1 Entwicklung eines Frameworks und Durchführung von Experimenten

Zur Zielsetzung Konfiguartionskonzept, Dinge die fehlen, Ableitung einer Vorgehensweise

Denn in großen Gebieten, wie z.B. Einkaufshäuser, Lagerhallen und Flughäfen braucht man Hunderte von iBeacons, um die gesamte Fläche zu 100 Prozent abzudecken. Im Idealfall kann ich später in Matlab noch Gebiete einpflegen, an der die Karte nach Regionen aufgeteilt wird, in denen die Genauigkeit der Lokalisierung eingestellt werden kann. Damit ließe sich Robotergestützt ein Gebiet abfahren, in denen Vorschläge für die Positionierung von iBeacons anhand eines Optimierungsalgorithmusses generiert werden. Hierbei ließe sich wunderschön eine Pareto-Front bilden, die aus Anzahl von iBeacons und Genauigkeit pro Region Positionierungsvorschläge entwirft. Je nach Kundenwunsch. ROTER FADEN ZU MEINER ARBEIT!

Hier kann angesprochen werden das wir ein adaptives Verfahren entwickeln wollen, denn viele bisherige Systeme bieten lediglich ein Landmarkenssystem oder ein Positionierungssystem. Mit den Beacons soll beides möglich sei und je nach Kundenwunsch anpassbar werden.

1.1 Verwendete Hardware

1.1.1 Motorola Moto G

Dieses Smartphone dient als Empfangsstation des Beacon-Signale.

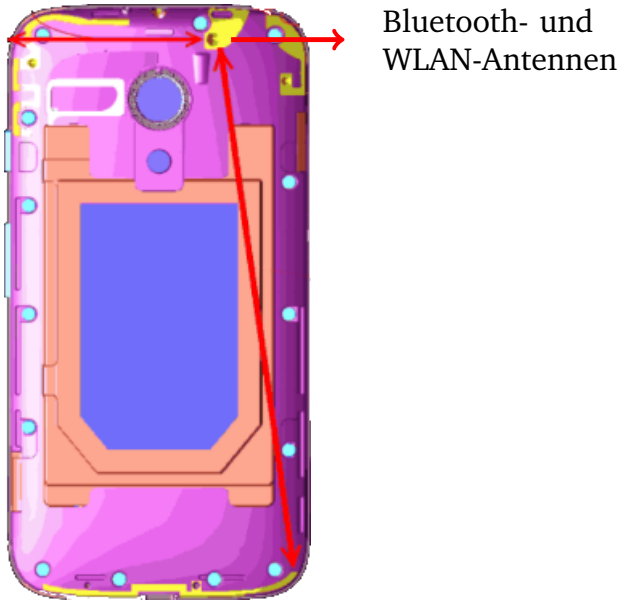

Abb. 1.1: Anordnung der Atennen an der Rückseite eines Motorola Moto G [3]

!Atennencharackteristiken! Idee für die Erklärung der seltsamen Empfangsqualität bei den verschiednen Ausrichtungen: durch die Verteilung der Antennen ergibt sich auch eine Verteilung der Empfangscharackeristiken. Die dabei aufgenommenen Messwerte sehen bei den

unterschiedlichen Ausrichtungen deshalb immer gleich aus, auch wenn die Signale aus einer anderen Richtung kommen, weil die Charackteristik speziell auf auf die normale Position optimiert wurde. Durch die Komplexität der Charackteristiken lässt sich dies nun schlecht visuell darstellen. Wenn man das Verhalten aber kennt (vorsicht, jedes Smartphone hat eine unterschiedliche Charakteristik), kann man Gegenmaßnahmen treffen. -> Heuristik

1.1.2 Youbot

Abb. 1.2: Youbot von Kuka [4]

1.1.3 Scitos G5

Bild vom Flur

1.2 Verwendete Software

1.2.1 ROS

1.2.2 Android-Studio

1.2.3 Estimote SDK for Android

1.2.4 App-Entwicklung

1.3 Versuchsplanung

Erklärung der Funktionsweise des Frameworks

1.4 Experimente

1.4.1 Energieverbrauch eines Beacon

Hier wird erklärt, dass sich durch die Einstellmöglichkeiten der Beacons auch deren Energiebedarfs der einzelnen Komponenten ändert (Mikrocontroller, BLE-Modul). Denn bei einer höheren Sendeleistung Verbraucht das BLE-Modul mehr Energie und bei einer Erhöhung der Sendefrequenz verbrauchen BLE-Modul und der Mikrocontroller (muss häufiger „aufwachen“) mehr. Graphik mit Leistungsverbrauch der beacons in Abghängigkeit mit ihren Einstellungen. Die Farbe soll dabei der Leistungsverbrauch und die Achsen die Intervallzeit und die Sendeleistung darstellen. Hier kann man sich auch überlegen die theoretische Akkulaufzeit mit anzugeben. Die Knopfbatterien sind vom Typ CR2450 mit einem Energiegehalt von ca. 1,8 Wh [6]. Der Mikrocontroller kann hier nicht mit eingerehnet werden, da lediglich ein Verbrauch in Abhängigkeit mit der Prozessorgeschwindigkeit [2] gegeben ist und für die Beacons keine Informationen vorliegen, in welcher Taktfrequenz diese betrieben werden. Natürlich hat der Prozessor auch eine höhere Taktfrequenz, wenn die Sendeintervalle der Beacons häufiger stattfinden und somit ist beides aneinander gekoppelt. Deswegen Messung des Stromverbrauchs am Beacon bei minimaler Sendeleistung mit den unterschiedlichen Intervallen (niedrigster Stufe, mittlerer und höhster) und dann einfach linear approximieren. Bei einer angenommenen Datenrate von 250Kbit/s [5] und bei einer Größe eines Datenpaketes von 30 Bytes [1] (siehe 1.3), dauert eine Sendesequenz ca. 0,96 ms. 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 eine direkte Leistungsmessung an den Beacons durch die gering fließenden Ströme nicht möglich ist, werden aus dem Datenblatt des Prozessors diese rechnerisch ermittelt. Die wache Phase des Prozessors für alle Operationen vor und nach einer Sendung werden dabei mit 1,4 ms Sekunden angenommen, was jedoch sehr großzügig bemessen ist und in der Realität weit darunter liegen wird. Der Energiebedarf ist im Datenblatt mit 5,1 μW/MHz angegeben und auf dem Beacon wurde ein zu 16 MHz oszillierender Quarz verbaut, sodass dieser als Taktgeber verwendet wird. So ergibt sich eine verbrauchte Leistung für eine Sendung zu 0,4 μWs.

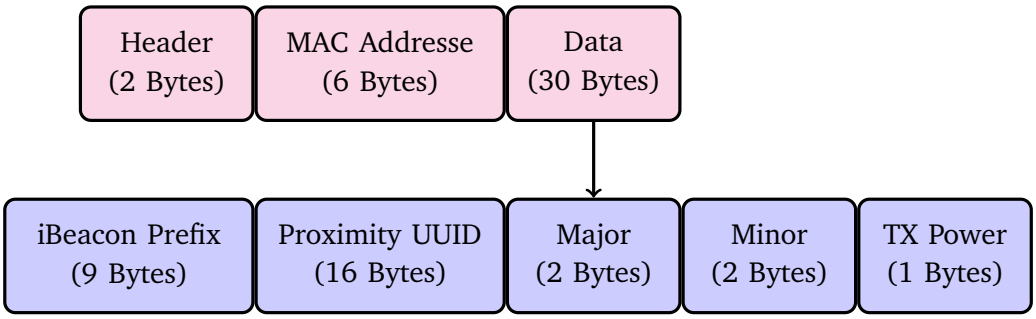

Abb. 1.3: Anteile eines Datenpaket in der Beacon-Kommunikation auf der MAC-Ebene

1.4.2 Distanzmessung mithilfe von Beacons

Zuerst werden zwei bis drei Distanzen und dazu die berechnete Distanz aus der App gegenübergestellt. Das wird schlecht sein und aus dem weitren Grund, dass es eine „Black Box“ ist

(d.h. man kann nicht einsehen, wie die Distanz aus dem RSSI-Wert berechnet wird), wird hier geplant ein eigenes Modell aufzustellen. Da wurde entschieden für verschiedene Distanzen nur die RSSI-Werte aufzunehmen, um später ein eigenes Modell zu implementieren.

1.4.3 Einfluss der Sendefrequenz

Wird das Rauschen vermindern.

1.4.4 Einfluss der Signalstärke

Dies soll auch in Abhängigkeit zur Signalsstärke passieren, um dessen Ausbreitung zu erforschen.

1.4.5 Auswirkung der Smartphone-Ausrichtung auf die Messungen

Hier kommen mal alle Messungen rein. Auch die mit den Beacons an allen drei Himmelsrichtungen.

1.4.6 Einfluss anderer Funkquellen auf die RSSI-Messung

Für die Erklärung aus dem zweiten Kapitel (2,4 Ghz) benötige ich auch die Störung und Interferenzen gleicher Signale. Ich muss ja schließlich erklären, warum ich nicht viele viele iBeacons nebeneinander klatschen kann, um eine möglichst genaue Standortbestimmung zu erhalten (denn die Signale stören sich untereinander). ir ist z.B. aufgefallen, dass die Beacons sich gegenseitig in der Sendestärke beeinflussen. Deswegen musste ich Einzelmessungen vornehmen und hier ließe sich auch begründen nicht zu viele Beacons in der realen Anwendung zu plazieren, um durch Elektrosmog nicht zu viele Einflussfaktoren in die Lokalisierung miteinfließen zu lassen. Viel hilft hier nicht immer viel, weniger ist manchmal besser, etc.

Literaturverzeichnis

[1] A. WARSKI: How do iBeacons work? http://www.warski.org/blog/2014/01/how-ibeacons-work/

[2] ARM LIMITED: Cortex-Mo Processor. http://www.arm.com/products/processors/cortex-m/cortex-m0.php

[3] B. KLUG: Motorola Moto G Review. In: www.anandtech.com http://www.anandtech.com/show/7586/motorola-moto-g-review/7

[4] KUKA AG: Youbot. http://www.kuka-labs.com/de/service_robotics/research_education/youbot/

[5] NORDIC SEMICONDUCTOR: nRF51822 Datasheet. http://www.nordicsemi.com/eng/Products/Bluetooth-Smart-Bluetooth-low-energy/nRF51822

[6] SONY CORPORATION: Datenblatt CR2450B. http://www.sony.net/Products/MicroBattery/cr/pdf/cr2450b.pdf