

Institut für Visualisierung und Interaktive Systeme
Universität Stuttgart
Universitätsstraße 38
D-70569 Stuttgart

Diplomarbeit Nr. 3063

Konsistente Beschriftung von 3D Karten

Ilona Heurich

Studiengang:	Informatik
Prüfer:	Prof. Dr. Daniel Weiskopf
Betreuer:	Dipl.-Inf. Harald Sanftmann

begonnen am:	14. Juli 2010
beendet am:	21. Januar 2011

CR-Klassifikation:	I.3.7, J.2, H.5.1
---------------------------	-------------------

Zusammenfassung

Geoinformationssysteme werden in Politik, Wirtschaft und Gesellschaftswissenschaften immer wichtiger. Eine Karte als Schnittstelle zu diesen Systemen muss mit gut lesbaren und schnell erfassbaren Informationen versehen sein. Ein unverzichtbares Hilfsmittel, Informationen auf einer Karte darzustellen, sind Beschriftungen. In gängigen Beschriftungssystemen wie GoogleMaps [Goob] kann es zu Artefakten wie plötzliches Auftauchen und Verschwinden von Beschriftungen kommen. Been et al. [BDY06] definiert einen Konsistenzbegriff für interaktive (Verschieben und Zoomen) zweidimensionale Karten, die ein solches Verhalten vermeiden. Ziel dieser Diplomarbeit ist, ein Beschriftungsverfahren zu entwickeln, welches diese gegebene Konsistenz auf 3D-Beschriftungen erweitert. Zunächst wird in dieser Arbeit das Verfahren von Been et al. [BDY06] implementiert. Schließlich wird der Konsistenzbegriff für gekippte Kartendarstellungen mit Geländemodellen diskutiert. Dabei wird die Einschränkung der Konsistenz deutlich und es zeigt sich, dass man, um für den Benutzer ansprechende und sinnvolle Beschriftungen zu erhalten, den Konsistenzbegriff erweitern muss. In dieser Arbeit werden Verfahren vorgestellt, die auf Basis dieses auf drei Dimensionen erweiterten Konsistenzbegriffs eine schnelle Beschriftung erreichen. Dabei werden Überdeckungen entweder in der Vorverarbeitungsphase durch geschlossene Formeln bzw. diskrete Zustände oder im Interaktionsschritt berechnet. Das beste Ergebnis liefert eine Kombination aus dem Verfahren von Been et al. [BDY06] und einer weiteren Schnittberechnung während der Interaktionsphase. Diese garantiert zwar für Verschieben, Rotieren und Kippen eine abgeschwächte Konsistenzeigenschaft, erhält sie allerdings für das Zoomen und lässt ein hohen Grad an Interaktionen mit dem Benutzer zu.

Abstract

Geoinformation systems are of increasing importance in politics, economics and society. A map which is an interface to these systems has to be provided with well readable and fast detectable information. Labels are essential for the illustration of such information. In commonly used illustrations like GoogleMaps [Goob] there exist artifacts like the sudden dis- and reappearance of labels. Been et al. [BDY06] defines a consistency term for interactive (pan and zoom), two-dimensional maps that avoid such behavior. The focus of this thesis is to develop a labeling procedure that extends this given consistency for 3D-Labeling. First of all the procedure of Been et al. [BDY06] is implemented and some practical considerations are made. Finally the consistency term for tilted map displays are discussed. Thereby, the restriction of the consistency becomes clear and it shows that, in order to produce a pleasant and meaningful labeling, artifacts have to be permitted. In this thesis procedures are presented that produce a fast labeling on basis of this for three dimensions extended consistency term. Intersections of labels are either computed in the preprocessing phase by closed formulas or discrete states or during the interaction step. The best result is provided by a combination of the procedure of Been et al. [BDY06] and a further intersection computation in the interaction phase. With this approach there is no consistency guarantee for panning, rotation and tilting but it conserves the consistency for zoom-operator and a high degree of user interactions are possible.

Inhaltsverzeichnis

1. Einleitung	1
1.1. Motivation	1
1.2. Aufgabenstellung	5
1.3. Aufbau der Diplomarbeit	6
2. Grundlagen	7
2.1. Geoinformationssysteme	7
2.1.1. Messung der Erde: geografische Länge und Breite	9
2.1.2. Projektion und Koordinaten	11
2.2. Georeferenzierung	13
2.2.1. Beschriftungen	14
2.2.2. NP-Vollständigkeit des Beschriftungsproblems	16
2.2.3. OpenStreetMap	19
2.2.4. Vorgehensweise bei der automatischen Beschriftung	21
2.2.4.1. Auswahlalgorithmen	23
2.2.4.2. Schnelles Labeling	29
3. Schnelle konsistente Kartenbeschriftung für interaktive Karten	33
3.1. Modell von Been et al.	33
3.1.1. Konsistenzbegriff für interaktive Karten	33
3.1.2. Ablauf des Verfahrens von Been et al.	34
3.1.3. Beschriftungsoptimierungsproblem	40
3.2. Praktische Betrachtungen	42
3.2.1. Terrainrenderer	42
3.2.2. Berechnungsschritte	43
3.2.2.1. Labeldarstellungen	43
3.2.2.2. Schnittberechnungen	45
3.2.2.3. Beschleunigungsstrategien	48
3.3. Vor- und Nachteile des Modells von Been et al.	54
4. 3D-Kartenbeschriftung	57
4.1. Anforderungen an eine 3D-Kartenbeschriftung	57
4.1.1. Erweiterter Konsistenzbegriff durch Rotieren um die Hochachse	57
4.1.2. Gekippte Kartendarstellungen	61

4.1.3. Gekippte Kartendarstellungen mit Geländemodellen	64
4.2. Ansätze für 3D-Kartenbeschriftungen	65
4.2.1. Erweiterung des Ansatzes von Been et al.	65
4.2.2. Sampling	68
4.2.3. Schnittberechnungen zur Interaktionsphase	69
4.3. Algorithmische Betrachtung der implementierten Verfahren	70
5. Ergebnisse	75
5.1. Testbedingungen	75
5.2. Laufzeiten	75
5.2.1. Vorberechnungsphase nach dem Modell von Been et al.	75
5.2.2. Sampling der Rotation um die Hochachse in der Vorverarbeitungsphase	79
5.2.3. Schnittberechnungen in der Interaktionsphase	82
5.3. Speicherverbrauch	84
5.3.1. Vorberechnungsphase nach dem Modell von Been et al.	84
5.3.2. Sampling der Rotation um die Hochachse in der Vorverarbeitungsphase	87
5.3.3. Schnittberechnungen in der Interaktionsphase	87
6. Mögliche Erweiterungen	89
7. Zusammenfassung	93
A. Anhang	95
Literaturverzeichnis	107

Abbildungsverzeichnis

1.1. OpenStreetMap	2
1.2. Google Earth	3
1.3. Google Earth Artefakte beim Zoomen	4
1.4. Google Earth Artefakte beim Verschieben	4
2.1. Aspekte von Geoinformationssystemen	8
2.2. Längengrad der Erde	10
2.3. Breitengrad der Erde	10
2.4. Kartenprojektionen	12
2.5. Beschriftung punkthafter Objekte	14
2.6. Bewertungsfunktion für Labelpositionen	15
2.7. Beschriftung linienhafter Objekte	16
2.8. Beschriftung flächenhafter Objekte	16
2.9. Beschriftungspositionen eines punkthaften Objektes	17
2.10. Das Beschriftungsproblem am Beispiel	18
2.11. Verschiedene Darstellungen der OpenStreetMap-Daten	20
2.12. OpenStreetMap-Datennutzung	20
2.13. Vorgehensweise bei der automatischen Textplatzierung	22
2.14. Backtracking	24
2.15. Lokales Minima	25
2.16. Algorithmus von Hirsch	28
2.17. Vergleich Beschriftungsqualität von Auswahlalgorithmen	28
2.18. Vergleich der Laufzeiten der Auswahlalgorithmen	29
2.19. Vergleich der Qualität von Auswahlalgorithmen	30
3.1. Inkonsistenz bei der Kartenbeschriftung	35
3.2. Statische Platzierung	36
3.3. Invariante Punktplatzierung	38
3.4. Ausgedehnte Labels	38
3.5. Aktive Bereiche von Labels	39
3.6. 2D-Ansicht der ausgedehnten Label	39
3.7. Labelarten	44
3.8. Straßenbeschriftung	45
3.9. Schnitt zwischen zwei Labels	46

3.10. Gültige Schnittpunkte	47
3.11. Schnittmöglichkeiten zwischen zwei Rechtecken	47
3.12. Schnittberechnung zwischen zwei Rechtecken	49
3.13. Detailstufen	50
3.14. Buckets	51
3.15. Buckets die zur Überdeckungsberechnung geprüft werden müssen	52
3.16. Auswahl der richtigen Skalierung	52
3.17. Schnittberechnung in Prioritätsreihenfolge	53
3.18. Bucket-Liste	54
4.1. Rotationsformen von Beschriftungen	58
4.2. Überlappungen bei Rotation	59
4.3. Rotierte Labels	59
4.4. Konsistenzbetrachtung bei Rotation	60
4.5. Verhalten der Labels beim Kippen	61
4.6. Überdeckungen durch Benutzerinteraktionen	62
4.7. Artefakte beim Kippen	63
4.8. Inkonsistenz beim gelockerten Konsistenzbegriff	64
4.9. gekippte Kartendarstellung mit Geländemodell	65
4.10. Allgemeiner Schnitt zweier Rechtecke	67
4.11. Gelockerter Konsistenzbegriff	68
4.12. Hybrides Verfahren	69
4.13. Überdeckungserkennung	71
4.14. Überdeckungsberechnung beim Kippen	73
5.1. Laufzeiten des Leonberg-Datensatzes	76
5.2. Laufzeiten der Detailstufen des Leonberg-Datensatzes	78
5.3. Laufzeitenvergleich für den Fall 3	79
5.4. Aufgeschlüsselte Laufzeiten	80
5.5. Laufzeitenvergleich für das Sampling	81
5.6. Laufzeitenvergleich für das Sampling	82
5.7. Laufzeitenvergleich für die Schnittberechnungen zur Interaktion	83
6.1. Konfliktgraph	90

Tabellenverzeichnis

5.1. Verschiedene Fälle für Schnittberechnungen	76
5.2. Labelarten im Leonberg-Datensatz	76
5.3. Detailstufen des Leonberg-Datensatzes	77
5.4. Labelarten in den verschiedenen Datensätzen	78
5.5. Speicherverbrauch der verschiedenen Datensätze	84
5.6. Speicherverbrauch der Labelliste	85
5.7. Speicherverbrauch der Straßen-Punkte	85
5.8. Speicherverbrauch der Eckgeraden und Seitenflächen	86
5.9. Speicherverbrauch der Label-Id- und Bucketliste	86
5.10. Speicherverbrauch Samples	87

Verzeichnis der Algorithmen

2.1. Gradientenabstieg	25
2.2. Simulated Annealing	26
3.1. Konsistente Beschriftung	40
3.2. Greedy-Algorithmus für konsistente Beschriftungen	42
3.3. Ablauf der Implementierung von Been et al. [BDY06]	43
3.4. Ablauf der beschleunigten Implementierung von Been et al. [BDY06]	50
4.1. Beschriftungsplatzierung	72

1. Einleitung

1.1. Motivation

Fragestellungen mit einem räumlichen Aspekt sind so alt wie die Menschheit

- Wo bin ich?
- Wo finde ich ... ?
- Wie komme ich nach ... ?
- Wie weit ist es nach ... ?
- ...

Informationen, die diese Fragen beantworten, werden Geoinformationen genannt. Sie bilden die Basis für Abläufe, Planungen, Maßnahmen oder Entscheidungen in der Verwaltung und Wirtschaft. Nach einer Schätzung des Bundesamts für Kartographie und Geodäsie haben rund 80% aller Entscheidungen im öffentlichen und privaten Leben einen räumlichen Bezug [Geo03].

Informationssysteme, die diese Geoinformationen nutzen, sind von immenser Bedeutung. Nach einer Schätzung von Marktstudien im Jahr 2002 lag der weltweite Umsatz dieser Systeme bereits bei über einer Milliarde Euro. Die Einsatzmöglichkeiten solcher Informationssysteme sind breit gefächert. Sie bilden zum Beispiel die Grundlage aller Navigationssysteme, liefern lokale Wettervorhersagen, unterstützen die Einteilung von Wahlkreisen, helfen bei der Erstellung des Waldschadenberichts oder sind bei der Ermittlung von Umweltsündern sowie Schmugglerbanden nützlich. Ebenso basieren zwei Drittel aller Wirtschaftsprozesse auf der Nutzung räumlicher Daten [Ase02].

Als Schnittstelle zwischen Benutzer und einem auf Geodaten basierenden Informationssystem werden Bildschirmkarten eingesetzt. Auf solchen Karten sind die Beschriftungen bzw. Labels wie Straßen, Hausnummern oder zahlreiche Zusatzinformationen wie z.B. Schwimmbäder, Restaurants oder Tankstellen gar nicht mehr wegzudenken [Ase02]. Diese Beschriftungen sind wesentliche Hilfsmittel zur Orientierung und zur Bestimmung eines exakten Ortes.

1. Einleitung

Die Herausforderung ist nun, visuell ansprechende Karten zu beschriften. Kleider machen Leute, und das gilt auch in der Kartographie. Schlechte, ungepflegte, laienhafte Schriftnordnung verdirbt selbst das beste Kartenbild und erschwert das Kartenlesen in unverantwortlicher Weise, schrieb einst Eduard Imhof, Pionier des systematischen Kartenzeichnens und Gründer des ersten kartographischen Instituts der Welt [Sch95].

So ist eine wichtige Anforderung an eine gute Kartenbeschriftung, dass sie leicht lesbar und so angeordnet ist, dass der Benutzer die Verknüpfung zwischen zu beschriftendem Objekt und Beschriftung leicht herstellen kann. Dabei ist wichtig, dass sich die verschiedenen Beschriftungen nicht überlappen und die Informationsdichte der Beschriftungen möglichst hoch ist (vgl. Abbildung 1.1).

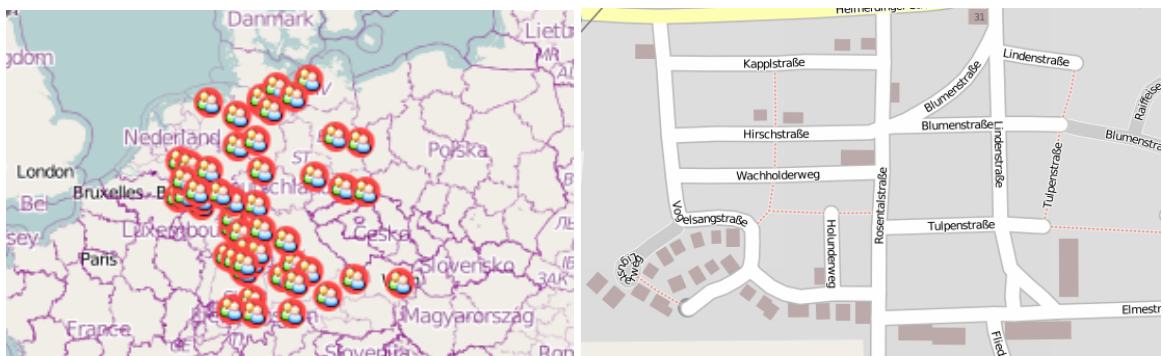


Abbildung 1.1.: OpenStreetMap-Ansichten. Links: Es werden so viele Labels angezeigt, dass sie nicht mehr zu den Interessenpunkten zugeordnet werden können. Rechts: Die Häuser-Labels sind sehr spärlich [Ope].

Das Problem, dass sich keine zwei verschiedene Beschriftungen überlappen dürfen, wird als allgemeines Beschriftungsproblem (Map Labeling Problem) bezeichnet und ist NP-vollständig [JS93]. Dies bedeutet, dass kein effizienter Algorithmus existiert und dadurch die Beschriftung selbst auf dem schnellsten Computer noch zu langsam ist für den praktischen Einsatz. Um trotzdem Beschriftungen in Echtzeit erzielen zu können, werden Heuristiken eingesetzt. Diese versuchen unter der Zuhilfenahme von Schätzungen, Wahrscheinlichkeiten, Regeln oder anderen Hilfsmitteln ein optimales Ergebnis anzunähern [Peto3]. Solche Heuristiken sind in den letzten Jahrzehnten Gegenstand vieler Studien.

Der Einsatz von interaktiven Karten (Zoomen, Verschieben, Kippen, Rotieren um die Hochachse) auf dem Computer, die durch die schnelle Entwicklung von 2D- und 3D-Modellierung sowie Rendering-Technologien möglich sind, erlaubt eine Vielzahl neuer Anwendungen. So können zum Beispiel Benutzer in Google Earth [Gooa] Städte erkunden oder Routen planen (siehe Abbildung 1.2) [QWC⁺09].

Durch Zoomen (freie Maßstabswahl) und Verschieben (Änderung des Kartenausschnitts) im 2D sowie Rotation um die Hochachse und Kippen im 3D entstehen neue Freiheitsgrade. Jede dieser Aktionen benötigt eine Neubeschriftung, die in Echtzeit erfolgen muss [Peto3].



Abbildung 1.2.: Google Earth Ansicht auf den Stuttgarter Hauptbahnhof [Gooa].

Die Herausforderung hierbei ist, einen Algorithmus zu finden, der schnell genug ist, dem interaktiven Charakter solcher dynamischer Karten (zum Beispiel zur Navigation) gerecht zu werden, zudem eine übersichtliche Darstellung liefert und dem Benutzer eine Orientierungshilfe bietet. Es sollen sich keine Labels überdecken oder beim Verschieben und Zoomen ständig unerwartet erscheinen und wieder verschwinden (vgl. Abbildung 1.3, 1.4).



Abbildung 1.3.: Das linke Bild zeigt wie die beiden Labels 49 und E331 platziert werden. Beim Hereinzoomen werden diese durch das Erscheinen von Label 520 unerwartet auf eine andere Position verschoben (siehe rechts). Darüber hinaus ergeben sich in der Beschriftung Überlappungen. Platzierte Labels verändern ihre Position wenn man mit der Maus darüber fährt [Gooa].

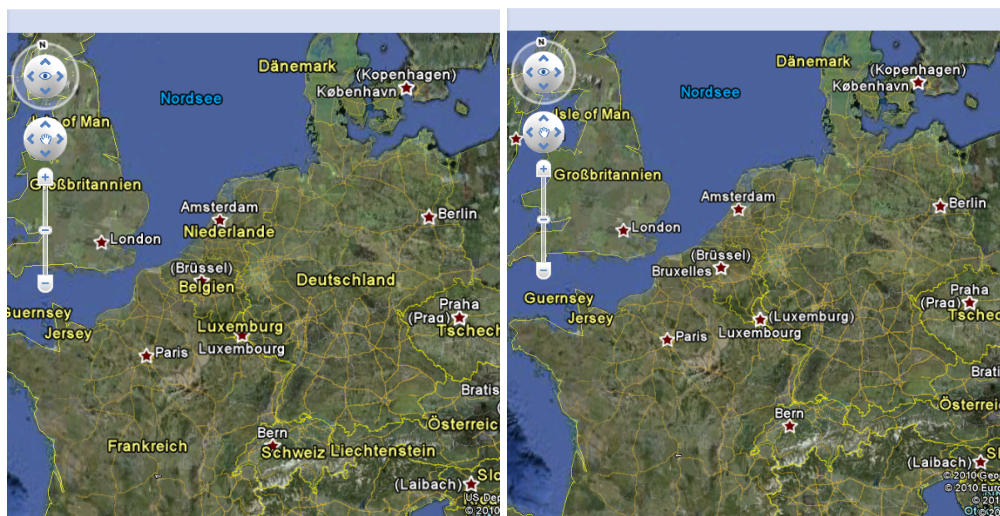


Abbildung 1.4.: Beim Verschieben des Sichtbereiches nach links verschwindet die Deutschland-Beschriftung, obwohl noch genügend Platz für die Beschriftung vorhanden ist [Gooa].

1.2. Aufgabenstellung

Beschriftungen spielen eine entscheidende Rolle bei der Kartendarstellung. Die Aufgabe besteht darin, eine Untermenge der Beschriftungen auszuwählen und auf der Karte zu platzieren, ohne dass sich dabei Beschriftungen überdecken. Dabei sollte die Untermenge so gewählt sein, dass die wichtigsten Beschriftungen angezeigt werden. In dynamischen Kartendarstellungen ist es wünschenswert, dass sich die Beschriftungen konsistent [BDYo6] verändern. Konsistenz fasst dabei verschiedene Kriterien zusammen:

- Beim Rauszoomen sollten keine Beschriftungen erscheinen, beim Reinzoomen keine verschwinden.
- Solange eine Beschriftung sichtbar ist, sollte sich seine Position und Größe kontinuierlich beim Zoomen und Verschieben ändern.
- Beim Verschieben der Karte sollten keine Labels verschwinden oder auftauchen.
- Die Position und Auswahl einer Beschriftung soll nur von dem aktuellen gezeigten Kartenausschnitt abhängen.

Been et al. stellen in [BDYo6] eine Lösung zum Beschriften von 2D Karten vor, welches diese Kriterien unter Zoomen und Translation erfüllt. Das Ziel dieser Diplomarbeit ist es ein konsistentes Verfahren für die Beschriftung von 3D Karten zu entwickeln.

Ziele der Arbeit im Einzelnen:

- Implementierung des Verfahrens aus [BDYo6]
- Konzeption und Implementierung einer Erweiterung für gekippte Kartendarstellungen. Dabei sollen der Kippwinkel und die Rotation um die Hochachse als zusätzliche Freiheitsgrade für den Benutzer zur Verfügung stehen.
- Konzeption und Implementierung einer Erweiterung für gekippte Kartendarstellungen mit Geländemodellen.

Mögliche Erweiterung:

- Konzeption und Implementierung einer Erweiterung für dynamische Beschränkung von Beschriftungen durch den Benutzer. Dabei wählt der Benutzer z.B. aus, dass nur Restaurant-POIs angezeigt werden sollen.
- Konzeption und Implementierung einer Erweiterung für sich veränderndes Geländemodell, wie z.B. uniforme Skalierung der Berge, nicht uniforme Skalierung, beliebige Verzerrung.

Die Implementierung soll grundsätzlich auf C/C++ basieren. Als Graphik-API soll OpenGL eingesetzt werden. Für die Implementierung sollte ein bestehender Terrainrenderer verwendet werden.

1.3. Aufbau der Diplomarbeit

Die Arbeit gliedert sich in folgender Weise:

Kapitel 2 – Grundlagen: In diesem Kapitel werden die notwendigen Grundlagen dieser Arbeit beschrieben. Zunächst werden Geoinformationssysteme (GIS) definiert und die geographischen Grundlagen, wie Bestimmung der geographischen Lage durch Längen- und Breitengrade sowie Projektionen und Koordinatensysteme eingeführt. Die Notwendigkeit einer guten Kartenbeschriftung als Schnittstelle zwischen dem Benutzer und dem GIS wird im Abschnitt Georeferenzierung behandelt. Des Weiteren werden die verschiedenen Formen, das generelle Vorgehen und bereits vorhandene Algorithmen zur Beschriftung vorgestellt. Dabei wird auf die Schwere des Problems einer guten Beschriftung eingegangen und entsprechend schnelle Heuristiken zur Lösung vorgestellt.

Kapitel 3 – Schnelle konsistente Kartenbeschriftung für interaktive Karten: Dieses Kapitel stellt ein Modell von Been et al. [BDYo6] vor, das zunächst einen Konsistenzbegriff einführt und schließlich einen schnellen Beschriftungsalgorithmus liefert, welcher diese Konsistenzbedingung erfüllt. Des Weiteren werden einige praktische Betrachtungen zu diesem Verfahren vorgestellt. Das Kapitel schließt mit einer Gegenüberstellung der Vor- und Nachteile dieses Verfahrens.

Kapitel 4 – 3D-Kartenbeschriftung: Dieses Kapitel befasst sich mit 3D-Beschriftungen. Zunächst werden die Anforderungen einer solchen Beschriftung im Vergleich zur zweidimensionalen Beschriftung betrachtet. Der in Kapitel 3 definierte Konsistenzbegriff für 3D-Karten wird erweitert. Anschließend werden Ansätze für eine Beschriftung geliefert und schließlich das algorithmische Vorgehen der in dieser Diplomarbeit implementierten Verfahren aufgezeigt.

Kapitel 5 – Ergebnisse: Die Ergebnisse der implementierten Verfahren werden vorgestellt und verglichen.

Kapitel 6 – Mögliche Erweiterungen: Ein Überblick über mögliche Erweiterungen zu den in dieser Diplomarbeit vorgestellten Verfahren wird gegeben.

Kapitel 7 – Zusammenfassung: Dieses Kapitel fasst die Diplomarbeit zusammen.

2. Grundlagen

2.1. Geoinformationssysteme

Geoinformationssysteme (GIS) gewinnen in der Wissenschaft und in der Praxis immer mehr an Bedeutung z.B. bei der Routenplanung, Navigation, Unterstützung von politischen, ökonomischen und ökologischen Entscheidungen [Peto3].

Diese Systeme lassen sich nicht einheitlich definieren. Die folgenden Definitionen geben einen Überblick über die verschiedenen Ansichten eines GIS-Systems [BM98].

Definition 1 (Toolbox basierte Definition von GIS)

- *Eine mächtige Menge von Tools für die Sammlung, Speicherung, Abfrage nach Bedarf, Transformation und die Darstellung von räumlichen Daten aus der realen Welt [Bur86].*
- *Ein System zur Erfassung, Speicherung, Nachprüfung, Manipulation, Analyse und Darstellung von Daten, welche räumlich zur Erde bezogen werden [DoE87].*
- *Eine Informationstechnologie, welche sowohl räumliche als auch nicht-räumliche Daten speichert, analysiert und darstellt [Par88].*

Definition 2 (Datenbank Definition von GIS)

- *Ein Datenbanksystem, in dem die meisten der Daten räumlich indiziert sind und über das eine Menge von Methoden arbeiten, um Anfragen über die räumlichen Entitäten in der Datenbank zu beantworten [SMSE87].*
- *Jede manuelle oder Computer basierte Menge von Methoden, die zur Speicherung und Manipulation geografischbezogener Daten benutzt wird [Aro89].*

Definition 3 (Organisations-basierte Definition von GIS)

- Eine automatisierte Menge von Funktionen, die Fachleuten hoch entwickelte Möglichkeiten für Speicherung, Abruf, Manipulation und Darstellung von geografisch lokalisierten Daten zur Verfügung stellt [OSS81].
- Eine institutionelle Einheit, die eine organisatorische Struktur widerspiegelt, welche Datenbank-Technologien, Fachkenntnisse und kontinuierliche finanzielle Unterstützung umfasst [Car89].
- Ein Entscheidungsunterstützungssystem, das die Integration von räumlich bezogenen Daten in eine Problemlösungsumgebung beinhaltet [Cow88].

Im Folgenden wird ein Geoinformationssystem als eine Toolbox, wie es in Definition 1 beschrieben wird, betrachtet.

Die Interaktion mit dem Menschen spielt bei allen Aspekten, die mit dem geografischen Informationssystem zu tun haben, eine wichtige Rolle (siehe Abbildung 2.1).

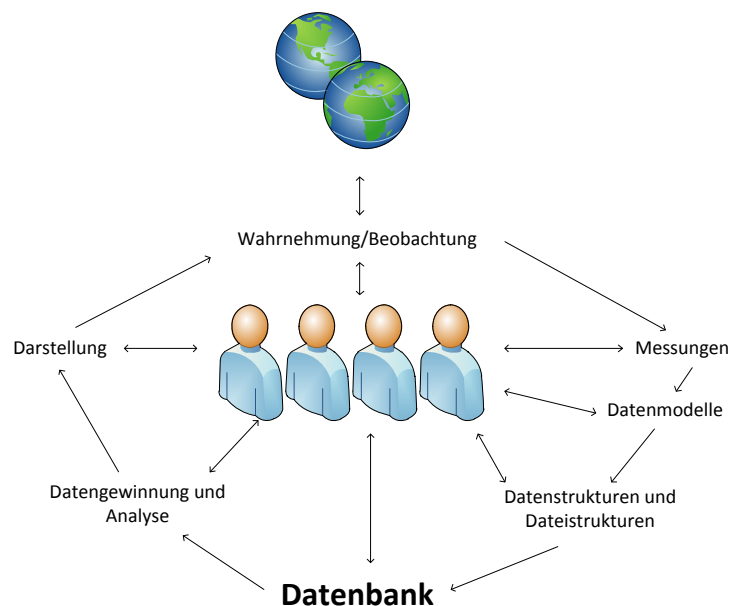


Abbildung 2.1.: Alle Aspekte von geografischen Informationssystemen sind mit der Interaktion von Menschen verknüpft [BM98].

Deshalb ist vor allem die Visualisierung von großer Bedeutung. Als visuelle Schnittstelle zwischen dem GIS und dem Nutzer bedient man sich dabei den Bildschirmkarten. Diese stellen Informationen aus der Datenbasis eines GIS in möglichst hoher kartografischer Qualität in Echtzeit dar und dienen als Interaktionsmedium. Dabei orientieren sich die qualitativen Erwartungen an den traditionellen Papierkarten, erweitert um die Anforderungen interaktiver Karten, die man zum Beispiel durch die Freiheitsgrade "Zoomen" oder "Verschieben" erhält [Peto3].

2.1.1. Messung der Erde: geografische Länge und Breite

Für Kartendarstellungen der Erde und deren Beschriftungen wird ein System benötigt, das eine sehr feine räumliche Auflösung besitzt und erlaubt, Abstände zwischen Ortspaaren zu berechnen sowie für andere Formen der Analyse geeignet ist. Dies bietet das System der geografischen Länge und Breite in einer umfassenden Weise. Das System basiert auf der Erdrotation und dem Schwerpunkt, welcher auf der Erdrotationsachse liegt. Die Ebene durch den Schwerpunkt, die senkrecht zu der Erdrotationsachse liegt, definiert den Äquator.

Die geografische Länge kann nun folgendermaßen definiert werden:

Definition 4 (geografische Länge)

Schnitte durch die Erde parallel zu der Achse der Erdrotation und senkrecht zu der Ebene des Äquators definieren Linien konstanter Länge. Der Schnitt durch das Royal Observatorium in Greenwich definiert die Länge Null. Jeder Winkel λ zwischen dieser und einer beliebigen anderen Scheibe definiert eine geografische Länge.

Die geografische Länge hat eine Ausdehnung von 180° westlich und 180° östlich (vgl. Abbildung 2.2). Im Computer werden geografische Längen als Dezimalgrad angegeben, wobei eine negative Zahl für die Ausdehnung in westlicher Richtung und eine positive Zahl für die Ausdehnung in östlicher Richtung steht.

Für die Definition der geografischen Breite muss die Form der Erde genauer betrachtet werden. Die Erde hat eine komplexe Form, die nur näherungsweise eine Kugel ist. Deshalb wird die Erde durch ein Ellipsoid angenähert, dessen Achse am Äquator länger als die an den Polen ist. Von den verschiedenen Varianten ist der WGS84 (World Geodetic System of 1984) der inzwischen am weitesten akzeptierte Ellipsoid. Die geografische Breite kann nun folgendermaßen definiert werden:

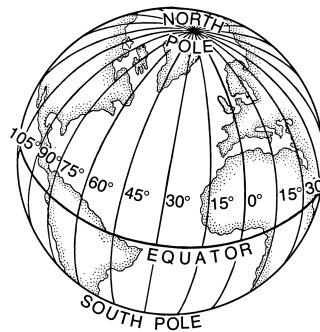


Abbildung 2.2.: Längengrade der Erde [Lae]

Definition 5 (geografische Breite)

Die geografische Breite eines Punktes auf der Erdoberfläche, ist der Winkel ϕ zwischen einer rechtwinklig zum Ellipsoid gezeichneten Linie durch diesen Punkt und der Ebene durch den Äquator.

Die geografische Breite variiert zwischen 90° Süd und 90° Nord (vgl. Abbildung 2.3). Der Südbreitengrad wird im Computer als negativer Dezimalgrad und der Nordbreitengrad als positiver Dezimalgrad dargestellt [LGMRO5].

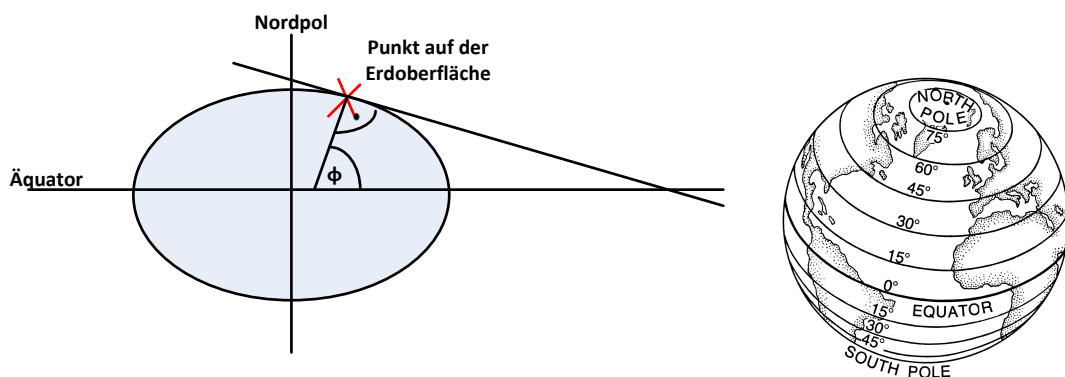


Abbildung 2.3.: Links: Der Breitengrad des mit einem roten Kreuz markierten Punktes auf der Oberfläche der Erde ist der Winkel zwischen der Tangente und der Ebene durch den Äquator. Rechts: Breitengrade der Erde [Bre]

2.1.2. Projektion und Koordinaten

Mit der geografischen Länge und Breite können Orte auf einem der Erde angenäherten Ellipsoid bestimmt werden. Viele Anwendungen benötigen allerdings eine zweidimensionale Darstellung, zum Beispiel für den Ausdruck der Karte auf Papier. Da die Menschen die Erde als flache Scheibe wahrnehmen wenn sie sich darauf bewegen, ist eine solche flache Karte ein wesentliches Hilfsmittel zur Orientierung. Um eine flache Darstellung der Erde zu erzielen, muss diese projiziert werden. Die Konsequenz einer solchen Projektion sind Verzerrungen.

Zunächst muss ein Koordinatensystem festgelegt werden. Das kartesische Koordinatensystem ist dafür gut geeignet. Es weist jedem Punkt Koordinaten auf einer Fläche zu, die durch zwei rechtwinklig zueinander stehenden Achsen definiert ist. In geografischen Anwendungen ist die y-Achse nach Norden ausgerichtet.

Eine Kartenprojektion kann man sich nun als eine Transformation einer Position auf der Erdoberfläche, die durch Breiten- und Längengrad (ϕ, λ) definiert ist, auf eine Position in Kartesischen Koordinaten (x, y) vorstellen.

Jede Kartenprojektion kann als mathematisches Funktionspaar dargestellt werden:

$$x = f(\phi, \lambda)$$

$$y = g(\phi, \lambda)$$

Zum Beispiel benutzt die berühmte Mercator-Projektion das Funktionspaar

$$\begin{aligned} x &= \lambda \\ y &= \ln \left(\tan \left(\frac{\phi}{2} + \frac{\pi}{4} \right) \right) \end{aligned}$$

wobei \ln die natürliche Logarithmus-Funktion ist.

Zwei Eigenschaften der Projektionen sind besonders wichtig, jedoch kann jede Projektion nur eine der beiden erreichen:

- Winkeltreue: stellt sicher, dass die Form von kleinen Eigenschaften auf der Erdoberfläche erhalten bleibt, d.h. dass die Skalierung der Projektion in der x- und y-Richtung immer gleich ist.
- Treue Flächen: stellt sicher, dass Flächen, die auf der Karte gemessen werden immer in der selben Proportion zu den Flächen stehen, die auf der Erdoberfläche gemessen werden.

2. Grundlagen

Die Eigenschaft der Winkeltreue ist nützlich für Navigationen, da die Flächenverzerrung in x- und y-Richtung gleich groß ist und Richtungen erhalten bleiben. Die Eigenschaft der gleichen Fläche ist nützlich für verschiedene Analyseformen, die Flächenberechnungen beinhalten.

Neben diesen Verzerrungseigenschaften ist eine weitere Art Kartenprojektionen zu klassifizieren die Ähnlichkeit zu einem physischen Modell, das beschreibt wie eine Position der Erdoberfläche auf eine Position der Kartenoberfläche abgebildet wird. Dabei werden drei Hauptklassen (vgl. Abbildung 2.4) unterschieden:

- Zylindrische Projektion: Wie in Abbildung 2.4 gezeigt, wird dabei ein Blatt Papier in Form eines Zylinders um die Erde gehüllt, die Eigenschaften der Erde darauf projiziert und das Papier anschließend wieder entrollt.
- Planare Projektion: Ein flaches Blatt Papier berührt die Erde.
- Kegelförmige Projektion: Ein Blatt Papier wird in einer Kegel Form um die Erde gehüllt.

Welche Art der Projektion benutzt wird, beeinflusst die Darstellung der Erde immens. So werden zum Beispiel Flächen, die das Blatt Papier berühren, gleich groß und Flächen, die in einiger Entfernung des Papiers liegen, größer projiziert als sie auf der Erde sind [LGMR05].

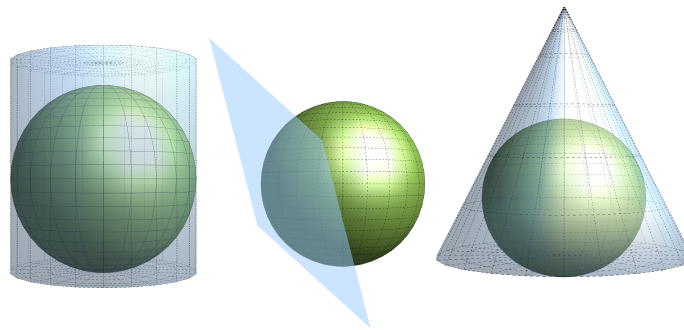


Abbildung 2.4.: Zylindrische, Planare und Kegelförmige Projektion. In jedem Fall wird die Positionen von Objekten auf der Erdoberfläche auf das Papier projiziert [LGMR05].

In dem dieser Diplomarbeit zu Grunde liegenden Terrainrenderer [Vaao8] wird die Universal Transversal Mercator Projektion (UTM) benutzt. Dabei handelt es sich um eine winkeltreue Zylinderprojektion, deren Nachteil allerdings eine starke Flächenverzerrung an den Polen ist [Lano6]. Dies spielt vor allem bei Wetterkarten eine große Rolle, wohingegen bei regionalen Karten die Flächenverzerrung sehr gering ist.

2.2. Georeferenzierung

Als Georeferenzierung wird die Verbindung von Positionen mit atomaren Informationen bezeichnet. Ein Beispiel dafür sind Postadressen. Die wichtigste Anforderung an die Georeferenzierung ist die Eindeutigkeit. Die Beschriftung ist eine Möglichkeit, diese Georeferenzierung für den Nutzer sichtbar und nutzbar zu machen [LGMR05].

Dabei müssen Beschriftungen so platziert werden, dass sie sich nicht überschneiden. In der Definition ist dieses sogenannte Beschriftungsproblem ("Map Labeling Problem") einfach:

Definition 6 (Beschriftungsproblem)

Gegeben: Menge von Punkten bzw. Features auf der projizierten Erde.

Gesucht: Eine Beschriftung für jeden Punkt, so dass sich keine zwei Beschriftungen gegenseitig überlappen.

Obwohl die Lösung dieses Problems in der Theorie einfach scheint, indem einfach alle Möglichkeiten durchnummeriert werden, macht es die kombinatorische Explosion dieses Ansatzes unpraktikabel für die Praxis. Viele verschiedene Varianten dieses Problems wurden als NP-vollständig bewiesen, sogar der eindimensionale Fall (Die NP-Vollständigkeit für den Fall von punkthaften Beschriftungen von Objekten wird in Abschnitt 2.2.2 umrissen). Es wurden deshalb Forschungen nach der besten Annäherung der optimalen Lösung betrieben [Moto7]. Eine $\mathcal{O}(n \log n)$ Lösung wird als sehr schnell angesehen, da viele Lösungen wesentlich schlechter sind [KT03].

Das allgemeine Beschriftungsproblem kann in verschiedene Kategorien in Abhängigkeit vom Typ Punkt, Linie oder Fläche des zu beschriftenden Interessensgebiets eingeteilt werden. Zusätzlich haben Forscher verschiedene Labeltypen in Bezug auf ihre Größe, ihre Form und ihrer Anordnung betrachtet. Zum Beispiel wurden Labels als Rechtecke, Kreise oder elastische Frames modelliert.

Selbst in dem einfachen Fall, dass ein Paar von achsenparallelen Rechtecken geschnitten werden soll, müssen mindestens vier atomare Operationen ausgeführt werden:

$$\begin{aligned} \text{SchnittpunktExistiert}(\text{rect}A, \text{rect}B) = & (\text{rect}A.\text{left} < \text{rect}B.\text{right}) \wedge \\ & (\text{rect}A.\text{right} > \text{rect}B.\text{left}) \wedge \\ & (\text{rect}A.\text{top} > \text{rect}B.\text{bottom}) \wedge \\ & (\text{rect}A.\text{bottom} < \text{rect}B.\text{top}) \end{aligned}$$

Die Überlegung, dass in einer Karte mehrere Millionen von Labels vorhanden sein können, zeigt, dass dies ein schwieriges Problem bleibt [Moto7].

Kann keine überlappungsfreie Beschriftung gefunden werden, müssen zu beschriftende Objekte entsprechend ihrer Wichtigkeit ausgewählt werden. Dazu werden Labelprioritäten eingeführt, damit kein höher priorisiertes Label ein niedriger priorisiertes verdeckt [Pet03].

2.2.1. Beschriftungen

Die zu beschriftenden Objekte können verschiedene Formen haben, welche man in den drei Kategorien punkthaft, flächenhaft und linienhaft zusammenfassen kann.

Objekte mit flächenhaft kleiner Ausdehnung werden als punkthaft bezeichnet. Das können zum Beispiel Städte, Inseln oder Interessenspunkte wie Tankstellen, Krankenhäuser oder Aussichtspunkte sein (vgl. Abbildung 2.5).



Abbildung 2.5.: Beispiele für die Beschriftung punkthafter Objekte: Städte und Dörfer; Gebäude; Inseln; Bergspitzen mit Höhenangaben [Pet96].

Die Beschriftung dieser Objekte erfolgt achsenparallel in einem gewissen Abstand um die Objekte herum. Der Abstand muss so gewählt werden, dass eine eindeutige Assoziation zwischen Label und zu beschriftendem Objekt möglich ist. In der Literatur werden die Positionen einer solchen Beschriftung meistens auf diskrete Positionen beschränkt. Es gibt aber auch Verfahren wie das von Petzold [Pet03], das ein Kontinuum an Beschriftungspositionen erlaubt [Pet03]. Um zu entscheiden, welche Labelposition gewählt werden soll, kann eine Bewertungsfunktion eingeführt werden. In der Literatur werden viele verschiedene Ansätze beschrieben, die meisten basieren auf den Richtlinien von Imhof [Imh62] (vgl. Abbildung 2.6):

- Label-Positionen auf der rechten Seite werden bevorzugt gegenüber den Positionen auf der Linken.
- Labels über dem zu beschriftenden Punkt werden bevorzugt vor den Labels unterhalb.

- Je weiter die Labelgrundlinie von einer horizontalen Linie durch den Mittelpunkt des assoziierten Punktes entfernt ist, desto weniger bevorzugt ist es [Imh62].

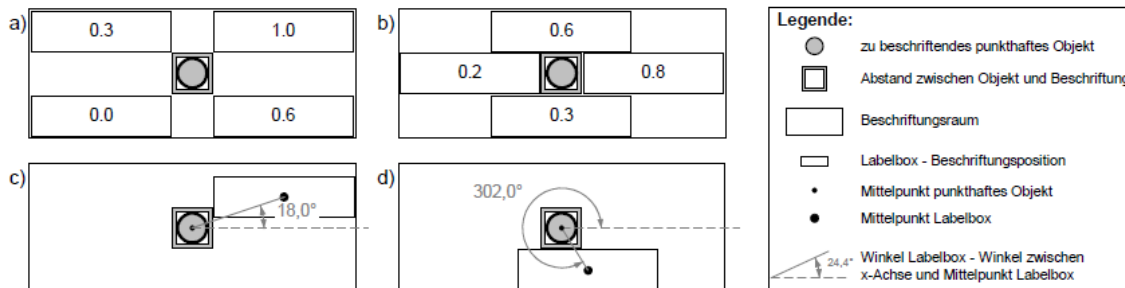


Abbildung 2.6.: In a) und b) werden diskrete Beschriftungspositionen mit den Werten zwischen 0.0 (schlecht) und 1.0 (sehr gut) dargestellt. Die Bewertung von kontinuierlichen Beschriftungen, wie sie in c) und d) gezeigt werden, basiert auf dem Winkel zwischen einer Linie horizontal zum Mittelpunkt des zu beschriftenden Objektes und dem Mittelpunkt des Labels [Peto3].

Linienhaft werden Objekte bezeichnet, die aus einer Kombination von Linien und Kreis-segmenten bestehen und sich durch Linienzüge approximieren lassen. Beispiele für Lini-enobjekte sind Straßen, Flüsse oder Eisenbahnstrecken [Imh62]. Eine Bewertungsfunktion für die Position der Beschriftung ist im Vergleich zu den punkthaften Objekten schwieriger und nicht eindeutig. Um eine solche Bewertungsfunktion zu erhalten, müssen gewichtete Kriterien zusammengesetzt werden [Peto3].

Weitere Details zur Beschriftung linienhafter Objekte liefern Imhof, E. [Imh62] und Ellsiepen [Ello1].

Beispiele von existierenden Verfahren für die Beschriftung linienhafter Objekte findet man in [ECMS96], [Kni98] und [Ello1].

Das Verfahren von Edmonson et al. [ECMS96] platziert entlang des zu beschriftenden Objekt Labels, bewertet jede Position und wählt die beste Bewertung aus. Die Verfahren von Knipping L. [Kni98] und Ellsiepen [Ello1] benutzen Splines (Kurvensegmente), wodurch Knicke vermieden werden und ein harmonischer Verlauf der Schrift sichergestellt wird (vgl. Abbildung 2.7).

Bei flächenhaften Objekten unterscheidet man zwischen zwei Fällen [Imh62]:

- Die Objekte besitzen eine explizite Grenzlinie die ihre Form bestimmt, wie zum Beispiel Länder oder Seen.
- Die Objekte besitzen nur implizit eine Grenze, wie zum Beispiel Regionen oder Gebirgsketten.



Abbildung 2.7.: a) zeigt die Beschriftung eines Flusses [Peto3]. In b) wird das Verfahren von Edmonson [ECMS96] dargestellt. Die ausgewählte Position ist rot markiert. In c) wird das Verfahren von Ellsiepen [Ello1] und Knipping [Kni98] abgebildet.

Im ersten Fall kann die Beschriftung horizontal erfolgen oder die geometrische Form des Objektes widerspiegeln. Im zweiten Fall ist keine Grenzlinie erkennbar. Das Label muss die Form widerspiegeln, um die Ausdehnung des Objektes sichtbar zu machen (vgl. Abbildung 2.8).

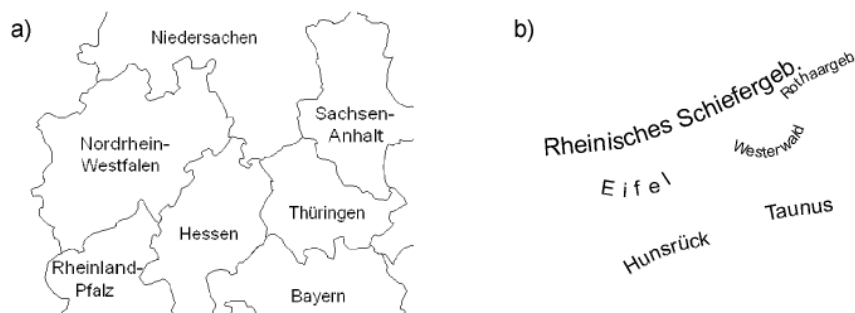


Abbildung 2.8.: In a) haben die zu beschriftenden Objekte eine Grenzlinie und werden achsenparallel beschriftet. In b) spiegeln die Beschriftungen die Form des Objektes wieder [Peto3], [Kni98].

Weitere Details findet man in [Imh62] und [Ello1]. Verfahren zur Beschriftung flächenhafter Objekte, die die Geometrie des Objektes widerspiegeln, findet man [Pet96] und [PP97]. Achsenparallele Beschriftung werden in [DPP03] und [Döo1] behandelt.

2.2.2. NP-Vollständigkeit des Beschriftungsproblems

Das Beschriftungsproblem für die Beschriftung punkthafter Objekte ("admissible point-feature placement (SLP)") ist folgendermaßen definiert:

Definition 7 (Beschriftungsproblem für die Beschriftung punkthafter Objekte)

Gegeben: Menge von Objekten, die zweidimensional mit je vier möglichen Positionen punkthaft beschriftet werden soll.

Gesucht: Gibt es eine punkthafte Beschriftung dieser Objekte, so dass keine Überlappung oder Verdeckung entsteht [JS93]?

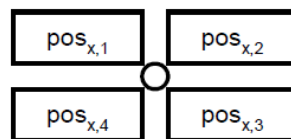


Abbildung 2.9.: Die vier Beschriftungspositionen eines punkthaftern Objektes [Peto3]

Satz 8 (NP-Vollständigkeit)

Das Beschriftungsproblem eingeschränkt auf überlappungsfreie Beschriftung punkthafter Objekte ist NP-vollständig (verursacht eine exponentielle Laufzeit) [JS93].

Technisch erfolgt der Nachweis ob ein Problem NP-vollständig ist, durch eine Reduktion auf ein bekanntes und bewiesenes NP-vollständiges Problem [Man89], [CS01].

Im Beweis von Marks et al. [JS93] wird das Beschriftungsproblem bei Beschriftungen punktbasierter Objekte auf das planare Erfüllbarkeitsproblem (3-SAT-satisfiability-problem) reduziert.

Definition 9 (planares 3-SAT) Das 3-SAT-Problem ist die Frage, ob es zu einer gegebenen aussagenlogischen Formel F mit genau drei Variablen, eine Belegung dieser mit den Werten wahr oder falsch gibt, so dass der ganze Ausdruck den Wert wahr annimmt. Eine Formel ist planar, wenn der daraus folgende bipartite Graph planar ist. Dies ist ein Graph (V,E) , dessen Knotenmenge V aus allen Klauseln $c_i \in F$ und allen Variablen $x_i \in F$ und dessen Kantenmenge folgendermaßen definiert ist: Für jede Klausel c in F und für jede Variable x gibt es eine Kante in E , welche die Knoten c und x genau dann verbindet, wenn die zu c_i gehörende Klausel ein positives oder negatives Vorkommen der Variablen x_i hat.

2. Grundlagen

Ein Algorithmus, der alle Kombinationen von Belegungen überprüft um festzustellen, ob der Ausdruck erfüllbar ist, liegt in $\mathcal{O}(2^n)$ (n = Anzahl der Variablen des Ausdrucks) [CS01].

Die Konstruktion des Beweises, dass das Beschriftungsproblem in der Klasse NP liegt, wird im Folgenden anhand eines Beispiels beschrieben. In Abbildung 2.10 wird das Ziel, eine aussagenlogische Formel zu konstruieren, die genau dann wahr wird, wenn sich die entsprechend ausgewählten Beschriftungen nicht überschneiden dargestellt.

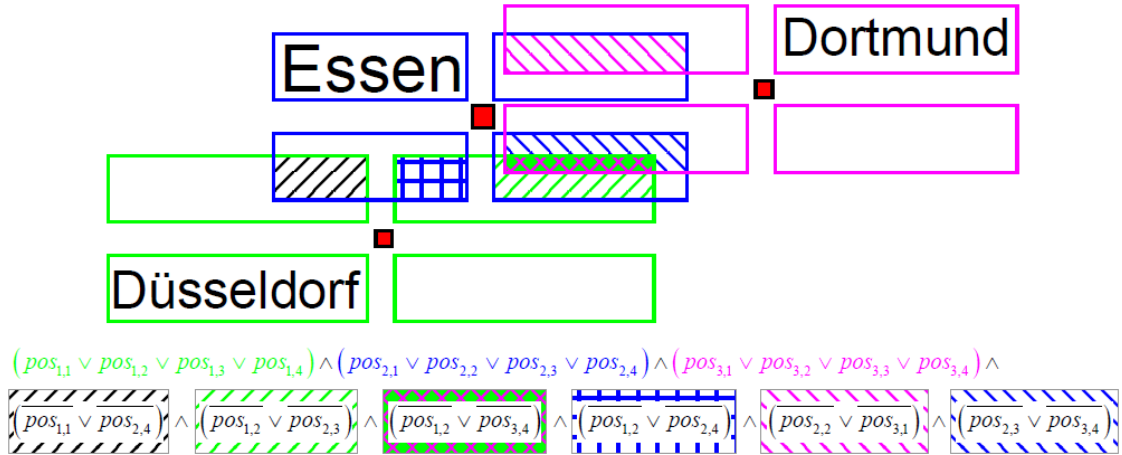


Abbildung 2.10.: Entwicklung eines aussagenlogischen Ausdrucks für das Beschriftungsproblem für punkthafte Beschriftungen am Beispiel der Beschriftungen der Städte Düsseldorf, Essen und Dortmund [Peto3].

Für jedes zu beschriftende Objekt $x \in \{1 = \text{Düsseldorf}, 2 = \text{Essen}, 3 = \text{Dortmund}\}$ wird eine Disjunktion aufgestellt,

$$(pos_{x,1} \vee pos_{x,2} \vee pos_{x,3} \vee pos_{x,4})$$

da mindestens eine der Positionen ausgewählt sein muss, damit das Objekt beschriftet wird. Die Disjunktions-Terme werden durch Konjunktionen miteinander verknüpft (vgl. Abbildung 2.10 erste Zeile der aussagenlogischen Formel). Dadurch wird sichergestellt, dass jedes zu beschriftende Objekt eine Beschriftung erhält.

Im nächsten Schritt werden alle Konflikte ermittelt und als aussagenlogische Terme dargestellt. So stehen zum Beispiel die erste Position von Düsseldorf und die vierte Position von Essen im Konflikt. Es dürfen also nicht beide gleichzeitig ausgewählt werden

$$\overline{(pos_{1,1} \wedge pos_{2,4})} = \overline{pos_{1,1}} \vee \overline{pos_{2,4}}$$

Jeder Konflikt zwischen zwei Beschriftungen wird, wie in diesem Fall, durch eine Disjunktion der negierten Positionsausdrücke dargestellt und mit einer Konjunktion an die

aussagenlogische Formel angehängt (vgl. Abbildung 2.10 zweite Zeile der aussagenlogischen Formel).

Zur Prüfung, ob eine konfliktfreie Beschriftung möglich ist, müssen im Worst-Case-Fall alle Kombinationen von Belegungen getestet werden. Dies führt wie oben bereits beschrieben zu einer exponentiellen Laufzeit. In diesem Fall, bei dem vier mögliche Beschriftungspositionen betrachtet werden, liegt die Laufzeit in $\mathcal{O}(4^n)$.

Eine mögliche konfliktfreie Beschriftungslösung für das Beispiel in Abbildung 2.10 ist

$$\begin{aligned} pos_{1,2} &= pos_{2,2} = pos_{3,2} = \text{wahr} \\ pos_{1,1} &= pos_{1,3} = pos_{1,4} = pos_{2,1} = pos_{2,3} = pos_{2,4} = pos_{3,1} = pos_{3,3} = pos_{3,4} = \text{falsch} \end{aligned}$$

Dagegen liefert die folgende Belegung der aussagenlogischen Formel falsch, da es zu einem Konflikt zwischen Beschriftungspositionen von Essen und Düsseldorf kommt

$$\begin{aligned} pos_{1,2} &= pos_{2,3} = pos_{3,1} = \text{wahr} \\ pos_{1,1} &= pos_{1,3} = pos_{1,4} = pos_{2,1} = pos_{2,2} = pos_{2,4} = pos_{3,2} = pos_{3,3} = pos_{3,4} = \text{falsch} \end{aligned}$$

Wird die Anzahl der Beschriftungspositionen pro zu beschriftendem Objekt auf zwei beschränkt, reduziert sich das Problem auf eine $\mathcal{O}(n \log^2 n)$ Laufzeit und ist somit in polynomineller Zeit lösbar [FW91].

2.2.3. OpenStreetMap

Um Labels auf einer Karte zu platzieren, werden Informationen über die Lage der zu beschriftenden Objekte benötigt. OpenStreetMap (OMS) [Ope] ist ein Projekt, welches eine Sammlung solcher räumlicher Daten bereitstellt. Das Ziel dieses Projektes ist, eine umfassende Weltkarte zu erschaffen. So liefert die Datensammlung Informationen über Straßen, Eisenbahnstrecken, Flüsse, Wälder, Häuser und vieles mehr. Das Projekt wurde im Jahr 2004 ins Leben gerufen. Der Vorteil dieser Datensammlung gegenüber anderen wie zum Beispiel GoogleMaps [Goob] liegt in der Lizenzkostenfreiheit und beliebigen Weiterverarbeitungsmöglichkeit. So ist zum Beispiel die Nutzung nicht an eine bestimmte Programmierschnittstelle ("Application Programming Interface (API)") geknüpft. Dabei stellt OpenStreetMap nicht nur eine Karte zur Verfügung, sondern auch die "rohen" Daten. Dadurch ist eine Vielzahl von Einsatzmöglichkeiten denkbar: vom einfachen Einbinden der Karten in eine Webseite bis hin zu komplexen Datenbankverbindungen (vgl. Abbildung 2.11).

Erstellt wird die Datenbank durch freiwillige Nutzer, die diese Geodaten sammeln und dem Projekt zur Verfügung stellen. Der Nachteil daran ist, dass keine Vollständigkeit der Daten garantiert wird. Allerdings gewinnen die Karten durch die immer größer werdende

2. Grundlagen

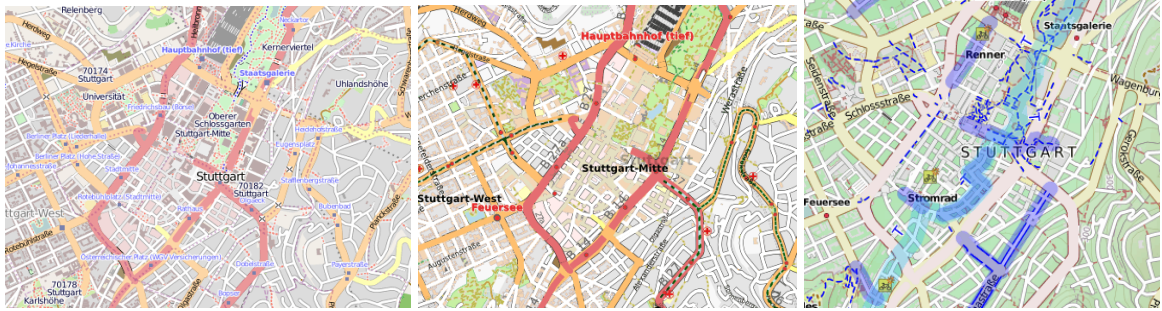


Abbildung 2.11.: Die Verwendung der OpenStreetMap-Daten bleibt dem Benutzer überlassen. So können zum Beispiel verschiedene Renderer benutzt werden (Links: Mapnik, Mitte: Osmarenderer) oder Karten für bestimmte Zwecke erstellt werden wie die rechts dargestellte Fahrradkarte [Ope].

Community (am 11. September 2010 waren es 300.000 Mitglieder) täglich an Umfang und haben jetzt teilweise schon eine größere Genauigkeit als GoogleMaps-Karten [Goob]. So ist zum Beispiel die USA sehr gut erfasst, da amerikanische Behörden sämtliche Geodaten gestiftet haben. Das gleiche gilt für die Niederlande, wo Daten des Navigationsherstellers AND genutzt werden konnten. In ländlichen Gebieten Ostdeutschlands dagegen muss noch viel erschlossen werden [Rico9]. Die Daten von OpenStreetMap können vom OSM-Server abgefragt werden. Wie diese Daten dann weiterverarbeitet werden, bleibt dem Nutzer offen. So können die Daten gleich online weiterverarbeitet oder zunächst gespeichert und später offline benutzt werden (vgl. Abbildung 2.11).

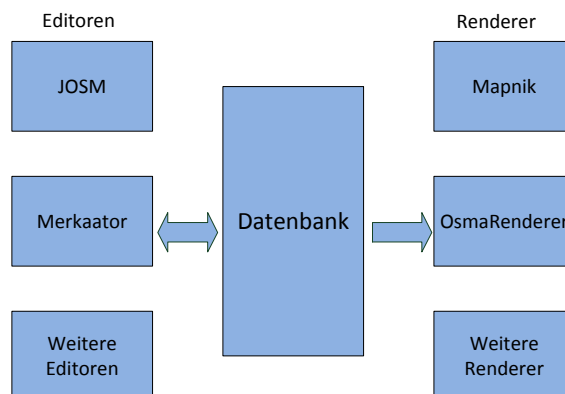


Abbildung 2.12.: Der Kern des OpenStreetMap-Projekts ist die Datenbank die mittels Editoren wie JOSM oder Merkaator bearbeitet werden können. Mittels Renderer wie Mapnik oder Osmarenderer können die Daten angezeigt werden.

Die Daten liegen dabei in einem auf dem "Geographic Data Files"(GDF) basierenden Datenformat vor. Die Grundelemente sind *Nodes*, *Ways* und *Relations*.

- *Node*: Ein *Node* ist ein geografischer Punkt, der durch Längen- und Breitengrad definiert ist. Er wird sowohl für die Definition von *ways* benötigt sowie auch als ein Punktfeature wie zum Beispiel eine Telefonzelle oder ein anderer Interessenpunkt (engl. point of interest (POI)).
- *Way*: Ein *Way* ist eine Kopplung von mindestens 2 und maximal 2000 *Nodes* die ein lineares Feature wie zum Beispiel eine Straße beschreiben.
- *Relations*: Eine *Relation* gruppiert andere Elemente wie *Nodes*, *Ways* oder andere *Relations*.

Durch sogenannte *tags* können, wie folgendes Beispiel zeigt, *nodes*, *ways* und *relations* semantische Eigenschaften zugeordnet werden, so dass die Daten in einer aussagekräftigen Weise interpretiert werden können [Ope].

```
<node>
  <tag k='highway' v='motorway_junction' />
  <tag k='name' v='Dreieck Leonberg' />
</node>
```

2.2.4. Vorgehensweise bei der automatischen Beschriftung

Der typische Ablauf bei der automatischen Beschriftung wird in Abbildung 2.13 dargestellt (vgl. [Chr95], [ECMS96], [PP97], [Coo88]).

Zunächst werden diskrete Beschriftungspositionen je nach Objekttyp (punkthaft, linienhaft, flächenhaft) erzeugt. Diese Beschriftungspositionen und der Informationsverlust der durch die Verdeckung von Kartenobjekten, die keine Beschriftungen sind, entsteht, werden bewertet.

In dynamischen Karten ist die Anzahl der zu zeichnenden Beschriftungen oft nur ein Bruchteil der Gesamtzahl der Labels. Deshalb wird ein Filterungsschritt benötigt, der die Anzahl der Beschriftungen reduziert. Dieser Filterungsschritt kann beliebig an die Aufgabe, welche die interaktive Karte erfüllen soll, angepasst werden. So können zum Beispiel nur die Labels gezeichnet werden, die den aktuellen Sichtbereich schneiden oder man verzichtet auf alle lokalen Straßennamen, wenn man weit genug herauszoomt [BDY06].

Im nächsten Schritt muss ein Algorithmus gewählt werden, der zu jedem Objekt eine Beschriftungsposition auswählt. Dies ist ein Optimierungsproblem, das folgende Bedingungen erfüllen muss:

- Keine Überlappung von ausgewählten Beschriftungspositionen

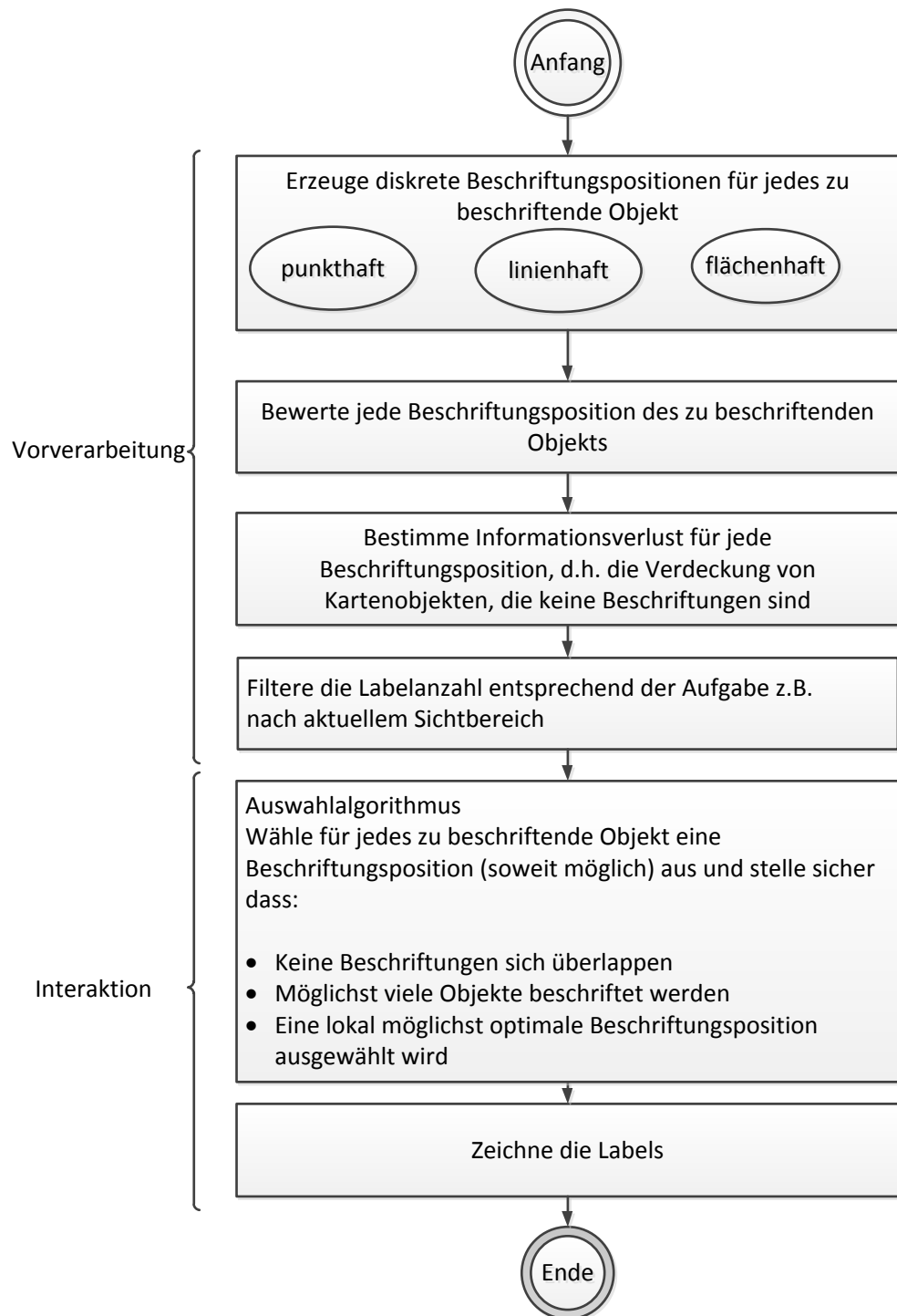


Abbildung 2.13.: Vorgehensweise bei der automatischen Textplatzierung (vgl. [Peto3]).

- Beschriftung möglichst vieler Objekte
- Wahl einer lokal optimalen Beschriftungsposition

Die erzielten Ergebnisse sind stark vom Auswahlalgorithmus und den gewählten Bewertungskriterien abhängig. Um eine kürzere Laufzeit zu erzielen werden oft auch Heuristiken eingesetzt, die ein suboptimales aber akzeptables Ergebnis erzielen [Pet03].

2.2.4.1. Auswahlalgorithmen

Die Forschung über das automatische Platzieren von Labels zieht sich über Jahrzehnte hinweg. Einen guten Überblick über die Literatur bietet die "Map Labeling Bibliography" [Wol09]. Wissenschaftler aus vielen verschiedenen Bereichen wie zum Beispiel Kartografie, "computational geometry" oder geografische Informationssysteme sind die prinzipiellen Entwickler. Deshalb findet man ein breites Spektrum an Verfahren: von Greedy, Backtracking und vollständig Regel basierten Ansätzen hin zu Divide and Conquer, Gradientenverfahren, Simulated Annealing, genetische Algorithmen, lineare Programmiertechniken, Tabu-Suche, Ameisenkolonien und viele mehr [Moto7].

Die Algorithmen können in zwei Gruppen eingeteilt werden:

1. Algorithmen, die ein globales Minimum an Überschneidungen von Beschriftungen suchen
2. Algorithmen, die das globale Minimum mit Hilfe von lokalen Minima und Heuristiken annähern

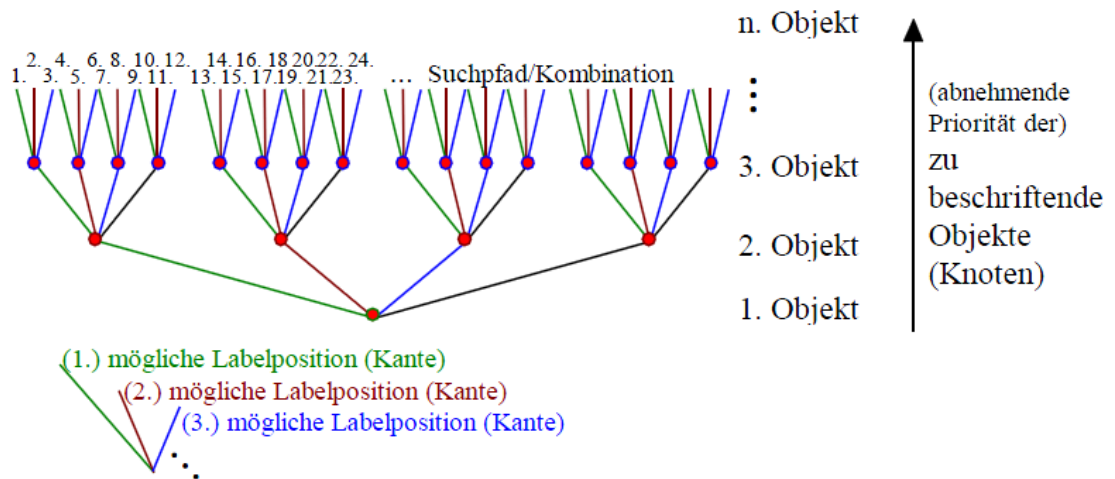
Der Vorteil der zweiten Gruppe ist eine wesentlich bessere Laufzeit im Vergleich zur ersten, allerdings auf Kosten der Optimalität [Pet96].

Im nächsten Abschnitt werden die wichtigsten Vertreter exemplarisch kurz umrissen [Pet03]. Eine umfassende Übersicht und einen Vergleich der Auswahlalgorithmen findet man in [Car87], [Chr95], [PP97] und [Wol99].

Backtracking

Als Vertreter der ersten Klasse ist *Backtracking* zu nennen. Dabei können zwei Varianten unterschieden werden:

- Alle möglichen Label-Kombinationen werden getestet und bewertet. Die Kombination mit dem besten Ergebnis wird ausgewählt.
- Es wird nach der ersten überschneidungsfreien Platzierung gesucht.



Algorithmus 2.1 Gradientenabstieg

Jedes zu beschriftende Objekt wird zufällig an einer seiner möglichen Labelposition beschriftet

while keine Verbesserung mehr oder Obergrenze an Schleifendurchläufen erreicht **do**
 Teste für das aktuelle zu beschriftende Objekt alle alternativen Beschriftungspositionen.
 Berechne das neue Gesamtbewertungsergebnis für alle möglichen Repositionierungen.
 Repositioniere das Label an die Position, die die größte Verbesserung bringt.
 Untersuche ein zu beschriftendes Objekt, das einen Konflikt mit der alten oder neuen Positionierung hat.

end while

Mit diesem Verfahren ist es möglich, dass man in einem lokalen Minimum stecken bleibt und kein globales Minimum an Überschneidungen findet. Abbildung 2.15 zeigt einen Fall, bei dem erst eine Verschlechterung in Kauf genommen werden muss, um von einem lokalen Minimum zu einem globalen Minimum zu kommen [Pet96].

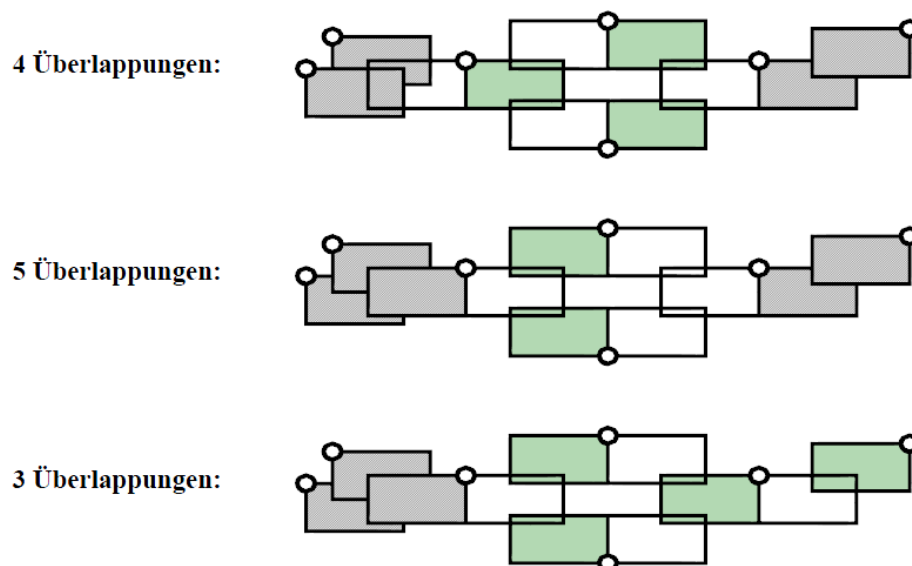


Abbildung 2.15.: Die schwarzen Rahmen kennzeichnen mögliche Labelpositionen. Aktuell ausgewählte Beschriftungspositionen sind farblich hervorgehoben. Die Schraffur macht Überlappungen deutlich. Um die 4 ursprünglichen Überlappungen zu reduzieren, muss zunächst eine Verschlechterung auf 5 Überlappungen in Kauf genommen werden, um das globale Minimum von 3 Überlappungen erreichen zu können (vgl. [Pet96]).

Simulated Annealing

Um das Problem des Steckenbleibens in lokalen Minima zu begegnen, wird der Ansatz *simulated annealing* verwendet. Dies ist ein stochastischer Optimierungsalgorithmus, dessen Grundidee aus der statistischen Physik kommt. Bei dem Verfahren werden Verschlechterungen der Gesamtbewertung mit einer gewissen Wahrscheinlichkeit akzeptiert. Mit zunehmender Zeit wird diese Akzeptanz-Wahrscheinlichkeit gesenkt und damit auch der Grad der akzeptierten Verschlechterung.

Algorithmus 2.2 Simulated Annealing

```
Setze für jedes zu beschriftende Objekt eine beliebige Beschriftungsposition
while  $\neq$  Abbruchkriterium erreicht do
  Nimm ein beliebiges zu beschriftendes Objekt.
  Platziere das Label an beliebiger Stelle neu.
  Berechne die Änderung der Gesamtbewertung.
  if Verbesserung then
    Führe Änderung aus.
  else
    Ändere mit bestimmter Akzeptanzwahrscheinlichkeit.
  end if
end while
```

In [Chr95], [CMS92], [CMS95] und [PP97] wird das Verfahren des simulated annealing umgesetzt. In Bezug auf Laufzeit und Beschriftungsqualität können damit gute Ergebnisse erzielt werden [Peto3].

Lineare Programmierung

Bei der Methode der *Linearen Programmierung* wie sie in [Zor86] und [Kre94] Anwendung findet, wird versucht das Minimierungsproblem in Gleichungen zu fassen. K ist die Menge der zu beschriftenden Objekte:

$\forall k \in K : \exists N_k$ mit $N_k =$ Anzahl der möglichen Beschriftungspositionen für das Objekt k

$\forall k \in K, (1 \leq i \leq N_k) : X_{i,k} = 0$ die i -te Labelposition ist nicht aktiviert

$X_{i,k} = 1$ die i -te Labelposition ist aktiviert

unter der Bedingung

$$\sum_{i=1}^{N_k} X_{i,k} = 1 \quad k = 1, 2, \dots, |K|$$

d.h. für jedes Objekt muss genau ein Label aktiviert sein

Falls sich zwei mögliche Labelpositionen (i und j) für verschiedene Objekte (k und m) überlappen, dann gilt die Ausschlussklausel:

$$X_{i,k} + X_{j,m} \leq 1$$

Hinzu kommt die Bewertung der einzelnen Beschriftungspositionen: $0 \leq W_{i,k}$. Die Bewertung 0 steht für die optimale Beschriftungsposition. Alle anderen Positionen erhalten eine positive Bewertung > 0 , je nach Qualität.

Somit ergibt sich das folgende Minimierungsproblem

$$\sum_{k=1}^K \sum_{i=1}^{N_k} W_{i,k} X_{i,k}$$

Dieses Minimierungsproblem bildet mit den obigen Nebenbedingungen ein ganzzahliges lineares Programm und ist somit NP-vollständig. Die Ganzzahligkeit kommt daher, dass eine Beschriftungsposition entweder 1 = ausgewählt oder 0 = nicht ausgewählt ist. Zur Lösung können typische Algorithmen aus der linearen Programmierung verwendet werden wie zum Beispiel der Simplex-Algorithmus. Dieser ist im Worst-Case Fall exponentiell [Pet96].

Verfahren von Hirsch

Das Verfahren von *Hirsch*, [Hir82], beschriftet ein punkthaftes Objekt frei entlang eines Abstandskreises um das Objekt herum (vgl. Abbildung 2.16). Dabei behalten Beschriftungen immer einen konstanten Abstand zum Objekt und werden immer waagrecht gezeichnet. Kommt es zu Überschneidungen mit anderen Beschriftungen, werden Überlappungs- bzw. resultierenden Vektoren gebildet. Dabei wird das Konzept der magnetischen Polabstoßung aus der Physik adaptiert. Abbildung 2.16 zeigt, wie sich Labels von der Schnittstelle entlang des resultierenden Vektors unter Berücksichtigung des konstanten Abstands vom zu beschriftenden Objekt weg bewegen [Pet96].

Vergleich der vorgestellten Verfahren

Christensen et al. [CMS92] haben die hier vorgestellten Verfahren *Greedy*, *Gradientenabstieg*, *Hirsch*, *Lineare Programmierung* und *Simulated Annealing* im Hinblick auf Beschriftungsqualität und Laufzeiten verglichen. Abbildung 2.17 zeigt das Ergebnis des Tests, n Punktobjekte ($n = 50, 100, 150, \dots, 1500$) mit einer Labelgröße von 30×7 Einheiten in einem Raster der Größe 792×612 Einheiten zu beschriften. Für jede Problemgröße werden 25 Beschriftungen erzeugt. Die Bewertung ergibt sich aus der durchschnittlichen Anzahl der Labels, die ohne Überlappung gezeichnet wurden, im Verhältnis zur Gesamtzahl.

Simuliert wurde dabei einmal mit und einmal ohne Labelselektion. Labelselektion bedeutet hier, dass zu beschriftende Punkte weggelassen werden können, die viele Konflikte mit

2. Grundlagen

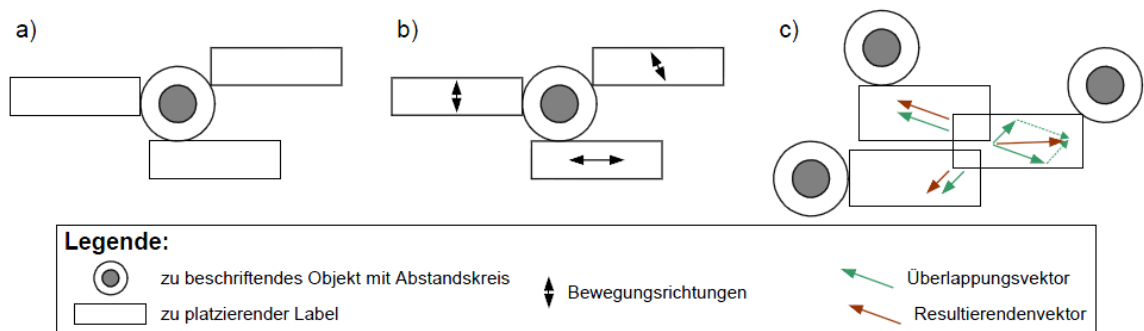


Abbildung 2.16.: a) zeigt mögliche Beschriftungspositionen. In b) werden die Bewegungsrichtungen dargestellt. c) zeigt die Auflösung eines Schnittes zwischen den Labelpositionen (vgl. [Pet03]).

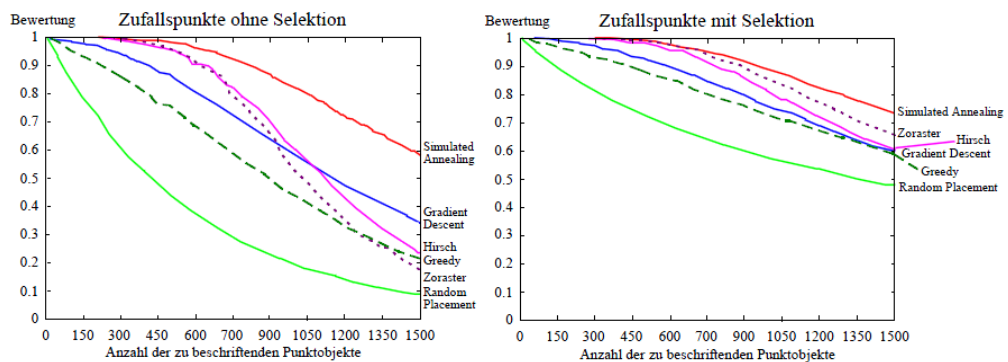


Abbildung 2.17.: Resultate der verschiedenen Auswahlalgorithmen mit zufällig generierten Karten. Mit Punktselektion rechts und ohne Punktselektion links (vgl. [Pet96]).

anderen Beschriftungen erzeugen oder eine sehr geringe Wichtigkeit besitzen. Das beste Ergebnis liefert Simulated Annealing.

Für dynamische Beschriftungen ist nicht nur die Qualität von Wichtigkeit, sondern auch die Laufzeit. Das Ergebnis der Laufzeiten bei 750 bzw. 1500 Punktbeschriftungen auf einer DEC 3000/400 AXP-Workstation zeigt Abbildung 2.18.

Abbildung 2.19 zeigt die Karten, die nach den verschiedenen Auswahlalgorithmen entstehen, für 750 Punktoberfläche.

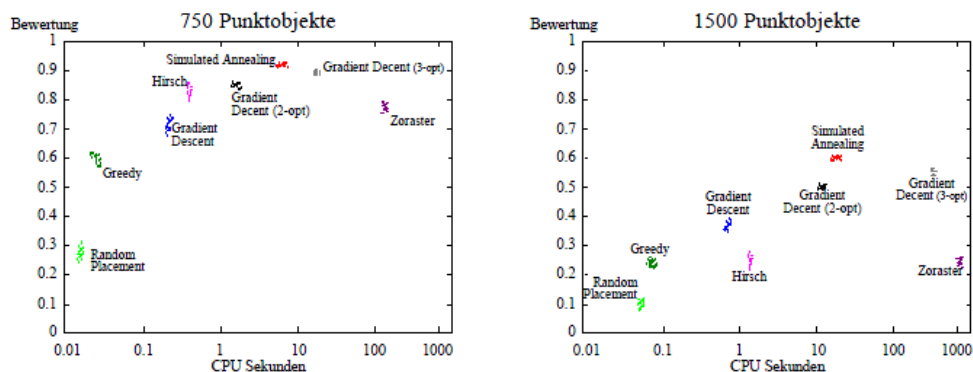


Abbildung 2.18.: Laufzeiten von 10 Testläufen mit 750 Punktoobjekten (links) und 1500 Punktoobjekten (rechts) (vgl. [Pet96]).

2.2.4.2. Schnelles Labeling

Die Geschwindigkeit wird aber gerade im Zeitalter interaktiver Kartendarstellungen immer wichtiger. Vor ein paar Jahren lag die beste Lösung in der wissenschaftlichen Literatur in der Größenordnung von Sekunden oder Minuten, was zwar für eine statische Karte akzeptabel, für eine dynamische Karte mit Zoom- und Schiebefunktionen aber zu langsam ist. In solchen Anwendungen muss die Position eines Labels bei jeder Änderung der Skalierung oder des Geltungsbereichs neu berechnet werden, das heißt der Algorithmus muss so schnell sein, dass er in "Echtzeit" (zum Beispiel mit der Bewegung des Mausekursors) läuft [Moto7].

Dr. Herbert Freeman, bietet eine exzellente Übersicht der letzten 25 Jahre des Fortschritts in [Fre05]. Er schließt seinen Überblick mit dem Versprechen ab, der Tag, an dem die Beschriftung digitaler Karten in einer Sekunde erfolgt, sei nicht mehr weit entfernt [Fre05].

Und tatsächlich gab es in den letzten Jahren einige Durchbrüche, um schnellere Algorithmen für das Karten-Beschriftungsproblem zur Verfügung zu stellen.

Wagner et al. [WWKS01] waren eine der ersten, die einen schnelleren Ansatz bereitstellten, indem sie verschiedene Heuristiken für eine signifikante Verbesserung gegenüber vorherigen Algorithmen einsetzten. Der Algorithmus läuft in zwei Phasen: in der ersten Phase wird eine Menge von Regeln auf alle zu beschriftenden Objekte angewendet, um möglichst viele Objekte beschriften zu können und die möglichen Beschriftungspositionen der Konfliktpartner zu reduzieren. In der zweiten Phase wird heuristisch die Anzahl der möglichen Beschriftungspositionen auf eine Beschriftung pro zu beschriftendes Objekt reduziert. Dieser Ansatz ist für große Datensätze mit einer Laufzeit von mehreren Minuten immer noch zu langsam.

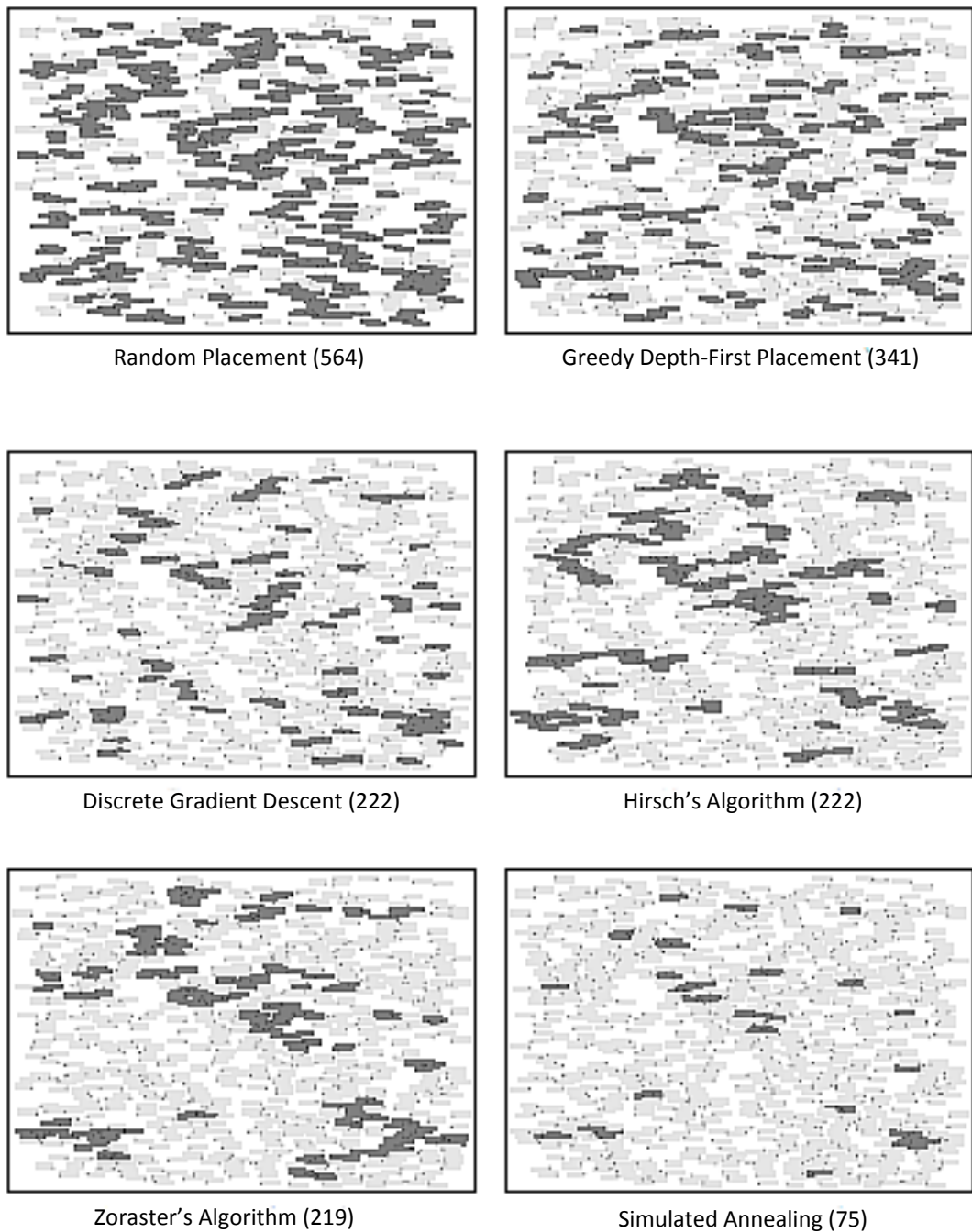


Abbildung 2.19.: Beispielkarten der verschiedenen Auswahlalgorithmen mit 750 Punkto-
jekten. Hellgraue Label sind frei von Überlappungen. Dunkelgraue Label
überlappen andere Label. Die Anzahl in Klammern gibt die Zahl der nicht
überlappungsfrei beschriftbaren Punktoobjekte wieder (vgl. [Pet96]).

Ein anderer Ansatz liefert Heber [Heb98] und Petzold et al. [PPH99]. Die wesentliche Eigenschaft dieses Ansatzes ist die Unterteilung des Beschriftungsvorgangs in eine Vorverarbeitungs- und Interaktionsphase (vgl. Abbildung 2.13). Die rechenintensive Überlappungsberechnung wird in die Vorverarbeitungsphase verschoben. Dort wird ein Konfliktgraph als Speicher- und Zugriffsstruktur aufgebaut. In der Interaktionsphase kann einer der zuvor beschriebenen Auswahlalgorithmen wie Greedy oder Simulated Annealing auf den vorher aufgebauten Konfliktgraphen adaptiert werden. Dies führt zu einer Verkürzung der Laufzeit in der Interaktionsphase. Allerdings sind die Vorverarbeitungskosten sehr hoch [Pet03].

In einem sich darauf beziehenden Artikel von Harrie et al. [HSKL05] wird ein Algorithmus präsentiert, der für die meisten Echtzeit-Kartenanwendungen tatsächlich schnell genug ist. Jedoch stellt der Autor diese Fähigkeit nur für Daten mit etwa 100 Punkten bereit. Der Algorithmus platziert die Beschriftungen in einem Gebiet, in dem so wenig kartografische Daten wie möglich verdeckt werden. Um dieses Gebiet zu finden, verwendet der Algorithmus eine Spiralsuche auf einem vorberechneten Gitter.

Der Ansatz von Cravo et al. [CRL08] nutzt einen adaptiven Zufalls-Greedy-Algorithmus auf Basis eines Konfliktgraphen und liefert damit gute Resultate. Er benötigt aber gut über eine Minute für Datensätze von 1000 Punkten und es werden keine größeren als diese getestet.

Roy et al. [RBDN05] beschreibt einen schnellen Graph-theoretischen Algorithmus, der eine theoretische Laufzeit von $\mathcal{O}(n \log n)$ hat. Bei einer Größe von 1000 Beschriftungen braucht dieser Algorithmus unter einer Sekunde. Wesentlich größere Mengen benötigen mehrere Sekunden. Jedoch berücksichtigt dieser Ansatz keine Labelprioritäten oder Beschriftungspositionenpräferenzen.

Poon und Shin [PS05] stellen einen Algorithmus vor, der für jede Zoomstufe eine Beschriftung für punkthafte Objekte berechnet und zwischen diesen Lösungen interpoliert. Dabei werden nur achsenparallele, rechtwinklige Labels betrachtet, die das zu beschriftende Objekt an seiner linken Grenze berühren. Beim Herauszoomen wachsen die Beschriftungen nach rechts und vertikal. Diese Lösung beansprucht relativ viel Zeit und Platz in der Vorberechnungsphase.

Mote, K. [Moto7] beschreibt eine schnelle Lösung für das Beschriftungsproblem, die ohne Vorverarbeitungsschritt auskommt. Allerdings mit der Einschränkung, dass nur Beschriftungen gleicher Größe behandelt werden, die achsenparallel sind. Der Algorithmus kann zwar auf nicht uniforme Beschriftungen erweitert werden, allerdings auf Kosten der Optimalität. Die Schnelligkeit dieses Algorithmus wird in dieser Veröffentlichung nicht bewiesen, allerdings liefert die Implementierung in den erfolgten Tests schnelle, effektive und zuverlässige Ergebnisse. In dem Algorithmus wird wie in den bereits vorgestellten Algorithmen ein Konfliktgraph aufgebaut. Dieser Schritt ist normalerweise teuer. Durch eine effiziente Gitterstrategie wird diese Berechnung beschleunigt. Dazu wird die Menge der zu beschriftenden Objekte im Bildraum unterteilt und so die Schnittentscheidung wesentlich vereinfacht.

2. Grundlagen

Dadurch können Karten mit mehreren tausend zu beschriftenden Objekten mit mehreren Frames pro Sekunde berechnet werden [Moto7].

3. Schnelle konsistente Kartenbeschriftung für interaktive Karten

3.1. Modell von Been et al.

3.1.1. Konsistenzbegriff für interaktive Karten

Dynamische Karten, die kontinuierliches Zoomen (Änderung der Skalierung) und Verschieben (Änderung des Interessengebietes) erlauben, bieten die Möglichkeit für viele verschiedene Anwendungen, wie zum Beispiel die Suche nach einem bestimmten Ort oder die kürzeste Route zwischen zwei Punkten. Labels spielen im Zusammenhang mit diesen Aufgaben eine wichtige Rolle, beispielsweise als Navigationsmarker oder zur Information. Dabei ist es wichtig, störendes Verhalten wie zum Beispiel unerwartetes Erscheinen oder unerwartete Bewegungen von Labels zu vermeiden. Um eine brauchbare Visualisierung von interaktiven Karten zu erhalten, müssen folgende zwei Probleme berücksichtigt werden:

1. Es muss vermieden werden, dass Labels unerwartet erscheinen oder andere Effekte auftreten, die Unordnung verursachen.
2. Es muss in einer Geschwindigkeit gelabelt werden, die das interaktive Zoomen und Verschieben möglich machen.

Die im vorherigen Kapitel vorgestellten Verfahren zur Labelplatzierung sind zwar schnell genug, um den Anforderungen von dynamischen Karten gerecht zu werden, aber nicht frei von Artefakten. Been et al. [BDYo6] liefert einen Ansatz, der beide Kriterien erfüllt.

Um das zweite Kriterium zu erfüllen, führen Been et al. [BDYo6] den Begriff der Konsistenz ein. In der Literatur findet man dazu Begriffe wie "frame-coherency" [AHS05], das visuelle Diskontinuitäten verhindern soll und "temporal continuity" [BFHo1]. Bei dieser zeitlichen Kontinuitätsbetrachtung werden bei in die Berechnung eines Frames die Ergebnisse des vorherigen Frames einbezogen. Um den Begriff der Konsistenz näher zu spezifizieren, werden im Folgenden vier Desiderata vorgestellt. Dabei bezieht sich Platzierung auf die Lokalisierung, Größe und Orientierung eines ausgesuchten Labels und Selektion auf die Entscheidung, wann man ein Label zeigt oder nicht.

Desideratum (D1)

Mit der Ausnahme vom Hineingleiten und Herausgleiten aus dem Sichtbereich sollten Labels nicht

3. Schnelle konsistente Kartenbeschriftung für interaktive Karten

verschwinden, wenn man hineinzoomt oder erscheinen, wenn man herauszoomt. Dies entspricht der normalen Annahme, dass wesentlich mehr Labels gezeichnet werden, wenn man hineinzoomt und weniger Labels gezeichnet werden, wenn man herauszoomt. Dieses Desideratum stellt sicher, dass Labels unter kontinuierlichem Zoomen nicht erscheinen, verschwinden und wieder erscheinen.

Dies ist vergleichbar mit der "monotonicity property" wie sie von van Krefeld und van Oostrum beschrieben wurde [KOS97] [Oos99]. In einigen Fällen ist es sinnvoll, eine weniger strikte Version dieses Desideratums zu betrachten, wenn zum Beispiel Beschriftungen für Kontinente beim Hineinzoomen auf Straßenlevel ausgeblendet werden sollen.

Desideratum (D2)

Solange ein Label sichtbar ist, sollte sich seine Position und Größe kontinuierlich während Schiebe- und Zoomoperationen ändern.

Desideratum (D3)

Bis auf das Hinein- oder Herausgleiten aus dem Sichtbereich sollten Labels während einer Schiebeoperation nicht verschwinden oder wieder auftauchen.

Wenn (D2) erfüllt ist kann man (D3) leicht erfüllen, indem man das Problem der Labelselektion mit Hilfe einer Funktion, die die Skalierung in Betracht zieht, löst.

Desideratum (D4)

Die Platzierung und Selektion jedes Labels ist eine Funktion des aktuellen Kartenzustandes (Skalierung und Sichtbereich). Insbesondere basiert es nicht auf früheren Kartenzuständen, die zu diesem Zustand geführt haben.

Abbildung 3.1 zeigt wie einfach die Desiderata mit einem naiven Ansatz verletzt werden können. Das Bild (a) zeigt die Beschriftungssituation der Objekte A, B und C beim Herauszoomen. Die Labelprioritäten sind folgendermaßen verteilt:

$$A > B > C$$

Anfänglich sind alle drei Beschriftungen sichtbar. Wird nun herausgezoomt, dann verdeckt das Label B das Label C. Wird weiter herausgezoomt, verdeckt das Label A das Label B, was somit nicht angezeigt wird. Dafür ist wieder Platz, das Objekt C zu beschriften und es poppt wieder auf. In (b) wird analog die Inkonsistenz beim Nach-Links-Verschieben dargestellt. Das Problem an diesem naiven Ansatz ist, dass der Auswahlalgorithmus von Frame zu Frame auf verschiedenen Labelmengen durchgeführt wird.

3.1.2. Ablauf des Verfahrens von Been et al.

Der in Kapitel 2.2.4 dargestellte Ablauf bei der Beschriftung von Objekten zeigt den in der Literatur üblich verwendeten Ansatz, zuerst die Labelmenge durch einen Filterungsschritt zu reduzieren. Been et al. [BDY06] brechen mit dieser Vorgehensweise und stellen eine

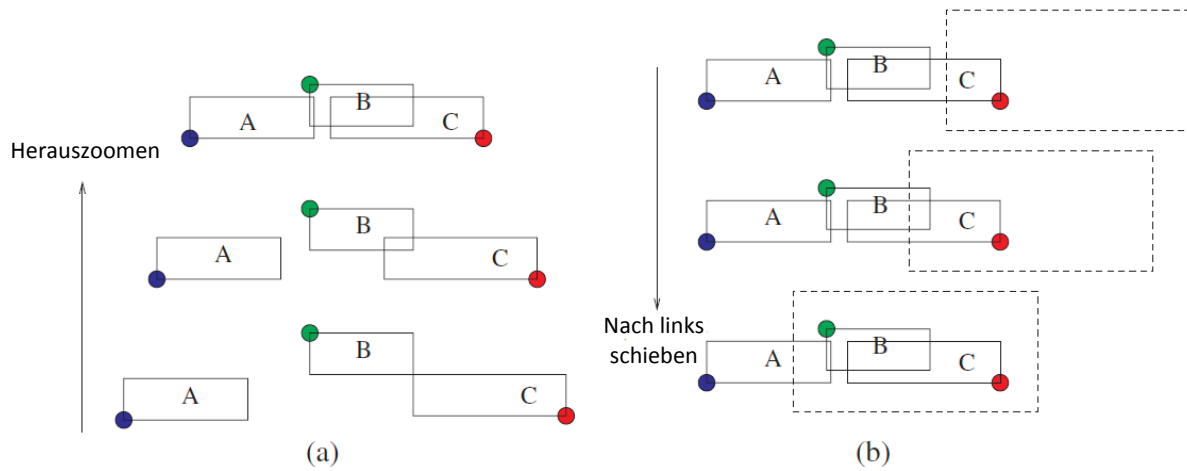


Abbildung 3.1.: Inkonsistenzen (a) beim Herauszoomen und (b) beim Nach-Links-Verschieben. Die Labelprioritäten sind $A > B > C$. Anfänglich ist C sichtbar, wenn man herauszoomt bzw. den Sichtbereich nach links verändert. C verschwindet aufgrund des Konfliktes mit B und poppt wieder auf sobald sich die Label A und B überschneiden [BDYo6]

Lösung des Beschriftungsproblem vor, in der zuerst alle Labels platziert und dann die platzierten Labels selektiert werden. Diese Schritte erfolgen in der Vorverarbeitungsphase. Während der Interaktionsphase wird schließlich ein Filterungsschritt vorgenommen, der auf der vordefinierten Selektion und Platzierung basiert. Mit dieser umgekehrten Reihenfolge kann eine brauchbare und praktische Lösung für die dynamische Kartenbeschriftung erzielt werden.

Im Folgenden wird dieses Verfahren genauer beschrieben. Zunächst wird die statische Beschriftung betrachtet, die aus einer Komposition von zwei Operationen besteht: der Selektierung und der Platzierung.

Label-Platzierungen können durch planare affine Transformationen dargestellt werden. Dabei sind folgende drei Transformationen $\tau : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ von Interesse:

- Translation $\tau = T(e, f)$ um den Vektor $\vec{v} = (e, f), e, f \in \mathbb{R}$
- Rotation $\tau = R(\Phi)$ um den Winkel $\Phi \in [0, 2\pi)$
- Uniforme Skalierung $\tau = D(s)$ durch den Faktor $s \in \mathbb{R}_{>0}$

Jede Transformation kann durch eine 3×3 -Matrix dargestellt werden. Die Komposition der Transformationen entspricht der Multiplikation von Matrizen.

In der Abbildung 3.2 wird eine statische Platzierung dargestellt. D_0 bezeichnet die Domäne der Karte M . Dies ist ein rechteckiges Gebiet von \mathbb{R} .

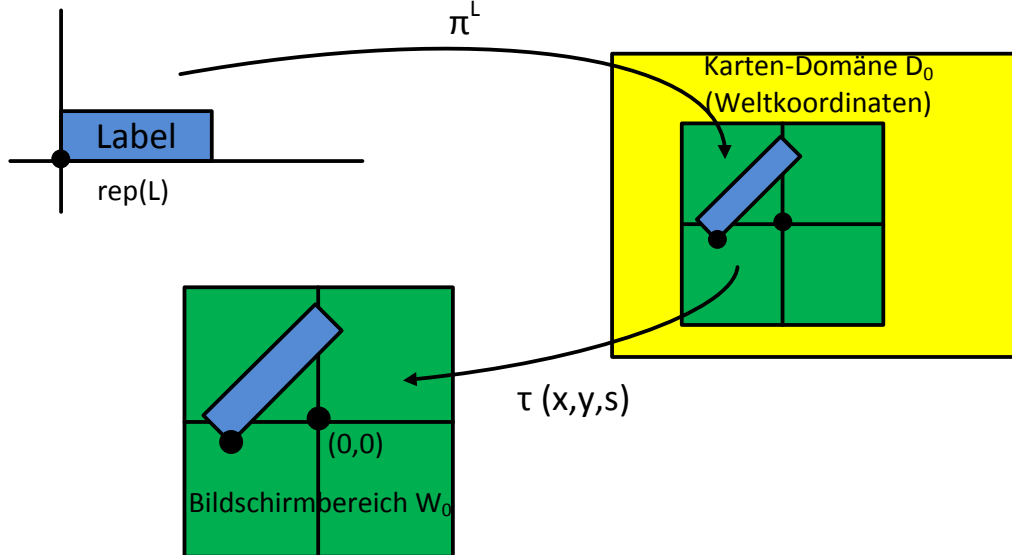


Abbildung 3.2.: Transformation von Labelkoordinaten über Weltkoordinaten zu Bildschirmkoordinaten [BDYo6]

Eine Beschriftung L besteht meistens aus einer Zeichenkette. Es kann aber auch ein Symbol oder eine Kombination von beidem sein. In Been et al. [BDYo6] wird L als eine zusammenhängende Menge $rep(L) \subseteq \mathbb{R}$ dargestellt. L wird in seinem eigenen Labelkoordinatensystem gezeichnet. Typischerweise ist $rep(L)$ eine rechteckige Box (siehe Abbildung 3.2). L wird mit einem zu beschriftenden Objekt $\phi(L) \subseteq D_0$ assoziiert, das es beschriftet.

W_0 ist der Bildschirmbereich in Fensterkoordinaten. Die achsenausgerichtete rechteckige Untermenge von D_0 , die in W_0 zu einem gewissen Zeitpunkt dargestellt ist, wird Sichtbereich genannt. Der Sichtbereich ist definiert durch die Breite w und die Höhe h von W_0 in Fensterkoordinaten, einem zentralen Punkt $(x, y) \in D_0$ und einer Skalierung s . Die Skalierung s wird bei Been et al. [BDYo6] so definiert, dass ein $w \times h$ Fenster zu einem $ws \times hs$ Sichtbereich führt. Deshalb entspricht eine Erhöhung der Skalierung dem Herauszoomen. Einfachheitshalber wird angenommen, dass w und h fix gewählt sind, so dass der Sichtbereich gegeben ist durch $W(x, y, s) \subseteq D_0$ und den aktuellen Status (x, y, s) . Während der Interaktionsphase gibt der Benutzer eine Sequenz von Kommandos in der Form "verschiebe nach (x, y) " oder "zooome nach s ". Wenn nun ein Fenster definiert ist, das sein Zentrum bei $(0, 0)$ in Fensterkoordinaten hat, dann ist die Transformation $\tau(x, y, s)$, die $W(x, y, s)$ nach W_0 bringt eine Translation gefolgt von einer Skalierung.

$$\tau(x, y, s) = D \left(\frac{1}{s} \right) \circ T(-x, -y)$$

Eine statische Platzierung für ein Label L ist somit eine zulässige Transformation π (oder π^L) von Labelkoordinaten in Weltkoordinaten. $rep(L)$ wird also in einer transferierten, rotierten und skalierten Form $\pi(rep(L))$ gezeichnet. Die Transformation π wird als gültig bezeichnet, wenn es bestimmte kartografische Bedingungen erfüllt. So erwartet man zum Beispiel, dass $\pi(rep(L))$ nahe am zu beschriftenden Objekt liegt.

Ein Label L ist sichtbar unter einer Platzierung π im Zustand (x, y, s) , wenn $\pi(rep(L)) \cap W(x, y, s)$ nicht leer ist (vgl. Abbildung 3.2). In Fensterkoordinaten wird die Platzierung von π in folgendem Bereich gezeigt:

$$(\tau(x, y, s) \circ \pi)(rep(L)) \cap W_0$$

Im Gegensatz zum statischen Labeling benötigt man beim dynamischen Labeling eine dritte Dimension. In dem Modell von Been et al. [BDYo6] wird für die dritte Dimension die Skalierung gewählt. Die dynamische Platzierung Π^L für ein Label L ist eine Funktion, die eine statische Platzierung π_s^L für jede Skalierung $s \in (0, \infty)$ bestimmt:

$$\Pi^L : s \in (0, \infty) \rightarrow \pi_s^L$$

Dies liefert eine fixe Platzierung pro Label und Skalierung und verhindert damit ein Verrutschen des Labels, wenn der Sichtbereich verändert wird.

Π^L ist gültig, wenn jedes π_s^L eine gültige statische Platzierung von L ist. Um das Desideratum (D2) zu erfüllen, muss Π^L kontinuierlich mit der Skalierung s sein.

Dies liefert in Weltkoordinaten ein ausgedehntes Label mit Skalierung als vertikale Dimension, dessen Labelgröße in Weltkoordinaten proportional zur Skalierung ist und somit in Bildschirmkoordinaten konstant unter dem Zoom-Operator. Π^L erfüllt die Labelgrößen-Invarianz-Eigenschaft, wenn es ein konstantes c gibt, so dass für jede Skalierung s gilt:

$$det(\pi_s^L) = cs^2$$

Durch die Skalierung der Labelkoordinaten kann man ohne Beschränkung der Allgemeinheit $c = 1$ annehmen. Dann ist $(\tau(x, y, s) \circ \pi_s^L)(rep(L))$ eine translierte, rotierte Kopie von $rep(L)$.

Für die Platzierung von Beschriftungen wird eine sogenannte "invariante Punktplatzierung" verwendet. Formell ist dies eine dynamische Platzierung von L , die repräsentiert wird durch (p, q, θ) , wobei p ein Punkt in Weltkoordinaten, q ein Punkt in L 's Labelkoordinaten und θ ein Winkel ist. Für jede Skalierung $s > 0$ definiert die statische Platzierung π_s^L die Form

$$\pi_s^L = T(p)R(\theta)D(s)T(-q)$$

wobei T , R und D die Translations-, Rotations- und Skalierungstransformationen sind. Es gilt $\pi_s^L(q) = p$ für alle s (dies ist die Invariante). Zum Beispiel gilt: Wenn $q \in rep(L)$ dann gilt auch an einer beliebigen Skalierung s , $p \in \pi_s^L(rep(L))$. Dies wird in Abbildung 3.3 gezeigt.

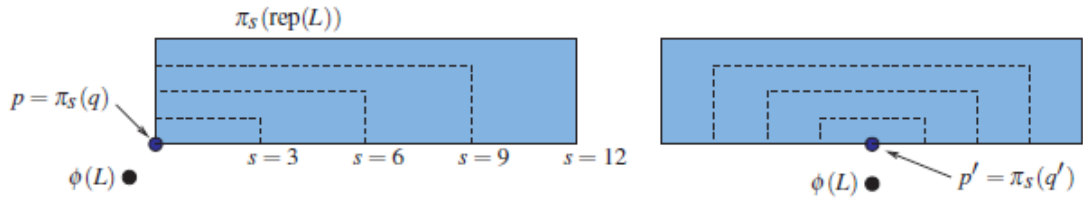


Abbildung 3.3.: Zwei mögliche invariante Punktplatzierungen für das zu beschriftende Objekt $\phi(L)$, die durch $(p, q, 0)$ und $(p', q', 0)$ an den Skalierungen $s = 3, 6, 9, 12$ in Weltkoordinaten dargestellt sind. [BDY06]

Die dynamische Platzierung in Abbildung 3.4 sind invariante Punktplatzierungen mit rechteckigen Labels. Die Ausdehnungen, welche die Labelgrößen-Invarianz-Eigenschaften besitzen, sind Pyramiden.

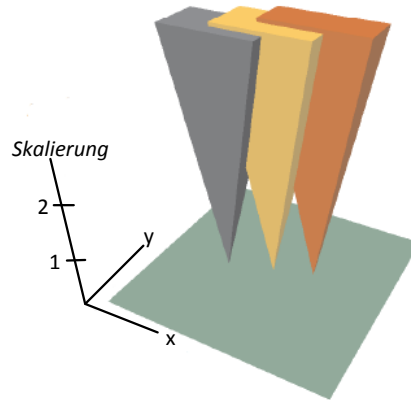


Abbildung 3.4.: Dynamische Platzierung dreier Beschriftungen in Weltkoordinaten. [BDY06]

Durch das Desideratum (D1), das besagt, dass Labels unter kontinuierlichem Zoomen nicht erscheinen, verschwinden oder wieder erscheinen sollen, wird das dynamische Selektionsproblem darauf beschränkt, dass jede Beschriftung zu exakt einem Intervall ausgewählt sein muss. Dieses Intervall wird der aktive Bereich genannt.

$$A^L := [s_{min}^L, s_{max}^L]$$

Unter der strikten Interpretation von (D1) darf ein Label nicht verschwinden, wenn man hineinzoomt. In diesem Fall ist

$$s_{min}^L = 0$$

Der aktive Bereich für ein Label wird nun bestimmt, indem seine Ausdehnung mit allen anderen Labels geschnitten wird. Existiert ein Schnitt mit einem Label, das eine höhere Priorität besitzt, dann wird dieser Schnittpunkt als s_{max}^L festgelegt (vgl. Abbildung 3.5).

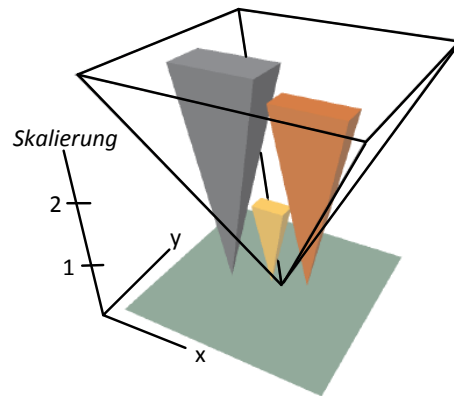


Abbildung 3.5.: Dieselben drei Beschriftungen wie in Abbildung 3.4, beschränkt auf ihre aktiven Bereiche, so dass sich keine Labels überlappen. [BDYo6]

Ein Schnitt dieser ausgedehnten Beschriftungen bei einer festen Skalierung ergibt die 2D-Karte. Zwei solcher Schnitte der Pyramiden aus Abbildung 3.5 werden in Abbildung 3.6 dargestellt. Die Skalierung $s = 1$ liegt in den aktiven Bereichen aller drei Labels und diese sind somit aktiv, aber nur das gelbe und das rote Label sind innerhalb des Sichtbereichs. Bei $s = 2$ wird der Sichtbereich durch Herauszoomen vergrößert. Dadurch werden alle drei Beschriftungen erfasst. Das gelbe Label wird nicht gezeichnet, da es im Konflikt mit dem Roten steht. Beim Herauszoomen von $s = 1$ auf $s = 2$ gleitet das graue Label in den Sichtbereich und alle drei Labels bewegen sich aufeinander zu. Während des Herauszoomens zwischen $s = 1$ und $s = 2$ verschwindet das gelbe Label aufgrund des Konflikts mit dem Roten.

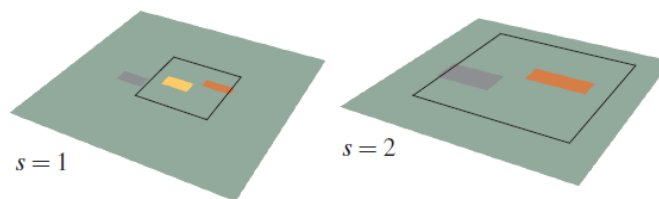


Abbildung 3.6.: Schnitt durch die ausgedehnten Labels ergeben die 2D-Karte bei $s = 1$ links und $s = 2$ rechts. Der Sichtbereich wird durch das schwarze Quadrat definiert. [BDYo6]

3. Schnelle konsistente Kartenbeschriftung für interaktive Karten

Zusätzlich zur Selektion und Platzierung wird noch eine Filterungsoperation benötigt, um die nötige Interaktivität gewährleisten zu können. Die Art der Filterung ist in diesem Ansatz offen gelassen und kann für verschiedene Aufgaben unterschiedlich implementiert werden. Im Fall einer Straßenkarte ist es beispielsweise sinnvoll, kleine Nachbarstraßen heraus zu filtern, wenn man weit genug herauszoomt.

Der Algorithmus kann nun folgendermaßen zusammengefasst werden:

Algorithmus 3.1 Konsistente Beschriftung

Vorverarbeitungsphase

- 1.) Bestimme eine dynamische Platzierung für jedes Label. In diesem Schritt wird jedes Label isoliert betrachtet, ohne Berücksichtigung von Konflikten mit anderen Labels.
- 2.) Wähle einen aktiven Bereich für jedes Label, so dass sich die resultierenden eingeschränkten Ausdehnungen paarweise nicht überlappen.

Interaktionsphase

- 1.) Filtere die Labels auf Basis von geografischen Bereichen und/oder Skalierung.
 - 2.) Stelle jedes Label, das nicht heraus gefiltert wurde, genau dann dar, wenn die aktuelle Skalierung in seinem aktiven Bereich liegt.
-

3.1.3. Beschriftungsoptimierungsproblem

Um zu zeigen, dass der Selektions- und Platzierungsalgorithmus optimal ist, muss gezeigt werden, dass so viele Labels wie möglich angezeigt werden. In einer statischen Karte entspricht diese Bedingung einfach der Anzahl der Labels, die selektiert wurden. Für eine interaktive Karte bedeutet dies, dass man über die Skalierung integrieren muss. Durch den aktiven Bereich wird dieses Integral im Modell von Been et al. [BDYo6] zu einer einfachen Summe reduziert.

Damit ergibt sich das folgende dynamische Beschriftungsoptimierungsproblem:

Definition 10 (Beschriftungsoptimierungs-Problem)

Gegeben sei eine Menge von Label S_0 .

Bestimme eine dynamische Platzierung und einen aktiven Bereich für jedes $L \in S_0$, so dass sich nicht zwei beschränkte Ausdehnungen überlappen und die Anzahl der aktiven Labels über allen Skalierungen maximal ist.

$$\max \sum_{L \in S_0} s_{\max}^L - s_{\min}^L$$

Bei dieser Formulierung wird jede Beschriftung unabhängig von Prioritäten gleich behandelt. Jeder aktive Bereich hat eine endliche obere Grenze s_{max} , so dass für jedes Label L gilt

$$s_{max}^L \leq s_{max}$$

Zur Einfachheit wird die strikte Version von (D1) angenommen, so dass gilt:

$$s_{min}^L = 0$$

In [BDYo6] wird gezeigt, dass das einfachere Problem, eine optimale Menge von aktiven Bereichen bei einer gegebenen Menge von dynamischen Platzierungen zu berechnen, bereits NP-Vollständig für eine generalisierte Klasse von Labelgrößen ist. Been et al. behandelt in "Optimizing Active Ranges for Consistent Dynamic Map Labeling" [BNPW08] dieses sogenannte "active range optimization (ARO)" Problem genauer. In diesem Artikel werden die NP-Vollständigkeitsbeweise für einfache Körperformen gezeigt und Lösungsverfahren vorgestellt, welche die naive Lösung um einen konstanten Faktor beschleunigen.

Für eine praktische Lösung müssen Labelprioritäten mit den folgenden Bedingungen eingeführt werden:

- Jedes Label hat eine eindeutige Priorität.
- Wenn zwei Labels an einem Kartenplatz kollidieren, muss das höher priorisierte Label angezeigt werden.

Dies ist eine Zuordnung $P : S_0 \rightarrow \mathbb{R}$. Prioritäten sind eindeutig $L \neq L'$ impliziert $P(L) \neq P(L')$. Eine Funktion A^* , die der Menge S_0 aller Labels einen aktiven Bereich zuweist, muss nun die folgende Prioritätsregel erfüllen: Wenn L blockiert ist, dann ist es blockiert durch ein L' mit $P(L') > P(L)$.

Damit ergibt sich das folgende eingeschränkte Beschriftungsoptimierungsproblem:

Definition 11 (eingeschränktes Beschriftungsoptimierungsproblem)

Gegeben sei eine Menge von Labels S_0 und eine dynamischer Platzierung für jedes $L \in S_0$. Wähle einen aktiven Bereich für jedes $L \in S_0$, so dass

- $s_{min}^L = 0$,
- für jedes L sich keine zwei beschränkte Ausdehnungen überlappen,
- die Prioritätsregel erfüllt ist,
- $\sum_{L \in S_0} s_{max}^L$ maximiert ist.

3. Schnelle konsistente Kartenbeschriftung für interaktive Karten

Die Lösung ist ein einfacher Greedy-Algorithmus (vgl. Algorithmus 3.2), der jede beschränkte Ausdehnung so hoch wie möglich setzt ohne Überlappung einer vorher definierten d.h. höher priorisierten Ausdehnung.

Algorithmus 3.2 Greedy-Algorithmus für konsistente Beschriftungen

```
for Jedes Label in Reihenfolge ihrer Priorität vom höchsten zum niedrigsten do  
  if L wird durch ein höher priorisiertes Label bei der Skalierung  $s_x$  blockiert then  
    Setze  $s_{max}^L = s_x$   
  end if  
end for
```

Beweis der Optimalität des Greedy-Algorithmus:

Lemma 12 (Lemma)

Unter der Prioritätsregel und dem beschränkten Desideratum (D1) ist das A^ , das durch den Algorithmus G berechnet wurde, optimal unter beliebigen Π^* , das jedem Label aus der gegebene Menge S_0 eine statische Platzierung zuweist.*

Beweis:

Zuerst bemerke, dass unter beliebiger optimaler Lösung jedes Label L entweder blockiert sein muss oder $s_{max}^L = s_{max}$ gilt. Ansonsten könnte man s_{max}^L erhöhen, so dass es zu einer besseren Lösung kommt. Angenommen, es gibt eine optimale Lösung θ , in welcher einige Labels einen größeren aktiven Bereich haben als unter G und L sei das höchst Priorisierte dieser Labels. L muss durch ein höher priorisiertes Label L' unter G blockiert sein, aber nicht unter θ . Damit ist der aktive Bereich von L' kleiner unter θ als unter G. Da θ hinsichtlich der Prioritätsregel optimal ist, muss L' blockiert sein unter θ durch ein höher priorisiertes Label L'' . Aber L' wird nicht blockiert durch L'' unter G. Aber $P(L'') > P(L') > P(L)$ was der Wahl von L widerspricht. Deshalb ist jedes aktive Intervall unter G mindestens so groß wie unter einer beliebigen optimalen Lösung, so ist G optimal. q.e.d.

3.2. Praktische Betrachtungen

3.2.1. Terrainrenderer

Als Basis für die Implementierung von Beschriftungen wird in dieser Diplomarbeit der Terrainrenderer von [Vaa08]. verwendet. Dieser wurde zur Darstellung von hochdetaillierten

Navigationskarten auf 3D-Geländemodellen implementiert. Für das Beschriftungsproblem sind vor allem die folgenden Eigenschaften wichtig:

- Ein flaches Terrain wird mittels Mercator-Projektion berechnet.
- Benutzerinteraktionen wie Verschieben, Zoomen, Rotieren und Kippen sind möglich.
- Es können Straßen-Datensätze von OpenStreetMap angezeigt werden.
- 3D-Darstellung ist möglich.
- Es können Höheninformationen abgefragt werden.

Um die Straßen-Datensätze aus OpenStreetMap für das Rendering zu erhalten, wurde ein OMS-Converter benutzt. Dieser Converter basiert auf einem SAX2-Parser und ist Basis für diese Diplomarbeit. Er wurde um die Funktionalität "Beschriftungs-Datensätze aus dem OMS-Datenformat lesen" erweitert.

3.2.2. Berechnungsschritte

3.2.2.1. Labeldarstellungen

Im folgenden Abschnitt werden einige praktische Betrachtungen der Berechnungsschritte des Modells von Been et al. [BDY06] sowie Beschleunigungsstrategien für den Greedy-Algorithmus vorgestellt.

Der Ablauf der Implementierung des Modells wird in folgendem Pseudocode dargestellt:

Algorithmus 3.3 Ablauf der Implementierung von Been et al. [BDY06]

- (1) Lese die Daten ein.
 - (2) Setze Labeleigenschaften wie zum Beispiel Prioritäten.
 - (3) Berechne die Schnitte zwischen den Labelpaaren.
 - (4) Schreibe die Ergebnisse in eine Datei.
-

Im ersten Schritt des Algorithmus 3.3 werden die Daten aus einer OpenStreetMap-Datei ausgelesen. Dies erfolgt mit dem oben beschriebenen OMS-Converter. Dabei werden drei Labelarten unterschieden (vgl. Abbildung 3.7):

- Namen wie Länder-, Städte- oder Ortsnamen,
- Straßen,
- Interessenpunkte "Point of Interest" (POI), wie zum Beispiel Restaurants, Kindergärten oder Krankenhäuser.

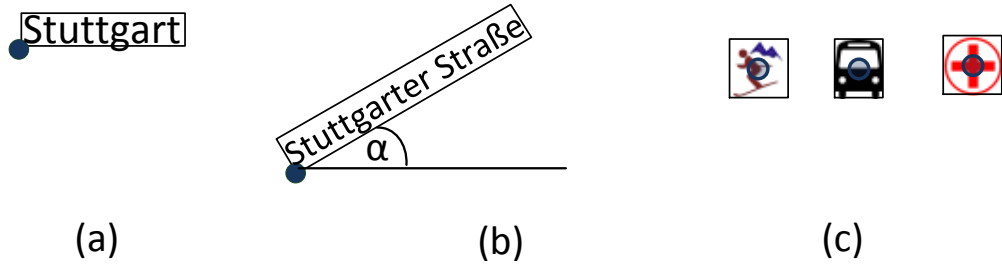


Abbildung 3.7.: (a) Namen-Label, (b) Straßen-Label, (c) POI-Label. Der blaue Punkt gibt die geografische Lage des zu beschriftenden Objektes an.

Die Repräsentation $rep(L)$ von Interessenpunkten sind achsenparallele Rechtecke, der Mittelpunkt definiert dabei die geografische Lage eines POIs und ist somit die Invariante. Die Ausdehnung dieser Rechtecke sind Pyramiden (Anhang A zeigt alle dargestellten POIs mit entsprechender Textur). Straßen-Beschriftungen werden als gedrehte Rechtecke dargestellt, die den Namen der Straßen repräsentieren. Die linke untere Ecke dieses Rechtecks im umgedrehten Fall definiert die Invariante. Die Ausdehnung ist wie bei den POIs eine Pyramide. Die Zeichen des Namens der Straße werden durch gleich große Rechtecke dargestellt. Ein Straßen-Label ist dann genauso hoch wie ein Buchstabe und so breit wie die Summe aller Zeichen-Rechtecke, die den Namen der Straße bilden plus die Breite des jeweiligen Zwischenraums zwischen zwei Zeichen. Ist ein Buchstabe z.B 16 Pixel breit und der Zwischenraum zwischen zwei Zeichen 4 Pixel, dann hat die "Stuttgarter Straße" eine Breite von

$$16 \text{ Pixel} * 18 \text{ Zeichen} + 4 \text{ Pixel} * 17 \text{ Zwischenräume} = 356 \text{ Pixel}$$

In der OpenStreetMap-Datei werden Straßen als eine Liste von Punkten dargestellt. Den Winkel, um den das Straßen-Label gedreht wird, erhält man folgendermaßen: Man drittelt die Anzahl aller Punkte, welche die Straße ergeben. Der Winkel zwischen der Gerade, durch die beiden Punkte welche das mittlere Drittel beschränken und der Achse wird als Drehwinkel verwendet (vgl. Abbildung 3.8). Sei p_1 der Anfangspunkt des mittleren Drittels und p_2 der Endpunkt, dann gilt für $(p_2.x - p_1.x) \neq 0$:

$$\alpha = \arctan\left(\frac{p_2.y - p_1.y}{p_2.x - p_1.x}\right)$$

ansonsten gilt:

$$\alpha = 90^\circ$$

Namen-Labels werden mittels eines achsenparallelen Rechtecks dargestellt. Höhe, Breite, Invariante und Ausdehnung sind dabei analog zu den Straßen-Labels.

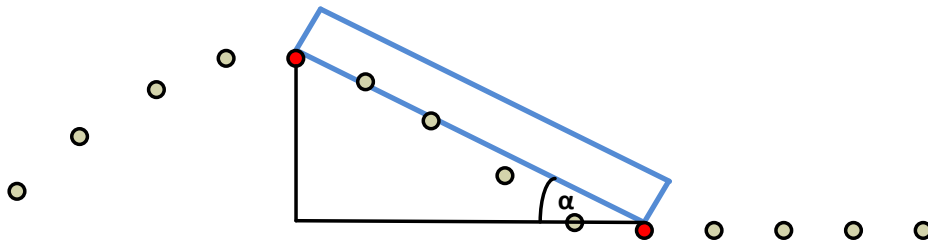


Abbildung 3.8.: Die grauen Punkte geben die Positionen einer Straße aus der OpenStreetMap-Datei an. Die roten Punkte sind die Begrenzungspunkte des mittleren Drittels dieser Positionen. Der Winkel α gibt an, wie weit das blau dargestellte Label gedreht wurde.

Im nächsten Algorithmusschritt werden die Labeleigenschaften vergeben die nicht direkt durch das Einlesen gesetzt werden, wie zum Beispiel die Priorität. Jedes Label besitzt folgende Eigenschaften:

- Typ: Straße, POI oder Name
- Name des zu beschriftenden Objektes
- Labelposition: Tupel aus geografischer Länge und Breite (φ, λ)
- Ausdehnung: Tupel aus Breite und Höhe (b, h)
- Rotationswinkel α
- Lebensbereich: Tupel aus maximaler und minimaler Höhe s_i, s_j mit $0 \leq i < j \leq s_{max}$
- Priorität p
- Textur

3.2.2.2. Schnittberechnungen

Als nächstes muss ein aktiver Bereich $A^L = [s_i, s_j]$ mit $0 \leq i < j \leq s_{max}$ für jedes Label berechnet werden. In dieser Implementierung wird $s_i = 0$ angenommen. Für s_j wird die liberale Version des Desideratums betrachtet, so dass ein Label wie zum Beispiel ein Ländername verschwindet, wenn man weit genug (z.B. auf Straßenniveau) hineinzoomt. Wird ein Label durch ein höher Priorisiertes verdeckt, wird s_j auf die Höhe des Schnittpunktes gesetzt. Der Schnitt zwischen zwei Labels wird berechnet, indem die Ausdehnungen geschnitten werden.

3. Schnelle konsistente Kartenbeschriftung für interaktive Karten

Da Labels beliebig gedreht sein können, wird der Schnitt realisiert, indem die Seitenebenen des ersten Labels mit den Eckgeraden des zweitens Labels geschnitten werden (vgl. Abbildung 3.9). Sei die Ebene gegeben durch die Hessesche Normalform,

$$ax + by + cz = d$$

und die Gerade durch ihre Parameterform,

$$\vec{g} = \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} + t \cdot \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix}$$

dann schneiden sich diese, genau dann wenn:

$$r_1 \cdot a + r_2 \cdot b + r_3 \cdot c = 0$$

Der Schnittpunkt ergibt sich, indem der folgende Parameter t in die Geradengleichung eingesetzt wird:

$$t = \frac{d - p_1 \cdot a - p_2 \cdot b - p_3 \cdot c}{r_1 \cdot a + r_2 \cdot b + r_3 \cdot c}$$

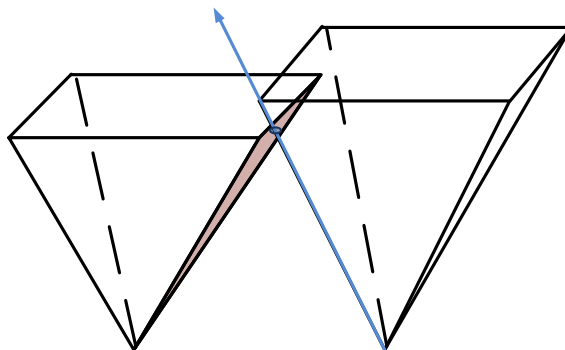


Abbildung 3.9.: Der Schnitt zwischen zwei Pyramiden wird durch die Berechnung der Schnittpunkte der Ebenen der linken Pyramide (rot) mit den Eckgeraden der rechten Pyramide (blau) realisiert.

Der niedrigste gültige Schnittpunkt dieser Schnittpunktmenge ergibt die maximale Höhe s_j des niedriger priorisierten Labels. Gültig ist ein Schnittpunkt genau dann, wenn sich die Labels in der berechneten Höhe auch wirklich schneiden. Abbildung 3.10 links zeigt, dass sich die rechte Seitenebene der linken Pyramide und die linke vordere Eckgerade der rechten Pyramide sich zwar als erstes schneiden, der Schnittpunkt aber nicht innerhalb der Pyramiden liegt. Die zugehörigen Labels in dieser Höhe überlappen sich aber noch nicht. Den richtigen Schnittpunkt zeigt Abbildung 3.10 rechts.

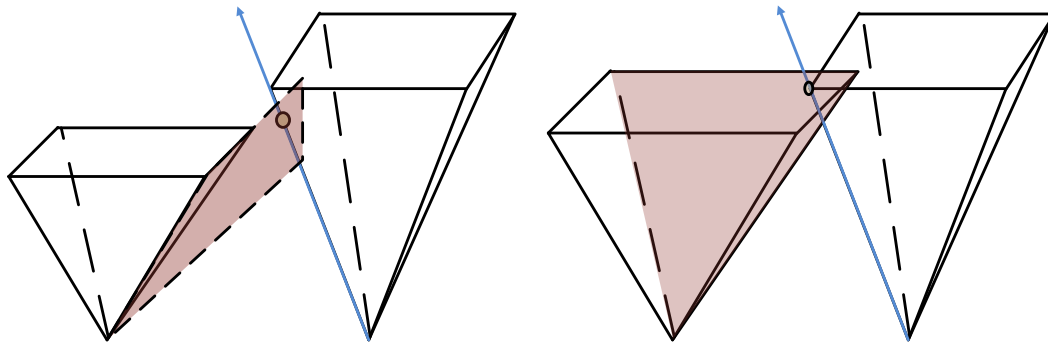


Abbildung 3.10.: Links: Der niedrigste Schnittpunkt zwischen den Seitenebenen (rot) der linken Pyramide und den Eckgeraden (blau) der rechten Pyramide ist kein gültiger Schnittpunkt, da sich die Labels in der Höhe noch nicht schneiden. Rechts: Der niedrigste gültige Schnittpunkt zwischen den zwei dargestellten Labels ist der Schnitt zwischen der hinteren Ebene (rot) und der rechten vorderen Eckgeraden (blau).

Um zu überprüfen, welcher Schnittpunkt der Richtige ist, wird zu jedem Schnittpunkt das aktuelle 2D-Bild erzeugt und geprüft, ob sich die Labels schneiden. Abbildung 3.11 zeigt verschiedene Möglichkeiten wie sich Labels schneiden können.

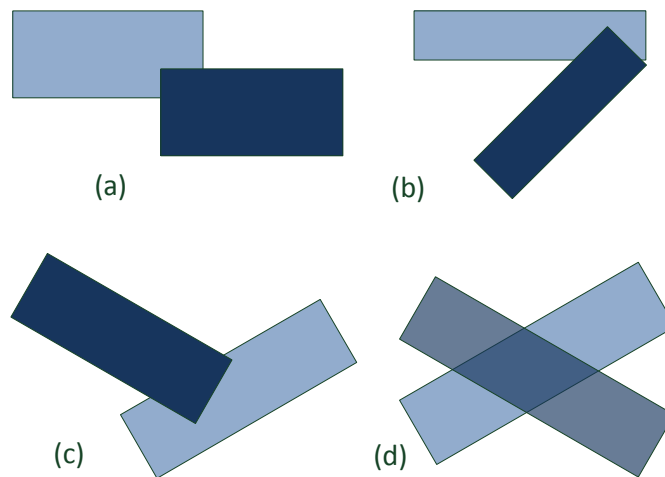


Abbildung 3.11.: (a) Zwei achsenparallele Rechtecke, (b) ein achsenparalleles und ein gedrehtes Rechteck, (c) zwei gedrehte Rechtecke, (d) zwei gedrehte Rechtecke die sich komplett durchschneiden.

Für den Fall (a) ist die Schnittberechnung einfach. Seien p_1, p_2, q_1, q_2 die Punkte, welche die beiden Rechtecke aufspannen, dann schneiden sich diese, falls gilt:

$$p2.x \geq q1.x \wedge p2.y \geq q1.y \wedge q2.x \geq p1.x \wedge q2.y \geq p1.y$$

Sobald ein Label wie in (b) und (c) verdreht ist, reicht dieser einfache Algorithmus nicht mehr aus. Abbildung 3.12 zeigt im rechten Bild, wie dieser Schnitt realisiert werden kann. Mit dem Mittelpunkt M des Rechtecks und den zwei Faktoren u, v lassen sich alle Punkte des Rechtecks durch folgende Linearkombination darstellen:

$$p_i = M + u \cdot \bar{x} + v \cdot \bar{y}, |u, v| \leq 1$$

Seien p_1, p_2, p_3, p_4 die Eckpunkte des ersten Rechtecks gegen den Uhrzeigersinn gezählt, dann gilt:

$$\begin{aligned}\bar{x} &= (p_2.x - p_1.x) \cdot \frac{1}{2} \\ \bar{y} &= p_3.y - p_2.y \cdot \frac{1}{2}\end{aligned}$$

Damit kann geprüft werden, ob die Eckpunkte des anderen Rechtecks durch eine Linearkombination mit $|u, v| \leq 1$ dargestellt werden können. Wird ein solcher Punkt gefunden, schneiden sich die Labels.

Für den Fall (d) wird die Berechnung des Schnittes noch aufwendiger. Dieser Fall tritt hier allerdings nie auf. Wenn sich zwei Pyramiden durchschneiden, muss es bereits einen niedrigeren Schnittpunkt geben.

Im letzten Schritt des Algorithmus werden alle Labeleigenschaften in eine Datei geschrieben, die vom Terrainrenderer, der die Labels darstellt, ausgelesen wird.

3.2.2.3. Beschleunigungsstrategien

Um die Laufzeit dieser Vorgehensweise von $\mathcal{O}(n^2)$ zu reduzieren, wurden Detailstufen "Level of Detail" (LOD), Gitter "Buckets" und weitere Verbesserungen eingeführt. So ergibt sich der Ablauf, wie er im Algorithmus 3.4 dargestellt ist.

Die erste Erweiterung ist die Einteilung des Skalierungsintervalls $[0, s_{max}]$ in h Unterintervalle (vgl. Abbildung 3.13),

$$\rho_i = [s_i, s_{i+1}], i = 0, \dots, h-1$$

die jeweils eine Detailstufe repräsentieren. Damit gibt es $h+1$ Skalierungsstufen $0 < s_0 < s_1 < \dots < s_h = s_{max}$ und jedes ρ_i deckt das Intervall $[s_i, s_{i+1}]$ ab. Die Einteilung der Intervalle kann empirisch auf den betrachteten Datensatz angepasst werden (vgl. . Abschnitt 5.3.1).

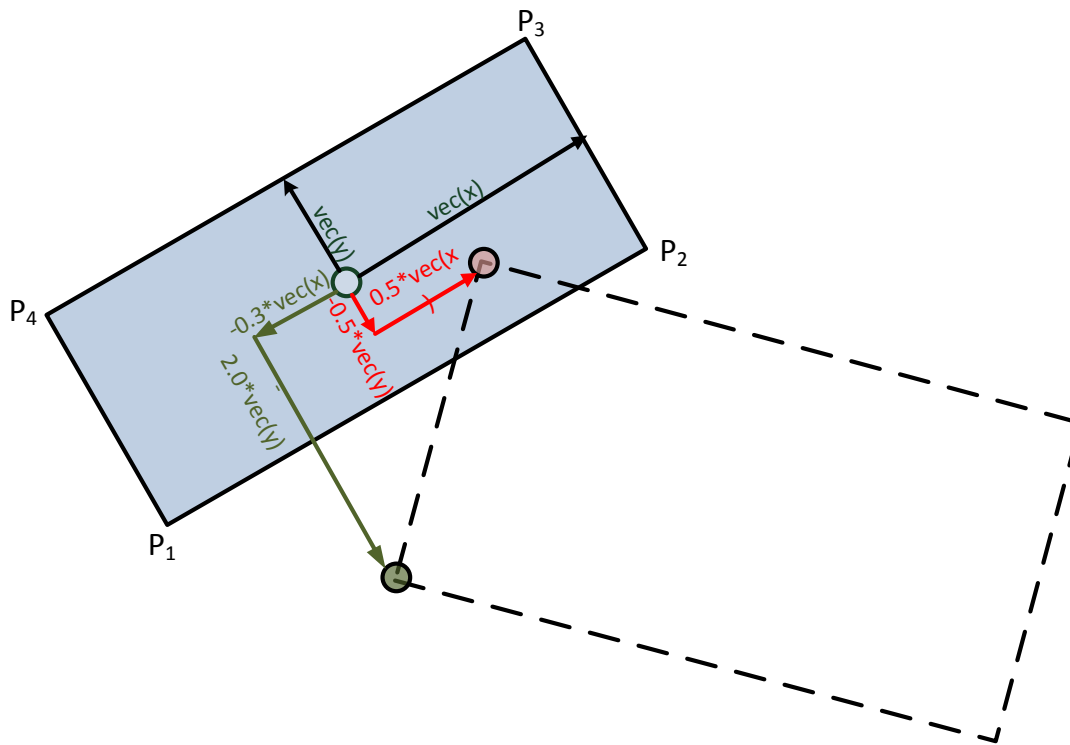


Abbildung 3.12.: Zwei Rechtecke der Fälle (b) und (c) aus Abbildung 3.11 schneiden sich. genau dann wenn sich ein Eckpunkt des einen Rechtecks aus der Linearkombination der Vektoren \vec{x} und \vec{y} mit $p_i = M + u \cdot \vec{x} + v \cdot \vec{y}$, $|u| \leq 1 \wedge |v| \leq 1$ darstellen lässt. Das rote Label zeigt einen Schnittpunkt. Beim grünen Label ist zwar $u < 1$, allerdings ist $v = 2$ und damit der Eckpunkt nicht im anderen Rechteck.

Für jede LOD-Stufe werden nun separat die Schnitte berechnet. Die Menge der Labels, die für eine LOD-Stufe geschnitten werden müssen, sind genau die Labels, die in diesem Unterintervall aktiv sind.

Mit dieser Unterteilung in Detailstufen müssen nur in den niedrigen LOD-Stufen viele Schnitte berechnet werden. In den höheren LOD-Stufen sind bereits viele Labels nicht mehr aktiv und so reduziert sich die Anzahl an Schnittberechnungen. Deshalb ist es sinnvoll für die Einteilung der Intervalle eine logarithmische Skalierung zu verwenden.

Um die Anzahl der zu berechnenden Schnitte weiter zu reduzieren, wird für jedes p_i die Kartendomäne D_0 in ein "Bucket"-Gitter aufgeteilt. Die Größe eines Buckets ist $b \times b$, wobei b die Breite des größten Labels bei der Skalierung der oberen aktuellen Intervallgrenze

Algorithmus 3.4 Ablauf der beschleunigten Implementierung von Been et al. [BDYo6]

- (1) Lese die Daten ein.
 - (2) Setze Labeleigenschaften wie zum Beispiel Prioritäten.
 - (3) Teile die Skalierungsskala in LOD-Stufen ein.
 - (4) Berechne alle Eckgeraden und Seitenebenen der Pyramiden.
 - for** jede LOD-Stufe **do**
 - (5) Unterteile die Karte in Buckets.
 - for** jedes Bucket **do**
 - (6) Berechne die Schnitte zwischen je zwei Labels in dem aktuellen Bucket und den Labels in den Nachbarbuckets.
 - end for**
 - end for**
 - (8) Schreibe die Ergebnisse in eine Datei.
-

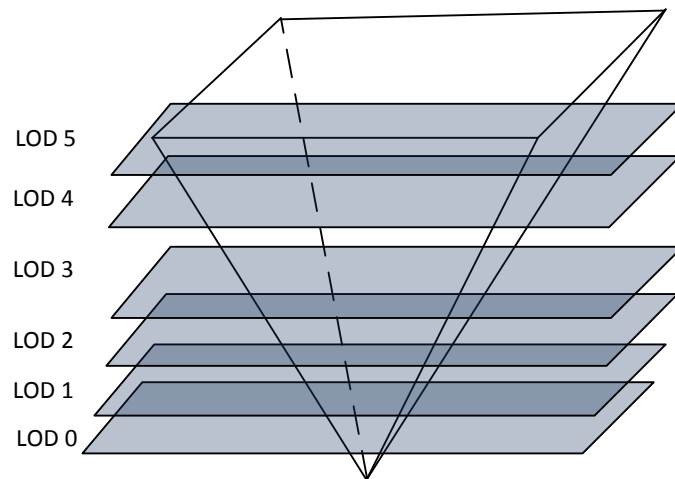


Abbildung 3.13.: Unterteilung des Skalierungsintervalls in Unterintervalle ρ_i .

s_{i+1} ist. Jedes Label wird dem Bucket zugeordnet, in dem seine Invariante liegt. Abbildung 3.14 zeigt die Unterteilung in Buckets. Das Label mit der Priorität 0 ist in diesem Beispiel das breiteste Label und definiert somit die Größe der Buckets. Label 1 zeigt, dass Labels die Bucketgrenzen überschreiten können, jedoch durch die Wahl von b als die maximale Breite eines Labels, kann es maximal die direkten Nachbarbuckets schneiden. Um nun alle möglichen Schnitte zu erfassen, müssen zuerst die Schnitte in den jeweiligen Buckets (siehe Label 2 und 3) und anschließend die Überlappung mit den Nachbarbuckets (siehe Labels 4 – 7) berechnet werden. In diesem Beispiel wird der Schnitt zwischen den Labels 4 und 6 zweimal berechnet. Einmal, wenn das Bucket betrachtet wird welches die Invariante des Labels 4 beinhaltet, und einmal, wenn das Bucket mit der Invariante des Labels 6 berechnet

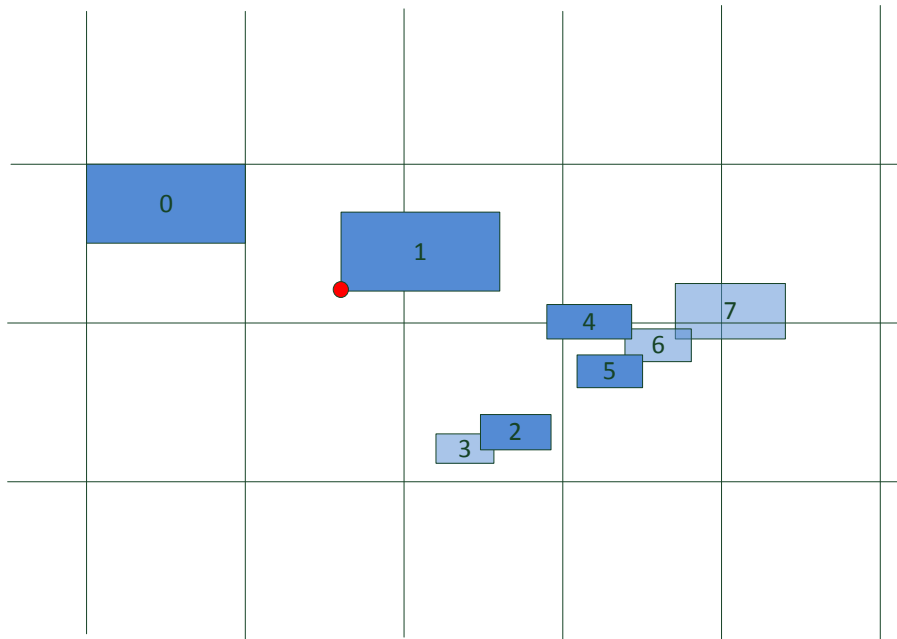


Abbildung 3.14.: Unterteilung der Kartendomain in Buckets der Größe $b \times b$. Das Label 0 definiert die Größe der Buckets. Label 1 zeigt, dass Labels auch über Bucketgrenzen hinweg liegen können. Label 2, 3 zeigen die Schnitte innerhalb eines Buckets. Label 4 – 7 zeigen die Schnitte im Bucket und den Nachbarbuckets.

wird. Um dies zu vermeiden genügt es, für ein aktives Bucket statt alle acht Nachbarbuckets auf Überlappungen zu prüfen nur die drei Buckets, wie in Abbildung 3.15 dargestellt, auf potentielle Schnittpartner zu betrachten.

Die Schnitte mit den anderen Labels der Nachbarbuckets werden betrachtet, wenn die jeweiligen Nachbarbuckets aktiv sind.

Für die Schnittberechnung ist es wichtig, dass die Überschneidungen in der richtigen Prioritätsreihenfolge berechnet werden. Geschieht dies nicht, kann es zur Auswahl der falschen maximalen Skalierung s_{max} , zu der ein Label sichtbar ist, kommen (vgl. Abbildung 3.16). Wird zum Beispiel erst der Schnittpunkt zwischen Label 1 und 2 berechnet, dann ergibt sich für das Label 2 $s_{max} = 1$. Danach wird das Label 1 von Label 0 auf $s_{max} = 0$ beschränkt. Da Label 1 in der Skalierungsstufe 1 nicht sichtbar ist, kann das Label 2 bereits ab der Skalierungsstufe 3 gezeichnet werden, ohne eine Überdeckung zu verursachen. Dieses Problem existiert auch über Bucketgrenzen hinweg. In den hier betrachteten Datensätzen tritt dieser Fall jedoch sehr selten auf.

3. Schnelle konsistente Kartenbeschriftung für interaktive Karten

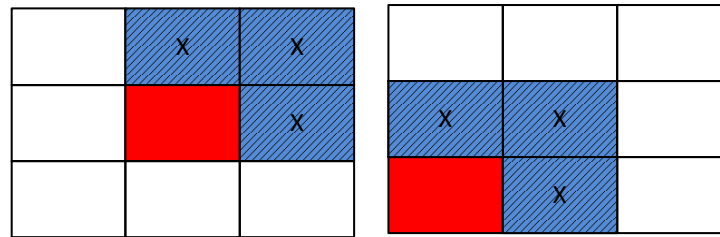


Abbildung 3.15.: Buckets die zur Überdeckungsrechnung geprüft werden müssen. Das aktive Bucket ist rot, die auf potentielle Schnittpartner zu überprüfenden Nachbarbuckets sind blau schraffiert dargestellt.

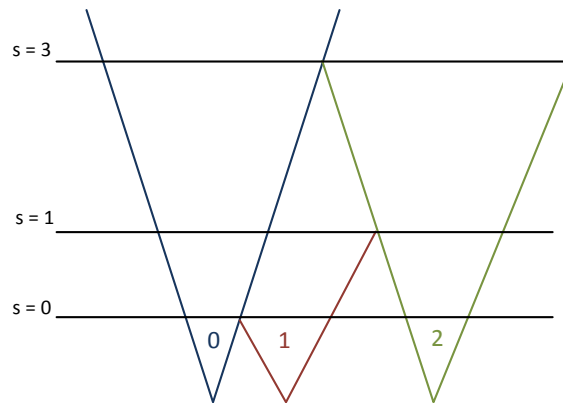


Abbildung 3.16.: Die maximale Skalierung zu der ein Label sichtbar ist muss in der Prioritätsreihenfolge berechnet werden. Werden erst Label 1 und 2 geschnitten, dann ist die maximale Skalierung von Label 2 nicht optimal.

Um alle Labels in richtiger Prioritätsreihenfolge auf Überlappung zu prüfen, kann eine Liste aller Schnittpaare erstellt werden, die dann anhand von Prioritäten sortiert wird. Dies erfordert allerdings einen sehr großen Platzbedarf, da jede Kombination gespeichert werden muss.

Eine Alternative ist die Erstellung einer Liste der Labels, die in den zu betrachtenden Buckets liegen und dann jedes Label mit jedem zu schneiden. Dabei werden allerdings auch Schnitte zwischen den Nachbarbuckets berechnet, die später noch einmal betrachtet werden. Damit keine Schnitte doppelt berechnet werden kann ein Konfliktgraph eingeführt werden. Die Knoten stellen die Labels in den betrachteten Buckets dar. Werden zwei Labels geschnitten, dann wird das niedriger Priorisierte in die Kantenliste des höher Priorisierten aufgenommen. Vor jedem Schnitt wird geprüft, ob dieses Labelpaar bereits betrachtet und somit schon im

Konfliktgraph gespeichert wurde. Nachteil dieses Konfliktgraphen ist ebenfalls ein hoher Platzbedarf.

Ein weiterer Ansatz, der hier aber nicht mehr implementiert wurde, benötigt weder zusätzlichen Speicher- noch Zeitaufwand durch Mehrfachschneidungen. Die Idee dabei ist, die einzelnen Buckets-Listen zu sortieren und dann jeweils die niedrigsten Labelpaare zwischen aktivem Bucket und Nachbarbucket zu schneiden. In Abbildung 3.17 wird das Verfahren dargestellt. Zunächst werden alle Listen des aktiven Buckets und der drei Nachbarbuckets sortiert. Ein Pointer wird auf die jeweils ersten Positionen der einzelnen Bucketlisten gesetzt. Danach wird das Minimum dieser Werte ausgewählt. Dies ist somit das höchst priorisierte Label, das als aktuelle Position gespeichert wird. Der Startpointer dieser Liste wird eine Position weiter gesetzt. Als nächster Schritt wird wiederum das Minimum der Pointer gesucht. Das Resultat ist das Label mit der zweithöchsten Priorität. Dieses Label wird mit dem vorher gespeicherten höher priorisierten Label geschnitten und der Pointer wieder um eine Position weiter gesetzt. Dies wird wiederholt, bis alle Listen ihr Ende erreicht haben. Dabei können Schnittpaare zwischen den Nachbarbuckets ignoriert werden.

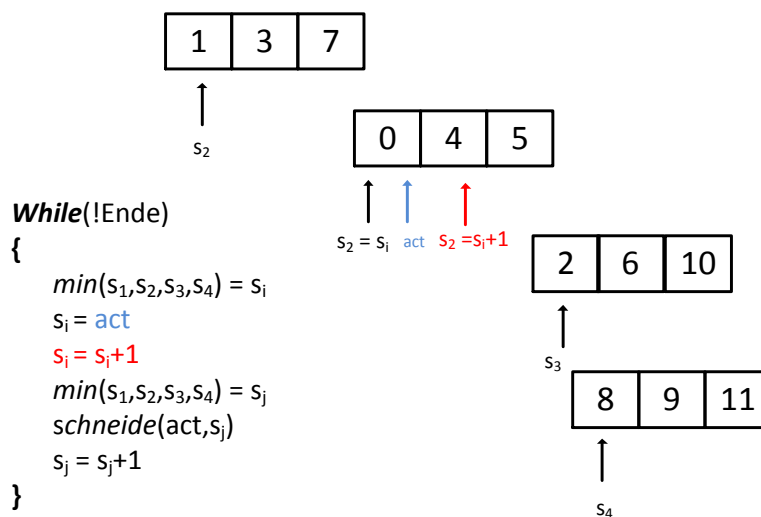


Abbildung 3.17.: Schnittberechnung in Prioritätsreihenfolge ohne zusätzlichen Speicher- und Zeitaufwand.

Die Buckets können dabei in einem zweidimensionalen Array abgespeichert werden. Damit ist der Zugriff auf die Labels beim Schnitttest sehr schnell, allerdings ist der Platzaufwand sehr groß. In dieser Implementierung konnte mit diesem Ansatz bei der Wahl der Höhe 7000 als kleinste LOD-Stufe schon nicht mehr die ganze Welt im Arbeitsspeicher gehalten werden. Eine Verbesserung liefert eine indizierte Datenstruktur, in der nur eine Liste erzeugt wird,

3. Schnelle konsistente Kartenbeschriftung für interaktive Karten

die nach den Bucketindizes sortiert wird. Dabei können Buckets, die keine Labels beinhalten, weggelassen werden (vgl. Abbildung 3.18).

Listenindex	0	1	2	3	4	...
Zeile	0	0	0	1	1	
Spalte	0	1	2	1	2	
Labelindex	0	5	14	1	312	

Abbildung 3.18.: Liste mit allen Bucketpositionen und Indizes der Labels. Buckets ohne Labels können dabei weggelassen werden.

Für diese Liste wird ein zusätzlicher Sortier- und Suchaufwand benötigt, der allerdings bei der Gesamtlaufzeit des Verfahrens nicht weiter ins Gewicht fällt. Eine weitere Möglichkeit die Bucketliste zu speichern ist eine Hash-Tabelle.

Mit dieser Unterteilung in Buckets muss also nicht mehr jedes Label mit jedem getestet werden, sondern nur noch diejenigen, die im selben Bucket oder in den unmittelbaren Nachbarbuckets liegen.

Damit die Seitenebenen und Eckgeraden, die für die Schnittberechnung benötigt werden, nicht in jedem Schritt neu berechnet werden müssen, werden diese einmal vorberechnet. So müssen sie später nur noch ausgelesen werden. Dies liefert eine verbesserte Laufzeit, allerdings auf Kosten eines Extra-Platzbedarfs. Bei sehr großen Labelmengen muss dieser Laufzeitvorteil aufgrund des hohen Platzbedarfs aufgegeben werden.

3.3. Vor- und Nachteile des Modells von Been et al.

Vorteile:

- Es können beliebige Labelformen dargestellt werden, sofern ein Schnittalgorithmus für jede Labelform-Kombination existiert.
- Die Darstellung ist frei von Artefakten wie zum Beispiel unerwartetes Erscheinen. So bekommt der Benutzer bei jeder Zoomstufe den Eindruck einer statischen Karte.
- In der Interaktionsphase fallen keine Berechnungskosten an.
- Ist unabhängig von der Art, welche bzw. wie viele Labelpositionen betrachtet werden.
- Filterungsoperation ist frei wählbar.

- Die Anzahl der zu beschriftenden Objekte ist nur durch den Arbeitsspeicher beschränkt, auf dem die Vorberechnung erfolgt.
- Pro Label muss zur Sichtbarkeitsbestimmung während der Interaktionsphase lediglich der Lebensbereich gespeichert werden. Betrachtet man die strikte Version der Desiderata reicht sogar die maximale Skalierung, zu der ein Label sichtbar ist. Dies eignet sich somit auch für Systeme, die wenig Ressourcen, wie zum Beispiel mobile Navigationssysteme

Nachteile:

- Durch das Konsistenz-Kriterium werden, von der gleichen Anzahl der zu beschriftenden Objekte ausgehend, wesentlich weniger Labels gezeichnet, als bei anderen Verfahren wie zum Beispiel von Mote, K. in "Fast point-feature label placement for dynamic visualizations" [Moto7].
- Hohe Vorverarbeitungskosten.
- Im Worst-Case-Fall ist die Laufzeit der Vorberechnung $\mathcal{O}(n^2)$. Dies ist vor allem bei einer großen Labelanzahl ein Nachteil gegenüber den Verfahren die zuerst Filtern.
- Da die Schnitte zwischen allen Labelpaaren betrachtet werden, müssen alle Daten geladen sein bzw. pro LOD-Stufe nachgeladen werden. Dies führt bei einer großen Labelmenge auch zu einem entsprechend großen Platzaufwand bzw. Zeitaufwand für das Nachladen.
- Es werden nur punkthafte Labels betrachtet.
- Keine Interaktion mit dem Benutzer möglich. Sollen zum Beispiel Labelprioritäten während der Interaktionsphase geändert werden (der Benutzer wechselt zum Beispiel von Auto- auf Fahrradkarten), müssen alle Überlappungsberechnungen erneut durchgeführt werden.

4. 3D-Kartenbeschriftung

4.1. Anforderungen an eine 3D-Kartenbeschriftung

4.1.1. Erweiterter Konsistenzbegriff durch Rotieren um die Hochachse

Um visuell ansprechende Beschriftungen für 3D-Karten zu erzeugen, müssen die Benutzerinteraktionen Rotieren um die Hochachse und Kippen, sowie deren Einfluss auf die Verdeckung von Beschriftungen betrachtet werden.

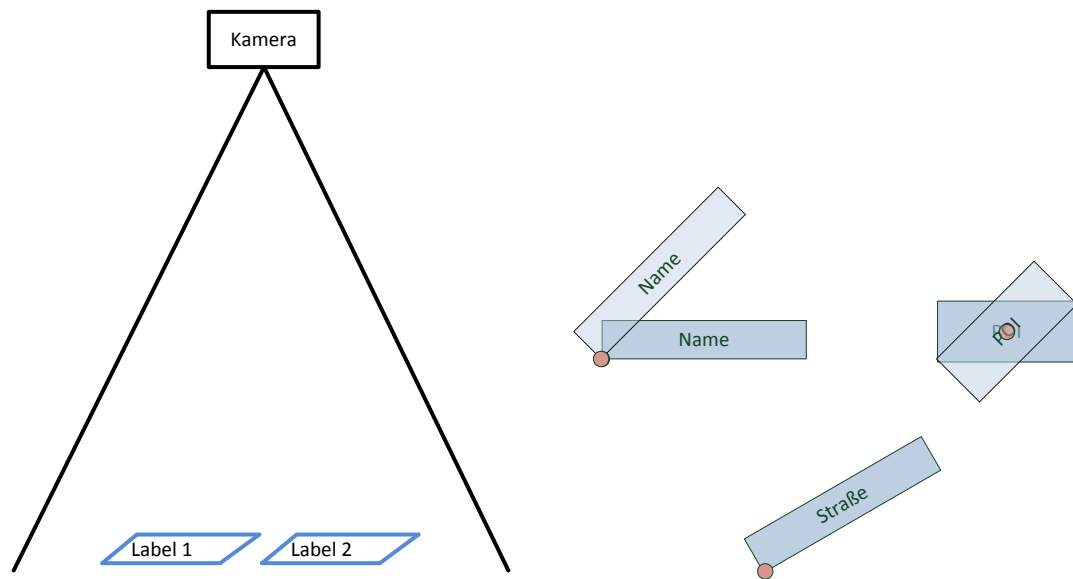
Damit Beschriftungen für den Benutzer immer lesbar bleiben, drehen sich Labels mit der Kamera, so dass sie immer achsenparallel zur Blickrichtung sind (vgl. Abbildung 4.1). Straßen-Labels bilden in dieser Implementierung eine Ausnahme. Sie drehen sich nicht mit der Kamera, sondern bleiben der Straße angeheftet, die sie beschriften.

Der neue Freiheitsgrad des Rotierens führt dazu, dass weitere Überdeckungsfälle betrachtet werden müssen. Abbildung 4.2 zeigt die Möglichkeiten, wie sich Labels unter der Operation Rotieren verhalten und wie es zu Überschneidungen kommen kann. Die Fälle (a), (c) und (e) beschreiben dabei die Labelplatzierung vor einem Rotationsschritt. Da sich zwei Beschriftungen (ausgenommen Straßen-Labels) immer um den gleichen Winkel drehen, kommt es in (b) zu keiner Überlappung. Im zweiten Fall (d) liegen zwei Labels so, dass sie sich durch die Rotation schneiden. Der dritte Fall ist der Schnitt zwischen Straßen-Labels und gedrehten Beschriftungen.

Da es zu neuen Verdeckungsmöglichkeiten kommt, muss auch der Konsistenzbegriff wie er in 3.1.1 definiert wird, auf die erweiterten Anforderungen angepasst werden. Abbildung 4.3 zeigt die Darstellung von drei Labels zu jeder Rotationsmöglichkeit. Im Folgenden wird angenommen, dass das Label mit der Priorität 0 die größte Wichtigkeit besitzt. Die rote Markierung der Label 1 und 2 stellen die Winkel dar, bei denen eine Überdeckung mit dem Label 0 vorliegt. Die blaue Markierung bei Label 2 zeigt den Schnitt dieser Beschriftung mit dem Label 1.

Es gibt nun drei Möglichkeiten, zu welchen Winkeln die Labels gezeichnet bzw. verdeckt werden:

- Strikte Konsistenz: Labels die bei irgendeinem Winkel von einem höher priorisierten Label verdeckt werden, werden nicht gezeichnet. In diesem Beispiel wird nur das Label 0 dargestellt.



Verhalten von Beschriftungen bei Kamerarotation: Der rote Punkt gibt die Invariante an. Straßen-Labels ändern sich nicht mit der Kamerarotation, alle anderen Labelarten drehen sich um ihre Invariante, so dass sie stets achsenparallel zur Blickrichtung sind.

Abbildung 4.1.: Links: Kamerasicht auf die Labels. Rechts:

- Gelockerte Konsistenz: Labels werden zu den Winkeln gezeichnet, bei welchen sie nicht durch eine höher priorisierte Beschriftung verdeckt werden (unabhängig davon ob diese zu dem Winkel selbst sichtbar sind oder nicht). Damit ist Label 1 während des rot markierten Bereichs und Label 2 während des blau markierten Bereichs nicht sichtbar. Für alle anderen Winkel werden die beiden Labels gezeichnet.
- Eingeschränkte Konsistenz: Jedes Label wird genau dann gezeichnet, wenn es nicht durch ein sichtbares, höher priorisiertes Label verdeckt wird.

In 2D kann durch die in Been et al. [BDY06] gegebenen Desiderata eine brauchbare Konsistenz definiert werden. In 3D ist diese Definition schwieriger. Alle drei vorgestellten Möglichkeiten haben ihre Vor- und Nachteile. Bei der strikten Konsistenz ist sichergestellt, dass kein Label unerwartet aufpoppt. Den Nachteil dieser Betrachtungsweise zeigt Abbildung 4.4 (a): ein breites, hoch priorisiertes Label erzeugt einen großen Radius bei der Rotation und verdeckt somit viele andere Labels, die dann niemals gezeichnet werden. Das entstehende Beschriftungsbild wird dadurch sehr spärlich und genügt nicht mehr den Anforderungen einer guten Beschriftung. Wird diese Bedingung auch noch für alle Zoomstufen gefordert, bleibt am Schluss nur noch ein Label übrig. Die zweite Möglichkeit liefert ein besseres Ergebnis. Jedoch zeigt Abbildung 4.4 (b) den Fall, dass durch Rotation eine Kette von Labels entstehen kann, die sich gegenseitig überdecken. Dadurch wird in diesem Beispiel nur noch

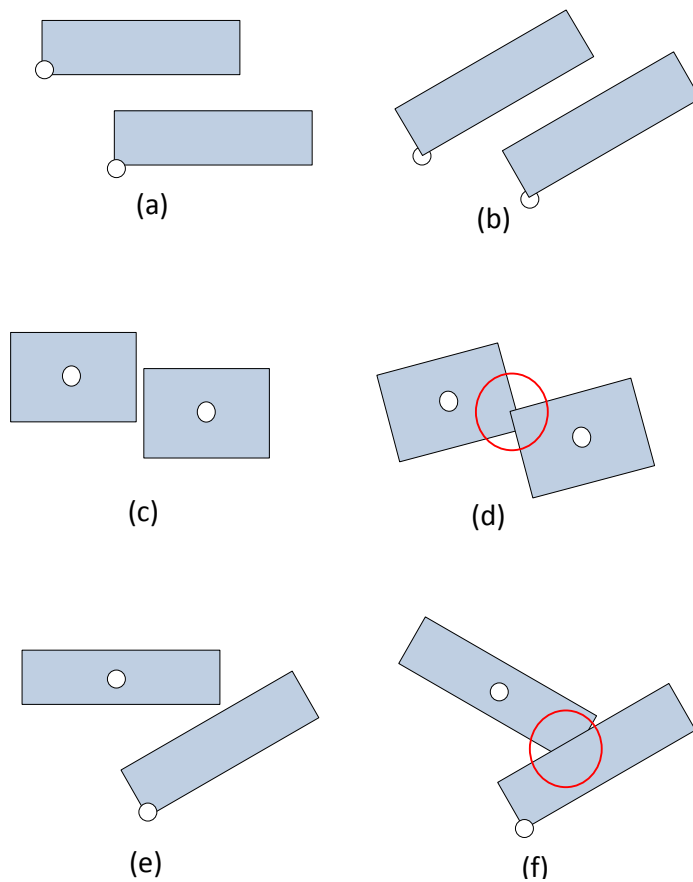


Abbildung 4.2.: (a), (c), (e) zeigen jeweils die Ausgangssituation zweier Labels. (b), (d), (f) zeigen die Lage der Labels nach einer Kamerarotation.

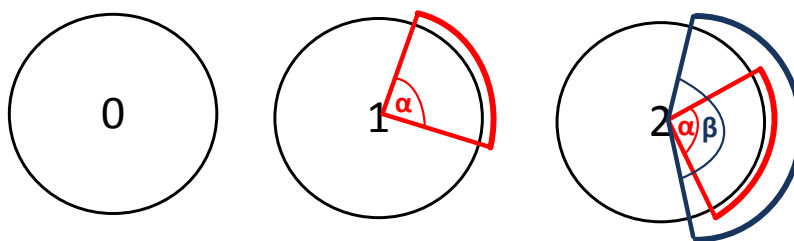


Abbildung 4.3.: Ein Kreis repräsentiert eine Label-Darstellung zu jeder möglichen Rotation. Rot wird der Winkelbereich dargestellt, in dem Label 0 (höchste Priorität) Label 1 bzw. Label 2 schneidet. Blau ist der Winkelbereich, in dem Label 1 und 2 sich überlappen.

4. 3D-Kartenbeschriftung

Label 1 gezeichnet, da alle anderen durch ein sichtbares höher priorisiertes Label verdeckt sind. Auch hierdurch entsteht unter Umständen ein Kartenbild, das wenig Beschriftungen enthält. Bei der dritten Möglichkeit der Label-Beschriftung, sind zu jedem Winkel, unter der Bedingung, dass kein Label von einem höher priorisierten Label verdeckt wird, maximal viele Labels sichtbar. Allerdings können bei diesem Ansatz Artefakte auftreten. Das Beispiel in Abbildung 4.4 (c) zeigt, wie das Label 2, das zunächst nicht sichtbar ist, erscheint, sobald das Label 1 bei einer Rotation der Kamera nach rechts durch das 0-te Label verdeckt wird und wieder verschwindet, wenn man die Kamera weiter dreht.

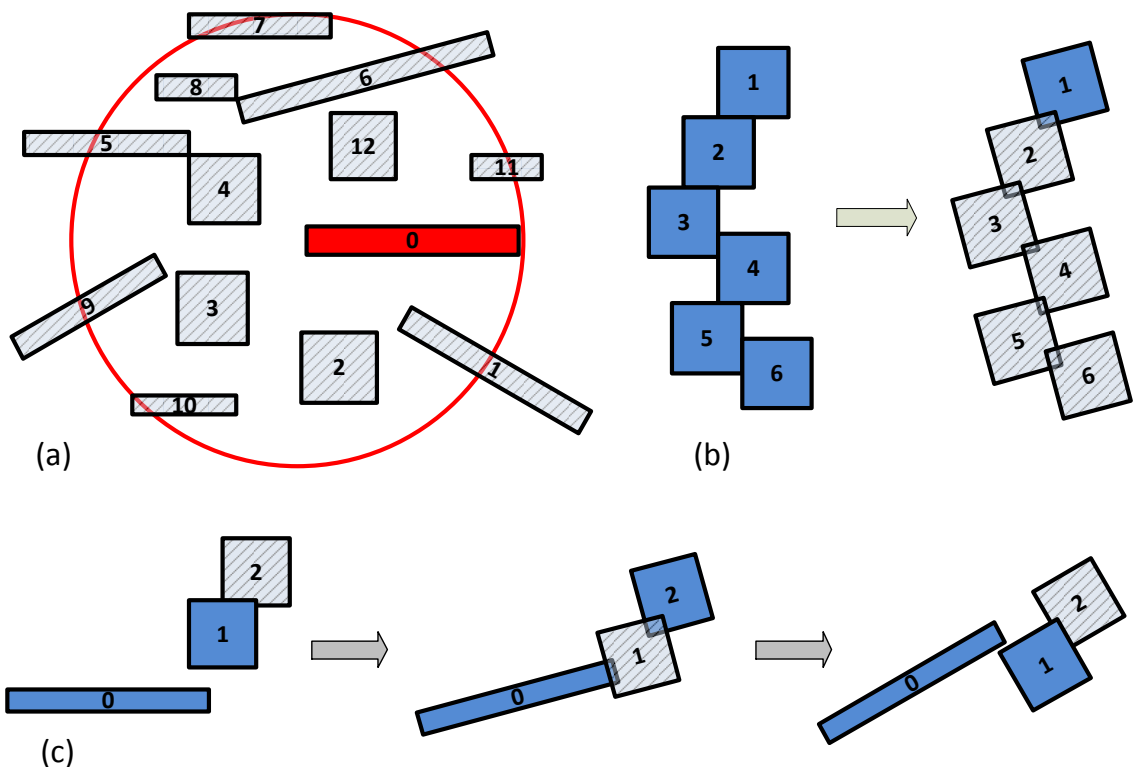


Abbildung 4.4.: (a) Beim strikten Konsistenzbegriff kann ein hoch priorisiertes Label viele andere verdecken. (b) Bei der gelockerten Konsistenzbetrachtung können sich Labels in einer Kette verdecken, so dass nur wenig Labels sichtbar sind. (c) Maximale Platzausnutzung auf Kosten von Artefakten wie unerwartetes Erscheinen.

Welche der drei Möglichkeiten nun das beste Ergebnis liefert, wird verfahrensabhängig in folgendem Abschnitt 4.2 diskutiert.

4.1.2. Gekippte Kartendarstellungen

Wie sich Labels bei der zweiten Benutzerinteraktion Kippen, die bei 3D-Karten als neuer Freiheitsgrad hinzu kommt, verhalten, wird in Abbildung 4.5 illustriert. Bisher wurde die Karte immer von oben betrachtet (vgl. Abbildung 4.5 links). Wird die Kamera gekippt, werden auch die Labels gekippt, so dass diese immer achsenparallel zur Blickrichtung ausgerichtet sind und so vom Benutzer lesbar bleiben. Wie schon bei der Rotation bilden Straßen-Labels in dieser Implementierung eine Ausnahme und bleiben flach den Straßen angeheftet.

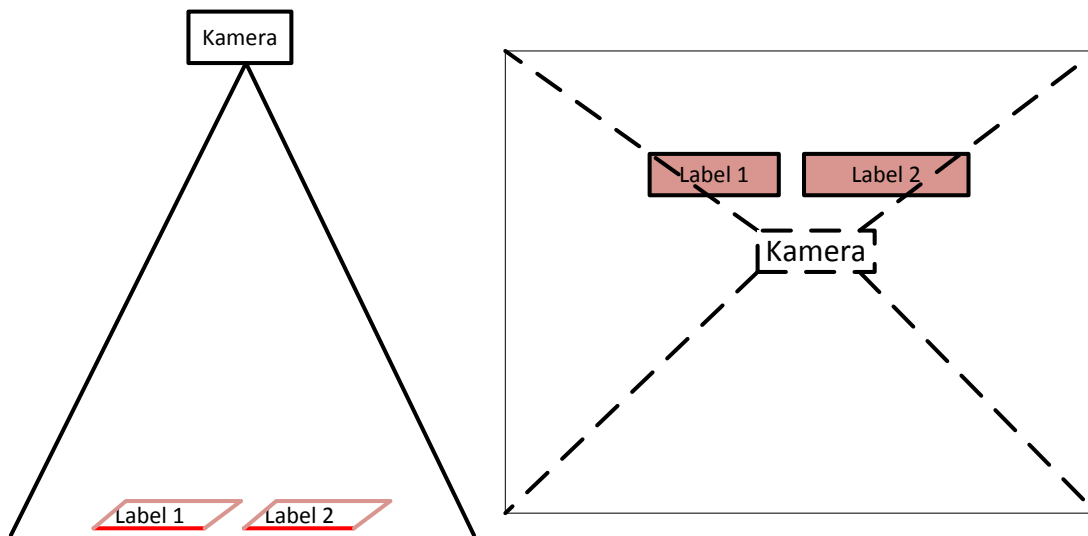


Abbildung 4.5.: Links: Betrachtung der Labels von oben. Rechts: Betrachtung der Labels von vorne durch Kippen der Kamera. Die Labels werden um die im linken Bild rot dargestellte Achse gedreht, so dass sie immer achsenparallel zur Blickrichtung sind.

Beim Kippen schneiden sich zwei Labels genau dann, wenn ihre perspektivische Projektion auf die Bildschirmfläche eine Überschneidung hat. Wird die Kipp-Operation, für Labels die auf dem Boden liegen, unabhängig von allen anderen Benutzerinteraktionen betrachtet, dann gilt: wird eine Beschriftung durch eine andere zum Winkel α verdeckt, dann gilt dies auch für jeden Winkel $\beta > \alpha$. Mit dieser Eigenschaft ist auch das Konsistenzverhalten einfach zu definieren: Sobald ein Label von einem höher priorisierten Label verdeckt ist, wird es nicht gezeichnet.

Bei 3D-Karten kann das Kippen jedoch nicht separat von allen anderen Interaktionsmöglichkeiten betrachtet werden. Durch die Kombination dieser Interaktionen entstehen neue Überdeckungsmöglichkeiten wie sie Abbildung 4.6 zeigt. Überschneidungen von gekippten

Labels werden durch das Kippen (a), Rotieren (b) und Verschieben (c) der Kamera erzeugt bzw. gelöst.

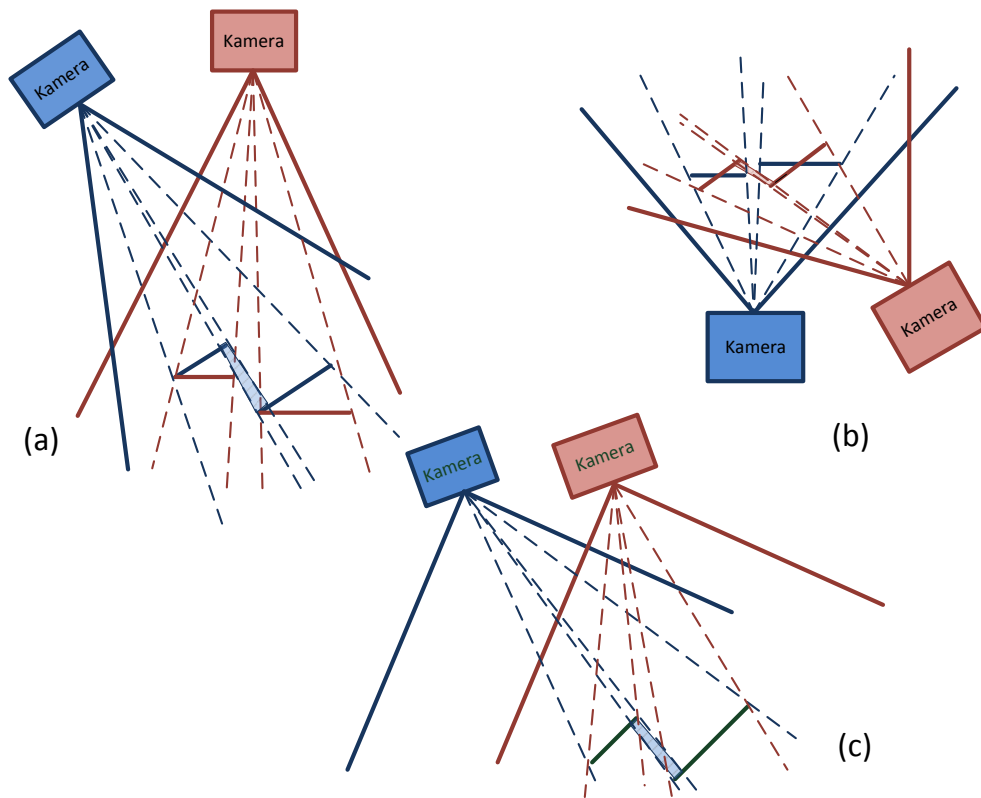


Abbildung 4.6.: Die blaue Kamera und der Sichtbereich ist die Ausgangssituation zur Darstellung eines gekippten Labels. Die rote Kamera erzeugt die Darstellung, welche durch (a) Kippen der Kamera, (b) Rotation der Kamera und (c) Verschieben der Kamera entstehen. Der schraffierte Bereich zeigt dabei die Überschneidungen.

Die Änderung des Öffnungswinkels ist analog zum Verschieben der Kamera nach hinten bzw. vorne und kann deshalb ebenfalls zu einer Veränderung der Überdeckungen führen.

Die Vielfalt der oben gezeigten Änderungsmöglichkeiten der Labelüberdeckung durch die Freiheitsgrade Rotieren, Verschieben, Kippen und Zoomen weist darauf hin, wie schwer es ist, einen Konsistenzbegriff für die Kombination aller Interaktionsmöglichkeiten zu formulieren. Bisher war die Verdeckungsrechnung unabhängig von der Position oder dem Öffnungswinkel der Kamera. Wird Kippen mit in Betracht gezogen, werden diese Informationen zur Schnittberechnung benötigt. Wird analog zur Rotation eine strikte Konsistenz definiert, kann nur ein Label gezeichnet werden, da es durch Rotation und Kippen immer

möglich ist, das höher priorisierte Label mit allen anderen zu schneiden. Betrachtet man die gelockerte Version der Konsistenz, kann es bereits, wie es Abbildung 4.7 vorführt, zu Artefakten kommen.

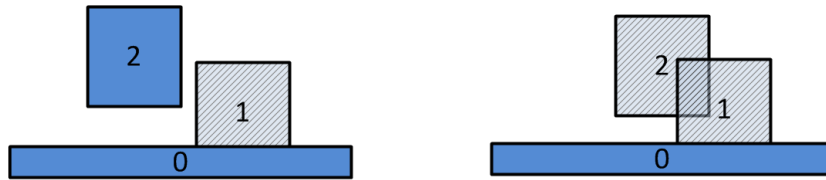


Abbildung 4.7.: Beim gelockerten Konsistenzbegriff können Artefakte auftreten. Links: Label in Bildschirmkoordinaten. Label 0 (höchst Priorisiertes) und 2 sind sichtbar. Label 1 ist durch den Konflikt mit Label 0 verdeckt. Rechts: Die Kamera wird nach rechts gedreht, dadurch verschwindet Label 2 durch den Konflikt mit Label 1. Für den Benutzer ist nicht ersichtlich, warum Label 2 verschwindet.

Links wird der aktuelle Bildschirmausschnitt von drei Beschriftungen gezeigt. Sichtbar sind das Label 0 und 2. Wird die Kamera nach rechts gedreht, überlappen sich Label 1 und 2. Nach der Konsistenzbetrachtung wird Label 2 nun nicht mehr gezeichnet. Aus Benutzersicht verschwindet dieses Label ohne Grund. Der Konflikt mit Label 1 kann nicht nachvollzogen werden, da der Benutzer durch die Überschneidung mit Label 0 das Label 1 nie gesehen hat. Obwohl die Konsistenzeigenschaft erfüllt ist, kommt es zu einem für den Benutzer unerwünschten Verhalten. Dieses Verhalten ist auch in der gelockerten Konsistenzbetrachtung während der Rotation um die Hochachse zu beobachten. Jedoch tritt es dort wesentlich seltener auf als bei der Kipp-Operation.

Ein weiteres Problem bei der Betrachtung der gelockerten Konsistenz zeigt Abbildung 4.8. Bei der Konsistenzbetrachtung soll vermieden werden, dass durch Verschieben der Kamera Labels aufpoppen. Dies ist allerdings bei Einführung des Kipp-Operators nicht mehr erfüllbar. Man müsste alle möglichen Überdeckungen ohne Betrachtung des sichtbaren Bereichs berücksichtigen. Fall (a) zeigt, wie sich die drei Labels nicht schneiden, wenn man die Überdeckungen von einer fixen Kameraposition aus berechnet. Verschiebt man nun die Kamera wie in (b) nach links, dann überdecken sich Label 2 und 1 und Label 1 verschwindet. Wird die Kamera weiter nach links geschoben, taucht es wieder auf, da Label 1 durch Label 0 verdeckt wird. Eine Möglichkeit der Lösung dieses Problems, ist die Kamera weit genug nach hinten zu verschieben, so dass die Projektionsstrahlen fast parallel verlaufen. Dabei entsteht allerdings ein Fehler in der Überdeckungsberechnung. Desweiteren kann damit nicht der Fall des Aufpoppens wie er in Abbildung 4.6 (c) dargestellt ist, vermieden werden.

Die dritte Möglichkeit der eingeschränkten Konsistenz, erzeugt dasselbe unerwünschte Erscheinen und Verschwinden von Beschriftungen wie bei der Rotation. Welche Einschränkungen in Kauf genommen werden, wird bei den Verfahren in Abschnitt 4.2 diskutiert.

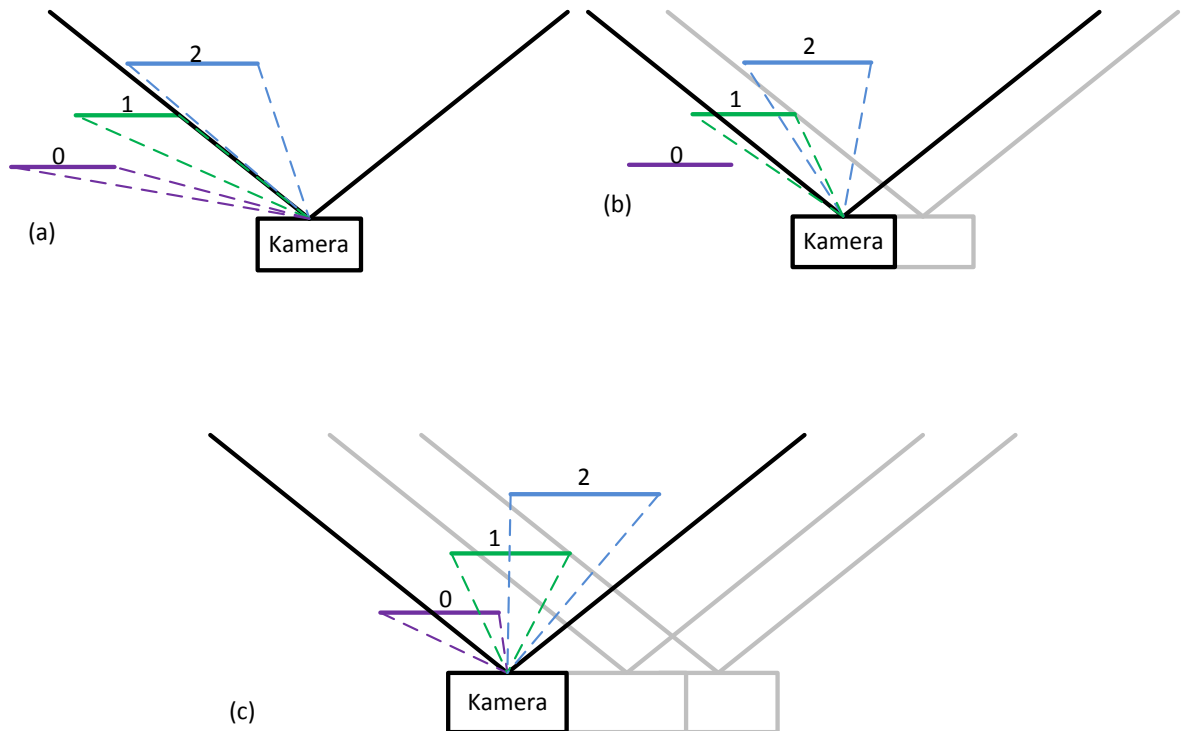


Abbildung 4.8.: (a) Betrachtung der Überschneidungen ohne sichtbaren Bereich zu berücksichtigen. (b) Wird die Kamera nach links geschoben, überdecken sich das blaue und das grüne Label. Das Grüne wird aufgrund der niedrigeren Priorität nicht angezeigt. (c) Wird die Kamera weiter nach links geschoben, taucht das blaue Label wieder auf, da das grüne von dem höher priorisierten lilafarbenen Label verdeckt wird.

4.1.3. Gekippte Kartendarstellungen mit Geländemodellen

Um eine 3D-Karte auch wirklich zu beschriften, müssen Höheninformationen mit in Betracht gezogen werden. Dabei werden Beschriftungen auf der Höhe ihrer Invariante gezeichnet. Beim Zoomen, Verschieben und Rotieren haben diese zusätzlichen Informationen keine Auswirkungen. Beim Kippen müssen jedoch zusätzlich Labelüberdeckungen auf unterschiedlichem Höhenniveau berechnet werden. Dies ändert jedoch nichts an der Konsistenzbetrachtung für gekippte Kartendarstellungen (vgl. Abbildung 4.9

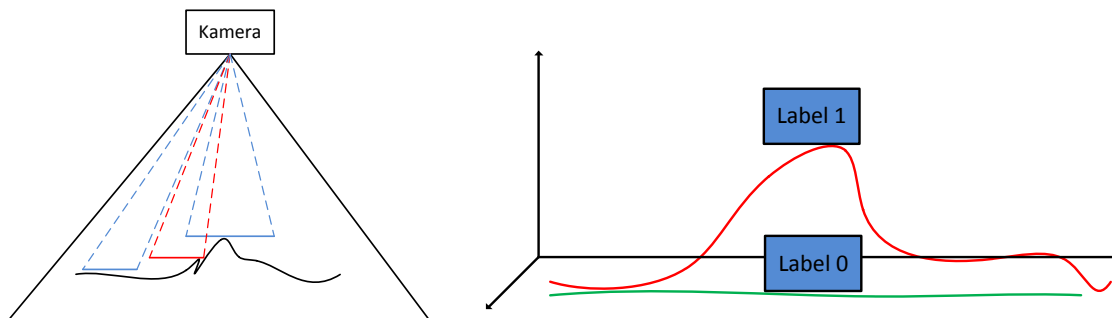


Abbildung 4.9.: Links: Bei der Draufsicht auf die Labels besteht kein Unterschied in der Verdeckungsberechnung. Die Höhenkoordinaten werden für die Berechnung nicht betrachtet. Die blauen Labels sind sichtbar. Das rote Label wird verdeckt. Rechts: die Verdeckungsberechnung ist von der Höhe abhängig. Die beiden Labels haben den selben x-Wert und würden sich ohne Betrachtung der Höhe überschneiden. Durch Betrachtung der Höheninformationen sind aber beide sichtbar.

4.2. Ansätze für 3D-Kartenbeschriftungen

4.2.1. Erweiterung des Ansatzes von Been et al.

Im vorherigen Abschnitt wurde das Verhalten von Beschriftungen unter Rotations- und Kippoperationen sowie die veränderten Anforderungen an einen Beschriftungsalgorithmus beschrieben. Im Folgenden werden verschiedene Ansätze aufgezeigt, wie die in Abschnitt 3.2 erzeugte Beschriftung auf 3D-Karten erweitert werden kann.

Die erste Idee für ein Verfahren ist, den Ansatz von Been et al. [BDYo6] zu erweitern. In diesem Modell wird die Skalierung als dritte Dimension verwendet. Dadurch entstehen im Fall von rechteckigen punkthaften Beschriftungen Pyramiden. Die Schnitte dieser Pyramiden ergeben die Überschneidung der Labels in Abhängigkeit der Skalierung. Analog dazu können die neuen Freiheitsgrade Rotation und Kippen als weitere Dimensionen betrachtet werden.

Pyramiden lassen sich in einer geschlossenen Formel darstellen:

$$|x| + |y| = z$$

Die Gleichung definiert die Mantelfläche einer Pyramide, die ihre Spitze im Ursprung hat und deren Schnitt bei $z = 1$ ein Quadrat mit den Ecken $P_0 = (1, 0, 1)$, $P_1 = (0, 1, 1)$, $P_2 = (-1, 0, 1)$ und $P_3 = (0, -1, 1)$ ergibt. Mit einer affinen Transformation

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = T \cdot \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

kann, wie folgendes Beispiel zeigt, eine beliebige Pyramide erzeugt werden:

Die Transformation

$$T = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 3 \\ 0 & 1 & 0 \end{pmatrix}$$

führt zu der Pyramidengleichung

$$|x| + |y| = z \rightarrow |x'| + |z'| = \frac{1}{3} y'$$

Damit wird eine Pyramide erzeugt, deren Spitze im Ursprung liegt und deren Eckpunkte bei einem Schnitt durch $z = 6$ die Punkte $P_1 = (0, 6, -2)$, $P_2 = (-2, 6, 0)$, $P_3 = (0, 6, 2)$, $P_4 = (2, 6, 0)$ ergeben.

Das Gleichsetzen und Lösen dieser Gleichung für Pyramidenpaare liefert die Schnittpunkte. Zusätzlich zu dieser hier angewandten direkten Lösung besteht auch die Möglichkeit, die Geometrie der Pyramide zu nutzen. Wie in Abschnitt 3.2 beschrieben, wird die Pyramide in Ebenen und Geraden zerlegt und diese auf Schnitte überprüft.

Wird die Interaktion um zwei weitere Dimensionen (Rotation, Kippen) erweitert, entsteht aus der Pyramide ein 5D-Objekt. Die geometrische Form dieses Objektes ist nicht mehr vorstellbar. Um die Form dennoch für die Schnittberechnung nutzen zu können, kann die konvexe Hülle des Objektes berechnet werden. Ludwig vergleicht in "Convexity and Computation" [Lud09] verschiedene Verfahren zur Erzeugung einer konvexen Hülle aus sample-Punkten. Für den zwei- bzw. dreidimensionalen Fall liefern diese Verfahren schnelle Laufzeiten, im fünfdimensionalen Fall wird dagegen die Berechnungszeit sehr schlecht. Diese Zeit hängt von der Anzahl der Punkte in der Menge ab, aus der die konvexe Hülle berechnet wird. Beim 5D-Objekt ist die benötigte Anzahl an Punkten nur schwer festzustellen, die nötig sind, um die Geometrie des Objektes exakt genug zu erfassen, um Schnittberechnungen durchführen zu können. Da die Hülle für jedes Label berechnet werden müsste, scheint dieser Ansatz unpraktikabel. Außerdem kann nicht garantiert werden, dass das 5D-Objekt konvex ist. Der mögliche Fehler, der dann durch die Approximation auf die konvexe Form entsteht, führt unter Umständen zu einer Beschriftung, die den Anforderungen nicht gerecht wird.

Eine weitere Möglichkeit besteht darin, eine geschlossene Formel aufzustellen. Zunächst wird nur die Rotation als zusätzliche Dimension betrachtet. Für den in Abbildung 4.10 dargestellten Schnittfall werden Orts- und POI-Labels (keine Straßenlabels) betrachtet, die sich um den selben Winkel drehen. Mit dem hier vorgestellten Verfahren kann der Anfangswinkel bestimmt werden zu dem sich zwei Labels schneiden.

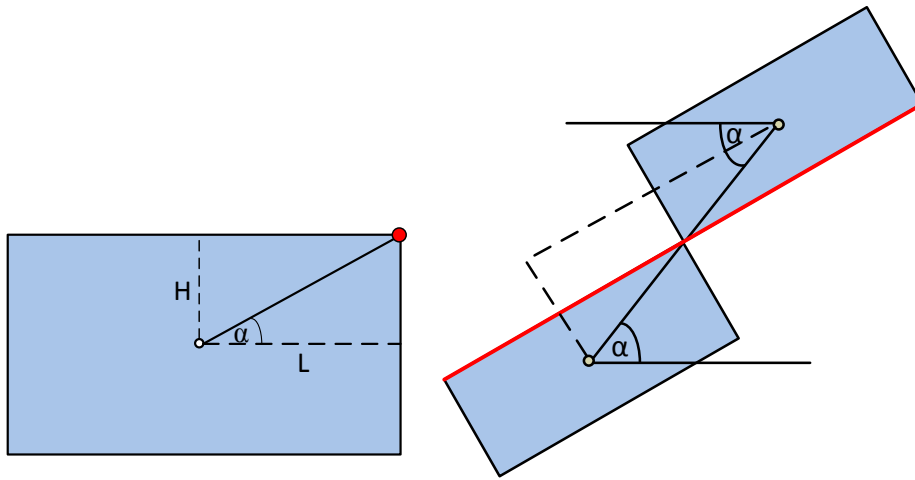


Abbildung 4.10.: Links: durch die Länge L, die Breite H des Rechtecks, und den Winkel α lässt sich die Position des rechten oberen Eckpunktes bestimmen. Rechts: Bei der Rotation lassen sich die Schnitte von zwei Rechtecken auf den dargestellten Fall reduzieren.

Abbildung 4.10 zeigt links wie ein Eckpunkt des Rechtecks (in diesem Fall der rot markierte rechte obere Punkt) zu jedem Rotationswinkel dargestellt werden kann:

$$p = L \cdot \vec{v} + H \cdot \vec{u}$$

mit

$$\vec{v} = \begin{pmatrix} \cos \alpha \\ \sin \alpha \end{pmatrix}, \vec{u} = \begin{pmatrix} -\sin \alpha \\ \cos \alpha \end{pmatrix}$$

Betrachtet werden alle Labels zur Kamerarotation Null. Wenn sich die Labels drehen, dann ist die notwendige Bedingung für den ersten Schnittpunkt, dass die oberen bzw. unteren Kanten der Rechtecke in einer geraden Linie liegen (siehe rote Markierung in Abbildung 4.10 rechts). Die hinreichende Bedingung ist, dass die Summe der Abstände des oberen Eckpunkts des linken Rechtecks und des unteren Eckpunkts des rechten Rechtecks zu ihren jeweiligen Mittelpunkten größer als der Abstand der Mittelpunkte sein muss:

$$s \cdot \sqrt{(l_1 \cdot \vec{v}_1)^2 + (h_1 \cdot \vec{u}_1)^2} + s \cdot \sqrt{(l_2 \cdot \vec{v}_2)^2 + (h_2 \cdot \vec{u}_2)^2} \geq d(m_1, m_2)$$

4. 3D-Kartenbeschriftung

Dabei ist s die Skalierung und $d(m_1, m_2)$ der Abstand der Mittelpunkte m_1, m_2 .

Bei POIs ist der Mittelpunkt die Invariante, bei Straßen- und Namen-Beschriftungen ändert sich der Mittelpunkt unter der Skalierung. Deshalb spielt bei Nicht-POIs die Skalierung auch auf der rechten Seite der Formel eine Rolle. Für POI- und Namen-Labels, die sich immer um den selben Winkel drehen, gilt: $\vec{u}_1 = \vec{u}_2$ und $\vec{v}_1 = \vec{v}_2$.

Um eine überlappungsfreie Beschriftung für jede Skalierung und jeden Winkel zu erhalten, muss diese Formel für jedes Labelpaar gelöst werden. Bereits für die Freiheitsgrade Zoomen und Rotieren ist diese Berechnung sehr komplex. Deshalb wurden noch weitere Ansatzmöglichkeiten betrachtet.

4.2.2. Sampling

Eine weitere Ansatzmöglichkeit ist, die Überschneidungen zu diskreten Rotationswinkeln und Skalierungsstufen in der Vorberechnungsphase zu berechnen und in der Interaktionsphase zwischen diesen Zuständen zu interpolieren. Für die Überdeckungsberechnung wird zunächst der gelockerte Konsistenzbegriff betrachtet. Abbildung 4.11 zeigt, dass bei dieser Konsistenz-Betrachtung relativ wenig Labels aufgrund von Verdeckungen, wie in Abbildung 4.4 dargestellt, zu sehen sind.

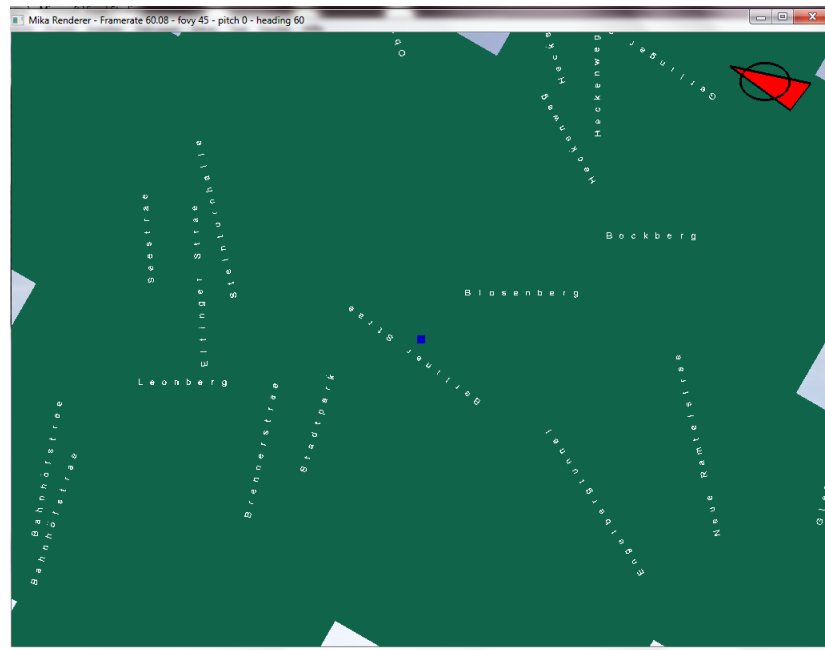
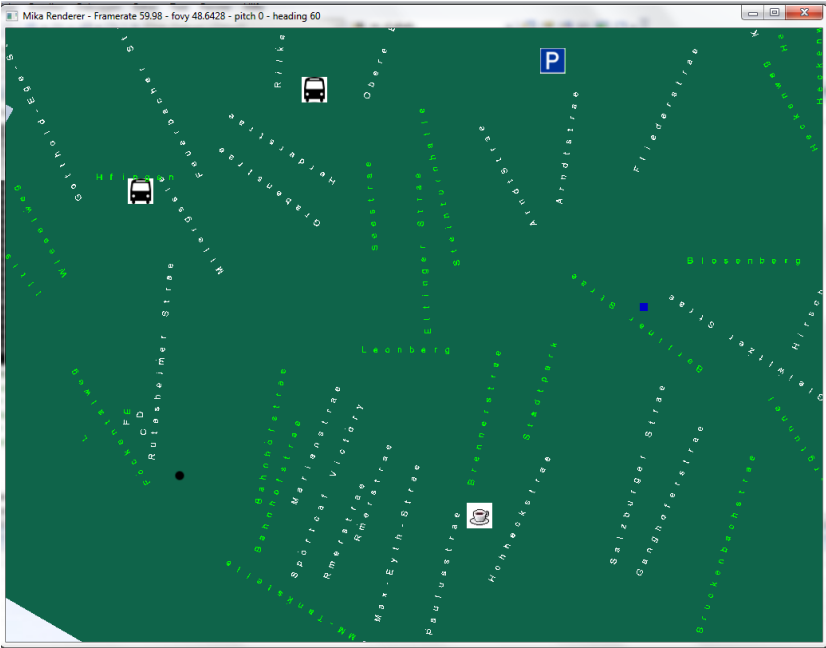


Abbildung 4.11.: Beschriftungen mit dem gelockerten Konsistenzbegriff.



Gegensatz zur Betrachtung der gesamten Labelmenge, gering genug, um Überdeckungen in der Interaktionsphase zu berechnen. In Abschnitt 2.2.4.2 werden zwar schnelle Algorithmen für die Schnittberechnung in der Interaktionsphase vorgestellt, jedoch gelten diese meist nicht für beliebig große, gedrehte Labels.

Betrachtet man die Konsistenzdiskussion für die Kipp-Operation wird deutlich, dass keine der oben definierten Konsistenzen, Artefakte vermeiden. Die Schnittberechnung basiert auf der Menge der Labels, die im Sichtbereich liegen. Deshalb ist es hier sinnvoll, statt aufwendige Schnittberechnungen (durch eine geschlossene Formel oder diskrete Schritte) in der Vorverarbeitungsphase zu berechnen, diese in die Interaktionsphase zu verlagern. Würde man zum Beispiel die Kippverdeckung durch diskrete Zustände berechnen, müsste jede mögliche Kameraposition, Skalierung und Rotation in Kombination durchgerechnet werden. Dagegen kann während der Interaktionsphase die Menge der Beschriftungen, die auf Überdeckung geprüft werden muss, durch Filteroperationen wie Sichtbarkeit und Rotationsverdeckung wesentlich reduziert werden.

Der Ansatz der Schnittberechnung in der Interaktionsphase hat zudem den Vorteil, dass der Öffnungswinkel, welcher bisher nicht in Betracht gezogen wurde, ebenfalls frei wählbar ist.

Bisher wurden nur die beiden Benutzer-Interaktionen Rotieren und Kippen diskutiert. Um eine Beschriftung für eine 3D-Karte zu erhalten, müssen jedoch auch Labels auf unterschiedlichem Höhenniveau betrachtet werden. Wird die Überdeckungsberechnung in der Interaktionsphase durchgeführt, ist die Erweiterung um Höheninformation einfach und verändert den Schnittalgorithmus nur gering. Eine Erweiterungsmöglichkeit wie sie auch schon in der Aufgabenstellung gegeben wurde, ist ein verändertes Geländemodell, wie zum Beispiel uniforme Skalierung der Berge, nichtuniforme Skalierung oder Verzerrung zuzulassen. Bei der Anwendung des Verfahrens die Schnittberechnung zur Laufzeit durchzuführen, ist durch diese Erweiterung keine Änderung an den Überdeckungsberechnungen nötig. Der Terrainrenderer von Vaaraniemi [Vaa08] bietet die Möglichkeit das Terrain durch einen Lupeneffekt zu plätten. Die Höhe eines Labels wird dann durch einen Faktor herabgesetzt.

4.3. Algorithmische Betrachtung der implementierten Verfahren

Aus den vorgeschlagenen Möglichkeiten, eine überlappungsfreie Beschriftung für 3D-Karten zu erstellen, wurden zwei Varianten implementiert. In der ersten Variante wurde die Rotation in diskreten Schritten berechnet. Nachdem dies jedoch zu keinem guten Ergebnis führte, wurde diese Diskretisierung für gekippte Beschriftungen nicht weiter implementiert. Bei der zweiten Implementierung wurde die Überschneidungsberechnung in die Interaktionsphase verlagert.

Die Schnittberechnung für die Rotation unterscheidet sich in beiden Verfahren nur darin, dass im ersten Fall die diskreten Werte durchlaufen und im zweiten Fall die aktuellen Werte abgefragt werden.

Der Schnittelalgorithmus, der bereits für die Implementierung wie in Abschnitt 3.2.2 beschrieben wurde, kann hier nicht verwendet werden. In diesem Algorithmus wird der Fall, dass sich zwei Rechtecke durchschneiden, d.h. kein Eckpunkt des einen Rechtecks im Rechteck des Anderen liegt, nicht erkannt (vgl. Abbildung 3.8 (d)). Betrachtet man nur diskrete Schritte, kann dieser Fall vorkommen, weshalb der Schnittelalgorithmus angepasst werden muss.

Um, wie in Abbildung 4.13 dargestellt, alle Schnittfälle abzudecken, werden die Seiten des Rechteckes in Geraden dargestellt und diese Seitengeraden auf Überschneidung getestet. Ob ein Schnittpunkt gültig ist, kann analog zum Verfahren in Abbildung 3.12 berechnet werden.

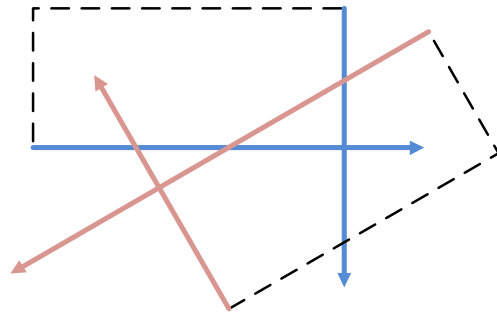


Abbildung 4.13.: Überdeckungserkennung durch Schneiden der Seitengeraden.

Für jedes Geradenpaar (g, h) berechnet sich der Schnittpunkt folgendermaßen:

$$g : \begin{pmatrix} p_{1,x} \\ p_{1,y} \end{pmatrix} + t \cdot \begin{pmatrix} r_{1,x} \\ r_{1,y} \end{pmatrix}$$

$$h : \begin{pmatrix} p_{2,x} \\ p_{2,y} \end{pmatrix} + s \cdot \begin{pmatrix} r_{2,x} \\ r_{2,y} \end{pmatrix}$$

$$p_{1,x} + t \cdot r_{1,x} = p_{2,x} + s \cdot r_{2,x}$$

$$p_{1,y} + t \cdot r_{1,y} = p_{2,y} + s \cdot r_{2,y}$$

$$t = \frac{(p_{2,x} - p_{1,x}) \cdot r_{2,y} - (p_{1,y} - p_{2,y}) \cdot r_{2,x}}{r_{1,x} \cdot r_{2,x} - r_{1,y} \cdot r_{2,x}}$$

Um nicht jedes Label mit jedem auf Überlappung testen zu müssen, können folgende Verbesserungen in Betracht gezogen werden:

Die notwendige Bedingung für das Schneiden zweier Labels ist, dass der Abstand der beiden Mittelpunkte m_1, m_2 kleiner oder gleich der Summe beider Radien r_1, r_2 ist:

$$d(m_1, m_2) \leq r_1 + r_2$$

Dabei sind die Radien die halbe Diagonale der Beschriftungen. Wird zunächst dieses Kriterium geprüft, kann die Anzahl der auf Überlappung zu testenden Labels wesentlich reduziert werden. Werden die Überschneidungen in der Interaktionsphase berechnet, kann diese Menge durch einen Filterungsschritt auf Labels im Sichtbereich weiter reduziert werden. Allerdings wird bei dieser Filterung die Konsistenzeigenschaft völlig außer Acht gelassen, so dass es beim Verschieben zu Artefakten kommen kann (vgl. Abbildung 4.8).

Wird die eingeschränkte Konsistenz berücksichtigt, kann eine weitere Verbesserung erzielt werden, indem für die Schnittberechnung nur bereits platzierte Beschriftungen betrachtet werden (vgl. Algorithmus 4.1).

Algorithmus 4.1 Beschriftungsplatzierung

```
for jedes Label in absteigender Priorität do
  if Kein Schnitt mit bereits sichtbaren Labels then
    Füge Label zur Liste der sichtbaren Labels hinzu.
    Zeichne das Label
  end if
end for
```

Bei der Implementierung der diskreten Schritte kann das Ergebnis in einer Liste, die alle Kamerazustände und die zu diesen Zuständen sichtbaren Labels enthält, gespeichert werden. Der Vorteil daran ist, dass in der Interaktionsphase nur noch der richtige Zustand ermittelt werden muss und schließlich die sichtbare Labelmenge einfach ausgegeben werden kann. Der Platzbedarf dafür ist allerdings relativ hoch. Eine Alternative dazu ist, für jedes Label den Skalierungs- und den sichtbaren Winkelbereich zu speichern. Der Nachteil daran ist, dass während der Interaktionsphase für jedes Label eine Liste der sichtbaren Zustände durchlaufen werden muss, um zu entscheiden, ob es zum aktuellen Zustand sichtbar ist oder nicht.

Für beide Varianten muss zu einem gegebenen, aktuellen Zustand in der Interaktionsphase entschieden werden, welcher vorberechnete Zustand dargestellt werden soll. In dieser Implementierung hat "Nearest Neighbour" bereits gute Ergebnisse für den Rotationswinkel

geliefert. Dabei wird immer der Zustand ausgewählt, der am nächsten zum aktuellen Wert liegt. Beim Zoomen kommt es bei diesem Ansatz allerdings zu vielen Überschneidungen, weshalb es besser ist, immer die nächst höhere Zoomstufe auszuwählen. Der Nachteil daran ist, dass bei Auswahl der Skalierungsstufen mit großen Abständen Labels verschwinden, obwohl für den Benutzer noch kein ersichtlicher Konflikt mit einem anderen Label vorliegt.

Für die Überdeckungsberechnung beim Kippen werden die Labels entlang des Sichtstrahls der Kamera auf eine Ebene projiziert (vgl. Abbildung 4.14). Dazu werden Geraden zwischen der Kameraposition k_{pos} und den Eckpunkten $e_i, i \in \{0, 1, 2, 3\}$ aufgestellt:

$$g_i : \begin{pmatrix} e_{i,x} \\ e_{i,y} \\ e_{i,z} \end{pmatrix} + t \cdot \begin{pmatrix} k_{pos,x} - e_{i,x} \\ k_{pos,y} - e_{i,y} \\ k_{pos,z} - e_{i,z} \end{pmatrix}$$

Diese Geraden werden mit der Projektionsebene analog zum Geraden-Ebenen-Schnitt aus Abschnitt 3.2.2 geschnitten. Die Schnittpunkte ergeben das projizierte Objekt, welches analog zur Schnittberechnung bei der Rotation auf Überdeckungen geprüft werden kann. Die Near-Plane des Sichtbereichs bietet sich als Projektionsebene an, da sie aus den Modell-View-Matrizen einfach berechnet werden kann.

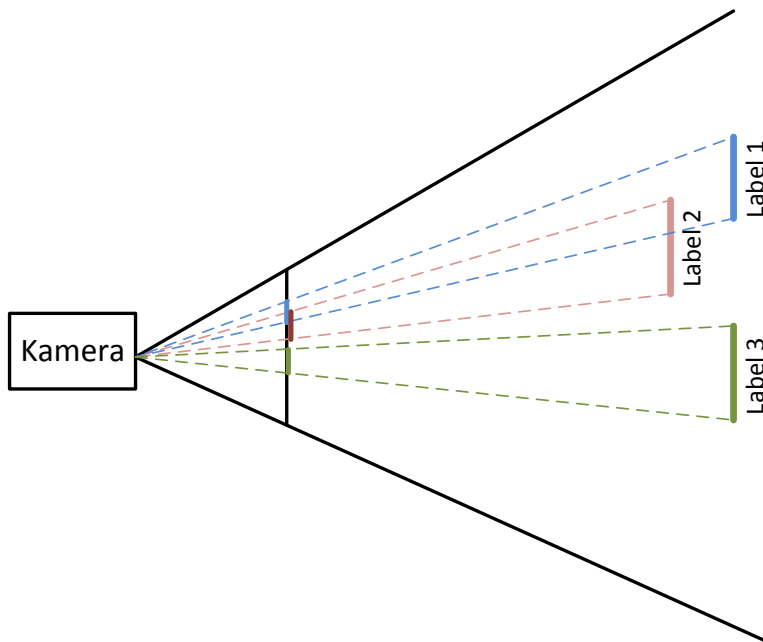


Abbildung 4.14.: Überdeckungserkennung durch Projektion auf die Near-Plane und Überdeckungsberechnung der projizierten Objekte.

Zur Verbesserung der Laufzeit der Verdeckungsberechnungen im Interaktionsschritt, kann eine Parallelisierungsstrategie verfolgt werden. Dabei übernimmt ein Thread die Zeichenrou-

tine und ein anderer Thread die Berechnung der Überdeckungen. Die Herausforderung liegt darin am Ende des Berechnungsschrittes die Daten zu synchronisieren und Seiteneffekte zu vermeiden. So kann die Zeichenroutine mit derselben Framerate zeichnen, als ob keine Berechnungen stattfinden würden. Am Ende eines Berechnungsschritts werden die Daten synchronisiert. Dieser Ansatz profitiert von den heutigen Multi-core Prozessoren, die auch Einsatz bei Navigationsgeräten der neuen Generation Verwendung finden.

5. Ergebnisse

5.1. Testbedingungen

Zur Durchführung aller Messungen wurde ein Dell Vostro 3700 Laptop mit einem Intel® Core™ I5-560M (2.66GHz) Prozessor mit einer Taktfrequenz von 2.66 GHz, einem Arbeitsspeicher von 4GB und einer Nvidia® Geforce™ GT 330M, Grafikkarte mit 1GB Grafikspeicher verwendet.

5.2. Laufzeiten

5.2.1. Vorberechnungsphase nach dem Modell von Been et al.

Die Schnittberechnungen der Implementierung des Modells von Been et al. [BDYo6] können aufgrund folgender Berechnungsmöglichkeiten in verschiedene Fälle unterteilt werden (vgl. Tabelle 5.1):

- i Für die Schnittberechnung werden die Labels in allen acht Nachbarbuckets betrachtet.
- ii Für die Schnittberechnung werden nur die Labels in den drei relevanten Nachbarbuckets betrachtet.
- iii Zur Vermeidung von doppelten Berechnungen wird ein Konfliktgraph erstellt, der pro LOD-Stufe neu aufgebaut wird.
- iv Die Label-Ids der betrachteten Buckets werden in einer Liste gespeichert und jedes Label mit jedem geschnitten.
- v Jede mögliche Schnittkombination, zwischen Labels in den Buckets und den Nachbarbuckets (keine Schnitte zwischen den Nachbarbuckets), wird in eine Liste gespeichert und durchlaufen.

Diese Fälle wurden anhand des Ortes Leonberg berechnet und miteinander verglichen. Tabelle 5.2 zeigt, welche und wie viele Labels dieser Datensatz enthält.

5. Ergebnisse

Fall	i	ii	iii	iv	v
1		x	x	x	
2	x		x	x	
3		x		x	
4		x			x

Tabelle 5.1.: Verschiedene Fälle für Schnittberechnungen.

Gesamtzahl der Labels	1720
Interessenspunkte (POI)	1073
Orte	9
Straßen	638

Tabelle 5.2.: Labelarten und -anzahl im Leonberg-Datensatz.

Jeder der vier Testfälle (vgl. Tabelle 5.1) wurde fünfmal berechnet und der Median als Ergebnis ermittelt. Abbildung 5.1 zeigt den Vergleich der Gesamtlaufzeiten der einzelnen Fälle.

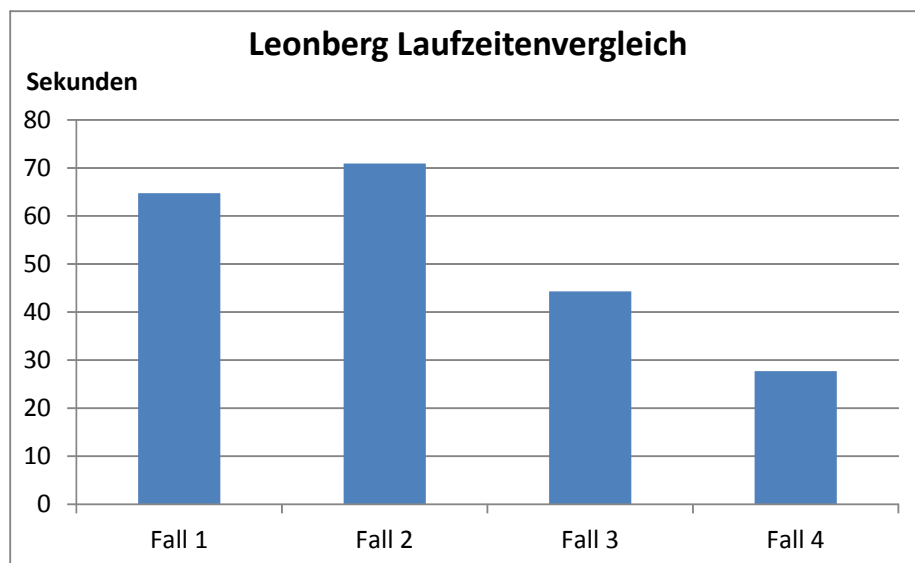


Abbildung 5.1.: Laufzeiten der verschiedenen Fälle des Leonberg-Datensatzes. Die x-Achse zeigt die verschiedenen Fälle, die y-Achse die Laufzeit in Sekunden.

Wie zu erwarten war ist der Fall, in dem alle acht Nachbarbuckets betrachtet werden wesentlich langsamer als die Fälle, in denen die Überschneidungsberechnungen auf drei Buckets

reduziert wurden. Im Fall 1 und 2 werden für eine LOD-Stufe Mehrfachberechnungen durch den Konfliktgraph vermieden, allerdings zeigt der Fall 3 in welchem kein Konfliktgraph benutzt wird, dass das Aufbauen und Suchen in einer solchen Datenstruktur eine längere Laufzeit verursacht, als Überschneidungen mehrfach zu berechnen. Die beste Laufzeit liefert das Verfahren des Falls 4, bei dem nur Schnitte zwischen aktuellem Bucket und Nachbarbuckets (keine Schnitten zwischen den Nachbarbuckets) berechnet werden.

Der Vergleich der Laufzeiten der einzelnen Detailstufen zeigt, dass in allen Fällen in den mittleren Detailstufen 5-7 die meisten Schnitte berechnet werden (vgl. Abbildung 5.2 und Tabelle 5.3). In den niedrigen LOD-Stufen ist die Anzahl der Labels zwar größer als in den Mittleren, aber durch die relativ kleine Größe der Buckets liegen die Labels gleichverteilt und sind somit schnell berechenbar. In den mittleren LOD-Stufen kommt es öfter zu Häufungspunkten in den Buckets, was zu einer längeren Laufzeit führt. In den hohen LOD-Stufen sind nur noch wenig Labels sichtbar und die Laufzeit ist entsprechend geringer. Um die Häufungspunkte zu reduzieren kann die Einteilung der LOD-Stufen durch kleinere Skalierungsschritte verfeinert werden.

LOD	Skalierung	Anzahl der Labels
0	1	1720
1	10	1480
2	100	1479
3	1000	1472
4	4000	1288
5	10000	941
6	100000	738
7	1000000	92
8	5000000	8
9	10000000	1
10	182563424,42	1

Tabelle 5.3.: Anzahl der Labels in den Detailstufen des Leonberg-Datensatzes.

Die beste Laufzeit, die keinen zusätzlichen Speicherbedarf benötigt und somit auch für große Labelmengen geeignet ist, ist das Verfahren des Falls 3. Dieser Fall wurde für verschiedene Datensätze durchlaufen und ist in Tabelle 5.4 und Abbildung 5.3 dargestellt. Aufgrund der langen Laufzeiten bei großen Labelmengen wurde jeweils nur ein Durchlauf betrachtet.

Die erwartete Laufzeit des Algorithmus ist $\mathcal{O}(n^2)$, wobei n die Anzahl der Labels ist. Die Ergebnisse aus Abbildung 5.3 zeigen, dass diese Laufzeit durch die Beschleunigungsstrategien verbessert werden können. Zwischen Berlin und Schwaben tritt ungefähr die erwartete Laufzeit ein: die Labelmenge von Schwaben ist eineinhalb mal so groß als die von Berlin, wobei die Laufzeit um den Faktor 2,5 steigt. Dagegen unterscheidet sich die Laufzeit zwischen

5. Ergebnisse

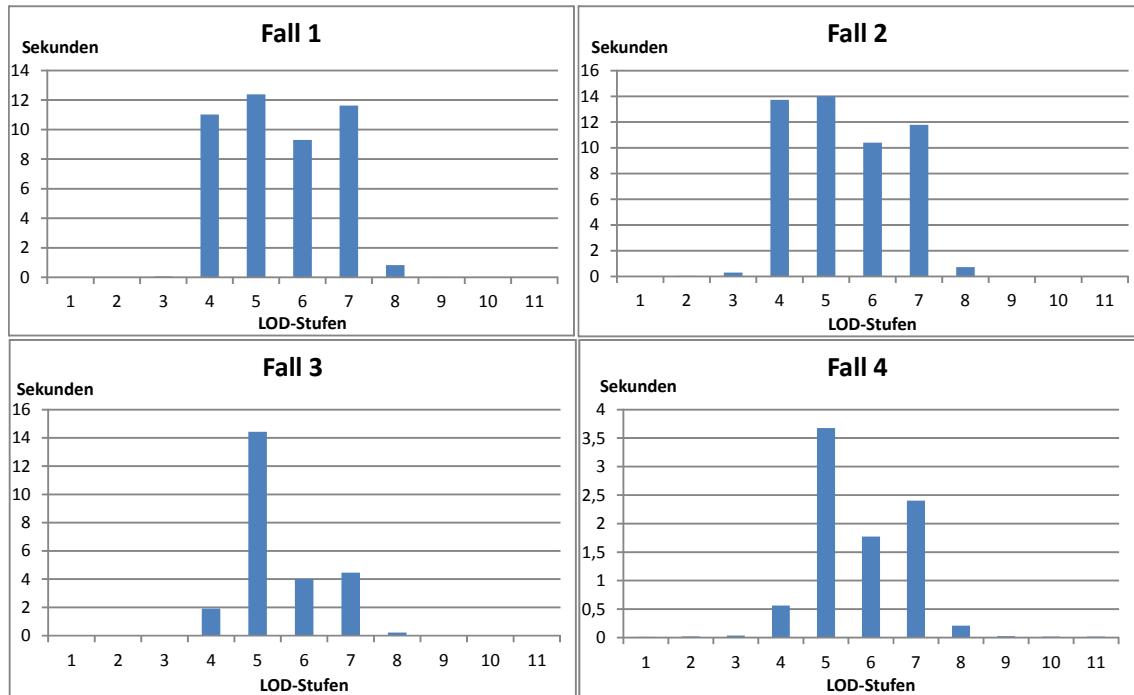


Abbildung 5.2.: Laufzeiten der verschiedenen Fälle des Leonberg-Datensatzes in Detailstufen aufgeschlüsselt. Die x-Achse zeigt die verschiedenen LOD-Stufen, die y-Achse die Laufzeit in Sekunden.

	Lichtenstein	Stuttgart	Bremen	Berlin	Schwaben
Gesamtzahl der Labels	1088	4909	20149	67427	96134
Interessenpunkte (POI)	285	2341	7535	32680	31381
Orte	20	2	104	104	3181
Straßen	776	2566	12510	34644	61575

Tabelle 5.4.: Labelarten in den verschiedenen Datensätzen.

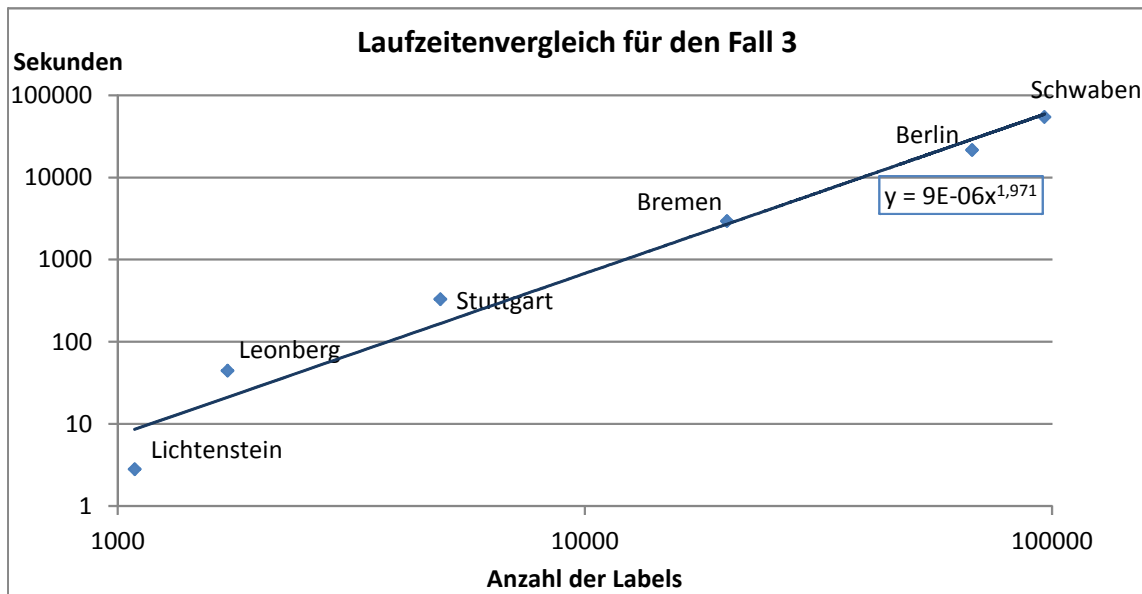


Abbildung 5.3.: Laufzeitenvergleich für den Fall 3. Die y-Achse zeigt die Anzahl der Labels und die x-Achse stellt die Laufzeit in Sekunden dar.

Bremen und Stuttgart um einen Faktor von 4,1 bei einer Labelmengeänderung um einem Faktor von 8,9. Alle anderen Werte liegen zwischen diesen beiden Laufzeitabschätzungen.

Eine genauere Aufschlüsselung der Laufzeiten zeigt die Abbildung 5.4. Dabei sind die Laufzeiten aufgeschlüsselt in den XML-Parser, das Setzen von Labeleigenschaften wie Priorität (Eigenschaften), das Berechnen der Eckgeraden und Seitenflächen der Labelpyramiden (Geraden), den LOD-Stufen und das Schreiben in eine Datei (schreiben). Man sieht, dass die Zeit, die ein XML-Parser zum parsen und das Schreiben der Ergebnisse in eine Datei benötigt, mit steigender Labelanzahl an Relevanz verliert. Die unterschiedlichen Laufzeiten beim XML-Parser kommen durch die unterschiedliche Struktur der XML-Dateien zustande.

5.2.2. Sampling der Rotation um die Hochachse in der Vorverarbeitungsphase

Zunächst wurde mit dem Ansatz von Been et al. [BDYo6] eine nordausgerichtete Karte zu jeder Skalierungsstufe berechnet und Überdeckungszustände für 17 Skalierungsstufen und jeweils 36 Winkel pro Skalierung gesampelt. Die Skalierungsstufen wurden folgendermaßen eingeteilt:

$$\frac{\text{größte Skalierung}}{2^i}, \quad i \in \{0, \dots, 16\}$$

5. Ergebnisse

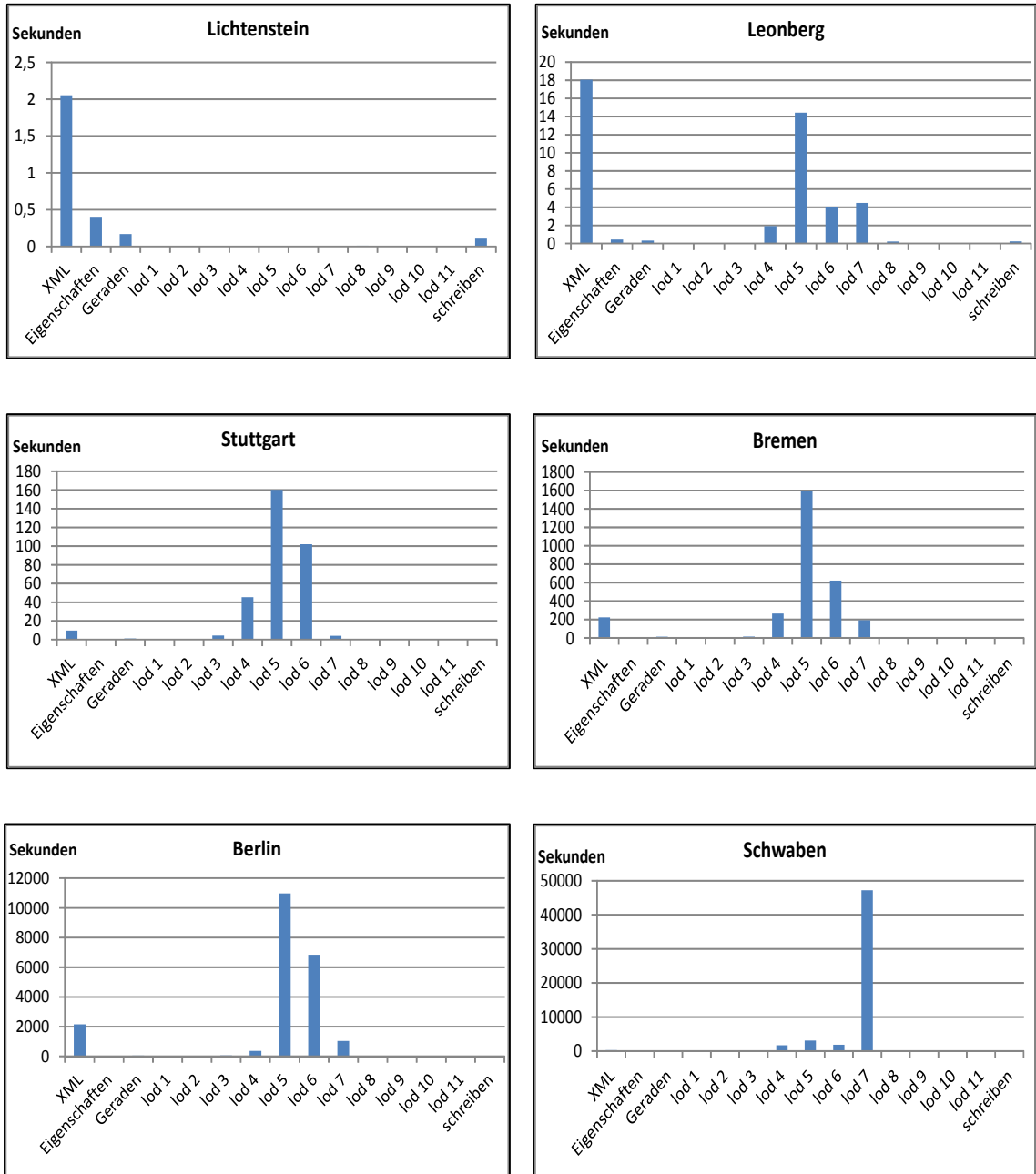


Abbildung 5.4.: Vergleich der Laufzeiten der einzelnen Berechnungsschritte. Die x-Achse zeigt die verschiedenen Schritte, die y-Achse die Laufzeit in Sekunden.

Das Ergebnis der Datensätze Lichtenstein, Leonberg, Stuttgart und Bremen sind in der Abbildung 5.5 dargestellt. Da die Datensätze Berlin und Schwaben bereits für die Berechnung der nordausgerichteten Karte eine sehr lange Laufzeit hatten wurden sie hier weggelassen.

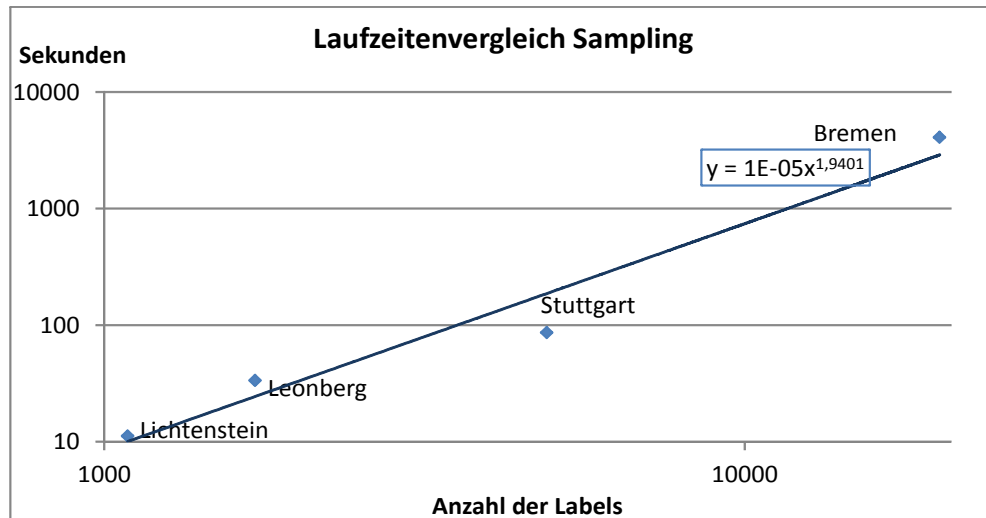


Abbildung 5.5.: Laufzeitenvergleich für das Sampling. Die x-Achse zeigt die Anzahl der Labels und die y-Achse stellt die Laufzeit in Sekunden dar.

Man kann feststellen, dass die Laufzeit mit der Anzahl der Labels wächst. Im Vergleich zur Laufzeit des implementierten Verfahrens von Been verläuft die Laufzeitkurve ähnlich. Im allgemeinen Fall müssen beim Sampling mehr Schnitte berechnet werden was eine längere Laufzeit erfordert. Die geringere Laufzeit einiger Fälle kommt dadurch zustande, dass durch die Vorberechnung der nordausgerichteten Karte wesentlich weniger Labels auf Überschneidung geprüft werden müssen. Die erwartete Laufzeit beträgt auch hier $\mathcal{O}(n^2)$ (vgl. Trendlinie der Abbildung 5.5). Eine Verbesserung erzielt man, indem nur Labels auf Überlappung geprüft werden, deren Radien sich schneiden. Tun sie dies nicht, kommt es bei keinem Winkel zu einem Schnittpunkt.

Abbildung 5.6 zeigt nochmals die Laufzeiten wie sie sich auf die Skalierungsstufen aufteilen. Die Ausgangslabelmenge sind die Labels, die bei einer nordausgerichteten Karte zur aktuellen Skalierungsstufe sichtbar sind. Je höher die Skalierung, desto weniger Labels müssen auf Schnitte überprüft werden und somit reduziert sich die Laufzeit für die Überdeckungsberechnung. In der gesampelten Skalierung 17 ist die Laufzeit trotz geringer Labelanzahl hoch. Dies liegt daran, dass am Ende der Berechnungsschritte für jedes Label geprüft wird, ob es in der Liste der Labels die überdeckt werden enthalten ist. Für die Skalierung 17 sind fast alle Labels verdeckt, was zu einer langen Suchlaufzeit führt. Für die

5. Ergebnisse

Winkelberechnungen pro Skalierungsstufe gibt es kaum Laufzeitunterschiede, da zu jedem Winkel dieselbe Labelmenge auf Überschneidung geprüft werden muss.

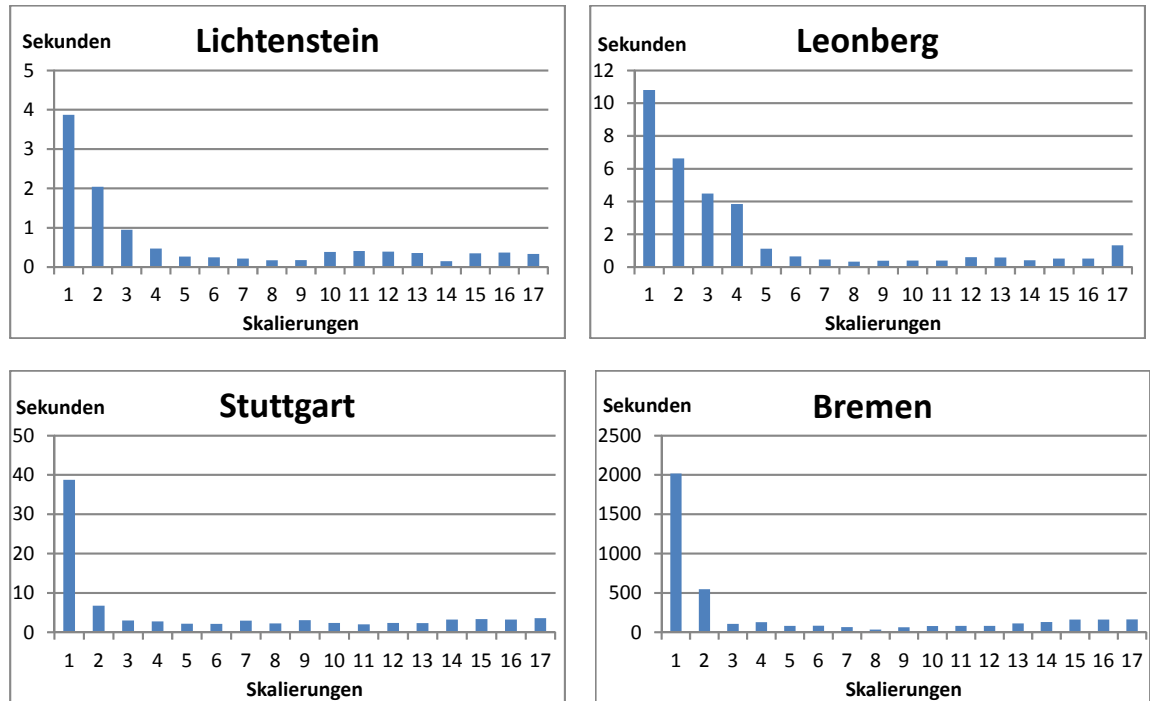


Abbildung 5.6.: Laufzeitenvergleich für das Sampling. Die x-Achse zeigt die Anzahl der Labels und die y-Achse stellt die Laufzeit in Sekunden dar.

Betrachtet man die Laufzeit des Sample-Ansatzes zur Interaktionsphase, dann kann durchschnittlich eine Framerate von 20-60 erzeugt werden. Zur Interaktionszeit wird aus den 612 vorberechneten Fällen der Zustand ausgewählt, der dem aktuellen Kamerazustand am nächsten ist und die Labels, die in diesem Zustand als sichtbar markiert sind, werden gezeichnet.

5.2.3. Schnittberechnungen in der Interaktionsphase

Bei der Schnittberechnung zur Interaktionsphase wird zunächst die Labelmenge auf Basis des Sichtbereichs und der aktuellen Skalierung gefiltert. Die Laufzeit der Schnittberechnung hängt von der Anzahl der Labels ab. Da diese durch den Filterungsschritt sehr klein und immer unterschiedlich ist, wurde im Folgenden die Framerate (gezeichnete Frames pro Sekunde) betrachtet. Diese zeigt an, wie flüssig die Interaktion mit dem Benutzer und das Darstellen der Labels läuft.

Zum Vergleich der Framerate wurden folgende Fälle für den Datensatz Leonberg betrachtet:

1. Nord-vorberechnete Karte mit Kamerablick auf den Punkt $\phi = 9.0127567, \lambda = 48.7997485$ mit Höheninformationen.
2. wie Fall 1 ohne Höheninformation.
3. wie Fall 2 mit Rotation um den Winkel 40 Grad.
4. wie Fall 3 mit Kippwinkel von 60 Grad.
5. wie Fall 4 mit Höheninformation.

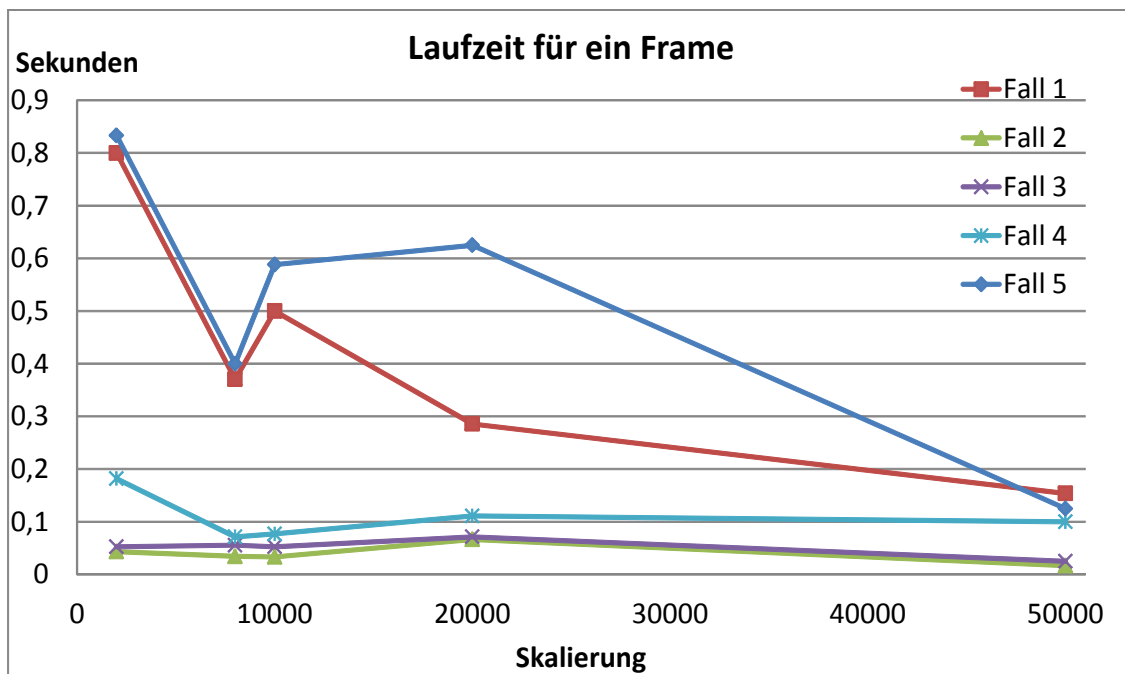


Abbildung 5.7.: Laufzeitenvergleich für die Schnittberechnungen zur Interaktion. Die x-Achse zeigt die Skalierungen und die y-Achse stellt die Zeit für die Berechnung von einem Frame in Sekunden dar.

Abbildung 5.7 zeigt die Zeit die für die Berechnung von einem Frame benötigt wird zu verschiedenen Skalierungen. Es zeigt sich das der Rechenaufwand durch Rotation- und Kippenoperationen steigt. Je größer die Skalierung ist, desto weniger Labels werden nach dem Filterungsschritt gezeichnet und somit reduziert sich die Schnittberechnungszeit. Vor

allein das Abfragen der Höhe der Labels aus dem Geländemodell erzeugt einen Einbruch in der Schnelligkeit der Darstellung.

5.3. Speicherverbrauch

Bisher wurde aufgrund der Laufzeit kein Datensatz größer als Schwaben betrachtet. Dieses Kapitel betrachtet den nötigen Speicherbedarf für das Einlesen der Labels und das Berechnen der Überdeckungen um die Möglichkeit zu prüfen, noch größere Datensätze zu rechnen wenn die Laufzeit keine Rolle spielt.

5.3.1. Vorberechnungsphase nach dem Modell von Been et al.

Für die Datensätze Lichtenstein, Leonberg, Stuttgart und Bremen wurde der Speicherbedarf für das Einlesen der Labels und das Berechnen der Überdeckungen bestimmt (vgl. Tabelle 5.5).

	Lichtenstein	Leonberg	Stuttgart	Bremen
Einlesen	3,95 MB	5,13 MB	7,7MB	24,67MB
Überdeckungsberechnung	3,27 MB	3,62 MB	7,23 MB	25,95 MB

Tabelle 5.5.: Speicherverbrauch der verschiedenen Datensätze beim Einlesen und bei der Überdeckungsberechnung.

Diese Messungen zeigen, dass größere Datensätze wie Bremen ohne Probleme noch berechnet werden können. Weiter wird betrachtet, welche Datenstruktur wie viel Speicher benötigt.

Labels werden während der kompletten Berechnung im Speicher gehalten. Für jedes Label werden die Eigenschaften, wie sie in Tabelle 5.6 dargestellt sind, gespeichert (Der Typ wird als string gespeichert, da er für viele Labels auch als Name dient).

Straßen-Labels bestehen aus Punkten, welche den entsprechenden Längen- und Breitengraden in einer Map "map<int, TPoint<double> >" gespeichert werden und 20 Byte pro Punkt benötigen. Die Zuordnung, welche Punkte zu welchem Label gehören, geschieht über eine Id-Liste die aus integer besteht und somit 4 Byte pro Eintrag benötigt (vgl. Tabelle 5.7).

Diese Punktliste wird nur zum Setzen des richtigen Winkels für gedrehte Straßen verwendet und wird anschließend wieder gelöscht, sie spielt also während der Schnittberechnungen keine Rolle.

Eigenschaft	Datentyp	Bytes
Name	string	$4 \cdot \text{Länge des Namens}$
Typ	string	POI = $3 \cdot 4 \text{ Byte} = 12 \text{ Byte}$, PLACE = $5 \cdot 4 \text{ Byte} = 20 \text{ Byte}$, STREET = $6 \cdot 4 \text{ Byte} = 24 \text{ Byte}$
Längengrad	double	8 Byte
Breitengrad	double	8 Byte
Rotationswinkel	double	8 Byte
maximale Skalierung	double	8 Byte
minimale Skalierung	double	8 Byte
Breite	double	8 Byte
Höhe	double	8 Byte
Textur	int	4 Byte
Priorität	int	4 Byte

Tabelle 5.6.: Speicherverbrauch der Labelliste.

	Anzahl an Straßen-Punkten	Byte
Lichtenstein	18597	446328
Leonberg	31825	763800
Stuttgart	44994	1079856
Bremen	162694	3904656

Tabelle 5.7.: Speicherverbrauch der Straßen-Punkte.

Für die Berechnung der Schnitte werden die Eckgeraden und Seitenflächen der Pyramiden, die die Labels entlang der Skalierung aufspannen, benötigt. Diese werden einmal vorberechnet und dann für alle LOD-Stufen verwendet. Eine Eckgerade benötigt 48 Byte. Für eine Seitenebene ergibt sich ein Speicherbedarf von 104 Byte.

Der Gesamtspeicherbedarf für die Eckgeraden und Seitenflächen für verschiedene Datensätze zeigt die Tabelle 5.8.

Für jede LOD-Stufe werden in einer Liste die Ids der Labels gespeichert, die zu dieser Stufe sichtbar sind. Die sichtbaren Labels werden pro LOD-Stufe auf Buckets aufgeteilt. Die Anzahl dieser Buckets hängt von der Breite der größten Beschriftung ab. Zur Schnittberechnung wird eine Liste der sichtbaren Label-Ids und die Bucketpositionen (x,y), in der die Labels liegen, gespeichert. Den Speicherverbrauch für Id- und Bucketliste für die verschiedenen Datensätze bildet die Tabelle 5.9 ab.

	Byte
Lichtenstein	165376
Leonberg	261440
Stuttgart	746168
Bremen	3062648
Berlin	10248904
Schwaben	14612368

Tabelle 5.8.: Speicherverbrauch der Eckgeraden und Seitenflächen.

	Lichtenstein	Leonberg	Stuttgart	Bremen	Berlin	Schwaben
LOD 1	17408	27520	78544	322384	1078832	1538144
LOD 2	592	23680	67920	271104	885600	1353952
LOD 3	592	23664	67920	271024	885056	1352752
LOD 4	592	23552	66752	268096	877872	1337776
LOD 5	592	20608	41744	209920	703888	1064480
LOD 6	592	15056	10416	81456	262352	514432
LOD 7	592	11808	2880	28176	78928	263104
LOD 8	592	1472	144	1408	2352	19328
LOD 9	76	128	32	96	128	400
LOD 10	64	16	48	48	32	80
LOD 11	48	16	48	32	16	48

Tabelle 5.9.: Speicherverbrauch der Label-Id- und Bucketlliste.

Für die Schnittberechnung wird in den Fällen 1-4 eine Liste der Label-Ids des aktiven Buckets und den jeweils betrachteten Nachbarbuckets gespeichert. Im Fall 4 wird jedes Konfliktpaar des aktiven Buckets und den betrachteten Nachbarbuckets in eine Liste gespeichert. Dies sind die zwei Ids der Labels und benötigen 8 Byte pro Konfliktpaar. Im schlimmsten Fall liegen alle Labels in einem Bucket und benötigen $4 \text{ Byte} \cdot \text{Labelanzahl}$ bzw. $8 \text{ Byte} \cdot \text{Labelanzahl}$ Speicherplatz.

Wird ein Konfliktgraph verwendet, wird für jede Label-Id eine Liste aller niedriger priorisierten Label-Ids erstellt, die mit diesem Label auf Überschneidung geprüft wurden. Im schlimmsten Fall werden alle Labels miteinander geschnitten und fordern einen Platzbedarf von $4 \text{ Byte} \cdot (\text{Labelanzahl})^2$.

5.3.2. Sampling der Rotation um die Hochachse in der Vorverarbeitungsphase

Zunächst wird eine nordausgerichtete Karte berechnet. Dafür ergibt sich derselbe Speicherbedarf wie in Abschnitt 5.3.1. Zusätzlich kommt der Speicherbedarf für die Sample-Berechnungen dazu. Jeder berechnete Kamerastatus wird mit der zugehörigen Liste sichtbarer Label-Ids in eine Liste gespeichert. Tabelle 5.10 zeigt den Speicherverbrauch eines Eintrags in der Kameraliste. Für die Vorberechnung werden 17 Skalierungen und für jede Skalierung jeweils 36 Winkel berechnet. Dies ergibt 612 Einträge in die Liste. Im schlechtesten Fall sind alle Labels sichtbar.

Kamerastatus	Typ	Byte
Rotation	double	8
Skalierung	double	8
Ids der Labels	integer	Anzahl der Labels · 4 Byte

Tabelle 5.10.: Speicherverbrauch Samples.

5.3.3. Schnittberechnungen in der Interaktionsphase

Für das Zeichnen und Berechnen zur Interaktionsphase werden Labels analog zum Modell von Been et al. [BDY06] in einer Liste gehalten und benötigen somit denselben Speicherbedarf, wie er in Tabelle 5.6 aufgezeigt ist. Zunächst wird die Labelmenge hinsichtlich der Skalierung und anschließend des Sichtbereichs gefiltert. Diese sichtbaren Beschriftungen werden auf Überschneidungen geprüft. Der dazu benötigte Speicherbedarf kann aufgrund der geringen Labelanzahl vernachlässigt werden.

6. Mögliche Erweiterungen

Die in dieser Diplomarbeit vorgestellten Verfahren erfüllen die meisten Anforderungen an eine gute Beschriftung für 3D-Karten. Sie bilden aber auch die Basis für eine Reihe von Erweiterungsmöglichkeiten.

Für den in der Aufgabenstellung spezifizierten zweiten Erweiterungsvorschlag, soll hier ein Ansatz umrissen werden. Dabei soll der Benutzer während der Interaktionsphase die Prioritäten der Beschriftungen ändern können. Er wählt zum Beispiel aus, dass nur Restaurant-POIs angezeigt werden sollen. Werden die Überdeckungen während der Interaktion berechnet, benötigt diese Anforderung keine Erweiterung des Verfahrens. Die Menge der Labels, welche in der Interaktionsphase auf Überdeckung geprüft werden basieren aber auf der Labelmenge, die in Abschnitt 3.2.1 in der Vorverarbeitungsphase berechnet werden. Im Folgenden wird eine Möglichkeit umrissen, wie die Prioritäten dieser Menge während der Interaktionsphase geändert werden können, ohne eine komplette Neuberechnung des Vorverarbeitungsschrittes durchführen zu müssen.

Bisher wird für jedes Label die maximale Skalierung gespeichert, in der es ohne Verdeckung durch ein höher priorisiertes Label gezeichnet werden kann. Dies wird nun um einen Konfliktgraphen erweitert. Die Knoten werden durch die Labels dargestellt. Eingehende Kanten sind Konflikte mit höher priorisierten Labels, ausgehende Kanten sind Konflikte mit niedriger priorisierten Beschriftungen. Kantengewichte stellen die Skalierungsstufe dar, bei der sich die Labelpaare schneiden.

Die maximale Skalierung, bei der ein Label gezeichnet werden kann, ergibt sich mit der Auswahl des Minimums der eingehenden Kantengewichte. Werden nun während der Interaktionsphase die Prioritäten geändert, müssen für jeden Knoten mit neuer Priorität alle Kanten betrachtet und gegebenenfalls umgedreht werden, so dass wieder jede ausgehende Kante zu einem niedriger priorisierten und jede eingehende Kante zu einem höher priorisierten Label zeigt. Abbildung 6.1 zeigt ein Beispiel, bei dem die Prioritäten der Label b und c vertauscht werden. Das Problem an diesem Ansatz ist der hohe Platzbedarf, der durch das Speichern jeglicher Schnittkombinationen entsteht.

Die bisher verwendeten Labelprioritäten wurden nach eigenem Ermessen von Hand gesetzt. Um eine für den Benutzer sinnvolle Prioritätenreihenfolge zu erhalten, können bisher bestehende Renderer wie zum Beispiel Mapnik von OpenStreetMap [Ope] analysiert oder Benutzerevaluationen durchgeführt werden.

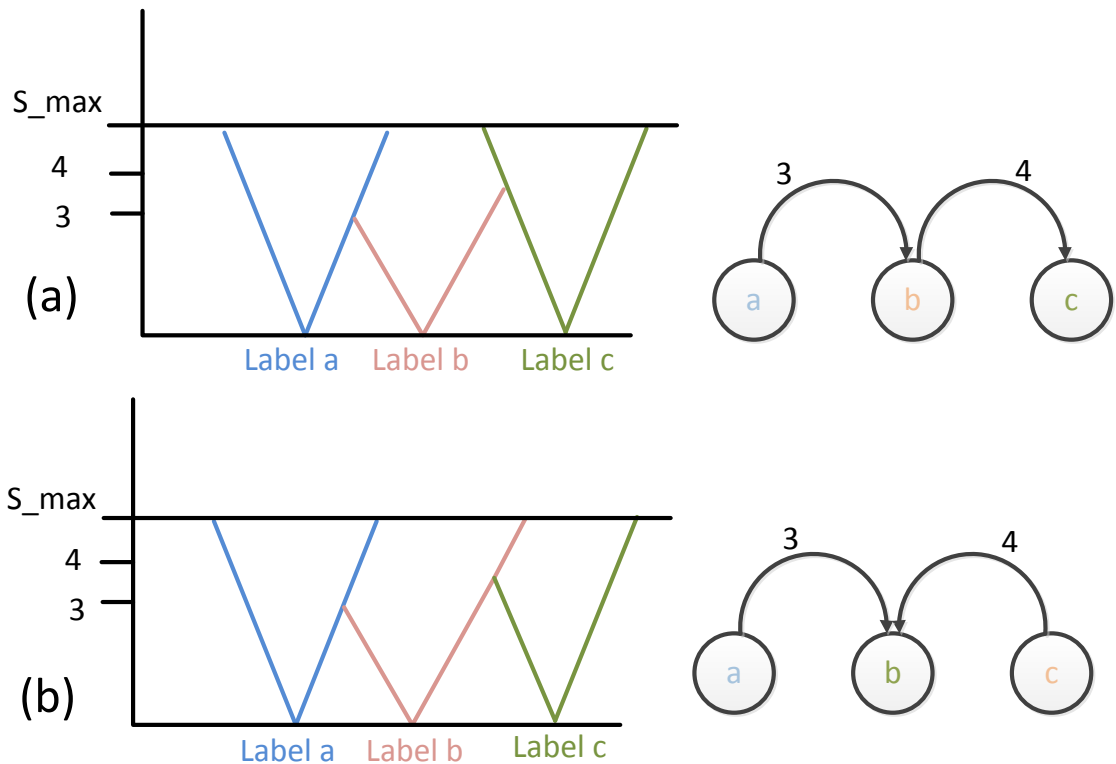


Abbildung 6.1.: (a) zeigt wie sich drei Labels schneiden und den entsprechenden Konfliktgraphen. In (b) wurde die Priorität von Label b und c vertauscht. Da das Label a immer noch größer ist als Label b mit der getauschten Priorität muss diese Kante im Konfliktgraphen nicht weiter betrachtet werden. Zwischen Label b und c wird die Kante getauscht.

In der Implementierung des Modells von Been et al. [BDY06] in Abschnitt 3.2.1 wird für die Invariante nur eine Labelposition betrachtet. Wie in Abschnitt 2.2.1 diskutiert, bestehen viele verschiedene Möglichkeiten für die Platzierung der Beschriftung. So könnte zum Beispiel das Greedy-Verfahren (vgl. Abschnitt 3.1.1) auf vier Beschriftungspositionen erweitert werden, so dass es die Platzierung wählt, die den größten aktiven Bereich ergibt. Die danach betrachteten niedriger priorisierten Beschriftungen werden nur mit der gewählten Platzierung verglichen. Statt diesem Greedy-Algorithmus können auch jegliche andere Auswahlalgorithmen, wie sie in Abschnitt 2.2.4.1 aufgeführt wurden, verwendet werden.

Bisher wurde für die Bewertung von Beschriftungspositionen die Verdeckung von Terrain, wie zum Beispiel Berge, nicht in Betracht gezogen. Vor allem in 3D-Karten ist dies von Bedeutung und kann in dieser Implementierung noch berücksichtigt werden.

In dieser Implementierung werden Straßen als punkthafte Beschriftungen behandelt. Dies führt dazu, dass Straßen nur durch eine Approximation gelabelt werden. Um ein besseres Ergebnis zu erzielen, kann untersucht werden, inwieweit diese Implementierung um die Strategien der Beschriftung linienhafter Objekte, wie sie in Abschnitt 2.2.1 vorgestellt wurden, erweitert werden können. So kann zum Beispiel analog zur Bewertungsfunktion punkthafter Objekte (vgl. Abschnitt 2.2.1) eine Bewertung für linienhafte Objekte angegeben werden [ECMS96].

Einige exemplarische Bewertungsfunktionen:

- Zentrierung: Verhältnis der Entfernung zwischen Labelbox und Anfang des linienhaften Objekts zur gesamten Länge des Objekts:

$$centerness = 1 - \left| 2 \cdot \frac{\text{distance}(\text{labelbox}, \text{start of lineObject})}{\text{length}(\text{lineObject})} - 1 \right|$$

- Flachheit: Die eingeschlossene Fläche zwischen Labelbox und dem linienhaften Objekt:

$$flatness = \frac{1}{\text{Fläche zwischen Labelbox und linienhaftem Objekt}}$$

Die Bewertung erstreckt sich von 0.0 (schlecht) bis 1.0 (sehr gut) analog zu der Beschriftung punkthafter Objekte [Peto3].

Für die implementierten Verfahren einer 3D-Beschriftung war bisher immer die Beschriftungsmenge nach dem Vorbereitungsschritt die Eingabe. Diese Beschriftungsmenge ist die Ansicht einer nach Norden ausgerichteten Karte. Nun kann eine Erweiterung erzeugt werden, die auch andere Ausrichtungen vorberechnet und danach die Ausrichtung wählt, mit der maximal viele Beschriftungen gezeichnet werden können.

Um die Geschwindigkeit der Vorbereitung noch etwas zu verbessern, können statt den bisher eingesetzten Buckets, Quadrees verwendet werden. Diese liefern eine bessere Genauigkeit für Gebiete mit vielen Labels und eine spärlichere Unterteilung für Bereiche mit nur wenigen Labels.

Da für einige Verfahren die Konsistenz eingeschränkt bzw. ganz aufgegeben werden musste, können Beschriftungen mittels "Fading" langsam aus- oder wieder eingeblendet werden, statt sie "hart" aufpoppen zu lassen. Damit kann der Benutzer besser nachvollziehen, wohin das Label verschwindet.

Eine andere Möglichkeit bieten Heuristiken, mit diesen können globale Labelprioritäten eingeschränkt und zur Interaktionszeit entschieden werden, welches Label wichtiger ist. Zum Beispiel könnte bei der in Abbildung 4.7 dargestellten Inkonsistenz, zunächst geprüft werden wie wichtig Label 0 im Verhältnis zu Label 1 ist. Ist Label 0 ein Stadtname und Label 1 nur ein Straßenschild, dann sollte die Inkonsistenz in Kauf genommen werden. Während

dagegen bei zwei Labels von ähnlicher Wichtigkeit, wie beispielsweise zwei Boutiquen, die nur aufgrund der Eindeutigkeitseigenschaft eine unterschiedliche Priorität haben, Label 1 trotz niedriger Priorität gezeichnet werden sollte und somit kann Konsistenz erhalten werden. Dazu muss jedes Label ein Flag besitzen, welches anzeigt, ob es im letzten Sichtbereich gezeichnet wurde. Wird jetzt die Kamera verschoben und das Label wird von einem höher priorisierten, durch eine Heuristik aber als nicht wesentlich wichtiger bestimmten Label verdeckt, dann wird es in diesem Sichtbereich weiter gezeichnet. Das Problem ist, eine gute Heuristik dafür zu finden, welche Labels wesentlich wichtiger sind, so dass keine wichtigen Labels immer verdeckt werden.


7. Zusammenfassung

Im Rahmen dieser Diplomarbeit wurde die Möglichkeit einer konsistenten Kartenbeschriftung für 3D-Karten untersucht. Dabei wurde der Konsistenzbegriff wie er in Been et al. [BDYo6] für zweidimensionale Karten definiert wurde auf die Anforderungen einer 3D-Karte erweitert. Dabei wurde festgestellt, dass die Konsistenzforderung nicht einfach übernommen werden kann, sondern diese, um eine ansprechende und für den Benutzer sinnvolle Beschriftung darzustellen, eingeschränkt werden muss. Es wurden Verfahren vorgestellt, die auf Basis dieses auf drei Dimensionen erweiterten Konsistenzbegriffs, eine schnelle Beschriftung erreichen. Dabei werden Überdeckungsrechnungen entweder in der Vorverarbeitungsphase durch geschlossene Formeln bzw. diskrete Zustände oder im Interaktionsschritt berechnet. Von den vorgestellten Ansätzen wurden zwei Möglichkeiten implementiert und miteinander verglichen. Die beste Lösung liefert ein Verfahren, welches auf Basis des Modells von Been et al. [BDYo6] eine Labelmenge für jede Skalierungsstufe vorberechnet und die Überdeckung für die Freiheitsgrade Rotieren, Kippen und Verschieben während der Interaktionsphase berechnet. Dies liefert eine Beschriftung, welche die Konsistenz für das Zoomen erhält und gleichzeitig viel Spielraum für Interaktionen mit dem Benutzer zulässt.

A. Anhang

Beschriftungsarten


Im folgenden werden alle verwendeten Labelarten mit entsprechender Priorität und Textur dargestellt.


Beschriftungstyp	Priorität	Textur
Orte		
Kontinent	0	
Land	1	
Staat	2	
Region	3	
Landkreis	4	
Großstadt	5	
Stadt	6	
Dorf	7	
Ortschaft	8	
Vorort	9	
Ort	10	
Insel	11	
Straßen		
Autobahn	12	
Autobahnanschluss	13	
Fernstraße	14	
Fernstraßenanschluss	15	
Primär	16	
Primäranschluss	17	
Sekundär	18	
Sekundäranschluss	19	
Tertiär	20	
Tertiäranschluss	21	
unklassifiziert	22	
Straße	23	
Wohngebiet	24	
Anwohnerstraße	25	
Service	26	
Weg	27	
Gehweg	28	
Rennstrecke	29	
Busspur	30	
Luft		
Flughafen	31	

Terminal	32	
----------	----	---

Hubschrauberlandeplatz	33	
------------------------	----	---

Zug


Zughaltestelle	34	
----------------	----	---

Straßenbahn	35	
-------------	----	---

Bahnkreuzung	36	
--------------	----	---

Zug-Wendeplatte	37	
-----------------	----	---

Annehmlichkeiten

Bankautomat	38	
-------------	----	---

Bank	39	
------	----	---

Bar	40	
-----	----	---

Fahrradvermietung	41	
-------------------	----	---















Biergarten	42	
------------	----	---















Bus	43	
-----	----	---

Café	44	
------	----	---




Autovermietung	45	
----------------	----	---

A. Anhang







Carsharing	46	
Kino	47	
College	48	
Gerichtshof	49	
Zahnarzt	50	
Arzt	51	
Trinkwasser	52	
Fast Food	53	
Fähre	54	
Feuerwehr	55	
Tankstelle	56	
Krankenhaus	57	
Kindergarten	58	
Nachtclub	59	

Parken	60	
Apotheke	61	
Polizei	62	
Briefkasten	63	
Post	64	
Gefängnis	65	
Kneipe	66	
Wertstoffhof	67	
Restaurant	68	
Schule	69	
Unterstand	70	
Studio	71	
Taxi	72	
Öffentliches Telefon	73	



A. Anhang

Theater	74	
Toiletten	75	
Tierarzt	76	





Geschäfte		
Einkaufsladen	77	
Einkaufszentrum	78	
Garten-Center	79	
Bekleidungsgeschäft	80	
Bäcker	81	
Getränkehandel	82	
Werkstatt	83	
Computergeschäft	84	
Konditor	85	
Möbelhaus	86	

Hifi-Geschäft	87	
Eiscafé	88	
Kiosk	89	
Wäscherei	90	
Outdoor-Geschäft	91	
Spielzeugwarenladen	92	















Kraft

Strommast	93	
Elektrizitätswerk	94	

Freizeit & künstlich erschaffen

Gemeinschaftsfläche	95	
Kran	96	
Garten	97	
Golfplatz	98	

A. Anhang

Marine	99	
Naturreservat	100	
Park	101	
Fußballplatz	102	
Spielplatz	103	
Schlittschuhbahn	104	
Stadium	105	
Wasserpark	106	
Vermessungspunkt	107	
Wasserspeicher	108	
Elektrizität	109	
Wasserkraft	110	
Windmühle	111	
Fabrik	112	

Tourismus

Museum

113



Freizeitpark

114



Hotel

115



Hostel

116



Gästehaus

117



Ferienhaus

118



Wohnwagen

119



Camping

120



Alpine Hütte

121



Historisch

Schlachtfeld

122



Burg

123



Monument

124



Ruinen

125



Wrack

126



Flächennutzung		
Wald	127	
Natur		
Bergspitze	128	
Klippe	129	
Strand	130	
Sport		
Fußball	131	
Golf	132	
Tennis	133	
Schwimmen	134	
Hockey	135	
Cricket	136	
Fahrrad	137	

Skifahren

138



Tischtennis

139



Volleyball

140



Transport

Stop

141



Ampel

142



Vorfahrt gewähren

143



Zebrastreifen

144



Kreuzung

145



Gefälle

146



Kreisverkehr

147



Wendeplatte

148



Blitzer

149



Bus

150



Rastplatz

151



Ford

152



Literaturverzeichnis

- [AHS05] K. Ali, K. Hartmann, T. Strothotte. Label Layout for Interactive 3D Illustrations. *The Journal of WSCG*, 13, 2005. (Zitiert auf Seite 33)
- [Aro89] S. Aronoff. Geographic Information Systems: A Managment Perspective. *WDL Puplications, Ottawa, Canada*, 1989. (Zitiert auf Seite 7)
- [Ase02] D. Asendorpf. Die vermessene Welt. *Die Zeit*, 14, 2002. (Zitiert auf Seite 1)
- [BDY06] K. Been, E. Daiches, C. Yap. Dynamic Map Labeling. *IEEE Transactions on Visualization and Computer Graphics*, 12, September/October 2006. (Zitiert auf den Seiten i, ii, vii, 5, 6, 21, 33, 34, 35, 36, 37, 38, 39, 40, 41, 43, 50, 58, 65, 75, 79, 87, 90 und 93)
- [BFH01] B. Bell, S. Feiner, T. Höllerer. View Management for Virtual and Augmented Reality. *ACM Symp. on User Interface Software and Technology (UIST 2001)*, pp. 101–110, 2001. (Zitiert auf Seite 33)
- [BM98] P. A. Burrough, R. A. McDonnell. *Principles of Geographical Information Systems*. Oxford University Press, 1998. (Zitiert auf den Seiten 7 und 8)
- [BNPW08] K. Been, M. Nöllenburg, S.-H. Poon, A. Wolff. Optimizing Active Ranges for Consistent Dynamic Map Labeling. *Annual Symposium on Computational Geometry Proceedings of the twenty-fourth annual symposium on Computational geometry*, SESSION 1:10–19, 2008. (Zitiert auf Seite 41)
- [Bre] URL http://de.wikipedia.org/wiki/Geographische_Breite. (Zitiert auf Seite 10)
- [Bur86] P. A. Burrough. *Principles of Geographical Information Systems for Land Resources Assessment*. Oxford University Press, 1986. (Zitiert auf Seite 7)
- [Car87] L. W. Carstensen. A comparison of simple mathematical approaches to the placement of spot symbols. *Carthographica*, Bd. 24, Nr.23:46–63, 1987. (Zitiert auf Seite 23)
- [Car89] J. Carter. On defining the geographic information system. In *In W.J. Ripple (ed.) Fundamentals of Geographic Information Systems: A Compendium*, pp. 3–7. ASPRS/ACSM, 1989. (Zitiert auf Seite 8)

- [Chr95] J. Christensen. *Managing Designs Complexity: Using Stochastic Optimization in the Production of Computer Graphics*. Ph.D. thesis, Center for Research in Computing Technology, Harvard University, 1995. (Zitiert auf den Seiten 21, 23 und 26)
- [CMS92] J. Christensen, J. Marks, J. Shieber. Labeling Point Features on Maps and Diagrams. *Technical Report Harvard University, Cambridge Massachusetts, USA*, tr-25-92, 1992. (Zitiert auf den Seiten 26 und 27)
- [CMS95] J. Christensen, J. Marks, J. Shieber. An Empirical Study of Algorithm for Point-Feature Label Placement. *ACM Transactions on Graphics*, Juli 1995. (Zitiert auf Seite 26)
- [Coo88] A. Cook. *Automated cartographic name placement using rule-based systems*. Ph.D. thesis, Department of Mathematics and Computer Science, Polytechnic of Wales, 1988. (Zitiert auf Seite 21)
- [Cow88] D. Cowen. GIS versus CAD versus DBMS: what are the differences? *Photogrammetric Engineering and Remote Sensing*, 54:1551–1554, 1988. (Zitiert auf Seite 8)
- [CRL08] G. Cravo, G. Ribeiro, L. Lorena. A greedy randomized adaptive search procedure for the point-feature cartographic label placement. *Computers & Geosciences*, 34:373–386, 2008. (Zitiert auf Seite 31)
- [CS01] V. Claus, A. Schwill. *Duden Informatik*. Dudenverlag, 2001. (Zitiert auf den Seiten 17 und 18)
- [Döo1] D. Dörschlag. *Automatische Platzierung von Objekten in Flächen einer Karte*. Master's thesis, Institut für Kartographie und Geoinformation, Universität Bonn, 2001. (Zitiert auf Seite 16)
- [DoE87] Handling Geographic Information. Department of Environment, HMSO, London, 1987. (Zitiert auf Seite 7)
- [DPP03] D. Dörschlag, I. Petzold, L. Plümer. Placing objects automatically in areas of maps. In *Durban International Conference ICC03*. 2003. (Zitiert auf Seite 16)
- [ECMS96] S. Edmonson, J. Christensen, J. Marks, S. Shieber. A general cartographic labeling Algorithm. *Manuscript, Harvard University, Cambridge, Massachusetts*, 1996. (Zitiert auf den Seiten 15, 16, 21 und 91)
- [Ell01] M. Ellsiepen. *Formalisierung kartographischen Wissens zur Schriftplatzierung in topographischen Karten*. Ph.D. thesis, Instituts für Kartographie und Geoinformation der Rheinischen Friedrich-Whilhelms-Universität Bonn, 2001. (Zitiert auf den Seiten 15 und 16)
- [Fre05] H. Freeman. Automated cartographic text placement. *Pattern Recognition Letters*, 26:287–298, 2005. (Zitiert auf Seite 29)

- [FW91] M. Formann, F. Wagner. A packing problem with applications to lettering of maps. In *Proceedings of the 7th Annual Symposium on Computational Geometry*. 1991. (Zitiert auf Seite 19)
- [Geo03] Geoinformation und moderner Staat. Bundesamt für Kartographie und Geodäsie: Eine Informationsschrift des Interministeriellen Ausschusses für Geoinformationswesen (IMAGI), 2003. (Zitiert auf Seite 1)
- [Gooa] URL <http://www.google.de/intl/de/earth/index.html>. (Zitiert auf den Seiten 2, 3 und 4)
- [Goob] GoogleMaps. URL <http://maps.google.de/>. (Zitiert auf den Seiten i, ii, 19 und 20)
- [Heb98] M. Heber. *Vorausberechnung reaktiver Datenstrukturen zur schnellen Beschriftung von Landkarten*. Master's thesis, Institut für Informatik, Universität Bonn, 1998. (Zitiert auf Seite 31)
- [Hir82] S. A. Hirsch. An Algorithms for Automatic Name Placement around point Data. *The American Cartographer*, 9:5–17, 1982. (Zitiert auf Seite 27)
- [HSKL05] L. Harrie, H. Stigmar, T. Koivula, L. Lehto. *An Algorithm for Icon Labelling on a Real-Time Map*. 11th International Symposium on Spatial Data Handling (SDH2005). Springer-Verlag Berlin Heidelberg, 2005. (Zitiert auf Seite 31)
- [Imh62] E. Imhof. Die Anordnung der Namen in der Karte. *Internationales Jahrbuch für Kartographie*, II, 1962. (Zitiert auf den Seiten 14, 15 und 16)
- [JS93] J. Marks, S. Shieber. The Computational Complexity of Cartographic Label Placement. Technical report, Harvard University, Cambridge, Massachusetts, 1991/1993. (Zitiert auf den Seiten 2 und 17)
- [Kni98] L. Knipping. *Beschriftung von Linienzügen*. Master's thesis, Freie Universität Bonn, 1998. (Zitiert auf den Seiten 15 und 16)
- [KOS97] M. van Kreveld, R. van Oostrum, J. Snoeyink. Efficient settlement selection for interactive display. In *AutoCarto 13: ACSM/ASPRS '97 Technical Papers*, pp. 287–296. 1997. (Zitiert auf Seite 34)
- [Kre94] W. Kresse. *Platzierung von Schrift*. Ph.D. thesis, Institut für Kartographie und Topographie der Rheinischen Friedrich-Wilhelms-Universität Bonn, 1994. (Zitiert auf Seite 26)
- [KT03] K. G. Kakoulis, I. Tollis. A unified approach to automatic label placement. *International Journal of Computational Geometry and Applications (IJCGA)*, 13(1):23–60, 2003. (Zitiert auf Seite 13)
- [Lae] URL <http://en.wikipedia.org/wiki/Longitude>. (Zitiert auf Seite 10)

- [Lano06] N. de Lange. *Geoinformatik in Theorie und Praxis*. Springer-Verlag Berlin Heidelberg, 2006. (Zitiert auf Seite 12)
- [LGMR05] P. A. Lonley, M. F. Goodchild, D. J. Maguire, D. W. Rhind. *Geographic Information Systems and Science*, volume 2nd Edition. John Wiley & Sons, Ltd, 2005. (Zitiert auf den Seiten 10, 12 und 13)
- [Lud09] M. Ludwig. CONVEXITY AND COMPUTATION, 2009. Studienarbeit. (Zitiert auf Seite 66)
- [Man89] U. Manber. *Introduction to Algorithms- A Creative Approach*. Addison-Wesley, 1989. (Zitiert auf Seite 17)
- [Moto7] K. Mote. Fast point-feature label placement for dynamic visualizations. *Information Visualization*, 6:249–260, 2007. (Zitiert auf den Seiten 13, 23, 29, 31, 32 und 55)
- [Oos99] R. W. van Oostrum. *Geometric Algorithms for Geographic Information Systems*. Ph.D. thesis, Universiteit Utrecht, 1999. (Zitiert auf Seite 34)
- [Ope] Open Street Map. URL <http://www.openstreetmap.de/>. (Zitiert auf den Seiten 2, 19, 20, 21 und 89)
- [OSS81] V. Ozemoy, D. Smith, A. Sichermann. Evaluation computerized geographic information systems using decision analysis. *Interfaces*, 11:92–98, 1981. (Zitiert auf Seite 8)
- [Par88] H. Parker. The unique qualities of a geographic information system: a commentary. *Photogrammetric Engineering and Remote Sensing*, 54:1547–1549, 1988. (Zitiert auf Seite 7)
- [Pet96] I. Petzold. *Textplatzierung in dynamisch erzeugten Karten*. Master’s thesis, Rheinische Friedrich-Wilhelms-Universität Bonn Institut für Informatik III, 1996. (Zitiert auf den Seiten 14, 16, 23, 24, 25, 27, 28, 29 und 30)
- [Peto3] I. Petzold. *Beschriftung von Bildschirmkarten in Echtzeit - Konzept und Struktur*. Master’s thesis, Hohen Landwirtschaftlichen Fakultät der Rheinischen Friedrich-Wilhelms-Universität zu Bonn, 2003. (Zitiert auf den Seiten 2, 3, 7, 9, 14, 15, 16, 17, 18, 22, 23, 24, 26, 28, 31 und 91)
- [PP97] I. Petzold, L. Plümer. Platzierung der Beschriftung in dynamisch erzeugten Bildschirmkarten. *Nachrichten aus dem Vermessungswesen*, Reihe I Heft Nr. 117, 1997. (Zitiert auf den Seiten 16, 21, 23 und 26)
- [PPH99] I. Petzold, L. Plümer, M. Heber. Label placement for dynamically generated screen maps. In *19th International Cartographic Conferences (ICC’99)*, pp. 893–903. 1999. (Zitiert auf Seite 31)

- [PS05] S. Poon, C. Shin. Adaptive zooming in point set labeling. *International Symposium on Fundamentals of Computational Theory*, 15, 2005. (Zitiert auf Seite 31)
- [QWC⁺09] H. Qu, H. Wang, W. Cui, Y. Wu, M.-Y. Chan. Focus+Context Route Zooming and Information Overlay in 3D Urban Environments. *IEEE Transactions on Visualization and Computer Graphics*, 15:No. 6, November/Dezember 2009. (Zitiert auf Seite 2)
- [RBDNo5] S. Roy, S. Bhattacharjee, S. Das, S. C. Nandy. A Fast Algorithm for Point Labeling Problem. In *17th Canadian Conference on Computational Geometry (CCCG'05)*, pp. 155–158. 2005. (Zitiert auf Seite 31)
- [Ric09] M. Richter. Geodaten sind ein gutes Geschäft, 2009. URL <http://www.zeit.de/online/2009/18/openstreetmap-anders>. (Zitiert auf Seite 20)
- [Sch95] V. A. Schmidt. Reine Forschung, praktische Resultate. *Die Zeit*, p. 45, 1995. (Zitiert auf Seite 2)
- [SMSE87] T. Smith, S. Mennon, J. Starr, J. Estes. Requirements and principles for the implementation and construction of large-scale geographic information systems. *International Journal of Geographical Information Systems*, 1:13–31, 1987. (Zitiert auf Seite 7)
- [Vaao8] M. Vaaranemi. *Hochdetaillierte Navigationskarten auf 3D Geländemodellen*. Master's thesis, Institut für Visualisierung und Interaktive Systeme Universität Stuttgart, 2008. (Zitiert auf den Seiten 12, 42 und 70)
- [Wol99] A. Wolff. *Automated Label Placement in Theory and Practice*. Ph.D. thesis, Fachbereich Mathematik und Informatik, Freie Universität Berlin, 1999. (Zitiert auf Seite 23)
- [Wol09] A. Wolff. Map Labeling Bibliography, 2009. URL <http://i11www.iti.uni-karlsruhe.de/map-labeling/bibliography/>. (Zitiert auf Seite 23)
- [WWKS01] F. Wagner, A. Wolff, V. Kapoor, T. Strijk. Three Rules Suffice for Good Label Placement. *Algorithmica Special Issue on GIS*, 30:334–346, 2001. (Zitiert auf Seite 29)
- [Zor86] S. Zoraster. Integer programming applied to the map label placement problem. *Cartographica*, Bd. 23 Nr. 4:16–27, 1986. (Zitiert auf Seite 26)

Alle URLs wurden zuletzt am 20.01.2011 geprüft.

Erklärung

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

(Ilona Heurich)