Hochschule für Technik
und Wirtschaft Berlin
University of Applied Sciences

Forschungsbericht

Visuelle und elektronische Objektidentifikation zur Nutzung in Augmented Reality Anwendungen

Christian Bunk

Kurs: Forschungsprojekt 2
Betreuer: Prof. Dr. Jürgen Sieck
Studiengang Angewandte Informatik (Master)
Abgabedatum: 05. April 2012

Inhaltsverzeichnis

1 Einleitung **1**

2 Verwandte Arbeiten **2**

3 Visuelle Objekterkennung **3**
3.1 Histogramme . . . 3
3.2 Feature Tracking . . . 6
3.3 Texterkennung . . . 11

4 Elektronische Objekterkennung **12**
4.1 Verfahren zur Lokalisierung . . . 12
4.1.1 Lateration . . . 12
4.1.2 Triangulation . . . 12
4.1.3 Proximity . . . 13
4.1.4 Dead Reckoning . . . 13
4.1.5 Received Signal Strength Indicator (RSSI) . . . 13
4.1.6 Einflüsse auf die Signalstärke . . . 14
4.1.7 Time Difference of Arrival (TDoA) . . . 14
4.1.8 Angle of Arrival (AoA) . . . 14
4.2 Objekterkennung durch Funkstandards . . . 15
4.2.1 WLan . . . 15
4.2.2 Bluetooth . . . 15
4.2.3 Near Field Communication . . . 16
4.3 Weitere Sensoren . . . 17
4.3.1 GPS . . . 17
4.3.2 Beschleunigung . . . 17
4.3.3 Proximity . . . 18
4.3.4 Licht . . . 18

5 Implementierung **20**

6 Zusammenfassung **22**

Literatur **23**

Internetquellen **26**

Abbildungsverzeichnis

1 Backprojection (Bildquelle: [Lag11]) 4
2 Objekterkennung - Mean Shift 5
3 Berechnete Featurepunkte 7
4 Matches der Featurepunkte (links Originalbild, rechts Vergleichsbild) . . 8
5 Weiß gute Matches, Schwarz verworfene Matches 9
6 Berechnungszeiten 10
7 Bücher mit gleichem Cover 11
8 Trilateration. Quelle: [5] 13
9 Gemessene Lichtwerte am Fenster 19
10 Gemessene Lichtwerte an der Wand 19
11 Er-Datenmodell 20
12 Android App 21

1 Einleitung

Augmented Reality [Azu97] ist mittlerweile nicht mehr nur ein abstrakter Forschungsbegriff der Informatik. Durch mobile Plattformen wie iOS und Android stehen Augmented Reality Anwendungen vielen Anwendern zur Verfügung. Die Arten der Augmented Reality Anwendungen reicht von Spielen bis hin zu Reiseführern. Die Interaktion von virtuellen mit realen Objekten stellt dabei interessante Anwendungsmöglichkeiten dar. Eine besondere Herausforderung wenn es um Augmented Reality geht ist das Erkennen von Objekten. Die Erkennung kann dabei durch einfache Marker oder durch beliebige Bilder realisiert werden. Dabei kommen unterschiedliche Verfahren zum Einsatz. Das Erkennen von Objekten stellt dabei eine besondere Herausforderung dar. Bei der Objekterkennung ist es wichtig zu unterscheiden wie genau ein Objekt erkannt werden soll. Eine wichtige Frage ist auch wie genau ein Objekt erkannt werden soll. Ein Objekt kann entweder allgemein oder sehr speziell betrachtet werden. Ist das Objekt ein Auto, eine Flasche oder ein Buch. Hierfür werden Objekte nach bestimmten Mustern Klassifiziert. Möchte man hingegen genau das eine Auto oder dieses besondere Buch identifizieren sind dafür weitere Verfahren notwendig. In diesem Forschungsprojekt soll gezeigt werden wie Objekte visuell erkannt werden sollen. Dazu soll im ersten Teil dieser Arbeit auf Verfahren wie das vergleichen von Histogrammen bis hin zum Erkennen von besonderen Merkmalspunkten in einem Bild eingegangen werden. In dieser Arbeit soll gezeigt werden wie eine visuelle Erkennung umgesetzt und welche Vor- und Nachteile die Verfahren haben. Im zweiten Teil soll gezeigt werden wie Objekte elektronisch erkannt werden können. Dazu soll Untersucht werden inwieweit Funktechnologien wie W-LAN, Bluetooth oder NFC dabei helfen können. Diese setzen eine vorhandene Infrastruktur voraus. Im letzten Teil soll untersucht werden wie mit Hilfe von Sensoren, ohne eine vorhandene Infrastruktur, Objekte erkannt werden können.

2 Verwandte Arbeiten

Das Erkennen von Objekten mit Hilfe von WiFi, Bluetooth oder anderen Sensoren lässt sich auf bekannte Probleme wie die Lokalisierung in einem Raum zurückführen. Möchte man wissen wo sich ein Objekt befindet muss man wissen wo man selber ist. Gleichzeitig wenn man weiß welche Objekte in meiner Umgebung sind, dann man daraus auch auf die eigene Position schließen. Für die Lokalisierung und das Tracking sowohl innerhalb und ausserhalb von Gebäuden wurden verschiedene Lösungsansätze erarbeitet. So beschreiben Bahl und Padmanabhan[BP00] ein System das die Position eines Benutzer innerhalb eines Gebäudes durch Auswertung der Signalstärke (signal strength information) mehrerer Quellen mittels Triangulation bestimmt. Einen Ähnlichen Ansatz für WLAN Networks haben Ganu et al.[GKK04] gewählt. Sie messen die Signalstärke der WLAN Accespoints um die Benutzer innerhalb von Gebäuden zu lokalisieren. Kotanen et al. haben in [KHLH03] versucht mittels Bluetooth und einem erweiterten Kalman filter eine Position zu bestimmen indem Sie die Received Signal Strength Indeicator (RSSI) auswerten. Ein anderer Ansatz für Lokalisierung ist das LANDMARC System[NLLP04]. Dabei werden RFID Tags gleichmäßig im Raum verteilt. Durch Triangulation der Signalestärke kann dann eine relative Position zu den jeweiligen RFID Tags bestimmt werden. Eine allgemeine Zusammenfassung über Konzepte der Lokalisierung haben Hightower und Borriello[HB01] zusammengestellt. Für das Erkennen von Objekten in Bildern wurden verschiedene Verfahren entwickelt. Die bekanntesten sind SIFT und SURF[BTG06]. Sie basieren auf dem von Lowe entwickelten Verfahren der Merkmalsextraktion in Bildern[Low99, Low03]. Eine gute Übersicht über die verschiedenen Feature Detektoren geben Tuytelaars und Mikolajczyk in [TM08].

3 Visuelle Objekterkennung

In diesem Kapitel soll analysiert werden wie Objekte visuelle erkannt werden können. Dazu sollen Bildanalyseverfahren wie Histogramme und das Feature Tracking untersucht werden.

3.1 Histogramme

Die einfachste Herangehensweise um zwei digitale Bilder miteinander zu vergleichen ist das Vergleichen ihrer Histogramme[Lag11, BK08]. Ein digitales Bild besteht aus einer bestimmten Anzahl an Pixeln. Jedes Pixel hat dabei einen eindeutigen Wert. Ein Histogram ist beschreibt die Verteilung der Pixel im einem digitalen Bild. Ein monochromes 8-Bit Bild kann 256 verschiedene Werte annehmen. Das Histogramm eines Bildes mit 8-Bit besteht somit aus 256 Einträgen (auch Bins genannt). Bin 0 gibt die Anzahl der Pixel mit dem Wert 0 an, Bin 1 die Anzahl der Pixel mit dem Wert 1 und so weiter. Die Anzahl der Pixel eines Bildes ist dabei gleich der Summe aller Einträge in einem Histogramm. So ist es möglich digitale Bilder gleicher Größe zu vergleichen. Möchte man Bilder unterschiedlicher Größe vergleichen muss das Histogramm vorher normalisiert werden. Dabei wird der Wert jedes Bins durch die Anzahl aller Pixel eines Bildes dividiert. Somit gibt jedes Bin nicht mehr die gesamte sondern die relative Verteilung in einem Bild an.

Berechnung eines Histogramms

Das Berechnen eines Histogramm mit OpenCV wird mit der Methode `cv::calcHist` ausgeführt. Die Methode bekommt eine Referenz auf das Bild sowie verschiedene Parameter mit. Das erzeugte Histogramm wird dann in einer Instanz der Klasse `cv::MatND` gespeichert. Diese Klasse wird verwendete um n-dimensionale Matrizen zu bearbeiten. Die Größe des erzeugten Histogramms hängt davon ab ob das Bild lediglich aus Grauwerten oder aus Farben besteht. So liegt die Anzahl der Bins für Graubilder bei 256 und die für Farbbilder bei 256^3, was in der Summe etwas mehr als 16 Millionen Bins macht. Das die Berechnung auf so großen Matrizen wesentlich länger dauert sollte klar sein. Sehr wahrscheinlich werden viele der 16 Millionene Bins gar nicht mit Werten gefüllt sein. Daher ist es angebracht entweder die Anzahl der Farben vor der Berechnung des Histogramms zu reduzieren, oder die Klasse `cv::SparseMat`[1] zu verwenden, welche nur Werte ungleich Null speichert.

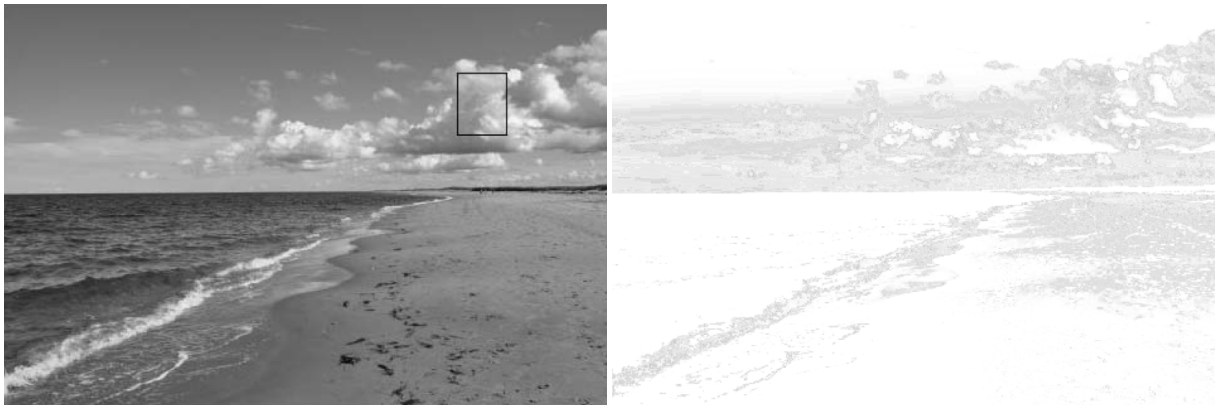
[1] Sparse engl. für dünn besetzt.

Backprojection: Erkennung bestimmter Inhalte in einem Bild

Ein Histogramm beschreibt wichtige Merkmale über den Inhalt eines Bildes. Betrachtet man eine bestimmte Region in einem Bild, so kann das Histogramm dieser Region als eine Funktion betrachtet werden, welche die Wahrscheinlichkeit angibt, das ein bestimmtes Pixel zu dieser Region gehört oder nicht. Angenommen es gibt ein Bild in dem ein bestimmter Inhalt wiedererkannt werden soll. Zunächst wird der Bereich (region of interest) ausgewählt die den zu erkennenden Inhalt enthält (siehe Abbildung 1(a)).

```
1 cv::Mat imageROI;
imageROI = image(cv::Rect(360,55,40,50));
```

Anschließend wird das Histogramm des ausgewählten Bereiches berechnet. Danach wird das Histogramm normalisiert so das wir eine Funktion erhalten die die Wahrscheinlichkeit bestimmt, für das ein Pixel zu der ausgewählten Region gehört. Bei der Backprojection geht es darum die Pixel aus einem Input-Bild durch ihren entsprechenden Wahrscheinlichkteiswert aus dem normalisierten Histogramm zu ersetzen. Das Ergebnis ist ein Bild das die Wahrscheinlichkeitswerte für jedes Pixel von hell (geringe Wahrscheinlichkeit) bis dunkel (hohe Wahrscheinlichkeit) wiedergibt ob es zu der ausgewählten Region gehört.



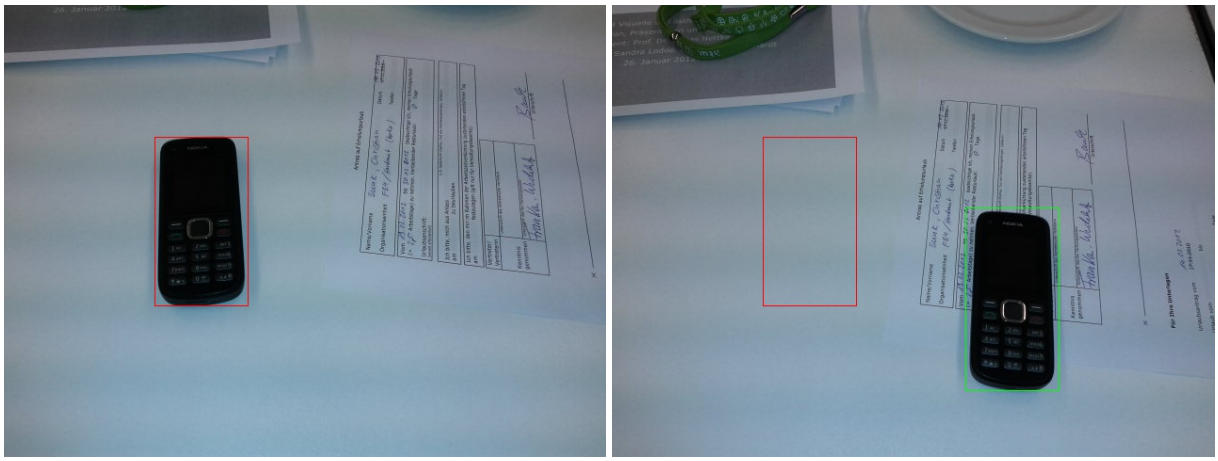
(a) Originalbild mit region of interest (b) Wahrscheinlichkeitsverteilung

Abbildung 1: Backprojection (Bildquelle: [Lag11])

Erkennen von Objekten - Mean Shift

Die Backprojection ermöglicht es mit Hilfe einer Wahrscheinlichkeitsfunktion bestimmte Bereiche in einem Bild wieder zu finden. Mit Hilfe der der Wahrscheinlichkeitsfunktion ist es sogar möglich die Position eines Objektes in einem Bild exakt zu bestimmen. Dazu wird ein Histogramm einer bestimmten Region berechnet (siehe Abbildung 2(a)). Diese Region muss manuell definiert werden. Anschließend wird in einem anderen Bild nach einem Bereich gesucht der mit dem zuvor berechneten Histogramm übereinstimmt. Der

Mean Shift Algorithmus startet dabei an einer Position und läuft dann iterativ durch das Bild. Die Region welche den höchsten Wahrscheinlichkeitswert annimmt gehört dann mit großer Wahrscheinlichkeit auch zu dem gesuchtem Objekt (siehe Abbildung 2(b)).



(a) Original (b) Ausgabe

Abbildung 2: Objekterkennung - Mean Shift

Erkennen gleicher Bilder

Mit Hilfe von Histogrammen können natürlich auch ganze Bilder auf ihre Ähnlichkeit hin untersucht werden. Die Ähnlichkeit zwei Bilder wird dabei durch Vergleichen ihrer Histogramme ermittelt. Eine Implementierung zum Vergleich zweier Histogramme bietet die Methode `cv::compareHist`. Die Methode gibt einen Score zurück welcher die Ähnlichkeit zweier Histogramme beschreibt. Auf diese Weise können mehrere Bilder mit einem Referenz-Bild verglichen werden. Um hier gute Ergebnisse zu erhalten ist es wichtig die Anzahl der Farben zu reduzieren. Es stehen verschiedene Verfahren für das Vergleichen zur Verfügung. Das Intersection Verfahren vergleicht für jedes Bin die zwei Werte aus dem Histogram und speichert den minimalsten. Aus der Summe der minimalsten Werte ergibt sich dann der Score Wert.

$$d_{intersection}(H_1, H2) = \sum_i min(H_1(i), H_2(i)) \tag{3.1}$$

Dieses Verfahren wird Intersect genannt. Andere Verfahren sind Chi-Quadrat oder Bhattacharyya. Das Verfahren hat den Nachteil das keine speziellen Objekte in einem Bild erkannt werden können. Es können eben nur zwei Bilder auf ihre Ähnlichkeit untersucht werden.

Fazit

Es wurde gezeigt das mit Hilfe von Histogrammen Objekte in einem Bild erkannt werden können. Ein Problem ist jedoch das Histogramme nicht robust gegenüber Veränderungen in der Beleuchtung sind. Wird das Bild heller oder dunkler hat dies große Auswirkungen auf die Erkennung. Auch Veränderungen in der Skalierung oder des Blickwinkels führen zu falschen Ergebnissen. Ein weiteres PRoblem ist das ein Histogramm gerade bei Farbbildern sehr groß wird. Diese Daten müssen irgendwo gespeichert werden. Aufgrund der genannten Probleme soll ein weiteres Verfahren analysiert werden. Dieses Verfahren heißt Feature Tracking.

3.2 Feature Tracking

Das erkennen von besonderen Merkmalen (oder auch features, interest points, keypoints) in einem Bild ist ein wichtiges Konzept in der Bildverarbeitung. So muss bei dem Verfahren mit Histogrammen immer das ganze Bild im Speicher sein. Natürlich können die Farben und Informationen in einem Bild reduziert werden, aber auch dann müssen immer noch viele Daten verarbeitet werden. Ein Nachteil der sich mit Histogrammen ergibt ist das die Erkennungsrate mit Veränderungen in der Skalierung und des Betrachtungswinkels abnimmt. Auch haben Veränderungen in der Beleuchtung schlechte Auswirkung auf die Erkennung. Daher soll in diesem Abschnitt gezeigt werden wie es mit Hilfe von OpenCV möglich ist Features in einem Bild zu erkennen zu speichern und zu verarbeiten. Ein Verfahren das Featurepunkte in einem Bild bestimmen kann ist das Speeded Up Robust Features (SURF) Verfahren [BTG06].

Berechnen der Featurepunkte

Bevor zwei Bilder verglichen werden können muss das Bild zuerst in den Speicher geladen werden.

```
// read image from file
cv::Mat image = cv::imread("image.jpg");
```

Anschließend werden auf dem Bild die Featurepunkte (keypoints) berechnet. In Abbildung 3 sind die berechneten Featurepunkte zu sehen.

```
std::vector<cv::KeyPoint> keypoints;   // save the keypoints
cv::SurfFeatureDetector surf(10.0);    // use SURF feature detector
surf.detect(image,keypoints);          // detect keypoints from image
```

Berechnen des Deskriptoren

Nun werden die Deskriptoren anhand der Featurepunkte berechnet.

(a) Keypoints (b) Deskriptoren inkl. Skalierung und Ausrichtung

Abbildung 3: Berechnete Featurepunkte

```
cv::SurfDescriptorExtractor extractor;
cv::Mat descriptors;  // save the descriptors
// compute descriptors from image using the keypoints
extractor.compute(image, keypoints, descriptors);
```

Jeder Deskriptor besitzt eine Ausrichtung (orientation) sowie einen Skalierungsfaktor wie in Abbildung 3(b) zu sehen ist. Der Skalierungsfaktor wird durch die Größe des Kreises dargestellt. Die Ausrichtung wird durch die radiale Linie des Deskriptors abgebildet. Diese Eigenschaften machen den Featurepunkt invariant gegenüber Veränderungen in der Skalierung sowie der Ausrichtung.

Vergleichen der Bilder

Bevor zwei Bilder miteinander vergleichen werden können müssen noch die Featurepunkte und Deskriptoren des zweiten Bildes berechnet werden. Anschließend können die beiden Bilder mit Hilfe des `cv::BruteForceMatcher` verglichen werden. Dabei wird jeder Deskriptor von Bild A mit den allen Deskriptoren von Bild B verglichen. Das Paar mit der kürzesten Distanz ist dann der passendste Match. Das Ergebnis ist in Abbildung 4 zu sehen.

```
std::vector<cv::DMatch> matches; // store matches
cv::BruteForceMatcher<cv::L2<float> > matcher;
// compute matches
matcher.match(descriptors_image1, descriptors_image2, matches);
```

Hier ist deutlich zu erkennen das einige Featurepunkte aus dem Original Bild nicht korrekt abgebildet werden.

Abbildung 4: Matches der Featurepunkte (links Originalbild, rechts Vergleichsbild)

Robust Matcher

Um die Anzahl der falsch zugeordneten Matches zu verringern wurde ein Robust Matcher [Lag11] implementiert. Bei diesem Verfahren wird nicht wie zuvor nur ein passender Match zu jedem Featurepunkt gesucht sondern nach den zwei besten Matches. Das geschieht in beide Richtungen. Aus Bild A werden zu jedem Punkt die zwei besten Matches aus Bild B gesucht. Umgekehrt wird das selbe für Bild B gemacht. Nun werden die zwei besten Matchen verglichen.

```
1 matcher.knnMatch(descriptors_image1, descriptors_image2,
      matches, // vector of matches (up to 2 per entry)
3     2); // return 2 nearest neighbours
```

So haben wir für jeden Featurepunkt je zwei Kandidaten aus dem jeweils anderen Bild. Ist die Distanz der des besten Matches gering und die für den zweitbesten Match sehr viel größer, kann mit Sicherheit angenommen werden das der beste Macht ein guter Kandidat ist. Ist hingegen der Abstand vom besten Match zum zweitbesten Match sehr gering ist die Wahrscheinlichkeit eines groß das der falsche Featurepunkt ausgewählt wird. In diesem Fall werden die beiden Matches verworfen. Mit Hilfe dieses Verfahrens kann die Anzahl falscher Matches reduziert werden (siehe Abbildung 5).

Featurepunkte in Datei speichern

Das Berechnen der Featurepunkte und der Deskriptoren in einem Bild ist ein sehr aufwendiger Vorgang der viel Zeit in Anspruch nimmt (siehe Abbildung 6). Da jedoch das Ergebnis der Berechnung bei gleichen Parametern stets gleich ist sollte man darüber nachdenken ob die Berechnung nur einmal gemacht werden muss. So ist es sicher nicht sinnvoll jedes mal für alle Bilder in der Datenbank die Featurepunkt und Deskriptoren

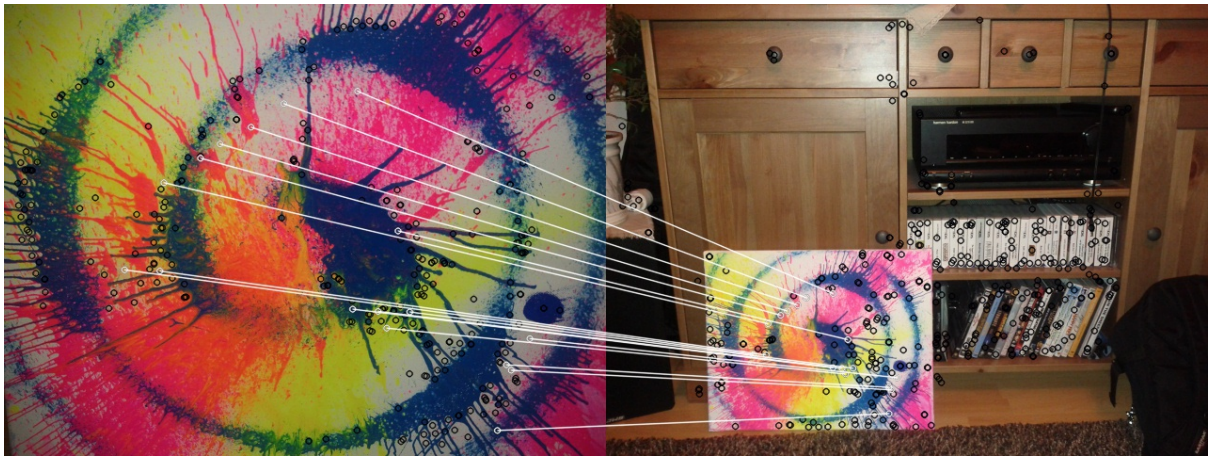


Abbildung 5: Weiß gute Matches, Schwarz verworfene Matches

neu zu berechnen. Eine Möglichkeit wäre es die Featurepunkte nur einmal zu berechnen und dann in geeigneter Form zu speichern um bei Bedarf schnell darauf zugreifen zu können. Mit der Klasse `cv::FileStorage` bietet OpenCV eine API um die berechneten Deskriptoren in eine Datei zu speichern. Diese können entweder in XML oder YAML formatiert werden. In eine Datei können dabei Daten von beliebig vielen Bildern gespeichert werden. Die Deskriptoren zu jedem Bild bekommen dafür eine eindeutige Bezeichnung. Das Auslesen erfolgt dann einfach über die Bezeichnung des gewünschten Bildes. Der folgende Codeausschnitt zeigt wie die berechneten Deskriptoren von zwei Bildern in eine xml Datei geschrieben werden.

```
cv::FileStorage fs("Features.xml", FileStorage::WRITE);
fs << "image1" << descriptors_image1;
fs << "image2" << descriptors_image2;
fs.release();
```

Beim Auslesen aus der Datei werden die Features zu jedem Bild in einem Array gespeichert. Nun kann über den eindeutigen Schlüssel auf das jeweilige Element zugegriffen werden.

```
cv::FileStorage fs("Features.xml", FileStorage::READ);
fs["image1"] >> descriptors_image1;
fs["image2"] >> descriptors_image2;
fs.release();
```

Auf diese Weise kann man einfach in einer Datenbank die Metadaten zu den Bildern und den Deskriptoren speichern, die dann aus der Datei ausgelesen werden können. In Abbildung 6 ist zu erkennen das die Berechnung der Featurepunkte 0.7 Sekunden und für die Deskriptoren 0.4 Sekunden benötigt werden. Insgesamt braucht die Berechnung ca. 1.1 Sekunden. Diese Zeit hängt natürlich von der Größe des Bildes, der Farben sowie der Anzahl der zu berechnenden Featurepunkte ab. Für dieses Beispiel wurden 4368

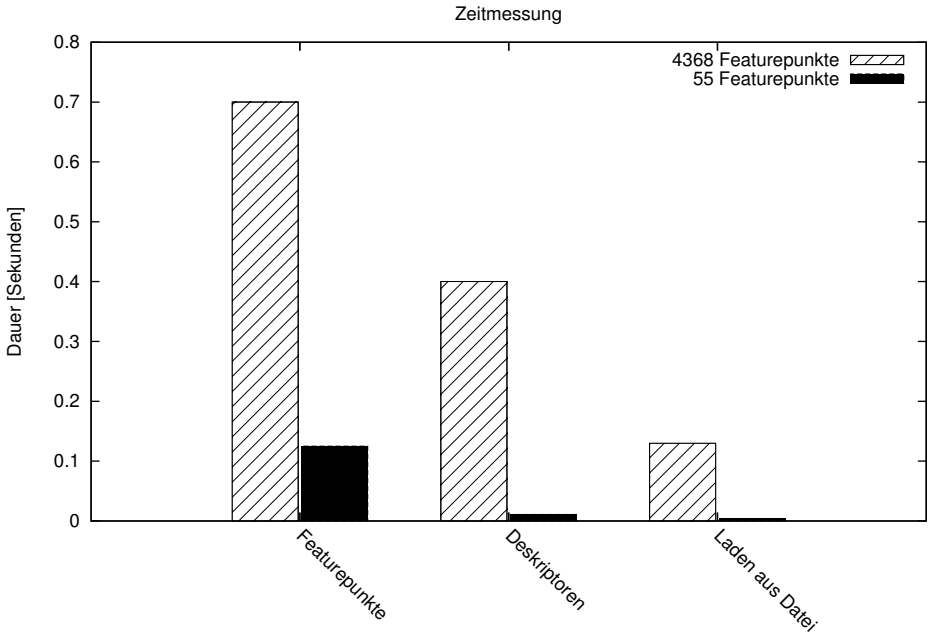


Abbildung 6: Berechnungszeiten

Featurepunkte berechnet. In der Regel reichen auch weniger Featurepunkte aus. Das Laden der Deskriptoren aus einer Datei ist mit 0.13 Sekunden dagegen deutlich schneller. Für 55 Featurepunkte dauert das Berechnen der Featurepunkte 0.125 Sekunden und 0.011 Sekunden für die Deskriptoren. Das Laden aus der Datei dauert nur 0.004 Sekunden. Hier ist deutlich zu sehen das das Laden der berechneten Deskriptoren aus der Datei einen deutlichen Geschwindigkeitsvorteil bringt als diese neu zu berechnen. Es macht also durchaus Sinn die Featurepunkte und Deskriptoren nur einmal zu berechnen und sie bei bedarf aus einer Datei zu laden. Noch schneller wäre es wenn die Featurepunkte der Objekte schon in den Speicher geladen werden wenn der Benutzer z.B. den Raum mit den Objekten betritt. Dazu müssen die Objekte jedoch elektronisch erkannt werden. Dies wird in Kapitel 4.2 und 4.3 besprochen.

Fazit

Mit Hilfe von Featurepunkten ist es möglich ein Objekt in einem Bild zu erkennen. Auch wurde gezeigt das die Qualität der Matches durch einen Robust Matcher sehr verbessert werden kann. So werden die meisten falschen Matches verworfen. Zwar braucht die Berechnung der Featurepunkte und der Deskriptoren viel Rechenzeit. Jedoch müssen die Features nur einmal berechnet werden. Das Ergebnis kann anschließend in eine Datei gespeichert werden. Der Zugriff auf die Features ist somit sehr viel schneller möglich. Hat man jetzt noch das Wissen welche Objekte sich in der unmittelbaren Umgebung befinden ist es möglich die Features in den Arbeitsspeicher zu laden noch bevor der Benutzer sein

Smartphone auf das Objekt hält. Tut er das kann sehr schnell mit den geladenen Features im Speicher verglichen werden.

3.3 Texterkennung

Bestimmte Objekte können die Identifizierung erschweren. Möchte man z.B. zwei Bücher (siehe Abbildung 7) mit gleichem Cover aber unterschiedlichem Titel erkennen reichen die bisher beschriebenen Verfahren meist nicht aus. So wurden die Bilder 7(a) und 7(b) mit Hilfe von Histogrammen auf ihre Ähnlichkeit untersucht. Der Score der Histogramme beider Buchcover ergab 0.996968. Die Buchcover haben also eine sehr hohe Ähnlichkeit. Die beiden Buchcover können mit Histogrammen also nur schwer unterschieden werden. Hier müssen also andere Verfahren zum Einsatz kommen. Eine Möglichkeit dabei wäre die Texterkennung (Object Character Recognition) zu nutzen. Damit könnten Buchtitel extrahiert werden. Dieser Ansatz wurde in dieser Arbeit jedoch nicht näher untersucht.

(a) (b)

Abbildung 7: Bücher mit gleichem Cover

4 Elektronische Objekterkennung

Die elektronische Objekterkennung soll zeigen wie mit Hilfe von

- vorhandenen Infrastrukturen wie WLan, Bluetooth oder NFC,
- oder mit Hilfe von Sensoren wie Licht

Objekte in der Umgebung erkannt werden können. Eine Herausforderung bei der Objekterkennung ist die Loaklisierung des Benutzers. Wenn die genaue Position des Benutzers und der Objekte bekannt ist kann eine Aussage darüber gemacht werden welche Objekte sich in der näheren Umgebung befinden. Dieses Kapitel soll zeigen mit welchen Verfahren eine genau Position ermittelt werden. Die Lokalisierung kann dabei in zwei Stufen erfolgen: grob und fein. Eine grobe Lokalisierung bedeutet das es schon reicht wenn man weiß in welchem Raum eines Gebäudes sich der Benutzer befindet. So kann eine allgemeine Aussage gemacht werden welche Objekte sich in diesem Raum befinden. Eine feine Lokalisierung bedeutet das die genaue Position des Benutzers in diesem Raum bekannt ist. So ist es möglich zu sagen das der Benutzer nahe an Objekt 1 ist und weiter weg von Objekt 2.

4.1 Verfahren zur Lokalisierung

Zur Lokalisierung gibt es bereits verschiedene Messverfahren. Die bekanntesten sollen im folgenden vorgestellt werden.

4.1.1 Lateration

Lateration oder Trilateration[5] ist ein Messverfahren zur Bestimmung eines Punktes in einer Ebene. Die Lateration verwendet die Entfernungsmessung zu bekannten Punkten. Ist die Entfernung r zu einem Punkt bekannt so liegt die eigene Position auf der Kreisfläche mit dem Radius r um diesen Punkt. Ist die Entfernung zu zwei Punkten $(P1, P2)$ bekannt so liegt der Standort auf den Schnittpunkten der beiden Kreisflächen (siehe Abbildung 8, Schnittpunkte A und B). Erst mit einem dritten bekannten Punkt kann die genaue Position ermittelt werden (Schnittpunkt B). Die Trilateration stellt die Grundlage der Entfernungsmessung für die Satellitennavigation dar.

4.1.2 Triangulation

Bei der Triangulation misst man die Winkal α und β von zwei verschiedenen Standorten 1 und 2 zu einem Bezugspunkt P. Ist die Basislänge b der beiden Standorte 1 und 2 bekannt, kann mit Hilfe der Winkel die Entfernung zum Bezugspunkt berechnet werden.

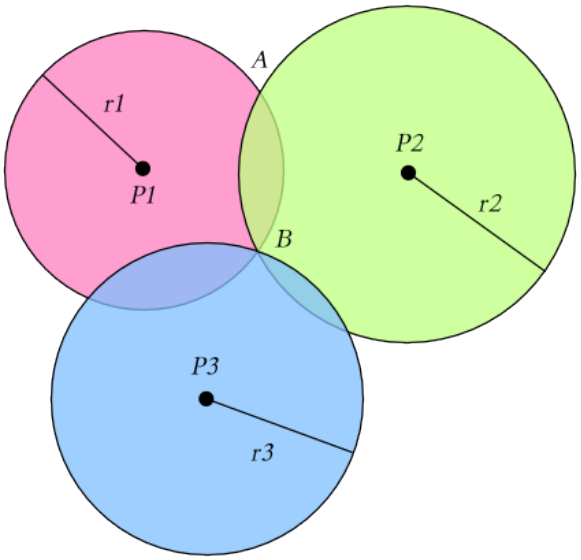


Abbildung 8: Trilateration. Quelle: [5]

4.1.3 Proximity

Beim Proximity Verfahren werden mehrere Sender/Empfänger (meist RFID Tags) verteilt. Da die Position der Tags bekannt ist, kann die Position des zu ortenden Objektes durch Signalstärken zu den Empfangsantennen der Tags berechnet werden. Solche Zellenbasierte Systeme sind z.B. GSM, UMTS oder auch Landmark[NLLP04].

4.1.4 Dead Reckoning

Sind Anfangskoordinaten des zu ortenden Objekts bekannt, so kann, wenn die Geschwindigkeit und die Bewegungsrichtung bekannt ist, eine Position zu einem bestimmten Zeitpunkt ermittelt werden.

4.1.5 Received Signal Strength Indicator (RSSI)

Eine Möglichkeit die Entfernung zu einem entfernten Punkt zu bestimmen, ist die empfangene Signalstärke eines Senders zu messen. Mehrere Arbeiten haben sich mit diesem Thema beschäftigt. So wurden in [MN, Tad] eine Positionierung mittels RSSI in ZigBee Infrastrukturen untersucht. Aber auch in wireless networks kann mittels RSSI eine Position ermittelt werden wie in [SGJ] gezeigt wird. Weitere Arbeiten zu RSSI finden sich in [FVE, HLIN].

4.1.6 Einflüsse auf die Signalstärke

Es gilt zu beachten das die Signalstärke von verschiedenen Faktoren beeinflusst werden kann. So kann schon das vorhandenen sein einer Person die Signalstärke beeinträchtigen, da diese am Menschlichen Körper (durch den hohen Wasseranteil) abgelenkt werden. Auch die Struktur und das Material der Wände eine Gebäudes haben Einfluss auf die Signale. So können Signale durch Reflexion oder Absorption beeinflusst werden. Da Signalstärke umgekehrt proportional zum Quadrat der Entfernung ist müsste diese dementsprechend mit höherer Entfernung schwächer werden. Dies ist jedoch nur unter idealen Bedingungen. Durch Reflexion innerhalb von Gebäuden kann sogar zu höheren Signalstärken kommen wenn der Abstand zum Sender größer wird, bzw. geringerer Signalstärke wenn der Abstand kleiner wird. Diese Fehler müssen bei Verfahren mit Signalstärken unbedingt berücksichtigt werden.

4.1.7 Time Difference of Arrival (TDoA)

Eine Entfernung zwischen zwei Knoten kann auch anhand der Signallaufzeit berechnet werden. Ist die Geschwindigkeit des Signals bekannt, kann mittels der Zeit die ein Signal zu einem Knoten und wieder zurück benötigt, die zurückgelegte Entfernung berechnet werden. Könnte man die Zeit mittels eines Smartphones bestimmen die ein elektromagnetisches Signal einer WLan oder Bluetooth Quelle benötigt so kann man die Entfernung berechnen. Möchte man die Entfernung mittels Signallaufzeit berechnen und eine Genauigkeit von unter $1m$ erreichen, bei rund $300000\frac{km}{s}$, sind Uhren notwendig die auf eine Nanosekunde (10^{-9} Sekunden) genau den Takt angeben. Leider sind Uhren mit so hoher Genauigkeit sehr teuer und werden nicht in Smartphones eingebaut. So bietet das Samsung Nexus S gerade einmal eine Genauigkeit von 1 Millisekunde (10^{-3} Sekunden). Dies würde ausreichen um Entfernung mit einer Genauigkeit von 300 km zu berechnen. Aufgrund der Ungenauigkeit der Uhren kann also keine Entfernung auf Basis der Signallaufzeit elektromagnetischer Signale erfolgen. Eine Alternative wäre aber die Verwendung von wesentlich langsameren Signalen wie (Ultra-) Schallwellen. So könnte bei einer Signallaufzeit von rund $300\frac{m}{s}$ einer Schallwelle, die Genauigkeit der berechneten Entfernung bis $0.3m$ liegen. Dieser Wert wäre sehr gut geeignet um eine Entfernung selbst auf kurze Distanzen berechnen zu können.

4.1.8 Angle of Arrival (AoA)

Während das TDoA Verfahren die Zeit misst die ein Signal benötigt, wird beim AoA Verfahren der Winkel gemessen mit der ein Signal beim Empfänger ankommt. Dies erfordert aber sehr gute Antennen.

Fazit

Es wurden verschiedene Verfahren gezeigt wie eine Entfernung zu anderen Knoten bestimmt werden kann. Dabei hat sich herausgestellt das die Laufzeitmessungen oder Winkelmessungen in mobilen Geräten mangels genauer Hardware nicht umsetzbar ist. Andere Verfahren wie Proximity sind da schon vielversprechender. Hier wird jedoch eine vorhandene Infrastruktur benötigt. Am aussichtsreichsten ist das Verfahren der Received Signal Strength um in Funknetzwerken (WLan oder Bluetooth) eine Lokalisierung vorzunehmen.

4.2 Objekterkennung durch Funkstandards

Besteht eine feste Funknetzwerk Infrastruktur kann mittels der Signalstärke eine Position bestimmt werden (siehe Abschnitt 4.1.5). Im weiteren Verlauf dieser Arbeit soll es jedoch genügen lediglich zu ermitteln in welchem Raum, innerhalb eines Gebäudes, man sich befindet. Dazu sollte jeder Raum mit einem Accesspoint (WLan oder Bluetooth) ausgestattet werden. Im folgenden sollen die Implementierungen vorgestellt werden, die zeigen wie mittels WLan und Bluetooth Objekte identifiziert werden können.

4.2.1 WLan

Der Zugriff auf die WLan Funktionen ist über die Klasse `android.net.wifi.WifiManager` möglich. Mit Hilfe der Methode `startScan()` kann eine Suche nach umliegenden WLan Acces Points gestartet werden. Anschließend werden die gefundenen Netzwerke in der Klasse `android.net.wifi.ScanResult` in einer Liste zurückgegeben. Die wichtigsten Parameter sind dabei:

- Basic Service Set Identifier (BSSID)
- SSID
- RSSI

Die BSSID ist eine eindeutige Bezeichnung eines Access Points und ist damit vergleichbar mit einer MAC-Adresse. Die ermittelte RSSI gibt die Signalstärke beim Empfänger an. Ist die Signalstärke des Senders bekannt, könnte mit diesem Wert eine Entfernung zum Sender berechnet werden. Es wurde ein Service entwickelt welcher nach Aktivierung automatisch nach WLan Acces Points sucht.

4.2.2 Bluetooth

Da Bluetooth Tags wesentlich kleiner sind und weniger Energie Verbrauchen als WLan Access Points, könnten die Tags auch direkt in der Nähe der Objekte platziert werden.

Gerätesuche

Die Suche nach anderen Bluetooth Geräten ist ein Prozess bei dem die lokale Umgebung nach eingeschalteten Bluetooth Geräten abgesucht wird. Anschließend können Informationen über jedes gefunden Gerät abgerufen werden. Nur wenn ein Bluetooth Gerät auch gefunden werden will, wird es auf die Suche antworten und Informationen wie den Namen, die Klasse und die eindeutige MAC Adresse mitteilen. Es wurde ein Service welcher periodisch nach Bluetooth Geräten in der Umgebung sucht. Dazu wurde die Klasse **android.bluetooth.BluetoothAdapter** verwendet. Da sich ein Benutzer ständig fortbewegt ist es nötig die Suche in bestimmten Abständen zu wiederholen. Hier ist jedoch zu beachten das die Gerätesuche sehr viel Energie kostet. Eine ständige Suche nach anderen Bluetooth Geräten ist daher nicht sinnvoll. Besser wäre wenn die Periode der Suche an das Laufverhalten des Benutzer angepasst wird. Bewegt sich ein Benutzer sehr schnell vorwärts verändert sich auch seine Umgebung sehr viel schneller. Dann muss auch das Intervall der Gerätesuche kleiner werden. Bewegt er sich dagegen kaum kann die Gerätesuche in größeren Intervallen gemacht werden um Ressourcen zu sparen. Folgende Parameter werden während der Suche gesammelt.

- Name des Gerätes
- Adresse des Gerätes
- RSSI

In einer Datenbank kann nun gesucht werden, welche Räume oder Objekte mit den gefunden Bluetooth Adressen übereinstimmen. So ist es möglich Informationen über umliegende Objekte zu bekommen.

4.2.3 Near Field Communication

Near Field Communication (NFC)[9] ist ein Übertragungsstandard welcher einen kontaktlosen Austausch von Daten ermöglicht. Die Kommunikation kann entweder von einem aktiven NFC Gerät zu einem passiven NFC Tag oder auch zwischen zwei aktiven NFC Geräten erfolgen. Ein passives NFC Tag bezieht seine Energie über Induktion von dem aktiven Transponder.

Um Objekte mit NFC zu identifizieren müssen diese mit einem NFC Tag ausgestattet sein. Aufgrund der geringen Kommunikationsreichweite muss der Benutzer wissen wo sich das NFC Tag befindet. Denken wir nun an das Szenario im Museum, müssten die entsprechenden Objekte mit einem NFC Tag ausgestattet werden. Da diese Tags aber sichtbar sein müssen würde sich hier die Frage stellen warum überhaupt NFC und nicht andere Verfahren wie Barcodes oder QR-Tags. Ein Vorteil von NFC ist das die Tags

versteckt werden können. So könnte man sie hinter Informationstafeln, die sich an den meisten Objekten befinden, verstecken. Durch ein Symbol könnte der Benutzer dann unauffällig darauf hingewiesen werden das er sein mobiles Gerät auf diese Fläche legen kann um zusätzliche Informationen zu bekommen.

Auswertung

Mit Hilfe der oben genannten Funkstandards ist eine Identifizierung von Objekten sehr gut möglich. Da sowohl WLan, Bluetooth und NFC eine eindeutige ID aussenden ist es möglich durch Anwendung dieser Verfahren Objekte zu identifizieren. Zu den Räumen bzw. Objekten werden die empfangbaren WLan und Bluetooth Acces Points gespeichert. Ein Objekt kann natürlich in der Reichweite mehrerer Bluetooth oder WLan Netze liegen. Das ist auch sinnvoll, denn fällt ein Acces Point aus kann ein Objekt immer noch durch andere Access Points identifiziert werden. Schwierig wird es wenn mehrere Objekte in der Reichweite der selben Netzwerke sind. Dann könnte die Verwendung der Signalstärke hilfreich sein um eine Abschätzung zu machen welches Objekt am nächsten ist.

4.3 Weitere Sensoren

In diesem Abschnitt soll für weitere Sensoren untersucht werden ob diese sich zur Lokalisierung und Objekterkennung eignen.

4.3.1 GPS

Mittels GPS ist eine Positionierung im freien sehr gut möglich. Werden Objekte mit einer GPS-Position gespeichert kann berechnet werden welche Objekte sich in der Nähe befinden. Anhand der Genauigkeit der GPS Position kann entschieden werden ob der GPS Sensor höhere Priorität bekommt gegenüber anderen Sensoren.

4.3.2 Beschleunigung

Mit Hilfe des Beschleunigungssensors ist es möglich zu bestimmen ob sich der Benutzer eines mobiles Gerätes bewegt. Diese Information kann genutzt werden um die Häufigkeit der Suche nach Access Points zu regulieren. Je schneller sich ein Benutzer bewegt desto häufiger sollte nach Access Points gesucht werden, da sich der Benutzer sehr wahrscheinlich an einem anderen Ort befindet. Bewegt er sich dagegen nur langsam dann muss auch nicht so oft nach Acces Points gesucht werden. Dies kann besonders bei Bluetooth viel Energiesparen. Mit Hilfe des Beschleunigungssensors kann die Laufgeschwindigkeit eines Benutzers ermittelt werden. Mit Hilfe eines Kompass kann bestimmt werden in welcher Richtung sich der Benutzer bewegt (siehe Abschnitt 4.1.4).

4.3.3 Proximity

Das Samsung Nexus S stellt einen Proximity Sensor bereit. Dieser befindet sich auf der Seite des Displays und ist dem Benutzer zugewandt. Er dient in der Regel dafür während eines Telefonats wenn sich das Telefon am Ohr des Benutzers befindet das Display auszuschalten um Strom zu sparen. Mit dem Proximity Sensore wäre es möglich Entfernungen zu messen. Allerdings liefert der Sensor nur binäre Werte für **nah** oder **fern** und ist somit nicht für eine Lokalisierung geeignet.

4.3.4 Licht

Mit Hilfe des Lichtsensors ist es möglich Objekte zu lokalisieren. Der Sensor misst die Stärke des Umgebungslichts. Je nach Lichtstärke kann man bestimmen ob ein Objekt nahe an einem Fenster, wo es tagsüber heller ist, oder nahe an einer Wand steht wo es in der Regel etwas dunkler ist. Das ganze hat allerdings mindestens zwei Nachteile. Das erste Problem ist das die Lichtverhältnisse sich über den Tagesverlauf ändern. Das zweite Problem ist das man davon ausgehen muss das sogar tagsüber durch künstliche Beleuchtung ein homogenes Lichtverhältnis existiert. So kann in einem gleichmäßig beleuchteten Raum wahrscheinlich keine konkrete Aussage mehr über die Position des Objekts im Raum gemacht werden. Dennoch stellt dies einen interessanten Ansatz dar. Mit dem Samsung Nexus S soll eine Analyse der messbaren Lichtwerte gemacht werden. Die Messwerte werden dabei in Lux[8, 2] ausgegeben, welche die Beleuchtungsstärke angibt.

$$lx = \frac{lm}{m^2} \tag{4.1}$$

Dabei gibt lm (Lumen[7], lateinisch für Licht, Leuchte) die Einheit des Lichtstroms an. Diese berücksichtigt die Empfindlichkeit des menschlichen Auges. So werden zwei Lichtquellen als gleich hell wahrgenommen, wenn sie den gleichen Lichtstrom aussenden.

Im folgenden soll die Stärke der Beleuchtung in einem Raum über einen Zeitraum von 24 Stunden gemessen werden. Eine Messung soll dabei am Fenster und eine andere an der gegenüberliegenden Fensterseite gemacht werden. Die Messungen erfolgten im TGS Raum 2.3.03. Es wurde pro Stunde eine Messwert angelegt. Für die Messung wurden 10 Werte in 10 Sekunden aufgenommen und anschließend gemittelt. Die Ergebnisse für die Fensterseite sind in der Abbildung 9 und die für die Wandseite in der Abbildung 10 zu sehen. Es ist sehr gut zu erkennen das die Lichtwerte an der Fensterseite, tagsüber, sehr viel stärker sind als auf der Wandseite. Nachts sind die Werte auf beiden Seiten identisch. So kann mit Hilfe des Lichtsensors zumnidest tagsüber, in nicht künstlich Beleuchteten Räumen, eine Aussage gemacht werden ob man sich gerade an der Fensterseite befindet oder nicht. Natürlich könnten auch die Lichtwerte von bestimmten Objekten gemessen werden. In einem Museum sind manche Objekte in iner Vitrine stark beleuchtet. Andere

Objekte befinden sich in abgedunkelten Räumen. Allein mit dem Lichtsensor ist eine Identifizierung der Objekte nicht möglich, jedoch aber zusammen mit anderen Sensoren.

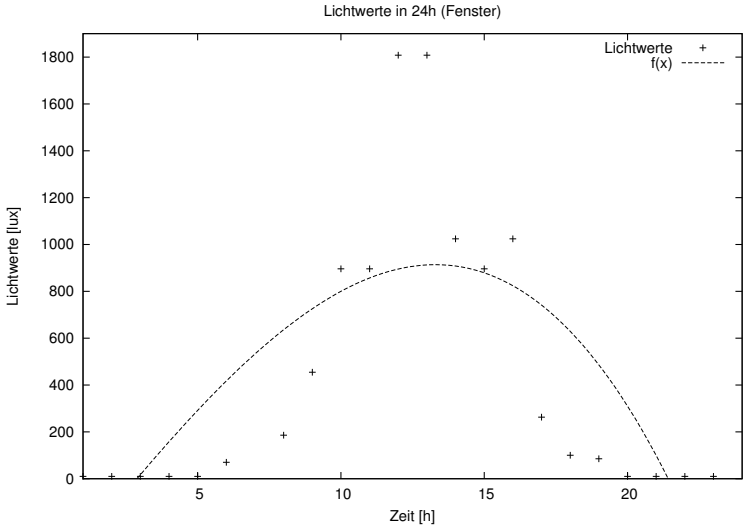


Abbildung 9: Gemessene Lichtwerte am Fenster

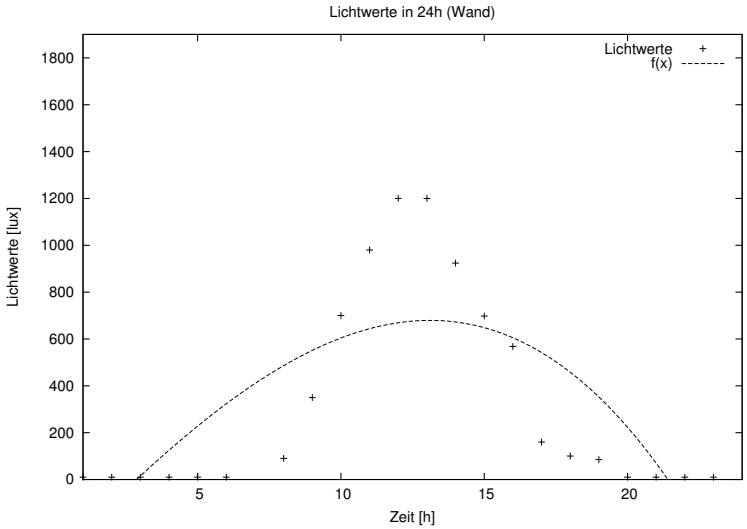


Abbildung 10: Gemessene Lichtwerte an der Wand

5 Implementierung

Es wurde eine Android Anwendung entwickelt, welche die WLan und Bluetooth Netzwerke in der Umgebung sucht. Dafür wurde je ein Service entwickelt welcher periodisch eine neue Suche startet. Das Ergebnis sind zwei Listen mit den aktuellen Bluetooth oder WLan Acces Points die in der Umgebung gefunden wurden. Nun sollen aber Objekte gefunden werden. Dafür wurde eine lokale Datenbank erstellt, welche die Objekte den jeweiligen Bluetooth oder WLan Acces Points zuordnet.

Datenmodell

In Abbildung 11 ist das entworfene Datenmodell zu sehen. Es wurden drei Tabellen definiert welche die Objekte, sowie die Bluetooth und WLan Access Points (im folgenden nur kurz als Access Points bezeichnet) enthalten. Da ein Objekt in der Reichweite mehrerer Access Points sein kann ein Access Point in seiner Reichweite mehrere Objekte enthalten kann, mussten noch zwei Tabellen hinzugefügt werden um die m:n Beziehung zu beschreiben.

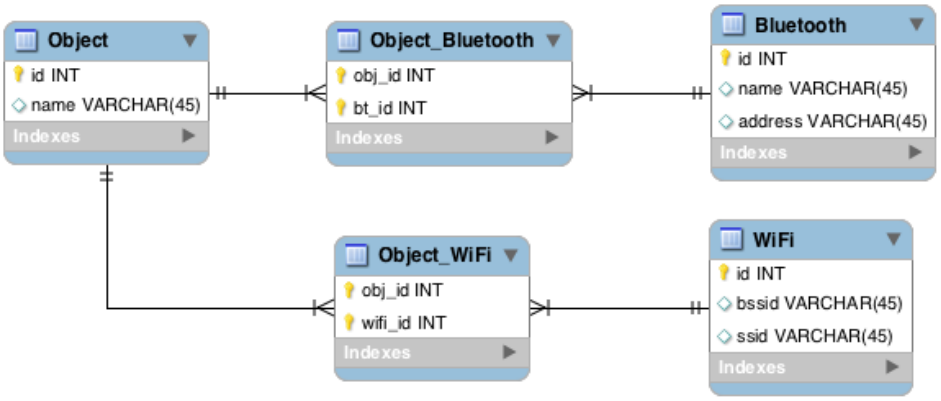


Abbildung 11: Er-Datenmodell

Anwendung

Die Suche nach Access Points wurde für jeden Sensor in einem Service gekapselt. Per Knopfdruck kann der entsprechende Sensor aktiviert werden. Wurde ein Access Point gefunden wird in der Datenbank nachgeschaut ob dieser Access Point dort enthalten ist, wenn ja werden die Objekte die mit diesen Access Point verknüpft sind in einer Liste zurückgegeben. Die Liste mit den gefundenen Objekten wird nach jeder Suche aktualisiert. Abbildung 12 zeigt die View der Anwendung. Die jeweiligen Sensoren können auf Knopfdruck aktiviert werden. Sie können einzeln oder zusammen genutzt werden.

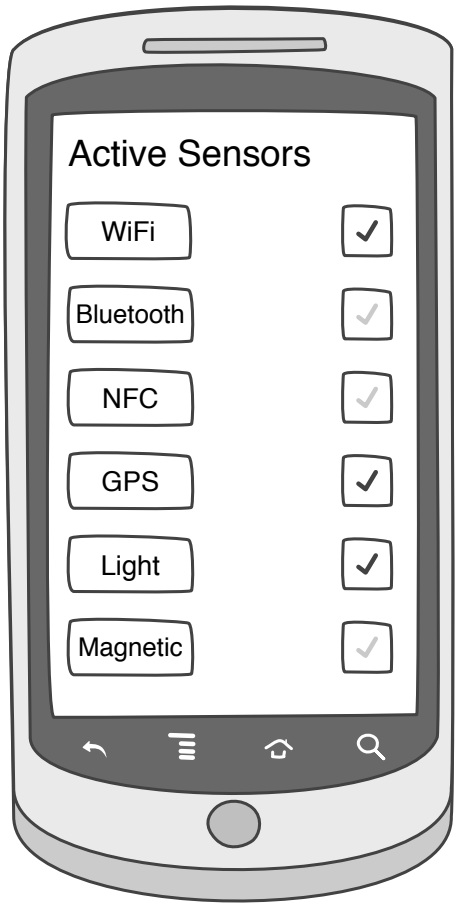


Abbildung 12: Android App

6 Zusammenfassung

In diesem Forschungsprojekt wurden Verfahren zur visuellen und elektronischen Erkennung von Objekten analysiert. Die visuelle Erkennung hat gezeigt das die Objekterkennung mit Histogrammen zwar möglich ist aber nicht robust genug gegen Veränderungen in der Beleuchtung, der Skalierung und dem Blickwinkel ist. Ein besserer Verfahren ist die Berechnung von Featurepunkten in einem Bild. Diese sind stabiler gegenüber den oben genannten Veränderungen. Auch können die Featurepunkte sehr gut serialisiert und bei Bedarf geladen werden. Ein weiteres Ziel war es Objekte elektronisch zu erkennen. Dazu wurden zunächst verschiedene Verfahren vorgestellt wie Lokalisierungen innerhalb von Gebäuden funktioniert. Das aussichtsreichste Verfahren ist dabei die Signalstärke der umliegenden Acces Points auszuwerten. In einer entwickelten Anwendung werden die Objekte den Access Points in ihrer Reichweite zugeordnet. Nun kann nach den empfangbar Acces Points gesucht werden. Wurde ein Access Point gefunden wird geprüft welche Objekte mit diesem verknüpft sind. So kann man Informationen darüber bekommen, welche Objekte in der Nähe sind. Ausserdem wurde untersucht ob der Lichtsensor in einem Smartphone bei der Lokalisierung helfen kann. Dabei kam heraus das unter der Bedingung das keine künstlichen Lichtquellen existieren, tagsüber eine beschränkte Lokalisierung möglich ist. So kann mit dem Lichtsensor eine Aussage gemacht werden ob sich der Benutzer in der Nähe eines Fenster befindet oder nicht.

Literatur

[Azu97] Azuma, Ronald: *A Survey of Augmented Reality.* 1997

[BB11] Björn Bittins, Prof. Dr. Jürgen S.: *Multisensor and Collaborative Localization for Diverse Environments.* 2011

[BGK+11] Bunk, Christian ; Günther, Andreas ; Kluge, Dennis ; Fellner, Steffanie ; Sandrock, Jessica ; Schreiber, Johanna ; Sieck, Prof. Dr. J.: *Augmented Reality children guide for the Museum of Islamic Art.* 2011

[BK08] Bradski, Gary ; Kaehler, Adrian: *Learning OpenCV: Computer Vision with the OpenCV Library.* 1st. O'Reilly Media, 2008 `http://amazon.com/o/ASIN/0596516134/`. – ISBN 9780596516130

[BP00] Bahl, Paramvir ; Padmanabhan, Venkata N.: *RADAR: an in-building RF-based user location and tracking system.* 2000. – 775–784 S.

[BTG06] Bay, Herbert ; Tuytelaars, Tinne ; Gool, Luc V.: Surf: Speeded up robust features. In: *In ECCV*, 2006, S. 404–417

[FVE] Faheem, A. ; Virrankoski, R. ; Elmusrati, M.: *Improving RSSI based distance estimation for 802.15.4 wireless sensor networks*

[GKK04] Ganu, Sachin ; Krishnakumar, A. S. ; Krishnan, P.: Infrastructure-based location estimation in WLAN networks. In: *IEEE Wireless Communications and Networking Conference (WCNC*, Society Press, 2004, S. 465–470

[HB01] Hightower, Jeffrey ; Borriello, Gaetano: *A Survey and Taxonomy of Location Systems for Ubiquitous Computing.* 2001

[HLIN] Helén, Marko ; Latvala, Juha ; Ikonen, Hannu ; Niittylahti, Jarkko: *Using Calibration in RSSI-based Location Tracking System*

[HRV00] Haartsen, Jaap C. ; Radio, Ericsson ; V, Systems B.: The Bluetooth radio system. In: *IEEE Personal Communications* 7 (2000), S. 28–36

[IJ99] I. Jami, R. F. O. M. Ali A. M. Ali: *Comparison of Methods of Locating and Tracking Cellular Mobiles.* IEE Colloquium on Novel Methods of Location and Tracking of Cellular Mobiles and Their System Applications (Ref. No. 1999/046), 1999, pp. 1/1-1/6, 1999

[KHLH03] Kotanen, Antti ; Hännikäinen, Marko ; Leppäkoski, Helena ; Hämäläinen, Timo D.: Experiments on Local Positioning with Bluetooth.

In: *Proceedings of the International Conference on Information Technology: Computers and Communications*. Washington, DC, USA : IEEE Computer Society, 2003 (ITCC '03). – ISBN 0–7695–1916–4, 297–

[KM12] Komatineni, Satya ; MacLean, Dave: *Pro Android 4 (Professional Apress)*. Apress, 2012

[KW08] Klingbeil, Lasse ; Wark, Tim: A Wireless Sensor Network for Real-Time Indoor Localisation and Motion Monitoring. In: *Proceedings of the 7th international conference on Information processing in sensor networks*. Washington, DC, USA : IEEE Computer Society, 2008 (IPSN '08). – ISBN 978–0–7695–3157–1, 39–50

[Lag11] Laganière, Robert: *OpenCV 2 Computer Vision Application Programming Cookbook*. Packt Publishing, 2011

[Low99] Lowe, David G.: *Object Recognition from Local Scale-Invariant Features*. 1999

[Low03] Lowe, David G.: *Distinctive Image Features from Scale-Invariant Keypoints*. 2003

[Mil11] Miller, Alexander: *Erkennung beweglicher Objekte*, Hochschule für Technik und Wirtschaft Berlin, Diplomarbeit, 2011

[MN] Mardeni, R ; Nizam, Othman S.: *Node Positioning in ZigBee Network Using Trilateration Method Based on the Received Signal Strength Indicator (RSSI)*

[MS01] Mikolajczyk, Krystian ; Schmid, Cordelia: Indexing based on scale invariant interest points. In: *In Proceedings of the 8th International Conference on Computer Vision*, 2001, S. 525–531

[Neu02] Neumann, Dirk: *Kalman-Filter und Partikelfilter zur Selbstlokalisation – Ein Vergleich*. (Letzter Zugriff 29. Februar 2012), 2002. – `http://www.allpsych.uni-giessen.de/dirk/projects/particle.pdf`

[NLLP04] Ni, Lionel M. ; Liu, Yunhao ; Lau, Yiu C. ; Patil, Abhishek P.: LANDMARC: indoor location sensing using active RFID. In: *Wirel. Netw.* 10 (2004), November, 701–710. `http://dx.doi.org/http://dx.doi.org/10.1023/B:WINE.0000044029.06344.dd`. – DOI http://dx.doi.org/10.1023/B:WINE.0000044029.06344.dd. – ISSN 1022–0038

[PhD10] PhD, Widyawan: *Indoor Localisation: State of the Art and Novel Algorithms*. 2010

[PW] Pele, Ofir ; Werman, Michael: *The Quadratic-Chi Histogram Distance Family*

[SB11] Stephan Bergemann, Prof. Dr. Jürgen S.: *Adopting the LANDMARC Positioning System for 2.4 GHz Band*. 2011

[SGJ] Saxena, Mohit ; Gupta, Puneet ; Jain, Bijendra N.: *Experimental Analysis of RSSI-based Location Estimation in Wireless Sensor Networks*

[Tad] Tadakamadla, Shashank: *Indoor Local Positioning System For ZigBee, Based On RSSI*

[TM08] Tuytelaars, Tinne ; Mikolajczyk, Krystian: Local invariant feature detectors: A survey. In: *FnT Comp. Graphics and Vision* (2008), S. 177–280

[VLH+07] Varshavsky, Alex ; Lara, Eyal de ; Hightower, Jeffrey ; LaMarca, Anthony ; Otsason, Veljo: *GSM indoor localization*. 2007

[WFG92] Want, Roy ; Falcao, Veronica ; Gibbons, Jon: The Active Badge Location System. In: *ACM Transactions on Information Systems* 10 (1992), S. 91–102

Internetquellen

[1] Google. Bluetooth. (Letzter Zugriff 4. März 2012). http://developer.android.com/guide/topics/wireless/bluetooth.html.

[2] Google. Sensorevent. (Letzter Zugriff 5. März 2012). http://developer.android.com/reference/android/hardware/SensorEvent.html.

[3] Google. What is the ndk? (Letzter Zugriff 26. Februar 2012). http://developer.android.com/sdk/ndk/overview.html.

[4] Wikipedia. Koppelnavigation — wikipedia, die freie enzyklopädie. http://de.wikipedia.org/w/index.php?title=Koppelnavigation&oldid=96356505, 2011. [Letzter Zugriff 16. Dezember 2011].

[5] Wikipedia. Lateration — wikipedia, die freie enzyklopädie. http://de.wikipedia.org/w/index.php?title=Lateration&oldid=95036798, 2011. [Letzter Zugriff 4. April 2012].

[6] Wikipedia. Assisted global positioning system — wikipedia, die freie enzyklopädie. http://de.wikipedia.org/w/index.php?title=Assisted_Global_Positioning_System&oldid=97907170, 2012. [Online; Stand 10. März 2012].

[7] Wikipedia. Lumen (einheit) — wikipedia, die freie enzyklopädie. http://de.wikipedia.org/w/index.php?title=Lumen_(Einheit)&oldid=99890732, 2012. [Online; Stand 7. März 2012].

[8] Wikipedia. Lux (einheit) — wikipedia, die freie enzyklopädie. http://de.wikipedia.org/w/index.php?title=Lux_(Einheit)&oldid=100195264, 2012. [Online; Stand 7. März 2012].

[9] Wikipedia. Near field communication — wikipedia, die freie enzyklopädie. http://de.wikipedia.org/w/index.php?title=Near_Field_Communication&oldid=100789144, 2012. [Online; Stand 13. März 2012].

[10] Wikipedia. Nexus s — wikipedia, die freie enzyklopädie. http://de.wikipedia.org/w/index.php?title=Nexus_S&oldid=100267756, 2012. [Online; Stand 10. März 2012].