



Masterarbeit

Lighthouse Keeper Ein neues Verfahren zur Planung und Evaluation von iBeacon Konfigurationen

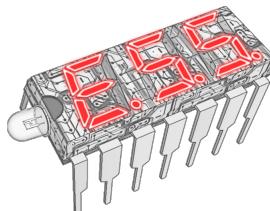
von

André Alexander Pieper
Geb. 28.03.1988 in Berlin
Matrikelnummer: 184960

30. April 2015

Erstprüfer: Jun.-Prof. Dr.-Ing. Sebastian Zug
Otto-von-Guericke-Universität
Fakultät für Informatik
Institut für Verteilte Systeme
Lehrstuhl Embedded Smart Systems
Universitätsplatz 2, D-39106, Magdeburg

Zweitprüfer: Prof. Dr.-Ing. Abbas Omar
Otto-von-Guericke-Universität
Fakultät für Elektro- und Informationstechnik
Institut für Informations- und Kommunikationstechnik
Lehrstuhl für Hochfrequenz- und Kommunikationstechnik
Universitätsplatz 2, D-39106, Magdeburg



Kurzdarstellung

Satellitengestützte Navigationssysteme sind in der heutigen Zeit ein fester Bestandteil des alltäglichen Lebens. Sie ermöglichen zum Beispiel die Orientierung eines Autofahrers auf ihm unbekannten Straßen und sind mittlerweile serienmäßig in Autos, Smartphones und sogar Kameras integriert. Aber die Signale der satellitengestützten Lokalisierungs-Systeme wie dem „Global Positioning System“ (GPS), „Globalnaja nawigazionnaja sputnikowaja sistema“ (GLONASS) und „Galileo“ verlieren sich in Gebäuden, da sie durch deren Materialien absorbiert, bzw. reflektiert werden. Dabei soll, so wie die „Outdoor“-Lokalisierungssysteme den Straßenverkehr revolutionierten, ein neues „Indoor“-System die Fortbewegung von Menschen in Gebäuden verändern: die „iBeacons“ oder zu Deutsch Leuchtfeuer. Obwohl bereits viele Unternehmen diese Technik herstellen und vermarkten, existiert im Moment noch kein zufriedenstellendes Konzept, das es ermöglicht, die nötige Infrastruktur für ein Lokalisierungssystem aus iBeacons effizient zu planen und zu testen. Um das Potential dieser aufstrebenden Technologie auszuschöpfen, liegt der Forschungsschwerpunkt der folgenden Ausführungen auf der Entwicklung eines dafür möglichen Verfahrens und dessen Automation mithilfe von Robotern. In Anspielung auf die Bedeutung der iBeacons wird das Gesamtkonzept als „Lighthouse Keeper“ benannt - frei übersetzt „Herr der Leuchtfeuer“.

Inhaltsverzeichnis

Abbildungsverzeichnis	IV
Tabellenverzeichnis	V
Abkürzungsverzeichnis	VI
1 Einleitung	1
2 Stand der Technik – Einführung in die Beacon-Technologie	4
2.1 Entwicklungsgeschichte	4
2.2 Aufbau und Funktionsweise von Beacons	5
2.3 Ortungsmethoden	7
2.3.1 Definitionen verschiedener Anwendungsfelder	8
2.3.2 Mikro-Lokalisierung	9
2.4 Anwendungsbereiche der Indoor-Lokalisierung	10
2.4.1 Mögliche Einsatzszenarien	10
2.4.2 Planungskonzepte für Beacon-Konfigurationen	12
3 Konzeptplanung	14
3.1 Vorüberlegungen	14
3.2 Zielsetzungen	15
3.2.1 Hardware-Anforderungen	15
3.2.2 Konzept der Software-Architektur	16
4 Software-Entwicklung und Experimente	19
4.1 Verwendete Hardware	19
4.1.1 Motorola Moto G	20
4.1.2 Youbot	21
4.1.3 Scitos G5	21
4.2 Verwendete Software	22
4.2.1 Robot Operating System – ROS	23
4.2.2 Miracenter	24
4.2.3 Android-Studio, Estimote SDK und ROSjava	25
4.3 App-Entwicklung	25
4.3.1 Beacon-Detektierung	25
4.3.2 Lagemessung	26

4.3.3	ROS-Anbindung	28
4.3.4	Programmablauf	28
4.4	Experimente	31
4.4.1	Messung der BLE-Signalausbreitung	31
4.4.2	Auswirkung der Smartphone-Orientierung	32
4.4.3	Wechselwirkungen zwischen Signalquellen	34
4.4.4	Energieverbrauch eines Beacon	35
Literaturverzeichnis		38

Abbildungsverzeichnis

1.1	Skizze satellitengestütztes Lokalisierungssystem [11]	1
2.1	Offizielles Bluetooth Smart Logo [9]	5
2.2	Explosionszeichnung Beacon [14]	5
2.3	Signalschwächung durch Objekte, z.B. Wände [6]	6
2.4	menschliche Körper blockieren zusätzlich die Signale [6]	6
2.5	Beispiel einer Trilateration zur Positionsbestimmung	9
2.6	Beacon Nutzung in Geschäften [1]	10
2.7	Beacon Nutzung im Flughäfen [35]	11
2.8	Demo der Estimote Indoor Localization App [12]	13
3.1	Kreislauf einer Prozessplanung	15
3.2	Überblick auf die Software-Anforderungen	17
3.3	Überblick auf die Software-Anforderungen	18
4.1	Vorderseite des Motorola Moto G [5]	20
4.2	Rückseite vom Moto G mit Antennen-Gerüst [7]	20
4.3	Youbot mit Halterung (gelber Aufsatz)	21
4.4	Zeichnung eines Youbot-Arms und seiner fünf Rotationsachsen [21]	21
4.5	Scitos G5 von der MetraLabs GmbH	22
4.6	Erstellter Grundriss eines Flures in Gebäude 29 der OvGU, Stockwerk 3	24
4.7	Aufbau des Komplementärfilters, in Anlehnung an [28]	27
4.8	Quellcode-Beispiel eines Message-Paketes für eine ROSjava-Implementierung	28
4.9	Programmablaufplan der Lighthouse Keeper Applikation für Android-Smartphones	30
4.10	Gleichzeitige Messung dreier naheliegender Beacon-Signale	35
4.11	Anteile eines Datenpaket in der Beacon-Kommunikation auf der MAC-Ebene	36
4.12	Einfluss der einstellbaren Parameter auf die Lebensdauer einer CR2450-Batterie	37

Tabellenverzeichnis

2.1 Beispiel der Informationsnutzung; in Anlehnung an: [6]	7
2.2 Beispiel der Bereichsdefinition; in Anlehnung an: [6]	9

Abkürzungsverzeichnis

GPS – Global Positioning System

GLONASS – Globalnaja nawigazionnaja sputnikowaja sistema

ULP – Ultra Low Power Bluetooth

BLE – Bluetooth Low Energy

WLAN – Wireless Local Area Network

RSSI – Received Signal Strength Indication

UUID – Universally Unique Identifier

ROS – Robot Operating System

P2P – Peer to Peer

PNG – Portable Network Graphics

XML – Extensible Markup Language

IDE – Integrated Development Environment

SDK – Estimote Software Development Kit

IMU – Inertial Measurement Unit

IP – Internetprotokoll

1 Einleitung

Herkömmliche Navigationssysteme für den „Outdoor“-Bereich haben viele Handlungen des alltäglichen Lebens weitgehend erleichtert. Die Tage gehören der Vergangenheit an, in denen Fahrtstrecken über unbekannte Straßen und durch fremde Länder mit analogen Karten weit im Voraus geplant werden mussten. Die Navigation, bzw. Lokalisierung basiert dabei auf einer Positionsbestimmung mithilfe von Funksignalen, die von Satelliten in der Erdumlaufbahn ausgesendet werden. Aus diesen Signalen können die Navigationssysteme ihre Position auf der Erde berechnen und schließlich dem Nutzer die Information zur Verfügung stellen. Die eigene Lokalisierung wäre nun auch für den Besucher in einem Gebäude hilfreich, damit er sich orientieren kann und beispielsweise in einem weitläufigen Kaufhaus schneller zu einem Geschäft seiner Wahl findet. Um die Vorteile einer Positionsbestimmung im „Indoor“-Bereich anzuwenden, können die bisherigen satellitengestützten Systeme (GPS, GLONASS, Galileo) jedoch noch nicht genutzt werden, da die massive Bauweise der Gebäude ihre Signale absorbiert bzw. reflektiert. Seit einigen Jahren arbeiten Forschungseinrichtungen und Unternehmen an einer Lösung für die Lokalisierung von Objekten im Innern von Gebäuden. Die Forschungsprojekte verfolgen dabei viele verschiedene Ansätze und Technologien zur Erreichung dieses Ziels, wie beispielsweise die Projekte „EVARILOS“ [34], „Google Indoor Maps“ [17] und „Mobile Indoor Localization“ [24] zeigen. Jedoch existiert noch keine Lösung, die sich bereits durchgesetzt hat und einen kommerziellen Nutzen erzielt.

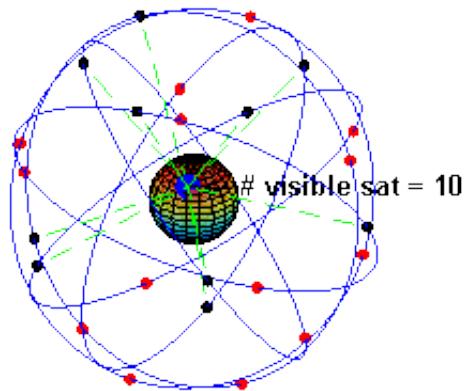


Abb. 1.1: Skizze satellitengestütztes Lokalisierungssystem [11]

Eine neue und vielversprechende Technologie zur innerräumlichen Lokalisierung stellt dabei die Entwicklung von iBeacons (z. Dt. Leuchtfeuer) dar. Die Vorteile dieses Systems gegenüber bisherigen Lösungen sind zum einen die geringen Kosten, sowie die hohe Flexibilität und Autonomie der einzelnen Elemente. Viele bisherigen Technologien zur Indoor-Lokalisierung, wie z.B. in den Boden eingelassene künstliche Magnetfelder [20], RFID-Transponder [2] oder funkbasierter Lösungen [36] sind zwar erprobt und erzielen eine hohe Lokalisierungsgenauigkeit, besitzen jedoch den Nachteil einer kostenintensiven und aufwendigen Infrastruktur, welche meist

auch eine bauliche Veränderung am Gebäude erfordert. Die Einfachheit der Anbringung der Beacons, die Kostenvorteile und die weitere Verbreitung des verwendeten Bluetooth-Protokolls auf mobilen Geräten sorgen dabei für ein weites Einsatzspektrum. Der größte Vorteil der iBeacons gegenüber konkurrierender Technologien stellt die variable Verwendung der Beacons beispielsweise für einfache, rudimentäre Positionsbestimmungen bis hin zur hochpräzisen Ortung dar. Je nach Einsatzszenario können die Beacons problemlos angepasst und für ihre Bestimmung optimiert werden.

Um mit den Beacons ein Lokalisierungssystem in Gebäuden aufzubauen, ähnlich dem im Outdoor-Bereich, benötigt es geeignete Verfahren, um die Beacons zu positionieren und entsprechend auf ihres Einsatzzweckes anzupassen. Während für die Satelliten der Outdoor-Systeme Modelle vorhanden sind, die deren Anzahl, Flughöhe und Sendeleistung für eine bestmögliche Lokalisierung berechnen, fehlen die Konzepte für den Indoor-Bereich. Zudem mangelt es an Verfahren, die Gebäude mit Beacon-Systemen zweifelsfrei auf die Qualität der Lokalisierung und der räumlichen Abdeckung hin validieren. Gegenstand dieser Arbeit ist es, ein geeignetes Konzept für die Erstellung von Beacon-Konfigurationen zu gestalten und die Anordnung anschließend experimentell zu validieren. Dabei wird auf ein automatisiertes und strukturiertes Verfahren mithilfe von Robotern zurückgegriffen, um die Validierung weniger fehleranfällig zu designen und somit zu standardisieren.

Aus diesen Anforderungen stellen sich drei zentrale Zielstellungen:

- Entwicklung eines Frameworks und einer systematischen Versuchsplanung für die Messung von Beacon-Signalen
- Auswahl und Parameterbestimmung eines Modells für die Signalausbreitung von Beacons mit anschließender Erstellung einer Simulationsumgebung, sowie eine sich daraus konkludierenden optimalen Verteilung von Beacons in einem Raum
- Validierung des Modells und der Simulationsergebnisse in einem Testszenario mithilfe eines Roboters

Um die Problematik eines bisher fehlenden Planungskonzeptes darzulegen, wird zuerst der Stand der Technik von Beacons beschrieben und erläutert. Daraus schlussfolgern sich in Kapitel 3 erste Ansätze, darüber wie ein mögliches Verfahren zur Beacon-Konfiguration aufgebaut sein muss. In Kapitel 4 werden für die erste Zielsetzung der Modellierung von Beacons alle Werkzeuge zur Messung der Beacon-Signale vorgestellt und ausgewertet. Mit den Messwerten werden anschließend die Parameter für die Signalausbreitung der Beacons in einem freien Raum bestimmt und damit die Grundlage für ein Simulationsprogramm gelegt. Da noch keine bekannten Veröffentlichungen über die Lokalisierungsgenauigkeit mittels Beacons existieren, wird zusätzlich aus den Messwerten dafür eine Qualitäts-Skala bestimmt und diese erklärt. Mithilfe der Simulation und eines Optimierungsalgorithmus werden anhand der selbstdefinierten Qualitätskriterien geeignete Beacon-Konfigurationen berechnet und diese experimentell in Kapitel 6 überprüft. Dabei wird besonders auf Art der Validierung eingegangen, da sie

automatisiert und standardisiert von einem Roboter durchgeführt wird. Am Ende der Arbeit werden die drei verfassten Zielstellungen mit dem Erreichten verglichen, die Ergebnisse daraus diskutiert und anschließend wird das Konzept danach bewertet.

2 Stand der Technik – Einführung in die Beacon-Technologie

Der erste Teil dieses Kapitels beschäftigt sich mit der Entstehungsgeschichte von iBeacons und nachfolgend mit deren Funktionsweise. Hier wird besonders auf die technischen Möglichkeiten der Technologie eingegangen. Im letzten Abschnitt wird erläutert, wo Beacon-Systeme zum Einsatz kommen und wie aktuelle Lösungsansätze für die Planung der dafür nötigen Infrastruktur aufgebaut sind. Anschließend wird aus den gegebenen Charakteristika der Beacons und der momentanen genutzten Fähigkeiten differenziert und eine Aussicht auf ein zukünftiges Konzept zu einer besseren Nutzung gegeben.

2.1 Entwicklungsgeschichte

Der Grundstein für die Beacon-Technologie legte die Firma Nokia im Jahre 2006. Damals entwickelte die Firma den neuen Standard *Wibree* für die Funkübertragung, der den alten Bluetooth-Standard ersetzen sollte. Mit der Neuentwicklung versprach man sich im Gegensatz zu Bluetooth einen geringeren Stromverbrauch und geringere Produktionskosten, bei gleichbleibenden Übertragungsraten. Ab dem Jahr 2009 wurde der Bluetooth-Standard um Wibree ergänzt und erst unter den Bezeichnungen *Ultra Low Power Bluetooth* (ULP) und dann später als *Bluetooth Low Energy* (BLE) darin aufgenommen [8] und anschließend als *Bluetooth Smart* vermarktet. Da viele Hersteller von mobilen Geräten in ihren Datenblättern die Unterstützung von BLE nicht explizit erwähnen, findet sich meistens folgendes Logo 2.1 in den Produktbeschreibungen und lässt erkennen, ob die Geräte den neuen Standard unterstützen.

Die Idee der Nutzung von Bluetooth Low Energy zur Indoor-Lokalisierung stammt dabei von der Firma Apple Inc. und wurde von ihr im Jahre 2013 auf der WWDC (Worldwide Developers Conference)[10] unter dem Namen *iBeacon* angekündigt. Obwohl zu dem Zeitpunkt noch kein fertiges Gerät zur Verfügung stand, wurde diese Technologie als Neuerung in Apples mobilem Betriebssystem iOS 7 vorgestellt. Jedoch verzichtet Apple seither auf die Produktion von iBeacons, was andere Unternehmen nutzen um selbst in den Markt einzusteigen. Deren Produkte wurden darauf in Beacons umbenannt und unterstützen zusätzlich die mobilen Betriebssysteme Android ab Version 4.3, Windows Phone 8 und neue die neueste Version von Blackberries OS [3]. Somit wäre die Beacon-Technologie mit der nötigen Hardware-Unterstützung in

mittlerweile über 99,5% aller mobilen Geräte (Smartphones, Tablets, Smartwatches, etc.) weltweit nutzbar [25].



Abb. 2.1: Offizielles Bluetooth Smart Logo [9]

2.2 Aufbau und Funktionsweise von Beacons

Die Grundbausteine der kleinen Leuchtfelder sind in der rechten Abbildung 2.2 ersichtlich, in der ein *Estimote Beacon* der Firma Estimote Inc. in seinen einzelnen Bestandteile untergliedert ist. Ein Beacon misst ungefähr $5,5 \text{ cm} \times 3,5 \text{ cm} \times 1,5 \text{ cm}$ in Länge, Breite und Höhe und ist dabei 50 g schwer. Zur Anbringung der Beacons an Wänden, Decken usw. dient die auf der Rückseite befindliche Silikonplatte. Diese haftet an nahezu jeder glatten Oberfläche und kann mithilfe von Wasser sehr einfach gereinigt werden. Somit können die Beacons im Grunde unzählige Male angebracht und wieder abgenommen werden. Die äußere Schutzhülle besteht dabei ebenfalls aus einem Silikon und schützt die inneren Bauteile. Zu den inneren Komponenten gehören die Platine mit dem darauf verlöteten Nordic nRF51822 Chip (32-Bit ARM Cortex M0 CPU mit 256 kB Flash-Speicher und dem 2,4 GHz BLE-Sendemodul) [27], einer Antenne und eine Knopfbatterie zur autarken Stromversorgung. Die Beacons haben eine variable Sendeleistung von 4dBm bis -30dBm (entspricht einer Leistung von 2,512 Milliwatt bis 1 Microwatt) und übertragen ihre Daten in Intervallen von 50 bis 0,5 Hertz. Die Kommunikation verläuft dabei bidirektional, d.h. vom Beacon zum Empfangsgerät und zurück. Während die Kommunikation von Beacon zu einem mobilen Gerät dazu dient, die Lokalisierung des Gerätes zu ermöglichen, dient die Kommunikation vom Smartphone, Tablet, etc. zum Beacon zu dessen Programmierung und zur Überprüfung des Betriebszustandes, wie z.B. dem Akku-Zustand.

Die bei der Signalübertragung verwendete Protokoll-Architektur Bluetooth Low Energy sendet dabei im 2,4 Ghz Band, welches ebenfalls von den Protokollen 802.11 ac/a/b/g/n, älteren Bluetooth-Standards, ZigBee, NFC, etc. genutzt wird. Eine kompakte Übersicht zu den einzelnen Varianten und deren Eigenschaften findet sich im

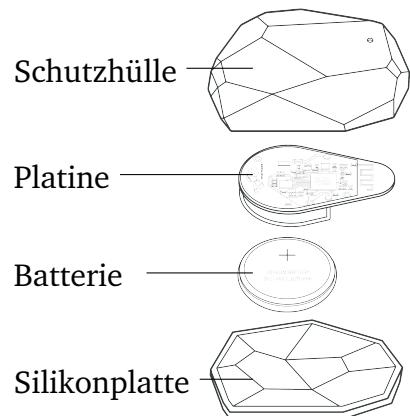


Abb. 2.2: Explosionszeichnung Beacon [14]

Anhang in Bild ???. Die Vorteile vom 2,4 Ghz Band gegenüber anderen Frequenzbändern ergeben sich aus einer großen Reichweite der Funksignale und einer geringen Größe der Antenne für deren Erzeugung. Jedoch besitzt dieses Band auch gewisse Nachteile, die bei der Auslegung von Beacon-Konfigurationen beachtet werden müssen. Die Verwendung des Protokolls im 2,4 Ghz Band ist dabei historisch bedingt und nahezu alternativlos, da sie zu den ISM-Bändern [19] zählt und nur diese somit frei nutzbar sind. Durch die Benutzung des 2,4 Ghz Bandes für die Datenübertragungen in WLAN oder Bluetooth treten bei vermehrter Nutzung des Bandes leicht Störungen in den Übertragungen auf. Somit hängt die Qualität der Signale und schlussendlich auch die Lokalisierungsgenauigkeit von der lokalen Auslastung des 2,4 Ghz Bandes ab. Des Weiteren können auch zwischen Sender und Empfänger befindlichen physischen Objekte einen störenden Faktor auf die Empfangsqualität ausüben. In den Bildern 2.3 und 2.4 ist eine mögliche Beeinträchtigung der Empfangsqualität von BLE-Signalen durch eine im Sichtfeld befindliche Wand oder eines menschlichen Körpers dargestellt. Denn durch die physikalischen Eigenschaften des 2,4 Ghz Bandes werden die Funksignale durch Materialien wie Wasser und Stahl besonders gut absorbiert. Da die meisten Gebäude aus Stahlbeton errichtet wurden und der menschliche Körper aus einem großen Anteil aus Wasser besteht, wirken sich diese Umstände besonders negativ auf die Qualität der Signale und am Ende auf die Anwendung der BLE-Technik für die Lokalisierung von Menschen in Gebäuden aus. Jedoch sei dies nur am Rande erwähnt und würde unter zusätzlicher Betrachtung dieser Aspekte den zeitlichen Rahmen dieser Arbeit in großem Ausmaß sprengen. Zusammenfassend werden hier noch einmal die Gründe für eine Qualitätsminderung der Lokalisierung festgehalten: die Schwächung der Signale an Objekten, die Phasenauslöschung durch andere Signalquellen und unberechenbaren Reflexionen.

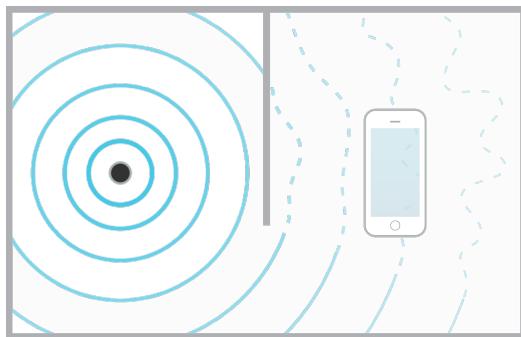


Abb. 2.3: Signalschwächung durch Objekte, z.B. Wände [6]

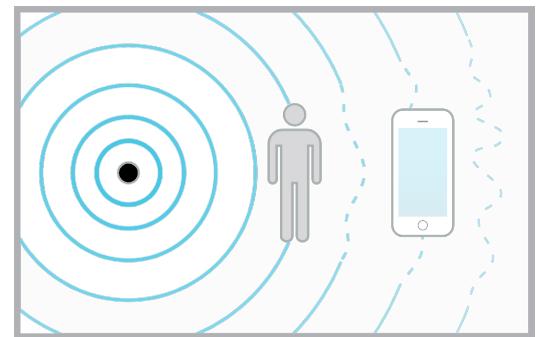


Abb. 2.4: menschliche Körper blockieren zusätzlich die Signale [6]

Die Daten, die bei der Kommunikation übertragen werden, sind dabei essentiell für die Positionsbestimmung von Objekten und bestehen aus einer:

- Identifikationsnummer (Länge von 16 Bytes)

- eingestellten Sendeleistung (2 Bytes)
- zusätzlichen Information, beschrieben als Major und Minor (jeweils 2 Bytes)

Mithilfe geeigneter Software lassen sich diese Parameter ändern und den Anforderungen entsprechend anpassen. Zusätzlich lässt sich noch die Intervalllänge der Sendefrequenz variieren. Im Falle der Estimote Beacons geschieht dies mit der kostenlosen Software „Estimote“ und ist für die Plattformen iOS und Android auf mobilen Geräten verfügbar. Mithilfe dieses Werkzeuges kann die Ortungsgenauigkeit signifikant erhöht bzw. verringert werden. Da die genauen Einflüsse dieser Parameter auf die Batterielebensdauer und die Ortungsgenauigkeit noch nicht erforscht wurden oder zumindest nicht öffentlich zugänglich sind, werden diese in Kapitel 4 untersucht und erklärt. Denn gerade der adaptiven Aspekt für die Indoor-Lokalisierung lässt diese Technologie so attraktiv erscheinen.

2.3 Ortungsmethoden

Mit der Identifikationsnummer kann zwischen den einzelnen Beacons unterschieden werden, wodurch erst die Möglichkeit einer Lokalisierung entsteht. Denn die Technik setzt voraus, dass die Identitäten der Beacons in einer Datenbank mit Positionsangabe in einem Gebäude hinterlegt sind. Beim Empfang eines gültigen Signals wird der Beacon in der Datenbank gesucht und anschließend die Distanz von Empfangsgerät und Beacon berechnet. Die Berechnung findet auf der Grundlage eines Ausbreitungsmodells von Signalen in Abhängigkeit zur eingestellten Sendeleistung des Beacons statt, in der die empfangene Signalstärke als RSSI-Wert interpretiert und dadurch eine Distanz geschätzt wird. Die Modelle der Hersteller sind typischerweise nicht frei zugänglich, weswegen sie auch hier nicht vorgestellt werden. Die als zusätzliche Informationen gekennzeichneten Daten sind eine Erweiterung der Identifikationsnummer und bieten lediglich einen gesteigerten Komfort für die Entwicklung der Datenbanken. Im nächsten Abschnitt wird dies in Tabelle 2.1 noch einmal veranschaulicht.

Geschäftsstandort		Berlin	Magdeburg	München
UUID		U8T7V56I-4689-10U9-7G63B4GAR21M		
Sendeleistung		-12dBm	-6dBm	1dBm
Major		1	2	3
Minor	Kleidung	10	10	10
	Elektronik	20	20	20
	Küche	30	30	30

Tabelle 2.1: Beispiel der Informationsnutzung; in Anlehnung an: [6]

Bei dem obigen Beispiel nutzt eine Warenhauskette mit mehreren Geschäftsstandorten die Beacon-Technologie zur Lokalisierung in ihren verschiedenen Standorten. Zur Ver-

einfachung besitzen die Beacons nur einen *Universally Unique Identifier* (UUID) und unterscheiden sich jeweils nur in ihren zusätzlichen Informationen und der Sendeleistung. Die gezeigten Informationen liegen auf den mobilen Geräten der Nutzer vor, genauso wie eine Applikation auf den Geräten, die diese Daten verarbeiten kann. Die Information Major steht dabei für den jeweiligen Standort und Minor für die Abteilung in der die Signale der Beacons empfangen werden können und der RSSI-Wert in einem definierten Bereich liegt. Abhängig vom Konzept der Ortung kann dieses Wissen unterschiedlich genutzt werden. Bei der Beacon-Technologie unterscheidet man daher zwei Anwendungskonzepte der Ortung.

2.3.1 Definitionen verschiedener Anwendungsfelder

Die einfachste Form einer Lokalisierung ist die Aufteilung eines Raumes in Bereiche, in der die Position eines Nutzers zu einem Beacon dadurch angegeben wird, ob er im Nah- oder Fernbereich zu ihm positioniert ist. Diese Art der Ortung ist dabei sehr ungenau, da die Distanzinformationen nicht explizit vorliegen, sondern nur ein homogener Distanz-Bereich um ein Beacon definiert ist. Die Distanz zu einem Beacon wird dabei durch die empfangene Signalstärke von einem Beacon geschätzt. Diese Einschätzung beruht derzeit noch auf den Erfahrungen des Beacon-Programmierers bzw. des Installateurs der Beacon-Systeme. Als Beispiel für eine solche Definition wurden einmal vier Zustände in der Tabelle 2.2 festgelegt und deren Bereiche in Abhängigkeit zu der empfangenen Sendeleistung eines Beacon eingegrenzt. In einem fiktiven Einsatzszenario könnte somit ein Kunde in einem Supermarkt oder Warenhaus gezielt auf ein Sonderangebot in seiner Nähe aufmerksam gemacht, oder zusätzliche Informationen zu einem Produkt in einer entsprechenden Applikation angezeigt werden.

Lagebeschreibung	Definition
Sehr nah	Dieser Bereich besitzt eine sehr hohe Wahrscheinlichkeit, dass der Nutzer direkt vor einem Beacon steht. Dies gilt für einen Bereich der in einer Distanz von unter 1 Meter zum Beacon liegt.
Nahbereich	Hier befindet sich der Bereich in einer Sichtlinie zu einem Beacon in einer Distanz von 1 bis 5 Metern. Da es aufgrund von Störungen, wie vorbeilaufender Menschen oder anderer Objekte zur Signalbeeinträchtigung kommen kann, könnte dieser Zustand nicht angezeigt werden, obwohl das Empfangsgerät in diesem Bereich liegt.
Fernbereich	Hier werden zwar die Signale von einem Beacon empfangen, jedoch kann aufgrund der Signalschwäche dem Empfänger kein eindeutiger Bereich zugeordnet werden. Dies impliziert jedoch keine große Entfernung zum Beacon, da wegen den genannten Störungen das Signal möglicherweise verfälscht wurde. Hier müssen weitere Verfahren und Strategien angewendet werden, um den Nutzer und sein Empfangsgerät genauer zu lokalisieren. Beispielsweise kann dem Nutzer empfohlen werden sein Empfangsgerät höher zu halten oder er sollte ein wenig umherlaufen.
Kein Empfang	Hier wurde ein Beacon nicht erkannt und somit liegt keine physische Nähe zum Beacon vor.

Tabelle 2.2: Beispiel der Bereichedefinition; in Anlehnung an: [6]

2.3.2 Mikro-Lokalisierung

Im Gegensatz zur relativen Lokalisierung mittels Lagebeschreibung, existiert noch die Methode der Mikro-Lokalisierung. Diese berechnet eine genaue Distanz zu einem Beacon und letztlich kann mithilfe von zusätzlichen Signalen mehrerer Beacons eine genaue Lokalisierung mit Koordinaten im Raum durchgeführt werden. Für die Bestimmung einer Position aus den empfangenen Beacon-Signalen existieren mehrere Verfahren. Ein gängige Methode ist die Tri-, bzw. Multilateration die aus der Entfernung eines Punktes zwischen drei oder mehreren Orientierungspunkten dessen Position in einem Koordinatensystem bestimmt. In diesem Fall wären

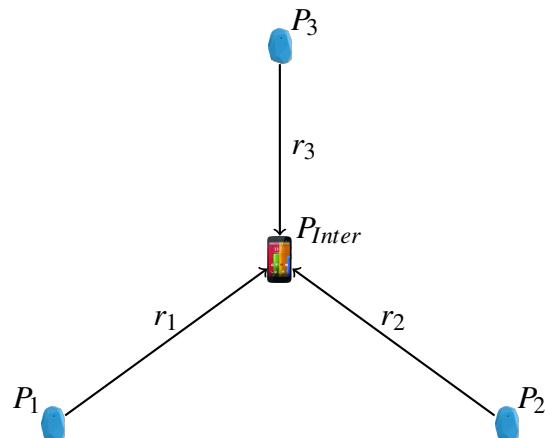


Abb. 2.5: Beispiel einer Trilateration zur Positionsbestimmung

es die Positionen der Beacons, die Orientierungspunkte und die Entferungen ließen sich aus den Signalen der Beacons berechnen. Mit einem Modell der Signalausbreitung wird dabei die Signalstärke, die durch den Weg vom Beacon zum Empfangsgerät geschwächt wurde, direkt in eine Distanz umgerechnet. Im übernächsten Kapitel wird ein solches Modell dafür beschrieben. An dieser Stelle soll noch einmal auf die Vorgehensweise zur Lokalisierung mittels Trilateration eingegangen werden. Dafür dient Abbildung 2.5 zur Veranschaulichung der Lösung. Im ersten Schritt werden dazu die Positionen der Beacons zur Vereinfachung in ein neues Koordinatensystem transformiert. Dabei wird ein Beacon in den Koordinatenursprung verlegt und ein zweiter durch Rotation um die Z-Achse auf die X-Achse gesetzt.

Translation:

$$\begin{aligned} T &= -P_1 \\ P'_1 &= P_1 + T = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \\ P'_2 &= P_2 + T \\ P'_3 &= P_3 + T \end{aligned}$$

Rotation:

$$\alpha = -\arcsin \left(\frac{y_2}{\sqrt{x_2^2 + y_2^2}} \right)$$

$$R_z(\alpha) = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix}$$

$$\begin{aligned} P''_1 &= \begin{pmatrix} 0 \\ 0 \end{pmatrix} \\ P''_2 &= R_z(\alpha) \cdot P'_2 \\ P''_3 &= R_z(\alpha) \cdot P'_3 \end{aligned}$$

Anhand zweier Formeln lassen sich dann die Koordinaten des Empfangsgeräts aus den einzelnen Distanzen berechnen und schließlich ins ursprüngliche Koordinatensystem zurück transformieren [29]. Die Distanzen sind dabei gegeben als r_1 , r_2 und r_3 und die

Koordinaten entsprechen der Nummerierung der Punkte nach der Transformation.

$$x_{Inter} = \frac{r_1^2 - r_2^2 + x_2^2}{2x_2^2}$$

$$y_{Inter} = \frac{r_1^2 - r_3^2 + x_3^2 + y_3^2}{2y_3^2} - \frac{x_3}{y_3} \cdot x_{Inter}$$

Rücktransformation:

$$P_{Inter} = R_z(-\alpha) \cdot \begin{pmatrix} x_{Inter} \\ y_{Inter} \end{pmatrix} - T$$

Ein großer Nachteil von dieser Methode ist die große Fehleranfälligkeit der Signalübertragung. In der Anwendung in Gebäuden werden die Signale oftmals von Wänden reflektiert oder gedämpft, während die hohe Dichte von Signalquellen weitere Interferenzen verursacht. Um diese Nachteile auszugleichen wird statt der Trilateration eine Multilateration angewendet. Durch die Nutzung mehrerer Beacons können Messrauschen und Störungen zum Teil ausgeglichen und die Genauigkeit der Lokalisierung somit erhöht werden. Jedoch kann in dieser Arbeit nicht auf dieses Verfahren eingegangen werden, da lediglich drei Beacons zur Verfügung standen.

2.4 Anwendungsbereiche der Indoor-Lokalisierung

Der Bedarf an einer innerräumlichen Ortung ist normalerweise dort vorhanden, wo es für den Besucher schwer ist sich zu orientieren. Dies können große Gebäudekomplexe sein oder verwinkelte und unüberschaubare Gänge. Infolge der Digitalisierung der Gesellschaft, oft bezeichnet als „Digitale Revolution“ oder auch unter anderem Kontext als „Zweite Moderne“ [15], entstehen durch den großen Grad der Verbreitung von mobilen „smartten“ Geräten riesige Netzwerke mit einer schier unendlichen Datenflut. Dies eröffnet folglich eine Möglichkeit zur Kommunikation zwischen Mensch und Gebäude, in der die Beacons als „Sinne“ des Gebäudes verstanden werden können.

2.4.1 Mögliche Einsatzszenarien

Viele Unternehmen nutzen hier die Möglichkeiten einer vernetzten Welt, um besser das Verhalten der Kundschaft zu verstehen. Um wieder auf das Beispiel von der Warenhauskette zurück zu kommen, wäre es für das Unternehmen von Vorteil, zu wissen, wie viele Kunden am Tag eine Filiale besuchen,



Abb. 2.6: Beacon Nutzung in Geschäften [1]

wie ihr Bewegungsprofil aussieht und was sie am Ende kaufen. Nützlich wären die Informationen für die Preisgestaltung, den Aufbau der Abteilungen im Warenhaus und dies könnte auch zu einer besseren „Just-In-Time“ Lieferkette führen und somit Lagerkapazitäten einsparen. Natürlich wäre dies auch mit herkömmlichen Methoden möglich, indem Umfragen stattfinden und die Mitarbeiter eines Geschäftes die Kunden genau beobachten würden. Dies wäre aufgrund hoher Personalkosten nicht nur unrentabel, sondern würde den Kunden außerdem ein Gefühl der Überwachung vermitteln. Mit der Beacon-Technologie wäre dies kostengünstig und voll automatisiert möglich. Damit potentielle Kunden auch die dafür nötige Software für ihr mobiles Endgerät installieren und an der Auswertung ihrer Daten zustimmen, können die Kunden mit einer entsprechenden Applikation an exklusiven Gewinnspielen, Rabattaktionen oder Punktesystemen teilnehmen, so wie es heute schon durch einige Unternehmen angeboten wird (z.B. DeutschlandCard¹, Payback², etc.). Die gesellschaftliche Akzeptanz wäre sicherlich gegeben, da es schon mit den modernen Stauwarnsystemen einen Vergleichfalls im Outdoor-Bereich gibt. Da die Position eines Fahrzeugs entweder durch das eingebaute Navigationssystem bekannt oder durch die Mobilfunkgeräte der Fahrzeuginsassen ermittelt werden kann, werden schon heute diese Systeme zur Erschaffung von Bewegungsprofilen und damit zur Stauvorhersage genutzt [18]. Für ein solches Szenario im Indoor-Bereich würde hauptsächlich die kontextbezogene Lokalisierung in Betracht gezogen werden, da hier die meisten Geschäfte überschaubar sind und schon gute Strukturen zur Orientierung der Kundschaft genutzt werden. Extreme Ausnahmen wie die „Golden Resources Mall“ in Peking mit 557,419 m² Ladenfläche, wo eine genauere Lokalisierung zur Navigation sicherlich sinnvoll wäre, bilden hier eher die Ausnahme.

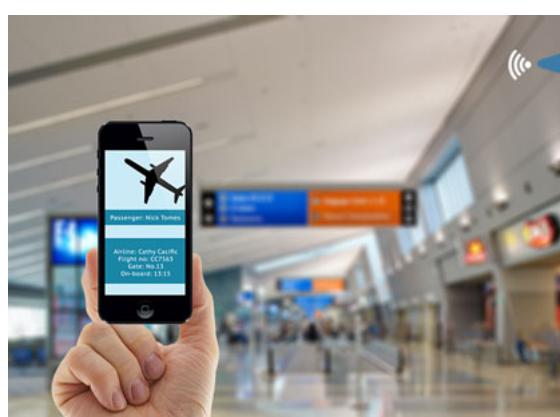


Abb. 2.7: Beacon Nutzung im Flughafen [35]

Im Gegensatz dazu stellt die Mikro-Lokalisierung ein weiteres mögliches Szenario dar. Sie soll dazu dienen Menschen oder Objekte genau zu lokalisieren, um unter Verwendung von speziellen Applikationen eine bessere Orientierung zu gewährleisten. Daran interessiert sind meist Branchen, die ihre Kundschaft gerne schnell und ohne Umwege zu dem führen möchten, weswegen sie in die Einrichtung des Unternehmens gekommen sind. Als Beispiele können hier Messehallen- und Flughafenbetreiber genannt wer-

¹www.deutschlandcard.de

²www.payback.de

den. In diesen Branchen sind die Gebäude meistens sehr unübersichtlich und die Kundschaft ist häufig ortsfremd. Der Vorteil den die Unternehmen aus dem System ziehen können, ist eine bessere Verteilung der Kunden durch eine intelligente Pfadplanung. So können die Platzkapazitäten der Einrichtungen besser genutzt und die Gäste somit schneller abgefertigt werden. Wenn zudem ein Unternehmen in seinen Gebäuden eine ähnliches System wie ein Navigationssystem für die Straße bieten kann, bedeutet dies auch einen Wettbewerbsvorteil und fördert die Kundenzufriedenheit.

Grundsätzlich sind dies hier nur Annahmen für mögliche Szenarien. Da es noch keine praktikablen Anwendungen der Indoor-Lokalisierung gibt, können keine genauen Vorhersagen diesbezüglich getroffen werden. Viele Unternehmen schrecken noch davor ab, weil kein einheitlicher Standard existiert und nur wenige Menschen mit der Technik vertraut sind.

2.4.2 Planungskonzepte für Beacon-Konfigurationen

Bisherige Ansätze um ein Lokalisierungssystem in einem Raum zu realisieren beruhen auf fachmännischen Einschätzungen des Beacon-Installateurs. Das bedeutet, dass sich das Konzept lediglich auf Erfahrungswerte beruft. Die Beacons werden dabei an strategische Punkte gesetzt und manuell an jeder Postion neu konfiguriert. Wie schon oben beschrieben, werden die Befestigungen der Beacons im Raum vermessen oder grob geschätzt und so zur Positionsbestimmung in eine Lokalisierungs-Applikation ebenfalls wieder manuell eingepflegt. Von der Firma Estimote gibt es seit Neuestem eine neue Applikation namens „Estimote Indoor Location“, die den Entwicklern einer Lokalisierungs-Applikation unterstützen soll. Der Entwickler installiert dazu an jeder der vier Wände eines Raumes ein Beacon der Firma und ausgehend vom Eingang startet der Entwickler die App und läuft anschließend den ganzen Raum ab. Dabei werden die Signale der Beacons an den Wänden aufgezeichnet und mithilfe der inertialen Sensoren des Smartphones zusätzlich die Bewegungsinformationen gespeichert. Daraus errechnet die App eine Karte vom Raum und liefert auch gleich einen Positionsbestimmung mithilfe der genannten Sensordaten in Echtzeit. Diese Karte kann auch in Eigenentwicklungen für das Smartphone verwendet werden, jedoch ist dafür eine Migration der Daten notwendig. Da für diese Arbeit nur drei Beacons zur Verfügung standen und diese App erst in der späten Phase der Arbeit veröffentlicht wurde, wird hier nicht mehr auf dieses Konzept eingegangen werden. Jedoch lassen die noch engen Begrenzungen der Anwendung (konstante Anzahl von vier Beacons, Einstellung der Parameter wieder manuell nach Erfahrungswert) keinen Handlungsspielraum für die Optimierung der Beacon-Konfigurationen zu. Zudem muss immer ein Entwickler die Räume ablaufen und später die Daten vom Smartphone in die eigene Entwicklung migrieren. Für kleine Räume ist diese Strategie sicherlich hilfreich, jedoch wird sie bei Großprojekten, wie z.B. die Schaffung eines Ortungssystems in einem Flughafen schlicht unbrauchbar.



Abb. 2.8: Demo der Estimote Indoor Localization App [12]

3 Konzeptplanung

Nach der Betrachtung des Standes der Technik fällt auf, dass sowohl die Planung, als auch die Inbetriebnahme von den innerräumlichen Ortungssystemen stets ein experimentierlastiger Prozess ist. Es existieren weder bekannte Theorien, noch computergestützte Hilfsmittel für deren Auslegung und Validierung. Zum einen wird es daran liegen, dass es dafür noch keine Notwendigkeit gab und zum anderen, weil es aufgrund der Vielseitigkeit der Anwendungsgebiete und Situationen noch kein gemeinsamer Standard gefunden wurde. Dabei ist das Problem der Erstellung einer guten Indoor-Lokalisierung nicht trivial und hoch Komplex, wenn all die Faktoren hinzugerechnet werden, die das System beeinträchtigen können. Zudem verliert sich der Überblick über alle Signalquellen und deren Position in großen Gebäuden, weswegen auch noch keine Großprojekte diesbezüglich entstehen. Im folgenden soll in diesem Abschnitt ein Konzept entwickelt werden, welche die Theorie mit ergänzender Simulationstechnik im Zusammenspiel mit Experimenten unterstützt und somit ein bekanntes Schema in Wissenschaft und Industrie aufgreift: der Prozessplanung.

3.1 Vorüberlegungen

Der erste Schritt für eine systematische Prozessplanung wäre die Schaffung einer Theorie bzw. eines Modells für die Signalausbreitung der BLE-Signale. Dieses muss erst durch Betrachtung der physikalischen Eigenschaften und mathematischen Beziehungen geschaffen oder anhand eines existierenden Modells aus vergleichbaren Technologien abgeleitet werden. Nachdem ein Modell aufgestellt wurde, muss dieses parametrisiert werden, um das Modell auf das reale Verhalten besser abbilden zu können. Mithilfe des Modells lässt sich anschließend eine Simulation erstellen, die die Signalausbreitung von BLE-Signalen in einem Raum prädiktionieren und daraus die Beacons unter Erfüllung von Optimalitätsbedingungen anschließend automatisch verteilen kann. Die Anordnung der Beacons kann so je nach Einsatzszenario angepasst und vor der eigentlichen Installation simuliert und getestet werden. Jedoch muss die fertige Konfiguration erneut durch Messungen auf die gewünschten Funktionen und Qualität der Indoor-Lokalisierung hin geprüft werden. Dieser Kreislauf aus Experimenten, Modellierungen und Simulationen wird dazu in Abbildung 3.1 grafisch veranschaulicht. Der Vorteil dieser Herangehensweise ist die fundierte Lösung eines Problems, sodass auf die gewonnenen Informationen aufgebaut werden kann und sich mit ihnen wissenschaftlich arbeiten lässt. Für die Planung eines Indoor-Lokalisierungsproblems mittels Beacons bedeutet das, dass die Konfigurationen vorab geplant werden können

und diese einheitlich aufgebaut sind. Dadurch ergeben sich auch kommerzielle Vorteile durch eine schnellere Entwicklung von Ortungs- und Navigations-Applikationen und weiterer Softwareprojekte. Zudem wird die Leistungsfähigkeit der Beacon-Technologie messbar, was für eine bessere Definition ihrer Einsatzmöglichkeiten führen wird.

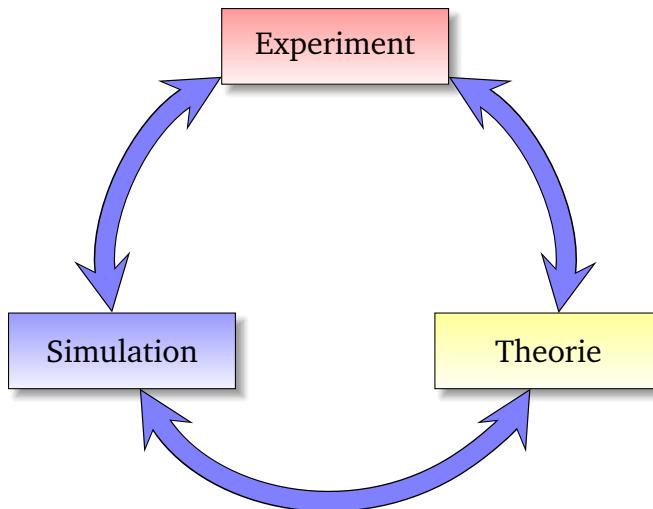


Abb. 3.1: Kreislauf einer Prozessplanung

3.2 Zielsetzungen

Aus den Vorüberlegungen ergibt sich eine klar definierte Struktur, die nun konkret mit einzelnen Abläufen in diesem Absatz beschrieben wird. Die Zielsetzungen sind dabei zum einen, ein Modell für die Ausbreitung von BLE-Signalen zu finden und zu parametrisieren. Dafür braucht es ein systematisches Vorgehen für die Aufnahme der Messwerte und eine Plattform als Empfänger der BLE-Signale. Nachfolgend muss das Modell in eine Simulationsumgebung eingepflegt und ein Optimierungsalgorithmus für die Positionierung der Beacons implementiert werden. Im letzten Schritt soll als Abschluss der Arbeit ein Testfeld mit einer simulierten Konfiguration aufgebaut und die Messungen daraus mit den Simulationsergebnissen verglichen werden. Dafür wird ein mobiles System benötigt, welches vorgegebene Punkte im Feld des Lokalisierungssystems ansteuern und seine Position mit der errechneten Position aus der Trilateration vergleichen kann. Um alle diese Punkte zu erfüllen, müssen Systeme und Werkzeuge gefunden werden, die die Konzeptplanung unterstützen. Im Anschluss werden diese näher benannt und deren Zusammenspiel erläutert.

3.2.1 Hardware-Anforderungen

An das Empfangsgerät wird lediglich die Anforderung gestellt, dass die Messung eines Beacon-Signals mit der Messung einer Position synchronisierbar ist und somit

über Netzwerkfähigkeit oder Speicher verfügen muss. Dies ist gerade in der Validierungsphase von hoher Wichtigkeit, um standardisierte und reproduzierbare Ergebnisse zu gewährleisten. Da die eigentliche Nutzung des Positionierungssystems mithilfe von Smartphones stattfindet, ist die Verwendung dieser Endgeräte für die Messungen durchaus zu empfehlen. Denn Smartphones besitzen eine große Bandbreite an Schnittstellen und einen internen Speicher, sodass jeweils eine Strategie für die Messwertaufnahme verfolgt werden kann. Zudem existieren von den Herstellern der Beacons bereit fertige Bibliotheken und Entwicklertools, um die Signale von ihren Beacons zu verarbeiten, was einen geringeren Aufwand für die Implementierung bedeutet. Zudem weisen Antennen der Empfangsgeräte Charakteristiken auf, die die Aufnahme der Messungen je nach Ausrichtung des Smartphones beeinflussen. Es gilt somit auch diese Einflüsse zu untersuchen und da die meisten Smartphones auch über Inertialsensoren verfügen, bieten sie gleich zusätzliche Sensordaten in einem Gerät. Die Problematik der Antennencharakteristiken findet sich dabei im nächsten Kapitel. Für die standardisierte Aufnahme der Messwerte bietet sich eine Kombination aus Maschine bzw. Roboter und Smartphone an. Da die Roboter-Systeme mobil sind, um eine Positionsbestimmung unabhängig von der Beacon-Ortung erweitert werden können und ebenfalls Kommunikationsfähigkeiten durch entsprechende Middleware besitzen, bietet sich ihre Verwendung für die Überprüfung von Beacon-Konfigurationen an. Es ist dabei nur darauf zu achten, dass es eine entsprechende Halterung für das Smartphone am Roboter existiert. Darüber hinaus kann der gesamte Prozess durch die Verwendung von Robotern in den Experimenten automatisiert werden, welches ebenfalls ein Ziel dieser Arbeit darstellt.

3.2.2 Konzept der Software-Architektur

Im Mittelpunkt der Software-Architektur steht die Datenübertragung der Messwerte aus verteilten Systemen auf ein zentrales System, um diese später im Verarbeitungsschritt zu synchronisieren. Um dies zu gewährleisten müssen alle Plattformen miteinander kommunizieren können und die Art der Übertragung darf dabei nicht zu starken Verzögerungen der Messwertaufnahme führen. Sprich es wird ein effektives und schnelles Framework benötigt, dass die eigentliche Messungen nicht beeinträchtigt. Es muss dabei zwischen den Experimenten unterschieden werden, denn für die Parameterbestimmung des Modells wird keine ständige Positionsmessung benötigt, da diese „per Hand“ ermittelt werden kann und sich im Laufe einer Messreihe für eine Distanz auch nicht ändert. Hingegen bei der Überprüfung eines bestehenden Lokalisierungssystems fließt die „externe“ Lokalisierung eines Roboters in den Vektor aus Messwerten mit ein, sodass über das Framework der Kommunikation verschiedene Datentypen von den verschiedenen Plattformen auf die zentrale Verarbeitungsstelle geleitet werden müssen. Die Anforderungen an die einzelnen Anwendungen und unteren Ebenen der Software wird in Abbildung 3.2 einmal dargestellt. Die Anforderungen an die einzelnen Anwendungen sind bei dem Smartphone die Erfassung der einzelnen BLE-Signale von den Beacons und der Messung der eigenen Ausrichtung.

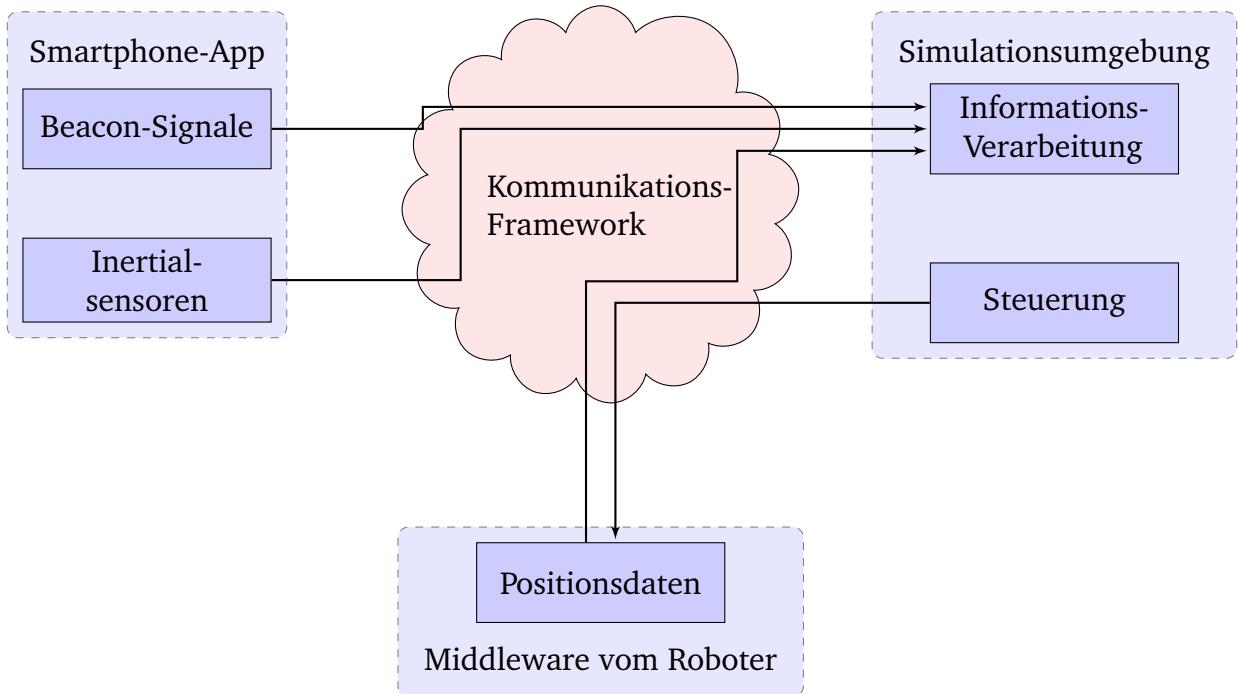


Abb. 3.2: Überblick auf die Software-Anforderungen

Im Gegensatz dazu muss der Roboter seine eigenen Positionsdaten übermitteln und zudem die auf einem externen Computer in der Simulation erstellten Raumelemente ansteuern und somit Befehle vom PC entgegennehmen. Die Simulationsumgebung speichert alle Sensordaten der verteilten Systeme und verarbeitet diese für die eigentliche Validierung der Beacon-Konfiguration.

Da die Simulation zur Erstellung eines Indoor-Lokalisierungssystems in Gebäude bzw. Räumen angewendet wird, muss die Simulationsumgebung zuerst den Grundriss eines Gebäudes analysieren und die gewonnenen Daten für die spätere Verwendung verarbeiten. Es bietet sich dabei an den Raum in einzelne Elemente zu unterteilen, sodass die Simulation vereinfacht wird und mit diskreten Werten gerechnet werden kann. Die Informationen über die Anordnung von Objekten und Wänden ist dabei essentiell für die manuelle oder automatische Verteilung der Beacons im Raum und der Simulation der einzelnen Signalstärken in Abhängigkeit zu den Distanzen der Raumelementen zu den Beacons. Der grundlegende Ablauf ist in Abbildung 3.3 hierfür skizziert. Die Optimierung einer Anordnung von Beacons ist dabei stets subjektiv, da für jedes Szenario andere Anforderungen an das Ortungssystem gestellt werden. Es gilt somit ein Optimierungsalgorithmus zu finden, der mit unterschiedlichen Ansprüchen je nach Situation umgehen und gegebenenfalls auf ein spezielles Optimierungsproblem angepasst werden kann. Auf die Optimierung wird speziell in Kapitel 5 eingegangen.

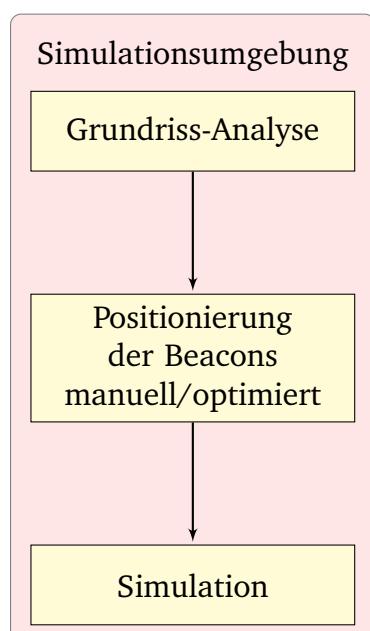


Abb. 3.3: Überblick auf die Software-Anforderungen

4 Software-Entwicklung und Experimente

Aus den anfänglichen Ideen für die Umsetzung des Lighthouse Keeper entstehen in diesem Kapitel die ersten Module für dessen Verwirklichung. Zum Anfang wird die verwendete Hardware vorgestellt und in Verbindung dazu, die Aufgaben der genutzten Software-Komponenten näher erläutert. Durch die geeignete Wahl aller Bestandteile, lassen sich viele Aufgaben auf vorgefertigten Module übertragen. Beispielsweise werden statt einer eigenen Entwicklung, ein bestehendes Kommunikations-Framework und eine bereits existierende Middleware für den Roboter genutzt. Mit der Nutzung dieser fertigen Lösungen wird im großen Maße Arbeitsaufwand eingespart und ermöglicht einen stärkeren Fokus auf das eigentliche Thema. Da dies im Hinblick auf die Smartphone-Applikation nicht immer möglich war, wird deren Implementierung in diesem Kapitel exemplarisch einmal dargestellt.

Die durchgeführten Experimente beziehen sich in diesem Teil der Arbeit auf die Messung der Signalstärke der Beacons in Abhängigkeit zu deren Entfernung. Diese Phase in der Prozessplanung legt den Grundstein für die Modellbildung, welche anschließend im nächsten Kapitel vorgenommen wird. Ferner werden einzelne Einflüsse auf die Messungen näher betrachtet und eine Analyse zu der Batterielaufzeit eines Beacons in Abhängigkeit zu seinen Einstellungen aufgestellt.

4.1 Verwendete Hardware

Da die Beacon-Technologie vorrangig zur Indoor-Lokalisierung von Personen eingesetzt wird und als Peripheriegerät meistens ein Smartphone Verwendung findet, wird auch ein solches für die gesamte Testdauer als Messgerät genutzt. Die Wahl fiel dabei auf ein Android-Smartphone mit dem Namen Motorola Moto G der gleichnamigen Firma Motorola Inc. Es wurde ausgewählt, weil es alle Hardware- und Software-Anforderungen zum Empfang von BLE-Signalen erfüllt und als ein Standard-Smartphone gilt, sodass sich mit den ihm erzielten Ergebnisse auch auf andere Produkte übertragen lässt. Des Weiteren werden zwei Roboter in den Experimenten genutzt, um die Messungen reproduzierbar und standardisiert durchzuführen. Für die reinen Distanz-Signalstärke-Messungen wird der Roboters bzw. lediglich sein Arm namens „Youbot“ der Kuka AG und später für die Validierung einer Beacon-Konfiguration der Roboter „Scitos G5“ der MetraLabs GmbH verwendet.

4.1.1 Motorola Moto G

Das Moto G dient als Empfangsstation der BLE-Signale, dessen grundlegende Spezifikationen ein 1,2 GHz Snapdragon 400 Prozessor mit 1 GB RAM und ein WCN3620 BT/FM/WLAN RF Modul aussmachen ([7]). Seine Abmaße betragen 129.9 mm × 65.9 mm × 11.6 mm bei einem Gewicht von 143 g. Desweiteren verwendet es standardmäßig ein Android 4.3 als Betriebssystem, welches jedoch für die Experimente auf die Version 4.4 geupdated wurde. Neben der technischen Ausstattung und Software sind für die späteren Messungen die Antennen und deren Charakteristiken von hoher Bedeutung, denn deren Eigenschaften wirken sich direkt auf den Empfang der Signale aus. Um die Einflüsse besser zu verstehen, sind in den Abbildungen 4.1 und 4.2 die Anordnung der Antennen einmal skizziert. Dabei fällt es auf, dass sich WLAN- und Bluetooth-Modul die selben Antennen teilen. Dies führt zu der Frage, ob es zu Konflikten in der Funktionsweise des Smartphones kommt, wenn gleichzeitig auf beide Module zugegriffen wird. Jedoch dazu mehr im Abschnitt der App-Entwicklung. Die zweite Frage die sich daraus stellt, ist die Veränderung der Empfangs- und Sendequalität des Moto G in verschiedenen Positionen. Eine Antenne hat je nach Bauform und Funktionsweise Bereiche, in der sie mit voller Leistung sendet und empfängt, aber auch Bereiche in der Funk-Signale sie weder verlassen noch erreichen können. Das hat verschiedene physikalische Gründe, jedoch sind diesen komplexen nichtlinearen Eigenschaften der Antennen des Moto G zumindest nicht öffentlich bekannt und können auch nicht einfach bestimmt werden. Es sei nur anzumerken, dass bei den Messungen auch darauf geachtet werden muss, wie und an welchen Halte-Punkten das Moto G am besten befestigt wird, ohne deren Transceiver-Fähigkeiten negativ zu beeinflussen. Eine gute Annahme dabei ist es das Smartphone so auszurichten, als ob es flach in der Hand eines Menschen liegen würde. Der Hersteller wird schließlich darauf bedacht sein, sein Produkt für einen normalen Betrieb auszulegen und somit auch die



Abb. 4.1: Vorderseite des Motorola Moto G [5]

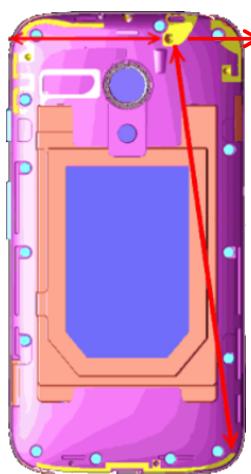


Abb. 4.2: Rückseite vom Moto G mit Antennen-Gerüst [7]

Konstruktion und Bau der Antennen danach optimieren. Diese Annahme muss jedoch noch anhand von Messungen verifiziert werden.

4.1.2 Youbot

In vorigen Abschnitt wurde schon angesprochen, dass die Lageposition des Messinstruments zu seiner Empfangsleistung überprüft werden muss. Zudem soll das Smartphone so gehalten werden, als wenn es sich in einer flachen Hand befindet und so auch die Experimente durchgeführt werden. Um all dies zu erreichen und auch unter der Anforderung an einen automatisierten und reproduzierbaren Prozess, empfiehlt es sich einen Roboterarm als Mess-Plattform zu benutzen. Aufgrund der Verfügbarkeit wurde das Modell „Youbot“ der Firma „Kuka“ gewählt und für die Messungen mit einer Halterung aus einem 3D-Drucker ergänzt (siehe Abbildung 4.3). Die Vorteile des Systems sind zum einen der montierte hochpräzise Roboterarm mit Greifer auf dem Youbot und zum anderen, dass ein vollwertiger Rechner mit einem Linux Betriebssystem und eine drahtlose WLAN-Schnittstelle im System verbaut sind und so die Kommunikation zum Roboter sehr einfach aufgebaut werden kann. Der künstliche Arm kann sich dabei um seine fünf Achsen drehen (siehe Abbildung 4.4) und bietet somit genug Möglichkeiten, die Lageposition vom Moto G zu verändern.



Abb. 4.3: Youbot mit Halterung (gelber Aufsatz)

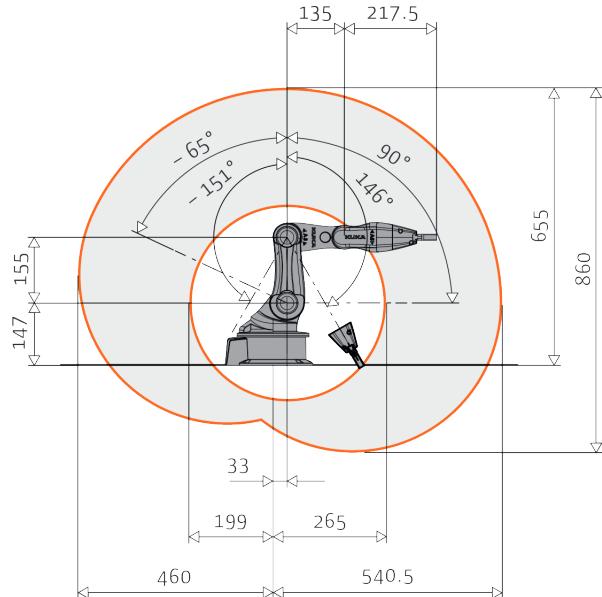


Abb. 4.4: Zeichnung eines Youbot-Arms und seiner fünf Rotationsachsen [21]

4.1.3 Scitos G5

Die Durchführung der Evaluation einer Beacon-Konfiguration wäre grundsätzlich auch mit dem modifizierten Youbot aus Abschnitt 4.1.2 möglich. Jedoch wurde es in der

Aufgabenstellung gefordert, den Roboter „Scitos G5“ von der MetraLabs GmbH zu verwenden. Der Scitos G5 ist mit den Maßen $55\text{ cm} \times 60\text{ cm} \times 60\text{ cm}$ etwas größer als der Youbot und durch seinen Dreirad-Lenkung weniger wendig (vgl. Abbildung 4.5). Die Vorteile gegenüber dem Youbot bestechen jedoch durch seine bessere Ausstattung und den größeren Software-Umfang, weswegen es keiner zusätzlichen Entwicklungen bedarf und somit den Arbeitsaufwand für das Lighthouse Keeper-Konzept verringert wird. Der Scitos G5 verfügt über Schrittmotoren, einem Laserscanner und Inertialsensoren zur Navigation und er besitzt zudem einen leistungsfähigen Intel Core i7-Prozessor, WLAN und ein Linux-Betriebssystem, sodass auch hier genug Ressourcen für eine Kommunikation vorhanden sind. Die Machbarkeit des gesamten Kontroll-Prozesses von Beacon-Konfigurationen beruht dabei auf der Leistungsfähigkeit von diesem Roboter, denn er muss in der Lage sein, seine Position unabhängig von der Triangulation von Beacon-Signalen zu bestimmen und diese als Referenzquelle zur Verfügung zu stellen. Für seine Positionsbestimmung verwendet er die Software-Umgebung „Miracenter“ (siehe 4.2.2 im nächsten Absatz), welche die Sensordaten aus der Wahrnehmung (Motoren, IMU, Laserscanner) so verwertet, dass sie eine Karte der Umgebung erstellen kann und somit im Abgleich von Sensorinformationen und Karte der Roboter stets seine Position kennt. Der Hersteller wirbt dabei mit einer Nutzlast von bis zu 50 kg und einer Maximalgeschwindigkeit von $1,4\frac{\text{m}}{\text{s}}$ bei einer batteriebetriebenen Laufzeit von ca. 20 Stunden [23], sodass ein künstlicher Roboterarm auf dem Scitos G5 ebenfalls denkbar wäre und somit auch die Aufgaben vom Youbot zukünftig übernimmt.



Abb. 4.5: Scitos G5 von der MetraLabs GmbH

4.2 Verwendete Software

Um die Hardware zu nutzen und das geplante Konzept umzusetzen, benötigt es einer Kommunikation zwischen den Geräten und weiterer Werkzeuge zur Aufnahme von Messungen und deren Verarbeitung. Bei der Umsetzung wurde besonders auf Konfirmität der verschiedenen Systeme und deren reibungslosen Zusammenspiels geachtet.

Um eine gemeinsame Basis zu schaffen, wurde das Software-Framework „Robots Operating System“(ROS) verwendet. Mit einem gemeinsamen Standard lassen sich die Messungen besser vergleichen, wodurch ihre Qualität und Aussagekraft zunimmt. Die Messungen müssen dabei auf der Smartphone-Plattform und den Roboter-Plattformen aufgenommen und diese synchronisiert werden. Während ROS die übergeordnete Schnittstelle darstellt, müssen auf den einzelnen Hardware-Elementen die Messungen eigenständig durchgeführt werden. Die Messung auf dem Scitos-Roboter entfällt dabei auf die Software „Miracenter“ und funktioniert ohne weiteres. Die Software für die Messungen auf dem Smartphone ist hingegen nicht vorgefertigt und muss mithilfe eines Editors für Android-Applikationen und einer speziellen Bibliothek für die Kommunikation Smartphone ↔ Estimote Beacon entwickelt werden.

4.2.1 Robot Operating System – ROS

Das „Robot Operating System“ oder kurz ROS, ist ein Projekt zur Schaffung eines flexiblen Frameworks für die Entwicklung von Software für Roboter. Daraus entstand eine der mächtigsten Sammlungen aus Werkzeugen, Bibliotheken und Normen, um komplexes Verhalten zwischen Robotern über verschiedenste Robotik-Plattformen robust zu gestalten [31]. Der Anwendungsbereich für ROS ist demzufolge sehr umfangreich, jedoch werden für diese Arbeit nur folgende Eigenschaften [22] des Projektes benötigt und hier näher betrachtet:

- „Peer to Peer“ (P2P)-Verbindungen
- Modularer Aufbau
- Unterstützung mehrerer Programmiersprachen
- freier Nutzung und Open-Source

Das Kommunikation-Framework des Lighthouse Keeper-Konzeptes, welches auf ROS aufbaut, besteht dabei aus mehreren miteinander verbundenen Rechnern (sog. „Hosts“) die zur Laufzeit über P2P miteinander kommunizieren. Bei der P2P-Verbindung können alle Teilnehmer ihre Dienste bzw. ihre Informationen gleichermaßen einander anbieten und nutzen, indem sie Daten im Netzwerk gleichzeitig empfangen und senden können. Dabei läuft auf einem zentralen Server der eigentliche Kern des Frameworks, über den der sämtliche Datenverkehr geleitet wird. Und auf den Host-Systemen laufen die eigentlichen modularen Anwendungen („Nodes“), die über Schnittstellen („Sockets“) des Betriebssystems ihre Daten auf das Netzwerk und schließlich an ROS verteilen. Dadurch können die Nodes in verschiedenen Sprachen programmiert werden, die lediglich auf die entsprechende Schnittstelle zugreifen. Ausgehend von den Nodes werden den gesendeten Nachrichten gesonderte Bezeichnungen („Topics“) zugeordnet und mit einem Datentyp versehen. Diese Informationen sind allen Teilnehmer des P2P-Netzwerkes bekannt und können auch von ihnen angefordert werden. Der Kern organisiert dabei eine einheitliche Uhrzeit, die dadurch für alle Nachrichten bzw. deren Zeitstempel in den verteilten Systemen konsistent bleibt

und so eine Synchronisierung der Messgeräte nicht mehr nötig wird. Somit ermöglicht die Verwendung von ROS den Aufbau der gesamten Kommunikation, wie es in der Konzept-Planung in 3.2.2 gefordert wurde. Zudem erleichtert ROS durch sein großen Funktionsumfang nachfolgende Erweiterungen und bietet somit Freiheiten für zukünftige Aufgaben.

4.2.2 Miracenter

Das Paket „CogniDrive“, als Bestandteil der Software „Miracenter“ von der Firma MetraLabs GmbH, dient als Navigator des Scitos G5 und ermöglicht es mit ihm den Grundriss eines Raumes zu erstellen (siehe Abbildung 4.6) und zusätzlich die Lokalisierung des Roboters in diesem durchzuführen. Während beispielsweise der Roboter entlang der Wände fährt, messen seine Schrittmotoren den zurückgelegten Weg, die Inertialsenso- ren verfeinern die Informationen und erweitern sie um die Ausrichtung des Roboters. Dabei misst zusätzlich der Laserscanner die Distanz zu den Wänden und allen anderen festen Objekten in Reichweite. Aus der Fusion aller Messwerte lässt sich so der Grundriss des vermessenden Raumes generieren. Im nebenstehendem Bild ist eine solche Karte als Beispiel aufgeführt. Die Daten die Miracenter dem Nutzer zur Verfügung stellt, bestehen dabei aus der Position des Roboters und seiner Ausrichtung in der Karte. Die Karte liegt dabei als „Portable Network Graphics“ (PNG)-Datei vor, wobei ein Pixel einer konstanten metrischen Länge entspricht, deren Verhältnis in einer „Extensible Markup Language“ (XML)-Datei definiert ist. Die Farbwerte der Bildpunkte beschreiben zudem, ob an einer Stelle ein Hinderniss oder ein frei befahrbarer Raum vorliegt. In der Abbildung ist zu erkennen, dass die erstellte Karte teilweise verrauscht ist, Wände nicht gerade verlaufen, oder offene Türen und herumlaufende Personen die Messungen verfälschen. Die Karte kann dafür nach der Erstellung von jedem beliebigen Bildbearbeitungsprogramm geöffnet und bearbeitet werden. Dabei ist darauf zu achten die Auflösung nicht zu verändern, da ansonsten die Dimensionen nicht mehr übereinstimmen oder gegebenenfalls die XML-Datei angepasst werden. Nach der Bearbeitung der Bilddatei kann sie anschließend im Miracenter geladen werden. Mittels Mausklick in die Karte wird die ungefähre Position des Roboters bestimmt. Danach orientiert sich der Scitos G5 automatisch und lokalisiert sich im

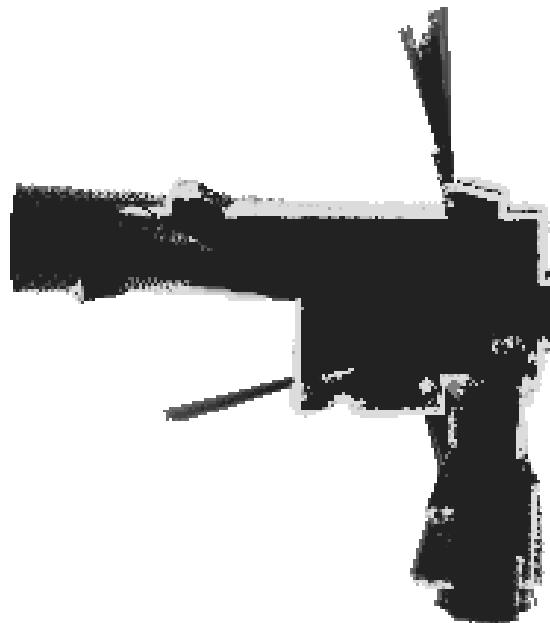


Abb. 4.6: Erstellter Grundriss eines Flures in Gebäude 29 der OvGU, Stockwerk 3

weiteren Verlauf selbst anhand seiner Sensordaten. Nun können ihm Positionen und Ausrichtung vorgegeben werden und durch seiner internen Pfadplanung steuert er sie autonom an. Durch ein Miracenter-ROS-Interface kann dabei die interne Lokalisierung und die Vorgabe von Position und Ausrichtung vom Scitos extern übermittelt werden. Die gesamte Software ist jedoch proprietär, d.h. nicht quelloffen, sodass ein tieferer Blick in die Funktionsweise der Software hier verwehrt wird.

4.2.3 Android-Studio, Estimote SDK und ROSjava

Als letzten Baustein in der Software-Architektur fehlt die App für das Moto G, deren Entwicklung auf dem Zusammenwirken dreier Grundpfeiler aufgebaut sein wird:

1. Das Android-Studio [16] ist eine Entwicklungsumgebung für Programme speziell von Android-Smartphones. Es bietet die Mittel eine Applikation, basierend auf der Programmiersprache Java, auf ein Android-Gerät zu portieren und dort auch zu testen. Aufgrund der Implementierung der „Integrated Development Environment“ (IDE) in Java, lässt es sich auf beinahe jedem Betriebssystem verwenden und bietet dadurch eine hohe Flexibilität in der Anwendung.
2. Damit die Applikation die Beacon-Signale verarbeiten kann, müssen die gesendeten Nachrichtenpakete und das verwendete BLE-Protokoll in das Programm eingebettet werden. Vom Hersteller der Beacons wird dafür eigens eine Bibliothek unter dem Namen „Estimote Software Development Kit (SDK)“ [13] bereitgestellt, die sich problemslos integrieren lässt und einfach zu bedienen ist.
3. Um die empfangenen Signale an den Server via P2P zu senden, muss eigens eine Schnittstelle von der Java-basierten Android-App zum in C/C++ gehaltenen ROS implementiert werden. Dafür wird das Paket „Rosjava“ [30] benötigt, um die Unterstützung von ROS-Funktionen für Java-Programme zu realisieren. Dies erlaubt es ROS-Pakete in eine Android-App zu integrieren und dadurch die zu übermittelnden Nachrichten-Formate darin auch zu verwenden.

4.3 App-Entwicklung

Um das geplante Programm auf dem Smartphone zu verwirklichen, werden die aus 4.2.3 beschriebenen Hilfen und zudem eigener Programmcode benötigt. In diesem Abschnitt werden die einzelnen Elemente der App erläutert und ihr Zusammenspiel in der fertigen Applikation anschließend betrachtet.

4.3.1 Beacon-Detektierung

Der kritischste, aber der auch am essentiell wichtigste Programmteil ist die Aufnahme von Beacon-Signalen. Hierfür muss die SDK von Estimote in die App über das Android-Studio als Bibliothek importiert werden. Daraufhin stehen neue Klassen und Objekte

der IDE zur Verfügung, die speziell auf die Verwendung von Beacons zugeschnitten sind. Zum einen muss dabei die App bei jedem Start für die Nutzung mit Beacons mit Abfragen neu initialisiert werden, das wären zum Beispiel die Überprüfung, ob Bluetooth angeschalten ist oder ob BLE überhaupt auf dem Gerät unterstützt wird. Des Weiteren muss der sogenannte Beacon-Manager, welcher im Hintergrund der App läuft, mit Zeiten für die Pause zwischen zwei Abtastungen und der Länge eines Scans, eingestellt werden. Die Problematik der Abtastraten wird hierbei später im Programmablaufplan noch erörtert. Zum anderen muss eine Art Empfangs-Funktion geschrieben werden, die bis zur Beendigung des Programms auf Meldungen der Leuchtfelder wartet und diese für die Verarbeitung im Programm aufbereitet. Die Anwendung von Filtern oder dergleichen entfällt hier, da schließlich das Verhalten der Signalausbreitung studiert werden soll und zusätzliche Einflüsse die Ergebnisse verfälschen könnten. Interessanter wird es bei der Frage, wie häufig die Abfragen der Eingänge nach Beacon-Signalen stattfinden sollen. Hierbei stellt sich noch eine viel wichtigere Frage und zwar der nach der Funktionsweise der SDK. Also wie die Datenströme, die von den Beacons in Intervallen gesendet, darauf vom Beacon-Manager aufgenommen werden. Denn die Sende-Intervalle der Beacons können variieren, während die Applikation mit festen Werten für alle Beacons initialisiert werden muss. Die Antwort nach den Abläufen im Beacon-Manager kann leider nicht beantwortet werden, da die Entwickler keinen Einblick in ihren Quellcode gewähren und auch sonst keine Angaben oder Dokumentationen veröffentlichen. Eine bessere Veranschaulichung dieser Problematik findet sich im Abschnitt über den Programmablauf.

4.3.2 Lagemessung

Parallel zur Beacon-Detektierung soll gleichzeitig die Lage des Smartphones gemessen werden, um die Antennen-Charakteristiken besser zu verstehen und ihre Einflüsse auf die Empfangsqualität zu untersuchen. Für diese Aufgabe müssen die Inertialsensoren oder auch englisch „Inertial Measurement Unit“ (IMU) aus Gyroskop, Beschleunigungssensor und Magnetometer vom Smartphone ausgelesen werden, was jedoch einfach umzusetzen ist, da die IDE nativ dafür Bibliotheken bereitstellt.

Bei der Messung stellt sich hierbei die Frage nach der Häufigkeit der Sensordatenabfragen, denn letztendlich senden die Beacon – zwar asynchron – ihre Signale alle 2 s bis 50 ms, jedoch können die Inertialsensoren in Intervallen von mehreren hundert Hertz ausgelesen werden. Aufgrund der höheren Verfügbarkeit der IMU-Daten, wäre es jedoch nicht zweckmäßig diese permanent über das Netzwerk zu senden und somit das Datenvolumen unnötigerweise künstlich aufzublähen. Es reicht daher völlig zu jedem Zeitpunkt an dem ein Messung nach Beacon-Signalen definiert ist, eine Lageposition vom Smartphone vorrätig zu haben. Jedoch lässt sich dabei durch die schnelleren Messungen die Qualität der Informationen mit einem Filter erhöhen. Im Hinblick auf noch genügend freie Rechenkapazitäten auf dem Smartphone (bisher lediglich Abfragen der Sensoren), wurde ein Komplementärfilter auch im

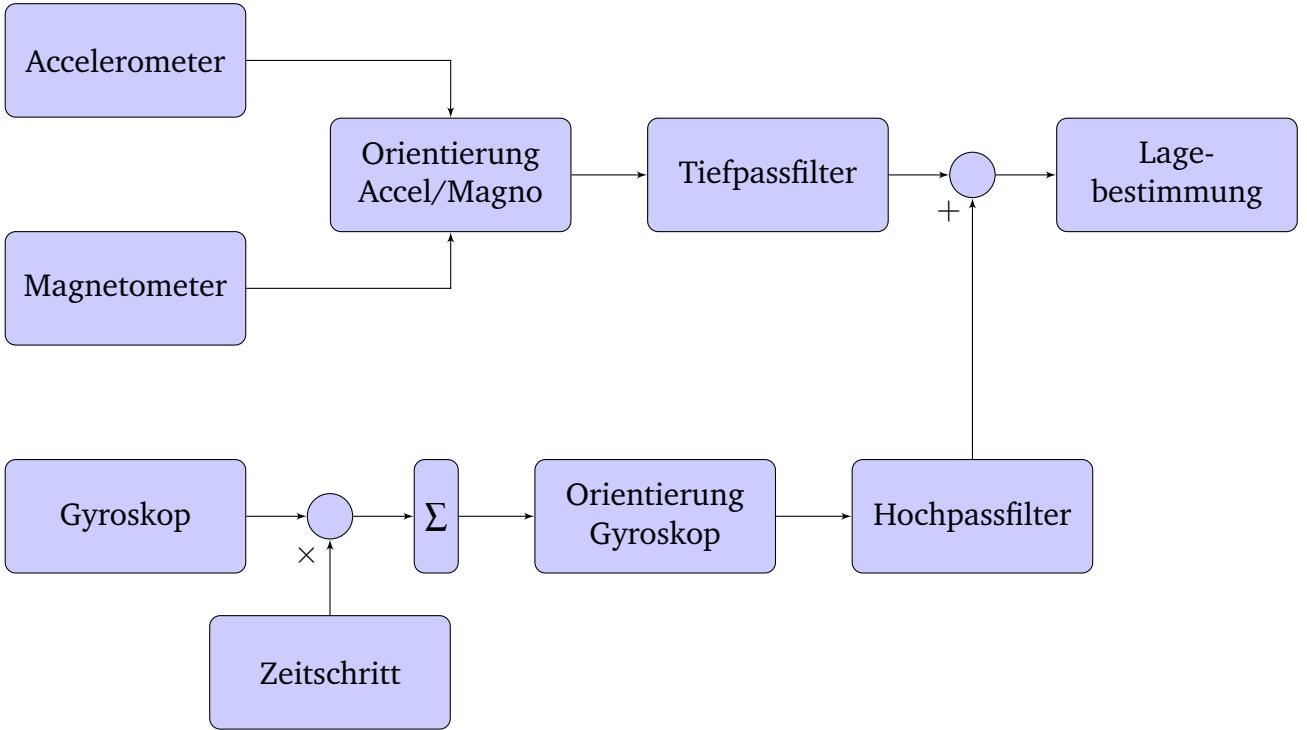


Abb. 4.7: Aufbau des Komplementärfilters, in Anlehnung an [28]

Hinblick auf die Sensordatenfusion von den drei IMU-Sensoren gewählt. Der dazu nötige Quellcode wurde aus [28] entnommen, der im Grunde den Drift des Gyroskops und das Signalrauschen des Beschleunigungssensors und des Magnetometers entfernt. Der Aufbau ist diesbezüglich in Abbildung 4.7 einmal skizziert. Die gesamte Funktionsweise beruht auf dem Prinzip der Fusion mehrerer Daten, um die Qualität der eigentlichen Information zu erhöhen. Es würde für den Zweck der Orientierungserfassung des Smartphones ausreichen, die Information vom Schwerkraft-Vektors des Beschleunigungssensors mit der des Magnetometers zusammenzufassen und lediglich diese auszugeben. Jedoch sind besonders die Daten des Magnetometers sehr verrauscht, sodass ein Tiefpass oder anders ausgedrückt ein Mittelwert aus den Daten gebildet wird. Da das Moto G zusätzlich über eine Gyroskop verfügt, welches die Bewegung des Smartphones viel genauer messen kann, wird auch diese Informationsquelle benötigt. Aber auch hier existiert es ein Nachteil in dessen Anwendung. Und zwar der Drift, sprich das Aufsummieren/Integrieren der Fehler oder des Messrauschen über die Zeit. Um auch diesem Problem zu begegnen, werden die Daten nur über kleine Zeitabschnitte mithilfe eines Hochpasses gesammelt. Dadurch werden anhand der Daten aus Accelerometer und Magnetometer als Stützinformationen und die Daten vom Gyroskop als Informationen über schnelle Änderungen verwendet.

4.3.3 ROS-Anbindung

Die Realisierung der Datenübertragung über eine ROS-Schnittstelle in der App wurde so umgesetzt, dass nach dem Start der Applikation ein Feld zur manuellen Eingabe der Adresse des Servers bzw. des ROS-Masters (z.B. die Internetprotokoll (IP)-Adresse) eingegeben werden muss. Dies ist nötig, um einen Zielort für die Nachrichten zu bestimmen. Nach der manuellen Eingabe wartet der sogenannte „Publisher“ – also der Programmteil, der die Nachrichten aufbereitet und an das Netzwerk übermittelt – auf die Messdaten. Dabei muss für den Publisher eine Bibliothek bereitstehen, die die Anzahl der zu übertragenden Informationen und deren Datentypen definiert und in Java integrieren kann, um sie später in der App zu verwenden. Dazu muss, wie anfänglich erwähnt die ROSjava-Erweiterung installiert sein und anschließend das Message-Format als Java-Paket damit erstellt werden. Da die versendeten Daten schon in vorigen Abschnitten erläutert wurden, wird in Abbildung 4.8 nur noch eine Zusammenfassung in Form der Implementierung präsentiert. Das dargestellte Message-Format enthält dabei die UUID empfangener Beacons als Zeichenkette oder String-Wert, sowie der empfangenen Signalstärke RSSI und der eingestellten Sendeleistung (bezeichnet als Power), als 32 Bit Integer und der daraus errechneten Distanz als 64 Bit Fließkomazahl (ein zusätzliches Feature der Estimote SDK). Des Weiteren beinhaltet sie die errechnete Orientierung vom Moto G und über allem den Zeitstempel der Zusammensetzung der Nachricht. Natürlich existiert zwischen Aufnahme der Messungen und dem Sendezeitpunkt eine zeitliche Differenz (kleiner als 1 ms Bearbeitungszeit), welche jedoch für die langsame Dynamik des Systems in den ersten Experimenten zu vernachlässigen ist.

```
string UUID
int32[] RSSI
int32[] Power
float64[] Distance
float64[] Orientation
float64[] Time
```

Abb. 4.8: Quellcode-Beispiel eines Message-Paketes für eine ROSjava-Implementierung

4.3.4 Programmablauf

Nachdem die einzelnen Module fertig gestellt wurden, musste aus ihnen ein gesamtes Programm zusammengefügt werden. Bei der Vereinigung der einzelnen Programmteile kristallisierten sich dabei verschiedene Probleme, die unter anderem auch auf den verstärkten Einsatz von proprietärer Software zurückzuführen sind. Denn bei der notwendigen Verwendung von nicht quelloffener Software, können Fehlfunktionen der App, die eigentlich auf dem Versagen der genutzten Software beruht, lediglich umgangen oder zumindest mit Einschränkungen im Ablauf der Applikation behoben wer-

den. Im konkreten Fall verursachte die gleichzeitige Verwendung von WLAN und Bluetooth, nach anscheinend willkürlichen Nutzungszeiten der App, Abstürze derjenigen. Das Problem scheint hierbei primär bei Android zu liegen, weil der Sachverhalt schon seit zwei Jahren in diversen Online-Foren bekannt ist, jedoch der Fehler weiterhin auch in den neuesten Versionen des Betriebssystems auftritt [32]. Der günstigste Fall den Konflikt zu lösen, wäre es einfach die Nutzung von Bluetooth durch die Estimote SDK und dem Gebrauch der WLAN-Verbindung besser zu organisieren. Jedoch bietet die SDK keine Einstellmöglichkeiten und keinerlei Dokumentation über die Verwendung des Bluetooth-Adapters, sodass auch nicht auf Seiten der WLAN-Übertragung die Anpassung stattfinden kann. Da die Zeit fehlte, sich weder mit dem Entwicklerteam der Firma Estimote auseinander zu setzen oder sich in das Android-Betriebssystem einzurbeiten, war die Lösung eher primitiv als elegant. In Abbildung 4.9 ist die gesammelte App als Ablaufplan mit Blockschaltbildern dargestellt. Während die eigentliche Aufgabe der Applikation schon im Vorfeld erklärt wurde, fällt hier eine zusätzliche Funktion, hier als Kontroll-Funktion bezeichnet, auf. Diese ist im Grunde eine Sicherheitsabfrage, wo vor der Messung und der Sendung der Nachricht über den Publisher kontrolliert wird, ob eine Verbindung zum P2P-Netzwerk existiert. Es fiehl in den unzähligen Testläufen des Programmes auf, dass sobald die WLAN-Verbindung wegbricht, der Publisher kein Ziel für seine Nachrichten ausfindig machen kann und dadurch es logischerweise zum Veragen der Funktionen führt. Durch Zuhilfenahme der Debug-Funktion vom Android-Studio konnte auch die Zeile ausfindig gemacht werden, die zum Absturz des Programmes führt. Und zwar versucht der Publisher nach der missglückten Transmission ständig auf das WLAN-Modul zuzugreifen und wenn dazu noch der Beacon-Manager versucht, den Bluetooth-Adapter bei der Detektierung zu aktivieren, geschieht der Zusammenstoß und das Programm wird außerplanmäßig beendet. Nachdem mit der Abfrage schon die Messung bei fehlender WLAN-Verbindung unterbunden wird und die Systeme erst deaktiviert und dann einzeln reaktiviert werden, lief die App durchgehend 24 Stunden ohne Vorkommnisse.

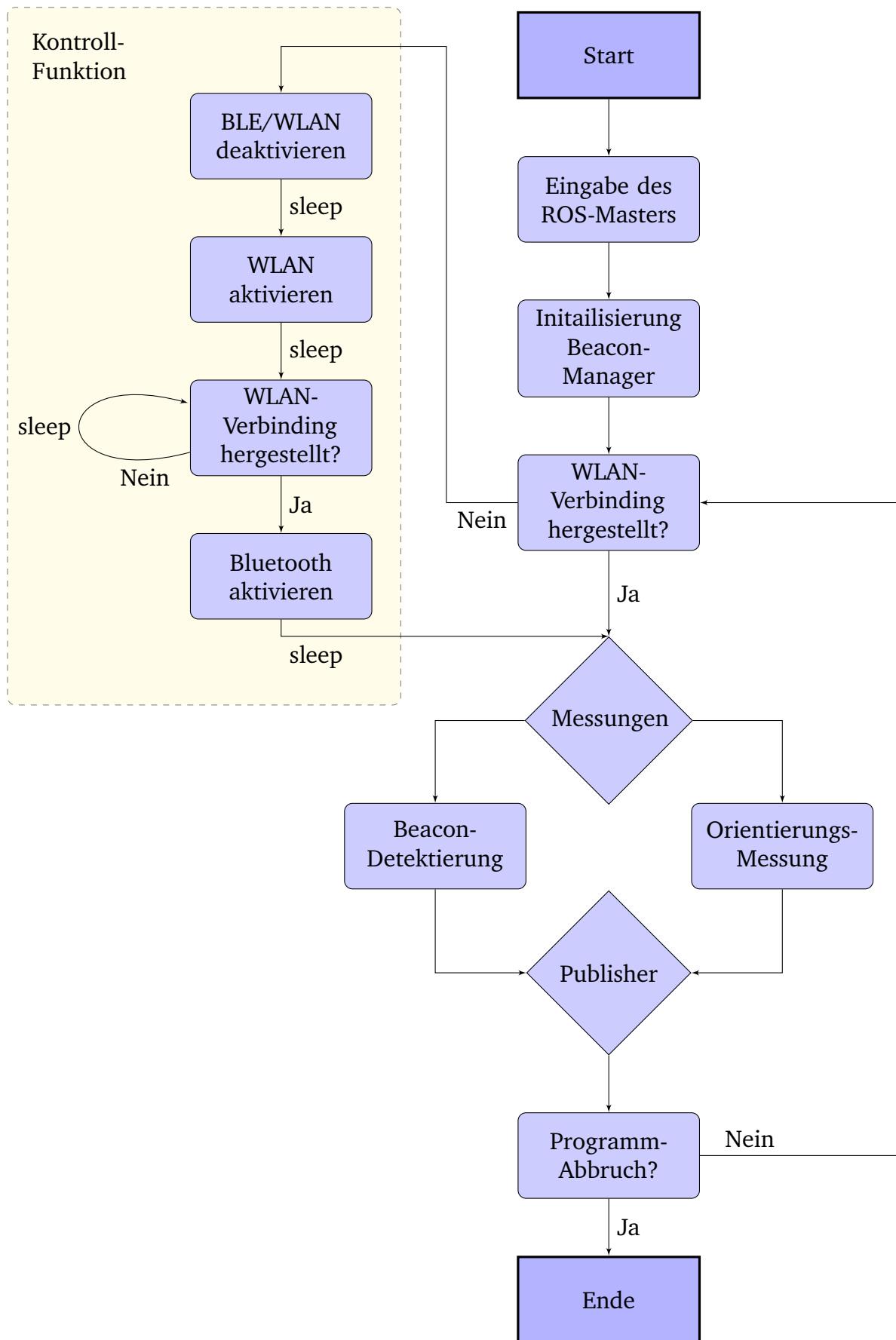


Abb. 4.9: Programmablaufplan der Lighthouse Keeper Applikation für Android-Smartphones

4.4 Experimente

Die hier vorgenommenen Versuche betrachten zum einen die Ausbreitung der Signale bei einer fest justierten, für die Arbeit vorher definierten Einstellung der Beacons. Zum anderen werden auch Aspekte der Beacon-Technologie betrachtet, die für spätere Forschungen von Interesse wären, aber auch die Notwendigkeit dieser Arbeit unterstreichen sollen. Denn es macht gerade den Reiz der Beacons aus, dass sie relativ einfach an ihre Bestimmung angepasst werden können. Schließlich ergab sich nach der Aufnahme von über 30 Stunden Sensordaten-Material die Erkenntnis, dass das gesamte Thema weit umfangreicher ist und die möglichen Modi genauere Untersuchungen verdienen. Gerade deswegen soll dies nach der ursprünglichen Aufgabe der Aufnahme von Messwerten für die Modellbildung nachträglich mit einigen Abschnitten über das Verhalten der Beacons gewürdigt werden.

4.4.1 Messung der BLE-Signalausbreitung

Für die Versuche wurde zunächst ein großer Raum benötigt, damit der Abstand von Beacon zu Smartphone um mehr als 15 Metern variiert werden kann. Das Equipment für die statischen Experimente bestand aus dem Youbot, drei Beacons, dem Moto G und einem Laptop mit installiertem Ubuntu Betriebssystem. In den Bilder ... im Anhang sind einige Fotos mit dem Aufbau hinterlegt. Die Messungen wurden dabei für jeden der drei Beacons für eine Distanz für jeweils zehn Minuten durchgeführt. Aufgrund der Masse an Daten werden hier nur drei Messreihen exemplarisch in den Abbildungen ... bis ... für die Signalstärke -12 dB und der Sende-Frequenz von 200 ms gezeigt. Es fällt dabei auf, dass die Abweichungen bzw. das Rauschen mit größerer Distanz zunimmt. Für die ersten fünf Distanz-Meter ist die über alle Messreihen die Regel, danach wird es sehr unbestimmt und nimmt sogar ab dem zehnten Meter bis zum fünfzehnten zwischenzeitlich ab. Diese können die „Verschmutzung“ des Raumes durch andere Signale im 2,4 GHz Band sein oder auch durch Reflektionen der Wände, der Decke oder des Fußbodens hervorgerufen werden. Aufgrund der schon erwähnten Asynchronität der Beacon-Signale, der nicht einsehbaren Messzyklendauer der Estimote SDK und/oder auch der parallelen Nutzung von Bluetooth- und WLAN-Modul des Smartphones, könnten die stärksten Signale von den Beacons zufällig nicht aufgenommen und nur deren „Echo“ wahrgenommen werden, weil ihnen zu dem Zeitpunkt, als die Signale die Antennen vom Smartphone passierten, niemand zuhörte. Dieser Verdacht sieht sich jedoch nicht bestätigt, wenn nur Abbildung ... betrachtet wird. Hier ist der Abstand zwischen Smartphone und Beacon sehr gering, nichtsdestotrotz werden keine reflektierten Signale aus größerer Entfernung aufgezeichnet, weder in dieser Messung noch in weiteren. Also muss die Verstärkung des Rauschens auf die physikalischen, statt den programmiertechnischen Erklärungen zurückzuführen sein. In dieser ersten Betrachtung des Systems kann schon von vornherein gesagt werden, dass das Rauschverhalten bzw. das Auftreten von Störungen raumabhängig und somit nur schwer vorhersagbar ist.

Um nun doch wie erwünscht eine qualitativ hochwertige Schätzung über die Signalausbreitung von BLE-Signalen zu gewinnen, muss es zum Anfang ein Modell geben, dass zuerst von einem völlig idealem Verhalten ausgeht. Es muss die Realität dabei nur im Groben abbilden und ausschließlich den Grundstein für die Simulation und deren Erweiterungen legen. Um eine Theorie dafür aufzustellen zu können, müssen die gesammelten Daten genauer betrachtet und die Ausbreitung des BLE-Signals der Beacons ohne Störungen herausgefiltert werden. In Abbildung ... sind einmal alle Messreihen in einem Diagramm zusammengefasst. Dabei wurde von jeder Messung eines jeden Beacons der Mittelwert der empfangenen Empfangsstärke über der Distanz zwischen Smartphone und Leuchtfeuer aufgetragen. Auf den ersten Blick fällt auf, dass lediglich bis fünf Meter dadurch eine genaue Aussage über die Entfernung getroffen werden kann. Die Unterschiede bzw. das Fehlen eines Unterschiedes macht es für größere Entfernungen schwer, anhand dessen eine qualitative Berechnung durchzuführen. Deswegen wird der Anteil an Informationen in Abbildung ... erhöht, indem nicht der Mittelwert aller Messungen, sondern die Verteilung der empfangenen Signalstärke farblich über die Entfernungen markiert wurde. Hier erscheint verstärkt das Phänomen, dass das Rauschen nicht mit der Distanz korreliert, sondern stark ortabhängig ist. Zumindest erhöht sich mit einem größeren Luftweg die Wahrscheinlichkeit, dass dies auftritt. Werden statt aller Messwerte einer Entfernung nur die 100 stärksten zur Mittelwertbildung herangezogen, wie es in Abbildung ... gezeigt wird, sieht die Entwicklung der Abnahme der Signalstärke deutlich vorhersehbarer aus und wird dadurch verwertbar.

(3 Bilder - 1, 5 und 10 Meter mit Mittelwert) (Bild Mittelwert aller Messungen über Distanz) (Bild der Verteilung) (Bild Mittelwert 100 bester Messungen über Distanz)

In dem Abschnitt über ROSjava wurde in Bild 4.8 eine von der SDK berechneten Distanz erwähnt. Diese ist jedoch nur eingeschränkt nutzbar, wie es Abbildung ... belegt. Hier wurden die tatsächlichen Distanzen mit den geschätzten der SDK gegenübergestellt und die Ergebnisse zeigen, dass diese Berechnungen nicht korrekt sind. Woran dies liegt lässt sich dabei nicht begründen und es kann wiederholt nur auf den nicht freien Zugang zu diesen Informationen verwiesen werden.

(Bild reale Distanz über berechneter)

Einfluss der Intervalllänge

Einfluss der Signalstärke

4.4.2 Auswirkung der Smartphone-Orientierung

Im Zuge der vorangegangenen Untersuchungen und unter der Vermutung des Einflusses der Antennen-Charakteristiken vom Moto G, stellte sich die Frage wie sehr die

Lage des Smartphones die Messungen beeinträchtigte. Schließlich ist der Empfänger, also das Moto G ständig in Bewegung und befindet sich dauernd in neuen Positionen und Ausrichtungen zu den Standorten der Beacons wieder. Dies gilt natürlich nicht für die statischen Versuchen in diesem Kapitel, sondern soll im Hinblick auf die dynamische Validierung mit dem Scitos G5 und auch später für die reale Anwendung erforscht werden. Es soll hierbei die Aussagekraft der gemachten Experimenten und zukünftigen Tests überprüft werden, welche die Grundpfeiler der Modellbildung und später der Simulation bilden. In diesem Sinne gilt es die Abhängigkeit der gemessenen Empfangsstärke des Signals zu der möglichen Orientierungen des Moto G zur Signalquelle zu ermitteln. Um den Aufwand der Messungen so gering wie möglich zu halten, wurden die drei verfügbaren Beacons so um das Smartphone positioniert, dass sie sich quasi auf den drei Raumachsen mit dem Moto G im Koordinatenursprung befinden (siehe Abbildung ...). Weil der Youbot seinen Roboterarm nur in Z-Richtung der Abbildung drehen konnte, ohne dabei die Abstände zu den Beacons zu ändern, wurden einige Konstellationen nicht betrachtet. Es soll hierbei jedoch nicht darum gehen, wie das Smartphone am besten zu halten sei, sondern ob die Intensität Signale aussgehend von der Decke, von den Wänden, von vorn oder von der Seite anders wahrgenommen wird. Die Ergebnisse finden sich in den Abbildungen ... bis ... wieder. Dort sind zum einen die gemessene Empfangsstärke der Beacons in festen Abständen zum Moto G zu sehen und dazu gegenüber die Ausrichtung an den drei Drehachsen. Es fällt auf, dass egal aus welcher Richtung die Signale auf das Smartphone treffen, der Empfang durch die Drehung gleichermaßen beeinträchtigt wird. Das Ergebnis überrascht insofern, dass eigentlich die Änderungen der erhaltenen Signalstärken zu jedem Drehwinkel die gleichen waren. Es gab keine Unterschiede, obwohl die Beacons alle in unterschiedlichen Winkeln zu dem Smartphone und somit zu dessen Antennen standen. Somit erklärt sich dieses Verhalten nicht anhand der Argumentation, dass eine Antenne bestimmte Charakteristiken aufweist und je nach Winkel der die Qualität des Signals ab- oder zunimmt. Um den Einfluss dieses Phänomens auf die statischen Versuche genauer abzuschätzen, wurden nachfolgend die Messungen aus 4.4.1 noch einmal wiederholt und nur lediglich die Drehung um besagte Z-Achse hinzugefügt (siehe Bild ... im Anhang). Die Ergebnisse in den Abbildungen ... bis ... verwunderten allerdings genauso, denn die fast proportionale Änderung von Ausrichtung zur Stärke der Signalaufnahmeleicht ist relativ leicht zu erfassen und sicherlich auch gut zu filtern, doch mit dem Verständnis wie Antennen funktionieren und der bekannten Theorien lässt sich dieses Verhalten nicht erklären. Die einzige Erkenntnis aus den hier durchgeföhrten Experimenten ist, dass die Vermutung der höchsten Empfangsgüte der Antennen bei normaler Halterung des Smartphones in der flachen Hand sich bewahrheitete und die Richtung, aus denen ein Signal stammt, dabei keinen Einfluss hat.

(Bild mit drei beacons auf den drei Achsen - Achsenbschriftung! - In der Mitte das Smartphone und die Drehung skizzieren)

(1 Bild von den Signalverläufen mit den drei Achsen und dazu der Verlauf der drei Drehwinkel) - Bei diesen Bildern noch erklären in welchem Abstand die Beacons aufgeteilt wurden und die Maxima und Minima kennzeichnen. Noch erklären das das

beste Signal bei 0 Grad liegt und wo halt das schlechteste liegt.

(1 Bild noch mit den drei ausgewählten Distanzen 1, 5 und 10 Meter und dazu auch die Drehwinkel) - Hier ebenso wie oben alles schön erklären und auch auf die Streuung eingehen, also ob die Signaleintrübung immer gleich blieb oder auch über die Distanz zunahm.

4.4.3 Wechselwirkungen zwischen Signalquellen

Unter dem Aspekt der angesprochenen „Verschmutzung“ eines Raumes durch andere Signalquellen, welche auch im 2,4 GHz Band senden, blieb ein Beweis dieser Behauptungen bisher aus. Der Beweis oder generell die Frage nach Wechselwirkungen mit anderen Transmittern ist insoweit auch wichtig, wenn es um einen Sicherheitsabstand zwischen zweier Beacons geht, bevor sie anfangen sich gegenseitig zu stören. In Bezug auf die generelle Störung von Bluetooth-Signalen durch beispielsweise WLAN-Geräten ist weitgehend bekannt und auch hinreichend erforscht (z.B. [26]). Jedoch bleibt die Frage, ob sich Beacons untereinander beeinflussen und wenn ja, wie groß der Mindestabstand gewählt werden muss, sodass die Auswirkungen vernachlässigbar sind. Zur Beantwortung wurden die drei verfügbaren Leuchtfeuer relativ nah zueinander befestigt (siehe Abbildung 4.10) und zu den gleichen Distanzen, wie in 4.4.1 die Signalstärken aufgenommen. In Abbildung ... sind die Ergebnisse aus einem Abstand von 10 cm der Beacons zueinander dargestellt. Im Vergleich zu den Messungen aus ... werden Unterschiede in der empfangenen Signalstärke sichtbar. Das Signal aus dem Beacon-Haufen ist bei gleicher Distanz von Beacon zu Smartphone mitunter 5 bis 10 dBm schwächer, als das Signal von einem alleinstehendem Beacon. Dies beweist somit die grundsätzliche negative Beeinflussung von nah beieinander liegenden Beacons auf deren Sendequalität. In Folge weiterer Untersuchungen ergab sich ein empfohlener Mindestabstand von 50 cm, durch den keine messbaren Verschlechterungen in der Signalstärke mehr vorkamen. An dieser Stelle ist anzumerken, dass hier generell mehr nicht gleich besser bedeutet. Im Hinblick auf die Planung eines Indoor-Lokalisierungssystems muss diese Betrachtung in den Gestaltungsprozess mithilfe einer Simulation mit einfließen, sodass der Sicherheitsabstand gewahrt bleibt.



Abb. 4.10: Gleichzeitige Messung dreier naheliegender Beacon-Signale

(Bild von allen drei signalen zusammen)

4.4.4 Energieverbrauch eines Beacon

Der Wartungsaufwand für ein Beacon hängt maßgeblich an der Laufzeit der in jedem Beacon eingebauten Batterie ab. Die in Kapitel 1 und 2 erwähnten Einstellmöglichkeiten von Sendeleistung und der Häufigkeit von Sendeintervallen beeinflussen dabei den Stromverbrauch der Beacon und somit auch die Betriebsdauer der Batterien. Dieser Abschnitt soll sich damit beschäftigen, inwieweit sich die Parameter auf die Wartungszyklen auswirken. Dies soll später dabei helfen, zwischen der Auslegung der beiden Parameter und des gewünschten Wartungsaufwandes abwägen zu können. Denn in großen Projekten mit mehreren hundert Beacons steigt der Wartungsaufwand proportional und wenn einzelne Beacons andere Einstellungen als die Mehrheit aufweisen (um beispielsweise wichtige Gebiete besser abzudecken), wird auch der Überblick über die nötigen Wartungszyklen verloren gehen. Um die Abhängigkeiten zu veranschaulichen, wird anhand vom Datenblatt des verwendeten Hardware-Chip nRF51822[27] in den genutzten Beacons eine Simulation entworfen, die die Einstellmöglichkeiten als Variablen betrachtet. Als Ausgangsbasis für den Energiespeicher wird eine CR2450-Batterie mit 1,8 Wh[33] angenommen. Hierbei wurde nicht vom Idealzustand ausgegangen, sondern ein üblicher Faktor von 0,7 hinzu multipliziert, um der Alterung der Zelle und weiterer Effekte Rechnung zu tragen. Durch die gering fließenden Ströme und der minimalen Bauweise der Beacons, können auch keine direkten Messungen vorgenommen werden. Zudem kann auch nicht verifiziert werden, wie lange ein Sendeaufrag oder ein Zustand vom Prozessor dauert. Um trotzdem ein gutes Simulationsergebnis zu erreichen, muss tiefer in die Funktionsweise der Beacons geschaut

werden. Als größte Unbekannte tritt dabei die Sendedauer auf. Bei einer angenommenen Datenrate von 250Kbit/s und bei einer Größe eines Datenpaketes von 38 Bytes [4] (siehe 4.11), dauert eine Sendesequenz rund 1 ms. Die benötigte elektrische Leistung für eine vorher definierte Sendeleistung beträgt dabei ein Vielfaches dessen und ist davon stark nichtlinear abhängig. Die dazu nötigen Beziehungen können anhand des Datenblattes des nRF51822 Chips hergeleitet werden. In diesem Fall wurde ein Polynom 4. Grades als Funktion der einstellbaren Sendeleistung aufgestellt, welches annähernd diese Abhängigkeiten beschreibt. Da die Kommunikation bidirektional ist, existiert auch ein Empfangsmodus der ebenfalls mit 1 ms Empfangsdauer angenommen wird, aber dessen Modus hingegen einen konstanten Stromverbrauch aufweist.

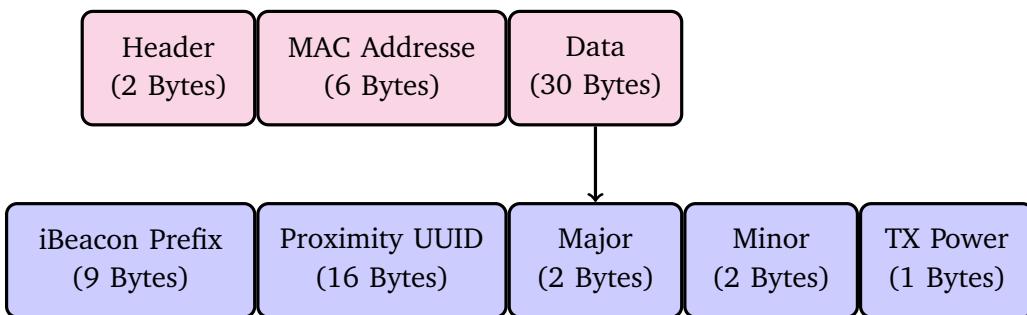


Abb. 4.11: Anteile eines Datenpaketes in der Beacon-Kommunikation auf der MAC-Ebene

Bei den tatsächlichen Laufzeiten des Sende- und Empfangsmodus muss jedoch ein Unsicherheitsfaktor mit eingerechnet werden, sodass nicht der berechnete Wert für eine Übertragung genommen wurde, sondern das Zweifache dessen, um ein weniger idealisiertes Bild für die einzelnen Vorgänge zu gestalten.

$$t_{TX} = 2 \text{ ms} \cdot \text{Intervalle [Hz]}$$

$$t_{RX} = 2 \text{ ms} \cdot 1 \text{ Hz}$$

$$t_{CPU} = t_{TX} + t_{RX}$$

$$t_{IDLE} = 1 - t_{CPU}$$

Da für den Sendevorgang noch zusätzlich der Cortex M0 Prozessor aufgeweckt werden muss und dieser das Paket vorbereitet und bis zum Ende der Sendung abwartet, muss hierfür zusätzlich der Energieverbrauch berechnet werden. Da wie erwähnt eine direkte Leistungsmessung an den Beacons durch die gering fließenden Ströme nicht möglich ist, werden aus dem Datenblatt des Prozessors diese rechnerisch ermittelt. Die aktive Phase des Prozessors für alle Operationen vor und nach einer Sendung werden dabei mit 2 ms Sekunden angenommen, welche exakt der Sende- und Empfangsdauer entspricht. Der Energiebedarf der Recheneinheit wurde im Datenblatt mit 4,1 mA bei einem Takt von 16 MHZ angegeben. So ergibt sich für den Prozessor eine Leistung von 12,3 mW. Zusätzlich wird der Stromverbrauch im IDLE-Modus (Schlafphase

des Prozessors) mit $2,6 \mu\text{A}$ angegeben. Aus den genannten Gegebenheiten lassen sich nachfolgende Gleichungen erstellen und damit eine Simulation eines Lebenszyklus von einer Batterie näherungsweise bestimmen:

$$\begin{aligned}
 E_{\text{Bat}} &= & 1,8 \text{ Wh} \cdot 0,7 &= & 1,26 \text{ Wh} \\
 L_{\text{CPU}} &= & 4,1 \text{ mA} \cdot 3\text{V} &= & 12,3 \text{ mW} \\
 L_{\text{TX}} &= \text{ Polynom 4. Grades (von } 4,7 \text{ mA } \sim 11,8 \text{ mA) } \cdot 3\text{V} &= & 14,1 \text{ mW } \sim 35,4 \text{ mW} \\
 L_{\text{RX}} &= & 6,1 \text{ mA} \cdot 3\text{V} &= & 18,3 \text{ mW} \\
 L_{\text{IDLE}} &= & 2,6 \mu\text{A} \cdot 3\text{V} &= & 7,8 \mu\text{W}
 \end{aligned}$$

ergeben den Zusammenhang

$$t_{\text{Laufzeit}} = \frac{E_{\text{Bat}}}{L_{\text{CPU}} \cdot t_{\text{CPU}} + L_{\text{TX}} \cdot t_{\text{TX}} + L_{\text{RX}} \cdot t_{\text{RX}} + L_{\text{IDLE}} \cdot t_{\text{IDLE}}}$$

und dies ergibt folgendes Simulationsergebniss:

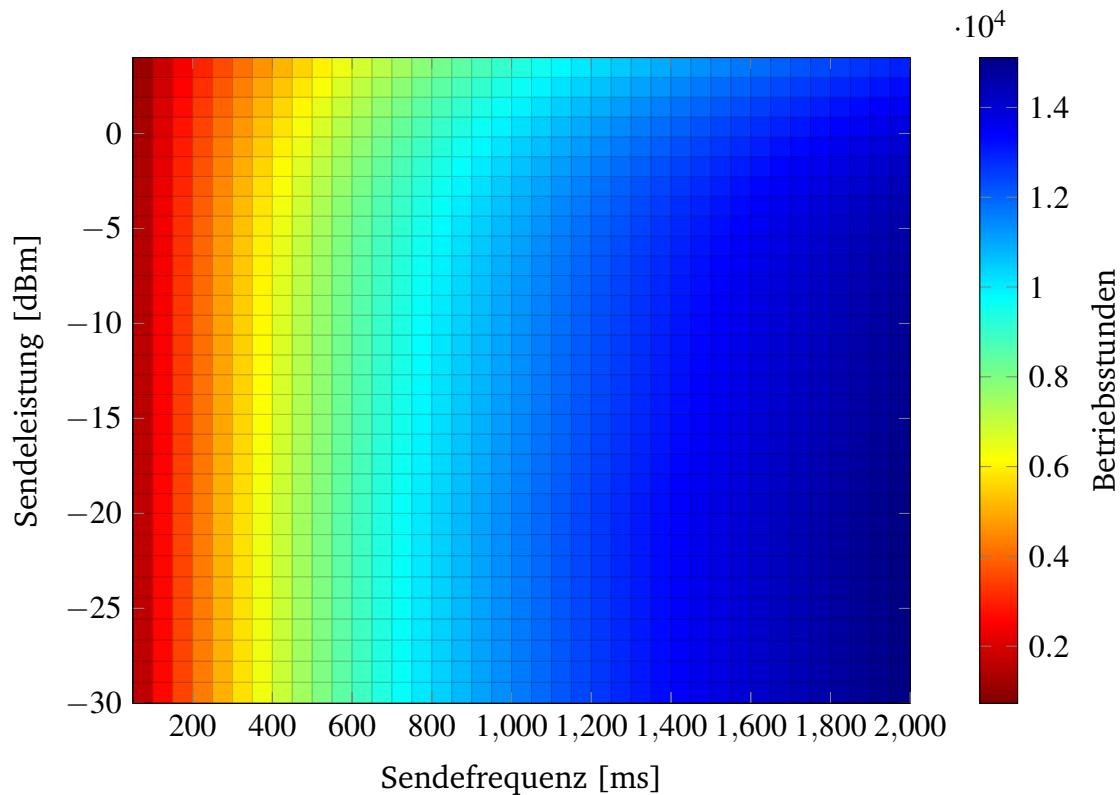


Abb. 4.12: Einfluss der einstellbaren Parameter auf die Lebensdauer einer CR2450-Batterie

Aufgrund der getroffenen Annahmen würde eine Batterie in den niedrigsten Einstellungen für rund 2 Jahre den Betrieb eines Beacon ermöglichen. Im realen Betrieb

hatte sich gezeigt, dass eine Batterie schon nach 4 bis 5 Monaten gewechselt werden musste. Während dieser Zeit wurden die Einstellungen nicht geändert und bei einer Intervalllänge von 5 Hz und Sendeleistung von -12 dBm belassen. Beides stimmt mit der Simulation ungefähr überein, was noch kein Beweis für die Gultigkeit der Annahmen bedeutet, jedoch wird der qualitative Verlauf der wechselseitigen Beziehungen von Einstellparameter und Batterielaufzeit skizziert, welcher als Orientierung zur Auslegung der Parameter benutzt werden kann.

Literaturverzeichnis

- [1] 5280MOBILE.NET: What is iBeacon. <http://5280mobile.net/ibeacon-proximity/>
- [2] A. ATHALYE ; V. SAVIC ; M. BOLIC ; P. M. DJURIC: Novel Semi-Passive RFID System for Indoor Localization,. In: IEEE Sensors Journal 13 (2013), Nr. 2, 528-537. <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6310003>
- [3] A. CAVALLINI: iBeacon Bible 2.0. <https://meetingofideas.files.wordpress.com/2014/06/ibeacon-bible-2-0.pdf>. Version: 2015
- [4] A. WARSKI: How do iBeacons work? <http://www.warski.org/blog/2014/01/how-ibeacons-work/>
- [5] AMAZON.COM, INC.: Motorola Moto G. http://www.amazon.de/Motorola-Smartphone-HD-Display-Megapixel-Quad-Core-Prozessor-Schwarz/dp/B00GJG0Q0I/ref=sr_1_1?ie=UTF8&qid=1428844430&sr=8-1&keywords=Motorola+Xt1032
- [6] APPLE, INC.: Getting Started with iBeacon. <https://developer.apple.com/ibeacon/Getting-Started-with-iBeacon.pdf>
- [7] B. KLUG: Motorola Moto G Review. In: www.anandtech.com <http://www.anandtech.com/show/7586/motorola-moto-g-review/7>
- [8] BLUETOOTH SIG, INC.: SIG INTRODUCES BLUETOOTH LOW ENERGY WI-FREELESS TECHNOLOGY, THE NEXT GENERATION OF BLUETOOTH WIRELESS TECHNOLOGY. In: bluetooth.com (2009). <http://www.bluetooth.com/Pages/Press-Releases-Detail.aspx?ItemID=4>
- [9] BLUETOOTH SIG, INC.: Bluetooth Smart Logo. <http://www.bluetooth.com/Pages/Bluetooth-Brand.aspx>. Version: 2014
- [10] D. E. DILGER: Inside iOS 7: iBeacons enhance apps' location awareness via Bluetooth LE. In: appleinsider.com (2013). <http://appleinsider.com/articles/13/06/19/inside-ios-7-ibeacons-enhance-apps-location-awareness-via-bluetooth-le.html>

- [11] <http://commons.wikimedia.org/wiki/File%3AConstellationGPS.gif>
- [12] ESTIMOTE, INC.: [Estimote Indoor Localization.](http://estimote.com/indoor/) <http://estimote.com/indoor/>
- [13] ESTIMOTE, INC.: [Estimote SDK for Android.](https://github.com/Estimote/Android-SDK) <https://github.com/Estimote/Android-SDK>
- [14] ESTIMOTE, INC.: [Estimote Beacon Illustration 2.](http://estimote.com/assets/gfx/press/press-beacon-illustration-2.cd71f93a.png) [http://estimote.com/assets/gfx/press/press-beacon-illustration-2.cd71f93a.png.](http://estimote.com/assets/gfx/press/press-beacon-illustration-2.cd71f93a.png) Version: 2015
- [15] G. BRAUNBERGER: [Macht der Maschinen.](http://www.faz.net/aktuell/wirtschaft/menschen-wirtschaft/digitale-revolution-macht-der-maschinen-12910372.html) <http://www.faz.net/aktuell/wirtschaft/menschen-wirtschaft/digitale-revolution-macht-der-maschinen-12910372.html>
- [16] GOOGLE, INC.: [Android Studio.](http://developer.android.com/tools/studio/index.html) <http://developer.android.com/tools/studio/index.html>
- [17] GOOGLE, INC.: [Google Indoor Maps.](https://www.google.com/maps/about/partners/indoormaps/) <https://www.google.com/maps/about/partners/indoormaps/>
- [18] H.-C. DIRSCHERL: [Stau-Warnung - Diese Verkehrslage-Dienste gibt es für Autofahrer.](http://www.pcwelt.de/http://www.pcwelt.de/ratgeber/Stau-Warnung-Google-Maps-Tomtom-Verkehrslage-Echtzeitverkehrsinformation.html) In: <http://www.pcwelt.de/http://www.pcwelt.de/ratgeber/Stau-Warnung-Google-Maps-Tomtom-Verkehrslage-Echtzeitverkehrsinformation.html>
- [19] http://www.bundesnetzagentur.de/SharedDocs/Downloads/DE/Sachgebiete/Telekommunikation/Unternehmen_Institutionen/Frequenzen/Allgemeinzuteilungen/2003_76_ISM_pdf.pdf?__blob=publicationFile&v=5
- [20] J. BLANKENBACH ; A. NORRDINE: [Indoor-Positionierung mit künstlichen Magnetfeldern.](http://geodesie.info/sites/default/files/privat/zfv_2013_1_Blanckenbach_Norrdine.pdf) In: [zfv 1 \(2013\).](http://geodesie.info/sites/default/files/privat/zfv_2013_1_Blanckenbach_Norrdine.pdf) geodesie.info/sites/default/files/privat/zfv_2013_1_Blanckenbach_Norrdine.pdf
- [21] KUKA AG: [Datenblatt Youbot.](http://www.kuka-labs.com/NR/rdonlyres/4833867A-D24F-410A-9AE4-300FBF671DFD/0/youBot_datenblatt_web_0514.pdf) http://www.kuka-labs.com/NR/rdonlyres/4833867A-D24F-410A-9AE4-300FBF671DFD/0/youBot_datenblatt_web_0514.pdf
- [22] M. QUIGLEY ; B. GERKEY ; K. CONLEY ; J. FAUST ; T. FOOTE ; J. LEIBS ; E. BERGER ; R. WHEELER ; A. NG: [ROS: an open-source Robot Operating System.](https://www.willowgarage.com/sites/default/files/icraoss09-ROS.pdf) [https://www.willowgarage.com/sites/default/files/icraoss09-ROS.pdf.](https://www.willowgarage.com/sites/default/files/icraoss09-ROS.pdf) – Forschungsbericht
- [23] METRALABS GMBH: [Scitos G5.](http://metralabs.com/index.php?option=com_content&view=article&id=70&Itemid=64) http://metralabs.com/index.php?option=com_content&view=article&id=70&Itemid=64

- [24] MICROSOFT RESEARCH: Mobile Indoor Localization. <http://research.microsoft.com/en-us/projects/indoorloc/>
- [25] MOBILE-STUDIEN.DE: Marktanteile mobiler Betriebssysteme Q1 2014. <http://mobile-studien.de/marktanteile-betriebssysteme/marktanteile-mobiler-betriebssysteme-q1-2014/>. Version: 2014
- [26] N. GOLMIE ; R.E. VAN DYCK ; A. SOLTANIAN ; A. TONNERRE ; O. REBALA: Interference Evaluation of Bluetooth and IEEE 802.11b Systems. In: Wireless Networks 9 (2003)
- [27] NORDIC SEMICONDUCTOR: nRF51822 Datasheet. <http://www.nordicsemi.com/eng/Products/Bluetooth-Smart-Bluetooth-low-energy/nRF51822>
- [28] P. LAWITZKI: Android Sensor Fusion Tutorial. <http://www.codeproject.com/Articles/729759/Android-Sensor-Fusion-Tutorial>
- [29] R. MARDENI ; S. N. OTHMAN: Node Positioning in ZigBee Network Using Trilateration MethodBased on the Received Signal Strength Indicator (RSSI). In: European Journal of Scientific Research http://godieboy.com/wp-content/uploads/2012/05/ejsr_46_1_05.pdf
- [30] ROSJAVA ORGANISATION: rosjava. <http://wiki.ros.org/rosjava>
- [31] ROS.ORG: About ROS. <http://www.ros.org/about-ros/>
- [32] R...@THEMMTEAM.COM: Nexus 4 loses wifi connection when bluetooth is in use. <https://code.google.com/p/android/issues/detail?id=41631>
- [33] SONY CORPORATION: Datenblatt CR2450B. <http://www.sony.net/Products/MicroBattery/cr/pdf/cr2450b.pdf>
- [34] TECHNISCHE UNIVERSITÄT BERLIN ; ADVANTIC SISTEMAS Y SERVICIOS S.L. ; iMINDS ; SICS SWEDISH ICT AB ; TELEVIC HEALTHCARE: Evaluation Of RF-Based Indoor Localization Solutions For The Future Internet (EVARilos). <http://www.evarilos.eu>
- [35] WWW.CONTECHLAB.COM: iBeacon Airport. <http://www.contechlab.com/ibeacon-application-in-airport/>
- [36] Y. CHEN ; D. LYMBEROPoulos ; J. LIU ; B. PRIYANTHA: FM-based Indoor Localization. <http://research.microsoft.com/pubs/163038/sys029fp-Chen.pdf>