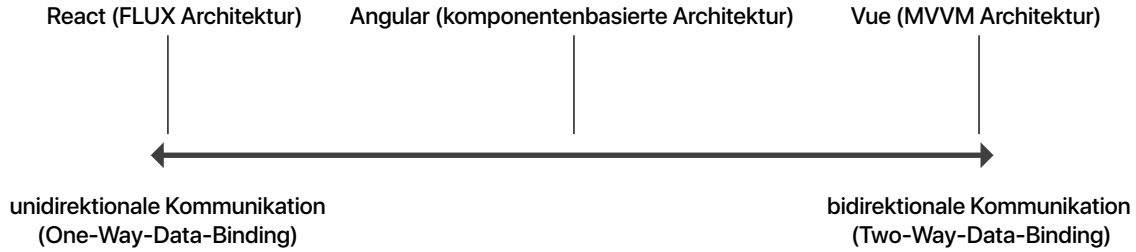

Abb. 7: *architekturelle Einordnung der JavaScript Frameworks*

3.1.2 Mobile Application Development Frameworks

Mobile Application Development Frameworks werden zur Entwicklung von betriebssystemübergreifenden Applikationen verwendet. Durch eine gemeinsame Code-Basis ermöglichen sie die Erstellung von Applikationen für unterschiedliche Betriebssysteme.

telerik.com definiert betriebssystemübergreifende Applikationen wie folgt:

„Hybrid apps, like native apps, run on the device, and are written with web technologies (HTML5, CSS and JavaScript). Hybrid apps run inside a native container and leverage the device's browser engine (but not the browser) to render the HTML and process the JavaScript locally. A web-to-native abstraction layer enables access to device capabilities that are not accessible in Mobile Web applications, such as the accelerometer, camera and local storage." [29]

Eine betriebssystemübergreifende Applikation ermöglicht kostengünstige Entwicklung und bietet trotz einer Code-Basis Zugriff auf die nativen Funktionen des entsprechenden Smartphones. Beliebte Mobile Application Development Frameworks sind zum Beispiel Cordova, React Native oder Titanium, wie in Abbildung 8 zu sehen ist. [30]

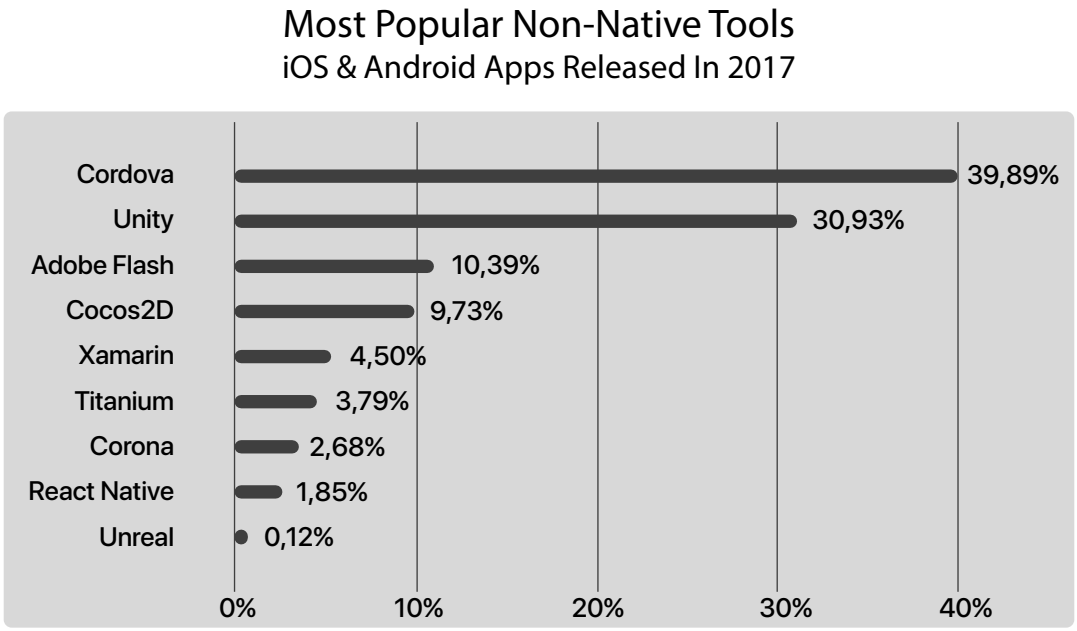

Abb. 8: *Statistik Werkzeuge zur Erstellung von betriebssystemübergreifenden Applikationen*

Cordova

Cordova ist ein Open Source Mobile Application Development Framework. Es ist das am meisten genutzte Werkzeug zur Erstellung von betriebssystemübergreifenden Applikationen. Zur Entwicklung werden die Programmiersprachen HTML, CSS und JavaScript verwendet. Cordova generiert im Anschluss ein plattformspezifisches Projekt, in welchem eine Web-View als initiale Komponente ausgeführt wird. Dabei verweist eine entsprechende Kontrollinstanz dieser Web-View auf die Anwendung. Durch die Verwendung eines Foreign Function Interfaces (FFI), kann die Cordova Applikation jedoch auch auf die nativen Funktionen des Smartphones wie z. B. die Kamera zugreifen. Eine Erweiterung der Funktionalität von Cordova erfolgt durch Plug-Ins. [30], [31], [32]

React Native

React Native wurde von Facebook als Open Source Projekt entwickelt und ermöglicht die Erstellung von Applikationen für Android, iOS und die Universal Windows Plattform (UWP). Für die Entwicklung wird die Syntax JSX verwendet. Wie bei React setzt sich auch hier die Nutzeroberfläche aus Komponenten zusammen. Dabei greift React Native auf die gleichen Grundelemente wie native Applikationen zurück. Der JSX-Code wird direkt interpretiert und verändert anschließend die nativen Views. Ähnlich wie bei Cordova kann auch React Native durch sogenannte Module um Funktionalitäten erweitert werden. Die Besonderheit ist, dass diese Module nicht nur in JavaScript, sondern auch in Objective-C, Swift oder C++ geschrieben sein können. Somit haben sie auch direkten Zugriff auf die nativen Funktionen des Smartphones. [33]

Titanium

Die Open Source Software Titanium von Appcelerator erlaubt die Entwicklung von Applikationen mit JavaScript für iOS, Android, UWP oder Blackberry. Dieser Code kommuniziert mit dem Titanium SDK, welches daraus native Komponenten generiert. Auch Titanium kann durch Module in seiner Funktionalität erweitert werden. [34], [35]

3.2 Augmented Reality

Vor Beginn der genaueren Betrachtung von Augmented Reality, muss eine Abgrenzung zu Mixed Reality, Augmented Virtuality und Virtual Reality stattfinden. Im Folgenden werden die Begriffe und deren Bedeutung genauer erläutert.

3.2.1 Definition wichtiger Begriffe

Reality

Reality (dt. Realität) wird von den Lexico Dictionaries folgendermaßen definiert:

„The state of things as they actually exist, as opposed to an idealistic or notional idea of them." [24]

Mixed Reality

Der Begriff Mixed Reality (MR, dt. Gemischte Realität) wird fälschlicherweise oftmals als Synonym für Augmented Reality verwendet. Im Jahr 1994 hat Paul Milgram die Begriffe Augmented Reality und Virtual Reality (VR) eingeordnet. Er hat die allgemein anerkannte Theorie aufgestellt, dass MR als ein Kontinuum zu betrachten ist. Abb. 9 zeigt den Übergang von der Realität zur Virtualität. [12]

Wenn die Realität überwiegt, wird der Begriff Augmented Reality genutzt. Sobald der Anteil der Virtualität größer ist, findet der Begriff Augmented Virtuality (AV) Verwendung. [14]

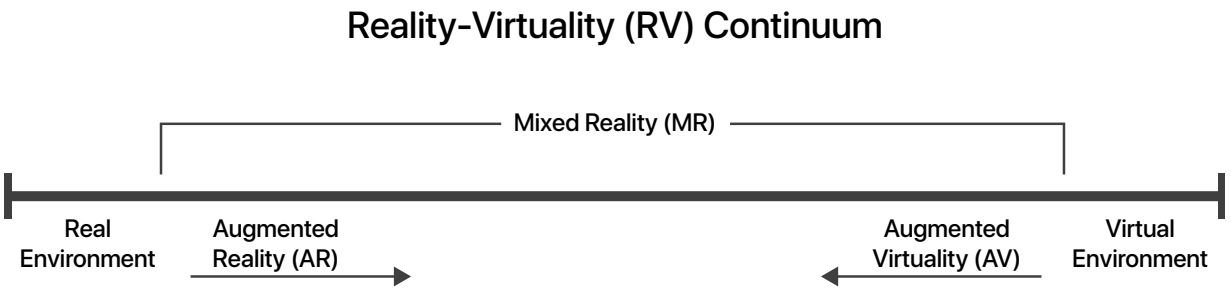

Abb. 9: *Reality-Virtuality Kontinuum*

Milgram und Kishino beschreiben in ihrem Paper „Augmented Reality: A class of displays on the reality-virtuality continuum" drei Dimensionen, mit denen MR-Anwendungen eingeordnet werden können:

- Das Extend of World Knowledge (EWK, dt. Wissen über die Welt) beschreibt, wie viel das MR-System von der Außenwelt weiß. Abb. 10 zeigt, dass sich das Kontinuum von keinem Wissen bis hin zu einem vollständigen Modell der Welt erstreckt. [12]

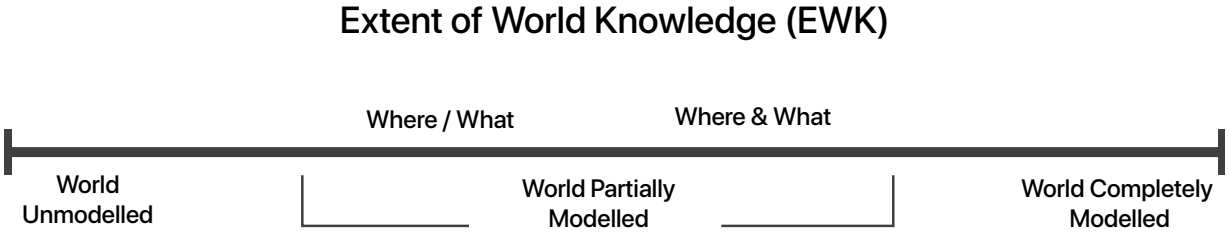

Abb. 10: *Extend of World Knowledge*

- Abb. 11 zeigt die Reproduction Fidelity (RF, dt. Reproduktionstreue), welche angibt, wie realistisch die echte Welt erfasst und wiedergegeben wird. Im Videobereich reicht sie von Monovideo bis hin zu 3D-High-Definition-Videos und im Grafikbereich von einfachen Skizzen bis hin zu fotorealistischen Grafiken. [12]

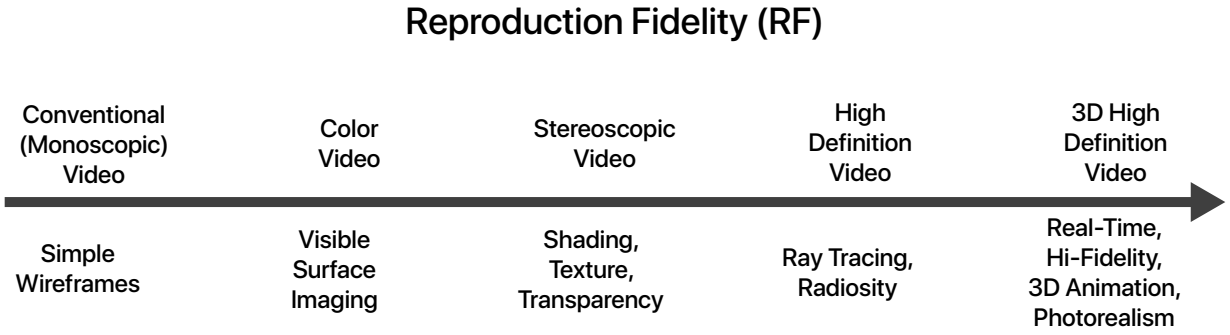

Abb. 11: *Reproduction Fidelity*

- Das Extent of Presence Metaphor (EPM, dt. Ausmaß der Präsenz) zeigt auf, wie stark die Immersion beim Nutzer ist. Wie in Abb. 12 dargestellt, reicht diese von einem geringen Präsenzgefühl (bei normalen Monitoren) bis hin zur vollständigen Immersion (bei HMD´s). [12]

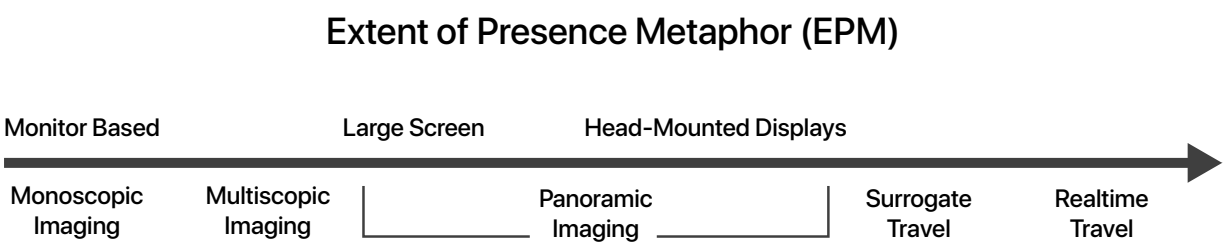

Abb. 12: *Extent of Presence Metaphor*

Augmented Reality

Für Augmented Reality (AR, dt. erweiterte Realität) existieren viele unterschiedliche Auslegungen. Die bekannteste Definition ist hierbei von Ronald T. Azuma aus dem August 1997 in „A Survey of Augmented Reality". So sagt er, dass es AR dem Nutzer ermöglicht, virtuelle Objekte mit der realen Welt zu überlagern oder zusammenzusetzen. Anstatt dem Nutzer eine vollständig virtuelle Umgebung zu zeigen, wird die Realität ergänzt. Nach Azuma muss AR bestimmte Merkmale aufweisen: Sie muss Realität und Virtualität kombinieren, interaktiv und in Echtzeit stattfinden und die virtuellen Inhalte in der realen Welt verankern. [9], [13]

Ralf Dörner definiert AR in seinem Buch „Virtual und Augmented Reality (VR/AR)" wie folgt:

„Augmentierte Realität ist eine (unmittelbare, interaktive und echtzeitfähige) Erweiterung der Wahrnehmung der realen Umgebung um virtuelle Inhalte (für beliebige Sinne), welche sich in ihrer Ausprägung und Anmutung soweit wie möglich an der Realität orientieren, so dass im Extremfall (so das gewollt ist) eine Unterscheidung zwischen realen und virtuellen (Sinnes-) Eindrücken nicht mehr möglich ist." [14]

Augmented Virtuality

Augmented Virtuality (AV, dt. erweiterte Virtualität) ist neben AR eine weitere Form der Mixed Reality. Hier überwiegt der Anteil der Virtualität.[14]

Das deutschsprachige Online-IT-Lexikon www.itwissen.info definiert Augmented Virtuality folgendermaßen:

„Die erweiterte Virtualität, Augmented Virtuality (AV), gehört ebenso wie die erweiterte Realität (AR) zu den gemischten Realitäten, den Mixed Realities (MR). Es ist eine Symbiose aus der virtuellen Welt, die mit Informationen aus der realen Welt angereichert wird." [36]

Virtual Reality

Virtual Reality (VR, dt. virtuelle Realität) wird von Paul Milgram in „Augmented Reality: A class of displays on the reality-virtuality continuum" aus dem Jahr 1994 als eine künstliche Welt, in welche der Benutzer vollständig eintaucht, beschrieben. Diese Umgebung kann entweder der Realität nachempfunden oder komplett fiktional sein. In der Virtualität ist es z. B. möglich, physikalische Gesetze außer Kraft zu setzen. [12]

Eine weitere Beschreibung von VR im Online-Lexikon von www.Onlinemarketing.de lautet:

„Virtual Reality meint die computergenerierte Darstellung einer Welt in Echtzeit. Virtual Reality blendet die Außenwelt komplett aus und beeinflusst die menschlichen Sinne durch computeranimierte Videos, Animationen und Töne. Die Realitätsveränderung wird von VR-Brillen umgesetzt, die auch Bewegungen des Nutzers erkennen und übertragen." [37]

Hierbei ist besonders das Wort „Echtzeit" wichtig. Der Nutzer kann so direkt mit der virtuellen Welt interagieren und wird nicht durch Verzögerungen aus der Immersion geworfen. [37]

3.2.2 Funktionsweise

Um zu verstehen, wie eine AR-Applikation eigentlich funktioniert, muss ein genauerer Blick auf AR-Systeme geworfen werden. Dirk Schart unterteilt diese in seinem Buch „Augmented und Virtual Reality für Marketing, Medien und Public Relations" in folgende Komponenten:

- „Hardware: eine oder mehrere Kameras sowie Sensoren eines Mobilgeräts oder einer Augmented- oder Mixed-Reality-Brille" [38]
- „Tracking-Software und Renderer für die Berechnung und Anzeige der korrekten Überlagerung sowie als Szenengenerator" [38]
- „Anzeigegerät (z.B. Display eines Mobilgeräts, ein Head-Mounted Display, ein Monitor), auf dem die virtuellen Objekte eingeblendet werden" [38]

Kombiniert man diese drei Komponenten, so ist man in der Lage eine AR-Anwendung zu erstellen. Der erste Schritt bei der Benutzung einer solchen Applikation ist die Videoaufnahme. Der Benutzer zeichnet mit der Kamera ein Video seiner Umgebung auf. Im Anschluss wird dieses Video analysiert. Position und Orientierung des Aufzeichnungsgeräts resultiert aus den Ergebnissen des Trackingvorgangs. Anschließend findet die Ausrichtung, oft auch als Registrierung bezeichnet, der computergenerierten Inhalte statt.

Anhand der beim Tracking gewonnenen Daten, werden die Inhalte fest im Raum verankert. So sieht es für den Benutzer aus, als hätte diese einen festen Platz in der Umgebung. Zum Schluss werden diese, nun perspektivisch korrekt ausgerichteten Inhalte, dem Video überlagert. [14], [38]

Tracking

Grundsätzlich unterscheidet man beim Tracking zwischen visuellem und nichtvisuellem Tracking. Beide Arten des Trackings sollen in diesem Kapitel genauer erläutert werden.

Visuelles Tracking

Markerbasiertes Tracking

Das markerbasierte Tracking ist eines der gängigsten Trackingverfahren. Diese Art wird schon seit Ende der 90er Jahre verwendet. Hierbei kommen Marker mit schwarz-weißen Mustern oder Symbolen zum Einsatz, um daran virtuelle Inhalte auszurichten. Mittlerweile finden auch immer öfter farbige Marker Verwendung. Diese bieten für die Kamera bei wechselnden Lichtverhältnissen jedoch teilweise zu wenig Kontrast für eine problemlose Erkennung. Die gängigste Form eines Markers ist ein Quadrat. Die Begrenzung erfolgt über einen schwarzen oder weißen Rahmen. Innerhalb dieses Rahmens ist es auch möglich, benutzerdefinierte Symbole oder Grafiken zu platzieren. Hierbei muss jedoch beachtet werden, dass dies Performanceverluste nach sich ziehen kann. Für eine Erkennung der Marker müssen diese vollständig sichtbar sein. Sobald ein Teil des Markers aus dem Sichtfeld der Kamera verschwindet, ist das Tracking nicht mehr möglich. Gleiches gilt, wenn der Marker nicht richtig fokussiert werden kann oder ein Objekt den Marker verdeckt. [14]

Ein Vorteil dieser Technik ist die günstige Erstellung und Vervielfältigung der Marker. Auf der anderen Seite müssen sie immer auf dem zu trackenden Objekt angebracht werden. Dies kann störend wirken und eventuell sogar wichtige Informationen verdecken. Außerdem muss der Nutzer wegen der Fokussierung immer den richtigen Abstand zum Marker haben. [14]

Im Folgenden soll die Funktionsweise des markerbasierten Tracking anhand von Abb. 13 dargelegt werden.

Der erste Schritt zur Erkennung des Markers ist die Suche nach vier zusammenhängenden Linien im Kamerabild (1). Findet die Software einen Marker, wird durch die Erhöhung des Schwellwerts ein Schwarzweißbild erzeugt (2). Dieses ermöglicht es, den Marker leichter zu tracken. Anhand der Eckpunkte errechnet die Software dessen Perspektive. Die Speicherung der Proportionen des Markers im System macht dies möglich. Anschließend folgt die Normalisierung der Region innerhalb des gefundenen Quadrats. Das

Muster wird aus dem Quadrat extrahiert und mit einer Datenbank abgeglichen. Die verwendeten Marker sollten möglichst eindeutige Muster haben (3). Wenn das erkannte Muster in der Datenbank vorhanden ist, kann der referenzierte Inhalt geladen werden. Abschließend folgt die Zusammenfügung von 3D-Objekt und Realbild (4). [9], [14]

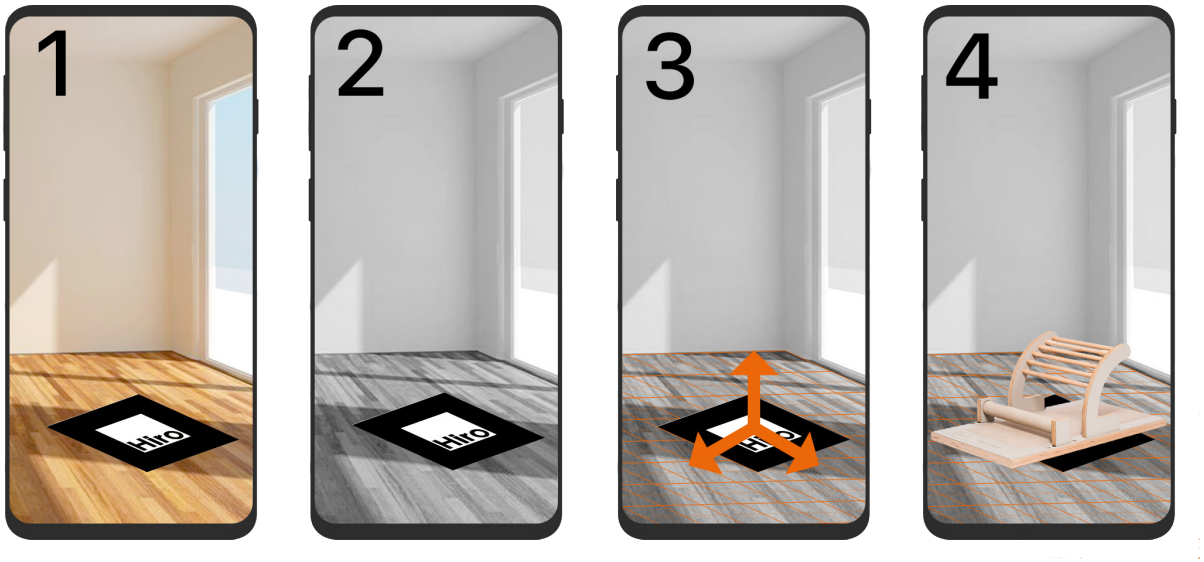

Abb. 13: *Tracking von Markern*

Markerloses Tracking

Beim markerlosen Tracking unterscheidet man grundsätzlich zwischen modelllosem und modellbasiertem Tracking. Bei letzterem werden der Software Referenzbilder und -modelle zur Verfügung gestellt. Diese vergleicht nur noch die Schlüsselpunkte des aktuellen Kamerabilds mit denen der Referenz. Auch beim modellosen Tracking wird ein temporäres Modell erstellt. Da dies aber erst während dem Vorgang erfolgt, ist diese Technik flexibler. Der Nachteil ist hierbei, dass die Objekte nur relativ zum Startpunkt positioniert und nicht in der Umgebung vorregistriert werden können. [38]

Modelloses Tracking nutzt natürliche Merkmale zur Erkennung eines Objekts oder der Umgebung (1). Für ein stabiles Tracking braucht es eine gute Bildqualität und ausreichend eindeutig identifizierbare Schlüsselpunkte (auch Keypoints oder Interest-Points genannt). Deshalb ist es von Vorteil, wenn die zu trackende Oberfläche Unregelmäßigkeiten oder einzelne, auffällige Merkmale besitzt (2). In Abb. 14 wäre dies z. B. das Parkett oder die Eckpunkte des Raumes. Sie sollten aus verschiedenen Blickwinkeln gut

erkennbar sein, um ein Tracking von mehreren Standorten zu ermöglichen. Nachdem genug Schlüsselpunkte erkannt wurden, sucht der Algorithmus nach einer zusammenhängenden Fläche (3). Auf dieser können anschließend Objekte platziert werden (4). Im Gegensatz zum Tracking von Markern verwendet diese Technik deutlich anspruchsvollere Bilderkennungsalgorithmen, welche höhere Anforderungen an die Performance stellen. [9]

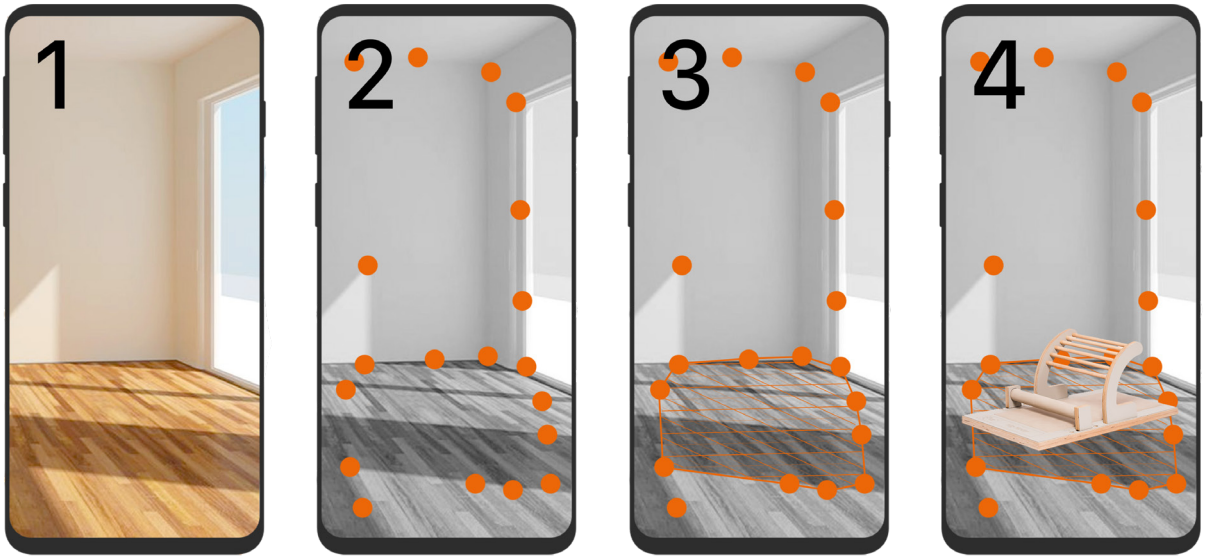

Abb. 14: *Markerloses Tracking*

Bildtracking

Auch das Tracking von Bildern ist mit dem modellbasierten Verfahren möglich. Aufdrucke auf Verpackungen oder in Magazinen lassen sich so als Marker nutzen und mit virtuellen Inhalten überlagern. Oftmals ist auch das gleichzeitige Tracking von mehreren Bildern möglich. [38]

Gesichtstracking

Gesichtstracking nutzt die modellose Trackingvariante zur Erkennung von Gesichtern im Bild. Viele Trackingsysteme sind mittlerweile in der Lage, neben der Position des Gesichts, noch weitere Informationen zu erkennen. Hierzu zählen z. B. Geschlecht, Alter oder Gesichtsausdrücke. Diese Informationen lassen sich bei der Überlagerung von virtuellem Inhalt zur individuellen Anpassung an das Gesicht der Person nutzen. [39]

Objekt- und Umgebungstracking

Objekttracking greift auf das modellbasierte Verfahren zur Erkennung von z. B. Stühlen oder Tischen zurück. Umgebungstracking ermöglicht das Ganze in größeren Dimensionen. Neben normalen Objekten ist auch die Erkennung von ganze Räumen, Gebäuden oder Landschaften möglich. [40]

Nichtvisuelles Tracking

Global Positioning System (GPS)

GPS kann zur Bestimmung der aktuellen Position eines GPS-Empfängers genutzt werden. Die im Erdorbit stationierten Satelliten senden codierte Funksignale und messen die Zeit bis zur Ankunft beim Empfänger. Sobald ein Empfänger Signale von mehr als vier Satelliten bekommt, ist die Berechnung der Position auf der Erdoberfläche möglich. Die Messung kann je nach verfügbarer Anzahl von Satelliten und dem Signalempfang auf der Erdoberfläche in ihrer Genauigkeit variieren. Auch die Nutzung von GPS in Innenräumen sorgt durch Signalreflexionen von den Wänden für Ungenauigkeiten. Da die Höhenwerte oft Messfehler enthalten, wird zur Positionsbestimmung meist nur Längen- und Breitengrad verwendet. Zur genaueren Bestimmung des Standorts empfiehlt sich die Verwendung von Differential GPS (DGPS). Dabei empfangen Bodenstationen ein separates Korrektursignal. Anschließend findet eine Messung der atmosphärischen Verzerrungen statt. Diese wird anschließend bei der Positionsbestimmung berücksichtigt. [9], [38]

Magnetometer

Das Magnetometer misst die Richtung des Magnetfelds der Erde. So ist die Bestimmung der aktuellen Richtung des Geräts, bezogen auf den magnetischen Norden, möglich. Die Messungen werden auf drei Achsen (3DOF) durchgeführt. Leider sind diese jedoch sehr fehleranfällig, da sie durch lokale Magnetfelder verzerrt werden können. Hierfür reicht schon eine Armbanduhr aus Metall in der Nähe des Smartphones. Deshalb empfiehlt es sich, das Magnetometer in Verbindung mit anderen Sensoren zu nutzen. [9], [38]

Gyroskop

Ein Gyroskop misst die Rotationsgeschwindigkeit. Dazu ermittelt es die Trägheitskraft eines kleinen vibrierenden Objekts, welches seine Orientierung auch beibehält, wenn das Gerät rotiert. Anschließend erfolgt die Berechnung der Veränderung der Geräteorientierung. Für eine 3DOF-Orientierungsmessung müssen drei orthogonale Gyroskope in einem mikroelektromechanischen System (MEMS) kombiniert werden. [9][38]

Accelerometer

Das Accelerometer ist das Gegenstück zum Gyroskop. Es ist auch nach dem MEMS-Ansatz gebaut und ist für die Beschleunigungsmessung zuständig. Es besteht aus einer kleinen Masse, welche zwischen zwei Federn aufgehängt ist. Sobald sich der Sensor beschleunigt, verschiebt sich die Masse. Diese Verschiebung sorgt für eine Änderung der Kapazität an der Elektrode. Ein Sensor misst diese Änderung und kann daraus die Beschleunigung berechnen. Neben der Beschleunigungsmessung kann es auch zu Messung der Neigung des Geräts genutzt werden. [9], [38]

Sensor Fusion

Da man bei AR-Applikationen ein sehr gutes Tracking benötigt, ist die Kombination mehrerer Sensoren sinnvoll. Die sogenannte Sensor Fusion gleicht die Schwächen der einzelnen Sensoren aus und gewährleistet ein besseres Nutzererlebnis. Ein Smartphone verfügt mindestens über eine Kamera, GPS, Accelerometer und Magnetometer (siehe Abb. 15). Die Kombination von allen Sensoren erhöht allerdings auch den Akkuverbrauch und erfordert eine zusätzliche Kalibrierung. [9], [38]

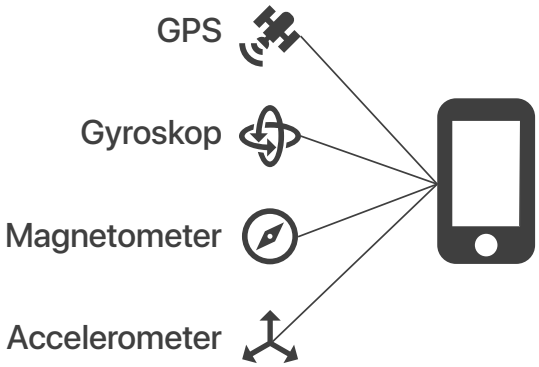

Abb. 15: *Sensor Fusion*

Simultaneous Location and Mapping-Tracking (SLAM)

SLAM vereint modellbasiertes und modellloses Tracking. Vor Benutzung der eigentlichen Anwendung wird die Umgebung gescannt. Dabei muss darauf geachtet werden, dass die Umgebung genügend eindeutige Merkmale (Schlüsselpunkte) hat und gleichmäßig ausgeleuchtet ist. Ein Algorithmus erstellt daraus eine Karte. Anhand dieser wird die Position des eigenen Geräts und die relative Position von anderen Objekten im Raum bestimmt. In Kombination mit den Sensoren des Smartphones lassen sich die realen Abstände in

der Umgebung berechnen. Sobald genug Merkmale der Umgebung erkannt wurden, kann sich der Nutzer mit dem Smartphone frei im Raum bewegen. Die Platzierung von virtuellen Inhalten auf physischen Oberflächen funktioniert dabei ohne Marker. Die Inhalte bleiben an der festgelegten Stelle, auch wenn sie aus dem Sichtfeld verschwinden. Sobald sich die Umgebung verändert, werden neue Schlüsselpunkte gesetzt und der Karte hinzugefügt. [9], [38]

Fazit

Markerbasiertes Tracking hat den Vorteil, dass es schnell und sicher funktioniert. Es gibt jedoch auch viele Nachteile, wie z. B. das Design der Marker oder die Notwendigkeit Objekte auf festgelegten Stellen zu platzieren. Durch die rasante Entwicklung bei visuellen Trackingtechnologien verliert markerbasiertes Tracking zunehmend an Bedeutung.

Das markerlose Tracking eröffnet viele neue Möglichkeiten. So kann sich der Nutzer frei in der Umgebung bewegen, während seine Position per SLAM genau bestimmt wird. Auch hier wird Sensor Fusion genutzt, da einzelne Sensoren nur eine unzureichende Genauigkeit beim Tracking und somit ein schlechtes Nutzererlebnis bieten.

Die Kombination von visuellen und nichtvisuellen Trackingtechnologien sorgt für ein stabiles Tracking. Durch die Erkennung von Bildern, 3D-Objekten und Flächen in der Umgebung, ist die Platzierung von virtuellen Inhalten nahezu überall möglich.

Registrierung

Für eine korrekte Abbildung der virtuellen Objekte ist eine sogenannte Registrierung notwendig. Sie sorgt dafür, dass die Objekte perspektivisch korrekt an einer bestimmten Position im Realbild verankert werden (Abb. 16, Bild 2). Wenn das Objekt nicht richtig registriert ist, wirkt es für den Nutzer, als ob die Objekte im Raum herumschweben (Abb. 16, Bild 1). Für die perspektivisch korrekte Darstellung des Objekts müssen alle Koordinatensysteme aneinander angepasst sein. Bei der sogenannten Modelltransformation erfolgt die Anpassung der lokalen Objektkoordinaten an das globale Koordinatensystem. Die Software positioniert das Objekt anhand des globalen Koordinatensystems in der realen Welt. Da sich der Nutzer in den meisten Fällen in der realen Welt bewegt, muss außerdem eine Ansichtstransformation stattfinden. Sie definiert die Beziehung zwischen der realen Welt und den Kamerakoordinaten. Das Tracking der Nutzerbewegung ermöglicht die perspektivisch korrekte Darstellung des globalen Koordinatensystems und damit auch des 3D-Objekts. [9]

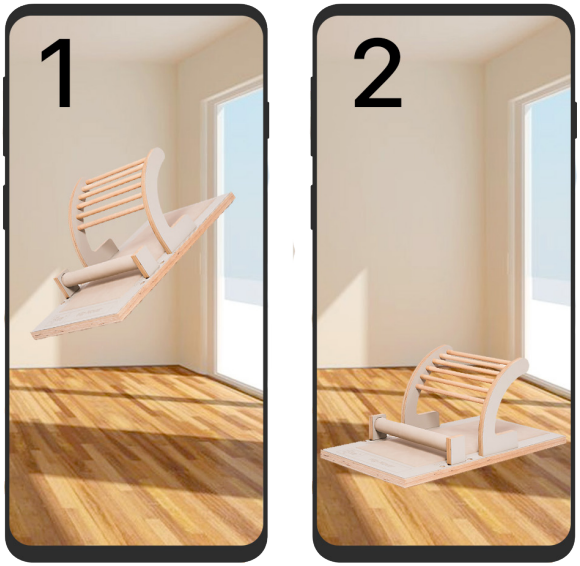

Abb. 16: *Registrierung von virtuellen Inhalten*

Darstellung

Es gibt verschiedene technische Ansätze, um virtuelle Objekte mit der Realität zu kombinieren. Diese Ansätze, sowie die Unterscheidung verschiedener Displayarten, werden im Folgenden dargelegt.

Bei monokularen Displays wird nur ein Screen genutzt, um die Bilder wiederzugeben. Bei binokularen Displays steht jedem Auge ein Screen zur Verfügung. [9]

Die Einordnung von AR-Displays geschieht anhand der Entfernung zum Auge. Abb. 17 zeigt die verschiedenen Arten in der Übersicht.

Head-Mounted-Displays (HDM) werden meist in der Form einer Brille getragen. Die direkte Überlagerung des Sichtfelds mit virtuellen Inhalten sorgt für eine Immersion. Außerdem reagiert das HDM auf alle Bewegungen der Person. [9]

In die Kategorie der Handheld-Displays fällt unter anderem das Smartphone. Auch hier kann der Nutzer das Sichtfeld durch das Bewegen des Smartphones verändern. Jedoch ist eine Überlagerung mit virtuellen Objekten nur für die von der Kamera erfassten Bereiche möglich. [9]

Ein stationäres Display ist fest an einer Position verankert. Hier kann der Nutzer z. B. einen Marker oder ein Bild vor die Kamera halten. Innerhalb des Sichtfelds der Kamera bekommt er die AR-Ansicht davon angezeigt. [9]

Auch projizierte Displays lassen sich nicht bewegen, da die Überlagerung der Realität mit einem Projektor stattfindet. [9]

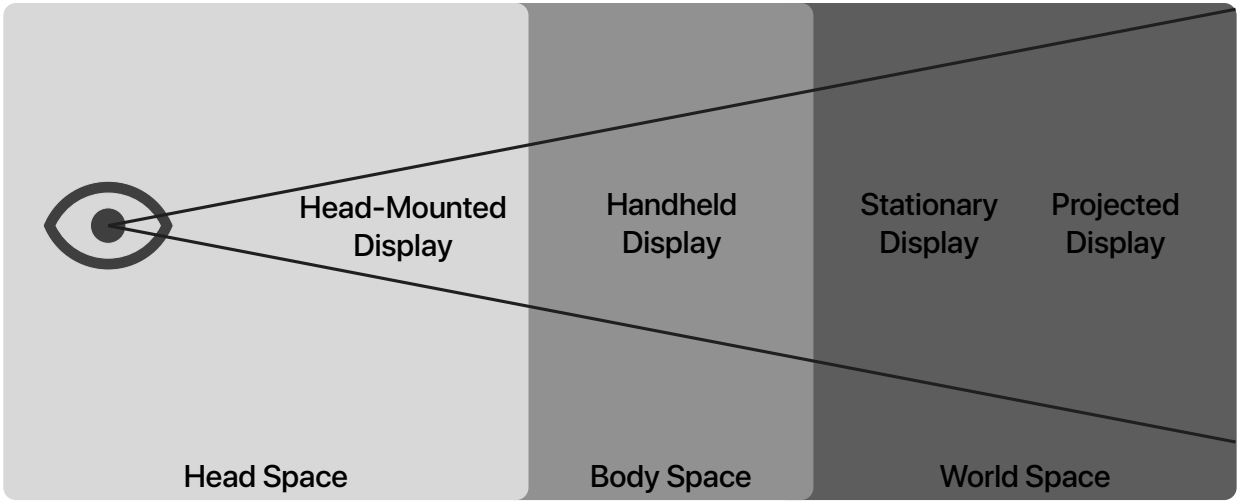

Abb. 17: *Kategorisierung von AR-Displays*

Optische See-Through-Displays

Abb. 18 zeigt ein optisches See-Through-Display, beim dem der Nutzer die reale Welt sieht und die virtuellen Objekte mit Hilfe eines transparenten Displays optisch überlagert werden. Es gibt unterschiedliche Bauweisen von Brillen, welche diese Technik verwenden. Die am häufigsten verwendete Art funktioniert mit semitransparenten Spiegeln. Diese Variante wird meist bei Head-Mounted-Displays verwendet. [14], [41]

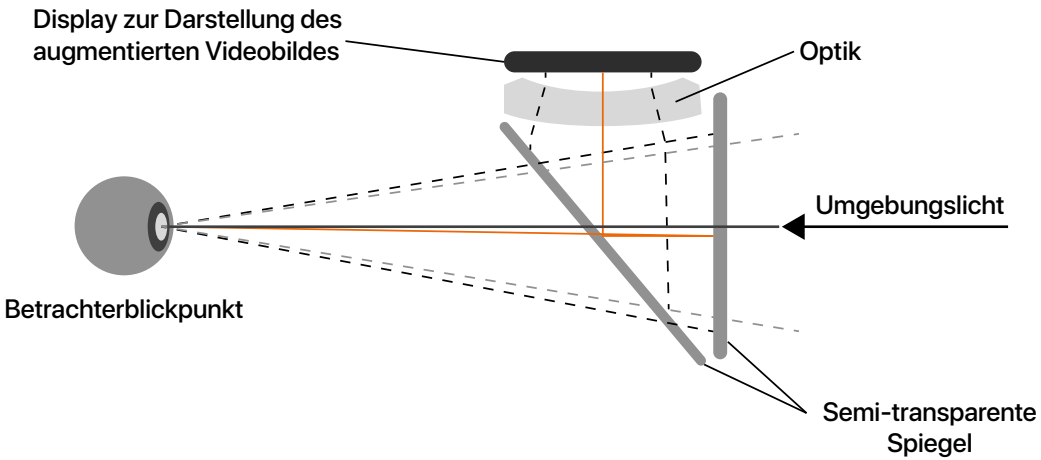

Abb. 18: *Optisches See-Trough-Display*

Video-See-Through-Displays

Wie der Name vermuten lässt, wird bei Video-See-Through Displays per Kamera ein Video aufgezeichnet, welches auf einem Display wiedergegeben wird (siehe Abb. 19). Für den Nutzer entsteht so der Eindruck, dass er die reale Umgebung betrachtet. Die 3D-Objekte werden in Echtzeit perspektivisch korrekt in das Video eingefügt. Dies ist z. B. bei AR-Anwendungen auf Smartphones der Fall. [14], [41]

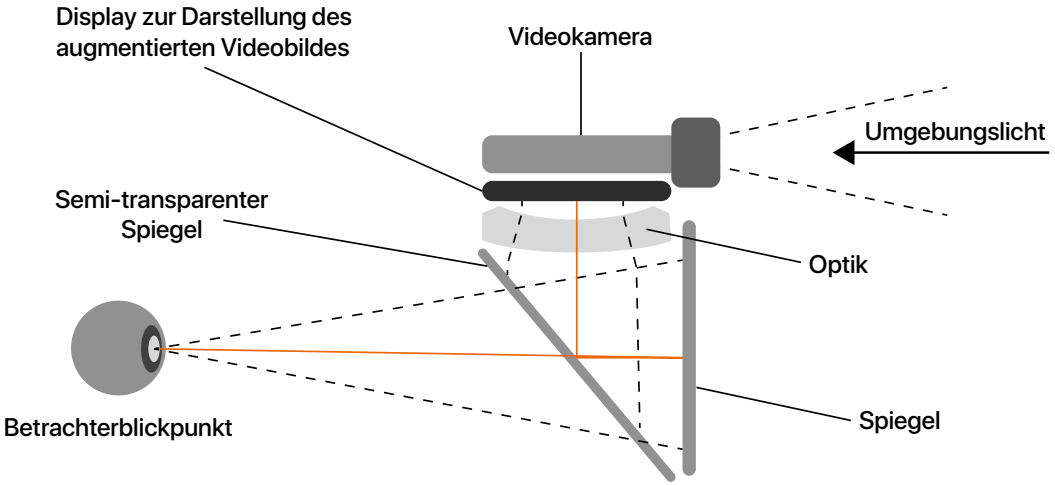

Abb. 19: *Video-See-Trough-Display*

Projektionsbasierte Displays

Bei dem in Abb. 20 dargestellten projektionsbasierten Display, werden bestehende Gegenstände durch Projektoren angeleuchtet, sodass „sich die Wahrnehmung der realen Gegenstände verändert“. [14] Da die Gegenstände an festen Positionen stehen, kann die Darstellung nur an einem bestimmten Platz stattfinden. Die unterschiedlichen Farben der Projektionsflächen können das projizierte Bild beeinflussen und sollten deshalb schon vorab Beachtung finden. Einfarbige Objekte sind am besten als Projektionsfläche geeignet.

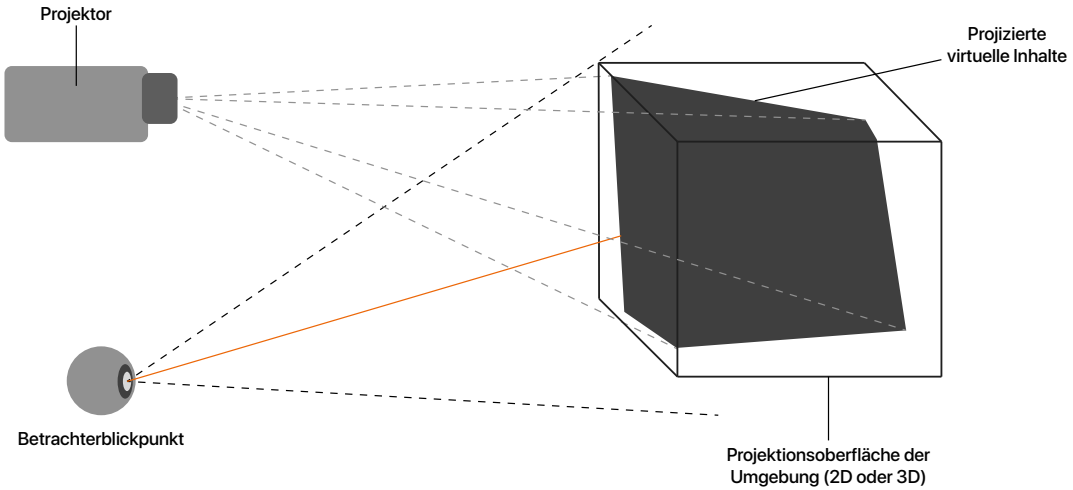

Abb. 20: *Projektionsbasiertes See-Trough-Display*

3.2.3 Interaktion mit Augmented Reality

Um mit einer AR-Applikation zu interagieren, gibt es unterschiedliche Möglichkeiten. Im Folgenden sollen zwei grundlegende Techniken zur Steuerung einer Applikation erläutert werden.

Tangible User Interfaces (TUI) sind die am häufigsten genutzte Schnittstelle für AR-Anwendungen. Sie erlauben dem Nutzer mithilfe von physischen Objekten die Interaktion mit der Applikation. Eine der bekanntesten Arten ist der Touchscreen, welcher Ein- und Ausgabe der Applikation vereint. Mithilfe von Multi-Touch-Gestik kann der Nutzer mit den virtuellen Objekten interagieren. [14], [41]

Eine andere Art der Interaktion findet mit verschiedenen Handzeichen statt. Diese können von der Anwendung als Befehle interpretiert und ausgeführt werden. In diesem Fall muss der Nutzer jedoch zuerst die entsprechenden Gesten lernen, um die Applikation bedienen zu können. [14], [41]

4 Stand der Technik

Markerlose Augmented Reality-Technologien für mobile Endgeräte

Durch den rasanten Fortschritt der Technik haben seit einiger Zeit auch Smartphones genügend Leistung, um AR-Anwendungen auszuführen. Bis vor kurzem war die Entwicklung einer solchen Applikation jedoch sehr komplex. So mussten eigene Algorithmen zum Tracking von Objekten, der Interpretation der Sensordaten und dem Rendering von 3D-Objekten geschrieben werden. Mittlerweile gibt es viele SDKs, welche Entwicklern diese Arbeit abnehmen und vorgefertigte Funktionen bieten.

Dieses Kapitel behandelt den aktuellen technischen Stand von markerlosen AR-Technologien für das Smartphone. Laut der definierten Zielsetzung sind native Technologien nur von geringer Relevanz, da sie die Applikationserstellung lediglich für ein bestimmtes Betriebssystem ermöglichen. Sie werden trotzdem in der Theorie behandelt, da einige der webbasierten und betriebssystemübergreifenden Technologien auf die Frameworks ARKit und ARCore aufbauen. Die gerade genannten Technologien sollen neben der Theorie auch anhand von Tests mit Beispielapplikationen bewertet werden. Der Fokus liegt dabei auf der Trackingqualität.

4.1 Native AR-Technologien

Unter nativen Applikationen versteht man Anwendungen, welche speziell für ein bestimmtes Betriebssystem wie z. B. Android oder iOS erstellt wurden. Bei jedem Betriebssystem unterscheiden sich die Programmierschnittstellen (API, engl. Application Programming Interface) grundlegend. Auch die Programmiersprachen der Betriebssysteme unterscheiden sich. Für die Entwicklung von iOS-Applikationen werden die Programmiersprachen Swift und Objective-C genutzt, bei Android hingegen Java und Kotlin. [42], [43]

Die Verteilung der Applikationen findet über sogenannte App-Stores statt. Die bekanntesten Beispiele sind hier der Play Store von Google oder der App Store von Apple. [44]

Ein Vorteil bei der Entwicklung von nativen AR-Anwendungen ist die Ansteuerung der Schnittstellen. AR-Applikationen benötigen neben dem Zugriff auf die Kamera noch eine Reihe an weiteren Sensoren, wie z. B. Gyroskop oder Accelerometer, um die Position des Nutzers zu bestimmen. Bei nativen Applikationen wird diese Ansteuerung über Googles AR-Core und Apples ARKit für Entwickler stark erleichtert. Außerdem bieten native Applikationen eine sehr gute Performance im Gegensatz zu webbasierten oder betriebssystemübergreifenden Alternativen. Dies liegt daran, dass eine zusätzliche Abstraktionsschicht entfällt. [45]

Bezogen auf den Aufwand schneiden native AR-Applikationen jedoch schlecht ab. So benötigt jedes Betriebssystem eine eigens geschriebene Anwendung. Das resultiert in höherem Zeit- und Kostenaufwand. Zusätzlich sorgen auch die unterschiedlichen APIs für einen erhöhten Einarbeitungsaufwand. [45]

4.1.1 ARKit

Das ARKit SDK von Apple ermöglicht Entwicklern die Erstellung von AR-Applikationen für iOS. ARKit funktioniert auf allen iPhones mit einem A9-Prozessor und ab iOS 11. [46]–[49]

Es stellt folgende Funktionalitäten zur Verfügung:

Tracking

- Tracking von Position und Ausrichtung des Geräts mithilfe von gleichzeitiger Lokalisierung und Kartierung (SLAM)
- Tracking von Bildern/2D-Oberflächen im Raum
- Tracking von 3D-Objekten nach vorherigem Scannen oder mit Vergleichsmodell
- Tracking von Flächen
- Speichern der Tracking-Daten, um die Sitzung zu einem späteren Zeitpunkt fortzusetzen
- Gesichtstracking von bis zu drei Gesichtern mit Überlagerung von virtuellem Inhalt (ab Herbst 2019 mit ARKit 3)
- Bewegungstracking bei Menschen und Übertragung der Bewegungen auf ein virtuelles 3D-Modell (ab Herbst 2019 mit ARKit 3)

Benutzerinteraktion

- Platzierung von 3D-Modellen im Raum und Interaktion über den Touchscreen
- Hit-Testing, um 3D-Positionen auf realen Oberflächen anhand eines Bildschirmpunkts zu finden

Rendering

- Rendering von Texturen, Reflektionen und Einbeziehung der Lichtverhältnisse
- Überlagerung von virtuellen Inhalten, welche von realen Objekten verdeckt werden (ab Herbst 2019 mit ARKit 3)

Multi-User

- Kommunikation zwischen mehreren iOS-Geräten in Echtzeit

Sonstiges

- Gleichzeitige Verwendung von Front- und Rückkamera
- Einsatz von künstlicher Intelligenz, um die Leistung zu verbessern

Mit ARKit 3 stehen ab Herbst 2019 alle der oben genannten Funktionen zur Verfügung. Außerdem wird eine Reihe an neuen Werkzeugen zur Erstellung von AR-Anwendungen veröffentlicht. [46]–[49]

Möglichkeiten zur Erstellung von AR-Applikationen (inkl. der mit ARKit 3 erscheinenden Möglichkeiten):

Xcode:

Mit Xcode können AR-Applikationen mithilfe des ARKit SDK in Objective-C oder Swift entwickelt werden. [46]–[49]

Reality Composer:

Der Reality Composer ermöglicht es dem Nutzer, auch ohne Programmierkenntnisse AR-Applikationen zu erstellen. Anschließend können die erstellten AR-Erlebnisse mit Xcode in Applikationen integriert oder für AR Quick Look exportiert werden. [46]–[49]

AR Quick Look:

AR Quick Look ermöglicht es, eine AR-Anwendung in Form einer Universal Scene Description Datei (***USDZ***) zu einer Webseite oder Applikation hinzuzufügen. Das Dateiformat mit ***USDZ***-Endung wurde extra für 3D-Modelle in AR-Anwendungen von Apple und den Pixar Animation Studios entwickelt. Ab iOS 12 kann der Nutzer das in der Datei enthaltene 3D-Objekt im Raum platzieren, durch Touchgesten verändern und mit anderen Nutzern teilen. [46]–[49]

4.1.2 ARCore

Das SDK ARCore ist die Antwort von Google auf das ARKit von Apple. Es ermöglicht Android-Entwicklern das einfache Erstellen von AR-Applikationen. Viele Geräte ab Android 7 unterstützten die Verwendung von ARCore. Damit ein Gerät ARCore nutzen kann, muss es von Google kalibriert und freigegeben wer-

den. Dies ist nötig, da z. B. die Positionierung der Sensoren bei vielen Herstellern unterschiedlich ist und so andere Werte bei den Messungen entstehen. Eine Zertifizierung von Google ermöglicht die Nutzung auf dem entsprechenden Gerät. Eine Liste mit den unterstützten Geräten bietet Google unter https://developers.google.com/ar/discover/supported-devices an. Die Liste mit kompatiblen Smartphones wird von Google stetig erweitert. Im großen Ganzen unterscheiden sich die Funktionen von ARCore nicht signifikant von Apples ARKit. [50]–[54]

Tracking:

- Tracking von Position und Ausrichtung des Geräts mithilfe von gleichzeitiger Lokalisierung und Kartierung
- Tracking von Bildern/2D-Oberflächen im Raum, auch bei bewegten Oberflächen (z.B. Werbung auf einem Bus)
- Tracking von 3D-Objekten nach vorherigem Scannen oder mit Vergleichsmodell
- Tracking von Flächen
- Speichern der Tracking-Daten, um die Sitzung zu einem späteren Zeitpunkt fortzusetzen
- Gesichtstracking mit Überlagerung von virtuellem Inhalt

Benutzerinteraktion

- Platzierung von 3D-Modellen im Raum und Interaktion mit Touchgesten
- Hit-Testing, um 3D-Positionen auf realen Oberflächen anhand eines Bildschirmpunkts zu finden und Interaktion mit virtuellen Objekten zuzulassen. ARCore kann bei einem Hit-Test auf einer schrägen Oberfläche die Neigung berechnen und die virtuellen Objekte entsprechend platzieren.

Rendering

- Rendering von Texturen, Reflektionen und Einbeziehung der Lichtverhältnisse

Multi-User

- Betriebssystemübergreifende Kommunikation mit dem Multi-User-Modus der Cloud Anchor API zwischen mehreren Android- und iOS-Geräten in Echtzeit.

ARCore für iOS ermöglicht die plattformübergreifende Erstellung von Augmented-Reality-Applikationen. So können Nutzer von Android- und iOS-Geräten eine Anwendung zusammen nutzen. Leider gilt dies nur für den Multi-User-Modus der Cloud Anchor API. Normale AR-Applikationen für iOS müssen weiterhin mit ARKit entwickelt werden. [50]–[54]

Auch bei Google gibt es einige Möglichkeiten zur Erstellung von AR-Applikationen:

Android Studio:

- Das ARCore SDK ermöglicht die Entwicklung von AR-Anwendungen mit Java oder Kotlin.
- Das ARCore SDK in Kombination mit dem Android Native Development Kit ermöglicht es dem Entwickler, Teile seiner AR-Anwendung in nativem Code wie C oder C++ zu schreiben.

Unity und Unreal:

Das ARCore SDK bietet Plug-Ins für die Entwicklungsumgebungen Unity und Unreal. Die Nutzeroberflächen der Programme ähneln gängigen 3D-Animationsprogrammen. So können auch ohne Programmierkenntnisse AR-Applikation erstellt werden. [50]–[54]

Software Development Kit	Google ARCore Version 1.10.0	Apple ARKit 3
Tracking		
Flächen	horizontal, vertikal, schräg	horizontal, vertikal
2D-Objekte	X	X
3D-Objekte	X	X
bewegte Objekte	X	-
Tracking mehrerer Bilder	X	X
Gesichtstracking	X	X
Tracking mehrerer Gesichter	-	X
Bewegungstracking	-	X
SLAM	X	X
Speichern von Trackingdaten	X	X
Benutzerinteraktion		
Interaktion über Touchscreen	X	X
Hit-Testing	X	X
Multi-User		
Multi-User gleiche Plattform	X	X
Multi-User plattformübergreifend	X	-
Rendering		
Texturen, Reflektion, Einbeziehung der Lichtverhältnisse	X	X
Verdeckung virtueller Inhalte mit realen Objekten	-	X

Tab. 1: *Gegenüberstellung von Googles ARCore und Apples ARKit*

4.2 Webbasierte AR-Technologien

Webbasierte Technologien ermöglichen dem Nutzer eine AR-Erfahrung, ohne dafür eine Applikation auf seinem Smartphone zu installieren. Der Browser muss dafür in der Lage sein AR-fähige Geräte zu erkennen, Position und Ausrichtung des Geräts abzufragen und auf die Kamera zuzugreifen.

Diese Technologie hat nicht nur Vorteile für den Nutzer. Auch der Entwickler muss nur noch eine Code-Basis schreiben, welche dann auf nahezu allen Endgeräten funktioniert.

Ein Nachteil ist jedoch die Datenmenge, die beim Aufrufen der Webseite über das mobile Internet anfällt. Der Entwickler muss deshalb besonders darauf achten, dass z. B. die Bilder und 3D-Objekte so schlank wie möglich bleiben.

Im Folgenden werden die zwei bekanntesten Programmierschnittstellen für webbasierte AR-Applikationen vorgestellt. Außerdem soll eine Bewertung des Nutzererlebnisses anhand von Beispielapplikationen stattfinden.

4.2.1 WebXR Device API

Die WebXR-Device API ist in Zusammenarbeit mehrerer großer Konzerne entstanden. Unter anderem sind Google, Microsoft und Mozilla an dem Projekt beteiligt. Zusammen bilden sie die Immersive Web Community Group. Das „X“ in WebXR steht als Variable für alle Arten von immersiven Anwendungen, unter anderem auch für Augmented Reality. Die WebXR Device API ist aus der WebVR API entstanden. Letztere war die erste Möglichkeit für Entwickler, mit VR im Browser Erfahrungen zu sammeln. Die Anforderungen haben sich jedoch insbesondere in Hinsicht auf AR geändert. Deshalb war es wichtig, die API offen für Erweiterungen zu machen. Die WebXR API soll laut der Immersive Web Community Group in Zukunft von allen gängigen Browsern als Standard implementiert werden. Außerdem soll so die Installation von Applikationen oder zusätzlichen Plug-Ins überflüssig und AR für die breite Masse zugänglich gemacht werden. Die WebXR API abstrahiert die betriebssystemspezifischen APIs ARKit und ARCore. Somit funktioniert sie aktuell nur auf Geräten, welche ARKit oder ARCore unterstützen. [55], [56], [57]

Die WebXR Device API kann horizontale und vertikale Flächen erkennen und Hit-Tests durchführen. Bei Hit-Tests wird vom Smartphone des Nutzers ein virtueller „Strahl“ ausgesendet, z. B. wenn der Nutzer auf den Bildschirm tippt. Anschließend wird der Kollisionspunkt mit der realen Welt zurückgegeben. Diese Informationen kann die API unter anderem zum Positionieren eines virtuellen 3D-Modells in der realen Welt nutzen. In Zukunft sollen weitere Funktionen hinzukommen, welche Umgebungs- und Objekttracking

inklusive Erkennung der Lichtverhältnisse ermöglichen. [55], [56], [57]

Da die WebXR Device API keine Rendering-Funktionalitäten mit sich bringt, werden diese durch WebGL APIs zur Verfügung gestellt. Was zunächst wie ein Nachteil wirkt, bringt viele Vorteile mit sich. WebGL ermöglicht den direkten Zugriff auf die Ressourcen der Grafikkarte und bietet umfangreiche Funktionen zur Erstellung von 3D-Grafiken. Da die Erstellung von 3D-Szenen in WebGL sehr komplex ist, bietet sich die Verwendung eines JavaScript Framework wie Three.js an. Die WebXR Device API unterstützt Three.js seit Mai 2018. Es wird unter anderem zur Erstellung und Anzeige von 3D-Modellen im Browser genutzt. [58], [59]

Praktischer Test

Die Nutzung der WebXR Device API funktioniert aktuell nur mit speziellen Browsern. Auf iOS wird der Mozilla XR Viewer und auf Android Chrome Canary benötigt, um AR über den Browser zu erleben. Die WebXR Device API ist voraussichtlich mit Version 76 in Chrome verfügbar. Ab dann müssen zur Nutzung nur noch die Developerflags ***#webxr***, ***#webxr-hit-test*** und ***#webxr-plane-detection*** aktiviert sein.

Beispielapplikationen werden für iOS unter https://ios-viewer.webxrexperiments.com/ und für Android unter https://immersive-web.github.io/webxr-samples/ zur Verfügung gestellt.

Der Test der Beispiele zeigt, dass die WebXR Device API zum jetzigen Zeitpunkt noch unstabil ist. Die Nutzung sollte deshalb nur zu Testzwecken erfolgen. Unter anderem werden Objekte falsch platziert oder bewegen sich bei Änderung der Smartphoneposition mit. Die WebXR Device API befindet sich noch in der Entwicklung und unterliegt aktuell starken Änderungen. Durch die notwendige Installation oben genannter Browser kann der größte Vorteil der webbasierten Technologien noch nicht genutzt werden. Sowohl aus Entwickler- als auch Konsumentensicht ist bei der WebXR Device API jedoch ein enormes Potential gegeben.

4.2.2 8th Wall Web JavaScript API

Das Start-Up 8th Wall hat das Ziel, Augmented Reality für alle Smartphones verfügbar zu machen. Deshalb arbeitet das Unternehmen unabhängig von ARCore und ARKit mit einer eigenen Implementierung. Die 8th Wall Web JavaScript API basiert auf WebGL und beinhaltet die 8th Wall SLAM Engine, welche auf die Darstellung von virtuellen Inhalten im Browser optimiert ist. Aktuell sind Funktionen wie das Tracking von sechs Freiheitsgraden (6DoF), Bildtracking, Flächentracking, Erkennung der Lichtverhältnisse und Hit-Tests verfügbar. Weitere Funktionen wie Objekt- und Gesichtstracking sollen in absehbarer Zeit ergänzt werden. Die API lässt sich auch in 3D-JS Frameworks wie A-Frame oder Three.js integrieren. [60]–[62]

Praktischer Test

Die 8th Wall Web JavaScript API kann ab iOS 11 in Safari verwendet werden. Die Nutzung anderer Browser ist auf iOS noch nicht möglich. Auf Android gibt es keine Beschränkungen hinsichtlich der Betriebssystemversion. Chrome, Firefox und auf Chrome basierende Browser, wie z. B. der Samsung Browser, unterstützen die Nutzung der 8th Wall Web JavaScript API.

Die Beispielapplikation steht unter https://apps.8thwall.com/8w/jini/ zur Verfügung.

Die 8th Wall Web JavaScript API ist der WebXR Device API um einiges voraus. Beim Test der Beispielapplikation änderten Objekte bei schnellen Bewegungen allerdings ihre Position. Eine gute Ausleuchtung der Fläche konnte diesen Effekt zwar reduzieren, jedoch nicht ganz beseitigen.

Trotzdem ist auch die 8th Wall Web JavaScript API sehr vielversprechend. So bekam sie schon während des Verlaufs der Thesis regelmäßig Updates mit Verbesserungen und neuen Funktionen.

4.3 Betriebssystemübergreifende AR-Technologien

Auch bei betriebssystemübergreifenden AR-Technologien geht es darum, mit einem Quellcode möglichst viele Endgeräte zu erreichen. Durch SDKs, wie z. B. Wikitude oder Vuforia, wird die Nutzung von AR-Applikationen auch auf Geräten mit älteren Betriebssystemversionen ohne ARCore oder ARKit möglich. Da sie über eine eigene Implementierung der AR-Funktionalitäten verfügen, sind sie nicht auf Google oder Apple angewiesen. Der Zugriff auf ARKit und ARCore ist bei vorhandener Unterstützung jedoch trotzdem möglich. Die meisten SDKs basieren dagegen ausschließlich auf ARCore und ARKit.

Zur Erstellung einer betriebssystemübergreifenden AR-Applikation werden von den SDKs unterschiedlichste Technologien zur Verfügung gestellt.[63], [64]

Ein großer Nachteil ist, dass eine betriebssystemübergreifende Applikation nicht an die Performance der nativen Konkurrenz herankommt. [65]

Dieses Kapitel soll einen Überblick über die Möglichkeiten zur Entwicklung einer betriebssystemübergreifenden AR-Anwendung geben und durch den Test von Beispielapplikationen die Leistungsfähigkeit der Frameworks bewerten.

4.3.1 AR mit Game Engines

Wenn es um die Erstellung von AR-Applikationen mit Game Engines geht, wird zwangsläufig über Unity und Unreal gesprochen. Unity ist in diesem Bereich mit einem Marktanteil von knapp 60 Prozent sehr etabliert, Unreal wird deshalb in dieser Arbeit nicht betrachtet. [66]

Unity ist eine Echtzeit-Entwicklungsplattform für 2D-, 3D-Spiele und Applikationen. Sie ermöglicht die Bereitstellung auf über 25 Plattformen wie Smartphones, Konsolen oder TVs. Die Nutzeroberfläche von Unity ähnelt dem Aufbau gängiger 3D-Animationssoftwares. So lassen sich auch mit wenig bis keinen Programmierkenntnissen AR-Applikationen erstellen. Neben dem Editor kann der Entwickler in Unity auch mit C# programmieren. Die Simulation der AR-Anwendung ist direkt in Unity möglich. So muss der Entwickler die Applikation zum Testen nicht aufwendig auf einem Gerät installieren. Außerdem lassen sich die Sensordaten vom Smartphone zum Debuggen direkt auf dem Rechner anzeigen.[67]

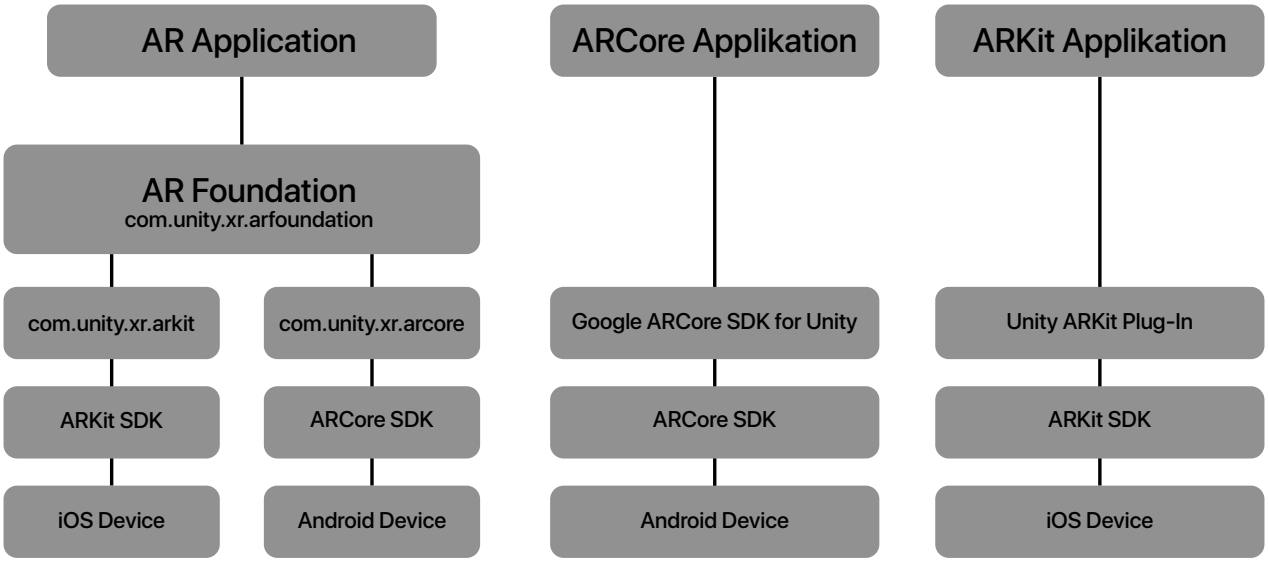

Abb. 21: *Funktion von Unity AR Foundation*

Wie im linken Teil von Abb. 21 dargestellt, bildet AR Foundation von Unity die Schnittstelle zu den plattformspezifischen SDKs ARCore und ARKit. AR-Anwendungen für iOS und Android können so mit einer Code-Basis in Unity erstellt werden. Im rechten Teil ist zu sehen, dass eine Applikation ohne AR Foundation für jedes Betriebssystem neu entwickelt werden muss. AR Foundation ermöglicht Flächentracking, SLAM, Hit-Testing, Gesichtstracking und die Erkennung der Lichtverhältnisse. Funktionen wie Bild-, Ob-

jekttracking und Cloud Anchors befinden sich in der Entwicklung. Zusätzlich ist die Ergänzung von weiteren Funktionen mithilfe von externen SDKs, wie Vuforia oder Wikitude, realisierbar. [68]–[72]

Praktischer Test

Mit Unity wurden schon viele bekannte AR-Applikationen wie z. B. Pokémon Go oder Lego AR Playgrounds erstellt. Diese dienten auch in der Thesis als Testapplikationen. Bei oben genannten Applikationen klappt das Tracking sehr gut und das 3D-Modell bleibt auch bei schnellen Bewegungen auf der festgelegten Position.

4.3.2 AR mit Cross-Plattform Frameworks

Bei Cross-Plattform Frameworks muss die AR-Funktionalität in Form von Plug-Ins hinzugefügt werden. Dies ermöglicht es, eine AR-Anwendung einmal zu schreiben und auf unterschiedliche Betriebssysteme auszuliefern. Die Code-Basis wird, wie z. B. bei den Frameworks React Native, Cordova oder Titanium, oftmals in JavaScript geschrieben. Webentwickler müssen so keine neue Programmiersprache erlernen, um eine AR-Anwendung erstellen zu können. Durch einen Großteil der Plug-Ins werden die gängigsten AR-Funktionen wie Bild-, Objekt- und Flächentracking abgedeckt. Einige Plug-Ins bieten noch weitere Funktionalitäten an. [73], [74]

Praktischer Test

Je nach verwendetem Plug-In fallen die Mindestanforderungen an das Betriebssystem zur Benutzung sehr unterschiedlich aus. Ab Android 7 und iOS 11 sollten die Apps jedoch auf jedem Smartphone nutzbar sein. Als Testapplikation wurden, die mit dem Wikitude AR SDK und Cordova erstellten AR-Applikationen, Roomle und die Washington Post Winter Olympics AR Applikation genutzt. Sie ermöglichen ein sehr genaues Tracking und eine gute Performance. In den oben genannten Applikationen bleibt das Modell auch bei schnellen Bewegungen oder Positionsänderungen am festgelegten Ort.

4.4 Fazit

Wie schon in der Einleitung dieses Kapitels erwähnt, werden native Technologien, wegen der Beschränkung auf ein Betriebssystem, nicht näher betrachtet.

Webbasierte Technologien haben ein großes Potenzial. So ist gegenüber anderen Ansätzen keine Installation einer Applikation notwendig. Des Weiteren können Updates viel schneller durchgeführt werden, da nur die serverseitige Aktualisierung der Applikation notwendig ist. Die Änderungen wirken sich clientseitig sofort aus. Beim Vertrieb der Anwendung über einen App Store ist dies nicht möglich. Durch die Bereitstellung der entsprechenden Schnittstellen von Browserherstellern und durch ausreichende Leistung der Endgeräte hat diese Technologie in den letzten zwei Jahren große Fortschritte gemacht. Die genannten APIs für Web-Anwendungen bieten jedoch noch nicht die gewünschte Performance und Qualität beim Tracking. Außerdem setzen viele Lösungen spezifische Browser voraus.

Betriebssystemübergreifende Technologien hingegen bieten eine gute Performance, umfangreiche AR-Funktionalitäten und ein sehr genaues Tracking. Hier macht sich bemerkbar, dass sie im Gegensatz zu den webbasierten Technologien schon deutlich länger auf dem Markt sind. Außerdem können diese Lösungen in einigen Fällen auch auf älteren Smartphones ohne ARCore und ARKit genutzt werden. Dies ist möglich, da sie über eine eigene Implementierung des SLAM-Algorithmus verfügen. Je nach verwendetem SDK wird zusätzlich die Nutzung der Applikation auf Augmented-Reality-Brillen oder die Verwendung einer externen Kamera für eine bessere Qualität des Videostreams unterstützt. Betriebssystemübergreifende Technologien sind auch für große Anwendungen gut geeignet. Da viele Anbieter das Hosting von Zielbildern oder -objekten in der Cloud ermöglichen, können große Mengen an Zielen erkannt werden.

Im direkten Vergleich bieten die betriebssystemübergreifenden AR-Technologien aktuell das bessere Paket zur Erstellung einer AR-Applikation. Im weiteren Verlauf der Thesis wird der Schwerpunkt deshalb auf diesen Ansatz gelegt.

5 Vergleich

SDKs für betriebssystemübergreifende Augmented Reality-Anwendungen

Einen guten Überblick über alle verfügbaren AR SDKs bietet die Webseite SocialCompare. Sie ermöglicht die Erstellung und Pflege von Vergleichstabellen durch die Community. Die Informationen von SocialCompare wurden mit anderen AR SDK-Vergleichen und eigener Recherche kombiniert und in einer Liste zusammengetragen. Diese bietet einen guten Überblick über alle SDKs für betriebssystemübergreifende AR-Anwendungen [75], [76]

5.1 Evaluationsverfahren

Nach der Analyse verschiedener Software-Evaluationsverfahren fiel die Entscheidung auf das Business Readiness Rating Model (BRR). Obwohl es als einheitlicher, offener Standard für die Evaluierung von Open-Source-Software gedacht ist, lässt es sich auch gut für den Vergleich von kommerzieller Software nutzen. Ziel dieses Modells ist es, eine schnelle, überschaubare Bewertung zu ermöglichen. Unter anderem war auch Intel an der Entwicklung beteiligt und wendet es zur Klassifizierung von Software an. [77], [78]

Im Folgenden werden die vier Phasen des BRR-Modells geschildert:

Phase 1 – Quick Assessment

Anhand festgelegter Kriterien wird hier eine Vorauswahl durchgeführt, welche es ermöglicht, die Liste der zu evaluierenden SDKs einzugrenzen. [78]

Phase 2 – Target Usage Assessment

In dieser Phase werden 12 vom BRR-Modell vorgegebene Kategorien nach Wichtigkeit bewertet. In diesem Fall wurde der Punkt Architektur durch die Punkte Entwicklungsplattformen und Betriebssysteme ersetzt, da diese Punkte bei der Entscheidung für ein SDK deutlich wichtiger sind. Anschließend folgt die Gewichtung der wichtigsten sieben Kategorien, während die restlichen laut dem BRR-Modell vernachlässigt werden können. Eine Kategorie ist in verschiedene Metriken unterteilt. Auch diese sind nach ihrer Wichtigkeit bewertet. [78]

Phase 3 – Data Collection and Processing

Die Bewertung der zuvor ausgewählten SDKs findet auf einer Skala von eins (Unakzeptabel) bis fünf (Exzellent) statt. Ist eine Einordnung auf dieser Skala nicht möglich, wird mit 1 Punkt (nicht vorhanden) oder mit 5 Punkten (vorhanden) bewertet. [78]

Phase 4 – Data Translation

Die Zusammenfassung und abschließende Bewertung der Kategorien resultiert im Business Readiness Rating Score für die einzelnen SDKs. [78]

5.2 Auswahlkriterien für SDKs (Phase 1)

Um die Auswahl an plattformübergreifenden SDKs etwas weiter einzugrenzen, wurden zwei Kriterien definiert. Nur wenn diese erfüllt sind, ist das SDK für den Vergleich relevant.

Plattformübergreifende SDKs	**NFT**	**SLAM**
Vuforia Engine v8.3 [82], [83]	X	X
Kudan AR SDK v1.6.0 [84]	X	X
MAXST AR SDK v4.1.3 [85]	X	X
EasyAR SDK v3.0 [87]	X	X
Wikitude AR SDK v8.5 [63], [89]	X	X
Onirix SDKs v1.1.0 [90]–[95]	X	X
ViroReact v2.15.0 [97]	X	X
artoolkitX v1.0.5.1 [98]	X	-
Catchoom AR SDK v2.4.13 [99]	X	-
DroidAR v2 [100]	X	-
blippAR AR SDK [103]	X	-
ARmedia SDK v2.1.0 [88]	X	-
FLARToolKit v4.1.0 [101]	-	-
IN2AR SDK [86]	-	-
XZIMG Augmented Vision [96]	-	-
Pikkart AR SDK v3.5.8 [102]	-	-

Tab. 2: *Vergleich von Cross-Plattform SDKs anhand der festgelegten Auswahlkriterien*

- Natural Feature Tracking (NFT): NFT ermöglicht die Erkennung von Bildern oder Objekten ohne Marker. Hierzu wird meist ein Referenzbild oder -objekt analysiert und eine Karte mit den Schlüsselpunkten gespeichert. Diese wird anschließend mit dem echten Kamerabild verglichen. Ist das Objekt bekannt, können die verknüpften, virtuellen Inhalte geladen werden. [9], [79]

- Simultaneous Localization and Mapping (SLAM): SLAM ermöglicht dem mobilen Endgerät eine Karte seiner Umgebung zu erstellen und gleichzeitig seine Orientierung und Position innerhalb dieser Karte festzulegen. [41], [80], [81]

5.3 Überblick ausgewählter SDKs (Phase 2)

Der erste Abschnitt des Kapitels stellt die ausgewählten SDKs und deren Lizenzmodelle vor. Bei Abonnements der Lizenzen geben die meisten Anbieter ihre Preise pro Jahr an. Damit ein schneller Vergleich möglich ist, werden die Preise bei monatlicher Abrechnung auf ein Jahr hochgerechnet und falls nötig von Dollar in Euro umgerechnet (Wechselkurs Stand 9.8.2019 - 1 Euro = 1,12 Dollar). Im Anschluss erfolgt eine theoretische Bewertung der SDKs anhand der folgenden Kriterien.

Grad	Kategorie	Gewichtung
1	Funktionalität	25%
2	Entwicklungsplattformen	20%
3	Betriebssysteme	20%
4	Dokumentation	15%
5	Support	10%
6	Community	5%
7	Benutzerfreundlichkeit	5%
8	Verbreitung	-
9	Qualität	-
10	Sicherheit	-
11	Skalierbarkeit	-
12	Performance	-
13	Professionalität	-

Tab. 3: *Kriterien mit Gewichtung zur Bewertung der SDKs*

Da die vollen Namen der SDKs meist sehr lange sind, werden diese im Folgenden mit dem Firmennamen abgekürzt. Die Gewichtung der Kategorien wurde maßgeblich durch die Zielsetzung beeinflusst. Dadurch liegt der Fokus auf der Funktionalität und den zur Verfügung stehenden Entwicklungsplattformen und Betriebssystemen. Neben diesen Aspekten wird auf eine ausführliche Dokumentation, guten Support, die Community und die Benutzerfreundlichkeit Wert gelegt. Die Kategorien 8-13 wurden nicht betrachtet. Dies liegt zum einen an fehlender Relevanz für die Zielsetzung und zum anderen daran, dass manche Kategorien nur nach einem praktischen Test richtig bewertet werden können. Die Bewertung findet aus der Sicht eines Entwicklers statt.

5.3.1 Wikitude AR SDK

Das Wikitude AR SDK wird von der Wikitude GmbH in Salzburg, Österreich entwickelt. Die erste Version erschien im Jahr 2008. Wikitude ist seitdem zu einer der weltweit führenden, unabhängigen AR-Plattformen auf Smartphones, Tablets und AR-Brillen aufgestiegen. Die Bewertung findet anhand der Wikitude AR SDK Version 8.5 statt. [104]

Die Lizenzmodelle von Wikitude ermöglichen eine Einmalzahlung oder ein Abonnement. Im Gegensatz zur Einmalzahlung bekommt der Nutzer bei Abschluss eines Abonnements auch die Updates des SDKs zur Verfügung gestellt. Das Abonnement wird jährlich abgerechnet. Die Start-up-Lizenz ist an bestimmte Bedingungen geknüpft, welche unter https://www.wikitude.com/product/sdk-startup/ genauer nachzulesen sind. [105]

Lizenzmodelle	**Pro**	**Pro 3D**	**Cloud**	**Trial**	**Start-up**
Geotracking	X	X	X	X	X
Bildtracking	X	X	X	X	X
Objekt- und Umgebungstracking	-	X	X	X	X
SLAM	-	X	X	X	X
Clouderkennung	-	-	X	-	-
Preis					
Einmalzahlung	1990€	2490€	-	Kostenlos, mit Wasserzeichen	Kostenlos
Abonnement	1990€ pro Jahr	2490€ pro Jahr	4490€ pro Jahr		

Tab. 4: *Lizenzmodelle Wikitude AR SDK* *Quelle: [105]*

5.3.2 ViroReact

Viro Media ist ein Start-up aus Seattle, Canada. Die Viro Plattform ermöglicht die Entwicklung von Augmented- oder Virtual Reality-Anwendungen mit den Plattformen ViroReact und ViroCore. Bei ViroReact nutzt der Entwickler React Native zur Erstellung von plattformübergreifenden Applikationen. ViroCore soll in dieser Thesis nicht genauer betrachtet werden, da es nur die Entwicklung für Android ermöglicht. Die Bewertung wird anhand der Version 2.15.0 von Viro React durchgeführt.[106]

ViroReact ist kostenlos für alle Nutzer. Allerdings muss Viro Media als Hersteller erwähnt werden. Dies geschieht über die Einbindung des Logos, welches entweder auf dem Ladebildschirm oder innerhalb der Applikation an einer gut sichtbaren Stelle platziert werden sollte. Das SDK unterstützt Markertracking, Bildtracking und SLAM. [107]

5.3.3 Vuforia Engine

Vuforia Engine wird von dem Softwareunternehmen PTC aus Boston, USA entwickelt. Seit PTC Vuforia im November 2015 von Qualcomm gekauft hat, ist es zum Marktführer im Bereich Augmented Reality aufgestiegen. Die Bewertung wird anhand der Version 8.3 des Vuforia Engines durchgeführt. [108]

Vuforia bietet nur das Abonnement von Lizenzen an. Das Abonnement wird pro Monat abgerechnet. Alle Lizenzmodelle enthalten SDK Updates. [109]

Lizenzmodelle	**Trial**	**Basic**	**Basic und Cloud**	**Pro**
Markertracking	X	X	X	X
Bildtracking	X	X	X	X
Objekt- und Umgebungstracking	X	X	X	X
Zylindertracking	X	X	X	X
SLAM	X	X	X	X
Clouderkennung	-	-	X	X
Externe Kamera	X	-	-	X
Preis	Kostenlos, mit Wasserzeichen	37€ pro Monat (450€ pro Jahr)	88€ pro Monat (1062€ pro Jahr)	kundenspezifischer Preis

Tab. 5: *Lizenzmodelle Vuforia Engine* *Quelle: [109]*

5.3.4 Onirix SDKs

Die Onirix SDKs werden von der Softwarefirma Neosentec aus Asturias, Spanien entwickelt. Die Funktionen sind auf drei SDKs (Targets, Places und Spaces) verteilt. Je nach Anwendungsfall muss also das SDK mit der entsprechenden Funktionalität genutzt werden. Die Bewertung wird anhand der Version 1.1.0 der Onirix SDKs durchgeführt. [110]

Alle Abonnements werden pro Monat abgerechnet. Nur die Education-Lizenz kann wahlweise pro Monat oder pro Jahr gekauft werden. Um die Education-Lizenz nutzen zu können, muss der Entwickler entweder Professor oder eingeschriebener Student an einer Bildungseinrichtung sein. [111]

Lizenzmodelle	**Onirix Eduaction**	**Onirix Abo + Add-On alle SDKs**
Markertracking	X	X
Bildtracking	X	X
Geotracking	X	X
SLAM	X	X
Preis	9€ pro Monat oder 99€ pro Jahr	128€ pro Monat (1.536€ pro Jahr)

Lizenzmodelle Onirix SDKs *Quelle: [111]*

5.3.5 Kudan AR SDK

Kudan Limited bezeichnet sich selbst als Technologielabor für Bilderfassungsalgorithmen. Der Hauptsitz der Firma ist in Tokyo, Japan. Die Kudan AR SDK ermöglicht es umfangreiche AR-Anwendungen zu erstellen. Außerdem können die Applikationen durch die hohe Konfigurierbarkeit von Kudan individuell an die Bedürfnisse angepasst werden. Die Bewertung wird anhand der Kudan AR SDK Version 1.6.0 durchgeführt. [112]

Bei allen Lizenzmodellen von Kudan ist der volle Funktionsumfang des SDKs enthalten. Bei der Enterprise-Lizenz ist zusätzlich kostenloser Support enthalten. Das Abonnement wird jährlich abgerechnet. Die Lizenzen sind an den mit der Applikation erwirtschafteten Gewinn geknüpft. Genauere Informationen finden sich unter https://www.xlsoft.com/en/products/kudan/price.html.

Lizenzmodelle	AR Indie	AR Business	AR Enterprise
Markertracking	X	X	X
Bildtracking	X	X	X
SLAM	X	X	X
Preis	Kostenlos, mit Wasserzeichen	1341€ pro Jahr	Kundenspezifischer Preis

Tab. 6: *Lizenzmodelle Kudan AR SDK* *Quelle: [113]*

5.3.6 MAXST AR SDK

MAXST aus Seoul, Korea wurde 2010 gegründet und konzentriert seine Forschung und Entwicklung auf den Bereich Augmented Reality. Neben der MAXST AR SDK gibt es noch weitere Produkte wie den AR Guide, welcher das Erstellen von AR-Anleitungen ermöglicht oder AR Collaboration, welches Videoanrufe mit virtuellen Inhalten überlagern lässt. Die Bewertung wird anhand der MAXST AR SDK Version 4.1.3 durchgeführt. [114], [115]

MAXST bietet sowohl Einmalzahlung als auch Abonnements zum Erwerb des SDKs an. Bei allen Abonnement-Lizenzen (Pro-Subscription und Enterprise) sind SDK-Updates im Paket enthalten. Es wird jährlich abgerechnet. Die Enterprise-Lizenz bietet zusätzlich noch kostenlosen Support und Code-Reviews durch MAXST-Mitarbeiter. Die separaten Lizenzmodelle zur Nutzung der cloudbasierten Erkennung von Bildern sind in Tabelle 7 aufgeführt. [116]

Lizenzmodelle cloudbasierter Erkennung	Trial	Pro	Enterprise
Anzahl Zielbilder	1000	10.000	Keine Begrenzung
Anzahl Erkennungen pro Monat	1000	300.000	Keine Begrenzung
Preis	0€	35€ pro Monat (418€ pro Jahr)	kundenspezifischer Preis

Tab. 7: *Lizenzmodelle MAXST AR SDK* *Quelle: [116]*

Lizenzmodelle	Trial	Pro-One	Pro-Subscription	Enterprise
Markertracking	X	X	X	X
Bildtracking	X	X	X	X
Objekttracking	X	X	X	X
QR-Codetracking	X	X	X	X
QR-Codereading	X	X	X	X
SLAM	X	X	X	X
Clouderkennung möglich	-	-	X	X
Plattformen	Unity, Android, iOS	Unity, Andro-id, iOS	Unity, Android, iOS	Unity, Android, iOS, Windows, macOS, Augmented-Reali-ty-Glasses
Preis	Kostenlos, mit Wasserzeichen	Einmalig 446€	535€ pro Jahr	kundenspezifischer Preis

Tab. 8: *cloudbasierte Erkennung MAXST AR SDK* *Quelle: [117]*

5.3.7 EasyAR AR SDK

Das EasyAR AR SDK wird von der Firma VisionStar Information Technology aus Shanghai, China entwickelt. Seit der Veröffentlichung im Jahr 2015 haben über 30.000 Entwickler das EasyAR AR SDK für ihre Augmented Reality-Applikationen genutzt. Die Bewertung wird anhand der EasyAR AR SDK Version 3.0 durchgeführt. [118], [119]

Der Erwerb einer EasyAR-Lizenz ist nur per Einmalzahlung möglich. Diese enthält kostenlose Upgrades, bis zu einem neuen Major-Update des SDKs. Zur Nutzung der cloudbasierten Erkennung von Bildern muss ein separates Abonnement abgeschlossen werden. Die unterschiedlichen Lizenzen sind in Tabelle 9 gelistet. Sie werden pro Monat abgerechnet. [87]

Lizenzmodelle cloudbasierter Erkennung	Trial	Pro	Enterprise
Anzahl Zielbilder	100.000	100.000	100.000
Anzahl Erkennungen pro Tag	500	10.000	20.000
Preis	Kostenlos für einen Monat	79€ pro Monat (954€ pro Jahr)	159€ pro Monat (1910€ pro Jahr)

Tab. 9: *Lizenzmodelle cloudbasierte Erkennung EasyAR AR SDK* *Quelle: [120]*

Lizenzmodelle	Basic	Pro
QR-Codereading	X	X
Bildtracking	X	X
Objekttracking	-	X
SLAM	X	X
Clouderkennung möglich	X	X
Aufnahmefunktion für Videos	-	X
Externe Kamera	X	X
Preis	Kostenlos	Einmalig 1341€

Tab. 10: *Lizenzmodelle EasyAR AR SDK* *Quelle: [87]*

5.4 Gegenüberstellung (Phase 3)

Die im letzten Abschnitt vorgestellten SKDs sollen nun anhand der festgelegten Kategorien bewertet werden. Die in diesem Abschnitt verwendeten Fachbegriffe wie z. B. SLAM oder Bildtracking wurden bereits in Abschnitt 3.2.2 erläutert.

Nach der Bewertung der festgelegten Metriken, werden zusätzlich noch Alleinstellungsmerkmale der einzelnen SDKs aufgelistet.

5.4.1 Funktion

Tracking

Laut Zielsetzung sind beim Tracking die Verfügbarkeit von Simultaneous Localization and Mapping (SLAM) und die Unterstützung von ARCore und ARKit am wichtigsten. Da sie die freie Platzierung von virtuellen Inhalten in der Umgebung des Nutzers ermöglichen, wurden die beiden Metriken am stärksten gewichtet. Insgesamt erhält Vuforia mit 4,7 Punkten die höchste Bewertung, danach folgen Wikitude mit 4,6 Punkten und Onirix mit 3,9 Punkten.

Eine Kombination aus SLAM und ARCore/ARKit-Support wird von Wikitude, ViroReact, Vuforia und Onirix angeboten. [63], [64], [94], [95], [97]

Metrik	Gewichtung in %	Wikitude AR SDK	Viro-React	Vuforia Engine	MAXST AR SDK	Onirix SDKs	Kudan AR SDK	EasyAR SDK
1.1 Marker	5	1	5	5	5	5	5	5
1.2 Bilder	10	5	5	5	5	5	5	5
1.3 3D-Objekte	10	5	1	5	5	1	1	5
1.4 Tracking von mehreren Bildern	10	5	1	5	1	1	5	5
1.5 Gesichts-tracking	5	3	1	1	1	1	1	1
1.6 SLAM	30	5	5	5	5	5	5	5
1.7 ARKit/ARCore Support	25	5	5	5	1	5	1	1
1.8 weitere Funk-tionen	5	3	1	3	3	3	1	2
Bewertung in Punkten		4,6	3,8	4,7	3,3	3,9	3,2	3,65

Tab. 11: *Bewertung der Funktionalität des Trackings* *Quellen: [40], [63], [82]–[85], [87], [89], [90], [91]–[95], [97], [121]–[124], [64], [125]–[133]*

Zur Verarbeitung von großen Mengen an Zielbildern, bieten Wikitude, Vuforia, MAXST und EasyAR die Bilderkennung in der Cloud an. Diese Option ist meistens in den Enterprise-Lizenzmodellen enthalten oder kann extra gebucht werden. [85], [87], [131], [133]

Wikitude und Onirix können die Position des Smartphones per GPS bestimmen. Abhängig davon, ob sich der Nutzer in einem Gebäude oder in der Natur befindet, kann Wikitude sogar zwischen GPS-Signal, Netzwerk oder Beacon (Bluetooth-Sender, welcher regelmäßig Pakete sendet) wechseln. [91],[131]

Das Scannen und Auslesen von QR-Codes in der Applikation wird von MAXST und EasyAR unterstützt. So können z. B. Webseiten direkt in der Applikation geladen werden. [85], [87]

Vuforia und MAXST ermöglichen dem Entwickler die Nutzung einer externen Kamera. Diese sorgt für eine bessere Qualität des Kamerabilds und die Möglichkeit, die Anwendung auch auf einem Gerät ohne Kamera zu nutzen. [85], [132]

Neben normalen Bildern kann Vuforia mit dem Zylindertracking auch Bilder auf zylindrischen Objekten, wie z. B. einer Flasche tracken. Außerdem lassen sich in speziellen Markern (VUMarks), mithilfe von minimalen Änderungen am Design, eindeutige Identifikationsnummer codieren. Diese Technik gestattet dem Entwickler die Referenzierung von unterschiedlichen Inhalten bei gleichbleibendem Design. [129], [130]

EasyAR bietet mit einer Aufnahmefunktion die Möglichkeit ein Video von den platzierten, virtuellen Inhalten zu erstellen. [87]

Ein Plug-In ermöglicht Wikitude als einzigem SDK das Tracking von Gesichtern. Die Verwendung des Plug-Ins funktioniert nur mit der Java API. Zur Gesichtserkennung wird OpenCV, eine Open-Source-Bibliothek für Bildverarbeitungsalgorithmen, eingesetzt. [134]

Rendering

Da die Platzierung von 3D-Modellen unterstützt werden muss, wurde das Rendering von 3D-Inhalten am stärksten gewichtet. Am besten schneidet ViroReact mit 4,7 Punkten ab. Wikitude und Onirix teilen sich mit 4,4 Punkten den zweiten Platz.

Metrik	Gewichtung in %	Wikitude AR SDK	Viro-React	Vuforia Engine	MAXST AR SDK	Onirix SDKs	Kudan AR SDK	EasyAR SDK
1.1 HTML-Inhalte	5	5	1	1	1	1	1	1
1.2 2D- Inhalte	10	5	5	5	5	5	5	5
1.3 3D- Inhalte	40	5	5	5	5	5	5	5
1.4 Animationen	20	5	5	5	5	5	5	1
1.5 Ton	5	5	5	1	1	5	1	1
1.6 realistische Licht- und Schattensetzung	10	1	5	1	1	5	5	1
1.7 weitere Funktionen	10	3	4	4	2	1	2	1
Bewertung in Punkten		4,4	4,7	4,1	3,9	4,4	4,3	3

Tab. 12: *Bewertung der Funktionalität des Renderings* *Quellen: [40], [63], [64], [82]–[85], [87], [89], [90], [91]–[95], [97], [121]–[124], [125]–[133]*

Während Vuforia und MAXST OpenGL als Render Engine nutzen, setzen die anderen SDKs auf eine eigene Implementierung zum Rendern der AR-Ansicht. Bei Kudan kann der Entwickler selbst entscheiden, welcher Render Engine verwendet werden soll. [84], [85], [87], [90], [92], [94], [135]–[138]

Die Verdeckung von virtuellen Inhalten durch reale Objekte (engl. Occlusion) ermöglichen aktuell nur Vuforia, MAXST AR und Kudan AR. [84], [85], [139]

ViroReacts Physik- und Partikel-Engine bietet dem Entwickler Zugriff auf Effekte wie Schwerkraft oder Rauch. [97]

Mit EasyAR und ViroReact können auch 360°-Videos in die AR-Anwendung eingebunden werden. [140]

Kudan unterstützt Maps und Shader. Mit Echtzeit-Textur-Morphing ist dadurch auch die Änderung von Materialien bei bereits platzierten 3D-Inhalten möglich. [84]

Vuforia erlaubt die Verwendung von Oberflächen als virtuelle Buttons. So kann der Nutzer z. B. per Touchgeste auf einen Tisch bestimmte Aktion auslösen. [139]

Fasst man nun die Funktionalitäten Rendering und Tracking zusammen, erhält Wikitude mit 4,5 Punkten die höchste Bewertung. Vuforia und ViroReact folgen mit 4,4 bzw. 4,25 Punkten.

Funktionalität Durchschnitt	Wikitude AR SDK	Viro-React	Vuforia Engine	MAXST AR SDK	Onirix SDKs	Kudan AR SDK	EasyAR SDK
Bewertung in Punkten	4,5	4,25	4,4	3,6	4,15	3,55	3,325

Tab. 13: *Bewertung der Funktionalität der SDKs*

5.4.2 Entwicklungsplattformen

Die Metriken 2.4-2.6 wurden am stärksten gewichtet, da sie eine plattformübergreifende Entwicklung ermöglichen. Wikitude bietet 12 Entwicklungsplattformen und erzielt mit 5 Punkten das beste Ergebnis. Vuforia belegt mit 3,2 Punkten den zweiten Platz, gefolgt von Kudan und EasyAR.

Metrik	Gewichtung in %	Wikitude AR SDK	Viro-React	Vuforia Engine	MAXST AR SDK	Onirix SDKs	Kudan AR SDK	EasyAR SDK
2.1 Android	10	5	1	5	5	5	5	5
2.2 iOS	10	5	1	5	5	5	5	5
2.3 UWP	5	5	1	5	1	1	1	1
2.4 Unity Plug-In	25	5	1	5	5	5	5	5
2.5 React Native	25	5	5	1	1	1	1	1
2.6 Cordova Plug-In	20	5	1	1	1	1	1	1
2.7 weitere Plattformen	5	5	1	5	1	1	5	5
Bewertung in Punkten		5	2	3,2	2,8	2,8	3	3

Tab. 14: *Von den SDKs zur Verfügung gestellte Entwicklungsplattformen* *Quellen: [87], [97], [141]–[145]*

Für die Entwicklung unter Android stellen alle SDKs eine Java API zur Verfügung. Bei iOS wird im Fall von Kudan, Onirix, MAXST, EasyAR und Wikitude eine Objective-C API und bei Vuforia eine C++ API verwendet. Wikitude, Vuforia, Kudan, MAXST und EasyAR bieten zusätzlich eine C++ API an, die z. B. für die Entwicklung von Anwendungen für Augmented-Reality-Brillen genutzt werden kann. [87], [97], [141]–[145]

Durch eine JavaScript API und offizielle Plug-Ins ermöglicht Wikitude zusätzlich die Entwicklung von Anwendungen mit Cordova, Xamarin und Titanium. Durch Drittanbieter-Plug-Ins werden auch React Native, Ionic, Adobe Air und Qt by Felgo unterstützt. [145]

5.4.3 Betriebssysteme

Die SDKs sollten auf den gängigsten Betriebssystemen für mobile Endgeräte funktionieren. Android und iOS sind deshalb am stärksten gewichtet. UWP ist wegen geringer Nutzerzahlen nur schwach gewichtet. Die Metriken 3.4-3.6 sind nach der Zielsetzung nicht relevant. Mit 4,8 Punkten haben MAXST und EasyAR das beste Ergebnis. Kurz danach folgen Vuforia und Wikitude mit 4,6 Punkten.

Metrik	Gewichtung in %	Wikitude AR SDK	Viro-React	Vuforia Engine	MAXST AR SDK	Onirix SDKs	Kudan AR SDK	EasyAR SDK
3.1 Android	40	5	5	5	5	5	5	5
3.2 iOS	40	5	5	5	5	5	5	5
3.3 UWP	5	5	1	5	1	1	1	1
3.4 Windows	5	1	1	1	5	1	1	5
3.5 macOS	5	1	1	1	5	1	1	5
3.6 weitere Systeme	5	5	1	5	5	1	5	5
Bewertung in Punkten		4,6	4,2	4,6	4,8	4,2	4,4	4,8

Tab. 15: *Von den SDKs zur Verfügung gestellte Betriebssysteme* *Quellen: [87], [97], [141]–[145]*

Um eine AR-Applikation auf Windows und macOS nutzen zu können, ist der Einsatz einer externen Kamera nötig. Von den ausgewählten SDKs unterstützten dies MAXST und EasyAR. [85], [87], [141]

Mit Wikitude, Vuforia, Kudan, MAXST und EasyAR können auch AR-Applikationen für Augmented-Reality-Brillen entwickelt werden. [84], [87], [141], [144], [145]

5.4.4 Dokumentation

Bei der Verwendung eines neuen SDKs ist es für den Entwickler sehr wichtig, dass er sich schnell zurechtfindet. Demoanwendungen erleichtern den Einstieg dabei deutlich. Auch eine ausführliche Dokumentation und API-Referenz sind wichtig. Deshalb sind die Metriken 4.1-4.3 am stärksten gewichtet worden. Wikitude erhält mit 5 Punkten die beste Bewertung. ViroReact und Vuforia teilen sich mit 4,65 Punkten den zweiten Platz.

Metrik	Gewichtung in %	Wikitude AR SDK	Viro-React	Vuforia Engine	MAXST AR SDK	Onirix SDKs	Kudan AR SDK	EasyAR SDK
4.1 Demoanwendungen verfügbar	35	5	5	4	1	2	2	4
4.2 Dokumentation der Funktionalität	35	5	4	5	4	2	4	3
4.3 API Referenz	20	5	5	5	5	5	5	5
4.4 Installations- und Schnellstart-Anleitung	10	5	5	5	5	1	5	5
Bewertung in Punkten		5	4,65	4,65	3,25	2,5	3,6	3,95

Tab. 16: *Bewertung der Dokumentation der SDKs* *Quellen: [121], [146]–[151]*

Alle SDKs stellen Demoanwendungen zur Verfügung. Leider decken diese nur in wenigen Fällen alle Funktionen ab. Wikitude bietet mit 58 Code-Beispielen die umfangreichste Auswahl. Auch ViroReact, EasyAR und Vuforia schneiden sehr gut ab. [121], [146]–[151]

5.4.5 Support

Beim Support sind die Metriken 5.1-5.3 am wichtigsten und dementsprechend gewichtet. Neben der Verfügbarkeit wurde auch darauf geachtet, ob der Support kostenfrei oder kostenpflichtig ist. Ist die Metrik vorhanden und kostenfrei, wird mit 5 Punkten bewertet, ist sie kostenpflichtig mit 3 Punkten. Wenn sie nicht vorhanden ist gibt es einen Punkt. Am besten hat Onirix mit 5 Punkten abgeschnitten, da alle Metriken kostenlos angeboten werden. Es folgt ViroReact mit 3,8 Punkten und Wikitude 3,6 Punkten.

Metrik	Gewichtung in %	Wikitude AR SDK	Viro-React	Vuforia Engine	MAXST AR SDK	Onirix SDKs	Kudan AR SDK	EasyAR SDK
5.1 FAQ	30	5	5	5	5	5	5	5
5.2 E-Mail	40	3	5	3	3	5	3	5
5.3 Telefon	20	3	1	3	3	5	3	5
5.4 Entwickler-training	10	3	1	1	1	5	1	1
Bewertung in Punkten		3,6	3,8	3,4	3,4	5	3,4	4,6

Tab. 17: *Bewertung des Supports der SDKs* *Quellen: [152]–[160]*

Das MAXST Enterprise-Abonnement bietet dem Entwickler zusätzlichen Support per Videochat und Code-Reviews durch MAXST-Mitarbeiter. [156]

EasyAR bietet regelmäßig kostenlose Live-Tutorials an. Darin erklären EasyAR-Mitarbeiter z. B. neue Funktionen des SDKs. [158]

5.4.6 Community

Die Foren von Vuforia, Wikitude, ViroReact, MAXST und EasyAR bekommen 5 Punkte, da die Community aktiv ist und pro Woche mehrere Beiträge im Forum hinzukommen. Im Forum von Kudan werden nur unregelmäßig Beiträge gepostet, daher gibt es nur 3 Punkte. Onirix hat als einziges SDK kein Forum.

Metrik	Gewichtung in %	Wikitude AR SDK	Viro-React	Vuforia Engine	MAXST AR SDK	Onirix SDKs	Kudan AR SDK	EasyAR SDK
6.1 Forum verfügbar	100	5	5	5	5	1	3	5
Bewertung in Punkten		5	5	5	5	1	3	5

Tab. 18: *Bewertung der Community der SDKs* *Quellen: [152]–[160]*

Die größten Communitys sind Vuforia mit über 500.000 und Wikitude mit über 125.000 Entwicklern. [104], [108]

Die Viro-Entwickler-Community hat im Instant-Messaging-Dienst Slack eine Gruppe, der jeder beitreten kann. [159]

5.4.7 Benutzerfreundlichkeit

Benutzerfreundlichkeit ist ein wichtiger Punkt bei der Entscheidung für ein SDK. Metrik 7.1 und 7.2 wurden anhand des Tiobe-Index unter https://www.tiobe.com/tiobe-index/ und dem Stack Overflow Survey 2019 unter https://insights.stackoverflow.com/survey/2019 bewertet, welche die Popularität von Programmiersprachen wiedergeben. Wikitude hat mit 4,4 Punkten die höchste Bewertung, knapp gefolgt von ViroReact mit 4,3 Punkten. Auf dem dritten Platz liegt Kudan mit 3,5 Punkten.

Metrik	Gewichtung in %	Wikitude AR SDK	Viro-React	Vuforia Engine	MAXST AR SDK	Onirix SDKs	Kudan AR SDK	EasyAR SDK
7.1 einfacher Einstieg in Programmierung des SDKs	35	5	5	3	3	3	3	3
7.2 Unterstützung von gängigen Programmiersprachen	35	5	3	3	3	3	4	3
7.3 Unterstützung von gängigen Dateiformaten für 3D-Objekte	15	2	5	5	5	2	4	2
7.4 Debugging	15	4	5	3	3	3	3	3
Bewertung in Punkten		4,4	4,3	3,3	3,3	2,85	3,5	2,85

Tab. 19: *Bewertung der Benutzerfreundlichkeit der SDKs* *Quellen: [85], [87], [97], [135], [136], [141]–[145], [161], [162], [163]*

Bei Metrik 7.1 schneiden vor allem Wikitude und ViroReact mit 5 Punkten gut ab, da beide die Entwicklung in JavaScript bzw. JSX ermöglichen. So können sie z. B. auch von Webentwicklern ohne Einarbeitungszeit genutzt werden. [97], [145]

Um einen einfachen Start bei der Entwicklung zu gewährleisten, ist die Unterstützung von gängigen Programmiersprachen sehr wichtig. So bieten alle SDKs eine Auswahl mit z. B. C++ oder Java an. [87], [97], [141]–[145]

Mit Unterstützung von mindestens drei gängigen Dateiformaten für 3D-Inhalte (z. B. obj, fbx oder gltf) schneiden ViroReact, Vuforia, Kudan und MAXST gut ab.[85], [135], [136], [161]

Professionelle Debuggingfunktionen gibt es nur bei Wikitude und ViroReact. [162], [163]

5.5 Gesamtbewertung (Phase 4)

Abschließend werden bei jedem SDK die Kategorien entsprechend der festgelegten Gewichtung zusammengerechnet. Das Ergebnis nennt sich Business Readiness Rating Score. Wikitude hat mit 4,63 von 5 Punkten einen großen Vorsprung vor den anderen SDKs. Dies ist vor allem auf die vielen Funktionen und die große Auswahl an Entwicklungsplattformen und Ausgabesystemen zurückzuführen. Auf dem zweiten Platz liegt Vuforia mit 4,11 Punkten. Dahinter kommt ViroReact mit 3,85 Punkten.

Metrik	**Gewichtung in %**	**Wikitude AR SDK**	**Viro-React**	**Vuforia Engine**	**MAXST AR SDK**	**Onirix SDKs**	**Kudan AR SDK**	**EasyAR SDK**
1.Funktionalität	25	1,13	1,06	1,10	0,90	1,04	0,94	0,83
2.Entwicklungs-plattform	20	1,00	0,40	0,64	0,56	0,56	0,56	0,60
3.Ausgabesys-teme	20	0,92	0,84	0,92	0,96	0,84	0,88	0,92
4.Dokumentation	15	0,75	0,70	0,70	0,49	0,38	0,54	0,59
5.Support	10	0,36	0,38	0,34	0,34	0,50	0,34	0,46
6.Community	5	0,25	0,25	0,25	0,25	0,05	0,25	0,25
7.Benutzer-freundlichkeit	5	0,22	0,22	0,17	0,17	0,14	0,18	0,14
Bewertung in Punkten		4,63	3,85	4,11	3,66	3,51	3,68	3,84

Tab. 20: *Von den SDKs zur Verfügung gestellte Entwicklungsplattformen* *Quellen: [87], [97], [141]–[145]*

6 MVP mit ausgewählten SDKs

Mit Wikitude, Vuforia und ViroReact soll nun die Umsetzung eines minimalistischen Anwendungsfalls erfolgen, welcher die Platzierung eines 3D-Objekts in der Umgebung des Nutzers ermöglicht. Das Objekt soll auch bei einer Positionsänderung des Smartphones an der festgelegten Position bleiben.

Im ersten Unterpunkt wird neben der Installation auch auf die Voraussetzungen für diese eingegangen. Danach folgt die Erläuterung der wichtigsten Aspekte bei der Implementierung der AR-Logik. Abschließend soll eine Bewertung der Funktionalität des MVPs stattfinden.

Um betriebssystemübergreifende Applikationen zu implementieren werden im Folgenden verschiedene Ansätze verwendet. Sie alle haben das Ziel, die Applikation auf den Betriebssysteme iOS und Android zu installieren. Bei Wikitude wird das Cordova-Plug-In verwendet, welches auf der Wikitude JavaScript API basiert. Bei ViroReact wird React Native und bei Vuforia das Unity-Plug-In zur Erstellung der Anwendung genutzt. So ist ein Vergleich aller Ansätze zur Erstellung von betriebssystemübergreifenden AR-Applikationen möglich.

6.1 Wikitude AR SDK

6.1.1 Installation

Um mit dem Cordova-Plug-In von Wikitude arbeiten zu können, müssen die Cordova CLI, Xcode und das Android SDK installiert sein. Diese werden kostenlos auf der jeweiligen Webseite zum Download angeboten. Dabei gilt es zu beachten, dass die Erstellung von Applikationen für iOS nur auf macOS möglich ist. Des Weiteren wird ein Lizenzschlüssel von Wikitude benötigt. Dieser ist nach einer Registrierung auf der Webseite von Wikitude kostenlos erhältlich. Mit diesem Lizenzschlüssel hat der Entwickler unbegrenzten Zugriff auf alle Funktionen der Wikitude AR SDK. Um das angezeigte Wasserzeichen von Wikitude zu entfernen, muss jedoch eine kostenpflichtige Lizenz erworben werden.

Der erste Schritt ist die Erstellung eines neuen Projekts mit der Cordova CLI. Zu diesem Projekt müssen die Plattformen iOS und Android hinzugefügt und anschließend das Wikitude Cordova Plug-In von Github installiert werden. Nach der Installation folgt der Eintrag des Lizenzschlüssel im Wikitude-Plug-In. Wikitude stellt auf Github eine umfangreiche Anwendung zur Verfügung, welche Beispielcode für alle verfügbaren Funktionen enthält.

Die Installation von Xcode und Android Studio nimmt ca. 20 Minuten in Anspruch. Das anschließende Aufsetzen eines Cordova-Projekts mit dem Wikitude-Plug-In dauert 15 Minuten.

6.1.2 Implementierung

Die von Wikitude auf Github bereitgestellte Applikation deckt mit dem Projekt „Instant Tracking – Simple Instant Tracking“ die Anforderungen der Testanwendung bereits ab. Im Folgenden wird trotzdem kurz auf die, für den Anwendungsfall relevante, Implementierung eingegangen.

Als erstes wird eine Instanz der Klasse ***AR.InstantTracker*** angelegt. Diese ist für die Positionierung im Raum verantwortlich. An den ***AR.InstantTracker*** wird ein ***AR.InstantTrackable*** angehängt. Dieses repräsentiert hierbei ein virtuelles Objekt, das an die Position des Trackers gebunden ist. Zum Schluss wird noch ein ***AR.Model*** hinzugefügt, welches dem Instant Trackable zugewiesen wird.

Die ausführliche Dokumentation und die Beispielanwendungen sorgen für ein schnelles Verständnis des SKDs. Eine genaue Anleitung zur Implementierung des MVPs wird von Wikitude unter https://www.wikitude.com/external/doc/documentation/latest/phonegap/instanttracking.html zur Verfügung gestellt.

6.1.3 Praktischer Test

Die Applikation startet auf dem Samsung Galaxy S10 in unter zwei Sekunden. Auch die Erkennung der Umgebung erfolgt bei guter Ausleuchtung innerhalb weniger Sekunden. Die Größe des Modells wird automatisch anhand der Umgebung angepasst. Das Tracking funktioniert sehr gut, sodass auch bei schnellen Bewegungen oder Positionsänderungen die Objekte an der platzierten Stelle bleiben.

6.2 ViroReact

6.2.1 Installation

Bevor die Nutzung von ViroReact möglich ist, muss die Installation der Command Line Interfaces (CLIs) von React Native und ViroReact erfolgen. Anschließend kann mit der ViroReact CLI ein Projekt erstellt, die Beispiel-Applikation heruntergeladen und der Lizenzschlüssel eingetragen werden. Letzteren bekommt man nach der Registrierung auf der Viro Media Webseite zugesendet. ViroReact bietet die Möglichkeit, die erstellte AR-Applikation ohne Installation auf einem Smartphone zu testen. Mit dem ViroReact CLI ist die Erzeugung eines Links zum Einfügen in der Viro Media Testbed Applikation möglich. Über ein Schütteln des Smartphones lässt sich diese nach der Implementierung von Änderungen ganz einfach aktualisieren. Xcode oder Android Studio sind erst bei der Installation auf einem Smartphone nötig.

Insgesamt dauern die Installation der CLIs und das Aufsetzen des ViroReact-Projekts rund 10 Minuten. Weitere 20 Minuten sollten für die Installation von Xcode und Android Studio eingeplant werden.

6.2.2 Implementierung

Auch ViroReact bietet ein Tutorial zur Platzierung von 3D-Objekten an. In das leere ViroReact-Projekt muss anschließend nur noch der Quellcode für die Platzierung von 3D-Objekten eingefügt werden. ViroReact bietet dafür unter https://blog.viromedia.com/how-to-build-an-interactive-ar-app-in-5-mins-w-react-native-viro-ar-e420147e1612 ein Tutorial an.

Eine AR-Szene wird in ViroReact durch die Komponente ***<ViroARScene>*** definiert. Innerhalb dieser Komponente müssen alle für die AR-Anwendung notwendigen Komponenten platziert werden. Im Fall des Tutorials sind dies ***<ViroAmbientLight>***, ***<ViroSpotLight>***, ***<ViroNode>*** und ***<Viro3DObject>***.

Die Komponenten ***<ViroAmbientLight>*** und ***<ViroSpotLight>*** sind für die Beleuchtung des platzierten 3D-Objekts zuständig. ***<ViroNode>*** erkennt Flächen und ermöglicht die Platzierung eines ***<Viro3DObject>***. Nach der Platzierung ermöglicht die Komponente die Bewegung des Objekts auf der Fläche.

6.2.3 Praktischer Test

Auch die Beispiel-Applikation von ViroReact startet auf einem Samsung Galaxy S10 in unter zwei Sekunden. Die Erkennung der Umgebung erfolgte bei guter Ausleuchtung in rund acht Sekunden. Auch hier bleiben die Objekte bei schnellen Bewegungen oder Positionsänderungen an der festgelegten Stelle.

6.3 Vuforia Engine

6.3.1 Installation

Zur Erstellung einer AR-Applikation mit Vuforia ist die Installation des Unity Hubs, Unity, der Android SDK Tools und des Vuforia Plug-Ins nötig. Die Anforderung des kostenlosen Lizenzschlüssels für Unity ist nach der Erstellung eines Unity-Accounts über das Unity Hub möglich. Ein Vorteil ist, dass durch Unity die Installation von Xcode oder Android Studio überflüssig wird. Im Gegensatz zu beiden anderen SDKs ist mit Unity nur ein Programm zur plattformübergreifenden Entwicklung nötig.

Insgesamt dauert die Installation der benötigten Komponenten rund 20 Minuten.

6.3.2 Implementierung

Vuforia bietet über den Unity Asset Store viele Beispiele zum Herunterladen an. Zusätzlich gibt es auf der Webseite von Vuforia viele Tutorials zu den unterschiedlichen Funktionen des Plug-Ins. Die Ground Plane Funktionalität von Vuforia ermöglicht die Platzierung von virtuellen Objekten auf horizontalen Oberflächen. Diese soll auch in der Beispielapplikation mit Unity Verwendung finden. Die Erstellung orientiert sich hierbei an folgender Anleitung https://library.vuforia.com/content/vuforia-library/en/articles/Solution/ground-plane-guide.html.

Nachdem der Vuforia Augmented Reality Support in Unity aktiviert wurde, ist das Hinzufügen von Vuforia-Elementen möglich. Als erster Schritt ist die ***Main Camera*** mit einer ***AR-Camera*** von Vuforia zu ersetzten. Anschließend wird ein ***Ground Plane Stage***-Objekt hinzugefügt. Dieses Objekt verfügt über Bemaßungen. Es soll dem Nutzer zur Abschätzung der Größe des virtuellen Objekts dienen. Nun kann ein beliebiges 3D-Objekt hinzugefügt und auf die passende Größe skaliert werden. Anschließend wird ein ***Plane Finder***-Objekt hinzugefügt. Dessen wichtigste Aufgabe ist das Tracking von horizontalen Flächen und die Positionierung des 3D-Objekts in der realen Umgebung. Außerdem sorgt es dafür, dass bei Berührung des Touchscreens das Objekt an der entsprechenden Stelle platziert wird. Zum Schluss wird das ***Ground Plane Stage***-Objekt an den ***Plane Finder*** angehängt.

6.3.3 Praktischer Test

Die von Unity generierte Applikation startet auf einem Samsung Galaxy S10 in zwei bis drei Sekunden. Die Erkennung der Umgebung erfolgte bei guter Ausleuchtung in unter einer Sekunde. Bei schnellen Bewegungen oder Positionsänderungen mit dem Smartphone verschwinden die virtuellen Objekte teilweise, bis die Position des Geräts wieder bestimmt wurde.

6.4 Fazit

Abschließend soll auf Basis des umgesetzten MVPs eine Bewertung der drei ausgewählten SDKs stattfinden. Die verschiedenen Ansätze, die zur Erstellung einer betriebssystemübergreifenden AR-Applikation zur Verfügung stehen, bieten dabei sowohl für Anfänger als auch für erfahrene Entwickler entsprechende Möglichkeiten.

Anfänger können mit Unity und dem Vuforia Plug-In eine AR-Anwendung erstellen, ohne dafür eine Zeile Code zu schreiben. Das Plug-In stellt dabei umfassende Funktionalitäten für verschiedenste Anwendungsfälle bereit. Durch die ausführlichen Tutorials von Vuforia ist der Nutzer sehr schnell in der Lage, eine funktionierende AR-Applikation zu den genannten Anforderungen zu erstellen. Die App startet schnell und die Erkennung der Umgebung funktioniert gut. Das Tracking ist im Vergleich zu den anderen Ansätzen allerdings weniger zuverlässig.

Die beiden anderen SDKs ermöglichen die Entwicklung in JavaScript. Viro React nutzt dabei das JavaScript Framework React Native, während Wikitude in purem JavaScript entwickelt wird.

Viro React ermöglicht durch vordefinierte Komponenten eine sehr schnelle Erstellung von Applikationen. So kann die AR-Logik mit nur acht Zeilen Code umgesetzt werden. Im Gegensatz zu Wikitude und Vuforia bietet Viro React aber weniger Funktionen an. Für die vorhandenen Funktionen werden dem Entwickler viele Attribute zur Individualisierung der Komponenten zur Verfügung gestellt. Die Erkennung der Umgebung dauert zwar etwas länger, dafür ist das Tracking aber auch bei schnellen Bewegungen stabil.

Die Entwicklung einer AR-Applikation auf Basis von Wikitude ist aufwendiger wie bei den beiden Alternativen. Der große Funktionsumfang, eine schnelle Erkennung der Umgebung und das stabile Tracking sprechen jedoch für die Verwendung. Mit Wikitude sind auch größere Applikationen mit individuellen Anforderungen gut umzusetzen. Sind beim Entwickler Kenntnisse in JavaScript vorhanden, sollte die Einarbeitung in Wikitude und die Erstellung einer AR-Applikation leicht von der Hand gehen.

Alles in allem bietet jede der drei Lösungen eine mehr als ausreichende Funktionalität. Viro React eignet sich dabei eher für kleine Anwendungen, bei denen nur auf die Erkennung der Umgebung und Platzierung von virtuellen Inhalten gesetzt werden soll. Die Nutzung von Vuforia und Wikitude ist mit ihrem breiten Katalog an Funktionen auch bei umfangreicheren Anwendungen möglich. Beide SDKs können dabei individuell an spezielle Anforderungen angepasst werden. Wikitude sticht durch eine schnelle Erkennung der Umgebung und sehr zuverlässiges Tracking hervor. Für die praktische Umsetzung des MVP kommt aus diesen Gründen Wikitude zum Einsatz.

7 Demonstrator

7.1 Lösungsansatz

Wikitude ist aus der Evaluation im letzten Kapitel als die aktuell beste Wahl hervorgegangen.

Um die Erstellung einer plattformübergreifenden Applikation zu ermöglichen, soll die Umsetzung mit Cordova und dem entsprechenden Wikitude-Plug-In stattfinden. Als GUI-Framework kommt React zum Einsatz, da es eine gute Skalierbarkeit und Performance bietet. Außerdem ist es aktuell das meistgenutzte JavaScript Framework und besitzt eine sehr große Community. [17], [18]

Da Wikitude auch ein React Native Modul besitzt und so auf Cordova und React verzichtet werden könnte, scheint diese Auswahl etwas umständlich. [164]

Die Gründe hierfür sind folgende:

Die Bereitstellung des React Native-Moduls erfolgt nicht von Wikitude selbst, sondern von Drittanbietern. Dadurch sind keine regelmäßigen Updates oder Support gewährleistet. Außerdem unterstützt das Modul nur AR-Anwendungen, welche von einer Wikitude Studio URL geladen werden. Wikitude Studio ermöglicht zwar die Erstellung von AR-Applikationen ohne Programmierkenntnisse, unterstützt aktuell aber nur Anwendungen mit Bild- oder Objekttracking. Da in der Zielsetzung die markerlose Platzierung von 3D-Modellen im Raum gefordert ist, scheidet diese Möglichkeit aus. [164]–[166]

Somit fiel die Entscheidung auf die Kombination von React, Cordova und Wikitude.

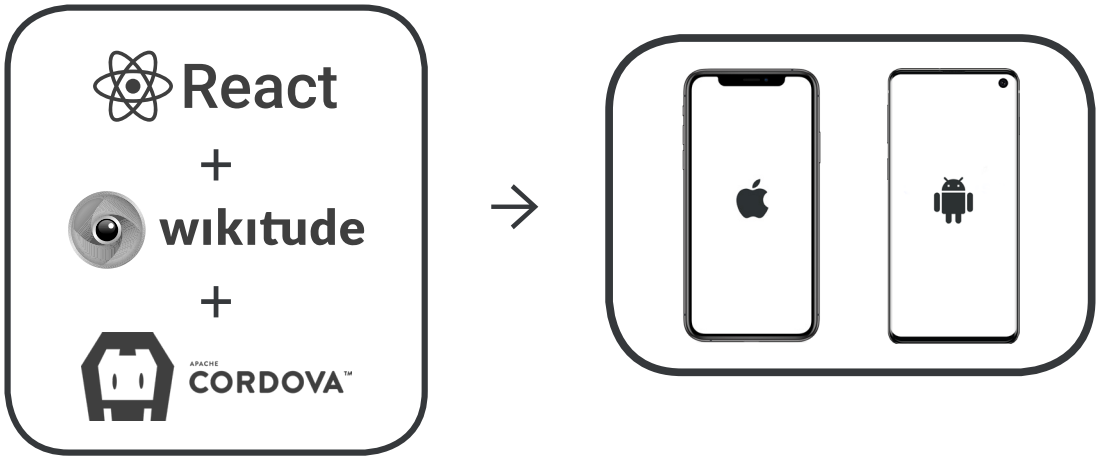

Abb. 22: *Komponenten zur Umsetzung der Applikation*

7.2 Zielsetzung

Die Applikation soll beispielhaft eine von vielen Anwendungsmöglichkeiten im Bereich Augmented Reality zeigen. Das Ziel ist die Erstellung einer Applikation, welche die markerlose Platzierung von 3D-Modellen in der Umgebung ermöglicht. Sie soll weiterhin soll auf iOS und Android lauffähig sein.

Den Usecase liefert die Firma Five-Konzept GmbH & Co. KG, nachfolgend Five-Konzept genannt. Vertriebsmitarbeiter oder auch Fitnessstudio-Besitzer erhalten eine komfortable Möglichkeit zum Einrichten von Five-Geräteparcours in ihren Räumlichkeiten. Sie können verschiedene Anordnungen des Parcours gegenüberstellen und testen, wie viele Geräte in einem Raum Platz finden.

7.3 Konzeption

In diesem Abschnitt folgt die Darlegung aller Schritte, welche vor der Implementierung der Applikation stattgefunden haben. Dazu zählt sowohl der Entwurf des Designs als auch der Funktionalität.

7.3.1 Funktionalität

Voraussetzung für die Anfertigung des Designs ist die Definition des gewünschten Funktionsumfangs. Die Erstellung einer Anforderungsliste aus der Sicht des Nutzers gehört somit zu den ersten Schritten. Da nicht alle Anforderungen gleich wichtig sind, wird zwischen Muss-, Soll- und Wunsch-Anforderungen unterschieden. Die Muss-Anforderungen sind dabei für die Funktion der Applikation unverzichtbar. Soll-Anforderungen sind wichtig, können aber bei zu hohem Aufwand weggelassen werden. Wunsch-Anforderungen werten die Applikation auf, sind jedoch nicht zwingend notwendig. [167]

Muss - Der Nutzer kann...

- ein 3D-Modell an einem gewissen Punkt im Raum platzieren.
- den Startpunkt für das Tracking selbst festlegen.
- die Position und Ausrichtung des 3D-Modells über den Touchscreen verändern.
- die Anwendung zurücksetzen und das Tracking neu starten.

Soll – Der Nutzer kann...

- zwischen verschiedenen 3D-Modellen auswählen.
- mehrere 3D-Modelle platzieren.
- weitere Informationen und Bilder zum platzierten 3D-Modell abrufen.
- die Farben und Materialien auf dem 3D-Modell vor der Platzierung ändern.
- eine Übersicht der platzierten 3D-Modelle aufrufen und diese bei Bedarf löschen.
- einen Screenshot der AR-Ansicht erstellen und auf dem Smartphone speichern.

Wunsch - Der Nutzer kann...

- die Farben und Materialien auch bei bereits platzierten 3D-Modellen ändern.
- mehrere 3D-Modelle gleichzeitig auswählen und diese nacheinander platzieren.

Das Use-Case-Diagramm in Abb. 23 zeigt alle Muss- und Soll-Anforderungen. Abb. 24 stellt den grundlegenden Ablauf bei Benutzung der Applikation aus Nutzersicht in einem Sequenzdiagramm dar.

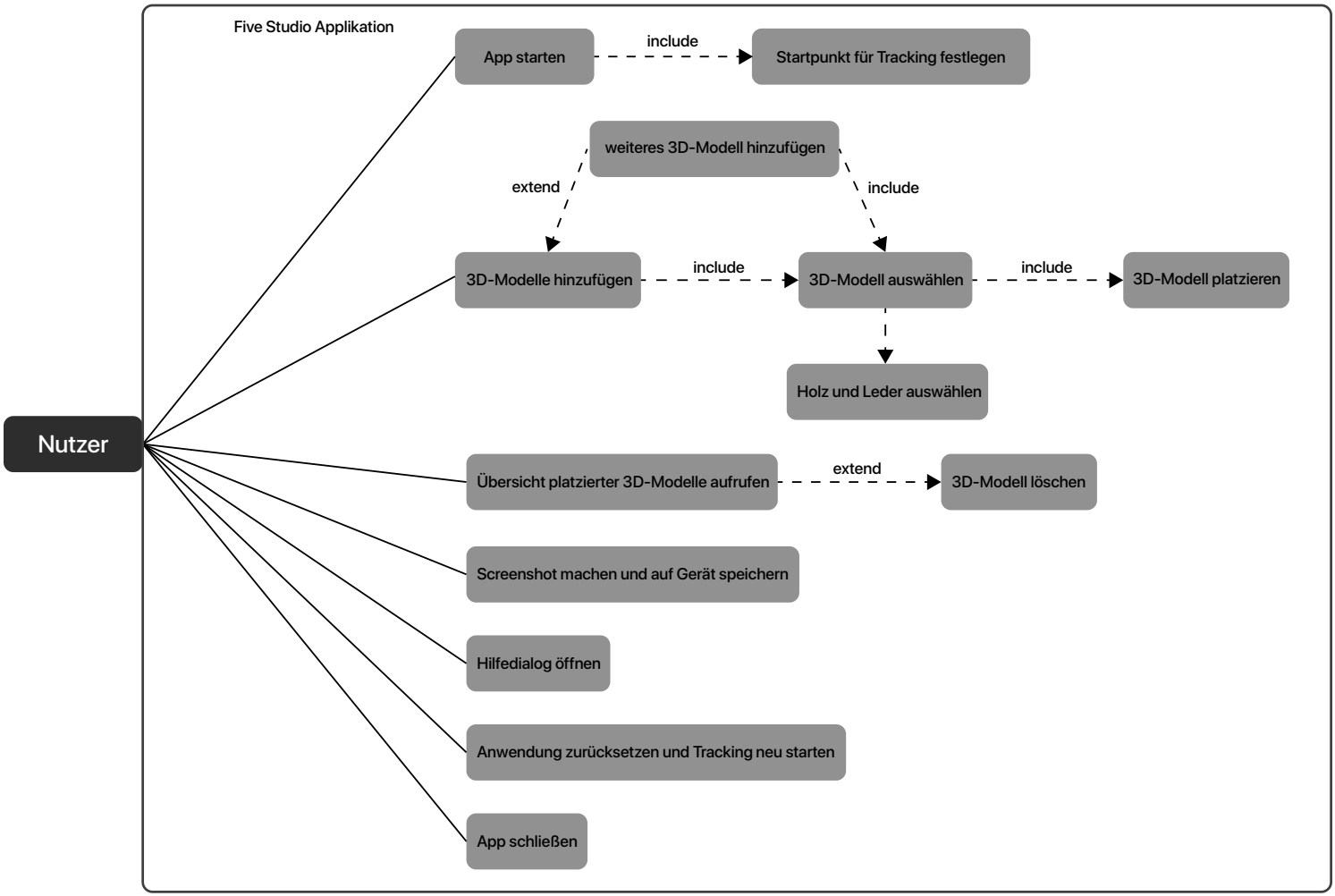

Abb. 23: *Use-Case Diagramm der Funktionen der Beispielapplikation*

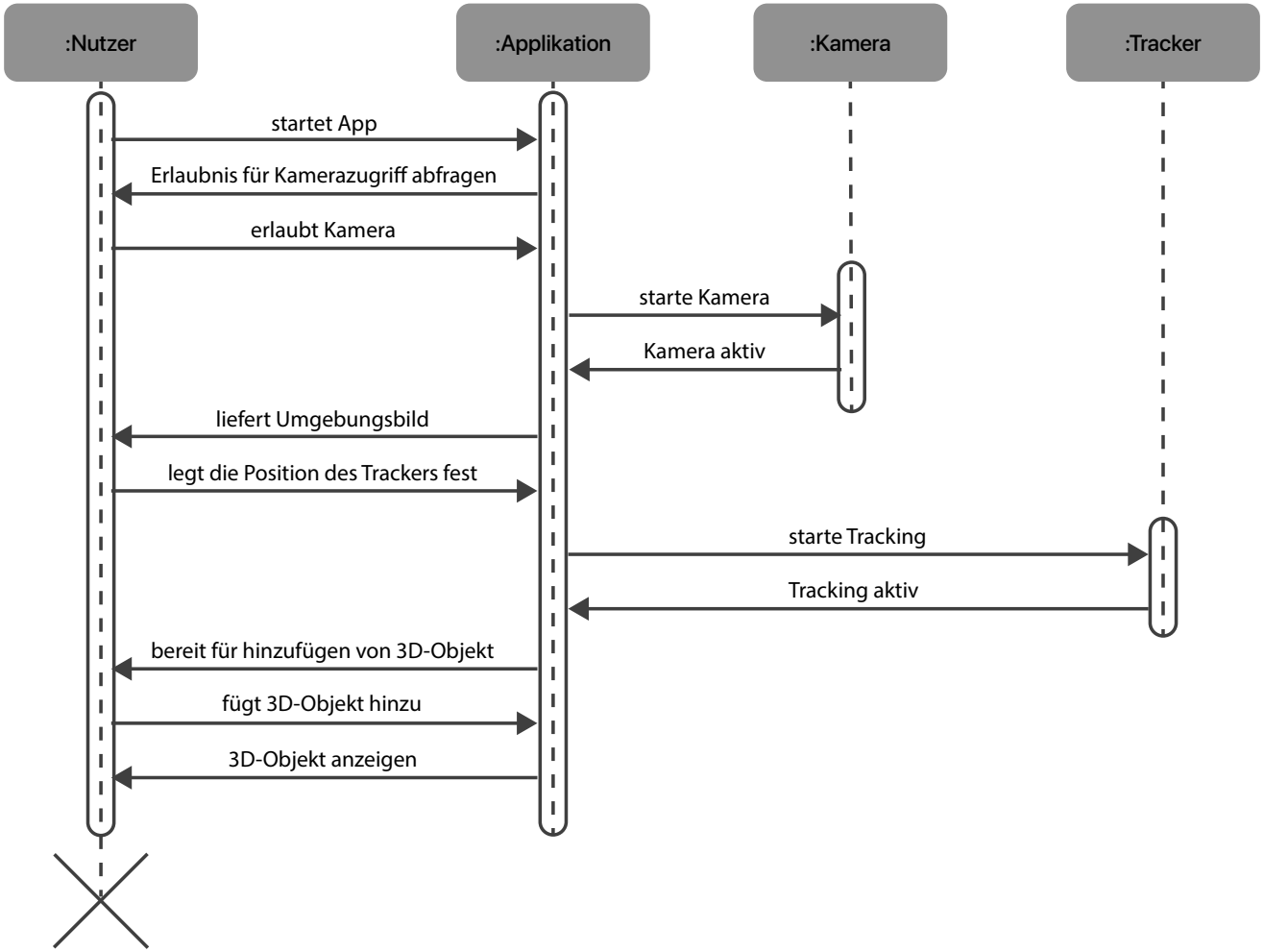

Abb. 24: *Sequenzdiagramm grundlegender Ablauf bei Benutzung der Applikation aus Nutzersicht*

7.3.2 Styleguide

Das Design baut auf den bereits vorhandenen Corporate Design-Vorgaben von Five-Konzept sowie dem Material Design Leitfaden von Google auf. Beim Material Design steht die Funktion der Applikation im Vordergrund. Die Nutzeroberfläche soll so intuitiv und übersichtlich wie möglich gestaltet sein. Durch die Schattierung wird eine Gruppierung der Ebenen erreicht. Abb. 25 belegt, dass sich dadurch die einzelnen Elemente besser voneinander abheben und für eine aufgeräumte Benutzeroberfläche sorgen. [168]

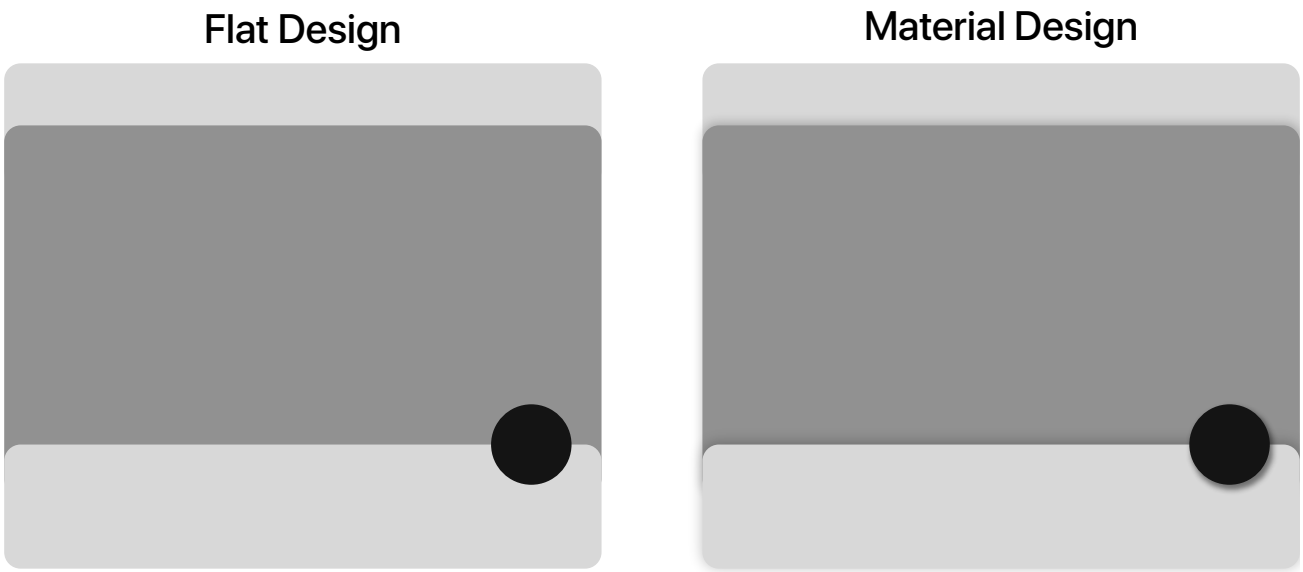

Abb. 25: *Flat Design vs. Material Design*

Grundlayout

Die linke Grafik in Abb. 26 stellt das Grundlayout der App dar. Es besteht aus der Hauptnavigationsleiste am unteren Bildschirmrand, der AR-Ansicht in der Mitte und einer weiteren Navigationsleiste am oberen Bildschirmrand. Die rechte Grafik veranschaulicht, wie sich die Nutzeroberfläche der Applikation verändert, wenn sich das Hauptmenü vergrößert. Die Kameraansicht wird verdeckt und je nach ausgewähltem Menüpunkt mit unterschiedlichem Inhalt und Navigationselementen befüllt.

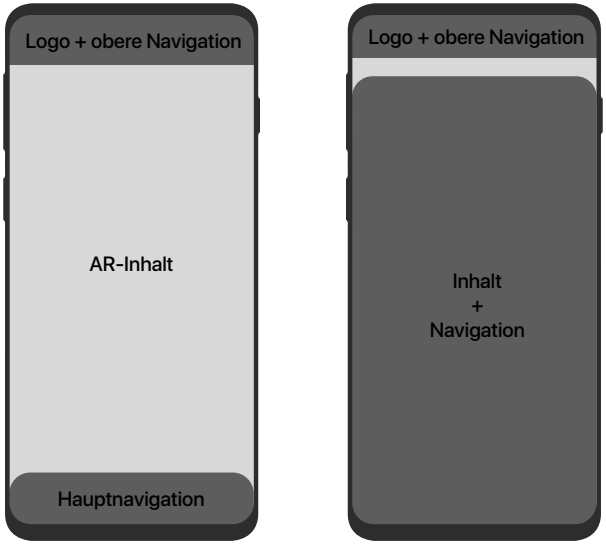

Abb. 26: *Grundlayout der Beispielapplikation*

Abb. 27: *Farben der Beispielapplikation mit RGB-Werten*

Farben

Die Farben der Anwendung wurden mit dem Corporate Design von Five-Konzept abgestimmt. Abb. 27 zeigt die verwendeten Farben mit ihren RGB-Werten. Der Hintergrund des Hauptmenüs ist in Weiß gehalten. Für Elemente wie z. B. Icons werden die Farben Orange und Grau genutzt. Damit der Kontrast zwischen Text und Hintergrund nicht zu extrem ist, sind die Textelemente in der Farbe Grau gestaltet. Die Verwendung der Farbe Schwarz erfolgt gezielt zur Hervorhebung von bestimmten Elementen.

Typographie

Auch die Typographie orientiert sich am Corporate Design von Five-Konzept. Abb. 28 zeigt die verwendete Schriftart „Quicksand". Die abgerundete, serifenlose Schrift wird in verschiedenen Größen und Schriftschnitten verwendet. Bei Überschriften kommt der Schriftstil „Bold" und beim Fließtext der Schriftstil „Regular" zum Einsatz.

Light Regular **Medium** **Bold**

Abb. 28: *Die Schriftart „Quicksand" und deren Schriftschnitte*

Icons

Wie am Anfang des Kapitels beschrieben, orientiert sich die Gestaltung der Applikation am Material Design-Leitfaden. Wo es möglich ist, kamen deshalb vorgefertigte Icons von Google zum Einsatz, siehe Abb. 29. Das Design ist sehr schlicht und drückt die wesentlichen Merkmale einer Aktion aus. Das Set von Google umfasst jedoch nur die gängigsten Aktionen. Zusätzlich benötigte Icons fanden sich auf materialdesignicons.com. Diese Webseite enthält eine breite Auswahl an Icons, welche die Community nach den Material Design Richtlinien erstellt hat. [168], [169]

Abb. 29: *Material Design Icons*

7.3.3 Design der Nutzeroberfläche

Auf Basis des im letzten Abschnitt definierten Styleguides entstand das Design für die Applikation. Im Folgenden soll ein Überblick der verschiedenen Ansichten der Anwendung gegeben werden. Die Beschreibungen beziehen sich jeweils auf den nebenstehenden Designentwurf. Die Möglichkeit zur Materialauswahl ist im Design bereits berücksichtigt, konnte aber im Rahmen der Thesis nicht mehr implementiert werden.

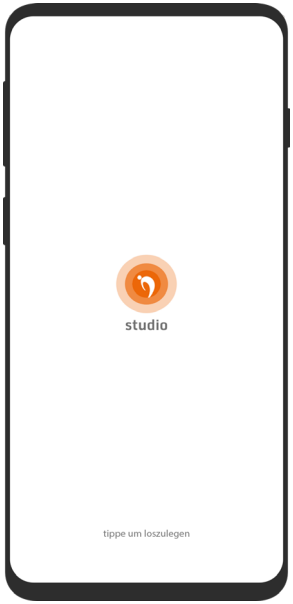

Abb. 30: *Five Studio Startbildschirm*

Startbildschirm

Die Startansicht zeigt das Five-Konzept Logo auf weißem Hintergrund. Der Benutzer muss den Bildschirm antippen, um die Anwendung zu starten.

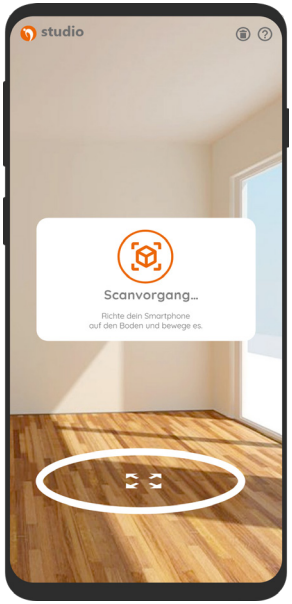

Abb. 31: *Five Studio Scanvorgang läuft*

Scanvorgang

Nach dem Start der Anwendung, beginnt Wikitude mit dem Scannen der Umgebung. Dem Nutzer wird ein Dialog angezeigt, bis der Scanvorgang abgeschlossen ist.

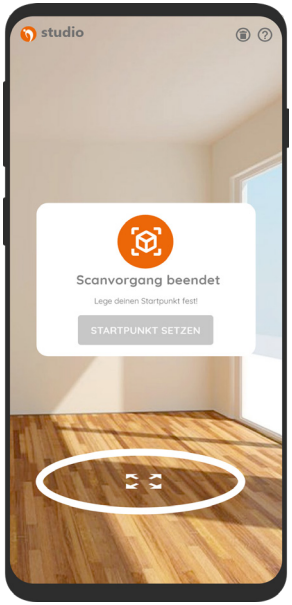

Abb. 32: *Five Studio Scanvorgang beendet*

Scanvorgang

Sobald der Scanvorgang beendet ist, aktualisiert sich der Dialog und ermöglicht es dem Benutzer, den Startpunkt für die Platzierung der Geräte auszuwählen.

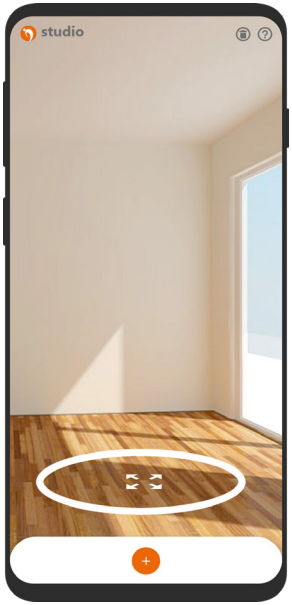

Abb. 33: *Five Studio Hauptmenü*

Hauptmenü

Diese Ansicht sieht der Nutzer, wenn das Tracking erfolgreich gestartet wurde. Für eine angenehme Bedienung ist das Hauptmenü an der unteren Bildschirmkante platziert und kann sich zur Anzeige der Geräteauswahl vergrößern.

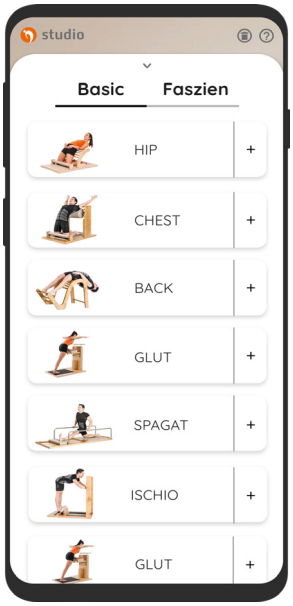

Abb. 34: *Five Studio Geräteauswahl*

Geräteauswahl

Dem Nutzer wird nun eine Übersicht aller Geräte angezeigt. Er kann zwischen den Kategorien Basic und Faszien wählen. Per Klick auf den Plus-Button platziert er das Gerät. Ein Klick auf den Bereich links vom Plus leitet auf eine Detailansicht des entsprechenden Geräts weiter.

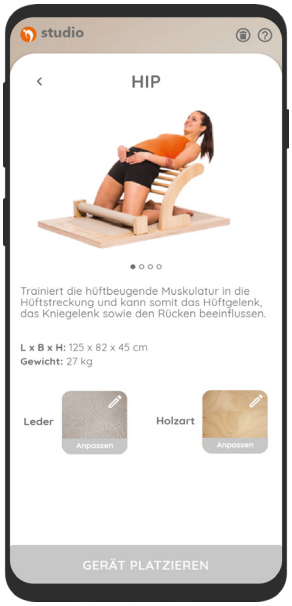

Abb. 35: *Five Studio Gerätedetails*

Gerätedetails

Nach der Auswahl eines Geräts wird auf diese Ansicht weitergeleitet. Mit einer Bildergalerie, Beschreibung, Maßen und Gewichtsangabe kann der Nutzer weitere Infos zum Gerät erfahren.

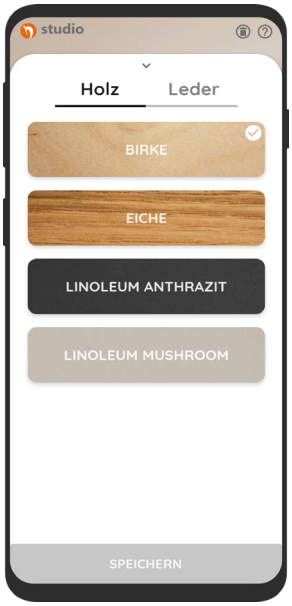

Abb. 36: *Five Studio Holzauswahl*

Holzauswahl

Diese Ansicht würde es dem Nutzer ermöglichen, die Holzart für das zu platzierende Gerät auszuwählen.

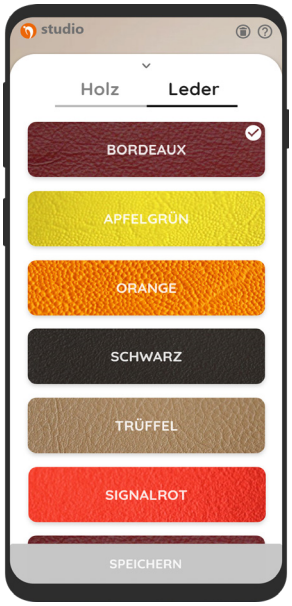

Abb. 37: *Five Studio Farbauswahl*

Farbauswahl

Diese Ansicht würde es dem Nutzer ermöglichen, die Lederfarbe für das zu platzierende Gerät auszuwählen.

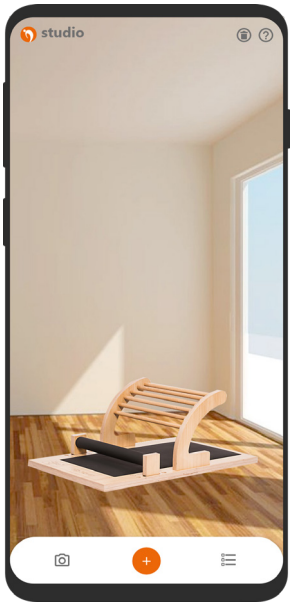

Abb. 38: *Five Studio Platziertes Gerät*

Platziertes Gerät

Diese Ansicht zeigt das platzierte Gerät.

Über das im oberen Menü platzierte Mülltonnen-Icon ist ein Neustart des kompletten Trackings möglich Das Fragezeichen-Icon öffnet den Hilfedialog.

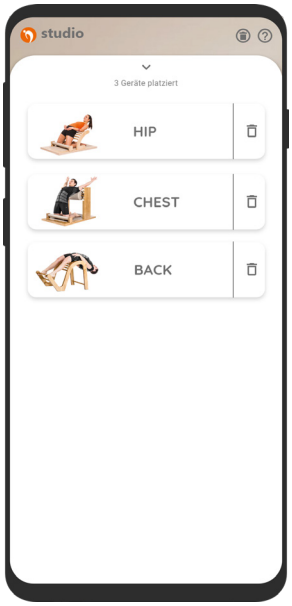

Abb. 39: *Five Studio Platzierte Geräte*

Platzierte Geräte

Diese Ansicht wird über das Listen-Icon im Hauptmenü erreicht. Es zeigt alle bereits platzierten Geräte an. Außerdem kann der Nutzer die einzelnen Geräte löschen.

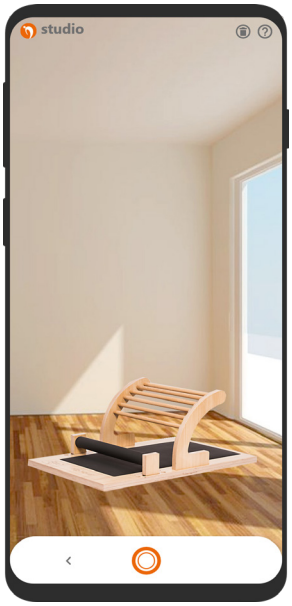

Abb. 40: *Five Studio Screenshot*

Screenshot

Diese Ansicht wird über das Kamera-Icon im Hauptmenü erreicht. Es ermöglicht dem Nutzer einen Screenshot von der aktuellen Ansicht zu machen.

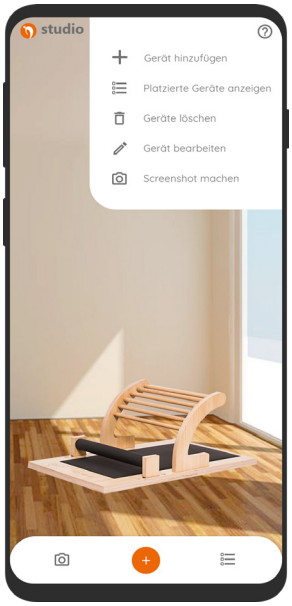

Abb. 41: *Five Studio Hilfedialog*

Hilfedialog

Den Hilfedialog erreicht der Nutzer über das Fragezeichen im oberen Menü am rechten Bildschirmrand. Hier werden alle verwendeten Icons erklärt. Über einen erneuten Klick auf das Fragezeichen-Icon wird der Dialog wieder geschlossen.

7.3.4 Inhalte

Die Grafiken der Applikation wurden freundlicherweise von Five-Konzept zur Verfügung gestellt. Eine Herausforderung stellt die korrekte Vorbereitung der 3D-Modelle dar. Die Anlieferung muss im Dateiformat ***fbx*** erfolgen. Der Wikitude 3D Encoder kann nur dieses Format in die von Wikitude benötigte ***wt3***-Datei konvertieren. Dieses Wikitude-eigene Format optimiert 3D-Modelle für ein schnelles Laden auf dem Smartphone. Um die richtige Darstellung des 3D-Modells und dessen Oberflächen zu gewährleisten, ist die Einhaltung der Vorgaben von Wikitude sehr wichtig. [170], [171]

Diese sind im Detail unter https://www.wikitude.com/external/doc/documentation/latest/phonegap/encoder.html aufgeführt.

7.4 Umsetzung

Zunächst findet die Erläuterung der benötigten Programme und deren Installationsprozesse statt. Danach wird genauer auf die Implementierung eingegangen. Der Fokus liegt dabei auf der Erstellung der Logik und der Nutzeroberfläche. Abschließend folgen Hinweise auf plattformspezifische Eigenheiten bei der Installation der Applikation auf einem Smartphone.

7.4.1 Aufsetzen der Entwicklungsumgebung

Diese Anleitung ist auf den Installationsvorgang unter macOS ausgerichtet. Zur Installation auf Windows gibt es jedoch keine großen Abweichungen. Lediglich einige Befehle in der Kommandozeile können sich geringfügig unterscheiden.

Schritt 1: IDE Jetbrains IntelliJ IDEA installieren

Als Integrated Development Environment (IDE, dt. Integrierte Entwicklungsumgebung) wurde JetBrains IntelliJ IDEA Ultimate genutzt. IntelliJ ist laut dem PYPL Index eine der beliebtesten IDEs. Die eingebaute Unterstützung von z. B. JavaScript, Git, React oder Android ist ein großer Vorteil. Für Studenten bietet Jetbrains unter https://www.jetbrains.com/idea/ einen kostenlosen Download der Entwicklungsumgebung an. Im Anschluss muss nur noch das Installationsprogramm ausgeführt werden. [172], [173]

Schritt 2: Installation von Node.js und npm

Node.js ist eine JavaScript-Laufzeitumgebung. Sie wird zur Erstellung eines Webservers genutzt. Die Installation beinhaltet den Paketmanager npm. Dieser ermöglicht das einfache Hinzufügen von JavaScript-Modulen zum eigenen Projekt. Unter https://www.npmjs.com/ findet man eine Bibliothek mit einer großen Auswahl an Modulen, welche unterschiedlichste Funktionalitäten anbieten. [174], [175]

Um Node.js und npm zu installieren, kann unter https://nodejs.org kostenlos ein Installer heruntergeladen werden. Ob die Installation erfolgreich abgeschlossen ist, lässt sich in der Kommandozeile mit den Befehlen ***node --version*** oder ***npm --version*** testen. Diese Befehle geben bei einer erfolgreichen Installation die aktuelle Node.js- oder npm-Version zurück. [174], [175]

Schritt 3: Installation des Java SDK

Zur Entwicklung von Android-Applikationen muss das Java SDK installiert sein. Ein kostenloser Download steht unter https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html zur Verfügung. Zur Installation muss lediglich den Anweisungen des Installationsprogramms gefolgt werden.

Schritt 4: Installation von Android Studio und des Android SDK

Zur Entwicklung und Installation einer Anwendung sind dank IntelliJ keine weiteren Anwendungen nötigt. Eine Installation von Android Studio ist jedoch trotzdem sinnvoll, da es sich zur Fehlersuche in der Applikation deutlich besser eignet wie IntelliJ. Die Android SDK wird mit Android Studio automatisch installiert, steht jedoch auch einzeln zum Download zur Verfügung. Der Download von Android Studio ist kostenlos unter https://developer.android.com/studio möglich. [176]

Schritt 5: Xcode

Voraussetzung zur Erstellung von Applikationen für iOS ist ein Computer mit macOS und Xcode. Die Entwicklungsumgebung ist kostenlos im App-Store zum Download verfügbar. Neben Xcode werden alle notwendigen Entwicklungswerkzeuge wie das iOS SDK oder die Programmiersprache Swift bei der Installation mitgeliefert. [177]

Schritt 6: Cordova

Cordova ermöglicht es, Applikationen mit einer Code-Basis auf mehreren Endgeräten zu nutzen. Es ist kostenlos über die Kommandozeile mit dem Befehl ***npm install -g cordova*** installierbar. Der Befehl ***cordova –version*** liefert bei erfolgreicher Installation die Versionsnummer von Cordova. [177]

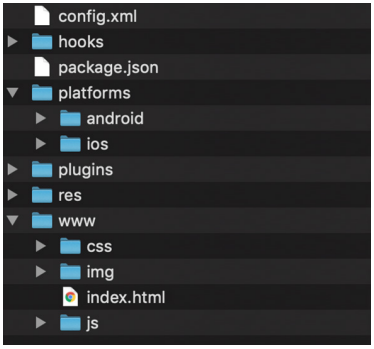

Abb. 42: *leeres Cordova-Projekt*

Die nachfolgenden Schritte können erst ausgeführt werden, nachdem ein Cordova-Projekt erstellt wurde. Mit dem Befehl ***cordova create <Applikationsname>*** lässt sich ein Cordova-Projekt erzeugen. Im Anschluss müssen die gewünschten Ausgabeplattformen im Verzeichnis des Cordova-Projekts mit dem Befehl ***cordova platform add <platform name>*** hinzugefügt werden. Im Fall der Beispielapplikation sind dies iOS und Android. Nach diesem Schritt sollte die Ordnerstruktur wie in Abb. 42 aussehen. [177]

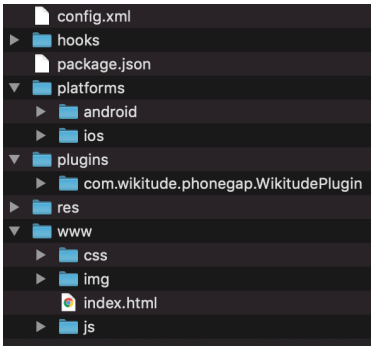

Abb. 43: *Cordova-Projekt mit Wikitude-Plug-In*

Schritt 7: Wikitude

Die Installation des Wikitude-Plug-Ins findet auch im Projektordner von Cordova statt. Es kann mit dem Befehl ***cordova plugin add https://github.com/Wikitude/wikitude-cordova-plugin.git*** hinzugefügt werden. Abb. 43 zeigt die Ordnerstruktur der Cordova-Applikation mit dem Wikitude-Plug-In im Ordner Plug-Ins. [145]

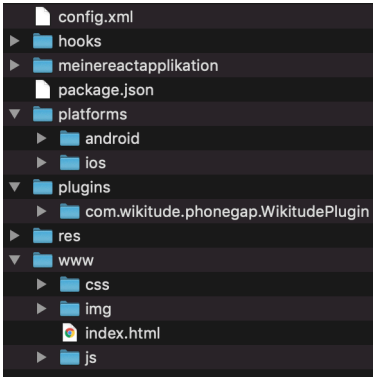

Abb. 44: *Cordova-Projekt mit Wikitude-Plug-In und React-Applikation*

Schritt 8: React

React muss zur Nutzung nicht zwingend installiert sein. Im Projektordner aufgerufen erzeugt der Befehl ***npx create-react-app <meinereactapplikation>*** ein React-Projekt. Der Befehl npx führt npm-Pakete direkt aus, ohne dass sie dafür installiert werden müssen. Abb. 44 veranschaulicht die fertige Ordnerstruktur. [178], [179]

7.4.2 Grundfunktionalität mit Cordova, Wikitude und React

Vor Beginn der eigentlichen Implementierung des Beispielprojekts wird noch etwas genauer auf die Architektur von Cordova eingegangen. Die Verortung von Wikitude und React innerhalb dieser Architektur ist hierbei ein wichtiger Aspekt. Anschließend folgt eine Erläuterung der durchzuführenden Schritte zu einer funktionsfähigen Applikation mit allen genannten Komponenten.

Die Architektur einer Cordova-Applikation wird in Abb. 45 dargestellt. Die Applikation wird innerhalb eines sogenannten Wrappers (oberer Bereich im Schaubild) ausgeführt. Dieser ist auf das Zielbetriebssystem ausgerichtet und kann auf die nativen APIs des Betriebssystems (unterer Bereich) zugreifen. [31], [180]

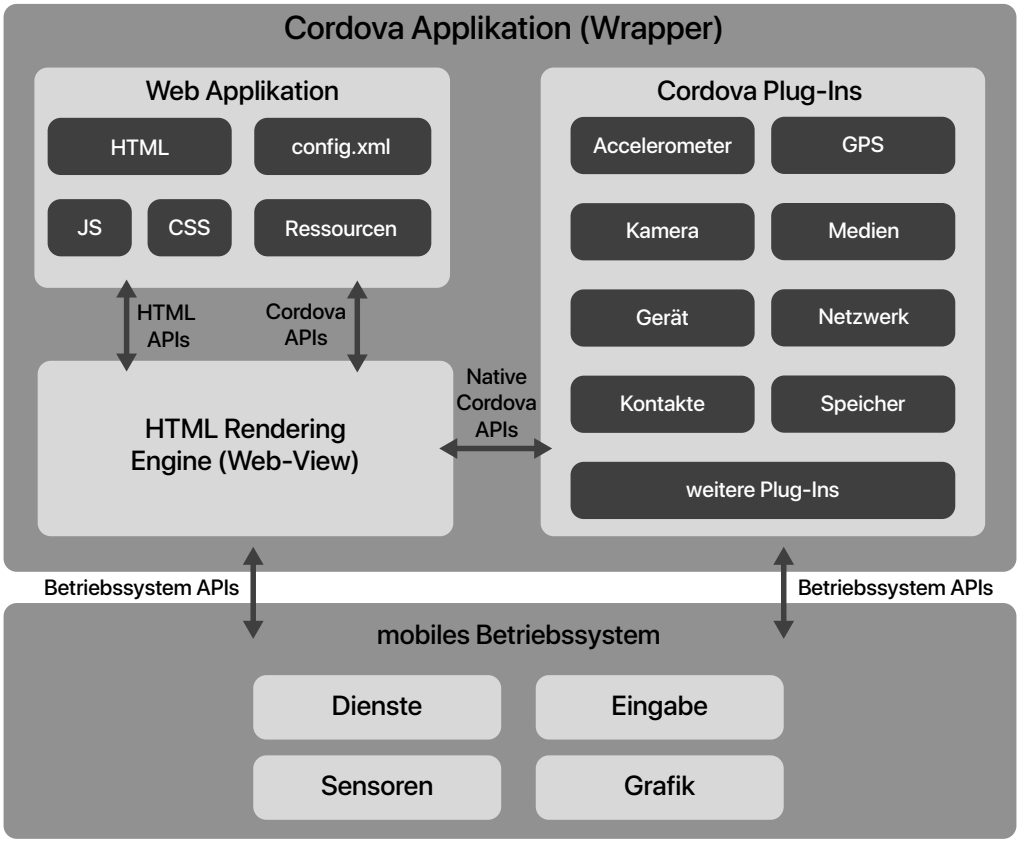

Abb. 45: *Cordova App Architektur*

Innerhalb des Wrappers befinden sich die Web-View, die Web-Applikation und die Cordova Plug-Ins, welche verschiedene Funktionalitäten zur Verfügung stellen. [31], [180]

Die Web-View führt zur Laufzeit die Webapplikation aus und greift auf verschiedene Plug-Ins und die nativen Betriebssystem APIs zu. Die Webapplikation stellt den Quellcode der Anwendung zur Verfügung. Dieser hat wie eine normale Webseite eine Startdatei mit dem Namen ***index.html***, welche auf die weiteren Ressourcen wie CSS, JavaScript und verschiedene Medien verweist. Die Datei ***config.xml*** von Cordova enthält neben wichtigen Informationen über die Applikation auch die Parameter des Wrappers. So bestimmt diese z. B., ob auf Orientierungsänderungen vom Smartphone reagieren soll oder nicht. [31], [180]

Die Plug-Ins bieten Cordova eine Schnittstelle, um mit nativen Komponenten zu kommunizieren. So ist der Zugriff per JavaScript z. B. auf die Kamera oder Sensoren möglich. [31], [180]

React kann man in diesem Schaubild in der Web-Applikation verorten, da es zur Erstellung der Nutzeroberfläche genutzt wird. Wikitude ist im Schaubild bei den Cordova-Plug-Ins untergebracht (siehe Abb. 46). So ist der Zugriff auf die nativen Funktionen des Geräts, wie z. B. die Kamera oder Sensoren, gewährleistet.

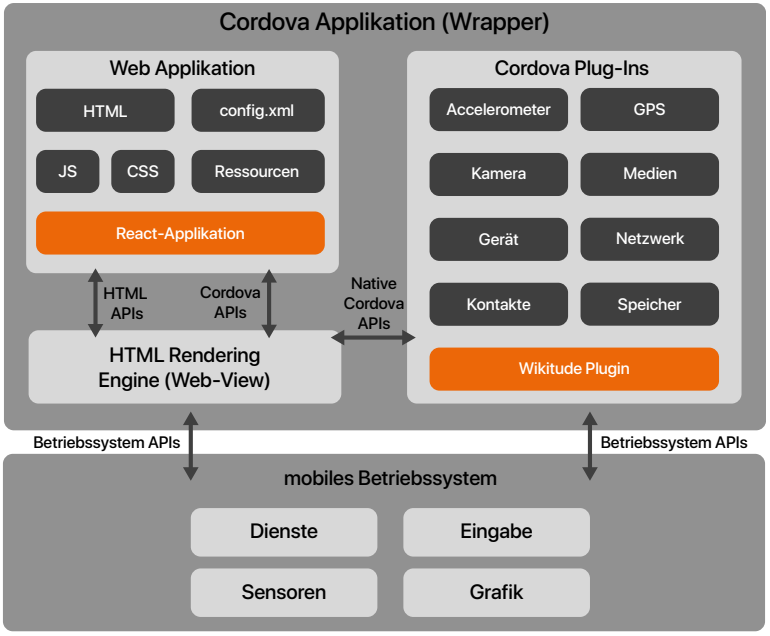

Abb. 46: *Cordova App Architektur mit React-Applikation und Wikitude Plug-In*

Als nächstes folgen die Schritte zu einer funktionsfähigen Applikation. Voraussetzung dafür ist ein vorbereitetes Cordova-Projekt (siehe letzter Abschnitt).

Zunächst wird die React-Applikation mit Cordova verknüpft. Dazu muss die Datei ***index.js*** so gestaltet werden, dass die Applikation erst dann ausgeführt wird, wenn das Gerät bereit ist. Dies lässt sich mit einem Event-Listener auf das ***deviceready***-Event umsetzen.

```
const startApp = () => {
    ReactDOM.render(
        <App />,
        document.getElementById(‚root‘)
    );
}

if(!window.cordova) {
    startApp()
} else {
    document.addEventListener(‚deviceready‘, startApp, false)
}
```

Code 1: *Datei index.js von React*

Bei der Erstellung eines Production-Builds zur Veröffentlichung komprimiert die Frontend-Build-Pipeline von Create React App (CRA, Werkzeug zur schnellen Erstellung einer React-Applikation mit vordefinierten Einstellungen) die Dateien und legt sie im ***build***-Ordner innerhalb der React-Anwendung ab. Um die Ausführung des Builds in der Web-View zu ermöglichen, muss ein Austausch vom aktuellen Inhalt des ***www***-Ordners gegen den Production-Build der React-Applikation stattfinden. Folgende Änderungen in der Datei ***package.json*** der React-Applikation automatisieren diesen Vorgang.

„homepage“: „../www“ sorgt dafür, dass der Production-Build auf den richtigen Speicherort verweist. Der Befehl ***cp -a ./build/. ../www/*** kopiert den Build aus der React-Applikation in den ***www***-Ordner von Cordova. Auf Windows lässt sich dieser Befehl durch ***robocopy .\|\|build ..\|\|www /MIR*** ersetzen.

Nun lässt sich im Ordner der React-Applikation mit ***npm run build*** ein Build ausführen. Das Ergebnis wird in den ***www***-Ordner des Cordova-Projekts kopiert.

VORHER

```
{
  „name“: „meinereactapplikation“,
  „version“: „0.1.0“,
  „private“: true,

  „dependencies“: {…},

  „scripts“: {
    „start“: „react-scripts start“,
    „build“: „react-scripts build“,
    „test“: „react-scripts test
             --env=jsdom“,
    „eject“: „react-scripts eject“
    „deploy“: „npm run build&&gh-pages
              -d build“
  },

  „eslintConfig“: {…},
  „browserslist“: {…},
}
```

NACHHER

```
{
  „name“: „meinereactapplikation“ ,
  „version“: „0.1.0“,
  „private“: true,

  „homepage“: „../www“,

  „dependencies“: {…},

  „scripts“: {
    „start“: „react-scripts start“,
    „build“: „react-scripts build &&
             cp -a ./build/. ../www/“,
    „test“: „react-scripts test
            --env=jsdom“,
    „eject“: „react-scripts eject“,
    „deploy“: „npm run build&&gh-pages
              -d build“
  },

  „eslintConfig“: {…},
  „browserslist“: {…},
}
```

Code 2: *Änderungen in der package.json-Datei von React*

Im nächsten Schritt soll die AR-Funktionalität mit Wikitude hinzugefügt werden. In der React-Applikation gibt es einen Ordner für statische (***public***) und einen Ordner für dynamische Dateien (***src***). Die JavaScript-Dateien von Wikitude liegen deshalb im ***public***-Ordner und die React-Komponenten im ***src***-Ordner.

Bei CRA sind die Einstellungen für Webpack, einem JavaScript-Modul-Packer, schon vorkonfiguriert. Das ermöglicht eine schnelle Erstellung einer React-Applikation mit fertiger Projektstruktur. Allerdings sind die vorgegebenen Einstellungen sehr restriktiv. So ist z. B. der Aufruf aus dem ***src***-Order von Dateien im ***public***-Ordner nicht möglich. Dies führt dazu, dass die Skripte für die Nutzeroberfläche und AR-Logik nicht miteinander kommunizieren können. [179], [181]

Der einfachste Weg zur Anpassung der Webpack-Konfiguration ist ein ***Eject*** der React-Applikation. Dieser irreversible Prozess sorgt dafür, dass die Bearbeitung aller Konfigurationsdateien von Webpack möglich ist und versteckte, vorinstallierte Plug-Ins sichtbar werden. Der Entwickler kann so alle Einstellungen selbst anpassen. Da jedoch viele Vorteile verloren gehen, wurde in dieser Thesis das Plugin React-App-Rewired verwendet. Es ermöglicht das Überschreiben der Konfiguration von Webpack, ohne einen ***Eject*** der React-Applikation durchzuführen.

Die Installation des Plug-Ins erfolgt über die Kommandozeile mit dem Befehl ***npm install react-app-rewired***. Die Konfiguration hierzu wird im Root-Verzeichnis in der Datei ***config-overrides.js*** der React-Applikation angelegt.

```
const ModuleScopePlugin = require(‚react-dev-utils/ModuleScopePlugin‘);

module.exports = function override(config, env) {
    config.resolve.plugins = config.resolve.plugins.filter(plugin => !(plugin
    instanceof ModuleScopePlugin));
    return config;
};
```

Code 3: *Datei: config-overrides.js für das Plugin React-App-Rewired*

Das ModuleScope-Plug-In sorgt dafür, dass nur Dateien aus dem ***src***-Ordner in React aufgerufen werden können. Die dritte Zeile des abgebildeten Codes entfernt dieses Plug-In. So ist der Zugriff aus dem ***src***-Ordner auch auf außerhalb liegende Dateien möglich.

Abschließend müssen noch folgende Anpassungen in der Datei ***package.json*** der React-Applikation vorgenommen werden. Durch diese Änderungen findet der Aufruf der Skripte des React-App-Rewired-Plug-Ins anstatt den Skripten von CRA statt. Dies sorgt dafür, dass die vorgenommenen Änderungen auch im Production-Build der Applikation enthalten sind.

VORHER

```
{
  „name“: „meinereactapplikation“ ,
  „version“: „0.1.0“,
  „private“: true,

  „homepage“: „../www“,

  „dependencies“: {…},

  „scripts“: {
    „start“: „react-scripts start“,
    „build“: „react-scripts build &&
             cp -a ./build/. ../www/“,
    „test“: „react-scripts test
            --env=jsdom“,
    „eject“: „react-scripts eject“,
    „deploy“: „npm run build&&gh-pages
              -d build“
  },

  „eslintConfig“: {…},
  „browserslist“: {…},
}
```

NACHHER

```
{
  „name“: „meinereactapplikation“ ,
  „version“: „0.1.0“,
  „private“: true,

  „homepage“: „../www“,

  „dependencies“: {…},

  „scripts“: {
    „start“: „react-app-rewired start“,
    „build“: „react-app-rewired build &&
             cp -a ./build/. ../www/“,
    „test“: „react-app-rewired test
            --env=jsdom“,
    „eject“: „react-scripts eject“,
    „deploy“: „npm run build&&gh-pages
              -d build“
  },

  „eslintConfig“: {…},
  „browserslist“: {…},
}
```

Code 4: *Änderungen in der package.json-Datei von React mit dem Plug-In React-App-Rewired*

Nach dieser Änderung können die Skripte für die Nutzeroberfläche und die Skripte, welche die AR-Logik steuern, miteinander kommunizieren.

Die Konfiguration von Create-React-App sorgt außerdem dafür, dass das Laden der React-Anwendung nur in der Datei ***index.html*** funktioniert. Da die Anwendung in diesem Fall in die Datei ***ar.html*** eingefügt werden sollte, sind auch hier Änderungen nötig.

Wikitude leitet von der Startseite (Datei ***index.html***) auf die Seite mit der AR-Ansicht (Datei ***ar.html***) weiter. Da die React-Anwendung durch die Konfiguration von Webpack nur in der Datei index.html gerendert werden kann, bietet es sich an, die Inhalte der Dateien ***index.html*** und ***ar.html*** zu vertauschen. Im Anschluss folgt die Anpassung der Datei ***config.xml*** um die Startseite von Cordova abzuändern.

Dafür muss die Anpassung folgender Zeile in der Datei ***config.xml*** stattfinden.

```
VORHER

<content src="index.html" />
```

```
NACHHER

<content src="ar.html" />
```

Code 5: *Änderungen in der Datei config.xml von Cordova*

Mit dieser Änderung wird statt der Datei ***index.html*** die Datei ***ar.html*** als Startseite der App geladen.

Da die AR-Logik in seltenen Fällen jedoch auch Daten an die React-Applikation weiterleiten muss, reichen die vorgenommenen Einstellungen noch nicht. Da Wikitude eine eigene Web-View zur Darstellung des Kamerabilds und der virtuellen Inhalte nutzt, lässt sich die Kommunikation mit der WebView von Cordova über ***postMessage*** realisieren (siehe Abb. 47).

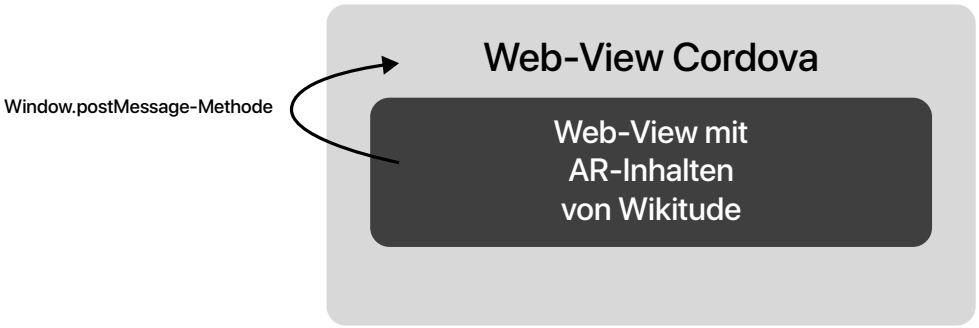

Abb. 47: *postMessage-Methode zur Kommunikation zwischen Cordova und Wikitude*

In nachfolgend gezeigtem Beispiel wird in React eine bestimmte Aktion ausgeführt, sobald die Web-View von Cordova eine Nachricht über die ***postMessage***-Methode empfängt.

```
REACT KOMPONENTE

bindEvents() {
  window.top.addEventListener(„message“,
  this.onMessageReceived, false);
}

onMessageReceived = (event) => {…}
```

```
WIKITUDE

window.top.postMessage(„message“, „*“);
```

Code 6: *postMessage-Kommunikation von Wikitude zu React*

Wenn die genannten Einstellungen erfolgreich vorgenommen sind, kann die Kombination von React, Wikitude und Cordova ohne Probleme zur Entwicklung einer AR-Applikation verwendet werden.

7.4.3 Implementierung der AR-Logik

In diesem Abschnitt soll etwas genauer auf die Implementierung der AR-Logik eingegangen werden. Der vollständige Quellcode findet sich im Anhang wieder. Der Quellcode basiert auf einer Beispielanwendung von Wikitude und wurde im Laufe der Implementierung stetig erweitert und angepasst.

Die AR-Logik benötigt zwei Dateien: Eine initialisiert das Wikitude-Plug-In (***index.js***) und leitet auf die AR-Ansicht weiter, während die andere (***3dmodelsandgesturesinstanttracking.js***) die Logik für die AR-Inhalte bereitstellt.

Die Datei ***index.js*** ist auf der Startseite ***ar.html*** eingebunden. Sobald das Smartphone bereit ist, sorgt ein Event-Listener für den Import des Wikitude-Plug-Ins von Cordova.

```
    initialize: function() { this.bindEvents(); },
    bindEvents: function() {
        document.addEventListener(‚deviceready‘, this.onDeviceReady, false);
    },
    onDeviceReady: function() {
        app.wikitudePlugin = cordova.require(„com.wikitude.phonegap.WikitudePlugin.
        WikitudePlugin“);
    },
```

Code 7: *Datei index.js, Funktionen initialize(), bindEvents() und onDeviceReady()*

Auf welcher Seite mit welchen Spezifikationen die AR-View geladen werden soll, definiert das Objekt ***arexperience*** in der ***index.js***. Im Fall der Beispielapplikation soll die AR-View in der Seite ***index.html*** mit den Funktionen für Instant Tracking geladen werden. Dabei kommt die Rückkamera des Smartphones in Full-HD-Auflösung und mit Autofokus zum Einsatz.

```
var arexperience = {
    „path“: „www/index.html“,
    „requiredFeatures“: [„instant_tracking“],
    „startupConfiguration“: {
        „camera2_enabled“: true,
        „camera_position“: „back“,
        „camera_resolution“: ‚full_hd‘,
        „camera_focus_mode“: ‚continuous‘
    }
};
```

Code 8: *Datei index.js, Objekt arexperience*

Sobald der Nutzer den Startbildschirm berührt, folgt der Aufruf der Funktion ***startAR()***. Die Funktion ***prepareArchitectWorld()*** kontrolliert, ob das Gerät Unterstützung für Wikitude bietet. Bei Bedarf wird die Berechtigung zur Nutzung der Kamera abgefragt. Die Funktion ***loadArchitectWorld()*** lädt das Wikitude-Plug-In mit den im letzten Abschnitt beschriebenen Spezifikationen.

```
startAR: function() {
        app.prepareArchitectWorld(arexperience, function() {
            app.loadARchitectWorld(arexperience);
        });
},
```

Code 9: *Datei index.js, Funktion startAR()*

Außerdem findet sich in der ***index.js*** auch die Funktion ***onJSONObjectReceived()*** wieder. Diese kann ein JSON-Objekt von der AR-View empfangen und an das Betriebssystem weitergeben. Im Fall der Beispielapplikation wird so das Speichern eines Screenshots der AR-View auf dem Smartphone ermöglicht.

```
onJSONObjectReceived: function (jsonObject) {
     if (typeof jsonObject.action !== ‚undefined‘) {
          if ( jsonObject.action === „capture_screen“ ) {
              app.wikitudePlugin.captureScreen(
                  function(absoluteFilePath) {
                    alert(„Der Screenshot wurde in deiner Galerie gespeichert!“);
                  },
                  function (errorMessage) {
                    alert(errorMessage);
                  },
                  true, null
              );
          }
     }
},
```

Code 10: *Datei index.js, Funktion onJSONObjectReceieved()*

Wie im Objekt ***arexperience*** in Codebeispiel 8 festgelegt, erfolgt anschließend eine Weiterleitung auf die Seite ***index.html***. In dieser wird die Datei ***3dmodelsandgesturesinstanttracking.js*** eingebunden und die AR-View vom Wikitude-Plug-In erzeugt.

Der erste Schritt in der Datei ***3dmodelsandgestureinstantracking.js*** ist die Prüfung, ob das verwendete Gerät ARCore oder ARKit unterstützt.

Die Klasse ***hardware*** enthält Funktionen, welche die Interaktion mit dem verwendeten Gerät ermöglichen. So kann unter anderem herausgefunden werden, ob Unterstützung für ARCore oder ARKit verfügbar ist.

Codebeispiel 11 zeigt eine Switch-Anweisung für die unterschiedlichen Fälle. In Fall 1 ermöglicht das verwendete Gerät kein plattformunterstütztes Tracking. Die Ausführung des Trackings erfolgt deshalb mit dem SLAM-Algorithmus von Wikitude. Fall 2 tritt ein, falls vom Gerät noch keine Rückmeldung über die Unterstützung vorliegt. In diesem Fall findet ein erneuter Aufruf der Funktion statt. Wenn Fall 3 zutrifft, unterstützt das verwendete Gerät kein ARCore oder ARKit. Für das Tracking nutzt Wikitude dann den eigenen SLAM-Algorithmus. Bei Fall 4 muss die ARCore-Applikation heruntergeladen oder aktualisiert werden. Die Funktion wird nach erfolgreicher Installation oder dem Update erneut aufgerufen. Der letzte Fall tritt ein, wenn das Gerät plattformunterstütztes Tracking ermöglicht.

```
AR.hardware.smart.onPlatformAssistedTrackingAvailabilityChanged = function(availabi-
lity) {
    switch (availability) {
        case AR.hardware.smart.SmartAvailability.INDETERMINATE_QUERY_FAILED:
            World.createOverlays();
            break;
        case AR.hardware.smart.SmartAvailability.CHECKING_QUERY_ONGOING:
            break;
        case AR.hardware.smart.SmartAvailability.UNSUPPORTED:
            World.createOverlays();
            break;
        case AR.hardware.smart.SmartAvailability.SUPPORTED_UPDATE_REQUIRED:
            break;
        case AR.hardware.smart.SmartAvailability.SUPPORTED:
            World.platformAssisstedTrackingSupported = true;
            World.createOverlays();
            break;
    }
};
```

Code 11: *Datei 3dmodelsandgesturesinstantracking.js, Verfügbarkeitsprüfung von ARKit oder ARCore*

Der Start des Trackings erfolgt über den Aufruf der Funktion ***createOverlays()***. Innerhalb dieser Funktion wird eine neue Instanz des Tracker-Objekts angelegt. Diese kann mit verschiedenen Events auf eine Zustandsänderung oder Fehlermeldungen reagieren. ARKit und ARCore ermöglichen durch das Scannen der Umgebung die Größe des 3D-Modells realitätsgetreu wiederzugeben. Die Eigenschaft ***deviceHeight*** wird benötigt, wenn kein plattformunterstütztes Tracking stattfindet. Da der SLAM-Algorithmus die automatische Berechnung der Größe des 3D-Modells nicht unterstützt, nutzt Wikitude den Abstand vom Smartphone zum Boden, um diese zu berechnen. Damit der Nutzer diesen Abstand nicht jedes Mal neu einstellen muss, wird hier ein Durchschnittswert von 1.3 Metern verwendet.

```
this.tracker = new AR.InstantTracker({
            onChangedState: function onChangedStateFn(state) {},
            deviceHeight: 1.3,
            onError: World.onError,
            onChangeStateError: World.onError,
});
```

Code 12: *Datei 3dmodelsandgesturesinstantracking.js, Funktion createOverlays(), Erstellung eines neuen Trackers*

In Codebeispiel 13 wird dem Tracker eine Instanz des Instant Trackable-Objekts zugewiesen. Diesem können verschiedene virtuelle Objekte angehängt werden. Die Eigenschaft ***drawables*** ermöglicht es, die virtuellen Objekte an das Instant Trackable zu binden. Weiterhin gibt es eine große Auswahl an Events, auf die reagiert werden kann. Wenn beispielsweise das Event ***onTrackingPlaneDrag*** ausgelöst wird, ist die Veränderung der Position des virtuellen Objekts über den Touchscreen möglich.

```
this.instantTrackable = new AR.InstantTrackable(this.tracker, {
    drawables: {…},
    onTrackingPlaneDragBegan: function onTrackingPlaneDragBeganFn(xPos, yPos) {
                oneFingerGestureAllowed = true;
                World.updatePlaneDrag(xPos, yPos);
    },
    onError: World.onError
});
```

Code 13: *Datei 3dmodelsandgesturesinstantracking.js, Funktion createOverlays(), Erstellung eines neuen Instant Trackables*

In Codebeispiel 14 wird im Intervall von einer Sekunde abgefragt, ob die Initialisierung des Trackers abgeschlossen ist oder nicht. Sobald der Start des Trackings möglich ist, wird per ***postMessage*** eine Nachricht an die Web-View der Nutzeroberfläche gesendet. Diese Nachricht bewirkt eine Änderung im Dialog, sodass der Nutzer das Tracking starten kann.

```
setInterval(
    function () {
         if (World.tracker.canStartTracking) {
             window.top.postMessage(„tracking“, „*“);
         } else {
             window.top.postMessage(„nottracking“, „*“);
         }
    }, 1000
);
```

Code 14: *Datei 3dmodelsandgesturesinstantracking.js, Funktion createOverlays(), Interval zur Überprüfung ob Trackingstart möglich ist*

Beim Start des Trackings erfolgt der Aufruf der Funktion ***changeTrackerState()***. Diese sorgt für eine Statusänderung des Trackers. Der wiederholte Aufruf dieser Funktion bricht das Tracking ab. Der Nutzer hat jedoch immer die Möglichkeit eines Neustarts.

```
changeTrackerState: function changeTrackerStateFn() {
     if (this.tracker.state === AR.InstantTrackerState.INITIALIZING) {
         this.tracker.state = AR.InstantTrackerState.TRACKING;
     } else {
         this.tracker.state = AR.InstantTrackerState.INITIALIZING;
         window.top.postMessage(„changeTrackerstate“, „*“);
     }
},
```

Code 15: *Datei 3dmodelsandgesturesinstantracking.js, Funktion changeTrackerState()*

Sobald der Nutzer ein 3D-Modell auswählt, wird der Funktion ***updateRequestedModel()*** dessen Kategorie und Nummer übergeben. Anschließend findet die Eintragung der Daten in den entsprechenden Variablen statt. Sobald der Nutzer das gewählte 3D-Modell per Touch auf den Bildschirm an einer bestimmten Position platziert, erfolgt aus dem Instant Trackable der Aufruf der Funktion ***updatePlaneDrag()***. Diese ist für das Verschieben und Platzieren von 3D-Modellen zuständig. Eine if-Abfrage erkennt, ob die Platzierung eines neuen 3D-Modells oder die Verschiebung eines bestehenden Modells stattfinden soll. Bei der Platzierung eines neuen Modells, wird die Funktion ***addModel()*** aufgerufen.

```
updateRequestedModel: function updateRequestedModelFn(categorie, number) {
        World.requestedModel = categorie[number].number;
        World.categorie = categorie;
    },
updatePlaneDrag: function updatePlaneDragFn(xPos, yPos) {
        if (World.requestedModel >= 0) {
            World.addModel(World.requestedModel, xPos, yPos);
            World.requestedModel = -1;
            World.initialDrag = true;
        }
        if (World.initialDrag && oneFingerGestureAllowed) {
            World.lastAddedModel.translate = {
                x: xPos,
                y: yPos
            };
        }
    },
```

Code 16: *Datei 3dmodelsandgesturesinstantracking.js, Funktionen updateResquestedModel() und updatePlaneDrag()*

Die Funktion ***addModel()*** erhält die Koordinaten zur Platzierung und den Pfad zum ausgewählten 3D-Modell. Im ersten Schritt kontrolliert die Funktion ***isTracking()***, ob der Tracker aktiv ist. Ist dies der Fall, erfolgt die Erstellung einer neuen Instanz des Modell-Objekts mit festgelegter Größe, Richtung und Position. Diese kann mit verschiedenen Events z. B. auf Nutzereingaben auf dem Touchscreen reagieren. Im Fall der Beispielapplikation kann der Nutzer das 3D-Modell beliebig drehen oder verschieben. Die erstellte Instanz wird anschließend zum Array mit allen platzierten 3D-Modellen hinzugefügt und an das Instant Trackable angehängt.

```
addModel: function addModelFn(pathIndex ,xpos, ypos) {
      if (World.isTracking()) {
            var modelIndex = rotationValues.length;
            World.addModelValues();
            var model = new AR.Model(World.categorie[pathIndex].path, {
                scale: { x: defaultValue, y: defaultValue, z: defaultValue },
                translate: { ,x: xpos, y: ypos },
                onDragBegan: function() { oneFingerGestureAllowed = true; },
                onDragChanged: function (intersectionX, intersectionY) {
                    if (oneFingerGestureAllowed) {
                        this.translate = { x: intersectionX, y: intersectionY };
                    }
                },
                onRotationChanged: function (angleInDegrees) {
                    this.rotate.z = rotationValues[modelIndex] - angleInDegrees;
                },
                onRotationEnded: function (){
                    rotationValues[modelIndex] = this.rotate.z
                },
                onError: World.onError
            });
            allCurrentModels.push(model);
            allCurrentModelCategories.push(World.categorie[pathIndex]);
            World.lastAddedModel = model;
            this.instantTrackable.drawables.addCamDrawable(model);
     }
},

isTracking: function isTrackingFn() {
    return (this.tracker.state === AR.InstantTrackerState.TRACKING);
},
```

```
addModelValues: function addModelValuesFn() {
    rotationValues.push(defaultRotationValue);
    scaleValues.push(defaultScaleValue);
},
```

Code 17: *Datei 3dmodelsandgesturesinstantracking.js, Funktionen addModel(), isTracking() und addModelValues()*

Die Funktion ***removeModel()*** ermöglicht das Löschen von einzelnen 3D-Modellen. Mit der Funktion ***resetModels()*** werden alle 3D-Modelle aus den Arrays und vom Instant Trackable entfernt. Der Aufruf der Funktion ***resetAllModelValues()*** sorgt anschließend für die Leerung der Arrays mit Größen- und Rotationsangaben zu den 3D-Modellen.

```
removeModel: function resetModelsFn(i) {
    this.instantTrackable.drawables.removeCamDrawable(i+1);
    allCurrentModels.splice(i, 1);
    allCurrentModelCategories.splice(i, 1);
},

resetModels: function resetModelsFn() {
    this.instantTrackable.drawables.removeCamDrawable(allCurrentModels);
    allCurrentModels = [];
    allCurrentModelCategories = [];
    World.resetAllModelValues();
},

resetAllModelValues: function resetAllModelValuesFn() {
    rotationValues = [];
    scaleValues = [];
},
```

Code 18: *Datei 3dmodelsandgesturesinstantracking.js, Funktionen removeModel(), resetModels() und resetAllModelValues()*

Der Aufruf der Funktion ***onError()*** geschieht immer dann, wenn ein Fehler in einer beliebigen Funktion im Skript passiert. Die Anzeige der Fehlerbeschreibung findet in einem Pop-Up-Fenster statt. Außerdem wird per ***postMessage*** eine Nachricht an die Nutzeroberfläche gesendet, sodass der Nutzer das Tracking bei Bedarf neu starten kann.

```
onError: function onErrorFn(error) {
   alert(error);
   window.top.postMessage(„changeTrackerstate“, „*“);
}
```

Code 19: *Datei 3dmodelsandgesturesinstantracking.js, Funktion onError()*

Codebeispiel 20 ist das Gegenstück zu Codebeispiel 10. Die Funktion ***captureScreen()*** wird aufgerufen, wenn der Nutzer den Auslöser für einen Screenshot in der Applikation drückt. Sie sendet ein JSON-Objekt an die Funktion in Codebeispiel 10, welche einen Screenshot erstellt und auf dem Gerät abspeichert.

```
captureScreen: function captureScreenFn() {
    AR.platform.sendJSONObject({
        action: „capture_screen“
    });
},
```

Code 20: *Datei 3dmodelsandgesturesinstantracking.js, Funktion captureScreen()*

7.4.4 Implementierung der Nutzeroberfläche

Das folgende Kapitel soll ein Überblick über die Struktur der Applikation, den Aufbau der React-Komponenten und die verwendeten Plug-Ins geben.

```
/***************Importe**************/
import React from ‚react‘;
import { Component } from ‚react‘;
import {World} from „../../public/js/3dmodelsandgesturesinstanttracking.js“;

/***************Styles**************/
const styles= { test: { color:‘white‘ } };

/***************Klasse**************/
class Examplecomponent extends Component {
    constructor(props) {
        super(props);
        this.state = {example: false};
    }

     /***************Event Handler**************/
    handleChange=()=>{
        this.setState({ example: !this.state.example});
       World.changeTrackerState();
    }

    /***************Render Methode**************/
    render() {
        return ( <Component> );
    }
}
export default Examplecomponent;
```

Code 21: *Beispielaufbau React-Komponente*

Nahezu alle Komponenten der Beispielanwendung folgen dem Aufbau des nebenstehenden Beispiels. Komponenten, welche einen oder mehrere Zustände besitzen, sind Klassenkomponenten. Sie werden durch das Erweitern mit der Klasse ***Component*** erstellt. Der Import von weiteren Klassen, Komponenten, Plug-Ins und unterschiedlichen Dateien ermöglicht den Zugriff auf diese. Hier wird auch die Datei mit der AR-Logik importiert und ermöglicht React so den Zugriff auf dessen Funktionen. All dies geschieht am im Abschnitt Importe.

Verschiedene Styles können mithilfe des Attributs ***style*** zu den Komponenten hinzugefügt werden.

Innerhalb der Klasse gibt es den Konstruktor, die Event-Handler-Methoden und die Render-Methode. Der Konstruktor wird in der Beispielapplikation normalerweise dafür verwendet einen Zustand zu initialisieren, Event-Handler-Methoden an eine Instanz zu binden und ***props*** für die ganze Klasse zugänglich zu machen. Die Event-Handler-Methoden sorgen meistens für die Änderung eines Zustands oder für die Verarbeitung von anderen Ereignissen. Im Beispiel sorgt die Methode für eine Zustandsänderung von ***example*** und den Aufruf einer Funktion der AR-Logik. Dies ist möglich, da das benötigte Objekt aus der Datei exportiert und von React importiert wurde. Alle in der Render-Methode platzierten Inhalte werden in der Komponente gerendert. Hier gilt es zu beachten, dass es immer eine Hauptkomponente geben muss, in welche die weiteren Elemente verschachtelt werden. Jede Komponente kann verschiedene Attribute besitzen, wie im Beispiel z.B. ***style*** oder ***onClick***.

Abschließend folgt der Export, damit die Komponente auch in anderen Komponenten wiederverwendet werden kann.

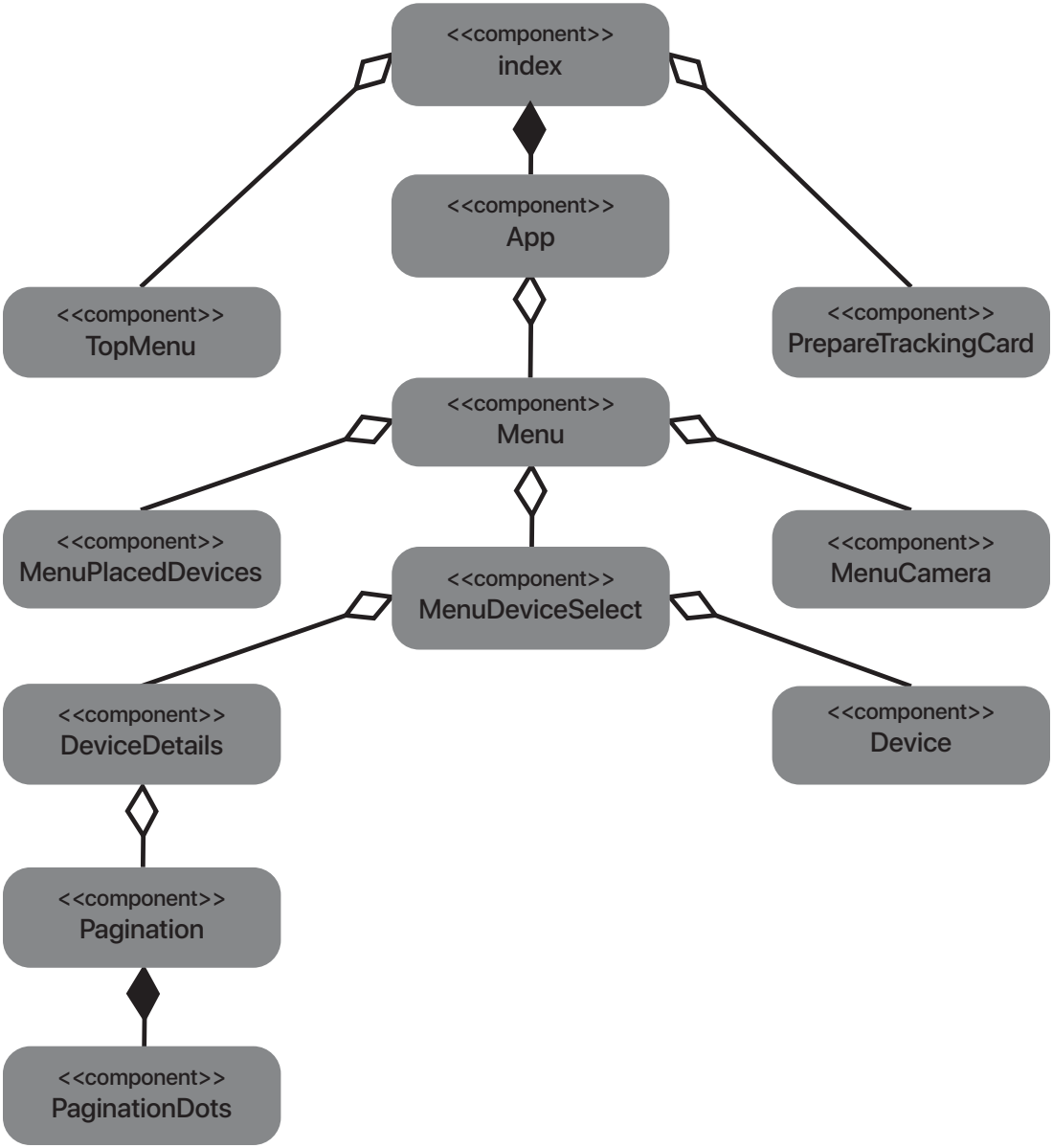

Abb. 48: *React Klassendiagramm Beispielapplikation*

Abb. 48 zeigt den Aufbau der Komponenten bei der Beispielapplikation. Den Startpunkt bildet die Datei ***index.js***. In dieser Datei werden alle Komponenten mit der Funktion ***ReactDOM.render()*** (siehe in Codebeispiel 1) in den Root-DOM-Knoten (Codebeispiel 22) gerendert. Dieser befindet sich in der Datei ***index.html***. Alles was sich innerhalb dieses Knotens befindet, wird von React verwaltet.

```
<div id="root"></div>
```

Code 22: *Datei: index.html Root-DOM-Knoten*

Die wichtigste Komponente in der Beispielapplikation ist die Datei ***Menu.js***. Sie ist sowohl für die Navigation als auch für die Darstellung der Inhalte zuständig. Die verschiedenen Ansichten des Menüs werden in weiteren Komponenten definiert. Alle Komponenten, welche sich im Menu-Kontext befinden, können im Fall der Beispielapplikation den Zustand des Menüs verändern. Ein Kontext ermöglicht die Freigabe von Daten für alle festgelegten Komponenten. So kann das Menü von jeder Komponente, je nach Auswahl des Nutzers, ein- oder ausgefahren werden.

Im Anschluss gibt es einen Überblick über die verwendeten Plug-Ins für React und deren Funktion im Beispielprojekt.

Material UI: Material UI bietet dem Entwickler React-Komponenten zur Erstellung einer Nutzeroberfläche im Stil von Googles Material Design an. So werden z. B. vorgefertigte Layout-Elemente, Navigations-Elemente, Eingabe-Elemente oder Icons angeboten. Durch das bekannte Design findet sich der Nutzer schnell zurecht. Da Material UI auf dem Mobile First-Konzept beruht, ist es ideal zur Erstellung von Applikationen für mobile Geräte geeignet. Das Design kann vom Entwickler beliebig angepasst werden. [182]

React Touch: Da React nicht explizit auf die Nutzung von Geräten mit Touchscreens ausgelegt ist, wurde das Plug-In React Touch verwendet. React Touch stellt eine Reihe an Wrapper-Komponenten zur Verfügung. Diese erleichtern es dem Entwickler, verschiedene Touchgesten zu unterscheiden und darauf zu reagieren. [183]

React Swipeable Views: Ähnlich wie React Touch React stellt auch Swipeable Views eine Wrapper-Komponente zur Verfügung. Mit ihr kann der Entwickler auf Swipes zu reagieren. Dabei kann zwischen Swipes von Oben, Unten, Links und Rechts unterschieden werden. Innerhalb eines Wrappers können sich mehrere Komponenten befinden. Zwischen diesen kann dann mithilfe eines Swipes gewechselt werden. [184]

React Albus: React Albus ermöglicht es dem Entwickler, den Nutzer Schritt für Schritt durch einen Prozess zu leiten. Dabei muss der Entwickler nur den Inhalt für die verschiedenen Schritte festlegen. Das Management der Zustände und den Wechsel der Inhalte übernimmt React Albus. [185]

7.4.5 Installation auf Android und iOS

Abschließend erfolgt die Nennung der wichtigsten Schritte zur Installation der fertigen Applikation auf Android und iOS. Eine AR-Applikation kann in einem Emulator wegen der fehlenden Kamera nicht richtig getestet werden. Deshalb sind die Installation und der Test auf dem Gerät sehr wichtig. Nicht betrachtet wird die Veröffentlichung der Applikation im App- oder Play-Store.

Android: Um eine Applikation vom Computer auf einem Android-Smartphone zu installieren, müssen zuerst die Entwickler-Einstellungen freigeschaltet werden. Hierzu geht man folgendermaßen vor:

1. Einstellungen öffnen
2. Menüpunkt Telefoninfo auswählen
3. 7 Mal hintereinander den Unterpunkt Build-Nummer berühren
4. In das übergeordnete Menü zurückkehren

Nun sollte der Menüpunkt Entwickleroptionen sichtbar sein.

5. Menüpunkt Entwickleroptionen auswählen
6. USB-Debugging erlauben [182]

iOS: Um die Beispielapplikation auf einem iPhone installieren zu können, wird ein sogenanntes Distribution Certificate von Apple benötigt. Um dieses Zertifikat zu erhalten, müssen folgende Schritte durchgeführt werden:

1. Mitgliedschaft im iOS Developer-Programm abschließen (kostenpflichtig)
2. Erstellung eines Provisioning-Profils im iOS Provisioning Portal
3. Mithilfe des Development Provisioning Assistenten ein Zertifikat erstellen.
4. Profil und Zertifikat in Xcode hinzugefügen [186]

Nachdem diese Schritte erfolgreich durchgeführt worden sind, sollte der Installation auf dem Smartphone nichts mehr im Weg stehen. Um alles für die Installation vorzubereiten, müssen folgende Befehle im Terminal des Root-Verzeichnisses der Beispielapplikation ausgeführt werden.

1. Einen Build der React-Applikation erzeugen:

cd meinereactapplikation

npm run build

cd ..

2. Einen Cordova-Build für iOS und Android erzeugen.

cordova prepare

cordova build

Die erzeugten, plattformspezifischen Projekte sollten bei iOS in Xcode und bei Android in Android Studio geöffnet werden. Dies hat den Vorteil, dass die Protokollierung sowohl während der Installation als auch während der Benutzung stattfindet.

Eine genaue Anleitung zur Installation auf dem jeweiligen Betriebssystem findet man unter folgenden Links.

iOS: https://help.apple.com/xcode/mac/current/#/

Android: https://developer.android.com/studio/intro

7.5 Fazit

In diesem Abschnitt folgt die Zusammenfassung der gewonnenen Erkenntnisse von Konzeption und Implementierung.

7.5.1 Konzeption

Durch die Unterscheidung zwischen Muss-, Soll- und Wunsch-Anforderungen ist klar festgelegt, welche Funktionen bei der Erstellung der Applikation zwingend berücksichtigt werden müssen. Das Use-Case-Diagramm stellt eine solide Grundlage für die Implementierung dar. Durch das bereits vorhandene Corporate Design von Five-Konzept gestaltet sich die Erstellung eines Styleguides für die Applikation sehr einfach. Angelehnt an die Website von Five-Konzept, ist das Design sehr minimalistisch. Mit Orange,

Grau, Schwarz und Weiß kommen lediglich vier Farben zum Einsatz. Insgesamt liegt der Fokus bei der Gestaltung der Applikation weniger auf Text, sondern vermehrt auf Bildern und Icons. Der grundlegende Aufbau orientiert sich dabei an erfolgreichen AR-Applikationen wie Ikea Place oder Roomle. Dies ermöglicht dem Nutzer eine schnelle und intuitive Bedienung.

7.5.2 Umsetzung

Die Erstellung der Grundfunktionalität mit React, Wikitude und Cordova nahm am meisten Zeit in Anspruch. Da diese Software bisher selten in Kombination miteinander eingesetzt wurde, mussten bis zu einem funktionierenden Prototyp einige Probleme gelöst und viele Anpassungen gemacht werden. Dazu zählt die Verknüpfung von React und Cordova durch einen automatischen Kopiervorgang des Production-Builds der React-Applikation in den ***www***-Ordner von Cordova. Änderungen an der Webpack-Konfiguration der React-Applikation ermöglichen die Kommunikation der AR-Logik mit der Nutzeroberfläche. Damit die AR-View und die Web-View mit der Nutzeroberfläche zusammen angezeigt werden, waren abschließend noch einige Anpassungen in Cordova und Wikitude notwendig.

Durch die umfangreichen Beispiele von Wikitude ist ein schneller Start gewährleistet. Webentwickler mit JavaScript-Kenntnissen benötigen für die Entwicklung mit Wikitude dank der guten Dokumentation nur wenig Einarbeitungszeit.

Die Wiederverwendbarkeit der Komponenten und die vorgefertigten Elemente von Material UI tragen zu einer schnellen Umsetzung der Nutzeroberfläche bei. Auch bei der Bedienung der Applikation sind durch die Nutzung von verschiedenen Plug-Ins alle bekannten Standardfunktionen, wie z. B. das Öffnen und Schließen eines Menüs durch Swipes, möglich.

Sobald die Applikation fertig entwickelt ist, muss ein Production-Build erstellt werden. Dieser wird zur Installation auf dem Smartphone benötigt. Mit React und Cordova funktioniert dies in wenigen Schritten. Vor der ersten Installation sind die plattformspezifischen Vorbereitungen nötig. Dazu gehört bei Android z. B. die Freischaltung der Entwickleroptionen auf dem Gerät und bei Apple die Erstellung eines Sicherheitszertifikats. Im Anschluss steht der Installation der Applikation mit Xcode oder Android Studio auf dem Gerät nichts mehr im Wege.

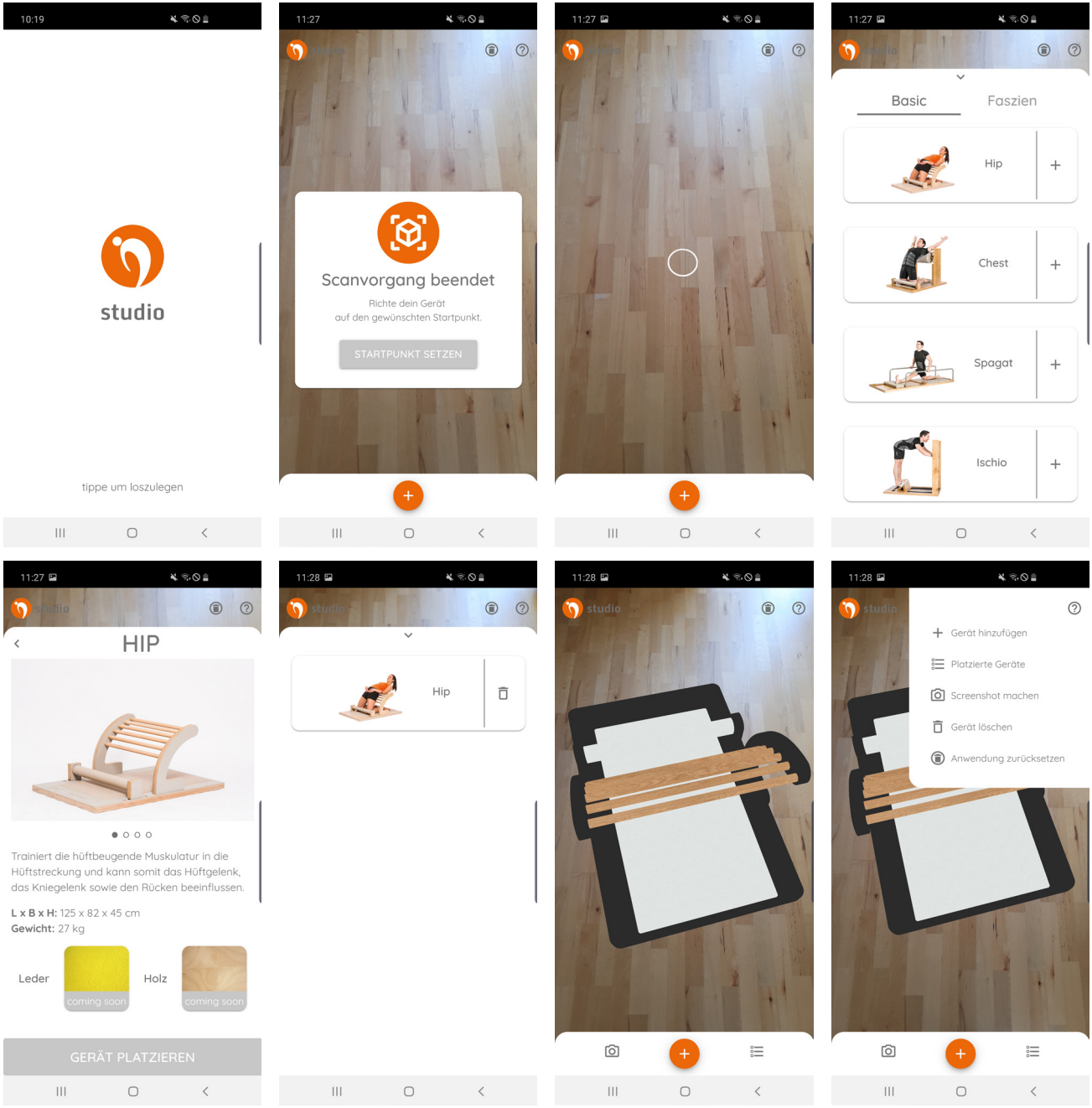

Abb. 49: *Screenshots der fertigen Beispielapplikation*

8 Resümee und Ausblick

Augmented Reality wird eine große Zukunft vorausgesagt. Dank der vielen Anwendungsfelder und den ständig besserwerdenden Werkzeugen zur Erstellung von Anwendungen werden AR-Technologien zunehmend zum Einsatz kommen. Ob das Wachstum dabei hinter den Statistiken zurückbleibt oder sie sogar übertrifft, wird sich in den nächsten Jahren zeigen.

In der Thesis wurden die möglichen Technologien zur Erstellung einer AR-Anwendung vorgestellt und getestet. Anschließend folgte der Vergleich verschiedener Softwares mit dieser Technologie, um das beste Leistungspaket zu ermitteln.

8.1 Stand der Technik

Im aktuellen Vergleich liegen die betriebssystemübergreifenden Technologien vorne. Besonders in den Punkten Performance und Funktionsumfang haben sie einen großen Vorsprung gegenüber den webbasierten Technologien.

Betriebssystemübergreifende Technologien bieten aktuell den besten Mix aus guter Performance, umfangreicher Funktionalität und plattformübergreifender Entwicklung an. Auch die große Auswahl an SDKs zur Entwicklung ermöglicht dem Entwickler, das für sich passende Paket auszuwählen. Neben den zahlungspflichtigen SDKs gibt es auch einige kostenlose Alternativen.

Webbasierte AR-Technologien entstanden erst in den letzten Jahren und können daher noch nicht den gewünschten Performance- und Funktionsumfang bieten. Bald soll jedoch die WebXR Device API Augmented Reality in allen wichtigen Browsern ins Web bringen. Durch die Nutzung einer eigenen, komplett unabhängigen Implementation zur Darstellung von AR-Inhalten im Browser, bietet auch das Start-Up 8th Wall großes Potential. Bis diese Technologien ausgereift sind, kann es nicht mehr lange dauern. Die Leistung auf den Smartphones ist auf jeden Fall gegeben.

Außer bei 8th Wall wird bei den meisten genannten Technologien im Hintergrund auf ARKit und ARCore zugegriffen. Falls eine Applikation also nur auf eine bestimmte Zielplattform ausgeliefert werden soll, sind native Technologien nach wie vor die beste Option. Sie sind kostenfrei nutzbar und bieten umfangreiche Funktionen und gute Performance.

8.2 SDKs für betriebssystemübergreifende Anwendungen

Anhand der festgelegten Zielsetzung wurden sieben SDKs zur Bewertung mit dem BRR-Model identifiziert. Es gibt zwar Kategorien zur Bewertung vor, diese können aber entsprechend der Zielsetzung frei gewichtet werden.

Schon während der Bewertung lagen die etablierten Anbieter Wikitude und Vuforia meist mit deutlichem Abstand zur Konkurrenz auf den ersten Plätzen. Auch in der Gesamtbewertung bestätigt sich dieses Bild. Gemessen an den festgelegten Kriterien liegt Wikitude klar auf Platz 1, Vuforia auf Platz 2 und Viro React auf Platz 3. Der Rest der bewerteten SDKs unterscheidet sich bezüglich der Gesamtpunktzahl nur marginal.

8.3 MVP mit ausgewählten SDKs

Bei der Umsetzung der MVPs konnte mit allen drei SDKs sehr schnell eine funktionsfähige Applikation mit den festgelegten Mindestanforderungen erstellt werden. Der Test der MVPs hat die zuvor theoretisch erstelle Auswertung noch einmal bestätigt. So hat Wikitude auch beim Praxistest insgesamt am besten abgeschnitten. Jedoch können die beiden anderen SDKs auch guten Gewissens weiterempfohlen werden. Mitunter hängt es auch von den persönlichen Präferenzen des Entwicklers ab, ob die Implementierung mit Cordova (Wikitude), Unity (Vuforia) oder in React Native (ViroReact) erfolgen soll.

8.4 Beispielapplikation

Nachdem die anfänglichen Schwierigkeiten bei der Verknüpfung von Cordova, Wikitude und React gelöst waren, schritt die Entwicklung der Applikation zügig voran. Einen maßgeblichen Beitrag dazu haben die guten Dokumentationen, ausführlichen Beispiele und die große Community von allen verwendeten Softwares geleistet. Das Ergebnis ist eine vollfunktionsfähige AR-Applikation zum Platzieren von Geräten der Five-Konzept GmbH & Co. KG in der Umgebung des Nutzers.

8.5 Ausblick

Die Investitionen von großen Konzernen wie Apple oder Microsoft und Applikationen wie Pokémon GO oder Ikea Place zeigen, dass AR schon in naher Zukunft nicht mehr aus dem Alltag wegzudenken ist. Höchstwahrscheinlich tragen die webbasierten AR-Technologien einen großen Teil zu dieser Entwicklung bei. Da bei Web-Applikationen keine Installation mehr notwendig ist, wird die Hürde zur Verwendung einer AR-Applikation für den Nutzer deutlich geringer. Das große Konzerne wie Google, Mozilla und Samsung die WebXR Device API vorantreiben und auch das Start-Up 8th Wall von großen Investoren wie Northwest Venture Partners Anfang 2018 mit knapp 8 Millionen ausgestattet wird, bestätigt diesen Trend. [60]

Die Beispielapplikation für die Five-Konzept GmbH & Co. KG unterstützt zwar die grundlegenden Funktionen, jedoch gibt es noch einige Möglichkeiten zur Erweiterung der Applikation.

Die Auswahl des Materials und der Lederfarbe ist schon im Design der Applikation berücksichtigt worden. Sie erlaubt es dem Nutzer, das ausgewählte Modell nach seinem Geschmack zu gestalten. Auch bei bereits platzierten Objekten könnte eine Implementierung dieser Auswahl stattfinden. Um ein komfortables Hinzufügen mehrerer Geräte zu ermöglichen, könnte eine vorherige Auswahl mit anschließender Platzierung genutzt werden.

Aktuelle AR-Applikationen wie z. B. Roomle oder Ikea Place zeigen, was hier sinnvoll und technisch möglich ist.

Auch die Speicherung von Tracking-Daten könnte eine gute Erweiterung für die App darstellen. Das würde die Fortsetzung der aktuellen Sitzung zu einem späteren Zeitpunkt ermöglichen. Durch eine Speicherung von verschiedenen Varianten des Five-Geräteparcours könnte der Nutzer diese besser vergleichen, ohne sie jedes Mal neu platzieren zu müssen.

Da Wikitude in naher Zukunft das Rendering von Reflektionen und die Einbeziehung der Lichtverhältnisse unterstützen soll, wäre eine Erweiterung der Applikation mit diesen Funktionen denkbar. Sie sorgt für eine noch stärkere Verschmelzung zwischen den virtuellen Objekten und der Realität. ARCore und ARKit bieten diese Funktionen bereits, weitere Softwareherstellen werden hier nachziehen.

Alles in allem wird sich Augmented Reality so entwickeln, dass man in naher Zukunft die Grenze zwischen Virtualität und Realität nur noch schwer erkennen kann.

9 Verzeichnisse

9.1 Abkürzungsverzeichnis

AR	Augmented Reality (erweiterte Realität)
VR	Virtual Reality (virtuelle Realität)
MR	Mixed Reality (gemischte Realität)
AV	Augmented Virtuality (erweiterte Virtualität)
HDM	Head-Mounted Display (Am Kopf befestigter Bildschirm)
GPS	Global Positioning System (Globales Positionierungssystem)
SDK	Software Development Kit (Softwareentwicklungskit)
FFI	Foreign Function Interface (Schnittstelle für Fremdfunktionen)
SLAM	Simultaneous Location and Mapping-Tracking (gleichzeitige Lokalisierung und Kartierung)
3DOF	Three Degrees of Freedom (drei Freiheitsgrade)
6DOF	Six Degrees of Freedom (sechs Freiheitsgrade)
MEMS	mikroelektromechanisches System
TUI	Tangible User Interfaces (anfassbare Benutzerschnittstelle)
API	Application Programming Interface (Programmierschnittstelle)
UWP	Universal Windows Platform (universelle Windows-Plattform)
URL	Uniform Resource Locator (Internetadresse)
FAQ	Frequently Asked Questions (häufig gestellte Fragen)
DOM	Document Object Model (objektorientiertes Dokumentenmodell)
VDOM	Virtual Document Object Model (virtuelles objektorientiertes Dokumentenmodell)
MVP	Minimum Viable Product (minimal überlebensfähiges Produkt)
CLI	Command Line Interface (Kommandozeile)
EWK	Extend of World Knowledge (Wissen über die Welt)
RF	Reproduction Fidelity (Reproduktionstreue)
EPM	Extent of Presence Metaphor (Ausmaß der Präsenz)
DGPS	Differential Global Positioning System (Differentielles globales Positionierungssystem)
NFT	Natural Feature Tracking (Erkennung natürlicher Merkmale)

wt3	Wikitude 3D Format Datei
bin	Binärdatei
usdz, obj, fbx, gltf, collada	Dateiformate zur Speicherung von 3D-Objekten

9.2 Glossar

Web-View: Eine Web-View ist eine Browser-Engine, welche in der Anwendung enthalten ist. So wird es ermöglicht, die Nutzeroberfläche der Anwendung in HTML, CSS und JavaScript zu schreiben. [187]

CRA (Create React App): Create React App ist ein Werkzeug zur schnellen Erstellung einer React-Applikation mit vordefinierten Einstellungen. [179]

Open-Source-Software: Der Begriff wird in der Enzyklopädie der Wirtschaftsinformatik folgendermaßen definiert: „Gemeint ist damit die freie Verfügbarkeit des Software-Quellcodes, der im Rahmen von Open-Source-Lizenzmodellen unentgeltlich genutzt und verändert wer-den kann." [188]

Tracking: Unter Tracking versteht man im Fall von Augmented Reality die Verfolgung von Objekten oder Schlüsselpunkten in der Umgebung.

Rendering: Rendering beschreibt den Vorgang, der aus einem 3D-Modell in einer Software ein Bild oder Video erzeugt. [189]

Marker: Marker bestehen aus einem eindeutigen Muster, welches von der Applikation erkannt werden kann. Die AR-Software hat Vergleichsmuster gespeichert. Diese werden mit dem Muster aus dem Kamerabild abgeglichen und lösen im Fall einer Übereinstimmung ein festgelegtes Ereignis aus.

Debugger: Im Lexikon von Gruenderszene.de wird ein Debugger folgendermaßen definiert: „Der englische Begriff Debugger kommt von dem englischen Begriff bug, welcher übersetzt Programmfehler bedeutet. Bei einem Debugger handelt es sich um ein Werkzeug, welches zum Auffinden von Fehlern in Computersystemen genutzt wird." [190]

Styleguide: Ein Styleguide enthält sämtliche Richtlinien zur Gestaltung von z. B. einem Produkt oder einer Marke, um für ein einheitliches Erscheinungsbild zu sorgen.

Wrapper: Ein Wrapper lässt sich „als eine Software, die wiederum mindestens eine oder auch mehrere Software-Komponenten umhüllt" definieren. [191]

Build: „Als Build bezeichnet die Softwareentwicklung einerseits den gesamten Prozess der Erzeugung einer kompletten, eigenständig lauffähigen Software und andererseits das Ergebnis dieses Prozesses: das oder die ausführbaren Programme einschließlich aller eventuell benötigten Ressourcen." [192]

Immersion: Immersion bedeutet im Zusammenhang mit Augmented und Virtual Reality, dass der Nutzer vollständig in die computergenerierte Welt eintaucht und sie als Realität wahrnimmt.

Hit-Testing: Bei Hit-Tests sendet das Smartphone des Nutzers ein „Strahl" aus, z. B. wenn dieser auf den Bildschirm tippt. Anschließend wird der Kollisionspunkt mit der realen Welt zurückgegeben. Diese Informationen können von der API zum Positionieren eines virtuellen 3D-Modells in der realen Welt genutzt werden. [56]

WebGL: „WebGL (Web Graphics Library) ist eine JavaScript-API zum Rendern interaktiver 3D und 2D Grafiken mittels eines kompatiblen Web-Browsers ohne Einsatz zusätzlicher Plug-Ins. Mit WebGL steht eine API zur Verfügung, die an OpenGL ES 2.0 angelehnt ist und deren Inhalte mittels eines <canvas> Elements dargestellt werden." [193]

OpenGL: „OpenGL (Open Graphics Library) ist [...] eine der Standard-Programmierschnittstellen (Application Programming Interface, API) der Computer-Industrie für die Definition von 2D- und 3D-Grafiken. Vor OpenGL musste jedes Unternehmen, das eine grafische Anwendung herstellte, den Grafikanteil für jede Betriebssystemplattform neu schreiben und sich zudem erst einen Überblick über die verbaute Grafikhardware verschaffen. Mit OpenGL kann eine Anwendung dieselben Effekte in jedem Betriebssystem mit einem beliebigen Grafikadapter mit OpenGL-Konformität einsetzen." [194]

Chrome Canary, Mozilla XR Viewer: Die Entwicklerversionen der Browser sind mit experimentellen Features z. B. im Bereich Augmented Reality ausgestattet. Sie dienen vor allem zum Test neuer Funktionen, bevor diese für alle Nutzer verfügbar gemacht werden.

Corporate Design: Das Corporate Design ist das „visuelle Erscheinungsbild eines Unternehmens im Rahmen und zur Unterstützung der von der Corporate Identity vorgegebenen Ziele. Das Corporate Design soll das Unternehmen nach innen und außen als Einheit erscheinen las-sen, bes. durch formale Gestaltungskonstanten, z.B. Firmenzeichen (Logo), Typografie, Hausfarbe." [195]

9.3 Abbildungsverzeichnis

9.4 Tabellenverzeichnis

9.5 Quellcodeverzeichnis

9.6 Literaturverzeichnis

[1] K. Leswing, "Apple CEO Tim Cook thinks augmented reality will be as important as 'eating three meals a day,'" 2016. [Online]. Available: https://www.businessinsider.de/apple-ceo-tim-cook-explains-augmented-reality-2016-10?r=US&IR=T. [Accessed: 29-Jul-2019].

[2] F. Richter and Statista, "Augmented Reality a $50 Billion Opportunity?," 2017. [Online]. Available: https://www.statista.com/chart/8633/virtual-and-augmented-reality-forecast/. [Accessed: 16-Jul-2019].

[3] Statista, "Forecast augmented and mixed reality software segment breakdown 2022," 2016. [Online]. Available: https://www.statista.com/statistics/610066/worldwide-forecast-augmented-and-mixed-reality-software-assumptions/. [Accessed: 17-Jul-2019].

[4] A. Donath, "Augmented Reality: Apple soll 2020 Brillen auf den Markt bringen," 2019.

[5] A. Floemer, "Google Maps: AR-Navigation für Fußgänger kommt bei ersten Nutzern an," 2019. [Online]. Available: https://t3n.de/news/google-maps-ar-navigation-fuer-fussgaenger-kommt-bei-ersten-nutzern-an-1143074/. [Accessed: 24-Jul-2019].

[6] O. Bimber and R. Raskar, "Spatial Augmented Reality Merging Real and Virtual Worlds," 2005.

[7] D. S. Tschanz Nathaly, "Augmented und Mixed Reality – was ist das?," in Augmented und Mixed Reality: für Marketing, Medien und Public Relations, 2nd ed., Konstanz/München: UVK Verlagsgesellschaft, 2017, pp. 19–44.

[8] A. B. Craig, Understanding augmented reality : concepts and applications. Morgan Kaufmann, 2013.

[9] D. Schmalstieg and T. Höllerer, Augmented reality : principles and practice. Addison-Wesley Professional, 2016.

[10] I. E. Sutherland, "The Ultimate Display," 1965.

[11] L. B. Rosenberg, "Virtual fixtures as tools to enhance operator performance in telepresence environments," 1993, vol. 2057, pp. 10–21.

[12] P. Milgram, H. Takemura, A. Utsumi, and F. Kishino, "Augmented Reality: A class of displays on the reality-virtuality continuum," 1994.

[13] R. T. Azuma, "A Survey of Augmented Reality," 1997.

[14] R. Dörner, W. Broll, P. Grimm, and B. Jung, Virtual und Augmented Reality (VR/AR). Springer-Verlag Berlin Heidelberg, 2013.

[15] B. Furht, Handbook of augmented reality. Springer, 2011.

[16] P. Christensson, "Framework Definition," 2013. [Online]. Available: https://techterms.com/definition/framework. [Accessed: 26-Jun-2019].

[17] L. Voss, "This year in JavaScript 2018 Review," 2018.

[18] A. Mepani, "Most Popular Front End JavaScript Framework In The World," 2019. [Online]. Available: https://www.c-sharpcorner.com/article/most-popular-front-end-javascript-framework-in-the-world/. [Accessed: 03-Jul-2019].

[19] React, "React Übersicht." [Online]. Available: https://reactjs.org/. [Accessed: 07-May-2019].

[20] C. Gackenheimer, Introduction to React, vol. 84. Springer Science+Business Media New York 2015, 2015.

[21] Facebook, "React: Components and Props," 2019. [Online]. Available: https://reactjs.org/docs/components-and-props.html. [Accessed: 03-Jul-2019].

[22] Facebook, "React: State and Lifecycle," 2016. [Online]. Available: https://reactjs.org/docs/state-and-lifecycle.html. [Accessed: 24-Jul-2019].

[23] J. Tillmann, "Was ist eigentlich ein (virtuelles) DOM?," 2017. [Online]. Available: https://t3n.de/news/eigentlich-virtuelles-dom-858160/. [Accessed: 07-May-2019].

[24] S. Deutsch, "Die Flux-Architektur und React," 2015. [Online]. Available: https://reactjs.de/artikel/react-flux-architektur/. [Accessed: 04-Jul-2019].

[25] O. Filipova, Learning Vue.js 2 : learn how to build amazing and complex reactive web applications easily with Vue.js. 2016.

[26] Vue, "Vue Übersicht." [Online]. Available: https://vuejs.org/. [Accessed: 07-May-2019].

[27] A. Freeman, Pro Angular. 2017.

[28] Angular, "Angular Übersicht." [Online]. Available: https://angular.io/. [Accessed: 07-May-2019].

[29] J. Cowart, "What is a Hybrid Mobile App?," 2012. [Online]. Available: https://www.telerik.com/blogs/what-is-a-hybrid-mobile-app-. [Accessed: 26-Jun-2019].

[30] Appfigures, "Statistik: Most popular non-native Tools," 2018. [Online]. Available: https://blog.appfigures.com/ios-developers-ship-less-apps-for-first-time/. [Accessed: 04-Jul-2019].

[31] R. Steyer, Cordova - Entwicklung plattformneutraler Apps. Springer Fachmedien Wiesbaden GmbH 2017, 2017.

[32] The Apache Software Foundation, “Apache Cordova Plugins,” 2018. [Online]. Available: https://cordova.apache.org/. [Accessed: 08-May-2019].

[33] Facebook, “React Native: A framework for building native apps using React.” 2017.

[34] Appcelerator, “Titanium SDK.” [Online]. Available: https://www.appcelerator.com/titanium/titanium-sdk/4/. [Accessed: 08-May-2019].

[35] R. Pot, “Building native apps with JavaScript: Titanium,” 2017. [Online]. Available: https://medium.com/all-titanium/building-native-apps-with-javascript-titanium-5139c855302e. [Accessed: 08-May-2019].

[36] DATACOM Buchverlag, “Definition Augmented Virtuality,” 2017. [Online]. Available: https://www.itwissen.info/AV-augmented-virtuality.html. [Accessed: 25-May-2019].

[37] Onlinemarketing.de, “Definition Virtual Reality.” [Online]. Available: https://onlinemarketing.de/lexikon/definition-virtual-und-augmented-reality. [Accessed: 25-May-2019].

[38] N. Tschanz, Dirk Schart, “Wie funktionieren Augmented und Mixed Reality?,” in Augmented und Mixed Reality, UVK Verlagsgesellschaft mbH, 2017, pp. 45–64.

[39] M.-H. Yang, “Face Detection,” in Encyclopedia of Biometrics, 2015, pp. 447–452.

[40] Wikitude, “Wikitude Objekt- und Szenenerkennung.” [Online]. Available: https://www.wikitude.com/augmented-reality-object-scene-recognition/. [Accessed: 11-Jun-2019].

[41] J. Carmigniani, B. Furht, M. Anisetti, P. Ceravolo, E. Damiani, and M. Ivkovic, “Augmented reality technologies, systems and applications,” in Multimedia Tools and Applications, vol. 51, no. 1, Springer Science+Business Media, LLC 2010, 2011, pp. 341–377.

[42] G. Wells, “The Future of iOS Development: Evaluating the Swift Programming Language,” C. Sr. Theses, Jan. 2015.

[43] J. Staudemeyer, Android mit Kotlin - kurz und gut. dpunkt Verlag, 2018.

[44] Statista, “Anzahl der Apps in den Top App-Stores,” 2018. [Online]. Available: https://de.statista.com/statistik/daten/studie/208599/umfrage/anzahl-der-apps-in-den-top-app-stores/. [Accessed: 01-Jul-2019].

[45] M. Rouse, “Definition Native App,” 2013. [Online]. Available: https://de.ryte.com/wiki/Native_App. [Accessed: 09-May-2019].

[46] W. Wang, Beginning ARKit for iPhone and iPad. Springer Science+Business Media New York 2018, 2018.

[47] Apple Developer, “ARKit Überblick.” [Online]. Available: https://developer.apple.com/arkit/.

[Accessed: 14-May-2019].

[48] Apple Developer, "Augmented Reality: ARKit 3," 2019. [Online]. Available: https://developer.apple.com/augmented-reality/arkit/. [Accessed: 06-Jun-2019].

[49] Apple Developer, "Augmented Reality," 2019. [Online]. Available: https://developer.apple.com/augmented-reality/. [Accessed: 06-Jun-2019].

[50] Google Developers, "ARCore: Fundamental concepts." [Online]. Available: https://developers.google.com/ar/discover/concepts. [Accessed: 14-May-2019].

[51] Google Developers, "ARCore: Cloud Anchors Developer Guide for iOS," 2019. [Online]. Available: https://developers.google.com/ar/develop/ios/cloud-anchors-developer-guide-ios. [Accessed: 14-May-2019].

[52] Google, "ARCore Überblick." [Online]. Available: https://developers.google.com/ar/discover/. [Accessed: 14-May-2019].

[53] Google Developers, "ARCore: Supported Devices." [Online]. Available: https://developers.google.com/ar/discover/supported-devices. [Accessed: 14-May-2019].

[54] iPhone-Ticker, "ARCore: Googles ARKit-Konkurrenz unterstützt iOS-Geräte," 2018. [Online]. Available: https://www.iphone-ticker.de/arcore-googles-arkit-konkurrenz-unterstuetzt-ios-geraete-132524/. [Accessed: 14-May-2019].

[55] Google Developers, "Building an augmented reality (AR) application using the WebXR Device API." [Online]. Available: https://codelabs.developers.google.com/codelabs/ar-with-webxr/#2. [Accessed: 15-May-2019].

[56] J. Medley and Google Developers, "Augmented reality for the web," 2018. [Online]. Available: https://developers.google.com/web/updates/2018/06/ar-for-the-web. [Accessed: 15-May-2019].

[57] J. Medley and Google Developers, "Welcome to the immersive web," 2018. [Online]. Available: https://developers.google.com/web/updates/2018/05/welcome-to-immersive. [Accessed: 15-May-2019].

[58] J. Dirksen, Learning Three.js: the JavaScript 3D Library for WebGL. Packt Publishing, 2013.

[59] R. Cabello, "Three.js: Javascript 3D library," 2018. [Online]. Available: https://threejs.org/. [Accessed: 15-May-2019].

[60] L. Matney, "Augmented reality developer tools startup 8th Wall raises $8 million," 2018. [Online]. Available: https://techcrunch.com/2018/02/06/augmented-reality-developer-tools-startup-8th-wall-raises-8-million/?guccounter=1&guce_referrer_us=aHR0cHM6Ly93d-

3cuZ29vZ2xlLmNvbS8&guce_referrer_cs=gQc34i1pJhTZAVObFwdnZQ. [Accessed: 15-May-2019].

[61] 8th Wall, "8th Wall: Products." [Online]. Available: https://www.8thwall.com/products-web.html. [Accessed: 15-May-2019].

[62] C. Fink, "8th Wall: Revolutionize Mobile AR Development," 2018. [Online]. Available: https://virtualrealitypop.com/8th-wall-to-revolutionize-mobile-ar-development-e36252ec8a3. [Accessed: 15-May-2019].

[63] Wikitude, "Wikitude Instant Tracking." [Online]. Available: https://www.wikitude.com/augmented-reality-instant-tracking/. [Accessed: 11-Jun-2019].

[64] PTC, "Vuforia Fusion." [Online]. Available: https://library.vuforia.com/articles/Training/vuforia-fusion-article.html. [Accessed: 11-Jun-2019].

[65] L. Corral, A. Sillitti, and G. Succi, "Mobile Multiplatform Development: An Experiment for Performance Analysis," Procedia Comput. Sci., vol. 10, pp. 736–743, Jan. 2012.

[66] Unity, "Unity Public Relations." [Online]. Available: https://unity3d.com/de/public-relations. [Accessed: 05-Aug-2019].

[67] J. W. Oak and J. H. Bae, "Development of Smart Multiplatform Game App using UNITY3D Engine for CPR Education," Int. J. Multimed. Ubiquitous Eng., vol. 9, no. 7, pp. 263–268, 2014.

[68] M. Vergara, "Vuforia in Unity: Build cross-platform AR apps," 2018. [Online]. Available: https://blogs.unity3d.com/2018/01/15/vuforia-in-unity-build-cross-platform-ar-apps/. [Accessed: 17-May-2019].

[69] Unity, "Unity Partners: Vuforia." [Online]. Available: https://unity3d.com/de/partners/vuforia. [Accessed: 17-May-2019].

[70] Unity, "Unity: Get real with creating AR games and apps," 2018. [Online]. Available: https://unity3d.com/de/how-to/create-AR-games-in-Unity-efficiently. [Accessed: 17-May-2019].

[71] D. Miller, T. Mowrer, and B. Weiers, "Unity: AR Foundation, ARCore and ARKit," 2018. [Online]. Available: https://blogs.unity3d.com/2018/12/18/unitys-handheld-ar-ecosystem-ar-foundation-arcore-and-arkit/?_ga=2.125848909.1327850971.1558076977-2014712053.1557922192. [Accessed: 17-May-2019].

[72] Unity, "Unity: Mobile AR." [Online]. Available: https://unity.com/solutions/mobile-ar. [Accessed: 17-May-2019].

[73] Wikitude, "Download: Wikitude AR SDK for Cordova." [Online]. Available: https://www.wikitude.com/download-wikitude-sdk-for-cordova/. [Accessed: 17-May-2019].

[74] Wikitude, “Getting started: Wikitude SDKCordova Plugin.” [Online]. Available: https://www.wikitude.com/external/doc/documentation/latest/phonegap/. [Accessed: 17-May-2019].

[75] SocialCompare, “Augmented Reality SDK Comparison,” 2019. [Online]. Available: http://socialcompare.com/en/comparison/augmented-reality-sdks#. [Accessed: 24-Jun-2019].

[76] B. Nagaraj, P. Vijayakumar, and E. Pro, “Comparative Study of Augmented Reality SDKs,” J. Autom. Mob. Robot. Intell. Syst., vol. 5, no. 2, pp. 95–104, 2011.

[77] J. P. Miguel, D. Mauricio, and G. Rodríguez, “A Review of Software Quality Models for the Evaluation of Software Products,” Int. J. Softw. Eng. Appl., vol. 5, no. 6, 2014.

[78] OpenBRR, “Business Readiness Rating for Open Source,” 2005.

[79] U. Neumann and S. You, “Natural feature tracking for augmented reality,” IEEE Trans. Multimed., vol. 1, no. 1, pp. 53–64, Mar. 1999.

[80] G. Klein and D. Murray, “Parallel Tracking and Mapping for Small AR Workspaces.”

[81] R. Liu, J. Zhang, K. Yin, J. Wu, R. Lin, and S. Chen, “Instant SLAM Initialization for Outdoor Omnidirectional Augmented Reality,” 2018.

[82] PTC, “Vuforia Erweitertes Tracking.” [Online]. Available: https://library.vuforia.com/content/vuforia-library/en/features/environments/extended-tracking.html. [Accessed: 11-Jun-2019].

[83] PTC, “Vuforia Flächenerkennung.” [Online]. Available: https://library.vuforia.com/content/vuforia-library/en/features/environments/ground-plane-guide.html. [Accessed: 11-Jun-2019].

[84] XLsoft, “Kudan Funktionalität,” 2018.

[85] MAXST, “MAXST Funktionen.” [Online]. Available: https://developer.maxst.com/Features. [Accessed: 12-Jun-2019].

[86] Beyond Reality, “IN2AR Funktionen.” [Online]. Available: https://www.beyondreality.nl/in2ar/. [Accessed: 25-Jul-2019].

[87] VisionStar Information Technology, “EasyAR Plattform, Features, Preise.” [Online]. Available: https://www.easyar.com/view/sdk.html. [Accessed: 13-Jun-2019].

[88] Inglobe Technologies, “ARMedia Funktionen.” [Online]. Available: http://dev.inglobetechnologies.com/features/#overview. [Accessed: 13-Jun-2019].

[89] Wikitude, “Wikitude SMART.” [Online]. Available: https://www.wikitude.com/smart-augmented-reality/. [Accessed: 11-Jun-2019].

[90] Onirix, “Onirix Places Android SDK.” [Online]. Available: https://docs.onirix.com/onirix-sdk/pla-

ces-android. [Accessed: 11-Jun-2019].

[91] Onirix, "Onirix Places." [Online]. Available: https://www.onirix.com/learn-about-ar/geolocation-based-augmented-reality-with-places/. [Accessed: 11-Jun-2019].

[92] Onirix, "Onirix Spaces SDK." [Online]. Available: https://docs.onirix.com/onirix-sdk/spaces-sdk. [Accessed: 11-Jun-2019].

[93] Onirix, "Onirix Spaces." [Online]. Available: https://www.onirix.com/learn-about-ar/spaces-markerless-AR-with-SLAM/. [Accessed: 11-Jun-2019].

[94] Onirix, "Onirix Targets SDK." [Online]. Available: https://docs.onirix.com/onirix-sdk/targets-sdk. [Accessed: 11-Jun-2019].

[95] Onirix, "Onirix Targets." [Online]. Available: https://www.onirix.com/learn-about-ar/marker-based-augmented-reality-with-targets/. [Accessed: 11-Jun-2019].

[96] XZIMG, "XZIMG Funktionen." [Online]. Available: https://www.xzimg.com/Products?nav=product-XAV. [Accessed: 25-Jul-2019].

[97] Viro Media, "ViroReact Plattformen, Features." [Online]. Available: https://viromedia.com/viroreact. [Accessed: 13-Jun-2019].

[98] P. Lamb and artoolkitX, "artoolkitX Funktionen," 2019. [Online]. Available: https://github.com/artoolkitx/artoolkitx/wiki. [Accessed: 25-Jul-2019].

[99] Catchoom, "Catchoom Funktionen." [Online]. Available: https://documentation.catchoom.com/documentation/sdk/sdk-image-recognition-comparison/. [Accessed: 25-Jul-2019].

[100] Bitstars, "DroidAR Funktionen." [Online]. Available: https://bitstars.github.io/droidar/. [Accessed: 25-Jul-2019].

[101] FLARToolKit, "FLARToolKit Funktionen." [Online]. Available: http://www.libspark.org/wiki/saqoosha/FLARToolKit/en. [Accessed: 25-Jul-2019].

[102] Pikkart, "Pikkart Funktionen." [Online]. Available: https://developer.pikkart.com/augmented-reality/products/sdk.aspx. [Accessed: 25-Jul-2019].

[103] Blippar, "Blippar Funktionen." [Online]. Available: https://www.blippar.com/sdk/static/sdk.html. [Accessed: 25-Jul-2019].

[104] Wikitude, "Wikitude Allgemein." [Online]. Available: https://www.wikitude.com/. [Accessed: 11-Jun-2019].

[105] Wikitude, "Wikitude Preise." [Online]. Available: https://www.wikitude.com/store/. [Accessed: 11-Jun-2019].

[106] Viro Media, "ViroReact Allgemein." [Online]. Available: https://viromedia.com/about/. [Accessed: 13-Jun-2019].

[107] Viro Media, "ViroReact Preise." [Online]. Available: https://viromedia.com/signup. [Accessed: 13-Jun-2019].

[108] PTC, "Vuforia Allgemein." [Online]. Available: https://www.ptc.com/en/products/augmented-reality. [Accessed: 11-Jun-2019].

[109] PTC, "Vuforia Preise." [Online]. Available: https://developer.vuforia.com/pricing. [Accessed: 11-Jun-2019].

[110] Onirix, "Onirix Allgemein." [Online]. Available: https://www.onirix.com/ar-sdks/. [Accessed: 11-Jun-2019].

[111] Onirix, "Onirix Preise." [Online]. Available: https://www.onirix.com/pricing/. [Accessed: 11-Jun-2019].

[112] XLsoft, "Kudan Allgemein." [Online]. Available: https://www.xlsoft.com/en/products/kudan/index.html. [Accessed: 12-Jun-2019].

[113] XLsoft, "Kudan Preise." [Online]. Available: https://www.xlsoft.com/en/products/kudan/price.html#faq. [Accessed: 12-Jun-2019].

[114] MAXST, "MAXST Allgemein." [Online]. Available: http://maxst.com/#/en. [Accessed: 12-Jun-2019].

[115] MAXST, "MAXST Allgemein 2." [Online]. Available: http://maxst.com/#/en/aboutus. [Accessed: 12-Jun-2019].

[116] MAXST, "MAXST Preise." [Online]. Available: https://developer.maxst.com/Pricing. [Accessed: 12-Jun-2019].

[117] MAXST, "MAXST Cloud Tracking." [Online]. Available: https://developer.maxst.com/Pricing/CL. [Accessed: 09-Aug-2019].

[118] VisionStar Information Technology, "EasyAR Allgemein." [Online]. Available: https://www.easyar.com/. [Accessed: 13-Jun-2019].

[119] VisionStar Information Technology, "EasyAR Allgemein2." [Online]. Available: https://www.easyar.com/view/contactus.html. [Accessed: 13-Jun-2019].

[120] EasyAR, "EasyAR Cloud Tracking." [Online]. Available: https://www.easyar.com/view/login.html. [Accessed: 09-Aug-2019].

[121] Wikitude, "Wikitude Demos." [Online]. Available: https://github.com/Wikitude/wikitu-

de-sdk-samples. [Accessed: 10-Aug-2019].

[122] Wikitude, “Wikitude Bilderkennung.” [Online]. Available: https://www.wikitude.com/augmented-reality-image-recognition/. [Accessed: 11-Jun-2019].

[123] Wikitude, “Wikitude Mehrfach Bilderkennung.” [Online]. Available: https://www.wikitude.com/augmented-reality-multiple-target-images/. [Accessed: 11-Jun-2019].

[124] Wikitude, “Wikitude Erweitertes Tracking.” [Online]. Available: https://www.wikitude.com/augmented-reality-extended-tracking/. [Accessed: 11-Jun-2019].

[125] Wikitude, “Wikitude Geo Tracking.” [Online]. Available: https://www.wikitude.com/geo-augmented-reality/. [Accessed: 11-Jun-2019].

[126] PTC, “Vuforia Objekterkennung.” [Online]. Available: https://library.vuforia.com/content/vuforia-library/en/features/objects/object-reco.html. [Accessed: 11-Jun-2019].

[127] PTC, “Vuforia Bilderkennung.” [Online]. Available: https://library.vuforia.com/content/vuforia-library/en/features/images/image-targets.html. [Accessed: 11-Jun-2019].

[128] PTC, “Vuforia Mehrfach Bilderkennung.” [Online]. Available: https://library.vuforia.com/content/vuforia-library/en/features/images/multi-target.html. [Accessed: 11-Jun-2019].

[129] PTC, “Vuforia Zylindererkennung.” [Online]. Available: https://library.vuforia.com/content/vuforia-library/en/features/images/cylinder-targets.html. [Accessed: 11-Jun-2019].

[130] PTC, “Vuforia VuMark.” [Online]. Available: https://library.vuforia.com/content/vuforia-library/en/features/objects/vumark.html. [Accessed: 11-Jun-2019].

[131] Wikitude, “Wikitude Cloudspeicher.” [Online]. Available: https://www.wikitude.com/products/wikitude-cloud-recognition/. [Accessed: 11-Jun-2019].

[132] PTC, “Vuforia Externe Kamera.” [Online]. Available: https://library.vuforia.com/content/vuforia-library/en/articles/Solution/external-camera.html. [Accessed: 11-Jun-2019].

[133] PTC, “Vuforia Cloudspeicher.” [Online]. Available: https://library.vuforia.com/content/vuforia-library/en/articles/Training/Cloud-Recognition-Guide.html. [Accessed: 11-Jun-2019].

[134] Wikitude, “Wikitude Gesichtstracking.” [Online]. Available: https://www.wikitude.com/documentation/latest/android/pluginsapi.html. [Accessed: 10-Aug-2019].

[135] PTC, “Vuforia Render Engine.” [Online]. Available: https://library.vuforia.com/articles/Solution/3D-File-Formats. [Accessed: 10-Aug-2019].

[136] Viro Media, “ViroReact Rendering 3D-Resources.” [Online]. Available: https://docs.viromedia.com/docs/3d-objects. [Accessed: 10-Aug-2019].

[137] Viro Media, “ViroReact Rendering 2D-Resources.” [Online]. Available: https://docs.viromedia.com/docs/importing-assets. [Accessed: 10-Aug-2019].

[138] Wikitude, “Wikitude Rendering.” [Online]. Available: https://www.wikitude.com/products/native-and-js-api-comparison/. [Accessed: 10-Aug-2019].

[139] PTC, “Vuforia Funktionen.” [Online]. Available: https://engine.vuforia.com/content/vuforia/en/features.html. [Accessed: 10-Aug-2019].

[140] Viro Media, “ViroReact 360 Grad Videos.” [Online]. Available: https://docs.viromedia.com/docs/video. [Accessed: 10-Aug-2019].

[141] MAXST, “MAXST Plattformen.” [Online]. Available: https://developer.maxst.com/MD/doc/4_1_x/intro. [Accessed: 12-Jun-2019].

[142] XLsoft, “Kudan Plattformen.” [Online]. Available: https://www.xlsoft.com/en/products/kudan/ar-sdk.html. [Accessed: 12-Jun-2019].

[143] Onirix, “Onirix Plattformen.” [Online]. Available: https://docs.onirix.com/onirix-sdk. [Accessed: 11-Jun-2019].

[144] PTC, “Vuforia Plattformen.” [Online]. Available: https://library.vuforia.com/content/vuforia-library/en/getting-started/overview.html. [Accessed: 11-Jun-2019].

[145] Wikitude, “Wikitude Plattformen.” [Online]. Available: https://www.wikitude.com/products/wikitude-sdk/. [Accessed: 11-Jun-2019].

[146] Onirix, “Onirix Demos.” [Online]. Available: https://github.com/onirix-com. [Accessed: 10-Aug-2019].

[147] XLsoft, “Kudan Demos.” [Online]. Available: https://github.com/XLsoft-Corporation?utf8=✓&q=&type=&language=c%23. [Accessed: 10-Aug-2019].

[148] MAXST, “MAXST Demos.” [Online]. Available: https://developer.maxst.com/MD/download/demo. [Accessed: 10-Aug-2019].

[149] VisionStar Information Technology, “EasyAR Demo.” [Online]. Available: https://www.easyar.com/view/download.html#download-nav2. [Accessed: 10-Aug-2019].

[150] PTC, “Vuforia Demos.” [Online]. Available: https://developer.vuforia.com/downloads/samples. [Accessed: 10-Aug-2019].

[151] Viro Media, “ViroReact Demos.” [Online]. Available: https://docs.viromedia.com/docs/quick-start. [Accessed: 10-Aug-2019].

[152] Wikitude, “Wikitude Support.” [Online]. Available: https://support.wikitude.com/support/home.

[Accessed: 11-Jun-2019].

[153] PTC, "Vuforia Support." [Online]. Available: https://developer.vuforia.com/support. [Accessed: 11-Jun-2019].

[154] Onirix, "Onirix Online Trainings." [Online]. Available: https://www.onirix.com/onirix-trainings/. [Accessed: 11-Jun-2019].

[155] XLsoft, "Kudan Support." [Online]. Available: https://www.xlsoft.com/doc/kudan/. [Accessed: 12-Jun-2019].

[156] MAXST, "MAXST Support." [Online]. Available: https://developer.maxst.com/BoardQuestions. [Accessed: 12-Jun-2019].

[157] VisionStar Information Technology, "EasyAR Support2." [Online]. Available: https://answers.easyar.com/questions?sort=votes. [Accessed: 13-Jun-2019].

[158] VisionStar Information Technology, "EasyAR Support." [Online]. Available: https://www.easyar.com/view/support.html. [Accessed: 13-Jun-2019].

[159] Viro Media, "ViroReact Support." [Online]. Available: https://viromedia.com/developers. [Accessed: 13-Jun-2019].

[160] Onirix, "Onirix Support." [Online]. Available: https://www.onirix.com/contact-us/. [Accessed: 10-Aug-2019].

[161] XLsoft, "Kudan unterstützte Dateiformate." [Online]. Available: https://www.xlsoft.com/doc/kudan/3d-models/. [Accessed: 10-Aug-2019].

[162] Viro Media, "ViroReact Debugging." [Online]. Available: https://docs.viromedia.com/docs/develop-with-viro. [Accessed: 10-Aug-2019].

[163] Wikitude, "Wikitude Debugging." [Online]. Available: https://www.wikitude.com/external/doc/documentation/latest/phonegap/workflow.html#debugging. [Accessed: 10-Aug-2019].

[164] Wikitude, "Wikitude Studio: Augmented Reality Creation Management Tool." [Online]. Available: https://www.wikitude.com/products/studio/. [Accessed: 24-Jul-2019].

[165] Wikitude, "Download: Wikitude SDK for React Native." [Online]. Available: https://www.wikitude.com/download-wikitude-sdk-for-react-native/. [Accessed: 24-Jul-2019].

[166] Brave Digital, "Wikitude: React Native Plug.In." [Online]. Available: https://github.com/brave-digital/react-native-wikitude. [Accessed: 24-Jul-2019].

[167] M. Glinz and H. Gall, "Software Engineering Kapitel 4 Spezifikation von Anforderungen," 2005.

[168] Google, “Material Design: Overview.” [Online]. Available: https://material.io/design/material-theming/overview.html#. [Accessed: 08-Jul-2019].

[169] Google, “Material Design Icons,” 2017. [Online]. Available: https://materialdesignicons.com/.

[170] Wikitude, “Download: Wikitude 3D Encoder for Windows.” [Online]. Available: https://www.wikitude.com/download-wikitude-3d-encoder-for-windows/. [Accessed: 24-Jul-2019].

[171] Wikitude, “3D Assets Workflow.” [Online]. Available: https://www.wikitude.com/external/doc/documentation/latest/phonegap/assetsworkflow.html#good-practice-when-working-with-3d-models. [Accessed: 24-Jul-2019].

[172] JetBrains, “IntelliJ IDEA: The Java IDE for Professional Developers by JetBrains,” 2000. [Online]. Available: https://www.jetbrains.com/idea/. [Accessed: 10-Jul-2019].

[173] C. Pierre, “TOP IDE Index,” 2015. [Online]. Available: http://pypl.github.io/IDE.html. [Accessed: 10-Jul-2019].

[174] npm, “npm.” [Online]. Available: https://www.npmjs.com/. [Accessed: 10-Jul-2019].

[175] N. j. Foundation, “Node.js.” [Online]. Available: https://nodejs.org/de/. [Accessed: 10-Jul-2019].

[176] Google Developers, “Download: Android Studio and SDK Tools.” [Online]. Available: https://developer.android.com/studio. [Accessed: 10-Jul-2019].

[177] Apple Developer, “Xcode,” 2018. [Online]. Available: https://developer.apple.com/xcode/. [Accessed: 10-Jul-2019].

[178] npm, “npx.” [Online]. Available: https://www.npmjs.com/package/npx. [Accessed: 11-Jul-2019].

[179] Facebook, “Create React App: Getting Started.” [Online]. Available: https://facebook.github.io/create-react-app/docs/getting-started. [Accessed: 11-Jul-2019].

[180] The Apache Software Foundation, “Apache Cordova.” [Online]. Available: https://cordova.apache.org/#getstarted. [Accessed: 11-Jul-2019].

[181] Facebook, “React: Create a New React App.” [Online]. Available: https://reactjs.org/docs/create-a-new-react-app.html. [Accessed: 24-Jul-2019].

[182] Material-UI, “Material-UI,” 2019. [Online]. Available: https://material-ui.com/. [Accessed: 24-Jul-2019].

[183] L. Aves, “React Touch,” 2018. [Online]. Available: https://www.npmjs.com/package/react-touch. [Accessed: 24-Jul-2019].

[184] O. Tassinari, “React Swipeable Views.” [Online]. Available: https://react-swipeable-views.com/. [Accessed: 24-Jul-2019].

[185] American Express, “React Albus.” [Online]. Available: https://github.com/americanexpress/react-albus. [Accessed: 24-Jul-2019].

[186] Apple, “Signing & Capabilities workflow.” [Online]. Available: https://help.apple.com/xcode/mac/current/#/dev60b6fbbc7. [Accessed: 24-Jul-2019].

[187] PCMag Digital Group, “Definition WebView.” [Online]. Available: https://www.pcmag.com/encyclopedia/term/70186/webview. [Accessed: 19-Jul-2019].

[188] M. Nüttgens, “Definition Open-Source-Software,” 2014. [Online]. Available: http://www.enzyklopaedie-der-wirtschaftsinformatik.de/lexikon/uebergreifendes/Kontext-und-Grundlagen/Markt/Open-Source-Software/index.html. [Accessed: 19-Jul-2019].

[189] Techopedia, “Definition Rendering.” [Online]. Available: https://www.techopedia.com/definition/9163/rendering. [Accessed: 21-Jul-2019].

[190] Vertical Media, “Definition Debugger.” [Online]. Available: https://www.gruenderszene.de/lexikon/begriffe/debugger?interstitial_click. [Accessed: 21-Jul-2019].

[191] DATACOM Buchverlag, “Definition Wrapper.” [Online]. Available: https://www.itwissen.info/Wrapper-wrapper.html. [Accessed: 21-Jul-2019].

[192] S. Augsten, “Was ist ein Build?,” 2018. [Online]. Available: https://www.dev-insider.de/was-ist-ein-build-a-702737/. [Accessed: 21-Jul-2019].

[193] Mozilla Developers, “Definition WebGL,” 2019. [Online]. Available: https://developer.mozilla.org/de/docs/Web/API/WebGL_API. [Accessed: 30-Jul-2019].

[194] M. Rouse, “Definition OpenGL,” 2016. [Online]. Available: https://whatis.techtarget.com/de/definition/OpenGL-Open-Graphics-Library. [Accessed: 30-Jul-2019].

[195] F.-R. Esch, “Definition Corporate Design.” [Online]. Available: https://wirtschaftslexikon.gabler.de/definition/corporate-design-30453. [Accessed: 30-Jul-2019].

10 Anhang

3DMODELSANGESTURESINSTANTRACKING.JS

```
var defaultScaleValue = 0.009;
var defaultRotationValue = 0;
var rotationValues = [];
var scaleValues = [];
var allCurrentModels = [];
var allCurrentModelCategories = [];
var oneFingerGestureAllowed = false;

AR.context.on2FingerGestureStarted = function() {
    oneFingerGestureAllowed = false;
};

var World = {
    platformAssisstedTrackingSupported: false,
    createOverlaysCalled: false,
    requestedModel: -1,
    categorie: null,
    initialDrag: false,
    lastAddedModel: null,

    init: function initFn() {
        AR.hardware.smart.isPlatformAssistedTrackingSupported();
    },

    captureScreen: function captureScreenFn() {
        AR.platform.sendJSONObject({
            action: „capture_screen“
        });
    },
```

```
createOverlays: function createOverlaysFn() {

    if (World.createOverlaysCalled) {
        return;
    }

    World.createOverlaysCalled = true;

    var crossHairsBlueImage = new AR.ImageResource(„assets/marker2.png“, {
        onError: World.onError
    });

    this.crossHairsBlueDrawable = new AR.ImageDrawable(crossHairsBlueImage,
    1.0);
    this.tracker = new AR.InstantTracker({
        onChangedState: function onChangedStateFn(state) {
        },
        deviceHeight: 1.0,
        onError: World.onError,
        onChangeStateError: World.onError,
    });

    this.instantTrackable = new AR.InstantTrackable(this.tracker, {
        drawables: {
            cam: World.crossHairsBlueDrawable,
            initialization: World.crossHairsRedDrawable
        },
        onTrackingPlaneDragBegan: function onTrackingPlaneDragBeganFn(xPos,
        yPos) {
            oneFingerGestureAllowed = true;
            World.updatePlaneDrag(xPos, yPos);
        },
```

```
        onTrackingPlaneDragChanged: function onTrackingPlaneDragChangedFn(xPos,
        yPos) {
            World.updatePlaneDrag(xPos, yPos);
        },
          onTrackingPlaneDragEnded: function onTrackingPlaneDragEndedFn(xPos,
          yPos) {
              World.updatePlaneDrag(xPos, yPos);
              World.initialDrag = false;
          },
          onError: World.onError
      });

      setInterval(
          function () {
              if (World.tracker.canStartTracking) {
                  window.top.postMessage(„tracking“, „*“);
                  World.instantTrackable.drawables.initialization = [World.cross
                  HairsGreenDrawable];
              } else {
                  window.top.postMessage(„nottracking“, „*“);
                  World.instantTrackable.drawables.initialization = [World.cross
                  HairsRedDrawable];
              }
          }, 1000
      );
  },

  updateRequestedModel: function updateRequestedModelFn(categorie, number) {
      World.requestedModel = categorie[number].number;
      World.categorie = categorie;
  },
  updatePlaneDrag: function updatePlaneDragFn(xPos, yPos) {
      if (World.requestedModel >= 0) {
```

```
            World.addModel(World.requestedModel, xPos, yPos);
            World.requestedModel = -1;
            World.initialDrag = true;
        }

        if (World.initialDrag && oneFingerGestureAllowed) {
            World.lastAddedModel.translate = {
                x: xPos, y: yPos
            };
        }
    },

    changeTrackerState: function changeTrackerStateFn() {

        if (this.tracker.state === AR.InstantTrackerState.INITIALIZING) {
            this.tracker.state = AR.InstantTrackerState.TRACKING;
        } else {
            this.tracker.state = AR.InstantTrackerState.INITIALIZING;
            window.top.postMessage(„changeTrackerstate“, „*“);
        }
    },

    addModel: function addModelFn(pathIndex ,xpos, ypos) {
        if (World.isTracking()) {
            var modelIndex = rotationValues.length;
            World.addModelValues();

            var model = new AR.Model(World.categorie[pathIndex].path, {
                scale: {
                    x: defaultScaleValue, y: defaultScaleValue, z: defaultScaleValue
                },
```

```
        translate: {
            x: xpos, y: ypos
        },

        onDragBegan: function ( /*x, y*/) {
            oneFingerGestureAllowed = true;
        },

        onDragChanged: function (relativeX, relativeY, intersectionX,
        intersectionY) {
            if (oneFingerGestureAllowed) {
                this.translate = {
                    x: intersectionX, y: intersectionY
                };
            }
        },

        onRotationChanged: function (angleInDegrees) {
            this.rotate.z = rotationValues[modelIndex] - angleInDegrees;
        },

        onRotationEnded: function () {
            rotationValues[modelIndex] = this.rotate.z
        },

        onError: World.onError
    });

    allCurrentModels.push(model);
    allCurrentModelCategories.push(World.categorie[pathIndex]);
    World.lastAddedModel = model;
```

```
            this.instantTrackable.drawables.addCamDrawable(model);
        }
    },


    isTracking: function isTrackingFn() {
        return (this.tracker.state === AR.InstantTrackerState.TRACKING);
    },

    addModelValues: function addModelValuesFn() {
        rotationValues.push(defaultRotationValue);
        scaleValues.push(defaultScaleValue);
    },

    removeModel: function resetModelsFn(i) {
        this.instantTrackable.drawables.removeCamDrawable(i+1);
        allCurrentModels.splice(i, 1);
        allCurrentModelCategories.splice(i, 1);
    },

    resetModels: function resetModelsFn() {
        this.instantTrackable.drawables.removeCamDrawable(allCurrentModels);
        allCurrentModels = [];
        allCurrentModelCategories = [];
        World.resetAllModelValues();
    },

    resetAllModelValues: function resetAllModelValuesFn() {
        rotationValues = [];
        scaleValues = [];
    },
```

```
    onError: function onErrorFn(error) {
        alert(error);
        window.top.postMessage(„changeTrackerstate“, „*“);
    }
};

AR.hardware.smart.onPlatformAssistedTrackingAvailabilityChanged = function(availabi-
lity) {
    switch (availability) {
        case AR.hardware.smart.SmartAvailability.INDETERMINATE_QUERY_FAILED:
            World.createOverlays();
            break;
        case AR.hardware.smart.SmartAvailability.CHECKING_QUERY_ONGOING:
            break;
        case AR.hardware.smart.SmartAvailability.UNSUPPORTED:

            World.createOverlays();
            break;
        case AR.hardware.smart.SmartAvailability.SUPPORTED_UPDATE_REQUIRED:
            break;
        case AR.hardware.smart.SmartAvailability.SUPPORTED:
            World.platformAssisstedTrackingSupported = true;
            World.createOverlays();
            break;
    }
};

World.init();
export {World, allCurrentModelCategories, allCurrentModels};
```

INDEX.JS

```
var arexperience = {
    „path“: „www/index.html“,
    „requiredFeatures“: [„instant_tracking“],
    „startupConfiguration“: {
        „camera2_enabled“: true, „camera_position“: „back“,
        „camera_resolution“: ‚full_hd‘, „camera_focus_mode“: ‚continuous‘
    }
};

var app = {
    isDeviceSupported: false,
    isArchitectWorldLoaded: false,

    initialize: function() {
        this.bindEvents();
    },

    bindEvents: function() {
        document.addEventListener(‚deviceready‘, this.onDeviceReady, false);
    },

    onDeviceReady: function() {
        app.wikitudePlugin = cordova.require(„com.wikitude.phonegap.WikitudePlugin.
        WikitudePlugin“);

        if ( cordova.platformId == „android“ ) {
            app.wikitudePlugin.setBackButtonCallback(app.onBackButton);
        } else {
            app.wikitudePlugin.setErrorHandler(app.onRuntimeError);
        }
```

```
            app.wikitudePlugin.setJSONObjectReceivedCallback(app.onJSONObjectReceived);
    },

    startAR: function() {
        app.prepareArchitectWorld(arexperience, function() {
            app.loadARchitectWorld(arexperience);
        });
    },

    prepareArchitectWorld: function(architectWorld, successCallback) {
        app.wikitudePlugin.isDeviceSupported(function() {
                app.wikitudePlugin.requestAccess(
                    function() {
                        successCallback();
                    },
                    function(error) {
                        var openAppSettings = confirm(error.userDescription + ,\nOpen
App Settings?‘);
                        if ( openAppSettings == true ) {
                            app.wikitudePlugin.openAppSettings();
                        }
                    },
                    architectWorld.requiredFeatures);
            }, function(errorMessage) {
                alert(errorMessage);
            },
            architectWorld.requiredFeatures);
    },
    loadARchitectWorld: function(architectWorld) {
        app.wikitudePlugin.loadARchitectWorld(function successFn() {
                app.isArchitectWorldLoaded = true;
            },
```

```
            function errorFn(error) {
                app.isArchitectWorldLoaded = false;
                alert(,Loading AR web view failed: , + error);
            },
            architectWorld.path, architectWorld.requiredFeatures, architectWorld.
             startupConfiguration
        );
    },

    onJSONObjectReceived: function (jsonObject) {
        if (typeof jsonObject.action !== ‚undefined‘) {
            if ( jsonObject.action === „capture_screen“ ) {
                app.wikitudePlugin.captureScreen(
                    function(absoluteFilePath) {
                        alert(„Der Screenshot wurde in deiner Galerie
                               gespeichert!“);
                    },
                    function (errorMessage) {
                        alert(errorMessage);
                    },
                    true, null
                );
            }
        }
    },

    saveError: function(error) {
        alert(„Could not save the current instant target.“);
    },

    loadError: function(error) {
        alert(„Could not load instant target, please save it first.“);
    },
```

```
onRuntimeError: function (error) {
        if (error.code == 960) {
            var openAppSettings = confirm(error.message + ,\nOpen App Settings?‘);
            if (openAppSettings == true) {
                app.wikitudePlugin.openAppSettings();
            }
        }
    },
};

app.initialize();
```