Fakultät für Elektrotechnik und Informationstechnik
Fachgebiet Verteilte Multimodale Informationsverarbeitung
Prof. Dr. Matthias Kranz

Augmented Reality: Navigation und standortbezogene Dienste mit Android-Smartphones

Augmented Reality: Navigation and Location-Based Services Using Android Smartphones

Alexandre Hoffmann

Diplomarbeit

Verfasser:	Alexandre Hoffmann
	[redacted]
	[redacted]
Professor:	Prof. Dr. Matthias Kranz
Betreuer:	Dipl.-Ing. Luis Roalter
Beginn:	01.08.2011
Abgabe:	01.02.2012

Fakultät für Elektrotechnik und Informationstechnik
Fachgebiet Verteilte Multimodale Informationsverarbeitung
Prof. Dr. Matthias Kranz

Erklärung

Hiermit erkläre ich an Eides statt, dass ich diese Diplomarbeit zum Thema

Augmented Reality: Navigation und standortbezogene Dienste mit Android-Smartphones
Augmented Reality: Navigation and Location-Based Services Using Android Smartphones

selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

München, den 1. Februar 2012

Alexandre Hoffmann

Alexandre Hoffmann
Leopoldstr. 169
80804 München

Kurzfassung

Diese Arbeit befasst sich mit dem Einsatz von Augmented Reality (AR) auf Smartphones. Eine weit verbreitete Anwendung von Augmented Reality auf dem Smartphone bilden AR-Browser. Diese ermöglichen dem Benutzer mit umliegenden Points of Interest (POI) zu interagieren. Die AR-Technologie bietet jedoch noch großes bisher ungenutztes Potential. In dieser Arbeit soll ein AR-System entwickelt werden, welches beispielsweise in einem Bürogebäude genutzt werden kann. Dieses AR-System muss dementsprechend für die Verwendung im Indoor-Bereich ausgelegt sein. Außerdem soll es zusätzlich zu den Interaktionen mit Räumen und Personen eine Navigation zur Verfügung stellen. Hierfür wurden bestehende AR-Systeme für Android-Smartphones analysiert und auf den gewonnenen Erkenntnissen aufbauend die benötigten Komponenten erarbeitet. Die vorhandenen Systeme können weder im Indoor-Bereich genutzt werden noch bieten sie eine Möglichkeit zur Navigation. Zudem ist die Interaktion mit den POI allgemein auf einige vordefinierte Funktionen beschränkt. In dieser Arbeit wurde ein AR-System entwickelt, welches sowohl Navigationsfunktionen als auch Interaktionen mit POI bereitstellt. Um den Funktionsumfang der POI-Interaktion flexibel zu gestalten, wurde das System an anpassbare standortbezogene Dienste angebunden. Das AR-System kann sowohl im Indoor- als auch im Outdoor-Bereich eingesetzt werden, wobei ein fließender Übergang zwischen beiden besteht. Die Indoor-Tauglichkeit wurde anhand einer neu entwickelten Android-Anwendung ermöglicht. Diese Lokalisierungsanwendung ermöglicht es neue Lokalisierungssysteme in das Android-Betriebssystem einzufügen. Zudem wurde ein AR-Browser entwickelt, welcher alle genannten Funktionen wie die Navigation oder die Interaktion mit POI anbietet. Der AR-Browser ermöglicht außerdem die Anbindung an die standortbezogenen Dienste, welche die Interaktion mit Räumen und Personen bereitstellen.

Abstract

This thesis studies the utilization of augmented reality (AR) on smartphones. A widespread use of augmented reality on smartphones are AR browsers. These browsers allow the user to interact with surrounding points of interest (POI). AR technology still has great unused potential. This thesis envisages the development of an AR-based system, which can be used in such spaces as, for example, an office building. For this purpose, the AR system has to be designed for indoor and outdoor use. In addition it should allow the user to interact with surrounding rooms and people, as well as to navigate both indoors and outdoors. To this end, existing AR systems for Android smartphones have been analyzed and the required components worked out based on the findings of these analyses. It was determined that present systems are not suitable for indoor use, nor do they offer navigation functions. The interaction with POIs is generally limited to certain predefined functions. In this thesis, an AR system has been developed providing navigation as well as interaction with POIs. To improve the flexibility of POI interaction, customizable location-based services have been developed. The system is suitable for indoor and outdoor use with a smooth transition between these two areas. The indoor suitability is provided through a newly developed Android application, which allows the system to insert new location providers into the Android operating system. In addition an AR browser offering all the above mentioned features, such as navigation and interaction with POIs, has been developed. Furthermore, the AR browser provides the connection to location-based services, which allows the user to interact with rooms and people.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Diese Arbeit wurde im Fachgebiet Verteilte Multimodale Informationsverarbeitung (VMI) der Technischen Universität München verfasst. Sie befasst sich mit dem Einsatz von Augmented Reality (AR) (erweiterte Realität) auf Android-Smartphones. In der Arbeit werden bereits vorhandene AR-Lösungen für Smartphones untersucht. Basierend auf den daraus gesammelten Kenntnissen werden Erweiterungen für die Verwendung von Augmented Reality auf Smartphones erarbeitet. In dieser Einleitung wird zuerst die Motivation der Arbeit beschrieben, anschließend wird eine Übersicht über den Aufbau der Arbeit gegeben.

1.1 Motivation

Forscher arbeiten bereits seit über vierzig Jahren im Bereich der Augmented Reality, doch erst die Verwendung auf dem Smartphone machte diese Technologie auch der breiten Bevölkerung zugänglich. Eine weit verbreitete Anwendung von Augmented Reality auf dem Smartphone bilden AR-Browser. Diese ermöglichen es dem Benutzer mit umliegenden Points of Interest (POI) zu interagieren. Die Interaktionen beschränken sich jedoch auf einige vordefinierte Funktionen, wie zum Beispiel das Aufrufen der Beschreibung eines POI. Eine Anbindung an anpassbare standortbezogene Dienste (Location Based Services – LBS) könnte eine wesentliche Erweiterung des Funktionsumfangs ermöglichen. Die AR-Technologie auf den Smartphones bietet überdies noch viel ungenutztes Potential. Die AR-Browser könnten nicht nur für Interaktionen, sondern beispielsweise auch für eine AR-basierte Navigation genutzt werden. Die vorhandenen AR-Browser sind für die Verwendung im Outdoor-Bereich ausgelegt. Sie können dementsprechend nicht oder nur beschränkt in Gebäuden und anderen Indoor-Einrichtungen genutzt werden. Durch diese Einschränkung können die vorhandenen AR-Browser in vielen Situationen nicht eingesetzt werden.

Nachfolgendes Beispielszenario veranschaulicht neue Anwendungsmöglichkeiten, welche durch die genannten Erweiterungen realisiert werden. Das Szenario zeigt die Verwendung von AR auf dem Smartphone innerhalb großer Bürogebäude. Demzufolge wird ein AR-System benötigt, welches innerhalb von Gebäuden, also im Indoor-Bereich, genutzt werden kann.

Ein derartiges AR-System erlaubt es dem Benutzer sich beispielsweise über die umliegenden Räume zu erkundigen und mit diesen zu interagieren. Verschiedene Dienste können dem Benutzer zur Verfügung gestellt werden, wie die Reservierung eines bestimmten Raums für einen gewissen Zeitraum oder die Kontaktaufnahme zu Personen in bestimmten Räumen. Die verfügbaren Dienste können dabei von Raum zu Raum variieren. Mit Hilfe der AR-Navigation kann sich der Benutzer anhand der auf seinem Smartphone-Bildschirm angezeigten Informationen bis zu einem gewünschten Raum leiten lassen. Zudem kann er sich den Weg aus dem Gebäude zu einer gewünschten Adresse anzeigen lassen. So kann die Indoor- mit der Outdoor-Navigation kombiniert werden. Nachfolgendes Beispiel verdeutlicht den Nutzen eines derartigen AR-Systems. Ein Meeting soll im Raum 001 eines großen Bürogebäudes stattfinden. Ein Teilnehmer dieses Meetings nutzt die öffentlichen Verkehrsmittel und kann mit Hilfe des AR-Systems von der nächstgelegenen Bushaltestelle zu dem Bürogebäude und weiter bis zum Raum 001 navigieren. Die Tür zum Raum 001 ist jedoch wegen Renovierungsarbeiten verschlossen. Der Benutzer des AR-Systems kann nun die standortbezogenen Dienste des Raums nutzen und ermitteln, in welchen Raum das Meeting verlegt wurde. Er kann daraufhin anhand des AR-Browsers zum gewünschten Raum navigieren.

Das Ziel dieser Arbeit besteht in der Realisierung eines AR-Systems, welches die in dem vorigen Beispiel erläuterten Erweiterungen ermöglicht. Das zu entwickelnde AR-System soll sowohl Navigationsfunktionen als auch Interaktionen mit POI bereitstellen. Zudem soll das System an anpassbare standortbezogene Dienste angebunden werden. Das System soll sowohl im Indoor- als auch im Outdoor-Bereich eingesetzt werden, wobei der Übergang zwischen beiden fließend sein soll. Für die Umsetzung des AR-Systems werden Android-Smartphones verwendet.

1.2 Aufbau der Arbeit

Die Arbeit beginnt mit der Analyse des aktuellen Forschungsstandes. Die einzelnen Themengebiete werden in Kapitel 2 im Detail beschrieben. Zuerst werden die für die Umsetzung des AR-Systems genutzten Android-Smartphones vorgestellt. Anschließend wird auf die Standortbestimmung sowie auf die Augmented Reality im Detail eingegangen. Zuletzt werden die standortbezogenen Dienste erläutert.

In Kapitel 3 wird ein auf diesen Grundlagen aufbauendes Konzept entwickelt, welches eine Lösung für die Einschränkungen der bestehenden Systeme bereitstellt. Zuerst wird das Gesamtkonzept vorgestellt, anschließend wird auf die einzelnen Themengebiete detaillierter eingegangen. Das Konzept umfasst einen möglichen Lösungsansatz für die Standortbestimmung im Indoor-Bereich. Zudem enthält es eine Lösung für einen AR-Browser, der sowohl eine Interaktion mit POI als auch eine Navigation im Indoor- und Outdoor-Bereich ermöglicht. Außerdem ist der konzipierte AR-Browser an anpassbare standortbezogene Dienste angebunden.

Die Implementierung des konzipierten Systems wird in Kapitel 4 beschrieben. Dabei werden die Android-Anwendungen für die Lokalisierung und den AR-Browser näher beschrieben. Zudem werden die zugehörigen Server-Komponenten sowie die implementierten standortbezogenen Dienste im Detail erläutert.

In Kapitel 5 werden abschließend die erzielten Resultate zusammengefasst und mögliche Erweiterungen des entwickelten Systems vorgestellt.

Kapitel 2

Grundlagen

In diesem Kapitel werden die Grundlagen für Android-Smartphones beschrieben. Ferner wird der Aspekt der Standortbestimmung näher erklärt, wobei aufgezeigt wird, welche Methoden und Systeme bestehen und wie genau die Standortbestimmung auf einem Smartphone funktioniert. Weiterhin wird die Funktionsweise der Augmented Reality (erweiterte Realität) sowie deren Anwendung in der Navigation beschrieben. Abschließend werden die standortbezogenen Dienste erläutert.

2.1 Android-Smartphones

Smartphones erfreuen sich immer größerer Beliebtheit [1]. Im Jahr 2005 wagte sich auch Google in diese Branche, indem es das Unternehmen Android Inc. übernahm. Im November 2007 wurde die Open Handset Alliance [2] gegründet. Dieses Konsortium, bestehend aus mehreren Hard- und Softwareherstellern sowie Mobilfunkanbietern, wird von Google geführt. Am 1. Oktober 2008 veröffentlichte die Open Handset Alliance das unter der Federführung von Google entwickelte Android-Betriebssystem. Das Android-Betriebssystem (fortan „Android“ genannt) ist quelloffen und steht unter einer Apache-Lizenz. Das erste Smartphone mit Android war das HTC Dream, auch unter dem Namen T-Mobile G1 bekannt und wurde von HTC [3] gefertigt. Zusätzlich zum Betriebssystem veröffentlichte Google auch das Android Software Development Kit (SDK), welches Entwicklern ermöglicht, eigene Applikationen für das System zu schreiben. Das SDK bietet dem Entwickler eine umfangreiche Programmierschnittstelle (Application Programming Interface – API), die auf der Programmiersprache Java beruht. Android-Smartphones werden von großen Smartphoneherstellern wie z.B. Samsung [4], HTC, LG [5] sowie auch von Google [6] selbst (in Kooperation mit den Herstellern) produziert. Laut dem Marktforschungsunternehmen Gartner werden Android-Smartphones bis zum Ende des Jahres 2012 fast die Hälfte des gesamten Smartphone-Marktes ausmachen [7].

2.1.1 Android-Betriebssystem

Das Betriebssystem gliedert sich in folgende Ebenen: Kernel, Bibliotheken und Android-Laufzeitumgebung, Anwendungsrahmenstruktur und Anwendungen [8]. Dieser Aufbau ist in der Abbildung 2.1 sichtbar.

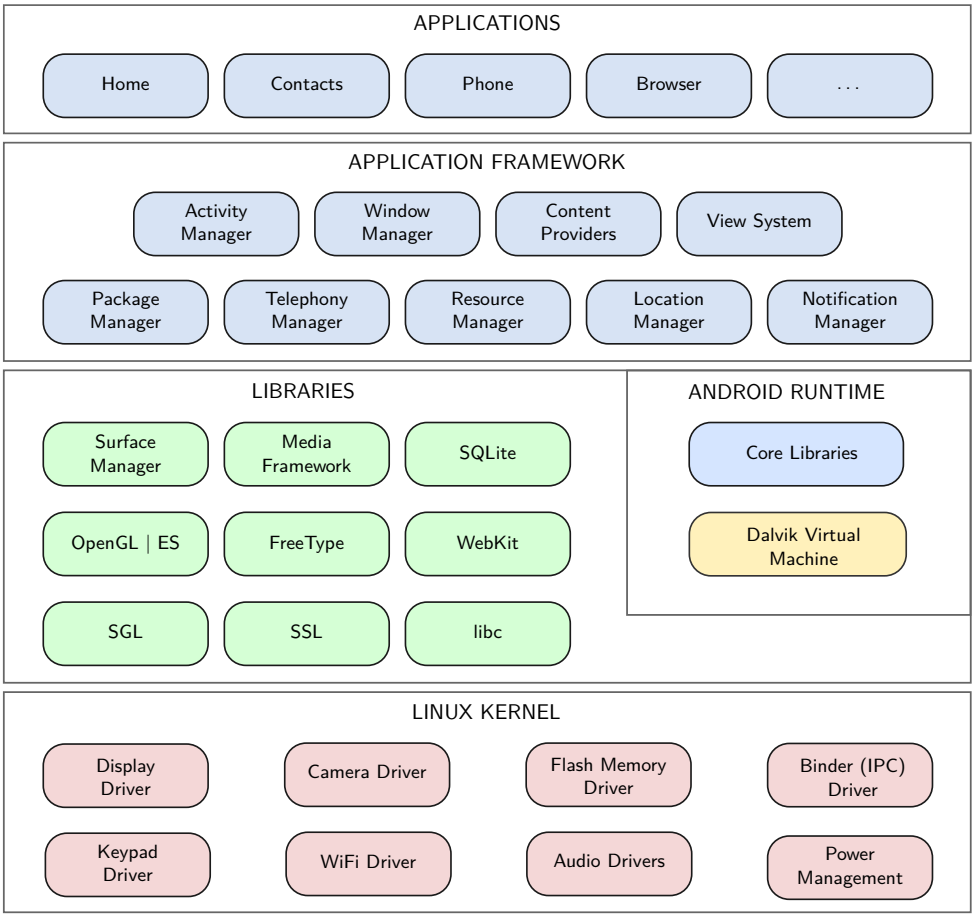

Abbildung 2.1: Architektur des Android-Betriebssystems modifiziert nach [8]

Kernel Die Kernsysteme von Android sowie Sicherheit, Speicherverwaltung, Prozessverwaltung und diverse Treiber bauen auf einem Linux Kernel 2.6 auf. Diese unterste Ebene bildet die Verbindung zwischen der Hardware und dem Betriebssystem.

Bibliotheken und Android-Laufzeitumgebung Die Ebene über dem Kernel beinhaltet die Standardbibliotheken des Betriebssystems. Diese sind alle in C oder C++ verfasst und speziell für die Hardware-Architektur des Smartphones kompiliert. Die Standardbibliotheken beinhalten die Grundfunktionalitäten wie zum Beispiel 2D- und 3D-Grafik, Browser-Engine, Media Codecs und

eine SQL-Datenbank.
Die Android-Laufzeitumgebung beinhaltet die Dalvik Virtual Machine (DVK) sowie die Java-Standardbibliotheken. Die DVK ist eine von Google entwickelte virtuelle Maschine. Diese dient der Ausführung des Java-Codes und wurde für die Nutzung auf mobilen Endgeräten optimiert.

Anwendungsrahmenstruktur (Application Framework) Über der Ebene der Standardbibliotheken und der Laufzeitumgebung befindet sich die Ebene des Application Frameworks. Diese Rahmenstruktur stellt High-Level-Bausteine für die Entwicklung von Applikationen zur Verfügung. Somit kann man leicht aus Anwendungen heraus auf Benachrichtigungsdienste, Ressourcen und Hardware-Komponenten zugreifen. Ein Beispiel für eine solche Hardware-Komponente wäre das GPS-Modul.

Anwendungen Die höchste Ebene im Android-Betriebssystem ist die Anwendungsebene. Alle Anwendungen sind in Java entwickelt und werden als APK-Dateien verpackt auf dem Smartphone installiert. Der Endbenutzer sieht nur die Anwendungen dieser Ebene.

2.1.2 Android-Anwendungen

Die Android-Anwendungen werden so eigenständig und gekapselt wie möglich konzipiert [8]. Dieses Sicherheitskonzept wird einerseits dadurch umgesetzt, dass jede Applikation als unabhängiger Benutzer im Android-Betriebssystem geführt wird. Andererseits wird es so umgesetzt, dass jede Applikation einen eigenen Prozess und eine eigene virtuelle Maschine besitzt. Eine Anwendung kann zudem nur auf diejenigen Systemressourcen zugreifen, für die sie explizite Zugriffsrechte besitzt.
Im Folgenden werden die wesentlichen Komponenten einer Android-Anwendung erläutert.

Activities Eine Activity besitzt eine graphische Oberfläche, welche Benutzerinteraktionen mit der Anwendung ermöglicht. Dazu stellt jede Activity eine graphische Benutzer-Schnittstelle (User Interface) zur Verfügung. Eine Anwendung besteht im Allgemeinen aus mehreren Activities, die miteinander verbunden sind. Jede Anwendung definiert eine bestimmte Activity als Einstiegspunkt, welche beim Start der Anwendung aufgerufen wird. Von dieser Activity aus können dann weitere Services oder Activities gestartet werden, um andere Funktionen auszuführen.

Services Ein Service dient der Ausführung langandauernder Aufgaben im Hintergrund und besitzt keine Benutzeroberfläche. Er wird durch eine Anwendungskomponente, wie zum Beispiel

eine Activity, gestartet. Der Service läuft im Hintergrund und wird bei Beenden der Anwendungskomponente, welche den Service gestartet hat, nicht gestoppt. Es wird zwischen zwei Arten von Services unterschieden: Started und Bound.

- Ein Started-Service wird aus einer Anwendung heraus gestartet und läuft dann solange im Hintergrund bis er seine Aufgabe erfüllt hat oder bis er manuell beendet wird. Dabei übergibt er keine Rückgabewerte an die Anwendungskomponente, welche den Service gestartet hat.
- Man spricht von einem Bound-Service, wenn eine Anwendung sich mit dem Service verbindet. Diese Verbindung ermöglicht eine Client-Server-Interaktion zwischen den beiden Komponenten. Zudem können fremde Anwendungen sich mit dem Service verbinden, wodurch eine Interprozess-Kommunikation (Interprocess Communication – IPC), also eine Kommunikation zwischen verschiedenen Anwendungen, ermöglicht wird.

Intents Intents sind Nachrichten zum Starten von Activities und Services. Die zu startenden Activities oder Services können sich sowohl in der eigenen Anwendung als auch in fremden Anwendungen befinden. Sie bieten somit auch eine Möglichkeit mit anderen Anwendungen zu interagieren. Aus diesem Grund existieren zwei Arten von Intents: explizite und implizite Intents

- Explizite Intents starten Komponenten anhand ihrer Namen. Da die Namen von Activities und Services fremden Entwicklern nicht bekannt sind, wird diese Methode hauptsächlich dazu verwendet, anwendungsinterne Komponenten zu starten.
- Bei impliziten Intents hingegen wird kein Komponentenname angegeben. In diesem Fall sucht das Android-System nach einer passenden Komponente. Dabei werden die Angaben in dem Intent mit denen in den verfügbaren Intent-Filtern verglichen, bis eine passende Komponente gefunden wird. Intent-Filter dienen dazu, dem System bekanntzugeben, welche implizite Intents die Komponente verarbeiten kann. Besitzt eine Komponente keinen Intent-Filter, so kann diese nur durch explizite Intents aufgerufen werden.

Android-Manifest Jede Android-Anwendung besitzt ein Android-Manifest in Form einer XML-Datei. Diese beinhaltet globale Eigenschaften der Anwendung, wie die Auflistung der verwendeten Komponenten (Activities und Services) oder die Zugriffsrechte (Permissions) der Anwendung auf Ressourcen. Zu jeder Komponente können zusätzliche Eigenschaften definiert werden, so kann zum Beispiel einer Activity ein Intent-Filter zugewiesen werden. Die Definition der Zugriffsrechte erlaubt es auf Systemressourcen, wie die Smartphone-Kamera, zuzugreifen. In der Manifest-Datei werden also anwendungsspezifische Einstellungen vorgenommen.

2.2 Standortbestimmung

Die Standortbestimmung oder Lokalisierung beschreibt die Ermittlung der physischen Position eines Objektes oder einer Person in der realen Welt [9]. Zuerst werden in diesem Abschnitt die verschiedenen Darstellungsmöglichkeiten eines Standortes erläutert. Anschließend folgt eine Beschreibung verschiedener Methoden der Standortbestimmung. Im Abschnitt 2.2.3 werden dann die Unterschiede zwischen client- und netzwerkbasierten Lokalisierungssystemen erklärt. Abschließend wird die Verwendung des Smartphones für die Standortbestimmung näher betrachtet. Insbesondere wird aufgezeigt, welche Lokalisierungssysteme mit dem Smartphone heutzutage kompatibel sind.

2.2.1 Darstellung von Standortinformationen

Ein Standort beschreibt eine Position in einem physikalischen Raum und kann in symbolischer, relativer oder absoluter Form angegeben werden [9]. Ein symbolischer Standort wird beispielsweise durch eine Adresse oder den Namen eines Platzes ausgedrückt. Ein relativer Standort hingegen bezieht sich auf die Lage zu anderen Standorten. Ein Beispiel hierfür wäre folgende Angabe: 100 Meter nördlich von einem bestimmten Gebäude. Für die Repräsentation eines absoluten Standorts werden allgemein geographische Koordinaten (Längen- und Breitengrad) verwendet. Als Referenzsystem für absolute Koordinaten kann zum Beispiel das World Geodetic System 1984 (WGS84) [10] verwendet werden. Dieses geodätische Referenzsystem wird zudem von dem Global Positioning System (GPS) [11] verwendet.

2.2.2 Methoden der Standortbestimmung

Bei der Standortbestimmung können folgende Methoden unterschieden werden [9, 12]:

Nachbarschaftserkennung Die Nachbarschaftserkennung oder Proximity ist die einfachste Methode der Lokalisierung. Bei dieser Methode wird die Nähe zu einem Referenzpunkt ermittelt, um so den Standort zu schätzen. Diese Schätzung kann zum Beispiel durch Erkennung von Referenzpunkten anhand der Funksignale erfolgen.

Lateration Bei der Lateration wird anhand von Entfernungen zu bekannten Referenzpunkten der Standort ermittelt. Dabei ist die Anzahl der benötigten Referenzpunkte um eins höher als die Anzahl der Dimensionen. Will man also den Standort in zwei Dimensionen (2D) bestimmen, so braucht man 3 Referenzpunkte (Trilateration).

Angulation Die Angulation funktioniert ähnlich wie die Lateration, nur dass bei dieser Methode anstelle der Entfernungen die Winkel zu den Referenzpunkten zur Positionsbestimmung verwendet werden.

Mustererkennung (Fingerprinting oder Pattern Matching) Diese Methode schätzt die Position durch Abgleichen von Signalen mit einer Datenbank. Die Methode besteht aus zwei Phasen, in der ersten Phase wird die Datenbank aufgebaut und trainiert, in der zweiten Phase wird der Standort bestimmt. Beim Aufbauen der Datenbank werden zum Beispiel Funksignale an verschiedenen Positionen gemessen und in der Datenbank gespeichert.

Koppelnavigation Koppelnavigation oder Dead Reckoning berechnet die Position bezogen auf eine vorher ermittelte Referenzposition. So kann zum Beispiel durch Messen von Beschleunigungs- und Richtungssensoren eine neue Position bestimmt werden. Da diese Methode nur relative Positionen berechnet, muss sie zusammen mit einer anderen Methode, welche absolute Positionen ermittelt, verwendet werden um den absoluten Standort zu bestimmen.

2.2.3 Client- und netzwerkbasierte Lokalisierung

In diesem Abschnitt werden die client- und netzwerkbasierten Lokalisierungssysteme beschrieben [9]. Zusätzlich bietet die netzwerkgestützte Lokalisierung eine hybride Lösung aus client- und netzwerkbasierter Lokalisierung.
Bei einem clientbasierten Lokalisierungssystem berechnet das Endgerät seine Position selbst, ohne dabei auf eine Netzwerkstruktur angewiesen zu sein. Ein Beispiel hierfür ist das GPS, da nur die empfangenen Signale ausgewertet werden.
Bei einer netzwerkbasierten Lokalisierung übernimmt das Netzwerk die Standortbestimmung. Ein Beispiel hierfür ist die Mobilfunkortung, wobei der Standort des Endgerätes vom Netzwerk ermittelt werden kann.
Bei einem netzwerkgestützten Lokalisierungssystem partizipiert sowohl das Endgerät als auch das Netzwerk an der Standortbestimmung. Ein Beispiel für dieses Verfahren ist das Assisted GPS (A-GPS), welches zum einen die GPS-Daten der Satelliten empfängt und zum anderen zusätzliche Informationen wie zum Beispiel Satellitenlaufbahnen über das Mobilfunknetz erhält.
Der Vorteil von clientbasierten Lokalisierungsverfahren gegenüber netzwerkbasierten ist der, dass die Privatsphäre des Benutzers nicht gefährdet wird, da das Endgerät nur Signale empfängt und nicht sendet. Allerdings erfordert die Positionsberechnung auf dem Endgerät einen höheren Energiebedarf (Batterie) sowie eine gewisse Speicher- und Rechenleistung.

2.2.4 Standortbestimmung mit dem Smartphone

Die weite Verbreitung von Smartphones macht diese interessant für die Standortbestimmung. Nach Canalys wurden bereits im Jahr 2008 40% der Smartphones im europäischen Wirtschaftsraum mit integriertem GPS ausgeliefert [13].
In modernen Smartphones sind jedoch weitere Hardware-Module verbaut, die zur Lokalisierung verwendet werden können. Für die Standortbestimmung können unter anderem folgende Funkkomponenten in dem Smartphone verwendet werden:

- GPS
- WiFi
- Mobilfunk

Zusätzlich zu den Funkkomponenten kann auch die Sensorik des Smartphones genutzt werden, um den Standort zu ermitteln:

- Digitaler Kompass
- Gyroskop
- Beschleunigungssensor
- Kamera

Diese Auflistungen zeigen mögliche Hardware-Komponenten, anhand derer eine Lokalisierung durchgeführt werden kann. Zur Zeit werden jedoch nur GPS, Lokalisierung anhand des Mobilfunksignals und WiFi standardmäßig eingesetzt.

Die vorgestellte Hardware ermöglicht verschiedene Systeme der Standortbestimmung auf dem Smartphone. Nachfolgend werden einige dieser Systeme genauer erläutert.

GPS Bei GPS handelt es sich um ein Satellitennavigationssystem [14]. Die Position wird dabei aus den von Satelliten empfangenen Daten berechnet. GPS wurde ursprünglich für das Militär entwickelt, wurde jedoch später für die zivilen Nutzung freigegeben. Man findet es heutzutage in vielen kommerziellen Anwendungen, wie zum Beispiel in Navigationsgeräten für Autos. GPS hat eine mediane Genauigkeit von 10 Metern im Außenbereich (Outdoor), welche sich jedoch durch Hindernisse, wie zum Beispiel hohe Gebäude oder auch Berge, wesentlich verschlechtern kann. Im Innenbereich (Indoor) ist GPS wegen dem schlechten Empfang der Satellitensignale nur begrenzt oder nicht einsetzbar.

WiFi Die WiFi-Ortung ermöglicht die Positionsbestimmung anhand eines WiFi-Netzwerkes [15]. Es existieren hierfür verschiedene Methoden, wie zum Beispiel Mustererkennung, Lateration der

Stärken oder Laufzeiten der Signale. Die Genauigkeit hängt von den Verfahren ab, liegt aber allgemein unter 10 Metern.
Das RADAR-System [16] ist ein bekanntes Beispiel für ein WiFi-Ortungssystem. Die Genauigkeit dieses Systems liegt zwischen 2 und 3 Metern.
Nach Elnahrawy et al. [17] liegen bei der WiFi-Lokalisierung die Grenzen der Genauigkeit bei einem medianen Fehler von ungefähr 3 Metern.

Mobilfunk Die Mobilfunkortung bezeichnet die Lokalisierung eines im Netz angemeldeten mobilen Endgerätes durch das Mobilfunknetz [12]. Es existieren mehrere Methoden, um eine solche Ortung durchzuführen. Diese unterscheiden sich in ihrer Genauigkeit und Komplexität. Die Genauigkeit der Mobilfunkortung ist im Allgemeinen in der Stadt besser als auf dem Land. Im Städtebereich liegt diese bei ungefähr 40 bis 1000 Metern[1], je nach Verfahren und Störungen. Das einfachste und auch für Android-Smartphones verfügbare Verfahren nennt sich Cell-Id. Es basiert auf Nachbarschaftserkennung und ermittelt die Position anhand der umliegenden Mobilfunkstationen. Um die Genauigkeit zu erhöhen können zusätzlich die Signallaufzeiten zwischen Smartphone und Basisstation ausgewertet werden. Das Cell-Id-Verfahren hat den Vorteil, dass keine speziellen Infrastrukturen benötigt werden. Die Genauigkeit des Verfahrens liegt in der Stadt zwischen 50 und 1000 Metern und kann auf dem Land bis über 10 Kilometer betragen. Die Genauigkeitswerte der angegebenen Verfahren sind für die Indoor-Lokalisierung nicht ausreichend. Varshavsky et al. [18] zeigen jedoch, dass die Mobilfunkortung im Indoor-Bereich sinnvoll genutzt werden kann. Es wird eine auf Mustererkennung basierende Methode vorgestellt, welche eine Genauigkeit von 2 bis 4 Metern ermöglicht.
Eine Alternative zu diesen rein auf Mobilfunk basierenden Verfahren bietet die Standortbestimmung mittels Assisted GPS (A-GPS). Bei dem Verfahren werden Hilfsdaten über das Mobilfunknetz übertragen, um eine genauere und schnellere GPS-Standortbestimmung zu ermöglichen.

Bild- und Markererkennung Ein weiteres System der Lokalisierung basiert auf der Wiedererkennung von Bilddaten. Als Bestimmungsmethode kommt hier die Mustererkennung zum Einsatz, welche es ermöglicht Marker oder Bilder mit vorher bekannten oder aufgenommenen abzugleichen. Eine Lokalisierung anhand von Markern ermöglicht nur eine diskrete Standortbestimmung, da nur beim Scannen eines Markers eine Aktualisierung der Position erfolgt. Hile und Borriello [19] zeigen, wie eine durch WiFi-Ortung ermittelte Position mit Hilfe der Bilderkennung verbessert werden kann. Dabei werden im Indoor-Bereich Orientierungspunkte, welche von der Kamera erkannt werden, mit dem Flurplan des Gebäudes abgeglichen. Mulloni et al. [20] und Chang et al.[21] verwenden hingegen 2D-Marker für die Positionierung. Hattori et al. [22] verwenden

[1] Diese Genauigkeitswerte resultieren aus folgenden Verfahren: Cell-Id, Enhanced Observed Time Difference (E-OTD), Uplink Time Difference of Arrival (U-TDoA), Observed Time Difference Of Arrival With Idle Period Downlink (OTDoA-IPDL), die von Kupper in [12] genauer erläutert werden.

zusätzlich zu den Markern eine WiFi-Ortung. Dieses hybride System erlaubt eine kontinuierlichere Positionsbestimmung und eine geringere Dichte an Markern. Eine hybride Lösung aus Bildfolgenerkennung und Markern wird von Kim und Jun [23] untersucht. Diese Lösung bietet eine kontinuierliche Lokalisierung, welche nur auf dem Einsatz der Kamera beruht.

Bewegungserkennung Ein weiteres System, welches auf dem Smartphone zur Verfügung steht, ist die Standortbestimmung mittels Bewegungserkennung. Hier werden durch die Sensorik gemessene Bewegungen erfasst und die Position bestimmt. Da durch Bewegungen nur relative Positionen ermittelt werden können, wird als Bestimmungsmethode die Koppelnavigation verwendet. Ein Beispiel für eine solche Lokalisierung wird von Parnandi et al. [24] vorgestellt. Die Bestimmung der Distanz erfolgt durch Zählen der gelaufenen Schritte. Die Richtung wird mit Hilfe des Kompasses ermittelt. Ein alternatives Verfahren zum Zählen von Schritten mit dem Smartphone wird von Bylemans [25] vorgestellt.

Fazit

Die vorgestellten Systeme zeigen die Möglichkeiten zur Durchführung einer Standortbestimmung mit einem Smartphone. Die Implementierungen, die im Moment standardmäßig in Smartphones vorhanden sind, reichen im Allgemeinen nicht für eine präzise Indoor-Lokalisierung aus. Android-Smartphones zum Beispiel verfügen über Mobilfunkortung mittles Cell-Id, WiFi-Ortung mittels Fingerprinting und GPS-Ortung. Wie bereits gezeigt, reicht die Genauigkeit der Mobilfunkortung mittels Cell-Id nicht für den Indoor-Gebrauch aus. Bei der WiFi-Ortung mittels Fingerprinting im Indoor-Bereich liegt das Problem meistens in der Datenbank. Diese wird bei Android-Betriebssystemen von Google bezogen und enthält somit für die meisten Gebäude keine Standortinformationen. Die GPS-Ortung kann im Indoor-Bereich wegen der schlechten Signalstärke nur eingeschränkt genutzt werden. Die vorhandene Hardware kann jedoch durch Erweiterung der bestehenden Lokalisierungssysteme und Implementierung neuer Systeme eine präzise Indoor-Lokalisierung ermöglichen. Wie beispielsweise von Martin et al. [26] gezeigt wird, kann anhand einer Kombination aus WiFi, Mobilfunk, und Bewegungserkennung im Indoor-Bereich eine Genauigkeit von 1,5 Metern erreicht werden.

2.3 Augmented Reality

Seit Ende der sechziger Jahre spricht man bereits von Augmented Reality (AR) [27]. Eine einheitliche Definition existiert jedoch bis heute nicht. Milgram [28, 29] definiert das Realitäts-Virtualitäts-Kontinuum, welches in der Abbildung 2.2 zu sehen ist. Er unterscheidet zwischen verschiedenen Mischungen von Realität und Virtualität. Die Mischung von Realität und Virtualität wird Mixed

Reality (gemischte Realität) genannt. Sie unterteilt sich in Augmented Reality und Augmented Virtuality (erweiterte Virtualität). Die Augmented Reality bezeichnet eine Erweiterung der Realität durch virtuelle Objekte (Computer-Grafiken).

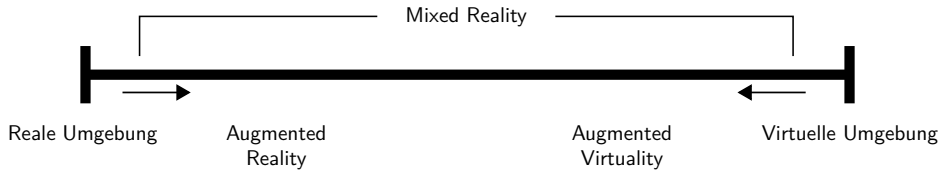

Abbildung 2.2: Realitäts-Virtualitäts-Kontinuum modifiziert nach [28]

Azuma [30] orientiert sich an Milgram und definiert die Augmented Reality durch folgende Punkte:

- Kombination aus Virtualität und Realität
- Echtzeit-Interaktion
- 3-dimensionaler Bezug zwischen virtuellen und realen Objekten.

Man spricht von Augmented Reality, wenn jeder dieser Punkte erfüllt ist. Die Definition hat jedoch den Nachteil, dass sie sich nur auf die visuellen Aspekte der Augmented Reality bezieht.

Augmented-Reality-Typen Man kann Augmented Reality grob in zwei Gruppen einteilen. Die erste Gruppe bezieht sich auf die Augmented Reality, welche Bilderkennung (Image Recognition) für die Darstellung der virtuellen Objekte verwendet. Die zweite Methode hingegen nutzt Positionsdaten (Geographical Information System – GIS), um die virtuellen Objekte abzubilden. Bei der AR mittels Bilderkennung unterscheidet man zudem, ob die Bilderkennung einen Marker verwendet oder nicht. Beispiele für die genannten Typen sind in nachfolgenden Abschnitten beschrieben.

2.3.1 Augmented Reality auf Smartphones

Smartphones ermöglichen es Augmented Reality einem breiten Publikum zur Verfügung zu stellen. Geiger et al. [31] verwendeten im Jahr 2001 erstmals Augmented Reality auf einem Personal Digital Assistant (PDA) und Mohring et al. [32] stellten 2004 das erste komplett auf einem Mobilfunkgerät laufende Augmented-Reality-System vor.
Moderne Smartphones erfüllen alle Voraussetzungen, die für ein AR-System notwendig sind. Sie verfügen über eine Kamera, um die Realität abzubilden und über Sensoren sowie Ortungstechnik, um die Ausrichtung und den Standort des Smartphones zu bestimmen. Zudem verfügen sie über ausreichend Rechenleistung, um die virtuellen Objekte darzustellen.

Augmented-Reality-Browser

Eine weit verbreitete Anwendung der AR auf Smartphones bilden die Augmented-Reality-Browser. Die Funktionsweise dieser Browser basiert auf der positionsabhängigen AR. Sie ermöglichen es verschiedenste Inhalte, wie zum Beispiel Restaurant-Informationen oder Reiseinformationen, im Umkreis des aktuellen Standorts darzustellen. Die dargestellten Objekte, auch Points of Interest (POI) (interessanter Ort) genannt, werden von Servern bezogen und sind in verschiedene Gruppen, auch Providers oder Channels genannt, aufgeteilt. Es existieren beispielsweise Gruppen für Restaurants oder für Reiseinformationen. Durch Aktivieren einer Gruppe werden die zugehörigen POI heruntergeladen und angezeigt. Die bekanntesten AR-Browser sind Layar [33], Wikitude [34] und Junaio [35], welche alle für Android-Smartphones zur Verfügung stehen.

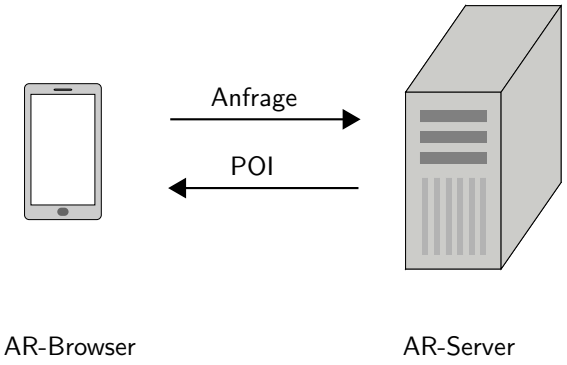

Abbildung 2.3: Kommunikation zwischen AR-Browser und Server

Die Architektur der vorgestellten AR-Browser wird in Abbildung 2.3 dargestellt. Die Struktur besteht aus 2 bis 3 Komponenten: dem AR-Browser auf dem Smartphone, dem AR-Server und optional mehreren separaten POI-Servern. Wenn separate POI-Server vorhanden sind, dient der AR-Server als Verbindungskomponente zwischen dem AR-Browser und dem POI-Server. Er ermöglicht eine Auslagerung der POI-Daten nach Gruppen, das heißt unterschiedliche Gruppen von POI können sich auf verschiedenen POI-Servern befinden. Der POI-Server verwaltet und speichert die AR-Daten. Wenn hingegen kein separater POI-Server vorhanden ist, sind AR- und POI-Server zu einer Komponente zusammengefasst.

Die Anfrage an den Server besteht im Allgemeinen aus einer HTTP-GET-Anfrage [36], welche die Koordinaten (Breiten- und Längengrad), an denen sich der Benutzer des AR-Browsers befindet, beinhaltet. Anhand dieser Koordinaten kann der Server die umliegenden POI ermitteln.

Das Format, in dem die POI übermittelt werden, unterscheidet sich bei jedem der vorgestellten Browser. Es bestehen jedoch Bestrebungen einen Standard zu definieren, wobei das von Wikitude entwickelte Format Augmented Reality Markup Language (ARML) [37] als Basis dient. ARML

basiert auf XML und soll in der Version 2.0 zum weltweiten Standard für AR-Anwendungen werden. Dieser Standard entsteht unter der Aufsicht des Open Geospatial Consortium (OGC) [38]. Das OGC ist ein internationales Industrie-Konsortium mit rund 442 Unternehmen, öffentlichen Einrichtungen und Universitäten.

Die vorgestellten AR-Browser haben verschiedene Nachteile. Zum einen können sie jeweils nur eine Gruppe von POI gleichzeitig anzeigen. Es ist also nicht möglich, Reiseinformationen gleichzeitig mit Restaurants anzuzeigen, wenn diese in verschiedene Gruppen unterteilt sind. Zum anderen sind die AR-Browser darauf ausgerichtet einen einmaligen Überblick über die umliegenden POI zu ermöglichen, denn die Standortinformationen werden je nach Browser nur selten oder gar nicht aktualisiert. Sie sind also für den kontinuierlichen Betrieb, zum Beispiel für Navigation, weniger geeignet.

2.3.2 Augmented Reality und Navigation

Augmented Reality kann sowohl im Outdoor- als auch im Indoor-Bereich für die Navigation verwendet werden. Ein Beispiel für die Outdoor-Navigation wird von Thomas et al. [39] vorgestellt. Bei diesem System handelt es sich um das erste AR-Navigationssystem. Es basiert auf positionsbezogener Augmented Reality, wobei die Navigation durch angezeigte Wegpunkte realisiert wird. Auch für die Indoor-Navigation existieren einige Beispiele. Höllerer et al. [40] stellen eine Indoor-Navigation vor, welche auf positionsbezogener Augmented Reality basiert. Die AR-Inhalte, Rauminformationen und Navigationshinweise werden hier also an den entsprechenden Koordinaten angezeigt. Reitmayr und Schmalstieg [41] sowie Kim und Jun [23] zeigen Beispiele für Indoor-Navigationssysteme, die auf Bilderkennung basieren. Die Hinweise der AR-Navigation werden bei diesen Systemen nicht in Bezug auf ihre Koordinaten, sondern an einer fixen Position im Bildschirm oder bezogen auf die erkannten Bilder angezeigt.

Da Smartphones Augmented Reality darstellen können und über Lokalisierungssysteme verfügen, besteht zudem die Möglichkeit Augmented-Reality-Navigation auf dem Smartphone einzusetzen. Beispiele für solche Systeme zeigen Hile und Borriello [19] sowie Mohareri und Rad [42]. Beide vorgestellten Systeme basieren auf Augmented Reality mittels Bilderkennung, wobei Mohareri und Rad für die Positionierung der virtuellen Objekte Marker verwenden.

2.4 Standortbezogene Dienste

Obwohl standortbezogene Dienste (Location Based Services – LBS) schon seit mehreren Jahren im Bereich der mobilen Kommunikation vertreten sind, existiert keine allgemein anerkannte Definition [12]. Die Ursache hierfür liegt wahrscheinlich an den unterschiedlichen Bereichen, in denen standortbezogene Dienste vorkommen. Dies führte zu unterschiedlichen Definitionen durch

verschiedene Fachrichtungen, insbesondere aus den Bereichen Telekommunikation und Ubiquitous Computing.

Schiller und Voisard [43] sowie Brimicombe und Li [44] verstehen unter standortbezogenen Diensten, dass anhand der Position des mobilen Endgerätes dem Benutzer selektive Informationen und Dienste zur Verfügung gestellt werden.

2.4.1 Reaktive und proaktive standortbezogene Dienste

Nach Kupper [12] unterscheidet man zwei Arten von standortbezogenen Diensten: reaktive und proaktive Dienste.

Ein reaktiver standortbezogener Dienst wird ausdrücklich vom Benutzer aktiviert. Um eine Interaktion mit einem solchen Dienst durchzuführen, stellt der Benutzer zuerst eine Verbindung zu diesem her. Daraufhin kann er auf die für seinen Standort zur Verfügung stehenden Informationen und Funktionen zugreifen. Der Dienst bearbeitet die Anfrage anhand der erhaltenen Positionsdaten und liefert dem Benutzer das Ergebnis zurück. Ein Beispiel für ein solches Szenario wäre eine Abfrage umliegender Restaurants und Bars.

Proaktive Dienste hingegen benötigen keine explizite Anfrage des Benutzers, sondern aktivieren sich automatisch, wenn ein vordefiniertes Szenario eintritt. So kann man zum Beispiel beim Betreten eines Einkaufszentrums automatisch per SMS über aktuelle Aktionsangebote informiert werden. Es besteht also eine asynchrone Interaktion zwischen dem Benutzer und dem Dienst. Da proaktive Dienste automatisch anhand von Positionsangaben gestartet werden, muss die Position kontinuierlich aktualisiert werden.

Kapitel 3

Konzept

Im vorherigen Kapitel wurden die Grundlagen dieser Arbeit genauer beschrieben. In diesem Kapitel wird auf den Grundlagen aufbauend eine Lösung vorgeschlagen, um das im Abschnitt 1.1 vorgestellte System zu realisieren. Bei dem System handelt es sich um ein Augmented-Reality-System auf einem Android-Smartphone, welches Interaktion mit POI sowie Navigationsfunktionen ermöglicht. Der erste Abschnitt beschreibt die Problemstellung sowie die Anforderungen dieses Systems. Anschließend wird auf die Architektur eingegangen und zuletzt werden die einzelnen Komponenten im Detail erläutert.

3.1 Problemstellung und Anforderungen

Die im vorherigen Kapitel vorgestellten Grundlagen zeigen, dass schon einige AR-Systeme auf Android-Smartphones existieren. Auch gibt es schon Ansätze für Navigationssysteme mittels AR auf Smartphones, welche jedoch nur für den Indoor-Bereich gedacht sind. Es besteht noch kein AR-Navigationssystem auf Android-Smartphones, welches sowohl im Indoor- als auch im Outdoor-Bereich genutzt werden kann. Außerdem existiert kein Gesamtsystem, welches die Interaktion mit POI, also eine Anbindung an standortbezogene Dienste, mit Navigation vereint. Für diese Probleme wurde eine Lösung erarbeitet, wobei das entwickelte AR-System zum einen eine Interaktion mit POI ermöglicht und zum anderen eine Navigation zwischen diesen bereitstellt. Zudem kann es nicht nur im Outdoor- sondern auch im Indoor-Bereich eingesetzt werden.

In Abschnitt 2.3 wurde gezeigt, dass sowohl auf Bilderkennung als auch auf Positionsdaten basierende AR-Systeme existieren. Auf Bilderkennung basierende AR-Systeme haben den Nachteil, dass sie auf Datenbanken, für die Mustererkennung, oder auf Marker angewiesen sind, was im Outdoor-Bereich nur schwer umsetzbar ist. Positionsbasierte AR-Systeme hingegen benötigen nur die Koordinaten, an denen die POI angezeigt werden sollen. Dies erleichtert die Umsetzung sowohl im Indoor- als auch im Outdoor-Bereich und macht das System leicht erweiterbar und flexibel. Zudem bietet ein positionsbasiertes System die Möglichkeit aus einer einstellbaren Distanz mit den POI zu interagieren. Das heißt, man kann bereits eine Interaktion durchführen, obwohl man

noch keinen direkten Sichtkontakt zu dem POI hat. Aufgrund der eben genannten Vorteile wurde die positionsbasierte AR als Lösung für das umgesetzte AR-System gewählt.
Ein wichtiges Kriterium für ein positionsbasiertes AR-System ist eine genaue Standortbestimmung. Im Outdoor-Bereich stellt diese wegen der GPS-Lokalisierung kein Problem dar. Im Indoor-Bereich wird die Standortbestimmung erschwert, da man keinen GPS-Empfang hat und dementsprechend auf andere Lokalisierungssysteme zurückgreifen muss. In Abschnitt 2.2.4 wurde bereits gezeigt, dass eine präzise Indoor-Lokalisierung mit einem Smartphone möglich ist, diese jedoch nicht standardmäßig implementiert ist. Um das Problem der Standortbestimmung im Indoor-Bereich zu beheben, wurde eine Anwendung auf dem Smartphone realisiert, die es ermöglicht die bestehenden Lokalisierungssysteme um neue Systeme zu erweitern. Die Funktionsweise dieser Anwendung wird im Abschnitt 3.3 genauer beschrieben.
Positionsbasierte AR-Systeme sind auf Smartphones standardmäßig als Augmented-Reality-Browser implementiert. Die bestehenden AR-Browser haben wie in Abschnitt 2.3.1 bereits erwähnt den Nachteil, dass sie ihre Standortinformationen nicht ausreichend oder überhaupt nicht aktualisieren und somit nicht für die Navigation genutzt werden können. Da die Navigation aber für das Gesamtsystem erforderlich ist und die bestehenden AR-Browser nicht ausreichend Freiraum für Weiterentwicklung bieten, um diese Funktionalität zum Beispiel nachzurüsten, wurde ein neuer AR-Browser erstellt. Dieser erfüllt die erforderlichen Eigenschaften und wird im Abschnitt 3.4 genauer beschrieben.
Die standortbezogenen Dienste, welche durch die Benutzerinteraktion mit POI aufgerufen werden, können durch die Entwicklung eines eigenen AR-Browsers flexibel und erweiterbar eingebunden werden. In Abschnitt 3.5 wird diese Struktur genauer beschrieben.

3.2 Architektur

Die Architektur des Systems gliedert sich in folgende drei Komponenten:

- Standortbestimmung
- Augmented-Reality-Browser
- Standortbezogene Dienste

Die Standortbestimmung und der Augmented-Reality-Browser sind als native Android-Anwendungen implementiert, da sie Zugriff auf Systemkomponenten, wie Beschleunigungssensoren, GPS und Kamera, benötigen. Die standortbezogenen Dienste hingegen können sowohl als native Anwendung als auch als Webseite implementiert werden, welches in Abschnitt 3.5 genauer beschrieben wird. Die Verbindungen der verschiedenen Komponenten werden in Abbildung 3.1 verdeutlicht. Der Augmented-Reality-Browser nutzt die Daten der Standortbestimmung, um die

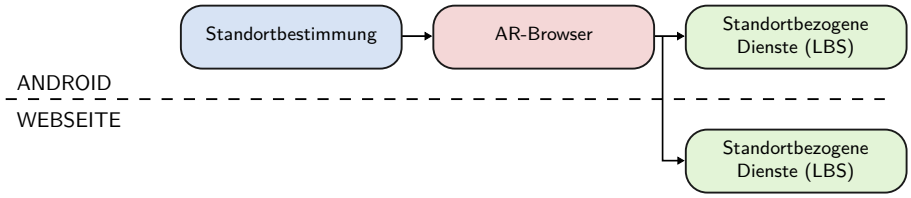

Abbildung 3.1: Architektur des AR-Systems

positionsabhängigen Objekte sowohl im Indoor- als auch im Outdoor-Bereich möglichst genau darstellen zu können. Zudem stellt der AR-Browser, durch die Benutzerinteraktion mit den POI, die Verbindung zu den standortbezogenen Diensten (LBS) her.
Um das System so flexibel und erweiterbar wie möglich zu gestalten, wurde es modular aufgebaut. Die Komponenten sind also unabhängig voneinander. Diese Struktur hat den Vorteil, dass die Komponenten wiederverwendbar sind. So kann zum Beispiel die Lokalisierungsanwendung nicht nur von dem AR-Browser, sondern auch von anderen Anwendungen verwendet werden. Auch kann der AR-Browser unabhängig von der Lokalisierungsanwendung genutzt werden, wobei er dann die Standortinformationen vom GPS-Modul bezieht. Zusätzlich bietet dieser Aufbau die Möglichkeit verschiedene Komponenten später ohne größeren Aufwand zu erweitern oder sogar auszutauschen.

3.3 Standortbestimmung

Die Standortbestimmung wurde als native und unabhängige Anwendung realisiert. Wie bereits erwähnt dient sie dazu, die bereits vorhandenen Lokalisierungssysteme um neue zu ergänzen, um so die Standortbestimmung auch im Indoor-Bereich zu ermöglichen.

3.3.1 Eigenschaften

Um die bestehenden Komponenten des Android-Betriebssystems gut einzubinden und die vorhandenen Strukturen weiterhin nutzen zu können, wurde die Anwendung so entwickelt, dass neue Lokalisierungssysteme die gleiche Struktur wie die bereits vorhandenen aufweisen. Die Lokalisierungssysteme des Android-Betriebssystems werden Location-Provider genannt. Android bietet eine Struktur, mittels derer man sich mit bestimmten Location-Provider verbinden kann und damit Positionsangaben von diesen erhält. Es besteht zudem die Möglichkeit neue Location-Provider hinzuzufügen, welche dann im System als Test-Provider geführt werden. Diese unterscheiden sich in ihrer Funktionsweise jedoch nicht von den bereits vorhandenen Location-Providern.
Die Anwendung unterstützt sowohl diskrete als auch kontinuierliche Lokalisierungssysteme. Diskre-

te Lokalisierungssysteme werden durch eine Android-Activity realisiert, da diese auf kurze Aktionen ausgelegt sind und eine Benutzerinteraktion ermöglichen. Ein Beispiel hierfür ist das Scannen eines Markers. Kontinuierliche Lokalisierungssysteme werden durch einen Android-Service ermöglicht. So besteht die Möglichkeit, die Lokalisierungsaufgabe im Hintergrund weiter auszuführen, auch wenn die Anwendung gerade nicht geöffnet ist. Ein Beispiel hierfür ist das kontinuierliche Messen von Beschleunigungs- und Richtungssensoren, was für die Realisierung eines Pedometers benötigt wird.

3.3.2 Funktionsweise

Die Funktionsweise der Anwendung wird in Abbildung 3.2 gezeigt. Die Lokalisierungsdaten der verschiedenen Location-Provider werden gesammelt und unter einem neuen Location-Provider, welcher fortan Haupt-Location-Provider genannt wird, im System verbreitet. Der Netzwerk-Provider entspricht der Android-Implementierung von Mobilfunk- und WiFi-Lokalisierung. Die Anwendung trifft durch eine Kombination aus Genauigkeit und Aktualität eine Auswahl, welche Positionen durch den Haupt-Location-Provider verteilt werden. Eine neue Position wird nur zugelassen, wenn diese genauer oder ungefähr gleich genau ist wie die zuletzt erhaltene Position. Wenn jedoch eine längere Zeit keine Position mehr zugelassen wurde, die diese Kriterien erfüllt, werden auch schlechtere Positionen angenommen. Verfügt die Anwendung über keine letzte Position, wird die erste erhaltene Position angenommen.

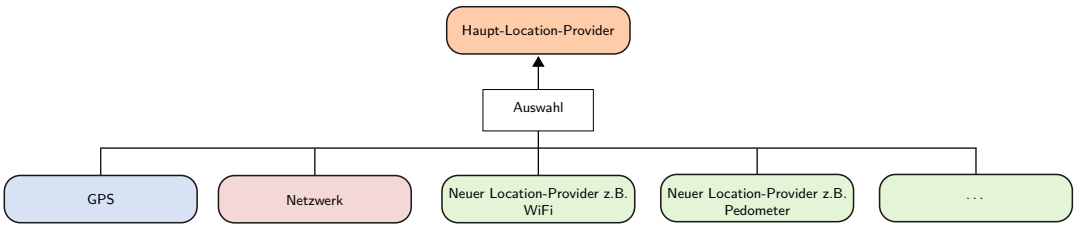

Abbildung 3.2: Funktionsweise der Standortbestimmung

Die Anwendung unterstützt auch Koppelnavigationssysteme (Dead Reckoning), wie zum Beispiel einen Pedometer. Um aus solchen Systemen Positionen zu erhalten, werden Referenzpunkte, wie zum Beispiel GPS-Positionsangaben, benötigt. Es kann für jeden neuen Location-Provider angegeben werden, ob es sich um eine Referenz handelt oder nicht. Erhält die Anwendung eine Positionsaktualisierung von einem Provider, wird überprüft, ob es sich hierbei um eine Referenz handelt. Ist dies der Fall, wird die Position durch den Referenz-Location-Provider im System verteilt. Um die Verteilung systemweit durchzuführen, ist der Referenz-Location-Provider als Android-Location-Provider realisiert. Die Koppelnavigationssysteme definieren selbst die geforderte Genauigkeit der Referenz-Positionen. Abbildung 3.3 zeigt die Funktionsweise der Koppelnavigation.

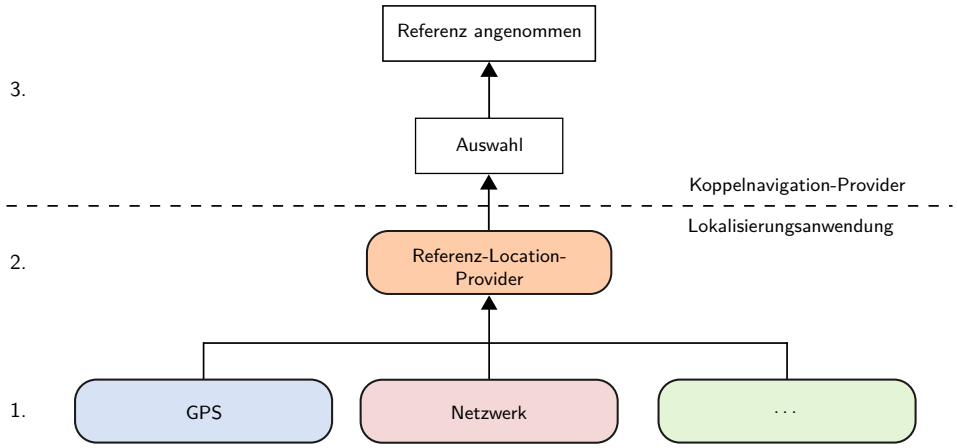

Abbildung 3.3: Funktionsweise der Koppelnavigation

Die Koppelnavigation durchläuft nachfolgende Punkte, welche auch in Abbildung 3.3 zu sehen sind.

1. Die Referenzen werden ausgewählt.
2. Die Referenzen werden im System durch den Referenz-Location-Provider verteilt.
3. Jeder Koppelnavigation-Provider trifft die Auswahl, ob er die Referenz annimmt oder nicht. Hier kann zum Beispiel ähnlich wie bei der Auswahl des Haupt-Location-Providers vorgegangen werden.

Diese ersten beiden Punkte sind allgemein in der Anwendung vorhanden. Der dritte Punkt hingegen ist spezifisch für den implementierten Koppelnavigation-Provider, zum Beispiel einen Pedometer. Hat der Provider eine Referenz angenommen, kann dieser nun relative Positionen in absolute Positionen umrechnen. Die Genauigkeit der errechneten Positionen kann nicht besser sein als die Genauigkeit der Referenz, da der Fehler des Koppelnavigation-Providers zu dem Fehler der benutzten Referenz addiert wird. Die errechneten absoluten Positionen können dann vom Provider im System verteilt werden. Diese Positionen werden, wie in Abbildung 3.2 gezeigt wurde, von der Anwendung ausgewertet und bei erfüllten Auswahlkriterien durch den Haupt-Location-Provider verteilt.

3.3.3 Schnittstellen

Es bestehen verschiedene Anbindungsmöglichkeiten von fremden Anwendungen an die Lokalisierungsanwendung. Diese können in Eingangs- und Ausgangsschnittstelle unterteilt werden.

Eingangsschnittstelle Die Eingangsschnittstelle erhält die Positionsdaten, die von fremden Anwendungen an die Lokalisierungsanwendung übertragen werden. Dies bietet den Vorteil, dass fremde Anwendungen Positionsdaten sammeln können und diese mit den in der Lokalisierungsanwendung vorhandenen Location-Providern verbinden können. Diese externen Positionsdaten können so in den Haupt-Location-Provider einfließen. Eine solche Verbindung wird über einen Bound-Service, wie er in Abschnitt 2.1.2 beschrieben wurde, realisiert. Die empfangenen Positionsangaben werden unter einem eigenen Location-Provider für externe Anwendungen an die Auswahl für den Haupt-Location-Provider übermittelt. Dies geschieht also nach dem gleichen Prinzip wie die Übermittlung der Positionsdaten der in der Lokalisierungsanwendung vorhandenen Location-Provider. Diese Funktionsweise wurde in Abbildung 3.2 dargestellt.

Ausgangsschnittstelle Die Ausgangsschnittstelle ermöglicht fremden Anwendungen den Empfang von Positionsangaben aus der Lokalisierungsanwendung. Hierfür wird die Location-Provider-Struktur genutzt, die im ganzen Android-System jeder Applikation zur Verfügung steht. Im Allgemeinen verbindet man die Anwendung, die Positionsangaben erhalten soll, mit dem Haupt-Location-Provider, welcher die genauesten Positionsangaben aus den in der Lokalisierungsanwendung vorhandenen Providern enthält. Es besteht auch die Möglichkeit sich mit einzelnen Providern, wie zum Beispiel dem WiFi-Provider, zu verbinden. Eine solche Verbindung ist möglich, da alle neuen Lokalisierungssysteme als Android-Location-Provider zur Verfügung stehen. Bereits bestehende Anwendungen, die ihre Positionsangaben standardmäßig vom GPS-Provider beziehen, können nicht problemlos auf den Haupt-Location-Provider zugreifen. Aus diesem Grund besteht zusätzlich die Möglichkeit die Positionsangaben des Haupt-Location-Providers über den GPS-Provider zu senden, indem die GPS-Daten mit den Haupt-Location-Provider-Daten überschrieben werden. Das Verfahren hat den Vorteil, dass jede Applikation ohne Änderungen im Indoor-Bereich genutzt werden könnte, da ihnen nun der Haupt-Location-Provider zur Verfügung steht. Der Nachteil ist, dass man keinen Zugriff mehr auf die richtigen GPS-Daten hat und diese deshalb nicht mehr in den Haupt-Location-Provider einfließen können.

3.4 Augmented Reality

Die Augmented-Reality-Komponente ist wie bereits erwähnt als AR-Browser realisiert und ist somit eine native Android-Anwendung. Da die vorhandenen AR-Browser nicht die gewünschten Funktionalitäten wie Navigation und Indoor-Tauglichkeit besitzen, wurde ein neuer AR-Browser entwickelt. Außerdem kann so die Benutzerschnittstelle an die gewünschten Funktionen angepasst werden.

3.4.1 Eigenschaften

Um eine standortbezogene AR zu ermöglichen, wird eine präzise Lokalisierung benötigt. Aus diesem Grund bezieht der AR-Browser die Positionsangaben vom Haupt-Location-Provider, welcher von der Lokalisierungsanwendung zur Verfügung gestellt wird. Ist dieser nicht verfügbar, benutzt der AR-Browser den GPS-Provider, was jedoch zu Einschränkungen im Indoor-Bereich führt.
Der AR-Browser bietet zudem die Möglichkeit, während der Benutzung Positionsangaben aus Markern zu scannen. Diese werden dann über die bereits erwähnte Schnittstelle an die Lokalisierungsanwendung weitergeleitet.
Ein weitere Eigenschaft des AR-Browsers ist die Möglichkeit mehrere Provider gleichzeitig darzustellen. Im Abschnitt 2.3.1 wurde bereits erwähnt, dass POI in Provider eingeteilt sind. Die vorhandenen Browser können jedoch nur einen Provider gleichzeitig darstellen. Durch die Entwicklung des neuen AR-Browsers wurde auch dieser Nachteil beseitigt. Die POI und Provider-Daten werden von einem Server bezogen, wie es auch bei den bestehenden AR-Browsern der Fall ist. Wie genau die Kommunikation mit dem Server aussieht, wird in nachfolgenden Abschnitten genauer beschrieben. Da die Inhalte von einem Server bezogen werden, wird der AR-Browser im Online-Modus verwendet, wofür er fortgehend über eine Internetverbindung verfügen muss. Außerdem verfügt der neue AR-Browser, wie auch die vorhandenen, über eine Karten- und Listenansicht. Die Listenansicht zeigt die in der Umgebung liegenden POI in Form einer Liste. Die Kartenansicht zeigt die umliegenden POI auf einer Karte. Die Kartenansicht verfügt über spezielle Erweiterungen für den Indoor-Bereich, welche in Abschnitt 3.4.4 genauer erläutert werden.
Die Hauptfunktionen, die Benutzerinteraktion mit POI sowie die Navigation werden nachfolgend im Detail beschrieben.

3.4.2 Interaktion mit POI

Dieser Abschnitt beschreibt, wie mit Hilfe des AR-Browsers POI dargestellt werden können und mit diesen interagiert werden kann. Zuerst wird auf die Funktionsweise des AR-Browsers, anschließend auf die Funktionsweise der Server-Komponente genauer eingegangen.

AR-Browser

Wie eine einfache Form der Interaktion mit einem POI aussieht, wird in dem Aktivitätsdiagramm 3.4 verdeutlicht.

Beim Starten des AR-Browsers wird geprüft, ob eine Internetverbindung zur Verfügung steht. Ist dies der Fall, kann die Verwendung fortgesetzt werden, andernfalls muss das Programm verlassen werden. Nach dem Start des AR-Browsers wählt der Benutzer die gewünschten Provider aus

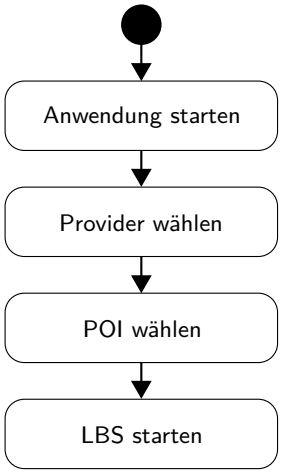

Abbildung 3.4: Aktivitätsdiagramm der POI-Interaktion

einer Liste aus. Die Provider werden bereits beim Start der Anwendung vom Server bezogen. Die Provider-Daten können dann später noch manuell aktualisiert werden. Durch die Auswahl der Provider wird bestimmt, welche POI heruntergeladen werden. Verfügt der AR-Browser über Standortinformationen, wird der Download der POI-Daten gestartet. Andernfalls wartet die Anwendung, bis sie einen Standort erhält. Sind die POI-Daten heruntergeladen, kann der Benutzer einen gewünschten POI auswählen. Er erhält bereits einige Informationen über den POI, wie den Namen und eine kurze Beschreibung. Handelt es sich um den gesuchten POI, kann der Benutzer die standortbezogenen Dienste (LBS) starten.
Aktualisierungen des Standortes lösen auch Aktualisierungen der POI-Daten aus. Es muss jedoch ein Mindestabstand zu der letzten Aktualisierungsstelle überschritten werden, um unnötige Übertragungen zu vermeiden.

Server

Die Server-Struktur ist ähnlich aufgebaut wie in Abschnitt 2.3.1 beschrieben, wobei der AR-Server und der POI-Server eine Komponente bilden. Für die Interaktion mit POI werden folgende zwei Funktionen benötigt: Herunterladen von POI und Herunterladen von Providern. Als Übertragungsformat für die Daten wird ARML verwendet. In Abbildung 3.5 wird die Kommunikation zwischen dem AR-Browser und der Server-Komponente dargestellt.

In der Abbildung ist sowohl die Anfrage für Provider als auch für POI dargestellt. Zuerst müssen die Provider vom Server abgerufen werden. Anschließend muss der Benutzer die gewünschten Provider auswählen. Daraufhin werden dann die entsprechenden POI-Daten beim Server angefragt. Die Anfrage beinhaltet die Koordinaten des Benutzers (lat, lon, alt) sowie die ausgewählten Provider. Anhand der Positionsangaben des AR-Browsers kann der Server die umliegenden POI

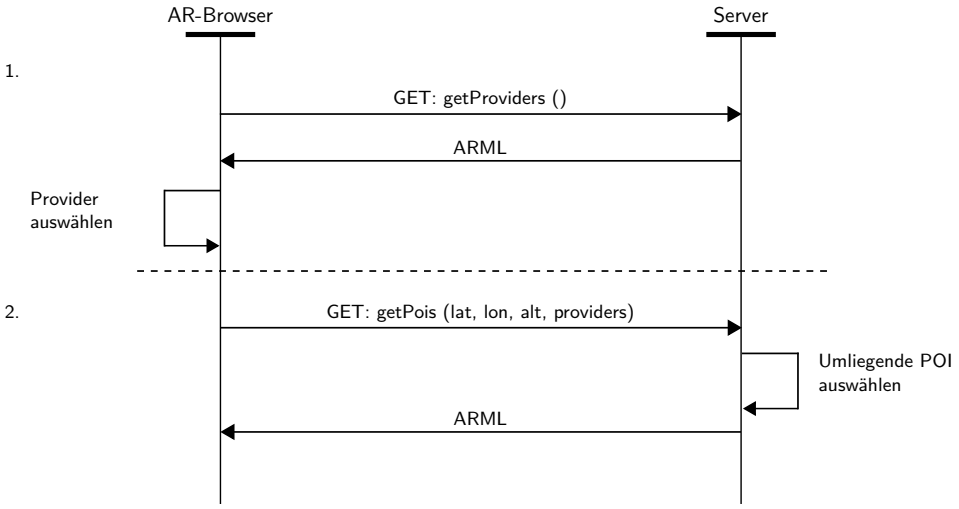

Abbildung 3.5: Kommunikation zwischen AR-Browser und Server bei der Interaktion mit POI, 1. Provider-Anfrage, 2. POI-Anfrage

auswählen, welche dann vom AR-Browser heruntergeladen werden. Weitere Aktualisierungen der POI-Daten werden, wie bereits erwähnt, durch Standortaktualisierungen initiiert, wobei jedoch nur die POI und nicht die Provider neu heruntergeladen werden. Diese Aktualisierungen entsprechen Wiederholungen des zweiten Teiles der Abbildung.

3.4.3 Navigation

Dieser Abschnitt beschreibt die Funktionsweise der Navigation. Es wird zuerst der AR-Browser und anschließend die Server-Komponente vorgestellt.

AR-Browser

Der Ablauf der Navigation im AR-Browser wird anhand des Aktivitätsdiagramms 3.6 gezeigt. Wie bereits bei der POI-Interaktion wird auch hier beim Starten des AR-Browsers geprüft, ob eine Internetverbindung zur Verfügung steht. Ist diese vorhanden, kann die Anwendung weiter genutzt werden. Der nächste Schritt dient der Eingabe der Navigationsinformationen wie zum Beispiel das Ziel der Navigation. Es besteht zudem die Möglichkeit der manuellen Eingabe des Startpunktes. Diese Eingaben entsprechen im Indoor-Bereich den vollständigen Namen oder Teilen der Namen der POI. Für Start- oder Zielpunkte, die sich im Outdoor-Bereich befinden, können Adressen oder Breiten- und Längengrade als Textwert angegeben werden. Wird kein Startpunkt eingegeben, werden die aktuellen Koordinaten als Startpunkt verwendet. Zusätzlich können weitere Optionen übergeben werden, wie zum Beispiel das Vermeiden von Treppen bei der Route. Falls

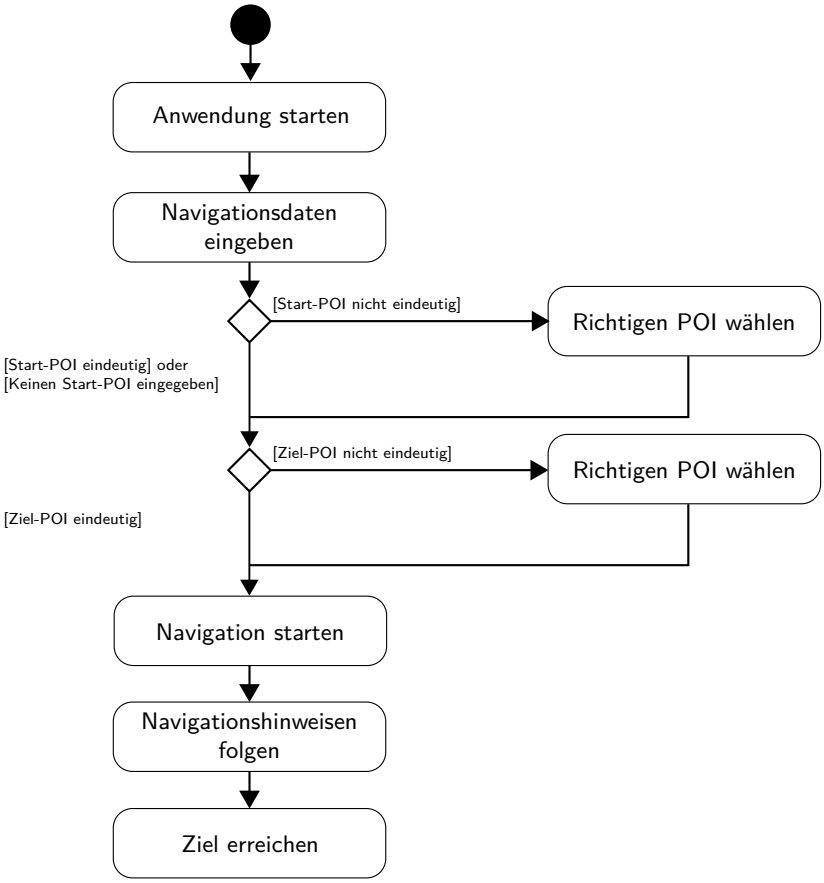

Abbildung 3.6: Aktivitätsdiagramm der Navigation

ein Startwert eingegeben wurde, wird zuerst vom Server überprüft, ob dieser eindeutig ist. Diese serverseitige Überprüfung wird im nachfolgenden Abschnitt genauer erklärt. Werden mehrere Resultate in der Datenbank gefunden, die auf die Benutzereingabe passen, werden diese zurückgeliefert und der Benutzer muss den richtigen POI auswählen. Das Ziel wird auf die gleiche Weise überprüft. Anschließend kann die Navigation gestartet werden. Hierbei folgt der Benutzer den Navigationshinweisen, bis er das Ziel erreicht.

Server

Die Server-Komponente dient der Bestimmung und Berechnung der Route. Um sowohl im Indoor- als auch im Outdoor-Bereich Routen ermitteln zu können, nutzt der Server einerseits eine interne Datenbank für den Indoor-Bereich, andererseits eine Verbindung zu Google-Maps für den Outdoor-Bereich. Google-Maps [45] kann auf HTTP-GET-Anfragen mit Navigationsangaben antworten, welche in der Keyhole Markup Language (KML) formatiert sind. KML [46] basiert auf XML und dient der Darstellung von geographischen Daten und Visualisierungen. Sie wurde von Google

weiterentwickelt und ist ein OGC-Standard. Da die Navigationsdaten von Google bereits im KML-Format übergeben werden, wird dieses Format auch für die Übertragung vom Server zum AR-Browser genutzt. Sowohl Indoor- als auch Outdoor-Navigationsdaten werden also in KML übertragen. Die Abbildung 3.7 zeigt die Kommunikation zwischen dem AR-Browser und dem Server. Zuerst werden die Start- und Zielangaben des Benutzers überprüft. Handelt es sich bei

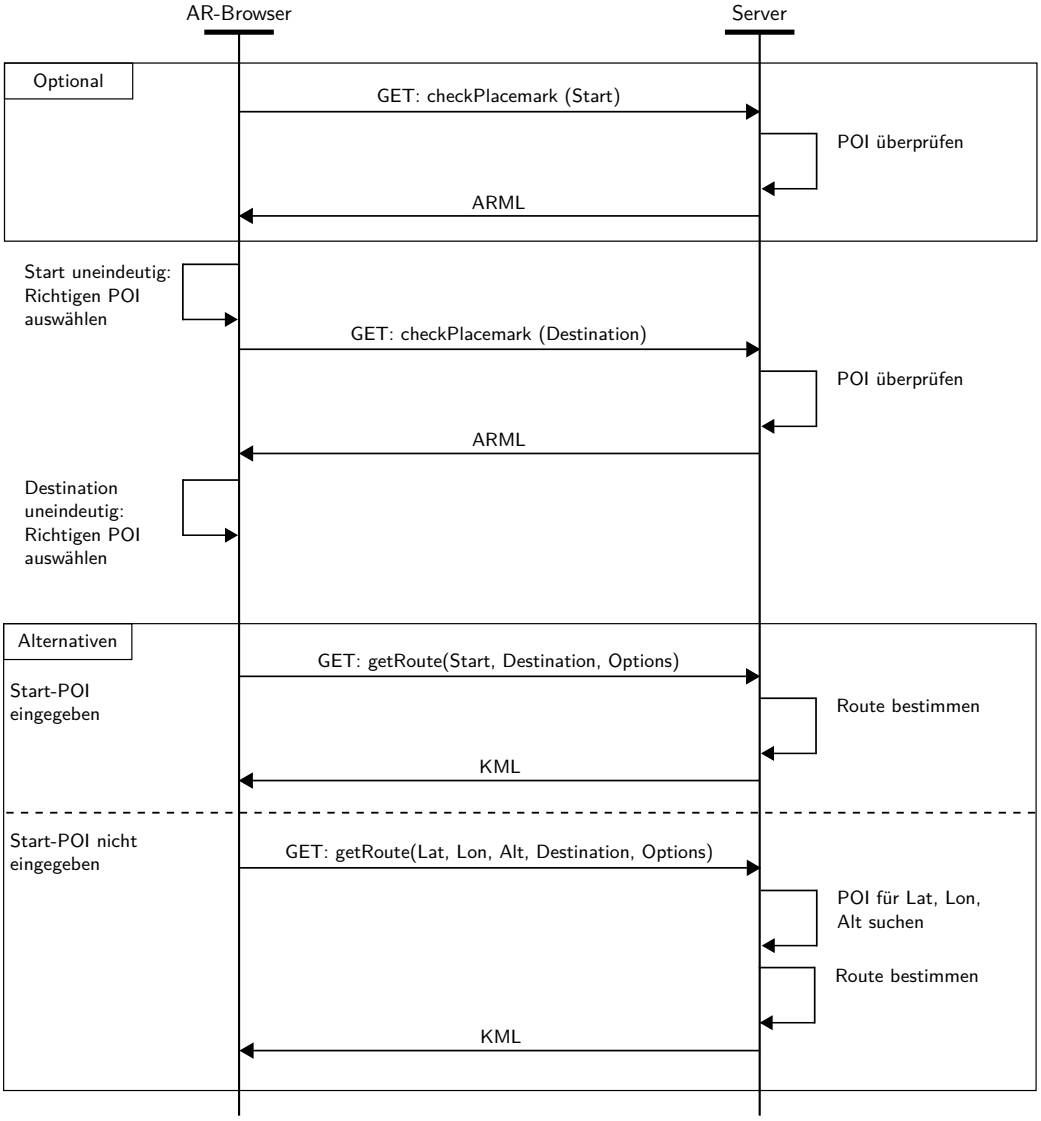

Abbildung 3.7: Kommunikation zwischen AR-Browser und Server bei der Navigation

diesen um POI-Namen, wird geprüft ob diese eindeutig einem POI in der Datenbank zugeordnet werden können. Das ist notwendig, da auch nur Teile des POI-Namens als Start oder Zielpunkt angegeben werden können. Falls ein Startpunkt eingegeben wurde, wird dieser an den Server übermittelt. Es wird überprüft, ob für die Eingabe ein eindeutiger Eintrag oder mehrere in der

Datenbank vorhanden sind. Werden mehrere POI gefunden, wird die Liste an den Benutzer zurückgegeben, welcher daraufhin den richtigen POI auswählen muss. Bei einem einzigen Eintrag wird dieser als richtig übernommen. Wird hingegen kein Eintrag in der Datenbank gefunden, handelt es sich um einen Startpunkt im Outdoor-Bereich und die Eingabe des Benutzers wird als solche übernommen. Die gleiche Prozedur wird für das Ziel wiederholt. Anschließend wird die Route beim Server angefragt. In der Abbildung werden zwei Alternativen aufgezeigt, zum einen die Anfrage mit einem manuell eingegebenen Startwert und zum anderen die Anfrage mit den aktuellen Koordinaten. Werden die Koordinaten an den Server übermittelt, sucht dieser in der Datenbank nach einem Navigationspunkt in der Umgebung. Wird ein Navigationspunkt gefunden, der nicht weiter als eine vordefinierte Maximaldistanz entfernt ist, wird dieser für die Navigation verwendet. Kann kein Navigationspunkt zugeordnet werden, werden die Koordinaten als Outdoor-Punkt übernommen. Anschließend kann die Route bestimmt werden. Dieser Schritt wird genauer im nachfolgenden Abschnitt beschrieben. Abschließend wird die ermittelte Route an den AR-Browser übertragen.

Routenbestimmung Der Server unterscheidet fünf verschiedene Fälle bei der Routenbestimmung:

1. Start und Ziel in der Server-Datenbank vorhanden und verbunden
2. Start nicht in der Server-Datenbank vorhanden
3. Ziel nicht in der Server-Datenbank vorhanden
4. Start und Ziel nicht in der Server-Datenbank vorhanden
5. Start und Ziel in der Server-Datenbank vorhanden, jedoch nicht verbunden

Die Routenberechnung zwischen Punkten, die in der Datenbank vorhanden sind, wird auf dem Server ausgeführt. Sind nicht alle Punkte in der Datenbank verfügbar, kann die Route nicht vollständig auf dem Server berechnet werden. Die fehlenden Routeninformationen werden beim Navigationsdienst von Google-Maps angefragt und gegebenenfalls an eine intern berechnete Route angefügt. Die interne Routenbestimmung wird mittels Dijkstra-Algorithmus durchgeführt. Im nachfolgenden Abschnitt wird der Verlauf der Routenbestimmung für die verschiedenen Fälle genauer beschrieben.

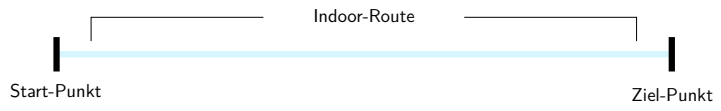

Abbildung 3.8: Fall 1 der Routenbestimmung: Indoor-Route

Fall 1 Dieser Fall beschreibt ein Szenario, bei dem man zum Beispiel von einem Raum des Gebäudes zu einem anderen navigieren will. Die Route zwischen dem Start-Navigationspunkt und dem Ziel-Navigationspunkt wird anhand des Dijkstra-Algorithmus bestimmt. Abbildung 3.8 zeigt die Route für diesen Fall.

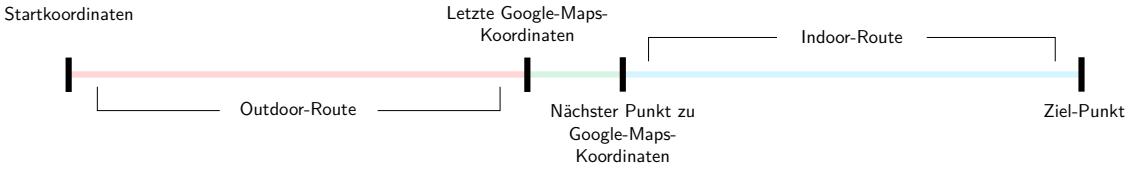

Abbildung 3.9: Fall 2 der Routenbestimmung: Outdoor-Route ⇒ Indoor-Route

Fall 2 Fall 2 tritt zum Beispiel ein, wenn man von einer bestimmten Adresse zu einem Raum in einem Gebäude navigieren will. Hierfür wird sowohl Outdoor- als auch Indoor-Navigation benötigt. Der Weg vom Startpunkt bis zum Gebäude wird von Google ermittelt. Die Koordinaten des Start- und des Zielpunktes werden an Google-Maps übertragen. Google-Maps sendet daraufhin die Navigationspunkte an den Server. Die an Google-Maps übertragenen Zielkoordinaten befinden sich in Fall 2 in einem Gebäude. Der letzte von Google-Maps übermittelte Navigationspunkt befindet sich jedoch auf der Straße vor dem Gebäude. Google-Maps kann allgemein nicht innerhalb eines Gebäudes navigieren, da es nicht über die notwendigen Informationen verfügt. Die Zielkoordinaten werden dann mit Hilfe der Indoor-Navigation erreicht. Diese nimmt die letzten von Google erhaltenen Koordinaten, also diejenigen, die sich vor dem Gebäude befinden und ermittelt den nächstgelegenen internen Navigationspunkt. Wird ein interner Navigationspunkt gefunden, wird von diesem Navigationspunkt aus bis zu dem Ziel-Navigationspunkt navigiert. Anschließend werden Indoor- und Outdoor-Route zusammengefügt, um so die gesamte Route zu erhalten. Abbildung 3.9 zeigt den Aufbau der gesamten Route.

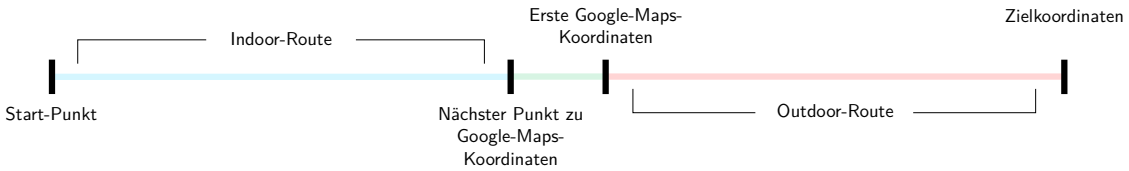

Abbildung 3.10: Fall 3 der Routenbestimmung: Indoor-Route ⇒ Outdoor-Route

Fall 3 Fall 3 beschreibt die umgekehrte Situation von Fall 2. Ein Beispiel hierfür ist die Navigation von einem Raum in einem Gebäude zu einer bestimmten Adresse. Auch hier wird sowohl Outdoor- als auch Indoor-Navigation benötigt. Die Start- und Zielkoordinaten werden an

Google-Maps übermittelt. Die ersten Google-Maps-Koordinaten befinden sich vor dem Gebäude. Anhand dieser Google-Maps-Koordinaten ermittelt der Server den nächstgelegenen internen Navigationspunkt, welcher dann das Ziel der Indoor-Navigation ist. Anschließend kann der Server die Indoor-Route ermitteln. Ist die Indoor-Route bestimmt, wird die Outdoor-Route angehängt, um so die gesamte Route zu erhalten. In der Abbildung 3.10 wird die Zusammensetzung der Route für Fall 3 dargestellt.

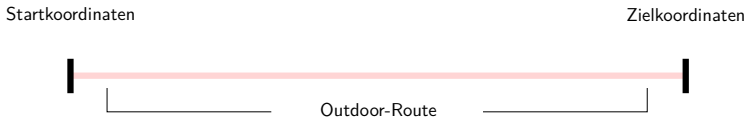

Abbildung 3.11: Fall 4 der Routenbestimmung: Outdoor-Route

Fall 4 Dieser Fall beschreibt ein Szenario, bei dem man zum Beispiel von einer bestimmten Adresse zu einer anderen navigieren will. Hier wird keine Indoor-Navigation benötigt. Das heißt, die Anfrage wird einfach an Google-Maps weitergeleitet. Die komplette Route wird also von Google-Maps bestimmt. Abbildung 3.11 zeigt die Route für diesen Fall.

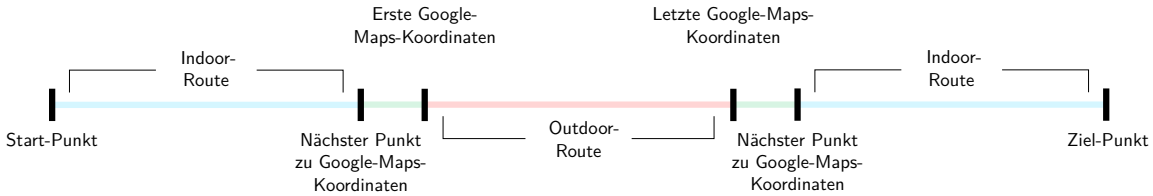

Abbildung 3.12: Fall 5 der Routenbestimmung: Indoor-Route ⇒ Outdoor-Route ⇒ Indoor-Route

Fall 5 Fall 5 beschreibt eine Situation, bei der man zum Beispiel von einem Raum eines Gebäudes bis zu einem Raum eines anderen Gebäudes navigieren will. Die Gebäude sind dabei nicht über die interne Datenbank verbunden. Dieser Fall tritt ein, wenn der Dijkstra-Algorithmus keine Route zwischen den internen Navigationspunkten findet. Dieses Szenario kombiniert den zweiten und dritten Fall. Zuerst wird, für die Start- und Zielkoordinaten, eine Route bei Google-Maps angefragt. Dann wird eine Indoor-Route wie in Fall 3 ermittelt, welche dann vor die Google-Maps-Route gesetzt wird. Anschließend wird eine Indoor-Route bis zum Ziel wie in Fall 2 berechnet. Diese wird hinter die Google-Maps-Route gesetzt, womit man dann die gesamte Route erhält. Die Zusammensetzung der Route wird in Abbildung 3.12 dargestellt.

3.4.4 Kartenansicht und Overlays

Der AR-Browser enthält wie bereits erwähnt als alternative Darstellungsform eine Kartenansicht. Diese basiert auf Google-Maps, sie wurde jedoch um die Funktionalität von Indoor-Karten erweitert. Die Indoor-Karten werden als sogenannte Overlays von einem Karten-Server bezogen. Anhand dieser Karten kann das Innere der Gebäude dargestellt werden. Auch die umliegenden POI werden bei der Kartenansicht dargestellt. Falls die Navigation gerade aktiviert ist, kann auch die Route in der Kartenansicht betrachtet werden.

Beziehen von Indoor-Karten Aktiviert man die Kartenansicht mit der Indoor-Option, wird automatisch eine Anfrage an den Server gerichtet. Die Anfrage beinhaltet den aktuellen Standort des Benutzers. Basierend auf dieser Position ermittelt der Server, ob er über eine Karte verfügt, die diese Position beinhaltet. Auch wird die Höhe der Position berücksichtigt, so dass die Indoor-Karte für das richtige Stockwerk ausgewählt werden kann. Findet der Server eine passende Karte, werden die Karteninformationen im KML-Format an den AR-Browser gesendet. Das KML-Format wurde für die Übertragung verwendet, da dieses über Strukturen für Overlays verfügt. Die empfangenen Karteninformationen enthalten die URL von welcher die Bilddatei der Karte bezogen werden kann. Außerdem enthalten sie die notwendigen Informationen um die Karte richtig auf der Google-Karte auszurichten.

3.5 Standortbezogene Dienste

Die standortbezogenen Dienste (LBS) können wie bereits erwähnt als native Android-Anwendung oder als Webseite umgesetzt werden. Nachfolgend werden die Vor- und Nachteile beider Möglichkeiten aufgelistet.

Android-Anwendung Ein Vorteil einer nativen Anwendung ist ihre Leistungsfähigkeit. Diese ist wesentlich höher bei einer für das Gerät eigens entwickelten Anwendung als bei einer Webseite. Ein weiterer Vorteil ist der mögliche Zugriff auf alle Systemressourcen. So können die LBS beispielsweise auf die Kamera zugreifen, was für eine Webseite nicht so einfach möglich ist.
Die nativen Anwendungen haben den Nachteil, dass sie manuell installiert werden müssen. Die Anwendung muss zum Beispiel aus dem Android-Market bezogen werden. Diese manuellen Installationen stellen einen zusätzlichen Aufwand für den Benutzer dar. Zudem haben native Anwendungen den Nachteil, dass sie unflexibel gegenüber verschiedenen Plattformen sind. Entwickelt man also eine Anwendung für das Android-Betriebssystem, so kann diese nicht auf Smartphones mit anderen Betriebssystemen verwendet werden.

Webseite Eine als Webseite realisierte LBS-Anwendung wird im Browser ausgeführt. Durch die Verwendung von HTML, CSS und JavaScript sehen die Layouts der Browser-Anwendungen denen der nativen Anwendungen ähnlich. Ein Vorteil von Webseiten ist, dass sie schnell aktualisiert werden können, ohne zum Beispiel durch den Android-Market gebremst zu werden. Zusätzlich sind sie auf jeder Plattform sofort verfügbar.
Der Nachteil der Webseite ist, dass sie langsamer als eine native Anwendung ist und dass sie nicht auf alle Ressourcen des Smartphones zugreifen kann.

Welche der vorgestellten Methoden zum Einsatz kommt, hängt vor allem von den zu realisierenden Diensten ab.
Um das gesamte AR-System so flexibel wie möglich zu gestalten, hat jeder Provider, also jede POI-Gruppe, eigene LBS. So kann zum Beispiel eine POI-Gruppe von Restaurants ohne Probleme andere Dienste anbieten, wie eine POI-Gruppe von Reiseinformationen. Dabei spielt es keine Rolle, ob die LBS als native Anwendung oder als Webseite realisiert sind. Die Verbindung des AR-Browsers zu den LBS wird durch eine URI bereitgestellt. Diese kann entweder zum Starten einer Android-Anwendung mit Hilfe eines Intents oder zum Aufrufen einer Webseite verwendet werden. Die Adresse der Webseite oder der Intent enthalten zudem die Information, welcher POI ausgewählt wurde. Die LBS können anhand dieser Identifikation auf diesen spezifischen POI reagieren.

Kapitel 4

Implementierung

Dieses Kapitel beschreibt die Implementierung des konzipierten Systems. Das System besteht aus einer Anwendung für die Standortbestimmung und einer Anwendung für die Augmented Reality. Die standortbezogenen Dienste wurden in dieser Implementierung als Webseite realisiert. Die implementierten Dienste umfassen Funktionen, welche zum Beispiel in einem Bürogebäude verwendet werden können.

Die Implementierung beschreibt eine Lösung, welche die Realisierbarkeit und Nutzbarkeit des konzipierten AR-Systems zeigt. Das AR-System kann sowohl im Indoor- als auch im Outdoor-Bereich genutzt werden und stellt die Benutzerinteraktion mit POI sowie die Navigation bereit.

Zuerst wird die verwendete Entwicklungsumgebung genauer beschrieben. Anschließend werden die beiden Android-Anwendungen näher erläutert. Hierbei wird sowohl auf den Aufbau der Anwendung als auch auf die verschiedenen Funktionen eingegangen. Anschließend wird die zugehörige Server-Komponente erklärt. Zuletzt werden die implementierten standortbezogenen Dienste beschrieben.

4.1 Entwicklungsumgebung

In diesem Abschnitt werden die verwendete Entwicklungsumgebung sowie die genutzten Frameworks vorgestellt. Zuerst werden die Android-Anwendungen und anschließend die Server-Komponente im Detail erläutert.

4.1.1 Android-Anwendungen

Die beiden Android-Anwendungen wurden in der Programmiersprache Java entwickelt. Für beide Anwendungen wurde das Google-API [47] für die Android-Version 2.3.3 verwendet. Es bietet im Vergleich zur normalen Android-SDK zusätzliche Funktionen.

Google-API Die Google-API ist eine Erweiterung des normalen Android-SDK. Es beinhaltet eine Reihe von Anwendungen, Bibliotheken und Diensten, die von Google zur Verfügung gestellt werden. Die Google-API umfasst außerdem Bibliotheken, um Google-Karten auf dem Android-Smartphone darstellen zu können. Da beide Anwendungen mit Kartenansichten ausgestattet werden, wird also die Google-API benötigt. Die Kartenbibliotheken stellen das Herunterladen, Rendern, Zwischenspeichern von Kartenabschnitten sowie einige Darstellungsoptionen und Steuerungen zur Verfügung.

Um die Karteninformationen von Google beziehen zu können, ist eine Registrierung für den Kartendienst erforderlich. Nach Einwilligung der Benutzerbedingungen erhält man einen Schlüssel (Maps-API-Key), welcher den Kartendienst in der Anwendung freischaltet.

Location-Provider-Framework Im Android-Betriebssystem bestehen bereits zwei Location-Provider: der GPS-Location-Provider und der Netzwerk-Location-Provider (Network-Location-Provider). Der GPS-Location-Provider nutzt für die Standortbestimmung den im Smartphone integrierten GPS-Empfänger. Steht eine Datenverbindung zur Verfügung, kann A-GPS vom GPS-Location-Provider verwendet werden. Der Netzwerk-Location-Provider beinhaltet eine Mobilfunkortung mittels Cell-Id sowie eine WiFi-Ortung mittels Fingerprinting. Für beide ist eine Internetverbindung erforderlich, da sie auf die Datenbanken von Google angewiesen sind.

4.1.2 Server-Komponenten

Die Server-Komponenten wurden in der Programmiersprache Java entwickelt. Es wurde das SDK der Java Enterprise Edition (Java EE) 6 verwendet. Der Server ist ein Apache-Tomcat-Server in der Version 7. Dieser implementiert die Servlets- und die JSP-Technologien, welche für die Realisierung der AR-Server-Komponente genutzt wurden. Der AR-Server wurde als Servlet realisiert. Servlets sind Java-Programme, die dynamisch Webseiten erstellen und auf der Serverseite ausgeführt werden. Die Kommunikation mit diesen Programmen erfolgt, wie bei einer normalen Webseite, über HTTP-Anfragen. Für die Datenbank, welche unter anderem die AR-Inhalte bereitstellt, wurde eine MySQL-Datenbank verwendet. Diese wird mit Hilfe der Java Database Connectivity (JDBC) an die Java-Infrastruktur des Servers angebunden.

Webseiten Die Webseiten, die zum Beispiel für die standortbezogenen Dienste wie auch für die Administration implementiert wurden, basieren auf JSP. JSP sind im Gegensatz zu Servlets keine Java-Klassen mit HTML-Ausgaben, sondern HTML-Seiten mit eingebettetem Java-Code. Für die Implementierung wurde das quelloffene Struts-Web-Framework von Apache [48] in der Version 2 verwendet. Dies bietet den Vorteil, dass die Web-Anwendung nach der MVC-Architektur aufgebaut werden kann. Zudem wurde das Tiles-Plugin für Struts verwendet, welches einen

komponentenbasierten Aufbau der Webseiten ermöglicht. Das ergibt den Vorteil, dass Header, Footer und Inhalt in verschiedene Dateien aufgeteilt werden können. Diese Aufteilung verhindert zum Beispiel unnötiges Kopieren des Header-Codes in jede Datei.

Für die Webseite der standortbezogenen Dienste wurde das jQuery-Mobile-Framework [49] verwendet. Dieses Framework ermöglicht es, der Webseite ein ähnliches Layout wie das einer nativen Anwendung zu geben.

4.2 Lokalisierungsanwendung

In diesem Abschnitt wird die Lokalisierungsanwendung, die es ermöglicht neue Location-Provider hinzuzufügen, genauer erklärt. Zuerst wird der Aufbau der Anwendung erläutert. Anschließend werden die implementierten Location-Provider vorgestellt. Zuletzt wird gezeigt, wie ein neuer Location-Provider hinzugefügt werden kann.
Damit die Anwendung auf die erforderlichen Systemressourcen zugreifen kann, muss die Manifest-Datei der Anwendung folgende Berechtigungen enthalten:

- `ACCESS COARSE LOCATION`: Diese Berechtigung ermöglicht den Zugriff auf den Netzwerk-Location-Provider. Dieser dient der groben Standortbestimmung mittels Mobilfunk (Cell-ID) und WiFi.
- `ACCESS FINE LOCATION`: Diese Berechtigung erlaubt den Zugriff auf den GPS-Location-Provider, welcher eine genauere Standortbestimmung ermöglicht.
- `ACCESS MOCK LOCATION`: Diese Berechtigung ermöglicht das Hinzufügen von neuen Location-Providern. Zusätzlich muss in den Einstellungen des Android-Betriebssystems der Parameter „Pseudostandorte zulassen“ unter dem Menüpunkt Anwendungen-Entwicklung aktiviert werden.
- `INTERNET`: Die Internet-Berechtigung ist notwendig, um die Karten der Kartenansicht herunterzuladen.

Es wird gleich beim Start der Anwendung geprüft, ob die Eigenschaft „Pseudostandorte zulassen“ in den Systemeinstellungen aktiv ist. Diese Einstellung ist für die Registrierung neuer Provider im System erforderlich. Falls diese nicht aktiviert ist, hat der Benutzer die Möglichkeit dies nachzuholen oder die Anwendung zu verlassen.
Um die Benutzerfreundlichkeit zu erhöhen wurde die Lokalisierungsanwendung in deutscher und englischer Sprache realisiert. Die Sprache wird automatisch anhand der Betriebssystem-Sprache bestimmt.

4.2.1 Aufbau

Die Anwendung gliedert sich in drei Tabs (Reiter), welche in Abbildung 4.1 zu sehen sind und nachfolgend beschrieben werden.

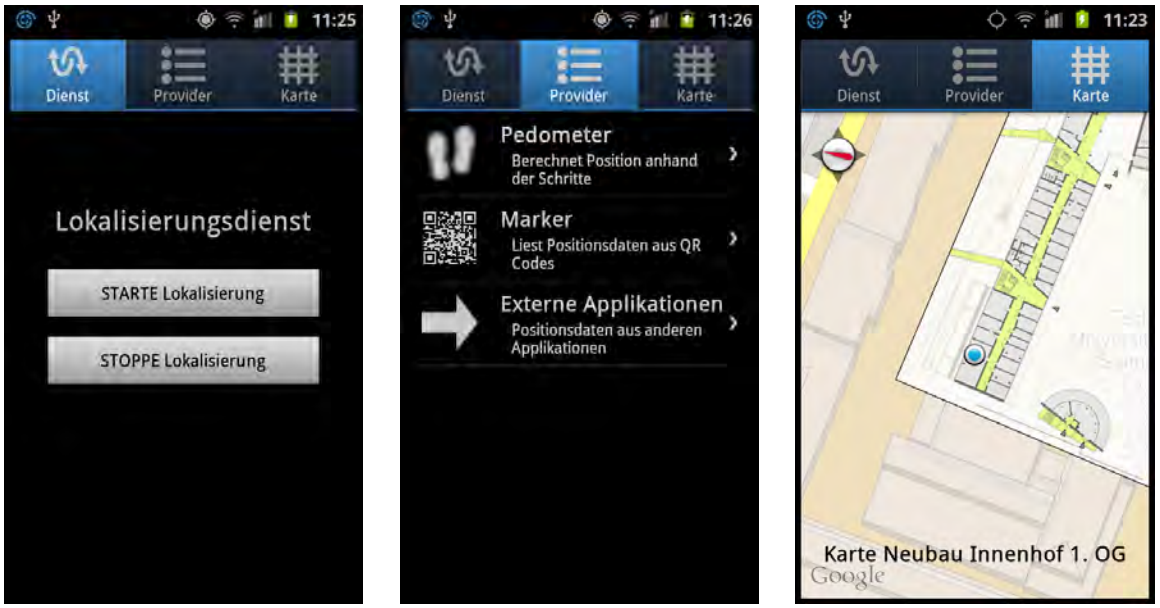

Abbildung 4.1: Übersicht der Lokalisierungsanwendung,
Lokalisierungsdienst (links), Location-Provider (Mitte), Kartenansicht (rechts)

Lokalisierungsdienst In diesem Tab kann der Lokalisierungsdienst gestartet und gestoppt werden. Durch den Start des Lokalisierungsdienstes werden alle implementierten und aktivierten Provider gestartet. Zusätzlich werden die neuen Location-Provider im System als Android-Location-Provider registriert. Durch Stoppen des Dienstes werden alle Provider gestoppt und die neuen Location-Provider wieder aus dem System entfernt. In der Status-Bar wird angezeigt, ob der Dienst gerade aktiviert ist oder nicht.

Location-Provider Der Provider-Tab beinhaltet eine Auflistung der implementierten Provider. Der Benutzer hat die Möglichkeit die implementierten Location-Provider in den Einstellungen zu aktivieren oder zu deaktivieren. Der Aktivierungsstatus jedes Providers sowie sein Name und eine kurze Beschreibung sind in der Auflistung sichtbar. Durch Anklicken eines Providers können zusätzliche Informationen oder Funktionen dieses Providers aufgerufen werden. Die implementierten Provider werden in Abschnitt 4.2.2 genauer beschrieben.

Kartenansicht Anhand der Kartenansicht kann die aktuelle Position verfolgt werden. Die Kartenansicht wurde wie bereits in Abschnitt 3.4.4 erwähnt durch Indoor-Karten erweitert.

Das heißt, die Anwendung fragt beim Karten-Server an, ob es für die aktuelle Position eine Indoor-Karte gibt. Falls eine solche zur Verfügung steht, wird diese heruntergeladen und als Overlay auf der Google-Karte angezeigt. In der Kartenansicht in Abbildung 4.1 wird eine solche Indoor-Karte eingeblendet. Zusätzlich dazu wird auch der Name der eingeblendeten Karte angezeigt. Der Server, der die Karten-Informationen enthält, kann in den Einstellungen vom Benutzer angegeben werden. Der Karten-Server wird in Abschnitt 4.4.3 im Detail beschrieben.

4.2.2 Implementierte Location-Provider

In diesem Abschnitt werden die im Rahmen dieser Arbeit entwickelten Location-Provider vorgestellt. Die implementierten Location-Provider sind jedoch nur Beispiele, welche um andere Location-Provider erweitert werden können. Es wurde ein Pedometer-Provider sowie ein Marker-Provider realisiert. Die in Abschnitt 3.3.3 erwähnte Schnittstelle zum Empfang von Positionsangaben fremder Anwendungen wurde ebenfalls als Location-Provider realisiert. Die implementierten Provider dienen zum Testen des Gesamtsystems. Sie wurden so ausgewählt, dass sie das Problem der Standortbestimmung im Indoor-Bereich beheben. Zusätzlich wurde darauf geachtet, dass für diese realisierten Provider wenig neue Infrastrukturen benötigt werden. So benötigt nur der Marker-Provider neue Infrastruktur, nämlich die zu scannenden Marker. Diese können jedoch auf Papier gedruckt werden, so dass diese Infrastruktur ohne großen Aufwand aufgebaut werden kann.

Die implementierten Location-Provider bilden also ein hybrides Lokalisierungssystem aus Markererkennung und Pedometer. Die Marker versorgen die Lokalisierungsanwendung mit präzisen Standortangaben. Der Pedometer hingegen dient der Standortbestimmung zwischen den Markern. Dadurch wird eine kontinuierliche Standortbestimmung ermöglicht. Je genauer der Pedometer arbeitet, desto weniger Marker werden für eine präzise Indoor-Lokalisierung benötigt.

Pedometer

Mit Hilfe des entwickelten Pedometers kann von einer Referenz-Position aus eine neue Position bestimmt werden. Dazu werden sowohl die Anzahl der Schritte als auch deren Richtung ermittelt. Nach jedem Schritt wird die neu berechnete Position durch den Pedometer-Provider verteilt.

Um eine neue Position bestimmen zu können, braucht der Pedometer zuerst eine Referenz. Dabei nimmt er nur die Referenzen an, welche die gleiche oder eine höhere Genauigkeit als die vorherige besitzen. Besitzt der Dienst noch keine Referenz, nimmt er die erste an, die er bekommt. Sobald er eine Referenz angenommen hat, kann er anhand dieser eine neue Position berechnen. Dabei merkt sich der Dienst die zurückgelegten Distanzen nach Norden sowie nach Osten. Eine zurückgelegte

Distanz nach Westen wäre in diesem Fall einfach ein negativer Wert der Distanz nach Osten und eine Distanz nach Süden ein negativer Wert der Distanz nach Norden. Zusätzlich speichert der Dienst die Anzahl gelaufener Schritte sowie die benutzte Referenz.

Der Pedometer reagiert sowohl auf den Kompass als auch auf die Beschleunigungssensoren. Er kann so die aktuelle Richtung sowie die gelaufenen Schritte erfassen. Der entwickelte Algorithmus zur Bestimmung eines Schrittes orientiert sich an dem von Bagi [50] vorgeschlagenen Lösungsansatz. Zuerst wird die Summe aus den X,Y,Z-Werten der Beschleunigungssensoren gebildet. Diese Summe wird mit den vorherigen Messwerten verglichen. Es wird geprüft, ob die aktuelle Summe ein Maximum oder Minimum ist. Ein Maximum oder Minimum kann durch einen Richtungswechsel der Beschleunigungswerte ermittelt werden. Hierfür wird die Richtung der aktuellen Summe mit der Richtung der vorherigen verglichen. Es handelt sich zum Beispiel um ein Maximum, wenn der vorherige Wert eine positive Richtung und der aktuelle eine negative hat. Wenn nun die Differenz zwischen dem gefundenen Maximum und Minimum größer als ein Schwellwert ist, handelt es sich um einen Schritt. Um die Schritte jedoch nicht doppelt zu zählen, wird ein Schritt nur gezählt, wenn es sich bei dem aktuellen Extremum um ein Maximum handelt. Dieses System kann noch verbessert werden, indem zum Beispiel eine automatische Bestimmung des Schwellwertes stattfindet. Momentan wird der Schwellwert und damit die Empfindlichkeit des Sensors vom Benutzer selbst festgelegt.

Nachdem ein Schritt erkannt wurde, wird die aktuelle Kompassrichtung diesem zugeordnet. Bei der Erfassung der Kompassrichtung werden die gemessenen Richtungswerte gefiltert. Dabei wird der Mittelwert mehrerer Messungen erzeugt, um eine stabilere Richtungsangabe zu erhalten. Zudem wird die Kompassrichtung mit Hilfe der Standortinformationen, also Breiten- und Längengrad, vom magnetischen Norden zum geographischen Norden umgerechnet. Diese Umrechnung ermöglicht eine genauere Richtungsangabe. Doch bevor die Richtung als solche angenommen wird, wird ermittelt, ob sich das Smartphone im Quer- oder im Hochformat befindet. Der Grund hierfür ist, dass die ermittelte Richtung auf das Hochformat des Smartphones bezogen ist. Befindet sich das Smartphone jedoch im Querformat, muss diese umgerechnet werden, was einer Addition von 90 Grad entspricht. Die Orientierung kann entweder automatisch anhand der Lagesensoren des Smartphones ermittelt werden oder durch den Benutzer manuell angegeben werden. Die Bestimmung der Orientierung ist erforderlich, da fremde Anwendungen sowohl im Hochformat als auch im Querformat verwendet werden können.

Ist die Richtung bestimmt, kann der gefundene Schritt in seinen Nord- und Ost-Anteil zerlegt werden. So kann mit Hilfe der Schrittlänge ermittelt werden, welche Distanz in welche Richtung zurückgelegt wurde. Die Schrittlänge wird dabei aus Größe und Geschlecht des Benutzers errechnet. Nach dem Pedometer Hersteller Digiwalker [51] zufolge, kann nämlich die Schrittlänge anhand der Größe ungefähr errechnet werden. Dafür wird folgende Formel verwendet: $Schrittlänge = Größe * k$ mit k gleich 0,415 für Männer und 0,413 für Frauen. Diese Berech-

nung wurde gewählt, da die wenigsten Benutzer ihre Schrittlänge kennen. Der Benutzer muss also nur sein Geschlecht sowie seine Größe angeben. Eine manuelle Eingabe der Schrittlänge ist aber auch möglich.

Die Nord- und Ost-Distanzen werden bei jedem Schritt aufaddiert. Somit wird bei jedem Schritt die aktuelle Distanz zu dem Referenzpunkt erfasst. Mit Hilfe dieser Distanz kann eine neue Position berechnet werden. Dafür wird die summierte Nord- und Ost-Distanz nach dem WGS84-Modell auf die Referenz-Position addiert. Empfängt der Pedometer eine neue Referenz, werden die Nord- und Ost-Distanz sowie die Anzahl der Schritte zurück auf Null gesetzt. Die Genauigkeit der ermittelten Position setzt sich aus der Genauigkeit der Referenz-Position und der Anzahl der gelaufenen Schritte zusammen. Dafür wird ein Viertel der Schrittlänge für jeden gemachten Schritt auf die Genauigkeit der Referenz-Position addiert:

$$Genauigkeit_{[\mathrm{PedometerPosition}]} = 1/4 * Schrittlänge * Anzahl_{[\mathrm{Schritte}]} + Genauigkeit_{[\mathrm{ReferenzPosition}]}$$

Nachdem die neue Position nun bestimmt ist, kann sie durch den Pedometer-Location-Provider verteilt werden.

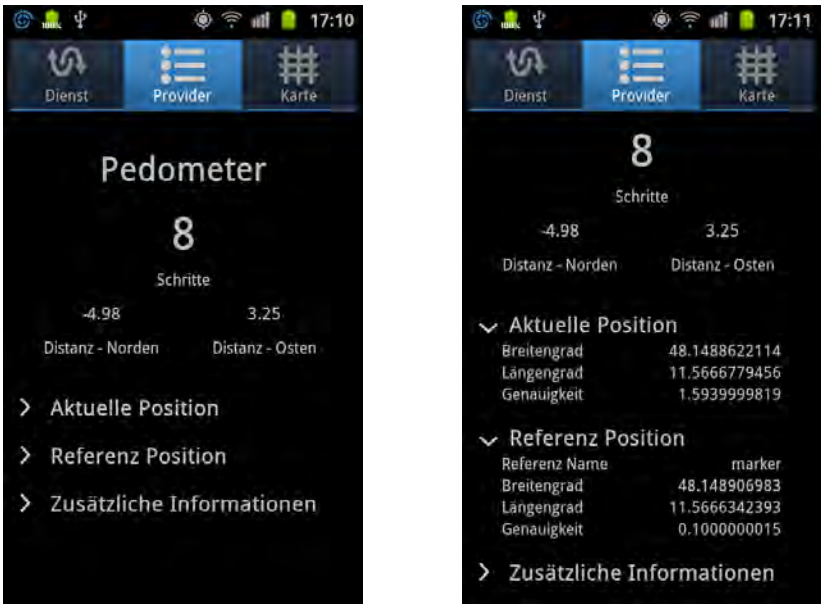

Abbildung 4.2: Pedometer-Provider

Der Pedometer benötigt keine Benutzerinteraktionen. Der Benutzer hat jedoch die Möglichkeit den Status des Pedometers abzufragen. Abbildung 4.2 zeigt die Informationen, die man erhält, wenn man in dem Provider-Tab den Pedometer-Provider anklickt. In dieser Ansicht sieht der Benutzer sofort, wie viele Schritte er seit der letzten Referenz gelaufen ist. Zusätzlich sieht er die zurückgelegte Distanz nach Norden und Osten in Metern. Die Koordinaten und die Genauig-

keit der aktuellen Position sowie auch der Referenzposition können ebenfalls betrachtet werden. Zusätzliche Informationen wie benutzte Schrittlänge und Orientierung stehen dem Benutzer auch zur Verfügung.

Marker

Der implementierte Marker-Provider nutzt die Informationen aus QR-Codes für die Standortbestimmung. Bei QR-Codes handelt es sich um zwei-dimensionale Codes, die von der International Organization for Standardization [52] standardisiert sind. In diesen Codes können Informationen wie Texte, URL oder andere Daten gespeichert werden. Sie können bis zu 7089 Dezimalziffern oder 4296 alphanumerische Zeichen beinhalten.

In der implementierten Infrastruktur beinhalten die QR-Codes den Breitengrad, Längengrad und die Höhe der Position, an der sie sich befinden.

Zum Scannen des QR-Codes wird eine externe Android-Anwendung verwendet. In dieser Implementierung wurde mit der Anwendung BarcodeScanner von Zebra Crossing [53] gearbeitet. Die Kommunikation mit dieser fremden Anwendung erfolgt durch implizite Intents. Dafür wird ein Intent mit folgendem URI an das System gesendet: com.google.zxing.client.android.SCAN. Die Antwort des BarcodeScanners enthält eine Zeichenkette mit dem Inhalt des gescannten QR-Codes. Falls die Anwendung nicht installiert ist, wird der Benutzer automatisch zum Android-Market weitergeleitet. Hier kann die Anwendung dann kostenlos heruntergeladen werden. Aus der empfangenen Zeichenkette werden dann die Standortinformationen extrahiert. Hierfür wird die Zeichenkette aufgespalten und der Breiten- und Längengrad entnommen. Die Zeichenkette kann als Höhenangabe sowohl eine Größe in Metern als auch eine Stockwerkangabe enthalten. Falls es sich um eine Stockwerkangabe handelt, wird diese mit der Stockwerkhöhe multipliziert, um eine Höhenangabe in Metern zu erhalten. Die Genauigkeit der gescannten Position wird immer auf 0,3 Meter festgelegt. Diese Distanz entspricht ungefähr dem Abstand, der bei einem Scanvorgang für die in der Implementierung verwendeten Markergröße eingehalten wird. Diese Festlegung der Genauigkeit setzt voraus, dass die im Marker angegebenen Koordinaten korrekt sind. Anschließend wird die ermittelte Position durch den Marker-Provider im System verteilt.

Das Scannen des QR-Codes wird explizit vom Benutzer ausgelöst. Er ruft die Benutzeroberfläche des Marker-Providers auf, indem er diesen im Provider-Tab anklickt. Die Benutzeroberfläche wird links in Abbildung 4.3 dargestellt. Sie enthält einen Button zum Scannen eines QR-Codes. Daraufhin wird die BarcodeScanner-Anwendung gestartet, welche rechts in Abbildung 4.3 zu erkennen ist. Nach dem Scannen wird wieder der vorige Bildschirm angezeigt, indem zusätzlich die Daten der gescannten Position eingeblendet werden.

Abbildung 4.3: Marker-Provider und BarcodeScanner

Externe Anwendungen

Der Location-Provider für externe Anwendungen dient wie bereits erwähnt dazu, fremde Anwendungen in den Haupt-Location-Provider einzubinden. Realisiert wurde diese Anbindung durch einen Bound-Service. Der Bound-Service stellt der fremden Anwendung ein Interface mit Funktionen der Lokalisierungsanwendung zur Verfügung. Die Interface-Datei wird im Anhang A aufgezeigt.

Es stehen der fremden Anwendung folgende vier Funktionen zur Verfügung:

- `stopLocalizationService`
- `checkServiceRunning`
- `checkServiceConfigured`
- `sendLocationUpdate` mit den Parametern: Latitude, Longitude, Altitude und Accuracy

Läuft der Lokalisierungsdienst noch nicht, wenn eine fremde Anwendung sich mit der Lokalisierungsanwendung verbindet, kann der Dienst gestartet werden. Damit fremde Anwendungen den Lokalisierungsdienst starten und stoppen können, muss diese Eigenschaft in den Anwendungseinstellungen aktiviert sein. Wenn der Lokalisierungsdienst von einer fremden Anwendung gestartet wurde, kann er auch wieder von dieser beendet werden, durch Aufruf der `stopLocalizationService`-Funktion. Wurde der Dienst hingegen in der Lokalisierungsanwendung selbst gestartet, kann er nicht durch eine fremde Anwendung beendet werden.

Die Funktionen `checkServiceRunning` und `checkServiceConfigured` dienen der Statusüberprüfung des Lokalisierungsdienstes. Die erste Funktion gibt an, ob der Service gerade läuft und genutzt werden kann. Die zweite Funktion dient der Abfrage, ob der Lokalisierungsdienst bereits konfiguriert wurde. Diese Konfiguration muss bei der ersten Ausführung der Lokalisierungsanwendung getätigt werden. Der Lokalisierungsdienst kann nicht gestartet werden, bevor die Lokalisierungsanwendung nicht konfiguriert wurde.

Läuft der Lokalisierungsdienst, kann die `sendLocationUpdate`-Funktion genutzt werden, um die Positionsangaben zu übertragen. Dabei müssen der Breitengrad (Latitude), der Längengrad (Longitude), die Höhe (Altitude) und die Genauigkeit (Accuracy) als Parameter übergeben werden.

Dieser Location-Provider benötigt keine Benutzerinteraktion. Der Benutzer hat jedoch die Möglichkeit, die verbundenen Anwendungen anzuzeigen, indem er den externen Location-Provider im Provider-Tab auswählt.

4.2.3 Hinzufügen von Location-Providern

Um einen neuen Location-Provider hinzuzufügen, müssen keine Änderungen am Kern der Lokalisierungsanwendung vorgenommen werden. Es muss nur eine neue Provider-Instanz im Provider-Manager der Anwendung erstellt werden. Folgende Parameter müssen für einen neuen Provider angegeben werden:

- **Provider-Name:** Name, unter welchem der Location-Provider im System verteilt wird.
- **Anzeigename:** Name des Providers, wie er bei Auflistungen angezeigt wird.
- **Beschreibung:** Kurze Beschreibung des Providers.
- **Logo:** Logo des Providers, welches zum Beispiel bei einer Auflistung angezeigt werden kann.
- **Einstellungsdatei:** Dateiname der XML-Datei, in welcher sich die vom Benutzer editierbaren Einstellungen des Providers befinden. Diese Datei wird dann automatisch in das Einstellungsmenü der Anwendung eingefügt.
- **Referenz:** Gibt an, ob der Provider als Referenz für Koppelnavigations-Provider verwendet werden kann.
- **Dienst:** Gibt an, ob es sich bei dem Provider um einen Dienst handelt. Ist dies der Fall, wird der Provider automatisch mit dem Lokalisierungsdienst gestartet.
- **Dienst-Klasse:** Handelt es sich bei dem Provider um einen Dienst, enthält der Parameter den Namen der Dienst-Klasse.
- **Anzeige-Klasse:** Besitzt der Provider eine Benutzerschnittstelle, enthält der Parameter den Namen der Activity-Klasse.

- **Erstausführungs-Klasse:** Muss der Provider vor der ersten Benutzung konfiguriert werden, kann hier der Name einer Klasse angegeben werden, mit der diese erste Konfiguration durchgeführt werden kann. Bei der ersten Ausführung der Lokalisierungsanwendung werden die Provider, die einen Wert für diesen Parameter besitzen, aufgelistet. Durch Anklicken eines Providers wird dessen Erstausführungs-Klasse als Activity aufgerufen.

Nach der Eingabe dieser Parameter wird der neue Location-Provider automatisch vom Lokalisierungsdienst als Location-Provider im System registriert.

4.3 AR-Browser-Anwendung

Dieser Abschnitt beschreibt den implementierten AR-Browser. Zuerst wird der Aufbau sowie die Darstellung der virtuellen Elemente erläutert. Anschließend wird die Benutzeroberfläche der Navigation und POI-Interaktion vorgestellt. Nachfolgend wird die Anbindung des AR-Browsers an die Lokalisierungsanwendung erläutert. Zuletzt werden der POI- und der Navigationsmodus im Detail beschrieben.
Folgende Berechtigungen sind für den AR-Browser erforderlich:

- `CAMERA`: Diese Berechtigung wird benötigt, um auf die Kamera des Smartphones zuzugreifen.
- `ACCESS FINE LOCATION`: Diese erlaubt den Zugriff auf den GPS-Location-Provider, falls der Haupt-Location-Provider der Lokalisierungsanwendung nicht zur Verfügung steht.
- `ACCESS NETWORK STATE`: Diese Berechtigung ermöglicht es zu überprüfen, ob das Smartphone über eine Internetverbindung verfügt.
- `INTERNET`: Die Verbindung zum Internet wird zum Beispiel für den Bezug der AR-Inhalte notwendig.

Nach dem Start der Anwendung wird sofort geprüft, ob eine für den Bezug der AR-Inhalte erforderliche Internetverbindung besteht. Falls keine Verbindung besteht, muss der Benutzer die Anwendung verlassen.
Der AR-Browser steht auf Deutsch sowie auf Englisch zur Verfügung. Die Sprache wird automatisch anhand der Betriebssystem-Sprache bestimmt.

4.3.1 Virtuelle Elemente

Die virtuellen Elemente umfassen alle in der AR-Ansicht angezeigten Elemente, also sowohl POI als auch Navigationsangaben. In diesem Abschnitt werden der Aufbau der virtuellen Elemente und ihre Darstellung in der AR-Ansicht beschrieben. Dabei werden die Parameter der virtuellen Elemente,

welche für eine korrekte Anzeige auf dem Smartphone-Bildschirm benötigt werden, beschrieben. Anschließend werden die Positionierung und Darstellung auf dem Smartphone-Bildschirm im Detail erläutert. Die virtuellen Elemente sollen an ihren zugehörigen Koordinaten angezeigt werden. Hierfür müssen der Standort sowie die Ausrichtung des Smartphones bekannt sein. Die Ausrichtung des Smartphones kann in Roll-Nick-Gier-Winkeln ausgedrückt werden. Für die Positionierung der virtuellen Inhalte werden der Roll-Winkel und der Gier-Winkel (Kompassrichtung) verwendet. Der Roll-Winkel dient der vertikalen Positionierung und der Gier-Winkel dient der horizontalen Positionierung. Die gemessenen Ausrichtungswerte werden auf die gleiche Weise wie beim Pedometer gefiltert. Bei der Filterung werden die Messwerte voriger Messungen mit einbezogen und der Mittelwert gebildet. Falls Standortinformationen zur Verfügung stehen, wird die Kompassrichtung mit Hilfe dieser Standortinformationen vom magnetischen Norden zum geographischen Norden umgerechnet. Die Messung der Kompassrichtung wird ausgehend vom Hochformat des Smartphones durchgeführt. Der AR-Browser wird jedoch im Querformat betrieben, also muss die Kompassrichtung immer umgerechnet werden. Zusätzlich zur Ausrichtung des Smartphones muss auch der Standort bekannt sein. Sobald dieser dem AR-Browser zur Verfügung steht, können die virtuellen Elemente angezeigt werden. Jedes virtuelle Element enthält folgende Parameter:

- **Id:** Enthält eine eindeutige Identifikationsnummer.
- **Name:** Enthält den Namen des virtuellen Elements.
- **Koordinaten:** Enthält die Koordinaten des virtuellen Elements: Breitengrad, Längengrad, Höhe.

Zusätzlich werden folgende Parameter für die Positionierung der virtuellen Elemente benötigt:

- **Distanz:** Beschreibt die Distanz der aktuellen AR-Browser-Position zu den Koordinaten des virtuellen Elements.
- **Kompasspeilung (Bearing):** Die Kompassrichtung in der sich die Koordinaten des virtuellen Elements bezogen auf die AR-Browser-Position befinden.
- **Sichtbarkeit:** Beschreibt, ob das virtuelle Element im sichtbaren Radius liegt und angezeigt wird. Dieser Radius kann vom Benutzer festgelegt werden. Er beschreibt die maximale Distanz zum Browser, innerhalb welcher virtuelle Elemente angezeigt werden.

Diese Parameter sind von dem Standort und der Ausrichtung des Smartphones abhängig. Das heißt, bei jeder Standortaktualisierung sowie neuer Ausrichtung müssen diese Parameter aktualisiert werden. Die Distanz und die Kompasspeilung werden mit Hilfe von Android-Funktionen berechnet. Die Sichtbarkeit wird anhand der errechneten Distanz sowie einer vom Benutzer festgelegten Maximaldistanz bestimmt. Ist die Distanz kleiner als die Maximaldistanz, wird das Element sichtbar.

Um die virtuellen Elemente im Browser anzeigen und mit ihnen interagieren zu können, besitzt jedes Element zusätzlich folgende Parameter:

- **Horizontale Bildschirmkoordinate:** Enthält die horizontale Bildschirmkoordinate in Pixeln, an der sich das virtuelle Element auf dem Smartphone-Bildschirm befindet.
- **Vertikale Bildschirmkoordinate:** Enthält die vertikale Bildschirmkoordinate in Pixeln, an der sich das virtuelle Element auf dem Smartphone-Bildschirm befindet.
- **Region:** Beschreibt die Fläche, in der das virtuelle Element gezeichnet wird. Diese dient der Interaktion mit dem virtuellen Element. Bei jeder Berührung des Bildschirms in der AR-Ansicht wird getestet, ob sich die Berührung in der Region befindet. Ist dies der Fall, kann eine Funktion des virtuellen Elements ausgeführt werden.
- **Ausgewählt:** Dieser Parameter gibt an, ob das virtuelle Element gerade ausgewählt ist. Dies kann zum Beispiel auftreten, nachdem die Region des virtuellen Elements berührt wurde.

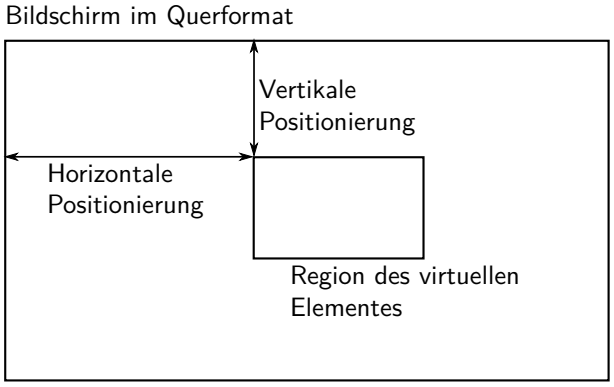

Abbildung 4.4: Positionierung des virtuellen Elements

Diese Parameter werden jedes Mal aktualisiert, wenn die AR-Ansicht neu gezeichnet wird. Es ist wichtig, in welcher Reihenfolge die virtuellen Elemente gezeichnet werden. Begonnen wird mit dem am weitesten entfernten Element. So bleiben bei Überschneidungen der virtuellen Elemente die vorderen Elemente sichtbar.

Bei jedem neuen Zeichnen eines virtuellen Elements wird eine neue Bildschirmposition berechnet. Die Bestimmung der Bildschirmposition besteht aus einer horizontalen und einer vertikalen Positionierung. Abbildung 4.4 zeigt die Positionierungen sowie die Region des virtuellen Elements auf dem Smartphone-Bildschirm.

Horizontale Positionierung Es wird zuerst eine mögliche neue horizontale Position berechnet. Diese wird anhand der Kompasspeilung des virtuellen Elements, der Bildschirmbreite in Pixel und der Kompassrichtung des Smartphones bestimmt. Die Berechnung der horizontalen Position wird

anhand des horizontalen Sichtfeldes, in dem die virtuellen Elemente abgebildet werden, und der Bildschirmbreite durchgeführt. Dieses virtuelle horizontale Sichtfeld kann vom Benutzer verändert werden. Der Winkel des virtuellen Sichtfeldes kann zwischen 30 und 90 Grad eingestellt werden. Bei einem Sichtfeld von beispielsweise 30 Grad werden nur virtuelle Inhalte angezeigt, die in einem Bereich zwischen -15 und +15 Grad in Bezug auf die Smartphone-Kompassrichtung liegen. Das virtuelle Sichtfeld ist in Abbildung 4.5 zu sehen. Es ist ersichtlich, dass durch eine Vergrößerung des Winkels mehr virtuelle Elemente sichtbar werden. Die Veränderung des Sichtfeldes bietet dem Benutzer also eine Zoom-Funktion. Das hat zum Beispiel den Vorteil, dass der Benutzer mit Hilfe der Zoom-Funktion weiter wegrücken kann, wenn er sich zu nahe an den virtuellen Objekten befindet. So erhält er einen besseren Überblick der umliegenden POI. Das Sichtfeld der virtuellen Inhalte ist in dem Fall nicht gleich dem Sichtfeld der Kamera.

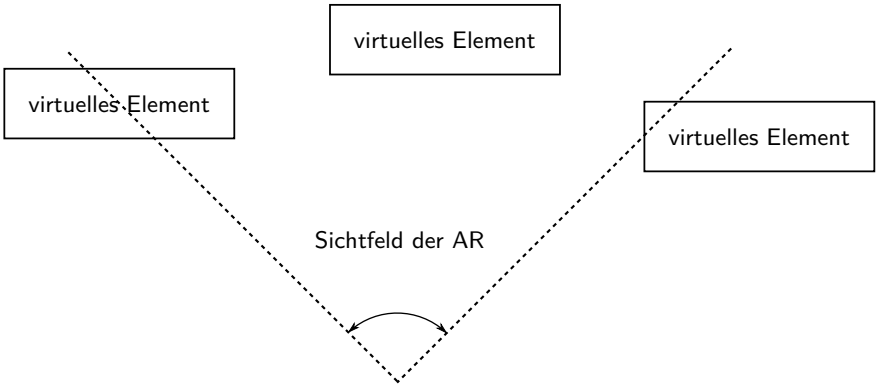

Abbildung 4.5: Virtuelles Sichtfeld

Für die Berechnung wird zuerst die Pixelanzahl pro Grad bestimmt. Hierfür wird die Pixelanzahl der Bildschirmbreite durch den Wert des eingestellten virtuellen Sichtfeldes geteilt. Anschließend wird die Winkeldifferenz zwischen der Kompassrichtung des Smartphones und der Kompasspeilung des virtuellen Elements errechnet. Die horizontale Bildschirmkoordinate wird durch die Multiplikation der Winkeldifferenz mit der Pixelanzahl pro Grad und die anschließende Verschiebung in den Mittelpunkt bestimmt:

$$Winkeldifferenz * PixelanzahlProGrad + Bildschirmbreite/2 \tag{4.1}$$

Die errechnete Zahl wird abschließend auf einen ganzen Pixelwert gerundet. Durch die Verschiebung in den Mittelpunkt wird ein virtuelles Element, das genau vor dem Benutzer liegt, in der Mitte des Smartphone-Bildschirms angezeigt.

Um die Ansicht zu stabilisieren und ein Flackern der virtuellen Elemente zu vermeiden, wird die Position eines Elements nicht bei jedem neuen Zeichnen der Ansicht geändert. Die berechnete neue Position wird zuerst mit der im virtuellen Element gespeicherten horizontalen Koordinate

verglichen. Die Position wird nur aktualisiert, wenn die Distanz zwischen der alten und der neuen Position größer als ein definierter Schwellwert ist. In der Implementierung wurde der Schwellwert für die horizontale Verschiebung auf 200 Pixel festgelegt. Enthält das virtuelle Element noch keine horizontale Position, wird standardmäßig die berechnete angenommen. Durch diesen Vergleich können nur größere Bewegungen die Positionen der virtuellen Elemente verändern und die AR-Ansicht wird so stabilisiert.

Vertikale Positionierung Die Höhe der virtuellen Inhalte, also ihre vertikale Positionierung, kann in der Anwendung ein- oder ausgeschaltet werden. Ist die vertikale Positionierung ausgeschaltet, werden die virtuellen Inhalte immer zentral angezeigt. Ist sie hingegen eingeschaltet, werden die virtuellen Elemente bezogen auf die Differenz zwischen ihrer Höhe und der Höhe des Benutzers im Bildschirm positioniert. So ist zum Beispiel leicht erkennbar, ob man sich auf dem gleichen oder auf einem anderen Stockwerk wie ein virtuelles Element befindet. Das Ausschalten der vertikalen Positionierung kann in verschiedenen Szenarien hilfreich sein. Ist man beispielsweise in Bewegung, fällt die Interaktion oder Navigation leichter, wenn die virtuellen Objekte immer zentral angezeigt werden. Diese zentrale Anzeige der virtuellen Elemente ermöglicht es, das Smartphone in einem beliebigen Roll-Winkel zu bedienen. So muss der Benutzer das Smartphone nicht in einer unhandlichen Position halten, um einem virtuellen Objekt zu folgen.

Bei aktiver vertikaler Positionierung wird ähnlich wie bei der horizontalen Positionierung zuerst eine neue Position berechnet. Dafür werden die Bildschirmhöhe, der Roll-Winkel des Smartphones, die Höhe und die Distanz des virtuellen Elements sowie die Höhe des Benutzers benötigt. Die Berechnung der vertikalen Position wird anhand des vertikalen Sichtfeldes und der Bildschirmhöhe in Pixel durchgeführt. Das vertikale Sichtfeld ist im Gegensatz zu dem horizontalen Sichtfeld auf 30 Grad festgelegt. Bei der vertikalen Positionierung kann der Benutzer zwischen zwei verschiedenen Ansichten wählen. Die erste Ansicht bestimmt die Bildschirmposition bezogen auf die Differenz zwischen der Höhe des virtuellen Elements und der Höhe des Benutzers. Die virtuellen Elemente gleicher Höhe werden immer an der gleichen vertikalen Bildschirmposition angezeigt. Die zweite Ansicht bestimmt die Position nicht nur bezogen auf die Höhendifferenz, sondern auch bezogen auf die Distanz zum virtuellen Element. Diese Ansicht bietet den Vorteil, dass mehrere hintereinanderliegende Objekte gleicher Höhe versetzt angezeigt werden. Die hinteren Objekte werden somit weniger verdeckt und für den Benutzer sichtbar. Der Winkel des virtuellen Elements wird durch folgende Formel bestimmt:

$$\arctan\left(\left(H\ddot{o}he_{[\text{VirtuellesElement}]} - H\ddot{o}he_{[\text{Benutzer}]}\right)/Distanz\right) \tag{4.2}$$

Bei der ersten Ansicht wird der Wert der Distanz in der Formel 4.2 auf einen konstanten Wert festgelegt. Ist bei der zweiten Ansicht die Differenz zwischen der Höhe des virtuellen Elements und der Höhe des Benutzers gleich Null, wird diese auf Eins gesetzt, um eine Verschiebung zu

erhalten.

Der nächste Schritt der vertikalen Positionierung besteht in der Bestimmung der Winkeldifferenz zwischen dem vorher bestimmten Winkel des virtuellen Elements und dem Roll-Winkel des Smartphones. Mit Hilfe dieser Winkeldifferenz, der Pixelanzahl pro Grad, und der Bildschirmhöhe wird dann ähnlich wie bei der horizontalen Positionierung die vertikale Bildschirmkoordinate berechnet. Hierfür kommt die Formel 4.1 zum Einsatz, wobei die Bildschirmbreite durch die Bildschirmhöhe ersetzt wird.

Die vertikale Positionierung wird auf die gleiche Weise wie bei der horizontalen Positionierung stabilisiert. Der Schwellwert wurde bei der vertikalen Stabilisierung auf 50 Pixel festgelegt, da die Bildschirmhöhe kleiner als die Bildschirmbreite ist. Sind die Bildschirmkoordinaten des virtuellen Elements bestimmt, kann das Element in der AR-Ansicht angezeigt werden.

4.3.2 Benutzeroberfläche

Dieser Abschnitt beschreibt die verschiedenen Benutzeransichten, die sowohl für die Navigation als auch für die POI-Interaktion von Bedeutung sind.

AR-Ansicht

Die Abbildung 4.6 zeigt die AR-Ansicht sowie die verfügbaren Bedienelemente. Diese enthält das aktuelle Kamerabild sowie diverse virtuelle Objekte. Bei den virtuellen Objekten kann es sich sowohl um POI als auch um Navigationsangaben handeln. Diese werden in den entsprechenden Kapiteln näher erläutert. Zusätzlich zu den POI und Navigationsangaben enthält die Oberfläche Bedienelemente. Diese ermöglichen dem Benutzer den Zugriff auf die Funktionen des AR-Browsers. Befindet sich die Anwendung in der AR-Ansicht, verhindert diese, dass der Bildschirm sich verdunkelt und in den Standby-Modus wechselt. Nachfolgend werden die in der Abbildung 4.6 aufgelisteten Bedienelemente kurz erläutert.

1. **POI-Modus:** Dieser Button dient dem Ein- und Ausschalten des POI-Modus. Dabei ändert sich die Deckkraft des Buttons je nach Status des POI-Modus. Bei aktiviertem Modus wird das Symbol in voller Deckkraft angezeigt. Zusätzlich werden im aktiven Modus links und rechts des POI-Symbols Zahlen angezeigt. Links des Symbols steht die Anzahl der heruntergeladenen POI und rechts die Anzahl der POI, welche in dem sichtbaren Bereich liegen. Die Distanz des sichtbaren Bereichs kann vom Benutzer in den Bildschirmeinstellungen festgelegt werden. Der POI-Modus wird in Abschnitt 4.3.4 detailliert beschrieben.
2. **Navigationsmodus:** Dieser Button aktiviert und deaktiviert den Navigationsmodus. Die Deckkraft des Buttons verändert sich wie bei dem POI-Modus, je nachdem, ob der Modus

Abbildung 4.6: AR-Ansicht des Augmented-Reality-Browsers mit Bedienelementen

aktiv ist oder nicht. Bei aktivem Modus wird das Symbol in voller Deckkraft angezeigt. Diese Funktion wird in Abschnitt 4.3.5 detailliert beschrieben.

3. **Lokalisierungsanwendung oder BarcodeScanner:** An dieser Stelle können zwei verschiedene Buttons angezeigt werden. Der angezeigte Button hängt von dem Status der Lokalisierungsanwendung ab. Ist diese nicht installiert oder konfiguriert, wird das Logo der Lokalisierungsanwendung (blaue Spirale) angezeigt. Bei fehlender Installation wird der Benutzer nach Betätigung des Buttons darauf hingewiesen, dass die Anwendung installiert werden muss. Ist die Anwendung bereits installiert, jedoch nicht konfiguriert, wird die Lokalisierungsanwendung gestartet. Bei erfolgter Installation und Konfiguration wird an dieser Stelle ein QR-Code-Symbol angezeigt. Durch Betätigung des QR-Code-Buttons wird die BarcodeScanner-Anwendung gestartet. Die Funktionsweise der Standortbestimmung mit Hilfe des AR-Browsers wird in Abschnitt 4.3.3 genauer beschrieben.

4. **Positionsinformationen:** Das Symbol dieses Buttons variiert, je nach Genauigkeit der aktuellen Position. Die Genauigkeit ist in vier Stufen aufgeteilt: eine Position mit einer Genauigkeit von unter 5 Metern (grünes Symbol), eine Position mit einer Genauigkeit zwischen 5 und 20 Metern (orangenfarbenes Symbol), eine Position mit einer Genauigkeit von über 20 Metern (rotes Symbol) oder keine Position (rotes Symbol mit Kreuz). Durch Betätigung des Buttons wird ein Dialog-Fenster gestartet, welches die Koordinaten sowie die präzise Genauigkeit der aktuellen Position anzeigt.

5. **Bildschirmeinstellungen:** Dieser Button startet ein Dialog-Fenster, in dem folgende vier Einstellungen für die Darstellung der virtuellen Elemente vorgenommen werden können: „Maximale Distanz“, „Aktiviere Höhe“, „Aktiviere dynamische Höhe“, „Zoom“. Die maximale Distanz und der Zoom werden über einen Schieberegler eingestellt. Die Grenzwerte des

Schiebereglers für die maximale Distanz werden dynamisch angepasst. Der Standardwert der oberen Grenze beträgt 100 Meter. Sind die heruntergeladen Elemente jedoch weiter entfernt, wird diese Grenze entsprechend angepasst. Mit Hilfe des Zoomreglers kann das virtuelle Sichtfeld zwischen 30 Grad (nah) und 90 Grad (weit) eingestellt werden. Die beiden anderen Einstellungen können über einen Schalter aktiviert oder deaktiviert werden. Der Schalter der dynamischen Höhe ist nur aktiv, wenn die Höheneinstellung aktiviert ist. Die Bedeutungen dieser Einstellungen wurden in Abschnitt 4.3.1 erläutert.

Kartenansicht

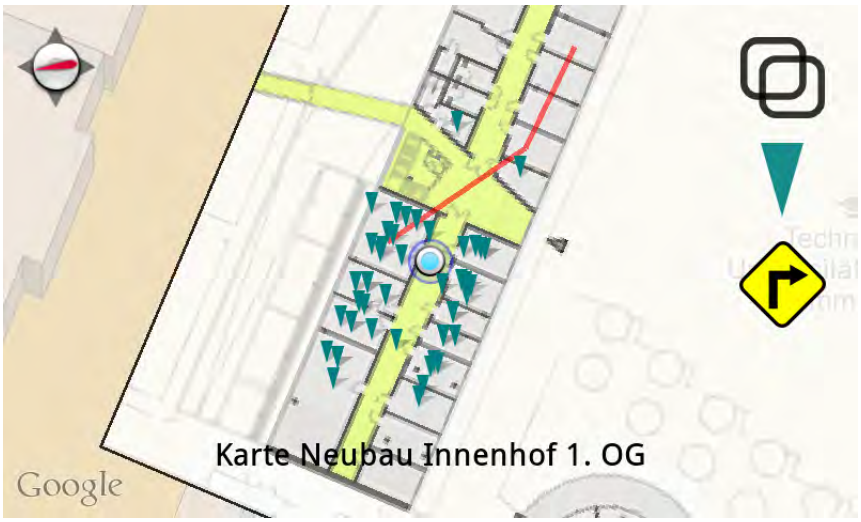

Abbildung 4.7: Kartenansicht des Augmented-Reality-Browsers

Zusätzlich zur AR-Ansicht wurde eine Kartenansicht implementiert, welche in Abbildung 4.7 zu sehen ist. Die Kartenansicht ist über den Menübutton des Android-Smartphones erreichbar. Die Anzeige der Indoor-Karten, der POI und der Route kann mit Hilfe der in der Abbildung zu sehenden Buttons ein- oder ausgeschaltet werden. Diese drei einblendbaren Anzeigen sind als Overlays für Google-Maps implementiert. Die Indoor-Karte entspricht einem Bild, welches über die Google-Karte gelegt wird. Sie wird wie in Abschnitt 3.4.4 beschrieben von einem Karten-Server bezogen. Für eine korrekte Darstellung der Indoor-Karte wird sowohl ihre Position als auch ihre Rotation berücksichtigt. Zudem wird der Name der Indoor-Karte in der Kartenansicht angezeigt. Die umliegenden POI werden anhand der POI-Symbole auf der Karte dargestellt. Die angezeigte Route besteht aus einer roten Linie, welche die einzelnen Navigationspunkte miteinander verbindet.[1]

[1] Die in der Abbildung 4.7 dargestellten Navigationspunkte befinden sich jeweils in der Mitte des Raums, weil die Koordinaten der Flure zum Zeitpunkt der Implementierung nicht zur Verfügung standen. Die dargestellte Route beschreibt also den Weg von Raum zu Raum, bis das Ziel erreicht ist.

Auch die aktuelle Position des Benutzers wird dargestellt. Die Karte wird automatisch auf die Benutzerposition zentriert, sie folgt also den Bewegungen des Benutzers.

4.3.3 Anbindung an die Lokalisierungsanwendung

Der AR-Browser bezieht die Standortinformationen vom Haupt-Location-Provider der implementierten Lokalisierungsanwendung. Falls dieser nicht zur Verfügung steht, kann der AR-Browser den GPS-Provider verwenden. Zudem ist der AR-Browser als externe Anwendung mit der Lokalisierungsanwendung verbunden. Der Grund und die Funktionsweise dieser Anbindung werden nachfolgend erläutert. Im Abschnitt 4.2.2 wurde die implementierte Indoor-Lokalisierung vorgestellt. Dieses implementierte hybride System besteht aus Marker und Pedometer. Es wurde gezeigt, dass das Scannen von Markern eine Benutzerinteraktion in der Lokalisierungsanwendung erfordert. Beabsichtigt der Benutzer einen Marker zu scannen und nutzt gerade den AR-Browser, so muss er den Browser verlassen und in die Lokalisierungsanwendung wechseln. Erst jetzt kann er den Marker scannen und wieder in den AR-Browser zurückkehren. Ein derartiges Scannen von Markern ist für den Benutzer jedoch umständlich. Um die Benutzerfreundlichkeit des Systems zu erhöhen, wurde deshalb die Möglichkeit implementiert, Marker direkt aus dem AR-Browser heraus zu scannen. Hierfür nutzt der Browser das im Anhang A beschriebene Interface, welches das Senden von Positionsangaben an die Lokalisierungsanwendung erlaubt. Damit der AR-Browser mit der Lokalisierungsanwendung verbunden werden kann, muss diese installiert und konfiguriert sein. Ist dies nicht der Fall, kann auch kein Marker aus dem AR-Browser heraus gescannt werden. Wurde der Lokalisierungsdienst in der Lokalisierungsanwendung nicht vorher gestartet, wird dies bei der Verbindung des AR-Browsers mit der Lokalisierungsanwendung durchgeführt. Beim Verlassen des AR-Browsers wird in diesem Fall der Lokalisierungsdienst auch wieder gestoppt. Dieses automatische Starten und Stoppen aus einer externen Anwendung kann nur dann erfolgen, wenn es nicht in der Lokalisierungsanwendung deaktiviert ist. Das Scannen eines Markers funktioniert im AR-Browser wie bei der Lokalisierungsanwendung über ein implizites Intent. Durch Betätigung des QR-Code-Symbols im AR-Browser wird das Intent im System verteilt und die BarcodeScanner-Anwendung gestartet. Ist diese nicht installiert, wird der Benutzer wie bei der Lokalisierungsanwendung an den Android-Market weitergeleitet. Nachdem der QR-Code mit Hilfe der BarcodeScanner-Anwendung gescannt wurde, können aus dem eingelesenen Wert die Standortinformationen ausgelesen werden. Diese werden dann über die im Interface beschriebenen Funktionen an die Lokalisierungsanwendung weitergeleitet. Die Positionsangaben aus dem gescannten Marker fließen so in den Haupt-Location-Provider ein, ohne in die Lokalisierungsanwendung wechseln zu müssen.

4.3.4 POI-Modus

Der POI-Modus ermöglicht die Interaktion des Benutzers mit den POI. Der POI-Modus muss aktiviert sein, damit eine Interaktion möglich ist. Ist der Modus hingegen deaktiviert, werden keine POI in der AR-Ansicht angezeigt. Der Modus kann mit Hilfe des POI-Buttons ein- oder ausgeschaltet werden.

Eigenschaften

Die für den POI-Modus notwendigen AR-Daten werden vom AR-Server bezogen. Diese Daten umfassen die POI und die Provider, welche beide im ARML-Format vom Server heruntergeladen werden. Der zu verwendende AR-Server kann in den Einstellungen der Anwendung geändert werden. Das Herunterladen und das Auswerten der Daten wird im Hintergrund durchgeführt. Die POI-Daten werden regelmäßig neu beim Server angefragt. Eine solche erneute Anfrage wird ausgelöst, sobald der Benutzer einen gewissen Mindestabstand zum letzten Standort, an dem die POI-Daten heruntergeladen wurden, zurückgelegt hat. Durch diesen Mindestabstand lassen sich überflüssige Übertragungen vermeiden. Befindet sich der nächstgelegene POI zum Beispiel in einer Distanz von 500 Metern zum Benutzer, so ist es nicht notwendig, alle 5 Meter eine neue POI-Anfrage an den Server zu senden. Aus diesem Grund wird ein Mindestabstand bestimmt, welcher auf ein Zehntel der Distanz zum nächstgelegenen POI festgelegt wird. Ist die Distanz zu dem nächstgelegenen POI jedoch kleiner als 100 Meter, wird der Mindestabstand trotzdem auf 10 Meter gesetzt, um zu häufige POI-Anfragen zu vermeiden. Ein POI-Element erweitert ein virtuelles Element um folgende Parameter:

- **Provider:** Enthält den Provider, dem der POI angehört.
- **Beschreibung:** Enthält eine kurze Beschreibung des POI.

Der AR-Browser kann die POI-Elemente zusätzlich zur AR-Ansicht auch in der Kartenansicht oder in der Listenansicht darstellen. Die Kartenansicht wurde im Abschnitt 4.3.2 vorgestellt. Die Listenansicht bietet die Möglichkeit die POI in einer Liste anzuzeigen. Sie ist über den Menü-Button des Smartphones zu erreichen. In der Liste werden die Namen und eine kurze Beschreibung des POI angezeigt. Zusätzlich wird die Distanz zu dem aktuellen Standort des Benutzers angezeigt. Die Liste wird nach diesen Distanzen sortiert. Die Ansicht verfügt zudem über einen Aktualisierungs-Button, der ein manuelles Herunterladen der POI-Daten ermöglicht. Durch Anklicken der POI-Elemente können die zugehörigen standortbezogenen Dienste gestartet werden.

Provider

Wie bereits erwähnt sind die POI in Provider aufgeteilt. Ein Provider beinhaltet folgende Parameter:

- **Id:** Zeichenkette für die Identifikation des Providers.
- **Name:** Name des Providers.
- **Beschreibung:** Kurze Beschreibung des Providers.
- **LBS-URI:** URI, an der sich die standortbezogene Dienste befinden.

Die Provider-Daten werden beim Start der Anwendung automatisch beim Server angefragt und heruntergeladen. Durch die Auswahl eines Providers in der Provider-Ansicht werden die zugehörigen POI heruntergeladen. Die Provider-Ansicht kann wie die Kartenansicht über den Menü-Button des Smartphones erreicht werden. Ist der POI-Modus aktiv und es wurde noch kein Provider ausgewählt, wird in der AR-Ansicht die Nachricht eingeblendet, dass ein Provider ausgewählt werden muss. Die Provider-Ansicht kann auch durch Anklicken dieser Meldung gestartet werden. Die Ansicht enthält eine Auflistung der verfügbaren Provider. Für jeden Provider wird der Name und eine kurze Beschreibung in der Liste angezeigt. Der Benutzer hat zudem die Möglichkeit die Provider-Liste manuell zu aktualisieren, indem er den dafür vorgesehenen Button betätigt. Daraufhin werden die Provider-Daten erneut vom Server heruntergeladen.

Funktionsweise und Ablauf

Der in Abschnitt 3.4.2 konzipierte Ablauf der POI-Interaktion wird in diesem Abschnitt anhand des entwickelten AR-Browsers veranschaulicht. Nach dem Start der Anwendung ist der POI-Modus automatisch aktiv. Somit besteht der erste Schritt in der Auswahl eines Providers. Dafür muss in die Provider-Ansicht gewechselt werden, welche bereits im vorherigen Abschnitt erläutert wurde. Ist mindestens ein Provider ausgewählt, können die zugehörigen POI-Daten heruntergeladen werden. Der Benutzer bleibt hiervon ungestört, da das Herunterladen und Auswerten der Daten im Hintergrund abläuft. Die POI-Daten werden nur heruntergeladen, wenn die Anwendung über einen Standort verfügt. Die aktuellen Positionsangaben werden in der AR-Ansicht angezeigt. Aus dieser Information kann der Benutzer auslesen, ob er über eine Position verfügt und ob diese ausreichend genau für den aktuellen Betrieb ist.

Beim Server wird nur eine beschränkte Anzahl von POI gleichzeitig angefragt. Diese Beschränkung dient zum einen dazu das Datenvolumen bei der Serverkommunikation zu reduzieren und zum anderen die Ressourcen des Smartphones zu schonen. Die Anzahl der gleichzeitig angefragten Einträge beträgt standardmäßig 50 POI. Dieser Wert kann in den Einstellungen der Anwendung

geändert werden. Je mehr POI in der Ansicht dargestellt werden, desto höher wird der Rechenaufwand für das Smartphone.

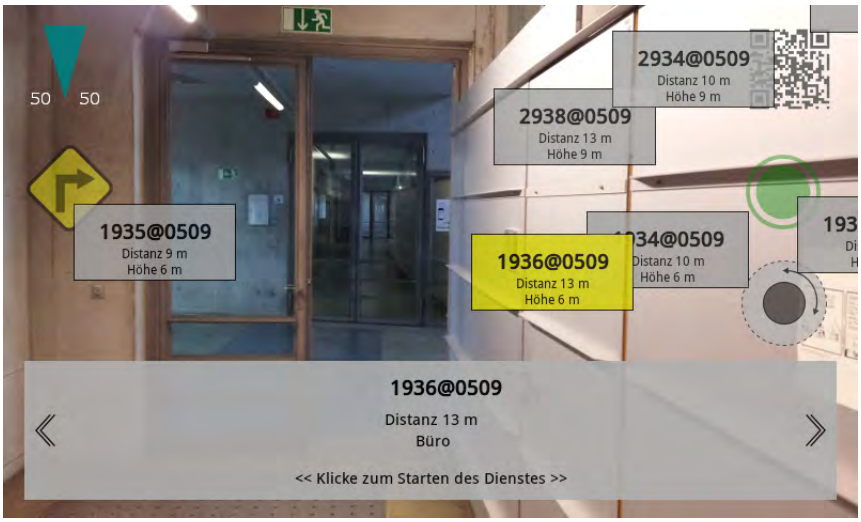

Abbildung 4.8: POI-Modus des Augmented-Reality-Browsers

Sobald die POI heruntergeladen und ausgewertet sind, werden sie wie in Abbildung 4.8 in der AR-Ansicht dargestellt. Bei dieser Darstellung ist die Einstellung der dynamischen Höhe aktiviert. Das heißt, Elemente gleicher Höhe werden vertikal versetzt angezeigt, um Verdeckungen zu reduzieren. Ist die vom Benutzer eingestellte oder die standardmäßig gesetzte Maximaldistanz kleiner als die Distanz zu dem nächsten POI, wird diese angepasst. Die neue Maximaldistanz besteht aus der Distanz zu dem nächsten POI zuzüglich eines definierten Offsets. Damit wird sichergestellt, dass dem Benutzer wenigstens ein POI angezeigt wird. Die Maximaldistanz kann in den Bildschirmeinstellungen anschließend vom Benutzer angepasst werden. Ein POI wird als graues Rechteck in der Ansicht dargestellt. Sofort sichtbar ist der Name des POI. Zusätzlich wird die Distanz vom aktuellen Standort zu dem POI und die Höhe des POI angezeigt. Durch Anklicken eines POI wird dieser ausgewählt. Das Rechteck des ausgewählten POI wird daraufhin gelb. Zudem erscheint im unteren Bereich des Bildschirms ein größeres Rechteck, in dem weitere Informationen zum POI enthalten sind. In dieser Fläche wird nun auch die Beschreibung des POI angezeigt. An dem linken und rechten Rand dieser Fläche befinden sich Pfeile, mittels derer zu dem nächsten oder vorigen POI gewechselt werden kann. Die POI sind nach ihrer Distanz zur aktuellen Position sortiert, so dass der vorige POI näher an der aktuellen Position liegt und der nächste weiter von der aktuellen Position entfernt ist. Erreicht man das letzte Element der Liste, springt man mit dem nächsten POI zurück an den Anfang. Befindet man sich beim ersten Element, springt man mit dem vorigen POI an das Ende der Liste.

Wird abgesehen von den Pfeilen ein Teil dieser Fläche berührt, werden die standortbezogenen Dienste gestartet. Dafür wird die vom Provider zur Verfügung gestellte URI aufgerufen. Die

Anwendung prüft, ob die angegebene URI die Zeichenkette „http“ enthält. Ist dies der Fall, wird die Browser-Activity in der AR-Browser-Anwendung gestartet und die entsprechende Seite aufgerufen. Die Id des ausgewählten POI wird der Adresse als Parameter angehängt. Andernfalls wird die URI als implizites Intent im System verteilt. Die Id des ausgewählten POI wird mit dem Intent übergeben. Existiert eine Anwendung, die auf den Intent reagieren kann, wird diese gestartet. Existiert hingegen keine solche Anwendung, erscheint eine Fehlermeldung.

4.3.5 Navigationsmodus

Der Navigationsmodus ermöglicht es zwischen verschiedenen POI und Adressen zu navigieren. Durch Betätigung des Navigationssymbols wird ein Dialog für die Navigationsangaben gestartet. Der Navigationsmodus wird erst aktiv, wenn man nach Herunterladen der Route die Navigation in dem entsprechenden Dialog startet. Durch erneute Betätigung des Navigationssymbols wird der Navigationsmodus verlassen und die aktuelle Route verworfen.

Eigenschaften

Die Navigationspunkte werden im KML-Format vom Server bezogen. Die in der Server-Datenbank vorhandenen Navigationspunkte besitzen wie jedes virtuelle Element einen Namen. Dieser wird in der KML-Datei mitgesendet. Die von Google-Maps erhaltenen Navigationspunkte besitzen hingegen keine Namen und werden deshalb vom Server durchnummeriert. Ein Name eines Google-Navigationspunktes wäre zum Beispiel Placemark 2.

Um Leistungsfähigkeit und Übersichtlichkeit der Anwendung zu gewährleisten, werden bei langen Routen nicht alle Navigationspunkte angezeigt. Die Anzahl der angezeigten Navigationspunkte ist in der Implementierung auf 20 Elemente festgelegt. Die anderen Elemente werden weder angezeigt, noch aktualisiert. Das heißt, ihre Distanz und Richtung werden nicht bei jeder Bewegung oder Standortänderung, wie es in Abschnitt 4.3.1 beschrieben wurde, angepasst.

Ein Navigationselement erweitert ein virtuelles Element um folgenden Parameter:

- **Distanz zum nächsten Navigationspunkt:** Dieser Parameter enthält die Distanz zu dem nächsten anzusteuernden Navigationspunkt.

Die Route kann alternativ wie bereits erwähnt in der Kartenansicht betrachtet werden.

Funktionsweise und Ablauf

Bevor mit der Navigation begonnen werden kann, muss eine Route beim Server angefragt werden. Die Abbildung 4.9 zeigt das Dialog-Fenster, in dem die Navigationseingaben getätigt wer-

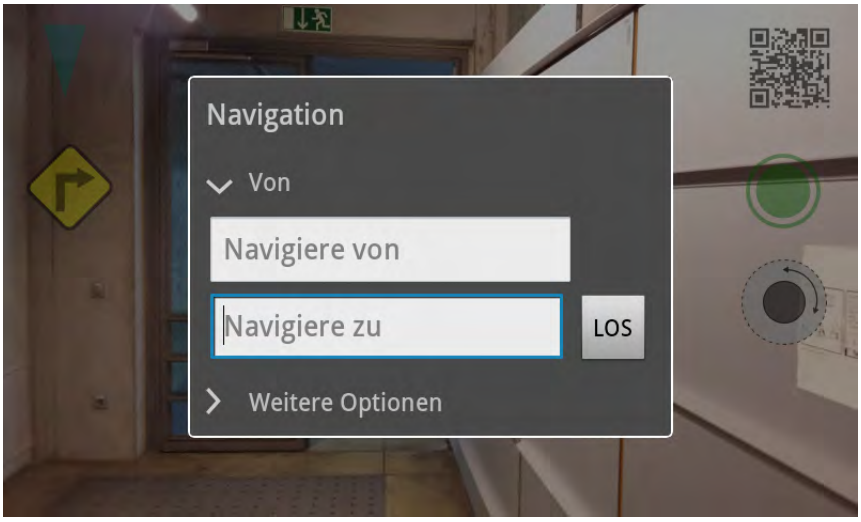

Abbildung 4.9: Dialog-Fenster der Navigationseingaben

den. Dieses wird durch Betätigung des Navigationssymbols aufgerufen. Nach dem Aufrufen des Dialog-Fensters wird das Feld für die Zieleingabe angezeigt. Die Eingabe des Zielpunktes ist unerlässlich für die Anfrage der Route. Die weiteren über ausklappbare Menüs erreichbaren Felder sind optional. Mit Hilfe dieser optionalen Felder kann ein manueller Startpunkt sowie zusätzliche Routenoptionen angegeben werden. In der Abbildung 4.9 wurde zum Beispiel die manuelle Starteingabe ausgeklappt.

Wurden alle Informationen eingegeben, kann durch Betätigung des LOS-Buttons eine Route angefragt werden. Wie bereits erwähnt werden daraufhin die Eingaben vom Server geprüft. Während dieser Prüfung erscheint auf dem Bildschirm ein Dialog-Fenster, das den Benutzer über diese Überprüfung informiert. Wurde ein POI-Name eingegeben, der vom Server nicht eindeutig zugewiesen werden konnte, wird eine Liste mit möglichen POI an den AR-Browser übertragen. Diese POI werden, wie auch in dem POI-Modus, im ARML-Format übertragen. Die empfangenen POI werden in Form einer Liste in einem Dialog-Fenster angezeigt. Der Benutzer wählt dann den richtigen POI aus dieser Liste aus und bestätigt die Eingabe. Wurde ein Startwert eingegeben, wird dieser auf die gleiche Weise geprüft. Der Dialog für die POI-Auswahl kann also insgesamt zweimal erscheinen, jeweils nach Überprüfung von Start- und Zielwert. Die beiden Fälle können durch den Titel des Dialog-Fensters voneinander unterschieden werden. Wurden alle Mehrdeutigkeiten beseitigt, kann die Route angefragt werden. Während die Route heruntergeladen wird, erscheint ein Dialog-Fenster, das den Benutzer über diese Aktion informiert.

Nach vollständigem Herunterladen der Route, werden die Routeninformationen in einem Dialog-Fenster angezeigt. Abbildung 4.10 zeigt ein Beispiel eines solchen Dialog-Fensters. Wurde eine Route gefunden, werden im Dialog-Fenster nochmals die wichtigsten Eigenschaften der Route

Abbildung 4.10: Dialog-Fenster der Route

aufgelistet. Die gesamte Distanz der Strecke sowie Start und Ziel der Route werden angezeigt. Zusätzlich können mit Hilfe des ausklappbaren Menüs Informationen über die Navigationspunkte eingeblendet werden. Die Namen und die Distanzen zwischen den einzelnen Navigationspunkten werden angezeigt. Die Distanzen zwischen den Navigationspunkten sind bereits im voraus berechnet worden und in dem entsprechenden Parameter des Navigationspunktes gespeichert. Handelt es sich um die gewünschte Route, kann der Navigationsmodus und damit die Navigation gestartet werden. Wurde eine falsche Route gefunden, kann die Navigation abgebrochen werden. Konnte hingegen keine Route gefunden werden, hat man die Wahl zwischen erneuter Eingabe der Start- und Zielangaben oder Abbruch der Navigation.

Startet man die Navigation, werden die Navigationspunkte in der AR-Ansicht sichtbar. Zuvor wird jedoch geprüft, ob der erste Navigationspunkt im sichtbaren Radius liegt. Ist dies nicht der Fall, wird die Maximaldistanz des sichtbaren Bereichs entsprechend angepasst. Die Maximaldistanz wird auf einen Wert bestehend aus der Distanz des zu folgenden Navigationspunktes und einem definierten Offset festgelegt. Somit ist sichergestellt, dass der Navigationspunkt, dem der Benutzer folgen muss, im sichtbaren Radius liegt.

Der aktuelle Navigationspunkt, dem der Benutzer folgen muss, ist wie in Abbildung 4.11 zu sehen grün gekennzeichnet. Liegen die nachfolgenden Navigationspunkte im sichtbaren Bereich, werden diese gelb und mit geringerer Deckkraft in der AR-Ansicht angezeigt. Maximal werden nur zwanzig dieser Punkte angezeigt, da sonst die Ansicht zu unübersichtlich wird. Die AR-Navigationspunkte sind im Bildschirm als Kreise dargestellt. Im Inneren des Kreises wird die Distanz in Metern zu den jeweiligen Navigationspunkten angezeigt. Der Benutzer wird von Navigationspunkt zu Navigationspunkt geleitet, bis er sein Ziel erreicht.

Abbildung 4.11: Navigation in der AR-Ansicht

Der aktuelle Navigationspunkt kann sich auch außerhalb vom Blickfeld des Benutzers befinden. Der Benutzer kann beispielsweise gerade in die entgegengesetzte Richtung gedreht sein. In diesem Fall wird ein Pfeil am Bildschirmrand angezeigt, der die Richtung angibt, in die sich der Benutzer drehen muss, um den Navigationspunkt sehen zu können. Muss er sich nach links wenden, zeigt ein Pfeil am linken Bildschirmrand nach links. Muss der Benutzer sich hingegen nach rechts drehen, zeigt ein Pfeil am rechten Rand nach rechts. Diese Orientierungshinweise bleiben solange sichtbar, bis der aktuelle Navigationspunkt wieder im Bildschirm angezeigt wird, er sich also wieder im virtuellen Sichtfeld befindet.

Befindet sich der Benutzer näher als zwei Meter am aktuellen Navigationspunkt, gilt dieser als erreicht und verschwindet. Dieser Navigationspunkt wird dann aus der Route gelöscht und der nächste Punkt wird aktiv. Anschließend wird erneut geprüft, ob der neue aktuelle Navigationspunkt im sichtbaren Bereich des Benutzers liegt. Ist dies nicht der Fall, wird die Maximaldistanz wie beim ersten Navigationspunkt entsprechend angepasst. Der neue Navigationspunkt wird nun als grüner Kreis angezeigt, dem der Benutzer folgen kann. Ist der Benutzer am letzten Navigationspunkt angekommen, informiert ihn eine Nachricht im Bildschirm über das Erreichen des Ziels.

4.4 Server-Komponenten

In diesem Kapitel werden die einzelnen Server-Komponenten beschrieben. Dabei werden die verschiedenen Anfragen an den Server sowie deren nachfolgende Bearbeitung im Detail erläutert. Zuerst wird die POI-Komponente beschrieben, welche zum Bezug der POI- und Provider-Daten genutzt wird. Anschließend wird die Navigationskomponente im Detail erklärt. Diese ist für die Routenbestimmung sowie die Überprüfung der POI-Namen zuständig. Dann wird der Karten-

Server näher beschrieben. Der Karten-Server ermöglicht den Bezug von Indoor-Karten. Dieser kann auch von anderen Anwendungen als dem AR-Browser, wie zum Beispiel der Lokalisierungsanwendung, genutzt werden. Abschließend wird die verwendete Datenbank sowie die Verwaltungswebseite vorgestellt.

4.4.1 POI-Komponente

Die Aufgaben der POI-Komponente unterteilen sich in Anfragen für POI und Provider.
Die Anfrage für Provider sieht beispielsweise folgendermaßen aus:

```
AR-Server-URL?action=getProviders
```

Diese URL enthält nur den `getProviders`-Befehl. Bei der Provider-Anfrage werden keine zusätzlichen Parameter benötigt. Erhält der Server eine solche Anfrage, werden alle Provider aus der MySQL-Datenbank geladen. Die geladenen Provider werden in Java-Objekte geschrieben. Die Übertragung der Provider-Daten wird jedoch im ARML-Format durchgeführt. Im nächsten Schritt werden deshalb die Provider-Elemente in das ARML-Format umgewandelt. Abschließend wird der resultierende XML-Datenstrom zurück an den AR-Browser gesendet.

Die POI-Anfrage an den AR-Server hat folgende Struktur:

```
AR-Server-URL?action=getPois&latitude=48.12312&longitude
   =11.12332&altitude=6&maxNumberOfPois=50&providers=
   provider1_provider2
```

Die Anfrage enthält den Befehl `getPois`, welcher die gewünschte Aktion angibt. Zudem enthält sie die im AR-Browser ausgewählten Provider. Die einzelnen Provider sind durch einen Unterstrich getrennt angegeben. Im obigen Beispiel wurden zwei Provider übertragen: Provider1 und Provider2. Außerdem wird die aktuelle Position des AR-Browsers und die maximale Anzahl der zurückzusendenden POI übertragen. Anhand dieser Daten kann der Server die um den AR-Browser liegenden POI ermitteln. Zuerst werden die POI-Elemente der übermittelten Provider aus der MySQL-Datenbank geladen und als Java-Objekte gespeichert. Anschließend wird die Distanz zwischen dem übermittelten Längen- und Breitengrad und jedem gefundenen POI berechnet. Im Anschluss wird die POI-Liste bezüglich der berechneten Distanz sortiert. Die am nächsten an der übertragenen Position liegenden POI werden aus der Liste herausgefiltert. Die Anzahl der zu entnehmenden POI wird im `maxNumberOfPois`-Feld angegeben. Anschließend werden die entnommenen POI, für die Übertragung, in das ARML-Format umgewandelt. Der resultierende XML-Datenstrom wird abschließend an den AR-Browser übermittelt.

4.4.2 Navigationskomponente

Die Navigationskomponente umfasst die Routenanfragen sowie die Anfragen zur Überprüfung der POI-Namen. Bei der Überprüfung der POI-Namen testet der Server diese auf Eindeutigkeit. Ein vom Benutzer eingegebener POI-Name ist eindeutig, wenn dieser einem einzigen Wert in der Datenbank zugeordnet werden kann. Diese Überprüfung wird durch folgende URL aufgerufen:

```
AR-Server-URL?action=checkPlacemark&placemark=Beispiel-POI-
    Name
```

Die Anfrage enthält neben dem Befehl auch den zu testenden POI-Namen. Der Server sucht in der Datenbank nach Einträgen, die den übermittelten POI-Namen enthalten. Bei der Suche wird die Groß- und Kleinschreibung vernachlässigt. Die gefundenen Einträge werden dann in das ARML-Format umgewandelt und zurück an den AR-Browser übermittelt. Werden keine Einträge in der Datenbank gefunden, wird ein XML-Datenstrom ohne POI-Einträge an den AR-Browser übermittelt. Der Server trifft keine Entscheidung anhand der gefundenen Daten. Allein der AR-Browser reagiert unterschiedlich, wenn er einen XML-Strom ohne Inhalte, mit einem Eintrag oder mit mehreren Einträgen erhält.

Sind alle Benutzereingaben geprüft, wird eine Route angefordert. Die Anfrage für eine Route unterscheidet sich, je nachdem, ob ein manueller Startwert eingegeben wurde oder nicht. Wird kein Startwert eingegeben, werden die aktuellen Koordinaten des AR-Browsers an den Server übertragen. Eine solche Anfrage sieht beispielsweise folgendermaßen aus:

```
AR-Server-URL?action=getRoute&to=Zielwert&latitude=48.1123&
    longitude=11.123&altitude=3&options=KeineTreppen
```

Wird hingegen ein manueller Startwert verwendet, wird dieser anstelle der Koordinaten übertragen und die Anfrage hat folgende Struktur:

```
AR-Server-URL?action=getRoute&to=Zielwert&from=Startwert&
    options=KeineTreppen
```

Erhält der Server eine Anfrage mit Koordinaten, sucht er zuerst nach einem Eintrag in der Datenbank, der diesen Koordinaten zugewiesen werden kann. Dafür werden die Einträge nach ihrer Distanz zu den übertragenen Koordinaten sortiert. Liegt der nächstgelegene Eintrag näher als 20 Meter an den übertragenen Koordinaten, wird dieser als Startwert verwendet. Wird kein Eintrag gefunden, werden die Koordinaten weiterhin verwendet. Eine solche Routenanfrage mit Koordinaten wird als Outdoor-Navigationspunkt angesehen und enthält automatisch einen Outdoor-Anteil, welcher bei Google-Maps angefragt werden muss. Erhält der Server eine Anfrage mit einem manuellen Startwert, wird ebenso geprüft, ob der eingegebene Wert einem Eintrag in der Datenbank zugeordnet werden kann. Ist dies der Fall, wird der gefundene Eintrag verwendet. Andernfalls

wird die Benutzereingabe als Outdoor-Navigationspunkt betrachtet. Bei den nachfolgenden Aktionen wird nicht mehr unterschieden, ob die Anfrage Koordinaten oder einen manuellen Startwert enthielt.

Im nächsten Schritt wird geprüft, ob für den eingegebenen Zielwert ein Eintrag in der Datenbank gefunden werden kann. Ist das der Fall, wird der gefundene Eintrag verwendet. Andernfalls wird die Benutzereingabe als Outdoor-Navigationspunkt betrachtet. Anschließend wird, wie in Abschnitt 3.4.3 beschrieben, die Routenbestimmung für den zutreffenden Fall ausgeführt.

Die interne Routenbestimmung wird mittels des auf dem Server implementiertem Dijkstra-Algorithmus durchgeführt. Für diese Berechnung werden die Verbindungen aller Navigationspunkte aus der Datenbank geladen. Anhand dieser Verbindungen kann der Dijkstra-Algorithmus den kürzesten Weg finden. Dabei können diese Verbindungen anhand übergebener Optionen gefiltert werden. Wird zum Beispiel die `KeineTreppen`-Option übertragen, werden alle Verbindungen, die Treppen enthalten, aus der Liste der zu verwendenden Verbindungen entfernt. Werden mehrere Optionen angegeben, werden diese in der URL durch einen Unterstrich voneinander getrennt.

Die externe Routenbestimmung erfolgt durch Google-Maps. Die Anfrage einer Route hat bei Google-Maps folgende Struktur:

```
maps.google.de/?saddr=Startwert&daddr=Zielwert&dirflg=w&
    output=kml
```

Start- und Zielwert werden an Google übermittelt. Zusätzlich wird in der URL angegeben, dass die zurückzugebenden Daten im KML-Format sein sollen. Der `dirflg`-Parameter gibt an, welcher Modus für die Route gewählt werden soll. Drei verschiedene Modi können unterschieden werden: zu Fuß, mit dem Auto oder mit dem Fahrrad. Da die Indoor-Navigation allgemein zu Fuß stattfindet, werden auch Fußwege für die Outdoor-Navigation angefragt. Die von Google-Maps erhaltenen Daten werden in Java-Objekte umgewandelt, um sie besser mit den intern ermittelten verknüpfen zu können.

Sind alle Navigationspunkte der Route ermittelt, werden sie für die Übertragung in das KML-Format umgewandelt. Abschließend wird der resultierende XML-Datenstrom an den AR-Browser gesendet.

4.4.3 Karten-Server

Der Karten-Server liefert Karteninformationen für Indoor-Karten, welche auf der Google-Karte eingeblendet werden können. Eine Anfrage an den Server hat folgende Struktur:

```
Karten-Server-URL?action=getMap&latitude=48.12321&longitude
    =11.3421&altitude=3
```

Der Server benötigt die aktuelle Position des AR-Browsers, um die Karte, welche die Benutzerposition beinhaltet, zu ermitteln. Nach Erhalt der Anfrage lädt der Server alle Karten aus der Datenbank. Dann wird getestet, ob die angegebene Position in einer der Karten liegt.

Jeder Karteneintrag enthält folgende Parameter:

- **Name:** Name der Karte
- **Norden:** Nördlich begrenzende Koordinate der Karte
- **Osten:** Östlich begrenzende Koordinate der Karte
- **Süden:** Südlich begrenzende Koordinate der Karte
- **Westen:** Westlich begrenzende Koordinate der Karte
- **Altitude:** Höhe der Karte
- **Rotation:** Winkel, um den die Karte gedreht werden muss
- **URL:** URL, an der sich die Bilddatei der Karte befindet

Die Parameter Norden, Osten, Süden und Westen beschreiben nicht die endgültige Position der Indoor-Karte, sondern ihre Begrenzungen parallel zu den Längen- und Breitengraden. Erst die anschließende Rotation erlaubt die richtige Positionierung der Indoor-Karte auf der Google-Karte. Um zu ermitteln, ob sich die Benutzerposition innerhalb der für Norden, Osten, Süden und Westen angegebenen Koordinaten befindet, muss die Benutzerposition um den entgegengesetzten Rotationswinkel gedreht werden. Die Benutzer-Koordinaten können nach dieser Rotation mit den Koordinaten-Parametern der Indoor-Karte verglichen werden. Liegt die Benutzerposition innerhalb mehrerer Karten, wird die Karte mit der geringsten Höhendifferenz zur Höhe des AR-Browsers ausgewählt. Somit wird sichergestellt, dass die Karte für das entsprechende Stockwerk ausgewählt wird. Der gefundene Karteneintrag wird zur Übertragung in das KML-Format umgewandelt und anschließend an den AR-Browser übermittelt.

4.4.4 Datenbank und Administration

Dieser Abschnitt behandelt die Datenbankstruktur sowie die Administration der vorgestellten Server-Komponenten. Abbildung 4.12 zeigt die vorhandenen Tabellen mit ihren Verbindungen. Die Placemark-Tabelle bildet die Basis für jedes virtuelle Element. Sowohl die Navigationspunkte als auch die POI sind in dieser Tabelle gespeichert. Die Placemark-Tabelle beinhaltet den Namen sowie die Koordinaten des virtuellen Elements. Die POI-Elemente besitzen zusätzlich zu den in der Placemark-Tabelle vorhandenen Parametern eine Beschreibung und einen Provider. Die POI-Tabelle ergänzt die Placemark-Tabelle um die fehlenden Parameter der POI. Durch die Id

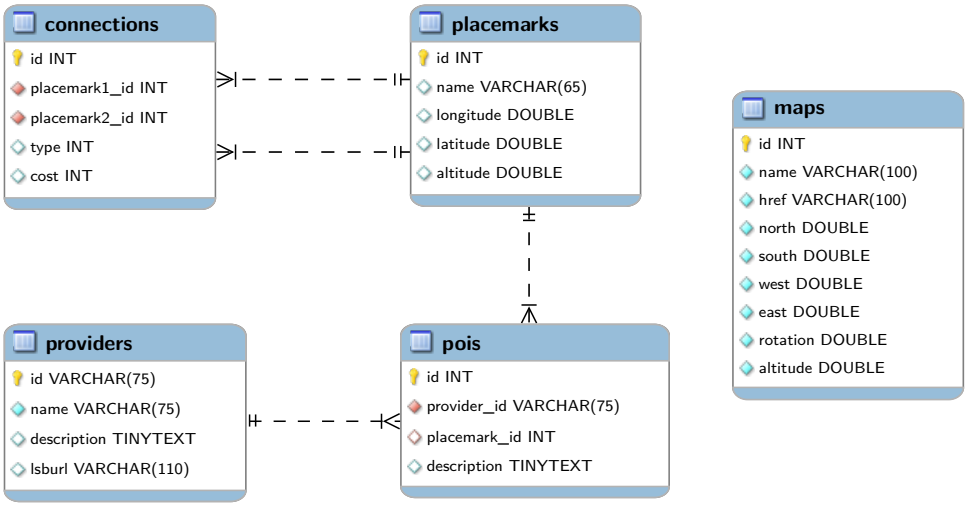

Abbildung 4.12: AR-Datenbank (modifizierte Grafik aus MySQL Workbench)

des zugehörigen Placemark-Eintrags sind beide Tabellen miteinander verbunden. Die POI-Tabelle enthält also nur die POI-spezifischen Erweiterungen. Jeder POI ist einem Provider zugewiesen. Die Provider sind in der Provider-Tabelle gespeichert. Ein Provider-Eintrag enthält den Namen, eine Beschreibung und die URL der standortbezogenen Dienste des Providers. Die POI-Tabelle ist auch über einen Fremdschlüssel mit der Provider-Tabelle verbunden. Ein POI-Eintrag beinhaltet also die Id des zugehörigen Provider-Eintrags.

Die Connection-Tabelle dient der Navigation und beinhaltet die Verbindungen der Placemarks untereinander. Ein Eintrag dieser Tabelle beschreibt die Verbindung zwischen zwei Placemarks. Er beinhaltet die Id der beiden verbundenen Placemark-Einträge. Zusätzlich enthält ein Eintrag die Kosten der Verbindung, zum Beispiel die Distanz zwischen den Placemarks. Außerdem besitzt jede Verbindung einen Typ, welcher als Zahlenwert in der Datenbank gespeichert wird. Der Typ kann zum Beispiel angeben, ob die Verbindung eine Treppe beinhaltet.

Es bestehen keine Verbindungen zwischen den Tabellen der AR-Komponenten und der Tabelle des Karten-Servers. Die Elemente eines Karten-Eintrags wurden in Abschnitt 4.4.3 bereits erläutert.

Für die Administration der vorgestellten Datenbanken wurde eine Webseite erstellt. Diese bietet die Möglichkeit, Einträge in der Datenbank anzulegen, zu ändern oder zu entfernen. Die Webseite enthält die Kategorien: Navigation und Placemarks. Die Placemarks-Kategorie ermöglicht die Verwaltung von Providern, POI und Placemarks. In der Navigations-Kategorie können die Verbindungen der Placemarks und die Einträge für den Karten-Server editiert werden.

Zusätzlich zu der Verwaltungswebseite besteht für die POI-Komponente eine alternative Lösung, um die Datenbank mit Inhalten zu versorgen. Diese alternative Lösung bietet die Möglichkeit, POI- und Provider-Daten im ARML-Format an den Server zu übertragen. Dies hat gegenüber

der Webseite den Vorteil, dass in einem Zug größere Datenmengen in die Datenbank geschrieben werden können. So kann zum Beispiel eine XML-Datei mit Provider oder POI komplett in die Server-Datenbank geladen werden. Dieser Upload kann durch eine der folgenden URL initiiert werden:

```
AR-Server-URL?action=addPois
AR-Server-URL?action=addProvider
```

In der Anfrage wird der XML-Datenstrom mit den Inhalten an den Server übermittelt. Dieser wertet die empfangenen Daten aus und speichert sie in Java-Objekte. Diese werden dann in einem weiteren Schritt in die MySQL-Datenbank geschrieben.

4.5 Standortbezogene Dienste

Als Erweiterung für den AR-Browser wurden verschiedene standortbezogene Dienste implementiert, welche typische Arbeitsschritte in Bürogebäuden erleichtern. Sie wurden in Form einer Webseite realisiert. Diese Umsetzung wurde gewählt, weil die Dienste keinen Zugriff auf die Hardware des Smartphones benötigen. Zudem bietet die Realisierung als Webseite wie in Abschnitt 3.5 beschrieben den Vorteil, dass die Dienste leicht erweiterbar und flexibel sind.

4.5.1 Implementierte Dienste

Die implementierten Dienste können in die beiden Kategorien personen- und raumbezogene Dienste unterteilt werden. Ein personenbezogener Dienst wäre zum Beispiel das Anrufen einer bestimmten Person. Ein Beispiel für einen raumbezogenen Dienst wäre eine Reservierung eines bestimmten Raums.

Personenbezogene Dienste

Personenbezogene Dienste enthalten die Dienste, die für Personen durchgeführt werden können. Nachfolgend werden die implementierten personenbezogenen Dienste im Detail erläutert.

Kontakt Der Kontakt-Dienst bietet die Möglichkeit eine Person direkt zu kontaktieren. Die Kontaktaufnahme kann über Telefon oder E-Mail erfolgen, unter Voraussetzung, dass die Kontaktdaten der Person, also Telefonnummer und E-Mail-Adresse, in der Datenbank hinterlegt sind.

Meeting Der Meeting-Dienst ermöglicht eine Terminvereinbarung mit einer bestimmten Person. Folgende Parameter können für einen Termin angegeben werden: Betreff, Name und E-Mail-Adresse des Benutzers sowie gewünschtes Datum und Uhrzeit. Die E-Mail-Adresse des Benutzers muss angegeben werden, um eine Antwort der kontaktierten Person zu ermöglichen, für den Fall, dass sie zum Beispiel zur gewünschten Uhrzeit den Termin nicht wahrnehmen kann. Der Betreff wird vom Benutzer angegeben, um die kontaktierte Person auf den Grund des Termins hinzuweisen.

Raumbezogene Dienste

Raumbezogene Dienste umfassen die Dienste, die für Räume durchgeführt werden können. Nachfolgend werden die implementierten raumbezogenen Dienste im Detail beschrieben.

Reservierung Dieser Dienst ermöglicht es eine Reservierung für einen Raum vorzunehmen sowie bereits getätigte Reservierungen anzusehen. Für eine Reservierung können folgende Parameter angegeben werden: Name und E-Mail-Adresse des Benutzers, Startdatum und Startzeit sowie Enddatum und Endzeit der Reservierung. Dieser Dienst ermöglicht beispielsweise die Buchung eines Seminarraums für einen bestimmten Zeitraum.

Notizen Der Notizen-Dienst ermöglicht es, Notizen an Räumen zu hinterlassen. So kann ein Benutzer zum einen die bereits angebrachten Notizen eines Raums lesen und zum anderen eine neue Notiz an diesem Raum hinterlassen. Eine solche Notiz wäre beispielsweise: „Büro für 10 Minuten nicht besetzt“. Folgende Parameter können für eine Notiz angegeben werden: Name und E-Mail-Adresse des Benutzers sowie Titel und Nachricht der Notiz.

4.5.2 LBS-Webseite

In diesem Abschnitt wird die realisierte Webseite im Detail beschrieben. Zuerst werden dabei verschiedene Eigenschaften der Webseite aufgelistet, anschließend wird näher auf den Aufbau und die Funktionsweise der Webseite eingegangen. Die entwickelte Webseite wurde wie bereits erwähnt auf jQuery-Mobile basierend realisiert.

Eigenschaften

Die Webseite der standortbezogenen Dienste wird jeweils für einen spezifischen POI aufgerufen. Dabei wird die Id des POI vom AR-Browser an die URL der standortbezogenen Dienste angehängt. Die URL hat somit beispielsweise folgenden Aufbau:

```
LBS-URL?id=123
```

Die implementierten Dienste sind für Räume ausgelegt. Aus diesem Grund wird die POI-Id in die entsprechende Raum-Id umgewandelt. Der Zusammenhang zwischen Räumen und POI wird in Abschnitt 4.5.3 verdeutlicht. Die Startseite liest die POI-Id ein und speichert die zugehörige Raum-Id in der aktuellen Sitzung (Session) des Benutzers. Zudem wird der Name des ausgewählten Raums in der Sitzung gespeichert. Das Speichern der Raumdaten in der Sitzung hat den Vorteil, dass diese so auf der gesamten Webseite zur Verfügung stehen.
Die entwickelte Webseite wurde zur Erhöhung der Benutzerfreundlichkeit in zwei Sprachen, Deutsch und Englisch, realisiert. Die Sprachauswahl erfolgt automatisch über die Spracheinstellungen des Browsers. In der im AR-Browser vorhandenen Browser-Activity wurden die Standard-Buttons des Web-Browsers deaktiviert. Das Layout der LBS-Webseiten wird damit dem Layout der nativen Android-Anwendungen angenähert.

Aufbau und Funktionsweise

Die Abbildung 4.13 zeigt die Startseite der standortbezogenen Dienste. Die Navigationsleiste, wel-

Abbildung 4.13: Startseite der standortbezogenen Dienste

che sich oben im Bildschirm befindet, enthält den Namen des ausgewählten Raums. Kann zu der übertragenen POI-Id kein Raum gefunden werden, wird dies in der Navigationsleiste angezeigt. Die Navigationsleiste enthält auf allen folgenden Seiten einen Zurück-Button, um auf die vorige Seite zurückzukehren. Dies ist erforderlich, da die Standard-Buttons des Browsers dem Benutzer nicht zur Verfügung stehen. Auch wird in der Navigationsleiste gegebenenfalls ein Hinzufügen-Button eingeblendet. Dieser dient dann zum Beispiel dem Hinzufügen einer Notiz oder Reservierung. Auf der Startseite kann der Benutzer zwischen raumbezogenen oder personenbezogenen Diensten wählen.

Die Struktur der Webseite für die raumbezogenen Dienste ist in der Abbildung 4.14 dargestellt. Dieses Zustandsdiagramm umfasst die Navigationsmöglichkeiten auf der Webseite für diese Diens-

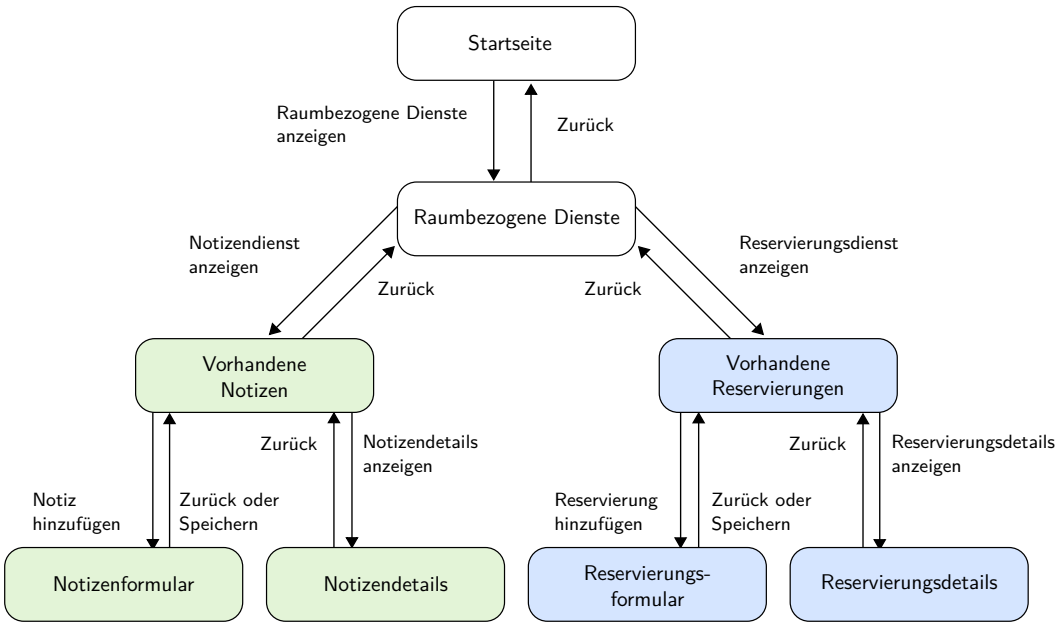

Abbildung 4.14: Zustandsdiagramm der raumbezogenen Dienste

te. Nach Auswahl der raumbezogenen Dienste erreicht der Benutzer eine Auflistung der verfügbaren Dienste. Die Abbildung zeigt den Fall, in dem beide implementierten Dienste für den Raum zur Verfügung stehen. Ein Raum kann auch beispielsweise nur den Notizen-Dienst unterstützen. Dies wäre zum Beispiel der Fall, wenn es sich bei dem ausgewählten Raum um ein nicht reservierbares Büro handelt. Im Abschnitt 4.5.3 wird näher auf die Zuweisung der Dienste eingegangen. Der nächste Schritt besteht in der Auswahl des gewünschten Dienstes. Der Benutzer hat auch die Möglichkeit zurück zur Startseite zu wechseln. Die Startseiten beider verfügbarer Dienste enthalten eine Übersicht der bereits vorhandenen Einträge. Der Benutzer kann nach Auswahl eines Dienstes entweder Details zu den angezeigten Einträgen ansehen oder einen neuen Eintrag hinzufügen. Um einen neuen Eintrag hinzuzufügen, also zum Beispiel eine neue Notiz zu hinterlassen, muss er ein Formular ausfüllen, welches die für diesen Eintrag notwendigen Felder enthält. Die für den jeweiligen Eintrag erforderlichen Felder wurden bereits in Abschnitt 4.5.1 erläutert. Der Benutzer hat zudem immer die Möglichkeit, wie auch in der Abbildung gezeigt, auf die vorige Seite zurückzugehen.

Die Struktur der Webseite für die personenbezogenen Dienste wird in Abbildung 4.15 gezeigt. Das Zustandsdiagramm zeigt die zugehörigen Navigationsmöglichkeiten. Nachdem der Benutzer die personenbezogenen Dienste ausgewählt hat, erhält er eine Übersicht über die diesem Raum zugeordneten Personen. Als nächstes kann der Benutzer eine gewünschte Person auswählen. Daraufhin werden die verfügbaren Dienste für diese Person angezeigt. In dem Zustandsdiagramm sind beide implementierte Dienste aufgezeigt, wobei einer bestimmten Person jedoch nicht unbedingt

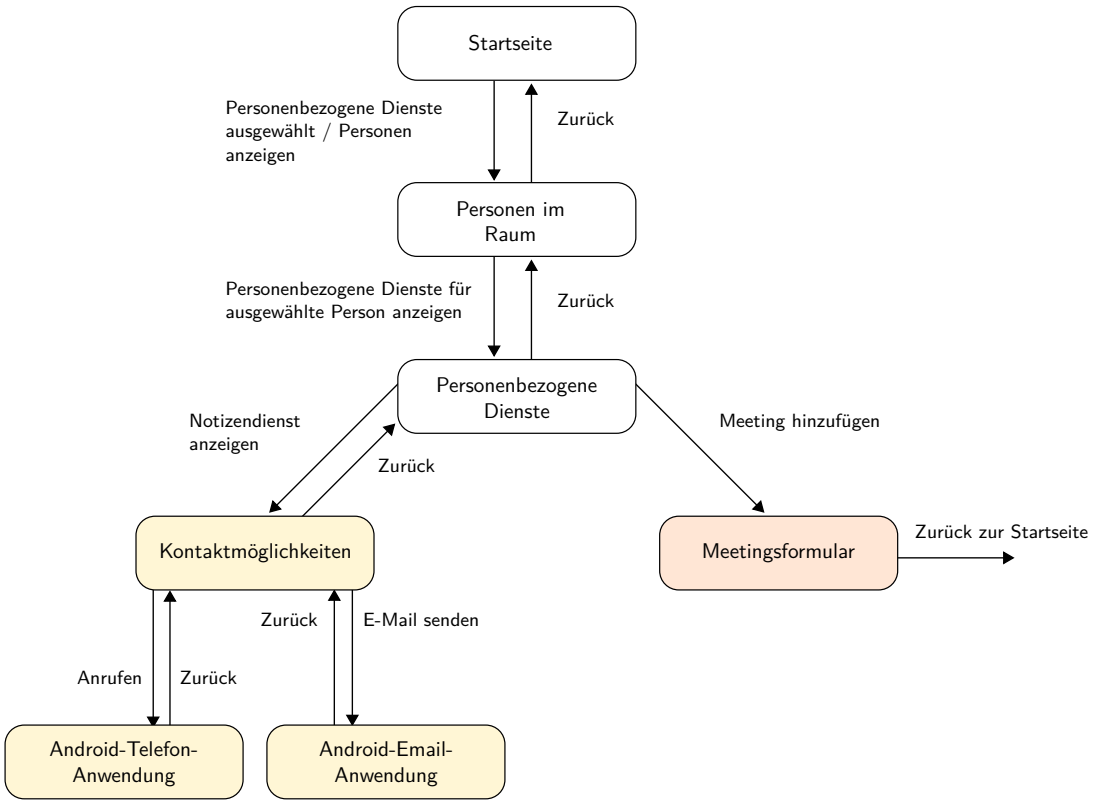

Abbildung 4.15: Zustandsdiagramm der personenbezogenen Dienste

beide Dienste zugewiesen sein müssen. Wie bereits erwähnt wird die Zuweisung der Dienste im Abschnitt 4.5.3 erläutert. Wird der Kontakt-Dienst ausgewählt, kann der Benutzer die Person über Telefon oder E-Mail kontaktieren. Durch die Auswahl einer dieser Kontaktmöglichkeiten wird auf dem Android-Smartphone die entsprechende Anwendung automatisch gestartet. Der Meeting-Dienst ist durch ein Formular realisiert, welches vom Benutzer ausgefüllt werden muss.

Der Benutzer kann zu jedem Moment die LBS-Webseite verlassen und zu dem AR-Browser zurückkehren, indem er den Zurück-Button des Smartphones betätigt.

4.5.3 Datenbank und Administration

In diesem Abschnitt wird zuerst auf die Struktur der Datenbank und anschließend auf die Administration der Datenbank eingegangen. Der Aufbau der LBS-Datenbank ist in Abbildung 4.16 zu sehen. Die Room-Tabelle ist mit der POI-Tabelle über die POI-Id verbunden. So kann jeder Raum eindeutig einem POI zugewiesen werden. Diese Verknüpfung stellt die Verbindung zwischen der AR-Datenbank und der LBS-Datenbank her. Ein Raum-Eintrag verfügt zusätzlich zu den POI-Parametern über eine Raumnummer und eine Platzanzahl. Die Reservation- und Note-Tabelle

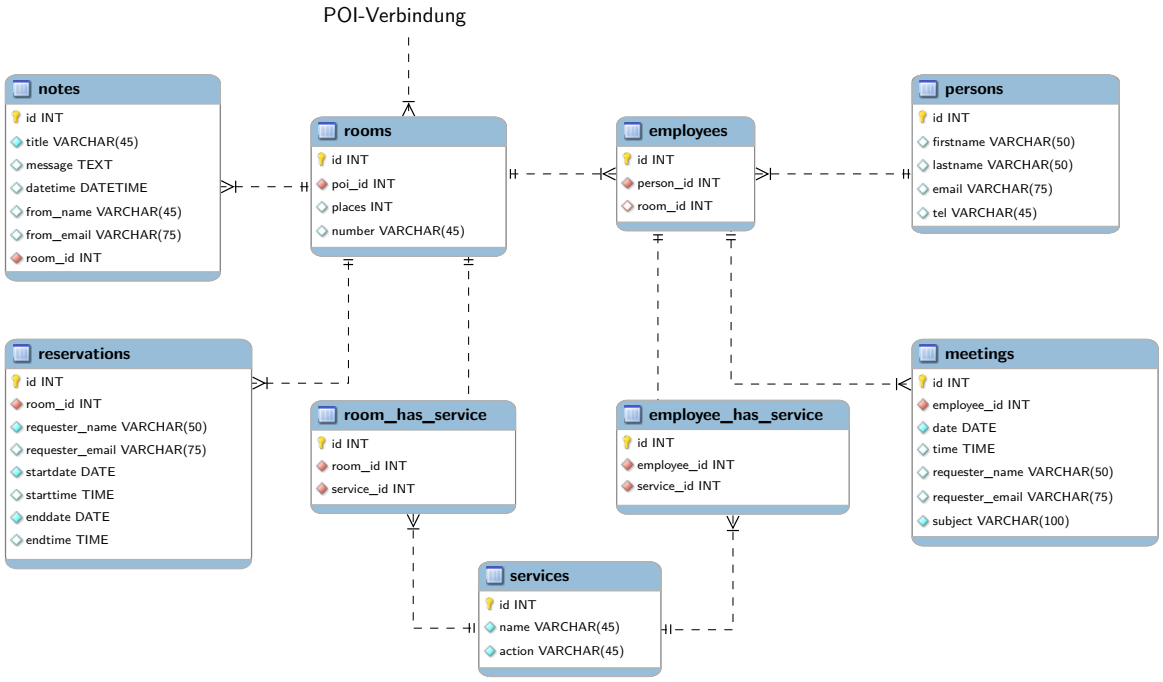

Abbildung 4.16: LBS-Datenbank (modifizierte Grafik aus MySQL Workbench)

repräsentieren die zugehörige Datenbank zu diesen Diensten. Die Tabellen sind über die Room-Id mit der Room-Tabelle verbunden. Die Employee-Tabelle verbindet Personen mit den zugehörigen Räumen. Die Person-Tabelle beinhaltet die Namen und Kontaktdaten von Personen. Die Tabelle des Meeting-Dienstes ist über die Employee-Id mit der Employee-Tabelle verbunden.

Für die Room- sowie die Employee-Tabelle existiert jeweils eine Tabelle, welche die entsprechenden Verbindungen mit einem Service speichert. Für die Room-Tabelle ist es die RoomHasService-Tabelle und für die Employee-Tabelle entsprechend die EmployeeHasService-Tabelle. Diese Tabellen ermöglichen es einen bestimmten Raum oder Angestellten mit einem Dienst zu verbinden. Die vorhandenen Dienste sind in der Service-Tabelle enthalten. Jeder Service-Eintrag besitzt einen Namen und eine Aktion. Der Aktion-Parameter des Dienstes kann entweder eine Struts-Aktion oder eine URL sein. Mit einer Struts-Aktion kann leicht ein neuer Dienst auf der LBS-Webseite eingebunden werden. Eine URL hingegen ermöglicht die Anbindung von externen Webseiten, die sich nicht in der Struts-Struktur des Servers befinden. Die Erweiterungen und der Ausbau der Dienste sind somit sehr flexibel.

Die Administration dieser Datenbanken erfolgt, wie auch bei der AR-Datenbank, über eine Webseite. Die Webseite gliedert sich in drei Bereiche: Dienste, Räume und Personen. Im Dienste-Bereich können neue Dienste hinzugefügt werden. Zudem können bestehende Dienste bestimmten Räumen oder Personen zugewiesen werden. Der Räume-Bereich dient zum einen dazu neue Räume hinzuzufügen und einem POI zuzuweisen. Zum anderen können in diesem Bereich die Notizen und

Reservierungen von Räumen angesehen und gegebenenfalls auch gelöscht werden. Im Personen-Bereich können neue Personen hinzugefügt werden und als Angestellte einem Raum zugewiesen werden. Zudem können die in der Datenbank vorhandenen Meetings betrachtet und gelöscht werden. Die standortbezogenen Dienste können also komplett über die Webseite verwaltet werden.

Kapitel 5

Zusammenfassung und Ausblick

In diesem letzten Kapitel werden die in dieser Arbeit erzielten Ergebnisse zusammengefasst sowie das entwickelte AR-System und dessen einzelnen Komponenten beschrieben. Zusätzlich werden potentielle Erweiterungen des AR-Systems aufgezeigt.

5.1 Zusammenfassung

In dieser Diplomarbeit wurden die bestehenden AR-Systeme für Smartphones analysiert und auf den gewonnenen Erkenntnissen aufbauend Erweiterungen erarbeitet. Es wurde ein AR-System für Android-Smartphones entwickelt, welches sowohl eine Interaktion mit POI als auch Navigation bereitstellt. Zudem kann das AR-System im Indoor- wie auch im Outdoor-Bereich verwendet werden. Der Übergang zwischen Indoor- und Outdoor-Bereich ist dabei ohne Unterbrechungen möglich. Außerdem wurden flexible standortbezogene Dienste an das AR-System angebunden.

Damit das AR-System im Indoor-Bereich genutzt werden kann, wurde eine Lokalisierungsanwendung entwickelt. Diese ermöglicht das Einfügen neuer Location-Provider in das Android-Betriebssystem. Die implementierten Provider basieren auf der Verwendung eines Marker-Scanners sowie eines Pedometers. Diese Provider können ohne großen Aufwand in ein neues Gebäude integriert werden. Es müssen lediglich Marker angebracht werden, welche in Form von QR-Codes auf Papier gedruckt werden können. Die Genauigkeit der Standortbestimmung anhand dieser Provider ist zum einen abhängig von der Dichte der Marker im Gebäude und zum anderen von der Genauigkeit des Pedometers.

Die AR-Komponente wurde als AR-Browser entwickelt, welcher die Interaktion mit POI und Navigationsfunktionen ermöglicht. Der AR-Browser musste von Grund auf neu entwickelt werden, da die bestehenden AR-Browser keinen Freiraum für Erweiterungen bieten. Die anzuzeigenden virtuellen Elemente, also POI und Navigationsdaten, werden von einem Server bezogen. Der AR-Browser nutzt die von der Lokalisierungsanwendung bereitgestellten Positionsangaben um die virtuellen Elemente im Bildschirm darzustellen. Die virtuellen Elemente werden also bezogen auf

ihre Koordinaten angezeigt. Die Genauigkeit der Darstellung der virtuellen Elemente hängt von der Genauigkeit der ermittelten Position, also der Standortbestimmung, ab. Ist die ermittelte Benutzerposition ungenau, werden die virtuellen Objekte versetzt angezeigt, wodurch die Bedienbarkeit des AR-Systems verschlechtert wird. Je genauer das Lokalisierungssystem die Position ermittelt, desto präziser sind die Anzeigen im AR-Browser.

Die POI des AR-Browsers sind an anpassbare standortbezogene Dienste angebunden. Die in dieser Arbeit implementierten Dienste wurden für große Bürogebäude ausgelegt. Sie unterteilen sich in raumbezogene und personenbezogene Dienste. Die raumbezogenen Dienste ermöglichen zum Beispiel das Reservieren eines Seminarraums. Anhand der personenbezogenen Dienste kann beispielsweise ein Termin mit einem Angestellten angefragt werden.

Insgesamt wurde ein modulares AR-System entwickelt, welches aus einer Lokalisierungsanwendung, einem AR-Browser sowie standortbezogenen Diensten besteht. Zusammenfassend lässt sich feststellen, dass die an das AR-System gestellten Anforderungen im konzipierten und implementierten System erfolgreich umgesetzt wurden.

5.2 Ausblick

In diesem Abschnitt werden mögliche Erweiterungen des entwickelten AR-Systems vorgestellt. Zuerst wird auf einen möglichen Ausbau der Standortbestimmung eingegangen. Anschließend werden mögliche Weiterentwicklungen des AR-Browsers beschrieben.

5.2.1 Standortbestimmung

Ein wichtiger Bestandteil für die Bedienbarkeit des AR-Browsers ist die Genauigkeit der Standortbestimmung. Diese stellt insbesondere im Indoor-Bereich eine Herausforderung dar. Um die Genauigkeit der Standortbestimmung im Indoor-Bereich zu erhöhen, können weitere Location-Provider in der Lokalisierungsanwendung hinzugefügt werden, die zum Beispiel Funksignale oder die Kamera nutzen. Ein Beispiel hierfür wäre die Standortbestimmung mittels WiFi. Zudem können die bereits implementierten Provider noch verbessert werden. Der Pedometer kann zum Beispiel durch komplexere Algorithmen für die Auswertung von Schritten optimiert werden. So könnte der Pedometer sich beispielsweise automatisch an den Benutzer anpassen oder sogar Höhenunterschiede anhand der Schritte feststellen. Beim Scannen von Markern könnte zusätzlich zu den im Marker vorhandenen Positionsdaten auch die Position, aus welcher der Marker gescannt wurde, verwendet werden. Diese genauere Benutzerposition könnte anhand des Winkels und der Distanz, aus welcher ein Marker gescannt wurde, ermittelt werden. Dies ist vor allem bei größeren Markern sinnvoll, da in diesem Fall der Scanvorgang aus größerer Entfernung erfolgen kann.

In neue Smartphones wird immer häufiger innovative Hardware verbaut, die für die Standortbestimmung eingesetzt werden könnte. So enthält zum Beispiel das kürzlich erschienene Galaxy Nexus einen Barometer. Dieses von Google und Samsung entwickelte Smartphone kann mit Hilfe des neuen Hardware-Moduls die Höhe des Benutzers ermitteln und damit die Genauigkeit der Standortbestimmung verbessern.

Zusätzlich könnten die Standortinformationen insofern optimiert werden, dass die Sensoren nicht nur einzeln betrachtet, sondern die Messungen der Sensoren miteinander verbunden werden.

5.2.2 AR-Browser

Es bestehen mehrere Möglichkeiten den AR-Browser zu erweitern. Zum Beispiel können vorhandene Komponenten wie die POI-Interaktion und die Navigation weiter ausgebaut werden. Zudem kann auch die Benutzerfreundlichkeit der Anwendung noch weiter erhöht werden oder der AR-Browser durch neue Funktionen erweitert werden.

POI-Interaktion Der POI-Modus könnte zum Beispiel um eine Suchfunktion ergänzt werden. Mit Hilfe einer solchen Suchfunktion könnte eine sofortige Suche und Anzeige eines bestimmten POI ermöglicht werden. Zudem könnte ein provider-spezifischer oder ein provider-übergreifender Modus der Suche realisiert werden. Zusätzlich könnte optional in deaktivierten Providern gesucht werden, welche dann bei erfolgreicher Suche automatisch aktiviert werden würden. Die Suche könnte sich sowohl auf die POI-Namen wie auch auf deren Beschreibung beziehen.

Navigation Die Navigation könnte zum Beispiel um verschiedene Ansichten erweitert werden. Im Outdoor-Bereich könnten, je nach Fortbewegungsart, verschiedene Ansichten zur Verfügung gestellt werden. Für eine Outdoor-Strecke, die mit dem Auto zurückgelegt werden soll, könnte eine für diese Fortbewegungsart angepasste Ansicht eingeblendet werden. In der implementierten Navigation sind die Anzeigen für eine Verwendung zu Fuß ausgelegt. Die Anwendung Wikitude Drive [34] zeigt, wie ein AR-Navigationssystem für die Nutzung im Auto auf dem Smartphone aussehen kann. Auch im Indoor-Bereich könnten mehrere Ansichten zur Verfügung gestellt werden. So könnten zum Beispiel Richtungsangaben wie „In 20 Metern rechts abbiegen“ eingeblendet werden. Ferner könnte beispielsweise eine Ansicht mit Pfeilen auf dem Fußboden die Navigation erleichtern.

Benutzerfreundlichkeit In der aktuellen Implementierung muss der Benutzer das Scannen von QR-Codes explizit durch Betätigen eines Buttons starten. Die Benutzerfreundlichkeit könnte erhöht werden, indem die Anwendung fortwährend nach Markern sucht und diese automatisch scannt. Weiterhin bestehen noch Möglichkeiten zur Erweiterung der Kartenansicht. Hier könnten

zum Beispiel durch Auswählen einer Stelle auf der Karte die Navigationsdaten eingegeben werden. Somit könnte der Benutzer das Ziel- und den Startpunkt direkt in der Kartenansicht festlegen. Die Benutzerfreundlichkeit könnte zudem durch einen Offline-Modus erhöht werden. In einem solchen Modus könnte der Benutzer dann beispielsweise AR-Inhalte für eine spätere Nutzung herunterladen. Somit hätte er die Möglichkeit auch ohne permanente Internetverbindung den AR-Browser nutzen zu können.

Neue Funktionen Zusätzlich zu der POI-Interaktion und der Navigation könnte die AR-Technologie noch für weitere Funktionen genutzt werden. Der AR-Browser kann zum Beispiel um eine Freunde-Funktion erweitert werden, welche in der Nähe befindliche Freunde in der AR-Ansicht anzeigt und eine Interaktion mit diesen ermöglicht. Eine weitere sinnvolle Anwendung für den AR-Browser wäre eine Notizen-Funktion. Anhand dieser Funktion könnten Notizen in der AR-Ansicht für andere Benutzer hinterlassen werden. Diese Notizen könnten dann verwendet werden, um Freunden Kommentare an verschiedenen Orten zu hinterlassen.

Zusammenfassend bleibt festzustellen, dass vielseitige Möglichkeiten für einen weiteren Ausbau der Standortbestimmung und des AR-Browsers bestehen.

Anhang A

Schnittstelle der Lokalisierungsanwendung

Das Interface muss in der Anwendung, welche die Positionsangaben an die Lokalisierungsanwendung senden soll, vorhanden sein. Das Interface wird in das Quellcode-Verzeichnis der Anwendung eingefügt und der nachfolgende Quellcode in einer Datei namens IExternService.aidl unter org.localization.interfaces gespeichert. Die AIDL-Endung steht für Android Interface Definition Language. Diese definiert die Programmierschnittstellen für die Interprozess-Kommunikation.

```
package org.localization.interfaces;

interface IExternService {
 void sendLocationUpdate(double latitude, double longitude,
    double altitude, double accuracy);
 void stopLocalizationService();
 boolean checkServiceRunning();
 boolean checkServiceConfigured();
}
```

Nachdem das Interface in der Anwendung eingefügt wurde, stehen dieser die enthaltenen Funktionen zur Verfügung.

Abbildungsverzeichnis

Abkürzungsverzeichnis

2D	Zweidimensional
3D	Dreidimensional
A-GPS	Assisted Global Positioning System
AIDL	Android Interface Definition Language
API	Application Programming Interface
APK	Android Package
AR	Augmented Reality
ARML	Augmented Reality Markup Language
CSS	Cascading Style Sheets
DVK	Dalvik Virtual Machine
E-OTD	Enhanced Observed Time Difference
GIS	Geographical Information System
GPS	Global Positioning System
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IPC	Interprocess Communication
Java EE	Java Enterprise Edition
JDBC	Java Database Connectivity
JSP	JavaServer Pages
KML	Keyhole Markup Language
LBS	Location Based Services
MVC	Model View Controller
OGC	Open Geospatial Consortium
OTDoA-IPDL	Observed Time Difference of Arrival With Idle Period Downlink
PDA	Personal Digital Assistant
POI	Point of Interest
QR	Quick Response
SDK	Software Development Kit
SQL	Structured Query Language
U-TDoA	Uplink Time Difference of Arrival
URI	Uniform Resource Identifier

URL . . . . . Uniform Resource Locator
VMI . . . . . Verteilte Multimodale Informationsverarbeitung
WGS84 . . . . . World Geodetic System 1984
XML . . . . . Extensible Markup Language

Literaturverzeichnis

[1] Gartner, "Gartner says worldwide mobile device sales to end users reached 1.6 billion units in 2010; smartphone sales grew 72 percent in 2010." http://www.gartner.com/it/page.jsp?id=1543014. [Letzter Zugriff: 23. November 2011].

[2] Open Handset Alliance, "Industry leaders announce open platform for mobile devices." http://www.openhandsetalliance.com/press_110507.html, November 2007. [Letzter Zugriff: 24. November 2011].

[3] HTC, "HTC-Webseite." http://www.htc.com. [Letzter Zugriff: 7. Dezember 2011].

[4] Samsung, "Samsung-Webseite." http://www.samsung.com. [Letzter Zugriff: 7. Dezember 2011].

[5] LG Electronics, "LG-Webseite." http://www.lg.com. [Letzter Zugriff: 7. Dezember 2011].

[6] Google, "Google-Webseite." http://www.google.com. [Letzter Zugriff: 7. Dezember 2011].

[7] Gartner, "Gartner says Android to command nearly half of worldwide smartphone operating system market by year-end 2012." http://www.gartner.com/it/page.jsp?id=1622614. [Letzter Zugriff: 23. November 2011].

[8] Android, "The developer's guide." http://developer.android.com/guide/index.html. [Letzter Zugriff: 23. November 2011].

[9] J. Krumm, *Ubiquitous Computing Fundamentals*. Chapman & Hall/CRC, 1. Aufl., 2009.

[10] NIMA, "Department of defense world geodetic system 1984 – its definition and relationships with local geodetic systems," Tech. Rep. 8350.2, National Imagery and Mapping Agency, 2004.

[11] National Coordination Office for Space-Based Positioning, Navigation, and Timing and the Civil GPS Service Interface Committee, "Offizielle GPS-Webseite." http://www.gps.gov. [Letzter Zugriff: 6. Dezember 2011].

[12] A. Kupper, *Location-based Services: Fundamentals and Operation*. John Wiley & Sons, 2005.

[13] Canalys, "Almost 40% of smart phones shipping in EMEA have GPS integrated." http://www.canalys.com/newsroom/almost-40-smart-phones-shipping-emea-have-gps-integrated, August 2008. [Letzter Zugriff: 27. November 2011].

[14] P. Misra und P. Enge, *Global Positioning System: Signals, Measurements and Performance*. Ganga-Jamuna Press, 2. Aufl., 2006.

[15] WiFi Alliance, "WiFi Alliance-Webseite." http://www.wi-fi.org. [Letzter Zugriff: 15. Dezember 2011].

[16] P. Bahl und V. Padmanabhan, "RADAR: an in-building RF-based user location and tracking system," in *Proceedings of the nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, Band 2 von *IEEE INFOCOM 2000*, Seiten 775–784, 2000.

[17] E. Elnahrawy, X. Li, und R. Martin, "The limits of localization using signal strength: a comparative study," in *Proceedings of the first Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks*, IEEE SECON 2004, Seiten 406–414, Oktober 2004.

[18] A. Varshavsky, E. de Lara, J. Hightower, A. LaMarca, und V. Otsason, "GSM indoor localization," *Pervasive and Mobile Computing*, Band 3, Seiten 698–720, Dezember 2007.

[19] H. Hile und G. Borriello, "Positioning and orientation in indoor environments using camera phones," *Computer Graphics and Applications, IEEE*, Band 28, Seiten 32–39, Juli–August 2008.

[20] A. Mulloni, D. Wagner, I. Barakonyi, und D. Schmalstieg, "Indoor positioning and navigation with camera phones," *Pervasive Computing, IEEE*, Band 8, Seiten 22–31, April–Juni 2009.

[21] Y.-J. Chang, S.-K. Tsai, Y.-S. Chang, und T.-Y. Wang, "A novel wayfinding system based on geo-coded qr codes for individuals with cognitive impairments," in *Proceedings of the 9th international ACM SIGACCESS conference on Computers and accessibility*, Assets '07, (New York, NY, USA), Seiten 231–232, ACM, 2007.

[22] K. Hattori, R. Kimura, N. Nakajima, T. Fujii, Y. Kado, B. Zhang, T. Hazugawa, und K. Takadama, "Hybrid indoor location estimation system using image processing and wifi strength," in *International Conference on Wireless Networks and Information Systems*, WNIS '09, Seiten 406–411, Dezember 2009.

[23] J. Kim und H. Jun, "Vision-based location positioning using augmented reality for indoor navigation," *IEEE Transactions on Consumer Electronics*, Band 54, Seiten 954–962, August 2008.

[24] A. Parnandi, K. Le, P. Vaghela, A. Kolli, K. Dantu, S. Poduri, und G. S. Sukhatme, "Coarse in-building localization with smartphones," in *MobiCASE*, Seiten 343–354, 2009.

[25] I. Bylemans, M. Weyn, und M. Klepal, "Mobile phone-based displacement estimation for opportunistic localisation systems," in *Third International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies*, UBICOMM '09, Seiten 113–118, Oktober 2009.

[26] E. Martin, O. Vinyals, G. Friedland, und R. Bajcsy, "Precise indoor localization using smart phones," in *Proceedings of the international conference on Multimedia*, MM '10, (New York, NY, USA), Seiten 787–790, ACM, 2010.

[27] I. E. Sutherland, "A head-mounted three dimensional display," in *Proceedings of the December 9-11, 1968, fall joint computer conference, part I*, AFIPS '68, (New York, NY, USA), Seiten 757–764, ACM, 1968.

[28] P. Milgram, H. Takemura, A. Utsumi, und F. Kishino, "Augmented reality: A class of displays on the reality-virtuality continuum," in *SPIE Proceedings of Telemanipulator and Telepresence Technologies*, Band 2351-34, 1994.

[29] P. Milgram und K. Fumio, "A taxonomy of mixed reality virtual displays," in *IEICE Transactions on Information Systems*, Band E77-D, 1994.

[30] R. T. Azuma, "A survey of augmented reality," in *Teleoperators and Virtual Environments 6*, Seiten 355–385, August 1997.

[31] C. Geiger, B. Kleinnjohann, C. Reimann, und D. Stichling, "Mobile AR4ALL," in *Proceedings of the IEEE and ACM International Symposium on Augmented Reality*, Seiten 181–182, 2001.

[32] M. Mohring, C. Lessig, und O. Bimber, "Video see-through ar on consumer cell-phones," in *Proceedings of the 3rd IEEE/ACM International Symposium on Mixed and Augmented Reality*, Seiten 252–253, November 2004.

[33] Layar, "Layar AR-Browser." http://www.layar.com. [Letzter Zugriff: 5. Dezember 2011].

[34] Wikitude GmbH, "Wikitude AR-Browser." http://www.wikitude.com. [Letzter Zugriff: 5. Dezember 2011].

[35] Metaio, "Junaio AR-Browser." http://www.junaio.com. [Letzter Zugriff: 5. Dezember 2011].

[36] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, und T. Berners-Lee, "Hypertext Transfer Protocol – HTTP/1.1." RFC 2616 (Draft Standard), Juni 1999. Updated by RFCs 2817, 5785.

[37] Wikitude GmbH, "ARML-Standard." http://www.openarml.org. [Letzter Zugriff: 5. Dezember 2011].

[38] Open Geospatial Consortium, "OGC-Webseite." http://www.opengeospatial.org. [Letzter Zugriff: 5. November 2011].

[39] B. Thomas, V. Demczuk, W. Piekarski, D. Hepworth, und B. Gunther, "A wearable computer system with augmented reality to support terrestrial navigation," in *Proceedings of the 2nd IEEE International Symposium on Wearable Computers*, Seiten 168–171, Oktober 1998.

[40] T. Höllerer, D. Hallaway, N. Tinna, und S. Feiner, "Steps toward accommodating variable position tracking accuracy in a mobile augmented reality system," in *Second Int. Workshop on Artificial Intelligence in Mobile Systems*, AIMS '01, Seiten 31–37, August 2001.

[41] G. Reitmayr und D. Schmalstieg, "Location based applications for mobile augmented reality," in *Proceedings of the Fourth Australasian user interface conference on User interfaces*, Band 18 von *AUIC '03*, (Darlinghurst, Australia, Australia), Seiten 65–73, Australian Computer Society, Inc., 2003.

[42] O. Mohareri und A. Rad, "Autonomous humanoid robot navigation using augmented reality technique," in *Proceedings of the 2011 IEEE International Conference on Mechatronics*, Seiten 463–468, April 2011.

[43] J. H. Schiller und A. Voisard, *Location-based services*. Morgan Kaufmann Publishers, 2004.

[44] A. Brimicombe und C. Li, *Location-based services and geo-information engineering*. John Wiley & Sons, 2009.

[45] Goolge, "Google-Maps." http://maps.google.de. [Letzter Zugriff: 15. Dezember 2011].

[46] Open Geospatial Consortium, "KML-Standard." http://www.opengeospatial.org/standards/kml. [Letzter Zugriff: 14. Dezember 2011].

[47] Goolge, "Google-API." http://code.google.com/android/add-ons/google-apis. [Letzter Zugriff: 15. Dezember 2011].

[48] The Apache Software Foundation, "Apache Struts-Webseite." http://struts.apache.org. [Letzter Zugriff: 15. Dezember 2011].

[49] jQuery, "Javascript library." http://jquerymobile.com. [Letzter Zugriff: 22. Dezember 2011].

[50] L. Bagi, "Pedometer – Android application." http://code.google.com/p/pedometer. [Letzter Zugriff: 27. November 2011].

[51] Digiwalker, "Digiwalker-Webseite." http://www.digiwalker.co.uk. [Letzter Zugriff: 15. Dezember 2011].

[52] ISO, *ISO/IEC 18004:2006 Information technology – Automatic identification and data capture techniques – QR Code 2005 bar code symbology specification*. Geneva, Switzerland: International Organization for Standardization, 2006.

[53] ZXing (Zebra Crossing), "1D/2D barcode image processing library." http://code.google.com/p/zxing. [Letzter Zugriff: 19. Dezember 2011].