Institut für Visualisierung und Interaktive Systeme

Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Bachelorarbeit Nr. 361

Ein interaktiver visueller Ansatz für das Map Matching von großen Bewegungsdatensätzen

Georgi Simeonov

Studiengang:	Informatik
Prüfer/in:	Prof. Dr. Thomas Ertl
Betreuer/in:	M. Sc. Robert Krüger, Dr. Fabian Beck
Beginn am:	8. August 2016
Beendet am:	7. Februar 2017
CR-Nummer:	H.5.2, I.3.6

Kurzfassung

In vielen Bereichen werden heutzutage große Mengen an Daten gesammelt. Das Verstehen solcher Daten erfordert meistens eine Vorverarbeitung und Visualisierung. Taxiunternehmen sammeln unter anderem Bewegungsdaten, die Analyse von Verkehrsbehinderungen oder allgemeinen Mustern erlauben. Um sowohl effizienter, als auch übersichtlicher die Daten zu analysieren und darzustellen, werden Bewegungsdaten oft durch Map Matching Algorithmen vereinfacht. Dabei wird der Datensatz auf einen Graphen des Straßennetzes abgebildet. Zusätzlich können Ungenauigkeiten und fehlerhafte Messungen behoben werden. Solche Algorithmen enthalten jedoch Parameter und müssen meistens auf die Daten angepasst werden. Diese Arbeit zeigt ein interaktives Map Matching Verfahren. Durch visuelle Unterstützung kann ein Analyst die Parameter für eine effiziente Anwendung einstellen. Das entwickelte Verfahren wurde implementiert und anschließend in mehreren Fallstudien ausgewertet.

Inhaltsverzeichnis

1 Einleitung **11**

2 Verwandte Arbeiten **13**
2.1 Datenexploration und Visualisierung 13
2.2 Geodaten-Analyse 13
2.3 Interaktive Verarbeitung 14
2.4 Map Matching 14

3 Grundlagen **17**
3.1 Visual Analytics 17
3.2 Bewegungsdaten 18
3.3 Vergleich von Map Matching Verfahren 19

4 Konzept **21**
4.1 St-Matching 21
4.2 Visualisierung 24
4.3 Interaktionen 27

5 Implementierung **31**
5.1 Datensatz 31
5.2 Programmierung 31

6 Fallstudien **35**
6.1 Iterative Parameteranpassung 35
6.2 Straßennetz Korrektur 38

7 Zusammenfassung und Ausblick **41**

Literaturverzeichnis **43**

Abbildungsverzeichnis

4.1 Arbeitsprozess 22
4.2 Überblick des entwickelten Systems 25
4.3 Overlays der Kartenansicht 26
4.4 Overlays der Kartenansicht 26
4.5 Automatische Erkennung von Fehlern 28
4.6 Vorgeschlagene Korrektur einer Straße 29

5.1 Aufbau des Systems 32

6.1 Fehler im Straßennetz 35
6.2 Fehlerrate und Ungenauigkeit während der Parameteroptimierung 37
6.3 Korrektur einer Brücke über Qiantang-Fluss. 38
6.4 Fehlerrate und Ungenauigkeit während der Straßennetz Korrektur 40
6.5 Ergebnisübersicht 40

Tabellenverzeichnis

6.1 Berechnete Iterationen für Parameteroptimierung. 37
6.2 Berechnete Iterationen für Straßennetzoptimierung. 39

1 Einleitung

Geographische Daten (Geodaten) werden häufig in großen Mengen durch GPS-Geräte in Fahrzeugen gesammelt. Die von Taxiunternehmen erfassten Daten werden oft analysiert, um in verschiedenen Bereichen Informationen zu sammeln. Sie können zum Beispiel genutzt werden, um Strecken, auf denen sich häufig Verkehrsstaus bilden, zu entdecken [WLY+13] oder um vorherzusagen wo und wann Taxifahrer schneller Fahrgäste finden können [LZS+11]. Ein großer Teil solcher Daten enthält jedoch häufig falsche Messungen, Ausreißer oder keine relevanten Informationen. Aus diesem Grund wird der Datensatz meistens bereinigt, bevor Wissen gewonnen werden kann.

Für eine Analyse der Daten spielen u.a. die Bereiche „Data Mining“ und „Information Visualization“ eine wichtige Rolle. Der erste Schritt ist jedoch fast immer eine Vorverarbeitung der Aufzeichnungen. Bevor sie nach Mustern durchsucht und anschaulich dargestellt werden können, ist es sinnvoll Daten ohne Informationsgehalt zu filtern und die restlichen in ein gewünschtes Format zu konvertieren. Solche Schritte können Performanz und Speicherplatz verbessern.

Ein möglicher Schritt für die Vorverarbeitung von Geodaten ist das **Map Matching**. Dabei werden die Messungen zu Punkten bzw. Pfaden auf einem Straßennetz zugewiesen. Wenn Straßen statt Koordinaten betrachtet werden, sind Zusammenhänge in den Daten besser zu erkennen. Bevorzugte Strecken und Eigenschaften des Verkehrsfluss können einfacher berechnet werden. Weil in der Regel weder Messungen, noch Straßeninformation exakt sind, kommt es meistens zu Fehlern in diesem Verfahren. Um das Ergebnis zu verbessern, wird das Map Matching für jeden Datensatz angepasst. Solche Algorithmen besitzen meistens Parameter, die in Abhängigkeit von der Qualität und Eigenschaften der Messungen eingestellt werden sollten. Dazu ist eine Visualisierung der Bewegungsdaten und der Eigenschaften hilfreich. Während andere Arbeiten versuchen die Map Matching Genauigkeit durch fortgeschrittene Berechnungen zu verbessern, wird in dieser Arbeit untersucht, wie ein Analyst, bei der Anwendung eines solchen Algorithmus, das möglichst beste Ergebnis erzielen kann. Es wurde ein System entwickelt, das dem Nutzer hilft Ausreißer in Daten zu bereinigen, Fehler im Straßennetz zu korrigieren und Parameter zu optimieren. Interaktionen, Eigenschaften und Ergebnisse des Verfahrens sollen sinnvoll visualisiert werden, so dass der Prozess verfolgt werden kann und Fehler erkennbar sind. Zur Verifizierung des entwickelten Systems wurde ein Datensatz, welcher von Taxis gesammelt wurde, und ein Modell des entsprechenden Straßennetzes verwendet.

Gliederung

Zu Beginn der Arbeit werden verwandte Arbeiten vorgestellt (Kapitel 2). Anschließend folgen die Grundlagen (Kapitel 3) dieser Arbeit und das Konzept (Kapitel 4) des entwickelten Verfahrens. Kapitel 5 beschreibt die Implementierung, welche anschließend in Kapitel 6 auf einen Datensatz angewendet wurde, um das System auszuwerten. Die Arbeit wird dann durch eine Zusammenfassung und den Ausblick möglicher Anknüpfungspunkte abgeschlossen.

2 Verwandte Arbeiten

Es wurden verwandte Arbeiten in verschiedenen Bereichen betrachtet. Zuerst werden Grundlagen der visuellen Analyse angeschaut. Ferner beschäftigt sich eine Vielzahl an Arbeiten mit der Analyse von Geodaten. Ein anderer, seit kurzem wachsender Bereich ist die interaktive Vorverarbeitung von Daten. Zuletzt werden verschiedene Map Matching Verfahren angeschaut.

2.1 Datenexploration und Visualisierung

Data-Mining hat viele Rollen in der realen Welt. Wozu es gebraucht und was die Grundlagen von Data-Mining sind, haben Fayyad et al. [FPS96] schon früh erläutert. Sie erklären den Prozess, wie Wissen aus Datenbanken gewonnen werden kann, und beschreiben beliebte Methoden für Data-Mining. Schneidermann [Shn01] zeigt später, dass Exploration der Daten wichtig ist und gibt Richtlinien für die Entwicklung von Werkzeugen zur Exploration vor. Zusätzlich zur Wichtigkeit der Visualisierung, betont er, dass der Nutzer im Vordergrund steht. Dieser soll die Werkzeuge verstehen und Kontrolle darüber haben, um finden zu können, wonach gesucht wird.

Die Kombination aus Data-Mining und Informationsvisualisierung vereinigt mit der menschlichen Wahrnehmung bildet die Grundlage für **Visual Analytics**. Keim et al. [KAF+08] beschreiben den Zusammenhang verschiedener Bereiche und erklären den Visual Analytics Prozess. Dieser Prozess besteht aus einer „Sensemaking"-Schleife, mit welcher durch Visualisierung, Wahrnehmung und Analyse Kenntnisse erlangt werden. Eine wichtige Grundlage in Visual Analytics bildet das Konzept des „Visual Information-Seeking Mantra" [Shn96], welches Regeln für das Entwickeln von Benutzeroberflächen stellt.

Auch für visuelle Exploration großer Mengen an Geodaten zeigen Arbeiten [AAW07], wie diese durch Aggregationstechniken anschaulich visualisiert werden können.

2.2 Geodaten-Analyse

Geodaten werden genutzt, um Muster und Eigenschaften im Verkehrsfluss zu zeigen. Statt mit Kameras, die teuer und inflexibel sind, zeigen Wang et al. [WLY+13] wie Daten betrachtet

werden können, um Verkehrsstaus zu erkennen. Ihre Arbeit verarbeitet die Daten in mehreren Schritten und nutzt geeignete Ansichten, um jeweils die Ergebnisse zu visualisieren. Dabei werden zuerst Bewegungsdaten und Straßennetz bereinigt. Anschließend wird ein Map Matching Algorithmus auf diese angewendet und aus den Ergebnissen kann die mittlere Geschwindigkeit für Straßen pro Zeitintervall berechnet werden. Eine ähnliche Pipeline, bestehend aus Map Matching und anschließenden Analyseschritten, verwenden auch Lu et al. [LLY+15], um die Auswahl verschiedener Strecken bei ähnlichem Start und Zielpunkt zu erklären. Statt mittlerer Geschwindigkeit, sind bei ihrer Fragestellung jedoch die möglichen Strecken vom Start zum Endpunkt relevant zur Analyse. Zuletzt folgt die Visualisierung und Exploration der Daten und Eigenschaften.

Bevor man in der Lage sein kann Erkenntnisse zu erlangen, müssen zuerst die Daten bereinigt werden. Li et al. [LWW15] stellen vor, wie bestimmte Qualitätsprobleme in GPS-Daten erkannt und entfernt werden können. Um die Probleme zu finden, verwenden sie aktives Lernen, welches Eingabe durch einen Analysten erfordert. Anomalien, die nicht als Qualitätsproblem definiert wurden, können auf diese Weise jedoch nicht erkannt werden.

2.3 Interaktive Verarbeitung

Um Zeitreihendaten zu verarbeiten, bietet es sich an Operationen zu definieren, so dass der Analyst die geeigneten Schritte der Verarbeitungspipeline wählen kann [BRG+12]. Mögliche Schritte sind Methoden zur Bereinigung, Normalisierung oder Bestimmung eines Ähnlichkeitsmaß. Die Wahl der Operationen und ihre Parameter kann auf eine relativ kleine Datenmenge angepasst und anschließend auf die restlichen Daten angewendet werden.

Methoden des maschinellen Lernens bieten einen alternativen Ansatz, um abweichende Daten zu bereinigen. Dies kann für GPS-Messungen angewendet werden, um abweichende Trajektorien zu identifizieren [LYC10]. Der Analyst hat dabei die Aufgabe das Modell mit Hilfe eines vorhandenen Datensets zu trainieren.

2.4 Map Matching

Navigationssysteme sind heutzutage in einem Großteil von Fahrzeugen und Mobiltelefonen. Schon innerhalb der ersten Navigationssysteme in Fahrzeugen wurde Map Matching eingesetzt, um die Position auf dem Straßennetz zu bestimmen [Col90]. Die anfangs simplen Matching Algorithmen, welche einfach die nächstgelegene Straße gesucht haben, wurden später für verschiede Anwendungen erweitert und verbessert. Navigationssysteme nutzen in der Regel inkrementelle Matching Verfahren, welche nur die aktuelle Messung oder die letzten n Messungen betrachten. Im Gegensatz dazu berücksichtigen globale Algorithmen eine

vollständige Folge von Messungen, um bessere Ergebnisse zu erreichen. Diese können jedoch nicht angewendet werden, wenn Messungen in Echtzeit gemessen werden.

Das Ziel vieler Arbeiten besteht darin, die Genauigkeit von Map Matching Algorithmen zu vergleichen oder verbessern. Wegen Hindernissen zwischen Fahrzeug und Satelliten oder aufgrund schlechter Verteilung der Satelliten, wird der Messfehler von GPS-Geräten erhöht [MKH06]. Um trotzdem richtige Ergebnisse zu erhalten, werden fortgeschrittene Algorithmen entwickelt, die zusätzliche Eigenschaften (z.B. Fahrtrichtung, Straßentopologie und Geschwindigkeit) berücksichtigen [WBK00]. Andere Arbeiten zeigen Optimierungen für geringe Abtastraten [LZZ+09] [MKYM12] oder für komplexe Straßennetze [OQN03].

3 Grundlagen

In diesem Kapitel werden die Grundlagen vorgestellt, die zum Verstehen dieser Arbeit relevant sind und für Entscheidung beim Vorgehen nötig waren. Dazu werden zuerst die Grundlagen von Visual Analytics vorgestellt. Danach werden die Eigenschaften von Bewegungsdaten beschrieben. Zuletzt folgt ein Vergleich verschiedener Map Matching Methoden.

3.1 Visual Analytics

Visual Analytics kombiniert Mensch und Computer. Durch eine enge Kopplung von analytisches Denken und interaktiver Visualisierung sollen komplexe Daten nachvollzogen werden.

3.1.1 Data-Mining

Data-Mining ist die Anwendung von Algorithmen, um Muster in Daten zu finden. Dabei gibt es einige beliebte Methoden.

Clustering ist ein möglicher Algorithmus, der ähnliche Elemente in Gruppen (Cluster) zuordnet. „Hierarchische Clusutering“ Verfahren beginnen mit einem großen Cluster, der schrittweise unterteilt wird, oder mit vielen kleinen Clustern, die vereinigt werden. Diese schließen also aus, dass Elemente zu mehreren Gruppen gehören, während in anderen Methoden sich die Gruppen überlappen können.

Klassifikation nutzt eine Funktion, um den Daten Klassen zuzuweisen. Der „naive Bayes-Klassifikator“ ist eine probabilistische Methode, die einfach anzuwenden ist. Sie berechnet zu einem Element die Wahrscheinlichkeit, mit der es zu einer Klasse gehört, und weist es zur Klasse mit der größten Wahrscheinlichkeit zu.

Diese und andere Algorithmen werden in vielen Bereichen angewendet [FPS96]. Für Bewegungsdaten kann Clustering genutzt werden, um ähnliche Fahrten zusammenzufassen. Dadurch können zusätzlich Daten anschaulicher dargestellt werden und es wird Speicherplatz gespart.

Visualisierung

Für die Entwicklung einer Nutzeroberfläche, stellt Ben Schneiderman [Shn96] sieben Aufgaben an den Programmierer. Zuerst soll ein Überblick über die Daten verschafft werden. Für interessante Daten soll ein Zoom und Filtern möglich sein, um diese genauer zu betrachten. Über Elemente sollen Details und Relationen angezeigt werden können. Zusätzlich ist eine Historie wichtig, um Aktionen rückgängig zu machen. Und zuletzt soll es möglich sein gefundene Daten und Ergebnisse zu speichern.

Für verschiedene Datentypen (1-dimensional, 2-dimensional, Zeitreihen, ...) gibt es jeweils bevorzugte Methoden, um diese darzustellen. Geographische Koordinaten können durch eine Kartenansicht anschaulich visualisiert werden. Die zeitliche Komponente von Bewegungsdaten kann durch Zeitfilter manipuliert werden [AAW07].

Interaktion

In Visual Analytics kommt nach der Visualisierung die Aufgabe des Nutzers. Dieser kann die dargestellten Informationen interpretieren und durch Exploration analysieren. Mittels Interaktion mit dem System kann er anschließend den Fokus auf die relevanten Daten setzen. Auf diese Weise kann die Visualisierung verbessert werden und der beschriebene Prozess wird wiederholt. Das Verständnis der Daten wird durch mehrere Iterationen verbessert, bis ein Ergebnis erreicht oder die Hypothese erfüllt ist.

3.2 Bewegungsdaten

GPS-Daten von Fahrzeugen werden aus verschiedenen Gründen gesammelt. Sie erlauben das Kontrollieren und Analysieren von Fahrzeugen. Diese Daten bestehen in der Regel aus Positionsinformation (Breitengrad, Längengrad) und Zeitstempel. Zusätzlich können weitere Eigenschaften wie Fahrtrichtung und Geschwindigkeit gemessen werden. Eine Folge solcher zusammenhängender Messungen bildet eine **Trajektorie**.

Die Daten werden meistens in einem Intervall von mehreren Sekunden bis wenigen Minuten gesammelt. Je kleiner das Intervall, desto besser können die Daten interpretiert werden. Kleine Intervalle führen jedoch zu großen Datenmengen, die größere Berechnungen als Folge haben. Die Performanz von sowohl Berechnungen der Eigenschaften, als auch der Visualisierung kann durch die großen Datenmenge limitiert werden. Daher sind Aggregationstechniken, z.B. Clustering [AAW07] oder Map Matching gut geeignet.

Ein anderes Problem, das auch bei GPS-Daten auftaucht, ist die Qualität der Messungen. Die GPS-Genauigkeit kann durch verschiedene Faktoren beeinflusst werden [MKH06] und ist daher oft nicht exakt. Bei Unternehmen mit tausenden von Fahrzeugen kommen auch Hardwarefehler

vor, die zu Fehlerhaften Messungen führen können. Zuletzt sind auch menschliche Fehler bei der Bedienung von GPS-Geräten nicht auszuschließen und können zu sinnlosen Daten führen.

3.3 Vergleich von Map Matching Verfahren

Definition Map Matching bezeichnet den Prozess, bei dem geographische Messungen auf ein Kartenmodell abgebildet werden. Typischerweise handelt es sich um GPS-Messungen, die auf einen Graphen mit Straßen als Kanten abgebildet werden. Das Map Matching vereinfacht den Umgang mit den Daten zur Analyse und ermöglicht eine einfachere und übersichtliche Visualisierung. Zusätzlich kann bei großen Datenmengen Speicherplatz erspart werden. Map Matching Verfahren können in verschiedene Kategorien unterteilt werden, die sich in der Regel durch das Anwendungsgebiet auszeichnen. Allgemein wird zwischen inkrementellen und globalen Algorithmen unterschieden.

Inkrementelle Verfahren werden oft angewendet, wenn die Daten nicht schon vorhanden sind, sondern in Echtzeit gemessen werden. Ein großes Anwendungsgebiet sind daher Navigationsgeräte in Fahrzeugen, welche stets die Position des Fahrzeugs ermitteln müssen. Für das Matching einer Messung wird daher entweder nur die letzte Messung betrachtet oder die letzten n Messungen. Eigenschaften zu berücksichtigen sind die Entfernung – zwischen Messung und Position auf der betrachteten Straße – und Verbindungen von der zuletzt bestimmten Position. Diese Algorithmen haben daher in der Regel relativ geringe Laufzeiten.

Bei globalen Verfahren ist die Vollständigkeit der Daten eine Voraussetzung, da diese immer eine Sequenz von Messungen auswerten. Für globale Algorithmen bestehen wiederum mehrere Ansätze. Die **geometrische** Methode sucht einen Pfad, der eine minimale Distanz (z.B. Fréchet-Distanz [WWFZ13]) zur gemessenen Trajektorie besitzt. Verbindungen zwischen Straßen werden bei **topologischen** Methoden betrachtet, aber nicht bei geometrischen. Andere **probabilistische** Verfahren können zusätzlich auch Geschwindigkeit und Richtung berücksichtigen [PH08]. Eine Kombination dieser Methoden kann wünschenswert sein, da sie Stärken und Schwächen besitzen [QON07].

4 Konzept

Das **Map Matching** geographischer Daten soll durch einen **Analysten** optimiert werden. Währenddessen sollen Ergebnisse und Eigenschaften der Daten visualisiert werden, um den Nutzer in diesem Prozess zu unterstützen. Dieses „Anpassen“ des Map Matching Verfahrens auf die Daten hat zwei Ziele. Zuerst soll die Genauigkeit des Algorithmus maximiert werden. Fahrten sollten nicht zu falschen Pfaden zugewiesen werden. Solche Fehler sind meistens nicht erkennbar, aber es kann unterschieden werden für welche Trajektorien das Map Matching gut verläuft und wo das Ergebnis fehlerhaft sein kann. Dieses Maß wird in dieser Arbeit als **Unsicherheit** bezeichnet. Die Unsicherheit einer Fahrt wird während dem Map Matching (Kapitel 4.1) berechnet. Das zweite Ziel besteht darin, die Fahrten zu reduzieren, für die der Algorithmus kein Ergebnis finden kann. Für den Anteil solcher Fahrten wird der Begriff **Fehlerrate** verwendet. Fehlende Straßenkanten im Graphen oder falsche Messungen führen zum Fehlschlagen des Algorithmus. Um beide Probleme gleichzeitig zu optimieren, werden Änderungen iterativ vorgenommen. Vor dem Map Matching kann optional ein Vorbereitungsschritt ausgeführt werden. Daten beinhalten oft Probleme (siehe Kapitel 3.2), die sowohl Fehlerrate und Unsicherheit erhöhen, als auch die Ergebnisse verfälschen würden. Deshalb werden solche Daten im Vorbereitungsschritt herausgenommen.

Zu Beginn des Arbeitsprozesses (Abbildung 4.1) wählt der Analyst aus, welche Geodaten für das Anpassen geladen werden sollen. Es kann sich dabei um ein relativ kleines Datensample handeln, welches stellvertretend für den gesamten Datensatz optimiert wird. Diese Daten und das Straßennetz werden zur Übersicht in einer Kartenansicht dargestellt. Anschließend kann der Analyst entweder die Parameter für Vorbereitungsschritte und Map Matching anpassen oder diese mit Standardparametern durchführen. Das Map Matching berechnet ein Ergebnis, Fehlerrate und Unsicherheit, welche ebenfalls visualisiert werden. Dies ermöglicht es dem Nutzer zu erkennen wo Fehlerstellen sind und diese können durch interaktiver Korrektur des Straßennetzes oder der Parameter verbessert werden. Durch mehrmaliges Wiederholen dieser Schritte sollen alle sichtbaren Fehlerquellen behoben werden. Wenn das Verfahren also optimiert ist, können die Parameter und das Straßennetz gespeichert werden oder das Map Matching kann auf den gesamten Datensatz angewendet werden.

4.1 St-Matching

Für das Map Matching wird der ST-Matching [LZZ+09] Algorithmus verwendet, welcher hohe Genauigkeit für geringe Abtastraten haben soll. Es handelt sich um ein probabilistisches

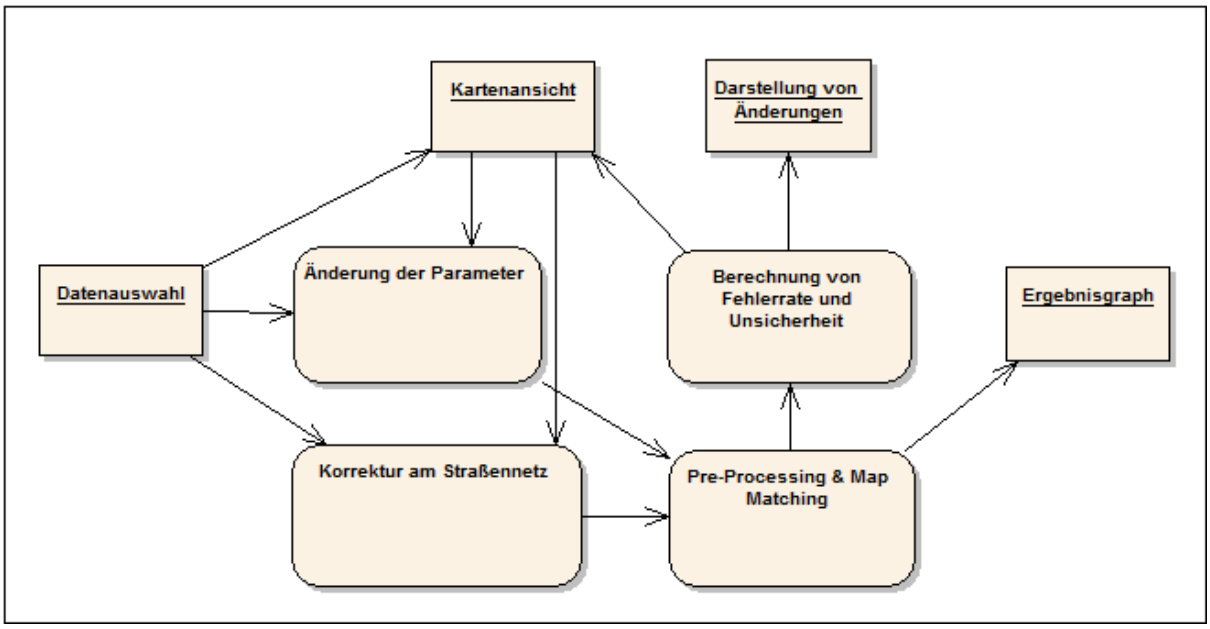

Abbildung 4.1: Arbeitsprozess

globales Verfahren, das sowohl Topologie des Straßennetzes, als auch Geschwindigkeit des Fahrzeugs berücksichtigt. Als globaler Algorithmus besitzt er eine höhere Komplexität als inkrementelle, aber weil Daten nicht zur Laufzeit gemessen werden, erreicht er eine bessere Genauigkeit. Durch Einstellen der Parameter können die Laufzeit und die Genauigkeit beeinflusst werden, womit sich dieser Algorithmus gut für interaktive Verfahren eignet. Falls der Prozess zu lange dauert, kann die Laufzeit auf Kosten der Genauigkeit reduziert werden. Zusätzlich wurde ST-Matching für Messungen mit geringer Samplingrate entwickelt, die auch in den untersuchten Daten (siehe Kapitel 5.1) gegeben ist.

Es wird eine räumliche und eine zeitliche Komponente betrachtet. Der Algorithmus wird von Lou et al. [LZZ+09] in drei Schritten beschrieben:

1. Candidate Preparation
2. Spatio-Temporal Analysis
3. Result Matching

Candidate Preparation

Der Algorithmus wird jeweils auf eine Trajektorie angewendet. Eine Trajektorie besteht aus n Messpunkten p_i. Im ersten Schritt werden zu den n Messungen in der Trajektorie jeweils k Kandidaten gesucht. Ein **Kandidat** ist eine Projektion auf einer Straßenkante mit der minimalen Distanz zu den gemessenen Koordinaten. Um diese zu finden, wird ein Gitter zur Indizierung der Straßenkanten erstellt. Dazu wird jede Kante entsprechend ihrer Position

in Gitterzellen, durch die sie verläuft, hinzufügt. Falls der Gitterabstand so groß ist wie die maximale Distanz, in der Kandidaten gesucht werden, müssen zu jeder Messungen nur ihre entsprechende Zelle und dessen Nachbarzellen nach Straßenkanten durchsucht werden. Das Ergebnis dieses Verfahrens ist eine Menge von bis zu k Kandidaten für jede Messung. Für Kandidaten werden in folgenden Schritten die Straßenkante, die Position auf der Kante und die Distanz zur Messung als Eigenschaften gebraucht.

Spatio-Temporal Analysis

Der zweite Schritt ist die Analyse der räumlichen und zeitlichen Eigenschaften. Ein Bestandteil der räumlichen Komponente ist die Wahrscheinlichkeit des Kandidaten $N(c_i^j)$. Diese ist nur von der Distanz x_i^j zwischen Position der GPS-Messung p_i und dem j-ten Kandidaten c_i^j von p_i abhängig. Sie wird durch eine Normalverteilung modelliert und repräsentiert die Wahrscheinlichkeit, dass der Kandidat ein „Match“ ist. Die Berechnung sieht wie gefolgt aus:

$$N(c_i^j) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{(x_i^j - \mu)^2}{2\sigma^2}} \tag{4.1}$$

μ ist der Erwartungswert und σ stellt die Standardabweichung dar. Für diese Parameter werden, die von Lou et al. [LZZ+09] empfohlenen Werte, $\mu = 0m$ und $\sigma = 20m$ verwendet.

Die zweite räumliche Eigenschaft repräsentiert die Topologie und hängt von den Verbindungen zwischen zwei Kandidaten ab. Berechnet wird diese aus der geographischen Distanz $d_{i-1 \to i}$ zwischen Messpunkten p_{i-1} und p_i dividiert mit der Länge $w_{(i-1,s)\to(i,t)}$ des kürzesten Pfades zwischen c_{i-1}^s und c_i^t. Die Längen der ersten und der letzten Kante in diesem Pfad sind entsprechend der Position des Kandidaten gewichtet, da dieser auf der Kante liegen.

$$V(c_{i-1}^s, c_i^t) = \frac{d_{i-1 \to i}}{w_{(i-1,s)\to(i,t)}} \tag{4.2}$$

$w_{(i-1,s)\to(i,t)}$ kann mit Hilfe des Dijkstra-Algorithmus aus dem gegebenen Straßennetz für alle $1 \leq s \leq k$ und alle $1 \leq t \leq k$ berechnet werden. Das Produkt dieser beiden Bestandteile bildet die räumliche Analyse

$$F_S(c_{i-1}^s \to c_i^t) = N(c_i^j) \cdot V(c_{i-1}^s, c_i^t) \tag{4.3}$$

mit $2 \leq i \leq n$.

Als letztes wird die zeitliche Analyse berechnet, welche die Geschwindigkeitsbeschränkungen berücksichtigt. Die durchschnittliche Geschwindigkeit $\overline{v}$ zwischen zwei Kandidaten beträgt

$$\overline{v}_{(i-1,s)\to(i,t)} = \frac{w_{(i-1,s)\to(i,t)}}{\Delta t_{i-1 \to i}} \tag{4.4}$$

und stellt einen Vektor mit m gleichen Elementen dar. Der Vektor $(v_1, v_2, ..., v_m)^T$ enthält die Geschwindigkeitsbeschränkungen auf dem Pfad $(e_1, e_2, ..., e_m)$ zwischen c_{i-1}^s und c_i^t. Die zeitliche Komponente sei dann die Kosinus-Ähnlichkeit dieser beiden Vektoren

$$F_T(c_{i-1}^s \to c_i^t) = \frac{\sum_{u=1}^{m}(v_u \cdot \overline{v})}{\sqrt{\sum_{u=1}^{m}(v_u)^2}\sqrt{\sum_{u=1}^{m}(\overline{v}_{(i-1,s)\to(i,t)})^2}} \tag{4.5}$$

und es gelte wieder $2 \leq i \leq n$.

Result Matching

Der letzte Schritt besteht darin, einen Pfad aus den jeweils k Kandidaten der n Messungen zu finden, der am wahrscheinlichsten ist. Dazu wird aus den Kandidaten ein Graph erstellt. Mit Hilfe der zeitlichen und räumlichen Analyse wird die sogenannte ST-Funktion definiert:

$$F(c_{i-1}^s \to c_i^t) = F_S(c_{i-1}^s \to c_i^t) \cdot F_T(c_{i-1}^s \to c_i^t), 2 \leq i \leq n \tag{4.6}$$

Der gesuchte Pfad besitzt eine maximale Summe der ST-Werte mit $1 \leq s \leq k$ und $1 \leq t \leq k$.

$$F(P) = \sum_{i=2}^{n} F(c_{i-1}^s \to c_i^t) \tag{4.7}$$

Der Algorithmus, der diesen Pfad $P = (c_1^s, \ldots, c_n^t)$ berechnet, wird in Kapitel 5.2 erklärt.

Das arithmetische Mittel über alle Werte der ST-Funktionen jedes Ergebnisses wird in dieser Arbeit als Sicherheit S für m Pfade P_i der Länge $|P_i|$ definiert.

$$S = \frac{\sum_{i=1}^{m} \frac{F(P_i)}{|P_i|-1}}{m} \tag{4.8}$$

Weil das in der Regel sehr kleine Werte sind, wird hier für besseres Verständnis folgende Unsicherheit U berechnet:

$$U = 1000 \cdot (1 - S) \tag{4.9}$$

Diese wird genutzt, um die Qualität der Ergebnisse zu vergleichen.

4.2 Visualisierung

Die verschiedenen Formen von Informationen sollen jeweils durch eine passende Darstellung visualisiert werden. Es gibt drei Arten von Daten, mit welchen gearbeitet wird.

1. Rohdaten (GPS-Koordinaten mit Zeitstempel, Straßennetz)
2. Verarbeitete Daten (Ergebnis des Map Matching)

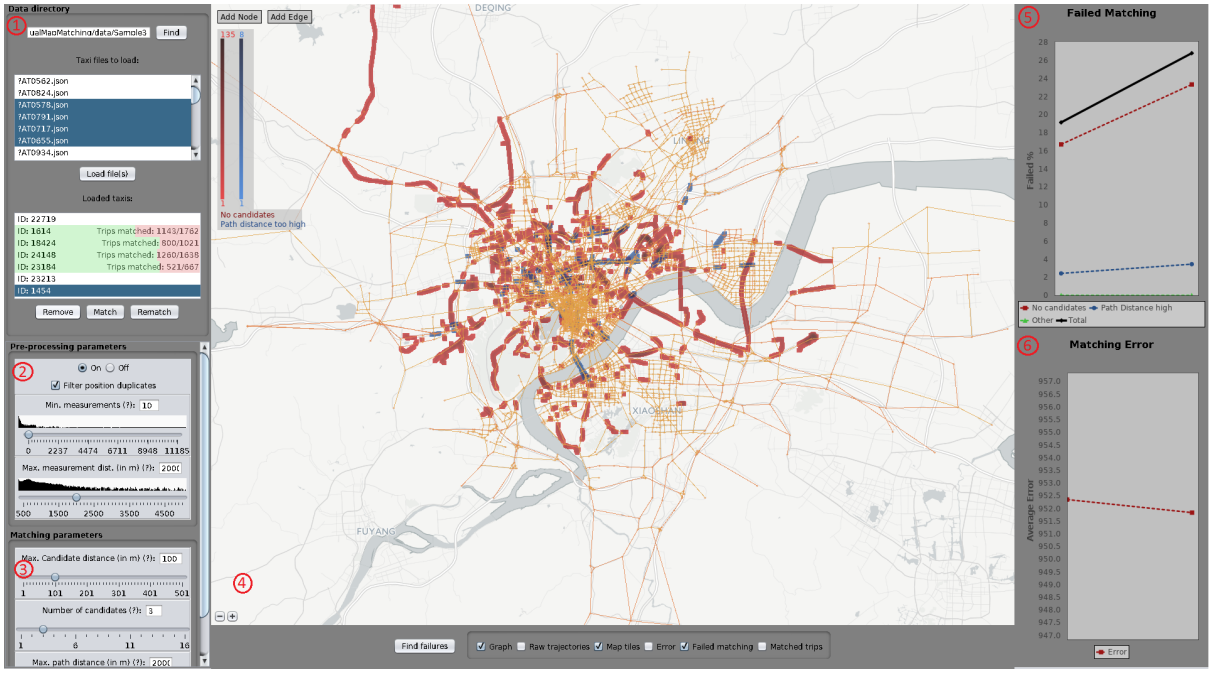

Abbildung 4.2: Überblick des entwickelten Systems. 1) Auswahl von Dateien, die geladen (oben) und „gematcht“ (unten) werden; 2) Parametereinstellung des Pre-Processing; 3) Parametereinstellung des Map Matching; 4) Kartenansicht zur Darstellung der Bewegungsdaten; 5) Liniendiagramm zur Darstellung von Fehlern im Map Matching Algorithmus; 6) Liniendiagramm zur Darstellung der Unsicherheit im Map Matching Algorithmus.

3. Eigenschaften (Fehlerrate etc. vom Map Matching Algorithmus)

Eine intuitive Weise diese darzustellen ist mit Hilfe einer Kartenansicht (4 in Abbildung 4.2). Um einen Überblick über die Rohdaten zu schaffen, wird jede Fahrt des Fahrzeugs durch Verbinden der Koordinaten dargestellt (Abbildung 4.4 links).

Nach dem Map Matching Algorithmus erhält man zum einen das Ergebnis und zusätzlich dessen Unsicherheit U. Die Ergebnisse bestehen aus Listen von Kandidaten, welche jeweils auf Kanten im Straßengraphen abgebildet und gezeichnet werden (Abbildung 4.3 rechts). Bei einer höheren Anzahl von Fahrten durch eine Straße, wird diese breiter gezeichnet.

Um diese Ergebnisse zu verbessern, sollten noch die Fahrten, für die der Matching Algorithmus kein Ergebnis findet oder das Ergebnis zu unsicher ist, ebenfalls betrachtet werden. Beide Probleme können auf der Karte gezeichnet werden. Durch Splatting (siehe Kapitel 5.2) der Kanten fehlerhafter Trajektorien ist es möglich Fahrten darzustellen, so dass Stellen mit vielen Fehlern erkennbar sind (Abbildung 4.4 Mitte). Auf dieser Weise kann auch die Unsicherheit (Abbildung 4.4 rechts) von erfolgreich berechneten Trajektorien dargestellt werden. Diese Darstellung kann parallel zum Map Matching erstellt werden, sodass ein Analyst schon während der Berechnung Problemstellen finden kann und eventuell vorzeitig eine neue Iteration

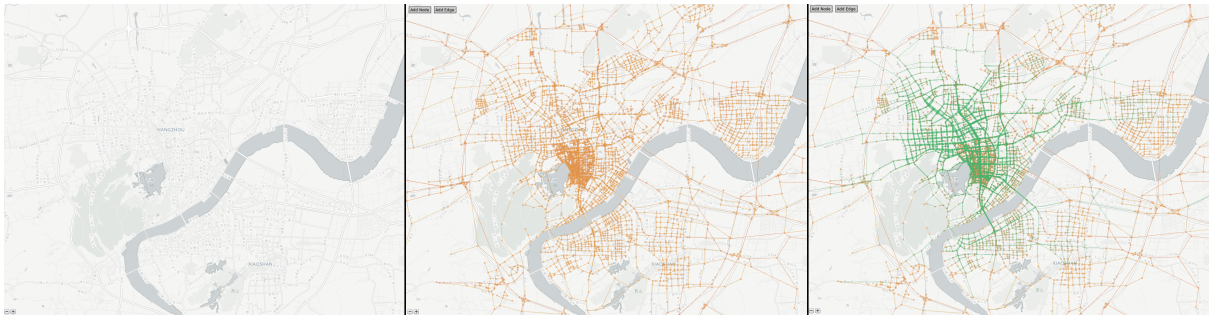

Abbildung 4.3: Links: Kartenansicht. **Mitte:** Straßengraph. **Rechts:** Straßengraph (gelb) mit Ergebnis (grün) des Map Matching.

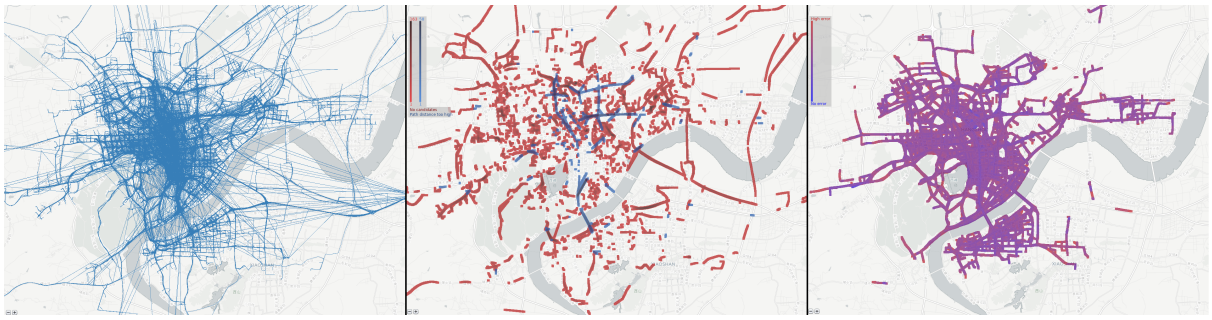

Abbildung 4.4: Links: Rohdaten. **Mitte:** Fehleranzahl des Map Matching. Fahrten ohne Kandidat zu einer Messung in rot. Fahrten ohne Verbindung zwischen zwei Messungen in blau. **Rechts:** Mittlere Unsicherheit des Map Matching. Geringe Unsicherheit in blau. Hohe Unsicherheit in rot.

beginnt. Zusätzlich besitzt jedes Fahrzeug einen Fortschrittsbalken, welcher durch Verwendung verschiedener Farben die Anteile an fehlerhafter und erfolgreicher Berechnungen darstellt (1 in Abbildung 4.2).

Der Analyst soll bei Änderungen sehen, wie sie sich auf das Verfahren auswirken. Um einen Überblick darüber zu schaffen, werden Fehlerrate und Unsicherheit betrachtet. Beide Größen sind von einander abhängig. Wenn Parameter (z.B. maximale Distanz zu Kandidaten) so eingestellt sind, dass der Algorithmus nur für Fahrten mit geringer Unsicherheit erfolgreich ist, so steigt die Fehlerrate. Eine zu starke Lockerung der Parameter kann jedoch dazu führen, dass den Trajektorien falsche Pfade zugewiesen werden und die Unsicherheit steigt. Die Werte von Fehlerrate und Unsicherheit werden deshalb in jeweils einem Liniendiagramm (5 & 6 in Abbildung 4.2) übereinander dargestellt.

Dieser Wert wird auf der y-Achse des Diagramms dargestellt, während die x-Achse die Iterationen repräsentiert. Die Fehlerrate wird in drei Kategorien unterteilt:

1. Keine Kandidaten gefunden
2. Keine Straßenverbindung zwischen Kandidaten gefunden

3. Andere

Kandidaten und Pfade zwischen Kandidaten existieren immer, jedoch sind diese jeweils durch eine maximale Distanz beschränkt, um falsche Ergebnisse zu vermeiden. Gründe dafür sind zum Beispiel fehlende Straßenkanten oder Messpunkte außerhalb des Straßengraphen. „Andere“ Fehler sind zum Beispiel Trajektorien mit nur einer oder keine Messung. Die Anteile (Prozent von allen Fahrten) dieser drei Fehler und ihre Summe werden auf der y-Achse im Diagramm der Fehlerrate dargestellt. Idealerweise sollen alle Werte in beiden Diagrammen bei jeder Iteration kleiner werden.

Um das Intervall geeigneter und gültige Werte für die Parameter darzustellen, eignet sich die Anwendung von Slidern (2 & 3 in Abbildung 4.2). Ein exaktes Einstellen für Parameter höherer Größenordnung kann durch Anwenden eines einfachen Textfeldes erreicht werden. Für Vorbereitungsschritte, die Fahrten vor dem Map Matching Algorithmus filtern, kann ein Histogramm über dem jeweiligen Slider hilfreich sein. Aufgrund der schiefen Verteilung der Fahrten wird eine logarithmische y-Achse für die Histogramme verwendet. Der Nutzer kann dadurch die Qualität der Daten überblicken und einschätzen welche Werte sinnvoll für den Parameter sind.

4.3 Interaktionen

Eine Aufgabe des Nutzers besteht darin, Probleme zu beheben, die der Algorithmus nicht erkennt. Idealerweise kann ein Experte Domänenwissen einbringen, um diese Schwierigkeiten zu beseitigen.

Das Straßennetz kann ein Teil dieser Probleme sein. Der Straßengraph besteht aus Knoten an Kreuzungen und die Straßen werden als Kanten modelliert. Wegen menschlicher Fehler kann der Graph Ungenauigkeiten enthalten. Es können zum Beispiel Knoten an Kreuzungen fehlen oder ganze Straßen werden ausgelassen. Zusätzlich können aus verschiedenen Gründen Abweichungen entstehen. Häufig sind Straßenkanten zu lang, weil sie nur an den Kreuzungen Knoten besitzen. Das verwendete Straßennetz kann veraltet sein oder komplizierte Verbindungen wurden vereinfacht. Bei solchen Problem kann der Map Matching Algorithmus scheitern, welches der Analyst erkennt und anschließend die nötigen Straßenkanten hinzufügen oder verschieben kann. Zur Hilfe ist es möglich Fehler automatisch zu erkennen. Durch Drücken eines Buttons („Find failures“ in Abbildung 4.2) sucht das System die Stellen mit den meisten Fehlern oder mit der größten Unsicherheit und zeigt diese dem Nutzer im Mittelpunkt der Kartenansicht (siehe Abbildung 4.5). Dazu werden die Berechnungen, die zum Darstellen der Unsicherheit und der Fehleranzahl nötig sind, verwendet (siehe Kapitel 5.2).

Um den richtigen Verlauf von Straßen zu finden, kann eine Kartenansicht (z.B. OpenStreetMap [Ope]) im Hintergrund verwendet werden. Alternativ ist es möglich die Originaltrajektorien zu betrachten, welche aber nicht unbedingt dem Straßenverlauf folgen. Bei Kurven, die im Straßennetz nicht richtig dargestellt sind, kann der Analyst nach dem Map Matching eine

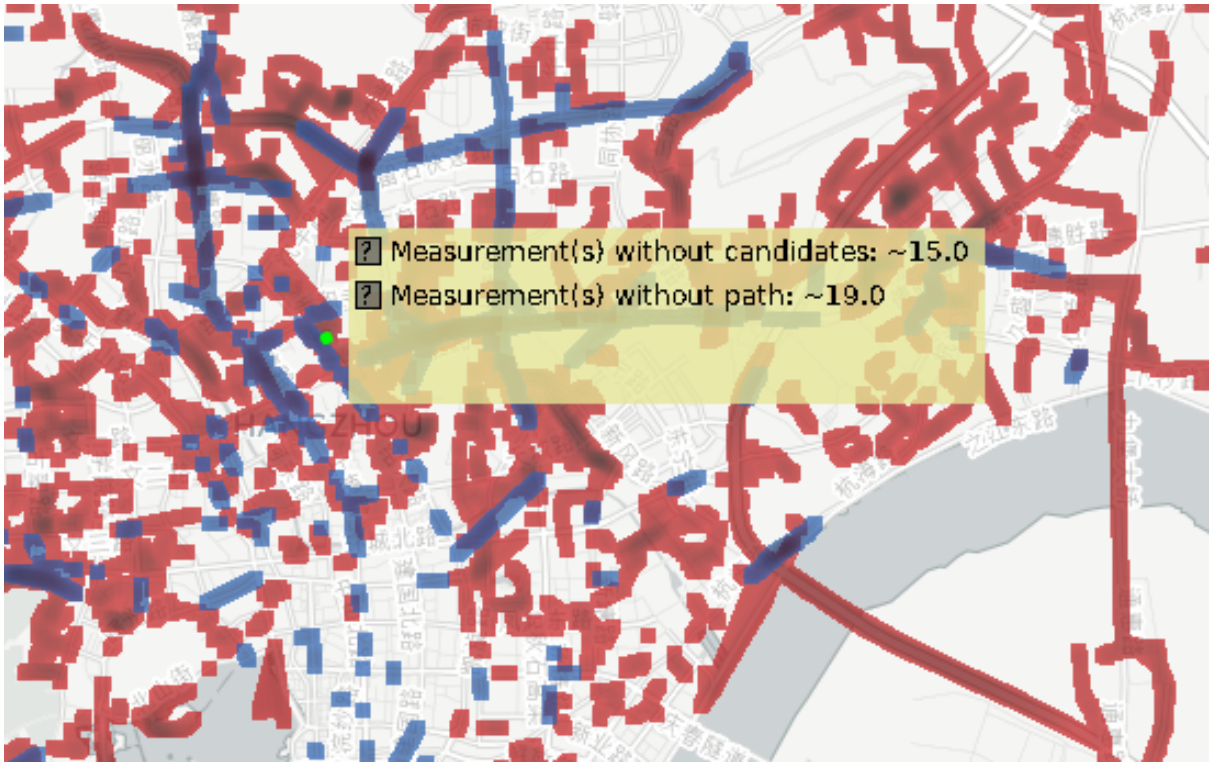

Abbildung 4.5: Automatische Erkennung von Fehlern. Hervorgehobene Fehlerstelle (grün), an welcher zu vielen Messungen kein Kandidat gefunden wurde (rot) und/oder keine Verbindung zwischen den Messungen (blau) gefunden wurde.

Fehlerstelle auswählen und das System schlägt eine Verbesserung durch Einfügen eines Knotens vor (siehe Abbildung 4.6). Um diese Verbesserung zu finden wird zuerst die nächste Straßenkante in der Nähe der Fehlerstelle gesucht. Anschließend werden alle Fahrten, welche durch die Fehlerstelle verlaufen, angeschaut. Von jeder dieser Trajektorien wird der Messpunkt gesucht, der eine maximale Distanz zur gefunden Straße hat, aber auch keine andere Straße näher an dem Punkt ist. Die mittlere Position dieser Messpunkte bildet den neuen Knoten, welcher mit den zwei Knoten der ursprünglichen Straßenkante verbunden wird. Alternativ kann das System die Kandidatendistanz an dieser Stelle zum nötigen Wert erhöhen, damit allen Messungen in der Nähe ein Kandidat zugewiesen wird. Dazu wird wieder zunächst die nächste Straßenkante in der Nähe gesucht. Anschließend wird der maximale Abstand zwischen der Straßenkante und den Messpunkten fehlgeschlagener Fahrten, die sich an der Fehlerstelle befinden, berechnet.

Durch Erhöhen der maximalen Distanz, in der Kandidaten gesucht werden (3 in Abbildung 4.2), können Fahrten trotzdem zur entsprechenden Straße zugewiesen werden. Gleichzeitig ist es jedoch möglich, dass andere Fahrten zu falschen Straßen zugewiesen werden, falls der Parameter zu groß ist. Um dies zu erkennen, kann die Unsicherheit beim Map Matching betrachtet werden (6 in Abbildung 4.2).

Abbildung 4.6: Vorgeschlagene Korrektur einer Straße. Links: Ursprüngliche Straßenkante (gelb) mit Fehlerdarstellung (rot). Rechts: Vorgeschlagener Verlauf der Straße (hellrot) ersetzt ursprüngliche Kante (blau). Grüner Punkt beschreibt die ausgewählte Fehlerstelle.

Der Ursprung eines anderen Problems liegt in den Daten. Durch defekte Hardware oder falsche Nutzung entstehen verschiedene Arten von Fehler. Zum Beispiel können fehlende Daten, falsche Koordinaten oder falsche Zeitstempel enthalten sein. Weil solche Daten nicht korrigiert werden können, sollten jeweils die Messungen oder die ganzen Fahrten gelöscht werden. Dazu wird der Vorbereitungsschritt vor dem Map Matching Algorithmus verwendet. Der Nutzer bestimmt dabei die Parameter für minimale Anzahl an Messungen pro Fahrt und maximale geographische Distanz zwischen Messungen. Messungen mit sehr geringer Änderung der zuletzt gemessenen Koordinaten können aus ihrer Trajektorie entfernt werden. Solche Messungen entstehen wenn sich Fahrzeuge nicht bewegen. Sie verursachen große Datenmengen und enthalten keinen Informationsgehalt, da das Stehen nach dem Entfernen anhand des Zeitstempels erkannt werden kann

Um die Genauigkeit des Algorithmus zu erhöhen, ist es möglich die Anzahl der Kandidaten k, die pro Messung betrachtet werden, zu erhöhen. Dies geht jedoch auf Kosten der Laufzeit, die quadratisch erhöht wird. Die Laufzeit des Algorithmus – angewendet auf eine Trajektorie der Länge n auf einem Graphen mit m Kanten – beträgt $\mathcal{O}(nk^2 m \log m + nk^2)$. Zusätzlich kann die maximale Geschwindigkeit (bzw. maximale Distanz) zwischen Kandidaten bestimmt werden, um unrealistische Möglichkeiten während der Berechnung zu verwerfen.

5 Implementierung

Für diese Arbeit wurde ein Datensatz verwendet, der von einem Taxiunternehmen aufgezeichnet und bereitgestellt wurde. Dieses Kapitel beschreibt zuerst den Aufbau dieser Datenmenge und erklärt anschließend die Implementierung des entwickelten Systems.

5.1 Datensatz

Die in dieser Arbeit verwendeten Daten wurden von über 8.400 Taxis in der Stadt Hangzhou, China gesammelt. Sie wurden zwischen dem 01.01.2013 und dem 31.01.2013 in einem Intervall von einer Minute gemessen und bilden ca. 24 Millionen Fahrten. Jedes Taxi wird durch eine Datei im JSON-Format repräsentiert. Diese Dateien bestehen aus Listen von Fahrten, die ebenfalls aus Listen von Messungen bestehen. Eine Messung beinhaltet Messungs-Id, Fahrzeug-Id, Längengrad, Breitengrad, Geschwindigkeit, Winkel, Fahrgast (in Taxi), Serverzeit und Taxizeit.

Das bereitgestellte Straßennetz besteht aus einem ungerichteten Graphen $G(V, E)$ mit Knoten V und Kanten E. Die Knoten bestehen aus den Koordinaten der entsprechenden Kreuzung und einer Knotennummer. Jede Kante besitzt zwei Knotennummern und den Straßentypen. Insgesamt besteht der Graph aus 5039 Knoten und 7036 Kanten.

5.2 Programmierung

Der praktische Teil dieser Arbeit wurde mit der Programmiersprache Java durchgeführt. Die Programmierung des Systems richtet sich nach dem **Model View Controller** Entwurfsmuster. Das Modell beinhaltet die geladenen Daten (Trajektorien, Straßengraph) und die Präsentationsschicht stellt diese Daten, sowie Ergebnisse, Fehler und Parameter des Systems dem Benutzer vor. Aktionen des Analysten (Kapitel 4.3) werden durch Kontrollelemente bearbeitet, welche direkt die Daten ändern (z.B. Parameter) oder Berechnungen ausführen (z.B. Map Matching).

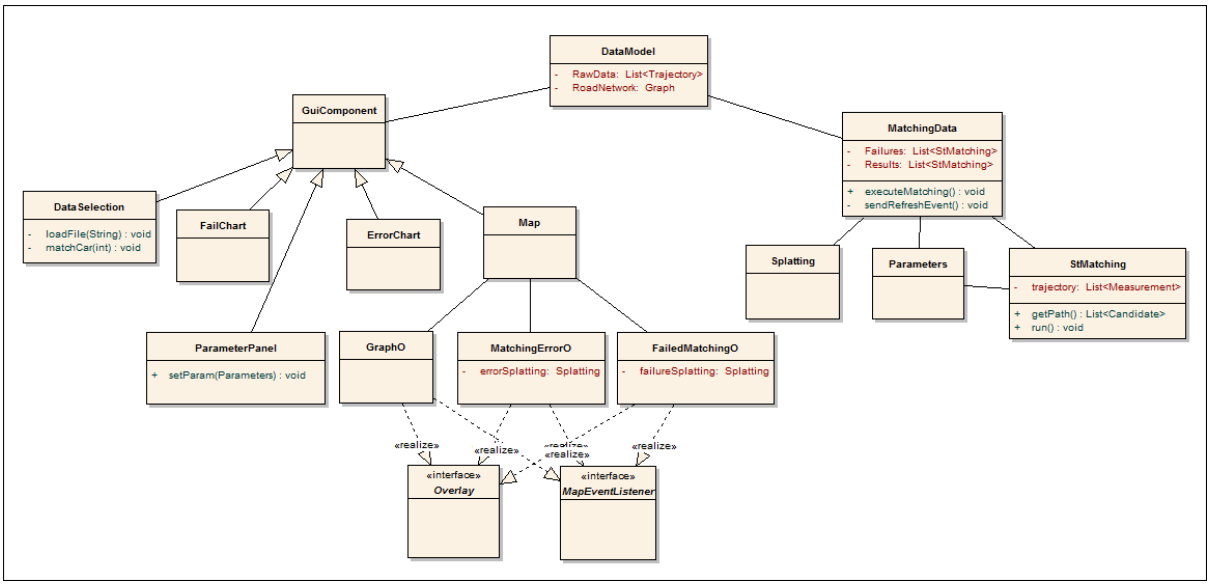

Abbildung 5.1: Aufbau des Systems

Modell

Die oberste Schickt der Klassenhierarchie (siehe Abbildung 5.1) besteht aus der „DataModel“-Klasse. Nach dem Singleton-Modell existiert nur eine Instanz dieser Klasse. Sie enthält die Eingabedaten (Straßengraph, Trajektorien) und verweist auf die unteren Schichten, welche für Oberfläche und Berechnungen zuständig sind. Die „MatchingData“-Klasse startet Threads zur Berechnung vom Map Matching und verwaltet diese, sowie dessen Ergebnisse. Sobald die Berechnungen fertig sind, wird ein Event an die Karte und ihre Komponenten (Overlays) geschickt, sodass diese die neuen Daten laden.

Zur Modellierung des Straßennetzes wurde Graph-Struktur des Prefuse Toolkit [Pre] verwendet.

GUI

Für die Kartenansicht wurde die JXMapViewer-Bibliothek [chr] verwendet. Diese erlaubt eine einfache Implementierung von „Overlays“, um die Taxifahrten und Fehler auf der Karte darzustellen. JXMapKit führt außerdem Umrechnungen zwischen geographischen Koordinaten und Pixelkoordinaten durch. Zoom- und Panning-Funktionen sind ebenfalls implementiert. Die Overlays werden im Hintergrund in jeweils eine BufferedImage Struktur gezeichnet. Diese Bilder werden neu gezeichnet, sobald die Kartenansicht geändert wird oder ein Event durch Map Matching Komponenten ankommt.

Die Liniendiagramme, um Fehler und Sicherheit anzuzeigen, wurden mittels JFreeChart [JFr] entwickelt. Mit jeder Iteration des Map Matching werden ihre Werte erneuert, sobald die Berechnung aller Fahrzeuge abgeschlossen ist.

Edge Splatting

Die Anzahl an Fehlern und die mittlere Unsicherheit werden durch „Edge Splatting“ berechnet. Dazu wird ein Array mit einem Eintrag pro Pixel auf der Kartenansicht initialisiert. Die Einträge enthalten eine Summe und die Anzahl der Summanden.

Für jede Kante wird über die Pixelpositionen iteriert, über die diese Kante verläuft. In der Umgebung von jedem Pixel wird im Array ein Wert zu dessen Summe addiert. Als Umgebung wird ein Quadrat mit Seitenlänge 7 Pixel betrachtet, wobei die Kante durch dessen Mittelpunkt verläuft. Der addierte Wert nimmt mit dem Abstand zum Mittelpunkt linear ab.

Beim Berechnen der Matrix für das Fehler-Overlay wird über die Kanten der im ST-Matching fehlgeschlagenen Trajektorien iteriert. Der addierte Werte beträgt immer 1, da das Splatting darstellen soll wie viele Fehler an Stellen entstehen. Dies wird für jeden Fehlertyp (siehe Kapitel 4.2) durchgeführt, um diese zu unterscheiden.

Um die Unsicherheit darzustellen, wird über die Pfade von Kandidaten (Ergebnis vom ST-Matching) iteriert. Die betrachteten Kanten sind jedoch die Verbindungen der Messpunkte, zu welchen die Kandidate gehören. Statt den mittleren Wert der Sicherheit aus Formel 4.8, kann hier der Wert der ST-Funktion (Formel 4.6) genutzt werden, um die Unsicherheit für eine Kante zweier Kandidaten zu berechnen. Diese Unsicherheit wird also entlang zu den Pixel der Kante addiert und anschließend wird sie durch die Anzahl der Summanden dividiert, um die mittlere Unsicherheit an den jeweiligen Stellen zu erhalten.

Wenn die jeweilige Matrix berechnet ist, werden die Werte normiert und eine Farbe wird für jeden Wert interpoliert und im Pixel des entsprechenden Bildes gezeichnet.

ST-Matching

Der Map Matching Algorithmus wurde entsprechend dem in Kapitel 4.1 besprochenem Verfahren implementiert. Die für Formel 4.5 benötigte Geschwindigkeitsbegrenzung ist nicht im Straßengraphen enthalten, daher wurde eine typische Geschwindigkeit für den jeweiligen Straßentyp angenommen. Die Länge des kürzesten Pfades zwischen zwei Kandidaten $w_{(i-1,s)\rightarrow(i,t)}$ (siehe Kapitel 4.1) wurde mit einem Dijkstra-Algorithmus berechnet. Durch Anwendung eines schnelleren Algorithmus, wäre es jedoch möglich die Laufzeit des ST-Matching zu verbessern. Weitere Optimierungen des Algorithmus beschreiben Lou et al. [LZZ+09] in ihrer Arbeit.

Um den besten Pfad im Kandidatengraph zu finden (siehe Kapitel 4.1) wird Algorithmus 5.1 verwendet. Eingabedaten sind k_i Kandidaten zu jedem Messpunkt p_i mit $1 \leq i \leq n$

Algorithmus 5.1 Result Matching

procedure FINDBESTPATH(c_i^j)
  $Highest[i][j]$ //Summe der ST-Werte für besten Pfad bis c_i^j
  for all $c_1^p, 1 \leq p \leq k_1$ **do**
    $Highest[1][p] := N(c_1^p)$
  end for
  for $i = 2 \ldots n$ **do**
    for all $c_i^t, 1 \leq t \leq k_i$ **do**
      $Max := -\infty$
      for all $c_{i-1}^s, 1 \leq s \leq k_{i-1}$ **do**
        $StValue := StFunction(c_{i-1}^s, c_i^t)$
        $Sum := Highest[i-1][s] + StValue$
        if $Sum > Max$ **then**
          $Max := Sum$
          $c_i^t.setStValue(stValue)$ //Zum Berechnen der Unsicherheit
          $c_i^t.setParent(c_{i-1}^s)$ //Vorgänger mit höchster Summe
        end if
        $Highest[i][t] := Max$
      end for
    end for
  end for
  $resultList$ //Bester Pfad der Kandidaten
  $candidate = argmax(Highest[n][s]), 1 \leq s \leq k_n$ //Bester letzter Kandidat
  for $i = n-1 \ldots 1$ **do**
    $resultList.add(candidate)$
    $candidate := candidate.getParent()$
  end for
  $reverse(resultList)$
end procedure

und $1 \leq j \leq k_i$. Das Ergebnis ist eine Liste von Kandidaten, die den besten Pfad bilden. Zusätzlich wird jedem Kandidaten der Messungen $p_2 \ldots p_n$ der Wert der ST-Funktion (Formel 4.6) zugewiesen, durch welchen sich die Unsicherheit (Formel 4.9) berechnet. Der restliche Teil des Algorithmus verläuft wie er in der Arbeit von Lou et al. [LZZ+09] beschrieben wurde.

Das Map Matching wird für jedes ausgewählte Taxi parallel berechnet. Die Regionen, wo eine Fahrt verläuft, werden gespeichert, so dass bei Änderungen am Straßennetz nur die betroffenen Fahrten neu berechnet werden müssen.

6 Fallstudien

Nach der Implementierung wurde untersucht, wie groß der Einfluss des Nutzers ist. Dazu wurde zuerst verfolgt, wie sich Änderungen an den Parametern auf das Ergebnis auswirken und ob diese sinnvoll sind. Anschließend wurde die Qualität des gegebenen Straßennetzes betrachtet und schrittweise verbessert.

6.1 Iterative Parameteranpassung

Zuerst wurde der Einfluss der Parametereinstellung untersucht. Je schlechter die Qualität der Messdaten und des Straßengraphen, desto mehr sollten sich Parameter auf das Ergebnis auswirken. Bei exakten Daten sollte der Algorithmus immer das richtige Ergebnis liefern. Da die verwendeten Daten ein relativ hohes Messintervall von einer Minute haben, mussten Parameter angepasst werden. Auch der Straßengraph weicht an Stellen von der Realität ab. Die zwei wichtigsten Parameter des ST-Matching sind die Anzahl der Kandidaten und die maximale Distanz zwischen Messpunkt und Kandidat. Zur Untersuchung wurden die Anfangsparameter auf drei Kandidaten und $100m$ maximaler Distanz gesetzt. Eine relativ kleine Anzahl an Kandidaten erlaubt schnelleres Berechnen der Iterationen, während andere Parameter bestimmt werden. Es wurden zehn zufällig ausgewählte Fahrzeuge betrachtet, die jeweils zwischen 2000 und 4000 Fahrten besaßen. Der erste Schritt nach Laden der Taxifahrten ist in der Regel das Map Matching, um eine Übersicht über die Qualität der Daten zu verschaffen. Weil die Daten oft nicht bereinigt sind, ist es meistens wünschenswert unrealistische Fahrten durch den Vorverarbeitungsschritt zu filtern. Nach dem Entfernen solcher Fahrten, schlug der

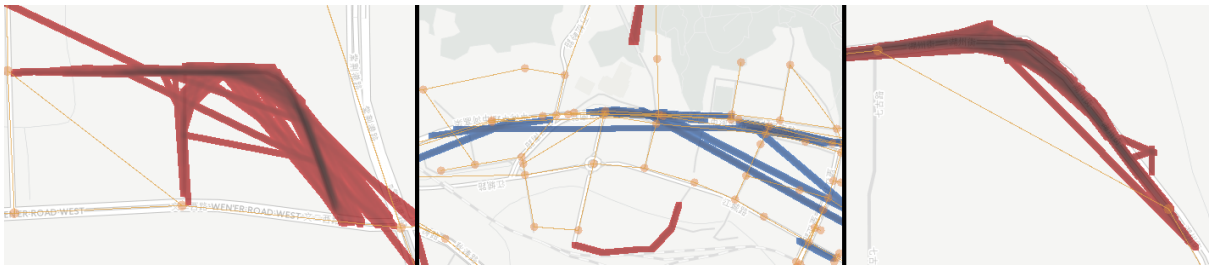

Abbildung 6.1: Fehler im Straßennetz. Links:Fehlende Straßenkanten & Knoten. Mitte: Straße mit vielen Kanten. Rechts: Ungerader Straßenverlauf.

Map Matching Algorithmus bei jeder vierten Fahrt fehl. Die meisten Fehler entstanden weil keine Kandidaten in der Nähe einer Messung gefunden wurden.

Kandidatendistanz

Obwohl die Kandidatendistanz zu $100m$ gesetzt wurde und GPS-Messungen in der Regel eine deutliche bessere Genauigkeit besitzen, entstanden viele Fehler. Die Ursache davon ist häufig das Straßennetz. Falls eine Straße nicht im Graphen vorhanden ist, so kann das richtige Ergebnis nicht berechnet werden. Anhand der Fehlerdarstellung konnte jedoch erkannt werden, dass viele Straßen vorhanden waren und trotzdem nicht vom Algorithmus gefunden wurden. In den meisten solchen Fällen war die Straße an den Knoten annähernd exakt, aber wegen ungeradem Verlauf weichte sie dazwischen von der Realität ab (Abbildung 6.1 rechts). Solche Fälle entstanden meistens am Rand der Stadt, während im Zentrum von Hangzhou und in Wohngebieten, wo die Straßen kurz und regelmäßig verlaufen, kaum Fehler auftauchten. Das Problem kann durch Erhöhen der Kandidatendistanz gelöst werden, aber es ist zu erwarten, dass die Unsicherheit zunimmt, wenn weit entfernte Straßen betrachtet werden. Weil immer die k Kandidaten, die am nächsten sind, betrachtet werden, sollte das korrekte Ergebnis durch den Algorithmus auch als Möglichkeit verarbeitet werden. Falls aber eine Straße im Straßennetz nicht vorhanden ist, so wird bei zu hoher Kandidatendistanz ein falsches Ergebnis berechnet. Das verwendete Straßennetz hatte in der Tat auch viele fehlende Straßen (Abbildung 6.1 links) außerhalb des Zentrums. Demonstrativ wurde die Kandidatendistanz auf den maximal zulässigen Wert ($500m$) erhöht. Dabei wurde für die meisten Fahrten ein Ergebnis berechnet (siehe Abbildung 6.2), sogar wenn Straßen nicht im Graphen modelliert waren. Obwohl also Fehler beim Map Matching entstanden, wurde jedoch auch Unsicherheit geringer, was ein besseres Ergebnis andeutet. Um solche offensichtlichen Fehler zu vermeiden, wurde $150m$ als Distanz verwendet, welche ebenfalls eine deutliche Verbesserung im Vergleich zu den Anfangswerten brachte (siehe Tabelle 6.1).

Anzahl an Kandidaten

Die Anzahl betrachteter Kandidaten ist relevant, wenn sich andere Straßen näher an der Straße, die tatsächlich befahren wurde, befinden. Falls Messungen auf den richtigen Straßenkanten liegen, so reicht es nur einen Kandidaten anzuschauen. Gleichzeitig sollte ein zu großer Wert in der Regel nicht zu falschen Ergebnissen führen, da nähere Kandidaten als wahrscheinlicher angenommen werden. In Fällen, wo viele Kanten im Graph nah an einander sind (z.B. mehrere Spuren oder Kreuzungen, siehe Abbildung 6.1 Mitte), kann eine kleine Kandidatenzahl zu Fehlern führen. Aufgrund der Performanz wurde erst am Ende der Studie die Anzahl an Kandidaten von drei auf sechs erhöht. Ein viel größerer Wert sollte sich nicht auf die Ergebnisse auswirken. Diese Änderung führte zu einer Verbesserung der Unsicherheit, da bei komplizierten Verbindungen häufiger die richtigen Pfade gefunden wurden. Die Berechnungszeit wurde dadurch nahezu vervierfacht und betrug 658 Sekunden, statt wie zuvor 182 Sekunden.

Iteration	Anzahl der Kandidaten	Max Distanz zu Kandidaten	Fehlerrate	Unsicherheit	Kommentar
1	3	100m	27.0%	928.7	Map Matching
2	3	100m	24.8%	907.8	Filtern unrealistischer Fahrten
3	3	500m	9.4%	903.9	Sehr große Kandidatendistanz
4	3	150m	18.6%	905.4	Sinnvolle Kandidatendistanz
5	6	150m	18.5%	893.9	Doppelte Anzahl an Kandidaten

Tabelle 6.1: Berechnete Iterationen für Parameteroptimierung.

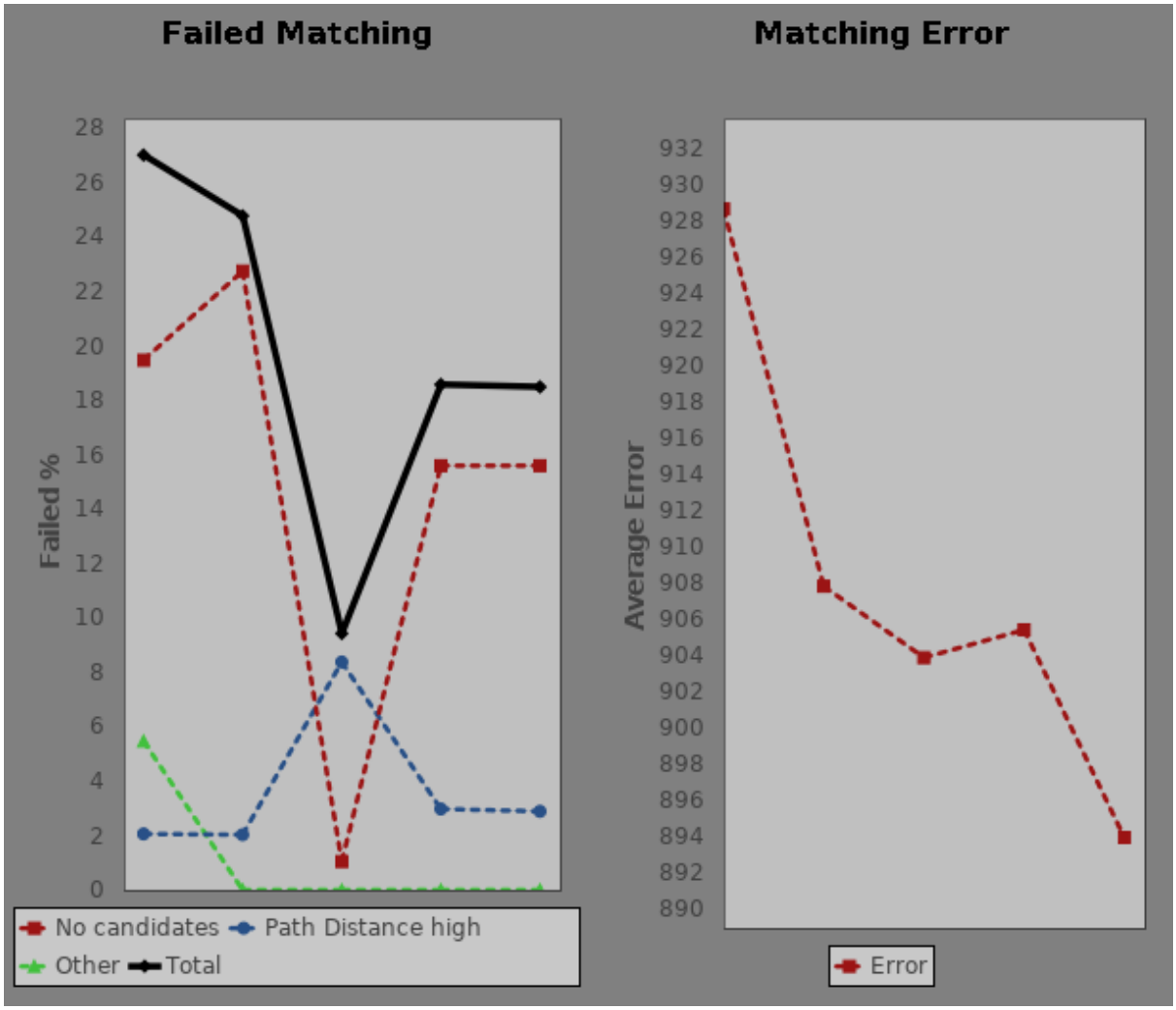

Abbildung 6.2: Fehlerrate (links) und Ungenauigkeit (rechts) der Iterationen während der Parameteroptimierung (Tabelle 6.1).

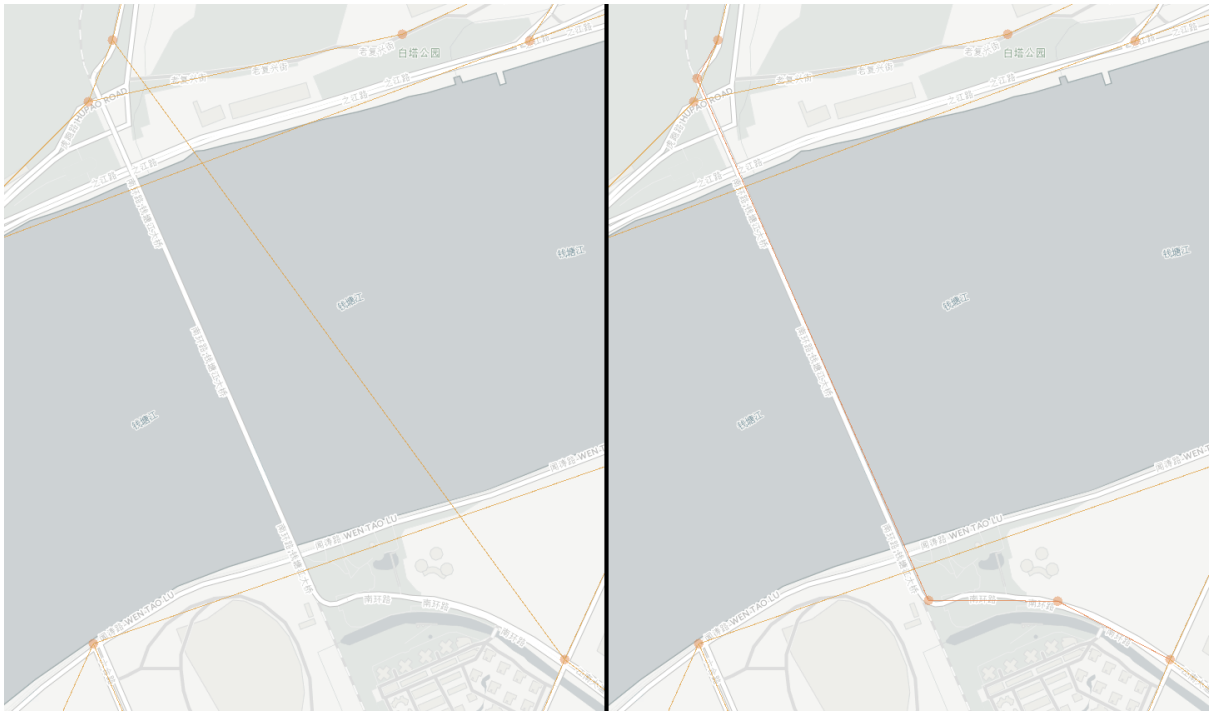

Abbildung 6.3: Brücke über Qiantang-Fluss. Links: Ungenauer Straßengraph. Rechts: Korrigierter Straßengraph.

6.2 Straßennetz Korrektur

Der für diese Arbeit verwendete Straßengraph von Hangzhou war Zentrum detailliert, aber in äußeren Stadtvierteln an vielen Stellen unvollständig. Daher wurde betrachtet, wie die Ergebnisse vom Straßennetz abhängen, wenn dieses verbessert wird. Nach dem Laden der Fahrzeugdaten wurde der Map Matching Algorithmus zur Übersicht durchgeführt. Dabei wurde die Anzahl an Kandidaten auf drei pro Messung gesetzt. Die maximale Distanz betrug $100m$ und maximale Geschwindigkeit $120km/h$. Fehlerhafte Messungen führten zu einer unübersichtlichen Darstellung der Fehler. Viele solcher Fahrten verliefen quer durch Hangzhou (siehe Abbildung 6.5 links). Um diese zu entfernen, wurde die Vorverarbeitung verwendet. Durch Begrenzung der maximalen Distanz wurden diese Probleme bereinigt. Zusätzlich wurden Trajektorien mit weniger als fünf Messungen gefiltert, da Taxifahrten in der Regel mindestens fünf Minuten dauern. Nach einem erneuten Matching war die Fehlerdarstellung deutlich übersichtlicher (siehe Abbildung 6.5 Mitte). Viele der sichtbaren Fehlerquellen waren längere Straßen, die als Kandidaten nicht gefunden wurden. Bei den meisten Brücken über den Qiantang-Fluss entstanden solche Fehler. Die auffälligen Probleme dieser Form wurden durch Unterteilung der jeweiligen Kanten in mehreren Abschnitten behoben (Abbildung 6.3). Anschließend wurde nochmals der Matching Algorithmus für Fahrten in der Nähe dieser Straßen berechnet. An den bearbeiteten Straßen entstanden in der Regel keine Fehler mehr. Auffällig war jedoch, dass oft an den meisten mehrspurigen Straßen keine Pfade zwischen Kandidaten gefunden wurden.

Iteration	Fehlerrate	Unsicherheit	Kommentar
1	30.2%	923.9	Map Matching
2	25.9%	906.1	Filtern unrealistischer Fahrten
3	24.2%	906.5	Korrektur ungerader Straßen
4	24.1%	906.4	Korrektur von komplexen Verbindungen
5	21.3%	906.9	Einfügen von fehlenden Straßen
6	15.5%	895.6	Parameter & weitere Iterationen

Tabelle 6.2: Berechnete Iterationen für Straßennetzoptimierung.

Grund dafür sind die vielen Knoten, die oft an Kreuzungen, Kreisverkehren und Ausfahrten vorhanden sind. Eine Verbesserung solcher Verbindungen erwies sich ohne Vorwissen der Struktur als schwierig und konnte kaum behoben werden. Nach diesen Bereinigungsschritten war die Identifizierung fehlender Straßen deutlich einfacher. An großen Parkplätzen, wie am Flughafen, ist das Map Matching ebenfalls häufig fehlgeschlagen, da diese nicht im Graphen modelliert sind. Durch Betrachten der Rohdaten und der Kartenansicht konnten viele davon erkannt und hinzugefügt werden. Das geschieht durch einfaches Einfügen von einen oder mehreren Knoten, welche durch Kanten mit dem bestehenden Graphen verbunden werden.

Von den 25.096 Fahrten (10 Taxis) ist der Algorithmus am Anfang bei $30,2\%$ der Berechnungen fehlgeschlagen. Das häufigste Problem waren nicht gefundene Kandidaten. Nach Vorverarbeitung wurden 7356 irrelevante Fahrten gefiltert. Ein Großteil davon bestand aus Trajektorien mit nur einer Messung. Von den verbliebenen 17.740 Fahrten war der Algorithmus mit insgesamt 25.9% fehlgeschlagenen Fahrten etwas erfolgreicher. Die Unsicherheit war nach dem Filtern ebenfalls geringer. In den nächsten Schritten blieb diese jedoch näherungsweise konstant (siehe Abbildung 6.4). Auch die Korrektur des Straßenverlaufs änderte die Unsicherheit kaum, weil diese Korrektur sich hauptsächlich auf die bisher fehlgeschlagenen Berechnungen auswirkt. Den größten Effekt auf die Fehlerrate hatte das Einfügen von fehlenden Straßen. Nur durch das Hinzufügen der Strecken mit großem Fehler war der Algorithmus bei ca. 500 Fahrten (2.8%) mehr erfolgreich.

Durch mehrfaches Wiederholen der durchgeführten Schritte, konnte das Ergebnis weiter verbessert werden. Zuletzt wurden auch die in Kapitel 6.1 erzielten Parameter verwendet. Sowohl der berechnete Anteil (siehe Tabelle 6.2), als auch die auf der Karte sichtbaren Fehler (Abbildung 6.5) wurden deutlich reduziert. Ein noch besseres Ergebnis könnte durch Bedienung von Experten, die mit dem Straßennetz vertraut sind, erreicht werden.

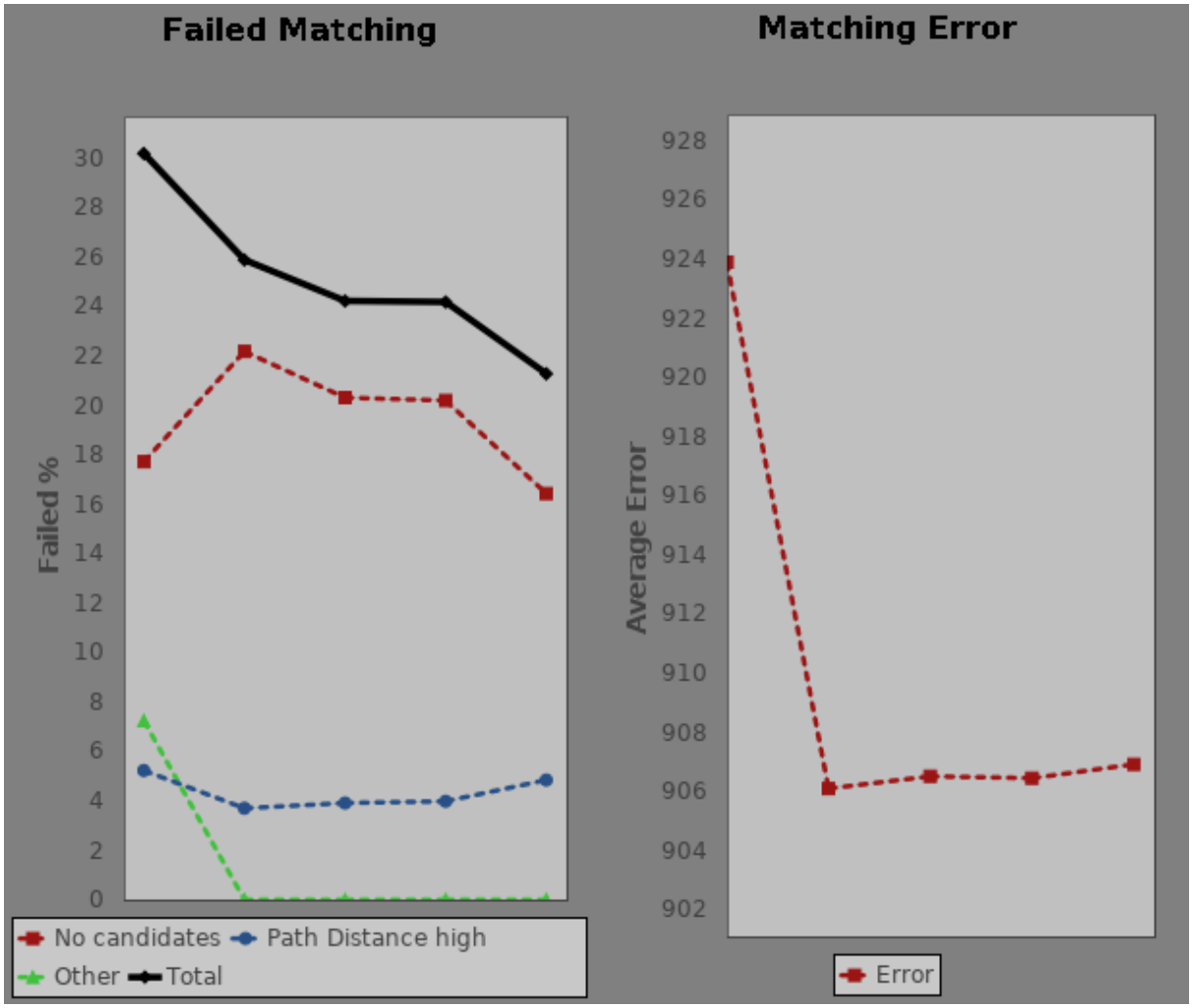

Abbildung 6.4: Fehlerrate (links) und Ungenauigkeit (rechts) der Iterationen während der Straßennetz Korrektur (Tabelle 6.2).

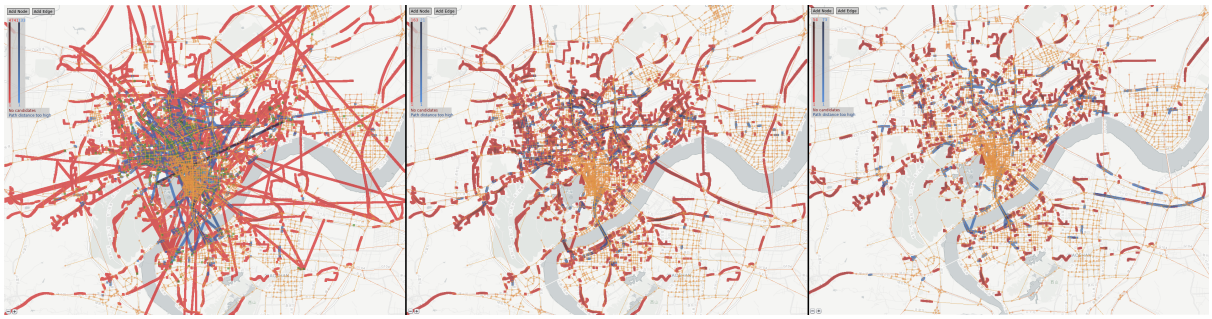

Abbildung 6.5: Links: Map Matching ohne Anpassung. Mitte: Map Matching nach Filtern unrealistischer Fahrten. Rechts: Map Matching nach allen Schritten (siehe Tabelle 6.2).

7 Zusammenfassung und Ausblick

In dieser Arbeit wurde zuerst erklärt, welche Probleme bei nicht bereinigten Geodaten vorkommen. Anschließend wurden Map Matching Verfahren als Vorbereitungsschritt für die Daten beschrieben. Es wurde ein System zur Darstellung eines Map Matching Algorithmus entwickelt, welches mit Hilfe verschiedener Ansichten die Daten, Ergebnisse und Fehler im Matching Algorithmus darstellt. Die Darstellung der Daten und Interaktion des Anwenders wurden dabei als Schwerpunkte betrachtet. Große Fehlerstellen werden im Verfahren hervorgehoben, sodass durch Änderungen am Straßennetz und Einstellen von Parametern das Matching Verfahren effektiv optimiert wird. Das entwickelte System verbessert an einem Datensample die Ergebnisse des Map Matching und führt dieses anschließend auf den großen Datensatz aus.

Nach einer Implementierung wurde das Konzept, anhand eines von Taxis gesammelten Datensatzes, in Fallstudien ausgewertet. Ergebnisse zeigten, dass das Map Matching bei einem Großteil der Rohdaten fehlschlägt, wenn das Verfahren nicht an die Daten angepasst wird. Danach wurde demonstriert welchen Einfluss der Analyst auf das Map Matching hat.

Ausblick

Das Pre-Precessing von Daten ist ein Prozess, der meistens gebraucht wird, um die Daten zu verstehen. Obwohl viele Arbeiten Ansätze bieten, um problematische Daten zu finden, werden solche Daten häufig verworfen, auch wenn sie eventuell relevant sind. Ein mögliche Erweiterung des Ansatzes dieser Arbeit besteht in der Kombination der Bereiche. Es können zum Beispiel falsche oder fehlende Komponenten im Straßennetz erkannt und manuell korrigiert werden. Durch maschinelles Lernen sollen anschließend ähnliche Probleme automatisch gelöst werden.

Ein Aspekt, der in dieser Arbeit nicht angeschaut wurde, sind die Eigenschaften des Straßennetzes in Abhängigkeit der Region. Es ist in der Regel im Zentrum und in Wohngebieten dichter als an anderen Stellen, sodass an solchen Regionen verschiedene Parameter für das Map Matching eventuell sinnvoller sind.

Literaturverzeichnis

[AAW07] G. Andrienko, N. Andrienko, S. Wrobel. „Visual analytics tools for analysis of movement data“. In: *ACM SIGKDD Explorations Newsletter* 9.2 (2007), S. 38–46 (zitiert auf S. 13, 18).

[BRG+12] J. Bernard, T. Ruppert, O. Goroll, T. May, J. Kohlhammer. „Visual-interactive preprocessing of time series data“. In: *Proceedings of SIGRAD 2012; Interactive Visual Analysis of Data; November 29-30; 2012; Växjö; Sweden*. 081. Linköping University Electronic Press. 2012, S. 39–48 (zitiert auf S. 14).

[chr] chrisadamson. *Building Maps into Your Swing Application with the JXMapViewer Blog*. https://community.oracle.com/docs/DOC-983180 (zitiert auf S. 32).

[Col90] W. C. Collier. „In-vehicle route guidance systems using map-matched dead reckoning“. In: *Position Location and Navigation Symposium, 1990. Record. The 1990's-A Decade of Excellence in the Navigation Sciences. IEEE PLANS'90., IEEE*. IEEE. 1990, S. 359–363 (zitiert auf S. 14).

[FPS96] U. Fayyad, G. Piatetsky-Shapiro, P. Smyth. „From data mining to knowledge discovery in databases“. In: *AI magazine* 17.3 (1996), S. 37 (zitiert auf S. 13, 17).

[JFr] JFreeChart. *JFreeChart*. http://www.jfree.org/jfreechart/ (zitiert auf S. 33).

[KAF+08] D. Keim, G. Andrienko, J.-D. Fekete, C. Görg, J. Kohlhammer, G. Melançon. „Visual analytics: Definition, process, and challenges“. In: *Information visualization*. Springer, 2008, S. 154–175 (zitiert auf S. 13).

[LLY+15] M. Lu, C. Lai, T. Ye, J. Liang, X. Yuan. „Visual analysis of route choice behaviour based on gps trajectories“. In: *Visual Analytics Science and Technology (VAST), 2015 IEEE Conference on*. IEEE. 2015, S. 203–204 (zitiert auf S. 14).

[LWW15] Q. Li, H. Wang, Y. Wu. „Visual data quality analysis for taxi gps data“. In: (2015) (zitiert auf S. 14).

[LYC10] Z. Liao, Y. Yu, B. Chen. „Anomaly detection in GPS data based on visual analytics“. In: *Visual Analytics Science and Technology (VAST), 2010 IEEE Symposium on*. IEEE. 2010, S. 51–58 (zitiert auf S. 14).

[LZS+11] B. Li, D. Zhang, L. Sun, C. Chen, S. Li, G. Qi, Q. Yang. „Hunting or waiting? Discovering passenger-finding strategies from a large-scale real-world taxi dataset“. In: *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2011 IEEE International Conference on*. IEEE. 2011, S. 63–68 (zitiert auf S. 11).

[LZZ+09] Y. Lou, C. Zhang, Y. Zheng, X. Xie, W. Wang, Y. Huang. „Map-matching for low-sampling-rate GPS trajectories“. In: *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM. 2009, S. 352–361 (zitiert auf S. 15, 21–23, 33, 34).

[MKH06] M. Modsching, R. Kramer, K. ten Hagen. „Field trial on GPS Accuracy in a medium size city: The influence of built-up“. In: *3Rd workshop on positioning, navigation and communication*. 2006, S. 209–218 (zitiert auf S. 15, 18).

[MKYM12] T. Miwa, D. Kiuchi, T. Yamamoto, T. Morikawa. „Development of map matching algorithm for low frequency probe data“. In: *Transportation Research Part C: Emerging Technologies* 22 (2012), S. 132–145 (zitiert auf S. 15).

[Ope] OpenStreetMap. *OpenStreetMap*. https://wiki.openstreetmap.org/wiki/Tiles (zitiert auf S. 27).

[OQN03] W. Y. Ochieng, M. A. Quddus, R. B. Noland. „Map-matching in complex urban road networks“. In: (2003) (zitiert auf S. 15).

[PH08] O. Pink, B. Hummel. „A statistical approach to map matching using road network geometry, topology and vehicular motion constraints“. In: *2008 11th International IEEE Conference on Intelligent Transportation Systems*. IEEE. 2008, S. 862–867 (zitiert auf S. 19).

[Pre] Prefuse. *Prefuse*. http://prefuse.org/ (zitiert auf S. 32).

[QON07] M. A. Quddus, W. Y. Ochieng, R. B. Noland. „Current map-matching algorithms for transport applications: State-of-the art and future research directions“. In: *Transportation research part c: Emerging technologies* 15.5 (2007), S. 312–328 (zitiert auf S. 19).

[Shn01] B. Shneiderman. „Inventing discovery tools: Combining information visualization with data mining“. In: *International Conference on Discovery Science*. Springer. 2001, S. 17–28 (zitiert auf S. 13).

[Shn96] B. Shneiderman. „The eyes have it: A task by data type taxonomy for information visualizations“. In: *Visual Languages, 1996. Proceedings., IEEE Symposium on*. IEEE. 1996, S. 336–343 (zitiert auf S. 13, 18).

[WBK00] C. E. White, D. Bernstein, A. L. Kornhauser. „Some map matching algorithms for personal navigation assistants“. In: *Transportation research part c: emerging technologies* 8.1 (2000), S. 91–108 (zitiert auf S. 15).

[WLY+13] Z. Wang, M. Lu, X. Yuan, J. Zhang, H. Van De Wetering. „Visual traffic jam analysis based on trajectory data“. In: *IEEE Transactions on Visualization and Computer Graphics* 19.12 (2013), S. 2159–2168 (zitiert auf S. 11, 13).

[WWFZ13] H. Wei, Y. Wang, G. Forman, Y. Zhu. „Map matching by Fréchet distance and global weight optimization“. In: *Technical Paper, Departement of Computer Science and Engineering* (2013) (zitiert auf S. 19).

Alle URLs wurden zuletzt am 01. 02. 2017 geprüft.

Erklärung

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

Ort, Datum, Unterschrift