Entwicklung eines mobilen Navigationssystems für Elektrofahrzeuge auf Basis von OpenStreetMap-Daten

Dominik Franke, Dzenan Dzafic, Carsten Weise, Stefan Kowalewski

Abstract

Fahrzeuge mit Elektromotoren (z.B. Elektrorollstühle) sind in Ihrem Bewegungsraum durch ihre Akku-Kapazität und ihren Stromverbrauch eingeschränkt. Diese beiden Eigenschaften sind wiederum stark abhängig von der Steigung und dem Belag der zu befahrenden Strecke. Je größer die zu bewältigende Streckensteigung und je unebener der Belag, desto schneller sinkt der Akkustand und damit die erreichbare Strecke.

Die Geodatensammlung des OpenStreetMap (OSM) Projektes hat bereits gezeigt, dass die Informationen wie Streckensteigung und Oberflächenbelag eines Weges teilweise in den Datenbestand aufgenommen werden. Doch inwieweit lassen sich diese Informationen für eine Routenplanung verwenden? Auf Basis des Routenplaners für Rollstuhlfahrer von *rollstuhlrouting.de* wird ein Algorithmus vorgestellt, welcher das mobile Open Source Navigationssystem AndNav (*www.andnav.org* - verwendet ebenfalls OSM-Daten) so erweitert, dass es neben der kürzesten und schnellsten auch die effizienteste Strecke bezüglich des Energieverbrauchs von Elektrofahrzeugen berechnet.

1. Einleitung

Wenn man mit einem Elektrofahrzeug unterwegs ist, interessiert einen oftmals nicht der kürzeste oder schnellste Weg, sondern der effizienteste bezüglich des Energieverbrauchs. So maximiert man z.B. die Reichweite des Fahrzeuges. Die vorliegende Arbeit modifiziert den Algorithmus A* [7], welcher u.a. bei dem Open Source Webservice OpenRouteService (ORS) [4] zur Navigation verwendet wird, sodass dieser einen entsprechenden effizientesten Weg bezüglich des Energieverbrauchs berechnet.

Diese Arbeit geht im zweiten Kapitel auf das Gesamtprojekt ein, indem das Zusammenspiel zwischen AndNav [6], ORS und OSM [2]erklärt wird. Des Weiteren beinhaltet das zweite Kapitel Grundlagen, wie den Aufbau und die Semantik des OSM Geodatensatzes, die zum tieferen Verständnis der folgenden Kapitel relevant sind. Kapitel drei stellt den erweiterten Algorithmus vor. Kapitel vier schließt die Arbeit mit einem Ausblick und möglichen Erweiterungen ab.

Das hier beschriebene Projekt findet im Rahmen einer Kooperation zwischen dem Lehrstuhl Informatik 11 der RWTH Aachen und des geographischen Institutes der Universität Heidelberg statt. Die Universität Heidelberg trägt insbesondere durch das Hosten von openrouteservice.de und dem verwandten Projekt rollstuhlrouting.de [9] zur Realisierung des hier vorgestellten Konzeptes bei. Zudem sind beide Seiten an einer intensiven Erforschung der Thematik interessiert, sodass redundante Arbeiten ausgeschlossen werden und sich Synergien ergeben. Abbildung 5 veranschaulicht schematisch die Kooperation, sowie die Kernkompetenzen der beteiligten Gruppen. Hierzu sollte erwähnt werden, dass beide Universitäten stark von der Qualität der OSM-Daten abhängig sind (Verfügbarkeit von Steigungsinformationen, Zuverlässigkeit dieser Informationen, …). Dies ist damit zu begründen, dass OSM einer freiwilligen Nutzung zu Grunde liegt und jeder Nutzer seine erfassten Daten selbst einpflegen kann. Obwohl OSM ein Open Source Projekt ist, hat die Erfahrung gezeigt, dass die Daten relativ zuverlässig sind. Der Datenbestand übertrifft seit einiger Zeit sogar jenen von kommerziellen Anbietern wie beispielsweise Google und Microsoft. Der hier vorgestellte

Algorithmus wird somit vom Lehrstuhl Informatik 11 der RWTH Aachen entwickelt und gepflegt und von Pascal Neis, M.Sc. des geographischen Institutes der Universität Heidelberg in ORS implementiert und über entsprechende Schnittstellen und das Webinterface openroute-service.org für Endbenutzer und Applikationen zur Verfügung gestellt.

2. Grundlage

In diesem Abschnitt geht es um die Abläufe zwischen den unterschiedlichen Komponenten, die für das vorgestellte Projekt eine wichtige Rolle spielen. Zusätzlich zu der Interaktion der beteiligten Komponenten wird eine grobe Übersicht der Kompetenzbereiche der RWTH Aachen und der Universität Heidelberg gegeben. Eine solche Zusammenarbeit bot sich an, da Letztere ihre Kernkompetenzen im Bereich Geo-Informatik vorweisen und einige Ressourcen bereitstellen können, um solch ein umfangreiches Projekt grundlegend zu unterstützen.

Interaktion zwischen den Komponenten

Um das Zusammenspiel der Komponenten zu erläutern, wird beispielhaft eine vollständige Routenanfrage durchgeführt. Diese Beispielanfrage wird in Abbildung 1 illustriert. Zu Beginn stellt der Benutzer eine Anfrage mit seinen Routenangaben via Mobiltelefon an AndNav. Da AndNav die Funktion eines Clienten übernimmt, leitet es die Anfrage an den Server von OpenRouteService (ORS) [4] weiter. Dieser sucht entsprechend der weitergeleiteten Anfrage verfügbares Kartenmaterial bei OpenStreetMap (OSM). Im nächsten Schritt der Routenanfrage werden die angeforderten Routenabschnitte des Kartenmaterials von OSM an ORS übermittelt. Diese dienen der Routenplanung, die durch die Routenangaben spezifiziert wurde, als Grundlage zur Berechnung einer geeigneten Route. Abschließend wird diese an AndNav zurückgegeben und anschließend auf dem Display visualisiert. Der Benutzer kann nun seine Reise beginnen und mit Hilfe von AndNav durch das Straßennetz navigiert werden. Von besonderem Interesse bei dieser Routenanfrage ist die Weiterleitung der Daten von ORS zu OSM, da Letzteres das gesamte Kartenmaterial beinhaltet und somit die Qualität der Auswertung nachhaltig beeinflusst. Die Auswertung kann infolgedessen nur so gut sein, wie es das Kartenmaterial zulässt. Deshalb wird die Weiterleitung im nachfolgenden Abschnitt genauer erläutert.

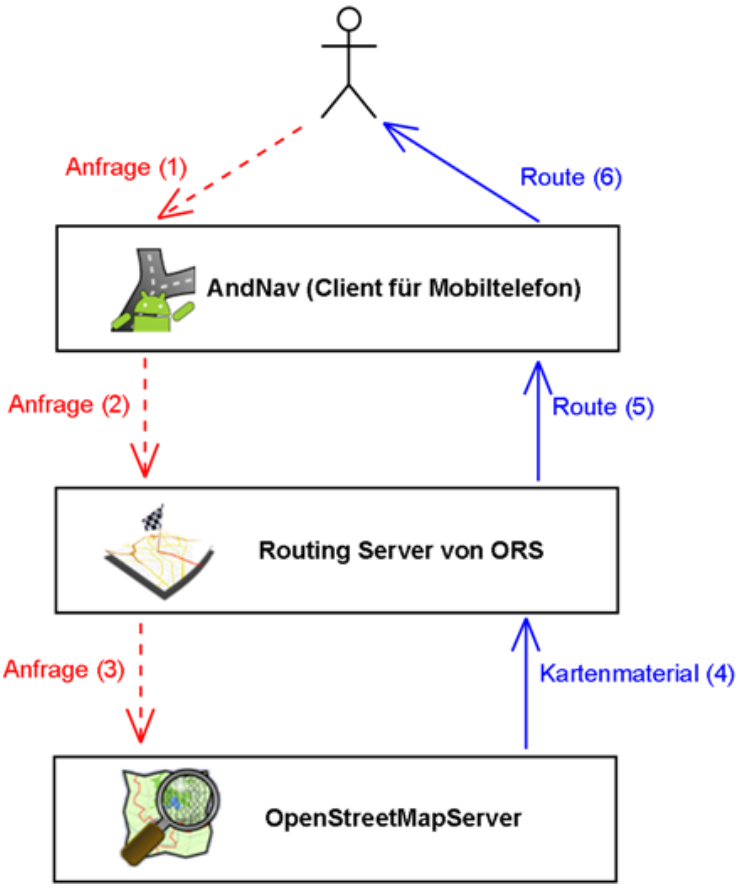

Abbildung 1: AndNav Kommunikationsmodell

Detaillierte Interaktion zwischen ORS und OSM

Das Kartenmaterial von OSM wird an ORS in Form von XML-Dateien übergeben und letzterer wandelt es in einen Navigationsfähigen Graphen um.

```
- <way id="23376829" user="theophrastos" uid="96069" visible="true" version="6" changeset="6334652" timestamp="2010-11-10T10:09:00Z">
    <nd ref="30812889" />
    <nd ref="30812890" />
    <tag k="footway:left:incline" v="0.5%" />
    <tag k="footway:left.sloped_curb.end" v="0.02" />
    <tag k="footway:left:smoothness" v="good" />
    <tag k="footway:left:surface" v="paving_stones" />
    <tag k="footway:left:width" v="1.20" />
    <tag k="footway:right:incline" v="0.6%" />
    <tag k="footway:right.sloped_curb.end" v="0.03" />
    <tag k="footway:right.sloped_curb.start" v="0.04" />
    <tag k="footway:right:smoothness" v="good" />
    <tag k="footway:right:surface" v="paving_stones" />
    <tag k="footway:right:width" v="1.20" />
    <tag k="highway" v="residential" />
    <tag k="maxspeed" v="30" />
    <tag k="name" v="Weberstraße" />
    <tag k="oneway" v="yes" />
  </way>
```

Abbildung 2: Weberstraße dargestellt im OSM XML-Datensatz

Abbildung 2 stellt einen Ausschnitt einer XML-Datei dar, welche die Weberstraße in Bonn abbildet. Die Informationen über die Straße sind klar über *Tags* [2] definiert. Der Tag „incline“ spielt für das Projekt eine wichtige Rolle. Incline definiert die Steigung, die in Form einer prozentualen Angabe gespeichert wird. Überdies gibt es noch weitere für dieses Projekt interessante Informationen, wie beispielweise „smoothness“, wodurch die Beschaffenheit der Straße bewertet wird und „surface“, das die Art des Straßenbelages bezeichnet. Derzeit liegt der Fokus dieses Projektes auf einer genauen Einschätzung des Energieverbrauchs unter der Berücksichtigung der Steigung. Für die Zukunft ist geplant, eine genaue Untersuchung des Rollwiderstandes durchzuführen, wodurch die Beschaffenheit und der Belag einer Straße durch den Algorithmus ebenfalls berücksichtigt werden sollen, wodurch die Berechnung der effizientesten Route genauer werden soll.

In der Abbildung 3 ist ein Ausschnitt eines Graphen zu sehen, der intern in der Software JOSM verwendet wird, einem Editor für OSM Daten (TODO Referenz für JOSM). Anhand der Abbildung soll veranschaulicht werden, wie eine XML-Datei von OSM als Graph bzw. als Straßennetz aussieht. Die Darstellung dieser Graphen ist im Gegensatz zu derer in ORS frei verfügbar. Die rote Linie entspricht im Graphen der Weberstraße, die in Abbildung 2 als XML spezifiziert ist, definiert wurde. Bei genauerer Betrachtung der Abbildung 3 sieht man, dass die Linie ein Pfeil ist, der nach links ausgerichtet ist. Die gelben Punkte sind Knoten. In OSM werden die Informationen zwischen den Knoten an den Kanten gespeichert. In der XML-Darstellung bilden die Punkte 30812889 und 30812890 den beschriebenen Abschnitt der Weberstraße.

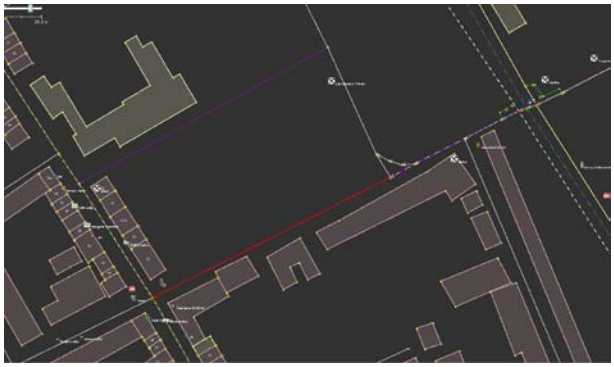

Abbildung 3: Weberstraße in JOSM

Abbildung 4: Weberstraße in ORS

Abbildung 4 visualisiert die Ansicht der Weberstraße in Kartenform auf openrouteservice.org.

3. E-Nav-Algorithmus

Im folgenden Abschnitt wird der von uns entwickelte Algorithmus vorgestellt, der die effizienteste Route bezüglich des Energieverbrauchs berechnet. Dazu wird auf den bereits bekannten Algorithmus A* [7] zurückgegriffen, welcher durch die hier vorgestellten Modifikationen so erweitert wird, dass er die gewünschte effizienteste Route berechnet.

```
1  /*
2   * Der Algorithmus bekommt als Eingabe ein Straßennetz als Graph. Dieser besitzt die
3   * Informationen über die Länge und in der Regel Informationen über die Steigung.
4   * Diese Informationen kommen alle von OpenStreetMap. Die zusätzlichen Informationen
5   * wie z.B. regularConsumption, start, target sowie maxIncline gibt der Nutzer bei
6   * AndNav ein und diese werden dann an den Server weitergeleitet.
7   */
8
9  Input: Map streetMap, int regularConsumption, Node start, Node target, int maxIncline
10 Algo E-Motor:
11
12 public Route mostEfficientRoute(Map streetMap, int regularConsumption, Node start, Node target, int maxIncline){
13     /*
14      * Berechnung des effizientesten Weges mit A*, indem die Kosten der
15      * Nachfolgerkante mit der Funktion consumptionOfEdge berechnet werden
16      */
17     A*(Graph graph, Node start, Node target);
18
19         if(path.incline > maxIncline){
20         //ignoriere Pfad mit höherer Steigung als maxIncline, da Weg zu Steil für Rollstuhl ist
21         }
22     /*
23      * A* durchläuft die Kanten und falls keine Steigung vorhanden ist, setze auf 0.
24      * A* gibt die effizienteste Route zurück. Folgende Abfrage muss hinzugefügt werden:
25      */
26         if(!inclineExists(Path path)){
27             path.setIncline (0);
28             // Der Pfad, welcher keine Informationen beinhaltet wird in einer Liste gespeichert
29             NoInfoPathList.add(Path path);
30         }return route;
31 }
32
33
```

```
33
34  /*
35   * Diese Methode berechnet für eine übergebene Route den Gesamtverbrauch,
36   * wird später auf Display ausgegeben
37   */
38  public double overallConsumption(Route route){
39      Node start = this.start;
40      Node end = this.end;
41      double sum = 0;
42      while(start != end){
43          sum += consumptionOfEdge(regularConsumption, start.edge.getLength(),
44          start.edge.getIncline());
45          start = start.getNext;
46      }return sum;
47  }
48
49  public double consumptionOfEdge(int regularConsumption, double incline, double pathLengthInKm){
50      double consumption = 0;
51      if(incline < -0,5){
52          consumption = regularConsumption * pathLengthInKm * 0,05;
53          return consumption;
54      }
55        else if(incline > -0,5 && incline <= 0){
56          consumption = regularConsumption * pathLengthInKm;
57      }
58        else{
59          consumption = regularConsumption * pathLengthInKm * 1,15^incline;
60      }
61  }
```

Abbildung 5: Source Code E-Nav

Ein Pseudocode des Algorithmus ist in Abbildung 5 dargestellt. Zunächst wird auf die Eingabe eingegangen und deren Bedeutung erläutert. Der Algorithmus bekommt von AndNav die folgenden Informationen: maximale Steigung, Normalverbrauch, einen Startknoten und einen Endknoten. Die maximale Steigung ist die größte Steigung, die der Elektromotor bewältigen kann. Diese Information ist aus dem Handbuch des Elektrofahrzeuges zu entnehmen. Der Normalverbrauch wird durch folgende Formel berechnet:

$$Normalverbrauch = \frac{MaximaleAkkukapazitätinAh}{\mathrm{Re}\,ichweiteunterNormalbedingungen}$$

Diese beiden Größen stehen ebenfalls im Handbuch. Das Kartenmaterial wird aus OSM entnommen. Unter Verwendung der vorgestellten Parameter wird anschließend der Algorithmus A* ausgeführt. A* ist eine Erweiterung des bekannten Dijkstra Algorithmus [8], welcher für die Berechnung eines kürzesten Weges benutzt wird. A* unterscheidet sich insofern von Dijkstra, dass eine Schätzfunktion für die Berechnung benutzt wird und somit häufig schneller eine Lösung gefunden wird. Die angewandte Heuristik für die Schätzfunktion ist in den meisten Fällen die Luftlinie zwischen dem Anfangs- und Endpunkt der Strecke. Damit die effizienteste Strecke berechnet werden kann, muss das Straßennetz so manipuliert werden, dass nicht mehr die Kantenlänge als Kriterium für die Auswahl verwendet wird, sondern der berechnete Verbrauch für jede Kante. In Abbildung 6 wird diese Modifikation durch die Funktion consumptionOfEdges vorgenommen. Diese unterscheidet drei Fälle:

1. negative Steigung (x<-0,5%)

   $$Energieverbrauch = \mathrm{Re}\,ichweite * Normalverbrauch * 0{,}05$$

2. neutrale Steigung (-0,5%<x<=0%)

   $$Energieverbrauch = \mathrm{Re}\,ichweite * Normalverbrauch$$

3. positive Steigung (x>0%).

   $$Energieverbrauch = \mathrm{Re}\,ichweite * Normalverbrauch * 1{,}15^{Steigung}$$

Nach der Auswertung dieser Funktion wird die Kantenlänge durch den entsprechenden Energieverbrauch ersetzt. Aufgrund von fehlenden oder nicht typkonformen Steigungsinformationen auf einigen Streckenabschnitten, anstatt Prozentangaben sind in OSM auch gelegentlich nur die Begriffe „Up“ und „Down“ an dem Tag incline auffindbar, findet eine Abfrage und Kontrolle der vorhanden Steigungsinformationen statt. Bei fehlenden Informationen wird die Steigung auf den Wert 0 gesetzt und die Kante wird in einer Liste vermerkt. Dies ist nötig, da der Graph vollständig sein muss, um A* über die veränderte Kostenfunktion laufen zu lassen. Diese Liste wird dazu verwendet, Aussagen über die Quality of Service zu machen. Dem Benutzer wird somit zusätzlich am Ende angezeigt, wie hoch die Relevanz der Auswertung ist. Die Relevanz wird ermittelt, indem ein prozentualer Anteil der vorhandenen Streckeninformationen angezeigt wird.

Eine Besonderheit ist, dass das Ergebnis einer einzigen Routenanfrage nicht aus einer, sondern aus zwei Routen besteht. Einerseits die kürzeste Route, samt Energieverbrauch und Relevanz, die mithilfe des nicht erweiterten A* berechnet wird und andererseits die effizienteste Route, die durch den im oberen Abschnitt beschrieben Vorgang mithilfe des erweiterten A* Algorithmus berechnet wird. Somit kann der Benutzer die beiden Routen vergleichen und anhand der Angaben (z.B. Quality of Service bei der effizientesten Route) eine auswählen. Es besteht kein direkter Zusammenhang zwischen der kürzesten und der effizientesten Route, weshalb diese sich auch unterscheiden können. Um den Unterschied zwischen einer kürzesten und einer effizientesten Route zu erläutern, wird in dem folgenden Beispiel ein Graph beschrieben, der ein Straßennetz darstellen soll. Dieses wird durch die Abbildungen 6 unterstützt und graphisch veranschaulicht.

Rechts ist ein modelliertes Straßennetz abgebildet, wobei die Knoten Wegpunkte darstellen und die Kanten die Straßenabschnitte zwischen den Punkten. Oberhalb der Kanten sind die Streckenlänge sowie die Steigung vermerkt. Unterhalb stehen Angaben zum Verbrauch des Akkus in Ah. Angenommen eine Person möchte von A nach G fahren und am selben Tag noch wieder zurück. Unter der Voraussetzung dass diese den kürzesten Weg nehmen möchte, hat die Berechnung ergeben, dass die Route von A nach G über C mit 3,7 km am kürzesten ist (blauer Pfad) und einen Verbrauch von 118,43 Ah verursacht. Diese Route ist allerdings nicht die effizienteste. Der effizienteste verläuft von A → D → E → G (grüner Pfad). Der Verbrauch für die Fahrt trägt 51,1 Ah und die Länge ist ungefähr 5,4 km. Auf dem Rückweg steht die Person wieder vor der Wahl zwischen der effizientesten und der kürzesten Route. Wie im Graph zu erkennbar, bleibt die kürzeste die Gleiche, jedoch in umgekehrter Richtung. Der Verbrauch ändert sich auf 21,6 Ah. Im Vergleich dazu ändert sich die effizienteste Route. Diese verläuft jetzt über G → F → B → A. Der Verbrauch liegt bei 2,15 Ah. Allerdings beträgt die Länge von knapp 8,6 km. Besonders auf dem Rückweg ist ein extremer Unterschied zu verzeichnen,

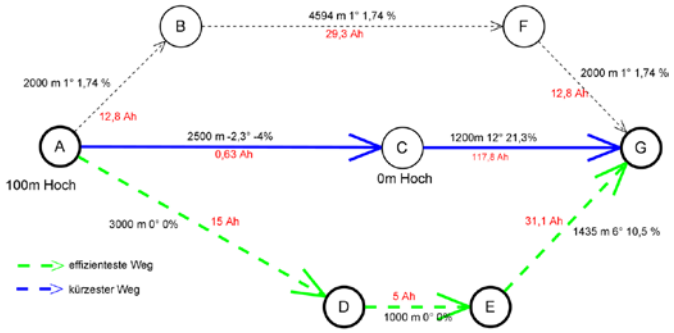

Abbildung 6: Straßennetz Hinweg

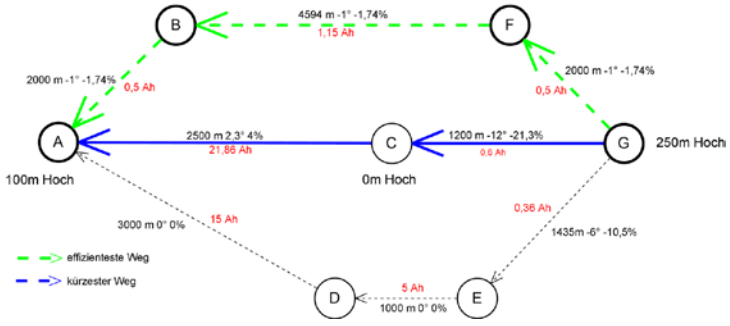

Abbildung 7: Straßennetz Rückweg

sowohl bezüglich des Verbrauchs als auch der Streckenlänge. Wie aus diesem Beispiel hervorgeht besteht ein Trade-Off bezüglich der Effizienz und der Streckenlänge. Aus diesem Grund wird die Entscheidung dem Benutzer überlassen und von AndNav (E-Nav) beide Routen angeboten.

4. Fazit

Der in dieser Arbeit vorgestellte Algorithmus bietet eine Möglichkeit durch eine Modifikation des A*-Algorithmus den effizientesten Weg bezüglich des Energieverbrauchs von Elektromotoren zu berechnen. Er wird zurzeit in den bestehenden ORS implementiert und anschließend online zur Verfügung gestellt.

Die nächste Überarbeitung des Algorithmus sieht vor, dass die unterschiedlichen Konfigurationen (z.B. Akku lädt sich auf während des Bremsens) von Elektrofahrzeugen bei der Berechnung der effizientesten Route berücksichtigt werden. Hierzu soll der Algorithmus parametrisiert werden, sodass das Verhalten von gängigen Elektrofahrzeugen durch die Übergabe bestimmter Parameter detaillierter modelliert werden soll und beim Algorithmus zu einem exakteren Ergebnis führt.

Außerhalb des Algorithmus gibt es jedoch noch weitere Kritikpunkte, die parallel zur Entwicklung des Algorithmus angegangen werden. Zum Beispiel ist die Erfassung des Akkustandes von Elektromotoren insbesondere bei Elektrorollstühlen oftmals ungenau. Daher arbeiten wir als Institut für Eingebettete Systeme zurzeit an einer Hardwarenahen Lösung für dieses Problem. Desweiteren sind statistische Daten, die den Energieverbrauch eines Elektrofahrzeuges in Abhängigkeit von der Steigung darstellen rar. Solche Daten benötigt man, um z.B. die oben angegebenen Funktionen und Fallunterscheidungen ableiten zu können. Für die angegebenen Funktionen haben wir einen Elektrorollstuhl entsprechend charakterisiert. Neben statistischen Daten mangelt es zudem oft an ausreichend Steigungsangaben auf der gesuchten Route. Wir arbeiten zurzeit an mobilen Applikationen für Mobiltelefone mit GPS, welche die GPS-Daten mit Steigungsinformationen mit Einverständnis des Benutzers anonym automatisch in OSM hochladen und einpflegen. In Anbetracht der zunehmenden Relevanz von Elektrofahrzeugen in den vergangenen und kommenden Jahren sehen wir in diesem Projekt interessante Herausforderungen und großes Potential.

Diese Arbeit wurde durch das UMIC Research Centre, RWTH Aachen Universität, unterstützt.

Kontakt zu den Autoren:

Dominik Franke
RWTH Aachen Universität
Ahornstr. 55
52074 Aachen
0049 241 8021172
franke [at] embedded.rwth-aachen.de

Dzenan Dzafic
RWTH Aachen Universität
Matschö-Moll-Weg 11
52064 Aachen
0049 173 5720384
dzenan.dzafic [at] rwth-aachen.de

Carsten Weise
RWTH Aachen Universität
Ahornstr. 55
52074 Aachen
0049 241 8021159

Stefan Kowalewski
RWTH Aachen Universität
Ahornstr. 55
52074 Aachen
0049 241 8021152

weise [at] embedded.rwth-aachen.de kowalewski [at]embedded.rwth-aachen.de

Literaturverzeichnis

[1] Ramm, Ferderik und Topf, Jochen. *OSM.* Berlin : s.n., 2008.

[2] OpenStreetMap. *http://wiki.OpenStreetMap.org.* [Online] [Abruf: 28. 01 2011.] http://wiki.OpenStreetMap.org/wiki/Category:Keys.

[3] Wiki. *wikimedia.org.* [Online] [vom: 28. 01 2011.] http://upload.wikimedia.org/wikipedia/commons/5/5b/Steigung_in_Prozent.png.

[4] OpenRouteService. *openrouteservice.org.* [Online] [Abruf: 28. 01 2011.] http://openrouteservice.org/

[5] Android. *android.com.* [Online] [Abruf: 28. 01 2011.] http://www.android.com/.

[6] AndNav. *andnav.org.* [Online] [Abruf: 28. 01 2011.] http://www.andnav.org/.

[7] HART, P.E., NILSSON, N.J., RAPHAEL, B. *A Formal Basis for the Heuristic Determination of Minimum Cost Paths*. IEEE (Hrsg), Transactions on Systems Science and Cybernetics. Menlo Park, California, USA, 1968

[8] DIJKSTRA, E.W.: *A note on two problems in connexion with graphs*. In: *Numerische Mathematik*. 1, 1959, S. 269–271

[9] MÜLLER, A, NEIS, P., ZIPF, A.: : *Ein Routenplaner für Rollstuhlfahrer auf der Basis von OpenStreetMap- Daten - Konzeption, Realisierung und Perspektiven.* AGIT 2010. Symposium für Angewandte Geoinformatik. Salzburg. Austria.