BEUTH HOCHSCHULE FÜR TECHNIK BERLIN
University of Applied Sciences

Bachelorarbeit

Medieninformatik

Ampelphasen-Informationssystem für FahrradfahrerInnen auf Grundlage persistenter geo- und zeitbasierter Daten

---

Berlin, den 15. Januar 2015

Autorin:
Jacoba Brandner
Matrikelnummer:
786635

Betreuerin:
Frau Prof. Dr. Gudrun Görlitz
Gutachterin:
Frau Prof. Dr. Petra Sauer

Inhalt

1 Einführung

1.1 Motivation

Im Berliner Verkehrswesen ist ein deutlicher Trend zu bemerken. Das Fahrrad wird zum ökologischen und gesundheitlichen, aktiven Lebensstil und wird dem hohen Verkehrsaufkommen der Automobile, insbesondere in der Stadtregion, entgegenwirken. “Fahrradfahren boomt in Berlin stärker als bislang bekannt” [J.A14]

Neue Fahrradwege und Vergrößerung des Fahrradstraßennetzes sind regionale Baumaßnahmen, die dabei aktuell diesen Fahrradtrend unterstützen. Grund der neuen Fahrradeuphorie ist nicht zuletzt die erfolgreiche Etablierung der E-Bikes[1]. E-Bikes erfreuen sich großer Beliebtheit und ermöglichen auch längere Touren ohne große Anstrengung.
Die Digitalisierung der Autoinnenräume mit Navigation und Bordelektronik sowie die Verbindungen zu Smartphones stellen aktuell keine Besonderheit mehr dar. Wird das Fahrrad nun als „vollwertiges“ Mitglied im Straßenverkehr angesehen, kann zusätzliche Elektronik wie Navigation die FahrradfahrerInnen unterstützen.
Sicherheit und eine rechtzeigige Ankunft am Ziel sind die Hauptaspekte der VerkehrsteilnemerInnen. Das Halten an der Ampel kann dabei schnell zu Verzögerungen führen. Doch wer die Restzeit im Voraus kennt, kann sich darauf einstellen und so die verlorenen Zeitabschnitte reduzieren.

1.2 Zielstellung

Der Fahrtfluss der RadfahrerInnen soll nicht unnötig unterbrochen werden. Rote Ampeln zwingen einen zum Anhalten – das Anfahren kostet Kraft und ist deshalb unbeliebt. So kommt es, dass viele RadfahrerInnen die Straße bei rot überqueren und die Verkehrssicherheit aller gefährden, wo doch das das Radfahren an sich zur Gesundheit beiträgt und auch gut für die Umwelt ist. Angesichts des Nutzenpotentials eines Ampelinformationssystems lässt sich die Zielstellung klar und deutlich formulieren. Durch das reibungslosen Passieren der Ampeln wird der Verkehr sicherer und das Radfahren attraktiver.
Um die Ampeldaten zu erfassen, gibt es verschiedene Möglichkeiten. Eine 100prozentige Deckung erreicht man nicht einmal durch manuelle Ablesung jeder Ampel, da circa 20 Prozent der Lichtsignalanlagen in Berlin manuell gesteuert werden. Die Berliner Verkehrslenkungszentrale stellt für diese Arbeit verkehrstechnische Unterlagen wie einen Lageplan, Signalzeitenpläne und Daten der verkehrsabhängigen Steuerung von Lichtsignalanlagen von circa zehn Anlagen als Basis für den zu

[1] Elektrofahrrad. Ein Fahrrad mit elektrischem Hilfsmotor

entwickelnden Prototypen zur Verfügung. Für die Auswertung der Daten werden die potentiellen Wartezeiten an der nächsten Ampel vorzeitig errechnet und den FahrerInnen mitgeteilt. Resultierend kann der Nutzer die Geschwindigkeit anpassen und die verbleibende Wegstrecke zur Ampel nutzen, um bei Grün ohne anzuhalten die Kreuzung zu überqueren. Für die Datenerhebung werden zugleich die mobilen Systeme der RadfahrerInnen genutzt. Wenn man das mit Ampeln auf gegebener Teststrecke umsetzt, kann zunächst der Prototyp des Ampelhinweissystem entwickelt werden.

Das Ziel der Arbeit ist ein Konzept und dessen prototypische Anwendung eines Ampelhinweissystem, welches einem auf Basis der zu erstellenden Ampeldatenbank Informationen über die Ampelschaltung zukommen lässt und ihn so interaktiv durch das Verkehrsnetz führt.

1.3 Aufbau der Arbeit

Zunächst wird analysiert welche Studien, Projekte oder Anwendungen es zu diesem Thema bereits gibt. Dann wird erläutert, ob sich eine mobile Anwendung oder eher eine Arduinoinstallation anbietet. Basierend darauf werden im vierten Kapitel die technischen Grundlagen erklärt. Hier werden also Definitionen und Entwicklungswerkzeuge beschrieben und ein Überblick über mögliche Einsatzgebiete gegeben. Für das Verständnis der Umsetzung ist die Klärung der theoretischen Berechnungsgrundlagen erforderlich.
Im Anforderungsanalysekapitel werden die Anforderungen an Funktionalität und Usability Kriterien für die Anwendung ermittelt und strukturiert.
Es werden Personas, fiktive Benutzer, eingeführt und schließlich eine Zusammenfassung der herausgearbeiteten Anforderungen gestellt.

Das sechste Kapitel bildet mit der Konzipierung den Kern dieser Arbeit. *Architektur, Design, (Theorie — funktioniert nicht wie in Konzeption beschrieben, weil...) es wird auf.. einngegangen.... mobile Anwendung. Applikation (App). die NutzerInnen auf Ampeln hinweist und die Dauer der Phase. Zusammenfassend wird das Konzept am Ende von allen Personas nocheinmal kritisch betrachtet und evaluiert.*
Kapitel sieben beschreibt die Umsetzung des exemplarischen Prototyps. Dieser wird dann in Architektur, Funktionalität und Design erläutert und schließlich in mehreren Testreihen evaluiert. Anhand einiger Systemtests wird eine Optimierung der Anwendung herausgearbeitet und gegebenenfalls umgesetzt.

Den Abschluss dieser Arbeit bildet eine Zusammenfassung der Ereignisse dieser Arbeit und einen Auslick auf zukünftige Entwicklung hinsichtlich des Themas.

2 Grundlagen

Dieses Kapitel befasst sich mit sowohl den mathematischen als auch den technischen Grundlagen der zu behandelnden Thematik, welche für das weitere Verständnis der Arbeit beitragen.

2.1 Technische Grundlagen

Erstellt wird eine Smartphone Anwendung, deren Grundlage für die Implementierung die Software-Plattform Android ist. Im folgenden Abschnitt werden Funktionsweise und Besonderheiten der verwendeten Technologien beschrieben.

2.1.1 Android

Die umfassende Open-Source-Plattform Android stellt eine vollständige Ausstattung für Mobilgeräte dar. Android-Anwendungen werden mit der Programmiersprache Java und der Auszeichnungssprache Extensible Markup Language (XML) entwickelt. Mit dem Android Software Development Kit (SDK)[1] werden die Werkzeuge und Application Programming Interface (API) zur Verfügung gestellt, die erforderlich sind Mobilanwendungen auf der Android-Plattform erzeugen zu können.
Zu den wichtigsten SDK Werkzeugen gehören der Android SDK-Manager[2], der AVD-Manager[3], der Emulator und der Dalvik Debug Monitor Server[4] (Ddms). Die Virtuelle Maschine Dalvik wurde unter Berücksichtigung von Rechenleistung und der Lebensdauer von Batterien speziell von Google für Android entworfen. [anda] Ab Android 4.4 wird die Android Runtime (ART) als alternative zur Dalvik Virtual Maschine (VM) angeboten. Diese bietet eine Reihe von neuen Funktionen die die Laufzeit verbessern. [**?**]
Mit dem Android Native Development Kit (NDK)[5] existiert auch ein Tool, mit dem Teile einer Anwendung in systemeigenen Programmiersprachen wie C oder C++ implementiert werden können. Programmcode, der in solchen Sprachen eignet, ist zum Beispiel bei CPU-intensiven Operationen wie Signalverarbeitungen oder Physik-Simulationen. Hier ist sicherzustellen, ob die erforderlichen Bibliotheken in dem SDK auch verfügbar sind. [andb]
Einen Überblick über die komplexe Android-Systemarchitektur, welche nachfolgend (nach [KC13] S. 3ff) kurz beschrieben wird, zeigt die folgende Abbildung 2.1.

[1] Das Android SDK steht unter `https://developer.android.com/sdk/index.html` zum Download bereit
[2] Der SDK-Manager verwaltet die SDK-Pakete, sowie die installierten Pakete imd System-Images
[3] Der AVD-Manager bietet eine grafische Oberfläche in der Android Virtuell Devices verwaltet und im Emulator ausgeführt werden können.
[4] Mithilfe des Ddms können Android Anwendungen auf Fehler untersucht werden
[5] Das Android NDK steht unter `https://developer.android.com/tools/sdk/ndk/index.html` zum Download bereit

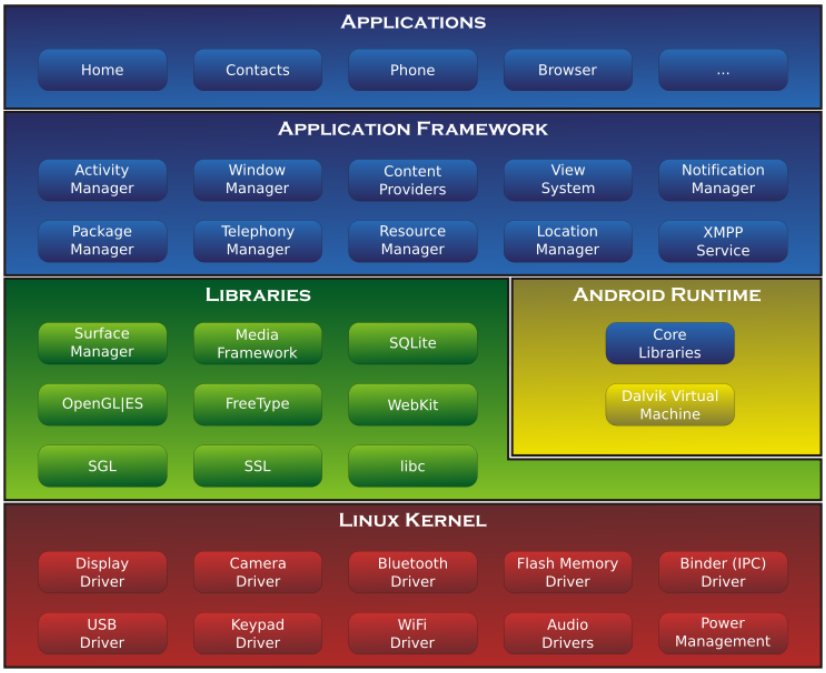

Abbildung 2.1: Die Android-Systemarchitektur Quelle: `http://en.wikipedia.org/wiki/File:Android-System-Architecture.svg`

Linux Kernel: Android basiert auf dem Linux 3.1-Kernel. Dieser eine bewährte Betriebssystemgrundlage indem er die erforderlichen Hardware-Treiber zur Verfügung stellt.

Bibliotheken: Systemeigenen Bibliotheken sind C/C++ Bibliotheken und vorinstalliert. Dazu gehören alle Bibliotheken im grünen Bereich von Abbildung 2.1:

- Surface Manager Der für das individuelle Display Verwaltung verantwortliche Oberflächen-Manager
- OpenGL/ES Eine 2D und 3D Grafikbibliothek
- SGL Eine 2D Grafikbibliothek
- Media-Framework eine Medien-Bibliothek zur Wiedergabe von Audio- und Video-Daten
- FreeType eine Bibliothek zur Darstellung von Computerschriften als Rastergrafik
- SSL Das Secure-Socket-Layer für die Internet-Sicherheit
- SQLite Ist eine ausgereifte Datenbank die den internen Speicher nutzt
- Webkit WebKit ist die Standard-Browser-Engine und erlaubt das schnelle Rendern und Anzeigen von HTML Seiten
- libc C-Bibliothek

Android Runtime: Die Android Laufzeitumgebung nutzt die Java-Core-Bibliotheken und die Dalvik VM. Die Dalvik VM ist Googles Implementation von Java, zur Anwendung auf mobilen Geräten optimiert. Jede gestartete Android-Anwendung läuft in einem eigene Prozess und bekommt darüber hinaus seine eigene Dalvik VM. Da die Anwendungen über keinen gemeinsamen Speicher verfügen erhöht das die Sicherheit und Verfügbarkeit und ist somit optimaler, obwohl mehr Ressourcen benötigt werden. Denn ein sterbender Prozess nimmt so nur seine “eigene“ Anwendung mit.

Die Anwendungen werden zunächst von einem normalen Java-Compiler in Java-Bytecode übersetzt und dann von dem Dex-Compiler in den Dalvik-Bytecode, welcher schließlich von der Dalvik VM ausgeführt wird.

Application Framework: Androids Application-Framework ist eine Umgebung die unterschiedliche DIenste zur Verfügung stellt. Sie bietet EntwicklerInnen Zugriff auf die im Kern verwendeten APIs sowie auf die Java-Bibliotheken die für Android erstellt wurden.

Applications: Auf der obersten Ebene in Abbildung 2.1 befinden sich die Anwendungen die den täglichen Telefon-Bedarfs wie Adressbuch, Messenger, E-Mail, Internet-Browser etc. decken. Zusätzlich unterstützt Android verschiedene Anwendungen von Drittanbietern. Diese sind hauptsächlich in Java geschrieben und werden am häufigsten über den Google Play Store verteilt.

2.1.2 Die native Datenbank SQLite

2.1.3 Mobile Sensorik

Im Zuge dieser Arbeit wird eine Ampelinformationsanwendung entwickelt, die auf mobilen Geräten lauffähig, speziell auf Smartphones lauffähig sein soll. Hierfür wird der Global Positioning System (GPS)-Sensor verwendet.... Die eigene Position und Geschwindigkeit wird mittels GPS-Sensor ermittelt, um in Verbindung mit der festen Position der nächsten Lichtsignalanlage (LSA) die optimale Geschwindigkeit für das Erreichen der “Grünen Welle“ zu errechnen. Die GPS-Sensoren in Smartphones sind kostengünstiger, kleiner und haben einen geringeren Stromverbrauch. Dafür allerdings eine geringere Messgenauigkeit. In der zu entwickelnden Anwendung kommt es auf jeden Meter an. Ob die Genauigkeit des integrierten GPS-Sensors genügt muss also getestet werden.

Mobile Sensorik unter Android

Geolokation mittels GPS

Funktionsweise: Position und Geschwindigkeit können aus Signallaufzeiten zwischen Satelliten und EmpfängerIn bestimmt werden.

G-Sensor

Blabla Beschleunigungssensor...
Anwendungsbeispiel: um Bewegungsänderungen festzustellenß
Messprinzip: misst die Kraft, die auf die Masse im Sensor wirkt (F=m*a)
Sensormasse ist bekannt, die Beschleunigung kann also bestimmt werden. Hohe Messgeschwindigkeit
Smartphone Sensoren sind potentiell für die Schlaglochdetektion geeignet

2.2 Berechnung der Geschwindigkeitsempfehlung

Präsentiert das System während der Anwendung eine Geschwindigkeitsempfehlung, ist diese abhängig von der Fahrtgeschwindigkeit und vom Abstand zur Ampel. Angenommen die Progressionsgeschwindigkeit v wird zum Zeitpunkt t_1 ermittelt, die LSA schaltet zum Zeitpunt t_2 auf Rot und Abstand zur Ampel beträgt s, dann gilt:

$$v = \frac{s}{t_2 - t_1}$$

Der Abstand zur Ampel wird also durch die verbleibende Zeit dividiert. Die von der Berliner Verkehrsleitzentrale zur Verfügung gestellten Ampelschaltpläne und Position der angesteuerten Ampel dienen als Grundlage dieser Berechnung und sind aus der Datenbank zu holen. Die aktuelle Position des Fahrrads wird vom GPS Sensor des Smartphones ermittelt und daraus der Abstand zur Ampel errechnet. Die Abbildung 2.2 soll die Berechnungsgrundlagen veranschaulichen:

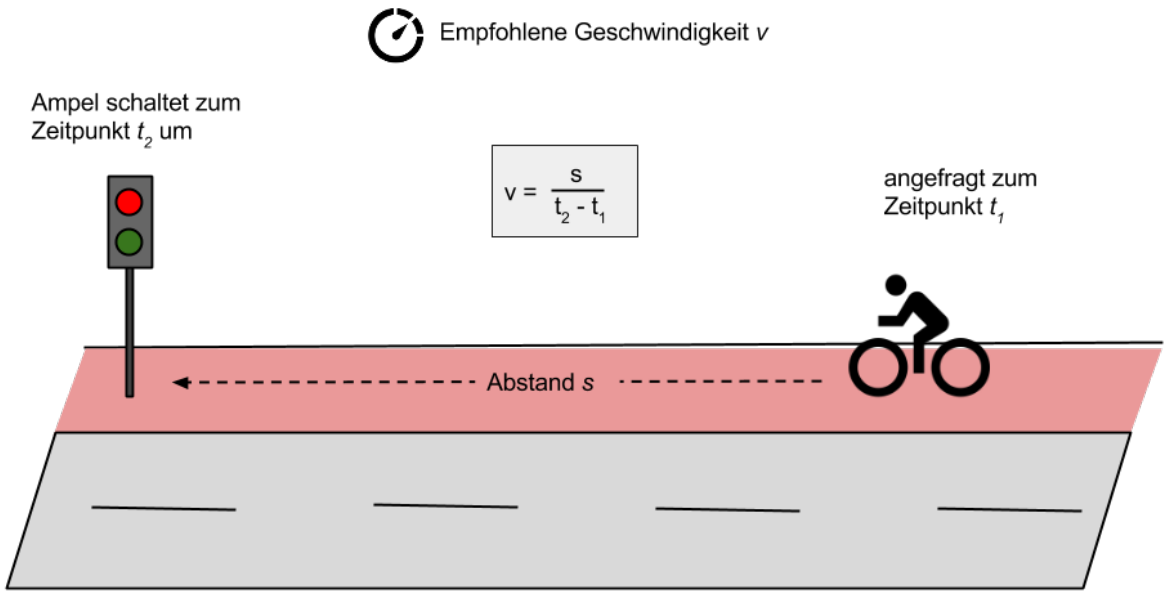

Abbildung 2.2: Veranschaulichung der Berechnung

Um die ensprechende LSA während der Grünphase zu passieren, muss letztendlich die empfohlene Geschwindigkeit v eingehalten werden.

3 Szenarien im Ampelbereich

Alle in Kapitel **??** angeführten Studien zu Ampelinformationssystemen und Konzepte zu Fahrraderweiterungen haben die Gemeinsamkeit des selbstkontrontollierten Fahrverhaltens der FahrerInnen. Ausgesprochen werden lediglich Empfehlungen, die möglichst intuitiv und schnell vermittelt werden. Grundlegend sollte die Anwendung in der Lage sein, die passende Empfehlung oder Handlungsaufforderung anzuzeigen, die sich aus folgenden Szenarien ergeben.

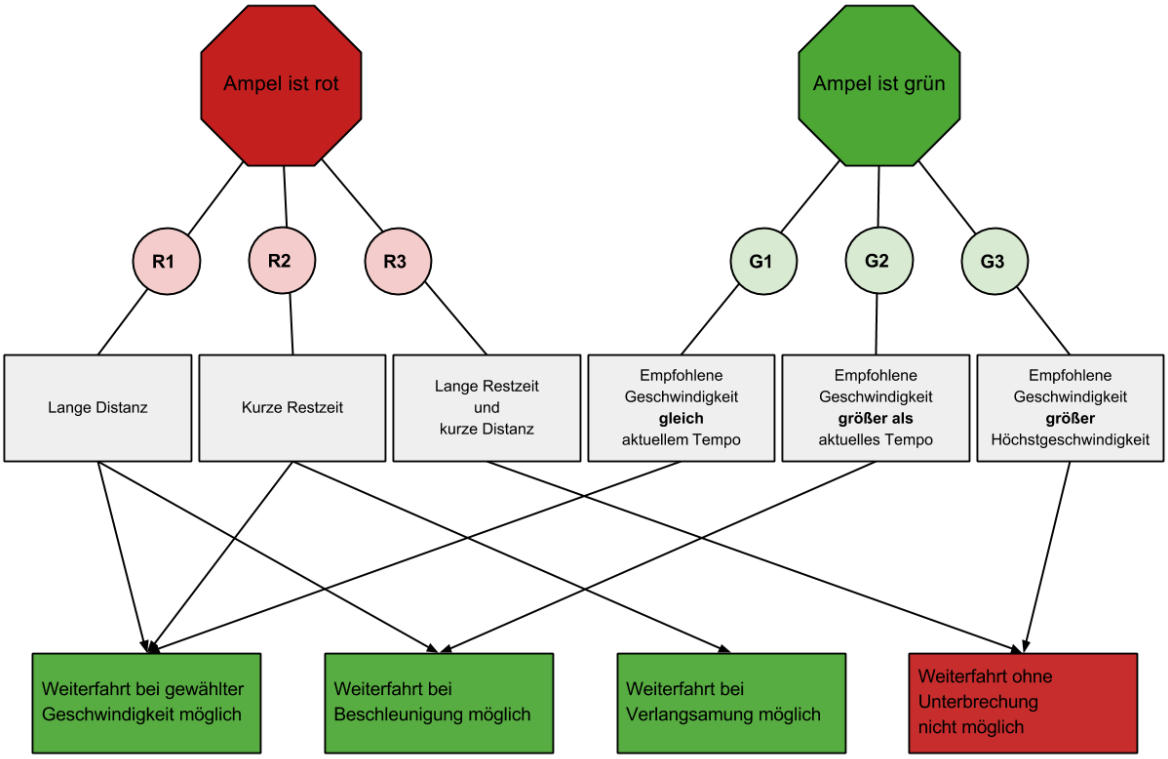

Abbildung 3.1: Szenarien im Ampelbereich

Szenario R1:

Zeigt die Ampel im Moment und noch eine ganze Weile auf Rot ist je nach Distanz zwischen Ampel und Fahrrad ist ein reibungsloses Passieren der Ampel durch Beschleunigung oder ohne Änderung der Geschwindigkeit zu erreichen.

Szenario R2:

Die Ampel zeigt im Moment auf Rot, aber die Restrotzeit ist kurz. Auch hier gilt: Je nach Entfernung zwischen Fahrrad und LSA, ist ein reibungsloses Passieren der Ampel durch Verzögerung oder ohne Änderung des Tempos zu erreichen.

Szenario R3:

Zeigt die Ampel im Moment Rot, die Restrotzeit ist lang, doch die Entfernung zwischen Fahrrad und LSA hoch, so ist keine Weiterfahrt ohne Unterbrechung möglich.

Szenario G1:

Ist die empfohlene Geschwindigkeit gleich der aktuellen, ist ein reibungsloses Passieren bei beibehaltenem Tempo möglich. Es besteht kein Aktionsbedarf.

Szenario G2:

Zeigt die Ampel im Moment Grün und ist die empfohlene Geschwindigkeit höher als die aktuelle, ist ein reibungsloses Passieren durch Beschleunigung zu erreichen. Bei der Anzeige der Progressionsgeschwindigkeit ist selbstverständlich die geltende Höchstgeschwindigkeitsbegrenzung oder die eingestellte Höchsteschwindigkeit zu beachten.

Szenario G3:

Zeigt die Ampel im Moment Grün und ist die empfohlene Geschwindigkeit höher als die aktuelle und gleichzeitig auch höher als die zugelassene, bzw. eingestellte Höchstgeschwindigkeit, ist die Distanz zur Ampel zu groß und ein reibungsloses Passieren nicht möglich.

Resultierend aus den Szenarien im Ampelbereich ergeben sich die folgenden Systemzustände.

- Zustand a: Weiterfahrt durch Verlangsamung möglich
- Zustand b: Weiterfahrt durch Beschleunigung möglich
- Zustand c: Kein Aktionsbedarf
- Zustand d: Keine Weiterfahrt ohne Unterbrechung möglich

Im weiteren Verlauf wird unter anderem beschrieben wie diese Systemzustände in den Designprozess eingebunden werden.

Abkürzungen

API	Application Programming Interface.
App	Applikation.
C2X	Car-to-Infrastructure oder Vehicle-to-Infrastructure.
C2X	Car-to-X oder Vehicle-to-X.
DSRC	Dedicated Short Range Communication.
LSA	Lichtsignalanlage.
NDK	Native Development Kit.
REST	REpresentational State Transfer.
SDK	Software Development Kit.
XML	Extensible Markup Language.

GLOSSAR

Arduino

Die Arduino-Plattform ist eine quelloffene, aus Soft- und Hardware bestehende Physical-Computing-Plattform. Die Hardware besteht aus einem einfachen I/O-Board mit einem Mikrocontroller und analogen und digitalen Ein- und Ausgängen. Die Entwicklungsumgebung basiert auf Processing, die auch technisch weniger Versierten den Zugang zur Programmierung und zu Mikrocontrollern erleichtern soll.

Smartphone

Mobiltelefon, das sich von einem klassischen Mobiltelefon durch einen größeren berührungsempfindlichen Bildschirm (Touchscreen) und diverse Sensoren, wie dem GPS unterscheidet. So ist eine Interaktion mit der Umgebung und den AnwenderInnen möglich.

ABBILDUNGSVERZEICHNIS

QUELLCODEVERZEICHNIS

LITERATURVERZEICHNIS

[anda] *Android Tools Help*. https://developer.android.com/tools/help/index.html, . – Zugriff: 15.01.2015

[andb] *Android NDK*. https://developer.android.com/tools/sdk/ndk/index.html, . – Zugriff: 15.01.2015

[J.A14] J.ANKER: Fahrradfahren boomt in Berlin stärker als bislang bekannt. In: *Berliner Morgenpost* (2014), Juni

[KC13] KRAJCI, Iggy ; CUMMINGS, Darren: *Android on x86: An Introduction to Optimizing for Intel Architecture*. New York : Apress Media L.L.C., 2013. – ISBN 978–1–4302–6130–8