Institut für Parallele und Verteilte Systeme
Abteilung Verteilte Systeme
Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Diplomarbeit Nr. 3248

Verschleierung von Benutzertrajektorien unter Topologie- und Geschwindigkeitsrestriktionen

Robert Schmidt

Studiengang:	Informatik
Prüfer:	Prof. Dr. Kurt Rothermel
Betreuer:	Dipl.-Inf. Marius Wernke
begonnen am:	12. Oktober 2011
beendet am:	12. April 2012
CR-Klassifikation:	C.5.3, K.4.1

Zusammenfassung

Diese Arbeit beschäftigt sich mit der Sicherheit von Benutzertrajektorien in standortbasierten Diensten. Solche Dienste erlauben es dem mobilen Nutzer sich beispielsweise an Stauerkennungsdiensten zu beteiligen, neue Freunde zu finden oder Informationen zu interessanten Orten in der Nähe zu erhalten. Jedoch stellt sich bei diesen Diensten oft die Frage, wie geschützt die Positionsinformationen der Benutzer vor Angriffen sind. Daher müssen die Positionsinformationen des Benutzers geschützt werden.

In der Arbeit wird gezeigt, dass bisherige Verfahren, welche einzelne Positionsinformationen verschleiern, nicht für Benutzertrajektorien anwendbar sind. Um die Trajektorien zu schützen wird ein off-line Verfahren entwickelt, welches Dummy-Ansätze mit der Vorausberechnung der Alternativen kombiniert. Vor Fahrtbeginn werden, wie bei einem Navigationssystem, mögliche Trajektorien des Benutzers berechnet. Diese werden während der Fahrt an einen Lokationsserver gesendet.

Da nur Positionsinformationen aufgrund von vorausberechneten Alternativen an den Lokationsserver gesendet werden, hat ein Angreifer keine Möglichkeit, auf die reale Trajektorie zu schließen.

Bei der Untersuchung des Verfahrens wird gezeigt, dass gute Alternativen berechnet werden und das Verfahren anwendbar ist. Anhand von realen Trajektorien wird festgestellt, dass die vom Verfahren angenommenen Streckengeschwindigkeiten jedoch angepasst werden müssen.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Kapitel 1

Einleitung

1.1 Motivation

In den vergangen Jahren fand eine rapide Entwicklung im Bereich mobiler Anwendungen statt. Dies ist zum einen darauf zurückzuführen, dass die verbauten Chips in den mobilen Geräten immer kleiner und leistungsfähiger werden. Zum anderen weil die Benutzer auch bestimmte Anwendungen einfordern. Ein Beispiel für diese Entwicklung sind Smartphones, welche immer mehr mobile Alleskönner darstellen. Sie ersetzen Navigationssysteme, Terminplaner, Kameras, Mp3-Player, Camcorder, Bahntickets und vieles mehr. Da viele mobile Geräte inzwischen über GPS[1] ihre Position genau bestimmen können, wurden Dienste und Anwendungen entwickelt, welche die Positionsinformationen der Benutzer direkt nutzen können. Solche Dienste nennt man auch *standortbasierte Dienste*. Sie ermöglichen es den Nutzern beispielsweise neue Freunde mit gleichen Interessen[2] zu finden, aktiv an Stauerkennungsdiensten teilzunehmen oder einfach nur ein standortbezogenes Spiel zu spielen[3].

Jeder Benutzer sendet seine Positionsinformationen an einen Lokationsserver, der diese verwaltet und Anfragen seiner Nutzer beantwortet. Jedoch kann nicht für jeden Dienst gewährleistet werden, dass ein solcher Lokationsserver vertrauenswürdig ist. Demnach wäre es für einen Angreifer möglich an die Positionsdaten der mobilen Nutzen zu kommen. Dies stellt eine Verletzung der Privatsphäre der Benutzer dar und muss deshalb verhindert werden. Deshalb muss eine Möglichkeit gefunden werden die

[1] Global Positioning System
[2] Friend-Finder-Applikationen
[3] `http://www.scvngr.com/`

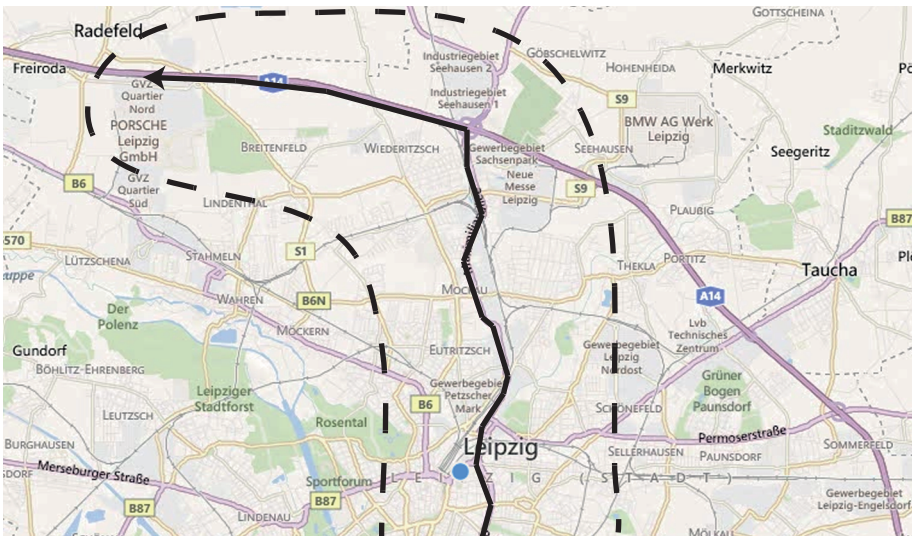

Abbildung 1.1: Schlauch um eine reale Trajektorie

Positionsinformationen der Benutzer zu schützen und gleichzeitig zu ermöglichen das standortbasierte Dienste verwendet werden können.

1.2 Zielsetzung

Das Ziel dieser Arbeit ist es, eine Verschleierung für Benutzertrajektorien zu entwickeln. Dabei wird davon ausgegangen, dass ein mobiler Benutzer einem Lokationsserver fortlaufend Positionsinformationen zur Verfügung stellt, um eine ortsbezogene Anwendung zu nutzen. Jedoch handelt es sich bei dem Lokationsdienst um eine nicht vertrauenswürdige Instanz. Das bedeutet, dass es einem Angreifer möglich sein könnte, an die Positionsinformationen der Benutzer zu gelangen. Es muss also sichergestellt werden, dass die Privatsphäre des Benutzers gewahrt bleibt. Gleichzeitig soll die Anwendung des Benutzers nicht beeinträchtigt werden.

Dazu wäre es optimal, eine Anwendung zu entwickeln, die um eine reale Trajektorie einen Schlauch legen könnte, wie in Abbildung 1.1 dargestellt. Die Herausforderung hierbei besteht darin, dass es einem Angreifer nicht möglich sein soll, durch Probieren der Strecken den Schlauch zu generieren. Dadurch würde er sonst auf die reale Trajektorie schließen können.

1.3 Aufbau der Arbeit

Dieses einleitende Kapitel stellt die Motivation für die Arbeit dar. Im folgenden zweiten Kapitel werden die Grundlagen dieser Arbeit beschrieben. Dabei wird vor allem auf bestehende Verfahren für die Verschleierung von einzelnen Positionen eingegangen. Daran anschließend werden auch Möglichkeiten beschrieben, wie komplette Trajektorien verschleiert werden können und welche Angriffsmöglichkeiten hier bestehen. Im Hauptteil der Arbeit wird ein eigenes Verfahren zur Verschleierung von Trajektorien entwickelt, bei dem die Angriffsmöglichkeiten aus dem zweiten Kapitel nicht angewendet werden können. Das Verfahren wird anschließend im vierten Kapitel evaluiert. Das letzte Kapitel fasst die Ergebnisse der Arbeit zusammen und stellt einen Ausblick für folgende Arbeiten.

Kapitel 2

Grundlagen und verwandte Arbeiten

In diesem Kapitel werden die Grundlagen dieser Arbeit beschrieben. Dazu werden zunächst im Abschnitt 2.1 Beispiele von standortbezogenen Diensten vorgestellt. Der darauffolgende Abschnitt 2.2 behandelt die Abwägung zwischen der Privatsphäre eines Benutzers und der geforderten Dienstgüte.

Nach der Diskussion über den Schutz der Privatsphäre werden im Abschnitt 2.3 bestehende Konzepte zur Verschleierung von einzelnen Positionsinformationen vorgestellt. Diese werden insbesondere auf deren Angreifbarkeit überprüft.
Im letzten Abschnitt 2.4 wird nicht mehr die Verschleierung von einzelnen Positionsinformationen angenommen, sondern es wird die Verschleierung von kompletten Trajektorien betrachtet. Dabei werden auch einige Verfahren der Positionsverschleierung auf Trajektorien übertragen und auf deren Angreifbarkeit überprüft.

2.1 Beispiele von standortbasierten Diensten

Standortbasierte Dienste sollen dem Benutzer einen Vorteil bieten, für den sie bereit sind, ihre Positionsinformationen zur Verfügung zu stellen. Dabei unterscheidet man zwei Arten von Diensten.
Die erste Art standortbasierter Dienste benötigt nur Positionsinformationen und keine eindeutige Benutzerkennung. Dazu zählen zum Beispiel Dienste, welche interessante

Anwendung	**Beschreibung**	**Benötigte Genauigkeit der Position**
Friend-Finder	Es sollen Nutzer ermittelt werden, die gleiche Interessen haben (z.B. im gleichen Cafe sind).	fein - mittel
Flottenmanagement	Ein Chef möchte wissen, wo sich die Fahrzeuge seiner Mitarbeiter aufhalten.	fein - grob
Mautstraßen	Mautstraßen sollen bei Benutzung abgerechnet werden.	fein - mittel
Staudienst	Staus und Behinderungen sollen dadurch erkannt werden, dass sich viele Nutzer auf dem gleichen Streckenabschnitt aufhalten.	mittel
Rabattsystem	Sobald ein Nutzer in der Nähe eines teilnehmenden Geschäftes ist, erhält er einen virtuellen Coupon, welchen er einlösen kann.	mittel
Versicherungen	Es soll ungefähre Laufleistung eines PKWs pro Jahr ermittelt werden. Aus diesen Daten werden die Versicherungskosten berechnet.	mittel - grob

Tabelle 2.1: Typische standortbasierte Dienste und deren Genauigkeit

Orte in der Nähe anzeigen oder das Wetter für den aktuellen Standort ermitteln.
Die zweite Art dieser Dienste benötigt neben den Positionsinformationen auch eine eindeutige Benutzerkennung. Dies ist zum Beispiel dann nötig, wenn man dem Benutzer eine Rechnung stellen will.

In der Tabelle 2.1 werden Beispiele für standortbasierte Dienste vorgestellt, welche eine Benutzerkennung benötigen.
Die benötigte Genauigkeit solcher Dienste ist stark abhängig von der Art der Anwendung. Dabei müssen bei der Konzeption viele Faktoren berücksichtigt werden. Bei mobilen Geräten ist beispielsweise der Energieverbrauch ein wichtiger Faktor. Dabei würde die regelmäßige Standortermittlung via GPS den Energieverbrauch stark steigern. Neben diesem Faktor muss ein Anbieter sicherstellen, dass die Anwendung eine

entsprechende Dienstgüte gewährleistet. Dem gegenüber steht die Privatsphäre des Benutzers. Diese beiden Dimensionen werden im folgenden Abschnitt diskutiert.

2.2 Dienstgüte und Privatsphäre von Positionsinformationen in standortbasierten Diensten

Die mobilen Nutzer eines standortbasierten Dienstes haben vor allem das Ziel, eine größtmögliche *Dienstgüte* zu erreichen. Sie sind sich über die Vorteile der mobilen Anwendungen im Klaren. Ermittelt eine ortsbezogene Anwendung beispielsweise die Tankstelle mit dem niedrigsten Benzinpreis in der Nähe, so erhält ein Benutzer den Vorteil, dass er dort günstig tanken kann. Um solche vorteilhaften Dienste zu nutzen, muss ein Nutzer bereit sein, seine Positionsinformationen zur Verfügung zu stellen. Dabei ist es in Bezug auf die *Dienstgüte* optimal, wenn der Benutzer seine exakte Position mitteilt.

Cheng et al. [CCLS11] zeigen jedoch, wie bedenklich es ist, seine exakte Positionsinformationen zu veröffentlichen. Sie untersuchten Positionsdatensätze der Nutzer von Foursquare[1], Facebook[2] und Gowalla[3]. Die Ergebnisse der Untersuchung lassen sich wie folgt zusammenfassen:

- **Die Benutzer der Systeme folgen reproduzierbaren Mustern.**
- **Der Sozialstatus der Benutzer, in Verbindung mit geographischen und ökonomischen Faktoren, bestimmt die Mobilität eines Benutzers.**
- **Zusätzliche Informationen, die von den Benutzern zur Verfügung gestellt wurden (z.B. einem Kommentar zur aktuellen Position) führten dazu, dass man vorher unbekannte Zusammenhänge zwischen den Benutzern und der Position erhielt.**

Selbst wenn nur einzelne exakte Positionen veröffentlicht werden, können aus zusätzlichen externen Informationen, komplexe Zusammenhänge entstehen. Nutzt man eine mobile Anwendung beispielsweise während der Arbeit im Büro, so lassen sich Rückschlüsse über den Arbeitgeber ziehen. Ist die Anwendung während der Fahrt zur

[1] `http://www.foursquare.com`
[2] `http://www.facebook.com`
[3] `http://www.gowalla.com`

Arbeit aktiv, so kann man Präferenzen über die verwendeten Verkehrsmittel ableiten (z.B. Bus, Bahn, Auto).
Neben diesen einfachen Informationen lassen sich aber auch weitere Assoziationen herstellen, die kritischer sind. Befindet man sich beispielsweise an einem Ort, an dem gerade eine Demonstration stattfindet, so könnte auf die politischen Einstellungen geschlossen werden. Hält man sich oft in einer Spezial-Klink auf, so könnte man an einer Krankheit leiden. Die Möglichkeiten, Positionen mit Aussagen zu assoziieren, sind praktisch unbegrenzt.

Jedoch existiert eine zweite Dimension, welche immer mehr in den Fokus rückt - die *Privatsphäre von Positionsdaten*.

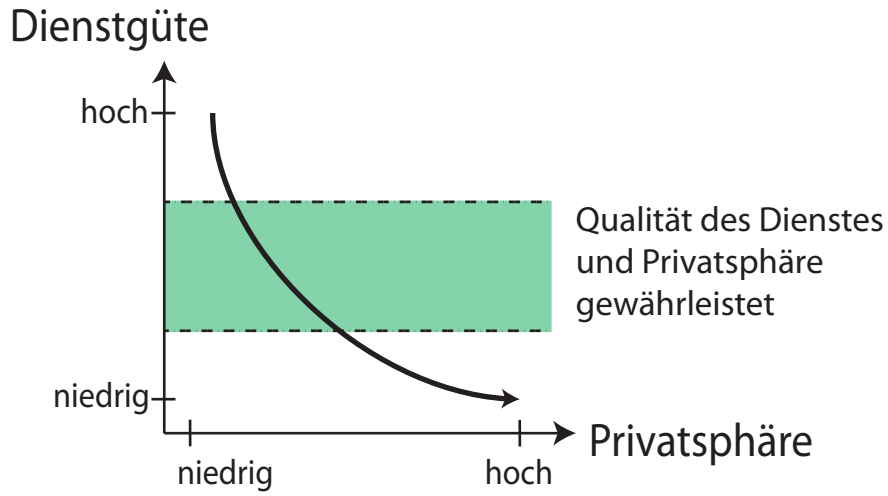

Abbildung 2.1: Zusammenhang von Dienstgüte und der Privatsphäre

> *"Die Privatsphäre von Positionsinformationen beschreibt die Möglichkeit eines Benutzers sicherzustellen, dass es einem Angreifer nicht möglich ist, die aktuelle oder eine vergangene Position dieses Benutzers zu ermitteln."* [BS03] (eigene Übersetzung)

Die Dimension der Privatsphäre bestimmt also, wie aufwendig es ist, aus konkreten Daten auf dem Lokationsserver Rückschlüsse auf eine exakte Position schließen zu können. Im Gegensatz zur Dienstgüte, ist es also für die Privatsphäre sehr schlecht, exakte Positionsinformationen zu veröffentlichen.

Werden beide Dimensionen berücksichtigt, muss man einen Mittelweg zwischen Dienstgüte und Privatsphäre finden (vgl. Abb. 2.1).
Dabei ist die Herausforderung, ein Verfahren zu entwickeln, welches die Positionsinformation eines Nutzers schützt und es gleichzeitig für Angreifer kaum möglich macht, auf eine exakte Position der mobilen Nutzer zu schließen. Gleichzeitig soll der Dienst weiterhin zufriedenstellende Ergebnisse liefern. Um dies zu gewährleisten, wurden einige Verfahren zur *Verschleierung von Positionsinformationen* vorgeschlagen, welche im folgenden Abschnitt vorgestellt werden.

2.3 Verfahren zur Verschleierung von einzelnen Positionsinformationen

Um von den Vorteilen standortbezogener Dienste zu profitieren, muss ein Benutzer seine aktuelle Position dem Lokationsserver mitteilen.

Der Abschnitt 2.2 zeigte, dass das Senden von genauen Positionsinformationen kritisch ist. Um den Schutz der Privatsphäre des Benutzers zu gewährleisten, wurden in der Literatur bereits Verschleierungsverfahren für einzelne Positionen vorgeschlagen. Diese werden in diesem Abschnitt vorgestellt und auf ihre Angreifbarkeit überprüft.

2.3.1 Verringerung der räumlichen Genauigkeit

Die Verringerung der räumlichen Genauigkeit ist ein einfaches Verfahren, um einzelne Positionsinformationen schnell verschleiern zu können [ACD+07]. Die Idee des Verfahrens ist es, einen Ungenauigkeitsbereich um eine reale Position zu legen. Innerhalb dieses Bereiches soll die Aufenthaltswahrscheinlichkeit für jeden Punkt gleich groß sein. In der Abbildung 2.2 wird ein solches Verfahren veranschaulicht. Dabei wurde ein Kreis mit einem gewissen Radius um die reale Position gelegt.

Wenn jedoch davon ausgegangen wird, dass sich ein Benutzer auf einem Straßengraphen bewegt, so ist die Annahme, dass jede Position innerhalb des Ungenauigkeitsbereiches gleich wahrscheinlich ist, falsch. Abbildung 2.3 zeigt, dass ein Angreifer die möglichen Aufenthaltsorte dann auf alle Straßen innerhalb des Ungenauigkeitsbereiches reduzieren kann.

Abbildung 2.2: Beispiel: Verringerung der räumlichen Genauigkeit

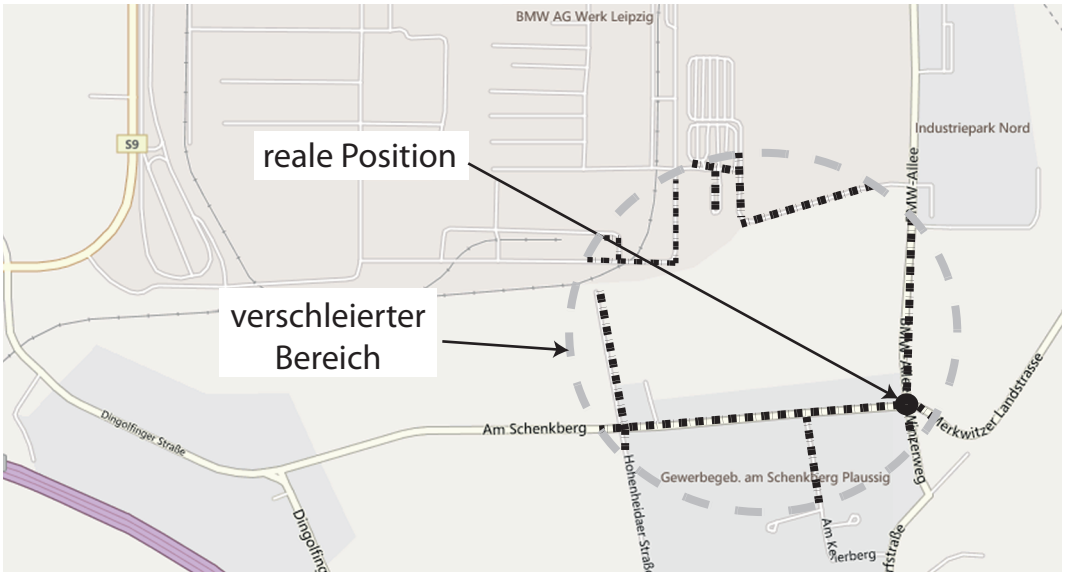

Abbildung 2.3: Reduzierung des Ungenauigkeitsbereiches auf den Straßengraph

Zudem ist es problematisch, dieses Verfahren in weniger bewohnten Gebieten anzuwenden. Dort gibt es wenige plausible Aufenthaltsorte, womit die Wahrscheinlichkeit von anderen Postionen innerhalb des Ungenauigkeitsbereiches des verringert wird. Insgesamt lässt sich also festhalten, dass solche Bereiche, durch Kenntnis von Topologieinformationen, degenerieren können.

2.3.2 Verschleierung durch Verteilung von Positionsinformationen

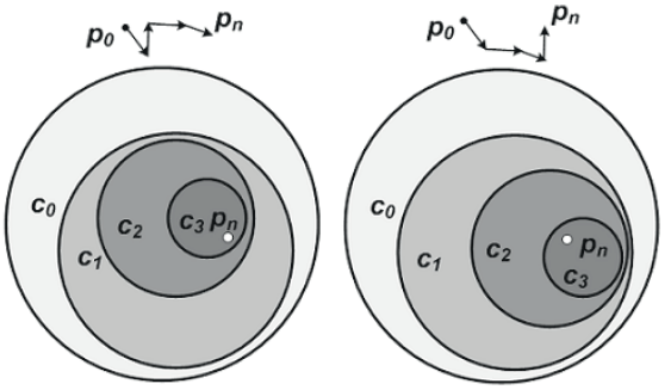

Abbildung 2.4: Darstellung der vektoriellen Addition von verschiedenen Shares p_j. c_j entspricht der Genauigkeit. [DSR11]

Dürr und Skvorzov kombinieren die räumliche Verschleierung mit der Verteilung der Positionsinformationen [DSR11]. Dabei werden die einzelnen Teile, sogenannte Shares, auf unabhängigen Lokationsservern gelagert. Die Verteilung auf unterschiedlichen Servern hat den Hintergrund, dass es unwahrscheinlich ist, dass ein Angreifer auf alle Server gleichzeitig Zugriff hat. Die Wiederherstellung der genauen Positionsinformation ist nur dann möglich, wenn man alle Shares besitzt. Konkret werden die Shares als einzelne Vektoren betrachtet. Durch vektorielle Addition werden sie wieder zusammengefügt und es erhöht sich die Genauigkeit des Ergebnisses um einen gewissen Grad (vgl. Abb. 2.4)
Für einen erfolgreichen Angriff ist hier entscheidend, auf wieviele Shares ein Angreifer Zugriff bekommt. Zudem muss auch bei diesem Verfahren beachtet werden, dass der Ungenauigkeitsbereich degenerieren kann, wenn sich der Benutzer beispielsweise auf einem Straßengraphen bewegt (vgl. Abschnitt 2.3.1).

2.3.3 Location k-Anonymity

> *Anonymität ist der Zustand der entsteht, wenn man innerhalb von mehreren Individuen nicht identifiziert werden kann.* [PK01] (eigene Übersetzung)

Das Verfahren der *Location k-Anonymity* stammt ursprünglich aus dem Bereich der rationalen Datenbanken [SS98].

Wenn statistische Daten veröffentlicht werden, so werden diese Daten in der Regel anonymisiert. Dazu werden identifizierende Attribute, wie Name oder Personalnummer weggelassen. Die Tabelle 2.2 zeigt eine anonymisierte Studie aus der nicht ersichtlich ist, wie derjenige heißt, der Schnupfen hat. Kennt man jedoch den Namen einer Person dieser Studie, weiß das diese in 71332 Waiblingen wohnt und 38 Jahre alt ist, so kann man die Person identifizieren. Gleichzeitig verringert sich die Anzahl der verbliebenen nicht zugeordneten Datensätze. Das Verfahren kann weiter verbessert werden, wenn man die Daten so anonymisiert, dass eine Anfrage nach einer beliebigen Kombination von Eigenschaften, entweder ein leeres Ergebnis oder mindestens k-Tupel zurückliefert [SSH10].

Name	**Alter**	**PLZ**	**Geschlecht**	**FamStand**	**Krankheit**
*****	38	71***	m	verheiratet	Schnupfen
*****	29	71***	w	ledig	Fieber
*****	29	39***	w	ledig	Knochenbruch
*****	25	98***	m	ledig	Fieber
*****	25	98***	m	ledig	Schnupfen

Tabelle 2.2: Beispiel: Anonymisierten Studie nach [SSH10]

Übertragen auf den Bereich der Positionsverschleierung bedeutet k-Anonymity, dass ein Benutzer nicht unterscheidbar von mindestens $k - 1$ anderen Benutzern ist. Dazu muss ein Verschleierungsbereich gewählt werden, der k-Benutzer beinhaltet. Um dies zu gewährleisten, gibt es mehrere Möglichkeiten. Die erste Möglichkeit ist, einem Server die Wahl der Verschleierungszone und der *k*-mobilen Geräte zu überlassen. In diesem Fall müsste der Server vertrauenswürdig sein, da er die exakten Positionsdaten der Nutzer besitzt. Eine weitere Möglichkeit ist es, die Verschleierungszone gemeinsam von allen Benutzern des Lokationsservers bestimmen zu lassen. Bei diesem Vorgehen müsste aber jedem Benutzer vertraut werden.

Auch dieses Verfahren hat Schwachstellen. Selbst wenn die Eigenschaft der Location k-Anonymity gewährleistet ist, so reicht dies für Gebiete, in denen es nur einen plausiblen

Aufenthaltsort gibt, nicht aus. In diesen Bereichen kann ein Angreifer die richtige Position mit hoher Wahrscheinlichkeit feststellen.

2.3.4 Location l-Diversity

Eine Weiterentwicklung von Location k-Anonymity ist *Location l-Diversity* [XKP09]. Um die Angriffsmöglichkeiten des k-Anonymity-Verfahrens zu vermeiden, wird zusätzlich verlangt, dass einer Verschleierungszone mindestens l verschiedene plausible Aufenthaltsorte zugewiesen werden. Damit lässt sich für einen Angreifer keine Information aus den Eigenschaften der Umgebung ziehen.
Ein Verfahren welches Location l-Diversity benutzt, ist PrivacyGrid. Dieses Verfahren geht davon aus, dass ein zusätzlicher Anonymisierungsserver Informationen über plausible Aufenhaltsorte einer Region besitzt. Der Anonymisierungsserver modifiziert Anfragen so, dass die Antwort mindestens l verschiedene Aufenthaltsorte enthält [BLPW08].
Dennoch muss bei diesem Verfahren sichergestellt werden, dass die Aufenthaltsorte nicht zu nah beieinander liegen, da sonst der Bereich, in dem sich der Benutzer aufhalten kann, zu klein ist.

2.4 Verfahren zur Verschleierung von Benutzertrajektorien

Bereits die Verschleierung von einzelnen Positionsinformationen bietet Angriffsmöglichkeiten. Verwendet man beispielsweise die räumliche Verschleierung, so kann der Ungenauigkeitsbereich degenerieren.

Werden nun nicht mehr nur einzelne Positionen verschleiert, sondern komplette Trajektorien, so erhöht sich der Aufwand der Verschleierung noch mehr. Es werden zwei Verfahren zur Verschleierung von Trajektorien vorgestellt und zwei Verfahren aus dem vorhergehenden Abschnitt auf Trajektorien übertragen. Auch hier steht im Vordergrund die Angreifbarkeit zu untersuchen.

2.4.1 Verringerung der räumlichen Genauigkeit

Im Abschnitt 2.3.1 wurde die räumliche Verschleierung von Einzelpositionen behandelt. Die Idee bei dem Verfahren war es, einen Ungenauigkeitsbereich um eine einzelne Position zu legen. An dieser Stelle soll das Verfahren betrachtet werden, wenn man eine komplette Trajektorie damit verschleiert.

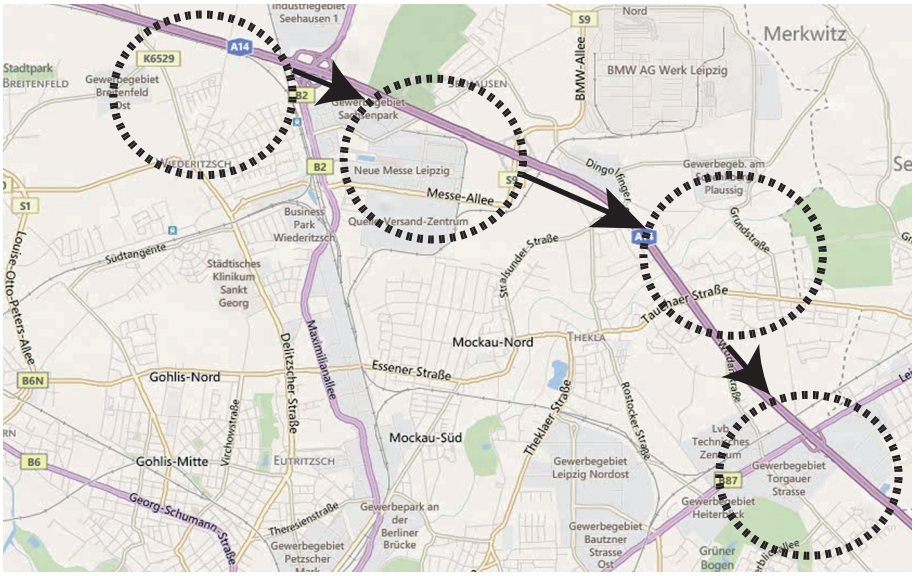

Abbildung 2.5: Beispiel für eine Trajektorie aus räumlichen Verschleierungen

Abbildung 2.5 zeigt vier einzelne Ungenauigkeitsbereiche. Werden diese einzelnen Bereiche miteinander verbunden, entsteht eine Benutzertrajektorie. Die einzelnen Bereiche umfassen ein Gebiet von mehreren Kilometern. Daher ist es schwierig, innerhalb der Bereiche einen genauen Aufenthaltsort zu bestimmen. Wird allerdings die komplette Benutzertrajektorie betrachtet, so kann ein Angreifer mit hoher Wahrscheinlichkeit schließen, dass sich der mobile Nutzer auf der Autobahn bewegt. Diese verbindet die einzelnen Ungenauigkeitsbereiche über einen kürzesten Pfad miteinander. Wenn fortlaufend Positionsinformationen gesendet werden müssen eignet sich diese Verschleierungsmethode nicht, da sie anfällig gegenüber Angriffe mit Kürzeste-Wege-Algorithmen ist.

2.4.2 Verteilung der Positionsinformationen

In diesem Abschnitt wird das Verfahren der *Positionsverteilung* aus Abschnitt 2.3.2 auf komplette Trajektorien übertragen. Dazu wird angenommen, dass in regelmäßigen Abständen eine Positionsinformation auf verschiedene Server aufgeteilt wird. Im Gegensatz zum vorhergehenden Verfahren (Verringerung der räumlichen Genauigkeit), kommt es hier darauf an, wie die Shares auf den Servern verteilt wurden. Es ist zum Beispiel möglich, dass der Angreifer nur Shares vom Anfang oder vom Ende einer Trajektorie bekommt. Damit erhält er also keine Rückschlüsse auf die komplette Trajektorie. Werden die Shares jedoch während der Fahrt des mobilen Nutzers gleichmäßig auf die Server verteilt, so kann der Angreifer mit Hilfe eines Kürzeste-Wege-Algorithmus die einzelnen Ungenauigkeitsbereiche miteinander verbinden. Somit würde er die wahrscheinlichste Trajektorie eines mobilen Nutzers wiederherstellen können.

2.4.3 Verschleiern durch Koordinatentransformation

Gutscher beschreibt die *Koordinatentransformation* als eine Möglichkeit zur Verschleierung von Benutzertrajektorien [Gut06a]. Die reale Position eines Benutzers wird dabei über eine Transformationsfunktion $\vec{d}_{d,e}$, die nur dem Benutzer bekannt ist, in einen anderen Koordinatenraum transformiert (vgl. Abb 2.6). Wenn ein Lokationsserver nun eine Anfrage bekommt, so beantwortet er die Anfrage nur auf diesem transformierten Koordinatenraum.

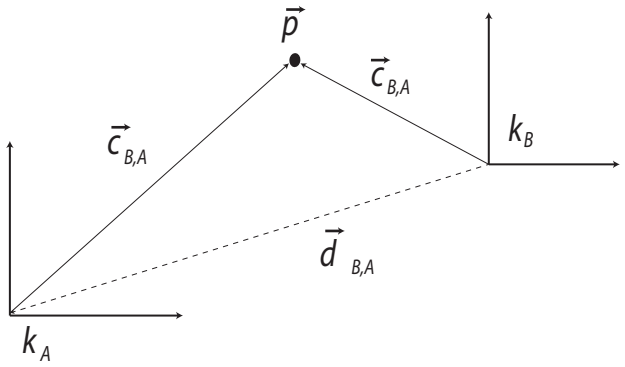

Abbildung 2.6: Koordinatentransformation nach [Gut06a]

Bei diesem Verfahren ist fraglich, ob ein Benutzer tatsächlich geschützt ist. Aus zusätzlichen Informationen (z.B. der Kenntnis des Arbeitsplatzes) kann ein Angreifer Bewegungsprofile ableiten. Ist es nun für einen Angreifer möglich, die transformierten Positionen auf reale Verkehrswege abzubilden, so kann er die Transformationsfunktion ermitteln [Gut06b].

2.4.4 Verschleierung durch Senden von falschen Trajektorien

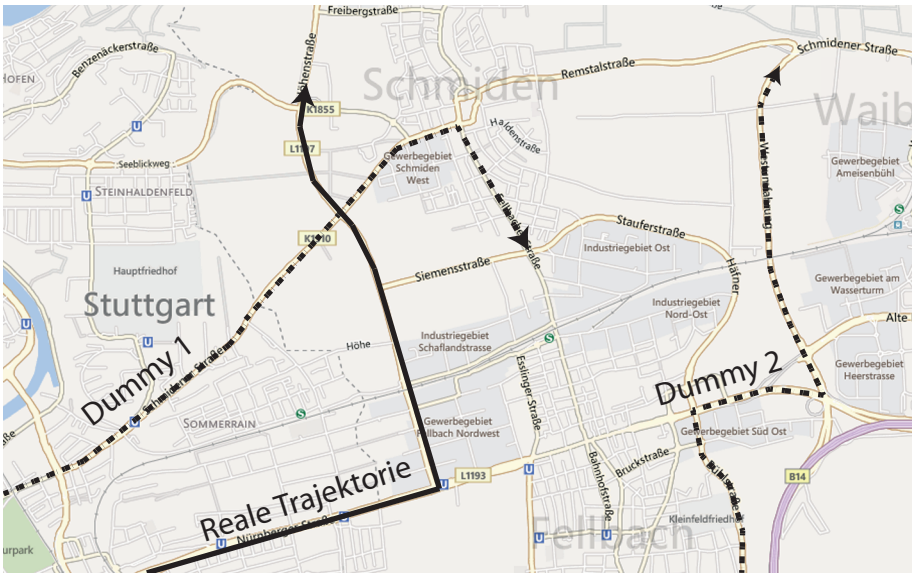

Abbildung 2.7: Beispiel für eine Verschleierung mit falschen Trajektorien

Eine weitere Möglichkeit die Benutzertrajektorie zu verschleiern ist es, falsche Trajektorien an den Lokationsserver zu senden (vgl. Abb 2.7). Diese falschen Trajektorien nennt man auch *Dummies*. Solche Dummies werden unter der gleichen Benutzeridentifikation veröffentlicht, wie die reale Trajektorie. Allein der mobile Nutzer kann somit bei Antworten des Lokationsservers erkennen, welche Trajektorien falsch waren [KYS05].
Allerdings muss bei diesem Verfahren beachtet werden, dass die Bewegungen von realen Personen in der Regel speziellen Mustern folgen. Daher müssen diese Muster bei der Generierung der falschen Trajektorien berücksichtigt werden. Werden die Trajektorien lediglich zufällig generiert, kann ein Angreifer die falschen Trajektorien erkennen [YPL07]. Es gibt jedoch auch die Möglichkeit auf bereits existierende reale Trajektorien zurückzugreifen und diese wiederzuverwenden. Jedoch stellt sich hier die Frage, woher

das mobiles Gerät diese Trajektorien bekommen soll. Sofern sie von einem externen System bereitgestellt werden, so müsste man diesem System vertrauen.

Kapitel 3

Entwicklung einer Verschleierung für Benutzertrajektorien

Nach dem Grundlagenteil der Arbeit, wird im folgenden Kapitel die Entwicklung eines Verschleierungsverfahrens für Benutzertrajektorien behandelt.
Dabei wird zunächst das in der Arbeit angenommene Systemmodell vorgestellt. Anschließend wird in Abschnitt 3.2 beschrieben, welches Angreifermodell angenommen wird. Basierend auf dem Systemmodell und dem Angreifermodell und weiteren Betrachtungen zu Verschleierungsverfahren, werden im darauf folgenden Abschnitt 3.3 Entwurfsentscheidungen für das Verschleierungsverfahren getroffen. In den darauffolgenden Abschnitten werden Definitionen und die verwendete Metrik vorgestellt. Anschließend wird im Abschnitt 3.7 der genaue Ablauf des Verschleierungsverfahrens erläutert.

3.1 Systemmodell

> *"Location-based Services are IT services for providing information that has been created, compiled, selected, or filtered taking into consideration the current locations of the users or those of other persons or mobile devices."* [Küp05]

Das Systemmodell umfasst die Komponenten eines *standortbezogenen Dienstes*. Ein solcher Dienst stellt einem mobilen Nutzer Informationen zur Verfügung, welche aufgrund seines aktuellen Standortes bestimmt werden. Abhängig von der Art des

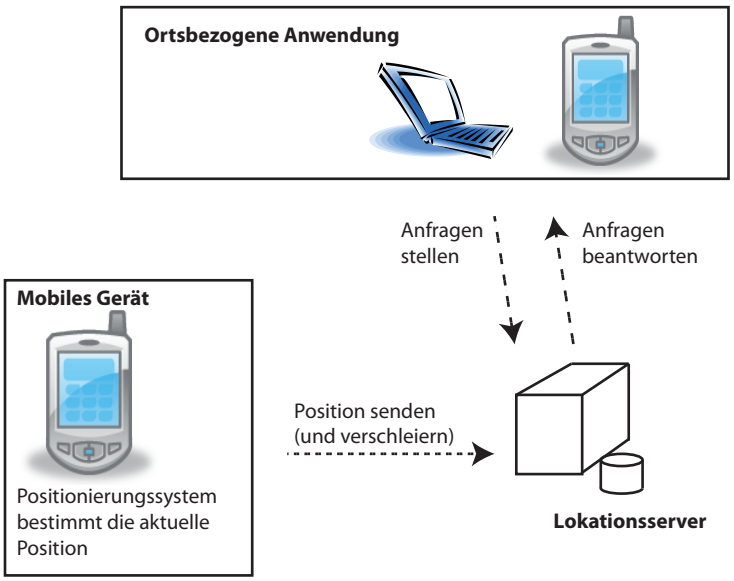

Abbildung 3.1: Darstellung des Systemmodell

Dienstes, wird nicht nur die eigene Position berücksichtigt, sondern auch die Positionen von anderen mobilen Geräten (vgl. 2.1).

Das Systemmodell besteht aus folgenden Teilen [SNE06] [DSR11]:

- ***Mobiles Gerät***

  Der Nutzer des Systems trägt ein mobiles Gerät bei sich. Dies kann entweder ein Smartphone, ein Notebook oder aber auch eine Steuerungseinheit im Auto sein. Ein mobiles Gerät besitzt ein Positionierungssystem und kann Anhand von GPS seine aktuelle Position bestimmen. Weiterhin hat das mobile Gerät die Aufgabe, Positionsinformationen an einen Lokationsserver zu senden, wobei die Möglichkeit besteht, dass das mobile Gerät die aktuelle Positionsinformation verschleiert. Einige existierende Verfahren zur Verschleierung wurden bereits im Grundlagenkapitel betrachtet.

Abbildung 3.2: Benutzertrajektorie aus 7 Punkten

- ***Lokationsserver***

  Der Lokationsserver stellt die zentrale Einheit dar. Er verwaltet die gesendeten Positionsinformationen der mobilen Geräte und stellt deren Daten in aufbereiteter Form zur Verfügung. Sind auf dem Lokationsserver eine Reihe von einzelnen Positionen gespeichert, die einem Benutzer eindeutig zugeordnet werden können, so entsteht durch Verbindung der einzelnen Punkte ein Pfad. Dieser Pfad wird als Benutzertrajektorie bezeichnet (vgl. Abb. 3.2).

- ***Ortsbezogene Anwendung***

  Eine ortsbezogene Anwendung stellt dem Lokationsserver Anfragen und verarbeitet seine Antworten. Dabei kann diese Anwendung auf einem externen System installiert sein oder auf dem mobilen Gerät des Benutzers. Ortsbezogene Anwendungen und Dienste können vielfältig sein. Einige Beispiele wurde bereits in Abschnitt 2.1 vorgestellt.

3.2 Angreifermodell

Ein Angriff auf einen standortbezogenen Dienst kann verschiedene Zielsetzungen haben. Czerny beschreibt zwei typische Angriffsziele [Cze07]:

a) Störung des Dienstes, so dass der Dienst nicht mehr zur Verfügung steht.

b) Infiltration des Dienstes, um positionsbezogene Daten eines Benutzers zu erhalten.

Im Rahmen dieser Diplomarbeit wird von einem Angreifer ausgegangen, welcher das Ziel hat an die Positionsdaten des Lokationsservers zu gelangen.

Dabei wird der folgende Angriff durchgeführt, welcher Anhand der Abbildung 3.3 erklärt werden soll [EFH+11] :

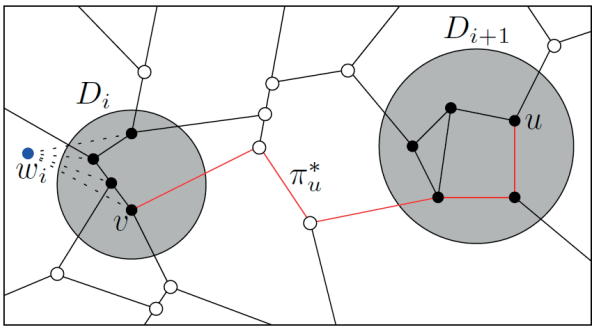

Abbildung 3.3: Verfahren zur Trajektorienbestimmung [EFH+11]

Gegeben sei ein Straßengraph G. Auf dem Lokationsserver sind einzelne Ungenauigkeitsgebiete D_i in geordneter Form gespeichert. Diese beinhalten eine Menge von Knoten und Kanten. Für einen verschleierten Bereich D_j, sind also V_j, die Knoten des Bereiches. Ziel ist es, diese Gebiete mit einem kürzesten Pfad zu verbinden. Ein einzelner kürzester Pfad zwischen den Ungenauigkeitsbereichen wird später als π_v^* bezeichnet. Nun wird vom einem Startbereich ausgehend ein neuer Knoten w_i eingeführt. Von diesem Knoten wird zu jedem Knoten aus V_i eine gerichtete Kante der Länge 0 eingefügt. Nun wird der kürzeste Pfad zu jedem Knoten aus dem nächsten Ungenauigkeitsgebiet berechnet.

Wird nun angenommen das bereits ein kürzester Weg π_v von w_1 zu jeder Kante $v_e \in V_i$ für $1 \leqq i < s$ existiert - wobei die kürzesten Pfade die verschleierten Bereiche $D_1, D_2, ..., D_{i-1}$ durchqueren. Dabei ändert sich durch die Hinzunahme von V_{i+1} der bestehende kürzeste Pfad von w_1 durch die Gebiete $D_1, D_2, ..., D_i$ nicht. Es wird lediglich der kürzeste Pfad erweitert.

Dieser Ansatz funktioniert, da angenommen wird, dass sich die Benutzer oft auf dem kürzesten Pfad zwischen den Gebieten bewegen.

3.3 Entwurfsentscheidungen

Im Einleitungskapitel wurde bereits erwähnt, dass es optimal wäre, ein Verfahren zu entwickeln, welches einen Schlauch um die Benutzertrajektorie legt. Ein solcher Schlauch repräsentiert dabei den Bereich, in welchem sich ein Benutzer aufhält. Genauer gesagt, wenn man sich auf einem Straßengraphen bewegt, ist es die Menge aller Straßen, auf denen sich der Benutzer aufhalten kann.

Die Generierung eines solchen Schlauches lässt sich mit verschiedenen Methoden bewerkstelligen. Ein einfaches Verfahren ist es, einen Schlauch mit einer festen Größe um die reale Trajektorie zu legen. Ein Angreifer kann bei dieser Vorgehensweise allerdings die reale Position leicht herausfinden, wenn er den Schlauch in bestimmte Gebiete einteilt und zwischen den Gebieten die kürzesten Pfade berechnet. Dieses Vorgehen wurde bereits im vorhergehenden Abschnitt 3.2 angesprochen. Wünschenswert wäre es also, wenn ein on-line Verfahren existieren würde, welches den Schlauch während der Fahrt erstellt. Jedoch müsste ein solches deterministisches Verfahren zu einem gewissen Zeitpunkt entscheiden können, welche Alternativen zu dem Schlauch hinzugenommen werden.

Das folgende Beispiel soll dieses Entscheidungsproblem näher erläutern:

Man befindet sich auf einer Autobahn und nutzt eine ortsbezogene Anwendung. Es soll eine Verschleierung der realen Trajektorie dadurch stattfinden, dass man während der Fahrt alternative Trajektorien hinzunimmt und diese dem Lokationsserver mitteilt. Man erreicht nun die erste Abfahrt (vgl. Abb. 3.4). Jetzt stellt sich die Frage, ob man die markierte Strecke weiterverfolgen soll.

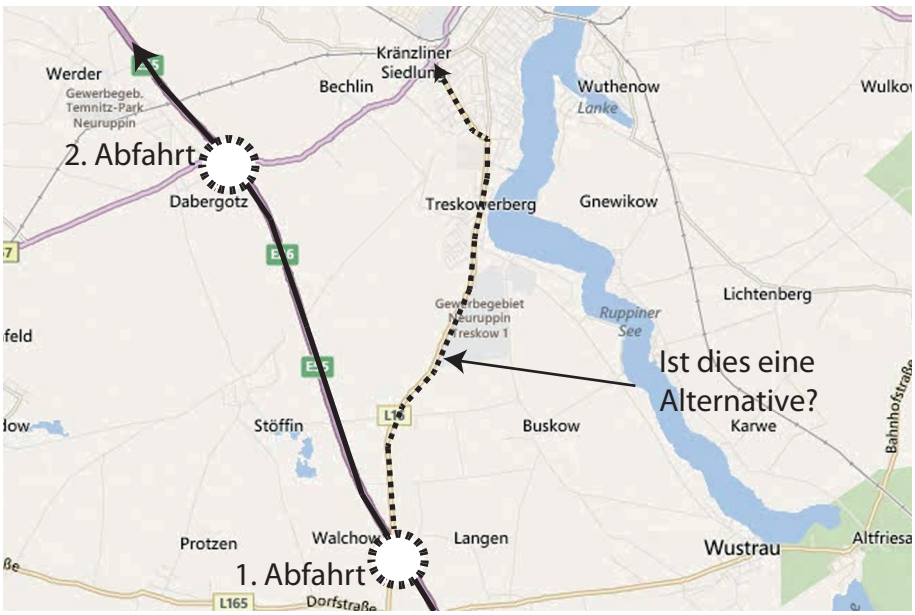

Abbildung 3.4: Entscheidungsproblem bei der deterministischen Berechnung von Alternativen

Hier gibt es nun mehrere Möglichkeiten wie man fortfahren kann. Die erste Möglichkeit ist, man fügt die markierte Strecke zur Menge der Alternativen hinzu. Falls man jedoch die Autobahn bei der nächsten Abfahrt nicht verlässt, müsste man die Alternative wieder zur Autobahn zurückführen. Ein Angreifer könnte eine solche Alternative leicht ausschließen, da es nicht plausibel ist, von der Autobahn zu fahren und diese bei der nächsten Auffahrt dann wieder zu befahren.
Die zweite Möglichkeit wäre, die markierte Strecke nicht zur Menge der Alternativen hinzuzufügen. Falls der Benutzer jedoch bei der zweiten Abfahrt von der Autobahn fährt, so hat man keine Alternative für diese Strecke ausgewählt. Damit wäre die Privatsphäre des Benutzers verletzt, da ein Angreifer nun eindeutig identifizieren kann, dass man an der zweiten Abfahrt von der Autobahn gefahren ist.

Des Weiteren muss ein deterministisches Verfahren sicherstellen, dass zu jedem Zeitpunkt mehrere Wege existieren, welche die gleiche Trajektorie erzeugen. Wäre dies nicht der Fall, so könnte ein Angreifer durch Probieren der einzelnen Routen auf die reale Strecke schließen.

Aufgrund der vorher genannten Probleme von on-line Verfahren, wird in dieser Arbeit ein Verschleierungsverfahren entwickelt, welches off-line die Verschleierung durchführt.

Es wird angenommen, dass sich der mobile Benutzer auf einem Straßengraphen bewegt. Der Schlauch soll dadurch entstehen, dass durch Vorgabe eines Zielpunktes alternative Strecken berechnet werden. Der Startpunkt ist hierbei bereits durch die aktuelle GPS-Position gegeben. Das Vorgehen kombiniert also den Ansatz des *Sendens von falschen Trajektorien* (vgl. Abschnitt 2.4.4) und die Berechnung von alternativen Routen, wie bei einem Navigationssystem. Damit werden die reale und auch die falschen Trajektorien im vornherein bestimmt. Somit ist es für einen Angreifer nicht möglich, Strecken aufgrund von bestimmten Mustern auszuschließen, da alle Strecken durch den gleichen Algorithmus entstanden sind.
Ein weiterer Vorteil dieses Ansatzes ist, dass die generierten Alternativen zielorientiert sind. Das bedeutet, dass jede generierte Trajektorie von einem Startgebiet in das gleiche Zielgebiet führt. Damit gibt es keine Teile einer Trajektorie, die diese bereits als falsche Trajektorie entlarven.
Zudem stellt das Verfahren sicher, dass sich die generierten Alternativen nur zu einem gewissen Grad gleichen. Dies ist notwendig, da sonst nur Alternativen der kürzesten Strecke berechnet werden würden.
Des Weiteren wird sichergestellt, dass die Strecken eine möglichst gleiche Fahrtdauer aufweisen. Für einen Angreifer ist es unwahrscheinlich, dass der Benutzer einen Umweg von einer Stunde Fahrzeit in Kauf nimmt.
Ein weiterer Sicherheitsaspekt ist, dass die einzelnen Positionen, die an den Lokationsserver gesendet werden, nicht abhängig von der realen Position des Benutzers sind. Das bedeutet, dass das Verfahren sich nur auf die möglichen Geschwindigkeiten, die im Straßengraph definiert sind, stützt. Deshalb ist es wichtig, um eine gewisse Dienstgüte einhalten zu können, das gewisse Zusicherungen eingehalten werden.

Um dieses Verschleierungsverfahren wie beschrieben umzusetzen, ist es zunächst notwendig ein Verfahren zu finden, welches verschiedene Alternativen im Straßengraphen zu berechnet. Sind diese Alternativen gefunden, so muss sichergestellt werden, dass die Start- und Zielpunkte ebenfalls nicht durch einen Angreifer erkannt werden können.

3.4 Berechnung von Alternativen in Straßengraphen

In den Entwurfsentscheidungen wurde beschrieben, was ein off-line Verfahren leisten muss, um die Privatsphäre des Benutzers zu schützen. Ein zentraler Aspekt ist hier die Generierung von verschiedenen Alternativen im Straßengraph. Bader et al. [Bad11] schlagen ein Verfahren vor, welches als Grundlage für die Generierung von verschiedenen Alternativen dienen soll:

Es sei ein Straßengraph $G = (V, E)$, ein Startpunkt **s** und einem Endpunkt **t** gegeben. Ziel ist es einen Alternativengraph (AG) zu berechnen der verschiedene Alternativen einer Strecke beinhaltet.

Es werden folgende Attribute definiert:

$$totalDistance := \sum_{e=(u,v)\in E'} \frac{w(e)}{d_H(s,u) + w(e) + d_H(u,t)}$$

$$averageDistance := \frac{\sum_{e\in E'} w(e)}{d_G(s,t) * totalDistance}$$

$$decisionEdges := \sum_{v\in V'\setminus t} outdegree(v) - 1$$

Das Attribut *totalDistance* entspricht einer Normierung. Das bedeutet, wenn ein Alternativengraph beispielsweise drei verschiedene Strecken beinhaltet, so erhält man hier 3 als Ergebnis. Das Attribut *averageDistance* beschreibt die durchschnittliche Länge der Strecken. Das Attribut *decisionEdges* beschreibt die Anzahl Entscheidungen, die ein Benutzer während der Fahrt treffen muss. Beispielsweise wenn er an eine Kreuzung kommt und entscheiden muss, welche Straße er weiterverfolgen muss.

Ziel von Bader et al. ist es die obenstehenden Attribute zu minimieren. Dabei wird wie folgt vorgegangen:

1. Schritt: Berechnung des kürzesten Weges **w** zwischen **s** und **t**

2. Schritt: Aufnahme des kürzesten Weges in den AG.

3. Schritt: Überprüfung der Attribute und eventuelles Verwerfen des Weges **w**

4. Schritt: Erhöhen der Gewichte Wege auf dem kürztesten Weg **w**.

5. Schritt: Weiter mit Schritt 1., bis eine Anzahl von Alternativen erreicht ist.

Nach der Generierung des Alternativgraphen kann man, durch Entfernen von überflüssigen Kanten, diesen minimieren [Bad11].

3.5 Schutz des Start- und Zielpunktes

Neben der Generierung von unterschiedlichen Alternativen, dürfen der Start- und Zielpunkt des Benutzers ebenfalls nicht erkannt werden. In [Kru07] wird ein Verfahren beschrieben, um die Startposition von Trajektorien zu verschleiern.

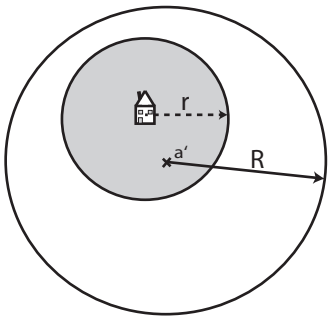

Abbildung 3.5: Verfahren zur Verschleierung von Start- und Zielpunkten nach [Kru07]

Durch GPS wird die aktuelle Position des Benutzers ermittelt. Um diese Position legt man einen Kreis mit dem Radius r. Innerhalb des Kreises ermittelt man einen Zufallspunkt a'. Um diesen Punkt a' wird ein zweiter Kreis gelegt mit dem Radius R (es gilt r < R). Alle Punkte, die innerhalb des zweiten Kreises liegen, dürfen nicht veröffentlicht werden. In [Kru07] wurde gezeigt, dass man bei einem Radius R von 2 km kaum Rückschlüsse auf die Startposition finden konnte.

3.6 Verwendete Metrik

Definition eines Straßengraphen

Ein Straßengraph G wird definiert, als ein gerichteter Graph $G = (V, E, w)$. Dabei entspricht V der Menge aller Knoten, die eine Kreuzung von verschiedenen Straßen darstellen. Die Menge E beschreibt die Kanten zwischen den Knoten, ein Teil einer Straße. w ist eine Funktion die jeder Kante ein Gewicht zuweist. In diesem Gewicht sollen möglichst alle Eigenschaften (z.B. Geschwindigkeit) des entsprechenden Streckenabschnittes enthalten sein. Die Kantengewichte innerhalb dieser Arbeit entsprechen einem Wert in Sekunden.

Der geometrische Abstand zweier Knoten wird mit $d(a, b)$ berechnet.

Die *Gleichheit zweier Trajektorien* T_1 und T_2 wird wie folgt definiert:

$T_1 = (V_1, E_1)$, $T_2 = (V_2, E_2)$

$$\forall\, e_1 \in E_1 \quad \exists \quad e_2 \in E_2 \mid e_1 = e_2 \wedge \forall\, e_2 \in E_2 \quad \exists \quad e_1 \in E_1 \mid e_2 = e_1$$

Die *Ähnlichkeit zweier Trajektorien* T_1 und T_2 wird wie folgt definiert:

$T_1 = (V_1, E_1)$, $T_2 = (V_2, E_2)$

Zunächst werden die Kanten die nur in einer Trajektorie vorkommen aufsummiert:

$$Summe_{T1} = \sum_{e=(u,v)\in E_1 \wedge \notin E_2} w(e)$$
$$Summe_{T2} = \sum_{e=(u,v)\in E_2 \wedge \notin E_1} w(e)$$

Anschließend werden die gemeinsamen Kanten aufsummiert:

$$Summe_{gleicheKanten} = \sum_{e=(u,v)\in E_2 \wedge e \in E_1} w(e)$$

Abschließend wird der Prozentsatz der Übereinstimmung ermittelt:

$$Ergebnis = Summe_{gleicheKanten} / (Summe_{T1} + Summe_{T2} + Summe_{gleicheKanten})$$

Der *maximaler Abstand* zweier Trajektorien T_1 und T_2 wird wie folgt definiert:
Seien $t_1..t_n$ Zeitpunkte,
v_{t_n} sei der Knoten der Trajektorie T_2 zum Zeitpunkt t_n
u_{t_n} sei der Knoten der Trajektorie T_2 zum Zeitpunkt t_n

Berechne für jeden Zeitpunkt den Abstand der korrespondieren Knoten und behalte den maximalen Abstand:

$\forall t$ Abstand = max(Abstand, d(v_{t_n},u_{t_n}))

3.7 Ablauf des Verfahrens

Bevor die einzelnen Komponenten des Verfahrens erklärt werden, soll nun zunächst der allgemeine Ablauf der Verschleierung beschrieben werden (vgl. Abb. 3.6).

Das mobile Gerät besitzt eine GPS-Komponente und ermittelt in regelmäßigen Abständen die genaue Position. Der Benutzer wählt einen Zielpunkt t aus. Der Startpunkt entspricht der aktuellen GPS-Position. Diese Information wird im ersten Schritt der Komponente zur Alternativenberechnung weitergereicht. Des Weiteren hat der Benutzer die Möglichkeit, die Anzahl der Alternativen[1] festzulegen. Im zweiten Schritt werden die Alternativen nach dem Verfahren berechnet, welches im nächsten Abschnitt 3.8 genauer erklärt wird.
Der Benutzer entscheidet abschließend, wie bei einem Navigationsgerät, welche Alternative er benutzen möchte. Es sollte jedoch verhindert werden, dass sich der Benutzer immer für die kürzeste Strecke entscheidet. Dies hat den Hintergrund, dass ein möglicher Angreifer die kürzeste Strecke als wahrscheinlichste Strecke annimmt. Bei der Alternativenwahl sollte jede Strecke mit der gleichen Wahrscheinlichkeit vom Benutzer gewählt werden können. Eine Möglichkeit dies zu erreichen ist, eine zufällige Alternative dem Benutzer vorzuschlagen. Eine weitere Möglichkeit ist es, dass die genauen Streckeninformationen, wie zum Beispiel die Dauer der Fahrzeit, nicht angezeigt werden. Die gewählte Strecke hängt somit nicht von der berechneten Fahrtdauer ab, sondern von der Einschätzung des Benutzers.

Wurde die zu fahrende Strecke festgelegt, muss das mobile Gerät in bestimmten Abständen eine aktuelle Position an den Lokationsserver senden. Dafür wird im vierten Schritt eine Anfrage an die *Komponente zur Positionsermittlung auf dem Alternativengraph* gestellt. Diese Komponente berechnet auf Grundlage des Alternativengraph und der

[1] Der Wert sollte nicht zu groß sein. - In der Evaluation wurde ein Wert von 3 angenommen.

übermittelten Fahrtdauer t, die aktuellen Positionen. Es werden immer genauso viele Positionen zurückgeliefert, wie es Alternativen gibt. Eine weitere wichtige Aufgabe dieser Komponente ist die Überwachung der Zusicherungen des Verfahrens. Dies wird in Abschnitt 3.9.2 erklärt.

Nachdem die Positionsinformationen nun der Anwendung auf dem mobilen Gerät zur Verfügung stehen, werden diese Informationen an den Lokationsserver gesendet.

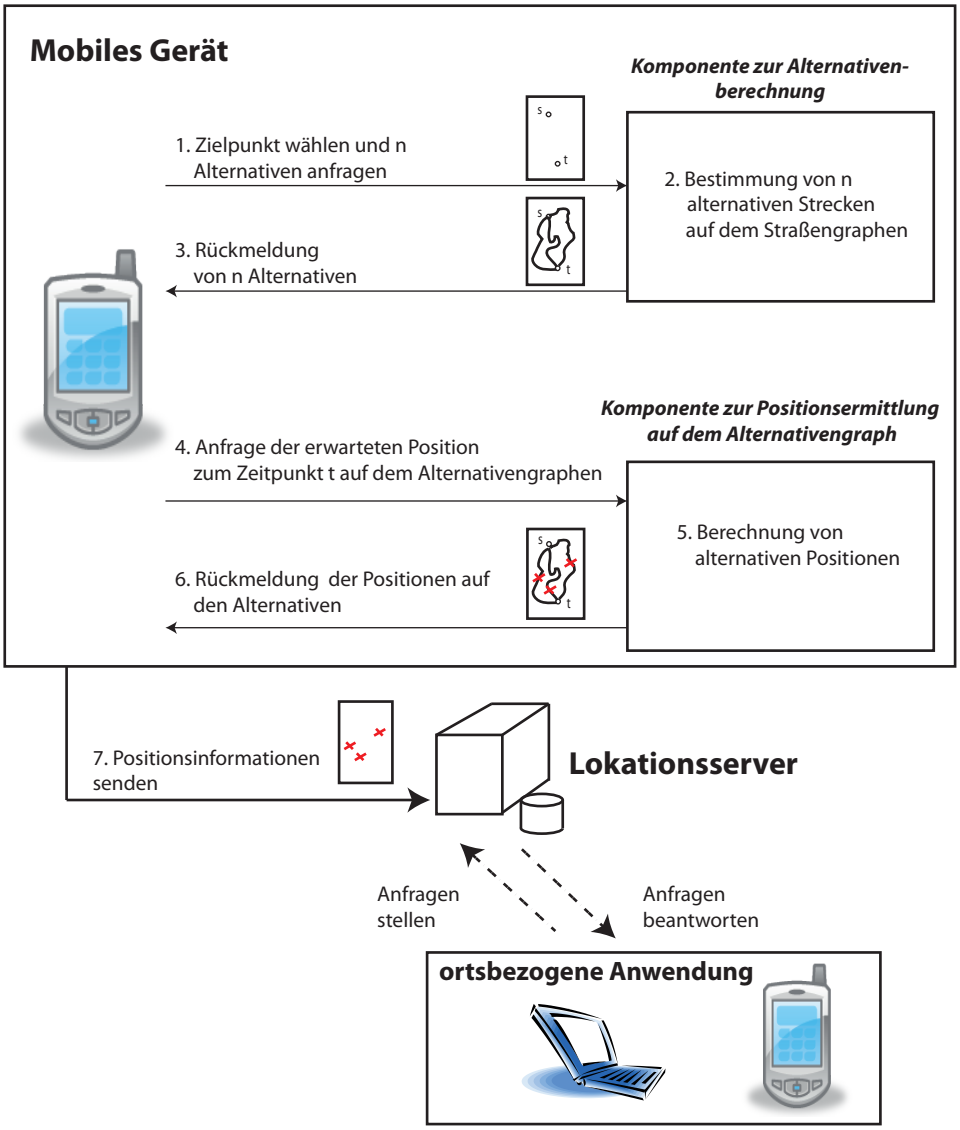

Abbildung 3.6: Ablaufmodell

3.8 Komponente zur Generierung von alternativen Routen

Der nachfolgende Pseudocode beschreibt die Generierung von Alternativen Routen des entwickelten Verfahrens. Das Verfahren basiert auf dem Verfahren von [Bad11] aus Abschnitt 3.4. Das Verfahren wurde so erweitert, dass mehrere Start- und Zielpunkte akzeptiert werden.

Algorithmus 3.1 Pseudocode: Bestimmung von alternativen Pfaden

Menge der Alternativen: AG
Graph $G = (V, E, w)$
Startpunkte $S_1..S_n$
Zielpunkte $T_1..T_n$
Maximale Deckung o_{max} = [0...1]
Rejoin Penalty: p_{rejoin}
Maximale Abwechung zur kürzesten Strecke d= $]1, 2[$
Anzahl von Alternativstrecken: a_{max}

Für jedes Paar (S_n, T_n) tue {
G' = G, Anz = 0
1. Berechne den kürzesten Weg w zwischen s und t.
2. Füge den Weg s...t in AG
3. Erhöhe das Gewicht der Kanten des Weges w um p_{use}
4. Erhöhe das Gewicht aller Kanten die den Weg w in G schneiden um $p_{penalty}$
5. Erhöhe Anz um 1

Solange Anz < a_{max} tue {
1. Berechne den kürzesten Weg w zwischen s und t.
2. Überprüfe ob der Weg w kürzer ist als (kürzester Weg * d) => Nein = Abbruch
3. Ja -> Erhöhe das Gewicht des Weges in G' um p_{use}
4. Erhöhe das Gewicht aller Kanten die den Weg w in G schneiden um $p_{penalty}$
5. Erhöhe Anz um 1;
6. Füge w in AG
}
}

Anschließend teilt die Komponente dem Benutzer die berechneten Wege mit. Dieser kann dann zwischen den berechneten Alternativen wählen. Hat der Benutzer sich für eine Alternative entschieden, ist die Komponente zur Positionsbestimmung zustän-

dig dafür, die aktuelle Position zu einem Zeitpunkt t auf dem Alternativgraphen zu ermitteln. Diese Komponente wird im nächsten Abschnitt beschrieben.

3.9 Komponente zur Positionsbestimmung

Die Komponente zur Positionsbestimmung hat im wesentlichen zwei Hauptaufgaben. Zum einen übernimmt Sie die Ermittlung von Positionsinformationen zu einem Zeitpunkt t. Zum anderen ist sie dafür zuständig die Zusicherungen des Verfahrens zu gewährleisten.

3.9.1 Ermittlung der Positionsinformation

Da der Lokationsserver nur Positionsinformationen aufgrund der vorausberechneten Alternativen bekommt, muss zu einem gewissen Zeitpunkt t die aktuelle Position auf den berechneten Alternativen ermittelt werden. Da jede Kante des Straßengraphen bereits die Zeit als Kantengewicht besitzt, müssen also nur die Kantengewichte aufaddiert werden bis man die vorgegebene Zeit t erreicht. Der letzte besuchte Knoten, bei dem die Summe der Kantengewichte kleiner als t ist, wird dann zur Ergebnismenge hinzugefügt.

3.9.2 Zusicherungen des Verfahrens

Die Positionskomponente kümmert sich weiterhin um die Einhaltung von Zusicherungen, die mit dem Verfahren erfüllt werden sollen. Diese Art von Zusicherungen können je nach ortsbezogener Anwendung unterschiedlich stark ausgelegt werden. Genauso kann man diese Zusicherungen entweder zeitlich oder mit Angabe eines Radius realisieren.

Eine *zeitliche Zusicherung* könnte also lauten:
Wenn man sich auf der gewählten Trajektorie bewegt, so kann man den ausgegeben Punkt p innerhalb von 100 Sekunden erreichen.

Eine *örtliche Zusicherung* könnte also lauten:
Wenn man sich auf der gewählten Trajektorie bewegt, so befindet man sich vom ausgegeben Punkt p maximal 500 Meter entfernt.

3.9.3 Verlassen der Zusicherungen

Die Zusicherungen garantieren, dass eine gewisse Dienstgüte eingehalten wird. Wird der Bereich der Zusicherungen verlassen, so muss es eine Strategie geben wie weiter vorgegangen wird.

Die erste Möglichkeit ist, dass die Verschleierung mit einer neuen Benutzeridentifikation neu gestartet wird. Dies trifft allerdings nur auf Dienste zu, die auf eine eindeutige Benutzeridentifikation verzichten können. Die Benutzeridentifikation dient bei diesen Diensten nur dazu, dass der Benutzer sich selbst identifizieren kann. Man muss jedoch berücksichtigen, dass ein Angreifer den Wechsel anhand der Zeitstempel der gesendeten Punkte und des Gebietes, in dem sich der Benutzer befunden hat, identifizieren könnte.
Die zweite Möglichkeit ist, dass man die Gewichte des Straßengraph anpasst. Ist ein Benutzer zu langsam, so müsste man die Kantengewichte entsprechend verringern. Dies bietet jedoch auch die Möglichkeit für einen Angreifer die Strecke zu erkennen, auf der sich der mobile Benutzer in Wirklichkeit bewegt. Hat der Angreifer beispielsweise Kenntnis darüber, dass auf einer Alternative ein Stau ist, so kann er daraus schließen, dass sich der Benutzer wahrscheinlich auf der Strecke mit der Verkehrsbehinderung aufhält.
Die dritte Möglichkeit ist, die Zusicherungen zu ignorieren. Dies ist vor allem für diese Dienste eine Lösung, in der die Zusicherungen nicht von Bedeutung sind. Werden dem Benutzer beispielsweise Sehenswürdigkeiten in der Nähe zurückgeliefert, so könnte das mobile Gerät diese Ergebnisse auch zwischenspeichern. Erreicht der Benutzer dann ein relevantes Gebiet, kann das mobile Gerät die Ergebnisse aus dem Zwischenspeicher verwenden. Da der mobile Nutzer vor Fahrtantritt bestimmt hat, auf welcher Route er fahren möchte, kann jederzeit überprüft werden, ob man noch auf der korrekten Route fährt.
Die letzte Möglichkeit ist, dass das Verfahren abgebrochen werden, muss und der Dienst nicht weiter genutzt werden kann.

Die Ursachen weshalb Zusicherungen nicht eingehalten werden können vielfältig sein. Typische Ursachen weshalb eine Fahrt zu langsam sein kann:

a) Stillstand durch Ampeln

b) Andere Geschwindigkeiten durch Behinderungen (z.B. Stau, Baustellen)

c) Behinderung durch andere Verkehrsteilnehmer (z.B. Schwertransporte)

d) Wetterbedingungen (z.B. Nebel/Regen)

e) Geänderte Geschwindkeitsvorgaben der Straße

f) Pausen des Fahrers

Ursachen für eine zu schnelle Fahrt:

a) Geänderte Geschwindkeitsvorgaben der Straße

b) Überhöhte Geschwindigkeit des Fahrers

3.9.4 Staudienst

Wie bereits gezeigt wurde, ist das Verlassen der Zusicherungen kritisch. Wenn unvorhergesehene Ereignisse auftreten, wie zum Beispiel ein Stau, wird das mobile Gerät schnell den Zusicherungsbereich verlassen. Um dies zu verhindern, wird ein Staudienst eingefügt. Dieser Dienst empfängt auftretende Änderungen der Geschwindigkeit und passt daraufhin die Gewichte des Straßengraphen und die Gewichte der berechneten Alternativen an.

3.10 Berücksichtigung von Topologie- und Geschwindigkeitsrestriktionen

Das vorgestellte Verfahren berücksichtigt neben Topologierestriktionen auch Geschwindigkeitsrestriktionen. *Topologierestriktionen* entstehen, wenn ein Angreifer zusätzliches

Wissen zu einem verschleierten Bereich hat. Dies passiert wenn ein Angreifer beispielsweise eine Straßenkarte hinter den Verschleierten Bereich legt und damit bestimmte Bereiche aus dem verschleierten Bereich ausschließen kann. Das entwickelte Verfahren berücksichtigt Topologierestriktionen dadurch, dass es selbst nur auf einem Straßengraphen arbeitet.

Geschwindigkeitsrestriktionen entstehen dadurch, dass ein Angreifer über Geschwindigkeitseigenschaften bestimmte verschleierte Bereiche ausschließen kann. Abbildung 3.7 zeigt ein Startbereich S und die Bereiche A und B welche gleichzeitig veröffentlicht wurden. Der Angreifer könnten nun durch Betrachtung der möglichen Geschwindigkeiten berechnen, ob er die Bereiche A und B erreichen kann. Man erkennt, dass die Strecke s_1 zu kurz ist um Bereich A zu erreichen. Strecke $s_2$ist jedoch lang genung. Damit kann der Angreifer den Bereich A ausschließen. Das entwickelte Verfahren berücksichtigt Geschwindigkeitsrestriktionen dadurch, dass immer die zulässige maximale Geschwindigkeit als Grundlage der Berechnung auf dem Straßengraphen herangezogenen wird. Somit werden nur gültige Punkte veröffentlicht.

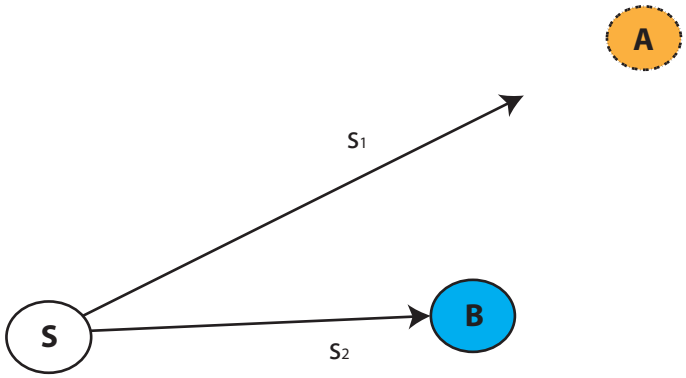

Abbildung 3.7: Berücksichtigung von Geschwindigkeitsrestiktionen

3.11 Berücksichtigung einer zeitlichen Komponente

Die Einhaltung von Zusicherungen ist für das Verfahren wichtig, da für die Verwendung einer ortsbezogenen Anwendung eine gewisse Dienstgüte erhalten werden soll. Dazu wurde bislang überprüft, ob man die Zusicherungen für eine gewählte Alter-

native einhält. Dieses Vorgehen hat jedoch den Nachteil, dass Zusicherungen schwer einzuhalten sind, wenn Probleme auf der gewählten Alternative auftreten. Beim bisherigen Vorgehen wurde von einem festen Startpunkt zu einem festen Zielpunkt m Alternativen berechnet. Um zu erreichen, dass sich die Alternativen unterscheiden, wurde vom Benutzer ein maximaler Grad der Überdeckung vorgegeben. Dies führt jedoch dazu, dass die Trajektorien weit voneinander entfernt sind und damit ein Wechsel kaum möglich ist.

Um dieses Problem zu lösen, wurden zusätzliche $n - 1$ Start- und Zielknoten eingefügt. Es werden also m Alternativen von jedem Start- zu jedem Zielpunkt berechnet. Insgesamt gibt es somit $m * n^n$ Alternativen.

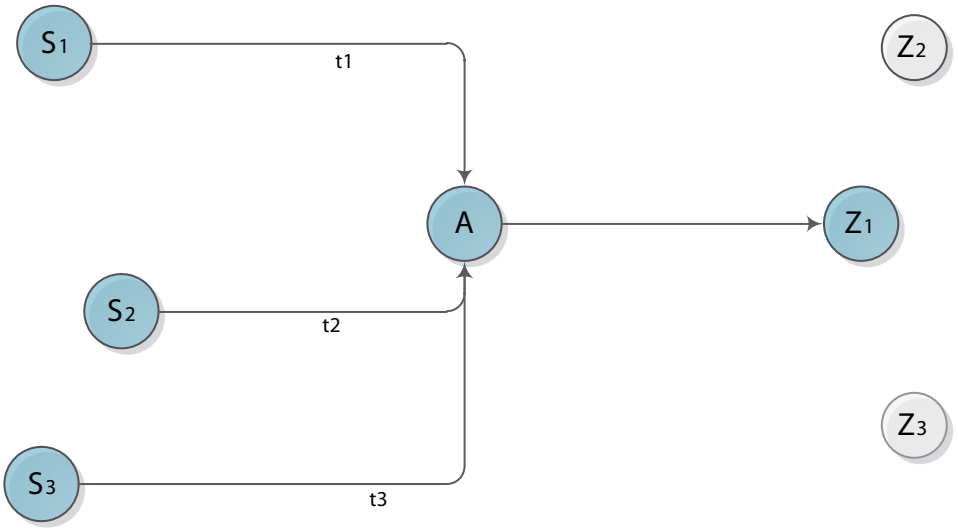

Abbildung 3.8: Zeitliche Komponente bei mehreren Start- und Zielpunkten (Ausschnitt aus den Alternativen)

Anhand von Abbildung 3.8 soll beschrieben werden, was unter dem Wechsel der Trajektorien zu verstehen ist. In diesem Beispiel wurden 3 Startpunkte generiert: S_1, S_2 und S_3. t_1, t_2 und t_3 beschreiben jeweils die Zeit, die man benötigt um auf dieser Kante zum nächsten Knoten zu gelangen. Ohne Beschränkung der Allgemeinheit soll gelten: $t1 \leq t2 \leq t3$.
Von jedem Startpunkte wurden m Alternativen zu jedem Zielpunkte berechnet (in der Abbildung ist allerdings nur ein Ausschnitt dargestellt). Die dargestellten Trajektori-

en haben den Knoten A gemeinsam, den sie allerdings zu unterschiedlichen Zeiten erreichen.

Angenommen, man befände sich auf der Trajektorie [S_1, A, Z_1] und kann aufgrund von Behinderungen die Zusicherungen, die am Knoten A gefordert sind, nicht mehr einhalten. Würde nur eine Trajektorie über diesen Knoten verlaufen, müsste man eine Strategie aus Abschnitt 3.9.3 wählen, um fortzufahren. In diesem Beispiel gibt es jedoch mehrere Trajektorien, die über Knoten A verlaufen. Somit kann man in diesem Fall überprüfen, ob die geforderten Zusicherungen am Knoten A noch für eine andere Trajektorie gelten, z.B. [S_3, A, Z_1]. Wenn dies der Fall ist, so kann man die Trajektorien wechseln. Andernfalls müsste man trotzdem eine Strategie aus dem Abschnitt 3.9.3 verwenden.

Der Nachteil des Hinzufügens von Start- und Zielpunkten ist der zusätzliche Aufwand. Zum einen ist der Rechenaufwand höher, denn es müssen $m * n^n$ Alternativen berechnet werden. Zum anderen müssen bei jeder Aktualisierung des Lokationsservers jeweils $m * n^n$ Alternativpositionen gesendet werden. Da die hinzugenommenen Startpunkte immer in der Nähe des orginalen Startpunktes sind, kann ein Benutzer durch dieses Vorgehen wahrscheinlich nur kleinere Behinderungen überbrücken.

Anhand der Abbildungen 3.9 und 3.10, sollen die Gemeinsamkeiten zwischen zwei Berechnungen herausgearbeitet werden. In Abbildung 3.9 werden von einem Startpunkt zu einem Zielpunkt drei alternative Wege berechnet. In der Abbildung 3.10 hingegen wurden jeweils noch zwei Startpunkte und zwei Zielpunkte hinzugefügt. Die zusätzlichen Punkte hatten einen Mindestabstand von 500 Meter zum Originalpunkt und durften maximal 1,5 Kilometer davon entfernt liegen. Es wurden dann jeweils drei Alternativen von jedem Startpunkt zu jedem Zielpunkt berechnet. Damit werden in Abbildung 3.10 insgesamt 27 Alternativen dargestellt.
Wie man anhand der Abbildungen erkennen kann, sind für verschiedene Startpunkte und Zielpunkte komplette Streckenabschnitte identisch. Dies wird später noch in der Evaluation im Abschnitt 4.4 gezeigt.

3.12 Implementierung

Das vorgestellte Verschleierungsverfahren wurde in der Programmiersprache Java implementiert. Dazu wurden entsprechende Klassen umgesetzt um einen routing-fähigen Straßengraphen einlesen und verwalten zu können. Zudem wurden die Komponenten zur Berechnung von alternativen Routen in Straßengraphen und die Komponente

Abbildung 3.9: Je drei Alternativen von einem Startpunkt zu einem Zielpunkt (Berlin)

Abbildung 3.10: Je drei Alternativen von drei Startpunkten zu drei Zielpunkten (Berlin)

zur Positionsbestimmung umgesetzt. Die Berechnungen der kürzesten Wege auf dem Graphen, wurden mit der Graphen-Bibliothek JGraphT[2] umgesetzt.

[2] http://www.jgrapht.org

Kapitel 4

Evaluation

In diesem Kapitel wird das entwickelte Verschleierungsverfahren aus Kapitel 3 evaluiert. Dabei werden zunächst die Evaluationsbedingungen im Abschnitt 4.1 geklärt. Im folgenden werden verschiedene Untersuchungen durchgeführt, um die reale Anwendbarkeit des off-line Ansatzes zu testen. Danach wird im Abschnitt 4.2 überprüft, ob für beliebige Start- und Zielpunkte, innerhalb verschiedener Gebiete, gültige Alternativen berechnet werden können. Anschließend wird anhand von realen Taxi-Trajektorien vom Projekt Geolife[1] überprüft, ob die ermittelten Alternativen des Verfahrens auch realen Trajektorien entsprechen. Abschließend werden im Abschnitt 4.5 die berechneten schnellsten Routen mit den schnellsten Routen von Google Maps verglichen.

[1] `http://research.microsoft.com/en-us/projects/geolife/`

4.1 Evaluationsbedingungen

Um das entwickelte Verfahren aus Kapitel 3 bewerten zu können, werden zunächst die Evaluationsbedingungen vorgestellt.

4.1.1 Kartenmaterial

Bei den verwendeten Kartendaten handelt es sich ausschließlich um orginale OpenStreetMap-Dateien[2]. Wenn eine Verkleinerung eines Bereiches notwendig war, so wurde das Tool *Osmosis*[3] benutzt. Dieses Tool erlaubt es, über die Kommandozeile eine OpenStreetMap-Karte einzulesen und durch die Angabe eines Rahmens die Karte zu verkleinern.

4.1.2 Vorbereitung des Kartenmaterials

Da das Kartenmaterial, neben den Straßeninformationen, viele weitere Informationen zu Häusern, Gelände usw. enthielt, musste das Material aufbereitet werden. Dafür wurde das Kommandozeilenprogramm OSM2Routing[4] benutzt. Das Programm erlaubt eine OpenStreetMap-Karte einzulesen und gibt als Ausgabe eine routingfähige OpenStreetMap-Karte zurück. Sofern keine Geschwindigkeitsangaben in den Kartendaten vorhanden waren, wurden von OSM2Routing vorkonfigurierte Standardwerte verwendet. Für die interne Verarbeitung wurden nur Kreuzungen aus den Kartendaten benutzt, da die Berechnung sonst unnötig viele Knoten berücksichtigt hätte. Dazu wurden unnötige Knoten zusammengefasst.

[2] `http://www.openstreetmap.org`
[3] `http://wiki.openstreetmap.org/wiki/Osmosis`
[4] `http://wiki.openstreetmap.org/wiki/Routing/Travel_Time_Analysis`

4.2 Alternativrouten in verschiedenen Gebieten in Deutschland

Eine grundlegende Frage für das entwickelte Verfahren ist, ob es für beliebige Start- und Zielpunkte Alternativrouten gibt. Durch die Alternativrouten wird der Schlauch erzeugt, in dem sich der Benutzer geschützt bewegt. Falls diese Alternativrouten immer existieren ist eine wichtige Frage, wieviele Alternativen gibt es.

Im folgenden wurde diese Fragen für verschiedene Gebiete in Deutschland überprüft, nämlich Berlin, Mecklenburg-Vorpommern und das Saarland. Diese Gebiete wurden gewählt, weil sie unterschiedliche Eigenschaften besitzen. Berlin besitzt ein dichtes Netz von Straßen. Mecklenburg-Vorpommern ist eine größere Region mit verschiedenen Aspekten. Zum einen ist der Großteil des Bundesland eher ländlich, aber es existieren auch einige größere Städte wie Lübeck, Rostock und Schwerin. Außerdem wurde das Saarland gewählt, weil es eine recht kompakte Region ist, welche trotz der geringen Größe, viele Autobahnen und Bundesstraßen besitzt.

4.2.1 Untersuchung der Existenz von Alternativen

Bei dieser Untersuchung wurden auf dem eingelesenen Straßengraphen Zufallspunkte generiert, welche einen Mindestabstand von 15 km (geometrisch) aufweisen mussten. Diese Punkte stellten den Start- und Zielpunkt eines Benutzers dar. Anschließend wurde das Verfahren zur Generierung von Alternativ-Routen durchgeführt. Es wurde die Anzahl der Generierungen gemessen, um 200 gültige Alternativ-Routen zu erzeugen.

Dabei wurden zwei Szenarien untersucht.

Szenario 1: Alternativrouten dürfen eine maximale Überdeckung von 30% aufweisen.
Szenario 2: Alternativrouten dürfen eine maximale Überdeckung von 20% aufweisen.

	Saarland	**Berlin**	**Mecklenburg**
Szenario 1	217	216	218
Szenario 2	230	227	230

Tabelle 4.1: Anzahl der benötigten Generierungen von alternativen Trajektorien um 200 gültige Datensätze zu erstellen

Die Tabelle 4.1 stellt die Anzahl der benötigten Durchläufe dar, um für 200 Start- und Zielpunkte mindestens drei Alternativen zu generieren. Dabei wurden in allen drei Gebieten ähnliche Werte ermittelt. Für Szenario 1 ergibt sich ein Durchschnitt von rund 92 Prozent. Für Szenario 2 ergibt sich ein Durchschnitt von ungefähr 86 Prozent. Damit scheint die Anzahl der notwendigen Generierungen abhängig zu sein von der gewählten maximalen Überdeckung.

4.2.2 Untersuchung der Streckenlänge und Fahrtdauer

Von den generierten Alternativen der vorhergehenden Untersuchung, wurden die jeweiligen Streckenlängen und Fahrzeiten ermittelt. Durch diese Untersuchung soll eine Einschätzung der Qualität der Routen stattfinden.

	Saarland	**Berlin**	**Mecklenburg**
S1 - Alternative 1	42 km (30 min)	27 km (26 min)	123 km (77 min)
S1 - Alternative 2	44 km (34 min)	28 km (29 min)	122 km (91 min)
S1 - Alternative 3	46 km (39 min)	29 km (31 min)	131 km (102 min)
S2 - Alternative 1	42 km (30 min)	27 km (27 min)	125 km (80 min)
S2 - Alternative 2	44 km (35 min)	29 km (30 min)	126 km (95 min)
S2 - Alternative 3	47 km (41 min)	30 km (33 min)	133 km (107 min)

Tabelle 4.2: Durchschnittliche Streckenlänge und Fahrtdauer von Alternativen Routen im Saarland, Berlin und Mecklenburg-Vorpommern (S1 - Szenario 1 / S2 - Szenario 2)

Die Tabelle 4.2 zeigt, dass die durchschnittliche Streckenlänge zwischen den Alternativen ungefähr gleich ist. Dennoch gibt es deutliche Unterschiede bei der Fahrzeit. Im Saarland beträgt der durchschnittliche Unterschied zwischen der schnellsten Alternative 1 und der langsamsten Alternative 3 ca. 36%. In Berlin beträgt der durchschnittliche Unterschied zwischen der schnellsten Alternative 1 und der langsamsten Alternative 3 ca. 22%. In Mecklenburg-Vorpommern beträgt der durchschnittliche Unterschied zwischen der schnellsten Alternative 1 und der langsamsten Alternative 3 ca. 33%.

Insgesamt lässt sich festhalten, dass zu jedem der 200 Start- und Zielpunkte mindestens 3 Alternativrouten gefunden werden konnten. Zwei Alternativen davon weisen eine ähnliche Fahrzeit auf. Daher sind sie für das Verfahren gut verwendbar. Die dritte Alternative ist ungefähr 30 % langsamer als die kürzeste Strecke. Dies ist noch akzeptabel für das Verfahren, jedoch sollte hier darauf geachtet werden, dass die Alternativen nicht deutlich schlechter werden. Daher ist es nicht sinnvoll mehr als 3 Alternativrouten zu

berechnen. Weitere Routen müssen noch mehr Einschränkungen beachten und können damit nur langsamere Fahrzeiten liefern.

4.2.3 Untersuchung des maximalen Abstandes der Alternativen

Bei dieser Untersuchung wird der maximale Abstand (vgl. Abschnitt 3.6) zwischen berechneten Alternativen betrachtet. Damit soll eine Einschätzung möglich sein, welche Dimension der Schlauch besitzt.

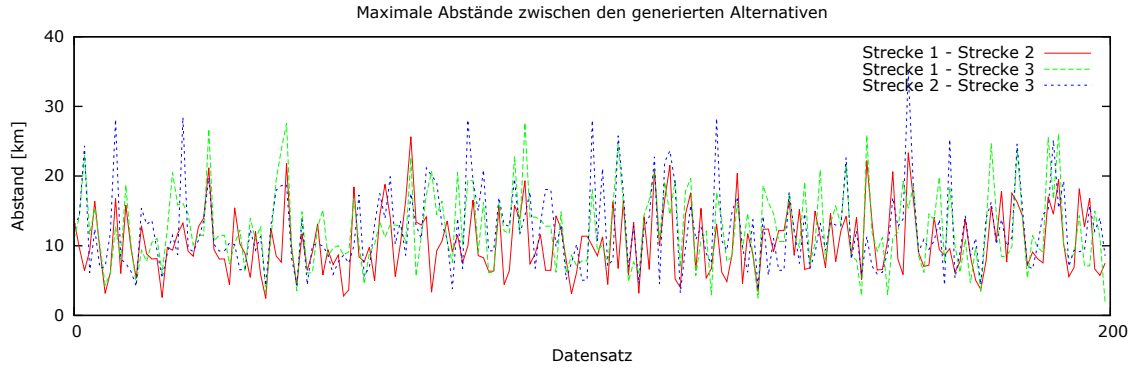

Abbildung 4.1: Abstände zwischen den einzelnen Alternativen - Saarland

Betrachtet man die Abbildungen 4.1, 4.2 und 4.3 so fällt auf, dass es deutliche Unterschiede zwischen den Abständen der Alternativen gibt. Die Ursache davon ist, dass die Streckenlänge unterschiedlich ist. Des Weiteren ist es nicht verwunderlich, dass bei einem dichten Straßennetz, wie Berlin, die Alternativrouten näher beieinander liegen.

	Saarland	**Berlin**	**Mecklenburg**
Strecke1 - Strecke2	9,36 km	4,44 km	29,36 km
Strecke1 - Strecke3	11,39 km	5,56 km	32,95 km
Strecke2 - Strecke3	11 km	4,54 km	26,30 km

Tabelle 4.3: Maximale Abstände der generierten Alternativen im Saarland, Berlin und Mecklenburg-Vorpommern(Median)

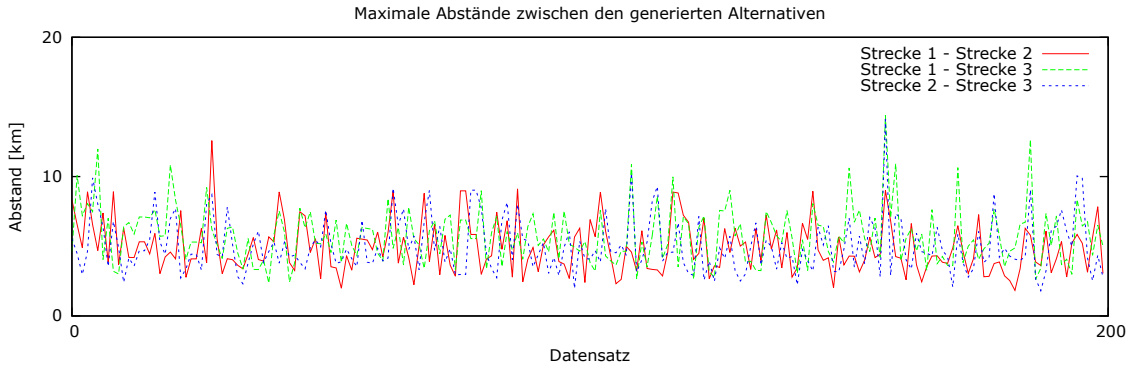

Abbildung 4.2: Abstände zwischen den einzelnen Alternativen - Berlin

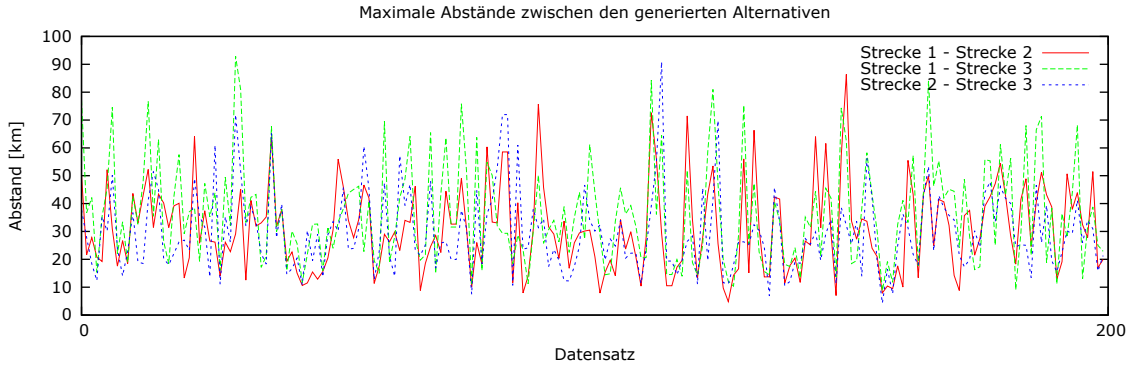

Abbildung 4.3: Abstände zwischen den einzelnen Alternativen - Mecklenburg-Vorpommern

Tabelle 4.3 zeigt den Median der maximalen Abstände zwischen den generierten Strecken. Die generierten Alternativen entfernen sich nicht zu weit von der kürzesten Strecke. Aufgrund der Streckenlängen und auch der unterschiedlichen Fahrzeiten ist der maximale Abstand (gemäß der vorgestellten Metrik) gut.

4.3 Bewertung des Verfahrens Anhand von realen Trajektorien

Anhand realer Trajektorien lassen sich Aussagen über die realen Anwendbarkeit des Verfahrens machen.

Dazu wurden Taxi-Trajektorien aus dem Projekt GeoLife verwendet [ZCL+10][ZLWX08][ZLC+08]. Dieses Projekt zeichnete über einen gewissen Zeitraum Positionsinformationen von Freiwilligen aus Peking auf. Teilweise wurden die Positionsdaten mit Markierungen versehen. Dies machte es möglich Taxifahrten herauszufiltern. Taxi-Trajektorien sind besonders interessant, da davon ausgegangen werden kann, dass sich ein Taxifahrer gut innerhalb einer Stadt auskennt. Das bedeutet, er umfährt zum Beispiel mögliche Staugebiete. Insgesamt wurden 74 Taxi-Trajektorien ausgewertet.

Zunächst soll überprüft werden, welchen maximal Abstand die generierten Alternativen zur realen Trajektorie aufweisen. Dazu wurden verschiedene Geschwindigkeiten getestet, nämlich die angenommene Maximalgeschwindigkeit, 80% der Maximalgeschwindigkeit und 50% der Maximalgeschwindigkeit.

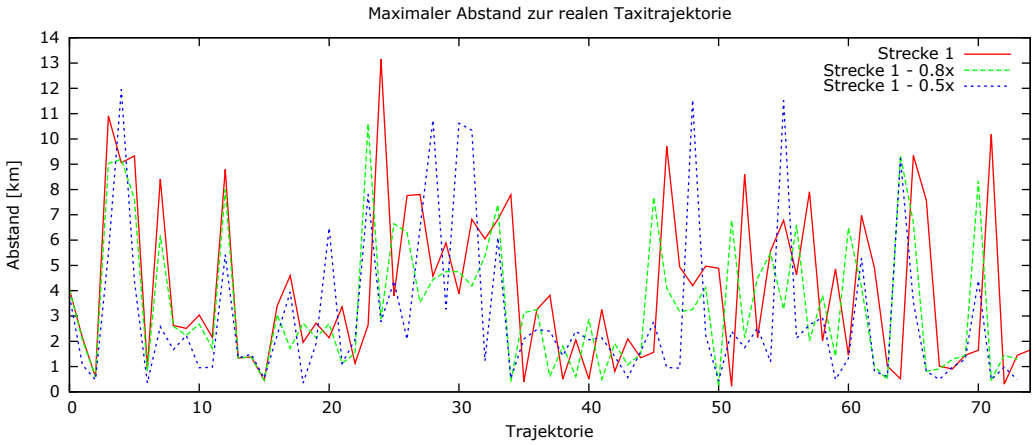

Abbildung 4.4: Maximaler Abstand der Taxi-Trajektorien zur ersten Strecke

Betrachtet man die Abbildung 4.4 so erkennt man, dass die Anpassungen der Geschwindigkeit für einzelne Trajektorien Vorteile gebracht haben. Dafür wurden bei anderen

die Abstände wieder größer. Der durchschnittliche Maximalabstand, bei Annahme der Maximalgeschwindigkeit, betrug: 3,86 km (Median: 3,16 km).
Bei 80% der Maximalgeschwindigkeit betrug der maximale Abstand lediglich 3,57 km (Median 2,55 km).
Bei 50% der Maximalabstand stieg der Abstand wieder auf durchschnittlich 4,4 km (Median: 3,15 km). Insgesamt sind die Abstände jedoch noch zu groß. Eine Ursache für die Abstände können beispielsweise sein, dass ein Taxifahrer länger an einer Ampel warten musste. Damit bringen auch Anpassungen an den Geschwindigkeiten keinen großen Vorteil. Ist die angenommene Geschwindigkeit zu langsam, so entfernt sich ein Taxifahrer schneller von der berechneten Alternative. Ist die angenommene Geschwindigkeit zu hoch, so entfernt sich die berechnete Alternative zu schnell von der Taxi-Trajektorie. Es wurde keine Anpassung der Geschwindigkeiten gefunden, welche für alle Taxi-Trajektorien galt. Von daher scheint es notwendig zu sein, dass der vorgeschlagene Staudienst aus Abschnitt 3.9.4 implementiert wird um weitere Untersuchungen vornehmen zu können.

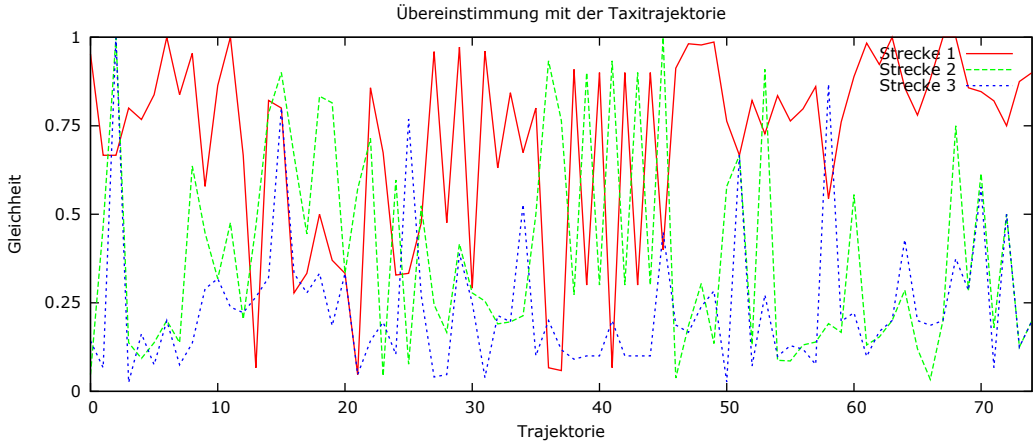

Abbildung 4.5: Übereinstimmung mit der Taxi-Trajektorie (50 m Ungenauigkeitsradius)

Die Abbildung 4.5 stellt die Übereinstimmung der generierten Strecke mit den realen Taxi-Trajektorien dar. Da die GPS Position ungenau sein kann, wurde ein Radius von 50 Meter angenommen. Das bedeutet, befindet sich ein Punkt der generierten Strecke in 50 Meter Entfernung zur Taxi-Trajektorie so ergibt sich eine Übereinstimmung. Insgesamt lässt sich feststellen, dass die realen Trajektorien mit allen generierten Alternativen

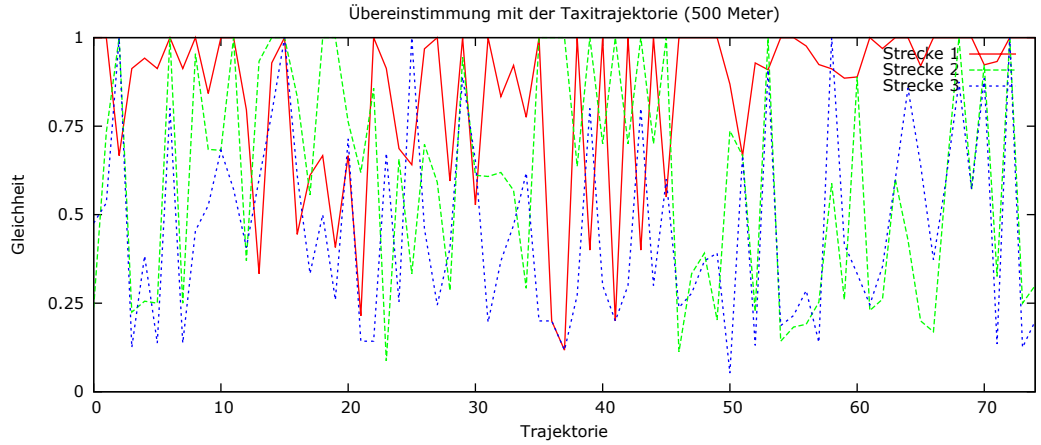

Abbildung 4.6: Übereinstimmung mit der Taxi-Trajektorie (500 m Ungenauigkeitsradius)

Übereinstimmungen aufweisen. Die meisten Übereinstimmungen existieren für die kürzeste Strecke (Strecke 1). Insgesamt lässt sich für die erste Strecke eine Überdeckung von durchschnittlich 70% feststellen. Die zweite Strecke weißt eine Überdeckung von durchschnittlich 38% auf. Die dritte Strecke weißt eine durchschnittliche Überdeckung von 23 % auf. Erweitert man den Ungenauigkeitsbereich auf 500 Meter (vgl. Abb. 4.6) um die Taxi-Trajektorie, so ändern sich diese Werte auf 83% für Strecke 1, 60% für Strecke 2 und 46% für Strecke 3. Das bedeutet also, dass die generierten Strecken nahe beieinander liegen, da sich bei der Ausweitung des Ungenauigkeitsbereiches, für jeder Strecke eine Änderung ergab.
Es lässt sich also festhalten, dass die generierten Alternativen zu einem großen Teil den realen Trajektorien von Taxifahrern entsprechen. Damit generiert das Verfahren relevante Alternativen. Jedoch muss, wenn man die vorhergehende Untersuchung des maximalen Abstandes berücksichtigt, die Bewertung der einzelnen Streckenabschnitte durch weitere Informationen angepasst werden. Hier könnte zum Beispiel der bereits angesprochene Staudienst Abhilfe schaffen.

4.4 Vergleich bei mehreren Start- und Zielpunkten

Wie bereits in Abschnitt 3.11 erwähnt, kann es notwendig sein, dass die gewählte Alternativroute verlassen werden muss. Dies kann notwendig sein, wenn die Zusicherungen einer Strecke nicht mehr eingehalten werden. Es wurde deshalb vorgeschlagen mehrere Start- und Zielpunkte einzufügen. Bei dieser Untersuchung wurden jeweils drei Startpunkte und drei Zielpunkte angenommen. Die zusätzlichen Start- und Zielpunkte lagen in einem Umkreis von 5 Kilometern um den eigentlichen Start-/Zielpunkt. Die Auswertung erfolgte mit Kartendaten von Peking. Es wurde ermittelt wie ähnlich die Strecken sind.

In Abbildung 4.7 wird die Überdeckung der kürzesten Strecke der zusätzlichen Startpunkte mit dem ursprünglichen Startpunkt dargestellt. Man erkennt, dass die kürzesten Strecken der neu generierten Startpunkte in vielen Fällen über die Strecke des kürzesten Weges des ursprünglichen Startpunktes verläuft. Daher ist hier der angesprochene Wechsel zwischen den verschiedenen Trajektorien möglich.

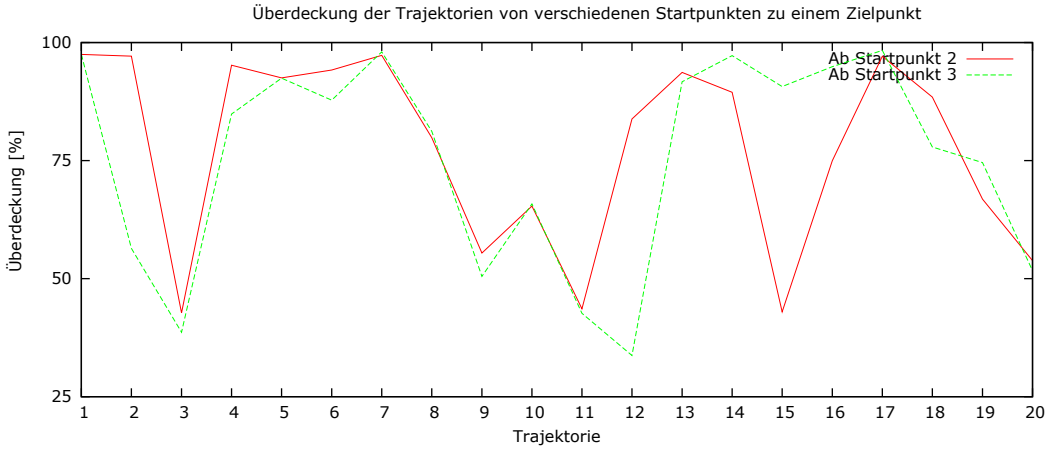

Abbildung 4.7: Überdeckung von Trajektorien mit verschiedenem Startpunkt und gleichem Zielpunkt

4.5 Vergleich mit der Routenplanung von Google Maps

Zum Abschluss der Evaluation soll das entwickelte Verfahren noch mit der Routenplanung von Google Maps vergleichen werden. Dazu werden zunächst durch das Verfahren zufällige Strecken generiert. Anschließend wird die Routenplanung von Google Maps mit dem gleichen Start- und Zielpunkt aufgerufen. Dabei wird zunächst die Länge der kürzesten Strecke vergleichen. Anschließend werden die ermittelten Fahrzeiten verglichen. Alle Routen wurden mit dem Straßengraphen des Saarland berechnet.

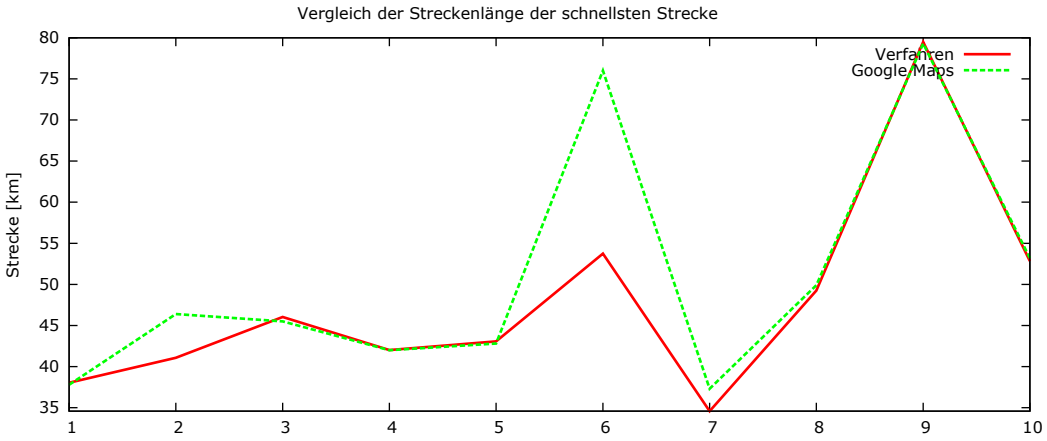

Abbildung 4.8: Vergleich der schnellsten Routen

In Abbildung 4.8 erkennt man, dass die ermittelten Strecken bis auf einen deutlichen Ausreißer ähnlich lang sind.

In Abbildung 4.9 erkennt man, dass die ermittelten Fahrtzeiten sich deutlich unterscheiden. Insgesamt sind die Fahrzeiten bei Google Maps ungefähr 35 % länger. Ähnlich wie in der Auswertung der realen Trajektorien erkennt man, dass von der bestehenden Navigationslösung von Google Maps eher pessimistische Werte für die Streckengeschwindigkeiten angenommen werden.

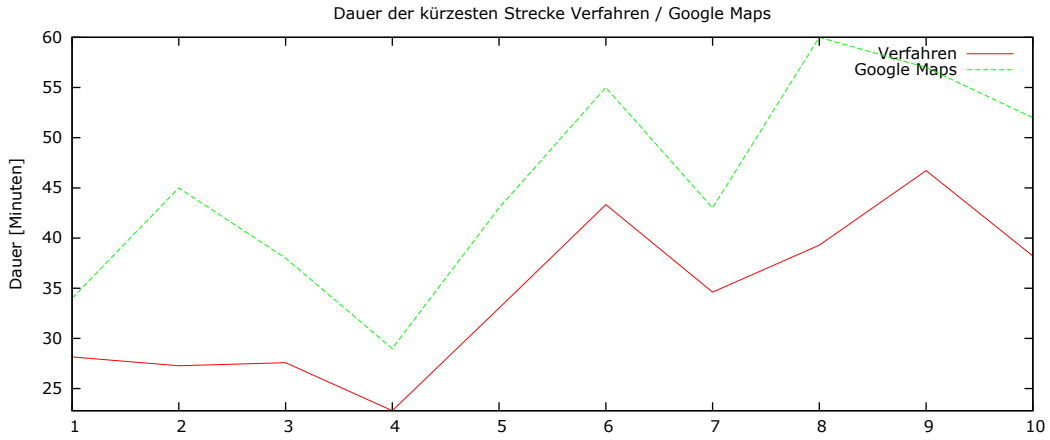

Abbildung 4.9: Vergleich der schnellsten Routen

Kapitel 5

Zusammenfassung

5.1 Fazit

Ziel der Arbeit war es, ein Verschleierungsverfahren für Benutzertrajektorien zu entwickeln. Es wurden zunächst verschiedene Verschleierungsverfahren für einzelne Positionsinformationen untersucht und festgestellt, dass diese für die Verschleierung von kompletten Trajektorien nicht ausreichend sind. Um Benutzertrajektorien zu schützen wurde gezeigt, dass ein optimales Verfahren es schafft einen Schlauch um eine reale Trajektorie zu legen.
Es wurde gezeigt, dass die on-line Generierung des Schlauches ein komplexes Problem darstellt, da ein deterministisches Verfahren zu bestimmten Zeitpunkten entscheiden muss, welche Strecken zum Schlauch hinzugenommen werden. Deshalb wurde in dieser Arbeit ein off-line Verfahren entwickelt, welches die Idee der Dummy-Ansätze mit der Generierung von alternativen Routen, wie bei einem Navigationssystem, verbindet. Bei diesem Verfahren werden Topologie- und Geschwindigkeitsrestriktionen berücksichtigt. Nachdem der Benutzer eine der generierten Routen gewählt hat, werden während der Fahrt Positionsinformationen, nur auf Grundlage der vorausberechneten alternativen Routen, an einen Lokationsserver gesendet. Somit können keine Rückschlüsse auf die reale Trajektorie gezogen werden.

Es wurde gezeigt, dass das Verfahren bei zufällig gewählten Start- und Zielpunkten in 85 % der Fälle unterschiedliche alternative Strecken generieren konnte. Bei der Untersuchung des Verfahrens mit real gefahrenen Trajektorien wurde festgestellt, dass die generierten Alternativen zu diesen ähnlich sind. Es zeigten sich jedoch große Unterschiede bei der Betrachtung der Fahrtdauer. Dies resultierte aus der Annahme,

dass ein Benutzer immer mit der maximal zulässigen Geschwindigkeit fährt. Dies stellt ein Problem bei der Einhaltung von Zusicherungen, die das Verfahren gewährleisten muss, dar.

Insgesamt lässt sich feststellen, dass das entwickelte Verfahren keine Angriffsmöglichkeiten bietet, da der Benutzer selbst die Alternative bestimmt, die er fährt.

5.2 Ausblick

Das im Fazit erwähnte Problem der Fahrtdauer, kann auf verschiedene Arten verringert werden. Zum einen besteht die Möglichkeit, dass der bereits erwähnte Staudienst implementiert wird. Dieser müsste die Aufgabe übernehmen, die Maximalgeschwindigkeiten anhand von verschiedenen Faktoren anzupassen. Dazu zählen zum Beispiel Wetterbedingungen, Berufsverkehr oder Baustellen.

Weiterhin besteht die Möglichkeit, das Verfahren so anzupassen, dass der Benutzer eine ungefähre Angabe seiner Fahrgeschwindigkeit macht. Diese Methode wird bereits in Navigationslösungen verwendet, um die Fahrtdauer abschätzen zu können.

Da es bislang nur möglich war auf reale Trajektorien vom Projekt GeoLife zurückzugreifen, wäre es zudem sinnvoll das Verfahren anhand von anderen realen Trajektorien zu überprüfen.

Literaturverzeichnis

[ACD+07] ARDAGNA, C. ; CREMONINI, M. ; DAMIANI, E. ; VIMERCATI, S. De Capitani d. ; SAMARATI, P.: Location privacy protection through obfuscation-based techniques. In: *Data and Applications Security XXI* (2007), S. 47–60 (Zitiert auf Seite 17)

[Bad11] BADER, Jonathan;Geisberger Robert; Sanders P. Roland ; Dees D. Roland ; Dees: Alternative Route Graphs in Road Networks, 2011 (Zitiert auf den Seiten 34, 35 und 40)

[BLPW08] BAMBA, B. ; LIU, L. ; PESTI, P. ; WANG, T.: Supporting anonymous location queries in mobile environments with privacygrid. In: *Proceeding of the 17th international conference on World Wide Web* ACM, 2008, S. 237–246 (Zitiert auf Seite 21)

[BS03] BERESFORD, A.R. ; STAJANO, F.: Location privacy in pervasive computing. In: *Pervasive Computing, IEEE* 2 (2003), Nr. 1, S. 46–55 (Zitiert auf Seite 16)

[CCLS11] CHENG, Z. ; CAVERLEE, J. ; LEE, K. ; SUI, D.Z.: Exploring millions of footprints in location sharing services. In: *AAAI ICWSM* (2011) (Zitiert auf Seite 15)

[Cze07] CZERNY, Jürgen: *Datenschutz in lokationsbasierten Diensten*. `http://dbis.ipd.uni-karlsruhe.de/img/content/SS07Czerny_DatenschutzLBS.pdf`. Version: 2007 (Zitiert auf Seite 30)

[DSR11] DURR, Frank ; SKVORTSOV, Pavel ; ROTHERMEL, Kurt: Position sharing for location privacy in non-trusted systems. In: *Proceedings of the 2011 IEEE International Conference on Pervasive Computing and Communications*. Washington, DC, USA : IEEE Computer Society, 2011 (PERCOM '11). – ISBN 978–1–4244–9530–6, 189–196 (Zitiert auf den Seiten 7, 19 und 28)

[EFH+11] EISNER, J. ; FUNKE, S. ; HERBST, A. ; SPILLNER, A. ; STORANDT, S.: Algorithms for Matching and Predicting Trajectories Citeseer, 2011 (Zitiert auf den Seiten 7 und 30)

[Gut06a] GUTSCHER, A.: Coordinate transformation-a solution for the privacy problem of location based services? In: *Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International* IEEE, 2006, S. 7–pp (Zitiert auf den Seiten 7 und 23)

[Gut06b] GUTSCHER, A.: Zugriffskontrolle für Ortsinformationen in Nexus-Eine Gratwanderung zwischen Sicherheit und Funktionalität. (2006) (Zitiert auf Seite 24)

[Kru07] KRUMM, J.: Inference attacks on location tracks. In: *Pervasive Computing* (2007), S. 127–143 (Zitiert auf den Seiten 7 und 35)

[Küp05] KÜPPER, A.: *Location-based services*. Wiley Online Library, 2005 (Zitiert auf Seite 27)

[KYS05] KIDO, H. ; YANAGISAWA, Y. ; SATOH, T.: An anonymous communication technique using dummies for location-based services. In: *Pervasive Services, 2005. ICPS'05. Proceedings. International Conference on* IEEE, 2005, S. 88–97 (Zitiert auf Seite 24)

[PK01] PFITZMANN, A. ; KÖHNTOPP, M.: Anonymity, unobservability, and pseudonymity—a proposal for terminology. In: *Designing privacy enhancing technologies* Springer, 2001, S. 1–9 (Zitiert auf Seite 20)

[SNE06] STEINIGER, S. ; NEUN, M. ; EDWARDES, A.: Foundations of location based services. In: *Lecture Notes on LBS* 1 (2006) (Zitiert auf Seite 28)

[SS98] SAMARATI, P. ; SWEENEY, L.: Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression / Technical report, SRI International. 1998. – Forschungsbericht (Zitiert auf Seite 20)

[SSH10] SAAKE, G. ; SATTLER, K.U. ; HEUER, A.: *Datenbanken–Konzepte und Sprachen*. mitp, 2010 (Zitiert auf Seite 20)

[XKP09] XUE, M. ; KALNIS, P. ; PUNG, H.: Location diversity: Enhanced privacy protection in location based services. In: *Location and Context Awareness* (2009), S. 70–87 (Zitiert auf Seite 21)

[YPL07] YOU, T.H. ; PENG, W.C. ; LEE, W.C.: Protecting moving trajectories with dummies. In: *Mobile Data Management, 2007 International Conference on* IEEE, 2007, S. 278–282 (Zitiert auf Seite 24)

[ZCL+10] ZHENG, Y. ; CHEN, Y. ; LI, Q. ; XIE, X. ; MA, W.Y.: Understanding transportation modes based on GPS data for web applications. In: *ACM Transactions on the Web (TWEB)* 4 (2010), Nr. 1, S. 1 (Zitiert auf Seite 55)

[ZLC+08] ZHENG, Y. ; LI, Q. ; CHEN, Y. ; XIE, X. ; MA, W.Y.: Understanding mobility based on GPS data. In: *Proceedings of the 10th international conference on Ubiquitous computing* ACM, 2008, S. 312–321 (Zitiert auf Seite 55)

[ZLWX08] ZHENG, Y. ; LIU, L. ; WANG, L. ; XIE, X.: Learning transportation mode from raw gps data for geographic applications on the web. In: *Proceedings of the 17th international conference on World Wide Web* ACM, 2008, S. 247–256 (Zitiert auf Seite 55)

Alle URLs wurden zuletzt am 11.04.2012 geprüft.

Erklärung

Hiermit versichere ich, diese Arbeit selbständig verfasst und nur die angegebenen Quellen benutzt zu haben.

(Robert Schmidt)