BEUTH HOCHSCHULE FÜR TECHNIK BERLIN
University of Applied Sciences

Bachelorarbeit

Medieninformatik

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

---

Berlin, den 17. Januar 2015

Autorin:
Jacoba Brandner
Matrikelnummer:
786635

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

Inhalt

1 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.

1.1 Technische Grundlagen

Im Zuge dieser Arbeit wird eine Smartphone Anwendung erstellt, deren Grundlage für die Implementierung die Software-Plattform Android und der im Smartphone integrierte Global Positioning System (GPS)-Sensor ist. Die eigene Position und Geschwindigkeit wird mittels GPS ermittelt, um in Verbindung mit der festen Position der nächsten Lichtsignalanlage (LSA) die optimale Geschwindigkeit für das Erreichen der "Grünen Welle" zu errechnen. Im folgenden Abschnitt werden Funktionsweise und Besonderheiten der verwendeten Technologien beschrieben.

1.1.1 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, der AVD-Manager, der Emulator und der Dalvik Debug Monitor Server(Ddms). Der SDK-Manager verwaltet die SDK-Pakete, sowie die installierten Pakete und System-Images. Der AVD-Manager bietet eine grafische Oberfläche in der Android Virtuell Devices verwaltet, und im Emulator ausgeführt werden können. Mithilfe des Ddms können Android Anwendungen auf Fehler untersucht werden. [anda]
Ursprünglich gab es die Virtuelle Maschine Dalvik, die unter Berücksichtigung von Rechenleistung und der Lebensdauer von Batterien speziell für Android entworfen wurde. [Quelle] Diese wurde mit Android 5.0 von der Android Runtime (ART) als effizientere Laufzeit komplett ersetzt. [andb]

Mit dem Android Native Development Kit (NDK)[2] existiert auch ein Tool, mit dem Teile einer Anwendung in systemeigenen Programmiersprachen wie C oder C++ implementiert werden können. Programmcode der in solchen Sprachen geschrieben ist, eignet sich zum Beispiel bei CPU-intensiven Operationen wie Signalverarbeitungen oder Physik-Simulationen besonders gut. Hier ist allerdings

[1] Das Android SDK steht unter `https://developer.android.com/sdk/index.html` zum Download bereit
[2] Das Android NDK steht unter `https://developer.android.com/tools/sdk/ndk/index.html` zum Download bereit

sicherzustellen, ob die erforderlichen Bibliotheken in dem SDK auch verfügbar sind. [andd]
Einen Überblick über die komplexe Android-Systemarchitektur, welche nachfolgend (nach [KC13] S. 3ff) kurz beschrieben wird, zeigt die folgende Abbildung.

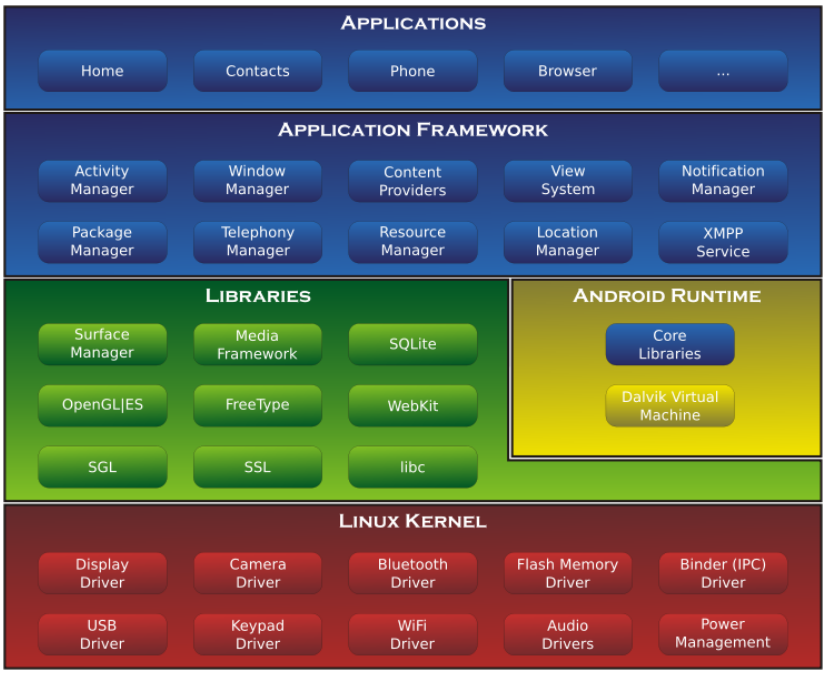

Abbildung 1.1 Die Android-Systemarchitektur Quelle: `http://en.wikipedia.org/wiki/File:Android-System-Architecture.svg`

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 1.1:

- Surface Manager Der für die Displayverwaltung 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 bis zur Version 5.0 auch die Dalvik Virtual Maschine (VM), weswegen diese noch auf der Grafik abgebildet ist. *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. [andc]

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 1.1 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.

1.1.2 Die native Datenbank SQLite

1.1.3 Mobile Sensorik <– Ich benutze nur GPS?

Ein Sensor[3] ist ein Bauelement, das physikalische Eigenschaften wie Helligkeit, Temperatur oder Beschleunigung sowohl quantitativ als auch qualitativ erfassen kann. 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 im Rahmen dieser Arbeit getestet werden.

Mobile Sensorik unter Android

Die meisten Android-Mobilgeräte verfügen über integrierte Sensoren, die die Bewegung, Ausrichtung und verschiedene Umgebungsbedingungen messen. Diese Sensoren sind praktisch wenn man dreidimensionale Gerätbewegungen, Positionierungen oder Änderungen in der Umgebung des Gerätes überwachen möchte. So können zum Beispiel Spieleanwendungen den Beschleunigungssensor nutzen, um komplexe BenutzerInnengesten und Bewegungen wie Neigung, Erschütterung, Drehung oder Schwenkung erfassen.
Die Android-Plattform unterstützt Bewegungssensoren zum Messen von Beschleunigungen und Drehungen in drei Achsen, Umgebungssensoren zur Ermittlung verschiedener Umweltparameter wie Luftdruck und -feuchtigkeit, oder Beleuchtung und Temperatur, und Positionssensoren zum Messen der physikalischen Position des Gerätes. Android bietet mit dem Android Sensor Framework eine Sammlung von Klassen und Schnittstellen an mithilfe dessen man diese Sensoren zugreifen und deren Daten erfassen kann. [andf]

[3] aus dem Lateinischen, deutsch: "*fühlen*"

Geolokation mittels GPS

GPS gewährleistet die Bestimmung des exakten Standpunktes und ist so wesentlicher Bestandteil ortsgebundener Anwendungen wie zum Beispiel die in Kapitel **??** beschriebenen.
Funktionsweise: Position und Geschwindigkeit können aus Signallaufzeiten zwischen Satelliten und EmpfängerIn bestimmt werden.
Android unterstützt mit dem `android.location` Paket den Zugriff auf die Ortungsdienste. Als zentrale Komponente des Location Frameworks stellt der `LocationManager` APIs zur Lokalisierung des Geräts bereit. Mit dem `LocationManager` ist die Anwendung in der Lage alle Location Provider[4] des letzten bekannten Standortes abzufragen, sich für regelmäßige Updates zur Position des Gerätes anzumelden und sich wieder abzumelden wenn sich das Gerät außerhalb gegebener Parameter befindet. [ande]

[4] deutsch: Standortanbieter. Ein Standortanbieter bietet regelmäßige Berichte über die geographische Lage des Gerätes

1.2 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 1.2 soll die Berechnungsgrundlagen veranschaulichen:

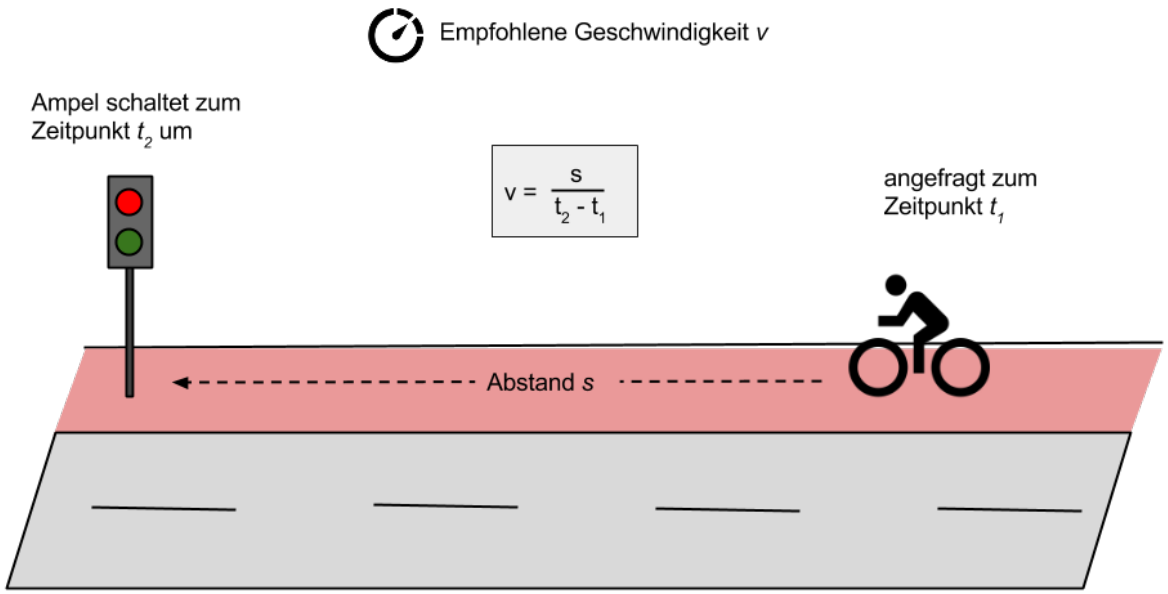

Abbildung 1.2 Veranschaulichung der Berechnung

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

2 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.

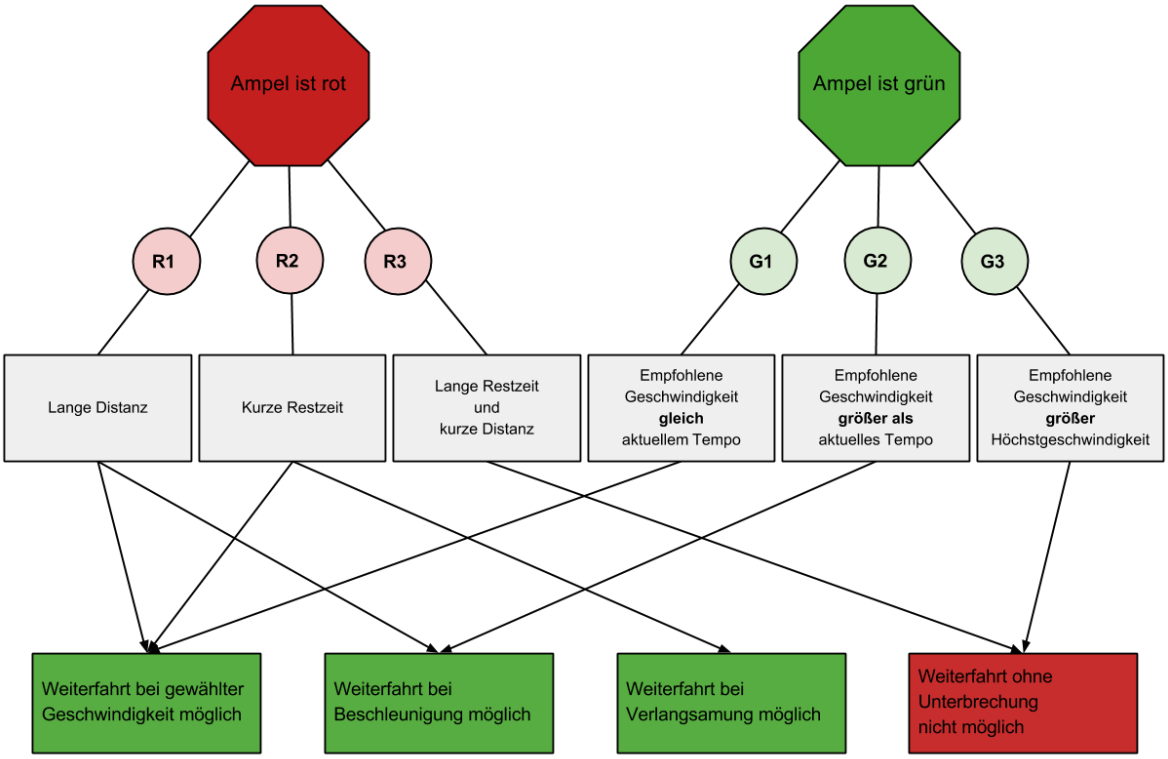

Abbildung 2.1 Szenarien im Ampelbereich

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

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

3 Anforderungsdefinition

Dieses Kapitel soll helfen eine Vorstellung für die Anforderungen an die Ampelhinweissystem -App zu bekommen. Im heutigen High-Technologie Zeitalter spielt die Benutzbarkeit und Funktionalität des Produkts eine große Rolle. Es geht nicht nur darum, Interesse zu wecken, sondern auch die NutzerInnen für das Produkt zu begeistern.
Aus den oben genannten Szenarien werden im Folgenden die Anforderungen hergeleitet, die die zu entwickelnde Smartphone-Anwendung erfüllen soll. Für die Umsetzung einer solchen Applikation ist die Positions- und Geschwindigkeitsbestimmung obligatorisch. Im Zusammenhang mit den Ampeldaten, bestehend aus Lage- und Schaltplan sollen die notwendigen Berechnungen durchgeführt und deren Ergebnisse als Empfehlung ausgesprochen.

3.1 Funktionalität

Die Bestimmung der Position, Fahrtrichtung und Geschwindigkeit des Fahrrads ist die zentrale Voraussetzung für eine zeitgerechte Empfehlungsaussprechung. Sie sollte schnell erfolgen, sodass es keine Verzögerung der berechneten Ergebnisanzeige gibt und die Informationen eine hohe Genauigkeit betragen.

- Genau, schnell
- Strecke muss später genauer erechnet werden. Luftlinie. Kartennavigation.
- Zugriff schnell (Datenbank)
-

3.2 Die graphische Oberfläche

Ebenso wie die Bestimmung der Position spielt auch die graphische Darstellung eine große Rolle. Gerade bei dem Gebrauch im Verkehr ist die Eindeutigkeit und schnelle Erfassbarkeit der zu übermittelnden Informationen bedeutend.
Um nicht von dem Verkehr abzulenken und diesen somit zu gefährden muss die Information, sowohl in Bedeutung als auch in Darstellung auf einen kurzen Blick erkennbar sein. Demnach sollte die Oberfläche möglichst einfach gehalten werden.
Da die Anwendung während der Fahrt nicht bedienbar sein muss – denn auch das würde vom Verkehr ablenken – besteht mehr Raum für die Informationsanzeige. Es ist davon auszugehen, dass die Anwendung ausschließlich im Freien Gebrauch findet. Dies und die dort herrschenden wechselnden

Helligkeiten, zum Beispiel bei der Fahrt durch Schatten, setzen eine Verwendung von hohen Kontrasten voraus.

Empfehlungsanzeige??? <– Frau Görlitz fragen

Im Allgemeinen sollte die Anwendung in der Lage sein die sich aus Kapitel 2 resultierenden vier Systemzustände zu visualisieren und gegebenenfalls eine entsprechende Empfehlung auszusprechen. Zur Verdeutlichung und einfacher Erfassbarkeit bietet es sich an, die entsprechenden Farben zu nehmen. Also ist für die Zustände *b*,*c* und *d*, die das Erreichen der Grünen Welle mit oder ohne Veränderung der eigenen Geschwindigkeit Grün und für den Zustand *a*, der ausdrückt dass keine Weiterfahrt ohne Absteigen möglich ist, Rot zu verwenden.

4 Konzeption

4.1 Architektur

Klassenstruktur..

4.1.1 Navigation

4.2 Das Design

Hohe Kontraste wegen Wetter.
einfache UI .
Minimalistisch sodass ein Blick genügt und man nicht abgelenkt wird.
Angezeigt wird: Geschwindigkeit schneller oder langsamer, Restrotanzeige, grüne Welle.

(a) keine Weiterfahrt möglich

(b) Kein Aktionsbedarf

(c) Weiterfahrt durch Verlangsamung möglich

(d) Weiterfahrt durch Beschleunigung möglich

Abbildung 4.1 Umsetzung des Designs anhand der Systemzustände

4.2.1 Anzeigeelemente

Geschwindigkeit

Ampeln

Informationen

4.3 Ampeldatenanfrage und Auswertung

4.3.1 Sensoren

GPS

wird gebraucht für...

4.4 Theorie

Um die korrekte Umsetzung des Prototyps zu ermöglichen, müssen zunächst einmal prinzipielle Theorien und Hintergründe diesen betreffend betrachtet werden. grundlegendes Wissen über geographische Koordinaten sowie mathematische Voraussetzungen im Umgang mit diesen, müssen zur Ideenverwirklichung berücksichtigt werden.

4.4.1 Die Berechnung der Entferung

4.4.2 Die Berechnung der Ankunft in Abhängigkeit der Geschwindigkeit

4.4.3 Die Anzeige der Restrotanzeige

5 Der Prototyp

Prototyp zeit, wie mittels GPS ... realisiert werden kann. Design und Funktionalitäten werden ebenfalls vorgestellt

5.1 Android-Manifest

5.2 Verwendete Bibliotheken

5.2.1 googlebla

5.3 MainActivity

5.4 Umsetzung Szenarien

5.5 Lokalisierung

6 Evaluierung

6.1 Systemtest

ist das GPS schnell/genau genug fürs radfahren? Optimierung ggf. umsetzen

7 Ergebnis und Ausblick

7.1 Ampelhinweissystem

7.2 Ausblick

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.
GPS	Global Positioning System.
LED	Licht-emittierende Diode.
LSA	Lichtsignalanlage.
NDK	Native Development Kit.
REST	REpresentational State Transfer.
SDK	Software Development Kit.
VM	Virtual Maschine.
WLAN	Wireless Local Area Network.
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.

C2I

direkter, drahtloser Datenaustausch zwischen Fahrzeugen jeglicher Art und infrastrukturellen Einrichtungen wie Funkbaken und Lichtsignalanlagen auf Basis von Wireless Local Area Network (WLAN), Bluetooth oder Dedicated Short Range Communication (DSRC).

C2X

direkter Informationsaustausch zwischen Fahrzeugen jeglicher Art, Verkehrsleittechnik wie z.B. Lichtsignalanlagen und Verkehrsleitzentralen.

DSRC

funkgestützte sicherheitsrelevante und private Dienste, die in der Automotive-Technik von mobilen Stationen ausgeführt werden können.

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 Lollipop*. http://developer.android.com/about/versions/lollipop.html, . – Zugriff: 16.01.2015

[andc] *Introducing ART*. https://source.android.com/devices/tech/dalvik/art.html, . – Zugriff: 15.01.2015

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

[ande] *Location and Maps*. http://developer.android.com/guide/topics/location/index.html, . – Zugriff: 17.01.2015

[andf] *Sensors Overview*. http://developer.android.com/guide/topics/sensors/sensors_overview.html, . – Zugriff: 17.01.2015

[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