
Eco-Routing auf Smartphones für Elektrofahrzeuge

Bernd DREIER, Uli GÖHNER, Andreas FISCHER und Robert SCHMID
Hochschule Kempten · bernd.dreier@hs-kempten.de

Zusammenfassung

Eine der größten Barrieren der heutigen Elektrofahrzeuge ist ihre limitierte Reichweite. Obwohl die Mehrheit der Autofahrer nicht mehr als 40 Kilometer pro Tag fährt (MOBILITÄT IN DEUTSCHLAND 2008), ist das Kaufverhalten für Elektrofahrzeuge mit viel kleinerer Reichweite als solcher mit Verbrennungsmotoren sehr zurückhaltend. Mit dieser Arbeit adressieren wir die zusätzlichen Informationsbedürfnisse der Fahrer von E-Fahrzeugen durch die Entwicklung einer Applikation für Standard-Smartphones, welche sowohl eine detaillierte und genaue grafische Repräsentation der Reichweite des Elektrofahrzeugs ermöglicht, als auch einen fortgeschrittenen Routing-Algorithmus implementiert, der den spezifischen Energieverbrauch des Elektrofahrzeugs berücksichtigt.

1 Besonderheiten der verwendeten Plattform

Die hier beschriebene Softwareplattform wird seit 2009 im Rahmen der Forschungsprojekte ee-tour (ERGEBNISBUCH EE-TOUR ALLGÄU 2011) und e-connect (VERBUNDVORHABEN „ECONNECT GERMANY“ 2013) an der Hochschule Kempten weiterentwickelt. In diesen Projekten wurden verschiedenste Fahrzeugtypen mit jeweils völlig unterschiedlichen, meist fest verbauten, Steuersystemen verwendet. Auch die Art und Weise der Nutzung deckte zwischen Pendler, Flughafentransfer und Ausflugsfahrt im Urlaub die unterschiedlichsten Profile ab. Daher waren für die zu entwickelnde Plattform folgende Punkte besonders wichtig:

- Unabhängigkeit von Datennetzen, da die Alpenregion eine schlechte Netzabdeckung hat und eine Routen- und Reichweitenberechnung an jedem Ort zu jeder Zeit möglich sein muss, auch grenzüberschreitend.
- Verwendung in verschiedensten Fahrzeugtypen.
- Unabhängigkeit von Kartenmaterial Dritter, da für die exakten Berechnungen an verschiedensten Stellen in die Algorithmen und auch in die Datenhaltung eingegriffen werden muss.

1.1 Entwicklung auf Google Android mit Offlinekarten

Auf Basis der obigen Überlegungen bot sich besonders Google Android an, das sowohl für verschiedenste Mobiltelefone als auch für Tablets zur Verfügung steht. Da die Anwendung unabhängig von Datennetzen sein soll und wir das Kartenmaterial durch zusätzliche Daten anreichern (s. u.), arbeitet unsere Anwendung ausschließlich mit Offlinekarten. Allerdings erfordert dieser Ansatz auch mehr benötigten Speicherplatz auf dem mobilen Gerät, welcher bei aktuellen Generationen an sich ausreichend vorhanden ist. Dennoch ergeben sich

für die Entwicklung einige Einschränkungen, insbesondere im Bezug auf die verwendbare Programm-Heap-Größe und den Arbeitsspeicher. So ist es mit dem Android-SDK nur möglich 16 MB Heap-Speicher pro Applikation zu allozieren. Es ist daher besonders wichtig ressourcensparende Algorithmen zu entwickeln.

2 Routen- und Reichweitenberechnung

2.1 Routing-Graph

Die Anwendung verwendet das in Java geschriebene Routing-Framework „graphhopper“¹, dessen kompaktes Routing-Graph-Format auf OSM² Daten basiert. Da OSM momentan keinerlei Höheninformationen mitführt, müssen diese zusätzlich beschafft und in das Framework integriert werden. Für die hier beschriebenen Anwendungen wurden die kostenfrei verfügbaren Höhendaten SRTM³ heruntergeladen und als zusätzliches Attribut für die Graph-Knoten hinzugefügt.

Der Routing-Graph besteht aus Knoten (Nodes) welche über Kanten (Edges) miteinander verbunden sind. Jeder Knoten setzt sich aus einer GPS-Koordinate und der Höhe zusammen. Eine Kante kennt ihre beiden Endpunkte, die Richtungen in der sie durchfahren werden darf, ihre Länge in Metern und die Geschwindigkeit die durchschnittlich auf ihr gefahren wird in m/s als Kanten-Gewichte. Abbildung 1 veranschaulicht die Datenstruktur anhand von Beispieldaten.

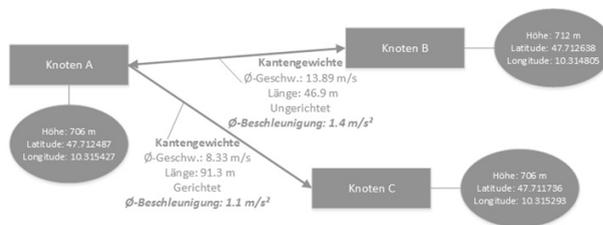


Abb. 1: Graph-Struktur mit Beispieldaten. Fett gedruckte Gewichte werden derzeit durch Kartendatenanreicherung hinzugefügt

2.2 Theoretisches Energieverbrauchsmodell

Für eine verlässliche Reichweitenprognose des Fahrzeugs ist ein exaktes Energieverbrauchsmodell unerlässlich, das die voraussichtlich benötigte Energiemenge abhängig von bestimmten Umweltfaktoren, dem Fahrzeug selbst und dem individuellen Verhalten des Fahrers möglichst genau vorhersagt. Die Energie aus der Hauptbatterie des Elektrofahrzeugs wird während der Fahrt dazu benötigt, den Fahrwiderstand zu überwinden. Dieser setzt sich zusammen aus vier Komponenten: Luft-, Roll-, Beschleunigungs- und Steigungswiderstand. Letztere beiden können in allen Berechnungen mit negativen Vorzeichen (Rekuperation) vorkommen.

¹ Siehe <http://graphhopper.com>

² Siehe <http://www.openstreetmap.de>

³ Shuttle Radar Topography Mission (SRTM) im Februar 2000 unter Beteiligung der DLR

Im derzeitigen Entwicklungsstand haben wir einige Vereinfachungen angenommen: Temperatur- sowie alterungsabhängige Einwirkungen auf die Kapazität der Batterie sind in der Reichweitenberechnung lediglich pauschal vorgesehen und werden derzeit von einem „Schwesterprojekt“ an der Hochschule Kempten genauer untersucht (ARMSTORFER 2013). Auch Regen und Schnee trägt zu verändertem Fahrverhalten des Fahrers und Rollwiderstand bei. Diese Komponenten wurden aktuell ebenfalls außer Acht gelassen, d. h. alle Messwerte und Annahmen legen nahezu trockene Fahrbedingungen zugrunde. Als am schwersten vorherzusehender Faktor gestaltet sich die Annahme über das Verhalten des Fahrers. Neben stark veränderlichen Beschleunigungswerten erlauben einige Fahrzeuge Bremswirkung durch Rekuperation. Schon ohne die Berücksichtigung dieser Art der Energierückgewinnung erzielen wir allerdings sehr gute Näherungen, was sich mit den Erkenntnissen spezifischer Untersuchungen zu diesem Gebiet deckt, Rekuperation in der Praxis als verhältnismäßig wenig relevant einstufen (BRÜCKNER 2011)

2.3 Berechnung und Anzeige einer Reichweitenkarte

In einer Straßenkarte werden dem Benutzer auf Basis der OSM-Kartendaten drei Polygone angezeigt. Die Außenkanten dieser Polygone beschreiben die maximale Reichweite des Elektrofahrzeugs bei verschiedenen Fahrprofilen. Wir unterscheiden dabei zwischen den drei Fahrstilen „sportlich“, „durchschnittlich“ und „energieeffizient“. Abbildung 2 zeigt die beschriebene Reichweitenkarte in Graustufen (in der Anwendung natürlich farbig).

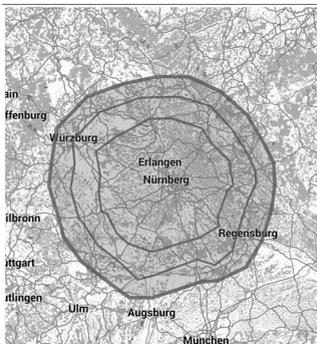


Abb. 2:

„Spiegeleikarte“ mit den Reichweiten für die verschiedenen Fahrstile „sportlich“, „durchschnittlich“ und „energieeffizient“ (von innen nach außen)

Damit die Reichweitenkarte schnell berechnet werden kann, müssen möglichst schnelle Routingtechniken und –Algorithmen eingesetzt werden. In diesem Fall wurden „Contraction Hierarchies“ (nach GEISBERGER 2008) mit bidirektionalen Routingalgorithmen eingesetzt, um das Berechnen von Pfaden in einem Graphen maßgeblich zu beschleunigen. Hierbei wird die Reichweitenkarte immer von einer Ausgangskoordinate M (i. d. R. von der Position des Fahrzeugs) aus berechnet. Um diesen Punkt werden kreisförmig neue Koordinaten mit einem Radius $r = 110\%$ der maximalen Reichweite des Fahrzeugs in km generiert (siehe Abb. 3). Anschließend sind alle Routen vom Ausgangspunkt zu diesen Koordinaten zu berechnen. Um überhaupt eine Aussage über die verbrauchte Energie treffen zu können, muss ein Simulator geschaffen werden, der virtuell eine Route abfährt und mithilfe des in Kapitel 4.4.6 beschriebenen Modells den Energieverbrauch vorausberechnen kann. Dieser Simulator in Form eines virtuellen Fahrzeugs liefert dann die Graph-Kante auf der die Energie des Autos voraussichtlich verbraucht sein wird. Genauer werden folgende Schritte durchlaufen:

Kreis-Vertices berechnen

Leider sind Kreisberechnungen in einem GPS-Koordinatensystem nicht trivial, da seine beiden Achsen abhängig von der elliptischen Form des Planeten unterschiedlich skaliert sind. An der Koordinate 47.0,10.0 beträgt beispielsweise ein Schritt von 0.1 Längengraden eine Distanz von ca. 11 Kilometern, wohingegen ein gleich großer Schritt in Breitengradrichtung eine Distanz von ca. 7.5 Kilometern beträgt. Um dennoch einen konstanten Radius r für alle Kreispunkte zu erhalten, findet hier die Haversine-Formel Anwendung. Mit dieser Formel lässt sich ausgehend von einer gegebenen GPS Koordinate unter Angabe eines Winkels und einer Distanz in Kilometern die korrekte GPS Koordinate errechnen. Alle errechneten Kreispunkte werden anschließend auf ihren nächstgelegenen Graph-Knoten verschoben ebenso wie der Ausgangspunkt der Reichweitenkarte. Abbildung 3 zeigt das Ergebnis des beschriebenen Verfahrens.

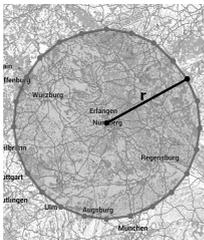


Abb. 3: Kreis definiert durch auf Graph-Knoten gelegte Vertices mit Abstand r zum Ausgangspunkt

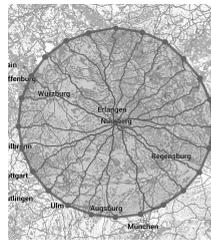


Abb. 4: Berechnete Routen vom Ausgangspunkt zu den vorläufig errechneten Kreispunkten

Wie viele Kreispunkte letztendlich generiert werden, wird durch den Auflösungsparameter n bestimmt. Je größer n , desto genauer die Reichweitenkarte und desto länger die Berechnungszeit.

Routenberechnung

Verteilt auf mehrere asynchrone Threads können unabhängig voneinander die Routen vom Ausgangspunkt der Karte zu den vorläufig berechneten Kreis-Vertices berechnet werden, wie in Abbildung 4 gezeigt. Diese Routen dienen als Input für das theoretische Energieverbrauchsmodell. Beim Routen wird zwischen verschiedenen Routentypen differenziert. Der Benutzer kann zwischen kürzester, schnellster und energieeffizientester Kalkulation wählen. Zur Berechnung werden verschiedene Kantengewichte herangezogen, wie etwa die Distanz, die Energie oder die Zeit. Um die bestmögliche Route zu erhalten, müssen präzise Gewichte verwendet werden. Diese erhält man durch die Anreicherung von Kartendaten (siehe Kapitel 3.2).

Tatsächliche Kartengrenzpunkte berechnen

Die berechneten Routen werden von virtuellen Fahrzeugen, deren Parameter entsprechend der Realität nachempfunden wurden (siehe Kapitel 2.2), „nachgefahren“. Dabei wird die pro Graph-Kante benötigte Energie dem Ladezustand der virtuellen Batterie des Fahrzeugs abgezogen. Initial wird der Batteriestand auf den realen Ladestand des Fahrzeugs gesetzt. Erreicht die Simulation eine Kante, welche den Ladestand der Batterie auf 0 sinken lässt, so wird der Eintrittsknoten dieser Kante als Vertex für das Reichweitenpolygon verwendet (siehe Abb. 2). Für die Polygone der beiden anderen Fahrstile „sportlich“ und „durchschnittlich“ werden analog erneut virtuelle Fahrzeuge über die bereits berechneten Routen „geschickt“. Dadurch ergeben sich die in Abbildung 2 ersichtlichen Abstufungen.

Die Berechnungszeit für die Reichweitenkarte beträgt auf einem Samsung Galaxy Note II mit einer maximalen Fahrzeugreichweite von 130 Kilometern, 100 % Akkuladestand, einem Auflösungsfaktor $n=24$ und 3 parallel arbeitenden Threads in etwa 7 Sekunden, was bei einem großen Updateintervall durchaus vertretbar ist.

3 Aktuell laufende Verbesserungen

3.1 Mehrkriterienoptimierung

Grundsätzlich gilt es zu bedenken, dass, anders als in herkömmlichen Fahrzeugen, im Elektrofahrzeug selten die schnellste oder kürzeste Route gewählt werden wird. Allerdings wohl auch niemals die energieeffizienteste, da sich aus dem theoretischen Energiemodell offensichtlich ergibt, dass diese Route von möglichst niedrigen Geschwindigkeiten dominiert wird, also z. B. alle Tempo-30-Zonen durchfahren würde. Gesucht ist also eine Route, die eine sinnvolle Kombination der Attribute kurz, schnell und energiesparsam bietet.

Unser Ansatz hierzu basiert auf Methoden der genetischen Algorithmen, die aus einer Menge existierender Routen verbesserte Lösungen generieren können (YUSSOF 2011). Der Ablauf ist in etwa wie folgt: Zunächst wird ein Pool von Lösungsvorschlägen für das Routingproblem angelegt. Beim Start werden in diesen Pool mehrere Routen, die mit bekannten Standardverfahren gefunden wurden, mit ihrem Energieverbrauch eingebracht. Dazu zählen z. B. die schnellste und die kürzeste Route. Aus diesem Pool werden zwei Routen ausgewählt, die einen gemeinsamen Teilstreckenverlauf besitzen. Innerhalb des gemeinsamen Streckenteils wird ein Punkt ausgesucht, der als Kreuzungspunkt dient. Durch die Kombination der Streckenteile entstehen zwei neue Routen (Crossover). Auch diese Routen werden dem Pool hinzugefügt. Im folgenden Schritt entsteht mit einer gewissen Wahrscheinlichkeit aus jeder bis hierher verwendeten Routen eine abgeänderte Variante (Mutation). Hierzu werden die Streckeneigenschaften untersucht und Bereiche identifiziert, die sich merklich negativ in der Energiebilanz niederschlagen. Durch Einfügen einer Streckenumleitung (z. B. Umfahren von Bergen) wird versucht, an dieser Stelle weniger Energie zu verbrauchen. Die leicht veränderte Route wird wieder energetisch bewertet und dem Routenpool eingefügt. Zum Abschluss werden nur diejenigen Routen übernommen, die bezüglich Zeit und benötigter Energiemenge pareto-optimal sind (Fitness). Nach einer fixen Anzahl von Iterationen des Gesamtzyklus wird aus der Pareto-Front eine Route gewählt, die die gewünschten Eigenschaften in Bezug auf die Kriterien Zeit und Energieverbrauch bietet. Dieser Algorithmus wurde bisher an ausgewählten Testfahrten erprobt und liefert für die untersuchten Strecken die für Ortskundige sinnvollen bzw. erwarteten Strecken.

3.2 Anreicherung von OSM-Kartendaten und Validierung der Energiemodelle

Mit der Aufzeichnung einer gefahrenen Strecke ist es möglich, eine Nachberechnung der Energiedaten anzustellen und daraus resultierende Diskrepanzen zu analysieren. Bei der Auswertung werden real gefahrene GPS-Daten unter Verwendung eines Mapmatchers auf den Routinggraphen projiziert und die gefundenen Kanten mit deren Energieverbrauch ausgegeben. Da wir dieses Mapmatching auch für Echtzeitfahrten verwenden, haben wir den in (ZHU 2011) beschriebenen Algorithmus implementiert, der nur die vergangenen

GPS-Punkte verwendet. Für die reine Analyse bereits abgeschlossener Fahrten wären einfachere Algorithmen ausreichend. Diese können mit den realen Energieverbräuchen verglichen werden. Aus den oben genannten Projekten liegen einige Hunderttausend-Fahrkilometer vor, die im Moment zur Validierung unserer bisherigen Ergebnisse untersucht werden.

Dies ist auch der erste Schritt hin zu einer Datenanreicherung um unseren Routing-Graphen iterativ zu verbessern. Dabei werden real gemessene Werte, wie Zeit-, Höhen- und Geschwindigkeitsdaten sowie die durchschnittliche Beschleunigung der Fahrer auf einer Kante, als Kantengewichte (siehe Abb. 1) in den Routing-Graphen gespeichert, um in Zukunft noch genauere Vorausberechnungen anstellen zu können. Auch Daten wie Fahrbahnbeschaffenheit, Kurvenradien in Teilstücken könnten hierzu verwendet werden. Diese Daten könnten auch automatisiert über Smartphones aufgezeichnet und in die Karten zurückgespielt werden.

4 Ausblick

Die Anreicherung der Kartendaten bzw. Routinggraphen mit verschiedensten Kantengewichten steht in nächster Zeit im Fokus unserer Arbeit, um noch genauere Prognosen treffen zu können. Außerdem arbeiten wir an einer adaptiven Fahrerklassifikation, mit deren Hilfe der Energieverbrauch abhängig vom Fahrstil des jeweiligen Fahrers in Echtzeit berechnet wird. Der Abgleich mit der Vorausberechnung erlaubt es dann, dem Fahrer Feedback zum Fahrtverlauf sowie Hinweise auf Abweichungen und deren Ursache zu geben.

Literatur

- ARMSTORFER, A., BIECHL, H., ALT, B., MÜLLER, H., SOLLACHER, R., MOST, D., SZABO, A., KÖBERLE, R. & FIEDELDEY, M. (2013), Internationaler ETG-Kongress 2013 „Einsatz von Batteriespeichern in Smart Grids“.
- BRÜCKNER (2011), Betrachtungen zum Energiehaushalt eines Elektrofahrzeugs. Tag der Forschung, HTW Dresden.
- ERGEBNISBUCH EE-TOUR ALLGÄU (2011), „iKT als Wegbereiter für die Elektromobilität“.
- GEISBERGER, R., SANDERS, P., SCHULTERS, D. & DELLING, D. (2008), Contraction Hierarchies: Faster and Simpler Hierarchical Routing in Road Networks. *Experimental Algorithms*, LNCS 5038.
- MOBILITÄT IN DEUTSCHLAND, KURZBERICHT (2008), infas.
- VERBUNDVORHABEN „ECONNECT GERMANY“ (2013), Zwischenbericht.
- YUSSOF, S., RAZALI, R. A. & SEE, O. H. (2011), An Investigation of Using Parallel Genetic Algorithm for Solving the Shortest Path Routing Problem. *Journal of Computer Science*, 7 (2), 206-215.
- ZHU, Z., XING, J., MENG, L., WU, Y., & LI, F. (2011), SRMM: Simple Real-Time Map-Matching Model and Algorithm in Urban Area Traffic Integrated Application. In *Intelligent System Design and Engineering Application (ISDEA)*, 2010 International Conference on, 2, 395-398.