Automatische Ampelphasenberechnung durch kollektive Geodatenerfassung für die Prädiktion der optimalen Geschwindigkeit zum Durchfahren der Grünphase

Daniel STEINER und Erich HARTLIEB
FH Kärnten · daniel.steiner@edu.fh-kaernten.ac.at

Zusammenfassung

Durch den zunehmenden Verkehr in heutigen Städten werden immer kreativere Lösungen benötigt, um trotz einer stetig steigenden Zahl von Fahrzeugen weiterhin Mobilität für jedermann zu gewährleisten. Bedingt durch die Fortschritte der Kommunikationstechnik wird die Vernetzung und Virtualisierung der realen Welt immer alltäglicher. Die Auswirkungen einer derart tief greifenden Integration von Informationstechnologie in unserem Alltag, bei welchem der Computer als sichtbares Gerät verschwindet, sind bisher noch kaum abzusehen (MATTERN 2003). Fakt ist jedoch, dass solche Systeme maßgeblich die Lebensqualität in verschiedensten Bereichen beeinflussen können. Zum Beispiel unterscheidet sich die heutige Fahrzeugtechnik stark von der letzten Generation durch die Einbettung von Fahrerassistenzsystemen, die den Fahrer bei der Fahrzeugführung entlasten sollen. Heutzutage ist es kaum vorstellbar, ein Auto ohne irgendwelche elektronischen Hilfssysteme zu steuern. Sei es ein Elektronisches Stabilitätsprogramm (ESP), ein Anti-Blockier-System (ABS), ein Navigationsgerät oder ein Tempomat, all diese Module sind maßgeblich für den Komfort und vor allem für die Sicherheit entwickelt worden. Die Herausforderung liegt vor allem in der Beherrschung der Datenflut, die hierbei anfällt. Speziell innerstädtische Verkehrssituationen zeigen, dass die hohe Komplexität und Vielfalt große Anforderungen an solchen Systemen stellen (SCHRÖDER 2009).

Ein Ansatz zur verbesserten Verkehrseffizienz ist ein intelligenteres Fahrverhalten an Ampeln: Wenn die Schaltzeiten von Ampeln bereits im Voraus bekannt sind, können viele Standzeiten und damit „Stop-and-Go“ Wellen im fließenden Verkehr verhindert werden. Hierdurch lassen sich wiederum große Energie- bzw. Treibstoffeinsparungen erzielen. Im Gegensatz zu zentralen Systemen, die den Gesamtverkehrsfluss „von oben“ betrachten, liegt der Fokus bei diesem Projekt am einzelnen Fahrzeug. Gegenwärtige Technologien ermöglichen eine Erfassung von Positionsdaten direkt während der Fahrt. Daraus können anschließend Muster gefiltert werden, die dem Fahrer in Echtzeit mitgeteilt werden, um einen geregelteren Verkehrsfluss zu gewährleisten.

1 Zielsetzung

Der zentrale Fokus dieser Arbeit ist die „*Minimierung von Wartezeiten und damit des Treibstoffverbrauchs an Ampeln mit fließendem Übergang in vernetzte Infrastrukturen*

Strobl, J., Blaschke, T., Griesebner, G. & Zagel, B. (Hrsg.) (2013): Angewandte Geoinformatik 2013.
 ISBN 978-3-87907-533-1.

durch automatische, verteilte Erfassung der Schaltzeiten pro Ampel". Mit der Umsetzung dieses Projekts ist es möglich, aus GPS-Trajektorien Muster zu erkennen und ohne zusätzliche Hilfsdatensätze Rückschlüsse auf Ampelpositionen bzw. deren Haltelinien zu erzielen. Als Grundszenario geht man davon aus, dass eine möglichst hohe Anzahl an Clients ihre Fahrstrecken aufzeichnen und diese anschließend an einen Backend übertragen. Dieser sammelt die Daten kontinuierlich ein und legt sie auf einer räumlichen Datenbank ab. Gleichzeitig werden die gesamten Routen mit ihren geloggten GPS-Positionen ständig analysiert und mögliche Haltelinien herausgefiltert. Da jeder aufgezeichnete Punkt die Koordinaten, Zeitstempel, Genauigkeit und Geschwindigkeit speichert, kann man mittels Geschwindigkeit-Zeit-Diagrammen charakteristische Muster auswerten. Nachdem die Tracks feststehen, werden die Segmente vor der Haltelinie ausgewertet, um darauf Rückschlüsse zur Wartezeit zu erzielen. Die gewonnenen Daten werden den Clients zugänglich gemacht, welche abhängig von ihrem Standort die verbleibende Zeit der nächsten Grünphase auf ihrem Smartphone angezeigt bekommen. Außerdem wird auch die ideale Geschwindigkeit zum Durchfahren der Grünphase dargestellt, die abhängig von der Position des Autos und dessen Abstand zur Ampel ständig neu berechnet wird. Nach dem heutigen Stand der Technik ist dieses Szenario gerade aufgrund der immer weiter steigenden Absätze von Smartphones interessant, auch wenn es in der Art keine vergleichbare Lösung gibt. Andere Systeme arbeiten unter anderem mit Bilderkennung oder durch Kommunikation mit der Infrastruktur (BRAUN et al. 2009, KOUKOUMIDIS et al. 2011). Dieses hier vorgestellte System ist im Vergleich dazu aber komplett dezentral, sodass keine zusätzlichen Kosten für Endverbraucher und Infrastruktur entstehen. Die Herausforderung liegt hier vor allem in der serverseitigen Implementierung. Die große Menge an Daten, welche am Server ankommt, muss auf entsprechende Muster analysiert werden. Dabei gilt es, auch bei wenigen Daten, die für einen Straßenabschnitt aufgezeichnet wurden, sinnvolle Aussagen über Genauigkeit treffen zu können. Zudem müssen alle errechneten Haltelinien so lange mit weiteren Datensätzen evaluiert werden, bis diese die erforderlichen Genauigkeits- und Qualitätskriterien erfüllen. Auch zur Evaluierung der Ampelphasen muss im Hintergrund kontinuierlich ein Datenabgleich mitlaufen, zudem muss überlegt werden, wie neben der Auswertung von statischen Ampelschaltzeiten die Speicherung von dynamischen Schaltzeiten erfolgen kann.

2 Problemstellung

Es gibt, wie vorhin schon erwähnt, schon einige Projekte, die sich mit der Vorhersage der Ampelphasen beschäftigen. Im Vergleich zu diesen Systemen bekommen die Clients die Informationen aber durch Geräte, die mit ihrer Umgebung kommunizieren und alle relevanten Daten schon zur Verfügung haben. Ein Beispiel wäre hierbei eine Ampel, die über WLAN ihre genauen Schaltzeiten in einem bestimmten Umkreis aussendet. Diese Variante ist äußerst effizient und kaum fehleranfällig. Der Nachteil ist jedoch, dass die Erstellung einer solchen Infrastruktur sehr kostenintensiv ist. Der Ansatz, die Schaltzeiten mithilfe der Clients zu lernen, funktioniert im Gegensatz dazu komplett dezentral. Die große Herausforderung ist aber, mit der Datenflut zurechtzukommen und nur die wichtigsten Informationen zu filtern. Außerdem muss das System garantieren, dass die berechneten Zeiten validiert sind und der Realität entsprechen, da sonst eine Vorhersage einer optimalen Geschwindigkeit keinen Sinn ergeben würde und in weiterer Folge sogar Unfälle auslösen könnte. Um

dem Ansatz nachzugehen, wie man mögliche Ampelpositionen und –phasen aus einer Vielzahl von GPS-Trajektorien herausfiltern kann, müssen zuerst grundlegende Fragen beantwortet werden:

- Wo sind Ampeln positioniert?
- Wie kann man einen Ampelpunkt (=Haltelinie) definieren?
- Wie ermittelt man die Ampelstehzeit?
- Wie berechnet man die optimale Geschwindigkeit?

Die Ampelpunkte werden in diesem Projekt mit Haltelinien gleichgesetzt, weil diese Position der vorderste mögliche Haltepunkt eines Fahrzeuges bei einer Ampel ist. Im Mittel aus allen Punkten der GP-Trajektorien muss somit das in Abbildung 1 gezeigte Muster erkennbar sein: Die Ampel schaltet auf Rot um, der Fahrer wird langsamer und kommt bei der Haltelinie zum Stehen. Sobald die Ampel auf Grün schaltet, fährt er wieder an. Allgemein bedeutet es, dass sich die Haltelinienposition genau auf dem Punkt befindet, auf dem das Fahrzeug wirklich zum Stillstand gekommen ist.

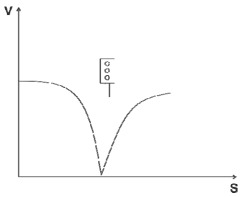

Abb. 1:
Geschwindigkeits-Strecke Diagramm

Um Rückschlüsse auf die geschalteten Ampelphasen bzw. Ampelstehzeiten ziehen zu können, werden in einem ersten Schritt die Haltepositionen extrahiert. Diese bilden die Grundlage für die Filterung bezüglich der Standzeiten. Nachdem also die Haltepunkte bekannt sind, können die Ampelphasen mittels der Standzeiten an diesen Punkten analysiert werden. In dieser Arbeit wird die Ampelstehzeit (t_A) als jene Wartezeit definiert, die man innerhalb eines bestimmten Umkreises um den Ampelpunkt wartet, d. h., wo sich die Geschwindigkeit nahe 0 bewegt (vgl. Abb. 2). Nahe 0 deswegen, weil es im Allgemeinen durch die GPS-Aufzeichnung aufgrund der Positionierungs-Ungenauigkeiten keine Geschwindigkeit 0 gibt.

Es gibt zwei verschiedene Arten, wie man die Haltezeit in einem Diagramm darstellen kann. Zum einen mit dem Geschwindigkeits-Zeit-Diagramm, zum anderen mit dem Strecken-Zeit-Diagramm. Zum Auswerten der Tracks eignen sich beide Arten, da die Geschwindigkeit als auch die Strecke mitgespeichert werden.

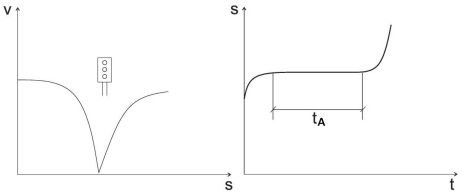

Abb. 2:
Geschwindigkeits-Zeit-Diagramm und Strecken-Zeit-Diagramm

Die Fahrzeug-abhängige Geschwindigkeit kann nur ermittelt werden, wenn die Ampelphasen bekannt sind. Das heißt, es muss die Dauer der Rot- und der Grünphase bekannt sein. Im Fall einer Rotphase muss die Geschwindigkeit bis zum Anfang der nächsten Grünphase berechnet werden, im Fall einer Grünphase die Geschwindigkeit, die für das rechtzeitige Durchfahren nötig ist. Faktoren, die diese Ermittlung beeinflussen, sind unter anderem:

- Position des Fahrzeuges und dessen Fahrtrichtung
- Position der nächsten Ampel (Entfernung zur nächsten Ampel)
- Ampelphase zum momentanen Zeitpunkt

Anzumerken ist, dass die Kreuzungsmittelpunkte und die Haltelinien statische Daten sind, d. h., wenn sie einmal ermittelt wurden und mit einer entsprechenden Genauigkeit feststehen, können sie in der Datenbank abgelegt werden und müssen nicht mehr verändert werden. Die Ampelphasen werden zum Teil auch statisch geschaltet, dynamische Zeiten, die sich je nach Tageszeit, Verkehrsaufkommen oder Fußgängeranzahl verändern, sind eine große Herausforderung bei der Implementierung und werden in dieser Arbeit nicht berücksichtigt. Die Daten müssen ständig aktualisiert werden, da eine Ungenauigkeit das System unbrauchbar machen würde. Ebenso muss die fahrzeugabhängige Geschwindigkeit wegen zeitabhängiger Ampelschaltung und positionsabhängigem Fahrzeugstandort ständig neu berechnet werden. All diese variablen Faktoren müssen für die effiziente Speicherung der Daten berücksichtigt werden.

2.1 Systemarchitektur

Die Systemarchitektur (Abb. 3) besteht im Wesentlichen aus zwei Bausteinen, dem Client/den Clients und dem Server (Backend). Im Client befindet sich ein Modul, welches während der Fahrt kontinuierlich die Position aufzeichnet und nach Abschluss eine GPX-Datei erzeugt. Diese Datei wird an den Server übertragen, dort ausgewertet und alle relevanten Daten in einer räumlichen Datenbank gespeichert. Der Client kann jederzeit, abhängig von seiner Position, die aktuellen Daten vom Server synchronisieren, und bekommt die ideale Geschwindigkeit angezeigt.

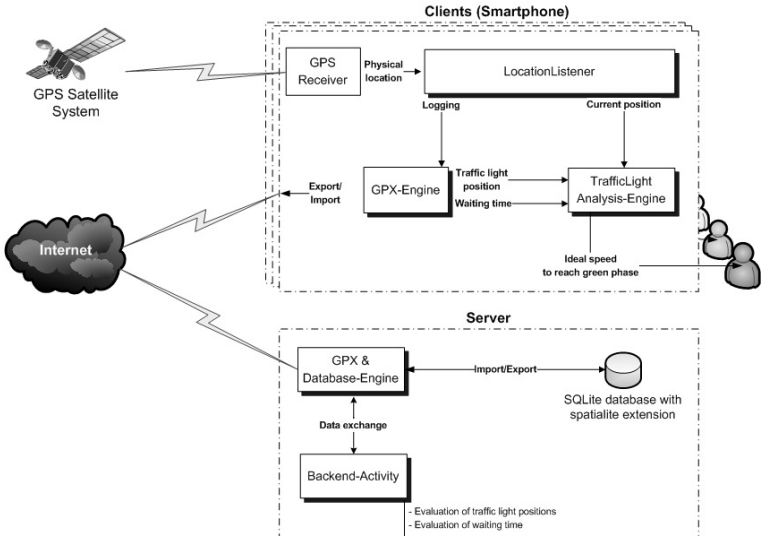

Abb. 3: Aufbau der Systemarchitektur

Der Client versorgt den Backend (Server) mit Tracks, die direkt nach Abschluss der Route über das mobile Internet übermittelt werden. Die Applikation ist für das Android-Betriebssystem implementiert und somit auf jeglichen Android-Geräten ausführbar. Sobald der Logging-Prozess gestartet wird, baut der GPS-Empfänger die Verbindung mit erreichbaren Satelliten auf. Wenn die Positionsbestimmung möglich ist, wird jede Sekunde ein Punkt mit seinen Koordinaten und zusätzlichen Attributen in die erzeugte GPX-Datei geschrieben. Danach wird die erzeugte Route fertig exportiert und an den Server übermittelt. Dieser kann anschließend mit der Auswertung der Datei beginnen. Die Traffic Light Analysis (Abb. 4) liefert dem Benutzer die Information über die Distanz zur nächsten Ampel, die Zeit, wie lange die Grünphase existiert bzw. wann die nächste Grünphase beginnt, und die erforderliche Geschwindigkeit, um die nächste Grünphase noch zu erreichen.

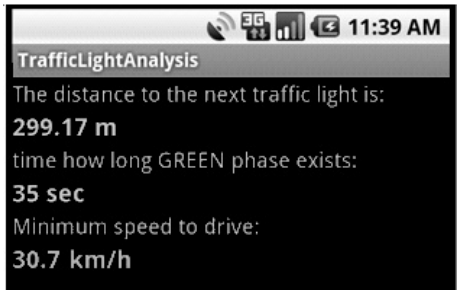

Abb. 4:
Visualisierung der Mindestgeschwindigkeit am Android- Client

Der Kernpunkt der Arbeit ist die automatisierte, serverseitige Berechnung der Position der Haltelinien. Im ersten Schritt werden alle Punkte aus den Rohdaten zusammengefasst, welche einen Abstand von weniger als 5 m aufweisen. Anschließend werden aus allen Punkten, die sich in einem Bereich gebildet haben, Cluster gebildet. Als letzter Schritt wird dann das Zentroid des Clusters berechnet, welches als Haltelinienposition in der Datenbank gespeichert wird.

3 Ergebnisse

Durchgeführte Analysen an zwei verschiedenen Ampeln zeigen vielversprechende Ergebnisse. In der ersten Abbildung (Abb. 5) erkennt man die ermittelte Position der Haltelinie in Form eines Sterns, der etwa einen Meter von der tatsächlichen Position abweicht.

Abb. 5:
Extraktion einer Ampelposition (Haltelinie) aus ausgewerteten Daten

Auch an einem anderen Ort wurde die Auswertung der Haltepunkte aus allen vorhandenen Routen vorgenommen (Abb. 6). Nach der Durchführung der Analyse ermittelte der Algorithmus eine weitere Position einer Haltelinie, welche nur etwa einen Meter von der tatsächlichen Position abweicht. Weitere Punkte konnten aufgrund der limitierten Anzahl vorhandener Routen nicht extrahiert werden.

Abb. 6:
Extraktion einer Ampelposition (Haltelinie) aus ausgewerteten Daten

4 Fazit

Die Herausforderung bei der Implementierung solcher Schritte ist das Denken in großem Maßstab. Es ist relativ einfach, aus einzelnen Versuchsobjekten mit Veränderung der Toleranzen auf Vorzeigeergebnisse zu gelangen. Schwierig wird es erst, wenn man die Implementierung so allgemein halten muss, dass sie auch alle möglichen Szenarien berücksichtigt. Für das Filtern von relevanten Daten aus einem großen Datensatz ist es nur mit statistischen Auswertungen möglich, daraus zu schließen, ob ein Haltepunkt wirklich bei einer Ampel ist bzw. ob er näher bei einer Ampel ist als ein anderer Haltepunkt. Dies könnte ein weiterer Ansatz für zukünftige Forschungsfragen sein. Um Ergebnisse zu erlangen, müssen bei nahezu jeder Funktion Toleranzwerte implementiert werden, die im Voraus manuell eingetragen werden müssen. Ob sich somit eine Halteposition aus einer Wartezeit von mindestens 30 sec in den letzten 40 m oder aus mindestens 20 sec in den letzten 30 m ergibt, liegt hierbei im Ermessen des Entwicklers. Für dieses Projekt wurden letztere Werte für die Berechnung herangezogen. Der nächste Schritt muss sich mit der Analyse der Ampelphasen beschäftigen. Diese ist nur konzeptuell angeschnitten worden, da dies ein sehr kompliziertes Thema ist, wenn man davon ausgeht, dass keine zusätzlichen Metainformationen über Schaltzeiten vorhanden sind. Grundsätzlich treten in erster Linie Fragen auf wie:

- Ab wann beginnt die Rot-Phase, wie lange dauert sie?
- Wie kann man die einzelnen Phasen kollektiv lernen?
- Wie werden Gelb-Phasen berücksichtigt?
- Wie werden die Zeiten pro Ampel gespeichert, wie oft werden sie aktualisiert?

Literatur

BRAUN, R. et al. (2009), TRAVOLUTION − Netzweite Optimierung der Lichtsignalsteuerung und LSA-Fahrzeug-Kommunikation, Köln.

KOUKOUMIDIS, E. et al. (2011), SignalGuru: leveraging mobile phones for collaborative traffic signal schedule advisory. Princeton.

MATTERN, F. (2003), Total vernetzt: Szenarien einer informatisierten Welt. Springer-Verlag, Berlin/Heidelberg, 1.

SCHRÖDER, J. (2009), Adaptive Verhaltensentscheidung und Bahnplanung für kognitive Automobile. Universitätsverlag, Karlsruhe, 9.