

Masterthesis  
im Studiengang  
Medieninformatik  
zur Erlangung des Grades  
Master of Science, M.Sc.

## **Untersuchung und Evaluation der Technologie Azure Spatial Anchors**

Erstbetreuer : Prof. Christoph Müller

Zweitbetreuer : Stefan Haiss

Abgabe : 27.02.2020

Von : Corinna Braun  
Matrikelnummer: 260947  
Rohrstraße 11, 79877 Friedenweiler  
[corinna.braun94@web.de](mailto:corinna.braun94@web.de)



---

## Eidesstattliche Erklärung

Ich erkläre hiermit an Eides statt, dass ich die vorliegende Masterarbeit selbstständig und ohne unzulässige, fremde Hilfe verfasst habe. Alle verwendeten Hilfsmittel und Quellen, welche im Wortlaut oder dem Sinn nach entnommen wurden, werden als solche kenntlich gemacht.

Friedenweiler, den 27. Februar 2020,

---

Corinna Braun



## Abstract - Deutsch

Die Technologie *Azure Spatial Anchors* ermöglicht die cloudbasierte Realisierung von Augmented Reality Inhalten, unabhängig von der Plattform. Es spielt keine Rolle, ob der User schlussendlich ein iOS, Android oder sogar eine Hololens verwendet. Jeder kann durch sein mobiles Endgerät den gleichen Inhalt betrachten und damit interagieren, sofern der Entwickler die Anwendung für die jeweilige Plattform etabliert. Doch wie genau wird die Technologie integriert und wie funktioniert sie? Dies wird im Rahmen dieser Arbeit behandelt. Dabei werden theoretische Grundlagen und allgemeine Begriffe geklärt und definiert. Dies bildet die Grundlage für eine umfassende und detaillierte Analyse der Technologie. Um die Analyse der verschiedenen Klassen und Funktionen tiefer zu festigen, wird die Entwicklung eines Prototyps beschrieben und dargestellt. Neben der Analyse wird zusätzlich die Robustheit der Technologie mit einem Experiment untersucht. Der Fokus liegt dabei auf der Lokalisierung von cloudbasierten Anchors bei Veränderung der Grundszene. Die Ergebnisse wurden statistisch ausgewertet und evaluiert. Als Schlussfolgerung konnte die Hypothese, dass signifikante Unterschiede bei der Lokalisierung von Anchors nach Veränderung der Szene auftreten, abgelehnt werden. Des Weiteren werden mehrere Probleme bei einem experimentellem Ansatz zu der Technologie diskutiert. Basierend auf der Evaluation in Bezug auf Analyse und Experiment, bietet ein Ausblick weitere Forschungsgebiete und Ansätze für die Entwicklung mit Azure Spatial Anchors. Diese Arbeit ist besonders für Entwickler und Begeisterte im Bereich der Augmented Reality Entwicklung mit Unity für Android interessant.



## Abstract - English

The *Azure Spatial Anchors* technology enables the cloud-based realization of augmented reality content, regardless of the platform. It does not matter whether the user uses an iOS, Android or even a Hololens. Anyone can view and interact with the same content through their mobile device, as long as the developer establishes the application for the respective platform. But how exactly is the technology integrated and how does it work? This will be discussed in the context of this thesis. Theoretical basics and general terms will be clarified and defined. This forms the basis for a detailed and comprehensive analysis of the technology. In order to deepen the analysis of the different classes and functions, the development of a prototype is described and illustrated. In addition to the analysis, the robustness of the technology is examined with an experiment. The focus is on the localization of cloudbased anchors when the basic scene changes. The results were statistically analyzed and evaluated. As a conclusion, the hypothesis that significant differences in the localization of anchors occur after changing the scene could be rejected. Furthermore, several problems with an experimental approach to the technology are discussed. Based on the evaluation in terms of analysis and experiment, an outlook offers further research areas and approaches for the development with Azure Spatial Anchors. This work is especially interesting for developers and enthusiasts in the field of augmented reality development with Unity for Android.



## Danksagung

An dieser Stelle möchte ich mich bei Prof. Christoph Müller und Stefan Haiss für die konstruktive Kritik und die kompetente Betreuung bei der Erstellung dieser Masterarbeit bedanken.

Außerdem gilt mein Dank der Firma meaPuna GmbH, welche mir stets mit Rat und Unterstützung beiseite standen. Auch möchte ich mich für die Bereitstellung von technischen Hilfsmitteln für Entwicklung und Test bedanken.

Mein Dank gilt auch meinen Kommilitonen, welche mit Hilfe von spannenden Diskussionen und beidseitigem Austausch dazu beigetragen haben, dass diese Arbeit in ihrer jetzigen Form vorliegt.

Auch meiner Familie und meinen Freunden möchte ich mich für die Unterstützung sowie den emotionalen Rückhalt während dem gesamten Studium danken.

Zusätzlich möchte ich mich auf diesem Wege bei Herrn Kirschenbauer und der Fakultät Digitale Medien für die Geduld und Bereitschaft bei der mehrfachen Nutzung des MSL bedanken.

Corinna Braun, Friedenweiler, 27.02.2020



**Inhaltsverzeichnis**

Inhaltsverzeichnis . . . . .	IX
Abbildungsverzeichnis . . . . .	XI
Tabellenverzeichnis . . . . .	XIII
Abkürzungsverzeichnis . . . . .	XV
Glossar . . . . .	XVII
Wichtige Hinweise . . . . .	XIX
1 Einleitung . . . . .	1
2 Theoretische Basis . . . . .	5
2.1 Definition nach Milgram . . . . .	5
2.2 Technische Umsetzung . . . . .	6
2.3 ARCore . . . . .	11
2.4 ARFoundation . . . . .	11
3 Analyse . . . . .	15
3.1 Erste Schritte . . . . .	16
3.2 Klassen . . . . .	19
3.3 Entwicklung Prototyp und Research-Anwendung . . . . .	35
4 Methodik . . . . .	49

4.1	Aktualität und Forschungsstand . . . . .	49
4.2	Hypothese . . . . .	50
4.3	Testkonzept . . . . .	51
4.4	Erhobene Daten . . . . .	53
5	Experiment . . . . .	55
5.1	Aufbau . . . . .	55
5.2	Erstes technisches Experiment . . . . .	57
5.3	Zweites technisches Experiment . . . . .	58
5.4	Probleme . . . . .	58
5.5	Statistische Auswertung . . . . .	60
6	Evaluation und Ausblick . . . . .	65
6.1	Abschließende Beurteilung Spatial Anchors . . . . .	65
6.2	Abschließende Beurteilung der Experimente . . . . .	68
6.3	Ausblick . . . . .	69
	Literaturverzeichnis . . . . .	71
A	Inhalt des beiliegenden Datenträgers . . . . .	75
B	Statistiken . . . . .	77
C	Klassenstruktur . . . . .	79
D	Screenshots . . . . .	83
E	Statistische Diagramme . . . . .	87
F	Tabellen . . . . .	89

## Abbildungsverzeichnis

Abbildung 2: Darstellung der Spatial Anchors Funktionalität mit cross-platform Anwendung [Micd] . . . . .	2
Abbildung 3: Realitäts-Virtualitäts Kontinuum nach Milgram et. al. [MTUK94]. . . . .	5
Abbildung 4: Konzept Optische-Durchsicht (links) und Video-Durchsicht (rechts) nach Azuma et. al. [ABB <sup>+</sup> 01]. . . . .	7
Abbildung 5: Eigenschaften, welche von AR Foundation unterstützt werden [Unib]. . . . .	14
Abbildung 6: Beispiel eines Anchors in der Web-App. . . . .	45
Abbildung 7: Szenenaufbau bei der Lokalisierung mit Stativ im Zustand ABC (Kategorie-Objekte). Das linke Bild wurde bei Experiment 2 und das rechte bei Experiment 1 aufgenommen. Die Markierungen und Abstände sind bei beiden Experimenten identisch. . . . .	56
Abbildung 8: Sicht auf die Szene durch die Research-Anwendung bei der Lokalisierung. . . . .	57
Abbildung 9: Grafische Darstellung der Datenerhebung aus dem zweiten Test samt Belichtung und Phase. Gelesen von links nach rechts. Gelbe Flächen stehen für 750Lux, dunkelgraue Flächen für 10Lux. . . . .	62
Abbildung 10: Übersicht über die verschiedenen Tarife der Azure-Ressource App Service-Plan für die Kategorie Entwickeln/Testen. Die Abbildungen stammen aus dem Azure Portal und benötigen eine Anmeldung. Unter [Micc] sind die Tarife aufgelistet. . . . .	66

Abbildung 11: Übersicht über die zwei Tarife der Azure-Ressource App Service-Plan für die Kategorie Produktion. Die Abbildungen stammen aus dem Azure Portal und benötigen eine Anmeldung. Unter [Micc] sind die Tarife aufgelistet. . . . .	67
Abbildung 12: Statistik zu PokemonGo von Statista [Dat19]. . . . .	77
Abbildung 13: Klassendiagramm Prototyp und Research-Anwendung. . . . .	80
Abbildung 14: Sequenzdiagramm für die Generierung eines Anchors in der Research-Anwendung. . . . .	81
Abbildung 15: Sequenzdiagramm für die Lokalisierung eines Anchors in der Research-Anwendung. . . . .	82
Abbildung 16: Flowchart Prototyp bei der Generierung eines Anchors. Buttons für Interaktion sind grün umrandet. . . . .	83
Abbildung 17: Flowchart Prototyp bei der Lokalisierung eines Anchors. Buttons für Interaktion sind grün umrandet. . . . .	84
Abbildung 18: Flowchart Research-Anwendung bei der Generierung eines Anchors. Buttons für Interaktion sind grün umrandet. . . . .	85
Abbildung 19: Flowchart Research-Anwendung bei der Lokalisierung eines Anchors. Buttons für Interaktion sind grün umrandet. . . . .	86
Abbildung 20: Boxplot-Darstellung der Millisekunden für die Lokalisierung. Ausreisser werden ausgeschlossen. Erstellt mit SOFA [PS]. . . . .	87
Abbildung 21: Diagramm zum Vergleich der Mittelwerte aller Anchor bei jeder Phase. Erstellt mit SOFA [PS]. . . . .	88

**Tabellenverzeichnis**

Tabelle 2: Eigenschaften der verschiedenen AR Displays, aufgelistet nach Krevelen. [Kre07] . . . . .	8
Tabelle 3: Deskriptive Kennzahlen der erhobenen Daten aus dem zweiten Experiment. . . . .	60
Tabelle 4: Tabelle mit p-Werten des Mann-Whitney-U-Tests mit SOFA. . . . .	61



## Abkürzungsverzeichnis

**AR** Augmented Reality

**AV** Augmented Virtuality

**MR** Mixed Reality

**SDK** Software Development Kit

**HMD** Head-mounted-display

**SLAM** Simultaneous localization and mapping

**6DOF** Six Degrees of Freedom

**WIMP** Windows, Icons, Menus and Pointing

**AAD** Azure Active Directory

**TAP** Task-based Asynchronous Pattern

**UI** User Interface

**API** Application Programming Interface

**BLE** Bluetooth Low Energy

**COM** Concurrent Odometry and Mapping

**URL** Uniform Resource Locator

**URI** Uniform Resource Identifier



## Glossar

Aurale Displays	Akustische Displays, welche auf Ton (Mono, Stereo oder Surround) limitiert sind [Kre07].
Cloud	Der Begriff Cloud stammt von Cloud Computing. Dabei handelt es sich um ein Geschäftsmodell zur Bereitstellung von IT-Dienstleistungen. Diese sind bedarfsorientiert und werden über das Internet zur Verfügung gestellt. Eine dieser Dienstleistungen ist der Cloud-Speicher. Dabei werden individuelle Daten über das Internet an externe Systeme übertragen und dort gespeichert [Chr, Mich].
Odometrie	Der Begriff Odometrie beschreibt die Bewegungsschätzung basierend auf bestimmten Parametern. In diesem Kontext wird auf die visuelle Odometrie referenziert, welche für die Schätzung visuellen Input benötigt. Dafür sind keine Vorkenntnisse der Szene oder der Bewegung notwendig [Dav04].
Olfaktorische Displays	Displays, welche sich auf den Geruchssinn beziehen. Diese sind kaum entwickelt [Kre07].



## **Wichtige Hinweise**

Im Umfang dieser Arbeit werden Themen im Informatikbereich diskutiert. Für ein umfassendes Verständnis ist eine Wissensgrundlage mit Unity und C# von Vorteil. Ein Verständnis zu spezifischen Informatik-Begriffen wird vorausgesetzt.

Einzelne Begriffe in dieser Arbeit werden nicht sinngemäß ins Deutsche übersetzt, damit eine Verbindung zu den im Programmiercode verwendeten Begriffen entsteht. Zum Beispiel der Begriff Anchor (dt. Anker). Auf Grund dessen sind fundierte englische Sprachkenntnisse erforderlich.



## 1. Einleitung

Der Begriff Augmented Reality (AR) etablierte sich bereits in den 90er Jahren und taucht seitdem immer häufiger in Literatur und Forschung auf. Persönlichkeiten wie Azuma oder Milgram veröffentlichten namhafte Werke zu diesem Themengebiet. Ihre Definition von Augmented Reality ist bis heute aktuell und dient als Grundlage der Forschung. Durch die Entwicklung des Smartphones ist es nun auch möglich, AR auf dem mobilen Endgerät für den Normalverbraucher anzubieten. Im Jahr 2018 wurden 57 Millionen Smartphone-Nutzer alleine in Deutschland gezählt<sup>1</sup>. Immer mehr Smartphones unterstützen ARKit (iOS) bzw. ARCore (Android). Dabei handelt es sich um Software Development Kits (SDKs) für die Entwicklung von AR-Anwendungen für das Smartphone. Microsoft geht mit der Technik *Azure Spatial Anchors* noch einen Schritt weiter. Sie ermöglicht die Entwicklung von plattformübergreifenden Multi-User-Anwendungen. Dabei können Orte durch virtuellen Kontext ergänzt und via HoloLens-, iOS- oder Android-Gerät zeitgleich parallel abgerufen werden. Dies ist möglich durch das Setzen von Cloud Anchors, dauerhaften Ankern in der realen Umgebung. Diese Anchors basieren auf visuell generiertem Bildmaterial, sog. Point Clouds<sup>2</sup>. Innerhalb einer solchen Point Cloud wird dieser Anchor platziert. Dabei handelt es sich um Position und Orientierung innerhalb verschiedener Feature Points<sup>3</sup> und in Relation zu diesen. Ausschlaggebend ist diese Positionierung, sie stellt den Ort des Anchors dar, welcher durch virtuelle Inhalte ergänzt wird. Ein solcher virtueller Inhalt könnte beispielsweise ein passendes 3D-Objekt oder eine Audiomeldung sein. Dies bleibt dem Entwickler überlassen, welcher *Azure Spatial Anchors* integriert. In Abbildung 2 wird die Funktionalität der Spatial Anchors vereinfacht dargestellt. In der Mitte des Tisches befindet sich der Cloud Anchor, welcher durch einen virtuellen Inhalt in Form eines gelben 3D-Objekts ergänzt wurde. Um den Tisch herum sind drei unterschiedliche mobile Endgeräte (Bsp. Hololens, Android-Smartphone und iOS-Tablet)

---

<sup>1</sup> <https://de.statista.com/statistik/daten/studie/198959/umfrage/anzahl-der-smartphonennutzer-in-deutschland-seit-2010/>

<sup>2</sup> dt. Punktewolken

<sup>3</sup> dt. Merkmalpunkte, Punkte innerhalb einer Point Cloud

zu sehen. Jedes dieser Endgeräte kann diesen Anchor lokalisieren und den virtuellen Inhalt, also das 3D-Objekt, zeitgleich sehen. Der Anchor bleibt dabei fixiert an seiner Stelle.



Abbildung 2.: Darstellung der Spatial Anchors Funktionalität mit cross-platform Anwendung [Micd].

Azure beschreibt bereits mehrere Ansätze zur Integrierung von Spatial Anchors im täglichen Leben. Ein Anchor kann mit mehreren Anchors verbunden werden und somit eine Beziehung herstellen. Dies ermöglicht beispielsweise die Verwendung von Spatial Anchors in Form von Navigation. Ob spielerisch durch einen Freizeitpark oder durch ein komplexes Firmengebäude, die Umsetzungsmöglichkeiten sind groß. In einer App kann der User die gewünschten Attraktionen oder Orte auswählen, zu welchen er navigiert werden möchte. Der User bewegt sich von Anchor zu Anchor und bekommt möglicherweise noch Informationen oder virtuelle Inhalte angezeigt. Wird ein Anchor erreicht und mit diesem interagiert, aktiviert sich der nächste Anchor. Das Finden von Anchors kann dabei ebenfalls spielerisch wie bei einer Schnitzeljagd oder förmlich mit Richtungsanzeigen oder Pfaden gestaltet werden. Eine weitere Verwendungsmöglichkeit ist die Erweiterung der realen Welt durch dauerhaften virtuellen Kontennt. Dies könnte zum Beispiel ein virtueller Kalender in einem Meetingraum oder Informationen zu einer spezifischen Maschine in einer Produktionshalle sein. Da es sich um digitale Inhalte handelt, können Informationen zentral geändert und aktualisiert werden, ohne zur Maschine oder zum Raum zu gehen.

Die Technologie der *Azure Spatial Anchors* wurde im Jahr 2019 veröffentlicht. Die Technologie ist sehr neu und kaum erforscht, weshalb diese Thematik große Aktualität besitzt. Im Umfang dieser Arbeit wird die Technologie analysiert und evaluiert. Die Entwicklung geschieht mit der Entwicklungsumgebung *Unity* und basiert auf der Plattform *Android*. Diese Abgrenzung schließt iOS Geräte und die HoloLens 2 aus, da das Erscheinungsdatum der HoloLens 2 zum Zeitpunkt der Bearbeitung noch ungewiss war. Die Entwicklung für iOS Geräte bedarf mehrere iOS Entwicklungs- und Testgeräte. Dies befand sich nicht im Entwicklungsspielraum dieser Arbeit. Es soll erforscht werden, wie die Technologie aufgebaut ist und wie sie in eine mobile Anwendung mit Unity integriert werden kann. Auch soll diese Arbeit als Hilfsmittel für künftige Entwickler dienen und ihnen den Einstieg mit der Technologie erleichtern. Auf Basis der Untersuchung wurde ein erster Prototyp entwickelt. Zusätzlich wurde die Robustheit der Technologie mit einem technischen Experiment untersucht. Die Forschungsfrage lautet "*Welchen Einfluss nehmen Veränderungen der Szene innerhalb der Point Cloud auf das Lokalisieren von Spatial Anchors, solange mindestens ein Objekt in der Szene bleibt?*".

Die Entwicklung und Erforschung dieser Technologie fand in Zusammenarbeit mit der Firma *meaPuna GmbH* sowie mit der *Hochschule Furtwangen* statt. meaPuna wurde 2015 gegründet mit Hauptsitz in Hechingen im Zollernalbkreis. Das Unternehmen startete als Dienstleistungsanbieter für Beratung, Entwicklung und Implementierung von SAP-Anwendungen. Dies wurde durch die kundenorientierte Entwicklung von Augmented- und Mixed-Reality Anwendungen erweitert. Dabei entwickelt meaPuna für Smartphone, Tablet und für die Datenbrille HoloLens (Microsoft) innovative Applikationen. Der Schwerpunkt der Entwicklung liegt dabei auf Branchen wie Maschinenbau, Medizintechnik, Facility Management und Architektur. Eine der führenden Lösungen von meaPuna in diesem Bereich ist das Service-Portal *REWEIV*. Dabei handelt es sich um eine Remote-Support-Lösung für Service und Support Teams, welche durch die integrierte Anwendung von Augmented Reality unterstützt werden.



## 2. Theoretische Basis

Im diesem Kapitel wird der Begriff Augmented Reality definiert. Seit den 90er Jahren erscheint der Ausdruck immer häufiger in Literatur und Forschung. Die Technologie Augmented Reality wird in diesem Zusammenhang unterschiedlich definiert.

### 2.1. Definition nach Milgram

Die augmentierte Realität steht in engem Zusammenhang zur virtuellen Realität. Das Realitäts-Virtualitäts-Kontinuum nach Milgram stellt die Terminologie in einem Schaubild dar (Abbildung 3).

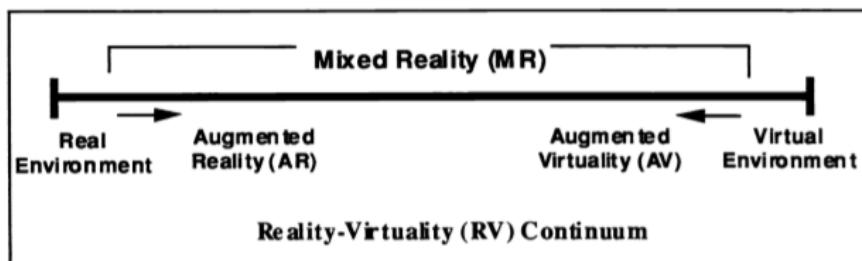


Abbildung 3.: Realitäts-Virtualitäts Kontinuum nach Milgram et. al. [MTUK94].

Milgram et. al. unterscheiden die reale Umgebung<sup>1</sup> und die virutelle Umgebung<sup>2</sup>, auch virtuelle Realität. Dazwischen liegt die gemischte Realität, auch Mixed Reality (MR). Die reale Umgebung besteht ausschließlich aus realen Objekten, welche in Person, durch ein Fenster oder über ein Display betrachtet werden. Im Gegensatz dazu steht die virtuelle Umgebung, welche alleine aus virtuellen Objekten besteht. Diese werden über ein Display oder immersiv durch ein Head-mounted-display (HMD) betrachtet. Augmented Reality befindet sich im Bereich der Mixed Reality. Hier unterscheiden

<sup>1</sup> engl. Real Environment

<sup>2</sup> engl. Virtual Environment

sich AR und Augmented Virtuality (AV). Im Jahr 1997 definiert Azuma drei Kriterien, welche ein AR System erfüllen müssen. [Azu97, ABB<sup>+</sup>01, Kre07]

- Kombiniert reale und virtuelle Objekte in einer realen Umgebung.
- Reale und virtuelle Objekte werden aneinander ausgerichtet und registriert.
- Ermöglicht Interaktivität in Echtzeit, sowie Registration in drei Dimensionen.

## 2.2. Technische Umsetzung

Für die Verwendung von AR sind einige technische Anforderungen notwendig. Krevelen schreibt auch, dass AR einen größeren Anspruch an technischen Hilfsmittel benötigt als VR. Die Kernelemente sind Displays, Tracking-Systeme, Grafikrechner und die jeweilige Software. Diese hat bereits Ivan Sutherland in den 60er Jahren definiert. [Kre07, Sut68]

### 2.2.1. Displays

Krevelen unterscheidet verschiedene Arten von Displays. Dazu zählen aurale, visuelle, haptische, olfaktorische und geschmackliche Displays. In dieser Arbeit werden nur visuelle Displays thematisiert. Krevelen bezieht sich hierbei auf die Kategorisierung nach Azuma. Es werden die drei Kategorien aufgeteilt: Head-worn-, Hand-held- und Projektive Displays. [Kre07, ABB<sup>+</sup>01]

#### Head-worn Displays

Diese Art von Display trägt der User auf dem Kopf. Die Bilder werden direkt vor den Augen angezeigt. Bei den Head-worn Displays gibt es zwei Arten, die Video-Durchsicht<sup>3</sup> und die Optische-Durchsicht<sup>4</sup>. Die Optische-Durchsicht basiert auf den ersten Ansätzen nach Sutherland. In Figur 4 ist ein Konzept zum Aufbau beider Technologien dargestellt. [Kre07, ABB<sup>+</sup>01]

*Video-Durchsicht* Die Video-Durchsicht nutzt Videoaufnahmen der am Kopf getragenen Videokamera als Hintergrund für die darübergelegten AR Elemente. Das Display ist Opaque.

---

<sup>3</sup> engl. video see-through

<sup>4</sup> engl. optical see-through



Abbildung 4.: Konzept Optische-Durchsicht (links) und Video-Durchsicht (rechts) nach Azuma et. al. [ABB<sup>+</sup>01].

*Optische-Durchsicht* Bei der Optischen-Durchsicht bestehen die Displays aus transparenten Spiegeln und Linsen. Die AR Elemente werden dann darüber gelegt. Der Hintergrund ist die mit den Augen wahrgenommene Realität.

### Hand-held Displays

Bei Hand-held Displays gilt es ebenfalls Video-Durchsicht und Optische-Durchsicht zu unterscheiden. Der Ansatz der Durchsicht bleibt gleich wie bei den Head-worn Displays. Die Hand-held Displays bieten einen Vorteil, da sie kostengünstig in der Produktion und leicht zu handhaben sind. Krevelen beschreibt bereits 2007, dass diese Displays die beste Möglichkeit zur Integrierung von AR im Massenmarkt sind. Möhring et. al. haben 2004 ein Paper über die Verwendung von Video-Durchsicht AR auf einem mobilen Endgerät verfasst. Dies ermöglicht die Integrierung von AR auf bereits existierenden Konsumerprodukten. Ein Optisches-Durchsicht Hand-held Display wurde von Stetten et al. 2001 im Bereich der Medizin entwickelt, welches das Ultraschallbild direkt über dem Organ abbildet. [Kre07, MLB04, SCHB01]

### Projektive Displays

Die projektiven Displays benötigen keine speziellen Brillen. Die Projektion kann sich auf großen Flächen befinden, welche eine hohe Field-of-view ermöglichen. Außerdem kann die Oberfläche aus einfachen Wänden sowie aus komplexen Konstruktionen bestehen.

## Abschließende Beurteilung

Bereits 2004 wurden erste Ansätze zur Integrierung von AR in mobile Geräte entwickelt. Dieser Ansatz hat sich in der heutigen Zeit, wie vorrausgesagt, am stärksten durchgesetzt. Das Smartphone ist ein fester Bestandteil der Gesellschaft geworden. Erfolgreiche Anwendungen haben AR mittels Video-Durchsicht implementiert. Ein Beispiel ist das Spiel PokemonGo, welches 2016 erschien. Es ist im Jahr 2019 immer noch das weltweit umsatzstärkste Spiel (Anhang B) [Dat19]. Abschließend sind in Tabelle 2 die Eigenschaften der verschiedenen Technologien mit jeweiliger Positionierung aufgelistet. Die Tabelle ist der nach Krevelen nachempfunden.

Positionierung	Head-worn			Hand-held
Technik	Optisch	Video	Projektiv	Alle
Mobile Verwendung	+	+	+	+
Aussengebrauch	+ / -	+ / -	-	+ / -
Interaktion	+	+	+	+
Helligkeit und Kontrast	-	+	Limitiert	+
Auflösung	Wachsend	Wachsend	Wachsend	Limitiert
Field-of-view	Limitiert	Limitiert	Wachsend	Limitiert
Akkomodation	+ / -	+	Limitiert	+ / -
Möglichkeiten	Aktuelle Dominanz			Realistisch, Massenmarkt
Nachteile und Herausforderungen	Abstimmung und Tracking	Verzögerungen	Retroreflektives Material	Prozessor- und Speicherlimits

Tabelle 2.: Eigenschaften der verschiedenen AR Displays, aufgelistet nach Krevelen.  
[Kre07]

### 2.2.2. Tracking und Registrierung

Eine der größten Herausforderungen bei AR ist das Tracking und die daraus resultierende Registrierung. Die Technologien in diesem Bereich entwickeln sich stets weiter, weshalb als Referenz für dieses Kapitel möglichst aktuelle Paper verwendet wurden. Für optimales Tracking werden sechs Grade der Freiheit<sup>5</sup> angestrebt: Drei Variablen für die Position (x,y,z) und drei Winkel für die Orientierung (yaw, pitch, und roll). Für eine optimale Registrierung ist die Erstellung eines Modells der Umgebung notwendig. Viele frühere Ansätze, zur Zeit von Krevelen, erforderten die Vorbereitung der Umgebung in Form von 3D Modellen. [Kre07] Wang et al. beschreiben 2016 neue Ansätze für das Tracking und die Registrierung. Diese Ansätze funktionieren auch in einer unvorbereiteten Umgebung. Im folgenden sind die Trackingmethoden nach Wang et al. aufgeführt. [WON16]

#### Tracking

Wang et al. definieren drei Arten des Tracking. Das Tracking mit Markern, ohne Marker und mittels Sensoren. Am stärksten wird das Tracking mit Markern verwendet. Seit etwa 2010 exsitiert jedoch auch ein Trend für die markerlosen und sensor-baiserten Trackingmethoden.

*Marker-basiertes Tracking* Die markerbasierte Trackingmethode wird am häufigsten verwendet. Sie bietet eine hohe Genauigkeit, Flexibilität und eine einfache Bedienung. Beispiele für eine Implementierung von markerbasiertem Tracking bietet die Plattform Vuforia.

*Markerloses Tracking* Die markerlose Trackingmethode wird als potenzieller Ersatz der Marker in Betracht gezogen. Dabei kommt die Simultaneous localization and mapping (SLAM) Technik zum Einsatz. Diese Technik erzeugt eine Karte der Umgebung und verfolgt gleichzeitig die aktuelle Position des Users. SLAM bedeutet übersetzt: Gleichzeitige Lokalisierung und Mapping in Echt-Zeit. Dabei handelt es sich um ein Kamera-Tracking System, welches die 3D Struktur einer unvorbereiteten Szene erstellen kann. Das monoSLAM System nach Davidson ist dabei ein Trackingansatz mit einer einzelnen Kamera. Es wird auch als gefiltertes SLAM bezeichnet, da es einen

---

<sup>5</sup> Six Degrees of Freedom (6DOF)

erweiterten Kalman-Filter verwendet. Der Kalman-Filter ist ein mathematisches Verfahren. Es soll Fehler bei Messwerten reduzieren und zusätzlich noch Schätzungen für nicht messbare Größen bestimmen.

*Sensor-basiertes Tracking* Die Trackingmethode mit Sensoren wurde bereits vor der Etablierung von Markern in AR Systemen verwendet. Diese Technik verwendet unterschiedliche Sensoren wie magnetische, akustische, optische und / oder mechanische Sensoren. Das bildbasierte Tracking basiert auf Computer-Bild-basierten Techniken. Der Vorteil der Sensoren sind die hohe Genauigkeit, die niedrige Latenz sowie die gute Anpassungsfähigkeit an verschiedene Umgebungssituation. Jedoch hat jede Sensorart ihre eigenen Vor- und Nachteile.

## Registrierung

Der Grund, weshalb das Tracking bei AR Systemen so ausschlaggebend ist, ist die richtige Registrierung der virtuellen Objekte. Dies bedeutet, dass sich die Inhalte mit gleichbleibender, relativer Position und Orientierung im augmentierten Raum befinden. Bei der Registrierung treten zwei Probleme auf: Genauigkeit und Verzögerungen. Fehler bei der Genauigkeit entstehen durch falsche Ausrichtung der Sensoren und fehlerhafte Tracking-Ergebnisse. Verzögerungen entstehen durch zeitliche Unterschiede zwischen Bewegung des Users und der Antwort des Bildes auf die neue Position. [WON16]

### 2.2.3. User Interface und Interaktion

Das klassische Desktop User Interface (UI) mit Windows, Icons, Menus and Pointing (WIMP) kommt für AR Systeme nicht in Frage. Bei 6DOF wird Interaktion benötigt. Jedoch im Gegensatz zu 2D Systemen schmälern typische Eingabegeräte wie Maus und Tastatur das AR Erlebnis. AR Interfaces müssen also folgende Interaktionen unterstützen: Selektion, Positionierung und Drehung von virtuellen Objekten, sowie Zeichnen von Pfaden und Eingabe von Werten und Text. In seinem Paper *Augmented Reality: Technologies, Applications, and Limitations* beschreibt Krevelen mehrere UI Ansätze genauer. [Kre07]

### 2.3. ARCore

ARCore ist eine Plattform von Google für die Entwicklung von Augmented Reality Inhalten. Dabei verwendet ARCore verschiedene APIs, welche zum Teil über Android und iOS verfügbar sind. Google beschreibt die Funktion von ARCore wie folgt: "*ARCore enables your phone to sense its environment, understand the world and interact with information*" [ARC]. Die Kernfunktionen, um virtuelle Inhalte in die reale Welt durch die Kamera eines Smartphones zu integrieren, sind Bewegungsverfolgung, Verständnis der Umwelt und Licht-Schätzung. Dabei nutzt ARCore eine markerlose Tracking-Methode namens Concurrent Odometry and Mapping (COM)<sup>6</sup> zur Verfolgung der Position des mobilen Endgeräts während es sich bewegt. Diese Technologie wurde von Google patentiert und ist verwandt mit der zuvor beschrieben Technologie SLAM [NLZ]. Sie erzeugt ein eigenes Verständnis der realen Welt. Für das Tracking kommt die Kamera des Smartphones zum Einsatz. Über die Kamera werden interessante Punkte der Umgebung, auch Feature Points, festgelegt. Bei Bewegung des Smartphones wird dann beobachtet, wie sich diese Punkte bei einer Bewegung verändern. Die Kombination aus dieser Bewegung und interner Trägheitssensoren des Smartphones erzeugt ein Verständnis über Position und Orientierung des mobilen Geräts im Raum. Zusätzlich zu der Identifizierung von den sogenannten Feature Points, kann ARCore flache Oberflächen erkennen und die durchschnittliche Beleuchtung der Umgebung abschätzen.

### 2.4. ARFoundation

ARFoundation ist ein Framework von Unity, welches verschiedene Augmented-Reality-Programmierschnittstellen vereint. Dazu zählen ARKit (iOS), ARCore (Android), Magic Leap und HoloLens. In Abbildung 5 sind alle Kernfunktionen, welche ARFoundation unterstützt, aufgelistet. Durch die Vereinigung verschiedener Plattformen ist eine cross-platform Entwicklung möglich. Bishop et. al. definieren diesen Begriff für Software, welche durch verschiedene Versionen für mehrere Plattformen verfügbar ist. Dem Gegenüber stehen die Begriffe Multiplattform bzw. Plattformunabhängig. Dies bedeutet, dass die Software an keine Plattform gebunden ist und ohne Veränderung auf mehreren Plattformen funktioniert. [Jud06]

Folgende Funktionalitäten werden durch ARFoundation unterstützt: [Unia]

---

<sup>6</sup> dt. Gleichzeitige Odometrie und Kartographie

- Welt-Tracking (World tracking): Verfolgung der Position und Ausrichtung des mobilen Endgeräts im physischen Raum.
- Ebenenerkennung (Plane detection): Erkennen von horizontalen und vertikalen Flächen.
- Erkennung von Punkte-Wolken.
- Referenzpunkte (Reference Points<sup>7</sup>): Eine beliebige Position und Ausrichtung, welche das mobile Endgerät verfolgt.
- Lichtschätzung (Light estimation): Schätzung der durchschnittlichen Farbtemperatur und Helligkeit in physikalischen Raum.
- Umgebungssonden (Environment probes): Ein Mittel zum Erzeugen einer Würfelkarte, um einen bestimmten Bereich der physikalischen Umgebung darzustellen.
- Gesichtsverfolgung (Face tracking): Erkennen und verfolgen von menschlichen Gesichtern.
- Bildverfolgung (Image tracking): 2D-Bilder erkennen und verfolgen.

AR Foundation basiert auf Subsystemen. Ein Subsystem ist eine plattformunabhängige Schnittstelle zum Abrufen verschiedener Informationen. Die AR-Subsysteme sind in dem Package `UnityEngine.XR.ARSubsystems` vereint. Allerdings sind auch einige Subsysteme in anderen Packages notwendig. Deshalb muss wenigstens eins der plattformspezifischen AR Packages im Package Manager installiert werden: ARKit XR Plugin und/oder ARCore XR Plugin. Jedes Subsystem ist für spezifische Funktionalitäten zuständig. Das `XRPlaneSubsystem` stellt beispielsweise die Oberflächenerkennung zur Verfügung. [Unia]

Die Packages ARCore XR Plugin und ARKit XR Plugin liefern dabei die Implementierung von ARCore und ARKit für viele der AR Subsysteme. Diese konkrete Implementierung eines Subsystems wird als Provider definiert. Die Provider unterscheiden sich in der Unterstützung bestimmter Funktionen. Daher besitzt jedes Subsystem einen Deskriptor, der angibt, welche Funktionen des Subsystems unterstützt

<sup>7</sup> Der Begriff *Reference Points* wird in dieser Arbeit dem Begriff *Feature Points* gleichgesetzt.

werden. Jeder Provider entscheidet selbst, wie ein Subsystem implementiert wird. Generell verpackt der Provider die native SDK einer Plattform.

**Beispiel ARCore PlaneTracking** Im Unity Projekt in dem Ordner *Packages* liegen die im *Package Manager* installierten Erweiterungen. Unter *AR Foundation > Runtime > AR* befinden sich die Skripte zur Verwendung von AR Funktionalitäten. Diese sind bereits plattformunabhängig. Unter *AR Subsystems > Runtime* sind die verschiedenen Subsystem-Klassen und deren Deskriptoren abgelegt. In *AR Subsystems > Runtime > PlaneTracking* befindet sich beispielsweise das `XRPlaneSubsystem` und der `XRPlaneSubsystemDescriptor`. Der Deskriptor speichert dabei, ob das Subsystem horizontale, vertikale und beliebig abgewinkelte Erkennung sowie Begrenzungspunkte für die Ebenen unterstützt. Durch das ARCore XR Plugin wird das Skript `ARCorePlaneProvider` in *ARCore XR Plugin > Runtime* hinzugefügt. Dies ist der Provider, welcher das `XRPlaneSubsystem` für ARCore implementiert. Diese Klasse erbt das Subsystem. Die Basis aller Subsysteme ist der `SubsystemLifecycleManager`. Diese Klasse ist generisch, ihr wird ein `TSubsystem` und ein `TSubsystemDescriptor` übergeben. Klassen, welche diese erben sind: `ARCameraManager`, `AROcclusionManager`, `ARRaycastManager`, `ARSession` und der `ARTrackableManager`. Letztere ist wiederum eine generische Klasse, welcher ein `TSubsystem`, ein `TSubsystemDescriptor` (Basis `SubsystemLifecycleManager`) und zusätzlich noch ein `TSessionRelativeData` und `TTrackable` übergeben werden. Das `TSessionRelativeData` ist ein struct und liefert die Daten, welche vom Subsystem bereitgestellt werden. Das `TTrackable` ist der Typ jener Komponente, welche diese Klasse verwalten soll (bsp. erstellen, aktualisieren oder zerstören). Die Klasse `ARPlaneManager`, welche bereits oben genannt wurde, erbt ebenfalls den `ARTrackableManager`. Dabei sind die übergebenen Klassen an die generische Klasse das `XRPlaneSubsystem`, der `XRPlaneSubsystemDescriptor`, die Klasse `BoundedPlane` und die Klasse `ARPlane`. Das `ARPlane` Objekt ist dabei die Fläche, welche von einem AR Device erkannt wird.

Unity's AR Foundation Supported Features				
Functionality	ARKit	ARCore	Magic Leap	HoloLens
Pass-through video	●	●		
Device tracking	●	●	●	●
Raycast	●	●	●	●
Plane tracking	●	●	●	●
Reference points	●	●	●	●
Point cloud detection	●	●	●	
Gestures	●	●	●	●
Face tracking	●	●		
2D image tracking	●	●	●	●
3D object tracking	●			
Environment probes	●	●	●	
Meshing			●	●
2D & 3D body tracking	●			
Human segmentation and occlusion	●			
Collaborative participants	●			

Abbildung 5.: Eigenschaften, welche von AR Foundation unterstützt werden [Unib].

### 3. Analyse

Dieses Kapitel beschreibt die Technologie der *Azure Spatial Anchors*. Durch die intensive Auseinandersetzung soll ein tieferes Verständnis gebildet werden. Dies kann für zukünftige Leser die Möglichkeit bieten, die Arbeit mit dieser Technologie zu erleichtern. Das Kapitel beinhaltet die ersten Schritte bei der Entwicklung für Android mit Unity, den Aufbau des Unity Projekts und die wichtigsten Klassen der SDK. Anschließend wird der `SpatialAnchorManager` beschrieben und die im Umfang dieser Arbeit entwickelten Anwendungen. Azure liefert den Manager mit der SDK als zusammenfassende Klasse. In diesem Teil werden auch mehrere Code-Beispiele aufgeführt und erläutert. Die Grundlage für die Analyse bietet die Microsoft-Dokumentation und die Azure Spatial Anchors Application Programming Interface (API) [Micf, Micb].

Um ein einheitliches Verständnis der nachfolgenden Kapitel zu gewährleisten, müssen verschiedene Begriffe definiert werden. Der Begriff **Anchor** steht hier übergeordnet für eine exakte Position und Orientierung innerhalb einer Point Cloud. Dieser Anchor wird meist durch ein 3D-Objekt repräsentiert (Bsp. meaPuna-Firmenlogo). Hier unterscheiden sich der native bzw. lokale und der cloudbasierte Anchor. Ist von **Spatial Anchors** die Sprache, so ist die Technologie *Azure Spatial Anchors* gemeint. Der Begriff **Prefab** ist von Unity definiert. Dabei handelt es sich um ein 3D-Objekt bzw. eine Sammlung von 3D-Objekten mit bestimmten Eigenschaften, Materialien oder Komponenten, welche als Ganzes gespeichert werden. Ein Prefab befindet sich dann in den Assets und hat den Typ `GameObject`. Jedes Objekt in Unity ist ein `GameObject`, sie dienen als eine Art Container für Komponenten. Für mehr Informationen ist es ratsam, die Unity-Dokumentation zu Rate zu ziehen. In dieser Arbeit treten vermehrt die Begriffe **Script** bzw. **Skript** und **Class** bzw. **Klasse** auf. Ein Skript ist hierbei die C#-Datei, welche den Code für die gleichnamige Klasse enthält.

### 3.1. Erste Schritte

Um die Technologie *Azure Spatial Anchors* nutzen zu können ist ein Azure-Konto notwendig. Microsoft bietet eine ausführliche Anleitung zum Einrichten der **Azure Spatial Anchors Account Ressource** [Micf]. Als Ressource werden bei Microsoft Azure Einheiten bezeichnet, welche von Azure verwaltet werden. Dazu zählen beispielsweise virtuelle Maschinen, Netzwerke, Server, Datenbanken und viele mehr. Ressourcen werden in einer Ressourcen-Gruppe verwaltet. In der Ressourcen-Gruppe `SpatialResourceGroupCB` befinden sich drei Ressourcen: `SpatialAnchorsWebAppCB` vom Typ *App Service*, `SpatialServicePlanCB` vom Typ *App Service-Plan* und `SpatialResourceCB` vom Typ *Spatial Anchors Account*. Letztere ist für die Entwicklung mit *Azure Spatial Anchors* unerlässlich. Für die Entwicklung mit Unity sind bestimmte Voraussetzungen notwendig. Die Unity Version muss 2019.1.10f oder höher sein. Zur Installation der wichtigen Android SDKs ist Android Studio notwendig. Folgende SDKs müssen in Android Studio installiert werden. Im Startbildschirm befindet sich die Option *Configure*, durch welche man in den SDK Manager gelangt. Dieser unterscheidet die Optionen *SDK Platforms* und *SDK Tools*.

- *SDK Platforms*: Die SDK für die jeweiligen Plattformen. Für die Verwendung ist Android 7.0 oder höher notwendig. Dies ist abhängig vom jeweiligen mobilen Endgerät. Auf Grund dessen sollten die Plattform-SDKs ab Android 7.0 aufwärts installiert werden. [ARC]
- *SDK Tools*
  - Android SDK Build-Tools
  - NDK
  - CMake
  - Android Emulator
  - Android SDK Platform-Tools
  - Android SDK Tools

Microsoft Azure bietet ein fertiges Unity Demo-Projekt über GitHub an. Beim Öffnen des Projekts sind bereits alle Plugins, Packages und Einstellungen richtig konfiguriert und geladen. Bei einem eigenen Unity-Projekt sind mehrere Einstellungen notwendig. Unter <https://github.com/Azure/azure-spatial-anchors-samples/releases> ist die SDK

mit allen notwendigen Skripten verfügbar. In dieser Arbeit wird die Version Azure Spatial Anchors v1.3.3 verwendet. Folgende Packages müssen im Package Manager unter *Window > Package Manager* installiert werden. In Klammern dahinter die für diese Arbeit verwendeten Versionen:

- AR Foundation (2.1.4)
- XR Legacy Input Helpers (2.0.6)
- Multiplayer HLAPI (1.0.4)
- ARCore XR Plugin (2.1.2)
- ARKit XR Plugin (2.1.2)
- Windows Mixed Reality (1.0.14)

Nach Microsoft Azure ist zusätzlich die .NET Core 2.2 SDK und ein Windows Computer mit ASP.NET und Webentwicklungs-Workload (Visual Studio 2017 oder höher) notwendig. Außerdem wurde das Package *Android LogCat* (v1.1.0) für die Entwicklung installiert. Dies ermöglicht das Auslesen von Debug.Log-Nachrichten im Unity Editor über das Android Endgerät.

### 3.1.1. Unity Projekt

Der Aufbau des Unity Projekts erscheint sehr simpel. Im Assets Ordner befindet sich die *AzureSpatialAnchors.SDK*, das Android Manifest und das *mainTemplate*. Diese Assets werden für die Entwicklung mit Spatial Anchors benötigt. Sämtliche Skripte der SDK befinden sich im Namespace *Microsoft.Azure.SpatialAnchors.Unity*. Im Ordner *Packages* befinden sich die wichtigen Skripte und Objekte, welche durch das Laden der Packages im *Package Manager* vorhanden sind. Bevor die Anwendung auf die bevorzugte Plattform gebaut wird, muss in den *Build Settings* jene Plattform ausgewählt werden.

### 3.1.2. Aufbau einer Szene

Die Szene einer *Azure Spatial Anchors* Anwendung benötigt zwei wichtige Objekte bzw. Prefabs. Das **Camera-Objekt** und das **AzureSpatialAnchors-Objekt**. Um eventuelle Positionsfehler zu vermeiden, ist es wichtig, beim Erzeugen leerer Game-Objects auf die Positionierung zu achten. Das leere GameObject sollte im Zentrum der Szene, also bei (0,0,0) liegen.

**Camera** Das Camera Prefab ist ein leeres GameObject, welches das Skript `XRCameraPicker` trägt. Dieses Klasse legt die verwendete Kamera fest. Dabei gibt es drei Kameras zu unterscheiden: Die HoloLens Kamera, die ARFoundation Kamera und eine Default Kamera. Diese drei Objekte werden der Klasse als Prefab übergeben. Diese Arbeit behandelt ausschließlich die Entwicklung mit Android, daher liegt der Fokus auf der ARFoundation Kamera. Als AR Foundation Kamera wird dem `XRCameraPicker` das Prefab `ARFoundationSessionStack` übergeben. Dabei handelt es sich um ein leeres GameObject, welches mehrere AR Skripte trägt. Es besitzt eine einfache Unity Camera als Child. Folgende Skripte liegen auf dem GameObject:

- `ARSession`
- `ARInputManager`
- `ARSessionOrigin`
- `ARRaycastManager`
- `ARReferencePointManager`
- `ARPlaneManager`
- `ARPointCloudManager`

Diese Skripte gehören zu AR Foundation und befinden sich im Package des Plugins. Sie steuern die AR Funktionalität auf dem mobilen Endgerät und sind Plattformunabhängig (siehe Kapitel 2.3). Die Klasse `ARSessionOrigin` benötigt als Übergabe-Objekt eine Kamera. Dafür wird die *Unity Camera*, welche als Child im `ARFoundationSessionStack` Prefab liegt, verwendet. Auf dieser liegen die Skripte `TrackedPoseDriver`, `ARCameraManager` und `ARCameraBackground`. Den Klassen `ARReferencePointManager`, `ARPlaneManager` und `ARPointCloudManager` können

ebenfalls über den Unity Inspector bestimmte Prefabs übergeben werden. Dies ist optional und hat keinen Einfluss auf die Funktionalität. Sie dienen zur Visualisierung, wie die Generierung der Oberflächen oder Punktewolken. Diese Prefabs können über die *Create* Funktionalität in der Hierarchie unter dem Punkt *XR* erzeugt werden.

**AzureSpatialAnchors** Das leere GameObject enthält die Klasse `SpatialAnchorManager`. Diese Klasse wird in diesem Kapitel im Abschnitt 3.2.6 genauer erläutert. Außerdem kann hier das Skript zur Kontrolle der Anwendung liegen. Dies ist abhängig vom Aufbau der Anwendung.

## 3.2. Klassen

Im Folgenden werden die Haupt-Klassen der Spatial Anchor SDK erläutert. Als Referenz hierzu dient die Dokumentation von Microsoft [Micb]. Außerdem wird die Klasse `SpatialAnchorManager` detailliert beschrieben.

### 3.2.1. Cloud Anchor

Ein Cloud Anchor ist ein Object vom Typ `CloudSpatialAnchor`. Diese Klasse definiert fünf Eigenschaften:

- **AppProperties** Ein Dictionary mit Eigenschaften, welches von der Anwendung definiert werden. Dieses Dictionary (`IDisctionary<string, string>`) besteht aus einem Schlüssel (Key) und einem Wert (Value). Jeder Schlüssel darf nur einmalig definiert werden.
- **Expiration** Mit dieser Eigenschaft wird dem Anchor eine Zeitspanne übergeben, nach welcher er automatisch gelöscht wird. Dies ist von Vorteil, falls beim Entwickeln die ID verloren geht. Der Eigenschaft wird ein Object vom Typ `DateTimeOffset` übergeben.
- **Identifier** Diese Eigenschaft speichert die einzigartige ID des Anchors. Der Identifier ist ein Object vom Typ `string`.
- **LocalAnchor** Die Eigenschaft ist ein IntPtr, der auf das lokale GameObject zeigt, welches mit dem Anchor verknüpft ist bzw. den Anchor visuell darstellt.

- **VersionTag** In dieser Eigenschaft wird die Versionskennzeichnung als String-Object gespeichert. Dies dient zur Parallelitätskontrolle.

Um einen Austausch zwischen dem lokalen und dem cloudbasierten Anchor zu gewährleisten, liefert Azure die Klasse `CloudNativeAnchor`.

### 3.2.2. `CloudNativeAnchor`

Diese Klasse stellt die Beziehung zwischen dem nativen bzw. lokalen Anchor und dem cloudbasierten Anchor her. Sie definiert zwei wichtige Methoden:

- **CloudToNative()**: Diese Methode speichert den Cloud-Anteil des Anchors und erstellt bzw. aktualisiert den nativen Anchor, damit diese die gleichen Informationen tragen.
- **NativeToCloud()**: Erstellt oder aktualisiert den Cloud Anchor und stellt sicher, dass der native Anchor dieselben Daten wie der Cloud Anchor reflektiert. Falls kein nativer Anchor auf dem `GameObject` existiert, wird einer erzeugt. Falls kein Cloud Anchor vorhanden ist, wird aus dem nativen Anchor ein Cloud Anchor generiert.

### 3.2.3. `Watcher`

Die Klasse `CloudSpatialAnchorWatcher` ist essentiell für das Auffinden von Anchors. Die Klasse besitzt eine Eigenschaft, einen `Identifier` vom Typ `Int32`. Dieser `Identifier` ist zur Identifizierung eines Watchers innerhalb einer Session. Die Klasse definiert auch die Funktion `Stop()`, welche jegliche Aktivität des Watchers beendet.

### 3.2.4. `AnchorLocateCriteria`

Diese Klasse speichert verschiedene Bedingungen und Kriterien zum Auffinden von Anchors. Sie hat fünf Eigenschaften.

- **BypassCache**: Diese Eigenschaft ist ein boolscher Wert. Sie entscheidet, ob Anchors im lokalen Cache/Speicher ebenfalls lokalisiert werden sollen oder nicht.

- **Identifiers** Bei dieser Eigenschaft handelt es sich um ein Array vom Typ `String[]`. Dieses Array speichert alle `CloudSpatialAnchor.Identifier`, also alle IDs von Anchors, welche lokalisiert werden sollen. Dabei ist die Lokalisierung pro Watcher auf maximal 35 Anchors begrenzt.
- **NearAnchor**: Diese Eigenschaft filtert alle Anchors, welche sich in der Nähe eines bestimmten Anchors befinden. Dabei wird ein Object vom Typ `NearAnchorCriteria` übergeben. Diese Klasse besitzt drei Eigenchaften. Die Eigenschaft `DistanceInMeters` legt die maximale Distanz fest, in welcher nach anderen Anchors gesucht werden soll. Sie hat den Typ `Single` und bei Default beträgt der Wert 5. Die Eigenschaft `MaxResultCount` bestimmt die maximale zu findende Anzahl von Anchors. Sie ist vom Typ `int` und beträgt bei Default den Wert 20. Die letzte Eigenschaft `SourceAnchor` speichert den `CloudSpatialAnchor`, welcher als Zentrum der Suche agiert.
- **RequestedCategories**: Bei dieser Eigenschaft handelt es sich um verschiedene Kategorien von gefragten Daten. Ihr wird ein Object vom Typ `AnchorDataCategory` übergeben. Dabei handelt es sich um einen `IEnumerator` mit drei Feldern. Das Feld `None` (0) gibt keine spezifischen Daten zurück. `Properties` (1) liefert die Eigenschaften des Anchors, einschließlich den `AppProperties`. Das letzte Feld `Spatial` (2) gibt räumliche Informationen über den Anchor zurück.
- **Strategy**: Diese Eigenschaft speichert eine Strategie zur Lokalisierung. Ihr wird ein Object vom Typ `LocateStrategy` übergeben. Auch hier handelt es sich um ein `IEnumerator` Object mit drei Feldern. Das erste Feld `AnyStrategy` (0) akzeptiert jede Methode. Das Feld `Relationship` (2) gibt an, dass primär Anchor mit einer Beziehung zu anderen Anchors lokalisiert werden sollen. Das letzte Feld `VisualInformation` (2) legt fest, dass die Lokalisierung eines Anchors hauptsächlich auf Basis ihrer visuellen Information stattfindet.

Innerhalb dieses Object werden die Eigenschaften, also die Kriterien mit einem *AND* Operator verknüpft. Als Beispiel: Das Object speichert ein Array von *X* `Identifier` und einen spezifischen Anchor als `NearAnchor`. Durch die Verknüpfung mit *AND* werden also alle Anchors gefunden, welche sich nahe dem `NearAnchor` befinden und den `Identifier` in dem Array entsprechen.

### 3.2.5. Session

Eine Session ist ein Object vom Typ `CloudSpatialAnchorSession`. Für jede Funktion bei der Verwendung von *Azure Spatial Anchors* wird eine Session benötigt. Diese muss erstellt, gestartet und am Ende wieder gestoppt werden. Es ist nicht notwendig die Session dauerhaft laufen zu lassen. Ein Object dieser Klasse besitzt fünf Eigenschaften (Properties):

- **Configuration:** Die Eigenschaft liefert ein Object vom Typ `SessionConfiguration`. Dieses Object enthält die Informationen für die Verbindung zu der Azure Ressource. Es besitzt die Eigenschaften `AccountDomain` (string), `AccountId` (string), `AccountKey` (string), `AccessToken` (string) und `AuthenticationToken` (string). Die Eigenschaften `AccessToken` und `AuthenticationToken` werden nur bei der Authentifizierung durch Azure Active Directory (AAD) benötigt. In dieser Arbeit wird die Authentifizierung mit Hilfe der `AccountId` und dem `AccountKey` durchgeführt. Diese Werte sind in der Spatial Anchors Ressource im Azure-Portal zu finden.
- **Diagnostics:** Diese Eigenschaft sammelt Daten für Fehlerbehebung und Verbesserung. Es wird ein Object vom Typ `CloudSpatialAnchorSessionDiagnostics` übergeben.
- **LogLevel:** Diese Eigenschaft definiert die Protokollierungsebene der aufgezeichneten Ereignisse einer Session. Es wird ein Object vom Typ `SessionLogLevel` übergeben. Dabei handelt es sich um ein `IEnumerator` mit sechs Feldern: `None` (0), `Error` (1), `Warning` (2), `Information` (3), `Debug` (4) und `All` (5). Diese Felder legen fest, welche Ereignisse protokolliert werden sollen.
- **Session:** Bei dieser Eigenschaft handelt es sich um einen `IntPtr` (`IntPtr`), welcher auf die aktuelle Session zeigt.
- **SessionId:** Diese Eigenschaft liefert die einzigartige ID der Session als `String`.

### Anchor Methoden

Innerhalb der Klasse `CloudSpatialAnchorSession` werden die Funktionen zur Erstellung, Lokalisierung und Bearbeitung von Anchors definiert.

- **CreateAnchorAsync(CloudSpatialAnchor)**: Diese Funktion erstellt einen dauerhaften Cloud Anchor auf Basis eines lokalen Anchors vom Typ CloudSpatialAnchor. Die Funktion ist vom Typ async Task.
- **CreateWatcher(AnchorLocateCriteria)**: Diese Funktion gibt ein neues Object vom Typ CloudSpatialAnchorWatcher zurück. Der Funktion wird ein AnchorLocateCriteria Object übergeben.
- **DeleteAnchorAsync(CloudSpatialAnchor)**: Diese Funktion löscht einen spezifischen Anchor und ist vom Typ async Task. Ihr wird ein Object vom Typ CloudSpatialAnchor übergeben.
- **GetActiveWatchers()**: Diese Funktion gibt eine IReadOnlyList mit allen aktiven CloudSpatialAnchorWatcher zurück.
- **GetAnchorPropertiesAsync(String)**: Eine async Task Funktion, welche einen CloudSpatialAnchor zurück gibt. Dafür ist keine vorherige Lokalisierung nötig. Der Funktion wird der CloudSpatialAnchor.Identifier übergeben.
- **RefreshAnchorPropertiesAsync(CloudSpatialAnchor)**: Für den Fall, dass die AppProperties nachträglich geändert wurden, ist der Aufruf dieser Funktion nötig. Ihr wird der CloudSpatialAnchor, welcher aktualisiert werden soll, übergeben. Die Funktion ist vom Typ async Task.

Zum Kontrollieren einer Session dienen die Funktionen Start(), Stop(), Reset() und Dispose(). Mit Hilfe der Reset()-Funktion werden alle gesammelten Umgebungsdaten gelöscht. Dies ist vor allem notwendig, wenn das Tracking innerhalb der Anwendung verloren geht.

### **GetSessionStatusAsync()**

Diese async Task Funktion liefert ein Object vom Typ SessionStatus, welches den aktuellen Status der Session beschreibt. Dieses Object ist besonders wichtig für die Erzeugung von genug visuellen Daten. Das Object hat fünf Eigenschaften: ReadyForCreateprogress, RecommendedForCreateProgress, SessionCreateHash, SessionLocateHash und UserFeedback. Die Eigenschaft RecommendedForCreateProgress speichert einen Single Wert zwischen 0 und 1. Ist der Wert < 1, sind nicht genügend Daten zum Erstellen eines Anchors vorhanden.

Sobald der Wert 1 oder größer beträgt, ist das Speichern des Anchors möglich. Die Eigenschaft ReadyForCreateProgress fungiert auf ähnliche Art. Die nachfolgende Funktion gibt zurück, ob ein lokaler Anchor bereit zum Erstellen und Speichern in der Cloud ist.

```
public bool IsReadyForCreate
{
    get
    {
        return ((sessionStatus != null)
            && (sessionStatus.RecommendedForCreateProgress >= 1));
    }
}
```

## Events

Die Klasse beinhaltet aktuell sieben Events: AnchorLocated, Error, LocateAnchorsCompleted, OnLogDebug, SessionUpdated, TokenRequired und UpdatedSensorFingerprintRequired. Ein Event ermöglicht einer Klasse oder einem Objekt jeweils andere Klassen und Objekte über ein bestimmtes Ereignis zu informieren. Der Sender eines Events wird als *Publisher* und der Empfänger als *Subscriber* bezeichnet. Der Publisher legt fest, wann ein bestimmtes Event ausgelöst wird und der Subscriber entscheidet, was daraufhin geschieht. Damit der Subscriber ein Event abonnieren kann, muss er eine Methode definieren, welche die Aktionen bei ausgelöstem Event steuert. Diese Event Handler Methode muss von gleicher Signatur wie das Event selbst sein. Mit Hilfe des Additionszuweisungsoperators ( $+ =$ ) abonniert der Subscriber das Event und fügt diesem die Event-Methode hinzu. Umgekehrt kann durch den  $- =$ -Operator ein Event deabonniert werden.

```
// Subscribe to AnchorLocated Event
CloudSpatialAnchorSession.AnchorLocated += OnAnchorLocated;

// Unsubscribe to AnchorLocated Event
CloudSpatialAnchorSession.AnchorLocated -= OnAnchorLocated;
```

Um eine Session zu erstellen, muss ein Object vom Typ CloudSpatialAnchorSession erzeugt werden. Es ist wichtig, dass nicht mehrere

Session-Objekte vorhanden sind. Daher muss vor der Erzeugung eine Abfrage stattfinden. Dazu ist es hilfreich, ein Manager-Script zu definieren, welches eine Methode zur Erstellung einer Session beinhaltet. Microsoft liefert in der API den `SpatialAnchorManager`.

### 3.2.6. Spatial Anchor Manager

Der `SpatialAnchorManager` ist eine Klasse, welche sämtliche Funktionen der verschiedenen Klassen sammelt und einen sicheren Aufruf gewährleistet. Dabei ist wichtig, dass sich der Manager im gleichen namespace befindet. Viele der Funktionen werden hier in einem asynchronen Task "geschachtelt". Dabei ist zu beachten, dass die Benennung einiger Methoden identisch zu den Methoden der oben genannten Klassen ist. Beispielsweise heißt die Funktion zur sicheren Speicherung eines Anchors `CreateAnchorAsync(...)`. Diese Methode heißt also gleich, wie die zur Speicherung eines Anchors in der Klasse `CloudSpatialAnchorSession`. Dieses Vorgehen dient der Übersicht und Struktur des Codes.

*Asynchrone Programmierung* Es wird das Modell Task-based Asynchronous Pattern (TAP) verwendet. Asynchroner Code verwendet Tasks (`Task<T>`, `Task`). Tasks sind Konstrukte, welche im Hintergrund arbeiten. Asynchrone Methoden ermöglichen die Verwendung des `await` Schlüsselworts. Dieses Schlüsselwort unterbricht die Methode, welche das `await` Schlüsselwort aufgerufen hat und gibt die Kontrolle zurück, sobald der Task abgeschlossen ist. Dieses Modell wird besonders bei dem Abfragen von Daten aus einem Netzwerk oder einer Datenbank verwendet. Es gilt CPU-gebundene und I/O-gebundene Arbeit zu unterscheiden. Bei I/O-gebundener Arbeit wird auf Daten aus einem Netzwerk oder eine Datenbank gewartet. Bei CPU-gebundener Arbeit wird eine sehr große Berechnung durchgeführt.

#### `CreateSessionAsync()`

Die Funktion erzeugt nach mehreren Überprüfungen eine neue Session. Im nachfolgenden Code wird die Funktion als `public async Task` definiert. Der Befehl `#pragma warning restore CS1998` verhindert die Ausgabe eines Warnhinweises in Unity. Fehlt dieser Befehl, ist in der Unity Console folgender Warnhinweis zu finden:

*"...: warning CS1998: This async method lacks 'await' operators and will run synchronously. Consider using the 'await' operator to await non-blocking API calls, or 'await Task.Run(...)' to do CPU-bound work on a background thread"*

Dies kann auch passieren obwohl ein await-Operator im Code vorhanden ist. Bei der Verwendung von plattformabhängiger Kompilierung kann es vorkommen, dass der Unity Editor bestimmte Zeilen im Code nicht sieht.

```
#pragma warning disable CS1998
public async Task CreateSessionAsync()
#pragma warning restore CS1998
{
    if (session != null)
    {
        Debug.LogWarning($"nameof(CreateSessionAsync)}
                           called but session is already created.");
        return;
    }
    await EnsureValidConfiguration(disable: false, exception: true);
    ...
}
```

*Plattformabhängige Kompilierung* Die plattformabhängige Kompilierung ermöglicht eine Partitionierung von Skripten. Dadurch können bestimmte Codeabschnitte ausschließlich für die jeweilige Plattform kompiliert und ausgeführt werden. Diese Partitionierungen werden durch ein Doppelkreuz angeführt. Mit Hilfe eines if-Statements werden die Plattformen festgelegt. Der Block beginnt mit #if, kann mit #elif (else if) erweitert werden und endet mit #endif. Die am häufigsten verwendeten Plattformen sind beispielsweise UNITY\_EDITOR, UNITY\_ANDROID, UNITY\_IOS oder UNITY\_WSA. Die plattformabhängige Komplilierung ermöglicht auch Versionskontrolle, indem ein Skript für verschiedene Unity-Versionen partitioniert wird. Das Schlüsselwort hierfür ist beispielsweise UNITY\_5\_0\_1. Der Codeabschnitt wird nur bei Unity-Version 5.0.1 ausgeführt. In dem folgendem Code ist ein Codeabschnitt zu sehen, welche nur bei Android und iOS Geräten ausgeführt wird. Der Unity Editor überspringt diesen Abschnitt.

Der nachfolgende Code zeigt die Fortsetzung der CreateSessionAsync() Methode. Der Teil kontrolliert die erhaltenen Frames der AR Camera. Danach wird ein neues CloudSpatialAnchorSession Objekt erzeugt und die Authentifizierung, also die Verbindung zum Azure Konto, konfiguriert. Abschließend werden die Events des Publishers CloudSpatialAnchorSession abonniert.

```
...  
#if UNITY_ANDROID || UNITY_IOS  
    // Ask for ar frames to process  
    arCameraManager.frameReceived += ArCameraManager_frameReceived;  
#endif  
  
    // Create the session instance  
    session = new CloudSpatialAnchorSession();  
  
    // Configure logging  
    session.LogLevel = logLevel;  
  
    // Configure authentication  
    if (authenticationMode == AuthenticationMode.ApiKey)  
    {  
        // API Key mode just applies credentials directly  
        session.Configuration.AccountId = spatialAnchorsAccountId.Trim();  
        session.Configuration.AccountKey = spatialAnchorsAccountKey.Trim();  
    }  
    else  
    {  
        // AAD mode requires an auth token workflow  
        session.TokenRequired += Session_TokenRequired;  
    }  
  
    // Subscribe to session events  
    session.OnLogDebug += OnLogDebug;  
    session.SessionUpdated += OnSessionUpdated;  
    session.AnchorLocated += OnAnchorLocated;
```

```

    session.LocateAnchorsCompleted += OnLocateAnchorsCompleted;
    session.Error += OnError;

    ...
}


```

### DestroySession()

Die Funktion zerstört eine Session und setzt alle Daten zurück. Zu Beginn wird überprüft, ob überhaupt eine Session existiert. Falls nicht, wird die Funktion mit `return` beendet. Handelt es sich bei der verwendeten Plattform um ein Android oder ein iOS System, werden darauffolgend die ARFoundation Funktionalitäten beendet. Sämtliche Daten von Reference Points im Cache werden gelöscht und die AR Camera beendet das Senden von Frames. Die Session stoppt, für den Fall, dass sie noch aktiv ist. Nachdem die Session von allen Events abgemeldet wurde, muss sie mit `CloudSpatialAnchorSession.Dispose()` gelöscht werden. Die Variablen `session` und `sessionStatus` nehmen den Wert `null` an.

```

public void DestroySession()
{
    if (session == null)
    {
        Debug.LogWarning($"'{nameof(DestroySession)}' 
            called but no session exists.");
        return;
    }

#if UNITY_ANDROID || UNITY_IOS
    pointerToReferencePoints.Clear();
    arCameraManager.frameReceived -= ArCameraManager_frameReceived;
#endif

    if (isSessionStarted) { StopSession(); }

    session.OnLogDebug -= OnLogDebug;
    session.SessionUpdated -= OnSessionUpdated;
}


```

```
    session.AnchorLocated -= OnAnchorLocated;
    session.LocateAnchorsCompleted -= OnLocateAnchorsCompleted;
    session.Error -= OnError;

    session.Dispose();
    Session = null;
    sessionStatus = null;
}
```

### ResetSessionAsync()

Diese Funktion setzt die aktuelle Session zurück und wartet, bis alle aktiven Anfragen gestoppt sind. Die `CloudSpatialAnchorSession.Reset()` Funktion wird hier verschachtelt. Die Variable `bool isSessionStarted` speichert, ob die aktuelle Session läuft (`true`) oder nicht (`false`). Zu Beginn der Funktion wird der aktuelle Zustand der Session zwischengespeichert. Falls die Session läuft, wird sie gestoppt. Danach wird die `CloudSpatialAnchorSession.Reset()` Funktion aufgerufen, sofern eine Session existiert. Die Variable `session` repräsentiert dabei ein Object vom Typ `CloudSpatialAnchorSession`. Nachdem die Session zurückgesetzt wurde, wird ein Task geschickt, welcher wartet, bis alle Lokalisierungs-Operationen beendet wurden. Die Variable `bool IsLocating` gibt `true` zurück, falls eine Session existiert und ein oder mehrere Watcher aktiv sind. Für den Fall, dass die Session zu Beginn der Funktion lief, wird sie in der letzten Zeile erneut gestartet.

```
public async Task ResetSessionAsync()
{
    bool wasStarted = isSessionStarted;
    if (isSessionStarted) { StopSession(); }
    if (session != null) { session.Reset(); }

    await Task.Run(async () =>
    {
        while (IsLocating)
        {
            await Task.Delay(20);
        }
    });
}
```

```
});  
  
if (wasStarted) { await StartSessionAsync(); }  
}
```

### StartSessionAsync()

Diese Funktion führt die `CloudSpatialAnchorSession.Start()` Funktion aus. Davor wird überprüft, ob die Session bereits gestartet wurde und ob eine Session vorhanden ist. Falls keine Session vorhanden ist, wird eine neue erzeugt. Nach dem Starten der Session wird der Status in der Variablen `sessionStatus` gespeichert. Dazu dient `CloudSpatialAnchorSession.GetSessionStatusAsync()`.

```
public async Task StartSessionAsync()  
{  
    if (isSessionStarted)  
    {  
        Debug.LogWarning($"{nameof(StartSessionAsync)}  
            called but session is already started.");  
        return;  
    }  
  
    if (session == null)  
    {  
        Debug.Log($"{nameof(StartSessionAsync)}  
            called with but no session. Creating one.");  
        await CreateSessionAsync();  
    }  
  
    session.Start();  
    isSessionStarted = true;  
    sessionStatus = await session.GetSessionStatusAsync();  
    OnSessionStarted();  
}
```

### StopSession()

Diese Funktion ist void und nicht async Task. Hier wird die Funktion CloudSpatialAnchorSession.Stop() ausgeführt. Wie bei StartSessionAsync() wird überprüft, ob die Session läuft und vorhanden ist. Nach dem Stoppen der Session werden die jeweiligen Variablen angepasst.

```
public void StopSession()
{
    if (!isSessionStarted)
    {
        Debug.LogWarning($"'{nameof(StopSession)}'
                           called but no session has been started.");
        return;
    }

    if (session == null)
    {
        Debug.LogWarning($"'{nameof(StopSession)}'
                           called but no session has been created.");
        return;
    }

    session.Stop();
    sessionStatus = null;
    isSessionStarted = false;
    OnSessionStopped();
}
```

### CreateAnchorAsync(CloudSpatialAnchor, CancellationToken)

Diese async Task Funktion erstellt nach mehreren Überprüfungen einen Cloud Anchor auf Basis des nativen Anchors. Ihr wird der lokal platzierte CloudSpatialAnchor übergeben. Außerdem wird ein Object vom Typ CancellationToken übergeben. Dieses Objekt verbreitet die Nachricht, dass eine Operation abgebrochen werden soll. Für mehr Informationen über das Objekt CancellationToken ist die Doku-

mentation von Microsoft hilfreich<sup>1</sup>. Im Umfang dieser Arbeit wird mit einem leeren CancellationToken gearbeitet.

```
public async Task CreateAnchorAsync(CloudSpatialAnchor anchor,
    CancellationToken cancellationToken)
{
    if (anchor == null) throw new ArgumentNullException(nameof(anchor));
    EnsureSessionStarted();

    // Wait for enough data
    while (!IsReadyForCreate)
    {
        cancellationToken.ThrowIfCancellationRequested();
        await Task.Delay(50);
    }

    // Actually create
    await session.CreateAnchorAsync(anchor);
}
```

### **CreateAnchorAsync(CloudSpatialAnchor)**

Diese Funktion ist vom Typ Task und besteht nur aus einer Zeile. Sie ruft CreateAnchorAsync(CloudSpatialAnchor, CancellationToken) auf und setzt CancellationToken.None als CancellationToken Objekt ein. Somit wird ein leerer CancellationToken Objekt an die Funktion übergeben.

### **DeleteAnchorAsync(CloudSpatialAnchor)**

Diese Funktion ist async Task und löscht einen übergebenen Anchor. Es wird überprüft, ob ein Anchor übergeben und die Session gestartet wurde.

```
public async Task DeleteAnchorAsync(CloudSpatialAnchor anchor)
{
    if (anchor == null) throw new ArgumentNullException(nameof(anchor));
```

---

<sup>1</sup><https://docs.microsoft.com/en-us/dotnet/api/system.threading.cancellationtoken?view=netframework-4.8>

```
    EnsureSessionStarted();  
  
    await session.DeleteAnchorAsync(anchor);  
}
```

### ProcessLatestFrame()

Diese Funktion bildet die Brücke zwischen *ARFoundation* und *Azure Spatial Anchors*. Sie speichert den letzten von ARFoundation aufgezeichneten Frame und über gibt ihn an die aktuelle Session. Dies sind die visuellen Bilddaten. Zuerst werden die Daten der Kamera in der Variable `cameraParams` vom Typ `XRCameraParams` gespeichert. Dabei handelt es sich um ein Struct aus dem `UnityEngine` Namespace. Danach wird ein Object vom Typ `XRCameraFrame` erzeugt. Mit der Funktion `TryGetLatestFrame(XRCameraParams, out XRCameraFrame)` wird versucht, den letzten Frame zu speichern. Da die Funktion vom Typ `bool` ist, gibt sie beim erfolgreichen Speichern `true` zurück und speichert den Frame in der `out` Variable `xRCameraFrame`. Innerhalb des `if`-Statements wird nun der letzte Frame an die Azure Spatial Anchor Session übergeben. Der aktuelle Zeitstempel wird in `latestFrameTimeStamp` gespeichert und mit dem Zeitstempel `lastFrameProcessedTimeStamp` verglichen. Ist der aktuelle Zeitstempel größer als der Zeitstempel des zuvor gespeicherten Frames, nimmt die Variable `newFrameToProcess` den Wert `true` an und die Funktion springt in das dritte `if`-Statement. In den letzten Zeilen der Funktion wird mittels `ProcessFrame(IntPtr)` ein `IntPtr` (`IntPtr`) auf den Frame gesetzt. Dieser Frame ist nun mit der aktuellen Session verbunden. Zusätzlich wird der Zeitstempel entsprechend aktualisiert. Die Funktion `ProcessLatestFrame()` wird von *ARFoundation* aufgerufen, sobald es einen neuen Frame zu verarbeiten gibt.

```
private void ProcessLatestFrame()
{
    if (!isSessionStarted)
    {
        return;
    }

    var cameraParams = new XRCameraParams
    {
        zNear = mainCamera.nearClipPlane,
        zFar = mainCamera.farClipPlane,
        screenWidth = Screen.width,
        screenHeight = Screen.height,
        screenOrientation = Screen.orientation
    };

    XRCameraFrame xRCameraFrame;
    if (arCameraManager.subsystem.TryGetLatestFrame(cameraParams,
        out xRCameraFrame))
    {
        long latestFrameTimeStamp = xRCameraFrame.timestampNs;

        bool newFrameToProcess =
            latestFrameTimeStamp > lastFrameProcessedTimeStamp;

        if (newFrameToProcess)
        {
            session.ProcessFrame(xRCameraFrame.nativePtr.
                GetPlatformPointer());
            lastFrameProcessedTimeStamp = latestFrameTimeStamp;
        }
    }
}
```

### 3.3. Entwicklung Prototyp und Research-Anwendung

Um die Technologie zu verstehen und zu untersuchen, wurde ein Prototyp basierend auf der Azure Demo entwickelt. Die Grundlage dazu bieten drei Skripte. Im Umfang dieser Arbeit entstanden zwei Anwendungen: der Prototyp und die Research-Anwendung. Das spezifische Skript für den Prototypen heißt MySpatialApp und erbt die Klasse MyAppBase. Diese Klasse erbt wiederum die Klasse für die Kontrolle der Eingabeaktionen MyInputInteractionBase. Durch diesen Aufbau der Skripte ist es möglich, mehrere Anwendungen mit unterschiedlichen Funktionen zu entwickeln. Die grundlegenden Funktionen sind in MyAppBase und MyInputInteractionBase definiert und werden an die jeweilige Anwendung vererbt. So basieren sowohl der Prototyp wie auch die Research-Anwendung auf diesen beiden Klassen. Die Research-Anwendung, welche für den Versuchsaufbau in Kapitel 4 entwickelt wurde, basiert auf dem ersten Prototypen MySpatialApp und die Basis ResearchAppBase auf MyAppBase. Die hier beschriebenen Klassen werden nur oberflächlich erläutert. Für ein besseres Verständnis ist es ratsam die jeweilige Skripte dazu zu nehmen. Im Anhang D befinden sich Flowcharts mit Screenshots der jeweiligen Anwendungen und Funktionalitäten. [Mice]

Mehrere Abbildungen im Anhang sind sehr klein und können auf Papier schlecht lesbar sein. Sämtliche Abbildungen, Tabellen sowie das Unity-Projekt der hier beschriebenen Anwendungen befinden sich entweder auf einem Datenträger verbunden mit der gedruckten Version oder im GitHub Repository unter <https://github.com/nincara/MasterThesis>.

Wie bereits erwähnt, basieren sowohl der Prototyp als auch die Research-Anwendung auf der Azure Demo<sup>2</sup>. Übernommene Methoden und Codeabschnitte werden explizit **nicht** kenntlich gemacht, da übernommenes und eigenes zu sehr ineinander fließt. In dieser Arbeit geht es nicht um die Erstellung und Verbreitung einer eigenen Anwendung, sondern um die Analyse und die Erforschung. In diesem Rahmen wird auf die genaue Kennzeichnung verzichtet.

---

<sup>2</sup> <https://github.com/Azure/azure-spatial-anchors-samples>

### 3.3.1. Klassendiagramm

Im Anhang C befindet sich das Klassendiagramm für beide Anwendungen. Im Umfang der Entwicklung mit Unity wurden bestimmte Symbole hinzugefügt und angepasst. Die Standard-UML-Symbole bleiben gleich.

- # protected Variable
- - private Variable
- + public Variable
- \* internal Variable

Variablen in fettgedruckter Schrift stellen eine serialisierte Variable mit dem Zusatz `SerializeField` dar. Unterstrichene Variablen symbolisieren den Zusatz `static`. Weitere Zusätze wie beispielsweise `event` oder `virtual` werden in geschweiften Klammern angezeigt. Viele der `private` Variablen werden in `public` Variablen mit `get` und `set` Methoden verpackt. Diese wurden im Klassendiagramm ausgelassen. Methoden, welche im Umfang dieser Arbeit keine Verwendung finden, wurden ebenfalls weggelassen.

Die Klassen, welche von Azure zur Verfügung gestellt werden, sind in dem Diagramm mit Grün dargestellt. Dazu gehören `SpatialAnchorManager` und `CloudNativeAnchor`. Die Klassen in Rottönen beinhalten die anwendungsbasierten Funktionalitäten. Basis ist die Klasse `MyInputInteraction`. Diese wird an `ResearchAppBase` (Magenta) und `MyAppBase` (Rot) vererbt. Beide definieren ein Objekt `anchoredObjectPrefab` (Blau) vom Typ `GameObject`. Das Objekt trägt die Klasse `AnchorData`. Die Klasse `MySpatialApp` erbt `MyAppBase` und `ResearchSpatialApp` erbt `ResearchAppBase`. Für das Speichern der Anchor-Keys dient die Klasse `MyAnchorExchanger` (Grau). `SaveDataToJson` (Grau) speichert alle Daten in einem Json-File und die Klasse `UIHandler` (Grau) kontrolliert die UI-Elemente in `ResearchSpatialApp`. Die detaillierte Erklärung der verschiedenen Klassen erfolgt in den nachfolgenden Abschnitten. Das Klassendiagramm kann dabei als Unterstützung für das tiefere Verständnis dienen.

### 3.3.2. Sequenzdiagramme

Im Anhang C befinden sich zwei Sequenzdiagramme zum Ablauf der zwei wichtigsten Funktionen der Research-Anwendung. Diese sind die Generierung und die Lokalisierung von Anchors. Sie können auf den Prototypen übertragen werden. Die Sequenzdiagramme sind einfach gehalten und stellen die Abläufe grob dar.

Die Sequenzdiagramme visualisieren den Ablauf der Anwendungen. Sie zeigen an, welche Funktionen durch Interaktion des Users ausgelöst werden und welches Feedback dieser daraufhin bekommt. Dabei steht am Anfang des Diagramms der User, welcher mit der Anwendung interagiert. Die Anwendung stellt die App auf dem Smartphone dar. Sie interagiert mit den Klassen `ResearchSpatialApp` und `ResearchAppBase`. Zu Beginn beider Diagramme werden die Anchor-Keys aus der Web-App geladen. Die `ResearchSpatialApp` interagiert dabei mit dem `anchorExchanger` vom Typ `MyAnchorExchanger`. Diese Klasse wiederum baut eine Verbindung zur Web-App `SpatialAnchorsWeb-AppCB` auf, in welcher die Anchor-Keys gespeichert sind. Des Weiteren agieren die ersten beiden Klassen regelmäßig mit dem `CloudManager`, ein Objekt vom Typ `SpatialAnchorManager`. Der `CloudManager` zentriert die wichtigen Funktionen zur Kontrolle einer Session oder für die Erzeugung eines Cloud Anchors. Dafür stellt er eine Verbindung zur Azure Spatial Anchors Ressource `SpatialResourceCB` her.

### 3.3.3. MyInputInteractionBase

Diese Klasse fängt die Interaktionen des Users ab. Sie wurde von der Azure Demo übernommen und erbt `MonoBehaviour`. Im Klassendiagramm im Anhang C ist diese Dunkelrot gefärbt. Neben der Interaktion mit dem Smartphone, sind auch Funktionen für HoloLens definiert. Der Fokus dieser Arbeit liegt jedoch auf den Funktionen für die Interaktion mit dem Smartphone. Folgende Funktionen sind dabei ausschlaggebend:

- `TriggerInteractions()`: Diese Funktion wird in der `Update()` aufgerufen, fängt die Interaktionen ab und ruft die jeweilige Funktion für die Interaktion auf. Im Falle einer Touch-Interaktion wird `OnTouchInteraction(Touch touch)` aufgerufen.
- `OnTouchInteraction(Touch touch)`: Die Funktion überprüft, ob die Touch-Interaktion beendet wurde. Dies wird ausgelöst, sobald sich der

Finger vom Bildschirm hebt. Ist dies der Fall, wird die Funktion `OnTouchInteractionEnded(Touch touch)` ausgeführt.

- `OnTouchInteractionEnded(Touch touch)`: Bei dieser Funktion wird ein Objekt vom Typ `ARRaycastHit` verwendet. Dies ist eine `ARFoundation` Klasse zur Kontrolle von Raycasts. Durch plattformabhängige Programmierung werden hier die Befehle für Android / iOS und HoloLens getrennt. Bei beiden wird geschaut, ob der Raycast ein Objekt trifft. In diesem Falle ist das getroffene Objekt eine AR Oberfläche. An dem Punkt, an welchem sich Raycast und Oberfläche schneiden, möchte der User eine Interaktion, wie das Setzen eines 3D-Objekts, ausführen. Dafür ist die Funktion `OnSelectObjectInteraction(Vector3 hitPoint, object target)` definiert.
- `OnSelectObjectInteraction(Vector3 hitPoint, object target)`: Diese Funktion ist `virtual` und muss in `MyAppBase` überschrieben werden. Sie definiert, was bei einer Touch-Interaktion an berührter Stelle geschehen muss.

### 3.3.4. `MyAppBase` und `ResearchAppBase`

Die Anwendungsbasen (sowohl `MyAppBase` als auch `ResearchAppBase`) erben die Klasse `MyInputInteractionBase`. Zu Beginn definieren beide Klassen zwei Inspektor-Variablen. Diese Variablen erfordern eine Eingabe im Unity-Inspector und sind durch ein `SerializeField` definiert. Dabei handelt es sich um eine Klasse im namespace `UnityEngine`. Diese Klasse ermöglicht es, private Variablen zu serialisieren. Dies ist gewöhnlich nur mit `public` Variablen möglich. Für mehr Informationen zur Serialisierung von Skripten ist die Dokumentation zu Unity<sup>3</sup> hilfreich. Kurz erläutert bedeutet die Serialisierung von Variablen, dass diesen im Unity Editor ein Objekt übergeben werden kann. Während dem Schreiben des Skripts ist also noch nicht klar, um welches Objekt es sich dabei handelt. Die serialisierte Variable fungiert als eine Art Platzhalter. Bei den Klassen `MyAppBase` und `ResearchAppBase` sind diese beiden Variablen das 3D-Objekt bzw. das Prefab, welches den Anchor darstellt und der `SpatialAnchorManager`. Die Variable für das Prefab ist vom Typ `GameObject`, was bedeutet, ihr kann nur ein Objekt vom selben Typ im Unity Editor übergeben werden. Das `SpatialAnchorManager` Objekt ermöglicht den Zugriff auf die Funktionalitäten dieser Klasse. Beide Klassen sind hier in verschiedene Regionen aufgeteilt.

---

<sup>3</sup> <https://docs.unity3d.com/Manual/script-Serialization.html>

## Start()

Die Start() Methode überprüft mehrere Gegebenheiten, welche für das erfolgreiche Ausführen des Skripts ausschlaggebend sind. Durch den Befehl base.Start () ; wird die Start() Methode der vererbten Klasse ausgeführt.

## Events

Das Abonnieren der Events (siehe Abschnitt 3.2.5) findet ebenfalls in der Start() Methode statt. Die Klasse definiert dabei fünf Event Handler Methoden: CloudManager\_SessionUpdated, CloudManager\_AnchorLocated, CloudManager\_LocateAnchorsCompleted, CloudManager\_LogDebug und CloudManager\_Error. Am Beispiel AnchorLocated wird die Verknüpfung dieser Methoden mit den Events der Klasse CloudSpatialAnchorSession dargestellt: Der Publisher des Events AnchorLocated ist die Klasse CloudSpatialAnchorSession. Dieses Event hat einen Subscriber, die Klasse SpatialAnchorManager. Dies findet in der Methode CreateSessionAsync() statt. Mithilfe des Additionszuweisungsoperator (+ =) wird eine Methode oder ein Event Handler dem Event hinzugefügt. Im Fall des AnchorLocated Events in der Klasse SpatialAnchorManager ist dies die void Methode OnAnchorLocated. Die angefügte Methode löst dabei das SpatialAnchorManager.AnchorLocated Event aus. Diesem Event wird in den beiden Basis-Klassen in der Start() die Methode CloudManager\_AnchorLocated angefügt. Sie führt die Methode OnCloudAnchorLocated aus. In dieser Funktion ist das Verhalten bei Lokalisierung eines Anchors für die jeweilige Anwendung definiert. Da sie in der Basis-Klasse als virtual definiert ist, kann sie in der erbenden, anwendungsbasierten Klasse überschrieben werden.

## Region Anchor Locate Criteria

Diese Region definiert Methoden für das Setzen von Kriterien beim Lokalisieren. Dies bezieht sich auf die in Abschnitt 3.2.4 beschriebene Klasse. Zusätzlich wird die Methode CreateWatcher() definiert, welche einen CloudSpatialAnchorWatcher (siehe Abschnitt 3.2.3) zurück gibt.

## Region Placing Anchor

In dieser Region wird die abstrakte Methode `IsPlacingObject()` vom Typ `bool` definiert. Sie wird in der erbenden Klasse überschrieben und gibt `True` zurück, falls der User gerade ein Anchor platziert. Die Methode `MoveAnchoredObject` verschiebt den vom User platzierten Anchor (3D-Referenz) bei erneuter Touch-Eingabe.

## Region Cloud Anchor Actions

Bei den Methoden in dieser Region handelt es sich um Funktionen, welche durch Events ausgelöst werden. Die Methoden sind `virtual` und können bzw. müssen in der erbenden Klasse erweitert werden, wie beispielsweise die Methode `OnCloudAnchorLocated`.

## Region Input Interaction Overrides

In dieser Region werden bestimmte Methoden aus der geerbten Klasse `MyInputInteractionBase` aus Abschnitt 3.3.3 überschrieben. Beispielsweise wird in der Funktion `OnSelectObjectInteraction` definiert, was bei einer Touch-Interaktion geschehen soll. Gibt die bereits erwähnte Methode `IsPlacingObject()` `true` zurück, so wird ein neues Objekt erzeugt oder ein bestehendes Objekt verschoben.

## Region Progress Data Collection

Die Methoden dieser Region befinden sich ausschließlich in der Klasse `ResearchAppBase`. Sie dienen zum Überprüfen, ob genug visuelle Daten gesammelt wurden. Dies geschieht in der Klasse manuell, also muss der User entscheiden, wann genug Daten gesammelt wurden. Durch betätigen eines Buttons wird die Funktion `EnoughCollected()` aufgerufen und das Sammeln von Daten wird abgebrochen.

## Region Save and Spawn Anchors

In dieser Region befinden sich die wichtigsten Anchor-Methoden: `SaveCurrentObjectAnchorToCloudAsync`, `SpawnNewAnchoredObject` und `SpawnOrMoveCurrentAnchoredObject`. Die erste Methode speichert den lokalen Anchor in der Cloud. Sie sammelt Umgebungsdaten mit Hilfe der Methoden aus der zuvor beschrieben Region. Eine `while`-Schleife wird solange ausgeführt, bis

der User durch das Betätigen eines Buttons das Sammeln beendet. Daraufhin werden dem Anchor mehrere AppProperties übergeben. Dazu zählen beispielweise die eingegebenen Daten durch den User wie Name, ID und Info. Aber auch das Erstellungsdatum, die gesammelten Daten bei der Generierung oder die maximalen Feature Points. Die Zeile `await CloudManager.CreateAnchorAsync(cloudAnchor);` speichert daraufhin den Anchor. Die letzte Methode `SpawnOrMoveCurrentAnchoredObject` wird bei der überschrieben Input-Methode `OnSelectObjectInteraction` ausgelöst. Falls bereits ein Anchor platziert wurde, wird die Methode `MoveAnchoredObject(...)` ausgeführt. Ist noch kein Anchor vorhanden, wird die Methode `SpawnNewAnchoredObject(...)` aufgerufen. Sie instanziert ein neues 3D-Objekt, welches den Anchor lokal repräsentiert. Dieses 3D-Objekt wird durch das serialisierte Feld zu Beginn der Klasse in Form eines Prefab `GameObject` definiert (Beginn des Abschnitts). Der Variable muss im Unity Editor ein Prefab zugewiesen werden. Die Methode wird in zwei Fällen aufgerufen: Bei der Erzeugung eines neuen Anchors oder bei der Darstellung eines lokalisierten Anchors. Ist zweiteres der Fall, werden die AppProperties Daten des Anchors an die auf dem Prefab liegende Klasse `AnchorData` übergeben. Die Daten des gefundenen Anchors müssen jedoch vor der Funktion `CloudToNative` gespeichert werden, da diese sonst verloren gehen.

### 3.3.5. MySpatialApp und ResearchSpatialApp

Beide Klassen erben die jeweiligen Basis-Klassen (siehe Klassendiagramm im Anhang C). Die Klassen unterscheiden sich in der Handhabung der UI Elemente. Für die Entwicklung der Research-App wurde die `UIHandler` Klasse geschrieben. Die Klasse speichert alle UI Elemente mit Hilfe der `GameObject.Find()` Methode. Außerdem kontrolliert es, welche UI Elemente bei Start der Anwendung sichtbar und unsichtbar sind. Mit der `GameObject.SetActive()` Methode können die Elemente ein und ausgeblendet werden. Beide Klassen basieren auf verschiedenen Zuständen. Ein `IEnumerator` mit der Bezeichnung `AppState` legt verschiedene App-Zustände fest wie `LoadingKeys` oder `PlacingAnchor`. Abhängig davon, in welcher Phase die Anwendung sich befindet, ändert sich der aktuelle Zustand. Diese Vorgehensweise ist von Vorteil, wenn beispielsweise ein bestimmter Code-Abschnitt in der `Update()`-Funktion nur in einem spezifischen Zustand ausgeführt werden soll.

### **Start() und Update()**

Die Start() Methode führt mehrere Überprüfungen durch und startet den MyAnchorExchanger. Danach werden alle Anchor-Keys aus der Web-App lokal in einer Liste gespeichert. Dafür ist die Methode StoreAllAnchorKeys() zuständig. Die Update() Funktion speichert bei der Lokalisierungsphase die maximale Anzahl von Feature Points.

### **Region UI Methods**

Die Methoden in dieser Region kontrollieren die UI Elemente der Anwendung. Hier unterscheiden sich beide Klassen, da die ResearchSpatialApp zusätzlich den UIHandler verwendet. Um die UI Elemente zu kontrollieren, fungiert ein Objekt dieser Klasse. Im Prototyp MySpatialApp werden die UI Elemente in der Start() zugewiesen. In der Region werden zusätzlich noch die Anchor Daten mit der SaveDataToJson Klasse gespeichert. Dies trifft ebenfalls nur auf die ResearchSpatialApp Klasse zu.

### **Region Key/ID Methods**

Die Region beinhaltet nur eine Methode. Die Methode zum Speichern aller Anchor Keys aus der Web-App. Die Funktion wird beim Starten der Anwendung ausgeführt.

### **Region Override Methods**

In dieser Region werden die virtual Methoden aus den Basis-Klassen um spezifische Funktionalitäten ergänzt. Dabei ist die Funktion OnCloudAnchorLocated äußerst wichtig. Sie legt fest, was bei der erfolgreichen Lokalisierung eines Anchors getan werden soll. Dazu zählt beispielsweise das Erzeugen einer lokalen Darstellung des Anchors mit der Methode SpawnOrMoveCurrentAnchoredObject(...). Außerdem wird die Funktion IsPlacingObject() überschrieben. Die bool Method liefert true zurück, falls sich die Anwendung aktuell im Platzierungs-Status PlacingAnchor befindet.

### **Region AppMethods**

Diese Region definiert die Methoden, welche den Ablauf der Anwendung steuern. Zu Beginn jeder Funktionalität (Generierung oder Lokalisierung) wird die Methode

InitializeSession() ausgeführt. Sie erzeugt und startet eine Session. Die Methode StartPlacingSession () führt diese Funktion aus und startet den Platziungsprozess. Dieser wird durch die Funktion DonePlacingObject() beendet und der platzierte Anchor wird gespeichert. Die Methode LookingForObject() wird bei der Lokalisierung ausgeführt. Sie ruft ebenfalls die InitializeSession()-Methode auf und erzeugt einen CloudSpatialAnchorWatcher. Alle Funktionen in dieser Region, mit Ausnahme InitializeSession(), werden durch die Betätigung eines Buttons gestartet.

### 3.3.6. AnchorData

Jeder Cloud Anchor wird durch einen lokalen Anchor dargestellt. Dabei trägt das Prefab, welches den Anchor repräsentiert, die Daten des verbundenen Cloud Anchors. Dazu dient die Klasse AnchorData. Dieses Skript liegt auf dem Prefab, welches beim Erstellen eines neuen Anchors verwendet wird. Falls das Prefab keine AnchorData-Komponente besitzt, wird eine hinzugefügt. Das Prefab wird dann als GameObject im Unity Editor der Klasse MyAppBase bzw. ResearchAppBase via Drag and Drop übergeben und in der Variablen anchoredObjectPrefab gespeichert. In dieser Arbeit ist das Prefab ein 3D Objekt des Firmenlogos der meaPuna GmbH. Beispielweise werden die Daten eines Anchors, also die AppProperties beim erfolgreichen Lokalisieren in der Klasse gespeichert. Das Skript speichert wichtige Daten zum Cloud Anchor: AnchorKey, AnchorName, AnchorId, AnchorDate, AnchorInfo, AnchorProgress, AnchorGenerateMilliseconds, AnchorFeaturePoints, AnchorDescription, AnchorPosition, AnchorRotation, AnchorPositionLocalization und AnchorRotationLocalization.

### 3.3.7. MyAnchorExchanger

Ein Cloud Anchor kann in der Azure Spatial Anchor Version 1.3.3 nur über seinen eindeutigen Key lokalisiert werden. Dies macht das Zwischenspeichern des Keys in einer cloud-basierten Lösung obligatorisch. Azure umgeht mit seiner Demo der Verwendung einer Datenbank und speichert die Anchor Keys in einer ASP.NET Core-Web-App. Die Schnittstelle zwischen dieser Web-App und Unity ist die Klasse MyAnchorExchanger.

#### SharingService Solution

Das Erzeugen der Web-App ist in der Azure Dokumentation genau beschrieben [Micg]. In der Demo auf GitHub liefert Azure eine Solution für einen Sharing Service. Diese Solution muss mit Hilfe von Visual Studio oder Visual Studio Code veröffentlicht werden.

#### API

Die SharingService API bietet drei Kernmethoden. Es gibt zwei GET und eine POST Methode. Die POST Methode postet einen übergebenen AnchorKey in der Web-App und wird mit einer Nummer versehen. Diese Nummern sind immer automatisch aufsteigend von 0 beginnend. Die GET Methoden liefern einen AnchorKey aus der Web-App. Mit /api/anchors/last wird der letzte AnchorKey in der Liste zurückgegeben und mit /api/anchors/{anchorNumber} ist die Abfrage eines spezifischen Anchors anhand seiner Nummer möglich. Dabei muss die Anwendung genau wissen, welcher Anchor zu welcher Nummer in der Web-App gehört. In Abbildung 6 ist ein Beispiel zu sehen, wie ein Anchor in der Web-App gespeichert wird. Dieser Anchor hat die Nummer 0 erhalten und kann über den GET Befehl /api/anchors/0 abgerufen werden.

#### baseAddress

Hauptbestandteil der Klasse MyAnchorExchanger ist die Basis URL, also die Adresse, welche Zugriff auf die SharingService API gewährleistet. Im Rahmen dieser Arbeit lautet die Adresse der SharingService API <https://spatialanchorsWebAppcb.azurewebsites.net>. Um die POST und GET Methoden in Unity auszuführen, ist die Erweiterung der URL<sup>4</sup> um den Zusatz /api/anchors notwendig. Azure

---

<sup>4</sup> Uniform Resource Locator (URL)

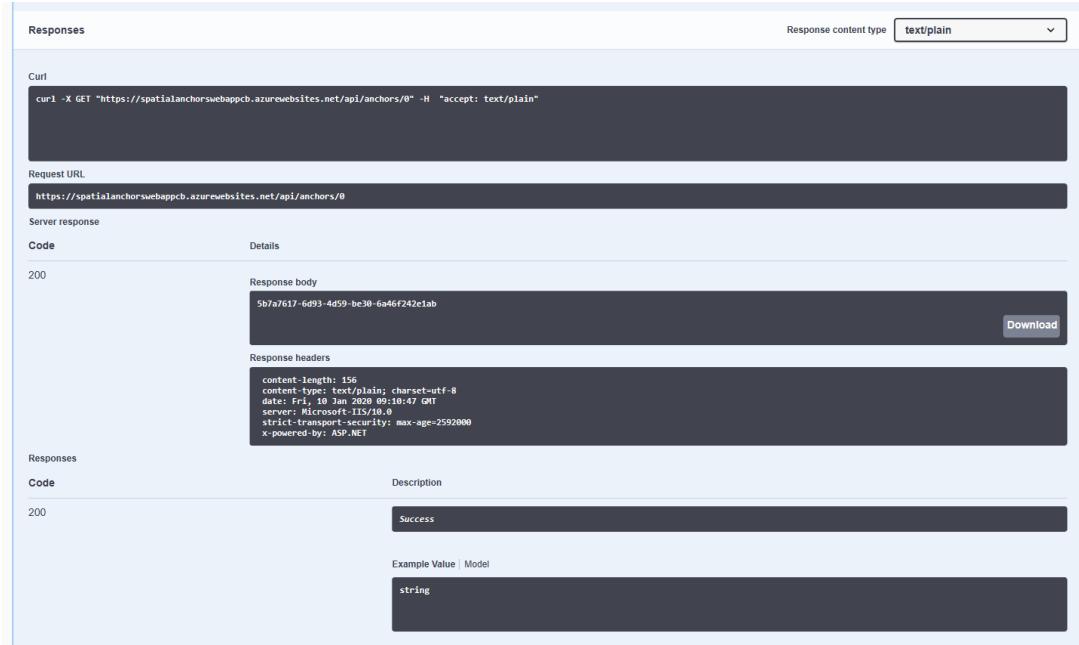


Abbildung 6.: Beispiel eines Anchors in der Web-App.

liefert hierzu die Configuration SpatialAnchorSampleConfig, welche die SharingService URL speichert. Als Adresse muss die URL mitsamt der Erweiterung angegeben werden, also: <https://spatialanchorsWeb-Appcb.azurewebsites.net/api/anchors>. Die Klasse ResearchSpatialApp zieht zu Beginn der Anwendung, also in der Start(), die Adresse aus der Configuration und speichert sie lokal in einer Variablen. Danach wird die Funktion WatchKeys(string exchangeUrl) aus der Klasse MyAnchorExchanger ebenfalls in der Start()-Methode der Klasse ResearchSpatialApp ausgeführt. Der Methode wird die zuvor gespeicherte Basis-Adresse aus der Configuration übergeben. Somit enthält der MyAnchorExchanger die Basis-Adresse.

### StoreAnchorKey

Der nachfolgende Code Snippet zeigt einen Teil der Funktion StoreAnchorKey(...) der Klasse MyAnchorExchanger. Diese Funktion speichert einen übergebenen AnchorKey in der Web-App mittels der POST Methode. Dazu dient die Klasse HttpClient aus dem Namespace System.Net.Http [Mica]. Ein HttpClient ermöglicht die Kommunikation mit einer URI<sup>5</sup> Ressource. Die Methode PostAsync speichert den AnchorKey in der Web-App. Ihr wird ein Objekt vom Typ Uri und

<sup>5</sup> Uniform Resource Identifier (URI)

ein `HttpContent` Objekt übergeben. Der übergebene URI ist die Basis-Adresse, also die `baseAddress`. Das `HttpContent` Objekt stellt der `AnchorKey` in Form eines `StringContent` dar.

```
internal async Task<long> StoreAnchorKey(string anchorKey)
{
    ...
    HttpClient client = new HttpClient();
    var response =
        await client.PostAsync(baseAddress,
            new StringContent(anchorKey));
    ...
}
```

### RetrieveAnchorKey und RetrieveLastAnchorKey

Beide Funktionen geben einen spezifischen `AnchorKey` zurück. Dazu werden ein `HttpClient` Objekt und die Funktion `GetStringAsync` benötigt. Sie unterscheiden sich lediglich um den hinteren Teil des angegebenen URI. `RetrieveAnchorKey(long anchorNumber)` gibt einen spezifisch geforderten `AnchorKey` zurück. `RetrieveLastAnchorKey()` liefert dagegen den zuletzt gespeicherten `AnchorKey`.

```
public async Task<string> RetrieveAnchorKey(long anchorNumber)
{
    HttpClient client = new HttpClient();
    return await client.GetStringAsync(baseAddress +
        "/" + anchorNumber.ToString());
}

public async Task<string> RetrieveLastAnchorKey()
{
    HttpClient client = new HttpClient();
    return await client.GetStringAsync(baseAddress + "/last");
}
```

### 3.3.8. UIHandler

Der `UIHandler` speichert alle UI Elemente mittels der `GameObject.Find()` Methode. Somit wird der Code ausgelagert. Beim ändern der UI Elemente in der Anwendung kann über ein Objekt der Klasse auf die einzelnen Elemente zugegriffen werden. Eine Ausnahme stellen die Feedback-Textboxen dar, welche die Klasse `ResearchAppBase` definiert.

### 3.3.9. SaveDataToJson

Diese Klasse speichert alle wichtigen Daten für die statistische Auswertung in einem Json Dokument. Sie implementiert das `SimpleJSONUnity` Plugin. Die Klasse besitzt zwei wichtige Funktionen. Die Funktion `StringToVector3` konvertiert einen `string` in ein `Vector3`. Dies ist notwendig, da die Position und Rotation in der Klasse `AnchorData` als `String` zwischengespeichert werden. Die Funktion `SaveData(GameObject dataObject, string seconds, float progress)` nimmt die Daten aus der `AnchorData` Komponente des übergebenen `GameObjects` und speichert diese in einem `JSONObject`. Die Position und Rotation werden mittels eines `JSONArray` korrekt abgespeichert. Um die Überschreibung von JSON Dokumenten zu verhindern, werden sämtliche Dokumente mit aktuellem Datum und Uhrzeit versehen.

```
string date = System.DateTime.Now.ToString  
("yyyy'-'MM'-'dd'_'HH'-'mm'-'ss");
```

Abschließend wird das fertige JSON Dokument in einem persistenten Datenverzeichnis lokal gespeichert. Der Befehl `Application.persistentDataPath` liefert den Pfad dieses Datenverzeichnisses. Der Vorteil an diesem Pfad ist, dass Dokumente in diesem Verzeichnis nicht durch Updates gelöscht werden können. Der Name des Dokuments setzt sich aus der ID des Anchors und aktuellem Datum und Uhrzeit zusammen.

```
string path = Application.persistentDataPath + "/DataSave"  
+ data.AnchorId + "_" + date + ".json";  
File.WriteAllText (path, dataJson.ToString ());
```



## 4. Methodik

Die Forschungsmethodik dieser Arbeit ist die Analyse und Evaluation einer neuen Technologie. Diese Arbeit bewegt sich im Bereich der angewandten Wissenschaften mit Forschung durch Entwicklung und der Erhebung von quantitativen Daten bei einem technischen Experiment. Die Forschungsfrage der Arbeit lautet:

*Welchen Einfluss nehmen Veränderungen der Szene innerhalb der Point Cloud auf das Lokalisieren von Spatial Anchors, solange mindestens ein Objekt in der Szene bleibt?*

### 4.1. Aktualität und Forschungsstand

Die Technologie Azure Spatial Anchors wurde Anfang 2019 erstmals vorgestellt mit der Veröffentlichung der Hololens 2. Speciale et al. erwähnen in ihrer Arbeit *Privacy Preserving Image-Based Localization* die Technologie im Zusammenhang mit der aufsteigenden Popularität von AR Systemen mit Bild-basierter Lokalisierung [SSK<sup>+</sup>19]. DeFanti beschreibt in seiner Dissertation *Co-Located Augmented and Virtual Reality Systems* das Problem, dass User innerhalb einer AR-Umgebung isoliert sind. Das Teilen von virtuellen Objekten und Inhalten auf mehreren Endgeräten beschreibt er als seltene Erfahrung in AR Systemen und Anwendungen. Viele SLAM-basierte AR Systeme können erkannte Landmarken nicht miteinander teilen. Somit ist eine gemeinsame Lokalisierung äußerst komplex. Cloud-basierte Technologien wie Azure Spatial Anchors sind der Anfang um dieses Problem zu lösen. DeFanti beschreibt jedoch auch, dass diese Technologien noch nicht weit verbreitet sind. [DeF19]

Die Thematik Azure Spatial Anchors hat eine große Aktualität und der Forschungsstand steht noch am Anfang. Besonders für Unternehmen ist diese Technologie zukunftsorientiert, da cloud-basierte AR-Systeme in vielen Bereichen einer Unternehmensstruktur eingebaut werden können. Diese Arbeit soll eine Lücke im Bereich dieser Technologie schließen.

## 4.2. Hypothese

Basierend auf der Forschungsfrage lassen sich folgende Hypothesen ableiten:

**H0:** Die Zeit der Lokalisierung eines Anchors unterscheidet sich signifikant von der Zeit der Lokalisierung nach Veränderung der Szene, solange mindestens ein Objekt in der Szene bleibt.

**HA:** Die Zeit der Lokalisierung eines Anchors unterscheidet sich nicht signifikant von der Zeit der Lokalisierung nach Veränderung der Szene, solange mindestens ein Objekt in der Szene bleibt.

Die Hypothese ist **ungerichtet** und **zweiseitig**. Sie legt fest, dass mindestens ein Objekt immer in der Szene bleibt. Dieses Objekt ist bei der Generierung aller Anchor vorhanden.

*Sub-Hypothesen* Die Sub-Hypothesen überprüfen kleinere Gegebenheiten. Die Anzahl der bei Sub-Hypothesen 1-4 entfernten bzw. hinzugefügten Objekte schließt sich aus der Anzahl von Objekten bei der Generierung. Die Sub-Hypothesen 5-6 basieren auf den Testergebnissen des ersten Tests.

1. Die Zeit der Lokalisierung eines Anchors unterscheidet sich nach der Wegnahme eines Objektes signifikant von der Zeit der Lokalisierung bei dem ursprünglichen Szenenaufbau.
2. Die Zeit der Lokalisierung eines Anchors unterscheidet sich nach der Wegnahme zweier Objekte signifikant von der Zeit der Lokalisierung bei dem ursprünglichen Szenenaufbau.
3. Die Zeit der Lokalisierung eines Anchors unterscheidet sich nach dem Hinzufügen eines Objektes signifikant von der Zeit der Lokalisierung bei dem ursprünglichen Szenenaufbau.
4. Die Zeit der Lokalisierung eines Anchors unterscheidet sich nach dem Hinzufügen zweier Objekte signifikant von der Zeit der Lokalisierung bei dem ursprünglichen Szenenaufbau.

5. Die Zeit der Lokalisierung bei höherem Lux-Belichtungswert unterscheidet sich von der Zeit der Lokalisierung bei geringerem Lux-Belichtungswert.
6. Die Zeit der Lokalisierung bei höherem Lux-Belichtungswert generierten Anchors unterscheidet sich von der Zeit der Lokalisierung bei niedrigerem Lux-Belichtungswert generierten Anchors.

### 4.3. Testkonzept

Für eine objektive Überprüfung der Hypothese werden technische Daten aus einem Experiment benötigt. Für diesen Test gab es multiple Ansätze. Zusätzlich traten mehrere Probleme auf, welche eine mehrfache Wiederholung des Tests unumgänglich machten.

#### 4.3.1. Grundidee

Für den Testaufbau wird ein schlichter Raum benötigt. Der Raum sollte weitestgehend leer und komplett abdunkelbar sein. Scheinwerfer kontrollieren die Lichtmenge in der Szene. Die Lichtverhältnisse werden in Lux gemessen. In der Mitte des Raumes befindet sich die Szene. Ein Tisch steht vor einem einfarbigen Hintergrund. Auf dem Tisch stehen kategorisierte Objekte. In der Mitte des Tisches befindet sich ein farbiger Klebepunkt. Auf diesem Punkt entsteht der Testanchor. Vor dem Tisch sind mehrere Markierungen auf dem Boden. Diese Markierungen sind essentiell für die möglichst gleichmäßige, einheitliche Generierung der Anchor. Außerdem befinden sich auf dem Boden mehrere Markierungen für ein Stativ, welches für die Lokalisierung notwendig ist. Um eine einheitliche, statistisch vergleichbare Lokalisierung zu ermöglichen, findet diese rein statisch mit Stativ statt. Zu Beginn steht eins von drei Objekten auf dem Tisch und der erste Anchor wird erzeugt. Dieses erste Objekt ist immer vorhanden, bei der Generierung und Lokalisierung jedes Anchors. Danach wird das zweite Objekt platziert und ein weiterer Anchor erzeugt. Dieser Vorgang muss für das dritte Objekt ebenfalls wiederholt werden. Somit entstehen drei Anchor unter drei verschiedenen Bedingungen. Bei der Lokalisierung wird jeder dieser Anchor unter allen drei Bedingungen mehrfach getestet ( $N \geq 10$ ). Somit ergeben sich für jeden Anchor mindestens 30 Messwerte.

### 4.3.2. Mögliche Einfluss- und Störfaktoren

Während des Tests gibt es mehrere Faktoren, die den Test beeinflussen oder stören könnten. Einige dieser lassen sich jedoch nicht kontrollieren oder spezifisch definieren.

- Internetverbindung
- Serververbindung von Azure
- Uneinheitliche Bewegungen bei der Generierung
- Kamera und Leistung des Smartphones

### 4.3.3. Research-Anwendung

Die Research-Anwendung muss in der Lage sein, Anchor zu generieren und spezifische Anchor über dessen ID zu lokalisieren. Die Anwendung lässt sich mittels eines Fadenkreuzes mit der Anchor-Markierung auf dem Tisch ansatzweise kalibrieren. Es ist wichtig, sämtliche visuellen Daten nach der Lokalisierung zu löschen. Die Anwendung basiert auf zwei Funktionalitäten: Der Erstellung eines Anchors und der Lokalisierung. In Kapitel 3.3 wird der genaue Aufbau der Research-Anwendung für das hier durchgeführte Experiment diskutiert. Die folgenden Paragraphen beschreiben den konzeptionellen Hintergrund dieser Anwendung.

*Erstellung* Die Erstellung eines Anchors besteht aus 4 Phasen. In der ersten Phase öffnet sich ein Fenster für die Eingabe von Name, ID und Info. Nach der Eingabe beginnt Phase 2. Durch Toucheingabe wird ein Anchor auf eine von ARFoundation generierte Plane gesetzt. Dieser kann bei erneuter Toucheingabe verschoben werden. Ist die Platzierung beendet, kommt der User über einen Button zur nächsten Phase. In dieser Phase werden visuelle Umgebungsdaten, basierend auf Punktewolken, gesammelt. Für eine möglichst einheitliche Generierung wird für jeden Anchor die gleiche Bewegung ausgeführt. Ist dieser Bewegungsablauf beendet, startet durch das Betätigen eines Buttons die nächste Phase. In Phase vier wird der Anchor nun gespeichert. Hierbei ist keine weitere Eingabe des User möglich. Wurde der Anchor erfolgreich gespeichert, startet die Anwendung neu.

*Lokalisierung* Die Lokalisierung eines Anchors besteht aus 3 Phasen. Jeder Anchor bekommt eine ID bei der Generierung. In der ersten Phase muss diese ID eingegeben werden. Nach der Eingabe startet Phase 2. In dieser Phase wird ein CloudSpatialAnchorWatcher erstellt (siehe Abschnitt 3.2.3) und ein Timer gestartet. Dieser Timer misst die Millisekunden bis zur Lokalisierung des Anchors. Bei erfolgreicher Lokalisierung startet Phase 3. Der Anchor wird angezeigt und ein Button für das Neustarten der Anwendung erscheint.

#### 4.4. Erhobene Daten

Zum Speichern der Daten aus Unity dient die Klasse SaveDataToJson (siehe Abschnitt 3.3.9). Folgende Daten speichert die Klasse:

- Name
- ID
- Erstellungsdatum und Uhrzeit
- Zusätzliche Information bei Eingabe bzw. Erstellung
- Zeit zum Erzeugen in Millisekunden
- Zeit zur Lokalisierung in Millisekunden
- Gesammelte Daten bei der Erzeugung
- Gesammelte Daten bei der Lokalisierung
- Anchor-Key
- Maximale Anzahl an Feature Points bei der Lokalisierung (Point Cloud)
- Maximale Anzahl an Feature Points bei der Erstellung (Point Cloud)
- Position und Rotation bei der Erstellung
- Position und Rotation bei der Lokalisierung



## 5. Experiment

In diesem Kapitel wird der praktische Teil des in Kapitel 4 beschriebenen Forschungsdesigns diskutiert. Der Ablauf der Testversuche wird geschildert und Probleme erläutert. Anschließend werden die erhobenen Daten der Stichprobe mittels deskriptiver und induktiver Statistik ausgewertet. Im Rahmen dieser Arbeit mussten zwei Testdurchläufe ausgeführt werden, da bei der Auswertung der Daten aus dem ersten Experiment mehrere Mängel auftraten. Das Konzept war nicht schlüssig durchdacht und bei der statistischen Auswertung machten sich Probleme bemerkbar. Der Testaufbau war bei beiden Tests identisch, lediglich die Erhebung der Daten unterscheidet sich.

### 5.1. Aufbau

Das erste Experiment fand vom 14.-16. Januar 2020 und das zweite am 28. Januar 2020 im Media Synthesis Lab (MSL) im I-Gebäude der Hochschule Furtwangen statt. Dieser Raum ist komplett abdunkelbar und die Belichtung erfolgt durch Scheinwerfer an der Decke. Für die Messung der Belichtung diente ein Belichtungsmesser von der Marke Minolta. In Abbildung 7 sind die drei Kategorie-Objekte erkennbar. Dabei handelt es sich um die Kategorien A, B und C. Kategorie A steht für eine Wasserkaraffe, B für drei Plastik-Kakteen und C für eine kabellose Tastatur. Im Test gibt es drei Zustände: ABC, AB und A. Dabei stehen die Buchstaben für die Kategorien, welche auf dem Tisch platziert sind. Das mobile Testgerät war ein Samsung Galaxy S9+ (Modellnummer SM-G965F) mit Android 9 (API Level 27). Während des Tests war der Flugmodus aktiviert und es wurde eine Internetverbindung zum hochschuleigenen eduroam-Netzwerk hergestellt. Bei der Auswertung und Beschreibung des Experiment wird zwischen dynamische und statischer Generierung bzw. Lokalisierung unterschieden. **Dynamisch** bedeutet, dass sich das Smartphone in der Hand des Users befindet und von diesem geführt wird. **Statisch** bedeutet die Platzierung des Smartphones in einem festen Stativ ohne Bewegungsmöglichkeiten.



Abbildung 7.: Szenenaufbau bei der Lokalisierung mit Stativ im Zustand ABC (Kategorie-Objekte). Das linke Bild wurde bei Experiment 2 und das rechte bei Experiment 1 aufgenommen. Die Markierungen und Abstände sind bei beiden Experimenten identisch.

### 5.1.1. Generierung

Im rechten Bild der Abbildung 7 ist der Aufbau samt Stativ ersichtlich. Auf dem Boden sind Markierungen um den Standort von Stativ und Tisch festzuhalten. Außerdem befinden sich im Abstand von 60cm zum Tisch Markierungen (rot), welche für die Generierung notwendig sind. Die Lokalisierung erfolgt statisch, um statistisch vergleichbare Daten zu erlangen. Bei der Generierung ist dies weitaus komplexer. Die Generierung von Anchors erfordert das Sammeln von Umgebungsdaten (Progress Data). Eine Option für das einheitliche Generieren wäre die Verwendung eines Roboter-Arms. Da dies den finanziellen und zeitlichen Rahmen dieser Arbeit überschreiten würde, gab es eine andere, weniger einheitliche Methode zur Generierung. Die Generierung startet bei der mittleren Markierung. Das Smartphone befindet sich auf Brusthöhe und es folgt ein immer gleicher Bewegungsablauf. Dieser Bewegungsablauf wird danach von der rechten und der linken Markierung wiederholt. Die Generierung endet auf der mittleren Markierung. Die bei der Generierung erzeugten Daten (siehe Kapitel 4) sind statistisch nicht einheitlich. Bei der Generierung wird das Stativ aus der Szene entfernt.

### 5.1.2. Lokalisierung

In Abbildung 8 ist der Aufbau bei der Lokalisierung sichtbar. Das Smartphone befindet sich in einer horizontalen Halterung. Durch die Smartphone-Kamera sind der Tisch und die darauf befindlichen kategorisierten Objekte sichtbar. Im Hintergrund befindet sich die Wand des GreenScreens. In der Mitte des Tisches ist eine Markierung, auf welcher der Anchor platziert wird. Das Fadenkreuz der Anwendung ist bei der Lokalisierung auf diese Markierung ausgerichtet. Im Laufe des Tests wird jeder Anchor unter jeder Bedingung getestet. Die Objekte haben genaue Markierungen auf dem Tisch, damit sie immer gleich platziert werden.

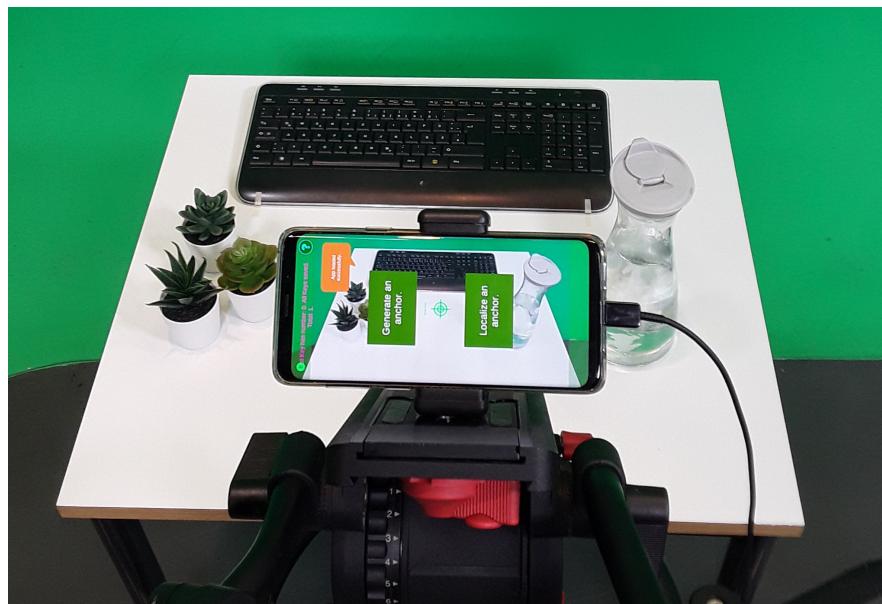


Abbildung 8.: Sicht auf die Szene durch die Research-Anwendung bei der Lokalisierung.

## 5.2. Erstes technisches Experiment

Das Konzept des ersten Experiments war aus statistischer Sicht mit zu vielen Fehlern behaftet. Es wurden zu viele unterschiedliche Daten gesammelt, jedoch zu wenig Daten pro Anchor. Jeder Anchor wurde dreifach pro Bedingung getestet. Das ergibt 9 Messwerte pro Anchor jeweils bei hellem und bei dunklem Licht. Jeweils drei für die Bedingungen ABC, AB und A. Um statistisch signifikante Ergebnisse zu erzielen,

sind mehr als drei Messwerte pro Bedingung erforderlich. Ein neuer Test schließt mehrere Bedingungen wie Licht und Untergrund vorerst aus und zeichnet dafür mehr Messwerte pro Anchor und Bedingungen auf. Die Messergebnisse aus diesem ersten Test können jedoch für eine statistische Auswertung der Lichtverhältnisse verwendet werden. Diese Testergebnisse sind jedoch mit Vorsicht zu behandeln.

### 5.3. Zweites technisches Experiment

Das zweite Experiment ist simpler aufgebaut. Der Fokus liegt auf den in Kapitel 4 beschriebenen Hypothesen. Es werden auf Unterschiede bei Lichtverhältnissen und Untergrund verzichtet. Die Szene wird mit 860-890 Lux bei 4500 Kelvin ausgeleuchtet. Es wurden drei Anchor bei drei Bedingungen erzeugt: ABC, AB und A. Danach wurde jeder Anchor unter jeder Bedingung 10mal lokalisiert. Dies ergibt für jeden Anchor 30 Messwerte. Zusätzlich wurde getestet, ob der jeweilige Anchor nach Wegnahme aller Objekte lokalisiert werden kann. Dies war statisch nicht möglich.

### 5.4. Probleme

Während der Experimente traten mehrere Probleme auf, sowohl Hardware- als auch Software-basiert. Im folgenden sind sämtliche Probleme mit Lösungsansatz aufgelistet:

- Die verwendete Smartphone-Halterung für das Stativ ermöglichte nur eine horizontale Ausrichtung des Smartphones während die Research-Anwendung für eine vertikale Nutzung konzipiert und entwickelt wurde. Dieses Problem konnte durch die Anpassung der Player Settings in Unity behoben werden. Die Anwendung erlaubt nur die Verwendung im Portrait-Modus und verhindert den Wechsel in Landscape-Modus beim Drehen des Smartphones.
- Das erste Experiment enthielt neben den Objekten und den Lichtverhältnissen eine dritte Bedingung: Veränderung des Untergrunds. Dazu wurde eine blau-weiß karrierte Tischdecke als alternativer Untergrund verwendet. Die Idee dabei war, dass ein Anchor mehr visuelle Daten sammeln und damit besser lokalisiert werden kann. Dies war jedoch ein Irrtum, da die Lokalisierung mit karriertem Untergrund große Probleme aufwies. Die Lokalisierung wurde erschwert und bereits nach der Wegnahme von zwei Objekten konnte kein Anchor mehr gefunden werden. Als Ursache dafür gibt es mehrere Ansätze:

- Das letzte Objekt auf dem Tisch, Objekt A, ist die mit Wasser gefüllt Karaffe. Die durchscheinende Tischdecke wird durch das Wasser gebrochen, was die Erkennung der Punktewolke bei der Lokalisierung stören könnte.
- Die Tischdecke war dehnbar und nur leicht an den Enden des Tisches befestigt. Minimale Berührungen könnten das Karomuster verändert haben, was zu einer Veränderung der Punktewolke führte.
- Da das Karomuster symmetrisch ist, hat der Anchor keinen genauen Anhaltspunkt im Verhältnis zur Wasserkaraffe. Hierbei wäre ein asymmetrisches Muster wirksamer.

Der genaue Grund für die fehlerhafte Lokalisierung mit unterschiedlichem Untergrund ist unklar und bedarf einer neuen und umfangreicheren Untersuchung. Die Ergebnisse des Experiments werden daher verworfen.

- Für Experiment 1 wurde anfänglich ein Samsung A8 verwendet. Dies hatte bei der Testreihe mit karriertem Untergrund große Probleme Oberflächen zu erkennen. Dies könnte an der Hardware des Smartphones in Kombination mit ARCore liegen. Das Experiment wurde mit einem Samsung S9+ wiederholt, welches ebenfalls für Experiment 2 verwendet wurde.
- Bei der ersten Analyse der Daten war nicht ersichtlich, welche Objekte sich zum Zeitpunkt der jeweiligen Lokalisierung in der Szene befanden. Es wurde eine weitere User-Eingabe hinzugefügt. Der User gibt dabei die Testphase gemeinsam mit der ID des gewünschten Anchors ein.
- Die Methode `StoreAllAnchorKeys()` benötigte einen großen Performance-Aufwand, was das Laden der Anwendung sehr zeitintensiv machte. Nach Überprüfung der Syntax stellte sich ein Denkfehler heraus. Der Code konnte reduziert und die Performance verbessert werden.

## 5.5. Statistische Auswertung

Für die statistische Auswertung wurde das Statistik Tool SOFA verwendet [PS]. Bei der Auswertung liegt das Augenmerk auf den gemessenen Millisekunden für die Lokalisierung von Anchors. Die Auswertung teilt sich auf in deskriptive und induktive Statistik. Im Anhang F befinden sich die Tabellen für die nachfolgenden Auswertungen. Die originalen JSON Daten sind im GitHub Repository unter <https://github.com/nincara/MasterThesis> gespeichert.

### 5.5.1. Deskriptive Statistik

Die deskriptive Statistik dient zur Beschreibung der erhobenen Daten einer Stichprobe. In Tabelle 3 sind die wichtigen Kenngrößen für die erhobenen Daten bei der Lokalisierung. Dazu zählen der Median, der Mittelwert, der maximale und minimale Wert und die Standardabweichung. Die Tabelle zeigt die erhobenen Daten aus dem zweiten Test. Da die Daten des ersten Tests pro Anchor nur  $N = 3$  Messwerte aufweisen, werden diese nicht deskriptiv dargestellt. Die Abbildungen im Anhang E zeigen Diagramme, welche die deskriptive Darstellung der Messwerte grafisch darstellen.

Phase - ID	N	Median	Mittelwert	Min	Max	Standardabweichung
A - 000	10	1267.0 ms	1274.8 ms	1192.0 ms	1405.0 ms	61.87 ms
AB - 000	10	1325.0 ms	1341.8 ms	1160.0 ms	1795.0 ms	180.23 ms
ABC - 000	10	1405.5 ms	1388.5 ms	1203.0 ms	1531.0 ms	94.82 ms
A - 001	10	1296.0 ms	1285.1 ms	1205.0 ms	1361.0 ms	52.49 ms
AB - 001	10	1431.0 ms	1471.8 ms	1231.0 ms	1882.0 ms	168.29 ms
ABC - 001	10	1363.5 ms	1378.8 ms	1330.0 ms	1501.0 ms	57.48 ms
A - 002	10	1291.0 ms	1288.9 ms	1196.0 ms	1392.0 ms	57.83 ms
AB - 002	10	1380.0 ms	1377.8 ms	1299.0 ms	1431.0 ms	46.18 ms
ABC - 002	10	1386.0 ms	1480.8 ms	1270.0 ms	2281.0 ms	302 ms

Tabelle 3.: Deskriptive Kennzahlen der erhobenen Daten aus dem zweiten Experiment.

### 5.5.2. Induktive Statistik - Mann-Whitney-U-Test

Mit Hilfe von SOFA-Statistics wurden die Messwerte der Stichprobe auf ihre Normalverteilung geprüft. Die Normalverteilung kann nicht bestätigt werden, jedoch auch

nicht vollends ausgeschlossen. Auf Grund dessen wird zur statistischen Auswertung der Variable *LocalizeMSeconds*, die Anzahl der Millisekunden für die Lokalisierung eines Anchors, der Mann-Whitney-U-Test verwendet. Der Mann-Whitney-U-Test untersucht zwei unabhängige Stichproben auf Unterschiede. Das Verfahren ist nicht-parametrisch, zweiseitig und setzt keine Normalverteilung voraus. Im Gegensatz zum beliebten unabhängigen t-Test wird der Mann-Whitney-U-Test bei einer geringen Stichprobenzahl bevorzugt. Der Test ist zudem robust gegenüber starken Ausreißern. Während dem Test wurden verschiedene Messwerte aufgezeichnet. Der Messwert für die maximalen Punkte der Punktewolken bei der Lokalisierung konnte nicht korrekt aufgezeichnet werden, da die Lokalisierung statisch verlief. Die Daten für die Position können nicht statistisch verglichen werden, da die Generierung dynamisch und die Lokalisierung statisch durchgeführt wurden.

## Experiment 2

In Tabelle 4 sind die p-Werte der verschiedenen Lokalisierungs-Phasen mit der gleichen ID und der gleiche Phasen verschiedener IDs. Für einen signifikanten Unterschied muss der p-Wert  $< 0.01$ , bestenfalls  $< 0.001$  sein. [PS, Hoc, Kro16]

	A, 000	AB, 000	ABC, 000	A, 001	AB, 001	ABC, 001	A, 002	AB, 002	ABC, 002
A, 000	-	0.4057	0.01014	0.6499	-	-	0.6231	-	-
AB, 000	0.4057	-	0.1124	-	0.02334	-	-	0.1403	-
ABC, 000	0.01014	0.1124	-	-	-	0.5967	-	-	0.9397
A, 001	0.6499	-	-	-	0.0746	0.0746	0.9097	-	-
AB, 001	-	0.02334	-	0.0746	-	0.04511	-	0.06959	-
ABC, 001	-	-	0.5967	0.0746	0.04511	-	-	-	0.7913
A, 002	0.6231	-	-	0.9097	-	-	-	0.1236	0.3237
AB, 002	-	0.1403	-	-	0.06959	-	0.1236	-	0.7623
ABC, 002	-	-	0.9397	-	-	0.7913	0.3237	0.7623	-

Tabelle 4.: Tabelle mit p-Werten des Mann-Whitney-U-Tests mit SOFA.

### Experiment 1: Lichtverhältnisse

Im ersten Testverlauf wurden drei Anchor bei 750 Lux und drei Anchor bei 10 Lux generiert. Alle Anchor wurden wiederum bei den hellen sowie bei den dunklen Lichtverhältnissen getestet sowie in Abschnitt 5.2 beschrieben. Dies ergibt eine Anzahl von 27 Messwerten pro Part. Die Daten und Ergebnisse sind mit Vorsicht zu behandeln, da pro Anchor zu wenig Messdaten erhoben wurden. Sie sind nicht statistisch signifikant und dienen hier nur zu Veranschaulichung des ersten Testversuch.

Die Daten teilen sich auf in zwei Gruppen. Gruppe 1 wurde bei 750 Lux und Gruppe 2 bei 10 Lux lokalisiert. Dabei teilen sich die Gruppen wiederum in zwei Parts auf. Part 1 wurde bei 750 Lux und Part 2 bei 10 Lux generiert. Beide Gruppen sind vom Aufbau und Ablauf identisch (siehe Abbildung 9). Bei dem ersten Ansatz für den Vergleich von Lichtunterschieden werden die beiden Gruppen und jeweils Part 1 beider Gruppen und Part 2 beider Gruppen mit dem Mann-Whitney-U-Test verglichen.

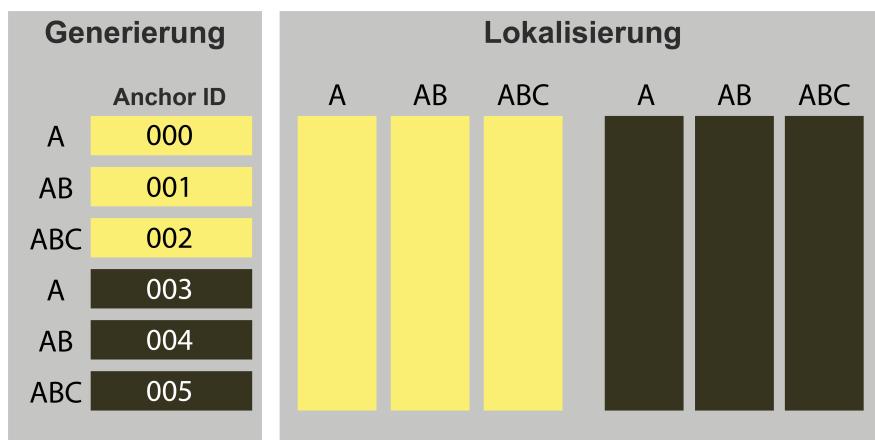


Abbildung 9.: Grafische Darstellung der Datenerhebung aus dem zweiten Test samt Belichtung und Phase. Gelesen von links nach rechts. Gelbe Flächen stehen für 750Lux, dunkelgraue Flächen für 10Lux.

*Gruppe 1 und Gruppe 2* Gruppe 1 ( $N = 54$ ) und Gruppe 2 ( $N = 54$ ) unterscheiden sich um den Belichtungswert bei der Lokalisierung. Es werden beide Gruppen direkt miteinander verglichen. Durch den Mann-Whitney-U-Test ergibt sich ein p-Wert von 0.1649.

*Gruppe 1 Part 1 und Gruppe 1 Part 2* Gruppe 1 Part 1 ( $N = 27$ ) und Gruppe 1 Part 2 ( $N = 27$ ) unterscheiden sich um den Belichtungswert bei der Generierung. Beide Parts wurden bei 750 Lux lokalisiert. Bei dem Vergleich mit Mann-Whitney-U-Test ergibt sich ein p-Wert von 0.5109.

*Gruppe 2 Part 1 und Gruppe 2 Part 2* Gruppe 2 Part 1 ( $N = 27$ ) und Gruppe 2 Part 2 ( $N = 27$ ) unterscheiden sich um den Belichtungswert bei der Generierung. Beide Parts wurden bei 10 Lux lokalisiert. Bei dem Vergleich mit Mann-Whitney-U-Test ergibt sich ein p-Wert von 0.8763.

*Gruppe 1 Part 1 und Gruppe 2 Part 1* Gruppe 1 Part 1 ( $N = 27$ ) und Gruppe 2 Part 1 ( $N = 27$ ) unterscheiden sich um den Belichtungswert bei der Lokalisierung. Beide Parts wurden bei 750 Lux generiert. Bei dem Vergleich mit Mann-Whitney-U-Test ergibt sich ein p-Wert von 0.5221.

*Gruppe 1 Part 2 und Gruppe 2 Part 2* Gruppe 1 Part 2 ( $N = 27$ ) und Gruppe 2 Part 2 ( $N = 27$ ) unterscheiden sich um den Belichtungswert bei der Lokalisierung. Beide Parts wurden bei 10 Lux generiert. Bei dem Vergleich mit Mann-Whitney-U-Test ergibt sich ein p-Wert von 0.2035.



## 6. Evaluation und Ausblick

### 6.1. Abschließende Beurteilung Spatial Anchors

Durch Azure Spatial Anchors ist das Setzen und Lokalisieren von Cloud Anchor in Cross-Plattform-Anwendungen möglich. Diese Technologie ist noch sehr neu und wird stetig weiterentwickelt. Bei Start dieser Arbeit war die Version der SDK v1.3.2 vom 30.08.2019. Am 05.11.2019 wurde diese auf Version v1.3.3 aktualisiert und am 19.11.2019 erschien die Version v2.0.0<sup>1</sup> [Micel]. Mit dieser Version reagiert Azure auf das Feedback mehrerer Entwickler, welche die Lokalisierung basierend auf einen Standort wünschen. Azure integriert nicht-visuelle Sensor-Signale wie GPS, WiFi und Bluetooth Low Energy (BLE). Durch diese Erweiterung erkennt Azure Spatial Anchors alle Anchors basierend auf den Standort-Daten. Die Integrierung dieser Funktion ist optional und im Umfang dieser Arbeit wurde sie ausgelassen.

Subjektiv betrachtet erwies sich die Entwicklung mit Azure Spatial Anchors zu Beginn als äußerst komplex. Die dazugehörige Dokumentation ist sehr dünn und beinhaltet noch keine Beispiele oder tiefere Erklärungen [Michb]. Außerdem gibt es wenige Forenbeiträge und Unterstützung online. Für die Einarbeitung und Entwicklung stellte die von Azure zu Verfügung gestellte Demo eine große Hilfe dar. Die Klassen der SDK wie `SpatialAnchorManager` und `CloudNativeAnchor` beinhalten viele aufschlussreiche Kommentare.

Ein größeres Problem der Entwicklung bei Beginn der Bearbeitung stellte die Lokalisierung von Anchors dar. Es war zwingend notwendig, den Key eines Anchors nach der Generierung zu speichern. Wie in Abschnitt 3.2.3 und 3.2.5 beschrieben, ist die Klasse `CloudSpatialAnchorWatcher` für die Lokalisierung eines Anchors notwendig. Die Klasse `AnchorLocateCriteria` aus Abschnitt 3.2.4 speichert sämtliche Kriterien für das Lokalisieren bestimmter Anchor. Hierbei speichert die Eigenschaft `Identifiers` die Keys der Anchor. In der Dokumentation [Micb] steht für diese Eigenschaft folgende

---

<sup>1</sup> siehe <https://github.com/Azure/azure-spatial-anchors-samples/releases>

Erläuterung: "*Cloud anchor identifiers to locate. If empty, any anchors can be located.*" [Micb]. Diese Beschreibung ist irreführend. Es könnte verstanden werden, dass wenn kein spezifischer Key übergeben wird, die Lokalisierung jedes Anchors möglich ist. Azure Spatial Anchors benötigt aber in v1.3.3 exakte Informationen zu den Anchors, welche lokalisiert werden sollen. Dies bedeutet, die Keys der Anchors erfordern einen globalen Speicherort in einer Datenbank oder ähnlichem. Azure schlägt die Verwendung einer Web-App vor (siehe Abschnitt 3.3.7) [Micg]. Dies setzt die Erzeugung der Ressourcen *App Service* und *App Service-Plan* voraus. Dabei umgeht Azure die detaillierte Beschreibung der Kosten dieser Web-App mitsamt Service-Plan. Es werden verschiedene Tarife für den Service-Plan genannt, jedoch nicht die Funktionsfähigkeit bei Verwendung der jeweiligen Tarife. In Abbildung 10 sind die drei Haupttarife zu sehen. Tarif F1 ist kostenfrei, beschränkt sich jedoch auf 1GB Arbeitsspeicher für 60 Minuten pro Tag. Dies bedeutet, die gespeicherten Keys werden nach 60 Minuten aus der Web-App gelöscht. Bei der Entwicklung ist dies ausreichend, jedoch nicht für die Ausführung des in Abschnitt 4 beschrieben Experiments. Auf Grund dessen wurde der Tarif kurzzeitig für die Durchführung des Experiments auf B1 gewechselt. Zu Beginn der Arbeit und der Analyse sind vermehrt Kosten angefallen, da standardmäßig die Kategorie Produktion mit dem Tarif P1V1 (siehe Abbildung 11) ausgewählt wird. Erst nach einiger Zeit ist dieser Kostenfaktor aufgefallen und der Tarif wurde auf F1 geändert. Azure schneidet dieses Thema nur knapp an und erwähnt nicht die Restriktionen bei der Verwendung des kostenfreien Tarifs.

Empfohlene Tarife		
<b>F1</b>	Freigegebene Infrastruktur 1 GB Arbeitsspeicher 60 Minuten/Tag für Computerressourcen Frei	
<b>D1</b>	Freigegebene Infrastruktur 1 GB Arbeitsspeicher 240 Minuten/Tag für Computerressourcen 8.00 EUR/Monat (geschätzt)	
<b>B1</b>	ACU gesamt: 100 1.75 GB Arbeitsspeicher Äquivalent zur Computerressource der A-Serie 46.17 EUR/Monat (geschätzt)	

Abbildung 10.: Übersicht über die verschiedenen Tarife der Azure-Ressource App Service-Plan für die Kategorie Entwickeln/Testen. Die Abbildungen stammen aus dem Azure Portal und benötigen eine Anmeldung. Unter [Micc] sind die Tarife aufgelistet.

Empfohlene Tarife	
<b>S1</b>	ACU gesamt: 100 1.75 GB Arbeitsspeicher Äquivalent zur Computerressource der A-Serie 61.56 EUR/Monat (geschätzt)
<b>P1V2</b>	ACU gesamt: 210 3.5 GB Arbeitsspeicher Äquivalent zur Computerressource der Dv2-Serie 123.12 EUR/Monat (geschätzt)

Abbildung 11.: Übersicht über die zwei Tarife der Azure-Ressource App Service-Plan für die Kategorie Produktion. Die Abbildungen stammen aus dem Azure Portal und benötigen eine Anmeldung. Unter [Micc] sind die Tarife aufgelistet.

Bei der Entwicklung mit Unity dient ARFoundation (siehe Kapitel 2.3) für die AR Funktionalitäten. Über ARFoundation bekommt der Entwickler umfassenden Zugriff sowie Informationen auf die Point Cloud. Die Reference Points können gespeichert und angezeigt werden. Dies ist bei den gesammelten Daten für einen Cloud Anchor nicht der Fall. Ein Cloud Anchor basiert auf gesammelten visuellen Daten, also auf einer Point Cloud. Diese Point Cloud, mit welcher der Anchor verknüpft ist und welche für die Lokalisierung ausschlaggebend ist, lässt sich nicht abrufen oder verändern. Die Begrenzung der Point Cloud eines Anchors wäre in mehreren Fällen interessant. Befindet sich beispielsweise ein Anchor auf einer industriellen Maschine, so könnten alle Reference Points der Umgebung ausgeblendet werden. Der Anchor bleibt immer in Beziehung zu der Maschine, auch bei Änderung der Umgebung. Die Ergebnisse aus Kapitel 5.5 zeigen jedoch, dass selbst bei Veränderung der Umgebung und der Szene ein Anchor lokalisiert werden kann, solange eine bestimmte Anzahl von Reference Points in der Punktwolke gleich bleiben. Dies macht die Cloud Anchor sehr robust gegenüber Veränderungen in der Point Cloud.

Während der Entwicklung gab es des öfteren Probleme mit Azure. Am 19. Februar 2020 traten Probleme mit der Authentifizierung zwischen der Anwendung und dem Azure Spatial Anchors Account auf. Im Azure Portal unter <https://status.azure.com/de-de/status> ist der Status jeder Ressource einsehbar. Es wurde kein Fehler mit der Mixed Reality Ressource *Spatial Anchors* gemeldet. In dem Thread *Issues* im GitHub Repository<sup>2</sup> werden Probleme und Bugs diskutiert. Dabei sind jedoch viele Anfragen offen und ungeklärt. Auch das Problem vom 19. Februar trat bei einem weiteren User über Android Studio auf. Das Problem wurde im Laufe des Tages gelöst und die Authentifizierung funktionierte wieder. Azure lieferte

<sup>2</sup> <https://github.com/Azure/azure-spatial-anchors-samples/issues>

im Zeitraum der Bearbeitung dieser Arbeit keine Informationen zu diesem Problem. Dies könnte häufiger auftreten, daher sollte diese Situation vom Entwickler bereits im Vorfeld abgefangen und berücksichtigt werden. Am 20. Februar 2020, einen Tag später, gab es Probleme mit dem Abfragen von AppProperties. Die Abfrage schlägt fehl und unterbricht den Vorgang. Die Anzeige von zuvor lokalisierten Anchors schlägt fehl. Nach einiger Zeit funktionierte dies jedoch wieder, ohne Änderung am Code. Bei zukünftiger Verwendung der hier beschriebenen und entwickelten Anwendungen sollte daher darauf geachtet werden.

## 6.2. Abschließende Beurteilung der Experimente

Abschließend zu den Ergebnissen aus Kapitel 5.5 lässt sich eine hohe Robustheit der Technologie schließen. Sowohl in Bezug auf die Veränderung der Szene sowie der Lichtverhältnisse. Die nachfolgenden Paragraphen diskutieren die Ergebnisse der Induktiven Statistik aus Kapitel 5.5.2.

*Experiment 1* Die Ergebnisse aus Abschnitt 5.5.2 zeigen die p-Werte resultierend aus dem Mann-Whitney-U-Test für die Daten aus dem zweiten Experiment. Alle p-Werte sind  $> 0.01$  und somit existiert kein signifikanter Unterschied zwischen den verschiedenen Phasen. Für den Vergleich von Phase ABC und A bei dem Anchor mit der ID 000 ergibt sich ein p-Wert von 0.01014. Dies ist der einzige p-Wert, welcher nahe an die 0.01-Grenze gelangt. Der Anchor wurde unter Bedingung A erzeugt, also mit einziger Wasserkaraffe in der Szene. Bei der Lokalisierung in Phase ABC befinden sich zwei fremde Objekte in der Point Cloud. Dies könnte ein Hinweis darauf sein, dass es signifikante Unterschiede bei der Lokalisierung von Anchors gibt, wenn fremde Objekte hinzufügt werden. Es ist jedoch zu beachten, dass es sich bei den erhobenen Daten um Millisekunden handelt. Der Mittelwert für Anchor 000 bei Testphase A beträgt  $1274.8ms$  und bei Testphase ABC  $1388.5ms$ . Daraus resultierend wird die Nullhypothese **H0** abgelehnt und die Alternativhypothese **HA** akzeptiert. Die Sub-Hypothesen 1 - 3 werden ebenfalls abgelehnt. Die Sub-Hypothese 4 kann, basierend auf den oben beschriebenen Resultaten, akzeptiert werden. Da es sich um eine sehr kleine Einheit handelt, müsste diese Gegebenheit erneut getestet werden. Im Unfang dieser Arbeit wird die Sub-Hypothese 4 abgelehnt.

*Experiment 2* Für alle Auswertungen der einzelnen Parts aus Gruppe 1 und 2 (siehe Abschnitt 5.5.2) ergibt sich ein p-Wert > 0.01, dies bedeutet, es besteht kein signifikanter Unterschied zwischen den Parts und den Gruppen. Um dieses Ergebnis jedoch signifikant beweisen zu können, muss der Test mit deutlich mehr erhobenen Messdaten wiederholt werden. Die Sub-Hypothesen 5 und 6 werden in diesem Rahmen abgelehnt.

Alle Hypothesen und Subhypothesen wurden abgelehnt. Es bestehen keine signifikanten Unterschiede für das in Kapitel 4 beschriebene Forschungskonzept. Das Forschungskonzept bietet jedoch eine Menge Potenzial für Verbesserungen und Änderungen, um noch signifikantere Ergebnisse zu erzielen.

### 6.3. Ausblick

Diese Arbeit deckt nur einen kleinen Teil der möglichen Entwicklungs- und Forschungsgebiete zum Thema Azure Spatial Anchors ab. Während der Bearbeitung und Untersuchung entwickelten sich neue Ansätze und Verbesserungsmöglichkeiten für weitere wissenschaftliche Forschungsarbeiten:

- Mit der Integrierung von standortbasierter Lokalisierung über nicht-visuelle Sensoren bieten sich neue Möglichkeiten zur Entwicklung. Dadurch könnte auch die Speicherung von Anchor-Keys in einer cloud-basierten Lösung entfallen. Für eine optimale Kontrolle ist die Kombination beider Ansätze eine Option.
- Die Analyse beschränkt sich auf die Entwicklung mit Unity und wird ausschließlich für Android getestet. Die Analyse und die Entwicklung könnten auf iOS Geräte sowie bei Verfügung auf die Hololens 2 ausgeweitet werden. Zusätzlich bieten sich hier neue Forschungsansätze. Die Lokalisierung von Anchors könnte zwischen verschiedenen Betriebssystemen verglichen werden.
- Für statistisch signifikantere Ergebnisse kann der Experimentaufbau optimiert werden. Für die Generierung könnte ein Roboterarm dienen, welcher immer gleiche Bewegungsabläufe vollzieht.
- Die resultierenden Ergebnisse aus dem in dieser Arbeit durchgeführten Experiment sind nicht nutzerorientiert sondern rein statistisch zu betrachten. Die Lokalisierung bei Nutzeranwendungen ist nicht statisch sondern dynamisch. Ein

Probandentest, in welchem User ein Szenario durchspielen müssen, könnte aufschlussreiche nutzerorientierte Daten generieren. Ein solcher Test könnte um standardisierte subjektive Fragebögen wie der NASA Taskload Test (NASA-TLX) oder der Simulator-Sickness-Questionnaire (SSQ) ergänzt werden.

- Da während des Experiments mehrere Komplikationen auftraten, entwickelten sich Ansätze für einen besseren, statistisch signifikanteren Forschungsansatz. Im Rahmen einer weiteren Masterarbeit, Studienarbeit oder eines Forschungsprojektes könnte die Technologie im größeren Umfang erforscht und mit mehr Experimenten statistisch evaluiert werden. In dieser Arbeit lag der Fokus auf der Lokalisierung von Anchors ohne größere Berücksichtigung auf die Generierung. Eine Option wäre die Generierung von mehreren Anchors unter derselben Bedingung und darauffolgend die Lokalisierung dieser wie in Kapitel 4 beschrieben. So könnten für jede Bedingung  $N \geq 10$  Anchors erzeugt und jeder dieser Anchors wieder für jede Bedingung  $N \geq 10$  mal getestet werden. Bei drei Bedingungen ergäbe das eine Anzahl von  $N \geq 900$  Messwerten. Diese Messung könnte daraufhin durch Unterschiede in Belichtung und Untergrund erweitert werden.
- Die Integrierung der standortbasierten Lokalisierung bietet weitere Möglichkeiten für Forschungsansätze. Das Forschungsdesign aus Kapitel 4 könnte sich darauf übertragen lassen. Dabei wird nicht nach einem Anchor mit spezifischer ID gesucht, sondern nach Anchors, welche sich im jeweiligen Wifi, GPS oder Bluetooth Radius befinden.

## Literaturverzeichnis

- [ABB<sup>+</sup>01] AZUMA, Ronald ; BAILLOT, Yohan ; BEHRINGER, Reinhold ; FEINE, Steven ; JULIER, Simon ; MACINTYRE, Blair: *Recent Advances in Augmented Reality*. Paper, 2001
- [ARC] ARCORE: *ARCore supported devices*. <https://developers.google.com/ar/discover/supported-devices>, Abruf: 03.01.20
- [Azu97] AZUMA, Ronald T.: *A Survey of Augmented Reality*. Paper, Presence, Vol. 6, No. 4, 1997
- [Chr] CHRISTOPH FEHLING AND DR. FRANK LEYMAN: *Cloud Computing*. <https://wirtschaftslexikon.gabler.de/definition/cloud-computing-53360/version-276453>, Abruf: 03.02.2020
- [Dat19] DATA, Priori: *Ranking der erfolgreichsten Apps im Google Play Store nach Umsatz weltweit im August 2019 (in Millionen US-Dollar) [Graph]*. <https://de.statista.com/statistik/daten/studie/689089/umfrage/apps-im-google-play-store-nach-umsatz-weltweit/>, 2019. – "[Online; accessed 11-September-2019]"
- [Dav04] DAVID NISTÉR AND OLEG NARODITSKY AND JAMES BERGEN: *Visual Odometry*. Paper, 2004
- [DeF19] DEFANTI, Connor: *Co-Located Augmented and Virtual Reality Systems*. Dissertation, 2019
- [Hoc] HOCHSCHULE LUZERN: *Mann-Whitney-U-Test*. <https://www.empirical-methods.hslu.ch/entscheidbaum/unterschiede/zentrale-tendenz/mann-whitney-u/>, Abruf: 29.01.2020
- [Jud06] JUDITH BISHOP AND NIGEL HORSPOOL: *Cross-Platform Development: Software that Lasts*. Paper, 2006
- [Kre07] KREVELEN, D.W.F. van: *Augmented Reality: Technologies, Applications, and Limitations*. Paper, 2007
- [Kro16] KRONTHALER, Franz: *Statistik angewandt - Datenanalyse ist (k)eine Kunst, Excel Edition*. Berlin : Springer Verlag, 2016. – ISBN 9783662471135

- [Mica] MICROSOFT: *HttpClient Klasse.* <https://docs.microsoft.com/de-de/dotnet/api/system.net.http.httpclient?view=netframework-4.8>, Abruf: 10.01.2020
- [Micb] MICROSOFT: *Microsoft Azure Spatial Anchors API.* <https://docs.microsoft.com/en-us/dotnet/api/microsoft.azure.spatialanchors>, Abruf: 03.01.20
- [Micc] MICROSOFT AZURE: *App Service – Preise.* <https://azure.microsoft.com/de-de/pricing/details/app-service/windows/>, Abruf: 21.02.2020
- [Micd] MICROSOFT AZURE: *Azure Spatial Anchors overview.* <https://docs.microsoft.com/en-us/azure/spatial-anchors/overview>, Abruf: 12.02.2020
- [Mice] MICROSOFT AZURE: *Azure Spatial Anchors Samples.* <https://github.com/Azure/azure-spatial-anchors-samples>, Abruf: 27.01.2020
- [Micf] MICROSOFT AZURE: *Quickstart: Create a Unity Android app with Azure Spatial Anchors.* <https://docs.microsoft.com/en-us/azure/spatial-anchors/quickstarts/get-started-unity-android>, Abruf: 22.10.2019
- [Micg] MICROSOFT AZURE: *Tutorial: Sitzungs- und geräteübergreifendes Freigeben von Azure Spatial Anchors.* <https://docs.microsoft.com/de-de/azure/spatial-anchors/tutorials/tutorial-share-anchors-across-devices>, Abruf: 10.01.2020
- [Mich] MICROSOFT AZURE: *Was ist Cloud-Speicher?* <https://azure.microsoft.com/de-de/overview/what-is-cloud-storage/>, Abruf: 03.02.2020
- [MLB04] MÖHRING, Mathias ; LESSIG, Christian ; BIMBER, Oliver: *Video See-Through AR on Consumer Cell-Phones.* Paper, 2004
- [MTUK94] MILGRAM, Paul ; TAKEMURA, Haruo ; UTSUMI, Akira ; KISHINO, Fumio: *Augmented reality: a class of displays on the reality-virtuality continuum.* Paper, Proc. SPIE 2351, Telemanipulator and Telepresence Technologies, 1994
- [NLZ] NERURKAR, Esha ; LYNEN, Simon ; ZHAO, Sheng: *SYSTEM AND METHOD FOR CONCURRENT ODOMETRY AND MAPPING.* <https://patentimages.storage.googleapis.com/4d/af/08/b9351479700bcf/US20170336511A1.pdf>

- [PS] PATON-SIMPSON, Grant: *Statistics Open For All.* <https://www.sofastatistics.com/home.php>, Abruf: 23.01.2020
- [SCHB01] STETTEN, G. ; CHIB, V. ; HILDEBRAND, D. ; BURSEE, J.: *Real time tomographic reflection: Phantoms for calibration and biopsy.* Paper, 2001
- [SSK<sup>+</sup>19] SPECIALE, Pablo ; SCHONBERGER, Johannes L. ; KANG, Sing B. ; SINHA, Sudipta N. ; POLLEFEYS, Marc: *Privacy Preserving Image-Based Localization.* Paper, The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019
- [Sut68] SUTHERLAND, Ivan E.: *A head-mounted three dimensional display.* Paper, 1968
- [Unia] UNITY TECHNOLOGIES: *About AR Foundation.* <https://docs.unity3d.com/Packages/com.unity.xr.arfoundation\spacefactor\@m{}2.2/manual/index.html>, Abruf: 09.12.2019
- [Unib] UNITY TECHNOLOGIES: *AR Foundation.* <https://unity.com/de/unity/features/arfoundation>, Abruf: 09.12.2019
- [WON16] WANG, X. ; ONG, S. K. ; NEE, A. Y. C.: *A comprehensive survey of augmented reality assembly research.* Paper, 2016



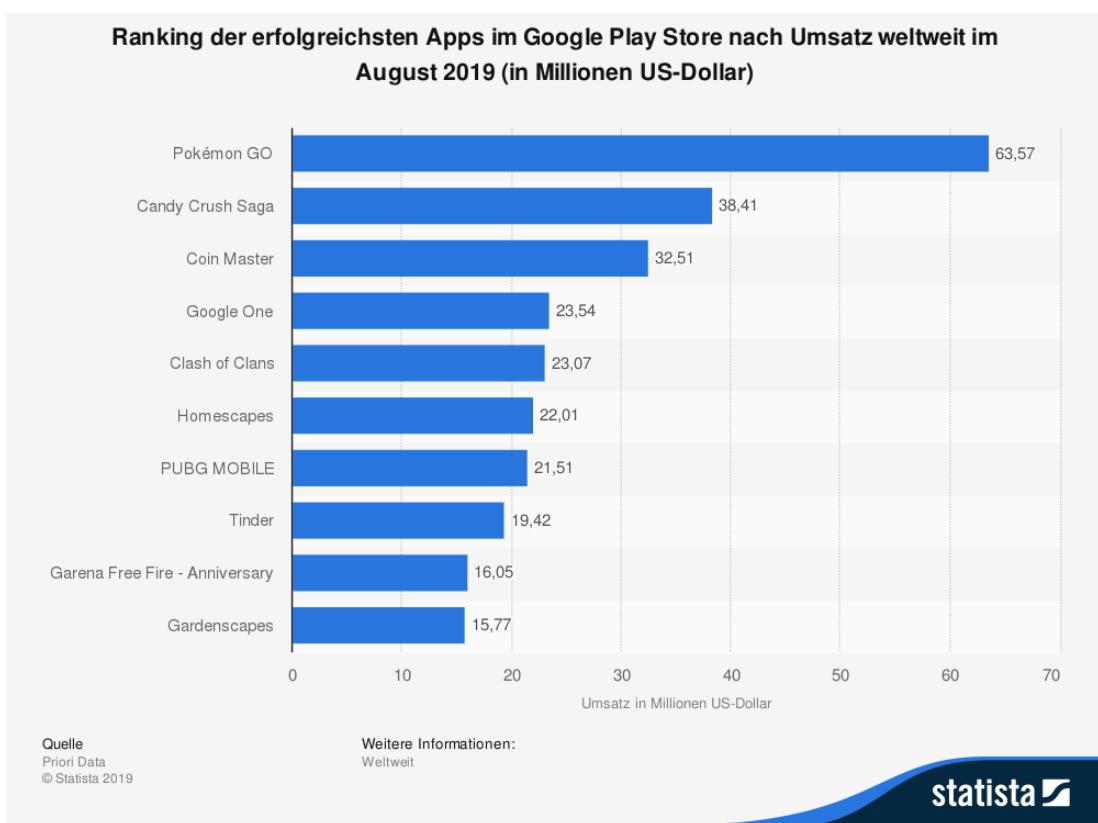
## **A. Inhalt des beiliegenden Datenträgers**

- Abbildungen
- Tabellen
- JSON Dateien
- Unity Projekt
- Masterarbeit in digitaler Form als PDF
- Anwendungen als APK



## B. Statistiken

Abbildung 12.: Statistik zu PokemonGo von Statista [Dat19].





## C. Klassenstruktur

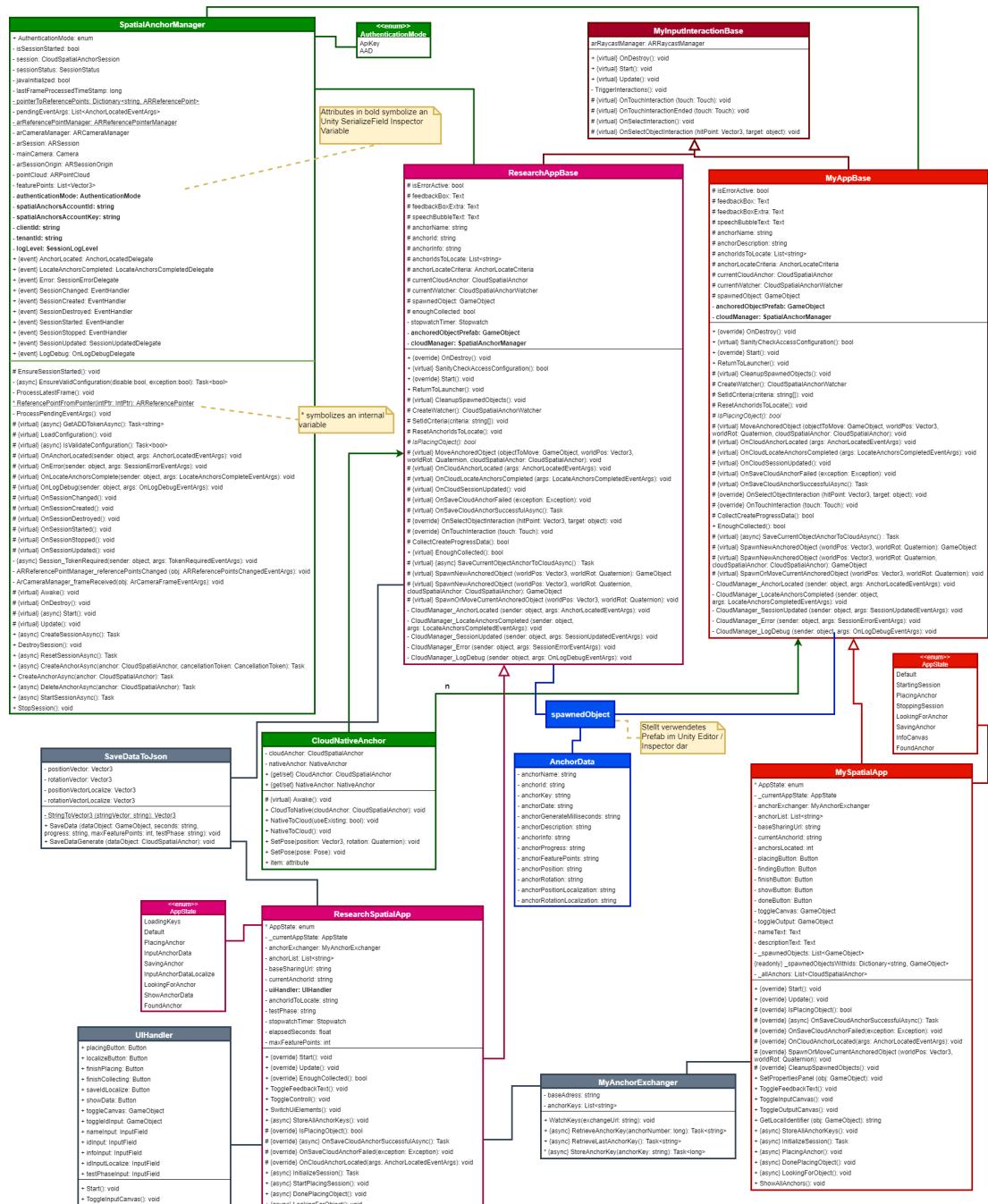


Abbildung 13.: Klassendiagramm Prototyp und Research-Anwendung.

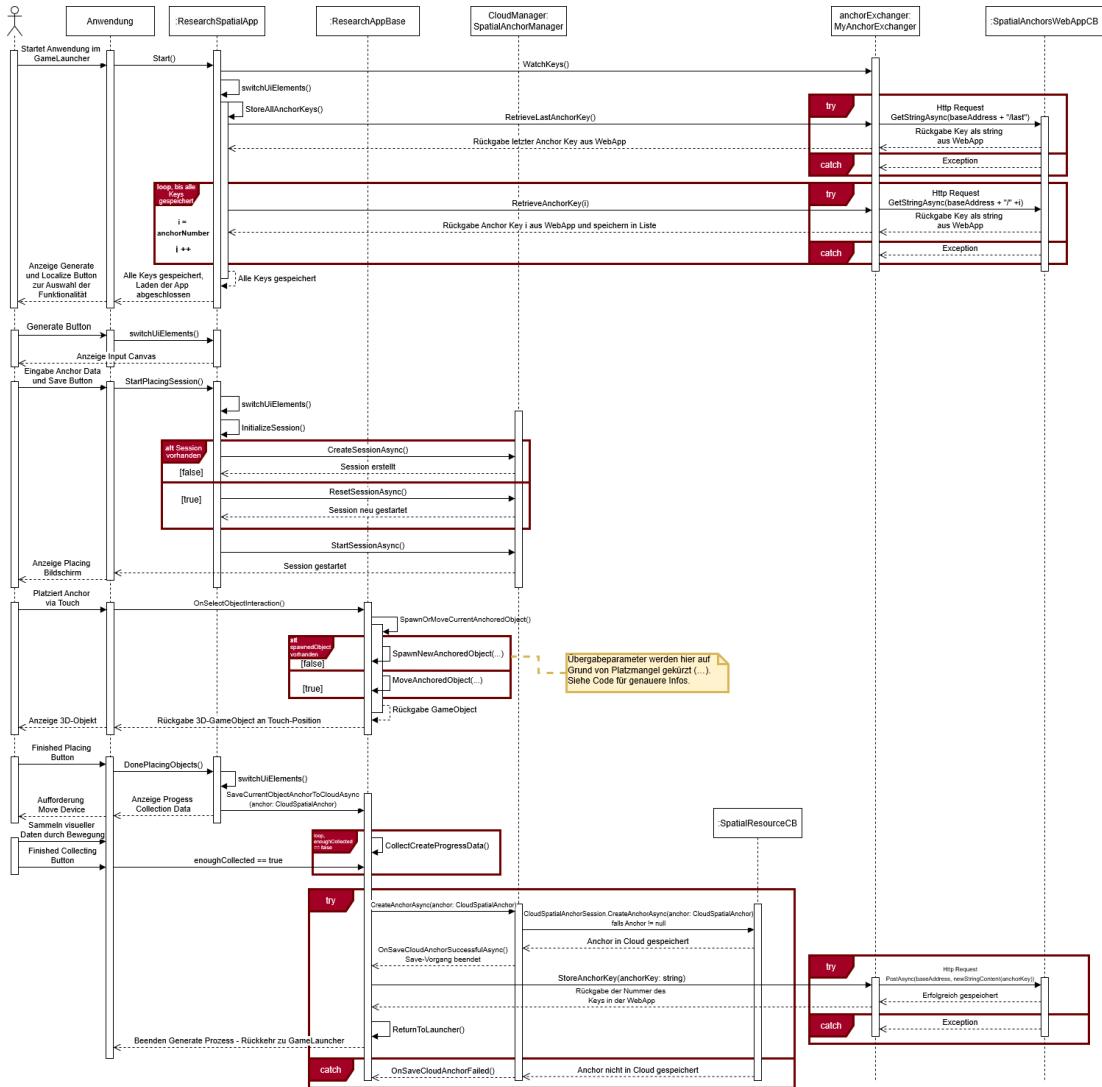


Abbildung 14.: Sequenzdiagramm für die Generierung eines Anchors in der Research-Anwendung.

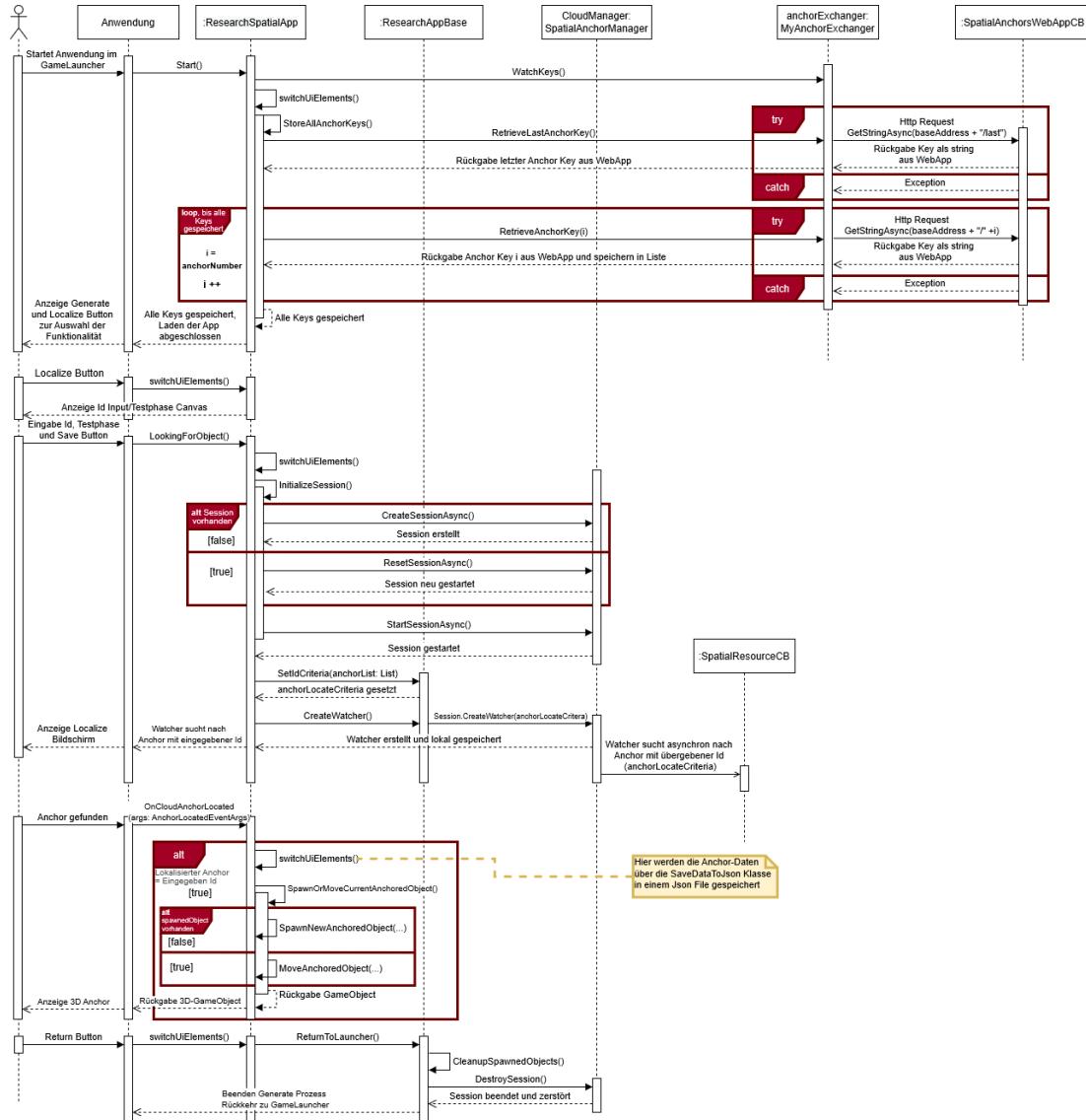


Abbildung 15.: Sequenzdiagramm für die Lokalisierung eines Anchors in der Research-Anwendung.

## D. Screenshots

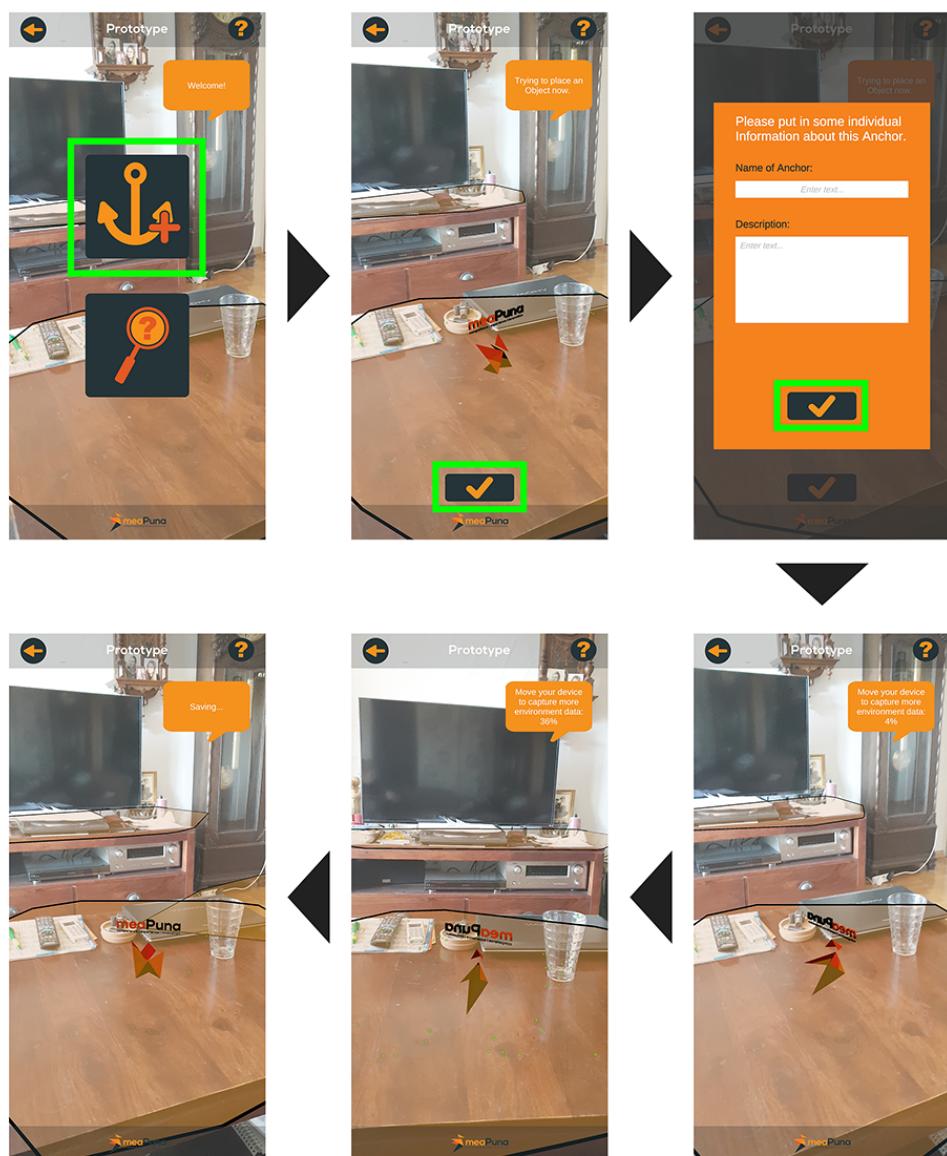


Abbildung 16.: Flowchart Prototyp bei der Generierung eines Anchors. Buttons für Interaktion sind grün umrandet.

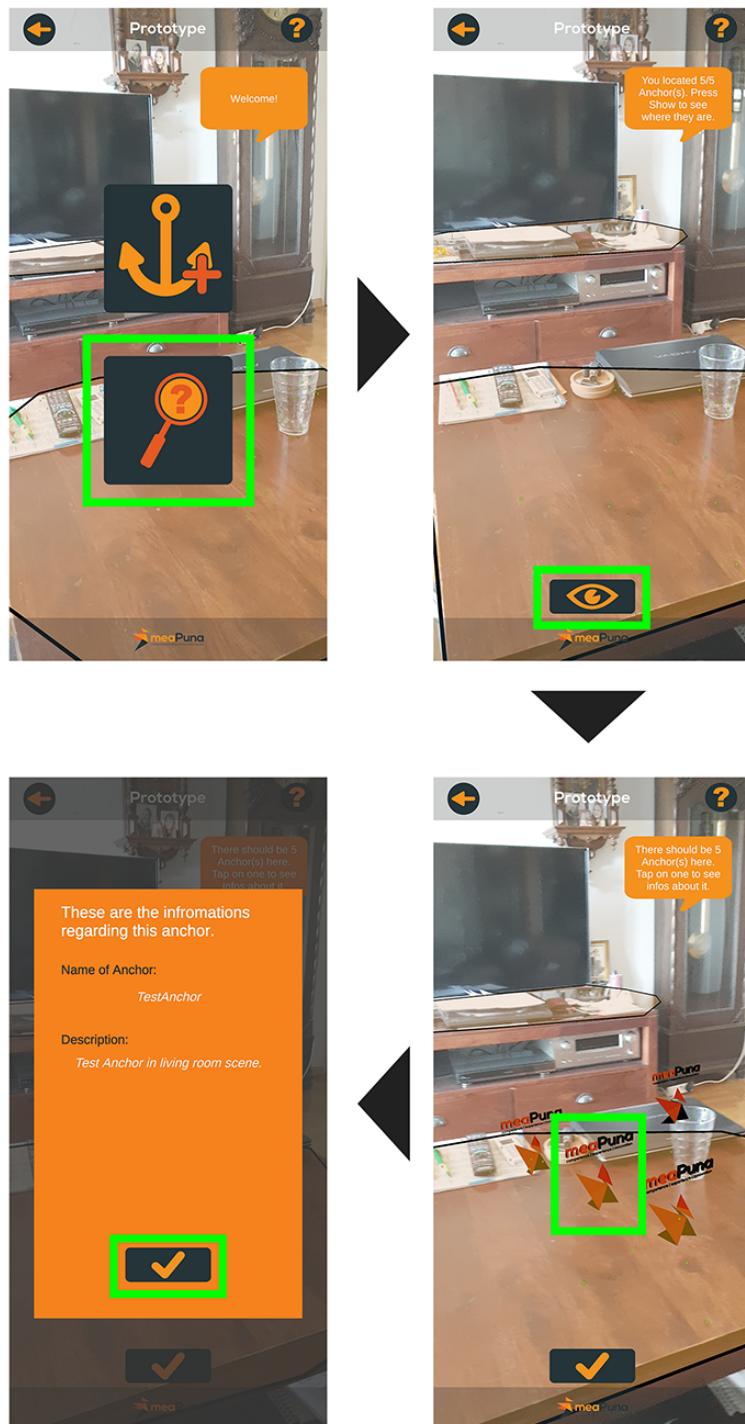


Abbildung 17.: Flowchart Prototyp bei der Lokalisierung eines Anchors. Buttons für Interaktion sind grün umrandet.

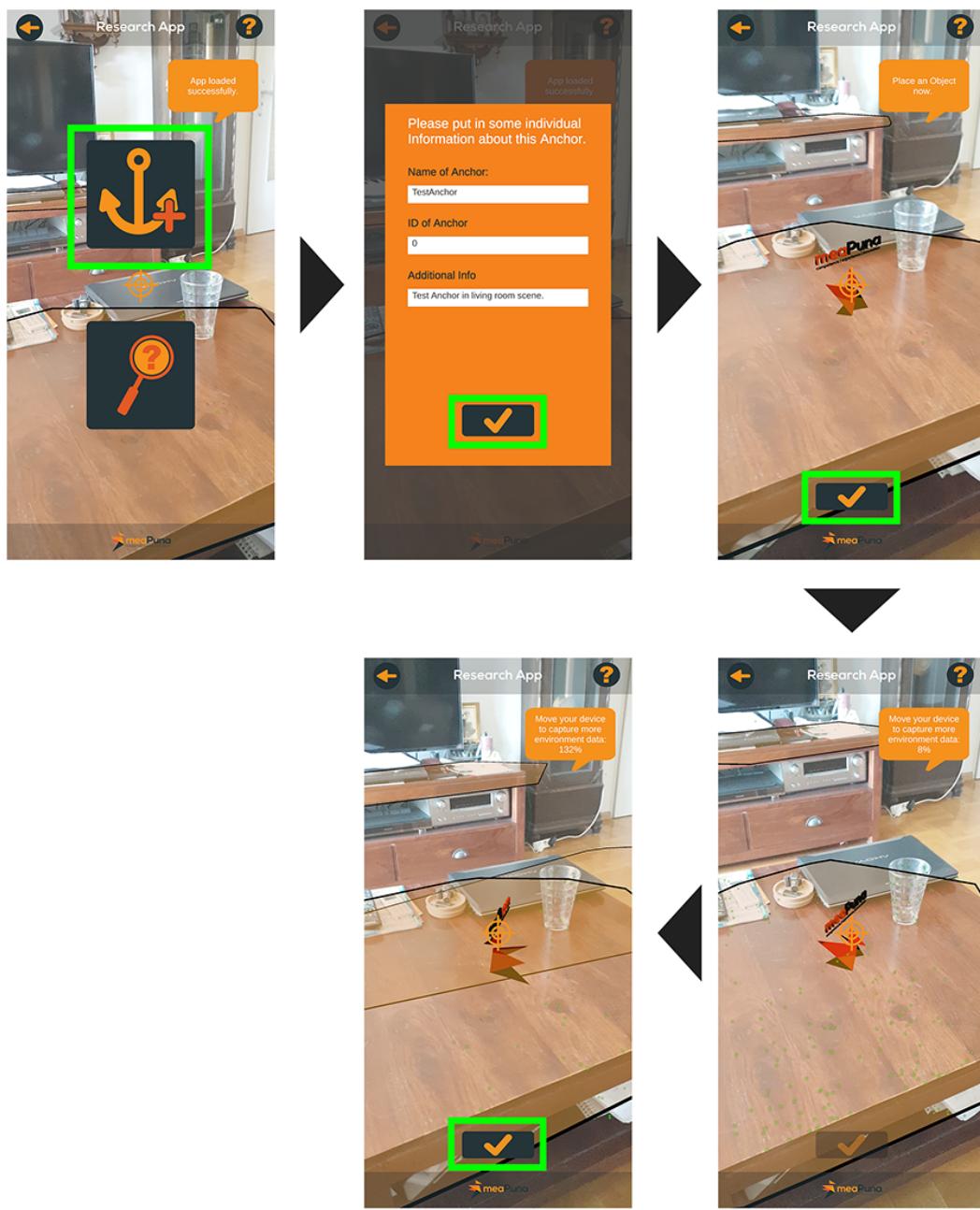


Abbildung 18.: Flowchart Research-Anwendung bei der Generierung eines Anchors.  
Buttons für Interaktion sind grün umrandet.

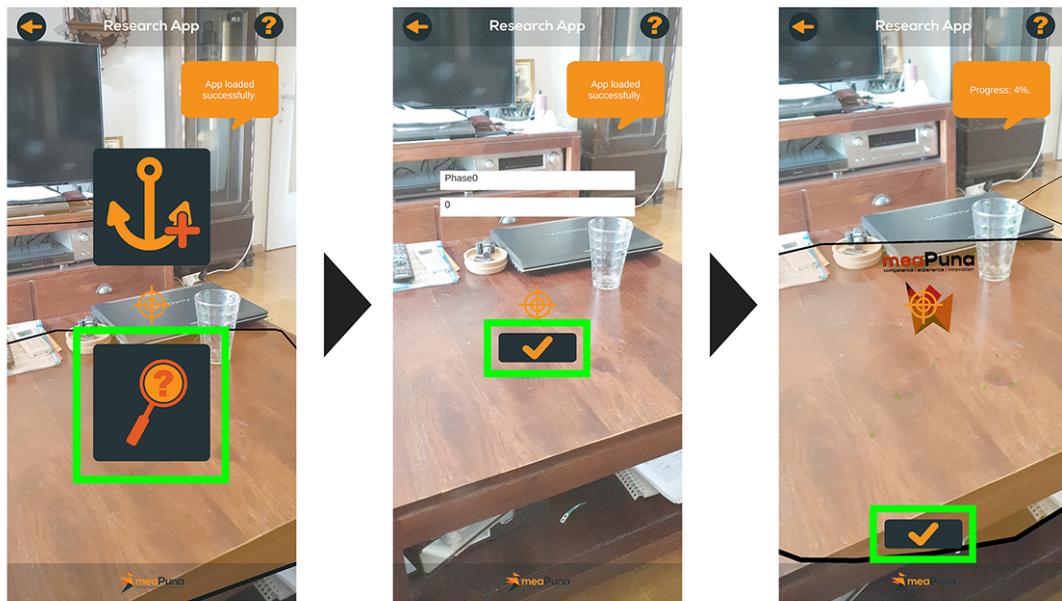


Abbildung 19.: Flowchart Research-Anwendung bei der Lokalisierung eines Anchors.  
Buttons für Interaktion sind grün umrandet.

## E. Statistische Diagramme

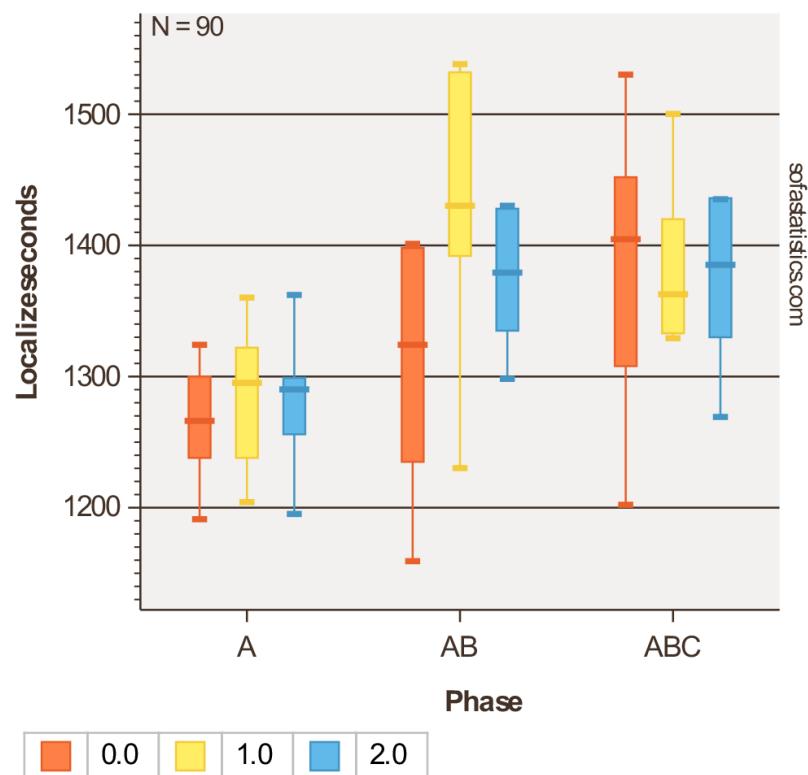


Abbildung 20.: Boxplot-Darstellung der Millisekunden für die Lokalisierung. Ausreißer werden ausgeschlossen. Erstellt mit SOFA [PS].

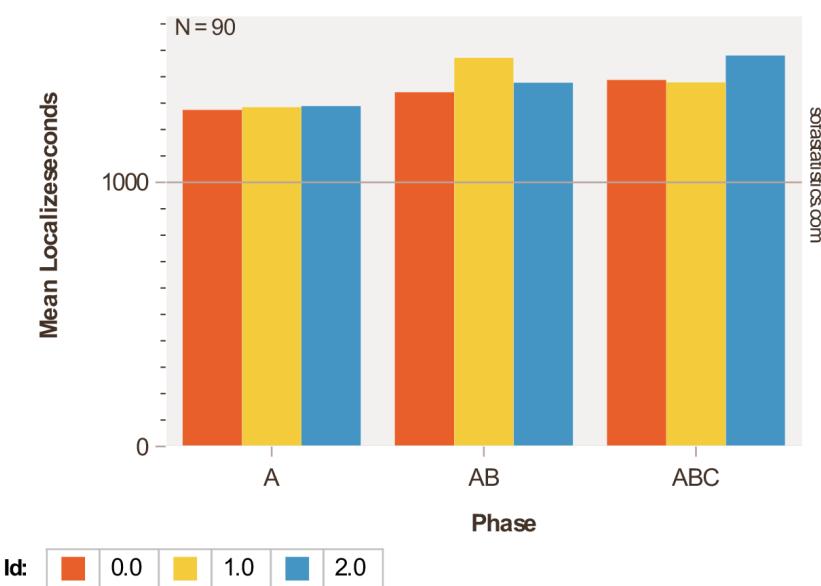


Abbildung 21.: Diagramm zum Vergleich der Mittelwerte aller Anchor bei jeder Phase.  
Erstellt mit SOFA [PS].

## F. Tabellen

**Tabelle: Experiment 2**

ID	Phase	Gruppe	GenerateMSeconds	GenerateProgress	GenerateFeaturePoints	TestPhase	LocalizeMSeconds	LocalizeProgress	LocalizeFeaturePoints	
000	ABC	1	32,781.00	164.25%		107.00	ABC10	1,406.00	4.04%	18.00
000	ABC	1	32,781.00	164.25%		107.00	ABC9	1,203.00	4.21%	8.00
000	ABC	1	32,781.00	164.25%		107.00	ABC8	1,308.00	4.25%	0.00
000	ABC	1	32,781.00	164.25%		107.00	ABC7	1,452.00	4.22%	0.00
000	ABC	1	32,781.00	164.25%		107.00	ABC6	1,531.00	4.21%	3.00
000	ABC	1	32,781.00	164.25%		107.00	ABC5	1,462.00	4.12%	0.00
000	ABC	1	32,781.00	164.25%		107.00	ABC4	1,405.00	4.01%	0.00
000	ABC	1	32,781.00	164.25%		107.00	ABC3	1,397.00	4.17%	0.00
000	ABC	1	32,781.00	164.25%		107.00	ABC2	1,297.00	4.07%	0.00
000	ABC	1	32,781.00	164.25%		107.00	ABC1	1,424.00	4.24%	0.00
000	AB	1	32,781.00	164.25%		107.00	AB10	1,398.00	4.07%	0.00
000	AB	1	32,781.00	164.25%		107.00	AB9	1,338.00	4.09%	0.00
000	AB	1	32,781.00	164.25%		107.00	AB8	1,269.00	4.19%	0.00
000	AB	1	32,781.00	164.25%		107.00	AB7	1,160.00	4.25%	0.00
000	AB	1	32,781.00	164.25%		107.00	AB6	1,339.00	4.15%	0.00
000	AB	1	32,781.00	164.25%		107.00	AB5	1,170.00	4.00%	0.00
000	AB	1	32,781.00	164.25%		107.00	AB4	1,312.00	4.24%	53.00
000	AB	1	32,781.00	164.25%		107.00	AB3	1,235.00	4.17%	0.00
000	AB	1	32,781.00	164.25%		107.00	AB2	1,795.00	4.25%	0.00
000	AB	1	32,781.00	164.25%		107.00	AB1	1,402.00	4.24%	0.00
000	A	1	32,781.00	164.25%		107.00	A10	1,325.00	4.02%	0.00
000	A	1	32,781.00	164.25%		107.00	A9	1,259.00	4.05%	0.00
000	A	1	32,781.00	164.25%		107.00	A8	1,275.00	4.04%	0.00
000	A	1	32,781.00	164.25%		107.00	A7	1,300.00	4.12%	0.00
000	A	1	32,781.00	164.25%		107.00	A6	1,297.00	4.09%	0.00
000	A	1	32,781.00	164.25%		107.00	A5	1,405.00	4.21%	0.00
000	A	1	32,781.00	164.25%		107.00	A4	1,192.00	4.10%	0.00
000	A	1	32,781.00	164.25%		107.00	A3	1,205.00	4.17%	0.00
000	A	1	32,781.00	164.25%		107.00	A2	1,238.00	4.01%	0.00
000	A	1	32,781.00	164.25%		107.00	A1	1,252.00	4.04%	0.00
001	ABC	2	30,903.00	172.16%		178.00	ABC10	1,442.00	4.17%	0.00
001	ABC	2	30,903.00	172.16%		178.00	ABC9	1,365.00	4.02%	2.00
001	ABC	2	30,903.00	172.16%		178.00	ABC8	1,333.00	4.20%	0.00
001	ABC	2	30,903.00	172.16%		178.00	ABC7	1,362.00	4.08%	0.00
001	ABC	2	30,903.00	172.16%		178.00	ABC6	1,367.00	4.02%	28.00
001	ABC	2	30,903.00	172.16%		178.00	ABC5	1,330.00	4.07%	0.00
001	ABC	2	30,903.00	172.16%		178.00	ABC4	1,420.00	4.10%	0.00
001	ABC	2	30,903.00	172.16%		178.00	ABC3	1,501.00	4.11%	1.00
001	ABC	2	30,903.00	172.16%		178.00	ABC2	1,332.00	4.24%	0.00
001	ABC	2	30,903.00	172.16%		178.00	ABC1	1,336.00	4.09%	12.00
001	AB	2	30,903.00	172.16%		178.00	AB10	1,478.00	4.10%	0.00
001	AB	2	30,903.00	172.16%		178.00	AB9	1,420.00	4.06%	0.00
001	AB	2	30,903.00	172.16%		178.00	AB8	1,382.00	4.05%	0.00
001	AB	2	30,903.00	172.16%		178.00	AB7	1,392.00	4.11%	0.00
001	AB	2	30,903.00	172.16%		178.00	AB6	1,435.00	4.19%	0.00
001	AB	2	30,903.00	172.16%		178.00	AB5	1,427.00	4.20%	0.00
001	AB	2	30,903.00	172.16%		178.00	AB4	1,882.00	4.25%	0.00
001	AB	2	30,903.00	172.16%		178.00	AB3	1,231.00	4.18%	0.00
001	AB	2	30,903.00	172.16%		178.00	AB2	1,539.00	4.03%	0.00
001	AB	2	30,903.00	172.16%		178.00	AB1	1,532.00	4.02%	0.00
001	A	2	30,903.00	172.16%		178.00	A10	1,295.00	4.19%	0.00
001	A	2	30,903.00	172.16%		178.00	A9	1,297.00	4.09%	0.00
001	A	2	30,903.00	172.16%		178.00	A8	1,205.00	4.14%	0.00
001	A	2	30,903.00	172.16%		178.00	A7	1,238.00	4.12%	0.00
001	A	2	30,903.00	172.16%		178.00	A6	1,304.00	4.12%	0.00
001	A	2	30,903.00	172.16%		178.00	A5	1,349.00	4.19%	0.00
001	A	2	30,903.00	172.16%		178.00	A4	1,322.00	4.19%	0.00
001	A	2	30,903.00	172.16%		178.00	A3	1,361.00	4.06%	0.00
001	A	2	30,903.00	172.16%		178.00	A2	1,228.00	4.06%	0.00
001	A	2	30,903.00	172.16%		178.00	A1	1,252.00	4.04%	0.00
002	ABC	3	34,852.00	164.06%		223.00	ABC10	1,403.00	4.08%	0.00
002	ABC	3	34,852.00	164.06%		223.00	ABC9	1,369.00	4.17%	1.00
002	ABC	3	34,852.00	164.06%		223.00	ABC8	1,410.00	4.02%	8.00
002	ABC	3	34,852.00	164.06%		223.00	ABC7	1,669.00	4.05%	0.00
002	ABC	3	34,852.00	164.06%		223.00	ABC6	2,281.00	4.17%	0.00
002	ABC	3	34,852.00	164.06%		223.00	ABC5	1,270.00	4.23%	0.00

002	ABC	3	34,852.00	164.06%	223.00	ABC4	1,338.00	4.19%	46.00
002	ABC	3	34,852.00	164.06%	223.00	ABC3	1,436.00	4.03%	0.00
002	ABC	3	34,852.00	164.06%	223.00	ABC2	1,302.00	4.10%	0.00
002	ABC	3	34,852.00	164.06%	223.00	ABC1	1,330.00	4.22%	7.00
002	AB	3	34,852.00	164.06%	223.00	AB10	1,362.00	4.00%	0.00
002	AB	3	34,852.00	164.06%	223.00	AB9	1,363.00	4.22%	0.00
002	AB	3	34,852.00	164.06%	223.00	AB8	1,398.00	4.12%	0.00
002	AB	3	34,852.00	164.06%	223.00	AB7	1,335.00	4.18%	0.00
002	AB	3	34,852.00	164.06%	223.00	AB6	1,428.00	4.14%	0.00
002	AB	3	34,852.00	164.06%	223.00	AB5	1,430.00	4.08%	0.00
002	AB	3	34,852.00	164.06%	223.00	AB4	1,299.00	4.01%	0.00
002	AB	3	34,852.00	164.06%	223.00	AB3	1,335.00	4.18%	0.00
002	AB	3	34,852.00	164.06%	223.00	AB2	1,431.00	4.24%	0.00
002	AB	3	34,852.00	164.06%	223.00	AB1	1,397.00	4.15%	0.00
002	A	3	34,852.00	164.06%	223.00	A10	1,363.00	4.20%	70.00
002	A	3	34,852.00	164.06%	223.00	A9	1,196.00	4.25%	0.00
002	A	3	34,852.00	164.06%	223.00	A8	1,226.00	4.05%	70.00
002	A	3	34,852.00	164.06%	223.00	A7	1,297.00	4.20%	0.00
002	A	3	34,852.00	164.06%	223.00	A6	1,278.00	4.05%	0.00
002	A	3	34,852.00	164.06%	223.00	A5	1,294.00	4.18%	0.00
002	A	3	34,852.00	164.06%	223.00	A4	1,256.00	4.24%	0.00
002	A	3	34,852.00	164.06%	223.00	A3	1,299.00	4.13%	1.00
002	A	3	34,852.00	164.06%	223.00	A2	1,288.00	4.17%	0.00
002	A	3	34,852.00	164.06%	223.00	A1	1,392.00	4.00%	0.00

**Tabelle: Experiment 1**

ID	TestPhase	Part	Gruppe	Phase	GenerateMSeconds	GenerateProgress	GenerateFeaturePoints	LocalizeMSeconds	LocalizeProgress	LocalizeFeaturePoints
000	ABC1	1	1	ABC	36,111.00	160.03%	162.00	1,706.00	4.12%	44.00
000	ABC2	1	1	ABC	36,111.00	160.03%	162.00	1,440.00	4.23%	288.00
000	ABC3	1	1	ABC	36,111.00	160.03%	162.00	1,399.00	4.01%	0.00
000	AB1	1	1	AB	36,111.00	160.03%	162.00	1,527.00	4.20%	228.00
000	AB2	1	1	AB	36,111.00	160.03%	162.00	1,545.00	4.05%	256.00
000	AB3	1	1	AB	36,111.00	160.03%	162.00	1,494.00	4.01%	240.00
000	A1	1	1	A	36,111.00	160.03%	162.00	1,405.00	4.04%	151.00
000	A2	1	1	A	36,111.00	160.03%	162.00	1,512.00	4.04%	146.00
000	A3	1	1	A	36,111.00	160.03%	162.00	1,345.00	4.02%	150.00
001	ABC1	1	1	ABC	38,262.00	176.01%	227.00	1,368.00	4.09%	0.00
001	ABC2	1	1	ABC	38,262.00	176.01%	227.00	1,401.00	4.23%	1.00
001	ABC3	1	1	ABC	38,262.00	176.01%	227.00	1,497.00	4.16%	7.00
001	AB1	1	1	AB	38,262.00	176.01%	227.00	1,431.00	4.17%	251.00
001	AB2	1	1	AB	38,262.00	176.01%	227.00	1,732.00	4.10%	245.00
001	AB3	1	1	AB	38,262.00	176.01%	227.00	1,429.00	4.18%	0.00
001	A1	1	1	A	38,262.00	176.01%	227.00	1,354.00	4.16%	158.00
001	A2	1	1	A	38,262.00	176.01%	227.00	1,352.00	4.20%	160.00
001	A3	1	1	A	38,262.00	176.01%	227.00	1,346.00	4.10%	150.00
002	ABC1	1	1	ABC	37,317.00	176.19%	239.00	1,297.00	4.05%	2.00
002	ABC2	1	1	ABC	37,317.00	176.19%	239.00	1,393.00	4.23%	0.00
002	ABC3	1	1	ABC	37,317.00	176.19%	239.00	1,405.00	4.16%	299.00
002	AB1	1	1	AB	37,317.00	176.19%	239.00	1,350.00	4.17%	242.00
002	AB2	1	1	AB	37,317.00	176.19%	239.00	1,631.00	4.18%	252.00
002	AB3	1	1	AB	37,317.00	176.19%	239.00	1,364.00	4.21%	259.00
002	A1	1	1	A	37,317.00	176.19%	239.00	1,326.00	4.16%	161.00
002	A2	1	1	A	37,317.00	176.19%	239.00	1,297.00	4.04%	146.00
002	A3	1	1	A	37,317.00	176.19%	239.00	1,377.00	4.09%	151.00
000	A3DUNKEL	2	1	A	36,111.00	160.03%	162.00	1,313.00	4.05%	111.00
000	A2DUNKEL	2	1	A	36,111.00	160.03%	162.00	1,479.00	4.23%	121.00
000	A1DUNKEL	2	1	A	36,111.00	160.03%	162.00	1,302.00	4.05%	109.00
000	AB3DUNKEL	2	1	AB	36,111.00	160.03%	162.00	1,497.00	4.25%	198.00
000	AB2DUNKEL	2	1	AB	36,111.00	160.03%	162.00	1,564.00	4.22%	22.00
000	AB1DUNKEL	2	1	AB	36,111.00	160.03%	162.00	1,509.00	4.17%	0.00
000	ABC3DUNKE	2	1	ABC	36,111.00	160.03%	162.00	1,456.00	4.13%	0.00
000	ABC2DUNKE	2	1	ABC	36,111.00	160.03%	162.00	1,395.00	4.01%	234.00
000	ABC1DUNKE	2	1	ABC	36,111.00	160.03%	162.00	1,494.00	4.16%	0.00
001	A3DUNKEL	2	1	A	38,262.00	176.01%	227.00	1,405.00	4.23%	0.00
001	A2DUNKEL	2	1	A	38,262.00	176.01%	227.00	1,290.00	4.24%	122.00
001	A1DUNKEL	2	1	A	38,262.00	176.01%	227.00	1,264.00	4.02%	116.00
001	AB3DUNKEL	2	1	AB	38,262.00	176.01%	227.00	1,567.00	4.25%	211.00
001	AB2DUNKEL	2	1	AB	38,262.00	176.01%	227.00	1,416.00	4.14%	22.00
001	AB1DUNKEL	2	1	AB	38,262.00	176.01%	227.00	1,421.00	4.08%	0.00
001	ABC3DUNKE	2	1	ABC	38,262.00	176.01%	227.00	1,367.00	4.23%	1.00
001	ABC2DUNKE	2	1	ABC	38,262.00	176.01%	227.00	1,410.00	4.05%	0.00
001	ABC1DUNKE	2	1	ABC	38,262.00	176.01%	227.00	1,373.00	4.19%	0.00
002	A3DUNKEL	2	1	A	37,317.00	176.19%	239.00	1,234.00	4.06%	121.00
002	A2DUNKEL	2	1	A	37,317.00	176.19%	239.00	1,294.00	4.01%	126.00
002	A1DUNKEL	2	1	A	37,317.00	176.19%	239.00	1,346.00	4.20%	126.00
002	AB3DUNKEL	2	1	AB	37,317.00	176.19%	239.00	1,414.00	4.07%	1.00
002	AB2DUNKEL	2	1	AB	37,317.00	176.19%	239.00	1,389.00	4.22%	76.00
002	AB1DUNKEL	2	1	AB	37,317.00	176.19%	239.00	1,389.00	4.07%	117.00
002	ABC3DUNKE	2	1	ABC	37,317.00	176.19%	239.00	1,314.00	4.24%	246.00
002	ABC2DUNKE	2	1	ABC	37,317.00	176.19%	239.00	1,500.00	4.06%	0.00
002	ABC1DUNKE	2	1	ABC	37,317.00	176.19%	239.00	1,420.00	4.03%	241.00
003	ABC1HELL	3	2	ABC	37,205.00	168.01%	92.00	1,511.00	4.16%	16.00
003	ABC2HELL	3	2	ABC	37,205.00	168.01%	92.00	1,525.00	4.19%	0.00
003	ABC3HELL	3	2	ABC	37,205.00	168.01%	92.00	1,397.00	4.18%	271.00
003	AB1HELL	3	2	AB	37,205.00	168.01%	92.00	1,366.00	4.21%	244.00
003	AB2HELL	3	2	AB	37,205.00	168.01%	92.00	1,331.00	4.25%	247.00
003	AB3HELL	3	2	AB	37,205.00	168.01%	92.00	1,366.00	4.14%	243.00
003	A1HELL	3	2	A	37,205.00	168.01%	92.00	1,526.00	4.04%	149.00
003	A2HELL	3	2	A	37,205.00	168.01%	92.00	1,572.00	4.25%	140.00
003	A3HELL	3	2	A	37,205.00	168.01%	92.00	1,345.00	4.13%	136.00
004	ABC1HELL	3	2	ABC	33,626.00	164.05%	146.00	1,396.00	4.25%	163.00
004	ABC2HELL	3	2	ABC	33,626.00	164.05%	146.00	1,430.00	4.18%	0.00
004	ABC3HELL	3	2	ABC	33,626.00	164.05%	146.00	1,401.00	4.22%	260.00
004	AB1HELL	3	2	AB	33,626.00	164.05%	146.00	1,367.00	4.07%	243.00
004	AB2HELL	3	2	AB	33,626.00	164.05%	146.00	1,442.00	4.05%	245.00
004	AB3HELL	3	2	AB	33,626.00	164.05%	146.00	1,425.00	4.15%	228.00
004	A1HELL	3	2	A	33,626.00	164.05%	146.00	1,481.00	4.01%	149.00
004	A2HELL	3	2	A	33,626.00	164.05%	146.00	1,320.00	4.24%	138.00
004	A3HELL	3	2	A	33,626.00	164.05%	146.00	1,298.00	4.13%	144.00

005	ABC1HELL	3	2	ABC	35,291.00	164.17%	178.00	1,622.00	4.16%	143.00
005	ABC2HELL	3	2	ABC	35,291.00	164.17%	178.00	1,399.00	4.21%	1.00
005	ABC3HELL	3	2	ABC	35,291.00	164.17%	178.00	1,749.00	4.12%	0.00
005	AB1HELL	3	2	AB	35,291.00	164.17%	178.00	1,520.00	4.16%	233.00
005	AB2HELL	3	2	AB	35,291.00	164.17%	178.00	1,432.00	4.05%	0.00
005	AB3HELL	3	2	AB	35,291.00	164.17%	178.00	1,451.00	4.06%	242.00
005	A1HELL	3	2	A	35,291.00	164.17%	178.00	1,333.00	4.07%	144.00
005	A2HELL	3	2	A	35,291.00	164.17%	178.00	1,520.00	4.18%	136.00
005	A3HELL	3	2	A	35,291.00	164.17%	178.00	1,476.00	4.04%	138.00
003	ABC1DUNKE	4	2	ABC	37,205.00	168.01%	92.00	1,863.00	4.07%	0.00
003	ABC2DUNKE	4	2	ABC	37,205.00	168.01%	92.00	1,605.00	4.12%	211.00
003	ABC3DUNKE	4	2	ABC	37,205.00	168.01%	92.00	1,516.00	4.19%	213.00
003	AB1DUNKEL	4	2	AB	37,205.00	168.01%	92.00	1,433.00	4.18%	177.00
003	AB2DUNKEL	4	2	AB	37,205.00	168.01%	92.00	1,496.00	4.08%	193.00
003	AB3DUNKEL	4	2	AB	37,205.00	168.01%	92.00	1,465.00	4.22%	178.00
003	A1DUNKEL	4	2	A	37,205.00	168.01%	92.00	1,397.00	4.21%	104.00
003	A2DUNKEL	4	2	A	37,205.00	168.01%	92.00	1,319.00	4.22%	114.00
003	A3DUNKEL	4	2	A	37,205.00	168.01%	92.00	1,422.00	4.02%	106.00
004	ABC1DUNKE	4	2	ABC	33,626.00	164.05%	146.00	1,558.00	4.04%	0.00
004	ABC2DUNKE	4	2	ABC	33,626.00	164.05%	146.00	1,368.00	4.15%	0.00
004	ABC3DUNKE	4	2	ABC	33,626.00	164.05%	146.00	1,437.00	4.20%	227.00
004	AB1DUNKEL	4	2	AB	33,626.00	164.05%	146.00	1,488.00	4.22%	10.00
004	AB2DUNKEL	4	2	AB	33,626.00	164.05%	146.00	1,340.00	4.10%	5.00
004	AB3DUNKEL	4	2	AB	33,626.00	164.05%	146.00	1,398.00	4.22%	192.00
004	A1DUNKEL	4	2	A	33,626.00	164.05%	146.00	1,289.00	4.23%	0.00
004	A2DUNKEL	4	2	A	33,626.00	164.05%	146.00	1,296.00	4.20%	118.00
004	A3DUNKEL	4	2	A	33,626.00	164.05%	146.00	1,415.00	4.17%	2.00
005	ABC1DUNKE	4	2	ABC	35,291.00	164.17%	178.00	1,633.00	4.15%	211.00
005	ABC2DUNKE	4	2	ABC	35,291.00	164.17%	178.00	1,499.00	4.01%	225.00
005	ABC3DUNKE	4	2	ABC	35,291.00	164.17%	178.00	1,344.00	4.10%	0.00
005	AB1DUNKEL	4	2	AB	35,291.00	164.17%	178.00	1,372.00	4.17%	0.00
005	AB2DUNKEL	4	2	AB	35,291.00	164.17%	178.00	1,402.00	4.19%	191.00
005	AB3DUNKEL	4	2	AB	35,291.00	164.17%	178.00	1,444.00	4.23%	6.00
005	A1DUNKEL	4	2	A	35,291.00	164.17%	178.00	1,355.00	4.07%	109.00
005	A2DUNKEL	4	2	A	35,291.00	164.17%	178.00	1,392.00	4.09%	106.00
005	A3DUNKEL	4	2	A	35,291.00	164.17%	178.00	1,512.00	4.00%	120.00
006	ABC1HELL	5	3	ABC	36,829.00	160.12%	293.00	1,468.00	4.03%	0.00
006	ABC1HELL	5	3	ABC	36,829.00	160.12%	293.00	1,491.00	4.03%	446.00
006	ABC2HELL	5	3	ABC	36,829.00	160.12%	293.00	1,582.00	4.17%	0.00
006	AB1HELL	5	3	AB	36,829.00	160.12%	293.00	1,435.00	4.15%	210.00
006	AB2HELL	5	3	AB	36,829.00	160.12%	293.00	1,579.00	4.05%	290.00
006	AB3HELL	5	3	AB	36,829.00	160.12%	293.00	1,532.00	4.25%	316.00
006	Y1	5	3	Y	36,829.00	160.12%	293.00	1,632.00	12.11%	139.00
006	Y2	5	3	Y	36,829.00	160.12%	293.00	1,448.00	12.19%	270.00
006	Y3	5	3	Y	36,829.00	160.12%	293.00	1,491.00	4.14%	68.00
006	YZ1	5	3	YZ	36,829.00	160.12%	293.00	1,866.00	12.24%	153.00
007	ABC1HELL	5	3	ABC	38,168.00	164.02%	413.00	1,593.00	4.09%	8.00
007	ABC2HELL	5	3	ABC	38,168.00	164.02%	413.00	1,473.00	4.01%	0.00
007	ABC3HELL	5	3	ABC	38,168.00	164.02%	413.00	1,499.00	4.09%	0.00
007	AB1HELL	5	3	AB	38,168.00	164.02%	413.00	1,434.00	4.17%	340.00
007	AB2HELL	5	3	AB	38,168.00	164.02%	413.00	1,572.00	4.15%	382.00
007	AB3HELL	5	3	AB	38,168.00	164.02%	413.00	1,435.00	4.07%	335.00
007	Y1	5	3	Y	38,168.00	164.02%	413.00	1,595.00	8.07%	78.00
007	Y2	5	3	Y	38,168.00	164.02%	413.00	1,662.00	16.09%	244.00
007	Y3	5	3	Y	38,168.00	164.02%	413.00	1,594.00	4.04%	89.00
007	YZ1	5	3	YZ	38,168.00	164.02%	413.00	2,267.00	20.24%	182.00
008	ABC1HELL	5	3	ABC	36,328.00	164.18%	435.00	1,479.00	4.08%	0.00
008	ABC2HELL	5	3	ABC	36,328.00	164.18%	435.00	1,591.00	4.05%	0.00
008	ABC3HELL	5	3	ABC	36,328.00	164.18%	435.00	1,467.00	4.05%	0.00
008	AB1HELL	5	3	AB	36,328.00	164.18%	435.00	1,543.00	4.17%	321.00
008	AB2HELL	5	3	AB	36,328.00	164.18%	435.00	1,501.00	4.08%	334.00
008	AB3HELL	5	3	AB	36,328.00	164.18%	435.00	1,454.00	4.17%	0.00
008	Y2	5	3	Y	36,328.00	164.18%	435.00	2,163.00	12.25%	240.00
008	Y1	5	3	Y	36,328.00	164.18%	435.00	1,458.00	8.24%	56.00
008	Y3	5	3	Y	36,328.00	164.18%	435.00	1,573.00	8.09%	67.00
008	YZ1	5	3	YZ	36,328.00	164.18%	435.00	5,412.00	32.07%	234.00
006	ABC1DUNKE	6	3	ABC	36,829.00	160.12%	293.00	1,553.00	4.00%	95.00
006	ABC2DUNKE	6	3	ABC	36,829.00	160.12%	293.00	1,635.00	4.16%	0.00
006	ABC3DUNKE	6	3	ABC	36,829.00	160.12%	293.00	1,626.00	4.06%	22.00
006	AB1DUNKEL	6	3	AB	36,829.00	160.12%	293.00	1,578.00	4.00%	12.00
006	AB2DUNKEL	6	3	AB	36,829.00	160.12%	293.00	1,712.00	4.13%	308.00
006	AB3DUNKEL	6	3	AB	36,829.00	160.12%	293.00	1,394.00	4.09%	0.00
006	Y1	6	3	Y	36,829.00	160.12%	293.00	1,818.00	12.20%	244.00

006	Y2	6	3	Y	36,829.00	160.12%	293.00	1,791.00	12.16%	225.00
006	Y3	6	3	Y	36,829.00	160.12%	293.00	1,740.00	12.07%	233.00
006	YZ1	6	3	YZ	36,829.00	160.12%	293.00	5,442.00	32.12%	206.00
007	ABC1DUNKE	6	3	ABC	38,168.00	164.02%	413.00	1,532.00	4.11%	450.00
007	ABC2DUNKE	6	3	ABC	38,168.00	164.02%	413.00	1,802.00	4.04%	461.00
007	ABC3DUNKE	6	3	ABC	38,168.00	164.02%	413.00	1,811.00	4.22%	0.00
007	AB1DUNKEL	6	3	AB	38,168.00	164.02%	413.00	2,316.00	4.02%	0.00
007	AB2DUNKEL	6	3	AB	38,168.00	164.02%	413.00	1,534.00	4.18%	378.00
007	AB3DUNKEL	6	3	AB	38,168.00	164.02%	413.00	1,373.00	4.21%	0.00
007	Y1	6	3	Y	38,168.00	164.02%	413.00	2,031.00	8.07%	213.00
007	Y2	6	3	Y	38,168.00	164.02%	413.00	1,595.00	12.04%	282.00
007	Y3	6	3	Y	38,168.00	164.02%	413.00	1,792.00	16.16%	276.00
007	YZ1	6	3	YZ	38,168.00	164.02%	413.00	2,102.00	16.02%	173.00
008	ABC1DUNKE	6	3	ABC	36,328.00	164.18%	435.00	1,493.00	4.24%	0.00
008	ABC2DUNKE	6	3	ABC	36,328.00	164.18%	435.00	1,434.00	4.09%	0.00
008	ABC3DUNKE	6	3	ABC	36,328.00	164.18%	435.00	1,487.00	4.14%	0.00
008	AB1DUNKEL	6	3	AB	36,328.00	164.18%	435.00	1,532.00	4.00%	378.00
008	AB2DUNKEL	6	3	AB	36,328.00	164.18%	435.00	1,566.00	4.03%	88.00
008	AB3DUNKEL	6	3	AB	36,328.00	164.18%	435.00	1,426.00	4.09%	0.00
008	Y1	6	3	Y	36,328.00	164.18%	435.00	1,665.00	12.19%	314.00
008	Y2	6	3	Y	36,328.00	164.18%	435.00	1,660.00	12.14%	279.00
008	Y3	6	3	Y	36,328.00	164.18%	435.00	1,806.00	12.18%	257.00
008	YZ1	6	3	YZ	36,328.00	164.18%	435.00	4,949.00	32.12%	195.00
009	ABC1HELL	7	4	ABC	37,399.00	184.01%	388.00	1,561.00	4.23%	0.00
009	ABC2HELL	7	4	ABC	37,399.00	184.01%	388.00	1,466.00	4.19%	0.00
009	ABC3HELL	7	4	ABC	37,399.00	184.01%	388.00	1,287.00	4.22%	0.00
009	AB1HELL	7	4	AB	37,399.00	184.01%	388.00	1,441.00	4.06%	0.00
009	AB2HELL	7	4	AB	37,399.00	184.01%	388.00	1,533.00	4.12%	6.00
009	AB3HELL	7	4	AB	37,399.00	184.01%	388.00	1,555.00	4.19%	329.00
009	Y1	7	4	Y	37,399.00	184.01%	388.00	1,864.00	16.17%	263.00
009	Y2	7	4	Y	37,399.00	184.01%	388.00	1,718.00	12.08%	235.00
009	Y3	7	4	Y	37,399.00	184.01%	388.00	1,470.00	8.10%	234.00
009	YZ1	7	4	YZ	37,399.00	184.01%	388.00	2,520.00	16.05%	251.00
010	ABC1HELL	7	4	ABC	35,259.00	164.07%	389.00	1,538.00	4.22%	0.00
010	ABC2HELL	7	4	ABC	35,259.00	164.07%	389.00	1,458.00	4.09%	426.00
010	ABC3HELL	7	4	ABC	35,259.00	164.07%	389.00	1,440.00	4.19%	0.00
010	AB1HELL	7	4	AB	35,259.00	164.07%	389.00	1,466.00	4.04%	3.00
010	AB2HELL	7	4	AB	35,259.00	164.07%	389.00	1,561.00	4.05%	400.00
010	AB3HELL	7	4	AB	35,259.00	164.07%	389.00	1,291.00	4.06%	368.00
010	Y1	7	4	Y	35,259.00	164.07%	389.00	1,669.00	16.01%	285.00
010	Y2	7	4	Y	35,259.00	164.07%	389.00	1,903.00	12.06%	215.00
010	Y3	7	4	Y	35,259.00	164.07%	389.00	2,665.00	20.03%	266.00
010	YZ1	7	4	YZ	35,259.00	164.07%	389.00	2,392.00	16.13%	202.00
011	ABC1HELL	7	4	ABC	35,818.00	160.18%	301.00	1,428.00	4.09%	0.00
011	ABC2HELL	7	4	ABC	35,818.00	160.18%	301.00	1,496.00	4.17%	0.00
011	ABC3HELL	7	4	ABC	35,818.00	160.18%	301.00	1,497.00	4.15%	0.00
011	AB1HELL	7	4	AB	35,818.00	160.18%	301.00	1,503.00	4.12%	315.00
011	AB2HELL	7	4	AB	35,818.00	160.18%	301.00	1,499.00	4.13%	338.00
011	AB3HELL	7	4	AB	35,818.00	160.18%	301.00	1,493.00	4.16%	58.00
011	Y1	7	4	Y	35,818.00	160.18%	301.00	1,424.00	8.22%	98.00
011	Y2	7	4	Y	35,818.00	160.18%	301.00	1,572.00	8.17%	213.00
011	Y3	7	4	Y	35,818.00	160.18%	301.00	1,464.00	8.07%	32.00
011	YZ1	7	4	YZ	35,818.00	160.18%	301.00	3,730.00	8.25%	125.00
009	ABC1DUNKE	8	4	ABC	37,399.00	184.01%	388.00	1,502.00	4.21%	0.00
009	ABC2DUNKE	8	4	ABC	37,399.00	184.01%	388.00	1,533.00	4.15%	0.00
009	ABC3DUNKE	8	4	ABC	37,399.00	184.01%	388.00	1,605.00	4.23%	0.00
009	AB1DUNKEL	8	4	AB	37,399.00	184.01%	388.00	1,557.00	4.14%	0.00
009	AB2DUNKEL	8	4	AB	37,399.00	184.01%	388.00	1,579.00	4.02%	0.00
009	AB3DUNKEL	8	4	AB	37,399.00	184.01%	388.00	1,625.00	4.04%	0.00
009	Y1	8	4	Y	37,399.00	184.01%	388.00	1,692.00	12.19%	313.00
009	Y2	8	4	Y	37,399.00	184.01%	388.00	1,667.00	12.05%	261.00
009	Y3	8	4	Y	37,399.00	184.01%	388.00	1,904.00	12.16%	283.00
009	YZ1	8	4	YZ	37,399.00	184.01%	388.00	1,998.00	20.08%	187.00
010	ABC1DUNKE	8	4	ABC	35,259.00	164.07%	389.00	1,665.00	4.08%	0.00
010	ABC2DUNKE	8	4	ABC	35,259.00	164.07%	389.00	1,528.00	4.23%	0.00
010	ABC3DUNKE	8	4	ABC	35,259.00	164.07%	389.00	1,561.00	4.09%	0.00
010	AB1DUNKEL	8	4	AB	35,259.00	164.07%	389.00	1,654.00	4.04%	0.00
010	AB2DUNKEL	8	4	AB	35,259.00	164.07%	389.00	1,462.00	4.04%	0.00
010	AB3DUNKEL	8	4	AB	35,259.00	164.07%	389.00	1,492.00	4.22%	384.00
010	Y1	8	4	Y	35,259.00	164.07%	389.00	1,767.00	12.13%	197.00
010	Y2	8	4	Y	35,259.00	164.07%	389.00	1,936.00	12.24%	238.00
010	Y3	8	4	Y	35,259.00	164.07%	389.00	2,096.00	20.18%	309.00
010	YZ1	8	4	YZ	35,259.00	164.07%	389.00	2,499.00	16.20%	194.00

011	ABC1DUNKE	8	4	ABC	35,818.00	160.18%	301.00	1,457.00	4.15%	0.00
011	ABC2DUNKE	8	4	ABC	35,818.00	160.18%	301.00	2,302.00	4.21%	0.00
011	ABC3DUNKE	8	4	ABC	35,818.00	160.18%	301.00	1,560.00	4.19%	0.00
011	AB1DUNKEL	8	4	AB	35,818.00	160.18%	301.00	1,667.00	4.09%	302.00
011	AB2DUNKEL	8	4	AB	35,818.00	160.18%	301.00	1,565.00	4.26%	393.00
011	AB3DUNKEL	8	4	AB	35,818.00	160.18%	301.00	1,472.00	4.18%	0.00
011	Y1	8	4	Y	35,818.00	160.18%	301.00	1,766.00	12.00%	210.00
011	Y2	8	4	Y	35,818.00	160.18%	301.00	2,001.00	12.09%	329.00
011	Y3	8	4	Y	35,818.00	160.18%	301.00	1,740.00	12.13%	249.00
011	YZ1	8	4	YZ	35,818.00	160.18%	301.00	5,190.00	28.15%	168.00