Developing an easy-to-use API for user input via Mobile Device

Ron Schiwkowksi (918691), Hannes Grothknopf (918740), Michael Schleiss (918825), Christian Heinrichs (919020), Dennis Hofmann (919285)

University of Applied Sciences, Sokratesplatz 1, 24149 Kiel, Germany

ABSTRACT

Keywords: Smartphone, JavaScript, HTML5, Nodejs, API

1. INTRODUCTION

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam.

1.1 Problem Statement

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam.

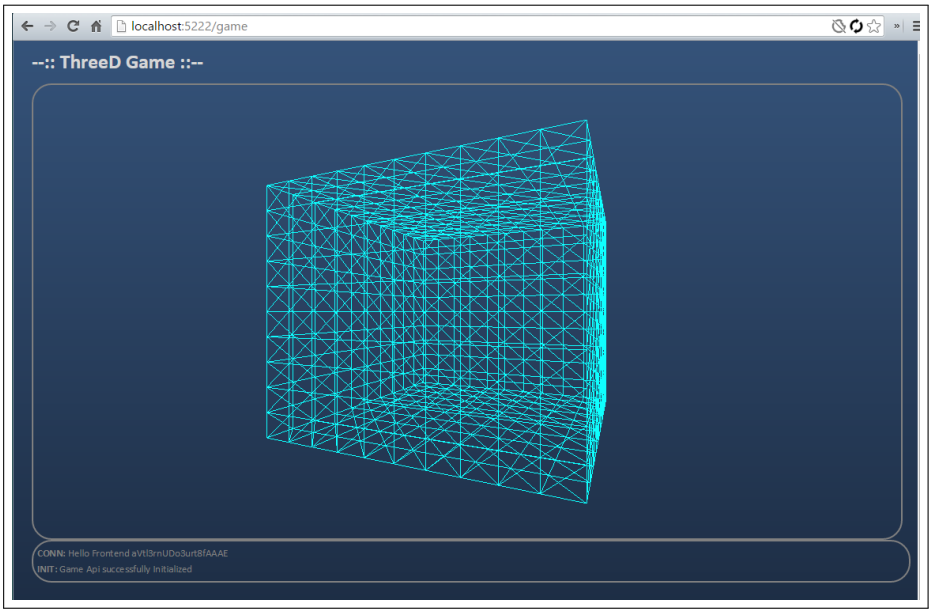

Abbildung 1. Nicht ausgelastet Bar

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.(siehe Abbildung 1).
Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam.

Tabelle 1. Matrix der Aufgabenverteilung

Aufgabe	Schleiss	Groth-knopf	Schiw-kowksi	Hofmann	Heinrichs
Recherche	X	X	X	X	X
Konzept	X	X	X	X	X
Serverdesign		X		X	X
Networking		X			X
GameAPI					X
DemoGame	X				X
ClientFrontend	X		X		
ClientBackend	X		X		
Database				X	
Login				X	
Projektmgmt.					X

2. METHODS

2.1 Project Management

Die Verteilung der Aufgaben ist in Tabelle 2.1 aufgestellt. Aufgaben sind in Redmine erfasst und Personen zugewiesen. Ein wöchentliches persönliches Treffen aller Projektmitglieder führt zu einer guten Arbeitsmoral und hohen Effektivität.

2.2 Technologies

2.2.1 NodeJS

Kismet ist eine frei erhältliche Anwendung zum passiven Abfangen von WiFi Paketen. Es fängt Pakete ab und liest deren Header Informationen aus. Dafür schaltet Kismet die WiFi Karte auf „RF monitoring“. Dadurch kann auch „management traffic“ abgefangen werden, was dazu führt, dass auch versteckte WiFi Netzwerke gefunden werden können.[?] Kismet kann mit sogenannten Drohnen betrieben werden. Dies können Klienten mit WiFi Karten sein, die nach Paketen scannen und die gewonnenen Informationen an einen Kismet-Server weiter leiten. So können zum Beispiel in einen Raum mehrere Drohnen verteilt werden, sodass über die gemessene Signalstärke der Pakete ein Rückschluss auf die Position des Ursprungs der Pakete getroffen werden kann. Ein solcher Systemaufbau ist in Abbildung 2 abgebildet.

Die Berechnung der Signalstärke wird nicht durch Kismet vorgenommen, sondern geschieht intern in der jeweiligen Netzwerkkarte. Es wird bei den Raspberry Pi ein LOGILINK WL0084B WiFi-USB-Stick verwendet. Die Dämpfung kann bei verschieden Netzwerkkarten stark abweichen. Deshalb ist es erforderlich, dass im Versuchsaufbau nur gleiche Netzwerkkarten verwendet werden.

2.2.2 HTML 5

Docker ist ein Werkzeug, dass Unterstützung beim Deploymentprozess von verschiedener Software bietet. Eine Möglichkeit, Software-Abhänigkeiten zusammenzufassen und einheitlich mit der Software auszuliefern, können weiterhin virtuelle Maschinen (VMs) sein. Diese VMs, die dann die Software und sämtliche Abhänigkeiten enthalten, können dann verteilt und ausgeführt werden. Einen ähnlichen Ansatz bietet die Docker Software. Jedoch ist das Ziel dieser Software nicht, komplette VMs herzustellen, sondern kleinere leichtgewichtige Container.

Diese Container basieren auf dem Konzept der Linux-Container (LXC) und teilen sich den Kernel der Host-Maschine. Daher benötigen die Container keine eigenen Betriebssysteme, welche bei paravirtuallisierten VMs mehrere Gigabyte Speicher in Anspruch nehmen können. Des Weiteren können sich Docker Container weitere Bibliotheken teilen, was wiederum Speicher und Rechenzeit einspart. Allerdings ist es nicht unbedingt einfach und eingängig Linux-Container zu konfigurieren und einzurichten. Diesen Teil übernimmt die Docker-Software. Darüber hinaus ist es mit Docker und dem zur Verfügung gestellten Software-Repository (Docker-Hub) schnell möglich, generierte Container als Ïmagesänderen Nutzern zur Verfügung zu stellen und zu verteilen.

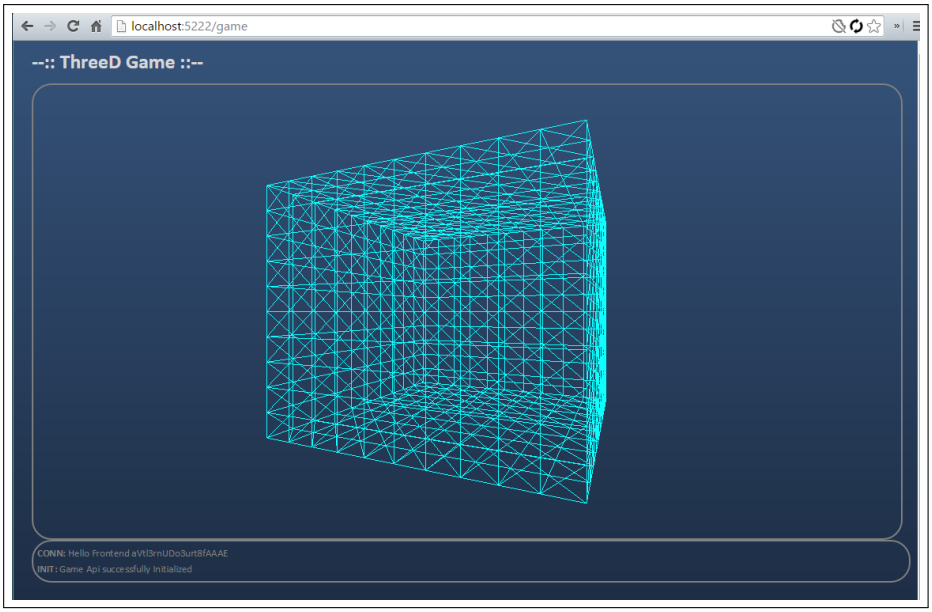

Abbildung 2. Systemarchitektur WiFi Tracking

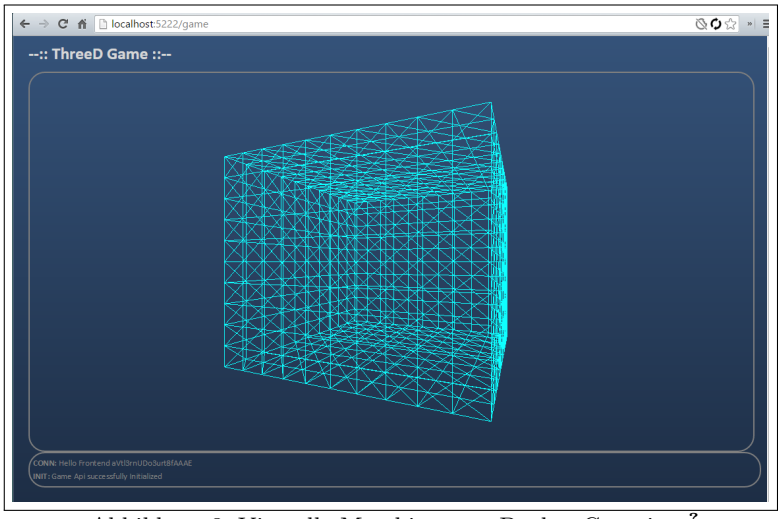

Abbildung 3. Virtuelle Maschinen vs. Docker Container[?]

2.2.3 Hardware

Als Kismet-Server dient ein übliches Notebook mit dem Betriebssystem Ubuntu. Als Drohnen werden Raspberry Pis (Model B) verwendet. Diese sind mit Wi-Fi USB-Sticks "LOGILINK WL0084B"bestückt. So wird sichergestellt, dass die Berechnung der Dämpfung bei allen Drohnen gleich ist. Die Raspberry Pis sind über ihre LAN-Schnittstelle mit einem Switch verbunden, mit dem auch das Notebook verbunden ist. Durch einen DHCP Server auf dem Ubuntu Laptop werden die IPs automatisch verteilt und eine Verbindung kann hergestellt werden.

Die Drohnen sind vorkonfiguriert in der Konfigurationsdatei des Kismet-Servers, eine Verbindung wird direkt beim Starten aufgebaut.

3. IMPLEMENTATION

3.1 Server and Networking

Die beim Kismet-Server auflaufenden Daten müssen noch für die Verarbeitung in eine zentrale Datenbank exportiert werden. Dafür wurde ein eigener Kismet-Client entwickelt. Dieser registriert sich bei dem Kismet-Server und

bekommt die Daten zugeschickt, sobald diese am Server anfallen. Diese Daten werden in der Mongo-Datenbank gespeichert. Das Programm ist in drei Pakete untergliedert:

Client-Paket
Das Client Paket beinhaltet die main Methode und initialisiert sowohl die Datenquelle als auch die Verbindung zum Kismet-Server. Des weiteren kümmert es sich um die Weitergabe der Daten von dem KismetClient an die Datenquelle.

Datasource-Paket
Die Datenquelle kümmert sich um das persistente Speichern der Daten. Hierfür wurde die MongoDB ausgewählt. Diese lässt sich einfach einrichten und mit Java verwenden. Außerdem kann sie über Webtechnologien leicht angesprochen werden. Dies ist wichtig für die spätere Auswertung der Daten, welche über Webtechnologien erfolgen soll. Die Datenquelle ist nicht an eine bestimmte Datenbank gebunden. Die Klasse muss lediglich das Datasource-Interface einbinden. Wie mit den Daten umgegangen wird, ist für den Client nicht wichtig und wird von der Klasse verwaltet.

KismetClient-Paket
Das KismetClient-Paket beinhaltet die Kommunikation mit dem Kismet-Server. Für die Kommunikation mit dem Server müssen die benötigten Protokolle mit den gewünschten Feldern bei diesem registriert werden. Danach werden automatisch Veränderungen an den Client gesendet und dieser muss diese auf Event-Basis auffangen und verarbeiten. Für die von uns benötigten Informationen, nutzt der Client die beiden Protokolle „SOURCE“ und „CLISRC“. Die benötigten Felder werden in den Tabellen 2 und 3 aufgezeigt.

SOURCE	
uuid	Die eindeutige ID der Quelle
username	Der Name der Quelle

Tabelle 2. Benötigte Felder des SOURCE Protokolls

CLISRC	
mac	Die MAC-Adresse des gefundenen Gerätes
uuid	Die ID der Quelle, welche das Gerät gesehen hat
lasttime	Timestamp der Sichtung
signal_dbm	Signalstärke des Geräts in DBM

Tabelle 3. Benötigte Felder des CLISRC Protokolls

3.2 Smartphone Client and Login

Zur Live-Evaluation der vom Kismet Client in die Datenbank geschriebenen Daten, wurde eine Webanwendung mit Hilfe des Webframeworks Meteor* geschrieben.
Die Anwendung wertet die vom Kismet Client bereitgestellten Messdaten reaktiv aus und lässt sich in einem Standard Webbrowser anzeigen. Zur Ermittlung der Position eines Gerätes werden die gemessenen Signalstärken für jedes Gerät von den unterschiedlichen Drohnen miteinander verglichen. Dabei wird der gleitende Mittelwert zur dritten Periode pro Drohne berechnet (siehe Abbildung 5), um mögliche Schwankungen in den Messungen zu behandeln. Wobei $x^d(t)$ für den Messwert zum Zeitpunkt t an Drogne d steht.
Da die Experimente ergeben haben, dass der gemessene dbm-Wert für ein Gerät geringer wird, je weiter man sich von der Quelle entfernt, kann so ermittelt werden, an welcher Drohne sich ein Gerät am nähesten befindet. Die berechneten Mittelwerte werden verglichen und der höchste Mittelwert bezeichnet die Drohne, und damit auch die Position, an der ein Gerät am nähesten ist. Anschließend kann gezählt werden, an welcher Drohne sich wie viele Geräte befinden. Mit Hilfe dieser Anwendung lässt sich das Ausgangsproblem lösen. Platziert man

*https://www.meteor.com/

$$\boxed{m_{MA}^{d}(t) = \frac{1}{3}\sum_{i=0}^{2} x^{d}(t-i)} \tag{1}$$

Abbildung 4. Gleitender Mittelwert zur dritten Periode

$$\boxed{m_{MA}^{d}(t) = \frac{1}{3}\sum_{i=0}^{2} x^{d}(t-i)} \tag{2}$$

Abbildung 5. Gleitender Mittelwert zur dritten Periode

an jeder Bar in der Einrichtung eine Drohne, kann diese Anwendung erkennen, wie viele Personen sich an den unterschiedlichen Bars befinden und darauf reagieren, indem auf einem Fernseher Ermäßigungen für eine weniger besuchte Bar angeboten werden.

3.3 GameAPI

Zur Live-Evaluation der vom Kismet Client in die Datenbank geschriebenen Daten, wurde eine Webanwendung mit Hilfe des Webframeworks Meteor[†] geschrieben.
Die Anwendung wertet die vom Kismet Client bereitgestellten Messdaten reaktiv aus und lässt sich in einem Standard Webbrowser anzeigen. Zur Ermittlung der Position eines Gerätes werden die gemessenen Signalstärken für jedes Gerät von den unterschiedlichen Drohnen miteinander verglichen. Dabei wird der gleitende Mittelwert zur dritten Periode pro Drohne berechnet (siehe Abbildung 5), um mögliche Schwankungen in den Messungen zu behandeln. Wobei $x^{d}(t)$ für den Messwert zum Zeitpunkt t an Drogne d steht.
Da die Experimente ergeben haben, dass der gemessene dbm-Wert für ein Gerät geringer wird, je weiter man sich von der Quelle entfernt, kann so ermittelt werden, an welcher Drohne sich ein Gerät am nähesten befindet. Die berechneten Mittelwerte werden verglichen und der höchste Mittelwert bezeichnet die Drohne, und damit auch die Position, an der ein Gerät am nähesten ist. Anschließend kann gezählt werden, an welcher Drohne sich wie viele Geräte befinden. Mit Hilfe dieser Anwendung lässt sich das Ausgangsproblem lösen. Platziert man an jeder Bar in der Einrichtung eine Drohne, kann diese Anwendung erkennen, wie viele Personen sich an den unterschiedlichen Bars befinden und darauf reagieren, indem auf einem Fernseher Ermäßigungen für eine weniger besuchte Bar angeboten werden.

3.4 Demonstration

3.4.1 Preparations

1. Ein Computer ist als Kismet-Server sowie als Meteor Server konfiguriert.
2. Die Raspberry Pis, die als Kismet Drohnen konfiguriert sind, werden an die entsprechenden Positionen platziert, an denen die Anzahl der Geräte gemessen werden soll. Im Anwendungsfall dieses Projekts wird eine Drohne jeweils an einer Bar platziert.
3. Die Drohnen werden per LAN - Kabel an das Netzwerk verbunden, in dem sich auch der Kismet-Server befindet.

3.4.2 Start applications

1. Der Kismet-Server wird mit dem Befehl *sudo kismet* über den Terminal gestartet.
2. Die Meteor Anwendung wird mit dem Befehl *meteor* über den Terminal gestartet. Die Mongo Datenbank wird dabei automatisch mit gestartet.
3. Der Kismet Client wird mit Hilfe des Java Archives gestartet. Als Parameter können beim Start über den Terminal die IP - Adressen und Ports des Kismet-Servers und der Mongo Datenbank übergeben werden, sollten diese von den Standardkonfigurationen abweichen.

[†]https://www.meteor.com/

3.4.3 Running

1. Der Kismet-Server gibt im Terminal Statusnachrichten zu den Drohnen und gefundenen Netzwerken aus.
2. Die Meteor Anwendung kann über einen Browser aufgerufen werden. *[IPDesMeteorServers]:3000*
3. Die Mongo Datenbank kann über den Befehl *meteor mongo* im Terminal aufgerufen werden.
4. Der Kismet Client gibt im Terminal Statusnachrichten zu den Informationen, die gespeichert werden aus.

4. RESULTS

4.1 Measurements

Structure
Im Masterlabor wurde die ersten Probemessung durchgeführt, um zu testen, ob eine Positionsbestimmung durch Vergleichen der Dämpfungswerte überhaupt möglich ist. Dazu wurden zwei Drohnen an verschiedenen Enden des Raumes platziert. Bei der Messung wurde eine App benutzt die sekündlich einen Broadcast nach Access Point durchführt. Dadurch wird traffic generiert, der nötig ist, damit die Drohnen genug Pakete abfangen.

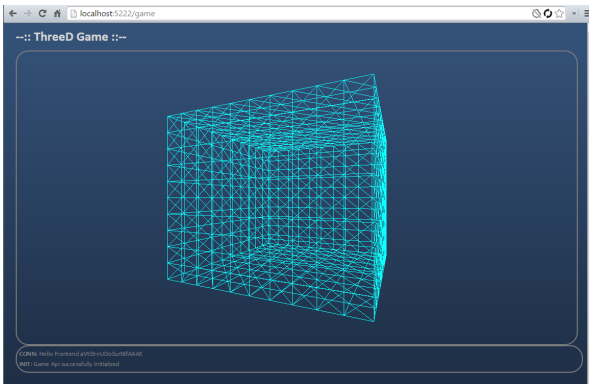

Abbildung 6. Probemessung

Ablauf der Messung
Anschließend wurde die in Abbildung 6 zur erkennende Strecke mit einem Smartphone abgelaufen. Beim Abgehen der Strecke wurde nach circa jedem Meter eine kurze Zeit verweilt, damit ein aussagekräftigeres Ergebnis erzielt werden konnte.

Results
Auf Abbildung 9 ist zu sehen von welcher Drohne sich das Gerät entfernt und welcher es sich nähert.

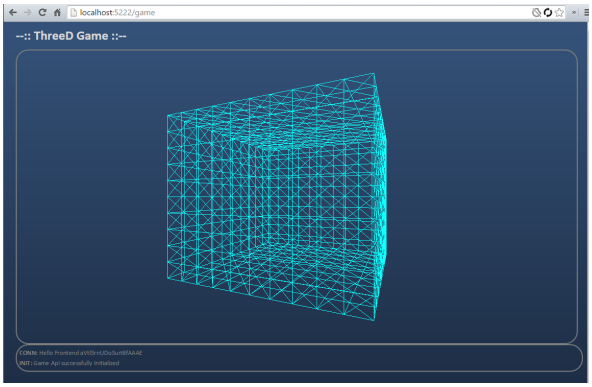

Abbildung 7. Berechnungen zum Ergebnis

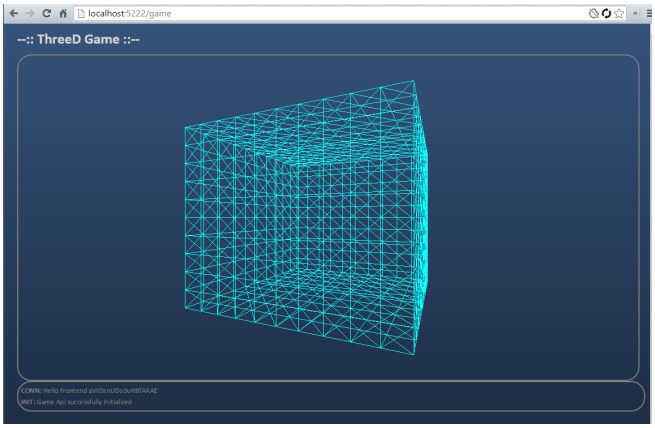

Abbildung 8. Ergebnis der Probemessung

4.2 Limitations

4.2.1 1. Test

Aufbau
Die Drohne und das Smartphone wurden auf dämpfenden Platten positioniert. Auf dem Gerät war die App zur Traffic-Generierung aktiviert. In Abständen von 50 cm, 100 cm und 200 cm wurden Messungen durchgeführt.

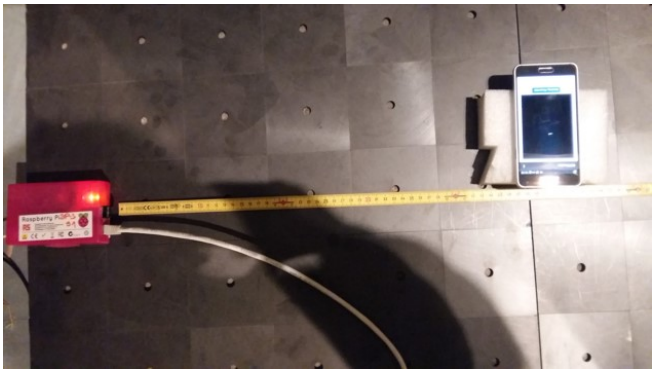

Abbildung 9. Aufbau 1. Test Störungsfreier Raum

Ergebnis
Selbst auf kurzer Distanz ist eine Differenz von ca. 10 dBm zwischen den Messpunkten zu sehen. Dies ermöglicht eine Ortsbestimmung auf kurzer Distanz, was beim nächsten Test ebenfalls zu sehen ist.

Entfernung	50 cm	100 cm	200 cm
Durchschnittliche Dämpfung	-62,22 dBm	-75,45 dBm	-82,52 dBm

Tabelle 4. Ergebnisse des 1. Tests im störungsfreien Raum

4.2.2 2. Test

Aufbau
Um zu ermitteln, wie genau der Vergleich von Dämpfungswerten für die Positionsermittlung benutzt werden kann, wurden zwei Kismet-Drohnen in 12 Meter Abstand voneinander aufgestellt. Diese waren über einen Switch mit dem Kismet-Server verbunden. Alle 120 cm wurde ein Messpunkt festgelegt. Dies entsprach der Größe einer Bodenplatte im störungsfreien Raum und wurde deshalb als Distanz gewählt.

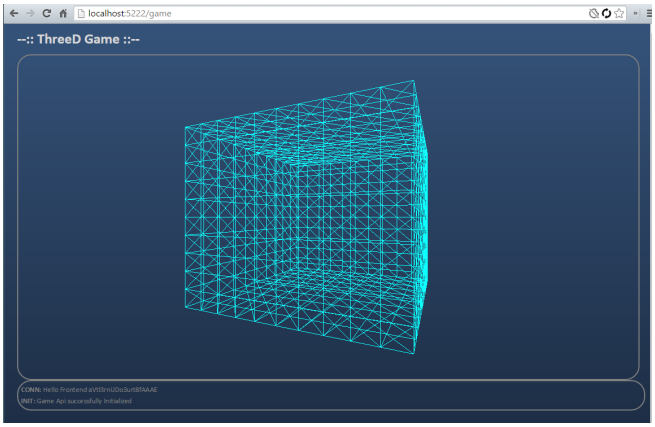

Abbildung 10. Aufbau 2. Test Störungsfreier Raum

Ablauf der Messung
Ein Smartphone wurde auf den entsprechenden Messpunkt gelegt. WiFi war auf dem Gerät eingeschaltet und die App zur Traffic-Generierung war aktiv. Es wurden pro Messpunkt circa 100 Messwerte aufgenommen, damit eine aussagekräftiges Ergebnis erzielt werden konnte und kurzzeitige Störungen abgefangen werden konnten.

Ergebnis
Wie schon bei der ersten Probemessung ist eine deutliche Tendenz zu erkennen. Bei der Drohne, von der sich das Smartphone entfernte, nahm die Dämpfung zu. Bei der anderen Drohne nahm sie ab. Die aufgenommen Dämpfungswerte sind bei circa 480 cm gleich groß (siehe Diagramm 14 im Anhang). Dies ist 120 cm von der eigentlichen Mitte entfernt. Somit besteht eine Messungenauigkeit von 120 cm.

4.3 Realistische Messung

Aufbau
Der Aufbau der Messung war der gleiche wie in der 2. Messung im störungsfreien Raum (siehe 4.2.2). Dieses mal fand die Messung allerdings in einem einfachen Flur der FH-Kiel statt und war somit nicht vor Störungen geschützt.

Ablauf der Messung
Ein Smartphone wurde an den einzelnen Messpunkten platziert. Das WiFi war auf dem Gerät aktiviert. Allerdings war das Gerät mit keinem WiFi-Netzwerk verbunden um die realen Umstände in einer Discothek nachempfinden zu können. Da von Anfang an festgestellt wurde, dass deutlich weniger Pakete abgefangen wurden, wurde die Dauer der Messung an jedem Messpunkt auf 2 Minuten festgesetzt.

Ergebnis
Zu beobachten war, dass in den 2 Minuten 3-9 Pakete abgefangen wurden. Das macht eine Echtzeit-Bewegungsverfolgung so gut wie unmöglich. Ebenso war die Differenz der gemessenen Dämpfungswerte geringer, was eine Positionierung des Geräts erschwert. Das liegt daran, dass auf dem Flur Strahlen der WiFi-Antenne des Smartphones weniger stark gedämpft wurden.

5. CONCLUSION

Eine passive Ortung von WiFi-Geräten anhand der Signalstärke ist möglich. Eine Installation einer zusätzlichen Anwendung auf den zu ortenden Geräten ist nicht notwendig. Es ist zu beachten, dass durch die starken Schwankungen in der Signalstärke nur eine Positionsbestimmung auf 1,2 Meter möglich ist (siehe Diagramm 14 im Anhang).
Ein weiteres Forschungsfeld ist die zeitliche Genauigkeit der bestimmten Position. Da nur unregelmäßig und nicht vorhersagbar Daten eingehen ist eine Echtzeitortung nicht trivial und bedarf weiterer Untersuchung. Die in dieser Arbeit vorgestellte Technologie eignet sich für die Raum bezogene Ortung von Geräten.
Rechtliche Rahmenbedingen sind bei der vorgestellten Positionsbestimmung einzuhalten. Es ist zu klären, in wie weit die MAC-Adresse eines Gerätes unter Datenschutzbestimmungen fällt.
In einer zukünftigen Weiterentwicklung kann versucht werden die genaue Position des Gerätes über Triangulation zu bestimmen. Es ist möglich aus den erfassten Daten die Position zu berechnen, solange es mindestens drei Messtationen gibt. Einige Schwierigkeiten dabei sind zum Beispiel Objekte im Messbereich, Wände sowie Personen. Diese verändern die Signalstärken und müssen im Algorithmus berücksichtigt werden. Dies führt zu einer weiteren nicht einfach zu lösenden Problemstellung welche die Aufstellung des Algorithmus abdeckt.

Im Weiteren kann die Positionsbestimmung über eine Fingerprint-Map durchgeführt werden. Hierbei ist ein genaueres Ergebnis zu erwarten als bei der Berechnung mittels der Triangulation. Dies liegt an der leichteren Berücksichtigung von starren Objekten in dem Messgebiet. Der Nachteil liegt in der vorherigen Aufstellung der Karte. Für diese muss das Gebiet vermessen worden sein, um eine Datenbank mit Signalstärken aufzustellen. Eine weitere Erforschung sollte beide Ansätze in Betracht ziehen und kombinieren, um die genaue Position des Nutzers bestimmen zu können.

APPENDIX A. OUTLOOK

A.1 Anforderungsanalyse

Die Einlasskontrolle ist in Meteor entwickelt. Eine Anforderungsanlyse mit Diskothekenbesitzern ergab folgende Anforderungen:

- Gäste können sich auf eine Gästeliste über eine Micro-Event-Page setzen (bereits existent in einer Java-Anwendung)
- Gäste bekommen bei der Anmeldung auf der Gästeliste eine E-Mail mit einem QR-Code zugesendet
- Gäste werden in einer Liste für jedes Event angezeigt
- In der Liste können Gäste durch anklicken eingechecked werden
- Gäste können alternativ über einen QR-Code-Scanner eingechecked werden
- Checkins werden in einem Diagramm zusammengefasst

A.2 Konzept und Implementierung

Als Framework wird Meteor eingesetzt. Über eine API wird bei jeder Anmeldung von einem Gast auf der Gästeliste der Gast an die Meteor-Anwendung übertragen. Bei einem Checkin wird der neue Status von der Meteoranwendung an die bestehende Java-Anwendung übertragen.
Der an den Gast via E-mail gesendete QR-Code enthält eine per Java generierte einmalige ID (UUID), welche mit der Gast-Anmeldung für ein Event verknüpft ist.
Für den Scanvorgang wird das Framework jsqrcode in Meteor eingebunden. Der Eventhandler qrScanner.on('scan') wird jede Sekunde gefeuert. In der Callback-Funktion wird die Variable messages zurückgegeben. Ist diese nicht null, enthält sie den Wert des QR-Codes, welcher zum Checkin verwendet wird.

A.3 Ausblick

Die Einlasskontrolle kann wie folgt mit dem Indoor-Tracking verbunden werden, um Personen in einem Raum zu orten:

1. Eine Person wird über einen QR-Code erkannt.
2. Das stärkste Signal von Smartphone (MAC) wird erkannt.
3. Die MAC-Addrese wird der Person zugeordnet.
4. Das Smartphone beziehungsweise die Person wird im Raum erkannt.

APPENDIX B. ERGEBNISSE DER MESSUNGEN IM STÖRUNGSFREIEN RAUM

Hier sind die Messergebnisse des 2. Tests aus dem störungsfreien Raum zu sehen. Es ist zu beobachten, dass sich die Linien der beiden Drohnen annähern und dann wieder auseinander driften. Beim Diagramm 14 sind die Dämpfungswerte etwa gleich groß, was auf einen Messungenauigkeit von 1,2 Metern schließen lässt, da der eigentliche Mittelpunkt bei 600 cm liegt.

Abbildung 11. Messergebnis beim 120 cm