Ubiquitous Navigation: Skalierbare ortsbezogene Dienste in Gebäuden

Martin Werner

Dissertation
an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig–Maximilians–Universität München

vorgelegt von
Martin Werner

Tag der Einreichung: 5. Juni 2012

Ubiquitous Navigation: Skalierbare ortsbezogene Dienste in Gebäuden

Martin Werner

Dissertation
an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig–Maximilians–Universität München

vorgelegt von
Martin Werner

1. Berichterstatter:	Prof. Dr. Claudia Linnhoff-Popien
2. Berichterstatter:	Prof. Dr. Alexander Schill
Tag der Einreichung:	5. Juni 2012
Tag der Disputation:	19. Juli 2012

Eidesstattliche Versicherung

(siehe Promotionsordnung vom 12.07.11, § 8, Abs. 2 Pkt. 5)

Hiermit erkläre ich an Eides statt, dass die Dissertation von mir selbstständig, ohne unerlaubte Beihilfe angefertigt ist.

Martin Werner

Zusammenfassung

Ortsbezogene Dienste (Location-based Services, LBS) sind in den letzten Jahren durch die weite Verfügbarkeit von GPS zu einem Massenphänomen gewachsen. Insbesondere für die Steuerung von mobilen Endgeräten wird mehr und mehr Kontextinformation hinzugenommen, da sowohl die Bedienbarkeit als auch die Informationsdichte auf den kleinen Smartphones im Kontrast zur Informationsfülle des Internets stehen. Daher werden vielfach Dienste nicht mehr allein auf der Nutzereingabe basierend erbracht (reaktiv), sondern bieten dem Nutzer relevante Informationen vollautomatisch und kontextabhängig (proaktiv) an.

Durch die proaktive Diensterbringung und ortsbezogene Personalisierung wird die Benutzbarkeit der Dienste verbessert. Leider kann man derzeit solche Dienste nur außerhalb von Gebäuden anbieten, da zum Einen kein einheitliches und günstiges Positionierungssystem verfügbar ist, und zum Anderen die notwendigen Kartendaten nicht vorliegen. Vor allem bei den Kartendaten fehlt es an einfachen Standards und Tools, die es dem Eigentümer eines Gebäudes ermöglichen, qualitativ hochwertige Kartendaten zur Verfügung zu stellen.

In der vorliegenden Dissertation werden einige notwendige Schritte zur Ermöglichung ubiquitärer und skalierbarer Indoornavigation vorgestellt. Hierbei werden die Themenfelder Positionierung, Modellierung und Kartographie, sowie Navigation und Wegfindung in einen umfassenden Zusammenhang gestellt, um so eine tragfähige, einfache und skalierbare Navigation zu ermöglichen. Zunächst werden einige Verbesserungen an Terminal-basierten WLAN-Indoorpositionierungssystemen vorgeschlagen und diskutiert, die teils auf der Verwendung neuer Sensorik aktueller Smartphones basieren, und teils auf der Verwendung besonders kompakter Umgebungsmodelle auf mobilen Endgeräten. Insbesondere werden Verfahren vorgeschlagen und evaluiert, mit denen sich aus typischen CAD-Daten mit geringem Bearbeitungsaufwand die notwendigen Zusatzinformationen für eine flächendeckende Indoornavigation modellieren lassen. Darüber hinaus werden Methoden untersucht, die diese semantischen Erweiterungen teil- bzw. vollautomatisch aus Zeichnungen extrahieren können. Ausgehend von dem Ziel, flächendeckende Navigation basierend auf CAD-Daten zu ermöglichen, stößt man auf das Problem, eine Menge an interessanten Punkten so zu ordnen, dass der Reiseweg kurz ist. Dieses Problem ist mit dem NP-vollständigen Travelling-Salesman-Problem verwandt. Allerdings ist die geometrische Situation in Gebäuden derart komplex, dass sich die meisten derzeit bekannten heuristischen Lösungsalgorithmen für das

Travelling-Salesman-Problem nicht ohne Weiteres auf die Situation im Inneren von Gebäuden übertragen lassen. Für dieses Problem wird ein heuristischer Algorithmus angegeben, der in linearer Laufzeit kleinere Instanzen des Problems mit akzeptablen Fehlern lösen kann. Diese Verfahren wurden im Rahmen eines Projekts mit dem Flughafen München erarbeitet und umgesetzt. In diesem Projekt sollten die ungelösten Probleme einer bereits existierenden kleinflächigen Demonstrator-Implementierung eines Fluggastinformationssystems gelöst werden [Rupp 09]. Auf diesen Algorithmen und Verfahren basiert die Navigationskomponente des Fluggastinformationssystems InfoGate [Flug 11b], das die flächendeckende Navigation in den Terminals des Flughafen München mit verteilten Anzeigesystemen ermöglicht und seit dem 6. Juni 2011 im Produktivbetrieb ist [Flug 11a]. So konnten die Verfahren dieser Arbeit in einer real existierenden, komplexen Umgebung evaluiert werden.

Abstract

Location-based services have become a mass phenomenon in the last few years, mainly due to the wide adoption of GPS technology. Particularly for the control of mobile devices more and more context information is being used, because the ease of use as well as the information presentation is a contrast to the multitude of information available on the Internet. As a consequence, services are emerging from reactive interaction, where mainly the user input is considered for the service, to proactive interaction, where relevant information is automatically identified and presented to the user.

Proactive services and location-based personalization enable a better service experience. Unfortunately such services are nowadays only available outside buildings, because there is no common and cheap positioning system and also map information is rare. Concerning map data there is a lack of standards and tools which enable the owners of a building to create valuable maps themselves.

In this thesis some necessary steps towards the creation of ubiquitous and scalable indoor navigation are introduced. Therefore, the topics of positioning, modeling, and cartography, as well as navigation and way-planning are considered in a large context, as to enable a simple, sustainable, and scalable navigation.

First, some improvements in the area of terminal-based positioning systems based on WLAN are proposed which are based on the integration of newly adopted sensor technology in smartphones and the usage of exceptionally compact environmental models. Moreover, a method for the generation of indoor environmental models out of typical CAD data with slight modeling effort is proposed. These models are based on the concept of semantic overlays and can in many cases be automatically extracted. During the conception of a spatially inclusive and comprehensive indoor navigation system one comes across the problem to order a set of points of interests in a way, such that the complete travel distance is short. This problem is comparable to the NP-complete traveling salesman problem. However, the geometric structure of buildings is such complex that most well-known approximation algorithms for the traveling salesman problem are not applicable inside buildings. For this problem a heuristic algorithm is developed which is able to solve small instances of the problem with acceptable quality.

These proposals have been evaluated in conjunction with Munich airport. The aim of the project was the solution of the unsolved problems of an existing, small, demonstrative implementation of a passenger information system first published in [Rupp 09]. Based on the models and algorithms of this thesis, a na-

vigation component for the passenger information system InfoGate [Flug 11b] was established which enables the area-wide navigation inside the terminal buildings of Munich airport with distributed display systems. This system is in production since June 2011 [Flug 11a] and it was possible to evaluate the algorithms of this thesis in a real complex setting.

Inhaltsverzeichnis

Danksagung

Einen ganz besonderen Dank möchte ich zunächst an Frau Prof. Dr. Claudia Linnhoff-Popien richten, die mich während der Betreuung der vorliegenden Dissertation stets unterstützt hat. Herrn Prof. Dr. Alexander Schill danke ich für die Übernahme des Koreferats und die vielen, wertvollen Hinweise zu meiner Arbeit.
Weiter möchte ich mich bei allen Kollegen am Lehrstuhl für Mobile und Verteilte Systeme für unzählige anregende Diskussionen und Gespräche bedanken. Ein besonderer Dank gilt dabei denen, ohne deren Unterstützung die vorliegende Arbeit nicht möglich gewesen wäre. Namentlich bedanke ich mich für wertvolle Diskussionen und fruchtbare Zusammenarbeit bei Ulrich Bareth, Florian Dorfmeister, Markus Duchon, Michael Dürr, Florian Gschwandtner, Moritz Kessel, Robert Lasowski, Marco Maier, Philipp Marcus, Johannes Martens, Peter Ruppel, Kim Schindhelm, Mirco Schönfeld, Anna Schwanengel, Kay Weckemann, Kevin Wiesner und Stephan Verclas.
Weiter möchte ich mich für die Ermöglichung der realen Umsetzung meiner Ideen bei den Mitarbeitern des Flughafens München bedanken. Nicht zuletzt gilt ein ganz besonderer Dank meiner Familie, die mich stets mit Verständis und Liebe unterstützt hat.

1 Einführung

Ortsbezogene Dienste (Location-based Services, LBS) gewinnen seit einigen Jahren zunehmend an Bedeutung. Die weite Verbreitung von Positionierungssystemen wie GPS, aber auch die Verbreitung von internetfähigen Endgeräten hat perfekte Voraussetzungen geschaffen, um diese neue Art von Diensten im Alltag der Menschen zu verankern. Ein solcher ortsbezogener Dienst ist ein Dienst, der neben Nutzereingaben auch den Aufenthaltsort des Nutzers zur Diensterbringung verwendet. Eine häufige Anwendung der Ortsinformation dient der Filterung und Bewertung von Informationen: Der Nutzer bekommt nur die Informationen angeboten, die für seinen aktuellen Standort relevant ist. Dadurch lässt sich insbesondere die Bedienbarkeit von Smartphones und Tablet-PCs verbessern, bei denen die Eingabe von Text mittels einer Bildschirmtastatur sehr mühsam ist.

Ortsbezogene Dienste sind eine spezielle Form der allgemeineren Idee kontextabhängiger Dienste. Kontext bezeichnet ganz allgemein jegliche Information, die geeignet ist, die Situation des Nutzers zu charakterisieren. Eine weit verbreitete Definition stammt von Dey et. al. [Dey 00]:

„Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves.“ [1]

Typische Quellen von Kontextinformationen umfassen direkt messbare Informationen wie den Ort, die Zeit, das Wetter, aber auch externes Wissen aus Kalendern, Anruflisten, E-Mailkonten, sozialen Netzen und vieles mehr. Die Modellierung von Kontext, die Ableitung von höherwertigen Kontextinformationen aus den direkt bestimmbaren Kontextinformationen und die Folgerung möglicherweise gewünschter Dienstaktivitäten ist ein sehr aktives Forschungsgebiet [Weis 09]. Die Probleme, die dabei gelöst werden, stehen in direktem Zusammenhang mit aktuellen Visionen der Informatik wie dem „Web-of-Things“, dem „Ubiquitous Computing“, den „Smart-Spaces“ und dem „Semantic Web“. Der Aufenthaltsort ist eine Art von Kontext, der sehr gut geeignet ist, wichtige Informationen über die Situation des Nutzers zu bestimmen, und verdient daher eine eigene Behandlung.

Die älteste und grundlegendste dieser Visionen ist wohl die des „Ubiquitous

[1] Kontext bezeichnet jegliche Information, die geeignet ist, die Situation einer Entität zu charakterisieren. Eine Entität ist eine Person, ein Ort oder ein Objekt, das als relevant für die Interaktion von Nutzer und Anwendung erachtet wird, insbesondere der Nutzer und die Anwendung selbst.

Computing“. Mark Weiser beginnt seinen wegweisenden Artikel über die Informatik des 21. Jahrhunderst mit dem folgenden Satz [Weis 91]:
„The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it.“[2]
Genau diese Vision beginnt sich durch kontextabhängige Dienste zu erfüllen, denn diese Dienste können beispielsweise im Hintergrund und gänzlich ohne Nutzerinteraktion funktionieren, indem sie die Situation des Nutzers beobachten und zur Diensterbringung auswerten. Waren klassische Dienste der Informationstechnologie immer „reaktiv“ und reagierten stets direkt auf Befehle des Nutzers, so lassen sich unter Hinzunahme von Kontext- und Ortsinformationen Dienste auch „proaktiv“ erbringen. Dabei wird der Nutzer aktiv über neuerdings relevante Informationen informiert, ohne vorher eine Anfrage an den Dienst gestellt zu haben. So wird ein kontextabhängiges System nur dann Informationen über Benzinpreise sammeln und diese analysieren, wenn der Nutzer auch gerade mit einem Auto unterwegs ist, oder Informationen über Zugverspätungen und Gleiswechsel nur dann verarbeiten, wenn im Kalender eine entsprechende Zugfahrt eingetragen ist.
In der vorliegenden Arbeit geht es um eine sehr klassische Kategorie ortsbezogener Dienste: Die Navigation. Für den Außenbereich ist diese bereits sehr hoch entwickelt. In Gebäuden sind aber viele Dinge so grundlegend anders, dass man bis heute an einzelnen Problemen bei der Navigation in diesem Bereich arbeitet. Die vorliegende Dissertation enthält aktuelle Forschungsergebnisse, die im Zusammenhang mit einer der ersten realen Anwendungen zur flächendeckenden, skalierbaren Indoor-Navigation entstanden sind.
Bei der Navigation geht es im Grunde immer um die Frage, an welchem Ort man sich befindet, wie die Umgebung aussieht und zu interpretieren ist, und was das für den Nutzer bedeutet. Vor allem die Zielführung, bei welcher der Nutzer unterstützt wird, ein bestimmtes Ziel zu einer bestimmten Zeit zu erreichen, ist von besonderem Interesse. Die Navigation lässt sich sinnvollerweise in drei einzelne Schritte unterteilen: Die *Positionierung*, bei der die gegenwärtige Position des Nutzers bestimmt wird und diese auch in Relation zu einem Modell der Umgebung gesetzt wird. Die *Wegfindung*, bei der Wege zwischen verschiedenen Positionen berechnet werden, die eine gewisse Optimalität (schnelle Route, kurze Strecke, etc.) erfüllen, und schließlich die vielen Fragestellungen bezüglich der *Nutzerschnittstellen*: Wie erfährt der Dienst die aktuellen Ziele des Nutzers, wie werden dem Nutzer die Navigationsinformationen zur Verfügung gestellt, und wie kann der Nutzer das Gesamtsystem bedienen?
Diese verschiedenen Aspekte der Navigation, Positionierung, Kartographie und Wegfindung werden in dieser Arbeit in einen umfassenden Zusammenhang gestellt. Denn im Inneren von Gebäuden ist eine isolierte Betrachtung der drei Aspekte kaum möglich, da die Komplexität der Umgebung nicht, wie im Außenbereich, vernachlässigt werden kann.

[2] Die tiefgreifendsten Technologien sind diejenigen, die entschwinden. Sie verweben sich in das Gefüge des Alltags, bis sie von diesem nicht mehr unterschieden werden können.

Die Bestimmung der Position eines mobilen Endgeräts ist in den letzten Jahren immer besser und immer zuverlässiger möglich geworden. Insbesondere der Kostenverfall bei GPS-Technologie mit der Folge der Verbreitung GPS-fähiger Geräte, hat die flächendeckende Positionierung mobiler Endgeräte zu einer wirtschaftlich sinnvollen Anwendung heranwachsen lassen. Auch Geräte ohne GPS können ihre Position mittlerweile über kostenlos angebotene Dienste aus den Mobilfunkzelleninformationen und den umliegenden WLAN-Netzen ableiten, sobald sie über eine Internetverbindung verfügen. Doch auch diese Dienste sind erst durch die weite Verbreitung von GPS möglich geworden, denn die notwendigen Daten werden typischerweise von GPS-fähigen Smartphones gesammelt.

Im Gegensatz zum Außenbereich gibt es für den Innenbereich noch keine ausgereiften Positionierungstechnologien, die in Bezug auf Qualität, Verbreitung und Einfachheit mit den Systemen für den Außenbereich vergleichbar wären. Denn die Systeme, die einer breiten Öffentlichkeit zugänglich wären, basieren zumeist auf Messungen von Funksignalstärken und Funklaufzeiten, und unterliegen dabei erheblichen Störungen durch Mehrwegeausbreitung, Dämpfung und Messfehler. Diese Systeme liefern in der Praxis bestenfalls eine Positionierung mit Abweichungen in der Größenordnung von Räumen und eignen sich beispielsweise selten für eine durchgehende Zielführung mit Anweisungen. Außerdem sind solche Systeme bereits gegenüber kleineren Änderungen in der Umgebung sehr anfällig, weshalb eine aufwändige Kalibrierung regelmäßig notwendig wird. Andere Systeme bieten hervorragende Positionierungsgenauigkeiten im Zentimeterbereich, basieren aber in der Regel auf spezieller Technologie (Ultraschall, Ultrabreitband-Funk, Laser, etc.) und stehen daher nur einer kleinen, geschlossenen Nutzergruppe auf einer kleinen Fläche offen, die mit der notwendigen Infrastruktur ausgestattet worden ist.

Die meisten Positionierungssysteme im Inneren von Gebäuden sind mit so großen Fehlern behaftet, dass eine Fehlerkorrektur notwendig wird. Dabei kann mit teilweise sehr komplexen Filterungsmechanismen für die Schätzung des Zustandes linearer und häufig auch nicht-linearer, stochastischer Systeme eine bessere Positionsbestimmung durch Karteninformationen und eine ausgleichende Behandlung von Fehlern unterschiedlicher Charakteristik erreicht werden.

Eine einfache Idee für die Minimierung von Fehlern kommt aus der Beobachtung, dass sich durch die gleichzeitige Verfolgung mehrerer hypothetischer Wege, die innerhalb der zu erwartenden Messgenauigkeit liegen, über die Zeit nur eine kleine Menge an Hypothesen nicht durch das Durchtreten von Hindernissen wie Wänden in einem Raumplan von selbst ausschließt und sich damit eine sehr genaue Positions- und Trajektorienbestimmung realisieren lässt. Solche Verfahren werden im Außenbereich beim Map-Matching eingesetzt [Heid 11], aber insbesondere auch bei der Fusion von Messwerten mit Karteninformationen in der Indoor-Positionierung.

Dies ist nur eines von vielen Beispielen, weshalb gerade in Gebäuden die Infor-

mationen über die Umgebung bereits bei der Positionierung eine zentrale Rolle spielen. Deshalb ist es auch von essentieller Bedeutung, solche Karteninformationen mit möglichst einfachen und klaren Methoden zu beschreiben. Denn eine solche Beschreibung muss vom Eigentümer des Gebäudes selbst erstellt werden.

Dieser Text ist wie folgt aufgebaut: Kapitel 2 erläutert zunächst die Grundlagen von Positionierungssystemen und gibt dann einen Überblick über derzeitige Ansätze, in Gebäuden eine Position zu bestimmen. In Kapitel 3 werden neue Ansätze vorgestellt, die die Positionsbestimmung in Gebäuden einfacher und genauer zu machen. Diese Ansätze basieren teils auf der Verwendung zusätzlicher Sensorik und teils auf der Verwendung visueller Informationen für probabilistische Ortsbestimmungen. Im folgenden Kapitel 4 werden bestehende Ansätze zur Modellierung von Gebäudeplänen diskutiert und zwei Ansätze aufgezeigt, die, mit einer radikalen Vereinfachung im Vergleich zu den verwandten Arbeiten, die Komplexität der Erstellung von semantischen Gebäudeplänen so stark vereinfachen, dass nicht nur Spezialisten in der Lage sind, solche Pläne zu bearbeiten. Dabei entsteht natürlich eine semantische Lücke zu den hochkomplexen, leistungsfähigen Modellen aus diesen Arbeiten, die aber durch die Konzeption und Implementierung zusätzlicher Algorithmen zumindest in Bezug auf die Anforderungen einer Indoor-Navigationsanwendung geschlossen werden kann. Insbesondere wird eine Methode vorgestellt, mit der eine flächendeckende Navigation trotz der im Hinblick auf Einheitlichkeit und Semantik eher schlechten CAD-Daten des Flughafens München mit einem angemessenen Bearbeitungsaufwand ermöglicht wurde. Im Kapitel 5 werden zwei Methoden angegeben, mit denen das NP-harte, geordnete Traveling-Salesman-Problem in Gebäuden mit vertretbaren Laufzeiten approximiert werden kann. Die besondere Komplexität in diesem Bereich beruht darauf, dass alle bekannten Approximationen des Traveling-Salesman-Problem, die eine nicht-exponentielle Laufzeit haben, eine Vereinfachung des Problems vornehmen müssen. Oft verwendet man an dieser Stelle geometrische Eigenschaften, die für typische Landkarten gelten, aber innerhalb von Gebäuden häufig verletzt werden: Die Wegegraphen in der Indoornavigation sind nicht metrisch, nicht symmetrisch und nicht euklidisch.

Insgesamt stellt diese Dissertation einen Beitrag zur Skalierbarkeit und Ubiquität von Indoor-Navigation dar. Der Aspekt der Skalierbarkeit bezieht sich sowohl auf den Rechenaufwand für die Wegführung, als auch auf den Aufwand, Kartendaten zu erzeugen und zu verwalten. Der Aspekt der Ubiquität drückt sich darin aus, dass die Forschungsergebnisse und Entwicklungen, die im Folgenden dargestellt werden, nicht auf aktive Unterstützung durch eine Infrastruktur angewiesen sind. Die notwendigen Berechnungen sollen und können auf den Mobiltelefonen der Nutzer durchgeführt werden.

Viele Aspekte dieser Arbeit konnten im Rahmen eines Projekts mit dem Flughafen München umgesetzt und in einer realen Umgebung getestet werden. So wurde mit dem Konzept zur Erstellung semantischer Overlays durch Flugha-

fenmitarbeiter ohne besondere Ausbildung oder Dokumentation der gesamte Flughafen München modelliert. Als Ergebnis steht das Informations- und Navigationssystem „InfoGate“ [Flug 11b] den Benutzern des Flughafens seit dem 6. Juni 2011 zur Verfügung [Flug 11a], dessen Navigationsfunktionalität auf den in dieser Arbeit vorgestellten Ergebnissen basiert.

2 Grundlagen der Indoor-Positionierung

Die Indoor-Positionierung beschäftigt sich mit der Frage, wie man in Gebäuden den Ort eines mobilen Endgerätes bestimmen kann und wie man dann eine Navigationshilfe anbieten kann, wie sie für den Außenbereich durch die Satelliten-Navigation mit GPS und die Verfügbarkeit qualitativ hochwertiger Karten bereits möglich ist. Neben der Frage nach einer Lösung, die eine Navigation in Gebäuden tatsächlich ermöglicht, beschäftigt man sich natürlich auch mit der Frage, wie man die Positionierung im Innen- und Außenbereich, die in der Regel auf unterschiedlichen Technologien beruht, zusammenführen kann und eine durchgängige „*Seamless*“-Positionierung ermöglichen kann.

Zunächst erscheint es auch etwas nutzlos, eine Navigation im Inneren von Gebäuden zu konzipieren, denn die meisten Gebäude werden nur von einer kleinen Nutzergruppe und häufig regelmäßig besucht, sodass mit Hilfe von klassischer Beschilderung und Plänen eine Orientierung meist problemlos möglich ist. Natürlich gibt es Gebäude, bei denen es wünschenswert ist, dass Besucher sich einfacher zurecht finden: Komplexe öffentliche Gebäude wie Flughäfen, Bahnhöfe, Museen und Krankenhäuser.

Aus einer ganz anderen Perspektive besteht auch ein sehr starkes Interesse an der Bestimmung von Positionen in Gebäuden: Ortsbezogenes Sicherheitsmanagement, Qualitätskontrolle und Evakuierung von Gebäuden mit extrem hoher Gefährdung für Personen. In diesem Bereich ist die Bestimmung eines Ortes häufig nur Mittel zum Zweck, die „Aktivität“ einer mobilen Entität (sei es nun eine Person, eine Maschine oder ein sonstiges Objekt) zu bestimmen. So liegt den Anwendungen des „Ambient Assisted Living“ die Idee zu Grunde, älteren Menschen die Möglichkeit zu geben, in ihrer Wohnung auch in Situationen weiterzuleben, in denen heutzutage eine ständige Überwachung notwendig wird und eine Einweisung in ein Pflegeheim oft die Folge ist [The 12a]. In diesen Szenarien ist natürlich weniger der Aufenthaltsort einer Person in seiner Wohnung von Interesse, sondern vielmehr das Erkennen von relevanten Abweichungen von der täglichen Routine. Auch in der Qualitätskontrolle kann eine automatische Erfassung unterschiedlicher Aktivitäten durch Positionierungstechnologie zu einer wesentlichen Verbesserung der Qualität führen, weil automatisch digital erfasst werden kann, ob eine bestimmte Menge an Tätigkeiten auch tatsächlich vorgenommen wurde. So wird in [Zinn 09] eine Anwendung beschrieben, bei der mittels Positionsdaten verschiedene Aufgaben bei der Qualitätskontrolle eines Fahrzeuges mit hoher Präzision erkannt werden.

Das wichtigste Problem, das die Verbreitung solcher Systeme in den letzten Jahren eingeschränkt hat, ist natürlich der Schutz der Privatsphäre. Die entscheidende Frage, die situativ stets anders zu bewerten und zu beantworten ist, lautet: Liefert das System einen ausreichenden Vorteil für denjenigen, dessen Privatsphäre durch die permanente digitale Überwachung beeinträchtigt ist? Leider entsteht der Vorteil häufig nicht tatsächlich und direkt demjenigen, der die Nachteile zu tragen hätte, sodass konsequenterweise auch diese Art von Systemen bisher nur eine untergeordnete wirtschaftliche Bedeutung hat. Das wissenschaftliche Interesse daran ist aber nach wie vor ungetrübt.
Eine weitere, häufige Anwendung für Positionierung in Gebäuden ist das „Tracking“ von Gegenständen. Es kann sehr nützlich sein, wenn man über den Ort und die Bewegungen von Gütern eine korrekt geführte Dokumentation hat. Denn so können Prozesse in der Logistik digital gesteuert und optimiert werden.
Auf Grund dieser doch sehr unterschiedlichen Anwendungsszenarien wurden sehr viele verschiedene Positionierungs- und Navigationstechniken entwickelt, die alle spezielle Vor- und Nachteile haben.
Der folgende Abschnitt 2.1 beschreibt die grundlegenden Algorithmen und Verfahren, die für die Indoor-Positionierung von Bedeutung sind. Abschnitt 2.2 beschreibt dann, wie man unterschiedliche Positonierungsalgorithmen zu einem besseren Gesamtergebnis kombinieren kann. Abschnitt 2.3 beschreibt Eigenschaften von Positionierungssystemen, die für den Vergleich und die Auswahl eines geeigneten Verfahrens und Systems relevant sind. Abschnitt 2.4 gibt abschließend einen Überblick über Beispiele aktueller Positionierungssysteme.

2.1 Methoden der Positionierung

Wie bereits ausgeführt, existiert eine große Vielfalt an Positionierungssystemen. Dennoch gibt es in der Methodik, mit der völlig unterschiedliche und zunächst nicht vergleichbare Positionierungssysteme arbeiten, erstaunliche Überdeckungen. Deshalb werden im folgenden Abschnitt die Vorgänge bei der Positionierung systematisch und ohne konkrete Beschränkung auf eine spezielle physikalische Größe oder eine Anwendung formuliert.
Eine wesentliche Schwierigkeit im Umgang mit physikalischen Größen besteht darin, dass sie in der Regel mit einem Messfehler oder auch einem Diskretisierungsfehler behaftet sind. So wird etwa bei der Lateration angenommen, dass das mobile Endgerät sich auf Kreisen mit gemessenen Radien um die jeweiligen Referenzpunkte befindet. In der Praxis werden sich diese Kreise aber nicht in einem Punkt schneiden, weil die Radien der Kreise gemessen werden und daher mit Fehlern behaftet sind. Um auf eine solide und wahrscheinlichkeitstheoretisch fundierte Weise damit umzugehen, stellt der folgende Abschnitt die Methode der kleinsten Quadrate von Gauß vor, mit der sich fehlerbehaftete, überbestimmte, lineare Gleichungssysteme mit einfachen Mitteln lösen lassen. Mit „Lösung“ bezeichnet man in diesem Zusammenhang die Bestimmung des

wahrscheinlichsten Wertes der zu bestimmenden Größe.
Auf dieser Methode aufbauend, werden zirkuläre und hyperbolische Lateration sowie Angulation eingeführt und erläutert, wie die Methode der kleinsten Quadrate in diesem nicht-linearen Umfeld in einer allgemeinen Vorgehensweise verwendet werden kann.

2.1.1 Ausgleichsrechnung und die Methode der kleinsten Quadrate

Die Beobachtung der Welt durch die Messung physikalischer Größen unterliegt im Allgemeinen verschiedenen Fehlern, da ein Messgerät nicht unendlich präzise hergestellt und geeicht werden kann. Darüber hinaus werden die Messungen in der Regel digital oder manuell weiterverarbeitet und es kommt zu Diskretisierungsfehlern, etwa durch die endliche Gleitpunktdarstellung einer kontinuierlichen Größe wie der Temperatur. Somit wird eine zuverlässige Beobachtung der Welt in der Regel nur durch längere Messreihen ermöglicht, aus der dann ein statistisch sinnvoller Wert – im einfachsten Fall ein Durchschnitt – für die zu messende Größe ausgerechnet wird.
Liegt ein linearer Zusammenhang zwischen gemessenen Größen und zu bestimmender Größe vor, lässt sich eine solche Messreihe als möglicherweise überbestimmtes lineares Gleichungssystem

$$Ax = b$$

schreiben. Für diese Situation haben schon Karl Friedrich Gauß und Adrien Marie Legendre im Jahr 1795 gleichzeitig und unabhängig eine Methode gefunden, mit der sich der wahrscheinlichste Wert x berechnen lässt. Da eine perfekte Lösung obiger Gleichung wegen der Messfehler im Allgemeinen nicht möglich ist, sei eine gute Näherungslösung ein solches x, welches das Quadrat der euklidischen Norm der Fehlergleichungen $r(x) = b - Ax$ minimiert:

$$||r(x)||^2 = (b - Ax)^T(b - Ax) = x^T A^T Ax - 2x^t A^t b + b^T b \rightarrow \min$$

Es lässt sich zeigen, dass (unter der Voraussetzung, dass A maximalen Rang hat, also die Spalten von A linear unabhängig sind) eine Lösung der Normalengleichung

$$A^T Ax = A^T b$$

die Fehlergleichung minimiert und dass umgekehrt jedes x, welches die Fehlergleichungen minimiert, auch die Normalengleichung erfüllt. Das wichtigste Ergebnis zur wahrscheinlichkeitstheoretischen Rechtfertigung dieser Minimierungsmethode ist der der Satz von Gauß-Markov. Dieser besagt, dass durch das beschriebene Vorgehen eine beste, lineare, unvoreingenommene Schätzung entsteht. Die wahrscheinlichkeitstheoretischen Voraussetzungen für diesen Satz, nämlich, dass die Fehler den Erwartungswert 0 haben, nicht korrelliert sind

und gleiche Varianz haben, ist bei der Verwendung gleichartiger physikalischer Messgrößen in der Regel erfüllt. Für Details sei an dieser Stelle auf die gängigen Bücher zu numerischer Mathematik oder Ausgleichsrechnung verwiesen [Freu 07, Schw 04, Ludw 71]. Auch das auf Navigation spezialisierte Buch [Hoff 03] enthält ein weiterführendes Kapitel über Ausgleichsrechnung in der Navigation, welches auch eine rekursive Variante und als Verallgemeinerung das Verfahren des diskreten Kalman-Filters (siehe auch Abschnitt 2.2) einführt.
Handwerklich ist ein solches Verfahren sehr einfach zu konstruieren: Zunächst kann man dafür sorgen, dass die Matrix A maximalen Rang hat, indem linear abhängige Spalten entfernt werden. Ist das erreicht, so lässt sich die notwendige Bedingung $\nabla r(x) = 0$ für ein Minimum als lineares Gleichungssystem mit positiv definiter, symmetrischer Matrix $A^T A$ schreiben:

$$\nabla r(x) = 2A^T Ax - 2A^T b = 0 \Leftrightarrow A^T Ax = A^T b$$

Dass der so gefundene Wert tatsächlich die Norm des Residuums minimiert, liegt daran, dass die Hessesche Matrix des Residuums sich als $2A^T A$ berechnet und somit – wie $A^T A$ auch – positiv definit ist. Im Allgemeinen findet man die Lösung x, die die Fehlergleichungen minimiert, indem man die Normalengleichung löst. Dies ist sehr einfach und auch numerisch stabil möglich, da die Matrix $A^T A$ symmetrisch und positiv definit ist, eine Inverse $(A^T A)^{-1}$ also existiert. Die Inverse wird wegen ihrer häufig problematischen Konditionszahl in der Regel nicht explizit berechnet, sondern es wird zur Lösung des Gleichungssystems eine unitäre Transformation verwendet [Freu 07].
Dazu lässt sich das Gleichungssstem mit unitären Transformationen (in der Praxis kann man beispielsweise Householder-Transformationen oder Givens-Rotationen verwenden) auf obere Dreiecksgestalt bringen.

$$A^T A = QD \text{ mit } D = \begin{pmatrix} R \\ \hline 0 \end{pmatrix} \text{ und } R \text{ obere Dreiecksmatrix}$$

Dann ist die obere Dreiecksmatrix R wegen maximalen Ranges von A invertierbar und man kann durch geeignete Substitution die Lösung direkt ausrechnen. Insgesamt lässt sich also die Berechnung der wahrscheinlichsten Lösung für ein überbestimmtes, lineares Gleichungssystem auf die Lösung eines anderen, nicht überbestimmten, linearen Gleichungssystems mit guten numerischen Eigenschaften zurückführen.

2.1.2 Lateration

Mit Lateration bezeichnet man den Vorgang, aus gemessenen Entfernungen einer mobilen Entität zu festen Punkten mit bekannten Koordinaten die Koordinaten der mobilen Entität auszurechnen. Wie in Abbildung 2.1 durch die gestrichelten Linien dargestellt, unterliegen die gemessenen Entfernungen typi-

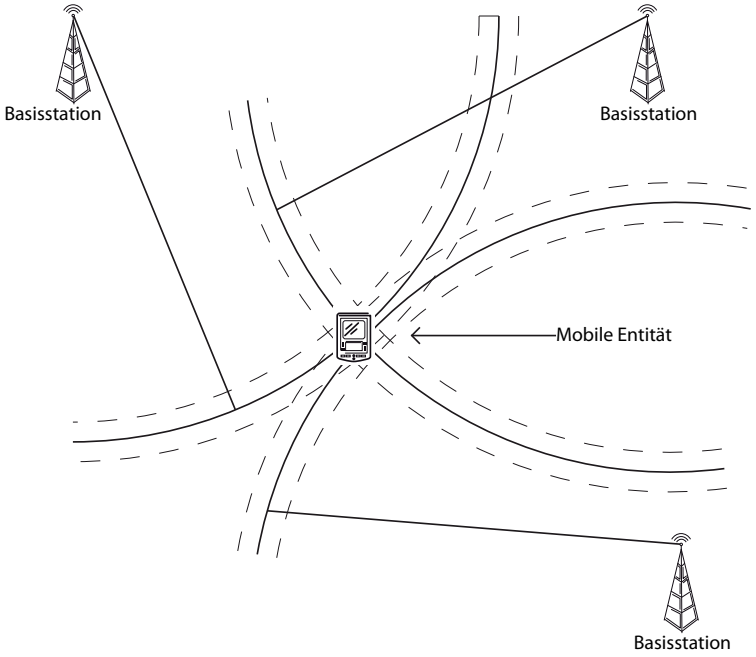


Abbildung 2.1: Beispiel für Lateration: Drei mit Fehlern behaftete Entfernungsschätzungen zu Basisstationen mit bekanntem Ort ergeben einen ungefähren Aufenthaltsort der mobilen Entität

scherweise Messfehlern. Obwohl drei Distanzen zu bekannten Referenzpunkten den Aufenthaltsort als den eindeutigen Schnittpunkt dreier Kreise um die Referenzpunkte definieren, wird eine solche Ortsauflösung wegen der Messfehler in der Praxis nicht möglich sein.

Im Prinzip muss also eine gute Näherungslösung für ein Gleichungssystem gefunden werden, welches eine exakte Lösung im Allgemeinen nicht zulässt. Es soll nämlich gefordert werden, dass der gesuchte Ort $p = (x, y)$ der mobilen Entität alle Kreisgleichungen k_i von Kreisen um Basisstationen an den Orten $p_i = (x_i, y_i)$ mit den gemessenen und fehlerbehafteten Radii d_i erfüllt:

$$d_i = k_i(x, y) = \sqrt{(x_i - x)^2 + (y_i - y)^2} \; i = 1 \ldots k \tag{2.1}$$

Ein solches, nicht-lineares Problem der Ausgleichsrechnung lässt sich im Allgemeinen nur iterativ lösen. Dabei beginnt man mit einer beliebigen Schätzung des Ortes der mobilen Entität als Startwert. In jedem Schritt verwendet man eine Linearisierung des Gleichungssystems 2.1, die naturgemäß nur in einer Umgebung der aktuellen Schätzung gilt. Dazu entwickelt man das Gleichungssystem 2.1 durch Anwendung des Satzes von Taylor bis zum Grad 1 um den aktuell geschätzten Punkt. Der Satz von Taylor besagt, dass sich – unter gewissen Differenzierbarkeitsvoraussetzungen, die im vorliegenden Fall erfüllt sind

– jede Funktion f als

$$f(x) = \sum_{i=1\ldots n} \frac{f^{(i)}(x_0)}{i!}(x - x_0)^i + R_{n+1}(x, x_0)$$

mit einem Restglied R schreiben lässt, für das eine bestimmte Beschränkung feststeht. Mit anderen Worten nähert sich die Taylorsumme unter geeigneten Vorraussetzungen der tatsächlichen Funktion f in einer Umgebung von x_0 beliebig gut an. Wenn die Taylorsumme bis zum Grad 1 der Gleichungen 2.1 berechnet werden soll, so sind dabei die partiellen Ableitungen nötig:

$$\frac{\partial}{\partial x} k_i(x, y) = -\frac{x_i - x}{\sqrt{(x_i - x)^2 + (y_i - y)^2}}$$

$$\frac{\partial}{\partial y} k_i(x, y) = -\frac{y_i - y}{\sqrt{(x_i - x)^2 + (y_i - y)^2}}$$

Wird der aktuell geschätzte Ort mit den Koordinaten $(\tilde{x}, \tilde{y})$ bezeichnet, so ergibt sich mit den Messungen d_i das folgende Gleichungssystem:

$$d_i = k_i(\tilde{x}, \tilde{y}) + \frac{\partial}{\partial x} k_i(\tilde{x}, \tilde{y})(x - \tilde{x}) + \frac{\partial}{\partial y} k_i(\tilde{x}, \tilde{y})(y - \tilde{y}) \; i = 1 \ldots k \qquad (2.2)$$

Setzt man nun $\hat{x} = (x - \tilde{x})$ und $\hat{y} = (y - \tilde{y})$, erhält man ein überbestimmtes, lineares Gleichungssystem, welches mit der Methode der kleinsten Quadrate (siehe Abschnitt 2.1.1) einen Korrekturvektor für die Ortsschätzung ergibt. Mit diesem korrigiert man die aktuelle Ortsschätzung und iteriert das Verfahren. Für weitere Details, Beispiele und Betrachtungen sei auf die grundlegenden Arbeiten und Lehrbücher [Freu 07, Foy 76, Krak 75, Ludw 71, Hoff 03] verwiesen.

Beispiel 2.1.1:
Die folgende Tabelle zeigt vier Basisstationen mit den Koordinaten und fehlerbehafteten, gemessenen Entfernungen.

Komponente	**Koordinate**	**Messung** (d_j)
Basisstation 1	$(x_1, y_1) = (0/0)$	2.92
Basisstation 2	$(x_2, y_2) = (10/0)$	8.14
Basisstation 3	$(x_3, y_3) = (15/10)$	15.46
Basisstation 4	$(x_4, y_4) = (0/12)$	9.89
Mobile Entität	$(x_e, y_e) = (2/2)$	-
Positionsschätzung	$(\tilde{x}, \tilde{y}) = (20, 20)$	-

Als erstes berechnet man die Faktoren

$$\alpha_i = \sqrt{(x_i - \tilde{x})^2 + (y_i - \tilde{y})^2} \text{ mit } i = 1 \ldots 4$$

die in den partiellen Ableitungen im Gleichungssystem 2.2 vorkommen. Geometrisch lassen sich diese Faktoren als Abstand von der aktuellen Ortsschätzung zur jeweiligen Basisstation interpretieren. Eine optimale Schätzung ist in diesem Sinne also eine Schätzung, in der $\alpha_i \approx d_i$ ist, die Abstände der Ortsschätzung zu den Basisstationen also den gemessenen Abständen entspricht. Mit den Beispieldaten errechnet sich für den ersten Schritt:

$$\begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \end{pmatrix} = \begin{pmatrix} \sqrt{800} \\ \sqrt{500} \\ \sqrt{125} \\ \sqrt{464} \end{pmatrix} \approx \begin{pmatrix} 28.284 \\ 22.361 \\ 11.180 \\ 21.541 \end{pmatrix}$$

Dann stellt man für das Gleichungssystem 2.2 eine Matrix A aus Zeilen A_i zusammen, die sich wie folgt ergeben:

$$A_i = \left(-\frac{(x_i - \tilde{x})}{\alpha_i}, -\frac{(y_i - \tilde{y})}{\alpha_i} \right)$$

Für das vorliegende Beispiel erhält man dann

$$A = \begin{pmatrix} 0.70711 & 0.70711 \\ 0.44721 & 0.89443 \\ 0.44721 & 0.89443 \\ 0.92848 & 0.37139 \end{pmatrix}$$

Auf der rechten Seite des linearen Gleichungssystem findet sich ein Vektor b mit Zeilen

$$b_i = d_i - \alpha_i$$

Im vorliegenden Fall konkret

$$b = \begin{pmatrix} -25.364 \\ -14.221 \\ 4.28 \\ -11.65 \end{pmatrix}$$

Wenn man dieses Gleichungssystem mit linearer Ausgleichsrechnung löst, erhält man den Korrekturvektor $(\hat{x}, \hat{y})$ im Gleichungssystem

$$A \begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = b$$

mit dem konkreten Wert

$$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \begin{pmatrix} -18.62235 \\ -0.23377 \end{pmatrix}$$

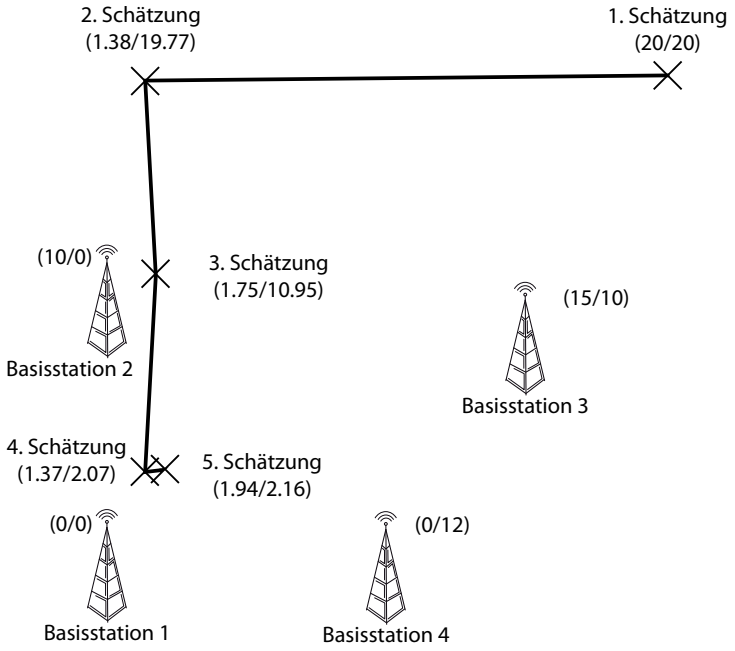


Abbildung 2.2: Die Situation des Beisipels und der iterative Prozess der Lateration

und die neue Ortsschätzung ergibt sich zu

$$\begin{pmatrix} \tilde{x}_2 \\ \tilde{y}_2 \end{pmatrix} = \begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} + \begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \begin{pmatrix} 1.3777 \\ 19.7662 \end{pmatrix}$$

Iteriert man dieses Verfahren, erhält man numerische Zwischenergebnisse wie in folgender Tabelle:

Schritt	**Ortsschätzung**	**Korrektur**	**Residuumsnorm**
1	$(20.00, 20.00)$	$(-18.62, -0.23)$	25.456
2	$(1.38, 19.77)$	$(0.38, -8.82)$	17.777
3	$(1.75, 10.95)$	$(-0.38, -8.88)$	8.954
4	$(1.37, 2.07)$	$(0.57, 0.09)$	0.631
5	$(1.94, 2.16)$	$(-0.01, -0.01)$	0.168
6	$(1.94, 2.15)$	$(0.00, 0.00)$	0.163
7	$(1.94, 2.15)$	$(-0.00, -0.00)$	0.163

Wie man erkennen kann, nähert sich die Ortsschätzung dem tatsächlichen Ort $(2, 2)$ an, erreicht ihn aber nicht, da die Distanzmessungen mit einem Fehler gestört worden sind. Im Schritt 7 ändert sich dann auch das Residuum nicht mehr im dargestellten Zahlenbereich. Daher wird an dieser Stelle die Berechnung abgebrochen und man erhält als Positionsberechnung $(1.94, 2.15)$. In Abbildung 2.2 ist eine grobe Darstellung der Situation dieses Beispiels zu finden. Im Anhang 7 ist auch eine sehr einfache, verständliche Implementierung zu finden, die sowohl in Octave als auch in MATLAB verwendet werden kann.

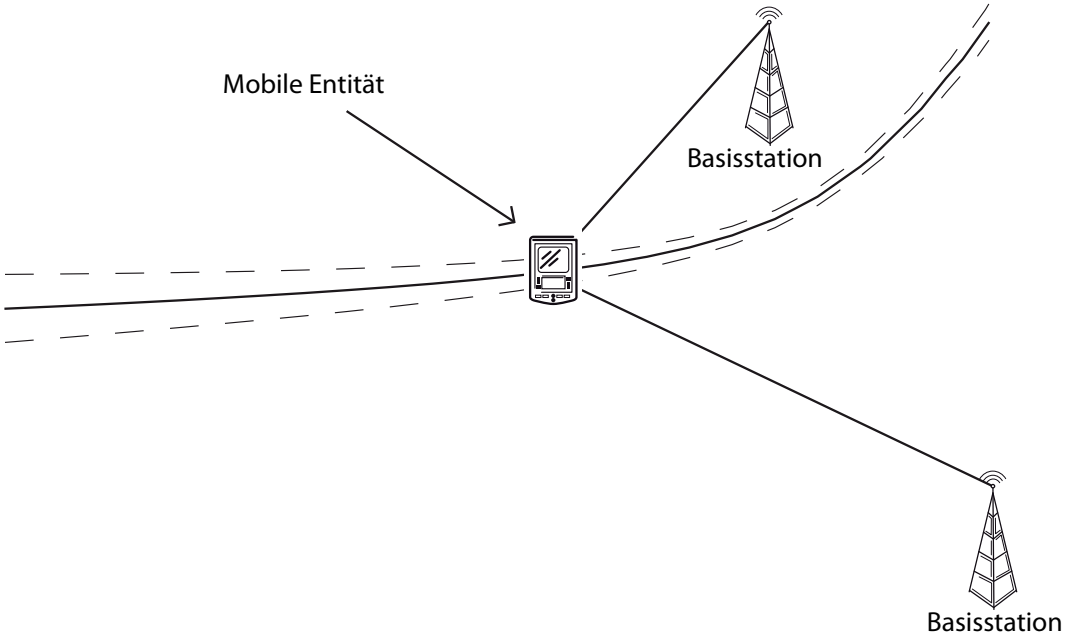


Abbildung 2.3: Beispiel für Hyperbellateration: Die Distanzdifferenz zwischen den eingezeichneten Linien ist bekannt. Die mobile Entität liegt dann auf der eingezeichneten Hyperbel. Durch Kombination mehrerer Hyperbeln lässt sich der Ort der mobilen Entität bestimmen

2.1.3 Hyperbolische Lateration

Hyperbolische Lateration ist eine Variante der Lateration, bei der nicht die Entfernungen zu festen, bekannten Orten gemessen werden, sondern der Entfernungsunterschied zu festen, bekannten Orten. Ist für zwei Basisstationen die Differenz Δd zwischen den Abständen der mobilen Entität zu den beiden Basisstationen bekannt, so liegt die mobile Entität auf einer Hyperbel wie in Abbildung 2.3 eingezeichnet. Eine solche Abstandsdifferenz kann in der Praxis beispielsweise durch die Messung der Zeitdifferenz zwischen dem Eintreffen zweier Signale an der mobilen Entität berechnet werden, die zur selben Zeit von den Basisstationen ausgesendet wurden (Time-Difference-of-Arrival, siehe Abschnitt 2.4).
Zur Berechnung der Position wird, genau wie bei der zirkulären Lateration (Abschnitt 2.1.2), die Methode der linearen Ausgleichsrechnung angewandt. Dabei werden zunächst die nicht-linearen und mit Messfehlern gestörten Aufenthaltsgleichungen

$$\begin{aligned}\Delta d_{ij} &= k_i(x,y) - k_j(x,y) \\ &= \sqrt{(x_i - x)^2 + (y_i - y)^2} - \sqrt{(x_j - x)^2 + (y_j - y)^2}\ i,j = 1 \ldots k, i < j \end{aligned} \quad (2.3)$$

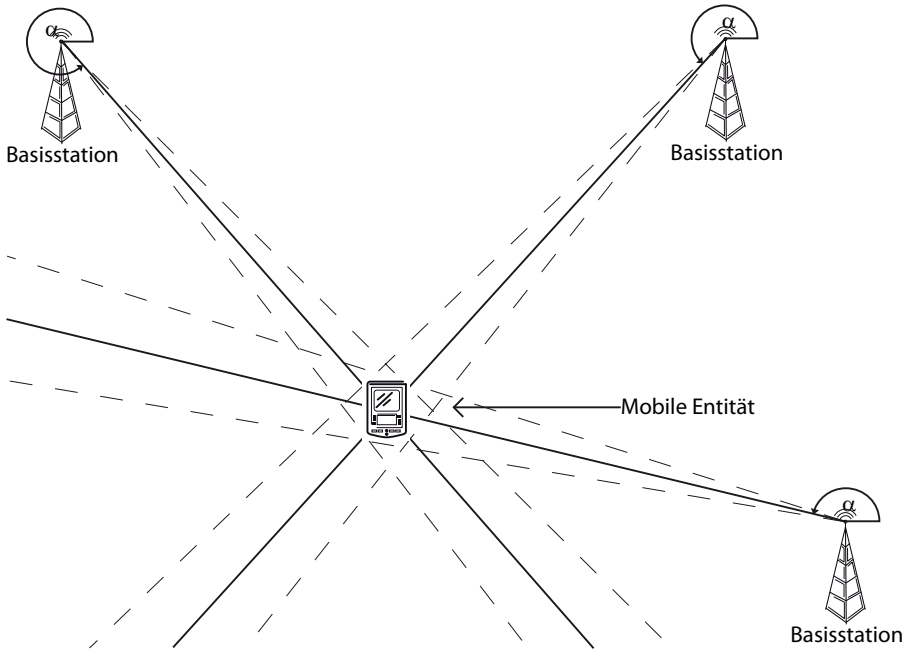


Abbildung 2.4: Beispiel für Angulation: Drei mit Fehlern behaftete Winkelmessungen zu Basisstationen mit bekanntem Ort ergeben einen ungefähren Aufenthaltsort der mobilen Entität

aufgestellt. Es ist dabei üblich, die Entfernungen zu einer festen Station, etwa $i = 1$, zu messen und nur dieses kleinere Gleichungssystem zu lösen.
Das auf diese Weise aufgestellte, nicht-lineare, überbestimmte, durch Messfehler gestörte Gleichungssystem wird dann analog zur zirkulären Lateration gelöst, indem man von einer Positionsschätzung ausgeht, dann sukzessive mit einer Taylorentwicklung um die letzte berechnete Position ein lineares Ausgleichsproblem für einen Korrekturvektor konstruiert, mit dem Verfahren der linearen Ausgleichsrechnung löst und das Ergebnis – letzte Ortsschätzung plus Korrekturvektor – als neue Ortsschätzung verwendet.

2.1.4 Angulation

Die Verfahren der *Angulation* verwenden gemessene Winkel zwischen einer mobilen Entität und festen Punkten, um eine Ortsschätzung zu errechnen. Die gemessenen Winkel unterliegen auch wieder Messfehlern, für die im Allgemeinen angenommen werden kann, dass sie einer Normalverteilung folgen. Es gibt hier zwei grundlegend verschiedene Perspektiven, aus denen Winkel verwendet werden. Entweder der Winkel zwischen mehreren festen Punkten und der mobilen Entität ist bekannt und wird an den festen Punkten gemessen oder der Winkel, in dem eine Gerade zwischen dem festen Punkt und der mobilen Entität an der mobilen Entität auftrifft, ist bekannt. Abbildung 2.4 zeigt ersteren Fall. Die Verfahrensweisen für eine Ortsbestimmung mit diesen Winkelinfor-

mationen sind vergleichbar. Denn im Grunde beschränkt eine Winkelmessung an einer Entität den möglichen Aufenthaltsbereich der anderen Entität auf einen Strahl.

An dieser Stelle soll der Fall betrachtet werden, in dem die Winkel an Basisstationen mit festen, bekannten Koordinaten gemessen werden. Dies ist auch der Fall, der in der Praxis häufiger vorkommt, weil sich beispielsweise bei Radiosignalen die Winkelinformation am Besten durch Antennenarrays ermitteln lässt, die in mobilen Endgeräten derzeit aus energetischen und wirtschaftlichen Gründen selten verbaut werden.

In diesem Fall liegen Winkelmessungen zwischen der mobilen Entität und n Basisstationen mit bekannten Koordinaten vor. Bis auf die zu erwartenden Messfehler und spezielle Effekte wie *Mehrwegeausbreitung* sollte die mobile Entität also auf den Geraden liegen, die durch die gemessenen Winkel definiert sind. Dies ist genau dann der Fall, wenn der Koordinatenvektor $p = (x, y)$ der mobilen Entität die Gleichungen

$$\alpha_i = \arctan \frac{y_i - y}{x_i - x} \; i = 1 \ldots k \tag{2.4}$$

erfüllt. Hierbei sollte eine Umgebung des Grenzfalls $x_i - x = 0$, der natürlich einem Winkel von $\alpha = 90°$ entspricht, ausgeschlossen werden. Die meisten Programmiersprachen bieten aber ohnehin eine numerisch sinnvollere Funktion `atan2` an, die einen Wert für Zähler und Nenner und nicht den verrechneten Quotienten übernimmt.

Im Zweidimensionalen reichen zwei Punkte, die nicht kolinear mit der tatsächlichen Position der mobilen Entität sind, aus, um eine Position zu bestimmen. Je mehr sich die geometrische Situation diesem Extremfall nähert, desto schlechter kann der Ort berechnet werden. In der Praxis verwendet man daher so viele Standorte, dass möglichst für die gesamte Positionierungsfläche Messungen existieren, die nicht in solch geometrischer Extremlage zueinander liegen, um eine gute Positionierung zu gewährleisten. In diesem Falle kommt wieder das Verfahren der Linearisierung der Gleichungen mit der Taylorentwicklung ersten Grades und der iterativen Verbesserung einer initialen Positionsschätzung zum Einsatz. So erhält man ein Verfahren, welches für das Gleichungssystem 2.4 eine Näherungslösung berechnet, die die wahrscheinlichste Position der mobilen Entität wiedergibt.

Im Falle, dass die mobile Entität selbst die Messung der Winkel zu Basisstationen mit bekannten Koordinaten vornimmt, kann genauso vorgegangen werden. Lediglich eine weitere Unbekannte, nämlich der Winkel γ, mit dem die mobile Entität zum Koordinatensystem der Fixpunkte rotiert ist, muss berücksichtigt werden. Man erhält dann das erweiterte Gleichungssystem

$$\alpha_i = \left(\arctan \frac{y_i - y}{x_i - x} \right) - \gamma \; i = 1 \ldots k \tag{2.5}$$

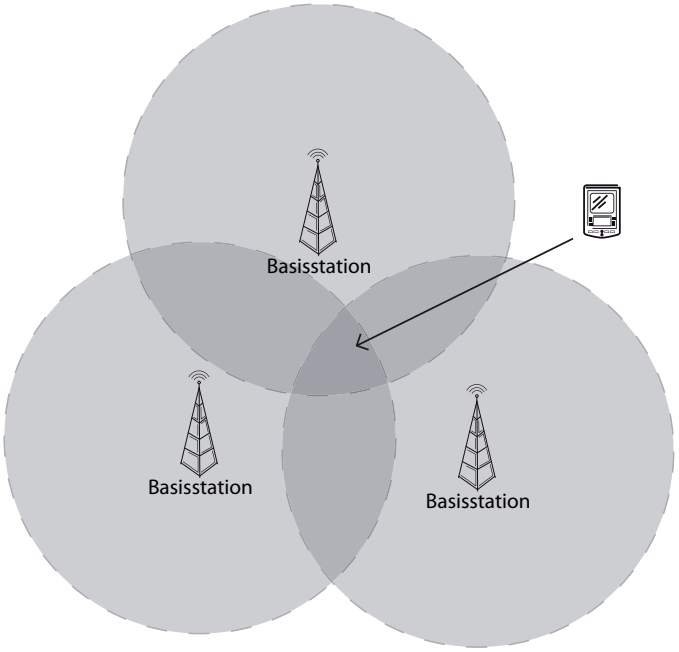


Abbildung 2.5: Beispiel für Proximity-Detection: Durch den Aufenthalt im Funkabdeckungsbereich dreier Sender wird die Fläche des möglichen Aufenthalts durch den Schnitt von Mengen eingeschränkt

und verfährt analog zum ersten Fall.

2.1.5 Proximity Detection

Bei den Verfahren der *Proximity Detection* wird die Nähe zu bekannten Objekten gemessen und daraus auf den Aufenthalt in einer Umgebung derselben geschlossen. So folgt beispielsweise aus der Sichtbarkeit eines WLAN-Netzwerkes, dass der Nutzer sich im Abdeckungsgebiet dieses Netzes befinden muss. Entsprechend ergibt die Proximity Detection nicht eine Position, sondern eine Menge möglicher Positionen. Die Nähe zu mehreren Objekten führt dann zu einem Schnitt der zugehörigen Mengen und somit zu einer kleineren Fläche, in der sich das zu positionierende Objekt befinden kann. Wird dabei die Nähe über die maximale Reichweite r eines Signals definiert, so ergeben sich durch Proximity Detection Kreisflächen um die bekannten Orte p_i der Sender, in denen ein Aufenthalt möglich ist:

$$p \in \bigcap_i \{x \text{ mit } ||x - p_i|| < r\}$$

Ein Beispiel für den Schnitt mehrerer Kreise zeigt Abbildung 2.5. In Extremfällen der Proximity Detection wird die Nähe zu sehr vielen Objekten mit einer sehr kleinen Fläche des möglichen Aufenthalts verwendet. Dabei kann

ein solches System durchaus Genauigkeiten im Meter- und Zentimeterbereich erreichen, benötigt aber auch Infrastruktur-Komponenten in dieser Dichte.

2.1.6 Inertiale Navigation

In der *inertialen Navigation* verwendet man Messwerte, die autonom bei der mobilen Entität messbar sind, um, ausgehend von einem (bekannten) Startpunkt, den Weg der mobilen Entität fortzuschreiben. In der Regel handelt es sich dabei um Beschleunigungen und Winkelbeschleunigungen, lediglich bei Fahrzeugen kann auch mit Odometrie und Lenkwinkel eine Strecke oder Geschwindigkeit direkt gemessen werden. Der große Vorteil der inertialen Navigation liegt in der Unabhängigkeit von einer Infrastruktur. Ein mobiles System kann sich autonom positionieren. Der größte Nachteil ist aber, dass sich Messfehler in den Sensordaten über die Zeit aufsummieren.
Ein typisches, inertiales Navigationssystem verwendet eine *Inertial Measurement Unit (IMU)*, die sechs Messwerte liefert. Die Beschleunigung wird typischerweise durch drei paarweise orthogonal angeordnete Beschleunigungssensoren ermittelt und kann auch gleichzeitig zur Lagebestimmung verwendet werden, da dieses Sensorsystem auch die Erdanziehungskraft aufzeichnet. Darüber hinaus werden oft drei Gyro-Sensoren auf den drei Hauptachsen angeordnet, die die Rotation um die jeweiligen Achsen messen. Zusammengenommen lässt sich damit die Lage und die Beschleunigung fortschreiben.
Das Verfahren der inertialen Navigation verwendet nun den mit einer IMU gemessenen Beschleunigungsvektor $\ddot{p}$, der auf die mobile Entität wirkt, und berechnet mit dem Integral

$$\dot{p}(t) = \int_{t_0}^{t} \ddot{p}(\tau) d\tau + \dot{p}_0$$

den Geschwindigkeitsvektor $\dot{p}(t)$ zum Zeitpunkt t. Verfährt man analog mit der Geschwindigkeitsfunktion $\dot{p}(t)$, erhält man die Ortsfunktion $p(t)$ der mobilen Entität:

$$p(t) = \int_{t_0}^{t} \dot{p}(\tau) d\tau + p_0$$

Um dieses Verfahren zu verwenden, benötigt man zunächst die folgenden zwei Informationen:

- p_0: den Ortsvektor, an dem sich die mobile Entität zum Zeitpunkt $t = t_0$ befindet.
- $\dot{p}_0$: den Geschwindigkeitsvektor, mit dem sich die mobile Entität zum Zeitpunkt $t = t_0$ bewegt.

In der Praxis ist es oft schwierig, ein inertiales Navigationssystem zu verwenden, da das Rechenverfahren, im Gegensatz zu globalen Positionierungssystemen, Fehler aggregiert und über die Integration massiv verstärkt. Kommt es

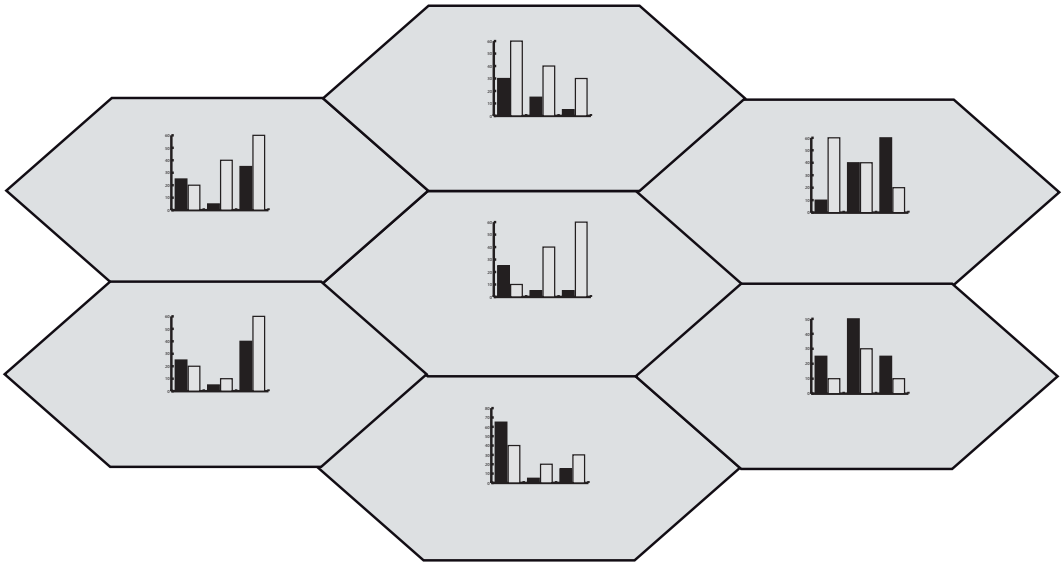


Abbildung 2.6: Beispiel für Mustererkennung: In verschiedenen Zonen werden ortsabhängige Messwerte erhoben, die sich in der Summe genügend unterscheiden, um die Zone des aktuellen Aufenthalts zu ermitteln

beispielsweise bei der Beschleunigungsbestimmung zu einem Fehler, wird das System auch dann nicht zur Ruhe kommen, wenn die mobile Entität ruht. Damit hat eine ruhende mobile Entität in der Praxis häufig noch eine Geschwindigkeit und mit der Zeit, abhängig von der Größe der falschen Geschwindigkeit, entfernt sich die Positionsberechnung von der Realität. Insbesondere ist es schwierig, die Erdanziehungskraft korrekt aus Beschleunigungsmessungen herauszurechnen.
Deshalb wird die inertiale Navigation oft mit einer anderen Positionsquelle zur Rekalibrierung, also zur Rücksetzung des Bewegungszustandes oder wenigstens des, Ortes kombiniert. Besonders die inertiale Navigation eignet sich zur Überbrückung von kürzeren Strecken, in denen wegen Mehrwegeausbreitung oder Empfangsschwierigkeiten andere Positionierungsverfahren fehlschlagen oder eine sehr hohe Aktualisierungsrate des Ortes erwünscht ist.

2.1.7 Mustererkennung, Szenenanalyse und Fingerprinting

Die bisher beschriebenen Verfahren verwenden einen bekannten, physikalischen Zusammenhang zwischen dem Aufenthaltsort einer mobilen Entität und einem Messwert, um direkt einen Ort auszurechnen. Hierbei kommen zwar Verfahren wie die Ausgleichsrechnung zur Auflösung von Messfehlern vor, dennoch wird angenommen, dass der Zusammenhang zwischen Messgröße und Ort *deterministisch* ist. Durch komplexe Funkausbreitung in Gebäuden oder durch

heuristische Verfahren zum Bildvergleich ist aber häufig keine deterministische Verbindung zwischen Messwerten und Orten bekannt In diesen Fällen verwendet man Verfahren der Mustererkennung und Szenenanalyse wie Fingerprinting, die ganz allgemein aus einem Trainigs-Datensatz mit Messwerten und Orten (sowohl als Teilflächen als auch als Koordinaten) ein Modell errechnen, das sich möglichst gut auf neue Beobachtungen zur Berechnung des Ortes anwenden lässt. Abbildung 2.6 illustriert diese Vorgehensweise an einer Zellenaufteilung der Fläche. Diese Verfahren bezeichnet man im Gegensatz zu den deterministischen Verfahren als *probabilistisch*, auch wenn sie nicht immer auf der Betrachtung von Wahrscheinlichkeiten beruhen.
Dabei können die klassischen Verfahren des maschinellen Lernens Verwendung finden. Diese Verfahren lassen sich am Besten an Hand der Art und Weise erklären, wie das aus den Daten extrahierte „Wissen“ dargestellt wird. Zunächst werden die Daten, aus denen eine empirische Erkenntnis abgeleitet werden soll, in einer Vorbereitungsphase in eine feste Form gebracht. Diese Form ist gegeben durch die Spezifikation einer Menge von *Attributen*. Solche Attribute können binär sein, z.B. ob Beacons eines bestimmten Access-Point gerade empfangen werden können, oder numerisch, wie die Signalstärke, mit der solche Beacons empfangen werden. Weiter gibt es die Möglichkeit, nominale Attribute zu definieren; solche Attribute könnten zum Beispiel Raumnamen erfassen.
Das grundlegende Problem besteht nun darin, das Attribut von Interesse aus den anderen Attributen zu bestimmen. Bei Anwendungen der Positionierung ist das Attribut von Interesse ein Attribut, das den Aufenthaltsort direkt oder indirekt beschreibt. In der *Klassifikation* wird der Wert eines nominalen Attributs aus den Werten der anderen Attribute abgeleitet. In der *numerischen Prediktion* wird ein numerischer Wert aus den anderen Attributen abgeleitet. Ein solches Problem der numerischen Prediktion kann durch Diskretisierung in ein Klassifikationsproblem überführt werden. Eine Wertbelegung der Attribute nennt man Instanz. In Instanzen werden die Attributwerte gleicher Zustände erfasst. Also etwa Messgrößen, die sich auf dasselbe Objekt beziehen, oder Messungen, die in einem gewissen Zeitpunkt an einem festen Ort gemacht wurden. Solche Instanzen müssen nicht unbedingt vollständig sein, es ist in vielen Fällen durchaus möglich, auch mit unvollständigen Instanzen zu arbeiten. Man verwendet entsprechend eine Datenbank von Instanzen, in denen möglichst viele Werte bekannt sind, als Trainingsdatenbank, um Zusammenhänge zwischen den Attributen zu erkennen und in einem Vorhersagemodell zu erfassen. Zur Klassifikation werden dann, nachdem ein solches Modell erstellt worden ist, Instanzen eingegeben, bei denen das Attribut von Interesse nicht bekannt ist.
Die Algorithmen, mit denen ein solches Modell aus den Trainingsdaten erstellt werden kann, organisieren sich in drei Gruppen, abhängig davon, welche Informationen zum Zeitpunkt der Modellerstellung vorhanden sind:

- *Supervised-Learning*: Zu jeder Instanz ist das Attribut, das vorhergesagt werden soll, bekannt.

- *Unsupervised-Learning*: Es wird nicht direkt nach einem Vorhersagemodell für ein Attribut gesucht, sondern es wird untersucht, welche Zusammenhänge zwischen den Attributen bestehen.

- *Semi-Supervised-Learning*: Die beiden genannten Verfahrenstypen werden kombiniert, um die Erstellung eines Vorhersage-Modells zu verbessern.

Generell muss man bei der Erstellung von Modellen darauf achten, dass die Modelle die Trainingsdaten nicht unrealistisch genau beschreiben, sogenanntes *„Overfitting“*. Denn dann erhält man zwar vielversprechende Erkennungsraten auf den Trainingsdaten, die Vorhersage generalisiert aber leider nicht auf unbekannte Instanzen, und eine sinnvolle Vorhersage ist nicht mehr möglich.
Eine einfache, verbreitete Methode, das notwendige „Wissen“ für eine Klassifikation zu organisieren, sind *Entscheidungsbäume*, bei denen Tests wie Vergleiche „Attribut $\leq$ Konstante“ in den inneren Knoten einer Baumstruktur angeordnet sind. Dabei wird für eine Instanz, deren Klasse bestimmt werden soll, von der Baumwurzel ausgehend der Test im jeweils aktiven Knoten des Baumes durchgeführt und abhängig vom Ergebnis mit einem der Teilbäume unterhalb des aktuellen Knotens fortgefahren. In den Blättern eines solchen Baumes stehen dann die Klassifikationsergebnisse.
Eine weitere Möglichkeit, aus Beobachtungen eine Form von Wissen abzuleiten und zu speichern, basiert auf dem Satz von Bayes

$$P(I|M) = \frac{P(M|I)P(I)}{P(M)}, \tag{2.6}$$

der besagt, dass die bedingte Wahrscheinlichkeit $P(I|M)$, dass ein gewisses Ereignis I vorliegt, wenn eine Messung M gemacht wird, sich aus anderen bedingten Wahrscheinlichkeiten berechnen lässt. Die Wahrscheinlichkeit $P(M|I)$, mit der eine Messung im Falle des Vorliegens des Ereignisses I gemacht werden kann, und die Wahrscheinlichkeit $P(M)$, eine solche Messung überhaupt zu machen, kann aus relativen Häufigkeiten in einem Trainingsdatensatz abgeleitet werden. Beschreibe zum Beispiel I den Aufenthaltsort und sei M das Ereignis einer aktuellen Messung. Dann ist $P(I|M)$ die Wahrscheinlichkeit, sich am Ort I aufzuhalten. Der Wert $P(M|I)$ auf der rechten Seite ergibt sich dann aus einer Trainingsdatenbank, in der Messungen und der Ort der Messung bekannt sind. Der Wert $P(I)$ wird aus der relativen Häufigkeit bestimmt, mit der eine Messung am Ort I in den Trainingsdaten enthalten ist. So lässt sich der Zähler obiger Gleichung für jeden Ort I_k berechnen, und durch Normierung erhält man eine Wahrscheinlichkeitsverteilung $P(I_k|M)$, die den Aufenthaltswahrscheinlichkeiten an den verschiedenen Orten I_k entspricht. Der Nenner der Gleichung 2.6 ist dabei irrelevant, weil nur eine feste Messung M betrachtet wird, und somit der Nenner in alle Werte $P(I_k|M)$ als Konstante einfließt, deren Effekt durch den Normierungsschritt korrekt neutralisiert wird. Im konkreten Fall des „Naive Bayes“-Algorithmus wird die Bestimmung

von $P(M|I)$ aus Attributen $a_1, \ldots, a_n$ unter Hinzunahme der Annahme, dass diese Attribute statistisch unabhängig sind, durch das Produkt

$$P(M|I) = \prod_{i=1\ldots n} P(a_i, I)$$

berechnet. In diesem Sinne arbeitet ein solches System naiv, weil genau diese statistische Unabhängigkeit häufig nicht vorliegt. Dennoch gibt es viele Beispiele, in denen ein solcher Klassifikator erstaunlich gute Ergebnisse liefert, auch wenn die Attribute nicht statistisch unabhängig sind.

Es gibt neben den beiden soeben vorgestellten Arten, Wissen aus Beobachtungen abzuleiten, eine große Anzahl anderer Ansätze. Eine geschlossene Darstellung solcher Verfahren würde den Rahmen dieser Arbeit sprengen. Gute Darstellungen finden sich in den gängigen Lehrbüchern [Russ 10, Witt 11, Nisb 09].

In der Positionsbestimmung im Inneren von Gebäuden treten die Verfahren des maschinellen Lernens häufig unter der Bezeichnung *Fingerprinting* auf. Die wegweisende und grundlegende Arbeit RADAR [Bahl 00], die man als erstes, funktionierendes Indoor-Positionierungssystem auf Basis der Signalstärken von WLAN-Beacons auffassen kann, verwendet einen gewichteten k-Nächste-Nachbarn Ansatz (kNN), um aus Trainingsdaten und Signalstärken die Koordinaten des mobilen Endgerätes zu ermitteln. Wegen der teils großen Ungenauigkeiten der Messwerte und der Signalausbreitung, kommen solche Verfahren alleine aber nicht über Raumgenauigkeit heraus. Diese Tatsache motiviert die Gegenposition, keine numerische Prediktion von Koordinaten zu versuchen, sondern eine Vorhersage einer groben, nominalen Position in Form beispielsweise eines Raumnamens. Details solcher Verfahren werden in Abschnitt 3.2 vorgestellt.

Die Verwendung von maschinellen Lernverfahren und moderner Statistik ist nicht nur für die Positionsbestimmung essentiell. Gerade für kontextabhängige und mobile Anwendungen aller Art sind solche Verfahren zwingend notwendig, wenn automatisiert aus einer Fülle von Sensorinformationen für den Nutzer oder die Applikation sinnvoll nutzbare Informationen abgeleitet werden sollen. So wurde zum Beispiel im Jahr 2011 ein System für die Qualitäts- und Prozesskontrolle geschaffen, welches lediglich den Aufenthaltsort von Arbeitern verwendet, um zu überprüfen, ob eine bestimmte Arbeitsabfolge auch tatsächlich eingehalten wurde [Zinn 09]. Auch der emotionale Zustand eines Fahrers wurde mit solchen Verfahren aus gemessenen Vitaldaten teilweise abgeleitet [Kumm 11] und auch in Bezug auf Automatisierung in mobilen Geschäftsanwendungen kann ein Kontextbezug nur durch fortgeschrittene Verfahren des maschinellen Lernens erreicht werden [Wern 12].

2.1.7.1 Simultaneous Localization and Mapping

Die Grundidee der Ansätze des *Simultaneous-Localization-And-Mapping (SLAM)* besteht darin, aus einer zeitlichen Sequenz von Beobachtungen gleichzeitig eine Karte der Umgebung und darin die aktuelle Position und die Trajektorie des zurückgelegten Weges zu ermitteln. Solche Systeme sind insbesondere in der Robotik von essentieller Bedeutung, denn sie dienen als Grundlage für die Wahrnehmung und Orientierung in unbekannten Umgebungen. Eine gute Übersicht über SLAM aus Sicht der Robotik bietet die Übersichtsarbeit [Boni 08].

Am häufigsten kommen visuelle SLAM-Ansätze zum Einsatz, bei denen die Beobachtung in Form von Bildern und Tiefeninformationen erfolgt. Solche Tiefeninformationen können aus Stereo-Bildern extrahiert oder mit Laser-Scannern bestimmt werden. Dabei wird durch Vergleiche von Merkmalspunkten oder optischen Flüssen zwischen aufeinanderfolgenden Bildern diejenige Transformation gesucht, die das Sichtfeld des ersten Bildes in das Sichtfeld des zweiten Bildes überführen.

Grundsätzlich entstehen bei SLAM-Ansätzen Graphen, deren Knoten den Zustand des Systems und deren Kanten die Übergänge zwischen Zuständen beschreiben. Typischerweise modellieren die Knoten den Aufenthaltsort und Blickrichtung und die Kanten die Abbildungen, die sowohl den Aufenthaltsort als auch die Blickrichtung ineinander überführen. So gesehen ist der SLAM-Ansatz mit der inertialen Navigation verwandt, weil auch hier, ausgehend von einer Startposition, nur die fehlerbehafteten Beobachtungen der Bewegung verwendet werden, um diese Position fortzuschreiben. Entsprechend ist auch beim SLAM-Ansatz eine Aufsummierung von Fehlern zu erwarten. Im Unterschied zu einfachen, inertialen Positionierungssystemen besteht bei SLAM aber die Möglichkeit, dass die mobile Entität denselben Ort mehrfach wahrnimmt. In solchen Situationen kann die vorhergehende Trajektorie zwischen dem ersten und zweiten Sichten desselben Ortes korrigiert werden.

Wie in der Computergrafik üblich, verwendet man dabei eine Einbettung der dreidimensionalen Welt in einen vierdimensionalen Vektorraum $(x, y, z) \in \mathbb{R}^3 \rightarrow (x, y, z, 1) \in \mathbb{R}^4$, weil sich eine Verschiebung in diesem Falle auch als lineare Abbildung schreiben lässt. Man beschränkt sich dann häufig auf diejenigen Transformationen, die aus einer Drehung und Verschiebung bestehen, weil sich weder die Skalierung der Welt noch die Kameraprojektion geändert hat.

Insgesamt führen Übereinstimmungen zwischen den Bildern mit Tiefeninformationen zu Punktwolken in jenem vierdimensionalen Raum, für die eine optimale, längentreue Transformation gefunden werden soll, die die Punktwolken am Besten in Deckung bringt. Dafür kann bei Tiefenbildern eine globale Metrik für die Abweichung verwendet werden oder bei einem punktweisen Vergleich für Punktwoken $A = \{a_i\}$ und $B = \{b_i\}$, $a_i, b_i \in \mathbb{R}^4$, eine Minimierung der

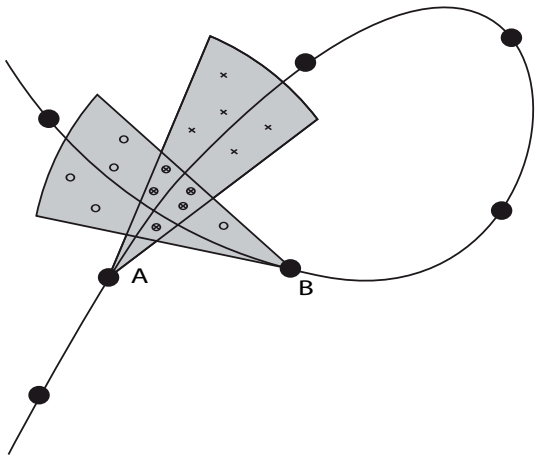


Abbildung 2.7: Schleifen-Erkennung bei SLAM zur Korrektur akkumulierter Fehler

Fehlersumme

$$T = \min_{\tilde{T}} \sum_{i} ||\tilde{T}a_i - b_i||$$

erfolgen. Diesen Vorgang, zwei Punktwolken in Deckung zu bringen, nennt man auch „Registrierung“.
Für die Registrierung von Punktwolken, die für sich allein ein spannendes Problem ist, werden unter anderem Verfahren wie *Random Sample Consensus (RANSAC)* [Fisc 81] oder eine der vielen Varianten des *Iterative-Closest-Point-Algorithmus (ICP-Algorithmus)* [Besl 92] verwendet.
Für die entscheidende Verbesserung von SLAM gegenüber inertialen Navigationssystemen zeigt Abbildung 2.7 eine Trajektorie mit Sichtfeldern. Das Sichtfeld von Punkt A und Punkt B zeigt dabei eine Überschneidung, die dazu verwendet werden kann, durch Registrierung der gemeinsamen Punkte der Punktwolken im Sichtfeld von A und B den Fehler in der Position auszugleichen, der sich auf der Strecke zwischen A und B angesammelt hat. Dazu steht beispielsweise der Algorithmus *„Tree-based NetwORk Optimizer“ (TORO)* zur Verfügung [Gris 09].

2.2 Algorithmen zur Kombination und Verifikation verschiedener Verfahren

Die verschiedenen Verfahren, mit denen in Gebäuden der Ort eines mobilen Endgeräts bestimmt werden kann, haben alle verschiedene Eigenschaften in Bezug auf Qualität und Fehlerverhalten. Daher gibt es häufig die Situation, dass die Kombination mehrerer Verfahren die Nachteile der jeweils anderen Verfahren mindert. Eine ganz einfache Möglichkeit besteht etwa darin, dass man ein langsames, präzises, globales Positionierungssystem wie GPS mit einem schnel-

len, lokalen System wie der Odometrie bei einem Fahrzeug kombiniert, indem stets von der letzten GPS-Position ausgehend mit den odometrischen Daten der Ort fortgeschrieben wird. So behält man einerseits die Genauigkeit der GPS-Positionierung, erhält aber andererseits in sehr hoher Frequenz und mit geringen Fehlern behaftete Positionen zwischen zwei GPS-Positionsberechnungen. Der Vorteil entsteht dadurch, dass die Fehler einer inertialen Navigation in den kleinen Zeitabschnitten zwischen zwei GPS-Positionierungen nicht so sehr ins Gewicht fallen, und daher das Gesamtergebnis näher an der Realität liegt, als das Ergebnis beider Positionierungsverfahren für sich.
Für die Kombination von Messwerten und Positionen gibt es auch viele mathematisch fundierte Verfahren. Im Folgenden sollen die für die Positionierung erfolgreichsten beiden Algorithmen erläutert werden, die eine Kombination nahezu aller verfügbaren Positionierungssysteme ermöglichen.

2.2.1 Dynamische lineare Systeme und Kalman-Filter

Eine klassische Herangehensweise besteht in der Verwendung eines Kalman-Filters. Ein Kalman-Filter [Kalm 60] ist ein Schätzer für den Zustand eines linearen, stochastischen Systems aus normalverteilt gestörten Beobachtungen. Dabei wird beim Kalman-Filter die folgende Systemgleichung verwendet:

$$x_{k+1} = A_k x_k + w_k$$

wobei A_k eine lineare Abbildung ist, die den zu erwartenden Zustandsübergang in einem diskreten Zeitschritt beschreibt, und w_k eine normalverteilte Zufallsvariable mit Mittelwert 0 und einer Kovarianzmatrix Q_{k-1} ist, die den Messfehler modellieren soll.
In dieser Situation lässt sich durch einen rekursiven Algorithmus, der aus den drei Schritten Gewichtung, Korrektur durch Messung und Vorhersage besteht, der Zustand des Systems optimal schätzen. Dabei wird – vereinfacht gesprochen – in den Zeiten, in denen keine neue Messung hinzugefügt wird, mit dem Vorhersagemodell, welches im Wesentlichen auf der Abbildung A_k basiert, der Zustand des Systems fortgeschrieben. In einem Navigationsbeispiel würde hier die letzte bekannte Geschwindigkeit und Richtung verwendet, um einen aktuelle Ortsvorhersage zu machen. Wird nun dem System eine neue Messung hinzugefügt, so werden mit einer geschickten Gewichtung der vorhergesagte Ort und der gemessene Ort kombiniert. Für die Details des Verfahrens sei auf [Hoff 03, Kapitel 3.6] verwiesen.
Der Kalman-Filter funktioniert zunächst nur für nicht-lineare Systeme mit normalverteilten Variablen. Man kann allerdings in einigen Fällen auch mit einer Taylor-Linearisierung wie bei der Lateration und Angulation eine Ausweitung auf einfache, nicht-lineare Fälle versuchen.
Durch die allgemein gestiegene Rechenleistung haben sich für viele Anwendungen etwas allgemeinere Verfahren durchgesetzt, für die weit weniger Voraussetzungen erfüllt sein müssen, und die im folgenden Abschnitte erläutert

werden. Dennoch ist der Kalman-Filter durch seine einfache Struktur und den relativ geringen Berechnungsaufwand ein beliebter Filter im Bereich mobiler Systeme.

2.2.2 Diskrete Wahrscheinlichkeitsverteilungen und Partikel-Filter

Diese allgemeineren Verfahren basieren auf der Simulation des Verhaltens von Partikeln, die zusammen eine Wahrscheinlichkeitsverteilung des Systemzustands beschreiben. Dabei können sowohl nicht-lineare Systeme als auch Systeme, die nur unvollständig beobachtbar sind, in natürlich parallelisierbarer Weise simuliert werden. Ein klassisches Verfahren ist das „Sequential Importance Resampling (SIR)“. Bei diesem Verfahren wird eine Menge von Partikeln ganz ähnlich zum Kalman-Filter durch Vorhersage-Modelle in der Zeit-Domäne und durch Messungen im Zustandsraum aktualisiert. Dabei kann ein Partikel beim Eintreffen einer Messung durchaus in mehrere Partikel unterschiedlicher Wahrscheinlichkeit – entsprechend einer beliebigen Verteilung – unterteilt werden. Um die unbegrenzte Entstehung von Partikeln zu vermeiden, werden Partikel mit geringer Gewichtung und geringer Wahrscheinlichkeit entfernt.
Mit genügend Partikeln ist dieses Verfahren optimal, jedoch auch sehr rechenaufwändig. Bei zu wenig Partikeln kann natürlich der Diskretisierungsfehler so groß werden, dass falsche Aussagen getroffen werden. Da die Simulation der Partikel unabhängig voneinander ist, kann ein Partikel-Filter leicht parallelisiert werden, was den Nachteil beim Rechenaufwand teilweise ausgleicht. Ein klassisches Verfahren, die Partikel zu aktualisieren, ist der „Bootstrap Filter“. Dieser wird in [Gord 93] vorgestellt und an sehr einfachen Beispielen erläutert. Für weitergehende Literatur sei auf [Arul 02, Rist 04] verwiesen.

2.3 Eigenschaften von Positionierungssystemen

Bevor verschiedene Positionierungssysteme beschrieben werden, soll im folgenden auf Aspekte hingewiesen werden, die für den Vergleich und die Auswahl von Positionierungssystemen relevant sind.
Ein solcher Aspekt ist natürlich zunächst die Genauigkeit, mit der eine Position bestimmt werden kann. Dabei unterscheidet man zwischen den Maßen „Genauigkeit“ (*„Accuracy“*) und „Präzision“ (*„Precision“*). Obwohl diese beiden Worte im Alltag eine sehr ähnliche Bedeutung haben, ist ihre Unterscheidung in der Positionierung essentiell: Mit Genauigkeit bezeichnet man die Abweichung einer Messung von der Realität, während Präzision die Abweichungen von Messungen untereinander beschreibt. Ein weiteres Maß ist in diesem Zusammenhang die Ortsauflösung, die typischerweise als die kleinste Änderung der Position angegeben wird, die vom System gemessen werden kann.
Wird nun an einem festen Ort eine Anzahl von Messungen vorgenommen, so lassen sich die beiden Größen Genauigkeit und Präzision auch numerisch

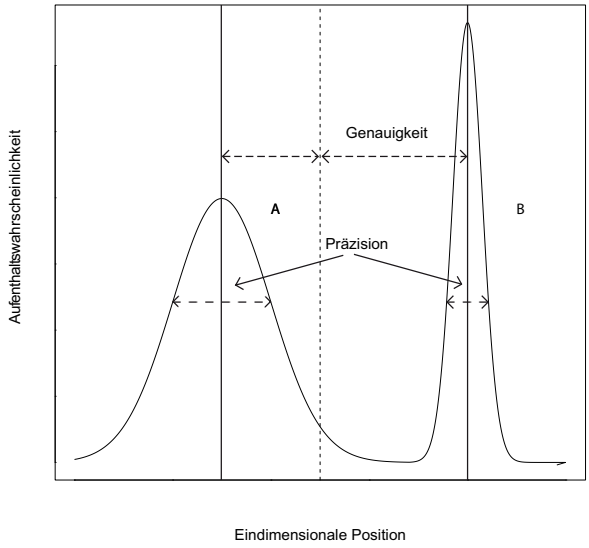


Abbildung 2.8: Genauigkeit („Accuracy“) und Präzision („Precision“) in der Indoorpositionierung

greifen: Die Genauigkeit wird nämlich durch die mittlere Abweichung vom realen Wert charakterisiert, während die Präzision durch die mittlere Abweichung vom Mittelwert der Messungen gegeben ist. Häufig unterliegen Positionierungssysteme einer gaußschen Verteilung, sodass Werte für Genauigkeit und Präzision auch als Abstand des Mittelwertes vom realen Wert beziehungsweise als Standardabweichung der Verteilung angegeben werden können.

Abbildung 2.8 zeigt beispielhaft eine Koordinatenschätzung eines recht genauen Positionierungssystems A und eines ungenaueren, aber hochpräzisen Positionierungssystems B.

Eine grundlegende Frage bei der Auswahl und Bewertung von Positionierungssystemen ist die *Mobilität*: Wenn die zu positionierenden Objekte beispielsweise Menschen sind, so bewegen sie sich relativ frei in alle Richtungen. Bei Maschinen, Fahrzeugen und Robotern ist hingegen häufig eine Einschränkung in der Bewegung vorgegeben: Fahrzeuge können sich nicht seitwärts bewegen, und passive Objekte auf einem Förderband können nur der typischerweise bekannten Bewegung des Förderbandes folgen. Natürlich hat das Einfluss auf die Auswahl eines Positionierungssystems. Während für den Menschen eine möglichst gleichmäßige Positionierung in der Fläche wünschenswert ist, kann für das Fahrzeug ein weniger allgemeines Positionierungssystem unter Verwendung der bekannten Bewegungsmöglichkeiten eine gute Wahl sein. Für Objekte auf einem Förderband ist ein ganz einfaches System, basierend auf RFID oder Barcodes, ausreichend, das nur an einer bestimmten Stelle das Objekt lokalisieren kann. Diese Information reicht aus, um den aktuellen Aufenthaltsort in

der Förderanlage aus der bekannten Bewegung derselben zu berechnen.
Ein anderer wichtiger Aspekt bei der Bewertung und Auswahl von Positionierungssystemen ist die Art der *Skalierung*: Für Systeme mit wenigen mobilen Entitäten und einer großen Fläche kann natürlich ein beliebig genaues, inertiales Positionierungssystem verwendet werden. Für viele Objekte ist eine solche Vorgehensweise nicht sinnvoll, da hier Kosten pro mobiler Entität entstehen.
Im Zusammenhang mit Kostenbetrachtungen, Skalierung und Privatsphäre ist die Klassifikation von Positionierungssystemen anhand der Aktivität oder Passivität der mobilen Entität in die folgenden drei Klassen sehr nützlich [Küp 05]:

- *Terminal-based Positioning*, in welcher die mobile Entität ohne fremde Hilfe eine Position berechnet.

- *Terminal-assisted Positioning*, in welcher die Positionsbestimmung zwischen mobiler Entität und einer Infrastruktur verteilt wird.

- *Terminal-free Positioning*, in welcher eine mobile Entität von einer Infrastruktur lokalisiert wird, ohne dass die mobile Entität beteiligt ist.

Die Verwendung von dedizierter Infrastruktur führt generell zu Anschaffungs-, Installations- und Wartungskosten, die mit der Fläche des Systems wachsen. Ein modernes, hochgenaues Positionierungssystem auf der Basis von Ultrabreitband-Technologie verwendet typischerweise vier oder mehr Sensoren für jeden Raum. Die Verwendung bestehender Infrastruktur wie WLAN führt hingegen zu keinen zusätzlichen Kosten, kann aber in der Genauigkeit und Zuverlässigkeit bei Weitem nicht mit dedizierten Systemen mithalten, da der WLAN-Standard nicht für die Positionierung von Geräten entwickelt wurde.
Bei Terminal-basierten Positionierungssystemen steigen die Kosten mit Genauigkeit, Stromverbrauch und der Anzahl der zu positionierenden Geräte. Jedes dieser Geräte benötigt nämlich zumindest eine Mess- und eine Recheneinheit, um seine Position zu bestimmen. Trotzdem hat dieser Ansatz einen entscheidenden Vorteil in Bezug auf die Privatsphäre: Nur die mobile Entität selbst bestimmt die Position und hat damit zunächst Zugriff auf die Positionsdaten.
Bei den hybriden „Terminal-assisted"-Positionierungssystemen sind meist alle diese Parameter relevant: Die Fläche, in der eine gewisse Infrastruktur vorhanden sein muss, die Anzahl mobiler Objekte, die eine gewisse Technologie benötigen, und die gewünschte Genauigkeit und Präzision der Ortsbestimmung. Deshalb versucht man in diesem Bereich häufig, eine bereits existierende Infrastruktur mit bereits existierenden mobilen Geräten zur Ortsbestimmung derselben zu verwenden. So werden in GSM-Mobilfunknetzen vielfach die Identifikationen umliegender Basisstationen zur Ortsbestimmung verwendet, oder es wird gerne auf WLAN zurückgegriffen, das ja in vielen Umgebungen und mobilen Geräten vorhanden ist. Für solche Systeme entstehen dann natürlich keine zusätzliche Kosten durch die Installation und den Betrieb von Infrastruktur.

2.4 Beispiele für Positionierungssysteme

Innerhalb von Gebäuden ist es erstaunlich schwer, eine Distanz zwischen zwei Punkten zu bestimmen. Deshalb führen die Ansätze der Lateration meist zu ungenauen Ergebnissen. In Bereichen, in denen die Ausbreitungscharakteristik der Signale bekannt ist – typischerweise bei Positionierung mit Sichtverbindung, führt Lateration zu guten Ergebnissen. Für die Bestimmung von Distanzen gibt es zwei Möglichkeiten: Die Messung von Signalstärken, die sich mit der Entfernung auf Grund von physikalischen Ausbreitungsgesetzen verändern, oder die Messung von Laufzeiten der Signale, deren Ausbreitungsgeschwindigkeit bekannt ist. Für die Bestimmung von Distanzen mit Zeitmessungen unterscheidet man die folgenden Verfahren, je nachdem, wessen Zeit verwendet wird und welche Entitäten eine synchronisierte, gemeinsame Zeit benötigen:

- *Time-of-Arrival:* Der absolute Zeitpunkt des Eintreffens eines Signals (Licht, Schall, Funk), das an einer bekannten Position ausgesendet wurde, kann von der mobilen Entität bestimmt werden.

- *Time-Difference-of-Arrival:* Der Zeitunterschied, mit dem zwei zu bekannten Zeitpunkten an bekannten Orten ausgesendete Signale (Licht, Schall, Funk) an der mobilen Entität eintreffen, kann bestimmt werden.

- *Roundtrip-Time-of-Flight:* Die Verzögerung, mit der eine Reflektion eines ausgesandten Signals (Licht, Schall, Funk) am Sender wieder eintrifft, kann bestimmt werden.

Die mit „Time-of-Arrival“ bestimmten Längen können mit Lateration (siehe Abschnitt 2.1.2) zu einer Position verarbeitet werden. Die größte Schwierigkeit ist hierbei, dass sowohl die Infrastruktur als auch die mobile Entität eine möglichst genaue gemeinsame Zeit benötigen, die gerade in drahtlosen Systemen schwer herzustellen ist. Ein klassisches Beispiel für diese Art von Positionierung ist das Satellitennavigationssystem GPS.
Der größte Vorteil von „Time-Difference-of-Arrival“-Methoden ist die Eigenschaft, dass hier die mobile Entität keine synchrone Uhr benötigt, sondern lediglich die Infrastruktur, die die Signale aussendet. Diese Form von Positionierung wird vor allem in Mobilfunksystemen verwendet, deren Basisstationen in der Regel hervorragend untereinander synchronisiert sind.
Im Falle von „Roundtrip-Time-of-Flight“ hingegen wird keinerlei Zeitsynchronisation benötigt. Die Reflektionen, die zur Distanzbestimmung verwendet werden, müssen nicht unbedingt physikalischer Natur sein, sondern können auch beispielsweise WLAN-Bestätigungspakete sein, die von einem Access-Point als Antwort auf ein eingehendes Paket versendet werden [Haus 11].
Neben der folgenden Aufstellung verschiedener Arbeiten zur Indoorpositionierung sei auch auf die qualitativ sehr hochwertigen Übersichtsarbeiten [High 01, Huan 10] hingewiesen.

2.4.1 Hochempfindliche Satellitennavigation, Pseudolites

Die Signale der verschiedenen Satelliten-basierten Positionierungssysteme leiden ganz besonders unter Dämpfungseffekten und Mehrwegeausbreitung innerhalb von Gebäuden. Die Signalstärken, die beispielsweise von GPS-Satelliten in Gebäuden gemessen werden, liegen außerhalb der typischen Empfindlichkeit von GPS-Empfängern. Außerdem verwenden diese Systeme häufig die Annahme, dass eine direkte Sichtverbindung zwischen Satellit und Gerät besteht, sodass Signale, die durch Reflektionen wahrgenommen werden, sich stark irreführend auf die Berechnung auswirken. Der Einfluss solcher Reflektionen kann sehr gut durch starke Sender am Boden, sogenannte *Pseudolites*, vermindert werden. Allerdings ist ein solches System recht aufwändig, denn diese Sender müssen sehr gut zeitsynchronisiert und ihr Ort sehr genau bekannt sein. Obwohl Pseudolites hervorragende Genauigkeiten im Zentimeterbreich erreichen, eignen sie sich weniger in Szenarien, wo auch zu den Pseudolites keine Sichtverbindung herrscht [Cobb 97].
Ein anderes Beispiel für die Erweiterung von GPS ins Gebäudeinnere ist das Locata-System [Loca 12]. Locata verwendet ein zeitsynchronisiertes Netzwerk von Basisstationen, die GPS-ähnliche Signale im lizenzfreien ISM-Band versenden. Diese Signale können dann von speziellen mobilen Komponenten verwendet werden, um in Kombination mit GPS-Signalen für den Außenbereich eine durchgehende Positionierung mit sehr hoher Genauigkeit zu ermöglichen [Rizo 10].
Obwohl die Signalausbreitung in Gebäuden extrem komplex ist, gibt es ein paar Ansätze, die mit hochempfindlichen Empfängern und genauen, dreidimensionalen Modellen der Gebäude zumindest grundsätzlich eine Positionierung mit den GPS-Signalen ermöglichen [Digg 02, Kee 01].

2.4.2 Licht-basierte Systeme

Die Verwendung von speziellem Licht für die Indoor-Positionierung ist naheliegend, weil Licht von den meisten typischen Baumaterialien reflektiert wird. Da diese Reflektionen bei Licht wegen der kurzen Wellenlänge viel deterministischer ausfallen als bei typischen Radio-basierten Systemen, kann mit den Methoden des „Roundtrip-Time-of-Flight“ eine ziemlich genaue Längenmessung erfolgen.
Solche Systeme werden häufig *LiDAR-Systeme („Light Detection and Ranging“)* genannt. Diese Bezeichnung soll daran erinnern, dass das grundlegende Funktionsprinzip dasselbe ist wie bei funkbasierten Radar-Systemen. Als typisches Beispiel für diesen Bereich sei die Arbeit [Trav 05] genannt, in der mittels LiDAR ermittelte Tiefeninformationen über einen Kalman-Filter mit den Daten eines inertialen Sensorsystems kombiniert werden. Im Ergebnis ist die Fehlercharakteristik des verwendeten LiDAR-Systems geeignet, einen guten Ausgleich zu den Fehlern der inertialen Komponente zu ermöglichen.
LiDAR-Systeme können auch verwendet werden, um dreidimensionale Kar-

ten der Umgebung mit SLAM-Methoden (siehe Abschnitt 2.1.7.1) zu erzeugen. Diese Art von SLAM-Lokalisierung wird häufig *„Red-Green-Blue-Depth-SLAM“ (RGBD-SLAM)* genannt. Alternativ zur Verwendung mit Methoden des SLAM können Eigenheiten des Tiefenbildes (Ecken, Kanten, Mittelwerte, etc.) verwendet werden, um eine Position zu bestimmen [Adam 98].
Dieselbe Art von Tiefenbild kann auch mit einer einfacheren Technologie erzeugt werden, die nicht die Signallaufzeit von Laser zur Tiefenbildbestimmung verwendet, sondern die Verformungen eines gewissen Licht-Musters. Die Kinect [Micr 12], ein Endgerät für die Spielekonsole XBox 360 von Microsoft, verwendet eine solche Technologie und liefert neben einem Farbbild auch ein Tiefenbild. Leider ist die Reichweite einer solchen Technologie durch die Auflösung der Kamera beschränkt, so dass eine Kinect nur bedingt zur Positionierung in größeren Gebäuden geeignet ist [Banh 12].
Eine ganz andere Art von Licht-basierten Systemen verwendet die Methode der Proximity-Detection. Dabei wird von einem mobilen Gerät ein moduliertes Lichtsignal ausgesendet, das von einem Sensornetzwerk aufgenommen wird und geeignet ist, die mobile Entität zu identifizieren. Eines der ersten Systeme dieser Art ist das ActiveBadge-System [Want 92], welches mit an der Decke montierten Infrarot-Sensoren die Infrarot-Signale auf der Schulter getragener „Badges“ wahrnehmen und dadurch auf die Position einer Person schließen kann.
Obwohl auch die Verwendung von Kamerabildern in gewisser Weise ein Licht-basiertes System darstellt, soll, wegen der grundlegenden Bedeutung der Bilderkennung in der Indoor-Positionierung, diese Verfahren im folgenden Abschnitt gesondert betrachtet werden.

2.4.3 Kamera-basierte Systeme

Kamera-basierte Systeme sind wahrscheinlich die Systeme, die der Wahrnehmung des Menschen am ehesten entsprechen. Der Mensch orientiert sich in allen Umgebungen hauptsächlich anhand visueller Informationen. So ist es nur natürlich, dass man auch in der Informatik eine Ortsbestimmung über das Aussehen der Umgebung versucht. Grundsätzlich unterscheidet man zwei Arten von Kamerapositionierungssystemen: Auf der einen Seite gibt es die Positionierungssysteme, bei denen die Kamera an einem beweglichen Objekt befestigt wird, dessen Position bestimmt werden soll. Auf der anderen Seite gibt es die Systeme, bei denen Kameras als Infrastruktur an bekannten Punkten befestigt sind. Hierbei werden Objekte lokalisiert und identifiziert, die sich durch das Blickfeld der Kameras bewegen [Maut 11].
Im ersten Fall, also wenn die Position der Kamera selbst von Interesse ist, wird typischerweise mit einer Methode der Szenenanalyse die aufgenommene visuelle Information weiterverarbeitet. Dabei können typische Umgebungsmerkmale (Landmarken, Bild-Features, geometrische Eigenheiten) mit einer Datenbank von Referenzbildern mit bekannten Positionen verglichen

werden und so zu einer Positionsbestimmung führen. In einigen Systemen [Hash 97, Kabu 87, Se 02] werden hierzu Landmarken in der Umgebung überwacht, die in einem aufgenommenen Kamerabild effektiv lokalisiert und unterschieden werden können. Diese Landmarken können entweder explizit ausgebracht oder aus der natürlichen Umgebung extrahiert werden.

Andere Systeme, wie auch das in Kapitel 3.3 vorgestellte System, verwenden die natürliche Varianz der Umgebung zur Positionierung, indem das zu einem Zeitpunkt von der mobilen Entität aufgenommene Bild mit einer Bilddatenbank verglichen wird. Ähnlich funktioniert auch das System [Kawa 10], in welchem ein aufgenommenes Bild mit einigen vorher aufgenommenen charakteristischen 360°-Panoramabildern verglichen wird.

Eine andere Art, mit Kamerainformationen umzugehen, besteht darin, die Bewegung der Kamera selbst aus den Bewegungen von Bildteilen in aufeinanderfolgenden Bildern zu bestimmen. In diesem Bereich kann der optische Fluss [Horn 81, Luca 81, Camp 04] verwendet werden, um die Bewegung der Kamera zu bestimmen. Der optische Fluss bezeichnet dabei das Vektorfeld, welches die zweidimensionale Bewegung von Pixeln oder Featurepunkten innerhalb konsekutiver Bilder beschreibt. Die Bestimmung einer dreidimensionalen Bewegung aus diesen Informationen ist allerdings sehr komplex und fehleranfällig. Ohne zusätzliche Annahmen, Stereo-Vision oder zusätzliches Wissen kommt man in der Regel nicht aus.

Selbstverständlich fallen in die Kategorie der kamerabasierten Systeme auch die Methoden des „Simultaneous Localization and Mapping (SLAM)“. Hierbei wird die Kamerabewegung aus einer Zeitsequenz von Bildern bestimmt und gegebenenfalls über das Erkennen von Schleifen nachkorrigiert (siehe Abschnitt 2.1.7.1).

Es gibt auch noch einige weniger verbreitete Systeme, die in gewisser Weise als Kamera-basiert anzusehen sind. Insbesondere gehören dazu die Positionierungssysteme, bei denen Dead Reckoning für Fahrzeuge in Beobachtung der Bodenfläche etwa mit hochwertigen Computermäusen durchgeführt wird [Naga 00, Dill 10].

Bei der Verwendung stationärer Kameras zur Positionsbestimmung ist das Problem der Positionsbestimmung relativ einfach zu lösen. Denn in der Regel kann die Position von beweglichen Objekten aus der Verdeckung von Bildteilen recht gut bestimmt werden. Problematisch ist vielmehr die Identifizierung und Unterscheidung der zu lokalisierenden Objekte voneinander und von der Umgebung. Daher eignen sich solche Systeme besonders in relativ statischen Umgebungen, etwa bei der Steuerung und Überwachung von Fertigungssystemen, bei denen die erwünschten Bewegungen stets gleich und bekannt sind und die Hauptaufgabe des Lokalisierungssystems darin besteht, Fehler in den Abläufen zu entdecken und zu melden.

2.4.4 Funk-basierte Systeme

Die meisten Systeme zur Positionsbestimmung in Gebäuden basieren heutzutage auf Funktechnologie. Dies begründet sich darin, dass mit Funk extrem hohe Genauigkeiten erreicht werden können, da Funk eine bessere Stabilität gegenüber geringfügigeren Störungen hat als etwa Licht. Ein weiterer Grund besteht darin, dass Funktechnologien im Grunde überall vorhanden sind. So kann man nahezu weltweit innerhalb von Städten die Position eines Smartphones allein anhand der MAC-Adressen umliegender WLAN-Zugangspunkte bestimmen, die von diesen meist ständig ausgesendet werden.
Die Bestimmung einer Position mit Funktechnologie ist mit der höchsten Qualität über die Zeit-basierten Methoden „Time-of-Arrival“, „Time-Difference-of-Arrival“ und „Roundtrip-Time-of-Flight“ möglich. Diese Methoden unterscheiden sich im Wesentlichen darin, wer Zeitmessungen vornimmt, und welche Komponenten eine gemeinsame Zeit und daher eine komplexe Uhrensynchronisation benötigen. Bei „Time-of-Arrival“ werden zwischen Infrastruktur und mobiler Entität – manchmal auch zwischen den mobilen Entitäten selbst zur Verminderung von Fehlern – Nachrichten ausgetauscht, deren Aussendezeit bekannt ist. Somit ist beim Eintreffen am Empfänger die Laufzeit des Signals bekannt, also die Zeit, die zwischen Senden und Empfangen vergangen ist. Bei Radiosignalen, die eine sehr hohe und verhältnismäßig konstante Ausbreitungsgeschwindigkeit in der Größenordnung der Lichtgeschwindigkeit $c_0 = 300.000\text{kms}^{-1}$ haben, sind damit gute Entfernungsbestimmungen möglich, die mit der Methode der Lateration (siehe Abschnitt 2.1.2) zu einer Ortsbestimmung weiterverarbeitet werden können.
Die Methode „Time-Difference-of-Arrival“ ist etwas einfacher umzusetzen, weil die mobile Entität keine genaue zeitliche Synchronisation mit den aussendenden Stationen benötigt. In diesem Fall wird nicht die Laufzeit der Signale selbst bestimmt, sondern lediglich der Laufzeitunterschied für Paare von Sendern. Aus diesen Informationen berechnet man mit der Methode der hyperbolischen Lateration (siehe Abschnitt 2.1.3) den wahrscheinlichsten Aufenthaltsort der mobilen Entität. Die Methode „Roundtrip-Time-of-Flight“ ist eine Methode, bei der (in der Regel) die gesamte Zeitmessung bei der mobilen Entität erfolgen kann. Damit ist keinerlei Zeitsynchronisation nötig. Allerdings benötigt ein solches System eine Reflektion. Dies führt insbesondere dann zu zusätzlicher Komplexität, wenn die Reflektion nicht auf einer physikalischen Interaktion beruht, sondern von einer aktiven Komponente herrührt. In diesem Fall ist nämlich eine Verarbeitungszeit an dieser aktiven Komponente zu schätzen und von der gemessenen Zeit abzuziehen, um eine zuverlässige Positionsbestimmung zu ermöglichen.
Funksignale ermöglichen auch ohne besonders großen Aufwand eine Bestimmung von Winkelinformationen. Denn wenn Sender und Empfänger in ausreichendem Abstand stehen, kann man das Funksignal als parallele Wellenfront modellieren, und mit einem Array aus Antennen den Zeitversatz des Eintreffens dieser Wellenfront an unterschiedlichen Antennen und daraus den Winkel

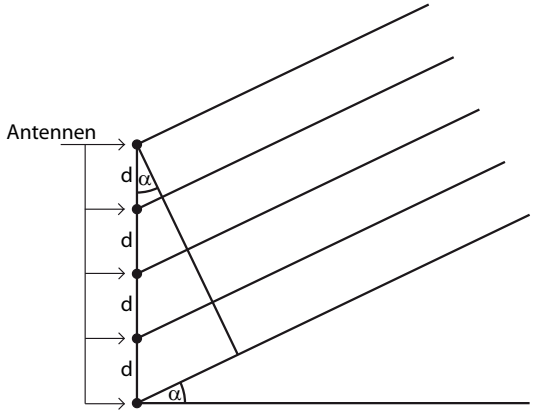


Abbildung 2.9: Prinzipielle Funktionsweise eines Antennenarrays

bestimmen.
Abbildung 2.9 zeigt die grundlegende Geometrie eines einfachen Antennen-Arrays: Mehrere Antennen werden im Abstand d voneinander aufgestellt und der Winkel α, aus dem die planare Wellenfront ankommt, berechnet sich aus dem Zeitversatz, der durch die unterschiedlich langen Wege zu den Antennen entsteht, durch Umstellen der folgenden Gleichung:

$$\Delta t = \frac{d}{c_0} \sin \alpha$$

Die Zeit-basierten Ansätze zur Positionsbestimmung haben in Gebäuden aber Schwierigkeiten, Effekte wie Reflektionen, Abschattungen und Mehrwegeausbreitung zu überwinden. Häufig ist ein Signal zwischen Sender und Empfänger nicht auf dem direkten Ausbreitungsweg, sondern auf irgendeinem anderen Weg gelaufen. Die Laufzeitmessung ist dann irreführend. Dennoch wird gerade bei spezialisierten Systemen in Szenarien, in denen zwischen Sender und Empfänger Sichtkontakt besteht, auf diese Techniken zurückgegriffen. Dann kann ihre Genauigkeit von den anderen bekannten Verfahren bei Weitem nicht erreicht werden.
Seit der Öffnung des gesamten Frequenzspektrums für die Ultra-Breitband-Technik in Europa im Jahr 2007 [The 07] sind auf diesen Techniken einige der leistungsstärksten Funkpositionierungssysteme entstanden.
Ubisense ist beispielsweise ein kommerzielles Positionierungssystem auf Basis von Ultra-Breitband-Technik, das mit den Methoden „Time-Difference-of-Arrival“ und „Angle-of-Arrival“ die Positionierung aktiver Tags, die kurze UWB-Pulse aussenden, ermöglicht, und damit eine Positionserfassung mit einer Genauigkeit in der Größenordnung von 15cm in drei Dimensionen ermöglicht [Steg 05].
Neben diesen Zeit-basierten Systemen gibt es auch die Möglichkeit, die Komplexität der Funkausbreitung, insbesondere innerhalb von Gebäuden, direkt auszunutzen. So kann man bei den Methoden des „Fingerprintings“ in einer

Trainingsphase – häufig als „Offline-Phase“ bezeichnet – Signalstärkeinformationen an bekannten Orten in der Umgebung messen und sammeln, um dann in einer Betriebsphase – häufig „Online-Phase“ genannt – mit den verschiedensten Methoden des maschinellen Lernens aus einer Messung den Aufenthaltsort zu bestimmen. Ein klassisches Beispiel für diese Art von Positionierung ist das RADAR-System [Bahl 00]. Bei diesem System werden WLAN-Signalstärken an bekannten Orten in Gebäuden aufgenommen, und für eine Messung mit unbekanntem Ort wird dann mit einer gewichteten Nachbarsuche in der Trainingsdatenbank eine Position ermittelt. Die Bedeutung von Messfehlern und der Einfluss verschiedener Regressionsmodelle auf derartige Methoden wurde durch Simulationen genau untersucht [Chen 02]. Auf der Idee von Signalstärke-Fingerprinting baut auch das Positionierungssystem „SMARTPOS“ [Kess 11b] auf, das zusätzlich noch Kompass- und Beschleunigungssensoren für die Positionsbestimmung verwendet. Dieses System wird im Abschnitt 3.2 eingehend beschrieben. Insbesondere die Verwendung von bereits vorhandenen WLAN-Systemen zur Positionierung erfreut sich zunehmender Beliebtheit, da sie zusätzliche Kosten vermeidet und mittlerweile durchaus akzeptable Genauigkeiten bei vertretbarem Wartungsaufwand erreicht.

Eine andere, funkbasierte Technologie, die mitunter zur Positionierung verwendet werden kann, ist *Radio Frequency Identification (RFID)*. Bei RFID werden passive oder aktive elektronische Komponenten („Tags“), die in der Lage sind, eine Identifikation auszusenden, von Lesegeräten erfasst. Die Reichweite solcher Systeme ist extrem begrenzt, da bei passivem RFID die gesamte Stromversorgung der Tags über Induktion erfolgen muss. Diese Begrenztheit des Arbeitsbereiches eines RFID-Lesegerätes eignet sich daher hervorragend zur Positionierung über Proximity Detection. Dabei werden Lesegeräte an strategisch sinnvollen Orten (Türdurchgänge, Grenzen von Fertigungsabschnitten, Förderbänder) angebracht, die die Fläche in Zonen unterteilen. So kann über den Aufenthalt mobiler Tags in den unterschiedlichen Zonen Buch geführt werden.

2.4.5 Inertiale Navigation

Unter dem Begriff *Inertiale Navigation* fasst man alle Positionierungs- und Navigationssysteme zusammen, die auf der Bestimmung von Änderungen des inertialen Systems der mobilen Entität beruhen. Es wird also keine Position in einem beliebigen Koordinatensystem berechnet, sondern es wird zu einem Zeitpunkt ein Koordinatensystem mit der mobilen Entität in einem gewählten Bewegungszustand, typischerweise im Ruhezustand im Nullpunkt, festgelegt, und die Position anhand der Beobachtung von Sensoren mit der Methode aus Abschnitt 2.1.6 fortgeschrieben. Diese Sensoren bestimmen Änderungen des Bewegungszustandes auf Grund globaler physikalischer Effekte. Für einen solchen Fortschreibevorgang, der auch den Begriff Koppelnavigation begründet, eignen sich Sensoren wie Beschleunigungssensoren, Gyroskope oder auch Ma-

gnetometer.
In diesem Bereich gibt es nur wenige, sehr spezielle Systeme für den Indoorbereich. Dies liegt an der prinzipiellen Genauigkeitsbeschränkung solcher Systeme: Jeder Sensor ist mit einem zufälligen Fehler behaftet und misst typischerweise nur eine Ableitung des Ortes nach der Zeit.
Da sich zufällige Fehler in diesen Messungen nicht aufheben, berechnet ein solches System nach einer bestimmten Zeit einen realitätsfernen Bewegungszustand, sodass das System absolut unbenutzbar wird. Bei einfachen Sensoren, wie sie in derzeitigen Smartphones verbaut sind, erreicht man Betriebszeiten von wenigen Sekunden, bei extrem aufwändigen Sensoren aus dem militärischen Bereich durchaus Minuten. Um die Akkumulierung solcher Fehler bei der inertialen Navigation zu überwinden, kombinieren Systeme inertiale Sensorik häufig mit anderen Positionierungssystemen. Dabei sind der Kalman-Filter [Kalm 60] oder die Verwendung von Partikelfiltern [Arul 02, Gust 02] sehr beliebt. Positionierungssysteme, die allein auf inertialen Sensoren beruhen, verwenden häufig Schritterkennungsmethoden, um den Einfluss der Fehler zu reduzieren.
In diesem Bereich ist die Forschung sehr aktiv, weil die Sensorik ständig besser wird und trotzdem noch kein „Erfolgsrezept“ gefunden wurde, wie man am Besten mit den Sensorwerten umgeht. Gute Beispiele für Navigation mit inertialen Sensoren sind die Arbeiten [Davi 10, Stor 10, Xiao 11, Link 11, Goya 11, Wood 08, Even 06]. Der Bereich der Koppelung von inertialer Sensorik mit globalen Positionierungssystemen wird sich in den nächsten Jahren stark weiterentwickeln, denn die klassischen Positionierungssysteme auf WLAN-Basis sind eher langsam, und benötigen für Tracking-Anwendungen dringend die Unterstützung von schnellen Sensoren, wie sie beispielsweise von inertialen Sensoren geleistet werden kann.

2.4.6 Schall-basierte Systeme

Schall-basierte Systeme verwenden die Ausbreitung von Schallwellen im Raum für die Bestimmung einer Position. Gerade bei Schall-basierten Systemen gibt es eine große Anzahl physikalischer Effekte, die für eine Positionierung verwendet werden.
Ganz klassisch lassen sich mit Schall, insbesondere mit dem für Menschen nicht hörbaren Frequenzen im Ultraschallbereich, sehr gut Proximity-basierte Systeme aufbauen. Die Bereiche, in denen ein bestimmtes Schallsignal empfangbar ist, entsprechen sehr häufig und sehr genau auch den Bereichen, die die Menschen logisch trennen (Zimmer, Flure, etc.). Hierbei ist es von großem Vorteil, dass nur die Offenheit für Schall und nicht eine direkte Sichtverbindung zwischen Sender und Empfänger benötigt wird. Insbesondere reflektieren viele Baumaterialien (Wände, Decken) Schallwellen sehr viel besser als etwa Infrarot-Licht, und ermöglichen so eine raumumfassende Ausbreitung.
Die Verwendung von Schall zur Indoor-Positionierung ist schon sehr früh

sehr erfolgreich gewesen. So verwendet beispielsweise das System ActiveBat [Ward 97] ein Sensornetzwerk aus Ultraschallempfängern zur Positionsbestimmung mobiler Einheiten, die Ultraschallsignale aussenden. Dieses System erreicht Genauigkeiten im Zentimeterbereich, ist aber durch die Verwendung einer sehr dichten Infrastruktur sehr teuer.
Wegen der relativ langsamen Ausbreitungsgeschwindigkeit von Schall – knapp 300ms^{-1} – ist es einfach, mit mehreren Mikrofonen den Eingangswinkel eines Schallsignales zu bestimmen. Ein Beispiel, auch wenn es sich nicht um ein Positionierungssystem, sondern um ein Lokalisierungssystem handelt, ist eine Anwendung, mit der sich die Richtung, aus der ein lautes Schallsignal (ein Schuss) gekommen ist, bestimmen lässt [Bogg 04].
Eine andere Art Schall-basierter Systeme verwendet die typische Geräuschkulisse eines Gebäudes zur Positionsbestimmung. In der Arbeit [Tarz 11] wird dazu eine Methode vorgestellt, mit der verschiedene Eigenschaften der Geräuschkulisse aus einem Mikrofonsignal extrahiert werden können. So entstehen, ähnlich zum WLAN-Fingerprinting, Feature-Vektoren, und aus einem Vergleich von Featurevektoren mit Messungen kann dann auf den Ort geschlossen werden. Ein solches System benötigt allerdings ausreichend unterschiedliche, reproduzierbare Geräuschkulissen und ist daher eher kritisch zu betrachten. Die Autoren selbst bemerken, dass schon Gespräche einen negativen Einfluss auf die Positionierung haben. Außerdem ist ein solches System bei Weitem einem WLAN-basierten System unterlegen, sofern denn eine WLAN-Infrastruktur existiert.

2.4.7 Druck-basierte Systeme

Ein eher einzigartiges Positionierungssystem names „Smartfloor“ verwendet Druckmessungen an Bodenplatten, um die Position und Identität von Nutzern in Gebäuden zu bestimmen [Orr 00]. Dabei konnte das System zwischen 15 Nutzern unterscheiden und deren Position im Gebäude bestimmen. Das Anwendungsgebiet diesen Systems ist im Bereich des „Ambient Assisted Living“ zu suchen, denn der entscheidende Vorteil liegt in seiner Einfachheit und Unsichtbarkeit. Ähnlich wie bei den Positionierungssystemen mit Kameras, die fest verortet einen gewissen Bereich überwachen, liegt die Schwierigkeit bei diesem System ohnehin nicht in der Ortsbestimmung, sondern in der Unterscheidung verschiedener Nutzer.

2.5 Zusammenfassung und Ausblick

Die Technologien und Verfahren zur Bestimmung der Position eines mobilen Endgeräts sind mannigfaltig. In diesem Abschnitt wurden zunächst die gängigen Verfahren zur Bestimmung von Positionen mit geometrischen und statistischen Betrachtungen eingeführt. Eine kurze Übersicht zeigt Tabelle 2.1, in

Verfahren	**Messgrößen**	**Charakteristische Eigenschafen**
Lateration	Längen, Zeitpunkte	Längenmessung schwierig durch Mehrwegeausbreitung
Hyperbolische Lateration	Längenunterschiede, Zeitdifferenzen	Benötigt nur Zeitsynchronsiation der Infrastruktur
Angulation	Winkel, Phasenverschiebungen	Ortsauflösung nur bei Sichtlinie sinnvoll möglich
Proximity Detection	Sichtbarkeit, physikalische Nähe	Sehr einfach und zuverlässig, relativ ungenau
Intertiale Navigation	Beschleunigungen, Bewegungen	Messfehler summieren sich auf, unabhängig von Infrastruktur
Fingerprinting	Vektoren ortsvariabler Messgrößen, Bilder	Empfindlich bei Änderungen der Umgebung, Stabil bei Mehrwegeausbreitung
SLAM	Bilder, Tiefenbilder	Sehr genau, unabhängig von Infrastruktur, extrem rechenintensiv

Tabelle 2.1: Tabellarischer Vergleich der Verfahren

der die jeweils herausstechendsten charakteristischen Eigenschaften der wichtigsten Verfahren genannt werden. Durch die Vielfalt dieser Verfahren gibt es in der Positionierung auch eine Vielzahl an Charakteristika in Bezug auf Fehlerverhalten, Geschwindigkeit und Aufwand. So führt eine Messung mit normalverteilten Fehlern bei den verschiedenen Verfahren auf verschiedene Fehlerflächen, die in den Abbildungen stets mit gestrichelten Linien dargestellt wurden. Da diese Fehler mitunter auch sehr groß werden und insbesondere bei den nicht-geometrischen Verfahren nicht unbedingt einer vernünftigen Beschränkung unterliegen, benötigt man Methoden, mehrere Positionierungsverfahren zu kombinieren, um die Unterschiede im Verhalten gewinnbringend einzusetzen. Dazu eignen sich der Kalman-Filter wegen seiner geringen Berechnungskomplexität und seiner oft nicht unrealistischen Annahme über die Linearität des Systems gerade für Echtzeit-Anwendungen auf mobilen Endgeräten. Bessere Ergebnisse, bei weit höherem Aufwand, liefern Partikelfilter, mit denen insbesondere auch Karteninformationen auf einfache Weise in die Positionierung integriert werden können. Für Anwendungen auf mobilen Endgeräten ist der hohe Rechenaufwand allerdings eine Hürde, besonders dann, wenn das Positonierungssystem längere Zeit verwendet werden soll. Denn schon die Messung der Umgebungsparameter führt zu einem hohen Stromverbrauch. In Zukunft wird man immer höhere Genauigkeiten in der Positionsbestimmung erreichen und auch benötigen, wobei den einzelnen Verfahren natürlich jeweils Grenzen durch physikalische und elektrotechnische Effekte gesetzt sind.
Die Bestimmung der Position kann man auch als eine notwendige Brückentech-

nologie begreifen, die die Vision des „Ubiquitous Computing“ über das derzeit durch die hohe Verbreitung von reinen Client-Server-Architekturen recht zentrale Internet ermöglicht. Entsprechend dieser Vision wird es auch bald mehr und mehr Architekturen geben, in denen die Dienste nicht über eine Client-Server-Architektur, sondern in verteilter Weise vor Ort erbracht werden. Damit wird sich für den Bereich der Positionierung eine weit höhere Flexibilität ergeben, denn dann kann vor Ort mit den optimalen Daten, Parametern und Systemen gearbeitet werden. Eine Standardisierung von Schnittstellen für die Kommunikation von Smartphones mit Gebäuden und digital augmentierten, physischen Objekten ist hier mit Sicherheit noch ausstehend.

3 Beiträge zur Positionierung

Im Folgenden werden drei Beiträge zur Positionierung in Gebäuden vorgestellt. Im Abschnitt 3.1 wird eine Simulations- und Designumgebung für Positionierungsalgorithmen und deren Untersuchung vorgestellt. Damit kann die Signalausbreitung von WLAN-Signalen simuliert werden und einige Evaluierungen können auch mit experimentell ermittelten Daten durchgeführt werden. Der entscheidende Vorteil dieser Vorgehensweise liegt in der einheitlichen Behandlung von Experiment und Simulation: Algorithmen werden nur einmal in einer Hochsprache für numerische Mathematik definiert. Abschnitt 3.2 stellt ein System vor, welches die neuartige Sensorik mit WLAN-Positionierung kombinieren kann, und dabei eine signifikante Verbesserung im Vergleich zur klassischen WLAN-Positionierung erreicht. Im Abschnitt 3.3 wird dann ein Kamerabasiertes Positionierungssystem auf Grundlage von Feature-Vergleichen vorgestellt, welches durch geschicktes Overfitting und eine Vorfilterung der Datenbank mit einer beliebigen, ungenauen Positionierung eine recht gute Bestimmung von Position und Blickrichtung eines Nutzers ermöglicht. Dabei wurde auch eine Distanz-Korrektur konzipiert, die die Verschiebung zwischen der Aufnahmeposition eines Datenbankbildes und der Aufnahmeposition eines aktuellen Bildes schätzt. Eine Analyse des Einflusses partieller Überdeckungen auf die Positionierung und die detaillierte Evaluierung in einer recht großen Positionierungszelle – ein Flur eines Gebäudes, der leicht durch WLAN-Positionierung von den anderen Gängen getrennt werden kann – runden die Darstellung ab.

3.1 Simulation von WLAN-Signalausbreitung unter Verwendung von CAD-Plänen

Wie im vorangehenden Kapitel beschrieben, kann die Bestimmung der Position einer mobilen Entität auf sehr verschiedene Arten und Weisen durchgeführt werden. Eine besonders attraktive Klasse von Positionierungssystemen verwendet allerdings weder spezielle Endgeräte noch spezielle Infrastruktur für die Bestimmung der Position. Obwohl das System Active-Badge [Want 92] mit derselben Idee gestartet ist, nämlich dass fast alle PDA's der damaligen Zeit mit Infrarot-Technologie ausgestattet waren, ist heutzutage Infrarot nicht mehr das Mittel der Wahl. Die derzeit am weitesten – sowohl in Smartphones und Handys als auch als Infrastruktur – verbreitete Technologie ist WLAN. Darüber hinaus bietet der WLAN-Standard (802.11b,g,n) einige Mechanismen, die sich zu einer durchgehenden und anonymen Positionierung ausnut-

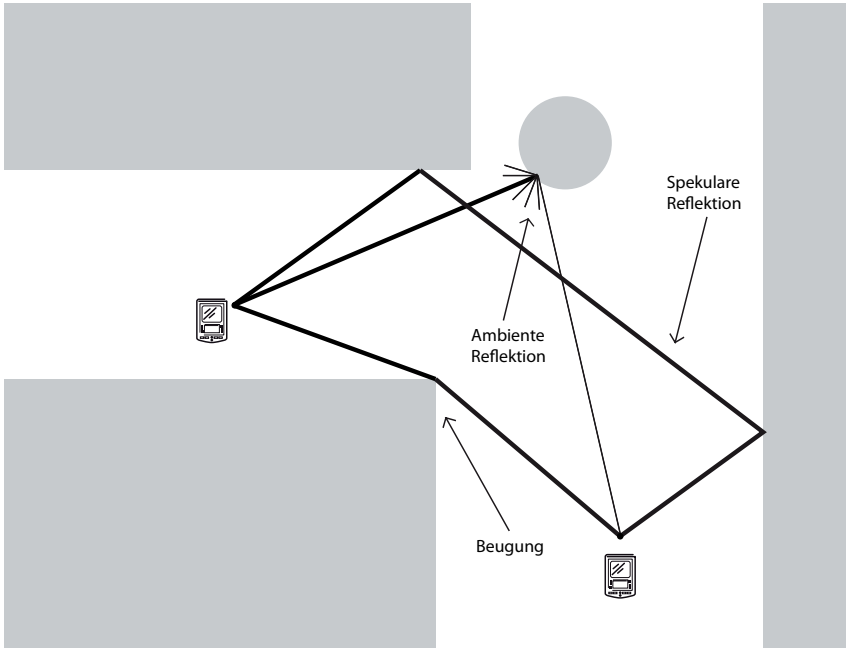


Abbildung 3.1: Besondere Störeinflüsse bei der Funkausbreitung in Gebäuden

zen lassen. Um ein Handover zwischen verschiedenen WLAN-Zugangspunkten zu ermöglichen, versenden alle WLAN-Zugangspunkte regelmäßig Nachrichten (Beacons), um umliegende Stationen von ihrer Präsenz zu unterrichten. Die WLAN-Empfänger können diese Beacons empfangen und einen Indikatorwert für die Signalstärke (Received-Signal-Strength-Indication, RSSI) dieser Beacons verwenden, um den Zugangspunkt mit dem jeweils besten Signal für die Kommunikation zu verwenden. Damit dieser Mechanismus zuverlässig funktioniert, ist jedes WLAN-Empfangsgerät in der Lage, Beacons zu empfangen und die Signalstärke mehrerer Access-Points zu überwachen.

Mit diesen Fähigkeiten kann man natürlich Positionsbestimmungen mit den verschiedensten Ansätzen durchführen. Leider ist aber die Signalausbreitung in Gebäuden so komplex und die Messgenauigkeit der Endgeräte so gering, dass eine spezielle Umgebung benötigt wird, in der reproduzierbare Ergebnisse für Positionierungsalgorithmen erzeugt werden können, und insbesondere auch die Bedeutung von Effekten und Umgebung analysierbar wird. In der Arbeit [Wern 11b] wurde eine Simulations- und Analyseumgebung für Indoorpositionierung geschaffen, in der in Forschung und Lehre an und mit Positionierungsalgorithmen gearbeitet werden kann.

Da es sich bei WLAN um ein schmalbandiges Funkübertragungssystem handelt, kann die Empfangssignalstärke als frequenzunabhängige Größe modelliert werden. Aus empirischen Studien weiß man, dass die Empfangssignalstärke stark von der Distanz zwischen Sender und Empfänger, von den Ausbreitungspfaden und weiteren Effekten wie Rauschen, Selbst-Interferenz, Verzerrung, Streuung und Reflektionen abhängt (siehe Abbildung 3.1).

Obwohl die physikalischen Vorgänge, die im Falle von WLAN auf die Empfangssignalstärke Einfluss haben, bekannt sind, können sie in der Praxis selten verwendet werden. Denn typischerweise fehlt ein Modell der Umgebung, das genau genug ist. Diese Genauigkeit müsste ein Bruchteil der Wellenlänge (ca. 13cm bei WLAN) sein, wenn auf Grund physikalischer Überlagerung die Signalstärke bei Mehrwegeausbreitung berechnet werden soll. Deswegen verlässt man sich in diesem Bereich typischerweise auf statistisch verifizierte, vereinfachte Ausbreitungsmodelle, um Ortsinformationen aus Signalstärken von WLAN-Beacons zu berechnen.

3.1.1 Modelle der Funkausbreitung

Eines der einfachsten Ausbreitungsmodelle, das im Falle von WLAN angewendet werden kann, ist die Ausbreitung von Funkwellen im Vakuum. Dabei wird die ausgesendete Energie P_t – unter der Annahme, dass die Funkquelle gegenüber der Ausbreitungsstrecke klein ist – auf einer Kugeloberfläche mit Radius d verteilt, und die empfangene Energie dann nochmals durch die effektive Apertur A_e der Empfängerantenne modifiziert. Die Apertur entspricht der Fläche, aus der die Energie „vollständig“ entnommen wird und hängt typischerweise von der Frequenz ab. Wird aber ein schmalbandiges Signal (wie WLAN) angenommen, die Frequenz $f = \frac{c}{\lambda}$ als konstant angenommen werden und folglich die Apertur ebenfalls durch eine Konstante A_e beschrieben werden. Zusammenfassend ergibt sich die empfangene Energie P_r wie folgt:

$$P_r = \frac{P_t A_e}{4\pi d^2}$$

Der Mechanismus der Ausbreitung ist nochmals in Abbildung 3.2 zu sehen. Die Apertur (gelbe Fläche in Abbildung 3.2) wird in Quadratmetern angegeben. Die detektierbare Energie nimmt dabei quadratisch mit dem Abstand ab.
Im Folgenden wird auf der üblicheren logarithmischen Skala gearbeitet, sodass Energie nicht mehr in Milliwatt (mW), sondern in Dezibel Milliwatt (dBm) gemessen wird. Zusätzlich wird die Apertur und die Aussende-Energie in eine Referenzenergie in festem Abstand zusammengefasst und ein variabler Exponent α eingeführt, der alle nicht modellierten Effekte (Antennen-Gain, Ausbreitungspfad, Dämpfung, Interferenz etc.) einer Umgebung erfassen soll. Mit einem durch Messung bestimmten Referenzwert $E(d_0)$ im Abstand d_0 und diesem Parameter α erhält man das sogenannte *One-Slope-Modell*:

$$E(d, \alpha) = E(d_0) - 10\alpha \log_{10}\left(\frac{d}{d_0}\right)$$

Die Modellparameter α und $E(d_0)$ werden in der Regel aus Experimenten mit konkreter Hardware im konkreten Einsatzgebiet bestimmt.
Als Referenzentfernung wird typischerweise $d_0 = 1\text{m}$ genommen. Im Vakuum und ohne Störeinflüsse müsste α gemäß dem Ausbreitungsgesetz den Wert

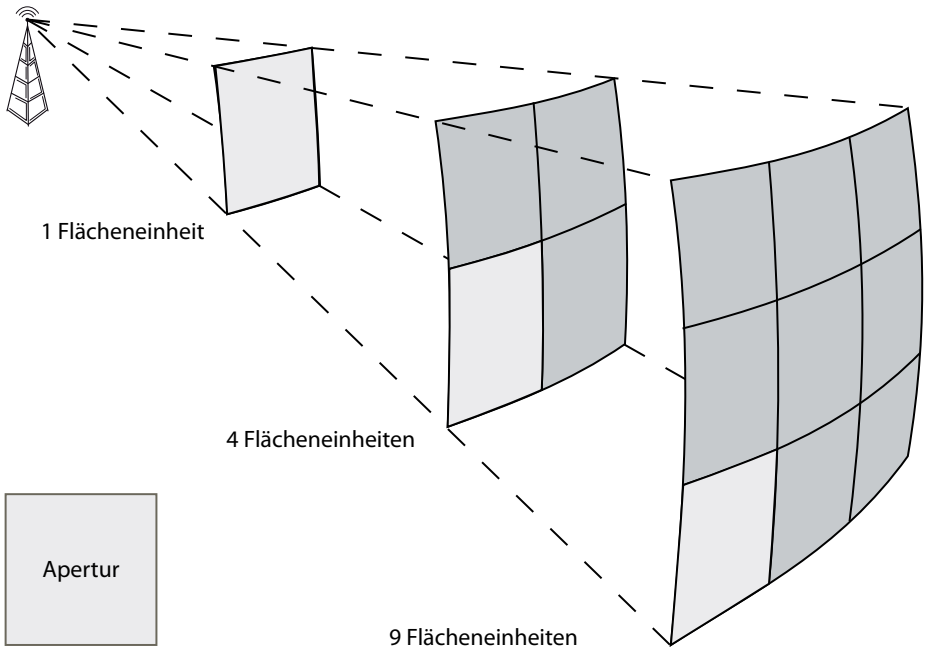


Abbildung 3.2: Ausbreitungsgesetz: Die Energiedichte nimmt quadratisch mit dem Abstand ab, weil sich die Energie auf der Kugeloberfläche verteilt

$\alpha = 2$ haben. Gemessene Werte für α in verschiedenen Umgebungen werden in [Rapp 02] zwischen 1.8 und 3 angegeben. Der Einfluss bestimmter Materialien auf die Empfangsleistung bei WLAN wird in [Zehn 05] analysiert.
Unter Verwendung einer Karte kann man dieses einfache Modell zu einem ziemlich leistungsstarken Modell erweitern. Man kann nämlich bei vielen Gebäuden davon ausgehen, dass die meisten Wände eine ähnliche Bausubstanz haben und daher die Durchdringungsdämpfung einer solchen Wand explizit modellieren. Dazu zählt man die Wände, die zwischen Sender und Empfänger durchtreten werden müssten, und addiert den als konstant angenommenen *Wall-Attenuation-Factor*. Diese Verfeinerung wurde zuerst von Motley und Keenan angegeben [Motl 88] und in der frühen Arbeit zur WLAN-basierten Indoor-Positionierung RADAR [Bahl 00] verwendet. Die verfeinerte Modellgleichung mit der Wanddämpfungskonstante W und der Anzahl der Wanddurchdringungen n lautet dann:

$$E(d, \alpha, n, W) = E(d_0) - 10\alpha \log_{10}\left(\frac{d}{d_0}\right) - nW$$

Ausgehend von diesen Basismodellen für die Signalausbreitung, die allesamt lediglich die direkte Sichtverbindung zwischen Sender und Empfänger betrachten, wurden einige Meta-Modelle vorgeschlagen, die auf die eine oder andere Weise den Ausbreitungsweg variieren lassen. Denn die WLAN-Funksignale haben sehr gute Eigenschaften in Bezug auf Mehrwegeausbreitung. Wenn man

eine Karte der Umgebung hat, in der beispielsweise Wände und komplexe Hindernisse eingetragen sind, so ist der Pfad, auf dem die meiste Leistung übertragen wird, häufig nicht die direkte Verbindung, die womöglich viele Wände durchdringen muss, sondern ein Pfad, welcher mit ein paar wenigen Reflektionen auf einer zwar etwas längeren Strecke mehr Energie transportiert als die direkte Verbindung. Das Auffinden solcher Pfade kann man zunächst mit einer Ray-Tracing-Methode unterstützen. Dazu muss modelliert werden, wie sich die Energie beim Auftreffen auf Wände etwa zwischen Durchdringung, ambienter Reflektion und spekularer Reflektion unterteilt. In dieser Situation unterscheidet man zwei Herangehensweisen: Man kann entweder Strahlen von allen Sendern (z.B. Lichtquellen) simulieren und mit der Umgebung interagieren lassen, oder umgekehrt nur die Strahlen simulieren, die an einem bestimmten Ort (z.B. im Auge) in eine bestimmte Richtung (z.B. die Blickrichtung) auftreffen. Letztere Variante hat natürlich den Vorteil, dass häufig weniger Strahlen simuliert werden müssen. Allerdings muss dann für jede Änderung des festen Ortes eine vollständige Neuberechnung erfolgen, während im anderen Fall eine vollständige Karte erstellt wird. Eine gute Übersicht über Ray-Tracing-Methoden in der Computergrafik gibt [Wald 09], für Anwendungen im Bereich der Funkausbreitung sei auf die Arbeit [Tran 08] verwiesen. Bei der Übertragung auf Funkausbreitung muss nur darauf geachtet werden, dass die Interaktionen sich bei längeren Wellen stark von den Interaktionen bei den kurzen Wellen im sichtbaren Spektrum unterscheiden können.
Solche Karten werden in diesem Zusammenhang häufig „Radio-Maps“ genannt. Beide Methoden sind natürlich sowohl in der Modellierung als auch in der Berechnung sehr komplex. Eine etwas weniger komplexe, vereinfachte Methode, die als Approximation einer Ray-Tracing-Methode aufgefasst werden kann, ist die Berechnung sogenannter dominanter Pfade [Wolf 05]. Ein dominanter Pfad ist dabei ein Ausbreitungspfad minimaler Dämpfung. Ist ein solcher Pfad ermittelt, verwendet man dessen Länge als Längenparameter für eines der obigen distanzbasierten Modelle.

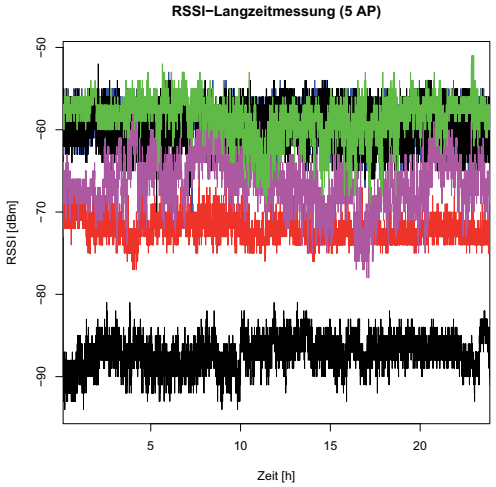
3.1.2 Analyse von Rauscheffekten

Die größte und wichtigste Fehlerquelle für die Berechnung von Positionen aus Messungen der Empfangssignalstärke ist das Rauschen. Die empfangene Signalstärke eines Funkübertragungssystems variiert nämlich auch dann recht stark über die Zeit, wenn die mobile Entität sich in Ruhe befindet. Dies hat verschiedene Ursachen [Stal 02, S.109 ff.]:

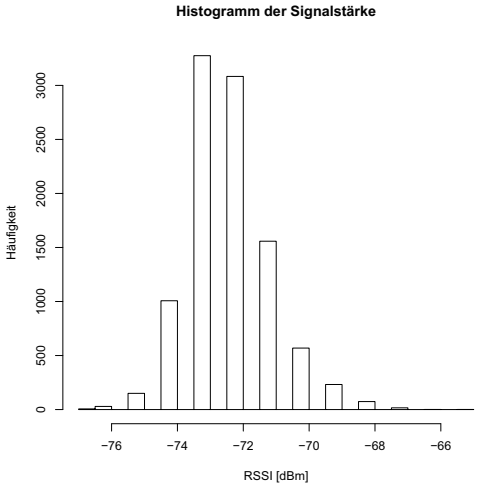
- Thermisches Rauschen: In jedem elektrischen Leiter entsteht weißes Rauschen allein durch die Bewegung von Elektronen.
- Intermodulations-Rauschen: Die modulierten Signale unterschiedlicher Funktechnologien auf unterschiedlichen Frequenzen können sich auf das verwendete Frequenzband auswirken.

- Impuls-Rauschen: Kurzzeitige Impulse teils hoher Energie, die aus Umwelteinflüssen herrühren (Blitz, Schalten von elektrischen Geräten, etc.).
- Umgebungs-Rauschen: Durch Veränderungen der Umgebung (Bewegungen von Menschen und Gegenständen) ändert sich das Ausbreitungs- und Interferenzverhalten und damit auch die empfangene Signalstärke an einem festen Ort.

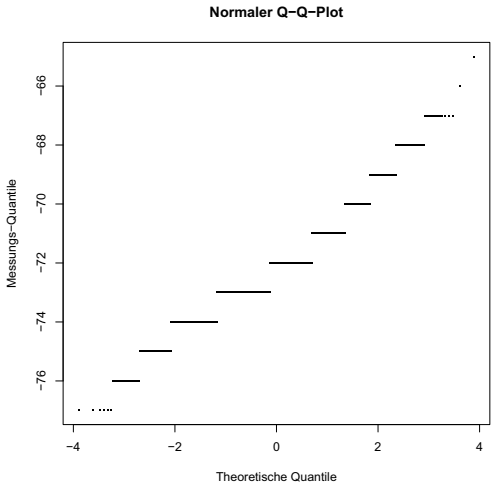
Abbildung 3.3(a) zeigt eine Langzeitmessung der WLAN-Signalstärke an einem festen Ort in einem Bürogebäude. Die Messung wurde 24 Stunden mit einem handelsüblichen Smartphone (HTC Desire) durchgeführt. Das Histogramm 3.3(b) zeigt die Verteilung der Messwerte, die zunächst wie eine Gauss-Verteilung aussieht. Einen weiteren Hinweis darauf, dass es sich womöglich um eine Gauss-Verteilung handelt, liefert der Quantile-Quantile-Plot 3.3(c), in dem bis auf die durch Rundung entstehenden waagerechten Abschnitte eine lineare Steigung vorzuliegen scheint. Für den formalen Nachweis einer Normalverteilung wurde der Shapiro-Wilk-Test [Shap 65] verwendet. Da die Teststärke dieses Testes aber stark beeinflusst wird, wenn Rundungen zu mehreren gleichen Werten führen, wurde dieser Effekt mit der Sheppard-Korrektur für den Varianz-Schätzer im Shapiro-Wilk-Test ausgeglichen [Roys 89]. Insgesamt ist es zulässig, das Rauschen als normalverteilt anzusehen, auch wenn der formale Wert nur knapp erreicht wurde ($n = 50$, $W_g \approx 0.957 > W_{\text{krit}} = 0.947$, Signifikanzniveau $\alpha = 0.05$).
Eine weitere Eigenschaft, um Rauschen zu charakterisieren, ist das Verhalten der spektralen Leistung. Dazu zeigt Abbildung 3.3(d) das Integral der Fouriertransformation der Messwerte und es ist erkennbar, dass die Rauschleistungsdichte (Steigung des Graphen) nahezu kostant ist. Es handelt sich also tatsächlich um *Weißes Gauss-Rauschen* (White Gaussian Noise, WGN).
Diese Charakterisierung von Rauschen bezieht sich natürlich nicht auf die Funkkanäle, sondern lediglich auf die Messung des Indikators für die Empfangssignalstärke. In der Literatur findet man für die Charakterisierung der Funkkanäle selber häufig andere Wahrscheinlichkeitsverteilungen. Die bekanntesten Rausch-Modelle, die so ihre Verwendung finden, sind *Riccian-Rauschen*, das gerade in Mehrwege-Situationen die Überlagerung eines starken (Sichtlinie) und schwachen (Mehrwegeausbreitung) Signals modelliert. Ein Spezialfall ist das *Rayleigh-Rauschen*, welches insbesondere dann gute Ergebnisse liefert, wenn es keine Sichtverbindung zwischen Sender und Empfänger gibt. Diese Rauschverteilungen sollten grundsätzlich verwendet werden, wenn man ein Kommunikationssystem für den Innenraum entwirft. Durch sehr große Mess- und Diskretisierungsfehler bei Messungen mit einem Smartphone ergab sich aber dennoch in der Beobachtung eine Gauss-Verteilung.
Es ist wegen der Betrachtungen gerechtfertigt, in einer Simulationsumgebung Weißes Gauss-Rauschen zu verwenden, um die Störung der Signalstärke zu modellieren. Denn auf der einen Seite wird tatsächlich (nahezu) Weißes Gauss-Rauschen in einem Ruheexperiment beobachtet, und auf der anderen Seite



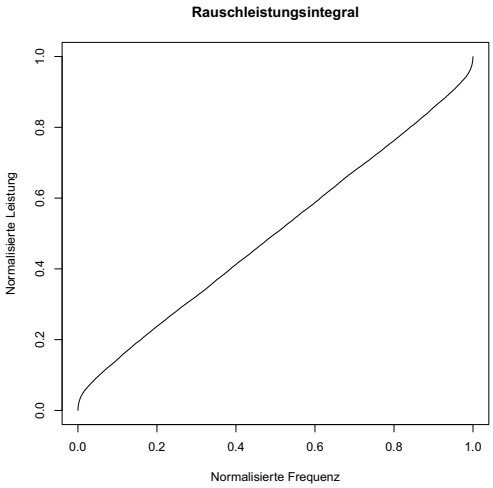
(a) Signalstärkeentwicklung über die Zeit



(b) Histogramm der Signalstärken



(c) Quantile-Quantile-Plot der Messung



(d) Rauschleistungsintegral (normalisiert)

Abbildung 3.3: Langzeitmessung von RSSI-Werten in Ruhe

enthält Weißes Gauss-Rauschen keine zusätzliche Information, die die Qualität einer probabilistischen Ortsbestimmung beeinflussen könnte, zumindest wenn die Rauschleistungsdichte in den Simulationsläufen variiert.

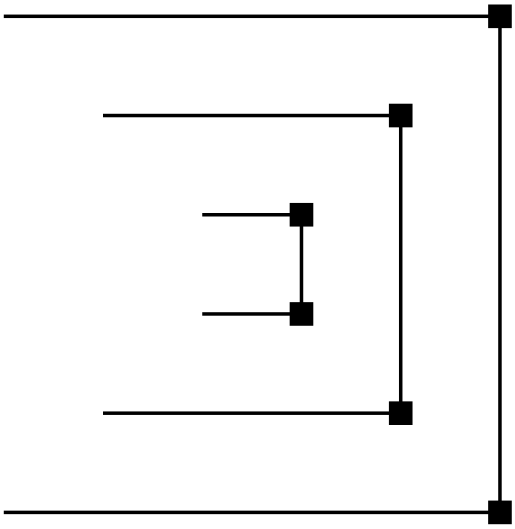
3.1.3 Simulationsumgebung in Octave

Das Motely-Keenan Modell für den Zusammenhang zwischen Entfernungen, Grundrissen und der Empfangssignalstärke eigent sich sehr gut, um verschiedene Indoorpositionierungsalgorithmen miteinander zu vergleichen. Eine solche Simulationsumgebung kann zwar keine Feldtests ersetzen, hat aber wegen

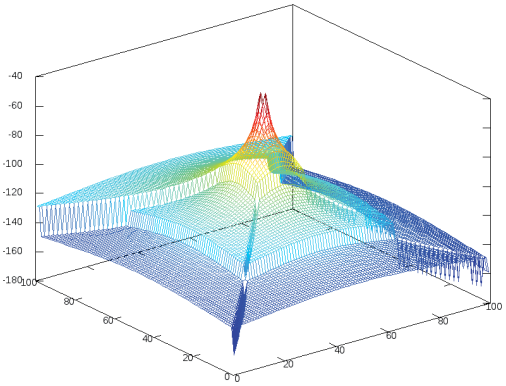
der in der Praxis unbegrenzten Anzahl an Experimenten, die automatisiert ablaufen können, einen entscheidenden Vorteil im Vergleich und in der Optimierung von Indoorpositionierungsalgorithmen. So können die Abhängigkeiten von Navigationsfläche, Rauschleistung, Messgenauigkeit und Ortskonfiguration der Access-Points untersucht werden.
Die Konzeption der Simulationsumgebung ist so, dass die „Experimente“ klar von der Evaluierung getrennt sind. Man kann sowohl die Daten eines theoretischen Ausbreitungsmodelles auswerten, als auch Daten analysieren, die mit einem Smartphone gesammelt worden sind. Dabei kommt die Software Octave [Eato 02] als technische Basis zum Einsatz, da sie es ermöglicht, auch komplexe mathematische und statistische Zusammenhänge und Algorithmen elegant zu formulieren. In weiten Teilen ist Octave dabei zu der kommerziellen Software MATLAB kompatibel, die sich in den letzten Jahren für Anwendungen mit Bezug zu numerischer Mathematik durchgesetzt hat [Math 12].

3.1.3.1 Raumpläne in der numerischen Simulation

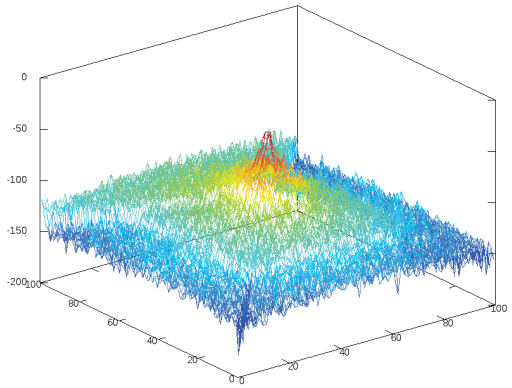
Für die Anwendung des Motley-Keenan Modells benötigt man die Möglichkeit, die Anzahl (und ggfs. auch die Art) an Wänden zwischen zwei Punkten im Simulationsbereich zu bestimmen. Deshalb wurde ein Octave-Plugin entwickelt, das auf der offen zugänglichen DXF-Bibliothek dxflib basiert, und beliebige CAD-Pläne im offenen DXF-Format in die Simulationsumgebung laden kann [Ribb 12]. Um eine einheitliche Behandlung der CAD-Daten zu gewährleisten, werden alle Primitiven in Folgen von Liniensegmenten zerlegt. Damit entsteht für eine CAD-Datei eine $n \times 4$-Matrix, wobei jede Zeile einer CAD-Linie mit Anfangs- und Endpunkt entspricht. Komplexe CAD-Entitäten wie Kreise, Bögen und Polyline-Bulges werden dabei durch Sequenzen von Liniensegmenten angenähert.
Diese Darstellung ist sehr gut geeignet, um eine Signalkarte auf Basis des Motley-Keenan-Modells zu berechnen. Abbildung 3.4 zeigt ein Motley-Keenan Modell mit und ohne Weißes Gauss-Rauschen, welches auf dem illustrativen „Raumplan“ in Abbildung 3.4(a) basiert.
Abbildung 3.5(a) zeigt, wie komplex die Indoor-Ausbreitung von Signalstärken mit dem Motley-Keenan-Modell wird, wenn man einen echten Raumplan verwendet. Dabei wurde diesem Modell kein Rauschen hinzugefügt.
Dieses Ausbreitungsmodell und die Erzeugung von Weißem Gauß-Rauschen erlaubt uns, Positionierungsalgorithmen, die auf Signalstärkeverteilungen basieren, zu simulieren, und auf diese Weise die Abhängigkeiten zwischen Position, Rauschleistung, Anzahl der Access-Points, Verteilung der Access-Points und ähnlichem zu untersuchen. In dieser Umgebung wurde unter anderem die wichtigste Positionierungsgrundtechnik kNN implementiert, die auch von RADAR [Bahl 00] und vielen nachfolgenden Arbeiten verwendet wird.
So kann man mit dem vorliegenden Simulator die Ausbreitung von Access-Points simulieren und Fingerprints erzeugen. Dazu werden beispielsweise in einem regulären Gitter von einer vorgegebenen Kantenlänge die Signalstärken



(a) Illustrativer Raumplan



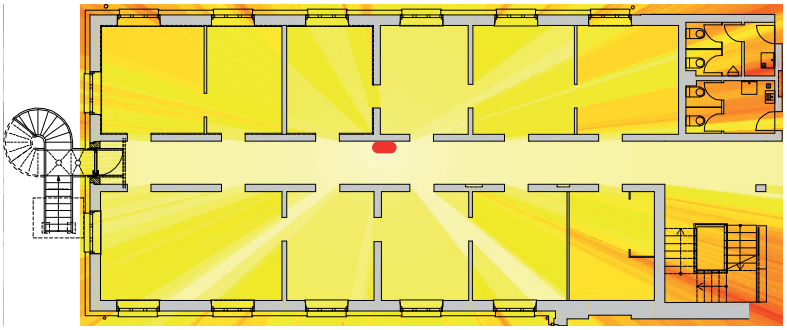
(b) Modell ohne Rauschen



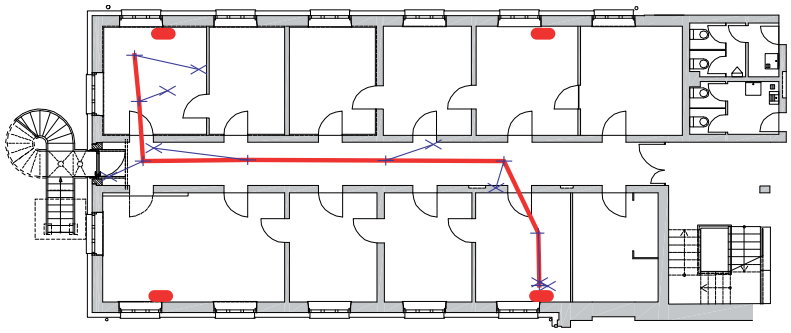
(c) Modell mit Rauschen

Abbildung 3.4: Ein illustrativer „Raumplan“ und das Motley-Keenan-Ausbreitungsmodell in IndoorSIM mit und ohne Rauschen

berechnet und mit Weißem Rauschen überlagert. So entstehen Daten, wie sie typischerweise auch in der Realität entstehen, wenn man in einem Gebäude an gewählten Referenzpunkten längere Messungen unternimmt. Auf der anderen Seite ist es auch möglich, Signalstärke-Verteilungen für verschiedene Räume, die in CAD beispielsweise als geschlossene Linien auf einem speziellen Layer spezifiziert wurden, zu berechnen und mit einem Label zu versehen, um so probabilistische Positionierungsmethoden auf Raumebene in dieser Umgebung zu entwickeln und zu testen. Dabei kann man Trainings- und Testdaten direkt im Dateiformat „ARFF“ erzeugen, die dann vom weit verbreiteten Weka-Toolkit für maschinelles Lernen [Hall 09] verarbeitet werden können. Es ist auch möglich, einen Track in CAD zu zeichnen und das Ablaufen dieses Tracks unter Vorgabe eines statistischen Modells für die Abweichungen, etwa eine zweidimensionale Gauss-Verteilung vorgegebener Abweichung, zu simulieren und



(a) Motley-Keenan-Ausbreitung in einem Bürogebäude



(b) Gewichtete k-Nächste-Nachbarn ($k = 3$) auf einem Raumplan in IndoorSIM

Abbildung 3.5: Heatmap und kNN-Visualisierung mit IndoorSIM

Signalstärke-Sequenzen für Tracking-Szenarien zu erzeugen. Dazu wurde noch ein Visualisierungs-Layer implementiert, mit dem man insbesondere den Unterschied zwischen realer Position und errechneter Position visualisieren kann. Dabei kann der Einfluss von Rauschen in der Zeitdomäne als Videosequenz gezeichnet werden. Abbildung 3.5(b) zeigt beispielhaft, wie solche Abweichungen im Simulator IndoorSIM dargestellt werden.

3.2 Einfluss der Blickrichtung bei WLAN-Fingerprinting

Für die Methoden der Szenenanalyse aus Abschnitt 2.1.7 können grundsätzlich alle Daten verwendet werden, die durch die mobile Entität oder die Infrastruktur messbar sind und sich mit dem Ort verändern. In den letzten Jahren hat es sich bei Smartphones durchgesetzt, diese mit einer gewissen Menge an inertialen Sensoren auszustatten, um auch Anwendungen der „Augmented Reality“ zu ermöglichen. So verfügen fast alle aktuellen Smartphones über Beschleunigungssensoren und einen digitalen Kompass.
Da ist es naheliegend, sich mit der Frage zu beschäftigen, ob diese zusätzliche Sensorik Vorteile für bestehende Positionierungsmethoden oder Mög-

lichkeiten zur Erweiterung derselben bietet. In diesem Zusammenhang wurde ein WLAN-basiertes Positionierungs- und Trackingsystem namens SMART-POS [Kess 11b] entwickelt, das als eine Erweiterung von klassischem kNN-Fingerprinting angesehen werden kann, und durch die verschiedenartige Verwendung von Kompassinformationen zu besseren Ergebnissen führt.

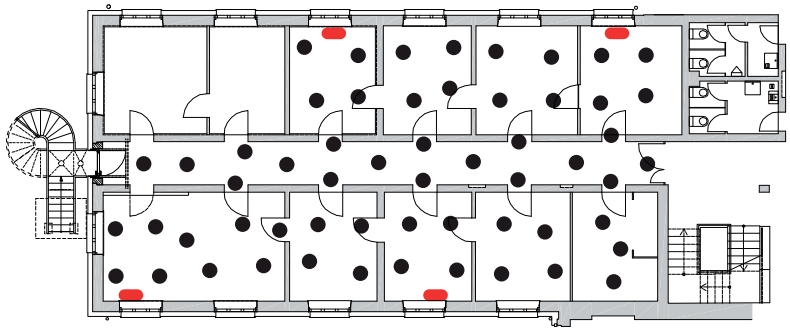
Von der Arbeit [King 06] ausgehend, durfte man erwarten, dass die natürlichen Abschirmungseigenschaften des menschlichen Körpers einen so starken Einfluss auf die gemessenen Signalstärken haben, dass durch die Verwendung der Orientierung eine Verbesserung entsteht. Über die genannte Arbeit gehen die vorliegenden Ergebnisse aber hinaus, da sowohl die Auswirkungen auf probabilistische Verfahren, als auch die Verwendung der Orientierung für integrierte Tracking-Anwendungen analysiert wurde.

In der Positionierung mit dem gewichteten kNN-Algorithmus wird eine höhere Genauigkeit bei geringerem Rechenaufwand im Wesentlichen dadurch erreicht, dass aus der Datenbank nur Fingerprints mit einer zur aktuellen Messung ähnlichen Orientierung verwendet werden.

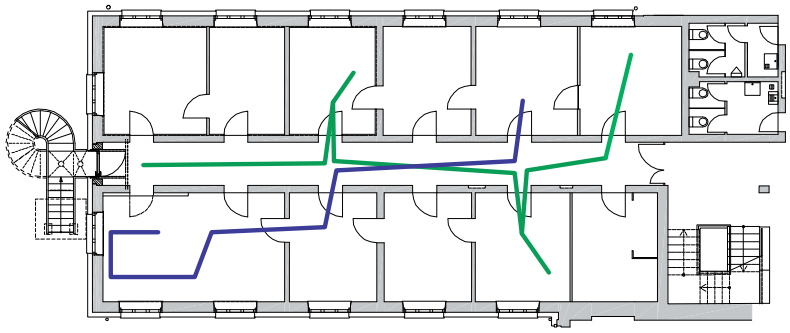
Ein sehr einfacher, probabilistischer Ansatz, die Aufenthaltswahrscheinlichkeit in Räumen mit einem naiven Bayes-Klassifikator zu errechnen, profitiert aber leider nicht von einer Filterung. Eine direkte Verwendung der Winkel ist nicht sinnvoll, da sie keine Korrelation mit dem Ort aufweisen, und daher der Winkel selbst nichts über den Ort aussagen kann. Allerdings sagt der Winkel sehr wohl etwas über die Situation der Messung in Bezug auf die elektromagnetische Abschattung des menschlichen Körpers aus. Im Falle einer Filterung der Winkel ähnlich zum kNN-Algorithmus überwiegt aber leider die zunehmende Unsicherheit bei der Bestimmung der Mittelwerte und Abweichungen einen möglichen Verbesserungseffekt, sodass insgesamt die Qualität der Positionierung nachlässt.

Für Tracking-Anwendungen konnte die Performance des kNN-Algorithmus dadurch gesteigert werden, dass der Abstand der Vergleichs-Fingerprints mit der letzten Ortsschätzung in die Auswahl der Nachbarn im kNN-Algorithmus integriert wurde. Eine weitere Verbesserung liefert ein einfaches Bewegungsmodell, welches auf einer konstanten Geschwindigkeit die gemessene Orientierung „abfährt“, und ähnlich einem Kalman-Filter zwischen der aktuellen Messung und der Vorhersage aus der vergangenen Messung einen Ausgleich findet. Ein Schrittzähler auf Basis der Smartphone-Beschleunigungssensoren ermöglicht eine weitere Verbesserung in diesen Tracking-Szenarien.

Ein wichtiger Unterschied zwischen dem vorliegenden System und anderen Systemen in der Literatur ist die Tatsache, dass alle Berechnungen auf einem handelsüblichen Smartphone gemacht werden. Insbesondere ändert sich dadurch die Rangfolge der Optimierungsgrößen: Für eine Server-basierte, mobile Navigationsanwendung ist typischerweise das Übertragungsvolumen wichtiger als der Rechenaufwand, weil es einen direkten Zusammenhang zwischen Energieverbrauch der mobilen Entität und Übertragungsvolumen gibt. Für eine autarke Positionierung des mobilen Geräts ist jedoch primär der Rechenaufwand



(a) Referenzpositionen (Kreise) und Access-Points (Rechtecke)



(b) Zwei Tracks im Abdeckungsgebiet des Positionierungssystems

Abbildung 3.6: Testdaten für SMARTPOS

für den Energieverbrauch verantwortlich, und sollte daher minimiert werden. Das System SMARTPOS verwendet aktive WLAN Scans und die Signalstärke-Indikatoren der Beacon-Frames, mit denen die erreichbaren Access-Points auf diese Scan-Anfrage reagieren. Gleichzeitig zeichnet das System die Orientierung des Smartphones und die Beschleunigungswerte in alle drei Hauptachsen des Gerätes auf.

In einer „Offline-Phase“ werden an vorher festgelegten, in einem Gitter angeordneten Referenzpositionen, jeweils mehrere solcher Messungen in die vier Hauptrichtungen des Gebäudes aufgenommen, und der Mittelwert der gemessenen Signalstärken gespeichert. So kann der Einfluss von starken, kurzzeitigen Fluktuationen in der WLAN-Ausbreitung effektiv reduziert werden. Es werden noch einige über das Gitter hinausgehende Kontrollpunkte aufgenommen, um in der Evaluierung als Testdaten zur Verfügung zu stehen, ohne dass sich der durchschnittliche Abstand der Referenzpositionen verändert. In einer „Online-Phase“ werden die genannten Daten mit Zeitstempeln versehen und für zwei vorgegebene Tracks aufgezeichnet. Abbildung 3.6 zeigt die Datenbank, die Testpunkte und die Tracks, wie sie in einem Gebäudeteil eines Universitätsgebäudes aufgenommen wurden.

In den folgenden Abschnitten werden zunächst genauer die Algorithmen beschrieben, bevor im Abschnitt 3.2.4 die Ergebnisse der Vergleiche und Konfigurationen der verschiedenen Algorithmen vorgestellt werden.

3.2.1 Positionierung mit dem gewichteten kNN-Algorithmus

Zunächst wird als Basis-Technologie ein gewichteter kNN-Algorithmus, wie im Folgenden beschrieben, auf verschiedenen Datenbanken verwendet.
Dabei wird eine Messung m und eine Orientierung o eines Messpunktes in den beiden folgenden Weisen verwendet: Zum Einen wird die aktuelle Orientierung o ignoriert und kNN im Signalraum mit der Messung m durchgeführt. Zum Anderen wird mit der Orientierung o eine Auswahl von Fingerprints aus der Datenbank getroffen, die von dieser aktuellen Orientierung um nicht mehr als 50° abweicht, und auf dieser Datenbank mit dem kNN-Algorithmus eine Position bestimmt.
Der Hintergrund für diese Art der Berücksichtigung der Orientierung liegt in der Vermutung, dass die Dämpfung des menschlichen Körpers einen starken Einfluss auf die gemessene Signalstärke hat und dass diese Dämpfung mit der Orientierung des Smartphones und damit mit der groben Sichtrichtung korreliert.
Diese Behandlung führt neben Verbesserungen in der Positionierungsqualität auch zu einer Reduktion der Datenbankgröße auf etwa 27% der ursprünglichen Datenbank und damit zu einer Reduktion in der benötigten Rechenleistung und Energie.
Für die Verwendung eines kNN-Algorithmus mit der Messung m und den möglicherweise gefilterten Fingerprints $f_i \in S$ aus der Datenbank S, muss der Abstand im Signalraum zwischen der Messung m und einem Fingerprint f_i definiert werden. Hierbei kann es vorkommen, dass in der Messung m und dem Fingerprint f_i nicht nur dieselben Access-Points vorkommen. In diesem Fall werden die Access-Points, die nur in einer der beiden Messungen vorkommen, verworfen. Danach wird die euklidische Distanz $d_i = \mathrm{dist}(m, f_i)$ zwischen den Vektoren mit den Signalstärken der verbliebenen Access-Points verwendet. Auf dieser Distanz basierend, wird die Teilmenge $N \subset S$ der k nächsten Nachbarn berechnet.
Zusätzlich kommt noch eine Gewichtung ω_i für jeden Fingerprint $f_i \in N, i \in \{1, \ldots, k\}$ zum Einsatz, wie in der folgenden Formel definiert:

$$\omega_i = \left(d_i \sum_{j=1}^{k} \frac{1}{d_j} \right)^{-1} \tag{3.1}$$

Diese Definition von Gewichtungen ist bereits normal im Sinne, dass die ω_i eine additive Zerlegung der 1 darstellen:

$$\sum_{i=1}^{k} \omega_i = 1$$

Der Ort der mobilen Entität l ergibt sich in SMARTPOS mit diesen Vorberei-

tungen als ω-gewichtete Summe der Orte l_i der Fingerprints in der Teilmenge N:

$$l = \sum_{i=1}^{k} \omega_i l_i \tag{3.2}$$

Die Ergebnisse für diese Form der Positionierung finden sich in Abschnitt 3.2.4.

3.2.2 Positionierung mit einem naiven Bayes-Klassifikator

Um in der Situation des Systems SMARTPOS einen Vergleich mit einem probabilistischen Verfahren zu ermöglichen, werden die Fingerprints der Datenbank mit einem Raum-Label versehen. So entsteht eine große Tabelle, in der die Access-Points und der Raum als Spalten auftreten während in den Zellen die Signalstärken bzw. die Raum-Label stehen. Nun kann mit einem naiven Bayes-Klassifizierer aus Abschnitt 2.1.7 die Aufenthaltswahrscheinlichkeit in einem Raum berechnet werden. Dabei werden zunächst pro Raum alle vorliegenden Fingerprints ohne Berücksichtigung der Orientierung zusammengefasst und die aktuelle Messung mit der Regel von Bayes und der Naivitäts-Annahme, dass die Signalstärken der verschiedenen Access-Points statistisch unabhängig sind, unter Verwendung der vorher bestimmten relativen Häufigkeiten zu Aufenthaltswahrscheinlichkeiten für die Räume weiterverarbeitet. Das Ergebnis dieser Positionierungsmethode ist der Raum mit höchster Aufenthaltswahrscheinlichkeit, sodass in diesem Falle Erfolgsraten gezählt werden können.
Zum Vergleich werden auch hier in einer zweiten Variante nur die Fingerprints der Datenbank verwendet, die eine ähnliche Orientierung aufweisen wie das Smartphone.

3.2.3 Kontinuierliche Positionierung und Tracking

In Anwendungsfällen des Trackings ist es natürlich sinnvoll, auch die Zeit-Domäne mit in die Positionierung einzubeziehen. Dazu werden zwei unterschiedliche Modelle verwendet: Eines, um aus einer Messung einen Ort zu bestimmen, und eines, um aus einer vorhergehenden Messung eine neue Position fortzuschreiben. Diese zwei Positionen müssen dann miteinander gewinnbringend kombiniert werden.
Klassischerweise kommt in dieser Situation ein Kalman-Filter oder ein Partikelfilter zum Einsatz (siehe auch Abschnitt 2.2 und [Even 06]). Dabei approximiert ein Kalman-Filter ein dynamisches, lineares System mit gestörten Messungen, während ein Partikelfilter eine diskrete Approximation einer Wahrscheinlichkeitsverteilung berechnet. Im Allgemeinen sind Partikelfilter sehr aufwändig, weil für eine möglichst große Anzahl an Partikeln eine Berechnung durchgeführt werden muss. Deshalb eignen sie sich eher nicht für Tracking-Anwendungen auf Smartphones, da weder genug Rechenleistung noch genug Energie für diesen Ansatz dauerhaft zur Verfügung stehen.

In der Indoor-Positionierung kommt ein Kalman-Filter typischerweise als gewichtete Linearkombination einer gemessenen Position $p_{\text{mes},i}$ und einer auf Grund eines vorherigen Zustandes und eines Bewegungsmodells fortgeschriebenen Position $p_{\text{pre},i-1}$ zum Einsatz.

$$p_{i|i-1} = p_{\text{mes},i}(1-\gamma) + p_{\text{pre},i-1}\gamma \tag{3.3}$$

Zu dieser klassischen Filterung wurde eine Erweiterung vorgeschlagen, die es ermöglicht, schon bei der Auswahl der Nachbarn im kNN-Algorithmus die Vorhersage der Position aus dem vorherigen Zustand zu verwenden. Dazu wurde bereits bei der Auswahl der nächsten Nachbarn im Signalraum die Abweichung der Position der Kandidaten-Fingerprints von der derzeitigen Ortsschätzung des Bewegungsmodells verwendet.

3.2.3.1 Modell für die Messung

Bei diesem Messmodell handelt es sich um den klassischen k-Nächste-Nachbarn-Algorithmus – wie in Abschnitt 3.2.1 beschrieben – mit der folgenden Verfeinerung: Sei d_i wie zuvor die euklidische Distanz zwischen einem bereinigten Paar aus einer Messung m und einem Fingerprint f_i. Sei weiter s_i die Kartendistanz zwischen der mit dem Bewegungsmodell vorhergesagten, aktuellen Position und der Position des Fingerprints f_i in der Datenbank. Dann sei die Distanz e_i für die Auswahl der k nächsten Nachbarn als Interpolation dieser Werte mit einem festen Interpolationskoeffizient α gegeben:

$$e_i = (1-\alpha)d_i + \alpha s_i$$

Der Wert für den Interpolationskoeffizienten α kann empirisch ermittelt werden und vermittelt zwischen der Tendenz von kNN zu weiten Sprüngen und der inertialen Fortschreibung der letzten bekannten Position.

3.2.3.2 Modell für die Vorhersage

Als Vorhersagemodelle wurden zwei Bewegungsmodelle verwendet. Das erste Bewegungsmodell ist ein sehr einfaches mit einem Geschwindigkeitsvektor $\vec{v}$ fester Länge

$$p_{pre,i-1} = p_{mes,i-1} + \vec{v}t,$$

wobei die Richtung des Geschwindigkeitsvektors durch die Orientierung des Smartphones gegeben ist.
Als zweites Bewegungsmodell kommt ein Schrittzähler zum Einsatz, der ähnlich zu der Arbeit von Link [Link 11] funktioniert und besondere Situationen in der vertikalen Beschleunigung detektiert:

$$p_{pre,i-1} = p_{mes,i-1} + \sum_{i=1}^{k} \vec{l_i},$$

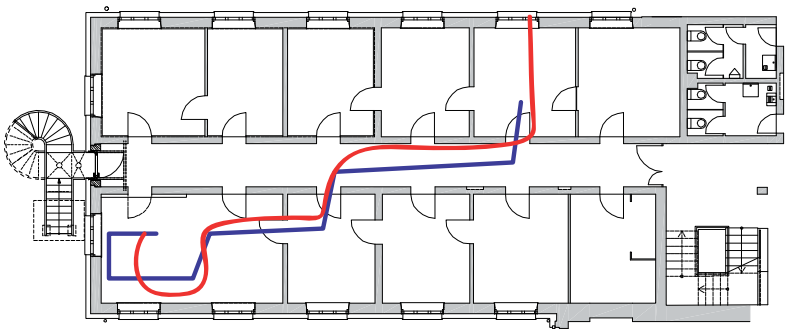


Abbildung 3.7: Der echte (blau) und der mit bekannter Geschwindigkeit und Kompassinformationen rekonstruierte Track (rot)

wobei in der Summe je ein Summand für jeden Schritt vorkommt, die Länge der Vektoren $\vec{l_i}$ eine fest vorgegebene Schrittlänge ist, und die Richtung der Vektoren $\vec{l_i}$ abermals von der Orientierung des Smartphones zum Zeitpunkt der Schritterkennung vorgegeben wird.

3.2.4 Evaluation und Ergebnisse

Um die verschiedenen Ansätze und Situationen des vorliegenden Positionierungs- und Tracking-System zu evaluieren, wurden Messdaten in einem Flügel eines Universitätsgebäudes gewonnen. Dazu wurden, wie bereits beschrieben, in Richtung der vier Hauptachsen des Gebäudeflügels Messungen an den 57 Referenzpositionen aus Abbildung 3.6(a) vorgenommen. So ergeben sich insgesamt 228 Fingerprints. Zusätzlich wurden die zwei Tracks aus Abbildung 3.6(b) in einer kontinuierlichen Messung aufgezeichnet. Das verwendete Smartphone HTC Desire erreichte dabei eine Sample-Rate für WLAN-Scans von etwa 1Hz und zeichnete Kompass- und Beschleunigungsdaten mit etwa 5Hz auf.
Im Rahmen dieser Evaluation wird der durchschnittliche Positionierungsfehler als Indikator für die Genauigkeit und die Standardabweichung als Indikator für die Präzision verwendet. Bei Tracking-Szenarien kommen auch der akkumulierte Fehler am Ende des Tracks und der prozentuale Fehler als Indikator für die Fehleranhäufung über die Zeit zum Einsatz.

3.2.4.1 Qualität der Orientierungsinformationen und der Schrittzählung

Die mit dem im Smartphone verbauten digitalen Kompass ermittelte Orientierung ist sehr gut. So konnte im Falle eines mit konstanter Geschwindigkeit aufgenommenen Tracks ein durchschnittlicher Fehler von 0.82m mit einer Standardabweichung von 0.55m erreicht werden, wobei die empirische ermittelte Geschwindigkeit von 0.97m/s mit den Messungen des Kompasses kombiniert wurde. Dieses überraschend gute Ergebnis ist auch in Abbildung 3.7 visuali-

siert und zeigt, dass die Qualität der Orientierungsinformation hervorragend ist. Der akkumulierte Fehler beträgt mit 2.96m bei einer Gesamtlänge von 27m etwa 11%.
Auch die Verwendung eines Schrittzählers führte zu guten Ergebnissen und unterstützt die These, dass die Bestimmung der Länge der zurückgelegten Strecke eines Fußgängers allein mit einem Schrittzähler recht gut möglich ist. Der akkumulierte Fehler liegt für beide Tracks bei unter 6%.

Versuch	**Ø-Fehler**	**Standardabw.**	**Akkum. Fehler**
T2 mit konstanter Geschwindigkeit	0.82m	0.55m	2.96m (11%)
T1 mit Schrittzähler	1.2m	0.77m	1.9m (5%)
T2 mit Schrittzähler	1.1m	0.61m	0.85m (3%)

Tabelle 3.1: Qualität der Kompassinformationen

3.2.4.2 Einfluss einer Filterung auf den gewichteten kNN-Algorithmus

Bei der Untersuchung der gewichteten kNN-Algorithmen mit und ohne Filterung der Datenbank wurde zunächst der für die unorientierte Positionierung optimale Wert von $k = 3$ fest gewählt. Im gefilterten Fall wurden bei der Auswahl der Nachbarn nur die Fingerprints berücksichtigt, die maximal um 50° von der aktuell gemessenen Orientierung des Smartphones abweichen. Zusammenfassend ergibt sich, dass die Verwendung der Orientierung als Filter die Qualität verbessert. Man erkennt in Tabelle 3.2, dass die Verwendung der Orientierung als Filter sowohl den mittleren Fehler um etwa 10%, als auch dessen Standardabweichung um etwa 33% signifikant senkt.

Versuch	**Ø-Fehler**	**Std.-abw.**
T1 ohne Orientierung	2.01m	1.52m
T1 mit Orientierung	1.72m	1.02m
T2 ohne Orientierung	1.74m	1.00m
T2 mit Orientierung	1.52m	0.67m

Tabelle 3.2: Verbesserung der Positionierung bei gewichtetem kNN durch Filterung der Orientierung

3.2.4.3 Bayes-Modell auf Raumebene

Um ein besseres Verständnis von den Effekten der Reduktion der Datenbank zu bekommen, wurde auch ein anderes, klassisches Verfahren implementiert, mit dem der Ort einer mobilen Entität bestimmt werden kann. Es handelt sich dabei, wie beschrieben, um einen Bayes'schen Ansatz, bei dem die Messwerte

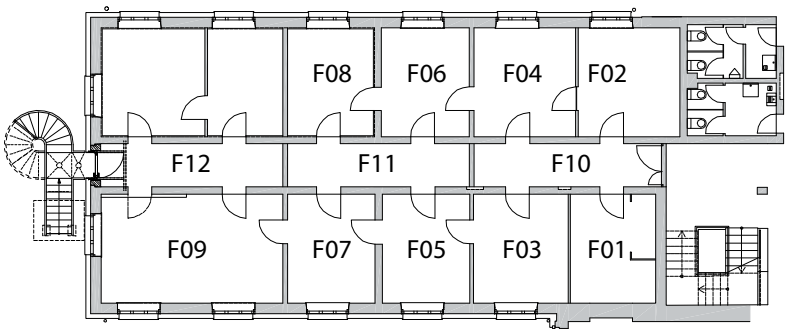


Abbildung 3.8: Raumaufteilung für den naiven Bayes-Algorithmus

pro Raum mit Mittelwert und Standardabweichung aggregiert werden und dieses Wissen dann zur Bestimmung des Raumes einer neuen Messung verwendet werden kann.
Der lange Korridor wurde dabei in drei Räume eingeteilt, sodass eine Raumaufteilung wie in Abbildung 3.8 ensteht. Aus den vorliegenden Fingerprints wurden fünf Modelle erstellt: Ein Modell, das alle Richtungen umfasst, und je ein Modell für jede Hauptrichtung des Gebäudes. Das Modell, das alle Richtungen umfasst, wurde dabei auf zufällig gewählten 25% der Daten errechnet, um dem globalen Modell keinen Vorteil durch eine signifikant größere Datenmenge zukommen zu lassen.
Es wurde eine zehnfache, stratifizierte Kreuz-Validierung auf den im regulären Gitter aufgenommenen Messpunkten verwendet. Dabei wurden in zehn Versuchen jeweils 90% der Daten für das Training und 10% für die Bestimmung von Erfolgsraten verwendet. Die Einschränkung auf die vier Hauptachsen reicht aus, da alle Fingerprints für Training und Evaluation nur in diesen Richtungen vorliegen.
Die Ergebnisse sind negativ: Dieser Ansatz kann leider nicht von der Orientierungsfilterung profitieren. Die Erfolgsraten sind in Abbildung 3.9 visualisiert.

3.2.4.4 Integration der Kartenentfernung in kNN-Gewichtung bei Tracking-Anwendungen

In der Testumgebung wurden die Modellparameter für das Messmodell und für die Integration der Kartenentfernung in die kNN-Gewichtung empirisch bestimmt. Bei einer Geschwindigkeit von 1ms^{-1} ergab eine Kopplung mit einem Wert von $\alpha = 0.7$ die besten Ergebnisse. Der durchschnittliche Fehler verbesserte sich für beide Tracks um weitere 30%-50%. Damit beträgt die Gesamtverbesserung des durchschnittlichen Positionierungsfehlers mit dieser Methode gegenüber der unorientierten Methode 50%-70%.
Dies bedeutet sicher nicht, dass man ohne Weiteres eine Leistungssteigerung in dieser Größenordnung erwarten kann, wenn man die Kartenentfernung in die kNN-Gewichtung mit aufnimmt. Denn die Position des Nutzers und damit insbesondere die zu verwendende Kartenentfernung ist nicht bekannt. Damit

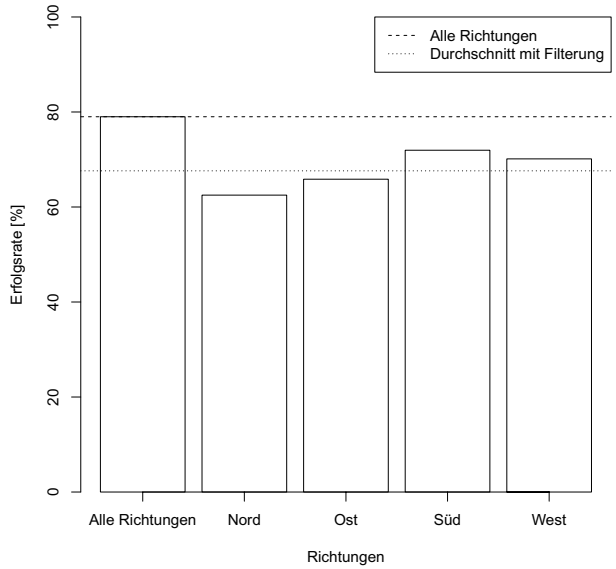


Abbildung 3.9: Ergebnisse der Bayes'schen Positionierung

hängt der Nutzen dieser Vorgehensweise davon ab, wie gut die Position des Nutzers zwischen einzelnen kNN-Positionierungen fortgeschrieben wird, und damit von der Kenntnis der Geschwindigkeit der Nutzer und der Richtung der Fortbewegung. Für die Richtung wurde mit dem digitalen Kompass bereits ein sehr gutes Ergebnis erzielt, der folgende Abschnitt wendet sich Bewegungsmodellen zu, die aus den vorliegenden Messdaten Auskunft über die Geschwindigkeit geben.

3.2.4.5 Verwendung von Bewegungsmodellen

Zunächst könnte man annehmen, dass der exakte Wert der fest vorgegebenen Geschwindigkeit des Nutzers einen sehr großen Einfluss auf die Positionierung mit kNN und Feedback hat. Glücklicherweise ist das nicht der Fall. Verwendet man Geschwindigkeiten aus dem Intervall von 0.8m/s bis 1.0m/s, so variiert der durchschnittliche Fehler nur um etwa 10%.

Verwendet man die Schritterkennung mit einer festen Schrittlänge von 1.0m, so ergibt sich ein optimaler Interpolationskoeffizient von $\alpha = 0.85$. Entsprechend wurde also der Einfluss des Messmodells reduziert, da die Track-Vorhersage viel besser ist, als bei konstanter Geschwindigkeit (in diesem Fall war $\alpha = 0.7$). Das Messmodell ist dennoch sehr wichtig, um Fehler in der Schrittlänge, der Orientierung oder auch durch die Schritterkennung übersehene oder doppelt gezählte Schritte auszugleichen.

3.2.4.6 Vergleich mit dem klassischen Kalman-Filter

Zum Vergleich mit der Literatur wurde auch ein klassischer Kalman-Filter implementiert. Dieser liefert unterschiedliche Ergebnisse. Im Falle einer konstanten Geschwindigkeit ist der durchschnittliche Fehler um mehr als 10% höher als beim vorgestellten Ansatz, der die Prediktion in die Auswahlmetrik des kNN-Algorithmus integriert. Im Falle des Schrittzählers ist der klassische Kalman-Filter jedoch überlegen. Dies liegt wohl daran, dass der Vorteil, einen Einfluss auf die Auswahl der k nächsten Nachbarn ausüben zu können, wegen der sehr hohen Präzision des Bewegungsmodells selten zum Tragen kommt.

3.2.5 Zusammenfassung der Evaluation

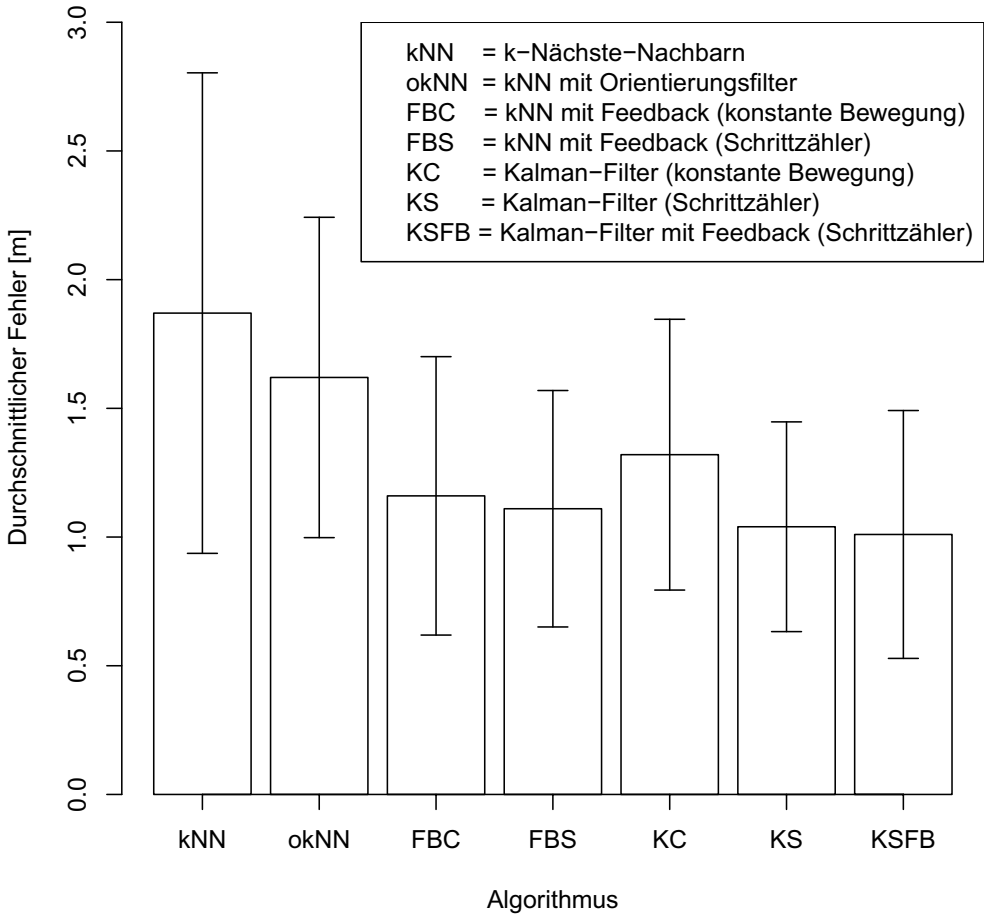


Abbildung 3.10: Durchschnittlicher Fehler der Algorithmen

Zusammenfassend lässt sich festhalten, dass die Filterung der Orientierung

bei klassischen kNN-Algorithmen eine signifikante Verbesserung der Präzision und Genauigkeit ermöglicht. Die Hinzunahme der Kartenentfernung zwischen Kandidaten-Fingerprints und einem Bewegungsmodell führt zu einer weiteren Verbesserung, je nachdem, wie gut das Bewegungsmodell ist. Ein klassischer Kalman-Filter führt in den Fällen, wo nur eine ungenaue Information über die Geschwindigkeit des Nutzers vorliegt, zu einem schlechteren Ergebnis als die vorgestellte Integration in die Auswahl der Nachbarn. Bei Hinzunahme eines besseren Geschwindigkeitsmodells, im vorliegenden Fall durch einen Schritt-Zähler, übertrifft der Kalman-Filter dann wieder diesen Ansatz. Eine Visualisierung der mittleren Fehler und derer Standardfehler aller algorithmischen Varianten dieses Abschnittes findet sich in Abbildung 3.10.

3.3 Bilderkennung zur Indoor-Positionierung

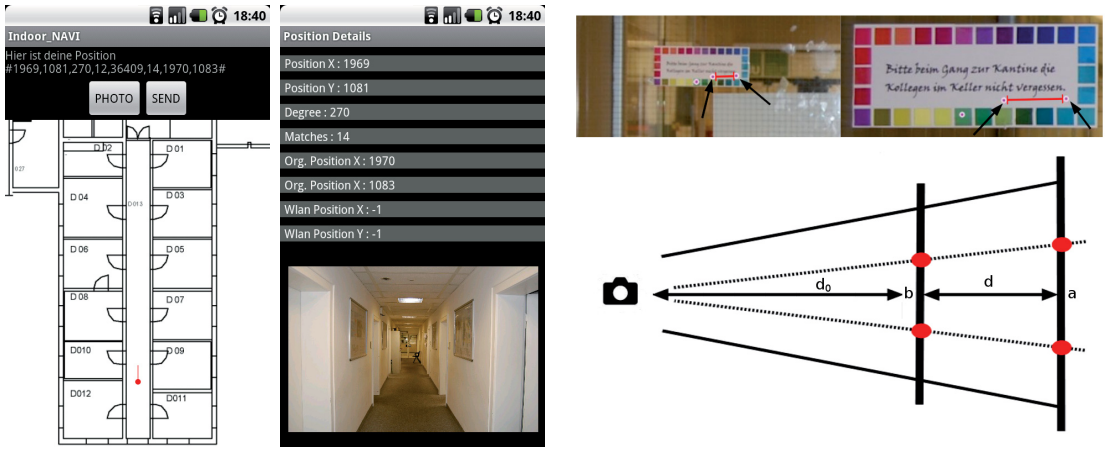
Die Bestimmung der Position einer mobilen Entität durch eine Kamera ist sicher das erstrebenswerteste System. Da der Mensch sich auch vornehmlich visuell orientiert und die natürliche Umgebung in der Regel genug Variation aufweist, um aus einem einzelnen Bild eine Position zu bestimmen, wäre es wünschenswert, wenn diese Form der Positionierung auch mit den Kameras von Smartphones angewendet werden könnte. Obwohl bereits die ersten Smartphones mit Stereo-Kamera-Systemen ausgestattet sind, und somit einen gewissen Aufschluss über die dritte Dimension erlauben, ist die Mehrheit der Smartphones auf eine gewöhnliche, mittelmäßige Kamera beschränkt. Davon ausgehend wurde in [Wern 11d] versucht, eine Bilddatenbank für eine recht homogene Umgebung (gleiche Türrahmen, gleiche Bodenbeläge, ähnliche Beleuchtung, etc.) aufzubauen, und untersucht, ob und wie sich eine solche Datenbank für die Positionierung eignet. Dabei ergeben sich überraschend gute Ergebnisse: Die Abweichungen in der Positionsbestimmung sind in der Größenordnung von Metern, zusätzlich bietet das System den Vorteil, dass die Blickrichtung des Nutzers bekannt ist und so Navigationsanweisungen als Overlays in das Kamerabild (ähnlich zu [Hile 07]) eingezeichnet werden können. Darüber hinaus ist das System sehr stabil bei semantischen Unterschieden. Es positioniert zwar insgesamt etwas ungenauer als typische WLAN-Positionierungs-Systeme, unterscheidet aber wesentlich besser zwischen unterschiedlichen Räumen, weil sich das Bild so stark ändert, dass eine Verwechselung mit einem Flur sehr unwahrscheinlich ist. Schließlich ist solch ein System sehr viel einfacher zu administrieren als ein WLAN-Positionierungssystem. So kann beispielsweise in der Datenbank nach ähnlichen Bildern gesucht werden. Das sind dann auch Kandidaten für eine Verwechslung im Betrieb. Solche Situationen können durch Hinzufügen von Objekten (Kunst, Bilder, Pflanzen, ...) oder auch durch Entfernen des einen oder anderen irreführenden Bildes aus der Datenbank ohne Spezialwissen entschärft werden.
In einer „Offline-Phase“ wird hier, ähnlich wie bei den Verfahren des WLAN-Fingerprinting, eine Bilddatenbank aufgestellt. Diese umfasst

- ein Foto,
- die Position, aus der das Foto aufgenommen wurde,
- die Blickrichtung, in der das Foto aufgenommen wurde, und
- eine automatisch erzeugte, rotations- und skaleninvariante Beschreibung des Bildes.

In einer „Online-Phase“ wird dann ein Bild aufgenommen, die Datenbank nach einem ähnlichen Bild durchsucht, die Position des Bildes übernommen oder gegebenenfalls noch eine heuristische Abstandsschätzung vorgenommen, und die Position der ursprünglichen Aufnahme damit korrigiert.
Um die ubiquitäre Verwendung des Systems zu ermöglichen, kann zusätzlich ein einfaches WLAN-Proximity-System vorgeschaltet werden. Dabei werden dann nur diejenigen Bilder im Detail verglichen, die in der Nähe des aktuellen Aufenthaltsortes des Nutzers aufgenommen wurden. Als positiver Nebeneffekt reduziert sich dadurch auch die Wahrscheinlichkeit für eine Verwechslung der aktuell aufgenommenen Szene mit einem entfernten Ort.
Die Ortsinformation wird in drei verschiedenen Weisen aus Bildinformationen abgeleitet. Im Foto-Modus wird ein einzelnes Foto aufgenommen und daraus die Position berechnet. Im Video-Modus wird ein Video aufgezeichnet und in eine Bildersequenz übersetzt. Diese Bilder werden dann einzeln positioniert und entweder wird der Durchschnitt („Averaging Mode“) der einzelnen Orte oder eine Mehrheitsentscheidung („Voting Mode“) verwendet, um den Ort zu bestimmen.
Es ist möglich, den zur Bildtransformation verwendeten SURF-Algorithmus auf dem Gerät auszuführen. Für die Analyse wurde aber auch eine Serverseitige Implementierung geschaffen, die die Bilder per Upload entgegennimmt und auf dem Server analysiert.

3.3.1 Das „Mobile Visual Indoor Positioning“-System

Im grundlegenden Foto-Modus wird ein hochauflösendes Bild mit der Geräte-Kamera aufgenommen und daraus mit dem SURF-Algorithmus [Bay 06] eine Menge an Feature-Points berechnet. Dabei wurde der SURF-Algorithmus verhältnismäßig aggressiv eingestellt, um möglichst viele Features zu finden. Diese haben dann zwar eine weniger starke Unterscheidungskraft, liefern aber insgesamt bessere Ergebnisse, weil in den doch relativ homogenen Szenen innerhalb von Gebäuden eine Konfiguration, wie sie zur Objekterkennung und Segmentierung sinnvoll wäre, nicht erfolgreich ist.
Diese Features der Datenbank-Bilder und des aktuellen Bildes werden mit dem klassischen Matching-Verfahren, wie beim SIFT-Algorithmus [Lowe 99, Abschnitt 7.1] vorgeschlagen, verglichen. Dabei wird für jeden Feature-Punkt zunächst der nächste Nachbar bestimmt. Um den Effekt von Features im



(a) Screenshots des mobilen Positionierers (b) Geometrie der Distanzschätzung

Abbildung 3.11: Screenshots des mobilen Clients und die Geometrie der Distanzschätzung

Hintergrundrauschen zu reduzieren, wird zusätzlich der zweitnächste Nachbar bestimmt und der nächste Nachbar nur dann als Match akzeptiert, wenn der Distanzunterschied (gemessen durch den Quotienten der Distanzen) zwischen dem nächsten und zweitnächsten Feature einen festgelegten, empirischen Schwellwert überschreitet. Denn durch dieses Vorgehen kann sichergestellt werden, dass es sich um einen reproduzierbaren Feature-Punkt handelt, und nicht um eine Häufung im hochdimensionalen Feature-Raum, wie sie durch irrelevante Strukturen und Hintergrundrauschen in den Bildern typischerweise entsteht. Die Unterscheidungskraft wird also durch die Eindeutigkeit der Nähe zum aktuellen Feature-Punkt bewertet.

Mit diesem Matching-Verfahren auf Bild-Ebene bestimmt sich die Anzahl der Matches in zwei Richtungen: Dabei dient jeweils einmal das aktuelle Datenbankbild und das aktuell aufgenommene Bild als Basis. Dann wird, ähnlich wie auf Feature-Ebene, von den Bildern mit der höchsten und zweithöchsten Anzahl an Matches dasjenige gewählt, bei dem sich die Anzahl der Matches zwischen den Richtungen am Wenigsten unterscheidet. Auf diese Weise wird ein Fall vermieden, der durch stark unterschiedliche Bildqualität zwischen Datenbank und aktuell aufgenommenem Bild entstehen kann: Ein Bild mit wenigen Features (beispielsweise unscharf oder schlecht aufgelöst) wäre zwar das „Richtige“, wird aber allein durch irrelevante Matches mit einem anderen Bild übertroffen.

Auf der Datenbasis dieser Anwendung ist die Berechnung des Abstandes zwischen den Aufnahmepunkten des aktuellen Bildes und des Datenbankbildes nicht möglich. Zunächst ist ja der Abstand d_0, aus dem ein Bild aufgenommen ist, nicht bekannt und durch die variierende Tiefe nicht einmal wohldefiniert. Mit dem Strahlensatz lässt sich für zwei aufgenommene Objete des Durchmessers b und a in Bildern, die aus der Entfernung $d_0 + d$ und d_0 aufgenommen

Abstand d [m]	$\frac{a_p}{b_p}$	α	**Schätzung [m]**	**Fehler [m]**
1	1.92	0.52	1.38	0.38
2	2.90	0.69	2.09	0.09
3	3.92	0.76	2.82	0.18
4	4.82	0.83	3.47	0.53
5	6.02	0.83	4.33	0.67
Schnitt:		0.73		
Std. Abw.:		0.13		

Tabelle 3.3: Distanzschätzung im relevanten Bereich. Für die Schätzung wurde α in Gleichung 3.5 per linearer Regression bestimmt.

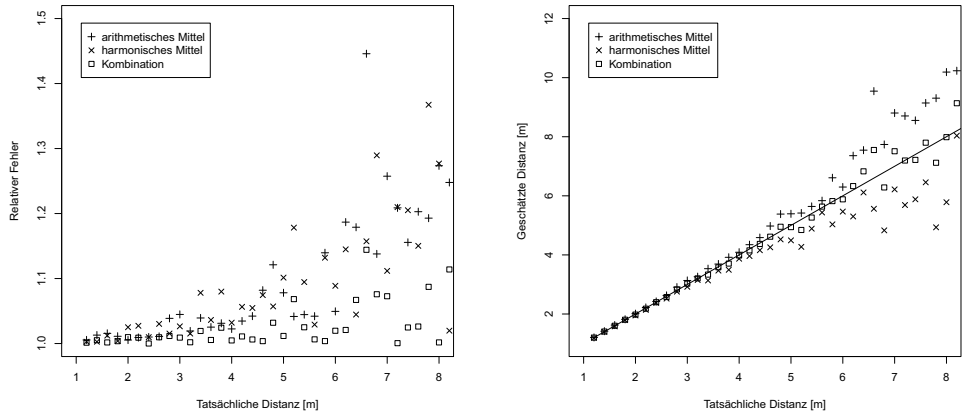
wurden, die folgende Gleichung aufstellen:

$$\frac{d_0 + d}{d_0} = \frac{b}{a} \Leftrightarrow d = \left(\frac{a_p}{b_p} - 1\right) d_0 \tag{3.4}$$

Da aber der Bruch $\frac{b}{a}$ nicht bekannt ist, wird er durch $\frac{a_p}{b_p}$ ersetzt, wobei a_p und b_p die Breite der Objekte auf Bildebene in Pixeln bezeichnet (siehe Abbildung 3.11(b)). Durch die vorliegenden Matches zwischen den Bildern lässt sich der Bruch $\frac{a_p}{b_p}$ approximieren, der Wert für d_0 ist aber im Allgemeinen unbekannt. Solange dieser Wert nicht aufgenommen wird, etwa als die durchschnittliche Bildtiefe definiert, kann leider keine korrekte Aussage über den Abstand d gemacht werden.
Im vorliegenden Testfeld ist es berechtigt, sich einen größeren Fehler zu erlauben. Ob dies im Allgemeinen sinnvoll ist, hängt davon ab, wie die zu erwartenden Aufnahmepositionen der mobilen Geräte sich zu den Positionen der Datenbankbilder verhalten. In empirischen Experimenten mit unterschiedlichen Entfernungen ergab sich, dass ein lineares Regressionsmodell überraschenderweise sehr gut auf der vorliegenden Datenbank funktioniert.

$$d = \alpha \frac{a_p}{b_p} \tag{3.5}$$

Obwohl diese Gleichung offenbar schon an der Stelle falsch ist, an der das Referenzbild genommen wurde (dann ist $a_p = b_p$ und konsequenterweise $\alpha = 0$, weil ja d=0 gilt), zeigen die Ergebnisse, dass es eine bessere Approximation als das feste Setzen von $d = 0$ ergibt (siehe Tabelle 3.3). Dabei wurde für Werte $1.8 \leq \frac{a_p}{b_p} \leq 6.5$ das lineare Modell verwendet, für Werte kleiner 1.8 die Datenbankposition des Bildes, und für Werte größer 6.5 wird eine Warnung ausgegeben. Natürlich ist die Leistungsfähigkeit des linearen Modells durch die Homogenität der Werte d_0 im relevanten Ausschnitt der Datenbank beschränkt. Für komplexe Gebäude mit sehr unterschiedlichen Werten für d_0 und sehr großen Hallen wird eine einheitliche Kalibrierung von α nicht den gewünschten Erfolg bringen.



(a) Relativer Fehler der Distanzschätzung (b) Abweichung von der realen Distanz

Abbildung 3.12: Fehler in der Distanzschätzung und Abweichungen von der realen Distanz

Ein weiterer, wichtiger Aspekt, der die Festlegung eines Wertes α betrifft, ist die Möglichkeit, dass sich das Sichtfeld zwischen unterschiedlichen Geräten unterscheiden kann. Im vorliegenden Fall wurde die Datenbank etwa mit einer handelsüblichen Spiegelreflexkamera erstellt und die Positionierung mit einem Smartphone des Typs HTC-Desire durchgeführt. Ein Unterschied im Sichtfeld zwischen Datenbank und mobilem Endgerät kann auch durch den Parameter α teilweise ausgeglichen werden.
Nach diesen Vorbereitungen muss noch der Bruch $\frac{a_p}{b_p}$ für zwei Bilder bestimmt werden. Nachdem ja zwischen den Bildern schon Matches berechnet worden sind, ist dieses Verfahren relativ einfach: Um eventuelle falsche Matches und Verzerrungen durch einen kleineren Unterschied in der Perspektive auszugleichen, bestimmt man den Durchschnitt der Abstände aller Matches zwischen den Bildern. Wie in Abbildung 3.12(b) dargestellt, tendiert das arithmetische Mittel dazu, den Abstand zu überschätzen, während der harmonische Mittelwert eine vergleichbare Tendenz zur Unterschätzung hat. Entsprechend kann man den Durchschnitt dieser beiden Werte als Quotient im linearen Modell 3.5 verwenden. Die relativen Fehler und ein Vergleich zwischen realer und geschätzer Entferung für die drei Fälle ist in Abbildung 3.12 dargestellt. Die so ermittelte Entfernungsschätzung wird verwendet, um den Aufenthaltsort, ausgehend vom Aufnahmeort des Datenbankbildes, entlang dessen Blickrichtung um diese Distanz nach hinten zu verschieben.
Im Video-Modus finden die Berechnungen grundsätzlich Server-seitig statt, weil ein Smartphone derzeit dem immensen Rechenaufwand nicht gewachsen ist. Dem liegt zu Grunde, dass die Qualität der einzelnen Video-Frames teilweise so schlecht ist, dass eine Positionierung nicht möglich ist oder nur sehr schlechte Ergebnisse erzielt werden. Deshalb wird in einem Zeitfenster konti-

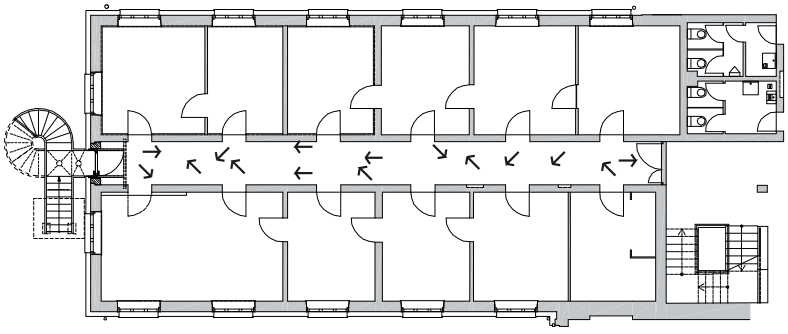


Abbildung 3.13: Verortung der Testbilder im Korridor

nuierlich positioniert und die Teilergebnisse werden dann zusammengefasst.
Für jeden Frame in einem festgelegten Zeitraum wird der obige Algorithmus ausgeführt und es wird pro Frame eine um die Distanz korrigierte Position errechnet. In der ersten Variante („Averaging Mode“) wird aus den Positionen einfach der Mittelwert gebildet. In der zweiten Variante („Voting Mode“) wird aus den Hypothesen für die Datenbank-Bilder dasjenige Bild gewählt, das von den meisten Frames gewählt wurde. In beiden Fällen muss man natürlich bei Bewegungen damit rechnen, dass man die mittlere Aufenthaltsposition im relevanten Zeitfenster erhält, und nicht den aktuellen Aufenthaltsort, der nur aus dem letzten Bild hätte bestimmt werden können. Eine mögliche Gewichtung des Alters der Bilder findet in diesem System noch nicht statt, ist aber eine Option für zukünftige Entwicklungen.
Nach erfolgreicher Positionierung des Gerätes zeigt die Anwendung den aktuellen Ort, die aktuelle Orientierung und gegebenenfalls das Datenbankbild (siehe Abbildung 3.11(a)). Darüber hinaus kann in das Kamerabild noch ein Overlay wie in Abbildung 3.14 eingezeichnet werden. Dieses Overlay wird dabei mit einem Ausschnitt des Datenbank-Bildes gekoppelt, und kann durch den SURF-Algorithmus in Echtzeit auf dem mobilen Endgerät in das Videobild eingezeichnet werden.

3.3.2 Evaluation und Ergebnisse

Für die Evaluation des vorgestellten Ansatzes wurde eine Bilddatenbank, bestehend aus 68 Bildern an gleichmäßig im Bereich der Positionierung verteilten Punkten und 16 Testbildern mit abweichender Position und Orientierung, erstellt. Abbildung 3.13 gibt einen Überblick über die Verortung der Datenbank, der Testbilder und eine Ansicht des betreffenden Flurs.
Die Bilder wurden allesamt in einer Auflösung von 5 Megapixeln mit einem HTC-Desire erzeugt. Für die vollständige Erstellung dieser Datenbank wurden nur 10 Minuten Zeit benötigt. Als guter Schwellwert für die Featurepunkt-Vergleiche in der Datenbank ergab sich 0.0004. Ein Wert, der zu sehr vielen Features führt, was uns eine gewisse Freiheit bei Überdeckungen erlaubt. Allerdings wird dadurch der Bildvergleich aufwändiger. Über die Qualität der

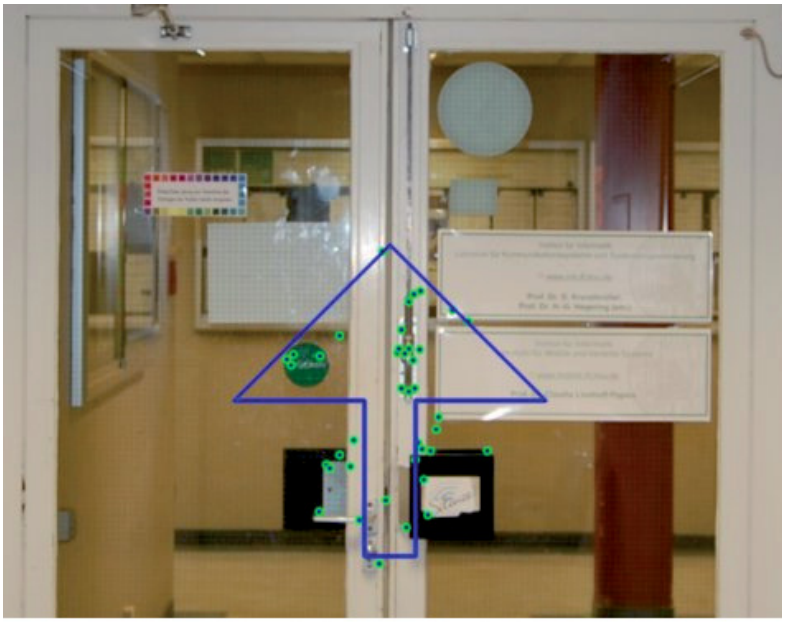


Abbildung 3.14: Einzeichnen von Navigationshilfen

Distanzschätzung in einem synthetischen Fall gibt Abbildung 3.12 Aufschluss. Die Hauptquellen für Fehler an dieser Stelle liegen in möglichen falschen Matches zwischen Punkten und einer perspektivischen Verzerrung zwischen aktuellem Bild und Datenbank-Bild. Im Video-Modus wurde eine Fenstergröße von 15 Frames, etwa 500ms, gewählt und nur eine Auflösung von 0.3 Megapixeln erreicht.

System	**25% [m]**	**50% [m]**	**75% [m]**
Foto-Modus	0.28	0.68	1.25
Video-Modus („Averaging“)	2.39	3.88	4.4
Video-Modus („Voting“)	1.0	2.85	4.4
RADAR [Bahl 00]	1.92	2.94	4.69

Tabelle 3.4: Drei Perzentilen der Positionierungsfehler der drei Modi im Vergleich zu WLAN-Positionierung

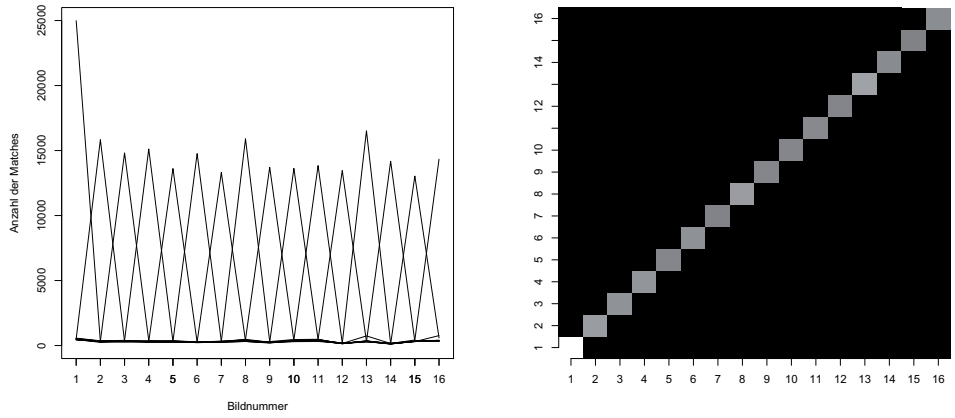
Die Ergebnisse für den mittleren Positionierungsfehler in den drei Modi mit Distanzkorrektur finden sich in Tabelle 3.4, in der auch die Ergebnisse des klassischen WLAN-Positionierungssystems [Bahl 00] zur Einordnung der Werte angegeben sind. Im Fotomodus wurden die 16 Testbilder verwendet. Wie zu erwarten, sind die Ergebnisse im Video-Modus wesentlich schlechter, als im Fotomodus, da die Bildqualität durch die geringe Auflösung, Bewegungsunschärfe und Kodierungsartefakte zur Auswahl falscher Bilder aus der Datenbank führt. Um die Mehrdeutigkeit der Datenbank zu analysieren, wurde eine Kreuzvalidierung wie folgt durchgeführt: Um die Mehrdeutigkeit zu beziffern, wurde

jedes Datenbankbild einmal mit der ganzen Datenbank positioniert und einmal nur mit den anderen Bildern aus der Datenbank. Dabei wurden – wie gewünscht – 97% der Bilder sich selbst zugeordnet und im zweiten Fall, in dem das Bild aus der Datenbank entfernt wurde, fanden nur 70% überhaupt einen Treffer, von denen aber die Meisten aus der direkten Umgebung des Datenbank-Bildes stammen. Der mittlere Positionierungsfehler ergab sich in diesem Fall zu 4.71m. Dieser Durchschnitt ist sogar noch besser als der durchschnittliche Abstand der Bilder, der 5.38m beträgt. Demzufolge wurden häufig Bilder an derselben Stelle, aber mit anderer Ausrichtung erkannt.

Um die Datenbank weiter zu analysieren, wurden drei weitere, unabhängige Test-Datenbanken an denselben 16 Orten zu unterschiedlichen Zeiten erzeugt. Als Basisinformation wurde die Eindeutigkeit der Features der Datenbankbilder analysiert. Dazu wurden alle Paarungen von Bildern aus der Datenbank miteinander verglichen, und die Anzahl der Matches zwischen Feature-Punkten gemessen.

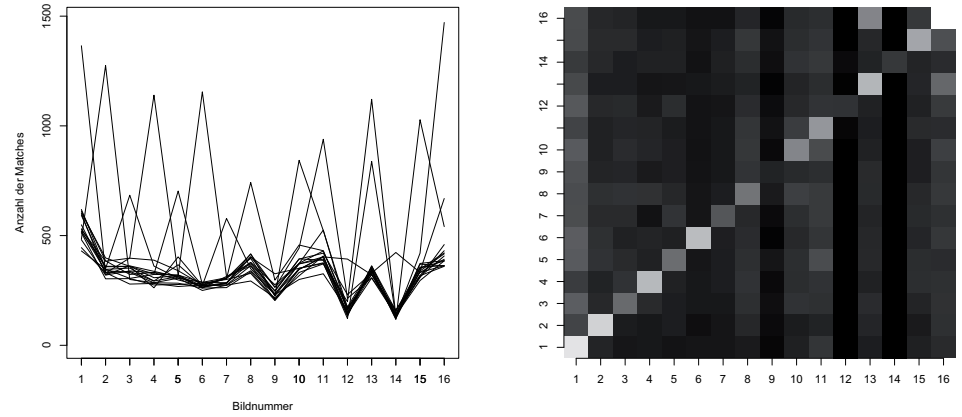
Abbildung 3.15 zeigt das Ergebnis in zwei Darstellungen: Die Konfusionsmatrix kommt der gewünschten Diagonalmatrix sehr nahe. Die Darstellung auf der linken Seite zeigt auf der Y-Achse die Anzahl der Matches und auf der X-Achse die Bilder. Für jedes Bild wurde eine Linie eingezeichnet, deren Maximum immer über dem richtigen Bild liegt. Es formiert sich zusätzlich eine Art Grundrauschen von Matches zwischen unterschiedlichen Bildern. Die wesentliche Erkenntnis besteht aber darin, dass die jeweilige Spitze, die immer vom richtigen Bild erzeugt wurde, wesentlich über diesem Grundrauschen liegt. Selbstvergleiche der anderen Datenbanken ergaben eine ähnliche Darstellung. Bei einem paarweisen Bildvergleich zwischen den zu verschiedenen Zeiten erzeugten Datenbanken ergibt sich ein etwas höheres und nicht mehr so homogenes Grundrauschen. Dieses liegt glücklicherweise immer noch jeweils signifikant unter den Spitzen, die sich aus am selben Ort aufgenommenen Bildern ergeben.

Als weitere Fragestellung ist natürlich interessant, inwiefern ein solches System mit partiellen Überdeckungen durch Personen oder sonstige Objekte zurecht kommt. Dazu wurden einige Testdatensätze mit zufälligen rechteckigen Flächen vorgegebenen Inhalts über die Bilder einer Datenbank gelegt und mit diesen Bildern eine Position berechnet. Die Ergebnisse sind wiederum vielversprechend, da durch den verhältnismäßig niedrigen Schwellwert bei der Extraktion von Feature-Punkten auch Teilflächen der Bilder ausreichend charakteristische Eigenschaften aufweisen, um eine Positionierung durchzuführen. In Abbildung 3.16 finden sich die Ergebnisse, die belegen, dass eine Überdeckung von 20%-30% keinen besonders negativen Einfluss auf die Bestimmung der Position hat. Allerdings ist an dieser Stelle anzumerken, dass ein durch ein nahes Objekt falsch eingestellter Autofokus natürlich die gesamten Bildinformationen zerstört, und in solchen Fällen eine Positionierung sicher nicht mehr möglich ist. Die Erfolgsrate, also die Rate, mit der das richtige Datenbankbild gefunden wird, zeigt eine absteigende Tendenz, bleibt aber im relevanten Bereich auf einem recht hohen Niveau.



(a) Anzahl der Matches, gleiche Datenbank

(b) Konfusionsmatrix, gleiche Datenbank



(c) Anzahl der Matches, verschiedene Datenbanken

(d) Konfusionsmatrix, verschiedene Datenbanken

Abbildung 3.15: Selbstvergleich einer Datenbank und Vergleich zweier verschiedener Datenbanken

3.3.3 Zusammenfassung

Mit dem System MoVIPS wurde gezeigt, dass die Ermittlung von Positionen innerhalb von Gebäuden aus Bildinformationen auch in vielen Situationen noch möglich ist, in denen eine solche Positionierung häufig als problematisch angesehen wurde. Durch geschicktes Einstellen („Overfitting“) der Bildvergleichsalgorithmen, und durch Optimierung der Datenbank in Bezug auf unerwünschte Mehrdeutigkeiten war es möglich, ein Positionierungssystem zu schaffen, das mit einigen Bildern kalibriert wird und eine recht hohe Qualität der Positionierung ermöglicht.
Der besondere Charme dieser Art der Positionierung liegt darin, dass die Vor-

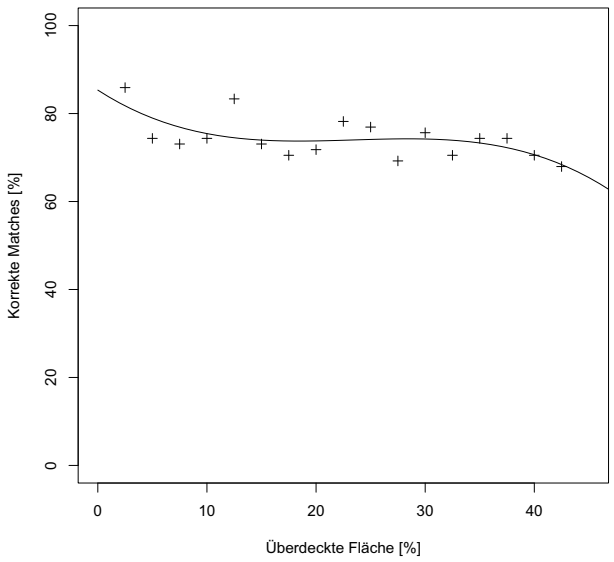


Abbildung 3.16: Einfluss partieller Verdeckungen auf die Bilderkennung und die Positionierung

gehensweise der Algorithmen verständlich ist. Insbesondere durch das Senden des erkannten Bildes an den Client ist eine Prüfung der erhaltenen Position durch den Nutzer möglich. Durch Vergleiche der Datenbank mit sich selbst und auch durch Speichern von Bildern der Online-Phase, die auf mehrere Datenbankbilder gepasst hätten, ist eine weitere Verbesserung der Datenbank möglich.

Das System MoVIPS beschränkt sich derzeit auf die Ermittlung der Position ohne Verwendung der Zeit-Domäne, ohne direkte Integration mit WLAN-Positionierung (abgesehen von der Filterung der Datenbank) und ohne Integration sonstiger Sensorik. Dies sind Bereiche, in denen in der nächsten Zeit noch einige Verbesserungen möglich sind. Beispielsweise ist davon auszugehen, dass die Verwendung eines Kalman-Filters wie im vorigen Abschnitt 3.2, der die Bewegung des Nutzers zwischen verschiedenen Positionsbestimmungen mit MoVIPS fortschreibt, eine deutliche Qualitätsverbesserung für Anwendungen des Trackings mit sich bringen würde. Auch die Integration weiterer Bild-Charakteristika (Farbinformationen, Kanten, etc.) könnte das System weiter verbessern. Hierbei muss nur genau evaluiert werden, wie sich das System dann in Bezug auf partielle Verdeckungen verhält.

Für die Kalibrierung eines solchen Systems sind die Verfahren des SLAM von großem Interesse, in denen ein dreidimensionales, farbiges Abbild der Umgebung aus einem Video erstellt wird. Auf Basis solcher Daten, wie sie problemlos beispielsweise mit der Kinect ermittelt werden können (siehe Abschnitt 2.1.7.1), wird die Kalibrierung noch viel einfacher, und das System kann auf

noch mehr Bilddaten akzeptabler Qualität zurückgreifen.
Insgesamt ist nach Meinung des Autors davon auszugehen, dass die Positionierung mit Bildinformationen sich in den nächsten Jahren durchsetzen wird. Denn die visuellen Unterschiede zwischen Orten mit unterschiedlicher symbolischer Position (Zimmer, Flure, Eingangsbereiche, etc.) sind sehr groß, während dies nicht unbedingt für andere, messbare Parameter wie WLAN-Signalstärken gilt.

3.4 Zusammenfassung und Ausblick

In diesem Kapitel wurden Erweiterungen zu bestehenden Positionierungs- und Tracking-Systemen für das Innere von Gebäuden vorgestellt.
Zunächst wurde eine Umgebung geschaffen, in der auf Basis von CAD-Daten eine recht realistische WLAN-Ausbreitung in Gebäuden simuliert werden kann, die auf verschiedenen Ausbreitungsmodellen beruht. Ganz bewusst wurde hier auf Objekt- und Strahlungsinteraktion (Spiegelung, Selbstinterferenz, Spiegelungsinterferenz, etc.) weitgehend verzichtet, weil Baupläne bei Weitem nicht in einer Genauigkeit gezeichnet sind, die für die vernünftige Simulation solcher Effekte nötig wäre, und darüber hinaus die Umgebung durch nicht modellierte Objekte wie Möbel, Bodenbeläge, Lampen, Wandbehänge, magnetische und elektrische Störquellen ohnehin sehr starken Störungen unterliegt. Mit dieser Umgebung wurden einige Evaluierungen in den vorliegenden Arbeiten zur Indoor-Positionierung mit WLAN ermöglicht und eine statistische Vor-Evaluierung sowie ein handliches Tool für die Implementierung und Konzeption von Positionierungsalgorithmen geschaffen.
Es ist gelungen, in der Domäne der WLAN-Positionierung eine Verbesserung der Positionierungsqualität zu erreichen, ohne auf extrem rechenaufwändige Verfahren wie Partikelfilter zurückzugreifen. Dazu wurde mit einem elektronischen Kompass die Orientierung des Smartphones ermittelt. Der Einfluss dieser Informationen auf Indoor-Positionierungs-Algorithmen wurde analysiert, und für Tracking-Anwendungen wurde eine Alternative zum klassischen Kalman-Filter konzipiert und evaluiert, die die Vorhersage der aktuellen Position mit einem Kompass-gestützten Bewegungsmodell nicht nur in die endgültige Position einbezieht, sondern schon in die Auswahl der k nächsten Nachbarn im Messmodell. Es hat sich herausgestellt, dass diese Vorgehensweise insbesondere dann sinnvoll ist, wenn keine detaillierten Informationen über die aktuelle Geschwindigkeit erhoben werden können oder sollen. Als Nebenergebnis wurde eine weitere Bestätigung dafür gegeben, dass sowohl die Kompassinformation, als auch die Information der typischerweise verbauten einfachen Beschleunigungssensoren sehr nützlich für die Positionierung sein können, was von einigen Autoren immer wieder in Zweifel gezogen wurde, weil eine direkte Positionsbestimmung mit den Methoden der inertialen Navigation nicht möglich ist.
Als weiteren Beitrag zur Indoorpositionierung wurde untersucht, ob die doch recht homogenen Umgebungen innerhalb von Gebäuden eine Positionierung

mit fortgeschrittenen Bildvergleichsverfahren ermöglichen. Die Antwort hierauf ist klar positiv, es ist sogar möglich, die notwendigen Berechnungen direkt auf dem Smartphone durchzuführen. Im Rahmen dieser Untersuchung wurde festgestellt, dass man durchaus auch bei eindimensionalen Bildern einen Vorteil dadurch erzielen kann, dass man eine Entfernungsschätzung auf unzureichenden Daten vornimmt und die Position eines Standortes eines Vergleichsbildes damit korrigiert. Weiter ist es auch gelungen, das wichtigste, typische Argument gegen Bild-basierte Positionierungssysteme, nämlich dass diese in der Regel mit Überdeckungen nur sehr schlecht zurecht kommen, teilweise zu widerlegen. Wenn mit einem optischen Fluß oder gar mit dem Zählen von senkrechten Kanten als Indikator für Türen positioniert wird, fehlt durch eine Überdeckung in der Regel eine wesentliche Information. Bei dem vorgeschlagenen Vergleichsverfahren eignen sich aber beliebige Ausschnitte von 70%- 80%, um noch eine akzeptable Position zu bestimmen.
Insgesamt haben die vorgestellten Methoden gemein, dass die Berechnungen auf einem handelsüblichen Smartphone durchgeführt werden können und dass die Systeme keine besonderen Infrastrukturkomponenten benötigen. Dies eröffnet den Weg zu anonymen Positionsbestimmungssystemen, in denen nur das Smartphone Zugriff auf die eigene Position hat. Darüber hinaus muss das Smartphone nicht über eine existierende Infrastruktur unterrichtet werden, lediglich Kartenmaterial in Form von Bild und Fingerprint-Datenbanken muss zur Verfügung stehen. Das folgende Kapitel beschäftigt sich daher genauer mit Karten und zeigt insbesondere, dass beispielsweise die relevanten Fingerprint-Daten für einen skalierbaren ortsbezogenen Dienst in Gebäuden in Kartenausschnitt-Bilder integriert werden können.

4 Umgebungsmodelle für die Indoor-Navigation

In Umgebungen des „Ubiquitous Computing“ werden häufig georeferenzierte Daten verwendet, also Daten, die einen Bezug zu einem geographischen Ort haben. Diese Daten können viele Gestalten annehmen. Da aber die Erfassung solcher georeferenzierter Daten mitunter sehr aufwändig ist, lohnt es sich, eine allgemeine Vorgehensweise im Umgang mit solchen Daten zu konzipieren, die eine einheitliche Behandlung zumindest für die Daten ermöglicht, die von mehreren unterschiedlichen Anwendungen eingesetzt werden. Ein Regelwerk, welches definiert, wie eine Organisation und Speicherung von ortsbezogenen Daten vorzunehmen ist, nennt man *Umgebungsmodell*. Dabei ist ein Umgebungsmodell nicht unbedingt ein Modell im Sinne von einem Element der Unified Modelling Language [Unif 12], sondern kann eine beliebige – auch informelle – Vorschrift zur Behandlung von Ortsdaten sein.
Bei Umgebungsmodellen unterscheidet man zunächst zwischen metrischen und topologischen Umgebungsmodellen. *Metrische Umgebungsmodelle* zeichnen sich durch einen klaren Bezug zu einem Koordinatensystem aus. Beispielsweise sind CAD-Zeichnungen typische metrische Umgebungsmodelle. *Topologische Umgebungsmodelle* modellieren keine Ortsgrößen in Koordinatensystemen, sondern die Beziehungen zwischen Orten. Ein typisches topologisches Umgebungsmodell ist ein Zugangsgraph, der einen Knoten pro Raum und eine ungewichtete Kante zwischen zwei Räumen hat, wenn ein direkter Übergang zwischen den Räumen möglich ist.
Diese Konzepte lassen sich auch zu *hybriden Modellen* kombinieren. Beispielsweise kann der Zugangsgraph zwischen Räumen in eine Karte eingebettet sein und mit der kürzesten Entfernung zwischen den Raummittelpunkten als Gewicht versehen werden. Eine wichtige Herausforderung beim Entwurf von Umgebungsmodellen liegt darin, dass ein Kompromiss zwischen den folgenden gegenläufigen Anforderungen getroffen werden muss:

- *Skalierbarkeit:* Ein Umgebungsmodell soll skalierbar sein, indem benötigte Features hinzugefügt oder herausgenommen werden, und so die Größe und Komplexität einer Instanz des Umgebungsmodells an den Anwendungsfall angepasst werden kann.

- *Verfügbarkeit:* Ein Umgebungsmodell soll so einfach sein, dass es aus typischerweise vorhandenen Daten mit möglichst geringem Aufwand für ein konkretes Gebäude instanziiert werden kann.

- *Zweckdienlichkeit:* Ein Umgebungsmodell soll so beschaffen sein, dass es die Operationen, die eine Anwendung durchführen will, optimal unterstützt.
- *Einfachheit:* Ein Umgebungsmodell soll möglichst einfach und klein sein. Es soll nicht mehr Daten umfassen, als unbedingt notwendig, und diese Daten sollten möglichst effizient organisiert werden.

Wenn man diese Anforderungen betrachtet, so kann man feststellen, dass keines der bisher vorgeschlagenen Umgebungsmodelle diese Anforderungen gleichermaßen erfüllen kann. Auch im Rahmen des Projektes HIPS stellt sich heraus, dass man bei der Verwendung von Umgebungsmodellen die Eigenschaft der Einfachheit immer betonen sollte. Wenn Daten vorliegen, sollten sie nicht aufwändig weiterverarbeitet werden, weil dann Änderungen an den Rohdaten fast immer zu Fehlern und Inkonsistenzen führen. Allenfalls dürfen die Rohdaten bearbeitet oder langlebige Informationen - getrennt von den Rohdaten - ergänzt werden.
Im Folgenden werden zwei Umgebungsmodelle vorgestellt, die gerade der Anforderung der Zweckdienlichkeit und Einfachheit entgegen kommen. Da es sich dabei naturgemäß um einfache bis minimalistische Umgebungsmodelle handelt, werden für komplexere Operationen im Bereich der Visualisierung und der Routenberechnung neue Algorithmen benötigt. Insgesamt ist aber die Kombination eines minimalistischen Umgebungsmodells mit intelligenten Algorithmen sicher sinnvoller, als die Erstellung unnötig komplexer Umgebungsmodelle.

4.1 Umgebungsmodelle

Umgebungsmodelle können beliebig komplex sein. So bietet der OGC-Standard GML [Geog 12] ein Anwendungsschema CityGML [City 12, Kolb 05], mit dem sich mehrstöckige Gebäude in einer festen Form und einer festen Semantik modellieren lassen. Dieses Umgebungsmodell ist dann so komplex, dass es für die Indoor-Navigation aus mehreren Gründen irrelevant ist: Zunächst sind keine ausreichenden Daten vorhanden, um das Modell zweckdienlich zu füllen, und eine manuelle Erstellung aller benötigten Daten wäre überaus aufwändig. Darüber hinaus bietet das Umgebungsmodell CityGML zwar einen passablen Rahmen, bildet aber nicht alle notwendigen Informationen für eine qualitativ hochwertige Indoornavigation direkt ab. Ein Beleg dafür ist die Erweiterung BIGML [Kess 11a] von CityGML für die Indoor-Navigation, die insbesondere die Notwendigkeit zusätzlicher Ergänzungen und Änderungen an CityGML zeigen soll. Die Argumentation ist allerdings fragwürdig, weil nicht nachgewiesen wird, dass die modellierten Daten über die Grenzen des Anwendungsgebietes hinweg nützlich sein können. Damit wird die Grund-Philosophie von CityGML, nur Daten zu modellieren, die durch ihre Modellierung mehreren Anwendungsgebieten nützen [Kolb 05], ignoriert.

4.1.1 Elementare Darstellung von Geometrie und Topologie

Unabhängig davon, welche zusätzlichen Informationen ein Umgebungsmodell beinhalten soll, muss jedes Umgebungsmodell in der Lage sein, Eigenschaften von Orten und Geometrien zu modellieren. Dabei gibt es zunächst die grundsätzliche Unterscheidung, ob das Umgebungsmodell metrisch oder topologisch sein soll.
Wie bereits beschrieben, speichern metrische Umgebungsmodelle Koordinaten in einem geometrischen *Koordinatensystem*. Eine Koordinate ist ein Vektor in einem metrischen Raum. Wie dieser metrische Raum zur Realität in Beziehung steht, wird über einen Referenz-Rahmen mit bekannter Position, Ausrichtung und Koordinaten beschrieben. *World Geodetic System 84 (WGS84)* [Depa 97] ist beispielsweise ein durch das GPS-System weit verbreitetes *globales Koordinatensystem*, mit dem sich jeder Ort auf der Welt beschreiben lässt. Im Gegensatz zu solchen globalen Koordinatensystemen gibt es häufig auch *lokale Koordinatensysteme*. Diese sind in der Regel nicht für die Abbildung der gesamten Erde gedacht, sondern nur für die Abbildung eines ganz bestimmten Teils. So verwenden die meisten metrischen Positionierungssysteme – etwa ActiveBat [Ward 97] oder UbiSense [Ubis 12] – ein frei definierbares lokales Koordinatensystem. Besonders nützlich können lokale Koordinatensysteme für die Modellierung dann sein, wenn man wiederkehrende Objekte nur einmal in einem lokalen Objekt-Koordinatensystem modelliert, und dann mit verändertem Referenzrahmen mehrfach in eine Umgebungsbeschreibung einfügt. Für grundlegende Betrachtungen zum Thema Koordinatensysteme sei auf [Küp 05, Kapitel 2] verwiesen.
Symbolische Koordinatensysteme verwenden hingegen eine freie Bezeichnung für den Ort. Eine symbolische Koordinate ist ein Element einer Menge L aus möglichen Orten. Diese Menge kann aus den Identifikatoren der Sensoren, etwa bei ActiveBadge [Want 92], oder auch aus Raumnamen, Straßennamen oder anderen Bezeichnungen für einen Ort bestehen.

4.1.2 Darstellung von Beziehungen

Ist die Umgebung in einem geometrischen Koordinatensystem modelliert, so können einige Beziehungen bereits direkt berechnet werden. So ist typischerweise eine Abstandsberechnung zwischen zwei verschiedenen Koordinaten möglich. Leider führt dieser Abstand zwischen Koordinaten in Gebäuden sehr häufig zu massiven Fehleinschätzungen: Der in Luftlinie nächste Raum auf derselben Ebene in einem Bürogebäude könnte wie in Abbildung 4.1 nur auf einem sehr weiten Weg über das Erdgeschoss erreichbar sein. Insbesondere ist in der Realität und in der topologischen Sichtweise der Raum C näher an Raum A, eine rein geometrische Modellierung würde aber zunächst Raum B näher an Raum A sehen.
Sind für symbolische Koordinaten die Relationen „Enthalten-Sein“ und

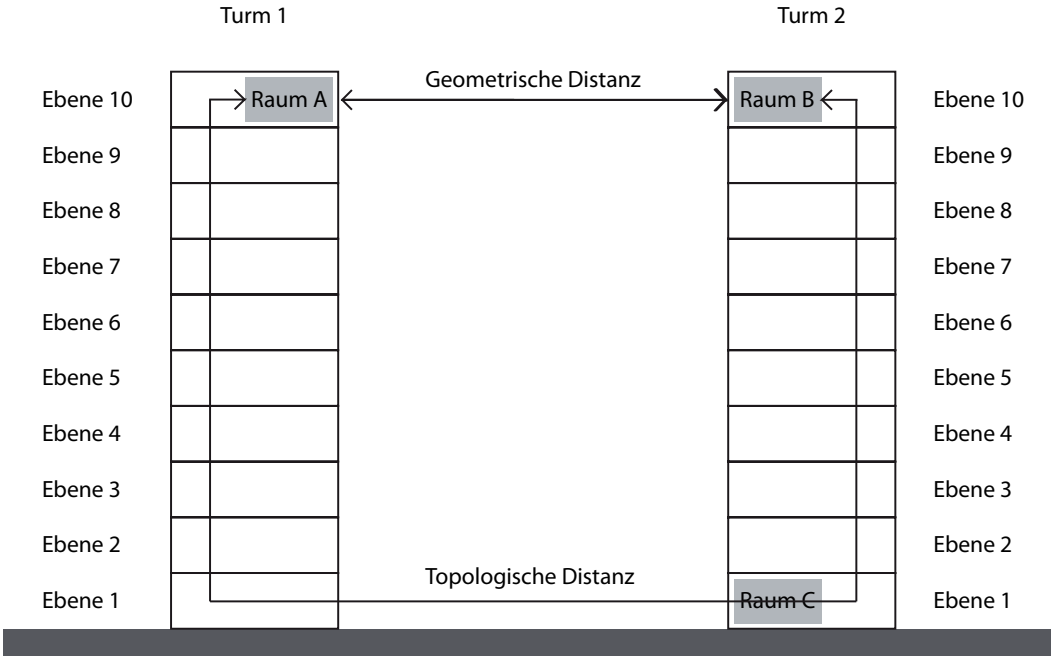


Abbildung 4.1: Topologische und geometrische Entfernungen im Vergleich

„Verbunden-Sein“ modelliert, so ermittelt eine Längenberechnung auf diesen topologischen Beziehungen eine vernünftigere Distanz, allerdings mit erhöhtem Aufwand: „Raum A ist enthalten in Turm 1,Ebene 10. Turm 1,Ebene 10 ist verbunden mit Turm 1, Ebene 9, [...] Turm 1, Ebene 0 ist verbunden mit Turm 2 Ebene 0, Turm 2, Ebene 0 ist verbunden mit Turm 2, Ebene 1, [...] Turm 2 Ebene 10 ist verbunden mit Raum B.“

4.1.3 Anforderungen an Umgebungsmodelle

In der wegweisenden Arbeit [Beck 05] analysieren die Autoren die Anforderungen verschiedener ortsbezogener Anwendungen in Bezug auf Umgebungsinformation, und zeigen systematisch auf, in welcher Weise diese Anforderungen erfüllt werden können. Dabei schlagen sie kein geschlossenes Umgebungsmodell vor, sondern bieten eine Übersicht über Methoden an. Für jede Anwendung kann man sich aus diesem Fundus auf Grund der speziellen Anforderungen der Anwendung für eine optimale Art und Weise der Modellierung der benötigten Aspekte entscheiden. Diese Arbeit ist also in höchstem Maße informativ und schafft es, dem Leser eine Übersicht zu geben, ohne seine Gedanken vorschnell in Schranken einer praktischen Machbarkeit oder einer früh getroffenen Design-Entscheidung zu weisen.
Wenn man die in der Arbeit [Beck 05] genannten Anfragen noch mit hybriden Anfragen vervollständigt, erhält man die folgende Übersicht über möglicherweise zu unterstützende Anfragen:

- Geometrische Beziehungen

  - Distanzen und Längen
  - Nächste Nachbarn
  - Gebiete

- Topologische Beziehungen
  - Enthalten-Sein
  - Verbunden-Sein

- Hybride Beziehungen
  - Kürzeste Wege, topologische Distanz
  - Abbildung zwischen geometrischen und topologischen Objekten

Die Autoren der Arbeit [Beck 05] gehen davon aus, dass es für die Ermittlung kürzester Wege reicht, die topologische „Verbunden-Sein"-Relation zu modellieren. In der Realität benötigt man aber neben dieser Verbundenheitsrelation oft eine metrische Maßzahl für den Aufwand, einer solchen Verbindung zu folgen. Diese kann beispielsweise durch die geometrische Distanz gegeben werden. In diesem Sinne ist also die Suche nach kurzen Wegen als eine hybride Operation aufzufassen, da sie in aller Regel auch die geometrische Distanz in irgendeiner Form verwendet. Außerdem wurde eine Abbildungsmöglichkeit zwischen geometrischen Angaben – etwa Koordinaten – und topologischen Angaben – beispielsweise Knoten in einem Graphen oder Raumbezeichner – ergänzt.
Eine von der Herangehensweise ähnliche Ausarbeitung von Anfragen und Beziehungen wurde für den Fall von Navigationsanwendungen in [Wern 10] vorgelegt. Im Folgenden wird eine kurze Übersicht über Umgebungsmodelle in der Literatur, deren Entstehungskontext und Anwendungszweck gegeben.

4.1.4 Umgebungsmodelle für Gebäude

Die Beschreibung von Umgebungsmodellen in der Literatur ist sehr vielfältig. Es gibt mindestens drei wichtige Perspektiven auf das Thema: Die erste Perspektive, die meist von Arbeiten für Prototypen ortsbezogener Dienste aufgestellt wird, besteht darin, eine gewisse, minimale Funktionalität umzusetzen, um einen bestimmten ortsbezogenen Dienst mit möglichst wenig zusätzlichem Aufwand zeigen zu können. Die zweite Perspektive ist die Perspektive der kreativen Industrie (Architektur, Kunst, etc.), die sich von Umgebungsmodellen eine bessere Austauschbarkeit der Daten ihrer Planungssysteme erhofft. Die dritte Perspektive ist die der öffentlichen Standardisierung, in der möglichst viele unterschiedliche Anwender von Geo-Daten ein gemeinsames Umgebungsmodell verabschieden sollen, das dann auch in einer breiten Anwendergruppe verwendet werden kann.

4.1.4.1 Towntology-Projekt

Im Towntology-Projekt [Town 12] besteht das Hauptziel darin, die Nutzung von Ontologien im Bereich der Städteentwicklung zu erforschen und zu bewerben. Basierend auf den Erfahrungen, die im Projekt COMBINE [COMB 12] gemacht wurden, nämlich, dass eine einheitliche Darstellung von Gebäuden selbst für den relativ kleinen Bereich der Anwendungen für die Energiebewertung wegen zu großer konzeptioneller Unterschiede nicht ohne Weiteres möglich ist, sollte eine ontologische Sicht auf Städte entwickelt werden, die eine höhere Dynamik und eine – zumindest europäisch – multilinguale und multikulturelle Behandlung der Glossare und Bedeutungen ermöglicht.
Obwohl es im Towntology-Projekt keineswegs um die Erstellung von Umgebungsmodellen für das Innere von Gebäuden geht, ist dieses Projekt doch sehr wichtig, da es derzeit das wohl umfangreichste Projekt ist, welches mit ontologischen Methoden die Komplexität von Städten abbildet. Das wichtigste Design-Ziel besteht darin, die Kommunikation und den Austausch von Daten zwischen unterschiedlichen Softwaresystemen zu ermöglichen. So kommen bei der Planung von Gebäuden für die unterschiedlichen Aspekte der Planung sehr unterschiedliche Software-Umgebungen zum Einsatz, die nach Möglichkeit eine gewisse Interoperabilität haben sollten.
Versuche in diese Richtung gibt es viele, so etwa die Geographic Markup Language (GML), das GML-Anwendungsschema CityGML, Industry Foundation Classes (IFC), Building Information Models (BIM) und vieles mehr. All diese Modellierungsversuche stellen auf die eine oder andere Art eine Ontologie auf. Im vorläufigen Abschlussbericht des Projektes [Tell 12] werden daher die Einflüsse vieler einzelner Modellierungs- und Standardisierungsprobleme in einer Gesamtsicht vorgestellt und diskutiert.
Für die Zwecke dieser Arbeit geht das Projekt aber weit über das Notwendige hinaus. Als Folge davon sind einige Konzepte derart komplex, dass der Implementierungsaufwand den Aufwand für eine Ad-Hoc-Lösung bei Weitem übersteigt, und es leider auch nicht zu erwarten ist, dass in den nächsten Jahren GIS-Daten entstehen, mit denen dieser Aufwand im Nachhinein gerechtfertigt wird. Es ist also an dieser Stelle die Frage zu stellen, ob eine Interoperabilität über Minimalismus und Simplizität – in der Arbeit [Beck 05] wird hierzu der Terminus „Minimal Modelling Effort“ eingeführt – einer Interoperabilität über die komplexe Modellierung aller Eventualitäten vorzuziehen ist. Diese Frage lässt sich derzeit nicht beantworten, die Zukunft wird zeigen, welche Herangehensweise sich durchsetzen wird.

4.1.4.2 Building Information Models (BIM) und Industry Foundation Classes (IFC)

Mit „Building Information Modelling“ bezeichnet man den Prozess, alle Aspekte einer Gebäudeentwicklung in einer zentralen Datenbasis zu erfassen, die dann automatisch konsistent sind, weil jedes Objekt nur einmal modelliert

wird. Durch Änderungen an einem Objekt werden dann automatisch alle Sichten wie Baupläne, Schnittzeichnungen und Kalkulationen aktualisiert. „Building Information Modelling“ ist demnach ein Tool, um den Lebenszyklus eines Gebäudes zu verwalten.
Weit wichtiger in diesem Zusammenhang ist die offene Spezifikation und Standardisierung eines digitalen, objektorientierten Austauschformates für solche Modelle durch die „International Alliance for Interoperability (IAI)“. Die *Industry Foundation Classes (IFC)* sind nach dem Entity-Relationship-Modell konzipiert und ermöglichen die semantische Beschreibung vieler Aspekte der Konstruktion von Gebäuden.

4.1.4.3 nD-Modellierung

Die nD-Modellierung wurde eingeführt, um die Integration von Design- und Konstruktionsprozessen weiter voranzubringen. Dabei ist ein nD-Modell eine Erweiterung des Building-Information-Model. Die Grundidee ist hier, die Medienbrüche, die in der traditionellen Praxis in der Konstruktion vorkommen, zu überwinden, indem eine zentrale Datenbank mit „intelligenten Objekten“ erstellt wird, die die traditionellen Dokumente wie Baupläne, Schnittzeichnungen und Zeitpläne automatisch als Sicht erzeugt [Lee 05].
Aus einer anderen Perspektive kann man die nD-Modellierung auch als Fortsetzung der Evolution von 2D-Modellierung, bei der für ein Gebäude viele Zeichnungen entstehen, die in der Regel manuell synchron gehalten werden müssen, über 3D-Modellierung, bei der diese vielen Zeichnungen aus einem einheitlichen 3D-Modell erzeugt werden können, hin zu 4D-Modellierung betrachten, wo der zeitliche Konstruktionsablauf mit modelliert wird, und damit bereits der Konstruktionsfortschritt modelliert und überwacht werden kann.
Insgesamt bezeichnet die nD-Modellierung also eine Herangehensweise, bei der die notwendigen, traditionellen Konstruktionsdokumente beibehalten werden, die klassischen Inkonsistenzen zwischen den verschiedenen Dokumenten aber so weit wie möglich dadurch verhindert werden, dass alle Informationen in einer zentralen, möglichst redundanzfreien Informationsbasis verwaltet werden.

4.1.4.4 GML und CityGML

Die Geographic Markup Language (GML) ist eine XML-Grammatik, mit der sich geometrische Datenbestände modellieren lassen. Sie fungiert gleichermaßen als Austauschformat wie als Datenmodell. GML erlaubt die Spezifikation von grundlegenden geometrischen Objekten wie Punkten, Linien und Polygonen, Flächen, Volumenkörpern, Texturen etc., und ist flexibel erweiterbar.
Eine solche GML-Erweiterung ist der OGC-Standard CityGML, der die Verwendung von Stadtmodellen in verschiedene Anwendungen ermöglichen soll, ohne dass die Inhalte jeweils neu aufbereitet werden müssen. CityGML ist als Anwendungsschema der Geography Markup Language Version 3.1.1 (GML3)

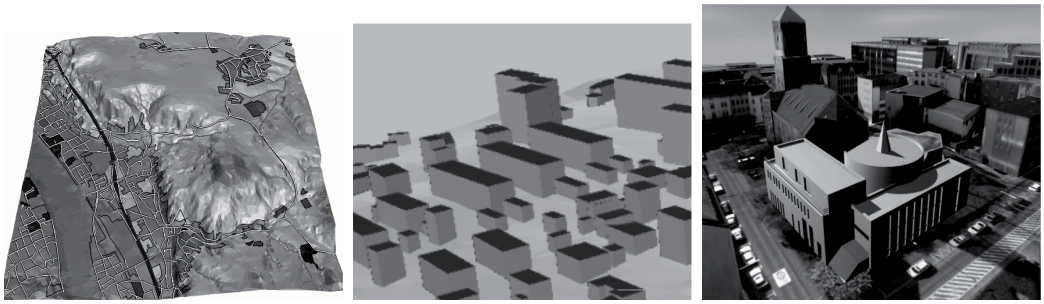
implementiert, und für die Visualisierung und Planung von Städten vorgesehen. Die Komplexität einer solchen Sprache wird im Fall von CityGML dadurch gerechtfertigt, dass Experten aus den unterschiedlichsten Anwendungsbereichen, unter anderem aus der Umweltsimulation, der Städteplanung, dem Gebäudemanagement, dem Notfallmanagement und der Personennavigation, an der Entwicklung von CityGML aktiv beteiligt sind [Kolb 05].
Es handelt sich um ein semantisches 3D-Modell, das aus einer Kombination eines geometrischen Modells und einer ontologischen Struktur besteht. Der höhere Aufwand für eine solche Modellierung ist ökonomisch natürlich nur dann gerechtfertigt, wenn mehrere Nutzer dieselben Daten nutzen können und wollen.
In der Grundstruktur unterscheidet CityGML vier Aspekte:

- Semantik
- Geometrie
- Topologie
- Erscheinungsbild (Appearance)

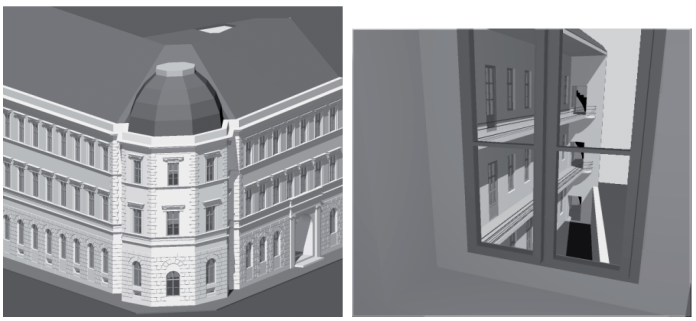
Die Modellierung dieser Aspekte wird auf fünf verschiedenen Genauigkeitsebenen („Level-Of-Detail“, LOD) ermöglicht. Die niedrigste Stufe eignet sich für ein digitales, flaches Abbild des Gebietes, in den weiteren Stufen kommen sukzessive Block-Modelle von Gebäuden (LOD1), Dächer, Balkone und Treppen (LOD2), architektonische Eigenheiten wie Fenster, Türen, Wandstrukturen, Dachstrukturen (LOD3), und schließlich der Innenraum von Gebäuden (LOD4) hinzu. Die unterschiedlichen Detailgrade sind in Abbildung 4.2 abgebildet.
Obwohl in CityGML das Innere von Gebäuden explizit durch das höchste Detaillevel vorgesehen ist, bleibt die vorgegebene Ontologie ganz klar auf das Äußere von Gebäuden beschränkt. So modelliert ein *Transportation Object* eine Brücke oder auch eine Straße, aber keine Rolltreppe und auch keinen Aufzug.

4.1.4.5 OpenStreetMap und OpenIndoorMap

OpenStreetMap ist ein offenes Projekt, das sich zum Ziel gesetzt hat, durch seine Nutzer eine vollständige und freie Weltkarte zu erstellen. Dazu haben die Initiatoren neben einer Web-Plattform ein sehr einfaches Format geschaffen, mit dem zunächst die grundlegenden geometrischen Features einer Weltkarte erstellt und bearbeitet werden können. Dieser Geometrie-Layer, der zunächst nur Punkte, Linien und Flächen beherrscht, wird durch Möglichkeiten zum freien Taggen für alle Anwendungsbereiche offen gehalten. Gruppierungen von geometrischen Objekten, zum Beispiel zu Gebäuden, sind durch Relationen möglich.



(a) LOD0 (b) LOD1 (c) LOD2



(d) LOD3 (e) LOD4

Abbildung 4.2: Verschiedene Detailgrade in CityGML, Abbildung aus [Kolb 05]

Für gewisse Zwecke von besonderem Interesse wurden einige dieser Tags festgelegt [Open 12c], so muss etwa eine Straße immer den Tag *highway=** enthalten. In den letzten Jahren hat sich auch innerhalb der OpenStreetMap ein besonderes Interesse an Gebäuden herausgebildet. Nicht nur, dass die Umrisse von vielen Gebäuden enthalten sind, es gibt mittlerweile ein Teilprojekt namens „IndoorOSM“, das möglichst viele für die Indoor-Navigation und Indoor-Visualisierung notwendige Informationen in der OpenStreetMap erfassen will. Dazu wird eine Relation *building* erstellt, die neben einigen globalen Tags (Adresse, Anzahl der Ebenen, Fassade, Name, etc.) wiederum für jeden Ebene eine Relation enthält, die sich aus diversen Informationen zusammensetzt. Im Grunde werden Gebäude als Relation zwischen Gebäudeteilen modelliert, die wiederum Begrenzungslinien, Zugänge, Räume, Ebenen und Türen enthält. Dabei werden flächige Features (Räume, Ebenen) als geschlossene Wege modelliert und punktuelle Objekte (Türen, etc.) als Punkte mit freien Tags. Interessant ist die Modellierung vertikaler Übergänge durch den Gebäudeteil *VerticalPassage*. Eine vertikale Passage muss nämlich Türen als Eingänge haben und ist selbst einem Raum ebenbürtig. So können dann auch Rolltreppen modelliert werden, die nur in eine Richtung zeigen und verschiedene Zugangskoordinaten auf verschiedenen Ebenen haben. Für eine genaue Beschreibung mit Beispielen sei auf die Projektseite [Open 12b] verwiesen.

4.1.4.6 The Guide Project

Im GUIDE-Projekt wurde ein kontext-abhängiges Touristen-Informations-System entwickelt [The 12b, Chev 00]. Dabei wurden die Ideen von Hypertext und Hypertext-Verlinkungen in den Mittelpunkt gestellt. Das Touristen-Informationssystem kann über eine zellenbasierte Ortung eine grobe Position des Nutzers ermitteln und die Plattform bietet sämtliche Informationen über aktive Landmarken an. Diese Landmarken unterstützen dann Operationen, die u.a. in HTML-Seiten eingebettet werden können. Beispielsweise unterstützen Landmarken die Anzeige einer Liste ihrer Nachbarn. So entsteht ein Netz aus Links und „Web-Seiten“, wobei die Verlinkungen den ortsbasierten Bezug herstellen. Somit handelt es sich um ein rein Graph-basiertes Umgebungsmodell, liefert aber eine sehr einfache und elegante Möglichkeit, Informationen zumindest für einfache ortsbezogene Anwendungen zu verwalten. Als besondere Eigenschaft ist dieses System für den Offline-Betrieb gedacht. Informationen werden nur über Nahfunk übertragen, und in diesem Sinne ist das Umgebungsmodell dann tatsächlich in der Umgebung integriert.

4.1.4.7 Nexus

Die Nexus-Plattform [Hohl 99] versucht eine einheitliche Infrastruktur für geospatiale Anwendungen bereitzustellen. Dabei wird das Konzept der „Augmented Area Models“ als eine Art hybrider Struktur, die sowohl virtuelle als auch physische Objekte verwaltet, eingeführt. Eine „Augmented Area“ ist in der Fläche beschränkt und kann so auf das Interesse einer bestimmten Anwendung oder Anwendergruppe beschränkt werden. Dabei unterscheiden sich virtuelle und physikalische Objekte nicht in ihrer Verwaltung oder Modellierung. Für digitale Dienste werden insbesondere „Virtual Information Towers“ vorgeschlagen. Das sind virtuelle Objekte, die Informationen enthalten, die dem Nutzer genau dann zur Verfügung gestellt werden, wenn er sich in der Nähe dieser virtuellen Objekte befindet. Diese Objekte kann die Nexus-Plattform dann einheitlich bearbeiten und aktualisieren, auch auf Grund von spatialen oder temporalen Ereignissen.
Etwas konkreter wird diese Plattform durch ein Schichtenmodell umgesetzt [Nick 01], in dem eine Föderation von „Spatial Model Servern“ und „Location Services“ über eine gemeinsame Middleware, die das „Augmented World Model“ zur Verfügung stellt, eine Schnittstelle für Applikationen anbietet, über die die Anwendungen mit der virtuellen und realen Umgebung interagieren können.
In gewisser Weise wird im Falle von Nexus kein Umgebungsmodell festgelegt, sondern lediglich durch die Spezifikation einer dreischichtigen Architektur eine Entkoppelung von Applikation, Dienstvermittlung und Modellverwaltung gefordert. Dass sich eine solche Architektur bisher nicht durchgesetzt hat, liegt offenbar daran, dass es in der Regel nur wenige Anbieter von ortsbezogenen Daten gibt, die sich gegenseitig nicht unterstützen wollen, und somit die Ver-

mittlungsschicht, die von den „Nexus-Nodes“ gebildet wird, keine Aufgabe hat.

4.2 Abstraktion des Raumbegriffes zur Verwendung von Bitmap-Karten

Die Anforderungen an ein Umgebungsmodell umfassen vielfach sowohl geometrische als auch topologische Anfragen. Dennoch ist es in der Praxis gängig, als Koordinatensystem die Bitmap-Koordinaten eines gemeinsam genutzen Raumplanes in Form einer Bitmap-Grafik zu verwenden, so etwa bei Ekahau [Ekah 12]. Die Wahl einer Bitmap als Datengrundlage ist damit begründet, dass gerade die kleinen Prozessoren von mobilen Endgeräten und auch die Frameworks und Betriebssysteme für diese Geräte sich schlecht für komplexe Datenstrukturen wie Vektorgrafiken eignen. Wegen der zentralen Bedeutung graphischer Benutzerschnittstellen sind aber die gängigen Bildbearbeitungsaufgaben auf Bitmaps überraschend performant möglich. Hinzu kommt die Tatsache, dass im Bereich des Internets und aller Browser Informationsdarstellungen in Form von Vektorgrafiken derzeit nur sehr selten zum Einsatz kommen. So basieren die großen Kartendienste Google Maps [Goog 12] oder auch die OpenStreetMap [Open 12a] auf Bitmap-Kacheln, welche die zu Grunde liegenden Karten in der für die aktuelle Zoom-Stufe optimalen Informationsdichte präsentieren, ohne dabei pro Anfrage serverseitige GIS-Berechnungen auszulösen. Wenn man davon ausgeht, dass auch für die kartographische Grundlage von ortsbezogenen Diensten in Gebäuden Bitmaps verwendet werden, so ergibt sich sofort der Vorteil, dass es weit verbreitete und frei erhältliche Tools zur Bearbeitung dieser Daten gibt. Jedes Bildbearbeitungsprogramm ist zunächst geeignet, solche Kartendaten zu erstellen und zu bearbeiten. Eine vorliegende Blaupause kann auch einfach eingescannt und mit ein wenig Nachbearbeitung als Modell verwendet werden.
Vor diesem Hintergrund wurde ein Bitmap-zentrisches Umgebungsmodell zur Benutzung für Indoor-Navigations-Anwendungen eingeführt [Wern 11a]. Der ausschlaggebende Vorteil zur Verwendung von Bitmaps als Träger des Umgebungsmodells besteht darin, dass diese sich für mobile Dienste auch deshalb eignen, weil ein derzeitiger Browser zumindest eine Anzeige vornehmen kann, während ein erweiterter Browser womöglich die in die Bilder eingebetten Informationen nutzen kann, um den Ort zu bestimmen oder eine Navigation vorzunehmen. Um allein in Bitmaps in effizienter Weise ein ausreichendes, semantisches Modell zu transportieren, werden einige Regeln festgelegt, wie solche Bitmaps interpretiert werden, und es wird eine Methode angegeben, mit der ausreichende Modelle zur Klassifikation von Symbolen wie Türen und zur Positionierung im Inneren von Gebäuden in die EXIF-Tags eingebettet werden.
Im Folgenden werden einige semantische Begriffe, die für die Modellierung des Inneren von Gebäuden relevant sind, durch Eigenschaften von Bitmaps

definiert. Dabei werden deutsche Versionen der englisch geprägten Begriffe aus der Arbeit [Wern 11a] verwendet, die in Klammern angegeben werden.

Definition 1:
Ein **Gebiet (Area)** ist eine Teilmenge der Pixel einer Bitmap.

Definition 2:
Der **Floodfill-Abschluss (Floodfill-closure)** eines Punktes (x, y) in einer Bitmap ist das Gebiet, das von einer gewöhnlichen 4-Nachbar-Floodfill-Operation am Punkt (x, y) gefüllt wird. Der 4-Nachbar-Floodfill ist dabei durch die Vorschrift definiert, dass die vier Nachbarn über, unter und neben dem aktuellen Punkt genau dann gefüllt werden, wenn sie dieselbe Farbe haben, wie die Farbe, die der Punkt (x, y) ursprünglich hatte. Für jeden Nachbarpixel, der gefüllt wurde, wird dann dieselbe Vorschrift rekursiv angewendet.

Definition 3:
Ein Gebiet ist **Floodfill-verbunden (Floodfill-connected)**, wenn es der Floodfill-Abschluss eines (und somit jedes) seiner Punkte ist.

Definition 4:
Ein **Raum (Room)** ist ein Floodfill-verbundenes Gebiet.

Definition 5:
Der **Außenbereich** ist der Floodfill-Abschluss des Punktes $(0, 0)$.

Definition 6:
Ein Raum R_1 ist **enthalten in (inside)** einem anderen Raum R_2, genau dann wenn R_1 Teilmenge der konvexen Hülle von R_2 ist.

Natürlich ergeben diese Definitionen nicht den gewöhnlichen Raumbegriff, und auch nicht die gewöhnliche Auffassung von Räumen, die in anderen Räumen enthalten sind. Aber es ist recht unwahrscheinlich, dass ein Raum, der nach obiger Definition in einem anderen Raum enthalten ist, nicht auch auf einem relativ kurzen Weg erreicht werden kann.
Um diesen Definitionen mehr semantische Korrektheit zukommen zu lassen, werden die folgenden drei Eigenschaften gefordert, die eine Bitmap für die direkte Verwendung als Raumplan nach obigen Definitionen haben muss.

- *Gebäudeabschluss (Closed-Building-Property)*: Das Gebäude ist durch schwarze Linien begrenzt und vollständig von Weiß umgeben. Im Wesentlichen bedeutet das, dass die Definition von *Außenbereich* tatsächlich den Außenbereich ergibt.

- *Türen-Sind-Räume (Doors-Are-Rooms)*: Eine Tür innerhalb eines Gebäudes ist so gezeichnet, dass sie selbst ein Raum im Sinne von Definiti-

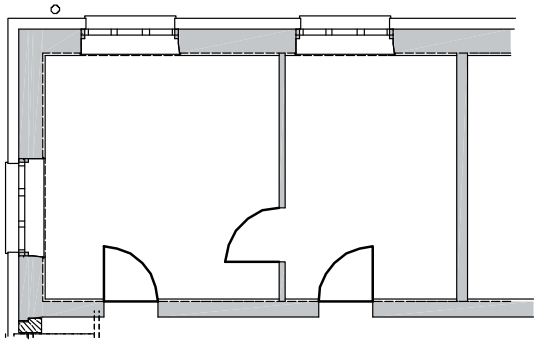


Abbildung 4.3: Eine „geschlossene Tür“ und eine typische Zeichnung einer „offenen Tür“

on 4 ist. Ein Beispiel für eine geeignete und eine ungeeignete Tür ist in Abbildung 4.3 zu sehen.

- *Begehbarkeit-Ist-Bekannt (Walkable-Space-Is-Known)*: Es gibt eine Möglichkeit, zu entscheiden, ob eine gerade Linie zwischen zwei Punkten begehbar ist.

Raumpläne, die den vorgegebenen Eigenschaften genügen, können verwendet werden, um Räume und Türen zu erkennen, einen Graphen zu erstellen, der alle Räume und Türen miteinander in der gewünschten Weise verbindet, auf dieser Ebene eine Positionierung und Routenberechnung durchzuführen und gegebenenfalls Textanweisungen zu erzeugen, und dann für die Visualisierung einen Weg zu finden, der ganz in der begehbaren Fläche des Modells enthalten ist, dessen Dreh- und Wendepunkte scharf sind und der gute Visualisierungseigenschaften hat.
Wie bereits in Abschnitt 4.1.3 beschrieben, sollte ein Umgebungsmodell eine gewisse Anzahl von Anfragen unterstützen. Entsprechend [Beck 05] sind die wichtigsten Anforderungen die Folgenden: Positionen von Objekten müssen modellierbar sein, möglichst mit symbolischen oder geometrischen Koordinaten. Distanzen zwischen Objekten müssen messbar sein. Die topologischen Beziehungen des „Enthalten-Seins“ und „Verbunden-Seins“ müssen modellierbar sein.
Geometrische Koordinaten werden im vorliegenden Modell als Pixel-Koordinaten spezifiziert, eine globale Positionierung, insbesondere über Kacheln hinweg, ist dann möglich, indem in den EXIF-Tags des Bildes ausreichende Referenzpunkte zum Beispiel in WGS84-Koordinaten definiert sind. Symbolische Koordinaten beziehen sich im vorliegenden Fall auf Räume und werden ebenfalls als Pixel-Koordinaten gespeichert. Zum symbolischen Ort gehören alle Orte der Floodfill-verbundenen Umgebung dieses Punktes.
Distanzen innerhalb einer Bitmap werden durch den euklidischen Abstand oder auch die Länge eines kürzesten Weges in der begehbaren Fläche gemessen, Entfernungen zwischen Orten in verschiedenen Bitmaps werden nach Übersetzung

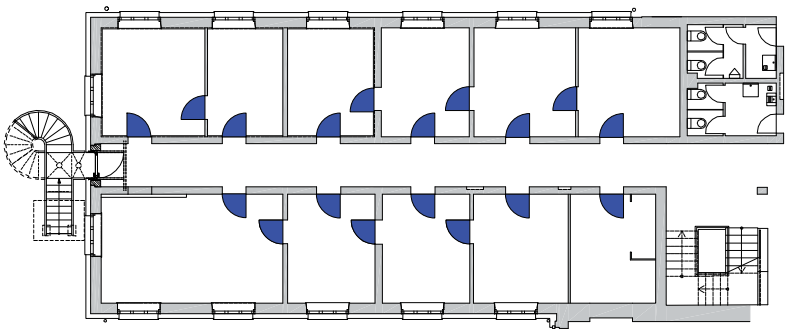


Abbildung 4.4: Ausschnitt eines Trainings-Bildes mit blau markierten Türen

in ein globales Koordinatensystem wie WGS84 in diesem berechnet.
Die topologische Beziehung des „Enthalten-Seins“ wird durch die konvexe Hülle im Sinne von Definition 6 modelliert. Es bleibt im Grunde nur die topologische Beziehung des „Verbunden-Seins“, die sich – bei ausreichend homogen gezeichneten Plänen – aus der Kartensymbolik ableiten lässt.

4.2.1 Automatisierte Erstellung eines Navigationsgraphen

Ein Navigationsgraph ist in diesem Zusammenhang ein Graph, der eine gewisse Menge an Ecken enthält, denen alle möglichen geometrischen und symbolischen Koordinaten zugeordnet werden können, und der zwischen zwei Ecken eine Kante enthält, wenn eine direkte, begehbare Verbindung zwischen diesen beiden Ecken besteht.
Der grundlegende Navigationsgraph in diesem Zusammenhang besteht aus einer Ecke pro Raum – also auch einer Ecke pro Tür, denn Türen sind ja in diesem Zusammenhang Räume, vgl. Definition 4 – und einer Kante zwischen benachbarten Räumen, falls einer der Räume eine Tür ist.
Da es durch den Floodfill-Zusammenhang relativ einfach ist, eine Liste an Räumen zu erstellen, benötigt das vorliegende Modell noch eine Methode, um Türen von sonstigen Räumen zu unterscheiden.

4.2.1.1 Erkennung von Türsymbolen in Bitmap-Karten

Für die Erkennung von Türsymbolen wird ein Klassifikationsmodell benötigt, das klein genug ist, um in den EXIF-Tags der Bilder platziert zu werden, das aber in der Lage ist, zwischen Türen und sonstigen Räumen effektiv zu unterscheiden. Dafür werden die einfacheren Algorithmen des maschinellen Lernens verwendet, weil diese gute Ergebnisse bei guter Verständlichkeit und mitunter sehr kleinen Modelldaten liefern. Dazu werden im Folgenden einige numerische Attribute zusammengestellt, die ausreichen, um die typischen Türsymbole von sonstigen Elementen eines Raumplanes zu unterscheiden. Diese Attribute werden dann automatisiert für jeden Raum aus dem Plan extrahiert, und ein Trainingsbild mit bereits markierten Türen verwendet, um ein Klassifikations-

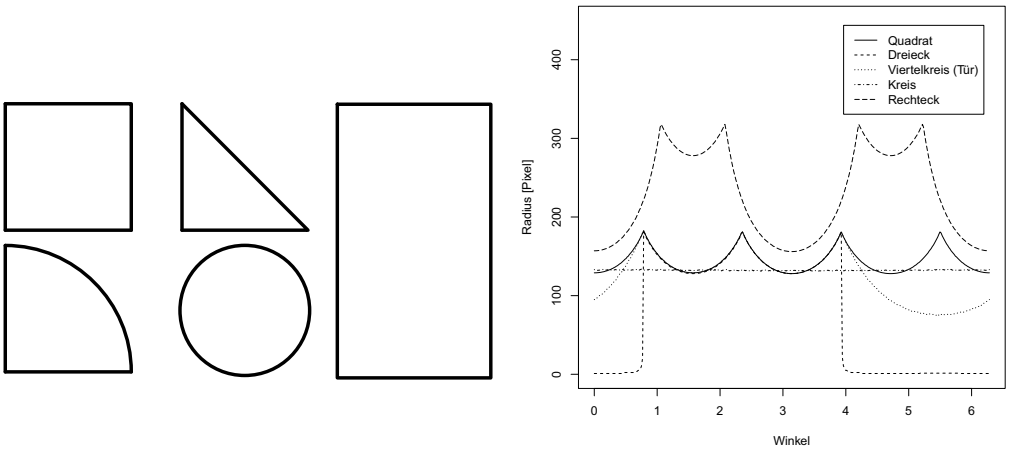


Abbildung 4.5: Das Ergebnis des radialen Sweepline-Algorithmus für einige einfache Formen

modell zu trainieren. Ein Beispiel für ein solches Trainingsbild ist in Abbildung 4.4 zu sehen.

Dabei wurde bei der Auswahl der Features Wert darauf gelegt, dass sie stabil sind gegen Rotationen, weil Türen in beliebiger Lage im Gebäude auftreten können. Sie sind aber absichtlich nicht skaleninvariant gewählt, weil das Modell ja ohnehin in der Bitmap gespeichert werden soll, und sich damit die Größenverhältnisse zwischen Training und Anwendung nicht unterscheiden.

Zunächst sind sehr grundlegende Features verwendet worden, weil diese bereits eine wichtige Unterscheidungskraft haben. So wird etwa die Fläche als numerisches Attribut verwendet. Die Fläche eines Raumes ist dabei die Anzahl der Pixel des Floodfill-verbundenen Gebietes. Ebenso wird die Breite und Höhe des minimalen, umgebenden Rechtecks verwendet.

Da diese Attribute noch nicht ausreichen, um alle Türen von sonstigen Räumen zu unterscheiden, wurde ein spezieller Sweepline-Algorithmus entworfen, der einige zusätzliche Informationen über konvexe Objekte sammelt.

4.2.1.2 Radialer Sweepline-Algorithmus

Im Folgenden wird zur einfacheren Darstellung davon ausgegangen, dass eine solche Region weiß gefüllt und durch schwarze Pixel begrenzt ist. Zu einer solchen Region wird zunächst ein sinnvoller Mittelpunkt bestimmt, etwa als der Mittelpunkt eines minimalen, umgebenden Rechtecks. Von diesem Mittelpunkt aus wird dann in festen Winkelabständen ein Strahl ausgehend vom Mittelpunkt abgelaufen und die erste Kollision mit den begrenzenden schwarzen Pixeln berechnet.

Algorithmus	**Datensatz 1**	**Datensatz 2**	**Datensatz 3**
RIPPER [Cohe 95]	99.85%	97.47%	99.91 %
Naive Bayes [Geor 95]	99.37%	99.15%	99.67 %
C4.5 [Quin 93]	99.85%	97.89%	99.90 %

Tabelle 4.1: Erfolgsrate der Türerkennung verschiedener Algorithmen des maschinellen Lernens

So entsteht eine Diskretisierung der Abbildung

$$r : [0, 2\pi] \to \mathbb{R},$$

die einen Winkel auf den Radius der derzeitigen Fläche ausgehend vom Mittelpunkt in Richtung des Winkels abbildet. Diese Abbildung ist dann von Natur aus eine 2π-periodische Funktion.
Abbildung 4.5 zeigt einige einfache Formen und das Ergebnis des soeben beschriebenen Algorithmus. Man erkennt, dass diese Vorgehensweise sehr viele Informationen über die Fläche sammelt, so denn der Mittelpunkt im Inneren der Fläche liegt. Denn die Fläche lässt sich aus den gesammelten Daten im Wesentlichen rekonstruieren.
Für die Aufgabe, Türsymbole von anderen Symbolen zu unterscheiden, reichen schon die sehr einfachen Attribute aus, die aus der Funktion r abgeleitet werden: Der maximale und der minimale Radius (radmax, radmin) und deren Quotient (sq) als Maß für die Abweichung von einem Kreis. In den Fällen, wo die Abbildung r den Wert 0 hat und somit der Quotient nicht definiert ist, wird ein fehlender Wert für den Datensatz zur Klassifikation deklariert.
Mit diesem Featureset wurden drei einfache Klassifikationsalgorithmen trainiert, die jeweils eine andere Wissensabstraktion verwenden. Die Erfolgsrate dieser Algorithmen wurde auf einem Stockwerk des Universitäts-Gebäudes mit zehnfacher stratifizierter Kreuzvalidierung gemessen. Sie ist extrem hoch und unabhängig von der Wissensabstraktion vergleichbar, wie in Tabelle 4.1 zu erkennen.
Dazu wurden vier verschiedene Datensätze verwendet, aus denen alle Räume im Sinne von Definition 4 extrahiert wurden:

- *Datensatz 1:* Ein Flügel des Universitätsgebäudes (238 Räume, davon 36 Türen)
- *Datensatz 2:* Der gleiche Flügel des Gebäudes um 20° rotiert (2447 Räume, davon 36 Türen) Die hohe Anzahl der Räume entsteht hierbei durch die Rotation durch Anti-Aliasing. Es entstehen sehr kleine, nicht-schwarze Bereiche, die von einem 4-Floodfill nicht verlassen werden können.
- *Datensatz 3:* Das vollständige Stockwerk des Gebäudes (Abbildung 4.6, 1755 Räume, davon 263 Türen)

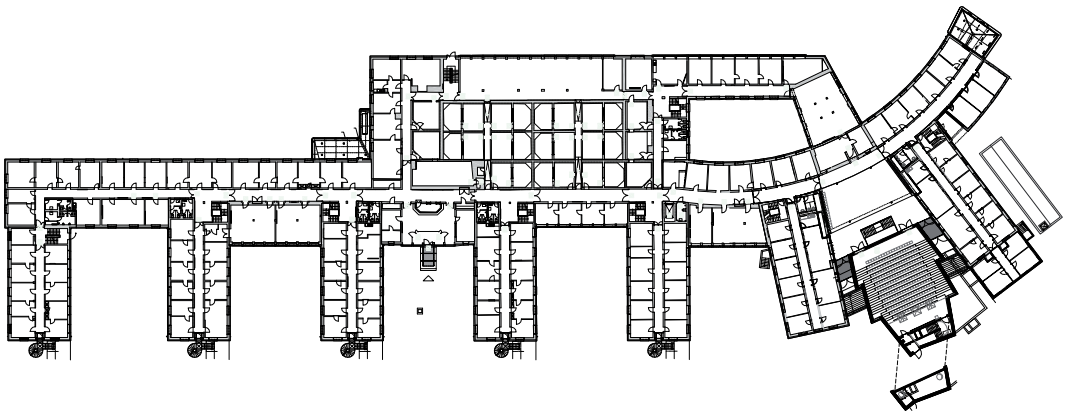


Abbildung 4.6: Der gesamte Gebäudeplan

Da es keine signifikanten Unterschiede in Bezug auf die Klassifikationsleistung zwischen den verwendeten Algorithmen gibt, und das hohe Niveau der Erfolgsrate für die vorgesehene Anwendung ausreicht, kann der Klassifikationsalgorithmus danach ausgewählt werden, welche Wissensabstraktion für die Anwendung am Besten geeignet ist. Dabei wurde der Regel-Generator RIPPER [Cohe 95] ausgewählt. Denn dann lässt sich das Regelset einfach als ASCII-String speichern und in die EXIF-Daten des Bildes einbetten. So entsteht eine Bitmap-Datei, die die notwendigen Informationen zur Erkennung von Türen schon enthält. Man hätte zwar sicher auch den Entscheidungsbaum in einen ASCII-String oder gar in eine Regelmenge übersetzen können, einzig die Bayes'sche Wissensabstraktion ist ein bisschen unhandlich und vor allem für den Nutzer unverständlich. Klassifiziert eine gewisse Regelmenge einen Raum falsch, so kann dies meist durch manuelles Betrachten der Features und eine geringfügige Veränderung korrigiert werden. Alternativ kann natürlich auch noch eine Liste mit Ausnahmen in Form von Pixelkoordinaten geführt und gespeichert werden.
Da das Regelset ja ohnehin fest an ein Bild gekoppelt wird, ist „Overfitting" erwünscht. Entsprechend wurden Pruning-Methoden abgeschaltet und man erhält für das gesamte Gebäude ein Regelset in der Größenordnung von 230 Bytes ASCII-Text, das sich problemlos in die EXIF-Daten des Bildes integrieren lässt.

4.2.1.3 Bestimmung benachbarter Räume

Nachdem jetzt alle Räume aufgezählt sind, und zwischen Räumen und Türen effektiv unterschieden werden kann, benötigt man zur Erzeugung des Navigationsgraphen auf Raumebene nur noch eine Methode, die zu einem Raum dessen Nachbarräume findet. Um Rechenleistung zu sparen, wird an dieser Stelle eine Variante des radialen Sweepline-Algorithmus verwendet, der ja ohnehin schon für die Klassifikation ausgeführt werden muss.
Da es ausreicht, nur die Nachbarn von Türen zu berechnen, ist es dabei auch

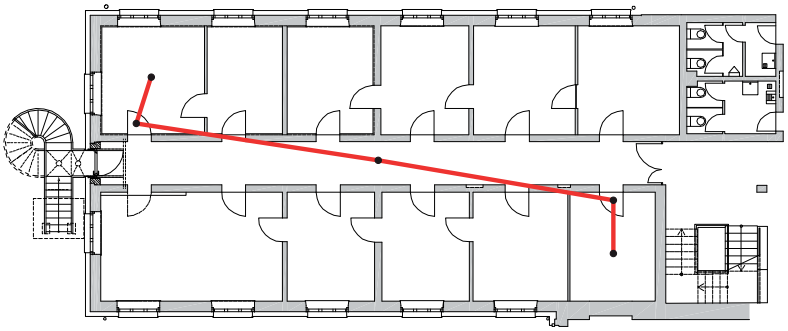
nicht relevant, dass dieser Algorithmus nur auf konvexe Räume sinnvoll anwendbar ist. Insbesondere trifft nämlich jeder Strahl des radialen Sweepline-Algorithmus auf die Begrenzungslinie des Raumes. Um die Nachbarn eines Raumes zu berechnen, wird diese schwarze Begrenzung übersprungen, und der erste Punkt festgehalten, der zu einem anderen Raum gehört, beispielsweise, indem der betreffende Pixel mit einer nirgends sonst verwendeten Farbe markiert wird.
Nachdem der radiale Floodfill berechnet wurde, wird mit Floodfill-Operationen bestimmt, welche Räume markiert worden sind, und so entsteht im Idealfall eine Liste der Nachbarn. Diese Nachbarn sind zwar nicht unbedingt durch eine Tür verbunden, aber für die Erzeugung des Navigationsgraphen werden ja ohnehin nur die Nachbarn von Räumen verwendet, die nachweislich Türen sind. Das einzige Problem entsteht dann, wenn die Auflösung der Karte so schlecht ist, dass ein Strahl eine Tür verlässt und über eine Raum-Ecke hinweg einen „falschen“ Raum trifft. Dies ist aber sehr selten und kann mit einem beliebigen Zeichenprogramm leicht korrigiert werden.

4.2.1.4 Erzeugung des Navigationsgraphen

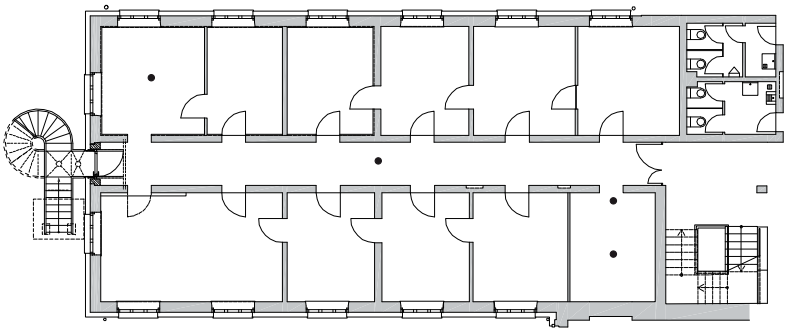
Es ist nun durch Kombination der vorgestellten Algorithmen sehr einfach, einen Navigationsgraphen zu erstellen. Es ist durchaus sinnvoll und denkbar, dass dieser Graph erst dann erzeugt wird, wenn ein Suchalgorithmus wie Dijkstra oder A^* einen bestimmten Raum betritt. So ist es möglich, dass nur für einen kleinen Teil der Karte die Klassifikationsalgorithmen und die Nachbarbestimmungen ausgeführt werden. Dabei wird vom bekannten Startraum aus die Menge der Nachbarräume bestimmt. In dieser Menge werden dann die Türen klassifiziert und eine Kante eingefügt, wenn einer von beiden Räumen eine Tür ist.
Obwohl der so erzeugte Navigationsgraph gut geeignet ist, um Navigationsanweisungen zu generieren, weil er nur wesentliche Änderungen des Aufenthaltsortes enthält und somit zu jeder Ecke im kürzesten Weg auch eine Anweisung enstehen sollte, gehen die Kanten typischerweise durch Wände, und eine Visualisierung ist kaum möglich. Abbildung 4.7(a) zeigt diese Situation.

4.2.2 Entfernung von Türen

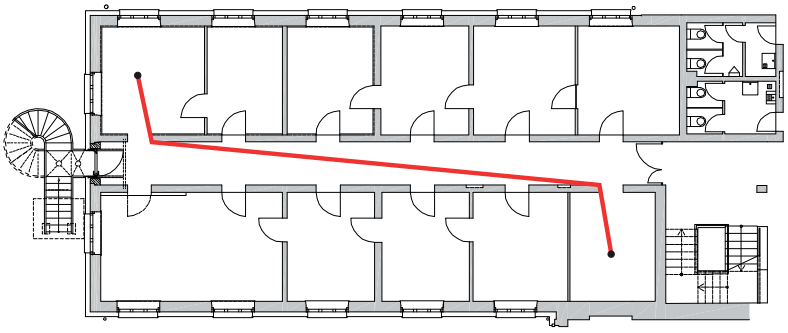
Nachdem nun mit dem erzeugten Navigatiosgraphen ein kürzester Weg berechnet werden kann, benötigt man noch ein Verfahren, diesen Weg vernünftig zu visualisieren, also zunächst einmal in die Karte einzuzeichnen.
Da bekannt ist, welche Räume im kürzesten Weg Türen sind und deren minimales, umgebendes Rechteck ermittelt wurde, können die Türen einfach aus der Karte entfernt werden. Dazu wird das minimale umgebende Rechteck weiß gefüllt. In der resultierenden Karte (Abbildung 4.7(b)) ist nun die Vereinigung der am kürzesten Weg beteiligten Räume Floodfill-verbunden. Nun wird



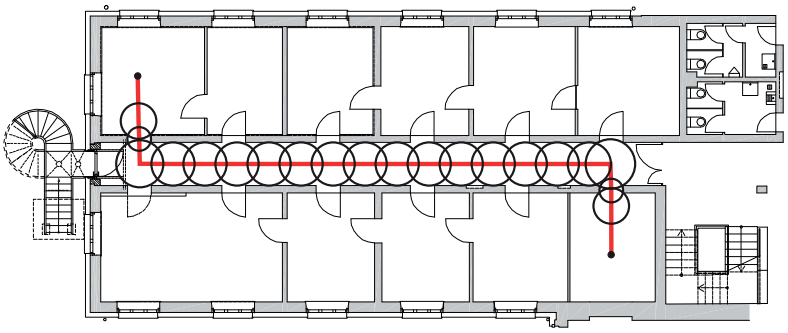
(a) Kürzester Pfad auf Raumbasis



(b) Türentfernung



(c) Kürzester Pfad in begehbarer Fläche



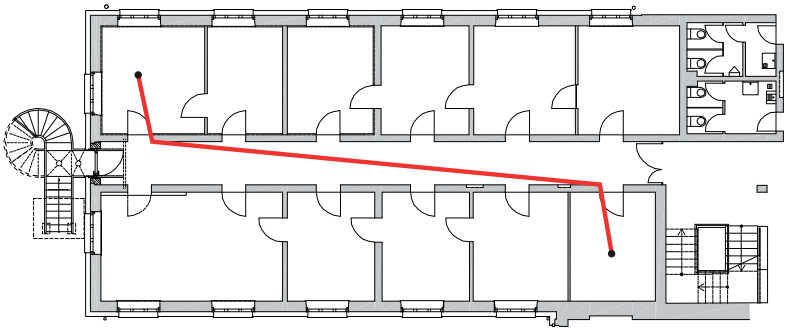
(d) Ergebnis des Growing-Circle-Algorithmus

Abbildung 4.7: Visualisierung eines raumbasierten kürzesten Wegs

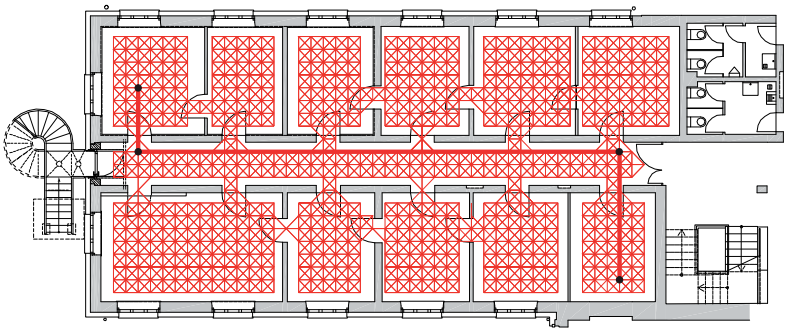
nochmals der kürzeste Weg zwischen Start- und Endpunkt berechnet, allerdings auf Pixelebene. Dazu wird implizit die Graphenstruktur verwendet, die jeden weißen Pixel mit den acht Nachbarn verbindet, die ebenfalls weiß sind. Auf dieser Pixelebene berechnet man dann einfach einen kürzesten Weg. Dieser Weg verbindet, genau wie der vorher berechnete Weg auf Raumebene, den Anfangs- und den Endpunkt. Dabei liegt aber – und das ist der große Vorteil – der Pfad innerhalb tatsächlich begehbarer Fläche. Ein solcher Pfad wird allerdings immer noch sehr schlecht für eine Visualisierung geeignet sein, da er sich typischerweise sehr nahe an Wänden entlang bewegt (Abbildung 4.7(c)). Doch für solche sehr schlechten Wege wurde in [Wern 11a] ein Verfahren entwickelt, mit dem sich die Visualisierungsqualität verbessern lässt. Da dieses Verfahren auch auf anders entstandene Wege mit schlechten Visualisierungseigenschaften übertragen werden kann, wird dieser letzte Schritt, der dann auch zu Abbildung 4.7(d) führt, im folgenden Abschnitt gesondert beschrieben.

4.2.3 Visualisierung von Wegen mit dem „Growing-Circle-Algorithmus“

Wegpunktgraphen führen häufig zu schlecht darstellbaren Wegen, da entweder zu wenige Kanten im Graphen enthalten sind, um eine schöne Visualisierung zu erhalten, oder so viele Kanten enthalten sind, dass sich kürzeste Wege immer sehr nahe an begrenzender Geometrie befinden. Selbst dann, wenn viel Wert darauf gelegt wird, dass innerhalb eines Graphen schöne Wege enthalten sind, sind diese nicht so leicht von den weniger schönen Wegen zu unterscheiden.
Mit dem „Growing-Circle-Algorithmus“, der in diesem Abschnitt vorgestellt wird, ist es möglich, unabhängig von der Qualität des Graphen und mit akzeptablem Aufwand kürzeste Wege so zu überarbeiten, dass sie gut visualisiert werden können. Der Algorithmus besteht im Wesentlichen aus einer Folge von parallelisierbaren Operationen auf der Bitmap und eignet sich daher von Hause aus für eine Berechnung auf dedizierter Grafikhardware, wie sie neuerdings auch in den ersten Smartphones verbaut wird. Allerdings ist eine solche Unterstützung durch Hardware nicht unbedingt notwendig.
Der vorliegende Algorithmus benötigt als Eingabe einen Weg als geordnete Liste von Koordinaten und eine Kollisionskarte. Eine Kollisionskarte ist dabei eine Bitmap, die Hindernisse mit schwarzen Pixeln und freie Flächen mit weißen Pixeln markiert. Die konfigurierbaren Parameter des Algorithmus umfassen eine maximale Strecke, die zu Beginn zwischen zwei Wegpunkten liegen darf, eine maximale Strecke, die ein Wegpunkt vom Algorithmus bewegt werden darf, und eine Bewertung, die entscheidet, welche Wegpunkte in welcher Weise bewegt werden dürfen.
Der Algorithmus geht wie folgt vor: Zunächst wird der vorgegebene Weg so tesselliert, dass die Vorgabe des maximalen Abstandes erfüllt ist. Dabei werden Wegabschnitte, deren Länge größer als die Vorgabe ist, in der Mitte geteilt, bis alle Wegabschnitte kürzer als die Längenvorgabe sind. Dann werden diese Weg-



(a) Ein kürzester Weg in einem Corner-Graphen



(b) Ein kürzester Weg in einem gitterbasierten Graphen

Abbildung 4.8: Visualisierungsprobleme von Wegpunktgraphen

punkte wie im Folgenden beschrieben bewegt, um die Visualisierungsqualität zu verbessern.
Für jeden Wegpunkt wird die Menge an Punkten bestimmt, zu denen er bewegt werden kann. Zu diesem Zweck werden Kreise um jeden Punkt der durch einen Parameter beschränkten Umgebung des aktuellen Punktes so lange vergrößert, bis diese mit Geometrie in der Karte kollidieren.
Aus dieser Menge werden dann die Orte mit maximalem Radius ausgewählt. Falls mehrere Orte diesen maximalen Radius erlauben, wird derjenige ausgewählt, der am Nächsten an der ursprünglichen Position liegt. Eine zu überwachende Nebenbedingung besteht darin, dass die Strecken zu den jeweilig benachbarten Wegpunkten keine Kollisionsgeometrie treffen.
Der Algorithmus wurde auf einem aktuellen Android-Smartphone (HTC-Desire) als native C-Bibliothek implementiert und verwendet einen Bildpuffer der Anwendung. Zu diesem Zweck wurde eine handliche, schnelle und stabile Bildbearbeitungsbibliothek für Android entwickelt. Die Geschwindigkeit des Systems ist im Wesentlichen zufriedenstellend, insbesondere vor dem Hintergrund, dass gerade die Grafikleistung sowohl der Hauptprozessoren als auch in Bezug auf dedizierte Stream-Computation-Hardware sich in der nächsten Zeit für Smartphones ganz massiv steigern wird.
Abbildung 4.9 zeigt die Laufzeit des Systems auf dem Screen-Puffer eines HTC Desire Smartphones. Dabei zeigt die X-Achse die Anzahl der Wegpunkte, die

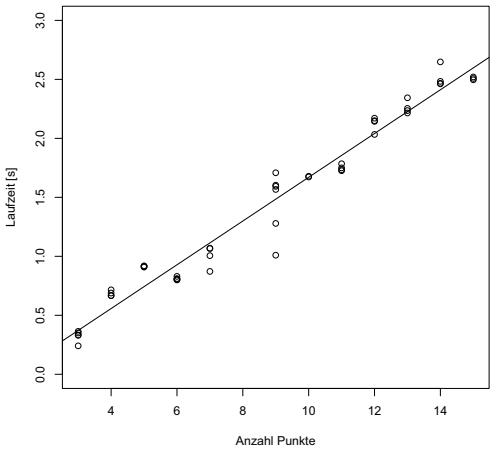


Abbildung 4.9: Laufzeit des Growing-Circle-Algorithmus auf dem Screenpuffer eines HTC-Desire

zufällig in der Freifläche gewählt wurden, und die Y-Achse die Berechnungszeit in Sekunden. Man erkennt, dass selbst bei fünfzehn sichtbaren Wegpunkten die Laufzeiten weniger als $3s$ betragen. Der im wesentlichen lineare Anstieg der Laufzeit mit der Anzahl der Wegpunkte liegt darin begründet, dass für die meisten Punkte dieselbe Anzahl an Berechnungen vorgenommen wird. Die Laufzeiten im Bereich weniger Sekunden sind vor dem Hintergrund zu verstehen, dass einige offensichtliche Optimierungen ausgelassen wurden, um die Allgemeinheit der Implementierung zu bewahren. So ist für die grundsätzliche Funktionsweise der Kollisionskarte eine 1-Bit-Grayscale-Bitmap ausreichend. Für die Bestimmung der maximalen Radii könnte der Kreis auch durch ein Rechteck ersetzt werden. Die Anfrage, ob ein Rechteck in der Kollisionskarte leer ist, kann dann durch die Verwendung von integralen Bildern in konstanter Zeit beantwortet werden. Weiter ließe sich Rechenzeit sparen, indem das Wachsen der Radii in exponentieller Weise mit Backtracking erfolgt, also der Radius in jedem Schritt zunächst verdoppelt wird, und dann bei der ersten Kollision der letzte nicht-kollidierende Kreis linear vergrößert wird. Wenn der Speicherbedarf einer zusätzlichen Bitmap oder eines zusätzlichen Farbkanals toleriert wird, kann darin auch pro Punkt der maximale Radius fest gespeichert und so im Betrieb in konstanter Zeit ermittelt werden.

4.2.4 Zusammenfassung

In diesem Abschnitt wurde ein minimalistisches, Bitmap-zentrisches Umgebungsmodell vorgeschlagen, mit dem sich eine vollständige Indoor-Navigation umsetzen lässt. Dabei wurden einfache Klassifikationssysteme für Türsymbolik verwendet und eine Visualisierungsheuristik konstruiert, die insgesamt eine Navigation und Wegevisualisierung auf der Bitmap erlauben. Selbstverständ-

lich und in offensichtlicher Weise kann das System auch auf mehrere Etagen erweitert werden. Hierzu ist eine Objekterkennung auf Bitmap-Ebene nicht ausreichend, da die Bedeutung der Verbindungen zwischen Etagen nur sehr schwierig automatisiert extrahiert werden kann. Dennoch kann in den EXIF-Daten eine Liste vertikaler Verbindungen zwischen Räumen gespeichert werden. Einzig zu beachten ist dabei, dass die Knoten auf beiden Ebenen in der begehbaren Fläche eines Raumes liegen müssen, und dass dann natürlich nicht jeder zweite Raum im Navigationsgraph eine Tür sein muss.

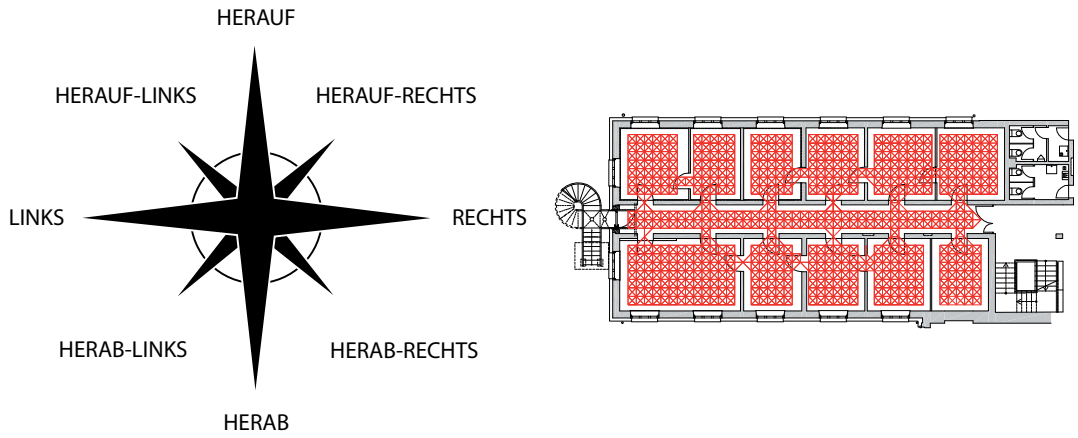
4.3 Semantische Overlays zur Ergänzung von Kartenmaterial

Im Projekt HIPS bestand ein wesentliches Problem darin, eine Indoor-Navigation zu ermöglichen, die bestehende CAD-Daten so verwendet, dass nicht bei jeder kleinen Änderung im CAD-Datenpool eine manuelle Überprüfung und Ergänzung eines Umgebungsmodelles nötig wird. Denn der Flughafen München verwendet die CAD-Datenbasis mit einigen internen Tools zum erweiterten Gebäudemanagement. So ist für alle Gebäude des Flughafens ein gemeinsames Koordinatensystem definiert worden und alle Tätigkeiten, die das Gebäude betreffen, werden in einer CAD-Umgebung gepflegt.
Für die Indoor-Navigation benötigt man zusätzliche Informationen über ein Gebäude, die für die Navigation nützlich sind. So ist es beispielsweise an einem Flughafen wichtig, bei der Auswahl der schnellsten Route auch Bauelemente wie Rolltreppen und Rollbänder zu kennen. Nur so kann man Routen mit solchen Bauelementen anderen Routen vorziehen, die womöglich in Bezug auf die geometrische Distanz etwas kürzer sind. Dabei ist zu beachten, dass der Einsatz von Indoor-Navigations-Technologie sich nur dann durchsetzen kann, wenn die notwendigen Daten in einer vernünftigen Zeit mit einem vernünftigen Aufwand erzeugt werden können.
Als Datenquelle liegen in der Praxis häufig CAD-Zeichnungen vor, die für den Bau oder auch einen Umbau des Gebäudes erstellt worden sind. Diese Informationen sind natürlich meist nur für den Konstruktionsprozess verwendet worden und beinhalten daher selten eine ausreichende semantische Struktur, die eine Indoor-Navigation direkt ermöglicht. Einige Forschungsarbeiten schlagen zwar vor, die notwendige semantische Information bei der Erstellung der digitalen Zeichnungen im CAD-System zu erstellen. Es ist aber nach Meinung des Autors wenig wahrscheinlich, dass solche Informationen von Architekten erstellt werden, da diese in der Regel nur am Bau und nicht am Betrieb eines Gebäudes beteiligt sind, und daher lediglich Mehraufwand durch eine solche Modellierung hätten.
Darüber hinaus liegen bisweilen auch nur Bitmap-Karten, beispielsweise eingescannte Blaupausen, der Gebäude vor. Diese können keine ausreichenden organisatorischen Informationen enthalten, wie sie beispielsweise in modernen

CAD-Plattformen modelliert werden könnten. Um auch in dieser Situation ohne besonderen Mehraufwand wie die Neuerstellung von CAD-Daten eine Indoornavigation anzubieten, wurden die notwendigen semantischen Informationen getrennt von der Datenbasis in einheitlicher Form modelliert.
Vor diesem Hintergrund wurde ein Prozess vorgeschlagen, in dem eine strikte Trennung von ergänzender Semantik und Kartenmaterial zur gewünschten Unabhängigkeit der Navigationsapplikation von der Art, Qualität und Änderungshäufigkeit der Datenbasis führt. Mit der vorgeschlagenen Vorgehensweise ist es möglich, mit geringem Aufwand auch großflächige Indoor-Navigations-Anwendungen zu erstellen, fehlerhafte CAD-Daten zu tolerieren und gegebenenfalls unabhängig von der zu Grunde liegenden Datenbasis zu korrigieren. Insgesamt entsteht ein Umgebungsmodell, auf dessen Basis ein hochperformantes Navigationssystem entwickelt wurde, das seit dem 6. Juni 2011 am Flughafen München im produktiven Einsatz ist. Dazu werden, zusätzlich zu den zu Grunde liegenden Zeichnungen, eine Menge zusätzlicher, polygonaler Regionen („Overlays“) modelliert, die ihre Fläche mit einer gewissen Semantik verknüpfen. Solche Overlays können Bereiche gewöhnlicher Begehbarkeit genauso markieren, wie Bereiche mit schnellerer Fortbewegung und eingeschränkter Bewegungsrichtung wie Laufbänder und Rolltreppen.
Im Projekt HIPS wurde mit der Anwendung GraphEdit2 eine Umgebung geschaffen, die es ermöglicht, bestehende unstrukturierte Gebäudedaten – CAD-Dateien oder Bitmaps – mit semantischen Informationen zu ergänzen, die eine Verwendung für eine Navigationsanwendung erlauben. Darüber hinaus wurden Algorithmen integriert, die es ermöglichen, wichtige, geometrische Objekte innerhalb von Karten automatisch zu erkennen, und die entsprechende semantische Information zu erzeugen.
Der nächste Abschnitt beschreibt, wie das Navigationssystem ein gitterbasiertes Navigationsnetz erstellt, und welche Schnittstelle zum Kartenmaterial dabei verwendet wird. Im darauf folgenden Abschnitt wird dann beschrieben, wie fehlerhafte CAD-Daten mit semantsichen Overlays korrigiert werden, und es wird eine Übersicht über die am Flughafen München verwendeten Overlays und deren Semantik gegeben.

4.3.1 Zugangsgraphen auf Basis des Eight-Corner-Systems

Als Suchraumabstraktion verwendet das Navigationssystem HIPS einen großen, gitterbasierten Navigationsgraphen niedrigen Verzweigungsgrades, der auf dem „Eight-Corner-System“ beruht. Diese Struktur wurde gewählt, weil das Navigationssystem deshalb die gesamte Fläche abdecken soll, weil zum Zeitpunkt der Erstellung der Navigationsdaten die Points-of-Interest nicht bekannt sind. Viele andere Arbeiten, wie auch der ursprüngliche Prototyp [Rupp 09] dieses Systems, verwenden zur Navigation ein möglichst kleines Netzwerk, das alle Points-of-Interest miteinander verbindet und möglicherweise noch gute



(a) „Eight-Corner-System“ zur Klassifikation von Richtungen

(b) Ein Zugangsgraph

Abbildung 4.10: Das „Eight-Corner-System“ und ein mit dieser Struktur erzeugter Zugangsgraph

Visualisierungseigenschaften für kürzeste Wege hat. Die Navigation auf solchen Daten ist zwar wesentlich einfacher, widerspricht aber der Intention, die Points-of-Interest zu einem späteren Zeitpunkt beliebig zu verändern.

Ein weiterer Vorteil eines moderat hoch aufgelösten Gitters liegt darin, dass in jedem Weg die Wegpunkte mit einem gleichmäßigen und kleinen Abstand auftreten, sodass etwa die Suche nach Landmarken oder auch die Visualisierung mit Methoden wie dem „Growing-Circle-Algorithmus“ aus Abschnitt 4.2.3 wesentlich einfacher wird.

Als Grundstruktur kommt das sogenannte „Eight-Corner-System“ zum Einsatz. Dabei unterscheidet man zwischen acht Richtungen wie in Abbildung 4.10(a) dargestellt. Der Verzweigungsgrad einer Suche im Graphen beträgt entsprechend höchstens 8 in der Freifläche. Diese Grundstruktur, die für die Vernetzung eines Graphen verwenden wurde, dessen Ecken ein reguläres Gitter bilden, entspricht von der Feinheit auch der Struktur, mit der häufig die Richtung in textuellen Anweisungen spezifiziert wird. Eine Kante wird natürlich nur dann dem Gitter hinzugefügt, wenn sie nicht mit einer Wand kollidiert, ein Beispiel für einen solchen Zugangsgraphen ist in Abbildung 4.10(b) dargestellt.

Damit ist offensichtlich, dass das mit semantischen Informationen erweiterte Umgebungsmodell eine Anfrage unterstützen muss, die überprüft, ob eine gerichtete Strecke zwischen zwei Punkten mit Geometrie kollidiert, die einer Begehbarkeit der Strecke widerspricht. Obwohl in der Karte typischerweise die Informationen implizit enthalten sind, welche Strecken innnerhalb einer begehbaren Fläche liegen und welche nicht, lassen sich solche Informationen in dieser Allgemeinheit kaum verarbeiten. So gibt es viele Linien in einer typischen CAD-Zeichnung, die durchschritten werden können. Dies gilt für Türlinien genauso wie für gegebenenfalls vorhandene Gebäudeumrisse. Deshalb

werden diese Informationen mit semantischen Overlays korrigiert und ergänzt, welche die in ihnen enthaltene Geometrie als begehbar markieren.

4.3.1.1 Erzeugung der Navigationsgraphen

Ausgehend von einem Startpunkt, der im Inneren des Gebäudes gewählt wird, verwendet die Grapherzeugung einen rekursiven Algorithmus, der für Strecken in die acht Richtungen des „Eight-Corner-Systems“ einen Kollisionstest ausführt. In dem Fall, in dem keine Kollision auftritt, wird der Endpunkt der Strecke als Knoten im Navigationsgraph aufgenommen und eine neue gerichtete Kante entsprechend der Strecke in den Navigationsgraphen einfügt und mit dem Endpunkt der Strecke fortgefahren. Dabei kann jede Kante nur einmal eingefügt werden, die Rekursion terminiert daher, wenn die Ausbreitung des Algorithmus durch ein Polygon beschränkt wird, das den Startpunkt enthält. Hierzu eignen sich beispielsweise die Bounding-Box bei CAD-Daten oder die durch die Größe gegebenen Grenzen einer Bitmap.

Diese Form der Erzeugung ist zwar weit weniger effizient als eine Methode, die ein globales Gitter mit legalen Kanten auffüllt, hat aber den entscheidenden Vorteil für den Nutzer, dass der Graph grundsätzlich zusammenhängend ist und – bei entsprechend gebremster Erzeugung – die Bewegung durch die Karte verfolgt werden kann. So können Fehler bei der Erzeugung sofort erkannt werden. Auch kann dieser Algorithmus in einem bestimmten Gebiet reaktiviert werden, falls Teile des Graphen gelöscht wurden. So können lokale Überarbeitungen des Navigationsgraphen sehr einfach ermöglicht werden. Abbildung 4.10(b) zeigt ein Beispiel eines automatisch erstellten Graphen. Dieser Algorithmus zur Grapherzeugung benötigt neben einem Startpunkt auch eine feste Kantenlänge l für die Berechnung der benachbarten Gitterpunkte. Für diagonale Kanten wird dabei natürlich der Wert $\sqrt{2}l$ verwendet.

Eine besondere Eigenschaft eines so erzeugten Navigationsgraphen verdient besondere Aufmerksamkeit: Die Ecken, die am „Rand“ eines Gebäudes liegen, sind genau die Ecken, die nicht den vollen Grad haben. Entfernt man also alle Ecken mit vollem Grad, so erhält man leicht einen Graphen, der nur die Ecken enthält, deren ausgehende Kanten mit der Umgebung kollidiert sind. Ist man daran interessiert, dass der Navigationsgraph nicht zu nahe an Wände gelangt, entfernt man einfach diese Ecken aus dem Navigationsgraphen. Damit ist ein Mindestabstand zu Wänden in Größe der Kantenlänge erzwungen. Man muss bei diesem Prozess aber darauf achten, dass man den lokalen Zusammenhang des Navigationsgraphen nicht zerstört. Dazu kann man einerseits die Kantenlänge geeignet klein wählen, oder bei der Entfernung einer Ecke Routen zwischen den Nachbarn berechnen und so den Fall erkennen und vermeiden, dass durch die Entfernung einer Ecke eine sinnvolle Verbindung zwischen zwei Räumen entfällt oder der Graph in zwei Zusammenhangskomponenten zerfällt.

4.3.2 Semantische Overlays

Damit der im vorigen Abschnitt beschriebene Algorithmus zur Erzeugung eines Zugangsgraphen funktioniert, muss er in der Lage sein, den Kollisionstest durchzuführen. Die dazu nötigen Zusatzinformationen werden in semantischen Overlays spezifiziert. Dabei werden zwei verschiedene Arten von Overlays unterschieden: Die erste Art von Overlay wird als geschlossenes Polygon modelliert und die Bedeutung bezieht sich auf alle Ecken und Kanten des Graphen, die im Inneren des Polygons liegen. Die zweite Art von Overlay ist durch Strecken und Liniensegmente gegeben. Die semantische Bedeutung bezieht sich dann auf alle Kanten im Navigationsgraphen, die diese Strecken schneiden.
Im Folgenden wird eine Menge semantischer Overlays beschrieben, die im Projekt HIPS verwendet wurden, um ein – mit den zu Grunde liegenden Karten zusammen – ausreichendes Umgebungsmodell zu erzeugen.

4.3.2.1 Obstruktionen

Da das zu Grunde liegende Kartenmaterial fehlerhaft sein kann, benötigt man eine Möglichkeit, diese Fehler zu korrigieren. Ein typischer Fehler, der bei gescannten Karten oder auch bei ungenauen CAD-Zeichnungen auftritt, besteht darin, dass bei der Grapherzeugung das Gebäude an Stellen betreten und verlassen wird, an denen das in der Realität nicht möglich ist. Auch sind häufiger größere Objekte in der Umgebung, beispielsweise ein Brunnen oder ein Gepäckband an einem Flughafen, die nicht begehbar sind, in den Bauzeichnungen aber nicht enthalten sind. Die Grapherzeugung kann durch sogenannte *Obstruktionen* daran gehindert werden, solche Bereiche zu betreten. *Obstruktionen* sind Strecken, die von der Grapherzeugung nicht durchlaufen werden können. Sie werden im Kartenmaterial der Navigationsanwendung nicht angezeigt und eignen sich daher auch, um Bereiche und Durchgänge kurzfristig zu sperren.

4.3.2.2 Brücken

Wie bereits beschrieben, enthalten typische Bauzeichnungen Linien, die vollkommen korrekt sind, aber überschritten werden können. Dies können organisatorische Linien und Konturen wie Bodenplatten und Gebäudeumrisse, oder auch Türen, Treppen, Gitter und ähnliche Objekte sein. Das semantische Overlay zur Korrektur solcher Probleme wird als *Brücken* bezeichnet. Eine Brücke ist ein geschlossenes Polygon, innerhalb welchem jegliche Geometrie ignoriert wird. Falls die unter einer Brücke liegende Geometrie teilweise erhalten werden muss, so kann sie mit Obstruktionen nachgezeichnet werden, die entsprechend stärker sind als Brücken. Die Grapherzeugung ignoriert zwar sämliche Geometrie innerhalb der Fläche einer Brücke, kollidiert aber weiterhin mit Obstruktionen.

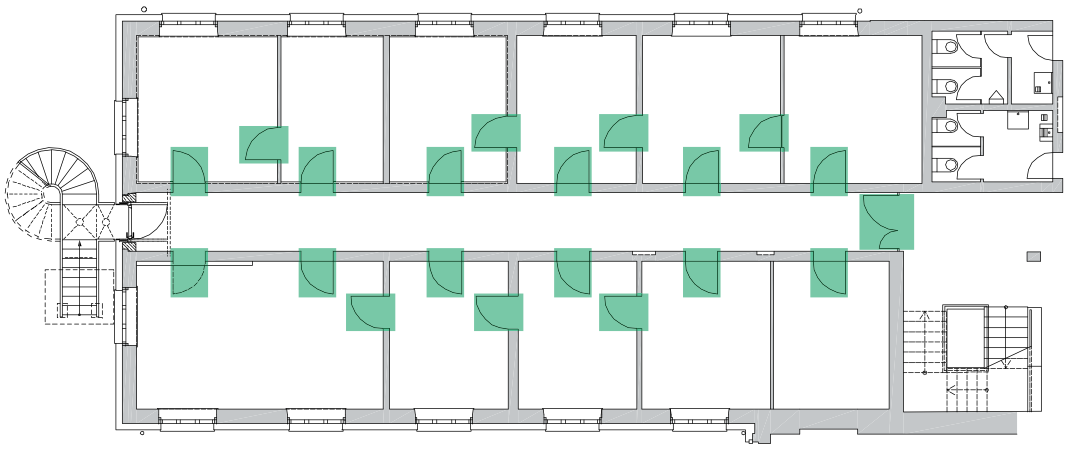


Abbildung 4.11: Ein mit *Brücken* überlagerter Grundriss-Ausschnitt

4.3.2.3 Bereiche manueller Grapherzeugung

Innerhalb eines polygonalen Bereiches des semantischen Typs *Manueller Bereich* wird der Graph-Erzeuger keine Kanten und Ecken anlegen. Vielmehr wird innerhalb dieser Bereiche der Graph-Ausschnitt von Hand gezeichnet. Insbesondere bei Treppen, bei denen der Graph ja über verschiedene Ebenen verbunden werden muss, wird dieses semantische Overlay erzeugt. Auch bei den später vorgestellten Erkennungsalgorithmen für diverse Zeichnungselemente können Gebiete markiert werden, in denen die manuelle Interaktion mit dem Benutzer notwendig wird.

4.3.2.4 Beschleunigter Bereich

Bei der Suche nach schnellen Wegen innerhalb eines komplexen Gebäudes wie einem Flughafen, gibt es Bereiche, in denen die effektive Geschwindigkeit der Fortbewegung beeinflusst wird. So kann beispielsweise vor der Sicherheitskontrolle davon ausgegangen werden, dass die Geschwindigkeit der Fortbewegung durch eine Warteschlange sehr niedrig ist, während beim Betreten eines Laufbandes die Geschwindigkeit massiv ansteigt.
Solche Bereiche markieren Overlays des Typs *„Beschleunigter Bereich“*. Hierbei kann die Größenordnung der Beschleunigung auch noch zur Laufzeit des Navigationssystems, beispielsweise durch vorhandene Messungen der Warteschlangen, bestimmt werden.

4.3.2.5 Einbahnstraßen

In komplexen Gebäuden wie Flughäfen müssen mitunter auch spezielle Regeln, die die Laufrichtung betreffen, modelliert werden. So ist es etwa einem Passagier zwar erlaubt, durch die Sicherheitskontrolle zu gehen, aber nicht mehr zurück – zumindest nicht an derselben Stelle im Gebäude. Aber auch sehr

gewöhnliche Gebäudeteile wie Rolltreppen und Laufbänder können sinnvollerweise nur in eine Richtung verwendet werden.
Um diese Art von Bewegungseinschränkungen effektiv zu modellieren, können Bereiche als Polygon modelliert werden, dem eine erlaubte Bewegungsrichtung aus den vier Hauptrichtungen des „Eight-Corner-Systems“ vorgegeben wird. Dabei zahlt sich die gewählte, interne Struktur des Graphen dahingehend aus, dass die Grapherzeugung neben der definierten Hauptrichtung noch die beiden benachbarten Nebenrichtungen verwenden darf. Dabei entsteht genau der gewünschte Effekt, dass das Polygon im Wesentlichen nur in die definierte Richtung durchlaufen werden kann, während eine kleinere Schräge durch gelegentliche Schritte in die benachbarten Richtungen ausgeglichen werden kann.

4.3.2.6 Zugangsobjekte

Um die verschiedenen Zugangsberechtigungen zu modellieren, wird für das Kartenmaterial ein polygonales *Zugangs-Overlay* definiert, welches nur dann betreten oder durchschritten werden darf, wenn der aktuelle Nutzer eine Berechtigung für dieses Zugangsobjekt besitzt. Die Berechtigungen können in Echtzeit aus einer Schließdatenbank zugewiesen werden oder auch nur den Rollen verschiedener Personen entsprechen. So dürfen die Mitarbeiter an einem Flughafen den Sicherheitsbereich nicht durch die Passagierschleuse betreten, sondern müssen durch spezielle Mitarbeiterschleusen geführt werden.

4.3.2.7 No-Transit-Overlays

Bei der Erstellung des Kartenmaterials am Flughafen München ist aufgefallen, dass einige Shops auch dann als Abkürzung verwendet werden, wenn die Bewegungsgeschwindigkeit im Inneren des Shops reduziert wurde. Damit das Navigationssystem keine Empfehlungen ausspricht, die einen bestimmten Shop benachteiligen, können diese vermieteten Gebiete mit einem polygonalen *No-Transit-Overlay* markiert werden. Diese Gebiete dürfen bei der Suche im Navigationsgraph zwar betreten, aber nur dann verlassen werden, wenn im Inneren des No-Transit-Gebietes ein Point-of-Interest besucht wurde.
Um auch komplexe Flächen, die sich nicht als einzelnes Polygon modellieren lassen, als einheitliches No-Transit-Gebiet zu modellieren, können mehrere No-Transit-Gebiete dieselbe ID haben, und deren Vereinigung wird dann zu einem No-Transit-Gebiet im Navigationsgraph.

4.3.2.8 Zusammenfassung

Diese Übersicht über die modellierbaren semantischen Overlays kann natürlich in jede Richtung erweitert werden, bis für eine tatsächliche Anwendung ausreichende Informationen vorliegen. Der große Vorteil dieser Herangehensweise besteht darin, dass die notwendigen semantischen Informationen voneinander isoliert sind und die Vorgehensweise sehr gut verständlich ist. Insbesondere

kann eine bestimmte Applikation auch nur auf einen der obigen Aspekte zugreifen, ohne sich um die anderen Aspekte oder gar eine ontologische Struktur zu kümmern. So wurde im Projekt HIPS die Personalisierung in Bezug auf Schließberechtigungen implementiert, indem, ausgehend von einer Menge an Schließberechtigungen, allein mit den Zugangsobjekten ein personalisierter Zugangsgraph erstellt wurde.
Ein weiterer Vorteil dieser Art der Modellierung besteht darin, dass die Anzahl der Änderungen an den semantischen Daten wesentlich geringer ist, als die Änderung an den CAD-Daten oder gar den Points-of-Interest. Im Grunde müssen die Overlays nur dann editiert werden, wenn die semantische Struktur des Gebäudes sich ändert. Dies kann durch neue oder wegfallende Wege, Rolltreppen, Laufbänder und vergleichbare bauliche Änderungen geschehen. Darüber hinaus reicht die Kombination aus Brücken und Obstruktionen aus, um jeden beliebigen Fehler in den zu Grunde liegenden CAD-Daten zu korrigieren, ohne die CAD-Daten selbst editieren zu müssen. Abbildung 4.11 zeigt ein Beispiel für eine Karte, die mit Brücken überlagert wurde, die es dem Grapherzeugungssystem ermöglichen, einige Türen und Organisationslinien zu überschreiten.

4.3.3 Erkennung funktionaler Symbolik in CAD-Plänen

Für komplexe Gebäude ist die Erstellung von semantischen Overlays im Sinne des vorigen Abschnittes mitunter sehr zeitaufwändig. Denn für jedes für die Navigation notwendige Objekt (Türen, Treppen, Rolltreppen, Aufzüge, Sicherheitsbereiche) muss zumindest ein Overlay erzeugt werden. Vor diesem Hintergrund wäre es natürlich nützlich, wenn man die Erkennung, zumindest von regelmäßig ähnlich gezeichneten Objekten wie Türen, automatisiert.
Leider gibt es in der Architektur aber keinen Standard und auch keinen Konsens, wie solche Symbole gezeichnet werden. Deshalb ist eine solche Erkennung derzeit immer an die tatsächlichen Begebenheiten des zu Grunde liegenden Materials anzupassen. Vor diesem Hintergrund ist es wichtig, dem Nutzer verschiedene Tools zur geometrischen oder flächenbasierten Erkennung von Objekten zu bieten. Der Nutzer selbst kann dann mit dem vorliegenden Plan in einem Ausschnitt ein sehr präzises Modell spezifizieren oder erzeugen, auf diesem Ausschnitt evaluieren und dann auf die gesamte Navigationsfläche verallgemeinern. Hierbei reicht es aber aus, dass die Erkennung eine Verallgemeinerung innerhalb der Karte schafft. Eine Verallgemeinerung über verschiedene Karten hinweg ist wegen des Mangels an vorliegenden Kartendaten kaum zu evaluieren und wegen der zu erwartenden geringeren Leistung für jede einzelne Karte auch nicht gewünscht.
Da nun aber, für jedes neue Gebäude getrennt, bei der Erstellung des Modells der Nutzer eingebunden wird, kann aus wirtschaftlichen Gründen nur eine Klassifikation von sehr häufigen und sehr homogenen Objekten erfolgen. Denn für diese Objekte lohnt sich sowohl die Erstellung eines Modells, als auch die manuelle Überprüfung des Ergebnisses.

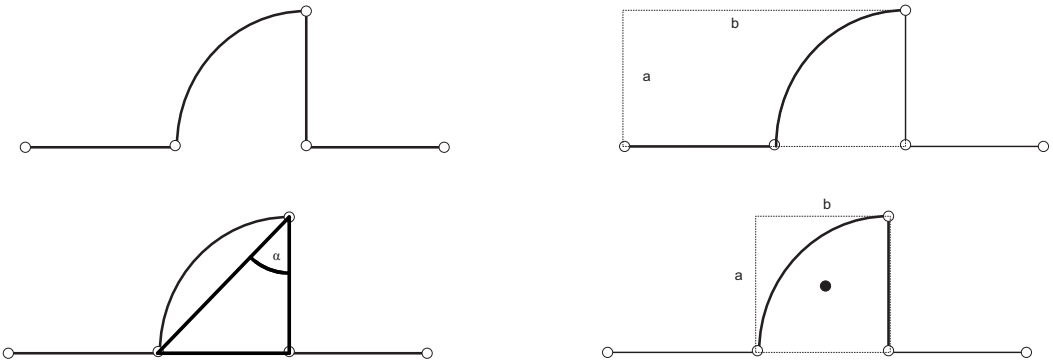


Abbildung 4.12: Geometrische Beziehungen zwischen Kanten in Türsymbolen

Für einige Symbole wird auch dann noch eine Nutzerinteraktion benötigt, wenn das Objekt korrekt erkannt wurde. So waren am Flughafen München beispielsweise die Zugänge zu Aufzügen nicht aus der Symbolik, sondern nur aus der umliegenden Geometrie ersichtlich. An einigen Stellen musste sogar vor Ort nachgesehen werden, in welche Richtung der Aufzug sich öffnet, also selbst für den Menschen reichen die Informationen in einer Karte nicht immer aus, um alle benötigten Informationen zu sammeln. Für solche Situationen wurde ein zusätzliches Overlay eingeführt, das eine notwendige Benutzerinteraktion beschreibt. Diese Overlays haben aber nur eine verwaltungstechnische Funktion und keine semantische Bedeutung für die Navigation, die Benutzerinteraktion sollte lediglich zeitversetzt erfolgen.

4.3.3.1 Erkennung von Türen in CAD-Daten

Abbildung 4.12 zeigt geometrische Beziehungen, die zur Erkennung von Türen in Vektorgrafiken verwendet werden können. Dabei wird davon ausgegangen, dass Türen innerhalb der Pläne recht homogen gezeichnet sind.
Eine Tür besteht meistens aus einem Kreissegment, das die Bewegung der Türspitze symbolisiert, einer Kante, die die geöffnete Tür symbolisiert und damit die Öffnungsrichtung der Tür angibt, und bisweilen einer Grundlinie, die der geschlossenen Tür entspricht. Wären jetzt die CAD-Daten insgesamt konsistent, so könnte man einfach nach Kreissegmenten der richtigen Größe und Öffnungsweite suchen. Leider funktioniert dieser Ansatz in der Praxis nicht, weil ältere Versionen von CAD-Systemen häufig gerundete Objekte in Liniensegmente tesselliert haben, und die ursprüngliche Information, dass es sich um einen Kreisausschnitt handelt, auf diese Weise verlorengegangen ist.
Da aber der Nutzer nur ein Toolkit benötigt, mit dem eine Erkennung von Türen zumindest semi-automatisch möglich ist, soll eine Liste mit geometrischen Features zusammengestellt werden, die sich für diese Anwendung als nützlich erwiesen. Diese Auflistung kann sicherlich von Anwendungsfall zu Anwendungsfall angemessen ergänzt werden, eine grundsätzliche Wiederverwend-

barkeit der elementaren, geometrischen Beobachtungen wird so aber gesichert. Viele dieser Features werden als Filter auf Listen von Strecken modelliert, die aus der Gesamtheit der Strecken diejenigen mit einer geometrischen Eigenschaft herausfiltern, sodass eine Kombination mehrerer geometrischer Eigenschaften möglichst effizient umgesetzt werden kann.
Zunächst einmal benötigt man einen Filter, der aus einer Vektorgrafik (CAD) Paare benachbarter Linien extrahiert. Denn ein Symbol ist im Allgemeinen zusammenhängend. Leider muss dabei mit geringfügigen Zeichenungenauigkeiten gerechnet werden, sodass eine graphentheoretische Untersuchung des Zusammenhangs durch Tiefensuchen nicht den gewünschten Erfolg bringt.
Solche zwei benachbarten Linien definieren eindeutig ein Dreieck, in dem die drei Endpunkte die Eckpunkte bilden. Für Türen kommen dann nur gleichschenklige Dreiecke (siehe Abbildung 4.12) in Frage, weil der Bogen ja die Öffnungslinie der Tür symbolisiert, und die Türbreite sich beim Öffnen nicht ändert. Ein weiterer Filter sucht entsprechend aus der Menge der benachbarten Linien nach solchen, die ein nahezu gleichschenkliges Dreieck definieren.
Wenn nun genau eine dieser Linien ein Bogen ist, so kann weiter geprüft werden, ob der Mittelpunkt des Kreises, aus dem der vorliegende Kreisbogen entnommen wurde, in etwa den dritten Punkt des Dreiecks ergibt. Dann handelt es sich sehr sicher um ein Türsymbol.
Ein alternativer Filter mit ähnlichem Effekt, der aber etwas schneller berechnet werden konnte, besteht darin, dass man das gefundene Dreieck, wie in Abbildung 4.12 illustriert, zu einem Rechteck mit Seitenlängen a und b vervollständigt und dann das Rechteck daraufhin untersucht, in wie weit es sich von einem Quadrat unterscheidet. Dazu sei die Maßzahl „Squarity“ wie folgt definiert:

$$\mathrm{Sq}(a, b) = (a - b)^2$$

Diese Definition ist offensichtlich symmetrisch in a und b, ergibt nur positive reelle Zahlen, ergibt den Wert 0 genau dann, wenn es sich um ein Quadrat handelt, und wächst quadratisch mit der Abweichung der Seitenlängen voneinander. Diese Maßzahl ist für Türen relativ klein und unterscheidet beispielsweise die Fälle auf der rechten Seite von Abbildung 4.12 sehr gut voneinander.
Eine weitere nützliche Beobachtung besteht darin, dass die nun häufig bekannte Lauflinie der Tür – ob sie als Bogen oder als Strecke gezeichnet wurde – eine Diagonale eines Rechtecks ist, das in der Regel bis auf diese Diagonale oder den Türbogen leer ist. Denn sonst kann man die Tür nicht durchlaufen. Auch dies kann man als Filter anwenden, sobald man eine Hypothese über die Lauflinie der Tür besitzt.
Auch wenn diese Herangehensweise an das Problem der Türerkennung reichlich statisch und in gewissen Beobachtungen auch ziemlich spezialisiert ist, ergibt sich ein hochverständliches System. Da die grundlegende Idee bei der Indoor-Navigation sein muss, dass jeder Eigentümer eines Gebäudes mit nur geringem Aufwand eine Navigation erzeugen und pflegen kann, so ist doch die durch diese Herangehensweise offenbarte Verständlichkeit der Klassifikation den mitunter

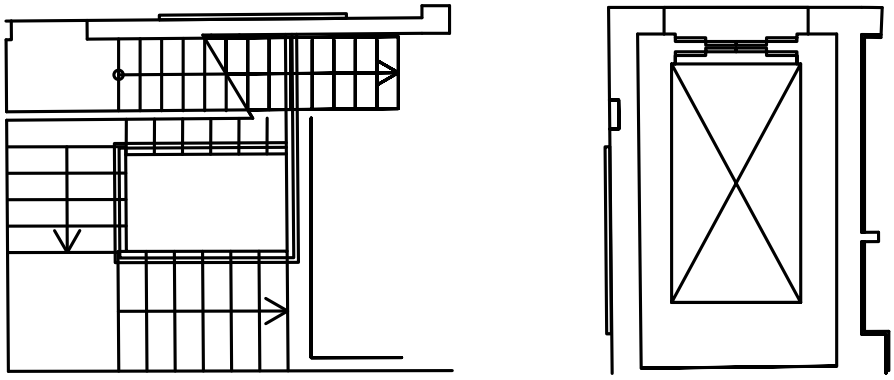


Abbildung 4.13: Typische Treppen- und Aufzugsymbolik in CAD

stärkeren Methoden des maschinellen Lernens vorzuziehen.
Mit diesen Filtern ist es im Projekt mit dem Flughafen München gelungen, alle Türen zu finden, die nicht in offensichtlicher Weise in der CAD-Datei falsch gezeichnet waren. Die einzige, subtile Fehlerquelle für dieses System besteht in Wänden, die zufällig auf die obige, geometrische Einschränkung passen, und in den Türen, deren Türbögen, wie bereits erwähnt, in tessellierter Form als Sequenz von Linien vorliegen.
In diesem Abschnitt wurden Tür-Symbole unter Verwendung geometrischer Eigenschaften von Vektorzeichnungen erkannt, also im Wesentlichen mit Bezug auf die Begrenzungslinien der Tür. Im Falle, dass das Türsymbol eine Fläche umschließt, kann man auch diese Fläche untersuchen. In diesem Zusammenhang ist natürlich der bereits vorgestellte radiale Sweepline-Algorithmus aus Abschnitt 4.2.1.2 zur Erkennung geeignet.
Die auf diese Weise erkannten Türen können automatisch mit einer Brücke als begehbar markiert werden, eine visuelle Überprüfung erfolgt dann bei der Erstellung des Navigationsgraphen, da man dessen Wachsen ja verfolgen soll, und somit gegebenenfalls fälschlicherweise als Türen markierte Gebiete auffallen.

4.3.3.2 Erkennung von Treppen

Während Türsymbole zumindest innerhalb eines Planes in der Regel relativ gleichmäßig gezeichnet werden, gibt es für die Erkennung von Treppen ganz erhebliche Schwierigkeiten, da es so viele Gestaltungsmöglichkeiten gibt. Ein typisches Beispiel für ein Treppenhaus ist in Abbildung 4.13 zu sehen.
Die einzige Ähnlichkeit, die zwischen allen Treppenformen beobachtet werden kann, besteht darin, dass es Stufen gibt. Eine effektive Erkennung von Stufen ist möglich, indem die Gruppen von Linien gesucht werden, die die folgenden Eigenschaften haben:

- Sie sind allesamt nahezu gleichlang,

- sie sind nahezu parallel,
- sie stehen in gleichmäßigem Abstand einer bestimmten Größenordnung zueinander,
- sie besitzen eine gemeinsame Begrenzungslinie.

Da es für Treppen in komplexen Gebäuden auch für Menschen mitunter schwer entscheidbar ist, ob die Treppe zwei verschiedene Stockwerke verbindet und wenn, ob es eine Verbindung mit dem darüberliegenden, dem darunterliegenden oder beiden Stockwerken gibt, werden solche Liniengruppen zur manuellen Bearbeitung markiert. Dementsprechend ist auch eine gewisse Anzahl an fälschlicherweise erkannten Treppen eher akzeptabel als eine Nicht-Erkennung. Die obigen Eigenschaften haben auf den uns zur Verfügung stehenden Raumplänen gut funktioniert, müssen aber sicher an jede einzelne Umgebung angepasst werden, insbesondere bei der Vorgabe der Schrittweite und ob es eine gemeinsame Begrenzungslinie gibt.

4.3.3.3 Erkennung von Aufzügen

Die Erkennung von Aufzugssymbolik ist erstaunlich einfach, weil es nur wenige sonstige Elemente im Gebäude gibt, die ihnen ähneln. Ein Aufzug wird in CAD häufig durch ein Rechteck, das den Aufzugschacht repräsentiert, symbolisiert. Dieses Rechteck wird dann mit beiden Diagonalen ergänzt, die die zentrale Aufhängung des Aufzuges symbolisieren sollen. Ein Beispiel ist auf der rechten Seite von Abbildung 4.13 zu sehen.
Eine Möglichkeit zur effizienten Erkennung solcher Symbole ist die Folgende: Zunächst durchsucht man die CAD-Daten nach Sequenzen benachbarter Linien, die ein Rechteck umschließen. Danach prüft man, ob diese Rechtecke mit den entsprechenden Diagonalen durchkreuzt sind.
Aufzüge können zwar relativ einfach erkannt werden, benötigen aber in der Regel immer manuelle Interaktion mit dem Nutzer, denn die Ein- und Ausstiegsmöglichkeiten sind nicht erkennbar und in einigen Fällen sind Stockwerke, in denen der Aufzug nicht hält, nicht aus dem Plan ersichtlich.

4.3.4 Grenzen der Objekterkennung

Die in den voranstehenden Abschnitten vorgestellten Werkzeuge zur Erkennung von Symbolik in CAD-Plänen sind natürlich in vieler Hinsicht fragwürdig: Zum Einen sind sie nicht in größerem Maßstab evaluierbar, weil nur sehr wenige CAD-Pläne öffentlich zugänglich sind. Zum Anderen ist nicht zu erwarten, dass die Symbolik, die in den einen Plänen vorherrscht, auch in anderen Plänen verwendet wird. In diesem Sinne soll der entstandene Prototyp keinesfalls in der Lage sein, CAD-Daten automatisch zu Navigationssystemen zu vervollständigen. Vielmehr wird durch die Integration elementargeometrischer

Filtersysteme und durch eine hohe Flexibilität in der Anpassung und Kombination derselben aufgezeigt, dass die Erstellung von semantischen Overlays in jedem einzelnen Anwendungsfall gewinnbringend durch die Konstruktion von Ad-Hoc-Methoden beschleunigt werden kann.
Am Flughafen München etwa wurde die Türerkennung auf einem sehr kleinen Ausschnitt konstruiert und danach auf die Gesamtfläche des Flughafens ausgeweitet. Aus den Daten der Schließanlage ist bekannt, dass es in dieser Umgebung mehr als 13.000 Türen zu erkennen gibt. Die Erstellung eines allein für die CAD-Daten der jeweiligen Umgebung passenden Erkennungssystems ist also durchaus sinnvoll. Eine manuelle Markierung aller Türen ist hingegen mit immensem Aufwand verbunden.

4.4 Zusammenfassung und Ausblick

Durch die enorme Komplexität von Indoor-Umgebungen ist es auch in naher Zukunft unwahrscheinlich, dass sich ein einheitliches Konzept für die Verwaltung und Verwendung von Indoor-Daten wie Raumplänen, Wegeplänen und Landmarken durchsetzt.
Eine aufwändige Standardisierung, wie sie im Falle von CityGML für die Stadtplanung durchgeführt wurde, ist schon deshalb unwahrscheinlich, weil die Anforderungen typischer Anwendungen der Indoor-Navigation nur wenige Überdeckungen untereinander oder mit anderen Anwendungsbereichen haben. Folglich werden für die meisten Gebäude nur Daten vorliegen, die grundsätzlich beispielsweise zur Errichtung von Gebäuden erfasst werden.
Daraus folgt nach Meinung des Autors, dass sinnvollerweise nur diese Daten für eine Indoor-Navigation verwendet werden sollen. Denn eine komplexe Modellierung aller notwendigen Informationen macht es zwar den Anwendungsentwicklern leichter, ein funktionierendes System zu erstellen, aber leider wird dieses System kaum umgesetzt werden, weil der Aufwand, solche Daten zu erzeugen, nur selten finanzierbar ist.
In diesem Sinne wurden exemplarisch zwei minimalistische Umgebungsmodelle für die Indoor-Navigation entworfen. Ein Modell basiert allein auf Bitmaps, die mit einfachen Bildbearbeitungs-Algorithmen für eine Indoor-Navigation durchaus ausreichen. Ein solches Modell eignet sich besonders für die Anwendung auf Smartphones und anderen mobilen Endgeräten, weil diese in der Regel über eine graphische Benutzerschnittstelle verfügen, und daher die grundlegenden Bildoperationen teils erstaunlich performant unterstützen. Eine Verwendung von vektorbasierten Karten auf mobilen Endgeräten hingegen führt zu wesentlich komplexeren Datenstrukturen, deren Verarbeitung auf kleineren Prozessoren schwierig ist. Insbesondere wurde in diesem Modell davon ausgegangen, dass kein Navigationsgraph explizit vorliegt, sondern dass eine Hierarchie aus zwei Graphen automatisch erstellt und verwendet wird: Zunächst ein Verbindungsnetzwerk aller Räume, Türen und Treppen, danach auf der Vereinigung der beteiligten Räume eines jeden Stockwerkes eine direkte Suche

auf Pixel-Ebene. Der implizite Navigationsgraph hat dann pro Pixel der Freifläche eine Ecke und je eine Kante zu allen Nachbar-Pixeln. Während auf der ersten Hierarchie-Ebene eine Verwendung des A^*-Algorithmus fragwürdig ist, weil der Graph keine zulässige, nicht-triviale Heuristik zulassen muss, ist dieses Problem auf der unteren Hierarchie-Ebene nicht vorhanden. Der Graph auf Pixel-Ebene ist dann pro Stockwerk euklidisch und man kann jegliche Metrik als Heuristik für A^* gewinnbringend einsetzen.

Als Gegenstück dazu wurde ein komplexeres Umgebungsmodell entworfen und umgesetzt, das alle Notwendigkeiten für den Betrieb einer flächendeckenden Indoor-Navigation beinhaltet. Entscheidende Unterschiede zu anderen Herangehensweisen bestehen darin, dass langlebige Informationen von den etwas kurzlebigeren Informationen getrennt gehalten werden, und dass die zu Grunde liegende CAD-Datenbasis keinerlei Richtlinien in Bezug auf Organisation, Homogenität oder Strukturierung erfüllen muss.

Somit kann die CAD-Datenbasis stets geändert werden, an den langlebigen Navigationsinformationen werden aber nur dann Änderungen vorgenommen, wenn die Grundstruktur eines Gebäudeteils sich ändert. Exemplarisch wurden für diese doch sehr große Datenbasis einige Algorithmen und Heuristiken zur Erkennung und Behandlung von Türen, Treppen, Rolltreppen und Aufzügen entworfen und umgesetzt, sodass eine semi-automatische Erstellung von flächendeckenden Navigationsgraphen möglich wurde. Da es leider keine einheitlichen Standards in der Zeichnung von Gebäudeobjekten gibt und nicht davon auszugehen ist, dass solche Standards sich bald durchsetzen, weil die Architektur eine gestaltende Kunstform ist, die sich keinen Standards unterwirft, ohne selbst einen Vorteil davon zu haben, ist eine solche Objekterkennung sicher nicht allgemeingültig. Da aber die Pläne einzelner Gebäude in sich meist einheitlich sind, ist eine Anpassung für eine andere Umgebung je nach Umfang des Projektes sicher möglich und sinnvoll.

Die verwendeten Navigationsgraphen sind einfache Flächenabstraktionen, welche die begehbare Umgebung in relativ hoher Auflösung modellieren. Man kann argumentieren, dass diese Graphen unnötig groß und unnötig komplex sind. Doch für die vorliegenden Anforderungen sind sie im folgenden Sinne minimal: Für die flächendeckende Indoor-Navigation muss, da die Verortung von Start- und Zielpunkten nicht bekannt ist, eine ausreichend hohe Dichte an Ecken im Graphen vorliegen. Ein Netzwerk, welches eine kleine Menge an interessanten Punkten verbindet, müsste bei jeder Änderung aktualisiert werden. Die Wahl einer solchen Mindest-Dichte gibt dann die Anzahl der Ecken im Wesentlichen vor. Somit wird die Performance der Wegesuche größtenteils durch den Verzweigungsgrad, also die durchschnittliche Anzahl möglicher Wahlen an den Ecken des Graphen, gegeben. Dieser ist im vorliegenden Fall stets kleiner als 8. Denn wenn ein Vertex betreten wird, kann es auf höchstens 8 Kanten verlassen werden. Diese acht Bewegungsrichtungen braucht man aber auch unbedingt, um Einbahnstraßen auch dann umzusetzen, wenn die Bewegung nicht ganz achsenparallel verläuft. Man soll also auf einer Einbahnstraße

durchaus auf die benachbarten Diagonalen ausweichen dürfen. Sonst würde die Graph-Erzeugung in vielen Fällen innerhalb einer Einbahnstraße scheitern. Bestätigend lässt sich sagen, dass selbst ohne die Verwendung von heuristischen Suchalgorithmen unter Verwendung von Dijkstra eine Routenberechnung auf dem Graphen des Flughafen Münchens auf gewöhnlicher Serverhardware eine Laufzeit von knapp 30ms hat. Dabei wurde ein Graph mit knapp 65.000 Ecken und gut 700.000 Kanten verwendet, der zu großen Teilen automatisch erzeugt und zu kleinen Teilen von Hand korrigiert wurde.

5 Wege in Gebäuden

Eine zentrale Aufgabe der Navigation ist die Bestimmung eines guten Weges von einem Startpunkt zu einem oder mehreren Zielen. Dabei sind im Außenbereich eigentlich nur zwei Bewertungen für die Güte eines Weges gängig: Die Suche nach dem besten Weg im Sinne der kürzesten Strecke oder im Sinne der kürzesten Reisezeit. Darüber hinaus werden häufig nur Nebenbedingungen erlaubt, etwa dass keine Autobahnen verwendet werden dürfen. An dieser Stelle existiert vielfach der aufrichtige Wunsch nach anderen Bewertungen, beispielsweise nach finanziellen oder ökologischen Gesichtspunkten. Genauso gibt es – zumindest in der Wissenschaft – auch Arbeiten in Bezug auf *multimodale Navigation* über verschiedene Verkehrsmittel wie Auto, Bus und Bahn hinweg.

Obwohl es viele kleinflächige Demonstratoren für diese komplexeren Systeme gibt, scheitern solche Systeme zumindest im großflächigen Betrieb an mathematisch-algorithmischen Details, die auch in Gebäuden zum Tragen kommen können: Das Fehlen der Dreiecksungleichung, einer guten Heuristik und einer vernünftigen Ordnungsrelation auf den verschiedenen Aspekten der gewünschten Bewertung.

Die einfachsten Verfahren zur Suche nach einem kürzesten Weg verwenden eine Aufzählung aller vom Start- oder Zielpunkt ausgehenden Wege. Algorithmen für diese Aufgabe sind die Tiefen- und Breitensuche, die alle Wege ablaufen. Die Tiefensuche fügt zunächst immer weitere Kanten zu einem Weg hinzu, bevor ein zweiter Weg versucht wird. Die Breitensuche dagegen setzt erst alle anliegenden Kanten zu Wegen zusammen, bevor neue Segmente an diese Wege angesetzt werden. Da eine Suche in diesem Sinne in der Praxis viel zu komplex ist, verwendet der Dijkstra-Algorithmus eine besondere Reihenfolge der Aufzählung aller Wege: Die Sequenz der Längen der Wege in der Aufzählung ist monoton steigend. Dies bedeutet, dass der kürzeste Weg bekannt ist, sobald man das Ziel zum ersten Mal besucht. Denn würde bei einer vollständigen Aufzählung aller Wege ein kürzerer Weg zum Ziel später gefunden, widerspräche das der Monotonie der Aufzählung. Um eine solche monoton steigende Aufzählung effizient zu ermöglichen, gibt es die Datenstruktur des *Fibonacci-Heap*, die in der Lage ist, eine ungeordnete Liste so zu verwalten, dass Elemente mit geringem Aufwand – in $O(1)$ – eingefügt werden können, aber dennoch stets das kleinste Element der Liste schnell – in $O(\log n)$ – abgerufen werden kann.

Nun ist offenbar, weshalb viele Bewertungen und insbesondere ihre Kombinationen nicht geeignet sind, um effizient kürzeste Wege zu berechnen: Es existiert auf den Bewertungen keine totale Ordnung, somit kann erst dann der kürzeste Weg bekannt sein, wenn tatsächlich alle Wege durchlaufen wurden.

Ein Vorgehen wie beim Dijkstra-Algorithmus mit Abbruch beim Ziel muss also scheitern.
Ganz ähnlich liegt der Fall in Bezug auf den A^*-Algorithmus, einer beliebten Beschleunigung des Dijkstra-Algorithmus durch eine Heuristik, welche die Entfernung zum Ziel schätzen soll. In der Theorie ist der A^*-Algorithmus optimal und optimal effizient, was bedeutet, dass er den kürzesten Weg mit kleinstmöglichem Rechenaufwand bestimmt. Diese Aussage gilt aber nur dann, wenn die folgenden Voraussetzungen erfüllt sind: Die Heuristik überschätzt niemals die tatsächliche Entfernung zum Ziel und lässt sich in konstanter Zeit berechnen, und der Navigationsgraph erfüllt die Dreiecksungleichung. Insbesondere die Voraussetzung der Dreiecksungleichung ist innerhalb von Gebäuden häufig verletzt und es ist schwierig, für eine Wegzeit-Bewertung eine Heuristik zu konstruieren, die nicht-trivial ist und gleichzeitig die Entfernung zum Ziel niemals überschätzt, weil Treppen, Rolltreppen, Warteschlangen und Laufbänder die Geschwindigkeit von Personen in Gebäuden in hohem Maße verändern können.
Die soeben dargestellten Probleme stellen selbstverständlich nur Warnungen dar und sind keineswegs Ausschlusskriterien: Man kann und wird selbstverständlich den einen oder anderen Algorithmus auch ohne Änderungen in Gebäuden verwenden. Allerdings muss man damit rechnen, dass ein solcher Algorithmus auch zu massiven Fehlern führen kann und nicht immer die beste Route auf effizienteste Weise berechnet. Ein Vorteil in Gebäuden ist aber zumindest für diese grundlegenden Navigationsaufgaben, dass die Komplexität, gemessen als Anzahl der Knoten und Kanten im Navigationsgraphen, im Vergleich zur Außenwelt eher gering ist, und somit bei sorgfältiger Implementierung auch die nicht-optimierten Verfahren wie Dijkstra oder sogar eine Tiefen- und Breitensuche effektiv ausführbar sind.
Problematischer ist die Situation, wenn ein kürzester Weg über mehrere Ziele gesucht wird. Der Rest dieses Kapitels widmet sich genau dieser Situation, teilweise unter Hinzunahme einer Ordnungsrelation auf den Zielen, denn in der Indoor-Navigation ist häufig eine gewisse Vorzeitigkeitsrelation zwischen verschiedenen Zielen gegeben.

5.1 Geordnete Navigation

Im Rahmen des Forschungsprojektes HIPS wurde ein Verfahren benötigt, eine teilweise geordnete Liste von Points-of-Interests so zu sortieren, dass die Reisezeit minimal wird. Das System erstellt auf verschiedenen Wegen (Videokonferenz, Touch-Eingabe, typische Fluggast-Prozesse) eine Liste an interessanten Punkten, zu denen der Nutzer geleitet werden soll. An einem Flughafen lassen sich die semantischen Beziehungen zwischen den Orten durch die Zuweisung einer personalisierten Ordnungszahl erfassen. So wird für einen abfliegenden Fluggast den Zielen, die sich im Sicherheitsbereich befinden, eine höhere Ordnungszahl zugeordnet als den Zielen im öffentlichen Bereich, weshalb dann jede vernünftige Lösung des Navigationsproblems „In welcher Reihenfolge soll ich

meine Ziele anlaufen?“ diese Ordnung respektiert. Für einen ankommenden Fluggast würde natürlich nur die umgekehrte Ordnung Sinn machen, während ein Gast ohne Ticket sich nur im öffentlichen Bereich aufhalten kann und innerhalb dieses Bereiches häufig keine Ordnungsvorgaben erfüllen muss. Auch wenn durch das Gebäude selbst keine Ordnung vorgegeben ist, kann eine solche Ordnung notwendig werden: Hat der Gast beispielsweise ein Parkticket, das er unter Vorlage eines Kassenbons in einer Parkzentrale rabattieren lassen kann, so muss er, bevor er zur Tiefgarage geleitet wird, zur Parkzentrale des Flughafens geleitet werden. Auch dies ist als Ordnung auf den Zielen seiner persönlichen Navigation modellierbar. Und wenn man die Situation eines Einkaufs in einem Möbelhaus betrachtet, so sind zwischen dem eigentlichen Einkauf, einer möglichen Beratung in einer Fachabteilung, dem Kassiervorgang und dem Abholvorgang gewisse Vorzeitigkeiten zu beachten, die sich ebenfalls als Ordnung auf den Zielen modellieren lassen.
Nun kennt das System also eine Menge an Orten und eine Ordnung, die aus dem Gebäude, der Nutzerrolle oder auch aus der Menge der interessanten Punkte selbst resultiert. Die Frage ist nun: Wie berechnet man einen vernünftig kurzen Weg durch das Gebäude, der alle vorgegebenen Ziele enthält? Dieses Problem ist mit dem klassischen Travelling-Salesman-Problem verwandt. Deshalb wird zunächst dieses klassische Problem betrachtet, und es werden die Ergebnisse und Lösungsansätze in diesem klassischen Fall beschrieben, um dann die Übertragung dieser Lösungsansätze auf das vorliegende Problem zu diskutieren.

5.1.1 Das klassische Travelling-Salesman-Problem

Das klassische *Travelling-Salesman-Problem* ist das Problem eines Handelsreisenden, eine Menge an Städten in der Reihenfolge zu besuchen, welche die benötigte Reisestrecke minimiert. Etwas präziser formuliert lautet das Traveling-Salesman-Problem:
Problem: *Sei G ein vollständiger, endlicher Graph mit nicht-negativen Kantengewichten. Sei eine Tour eine Permutation der Ecken von G und die Länge einer Tour die Summe der Kantengewichte der Kanten, welche die Ecken von G in der Reihenfolge der Tour und die letzte mit der ersten Ecke verbinden. Finde eine Tour mit minimaler Länge.*
In Abbildung 5.1 ist ein klassisches Beispiel zu sehen: Die fünfzehn größten Städte Deutschlands [List 12] sollen von einem Kaufmann auf möglichst kurzem Gesamtreiseweg besucht werden. Der kürzeste Weg ist in der Zeichnung angedeutet. Es ist unerheblich, ob die Städte im Uhrzeigersinn oder gegen den Uhrzeigersinn angelaufen werden, und es ist auch unerheblich, in welcher Stadt die Reise beginnt. Da es sich um 14 Städte und eine feste Start-Stadt handelt und im Prinzip in jedem Schritt von jeder Stadt aus jede andere anvisiert werden kann, die noch nicht besucht wurde, gibt $14! \approx 87,18$ Mrd. Möglichkeiten. Beachtet man hier, dass der Umlaufsinn in diesem Fall nicht

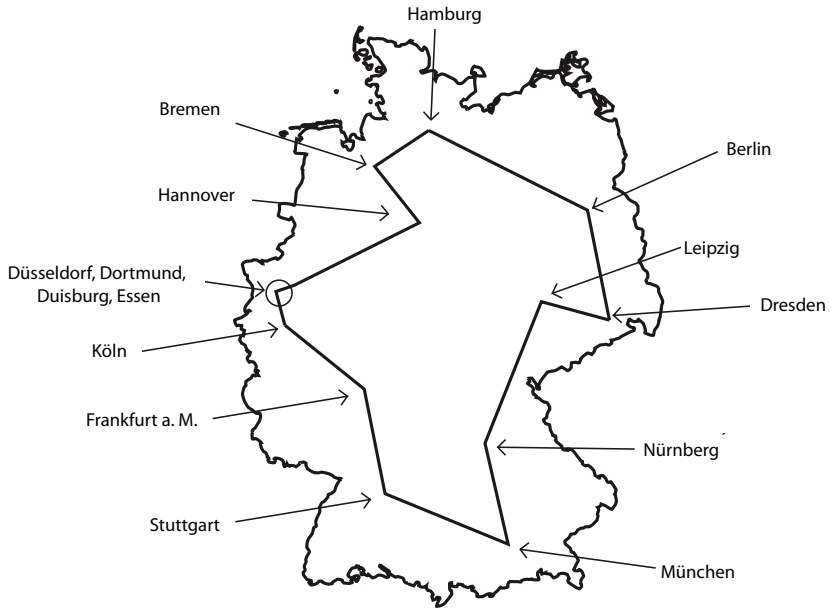


Abbildung 5.1: Beispiel für das Travelling-Salesman-Problem: Der kürzeste Weg, der die fünfzehn größten Städte Deutschlands miteinander verbindet

relevant ist, bleiben zumindest 43, 59 Mrd. tatsächlich unterschiedliche Touren. Das Travelling-Salesman-Problem ist NP-hart, und zum jetzigen Zeitpunkt sind keine Algorithmen mit polynomieller Laufzeit bekannt. Die benötigte Rechenzeit für die bekannten, exakten Algorithmen steigt so enorm an, dass nur sehr kleine Instanzen exakt gelöst werden können. Deshalb wird das Travelling-Salesman-Problem in der Praxis häufig nur in spezielleren Fällen betrachtet. Diese speziellen Fälle ermöglichen es dann, auch schnellere Algorithmen anzugeben, die dennoch gute Approximationen an die optimale Lösung des Travelling-Salesman-Problems konstruieren. Die wichtigsten Arbeiten und Ergebnisse in diesem Bereich werden im Folgenden zusammengefasst.

5.1.2 Lösungsalgorithmen zum Travelling-Salesman-Problem

Zunächst kann man festhalten, dass das Travelling-Salesman-Problem ein NP-vollständiges Problem ist. Dieses Ergebnis folgt aus der wegweisenden Arbeit von Karp [Karp 72] und der Tatsache, dass das Travelling-Salesman-Problem und das Problem des Hamiltonkreises equivalent sind [Kort 08]. Also weiß man derzeit nicht, ob ein Algorithmus mit polynomieller Laufzeit auf einer nichtdeterministischen Touring-Maschine existiert. Die Frage, ob ein solcher Algorithmus existiert, hängt stark mit dem bis heute ungelösten P-NP-Problem der theoretischen Informatik zusammen, da die Angabe eines polynomiellen Algo-

rithmus für ein NP-vollständiges Problem ausreicht, um $P = NP$ zu zeigen.
Der einfachste und offensichtlichste Algorithmus, das Travelling-Salesman-Problem zu lösen, ist die *ausschöpfende Suche*. Diese besteht darin, dass man alle möglichen Touren aufzählt, die jede Ecke einmal besuchen, jeweils deren Länge ermittelt und so die kürzeste Tour findet. Da der Graph vollständig ist, also zwischen jedem Paar von Ecken eine Kante existiert, entspricht diese Aufzählung einfach einer Aufzählung der Menge aller Permutationen der Städte und die Komplexität, diese aufzuzählen, ist folglich die Anzahl an Permutationen von n Städten: $O(n!)$. Im Folgenden soll – der Einfachheit halber – das subtile Detail, ob die Anfangsstadt feststeht und der Drehsinn relevant ist und also nur $(n-1)!/2$ tatsächlich unterschiedliche Touren existieren, übergangen werden, und alle Touren mit allen Startstädten und jedem Drehsinn als unterschiedlich aufgefasst werden.
Über einen solchen trivialen Ansatz hinaus gibt es viele Approximations-Algorithmen, die bestimmte Eigenschaften des Graphen verwenden, um an Stelle des Travelling-Salesman-Problems ein einfacheres Problem mit einem polynomiellen Algorithmus zu lösen, dessen Lösung immer noch in einer bekannten Weise mit der Lösung des Travelling-Salesman-Problems zusammenhängt. Dazu betrachtet man sogenannte k-Faktor-Approximationen. Das sind Algorithmen A mit polynomieller Laufzeit, für welche die folgende Ungleichung für alle Instanzen T des Problems gilt. $\mathrm{Opt}(T)$ bezeichnet dabei die Länge der optimalen Lösung der Instanz T, $A(T)$ die Länge des Ergebnisses des Algorithmus A.

$$\frac{1}{k}\,\mathrm{Opt}(T) \leq A(T) \leq k\,\mathrm{Opt}(T) \text{ für ein festes } k \geq 1 \tag{5.1}$$

Leider kann man zeigen, dass die Situation des klassischen Travelling-Salesman-Problems eine solche Approximation nicht zulässt – es sei denn $P = NP$ [Sahn 76]. Mit anderen Worten existiert kein Algorithmus A, der jede Instanz T des Travelling-Salesman-Problems in polynomieller Laufzeit löst, und gleichzeitig die Ungleichung 5.1 für einen festen Wert k erfüllt. Deswegen verwenden alle Approximationen zum Travelling-Salesman-Problem gewisse spezielle Eigenschaften des Graphen oder des Problems.
Zunächst ist die Situation herauszustellen, dass das Travelling-Salesman-Problem *metrisch* ist. Das bedeutet, dass die Gewichte für alle Dreiecke, die aus Kanten a, b, c des gegebenen Graphen zusammengesetzt sind, folgende Dreiecksungleichung erfüllen:

$$\mathrm{w}(a) \leq \mathrm{w}(b) + \mathrm{w}(c)$$

In diesem Fall reicht es, einen geschlossenen Pfad mit möglicherweise doppelten Kanten im Graphen G zu konstruieren. Dieses Vorgehen wendet der Christofides-Algorithmus [Chri 76] an und ist für diese Situation der beste, bekannte Approximations-Algorithmus mit einem Performance-Quotienten von $k = 3/2$ und einer Laufzeit von $O(n^3)$. Insgesamt bedeutet das, dass der

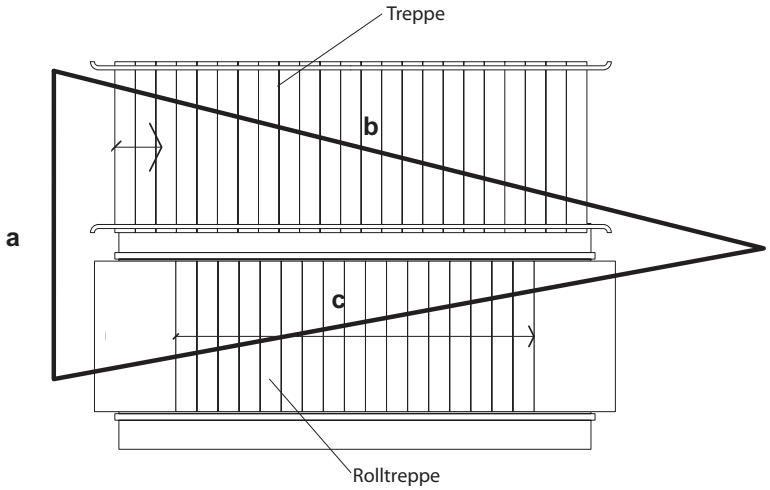


Abbildung 5.2: Beispiel für eine Verletzung der Dreiecksungleichung in Gebäuden

Christofides-Algorithmus in einer Laufzeit von $O(n^3)$ ein Ergebnis produziert, welches bis um die Hälfte der Länge der kürzesten Tour schlechter sein könnte als die tatsächliche, optimale Lösung.

In Gebäuden ist ein Navigationsgraph, dessen Gewichte den tatsächlichen Wegzeiten entsprechen, nur sehr selten metrisch. In Abbildung 5.2 ist ein Beispiel angegeben: Im dargestellten Dreieck ist die Wegzeit für die Kante b hoch, da es sich um eine Treppe handelt. Die Wegzeit für die Kante c hingegen ist recht kurz, da es sich um eine Rolltreppe handelt. Die Kante a ist kurz im Vergleich zu den beiden anderen Kanten und kann daher vernachlässigt werden. Offensichtlich kann die Dreiecksungleichung $b \leq a + c$ verletzt sein. Dazu muss nur der Geschwindigkeitsunterschied zwischen Treppe und Rolltreppe groß genug sein. Übersetzt in natürliche Sprache behauptet nämlich diese Ungleichung: „Es ist immer schneller, die Treppe zu nehmen, wenn man am Fuß der Treppe steht, als eine Rolltreppe, die sich direkt neben der Treppe befindet.“.

Eine andere geometrische Eigenschaft, die der Graph erfüllen kann, führt zu einer nahezu beliebig guten Lösung. Denn mit dieser Eigenschaft des Graphen kann ein polynomielles Approximations-Schema mit Performance-Quotient $1 + \epsilon, \epsilon > 0$ für das Travelling-Salesman-Problem angegeben werden. Die zu Grunde liegende geometrische Eigenschaft ist *euklidisch*. Sei G ein vollständiger, gewichteter Graph, dessen Ecken eine Teilmenge der euklidischen Ebene $\mathbb{R}^2$ sind und dessen Gewichte von der euklidischen Norm herrühren. Dann kann zu jedem $\epsilon > 0$ in einer Laufzeit von $O(n^3(\log n)^c)$ mit einer maschinenabhängigen Konstante c eine Tour konstruiert werden, die höchstens um den Faktor $1 + \epsilon$ länger ist als die kürzeste Tour.

Die Grundidee stammt von Karp [Karp 77] und besteht darin, eine reguläre Gitterunterteilung der Fläche vorzunehmen, und dann die Teilprobleme in den Regionen zu lösen und die Ergebnisse zusammenzusetzen. Arora arbeitet

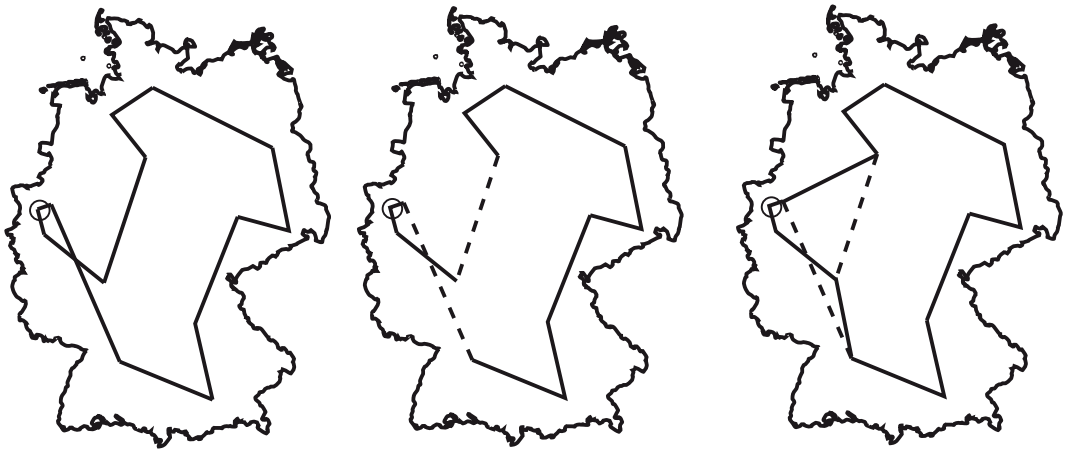


Abbildung 5.3: Ein k-Opt Austausch ($k = 2$): Ausgehend von einer Tour (links), werden zwei zufällig gewählte Kanten entfernt (Mitte) und dann die Tour mit den Kanten vervollständigt, welche die kürzeste Tour ergeben (rechts)

diese Idee aus und konstruiert damit ein Approximationsschema mit obigen Eigenschaften [Aror 98].

Im Gegensatz zu solchen geometrischen Vereinfachungen gibt es Methoden der *lokalen Suche*. Die grundlegende Idee besteht darin, dass man, ausgehend von einer heuristisch generierten initialen Tour, diese Tour durch lokale Veränderungen verbessert. Die Worst-Case-Performance dieser Algorithmenklasse ist in der Regel nicht polynomiell, aber in der Praxis funktioniert eine solche Vorgehensweise oft sehr gut. Es handelt sich also nicht um polynomielle Approxmiationsalgorithmen, sondern um Algorithmen, die in typischen Fällen ein besseres Laufzeitverhalten haben, als beispielsweise die Methode des Aufzählens. In der Analyse dieser Algorithmen wird man aber mit dem sogenannten Initialisierungsproblem konfrontiert. Die Qualität und die Laufzeit dieser Algorithmen hängen natürlich von der Qualität der initialen Tour ab. Deshalb können diese Arten von Algorithmen nur empirisch untersucht und verglichen werden. Für diese Vergleiche gibt es mit der TSPLIB [Rein 91] eine größere Sammlung von „realen“ Travelling-Salesman-Problemen und häufig wird diese Bibliothek für Vergleiche von Algorithmen herangezogen.

Einer der bekanntesten Algorithmen der lokalen Suche ist der *k-Opt-Algorithmus*. Dieser Algorithmus verbessert beispielsweise eine bestehende, typischerweise heuristisch erzeugte Startlösung so lange mit den k-Opt Operatoren, bis eine Verbesserung durch diese Operatoren nicht mehr möglich ist. Der k-Opt Operator entfernt dabei in jedem Schritt k zufällige Kanten und prüft, ob eine andere Verbindung der nun offenen Ecken zu einer besseren Tour führt. Ein Beispiel ist in Abbildung 5.3 zu sehen. Aus einer möglichen Tour werden zwei zufällige Kanten entfernt und die offenen Enden dann so neu verbunden, dass die kürzeste Tour aller möglichen Verbindungen entsteht. Im

Beispiel ist dies auch die global optimale Tour.

Das Endergebnis dieses Verfahrens nennt man im Allgemeinen k-optimal. Allerdings gibt es für jedes k Beispiele, die k-optimal, aber nicht $k+1$-optimal sind. Somit findet dieser Algorithmus in solchen Fällen nicht die kürzeste Tour. Wählt man k aber groß genug, so wird die beste Tour gefunden, beispielsweise bei $k = n$. Da aber die Laufzeit des Algorithmus mit k exponentiell steigt, muss für k ein guter Kompromiss zwischen Fehler und Laufzeit gefunden werden.

Ein solcher Kompromiss liegt der *Lin-Kernighan-Heuristik* zu Grunde, die im Prinzip ein solches k adaptiv wählt. Die Autoren nennen für ihren adaptiven Algorithmus eine empirische Laufzeit von $O(n^{2.2})$, wobei es aber sehr unwahrscheinlich ist, dass die Worst-Case-Laufzeit polynomiell ist [Kort 08].

Eine andere, moderne Technik zur Lösung des Travelling-Salesman-Problems wird mit *„Branch-And-Bound“* bezeichet. „Branch-And-Bound“ ist ein algorithmisches Framework, mit dem eine Aufzählung aller Touren simuliert werden kann. Dabei benötigt man eine Partitionierungsvorschrift und eine Methode, eine untere Schranke auf die Lösungen in den Teilen der Partitionen zu berechnen. Die untere Schranke wird verwendet, um zu dem Zeitpunkt, in dem das Verfahren bereits ein Beispiel gefunden hat, alle Partitionen auszuschließen, die nur längere Lösungen enthalten. Zur Berechnung der unteren Schranke kommt beispielsweise die *Held-Karp-Schranke* zum Einsatz. Diese basiert auf einer Lagrange-Relaxation des Travelling-Salesman-Problems und kann in einer Zeit berechnet werden, in der ein minimaler Spannbaum auf $n-1$ Vertices berechnet wird. Für Details zu diesem Verfahren, insbesondere zur Methode der Lagrange-Relaxation und zur Berechnung von minimalen Spannbäumen, sei auf das Lehrbuch [Kort 08] verwiesen.

Einige exemplarische Implementierungen der Lin-Kernighan-Heuristik, des Held-Karp-basierten Branch-And-Bound-Verfahrens, sowie die derzeit wohl „schnellste“ Implementierung eines Lösungsverfahrens für das Travelling-Salesman-Problem, die auf einer Erweiterung des Branch-And-Bound-Frameworks basiert, ist mit allen notwendigen Details in der öffentlich zugänglichen Concorde-Bibliothek [Appl 06] enthalten.

Eine grundlegend andere Methode, mit dem Travelling-Salesman-Problem und beliebigen anderen Optimierungsproblemen umzugehen, ist die Methode der genetischen Algorithmen. Im Grunde ist ein genetischer Algorithmus ein ausreichend zufälliger Prozess, in dem Teillösungen kombiniert werden, um bessere Ergebnisse zu erhalten. So funktinoiert beispielsweise die genetische Evolution sehr erfolgreich.

Ein *genetischer Algorithmus* verwendet eine *Bevölkerung* von *Chromosomen*. Chromosomen sind in diesem Zusammenhang Zeichenketten fester Länge über einem endlichen Alphabet, und kodieren eine mögliche Lösung des zu Grunde liegenden Problems. Im Fall des Travelling-Salesman-Problems wird beispielsweise jeder Stadt ein Buchstabe zugeordnet, und ein Chromosom kodiert so eine Reihenfolge von Städten. Abbildung 5.4 zeigt ein Beispiel. Jedem Chromosom wird ein Wert *Fitness* durch eine Zielfunktion zugeordnet. Im Fall

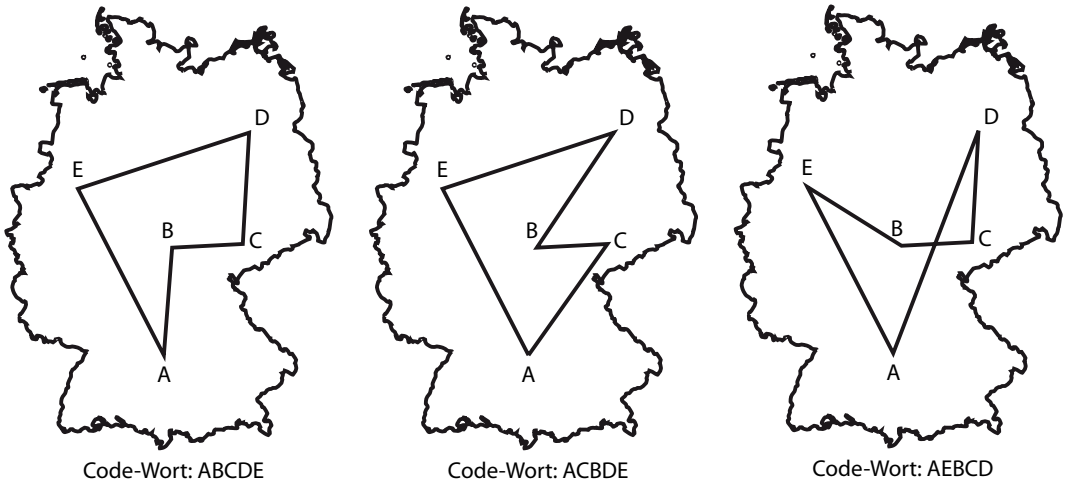


Abbildung 5.4: Verschiedene Touren und ihre Codewörter

des Travelling-Salesman-Problems ist eine typische Zielfunktion die Länge der Tour. Mit Wahrscheinlichkeiten, die dem Verhältnis der verschiedenen Fitness-Werte entsprechen, werden nun in einer Iteration („Generation“) Paare von Chromosomen gebildet, die mit einem *Crossover-Operator* zu neuen Chromosomen kombiniert werden. In die Bevölkerung der nächsten Iteration können nun entweder die beiden neuen Chromosomen eingehen oder die zwei mit den höchsten Fitness-Werten. Der Prozess wird nach einer gewissen Anzahl an Generationen gestoppt, und das Chromosom mit dem besten Fitness-Wert wird als Lösung des Problems ausgegeben. Ein guter Crossover-Operator ist eine Operation, die zwei Chromosomen zu typischerweise zwei neuen Chromosomen kombiniert, sodass das Ergebnis eine Chance hat, besser zu sein als die zwei zuvor ausgewählten Chromosomen. Damit das Verfahren bei komplexen Situationen, etwa die Existenz vieler Extrema in der zu optimierenden Zielfunktion, nicht durch Zufall in einem lokalen Minimum gefangen wird, wird in jeder Generation auch noch eine Mutation mit recht niedriger Wahrscheinlichkeit durchgeführt. Dabei wird ein neues, zufälliges Chromosom erzeugt, und ein anderes Chromosom aus der Generation entfernt.

Diese Algorithmen entziehen sich leider häufig einer genauen Performance-Analyse, da in dem Verfahren so viele zufällige Einflüsse in Abhängigkeit voneinander auftreten, dass auch die Berechnung eines Erwartungswertes oder die empirische Ermittlung einer tatsächlichen Qualität der Ergebnisse nahezu unmöglich wird. Beispielsweise hängt die empirische Gesamtlaufzeit häufig von der Qualität eines Initialisierungsverfahrens ab. Darüber hinaus kennt man in vielen interessanten Fällen die korrekte Lösung eines Problems nur für sehr kleine Instanzen. In diesem Fall können zwar Konstruktionsverfahren von Problemen, deren optimale Lösung durch die Konstruktion bekannt ist, eine gewisse Einsicht geben. Aber solche Verfahren bilden selten einen repräsentativen Teil des Raumes der Probleme ab und eignen sich daher nur bedingt für

allgemeine Aussagen.
Genetische Algorithmen zur Lösung des Travelling-Salesman-Problems liegen schon lange vor. Auch die wichtigsten Austauschoperatoren für das Travelling-Salesman-Problem sind lange bekannt [Gold 85, Gold 89, Davi 85]. Es handelt sich dabei um Operatoren, die eine schwierigere Variante des Travelling-Salesman-Problems lösen: Das *blinde Travelling-Salesman-Problem*. In dieser Variante werden die Kantenlängen erst dann verwendet, wenn eine volle Kandidaten-Tour generiert worden ist und deren Länge ermittelt werden soll. Im Folgenden wird klar werden, dass gerade diese Eigenschaft wichtig sein kann, wenn das Ordnungsproblem von Points-of-Interest in Indoor-Umgebungen kontextabhängig oder mit Unterstützung von dynamischen Wartezeiten beispielsweise an Bushaltestellen gelöst werden soll. Auf die gängigsten Crossover-Operatoren für das Travelling-Salesman-Problem wird im Zusammenhang mit einer genetischen Approximation des geordneten, blinden Travelling-Salesman-Problem im Abschnitt 5.2.2 detailliert eingegangen.

5.2 Die Situation in Gebäuden

Beschränkt man sich auf die Situation in Gebäuden, so wurde bereits ausgeführt, dass einige Eigenschaften, die zu guten Approximationen des Problems führen, nicht vorliegen, und deshalb die klassische Betrachtungsweise nicht gerechtfertigt ist. Beispielsweise ist ein Navigationsgraph nicht unbedingt zusammenhängend: Wenn ein Gast an einem Flughafen kein Ticket hat, so kann er die Sicherheitskontrolle an einem Flughafen nicht passieren. Der Navigationsgraph zerfällt dann in zwei Komponenten. Ein Navigationsgraph ist auch nicht symmetrisch, wie man leicht an einer Rolltreppe erkennt, die nur in eine Richtung gebaut ist. Man kann an dieser Stelle im Gebäude die Ebene nur in eine Richtung wechseln, und muss für den Weg zurück einen anderen Weg wählen. Wie Abbildung 5.2 zeigt, ist ein Navigationsgraph im Inneren von Gebäuden auch nicht metrisch. Folglich kann er auch nicht euklidisch sein, weil euklidische Travelling-Salesman-Probleme offenbar metrisch sind, da die Dreiecksungleichung bei der verwendeten euklidischen Norm erfüllt ist.
Durch die komplexere Geometrie der Navigationsgraphen funktionieren die meisten der bekannten Approximationen nicht wie gewünscht. Deshalb wurde das Problem in dieser Situation grundlegend neu analysiert. Im Ergebnis liefert eine sehr klassische Art und Weise, mit dem Problem umzugehen, die besten Ergebnisse. Erfreulicherweise kommt die Methode der genetischen Algorithmen, die in den letzten Jahren im Bereich der Forschung zum Travelling-Salesman-Problem stark an Bedeutung verloren hat, in der speziellen Situtation der Indoor-Navigation wieder zur Geltung.
Im System HIPS wird ein flächendeckender Ansatz verfolgt: Da die topologischen Gebäudeinformationen eine deutlich höhere Lebensdauer haben als die Positionen von interessanten Punkten – etwa durch saisonale Änderungen wie Weihnachtsmärkte – muss der Dienst transparent für die gesamte Fläche

der Navigationsanwendung erbracht werden können. Zu diesem Zweck wird in HIPS ein regelmäßiger Zugangsgraph verwendet, der mit einer konstanten, hohen Auflösung die begehbare Fläche abstrahiert – siehe Abschnitt 4.3.1 und Abbildung 4.10(b). Eine weitere Anforderung, die an das System HIPS gestellt wurde, ist die Unterstützung von Kontextänderungen wie Gate-Änderungen, Wartezeiten und die Integration von Intervall-Diensten wie Buslinien. In der Summe folgt aus diesen Anforderungen und deren Umsetzung, dass keine Ergebnisse zwischengespeichert werden können, da die Reisezeit zwischen zwei Orten nicht unbedingt konstant ist: Wenn man einen eingeplanten Bus verpasst hat, dauert die Reise in der Regel etwas länger. Zunächst sei jedoch das geordnete Problem noch einmal exakt formuliert:

Problem: *Sei G ein vollständiger, endlicher Graph mit nicht notwendig konstanten, nicht-negativen Kantengewichten. Sei eine Tour eine Permutation der Ecken von G, welche die Ecken von G in einer Reihenfolge anläuft, die der vorgegebenen Ordnung nicht widerspricht. Sei weiter die Länge der Tour die Summe der Kantengewichte der Kanten zum Zeitpunkt der Ankunft an der Anfangsecke jeder Kante. Finde eine Tour mit minimaler Länge.*

In der Regel sind die Gewichte des Graphen eine Funktion der Zeit, und die Längenberechnung wird mit einer Schätzung der Ankunftszeit an jeder Ecke die tatsächliche Wegzeit für die Tour angeben. Es können aber auch andere Gründe für Änderungen von Kantengewichten verantwortlich sein, wie etwa die durch eine möglicherweise gemessene Länge einer Warteschlange induzierte Verzögerung.

Falls im vorangegangenen Problem die Gewichte konstant sind, so handelt es sich um das *Sequential Ordering Problem*. Dieses Problem ist im Allgemeinen gut untersucht. Eine Serie von Papern von Escudero [Escu 88, Asch 93, Escu 94] zeigt, dass sich das Problem im Wesentlichen mit denselben Methoden lösen lässt wie das klassische Travelling-Salesman-Problem. Dazu gibt Escudero an, wie man bei geeigneter Formulierung des Travelling-Salesman-Problems eine Lagrange-Relaxation im Stile der „Branch-And-Bound“ Methode mit der Held-Karp-Schranke verfolgen kann. Dabei wird in der Arbeit eine Methode vorgestellt, mit der man die Ordnungsrelation als Polyeder beschreiben kann, und mit der dann die Ordnugsrelation weggelassen („Relaxation“) werden kann, und, im Falle, dass der bei der Relaxation entstandene Kandidat nicht wohlgeordnet ist, eine effiziente Schnittebene findet, mit der der Kandidat und ein gewisser Teil der nicht-optimalen Kandidaten effizient ausgeschlossen werden können.

Es gibt einige sehr wichtige Unterschiede zwischen dem klassischen Travlling-Salesman-Problem und dem hier betrachteten geordneten Travelling-Salesman-Problem. Zunächst hat natürlich das klassische Problem keine Ordnungsrelation, die ein Weg erfüllen muss. Auch werden im vorliegenden Problem nicht immer Lösungen gesucht, die auch einen Weg zurück zum Ursprung haben. Dies führt dazu, dass das vorliegende Problem komplexer ist, denn für das klassische Travelling-Salesman-Problem ist der Startort nicht relevant, weil ei-

ne kürzeste Tour mit einem Startpunkt auch die kürzeste Tour für alle anderen Startpunkte ist. In der vorliegenden Situation wird eine solche Rotation des Startpunktes entlang der Lösung jedoch in der Regel die Ordnung verletzen.
Im Folgenden werden zwei Lösungsansätze zum geordneten Travelling-Salesman-Problem beschrieben, und dann die Ergebnisse an Hand von realistischen Daten des Flughafens München evaluiert. Anschließend wird gezeigt, wie man im Falle eines euklidischen geordneten Travelling-Salesman-Problems die Ordnungsrelation mit erstaunlicher Qualität durch eine geometrische Translation des Problems relaxieren kann, und dann ein gewöhnliches euklidisches Travelling-Salesman-Problem mit einer beliebigen Methode, beispielsweise dem Arora-Approximations-Schema, lösen kann.
Als Eingabe stehen den im Weiteren beschriebenen Algorithmen die folgenden Informationen zur Verfügung.

- Eine nicht sortierte Menge an Zielen, die Knoten im Navigationsgraphen sind,
- eine vollständige Distanzmatrix für Verbindungen und Verbindbarkeit dieser Punkte – diese darf natürlich unendliche Werte enthalten,
- eine natürliche Ordnungszahl für jeden Point-of-Interest in der Menge.

5.2.1 Erschöpfende Suche (Brute-Force)

Der naheliegendste Algorithmus erzeugt zunächst alle Permutationen von Städten, prüft für jede Permutation, ob sie wohlgeordnet ist, und berechnet dann die Gesamtlänge der Tour. Dabei führt er über die bisher kürzeste wohlgeordnete Tour Buch. Nachdem der Suchraum durchlaufen ist, ist natürlich diese Tour die insgesamt kürzeste wohlgeordnete Tour.
An dieser Stelle ist es auch möglich, nicht zuerst alle Permutationen zu erzeugen und erst später die Wohlordnung zu überprüfen. Ein Grund, weshalb man alle Permutationen erzeugen sollte, liegt darin, dass es zu diesem Vorgehen keine asymptotisch schnellere Alternative gibt. Man müsste nämlich – wegen der Definition der Navigationsschnittstelle – eine Sortierung durchführen, um die Points-of-Interest gemäß ihrer Ordnungszahlen zu sortieren, und danach müsste für jede dieser Klassen die Menge aller Permutationen erzeugt werden. Asymptotisch ist das nicht schneller, denn die Größe einer Klasse ist keineswegs beschränkt, und auch in der Praxis kommen viele Beispiele vor, in denen effektiv nur eine Ordnungsklasse beiträgt: Beispielsweise unterliegt ein Besucher im Besucherbereich in der Regel keinen Einschränkungen außer der, dass die erste Ecke der Tour durch seinen Aufenthaltsort feststeht.
Dieser Algorithmus ist offenbar korrekt und kann für kleine Instanzen des Problems verwendet werden, um eine optimale Lösung zu berechnen. Diese Tatsache wird zur Errechnung relativer Fehler in der Evaluation verwendet.

5.2.2 Ein genetischer Algorithmus für das geordnete Travelling-Salesman-Problem

Wie im Abschnitt 5.1.2 beschrieben, basiert ein genetischer Algorithmus auf einer Strategie der natürlichen Selektion („Survival of the Fittest“) und verwendet einen partiellen und zufällig beeinflussten Informationsaustausch zwischen potentiellen Lösungen des Problems. Im vorliegenden Fall ist ein Chromosom definiert als eine Sequenz von ganzen Zahlen, welche die gegebenen Points-of-Interest in beliebiger Weise aufzählen. Der Fitness-Wert ist dann das Inverse der Länge der Tour. So führen kürzere Touren zu höheren Fitness-Werten. Dabei wird grundsätzlich mit einer festen Populationsgröße von Chromosomen in der folgenden Weise gearbeitet: Für eine fest vorgegebene Anzahl an Generationen wird der Fitness-Wert jeden Chromosoms der Population berechnet, und mit einer Roulette-Technik zwei Individuen der jeweiligen Generation ausgewählt. Die Wahrscheinlichkeit, dass ein bestimmtes Chromosom ausgewählt wird, ist dabei durch das Verhältnis des Fitness-Wertes des Chromosoms zur Summe aller Fitnesswerte gegeben. Ein Chromosom mit hohem Fitness-Wert wird also mit höherer Wahrscheinlichkeit ausgewählt als ein Chromosom mit niedrigem Fitness-Wert. Mit den zwei ausgewählten Chromosomen wird dann eine Austauschoperation durchgeführt, bei der genetisches Material nach den Vorgaben eines sogenannten „Crossover-Operators“ ausgetauscht wird. Aus den ausgewählten und den neu erzeugten Chromosomen werden nun im Selektionsschritt die beiden mit dem höchsten Fittness-Wert ausgewählt. Diese Vorgehensweise ist nicht immer optimal, weil man mit der gesamten Bevölkerung in einem lokalen Minimum gefangen sein kann, aus dem der Crossover-Operator durch die nicht-randomisierte Auswahl der stärksten Individuen nicht mehr herauskommt. Auf der anderen Seite wird aber dafür gesorgt, dass die beste, bisher gefundene Lösung des Problems nicht auf Grund von Zufällen verloren geht. Um die mangelnde Zufälligkeit bei der Auswahl der Chromosomen für die nächste Population auszugleichen, wird eine Mutation eingeführt. Hierbei wird in jeder Generation mit einer kleinen Wahrscheinlichkeit ein zufällig ausgewähltes Chromosom durch ein zufällig erzeugtes neues Chromosom ersetzt.

Die wichtigste Operation für diese Art von genetischem Algorithmus ist die Crossover-Methode. Denn nur bei der Crossover-Operation wird – von zufälligen Verbesserungen abgesehen – für eine Verbesserung der Population, und damit auch für eine Verbesserung des Ergebnisses, gesorgt. Für das Travelling-Salesman-Problem wurden bereits sehr viele Crossover-Methoden angegeben.

Der bekannteste Operator ist wohl der „Partially-Matched-Crossover (PMX)“-Operator von Goldberg [Gold 89]. Er berechnet zwei zufällige Positionen innerhalb der Chromosomen und vertauscht die Anteile der Chromosomen, die zwischen diesen Grenzen liegen. So ergeben sich zwei neue Chromosomen. Da diese beiden neuen Chromosomen in der Regel keine Touren sind – sie sind es genau dann, wenn in beiden Chromosomen zwischen den zufällig gewählten Positionen dieselben Ziffern vorkommen – wird eine Reparatur-Operation not-

wendig. Dafür werden bei PMX die Austäusche zwischen Ziffern innerhalb der Austauschregion außerhalb der Austauschregion rückwärts auf Doppelungen angewendet. Damit erhält man schlussendlich eine Tour, da jede Doppelung mit diesem Verfahren aufgelöst werden kann. Ein großer Nachteil dieses Operators liegt darin, dass ein großer Crossover-Bereich mit hoher Wahrscheinlichkeit zu solchen Rücktauschoperationen führt und damit lokal optimale Teillösungen zerstört werden können.

Ein besserer Crossover-Operator für diesen Fall ist der „Ordered-Crossover (OX)"-Operator. Ordered-Crossover arbeitet ähnlich wie PMX: Zwei Positionen werden als Austauschgrenzen ausgewählt, und die Chromosomenblöcke dazwischen werden ausgetauscht. Dann wird aus dem Ursprungschromosom mit Ziffern aufgefüllt, wobei Doppelungen ausgelassen werden. Damit ergibt sich auf jeden Fall eine Tour, und etwas mehr von der Ordnungsstruktur der Chromosomen bleibt erhalten.

Eine weitere populäre Möglichkeit, einen Crossover-Operator für das Travelling-Salesman-Problem zu definieren, basiert auf der Verwendung von heuristischen Informationen. Der „Heuristic Crossover"-Operator [Pál 93, Gref 85] wählt eine zufällige Stadt aus, und fügt die kürzeste Kante, auf der die Stadt in den Eltern-Chromosomen verlassen wird, in die Kind-Chromosomen ein. Nachdem eine erste solche Kante gewählt ist, verfährt der Algorithmus analog mit der zweiten Kante, bis eine Stadt in der Tour ein zweites Mal besucht wird. Da das Einfügen dieser Kante nicht zu einer zulässigen Tour führt, wird eine zufällige Kante ausgewählt, die nicht zu einem solchen Zyklus führt. Mit diesem Verfahren wird dann die Kind-Generation berechnet.

Im Falle des geordneten Travelling-Salesman-Problems ist dieser Ansatz aber nicht sinnvoll anwendbar ohne eine Backtracking-Strategie für den Fall, dass überhaupt keine Kante ohne Zyklus mehr eingefügt werden kann. Diese Situation kann dann auftreten, wenn alle noch sinnvoll einfügbaren Kanten die Ordnung verletzen. Damit würde aber die Crossover-Operation sehr komplex und ein unkalkulierbares Risiko für die Laufzeit des Gesamtalgorithmus.

Für das geordnete Travelling-Salesman-Problem wird die Hauptidee des OX-Operators aufgegriffen, der versucht, Teile der Anordnung von Städten zu erhalten [Wern 11c]. Das reicht zwar nicht immer aus, aber man kann den OX-Operator um eine Rotationsoperation erweitern, sodass aus wohlgeordneten Elternchromosomen auch wohlgeordnete Kindchromosomen erzeugt werden:

Seien $A = (a_i)_{i=0..n}$ und $B = (b_i)_{i=0..n}$ Chromosomen. Seien weiter zwei zufällige Austauschgrenzen $0 \leq k < l \leq n$ gewählt. Dann werden Kindchromosomen C und D erzeugt, indem $c_i = b_{i+k}$ für $i = 0..l-k$ und $d_i = a_{i+k}$ für $i = 0..l-k$ gesetzt wird und dann die noch fehlenden Ziffern von C (bzw. D) in der Reihenfolge aufgefüllt werden, in der sie auch in A (bzw. B) nach der oberen Austauschgrenze auftreten. Wenn über das Ende der Chromosomen A und B hinausgegangen wird, wird wieder am Anfag begonnen. Beim Auffüllen werden die Elemente übersprungen, die bereits in C und D enthalten sind.

Ein Beispiel für diese Chrossover-Operation ist in Abbildung 5.5 zu finden. Es

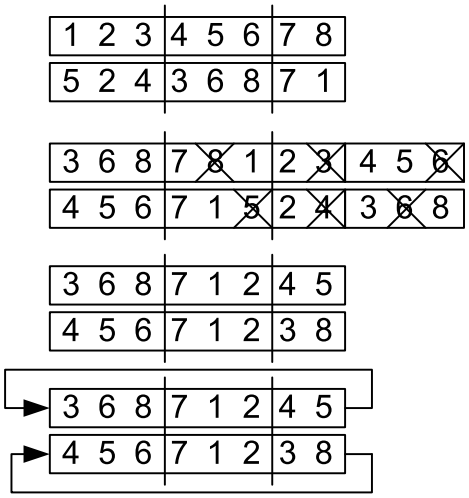


Abbildung 5.5: Beispiel des definierten Crossover-Operators

ist nun wichtig, den Fehler, der durch den Operator an der Ordnung erzeugt wird, wieder zu reparieren. Das ist möglich, denn die folgenden Aussage ist leicht zu sehen:

Lemma: *Für zwei wohlgeordnete Chromosomen* $A = (a_i)_{i=0..n}$ *und* $B = (b_i)_{i=0..n}$ *(also* $a_i \leq a_{i+1}$ *und* $b_i \leq b_{i+1}$ *für* $i = 0..n-1$ *in der vorgegebenen Ordnung) erlaubt das Ergebnis des Crossover-Operators eine Rotation, sodass die Kind-Chromosomen wohlgeordnet sind.*

Es wird ja alles in der selben Ordnung rekombiniert, in der es in der Eltern-Generation vorkam. Wenn ein Element ausgelassen wird, so ist es eine Folgerung der Transitivität der Ordnungsrelation, dass die Ordnung auch nach dieser Auslassung unverletzt bleibt. Ausgenommen ist lediglich die Stelle, an der an den Anfang des Elternchromosoms gesprungen wird. Man kann nun die Ergebnisse so rotieren, bis sich diese eine Stelle im Kind-Chromosom am Anfang befindet.

Neben dem Crossover-Operator ist es noch wichtig, dass man das Problem der Initialisierung löst. Denn bevor der genetische Verbesserungsprozess seine Arbeit aufnehmen kann, muss eine möglichst zufällige Grundpopulation erzeugt werden. Das Erzeugen eines zufälligen Chromosoms ist auch für die Mutationsoperation notwendig, die bei der vorliegenden Konstruktion des Algorithmus essentiell ist, weil der Selektionsschritt deterministisch die besseren Kandidaten verwendet.

Ein zufälliges Chromosom ist sehr leicht erzeugt. Man berechnet einfach eine zufällige Permutation und wendet diese auf die Grundtour $(0, 1, 2, 3, \ldots)$ an. Um nun die vorgegebene Ordnung zu erfüllen, kann man beispielsweise diese Permutation nach Ordnungsklassen sortieren. Eine andere Möglichkeit besteht darin, dass man die vorgegebenen Points-of-Interest einmal nach den Ordnungsklassen sortiert, und dann zufällige Permutationen der in den einzelnen Klassen beteiligten Points-of-Interests aneinander hängt. Im Rahmen des

Projektes HIPS wurde aber darauf verzichtet, diese Sortierung vorab herbeizuführen und es wurde auch kein vollständiger Sortieralgorithmus, sondern eine Variante von Bubble-Sort verwendet, die nach einigen Korrekturen abbricht und ein neues Chromosom erzeugt.
Das Problem, weshalb die vorherige Sortierung der Points-of-Interest nicht gewählt wurde, besteht darin, dass es in einem nicht-vollständigen Graphen auch dazu kommen kann, dass die partielle Ordnung nicht erfüllt werden kann. Also kann auch keine zulässige Tour gefunden werden. Ferner ist für den Fall der kontextabhängigen Berechnung der Routenlänge auch ein Einfluss auf die partielle Ordnung denkbar. Beispielsweise gibt es einige Passstraßen, die zu bestimmten Tageszeiten nur in eine Richtung befahren werden können, oder die Richtung einer Rolltreppe kann dem vorherrschenden Bedarf angepasst werden. Deshalb wird hier ein „blindes“ Verfahren gewählt, welches die partielle Ordnung erst auswertet, wenn der Kandidat für die Tour vollständig definiert ist. Außerdem kann der Algorithmus in dieser Form auch mit partiellen Ordnungen umgehen, wenn also nicht zwischen allen Paaren von Points-of-Interest feststeht, in welcher Relation sie zueinander stehen. In diesem Fall ist eine Sortierung nicht ohne Weiteres möglich, es kann aber immer noch untersucht werden, ob die partielle Ordnung verletzt ist oder nicht. Insgesamt war diese Vorgehensweise für den speziellen Anwendungsfall und die zu erwartenden Anfragen sehr gut geeignet. Dies liegt zum einen daran, dass in der Praxis der Anwendung auch viele Anfragen überhaupt nur eine Ordnungsklasse enthalten und dass – zumindest am Flughafen München – innerhalb einer Ordnungsklasse kaum Kanten existieren, die nicht in beide Richtungen begehbar sind und somit die Wahrscheinlichkeit, dass ein Problem unlösbar ist, relativ gering war.

5.2.2.1 Evaluation der Methode

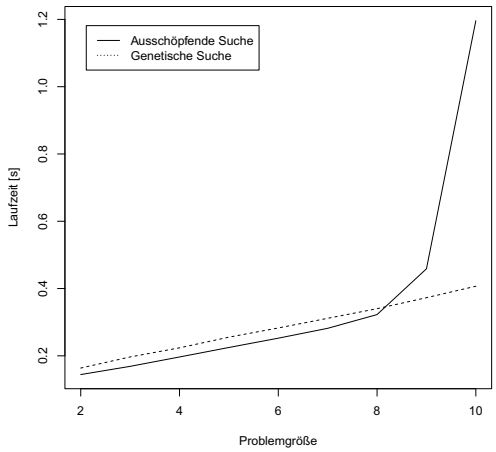
Für die Evaluierung wurden beide Algorithmen für das Navigationssystem HIPS am Münchner Flughafen implementiert. Dieses System basiert auf einem großen regelmäßigen Zugangsgraphen und modelliert die Ordnungsrelation dadurch, dass den Points-of-Interest in dieser Umgebung Ordnungsklassen zugewiesen werden können. So hat typischerweise der Point-of-Interest, an dem der Nutzer mit dem System in Interaktion tritt, die Ordnungsnummer 0, die Points-of-Interest im öffentlichen Bereich haben die Ordnungsnummer 1, nach der Sicherheitskontrolle die Ordnungsnummer 2 und so weiter.
In der Implementierung wird zunächst eine asymmetrische Distanzmatrix aufgestellt, indem n Dijkstra-Suchen [Dijk 59] die Entfernungen zwischen den Punkten berechnen. An dieser Stelle hätte man zwar auch bessere Methoden wie den Floyd-Warshall-Algorithmus [Floy 62] verwenden können, aber es wurde bewusst auf bestehende Routing-Funktionalität zurückgegriffen, damit bei Änderungen der Routing-Funktionalität oder der Bewertungen nur an einer Stelle Quellcode angepasst werden muss.
Der Graph besteht aus fast 65.000 Ecken, die durch 415.000 Kanten miteinander verbunden sind. Er umfasst und modelliert das gesamte öffentlich zugäng-

liche Gebiet des Flughafens München. Der Graph wurde mit den Methoden aus Abschnitt 4.3 semi-automatisch aus CAD-Daten des Flughafens München extrahiert und besitzt daher eine sehr regelmäßige Struktur. Für den Vergleich der Verfahren wurden 220 zufällige Beispielprobleme erzeugt, jeweils 20 für Problemgrößen zwischen 2 und 12 Points-of-Interest. Diese Zufallsprobleme wurden so gewählt, dass sie lösbar sind. Denn es kommt in der Praxis im Graphen des Flughafens nicht vor, dass das Navigationssystem ein unlösbares Problem konstruiert, insbesondere, weil das Nutzerinterface nur die Points-of-Interest anzeigt, die vom aktuellen Ort auch angelaufen werden können.

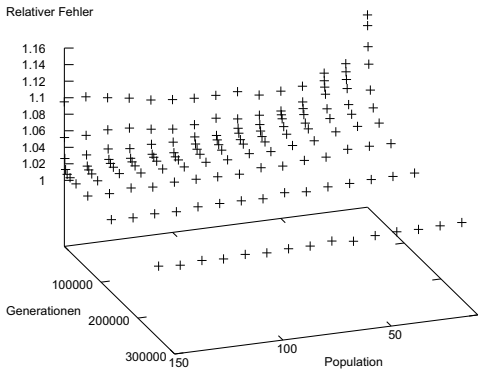
Die Ergebnisse und Laufzeiten, die benötigt wurden, um diese Probleme sequentiell auf typischer Desktop-Hard- und Software (Intel Core2 Duo P8400 @ 2.26 GHz, Linux 2.6) zu lösen, sind in Abbildung 5.6(a) angegeben. Der genetische Algorithmus verwendet 50 Chromosomen und 2000 Generationen, und hat eine Mutationswahrscheinlichkeit von 0.01 bezogen auf das Individuum.

Auf den ersten Blick scheint es, dass der genetische Algorithmus für kleine Probleme langsamer ist als die Methode der erschöpfenden Suche. Das ist damit zu begründen, dass die Aufstellung der Initialpopulation und das Durchlaufen der Generationen für kleine Instanzen aufwändiger ist als die erschöpfende Suche. Für Instanzen mit 9 Points-of-Interest allerdings übersteigt die Laufzeit der ausschöpfenden Suche die Marke von $1s$. Probleme der Größe 9 wurden im Durchschnitt in $1.19s$ gelöst. Im Vergleich dazu berechnet der genetische Algorithmus ein Ergebnis für alle gewählten Größen vom genetischen Algorithmus in weniger als einer halben Sekunde. Entsprechend kann man an dieser Stelle festhalten, dass bis zu einer Größe, die durch eine Laufzeitbegrenzung festgelegt ist, die ausschöpfende Suche ausgeführt werden sollte, und dann für größere Instanzen die genetische Approximation.

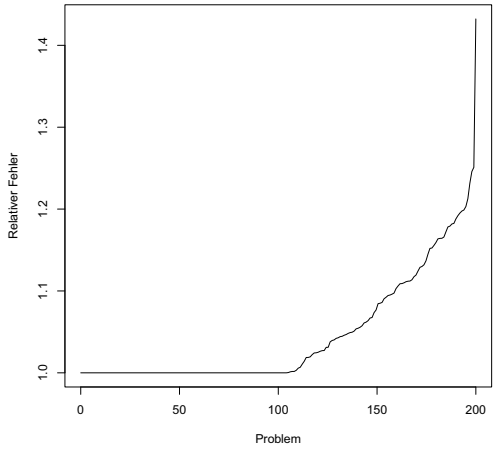
Allerings ist der genetische Algorithmus nicht korrekt: Er berechnet nicht unbedingt die kürzeste Lösung. Außerdem hängen die Laufzeit und die Ergebnisqualität des genetischen Algorithmus stark von den Konfigurationsparametern ab. Deshalb zeigt Abbildung 5.6(b) die Abhängigkeit der relativen Fehler von diesen Parametern für eine Menge von 200 lösbaren Problemen. Der relative Fehler ist der Quotient aus der Länge der gefundenen Tour und der Länge der tatsächlich kürzesten Tour, und konnte daher nur für die Instanzen berechnet werden, für die auch der korrekte Algorithmus der ausschöpfenden Suche in einer vernünftigen Zeit terminiert. In Abbildung 5.6(d) wird der durchschnittliche, relative Fehler auf der Y-Achse gegen die beiden wichtigsten Konfigurationsparameter „Anzahl Chromosomen“ und „Anzahl Generationen“ dargestellt. Im Wesentlichen sieht man einen klaren Anstieg in beide Richtungen: Mit beiden Parametern sinkt der relative Fehler und es steigt damit die Qualität der Approximation. Mit den Werten einer Population von 50 Chromosomen und von 2000 Generationen, wie sie für die Laufzeit-Evaluation verwendet wurden, findet man einen durchschnittlichen relativen Fehler von 1.046. Das bedeutet, dass im Benchmark-Set Touren gefunden werden, die im Durchschnitt um 5% länger sind als die kürzeste Tour. Es gibt hierbei aber auch eine relativ starke



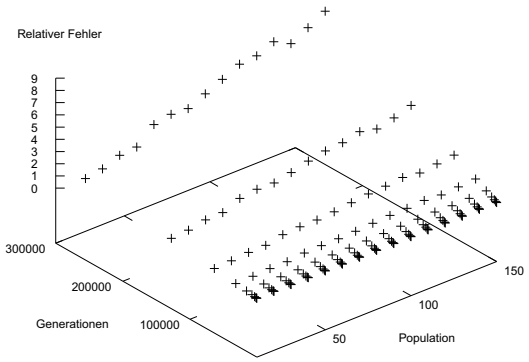
(a) Erschöpfende Suche im Vergleich zum Genetischen Algorithmus



(b) Relativer Fehler des Genetischen Algorithmus



(c) Verteilung relativer Fehler



(d) Einfluss der Parameter des genetischen Algorithmus auf den relativen Fehler (20 zufällige Probleme der Größe 8)

Abbildung 5.6: Empirische Evaluation der Verfahren auf einer Benchmark-Menge von 200 lösbaren, zufälligen PoI-Ordnungs-Problemen

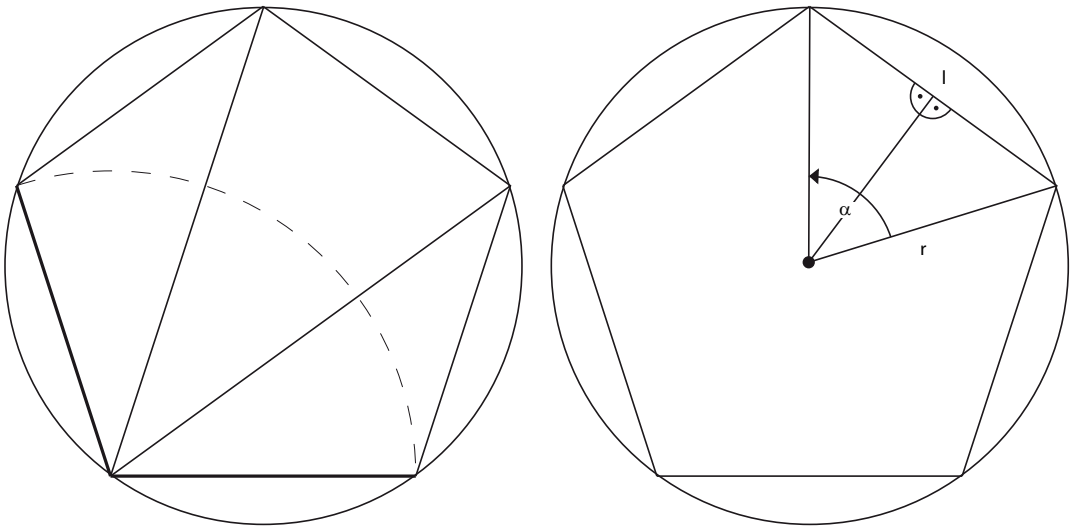
Streuung: 5 von den 200 Problemen waren um 20% bis 25% länger als die optimale Lösung, und ein Problem war um 45% zu lang. Die Verteilung der relativen Fehler im Benchmark-Set ist in Abbildung 5.6(c) zu sehen. An dieser Stelle sollte man beachten, dass die beste klassische Approximation für das metrische Travelling-Salesman-Problem ohne Ordnungsrelation einen relativen Fehler von bis zu 1.5 hat, also auch Touren angibt, die bis zu 50% zu lang sind. Die Mehrheit (52%) der Probleme wurde optimal gelöst, und 82% wurden mit

einem Mehrweg von weniger als 10% gelöst.

5.2.3 Schwache geometrische Relaxation der Ordnungsrelation für den euklidischen Fall

Dieser Abschnitt stellt eine Möglichkeit vor, bei Hinzunahme gewisser Voraussetzungen die Ordnungsrelation auf andere Weise zu behandeln. Insbesondere vor dem Hintergrund der jahrzehntelangen Forschung zum klassischen Travelling-Salesman-Problem wäre eine Übertragung der allgemeinen Ergebnisse auf ein geordnetes Problem interessant. Denn für das allgemeine Travelling-Salesman-Problem gibt es viele Bibliotheken, mit denen man mit moderatem Aufwand gute Ergebnisse erzielen und natürlich die Algorithmen in gut getesteter Implementierung übernehmen kann.
Vor diesem Hintergrund wird ein euklidisches, geordnetes, vollständiges Travelling-Salesman-Problem betrachtet. Sei also eine endliche Menge $V \subset \mathbb{R}^2$ an Ecken gegeben.[1] Sei weiter eine Ordnung durch k Ordnungszahlen als surjektive Abbildung $O : V \to \{1, \ldots, k\} \subset \mathbb{N}$ gegeben. Im Folgenden wird das *vollständige geordnete Travelling-Salesman-Problem* auf V mit Kantengewichten, die von der euklidischen Norm auf $\mathbb{R}^2$ induziert werden, betrachtet.
Zu diesem geordneten Travelling-Salesman-Problem wird eine Abbildung ϕ angegeben, die das Problem in ein ungeordnetes, euklidisches Travelling-Salesman-Problem überführt, sodass das Urbild der optimalen Tour dieses ungeordneten Problems zumindest die Ordnungsrelation erfüllt. Es wird dann empirisch gezeigt, dass auch die Länge des Urbildes der optimalen Tour relativ kurz ist. Allerdings wird es nicht so sein, dass das Urbild der optimalen Tour des ungeordneten Problems immer optimal ist. In diesem Sinne handelt es sich um eine *schwache Relaxation* der Ordnungsrelation. Man kann mit einem moderaten Erwartungswert für den relativen Fehler die Ordnungsrelation durch die Abbildung ϕ aus dem Problem eliminieren.
Als Vorbereitung für die Konstruktion der Abbildung sei festgehalten, dass eine Tour $T = (v_i)$ per Definitionem genau dann wohlgeordnet ist, wenn $O(v_i) \leq O(v_j)$ für alle $i \leq j$ gilt. Es folgt, dass in einer wohlgeordneten Tour der Nachfolger v_{i+1} einer jeden Ecke v_i, außer im Sonderfall der Rückkante, ein minimales Element der noch nicht verwendeten Ecken $V - \{v_1 \ldots v_i\}$ sein muss. Somit treten dann in einer wohlgeordneten Tour die Ecken einer jeden Ordnungsklasse zusammen, also ohne Unterbrechung durch Elemente anderer Odrnungsklassen, auf. Um diese Tatsache zu verwenden, bildet die Abbildung ϕ die Klassen selbst isometrisch innerhalb des Raumes $\mathbb{R}^2$ ab.
Um nun die Ordnungsrelation zwischen Klassen durch einen geometrischen Trick zu eliminieren, werden die euklidischen Travelling-Salesman-Probleme betrachtet, die durch die Ecken von regulären n-Ecken (für $n > 3$) gegeben sind. Es ist offensichtlich, dass die kürzeste Tour in diesen Fällen die übliche Linie des n-Ecks ist, da alle Sekanten zwischen Nicht-Nachbarn länger sind

[1] $\mathbb{R}^2$ kann leicht zu $\mathbb{R}^n$ verallgemeinert werden.



(a) Längenverhältnisse zwischen Sekanten (b) Geometrie der Sekantenlänge

Abbildung 5.7: Sekanten in n-Ecken

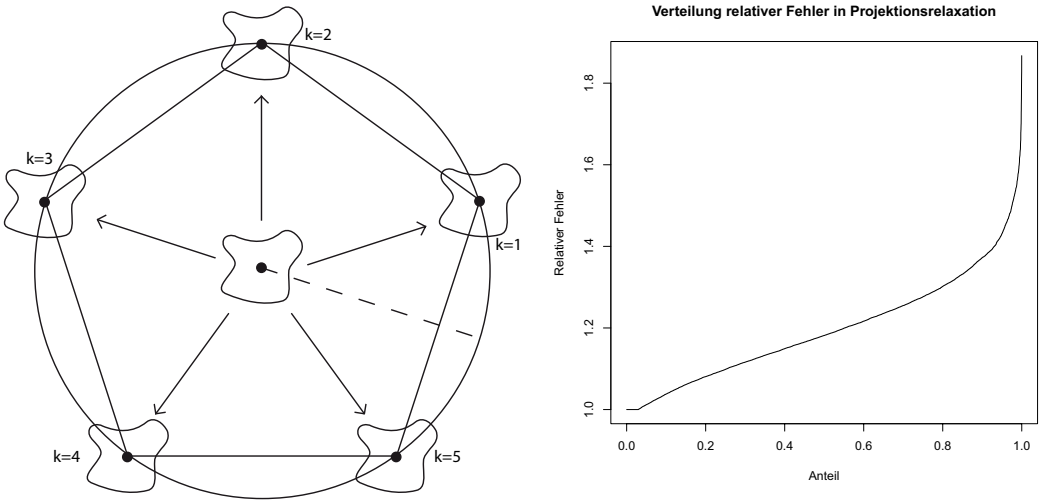
als die Sekanten zwischen direkten Nachbarn. Ein Beispiel ist durch den Kreis und alle Sekanten in Abbildung 5.7(a) gegeben. In Abbildung 5.7(b) sieht man, dass eine Sekante maximal den Mittelpunktswinkel $\alpha = \pi$ haben kann, und dass unter Verwendung der Winkelhalbierenden und der entstehenden zwei rechtwinkligen Dreiecke die folgende Gleichung für die Sekantenlänge l unter Verwendung der beiden rechtwinkligen Dreiecke gilt:

$$l = 2 \sin\left(\frac{\alpha}{2}\right) r \tag{5.2}$$

Da in dieser Gleichung für den Sinus nur Argumente aus $[0, \pi/2]$ auftreten, folgt aus der strengen Monotonie des Sinus auf diesem Intervall auch die strenge Monotonie für die Sekantenlängen. Diese Tatsache wird verwendet, um mit einer Abbildung ϕ die Ordnungsrelation durch ebendiese geometrische Tatsache zu erzwingen.

Sei nun ϕ wie folgt gegeben: Zu jeder Ordnungsklasse l wird ein Klassenwinkel $\alpha_l = 2\pi l/k$ gewählt, und jede Ecke in dieser Klasse entlang des Strahles vom Ursprung in Richtung $(\cos(\alpha_l), \sin(\alpha_l))$ um einen noch zu wählenden, festen Radius R verschoben. Dann ist diese Abbildung innerhalb der Klassen isometrisch, da es sich um eine für die Klasse identische Translation handelt. Fasst man nun die Klassen als Punkte auf, entsteht aber genau ein k-Eck mit dem vorgegebenen Radius R. Abbildung 5.8(a) zeigt die Abbildung ϕ in schematischer Weise.

Diese Abbildung ist schon ein guter Kandidat dafür, dass eine Lösung des transformierten Travelling-Salesman-Problems wohlgeordnet ist: Die Klassen kommen „geschlossen“ vor, wenn R so groß gewählt ist, dass eine unnötige, zusätzliche Kante zwischen zwei Klassen im Vergleich zu den Alternativen zu



(a) Schematische Darstellung der Abbildung ϕ

(b) Verteilung der relativen Fehler bei zufälligen, geordneten Travelling-Salesman-Problemen

Abbildung 5.8: Schematische Darstellung der Abbildung und Ergebnisse der Evaluation

teuer wird, und die Geometrie der Klassen bildet in grober Näherung ein k-Eck, es besteht also die Chance, dass das Urbild der kürzesten Tour wohlgeordnet ist, wenn die Abweichung vom tatsächlichen k-Eck, die durch den Abstand der Ecken des ursprünglichen Problems vom Nullpunkt entsteht, vernachlässigt werden kann.

In der Tat kann man den Radius R ausreichend groß wählen, denn das Argument, warum bei einem regulären k-Eck immer die gewöhnliche Randlinie des k-Ecks die optimale Tour ist, basiert auf der Darstellung der Sekantenlänge in Gleichung 5.2. Denn diese zeigt, dass die Sekanten zu den Nachbarn die kürzesten Sekanten unter allen Sekanten mit Beteiligung der Ecke v_i sind.

Sei nun r_k die maximale Distanz einer Ecke des ursprünglichen Problems V zum Nullpunkt. Dann kann durch ungünstige Konstellation der Punkte in den Klassen der Abstand zum Nachbarn um maximal $2r_k$ vergrößert und der Abstand zu den anderen Nachbarn um $2r_k$ verkleinert werden. Beide Fälle können zwar nicht gleichzeitig eintreten, dennoch kann R so gewählt werden, dass die folgenden Ungleichungen gelten – K_i bezeichne hier die Klassenmittelpunkte:

$$d(K_i, K_{i+1}) + 2r_k < d(K_i, K_{i+2}) - 2r_k$$

Nimmt man nun die Darstellung der Sekantenlänge aus Gleichung 5.2 und formt diese Ungleichung um, sodass der Radius R auf der linken Seite steht,

erhält man (wegen $\sin\left(\frac{2\pi}{n}\right) > \sin\left(\frac{2\pi}{2n}\right)$ für $n > 3$) mit

$$R > \frac{2r_k}{\sin\left(\frac{2\pi}{n}\right) - \sin\left(\frac{2\pi}{2n}\right)}$$

eine mögliche Wahl.
Insgesamt ist also folgende Situation erreicht: Es ist möglich, ein geordnetes, euklidisches Travelling-Salesman-Problem mit der vorgestellten Abbildung ϕ zu transformieren, und den Radius in dieser Transformation so zu wählen, dass die Unterschiede zum regulären k-Eck so gering sind, dass die optimale Tour die Klassen in der vorgegebenen, richtigen Reihenfolge durchlaufen muss. Also kann damit eine korrekt geordnete Tour für das Ursprungsproblem konstruiert werden, da die Abbildung ϕ auf V bijektiv ist, und das Urbild der optimalen Tour des transformierten Problems verwendet werden kann.
Bei der Abbildung auf das n-Eck werden aber eventuell Fehler gemacht: Ursprünglich günstige Wechselkanten zwischen Klassen könnten durch die Abbildung benachteiligt werden. Diese Fehler können in Ausnahmefällen zu großen relativen Fehlern führen, in der Praxis ist diese Fehlerquelle aber nicht sehr stark. Besonders dann, wenn man den Vergleich mit anderen Approximationen sucht.
Um diesen Fehler zu untersuchen, wurden 10.000 zufällige Probleme auf dem Quadrat $[0, 100] \times [0, 100]$ durch zwei unabhängige Gleichverteilungen für die X- und Y-Koordinate erzeugt, durch die vorgestellte schwache geometrische Relaxation gelöst und das Ergebnis mit der durch den Algorithmus der ausschöpfenden Suche aus Abschnitt 5.2.1 bekannten, optimalen Lösung verglichen. Der Mittelwert der relativen Fehler lag bei 1.197, die Touren waren also im Durchschnitt um 19,7% zu lang, das schlechteste Resultat hatte einen relativen Fehler von 1.867, war also um 86,7% länger als die optimale Tour. Dennoch ist die Verteilung, wie in Abbildung 5.8(b) zu sehen, vielversprechend.
An dieser Stelle kann eine Nachoptimierung mit einem Algorithmus der lokalen Suche, beispielsweise k-Opt, durchgeführt werden, oder die Ergebnisse werden zur Initialisierung des genetischen Algorithmus aus Abschnitt 5.2.2 verwendet. Insgesamt konnte mit rein geometrischen Methoden und niedrigem Erwartungswert ($E = 1.197$) für den relativen Fehler eine komplexe Nebenbedingung eliminiert werden.

5.3 Zusammenfassung und Ausblick

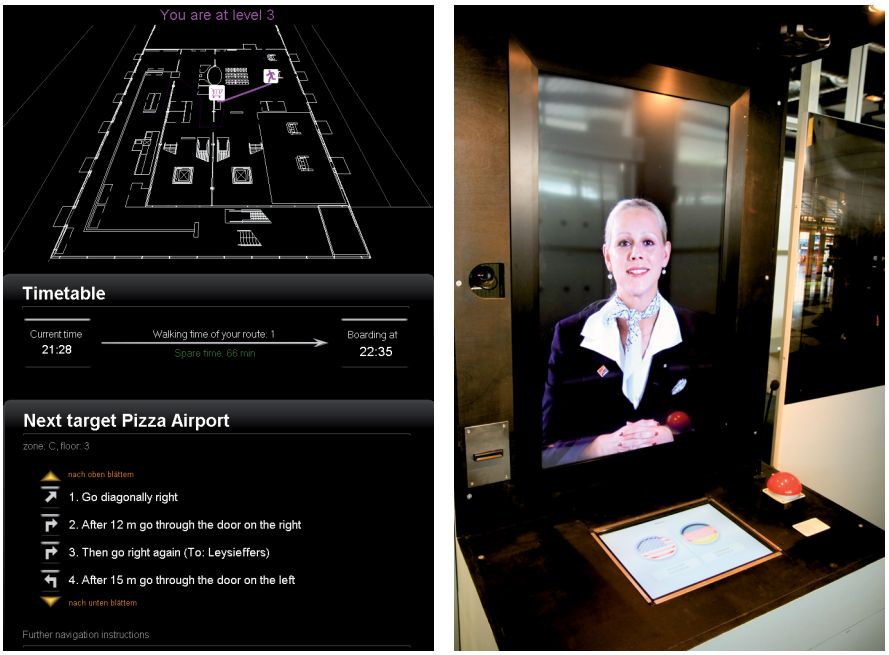
In diesem Kapitel wurde das im Rahmen des Projektes HIPS aufgetretene Problem der geordneten Navigation auf nicht-euklidischen Karten im Detail analysiert und verschiedene Herangehensweisen aufgezeigt. Das Problem der geordneten Navigation selbst ist schon länger bekannt und es gibt auch exakte Lösungsansätze. Diese haben aber eine sehr hohe Komplexität und passen auf das tatsächliche Problem nur in einer kurzsichtigen Betrachtung. Denn diese

Lösungen zerschneiden, wie beschrieben, den Raum der Lösungen sukzessive in einer „Branch-and-Cut“-Methode durch Ebenen und benötigen daher einen Graphen mit konstanten Bewertungen.
Da aber eine kontext-sensitive Navigation ermöglicht werden soll, die mit Voraussagen von Wartezeiten und mit Intervall-Diensten umgehen kann, verändert sich stets die Länge einer Tour durch die vorausgesagte Ankunftszeit an solchen Orten. Somit ist der Graph während der Optimierung des geordneten Travelling-Salesman-Problems nicht konstant bewertet und eine globalere, blinde Betrachtung des Problems muss vorgenommen werden.
Die Methode der genetischen Algorithmen wurde gerade wegen ihrer Blindheit, also der Eigenschaft, dass die Details des Graphen wie die Gewichtung nur dann ausgewertet werden, wenn ein vollständiger Lösungskandidat erstellt wurde, näher betrachtet. Es hat sich gezeigt, dass sie hervorragende Lösungen mit den Betriebsdaten des Flughafens München erstellen kann und dabei nur selten größere Fehler macht. Das entscheidende Problem ist aber die Erzeugung ausreichend vieler ausreichend variabler Touren. In der praktischen Anwendung am Flughafen konnte dieses Problem ähnlich zur Erzeugung von großen Primzahlen, dadurch umgangen werden, dass die Wahrscheinlichkeit der Wohlordnung von zufälligen Touren relativ hoch war. Eine Sortierung der Points-of-Interest nach ihrer Ordnung kam zwar in Frage, wurde aber wieder verworfen, weil durch die unterschiedlichen Rollen von Mitarbeitern und die Vielschichtigkeit der Wegwahl eine totale Ordnung nicht immer vorliegt, und eine Sortierung selbstverständlich eine solche totale Ordnung, also eine Relation zwischen allen Points-of-Interests, voraussetzt.
Unter Betrachtung der noch immer moderaten Fehler, die das Gesamtsystem macht, und der punktuell großen Umwege, wurde eine geometrische Relaxation der Ordnungsrelation zumindest für euklidische, total geordnete Probleme vorgeschlagen und mit empirischen Methoden auf einer Vielzahl zufälliger Graphen analysiert. Dabei konnte durch eine geometrische Modellierung der totalen Ordnung durch Abbildung auf einen Kreis mit hohem Erfolg die Ordnungsrelation relaxiert werden. Diese Idee und Vorgehensweise kann auch für komplexere Ordnungen umgesetzt werden, sofern diese eine geometrische Realisierung zulassen. Weiter ist auch eine Kombination mit adaptierten Verfahren der lokalen Suche wie einer Modifikation von k-Opt für die Nachoptimierung denkbar, um die Ergebnisse noch weiter in der Qualität zu steigern. So ist nach Meinung des Autors davon auszugehen, dass eine solche Relaxation durchaus ausreicht, um eine gute Dienstqualität zur Verfügung zu stellen. Insbesondere dann, wenn es sich um eine feste Umgebung handelt, an welche die Parameter der Verfahren angepasst werden können. Insgesamt ist eine geometrische Herangehensweise an kombinatorische Probleme sicher sinnvoll. Als Beleg mag die Beschreibung des klassischen Travelling-Salesman-Problems als Polytop dienen, die zu den schnellen Schnittebenenverfahren „Branch-and-Bound“ und „Branch-and-Cut“ geführt hat, die ganz offensichtlich geometrischer Natur sind.

6 Prototypen, Projekte und Fallbeispiele

6.1 Indoor-Navigation mit spezieller Interaktions-Infrastruktur

Seit dem Jahr 2008 beschäftigt sich der Flughafen München mit Möglichkeiten, das vorhandene Informationspersonal effektiver einzusetzen. Dabei entstand eine langjährige Kooperation mit dem Lehrstuhl für Mobile und Verteilte Systeme der Ludwig-Maximilians-Universität München, an welchem auch die vorliegende Dissertation entstanden ist. Im Rahmen eines studentischen Praktikums zur Positionierung und Navigation, wurde in Anlehnung an existierende Forschungsarbeiten, insbesondere an das System GAUDI [Kray 05], ein Prototyp eines Indoor-Navigationssystems auf verteilten Anzeigesystemen erstellt. Der Vorteil der Verwendung von verteilten Anzeigesystemen liegt darin, dass die Positionierung der Nutzer ohne irgendwelche zusätzlichen Geräte allein durch Interaktion mit öffentlich zugänglichen Informations-Displays erfolgt.
Während das System GAUDI lediglich eine dynamische Beschilderung vorsieht, in der zeit- und kontextabhängig Navigationsanweisungen auf Displays angezeigt werden, ist durch den Einsatz von Touch-Screens in Kombination mit einer lebensgroßen Videokonferenz ein System entstanden, welches eine gute Symbiose zwischen sozialer Interaktion mit Menschen, Bedienbarkeit und Effizienz eingeht.
Ein Fluggast, der sich orientieren möchte, kann dann an einen von vielen im Flughafengebäude verteilten „Video-Countern“ treten und dort eine digitale „Concierge-Glocke“ läuten. Daraufhin wird eine hochauflösende Videokonferenz in Lebensgröße mit einem Mitarbeiter des Informations-Callcenters initiiert. In dieser Videokonferenz, die noch durch einen schnellen Scanner und einen Drucker unterstützt wird, kann der Fluggast seine Ziele mitteilen, und der Service-Mitarbeiter des Flughafens erstellt einen Navigationsdatensatz für diesen Fluggast. Dieser Navigationsdatensatz wird mit einer Identifiaktion des Fluggastes verknüpft. Dies kann entweder durch ein geeignetes, vorhandenes Hilfsmittel des Fluggastes wie EC-Karte, RFID-Tag, NFC-fähiges Handy, Barcode oder ähnliches geschehen, oder dem Nutzer wird direkt am Video-Counter eine kostengünstige RFID-Karte ausgegeben.
Alle im Flughafen verteilten Informations-Displays sind in der Lage, besagte Identifikationen zu lesen, so kann der Nutzer einfach an ein solches Display her-



(a) Nutzerschnittstelle (b) Video-Counter

Abbildung 6.1: Das prototypische System „HIPS“ [Rupp 09]

antreten und seine Identifikation vorzeigen. Damit ist dem System sowohl der Nutzer als auch der aktuelle Aufenthaltsort bekannt und der Nutzer kann seine Route weiterplanen. Da keine durchgängige Positionierung des Nutzers erfolgt, wird das System von den Nutzern sehr gut angenommen, wie im Rahmen einer Nutzerstudie ermittelt wurde. Dadurch entsteht allerdings auch der Nachteil, dass das System nicht weiß, ob ein Nutzer einen bestimmten Point-Of-Interest auf seiner Route bereits erreicht hat. Entsprechend wird eine Nutzereingabe erforderlich, die bei jeder Interaktion als erstes angezeigt wird: „Haben Sie ihr Zwischenziel Apotheke schon erreicht?“

Abbildung 6.1 zeigt auf der linken Seite die Nutzerschnittstelle des Prototypen, in der zusätzlich zum aktuellen Standort, der Route, einer Karte und einer textuellen Beschreibung des Weges zum nächsten Point-of-Interest auch eine Zeitplanung angezeigt wird, falls dem System ein Flug mitgeteilt wurde. Die Flugdaten werden dabei direkt aus dem Flughafensystem abgerufen, das System reagiert automatisch auf Gate-Änderungen und Verspätungen und passt die Zeitplanung entsprechend an. Auf der rechten Seite derselben Abbildung ist ein Video-Counter aus der Prototyp-Phase zu sehen, mit dem die Videokonferenz abgewickelt wird. Dieser Prototyp wurde ursprünglich in [Rupp 09] beschrieben.

Nachdem dieses System, welches die grundlegenden Funktionen auf einer kleinen Fläche von etwa 200m^2 darstellt, auf ein sehr gutes Echo seitens des Flughafens, der Teilnehmer einer Nutzerstudie und der Presse stieß, entstand der

Wunsch, die noch offenen Probleme für eine Skalierung des Systems auf die Gesamtfläche des Flughafens und für einen dauerhaften Betrieb mit vertretbarer Wartungskomplexität zu lösen.
Die Probleme der Informatik, die in diesem Zusammenhang auftreten, wurden untersucht und Lösungen vorgeschlagen, die sich durch einen dauerhaften Praxiseinsatz im Navigationssystem „Info-Gate“ am Flughafen München bewährt haben. Dabei wurde eine erhöhte Rechenkomplexität für die Navigation in Kauf genommen, um die Langlebigkeit und allgemeine Anwendbarkeit zu unterstützen, und die Verwendung von existierenden CAD-Zeichnungen mittlerer Qualität zur Navigation über das Konzept der semantischen Overlays ermöglicht, siehe Abschnitt 4.3. Ebenso wurde erst durch die Gestaltung und Implementierung von neuartigen Ansätzen für die kontextabhängige, effiziente Auswahl von Routen über mehrere Points-of-Interest ein Produktiveinsatz in großen Umgebungen ermöglicht, siehe Abschnitt 5.1. Um die Flexibilität und Einfachheit des Systems noch weiter zu steigern, wurden Ansätze untersucht und implementiert, welche die Erstellung von semantischen Overlays für CAD-Daten zumindest teilweise automatisieren. Ebenso wurde dem System eine Berechtigungs- und Nutzerverwaltung hinzugefügt, sodass die großen Unterschiede in der Wegwahl zwischen Personal und Fluggästen an einem Flughafen abbildbar wurden.
Insgesamt entstand ein Navigationssystem auf verteilten Anzeigesystemen, welches als erstes System seiner Art die Lücke zwischen kleinflächigen Demonstratoren und einem tatsächlichen Betrieb überzeugend schließen konnte.

6.2 GraphEdit: Editor für semantische Navigations-Overlays

Für die Erstellung und Bearbeitung der semantischen Overlays aus Abschnitt 4.3 wurde eine plattformunabhängige, prototypische CAD-Anwendung umgesetzt. Diese ermöglicht die hybride Verarbeitung der vorliegenden CAD-Daten sowohl mit Algorithmen auf der vektoriellen Darstellung, als auch durch eine integrierte Rendering-Komponente auf einer automatisch erstellten, semantisch angereicherten Repräsentation als Rastergrafik. Dabei wurde auf hohe Geschwindigkeit und Toleranz einer extrem großer Datenbasis geachtet. So konnte durch die geschickte Verwendung eines Quadtree als spatialer Indexstruktur und hardwarebeschleunigtes Rendering von CAD die gesamte CAD-Datenbasis des Münchener Flughafens in einer einheitlichen Umgebung editiert werden.
Das System umfasst neben den Möglichkeiten zur Erstellung und Bearbeitung der in Abschnitt 4.3.2 vorgestellten Overlays auch die Erzeugung eines „Eight-Corner-System“-basierten Zugangsgraphen unter Berücksichtigung aller semantischen Overlays in einem rekursiven Prozess, der durch entsprechende Verlangsamung verfolgt werden kann. Eine Komponente zur versuchsweisen

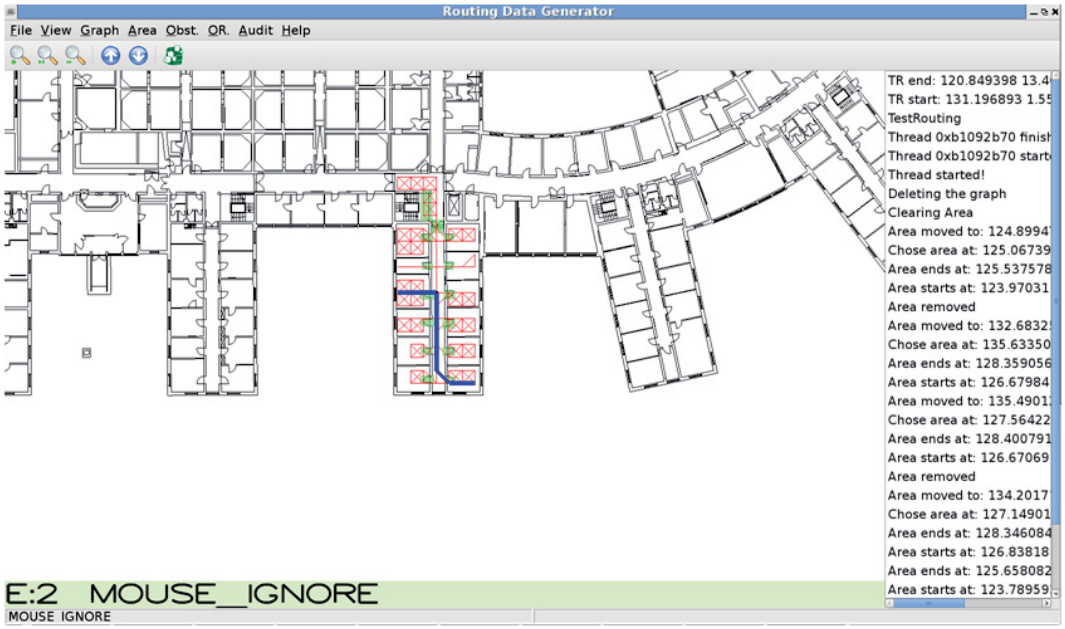
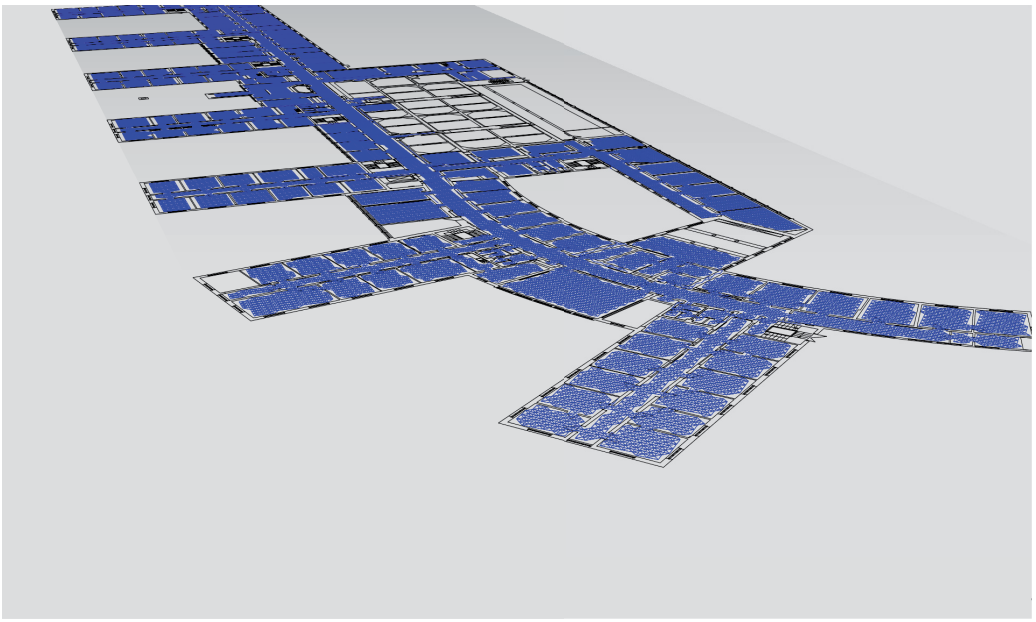


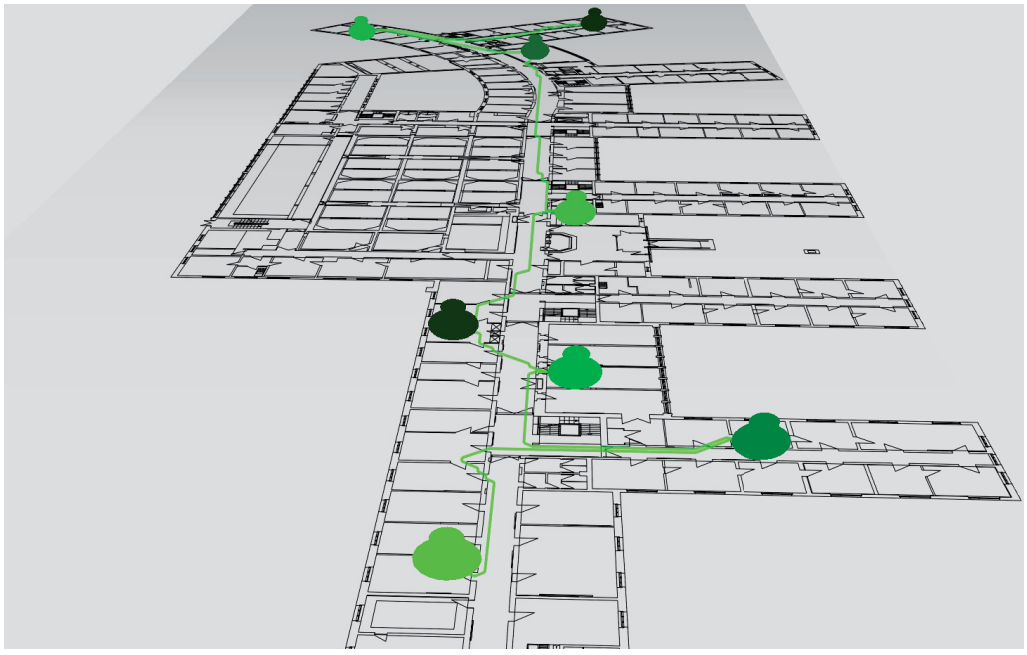
Abbildung 6.2: Die CAD-Umgebung GraphEdit bei der Erzeugung eines Navigationsgraphen

Berechnung kürzester Wege in den vorliegenden Graphen rundet das System in Bezug auf die Erstellung von Navigationsgraphen ab. Durch die modulare Konstruktion ist es auch ohne weiteres möglich, das Modul zur Grapherzeugung durch ein anderes zu ersetzen. Das System der Grapherzeugung kommuniziert mit der CAD-Umgebung über eine Schnittstelle, mit der insbesondere Kollisionsanfragen für Liniensegmente und Koordinatenanfragen für semantische Overlays gestellt werden können.
Diese Software wurde am Flughafen München eingesetzt, um einen vollständigen, semantisch korrekten Zugangsgraphen für den Münchener Flughafen zu erstellen, der darüber hinaus auch durch Anbindung an eine rollenbasierte Rechteverwaltung die Unterschiede zwischen den verschiedenen Berechtigungen am Flughafen korrekt erfassen konnte.
Die Erstellung der semantischen Overlays hat etwa zwei Monate in Anspruch genommen, zum Teil aber auch wegen der noch etwas rudimentären Zeichentools innerhalb der prototypischen Software GraphEdit. Dieser Zugangsgraph modelliert die für die meisten Mitarbeiter und Fluggäste begehbare Fläche von etwa 200.000 m^2 am Flughafen München. Mit dem Einsatz in einer Produktivumgebung dieser Größenordnung ist es gelungen, die große Lücke zwischen der Modellierung verhältnismäßig kleiner und einfacher Umgebungen für wissenschaftliche Zwecke und der Modellierung existierender, produktiver und hochkomplexer Umgebungen mit weniger präzisen, wirtschaftlicheren Umgebungsmodellen, zu schließen.
Mehr als zwei Millionen Zeichnungselemente beschreiben die verschiedenen Etagen der beiden Terminal-Gebäude des Flughafens München und beinhal-

ten dabei besonders viele spezielle Elemente wie gekrümmte Wände, Säulen, besonders große Hallen, besonders kleine Türen, Geschäfte, Büroräume und vieles mehr. Die in den Abschnitten 4.3.3 und 4.2.1.2 vorgestellten Erkennungsalgorithmen wurden in dieser Umgebung implementiert und ermöglichen das automatische Auffinden fast aller Türen. Die Erkennung von Treppen wurde am Flughafen im Produktiveinsatz nicht durchgeführt, weil eine manuelle Überarbeitung von Treppen ohnehin notwendig ist, und der Aufwand durch die klare Erkennbarkeit von Treppen für das menschliche Auge bei Verwendung von Symbolerkennung nicht wesentlich geringer ausfällt. Ein weiterer interessanter Aspekt in diesem Zusammenhang besteht darin, dass die CAD-Abteilung des Münchener Flughafens die vorgestellten Symbolerkennungsalgorithmen sehr nützlich findet, um Qualitätsmängel in den Zeichnungen zu finden. Diese Abteilung hat sich insbesondere für die Türen interessiert, die beispielsweise nicht mit einem Kreissegment in CAD vorliegen, da diese auch in Bezug auf andere Software Probleme verursachen.
Im Umfeld des Navigationssystems „Info-Gate“ entstand auch ein qualitativ hochwertiger, performanter Routing-Server, der auf dem erstellten Navigationsgraphen das geordnete Travelling-Salesman-Problem löst, und danach die kürzesten Routen zwischen den Navigationspunkten berechnet. Dieser Server stellt die Berechnungsfunktionalität und eine Möglichkeit zur schnellen Überprüfung von Sichtverbindungen zwischen zwei Punkten im Navigationsraum über Netzwerk für die verteilten Displays zur Verfügung.
Da die Erstellung der Software für die verteilten Displays keinerlei Bezug zu Forschungsproblemen hat, wurde sie durch eine externe Firma erstellt, sodass für die Entwicklung und visuelle Überprüfung ein Testclient entstanden ist, mit dem die Funktionalität des Systems geprüft werden konnte.
Abbildung 6.3 zeigt zwei Screenshots dieser Software, die in regelmäßigen Abständen zufällige Anfragen mit mehreren Points-Of-Interest an den Routing-Server stellt und das Ergebnis zu Testzwecken visualisiert.



(a) Ausschnitt des Navigationsgraphen eines Universitätsgebäudes



(b) Routenvisualisierung mit geordneten Points-Of-Interest

Abbildung 6.3: Testclient, Visualisierung und Zugangsgraph des Navigationssystems

7 Ausblick

In der vorliegenden Dissertation wurden verschiedene Beiträge zu ortsbezogenen Diensten in Gebäuden erbracht. Denn da durch die besondere Situation in Gebäuden eine Übertragung der Ergebnisse und Mechanismen aus dem Außenbereich nicht möglich ist, besteht ein großer Bedarf an neuen Verfahren, die eine ähnliche Qualität der Navigation und Wegführung in Gebäuden ermöglichen, wie sie im Außenbereich durch Navigationsysteme heutzutage erreicht wird.

So existiert für die Positionierung in Gebäuden beispielsweise keine global verfügbare Infrastruktur, mit der die Position des mobilen Endgerätes in ausreichender Genauigkeit bestimmt werden kann. In den letzten Jahren sind deshalb viele Verfahren auf Basis existierender und speziell für diesen Zweck konzipierter Funktechnologien entstanden, die eine genaue Positionierung in Gebäuden zulassen. Da aber gerade die Funkausbreitung in Gebäuden sehr komplex ist, werden im Bereich der Positionierung in Gebäuden alle möglichen Messgrößen verwendet, die von einem mobilen Endgerät erfasst werden können, und die sich mit dem Ort verändern. Kapitel 2 gibt einen Überblick über die zu Grunde liegenden Algorithmen und konkreten technischen Ausgestaltungen.

Diese Verfahren zur Positionierung in Gebäuden wurden im Kapitel 3 um die Unterstützung von Orientierungs- und Schrittsensorik erweitert. Mit dieser Sensorik, die seit kurzem in Smartphones verfügbar ist, konnten wesentliche Verbesserungen in Bezug auf Rechenaufwand und Positionierungsgenauigkeit erzielt werden. Auch für die Verwendung von digitalen Bildern der Umgebung zur Positionierung reicht die Leistungsfähigkeit moderner Smartphones aus, sodass auch diese Form der Positionierung wieder zunehmend an Relevanz gewinnt. Denn diese ist zunächst vollkommen autark und benötigt – bis auf Zugang zu den Daten – keine Infrastruktur. In diesem Zusammenhang wurde eine heuristische Abstandsschätzung eingeführt, mit der die Qualität der Position in den Situationen verbessert werden konnte, in denen ein Fehler durch Abweichungen zwischen der Aufnahmeposition eines Kalibrierungsbildes und der Aufnahmeposition eines Bildes bei der Positionierung entsteht. Eine Kombination derartiger Verfahren mit anderen Positionierungsverfahren kann an dieser Stelle noch zu weiteren Verbesserungen führen. Auch sollte über die hoch entwickelten Verfahren zur Kombination verschiedener Positionierungsverfahren womöglich ein Gesamtsystem geschaffen werden, welches aus den an einem bestimmten Ort zugänglichen Daten und Messgrößen ein insgesamt optimales Positionierungssystem zusammensetzt. Eine solche Positionierungsplattform könnte dann auch über den Weg der Standardisierung in naher Zukunft

dazu führen, dass ein Anwendungsentwickler mobile, ortsbezogene Dienste in Gebäuden genau so einfach implementieren kann, wie es im Außenbereich über GPS möglich ist.
Eine Integration mit den Arbeiten der Aktivitätserkennung, die derzeit ohnehin in der Regel mit Beschleunigungssensoren vorgenommen wird, kann eine neue Art ortsbezogener Dienste ermöglichen: Nämlich solche Dienste, die neben dem Aufenthaltsort auch die derzeitige Aktivität des Nutzers verwenden, um die passenden Informationen zum passenden Zeitpunkt proaktiv anzubieten.
Grundlegend für höherwertige ortsbezogene Dienste in Gebäuden ist natürlich ein algorithmisches Verständnis des Gebäudes selbst. Nur dann, wenn alle wichtigen Aspekte des Gebäudes von kontextabhängigen Diensten verwendet werden können, wird sich die Vision des „Ubiquitous Computing“, nämlich, dass die Unterstützung durch Informationstechnologie unsichtbar wird, umsetzen lassen. Als mögliche Grundlage für solche Integrationen wurden im Kapitel 4 sehr einfache Umgebungsmodelle konzipiert, die aber dennoch in der Lage sind, die Anforderungen einer Indoor-Navigation zu erfüllen, und die darüber hinaus leicht erweiterbar sind. Wenn sich besonders einfache Umgebungsmodelle durchsetzen, besteht die Hoffnung, dass viele Privatpersonen und Gebäudeeigentümer geeignetes Kartenmaterial zusammenstellen. Für den Außenbereich ist mit der OpenStreetMap ja genau solch ein Vorgehen bereits extrem erfolgreich.
Für das Problem der geordneten Navigation, das bei Planung von Wegen in Gebäuden entsteht, wurden Verfahren vorgeschlagen, die zumindest eine ungefähre Lösung des Problems ermöglichen. Diese Klasse von Problemen ist dabei nicht nur für die Personennavigation interessant, sondern auch für Planung von Prozessen und Ressourcen in Gebäuden. In den nächsten Jahren wird nach Meinung des Autors durch die Integration von Kontext- und Ortsinformationen mit Nutzerprofilen eine ganz neue Qualität kontextbezogener Dienste entstehen, und die Navigation im Innen- und Außenraum immer mehr zusammenwachsen.

Literaturverzeichnis

[Adam 98] M. Adams, A. Kerstens. "Tracking naturally occurring indoor features in 2-d and 3-d with lidar range/amplitude data". *The International Journal of Robotics Research*, Vol. 17, Nr. 9, pp. 907–923, 1998.

[Appl 06] D. Applegate, R. Bixby, V. Chvatal, W. Cook. "Concorde TSP Solver". `http://www.tsp.gatech.edu/concorde`, 2006.

[Aror 98] S. Arora. "Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems". *Journal of the ACM*, Vol. 45, pp. 753–782, 1998.

[Arul 02] M. Arulampalam, S. Maskell, N. Gordon, T. Clapp. "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking". *IEEE Transactions on Signal Processing*, Vol. 50, Nr. 2, pp. 174–188, feb. 2002.

[Asch 93] N. Ascheuer, L. F. Escudero, M. Grötschel, M. Stoer. "A cutting plane approach to the sequential ordering problem (with applications to job scheduling in manufacturing)". *SIAM Journal on Optimization*, Vol. 3, p. 25, 1993.

[Bahl 00] P. Bahl, V. N. Padmanabhan. "RADAR: an in-building RF-based user location and tracking system". In: *Proceedings of the Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pp. 775–784, 2000.

[Banh 12] M. Banholzer. "Evaluierung möglicher Algorithmen zur Indoor-Positionierung mittels Kinect". *Diplomarbeit*, 2012.

[Bay 06] H. Bay, T. Tuytelaars, L. Van Gool. "SURF: Speeded Up Robust Features". In: *Proceedings of the European Conference on Computer Vision*, pp. 404–417, 2006.

[Beck 05] C. Becker, F. Dürr. "On location models for ubiquitous computing". *Personal and Ubiquitous Computing*, Vol. 9, Nr. 1, pp. 20–31, 2005.

[Besl 92] P. Besl, N. McKay. "A method for registration of 3-D shapes". *IEEE Transactions on pattern analysis and machine intelligence*, Vol. 14, Nr. 2, pp. 239–256, 1992.

[Bogg 04] J. Boggs. "Geolocation of an Audio Source in a Multipath Environment Using Time-of-Arrival". Tech. Rep., DTIC Document, 2004.

[Boni 08] F. Bonin-Font, A. Ortiz, G. Oliver. "Visual navigation for mobile robots: a survey". *Journal of Intelligent & Robotic Systems*, Vol. 53, Nr. 3, pp. 263–296, 2008.

[Camp 04] J. Campbell, R. Sukthankar, I. Nourbakhsh. "Techniques for evaluating optical flow for visual odometry in extreme terrain". In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3704–3711, 2004.

[Chen 02] Y. Chen, H. Kobayashi. "Signal strength based indoor geolocation". In: *Proceedings of the IEEE International Conference on Communications*, pp. 436–439, 2002.

[Chev 00] K. Cheverst, N. Davies, K. Mitchell, A. Friday. "Experiences of developing and deploying a context-aware tourist guide: the GUIDE project". In: *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, pp. 20–31, ACM, 2000.

[Chri 76] N. Chrisofides. "Worst-case analysis of a new heuristic for the traveling salesman problem". *Technical Report 388, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh*, 1976.

[City 12] "CityGML". Online, 2012. http://www.opengeospatial.org/standards/citygml, letzer Abruf: 14.3.2012.

[Cobb 97] H. S. Cobb. "GPS pseudolites: theory, design, and applications". *Ph. D. Thesis, Stanford University*, 1997.

[Cohe 95] W. W. Cohen. "Fast Effective Rule Induction". In: *Twelfth International Conference on Machine Learning*, pp. 115–123, 1995.

[COMB 12] "COMBINE Project". Online, 2012. http://www.combine-project.eu/, letzter Abruf: 2.4.2012.

[Davi 10] P. Davidson, J. Collin, J. Takala. "Application of particle filters for indoor positioning using floor plans". In: *Ubiquitous Positioning, Indoor Navigation, and Location Based Service*, IEEE, 2010.

[Davi 85] L. Davis. "Job shop scheduling with genetic algorithms". In: *Proceedings of an International Conference on Genetic Algorithms and Their Applications*, 1985.

[Depa 97] Department of Defense. "World Geodetic System 1984, Its Definition and Relationships With Local Geodetic Systems". Tech. Rep., NIMA Technical Report TR8350.2, 1997.

[Dey 00] A. Dey, G. Abowd. "Towards a Better Understanding of Context and Context-Awareness". In: *Proceedings of the Workshop on the What, Who, Where, When, Why and How of Context-Awareness*, 2000.

[Digg 02] F. van Diggelen. "Indoor GPS Theory & Implementation". In: *Position, Location and Navigation Symposium*, pp. 240 – 247, 2002.

[Dijk 59] E. W. Dijkstra. "A note on two problems in connexion with graphs". *Numerische Mathematik*, Vol. 1, Nr. 1, pp. 269–271, 1959.

[Dill 10] M. Dille, B. Grocholsky, S. Singh. "Outdoor downward-facing optical flow odometry with commodity sensors". In: *Field and Service Robotics*, pp. 183–193, Springer, 2010.

[Eato 02] J. W. Eaton. *GNU Octave Manual*. Network Theory Limited, 2002.

[Ekah 12] "Ekahau". Online, 2012. http://www.ekahau.com/, letzer Abruf: 2.4.2012.

[Escu 88] L. F. Escudero. "An inexact algorithm for the sequential ordering problem". *European Journal of Operational Research*, Vol. 37, Nr. 2, pp. 236–249, 1988.

[Escu 94] L. F. Escudero, M. Guignard, K. Malik. "A Lagrangian relax-and-cut approach for the sequential ordering problem with precedence relationships". *Annals of Operations Research*, Vol. 50, Nr. 1, pp. 219–237, 1994.

[Even 06] F. Evennou, F. Marx. "Advanced integration of WIFI and inertial navigation systems for indoor mobile positioning". *EURASIP Journal of Applied Signal Processing*, 2006.

[Fisc 81] M. Fischler, R. Bolles. "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". *Communications of the ACM*, Vol. 24, Nr. 6, pp. 381–395, 1981.

[Floy 62] R. W. Floyd. "Algorithm 97: Shortest Path". *Communications of the ACM*, Vol. 5, Nr. 6, p. 345, 1962.

[Flug 11a] "Flughafen München: Eröffnung InfoGate". Online, 2011. http://www.reisenews-online.de/2011/06/07/neue-infogates-am-flughafen-muenchen/, letzer Abruf: 8.2.2012.

[Flug 11b] "Flughafen München: InfoGate". Online, 2011. http://www.munich-airport.de/infogate, letzer Abruf: 8.2.2012.

[Foy 76] W. Foy. "Position-location solutions by Taylor-series estimation". *IEEE Transactions on Aerospace and Electronic Systems*, pp. 187–194, 1976.

[Freu 07] R. W. Freund, R. H. W. Hoppe. *Stoer/Bulirsch: Numerische Mathematik 1.* Zehnte, neu bearbeitete Auflage, Springer-Verlag, Berlin Heidelberg, 2007.

[Geog 12] "Geographic Markup Language". Online, 2012. http://www.opengeospatial.org/standards/gml, letzer Abruf: 14.3.2012.

[Geor 95] H. George, P. Langley. "Estimating Continuous Distributions in Bayesian Classifiers". In: *Eleventh Conference on Uncertainty in Artificial Intelligence*, pp. 338–345, 1995.

[Gold 85] D. Goldberg, R. Lingle. "Alleles, loci, and the traveling salesman problem". In: *Proceedings of an International Conference on Genetic Algorithms and Their Applications*, pp. 154–159, 1985.

[Gold 89] D. E. Goldberg. *Genetic Algorithms.* Addision Wesley Longman, Inc., 1989.

[Goog 12] "Google Maps". Online, 2012. http://maps.google.com, letzer Abruf: 2.4.2012.

[Gord 93] N. Gordon, D. Salmond, A. Smith. "Novel approach to nonlinear/non-Gaussian Bayesian state estimation". In: *Proceedings of the Institute of Electrical Engineering*, pp. 107–113, 1993.

[Goya 11] P. Goyal, V. Ribeiro, H. Saran, A. Kumar. "Strap-Down Pedestrian Dead-Reckoning System". In: *Proceedings of the International Conference on Indoor Positioning and Indoor Navigation*, IEEE, 2011.

[Gref 85] J. Grefenstette, R. Gopal, B. Rosmaita, D. Gucht. "Genetic algorithms for the traveling salesman problem". In: *Proceedings of an International Conference on Genetic Algorithms and Their Applications*, pp. 160–168, 1985.

[Gris 09] G. Grisetti, C. Stachniss, W. Burgard. "Non-linear Constraint Network Optimization for Efficient Map Learning". *IEEE Transactions on Intelligent Transportation Systems*, Vol. 10, pp. 428–439, 2009.

[Gust 02] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, P.-J. Nordlund. "Particle filters for positioning, navigation and tracking". In: *IEEE Transactions on Signal Processing*, pp. 425–437, 2002.

[Hall 09] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten. "The WEKA Data Mining Software: An Update". *SIGKDD Explorations*, Vol. 11, 2009.

[Hash 97] M. Hashima, F. Hasegawa, S. Kanda, T. Maruyama, T. Uchiyama. "Localization and obstacle detection for robots for carrying food trays". In: *Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 345 – 351, 1997.

[Haus 11] M. Haustein, M. Werner. "Lokalisierung durch die Messung von Signallaufzeiten mitttels handelsüblicher Wireless LAN Adapter". *Tagungsband des 8. GI/KuVS-Fachgespräch „Ortsbezogene Anwendungen und Dienste"*, Vol. 8, pp. 101–113, 2011.

[Heid 11] W. Heidrich, M. Schulze, M. Kessel, M. Werner. "Robustes Mapmatching hoch aufgelöster, fahrzeugbasierter GPS-Tracks". *Tagungsband des 8. GI/KuVS-Fachgespräch „Ortsbezogene Anwendungen und Dienste"*, Vol. 8, pp. 23–36, 2011.

[High 01] J. Hightower, G. Borriello. "Location systems for ubiquitous computing". *Computer*, Vol. 34, Nr. 8, pp. 57–66, 2001.

[Hile 07] H. Hile, G. Borriello. "Information Overlay for Camera Phones in Indoor Environments". In: *Third International Symposium on Location- and Context-Awareness*, pp. 68–84, 2007.

[Hoff 03] B. Hoffmann-Wellenhof, K. Legat, M. Wieser. *Navigation – Principles of Positioning and Guidance*. Springer-Verlag, Wien, 2003.

[Hohl 99] F. Hohl, U. Kubach, A. Leonhardi, K. Rothermel, M. Schwehm. "Next century challenges: Nexus—an open global infrastructure for spatial-aware applications". In: *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, pp. 249–255, ACM, 1999.

[Horn 81] B. Horn, B. Schunck. "Determining optical flow". *Artificial Intelligence*, Vol. 17, Nr. 1-3, pp. 185–203, 1981.

[Huan 10] H. Huang, G. Gartner. "A Survey of Mobile Indoor Navigation Systems". *Cartography in Central and Eastern Europe*, pp. 305–319, 2010.

[Kabu 87] M. Kabuka, A. Arenas. "Position verification of a mobile robot using standard pattern". *IEEE Journal of Robotics and Automation*, Vol. 3, Nr. 6, pp. 505–516, 1987.

[Kalm 60] R. Kalman *et al.* "A new approach to linear filtering and prediction problems". *Journal of Basic Engineering*, Vol. 82, Nr. 1, pp. 35–45, 1960.

[Karp 72] R. M. Karp. "Reducibility among combinatorial problems". *Complexity of Computer Computations*, pp. 85–103, 1972.

[Karp 77] R. M. Karp. "Probabilistic analysis of partitioning algorithms for the TSP in the plane". *Mathematics of Operations Research*, Vol. 2, pp. 209–224, 1977.

[Kawa 10] H. Kawaji, K. Hatada, T. Yamasaki, K. Aizawa. "Image-Based Indoor Positioning System: Fast Image Matching Using Omnidirectional Panoramic Images". In: *1st ACM International Workshop on Multimodal Pervasive Video Analysis*, 2010.

[Kee 01] C. Kee, D. Yun, H. Jun, B. Parkinson, S. Pullen, T. Lagenstein. "Centimeter-Accuracy Indoor Navigation Using GPS-Like Pseudolites". In: *GPSWorld*, 2001.

[Kess 11a] M. Kessel, P. Ruppel, F. Gschwandtner. "BIGML: A location model with individual waypoint graphs for indoor location-based services". *PIK-Praxis der Informationsverarbeitung und Kommunikation*, Vol. 33, Nr. 4, pp. 261–267, 2011.

[Kess 11b] M. Kessel, M. Werner. "SMARTPOS: Accurate and Precise Indoor Positioning on Mobile Phones". In: *The First International Conference on Mobile Services, Resources, and Users (MOBILITY), 2011*, pp. 158–163, 2011.

[King 06] T. King, S. Kopf, T. Haenselmann, C. Lubberger, W. Effelsberg. "COMPASS: A Probabilistic Indoor Positioning System Based on 802.11 and Digital Compasses". In: *1st International Workshop on Wireless Network Testbeds, Experimental Evaluation and Characterization*, pp. 34–40, 2006.

[Kolb 05] T. Kolbe, G. Gröger, L. Plümer. "CityGML–Interoperable access to 3D city models". In: *First International Symposium on Geo-Information for Disaster Management*, Springer Verlag, 2005.

[Kort 08] B. Korte, J. Vygen. *Combinatorial Optimization: Theory and Algorithms*. Springer-Verlag Berlin Heidelberg, 2008.

[Krak 75] E. Krakiwsky. “A synthesis of recent advances in the method of least squares”. *Geodesy & Geomatics Lecture Notes; 42*, 1975.

[Kray 05] C. Kray, G. Kortuem, A. Krüger. “Adaptive Navigation Support with Public Displays”. In: *Proceedings of the 10th International Conference on Intelligent User Interfaces*, pp. 326–328, ACM, 2005.

[Kumm 11] F. Kummler. “Fahrerzustandserkennung durch lernende Systeme”. *Diplomarbeit*, 2011.

[Küp 05] A. Küpper. *Location-based Services: Fundamentals and Operation*. John Wiley & Sons, 2005.

[Lee 05] A. Lee, G. Aouad, R. Cooper, C. Fu, A. Marshall-Ponting, J. Tah, S. Wu. “nD-modelling: a driver or enabler for construction improvement?”. *RICS Research Paper Series, London*, pp. 1–16, 2005.

[Link 11] J. A. B. Link, P. Smith, N. Viol, K. Wehrle. “FootPath: Accurate Map-based Indoor Navigation Using Smartphones”. In: *Proceedings of the International Conference on Indoor Positioning and Indoor Navigation*, 2011.

[List 12] “Liste der größten deutschen Städte”. Online, 2012. http://www.wikipedia.de, letzter Abruf: 2.4.2012.

[Loca 12] “Locata”. Online, 2012. http://www.locatacorp.com/, letzer Abruf: 16.5.2012.

[Lowe 99] D. G. Lowe. “Object Recognition from Local Scale-Invariant Features”. In: *International Conference on Computer Vision*, pp. 1150–1157, 1999.

[Luca 81] B. Lucas, T. Kanade. “An iterative image registration technique with an application to stereo vision”. In: *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, 1981.

[Ludw 71] R. Ludwig. *Methoden der Fehler- und Ausgleichsrechnung*. Deutscher Verlag der Wissenschaften, 1971.

[Math 12] “MathWorks: MATLAB Produktseite”. Online, 2012. http://www.mathworks.de/products/matlab/, letzer Abruf: 15.2.2012.

[Maut 11] R. Mautz, S. Tilch. “Optical Indoor Positioning Systems”. In: *Proceedings of the International Conference on Indoor Positioning and Indoor Navigation*, 2011.

[Micr 12] "Microsoft Kinect". Online, 2012. http://www.xbox.com/de-DE/Kinect, letzer Abruf: 14.3.2012.

[Motl 88] A. Motley, J. Keenan. "Personal communication radio coverage in buildings at 900 MHz and 1700 MHz". *Electronics Letters*, Vol. 24, Nr. 12, pp. 763–764, 1988.

[Naga 00] K. Nagatani, S. Tachibana, M. Sofne, Y. Tanaka. "Improvement of odometry for omnidirectional vehicle using optical flow information". In: *Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 468–473, 2000.

[Nick 01] D. Nicklas, M. Großmann, T. Schwarz, S. Volz, B. Mitschang. "A model-based, open architecture for mobile, spatially aware applications". *Advances in spatial and temporal databases*, pp. 117–135, 2001.

[Nisb 09] R. Nisbet, J. Elder, J. Elder, G. Miner. *Handbook of statistical analysis and data mining applications*. Academic Press, 2009.

[Open 12a] "OpenStreetMap". Online, 2012. http://www.openstreetmap.org, letzer Abruf: 2.4.2012.

[Open 12b] "OpenStreetMap Foundation: IndoorOSM". Online, 2012. http://wiki.openstreetmap.org/wiki/IndoorOSM, letzer Abruf: 14.3.2012.

[Open 12c] "OpenStreetMap Foundation: OSM Map Features". Online, 2012. http://wiki.openstreetmap.org/wiki/Map_Features, letzer Abruf: 16.2.2012.

[Orr 00] R. Orr, G. Abowd. "The smart floor: a mechanism for natural user identification and tracking". In: *CHI: Extended Abstracts on Human Factors in Computing Systems*, pp. 275–276, ACM, 2000.

[Pál 93] K. Pál. "Genetic algorithms for the traveling salesman problem based on a heuristic crossover operation". *Biological Cybernetics*, Vol. 69, pp. 539–546, 1993.

[Quin 93] R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, 1993.

[Rapp 02] T. S. Rappaport. *Wireless communications: principles and practices*. Prentice-Hall, 2002.

[Rein 91] G. Reinelt. "TSPLIB – A traveling salesman problem library". *ORSA Journal of Computation*, Vol. 3, pp. 376–384, 1991.

[Ribb 12] "RibbonSoft: Was ist dxflib?". Online, 2012. http://www.ribbonsoft.com/de/what-is-dxflib, letzer Abruf: 15.2.2012.

[Rist 04] B. Ristic, S. Arulampalam, N. Gordon. *Beyond the Kalman filter: Particle filters for tracking applications*. Artech House Publishers, 2004.

[Rizo 10] C. Rizos, G. Roberts, J. Barnes, G. N. "Locata: A new high accuracy indoor positioning system". In: *Proceedings of the International Conference on Indoor Positioning and Indoor Navigation*, 2010.

[Roys 89] J. Royston. "Correcting the Shapiro-Wilk W for Ties". *Journal of Statistical Computation and Simulation*, Vol. 31, Nr. 4, pp. 237–249, 1989.

[Rupp 09] P. Ruppel, F. Gschwandtner, C. K. Schindhelm, C. Linnhoff-Popien. "Indoor Navigation on Distributed Stationary Display Systems". *International Computer Software and Applications Conference*, pp. 37–44, 2009.

[Russ 10] S. Russell, P. Norvig. *Artificial intelligence: a modern approach. Prentice Hall Series in Artificial Intelligence*, Prentice Hall, 2010.

[Sahn 76] S. Sahni, T. Gonzalez. "P-complete approximation problems". *Journal of the ACM*, Vol. 23, pp. 555–565, 1976.

[Schw 04] H.-R. Schwarz, N. Köckler. *Numerische Mathematik*. Fünfte Auflage, B.G. Teubner-Verlag, Wiesbaden, 2004.

[Se 02] S. Se, D. Lowe, J. Little. "Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks". *The International Journal of Robotics Research*, Vol. 21, Nr. 8, pp. 735–758, 2002.

[Shap 65] S. Shapiro, M. Wilk. "An analysis of variance test for normality (complete samples)". *Biometrika*, Vol. 52, Nr. 3/4, pp. 591–611, 1965.

[Stal 02] W. Stallings. *Wireless Communication and Networks*. Prentice Hall, 2002.

[Steg 05] P. Steggles, S. Gschwind. "The Ubisense smart space platform". In: *Adjunct Proceedings of the Third International Conference on Pervasive Computing*, pp. 73–76, 2005.

[Stor 10] W. Storms, J. Shockley, J. Raquet. "Magnetic field navigation in an indoor environment". In: *Ubiquitous Positioning, Indoor Navigation, and Location Based Service*, IEEE, 2010.

[Tarz 11] S. P. Tarzia, P. A. Dinda, R. P. Dick, G. Memik. "Indoor localization without infrastructure using the acoustic background spectrum". In: *9th International Conference on Mobile Systems, Applications, and Services*, pp. 155–168, 2011.

[Tell 12] J. Teller, C. Roussey, G. Falquet, C. Tweed, Eds. *COST Action C21: Towntology - Final Report (Draft)*. Towntology Project, 2012.

[The 07] The Comission of the European Community. "Comission Decision of 21 February 2007 on allowing the use of radio spectrum for equipment using ultra-wideband technology in a harmonised manner in the Community". *Official Journal of the European Union*, Vol. 2007/131/EC, 2007.

[The 12a] "The Ambient Assisted Living Open Association". Online, 2012. http://www.aaloa.org/, letzer Abruf: 14.3.2012.

[The 12b] "The GUIDE Project". Online, 2012. http://www.guide.lancs.ac.uk/overview.html, letzter Abruf: 2.4.2012.

[Town 12] "Towntology Project". Online, 2012. http://www.towntology.net/, letzter Abruf: 2.4.2012.

[Tran 08] N. Tran-Minh, T. Do-Hong. "Application of raytracing technique for predicting average power distribution in indoor environment". In: *Second International Conference on Communications and Electronics*, pp. 121–125, 2008.

[Trav 05] W. Travis, A. Simmons, D. Bevly. "Corridor navigation with a LiDAR/INS Kalman filter solution". In: *Intelligent Vehicles Symposium*, pp. 343–348, IEEE, 2005.

[Ubis 12] "Ubisense RTLS". Online, 2012. http://www.ubisense.net/, letzter Abruf: 2.4.2012.

[Unif 12] "Unified Modelling Language". Online, 2012. http://www.uml.org/, letzer Abruf: 14.3.2012.

[Wald 09] I. Wald, W. Mark, J. Günther, S. Boulos, T. Ize, W. Hunt, S. Parker, P. Shirley. "State of the art in ray tracing animated scenes". In: *Computer Graphics Forum*, pp. 1691–1722, Wiley Online Library, 2009.

[Want 92] R. Want, A. Hopper, V. Falc, J. Gibbons. "The active badge location system". *ACM Transactions on Information Systems*, Vol. 10, Nr. 1, pp. 91–102, 1992.

[Ward 97] A. Ward, A. Jones, A. Hopper. "A new location technique for the active office". *Personal Communications, IEEE*, Vol. 4, Nr. 5, pp. 42–47, 1997.

[Weis 09] D. Weiß. "Kontextabhängige Personalisierung multimedialer Inhalte auf mobilen Endgeräten". *Dissertation*, 2009.

[Weis 91] M. Weiser. "The computer for the 21st century". *Scientific American*, Vol. 265, Nr. 3, pp. 94–104, 1991.

[Wern 10] M. Werner, M. Kessel. "Organisation of Indoor Navigation data from a data query perspective". In: *Ubiquitous Positioning, Indoor Navigation and Location Based Service*, pp. 1–6, 2010.

[Wern 11a] M. Werner. "Efficiently Using Bitmap Floorplans for Indoor Navigation on Mobile Phones". In: *The Seventh International Conference on Wireless and Mobile Communications (ICWMC)*, pp. 225–230, 2011.

[Wern 11b] M. Werner. "IndoorSim: Simulation of Wireless-LAN-Based Indoor Positioning Systems Incorporating CAD-Floorplans". *Tagungsband des 8. GI/KuVS-Fachgespräch Ortsbezogene Anwendungen und Dienste*, 2011.

[Wern 11c] M. Werner. "Selection and Ordering of Points-of-Interest in Large-Scale Indoor Navigation Systems". In: *IEEE 35th Annual Computer Software and Applications Conference (COMPSAC), 2011*, pp. 504 –509, 2011.

[Wern 11d] M. Werner, M. Kessel, C. Marouane. "Indoor positioning using smartphone camera". In: *International Conference on Indoor Positioning and Indoor Navigation*, 2011.

[Wern 12] M. Werner, M. Kessel, F. Gschwandtner, M. Dürr, K. Wiesner, T. Mair. "Technologische Herausforderungen für kontextsensitive Geschäftsanwendungen". 2012.

[Witt 11] I. Witten, E. Frank, M. Hall. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2011.

[Wolf 05] G. Wölfle, R. Wahl, P. Wertz, P. Wildbolz, F. Landstorfer. "Dominant path prediction model for indoor scenarios". In: *German Microwave Conference (GeMIC)*, 2005.

[Wood 08] O. Woodman, R. Harle. “Pedestrian localisation for indoor environments”. In: *Proceedings of the 10th International Conference on Ubiquitous Computing*, pp. 114–123, ACM, 2008.

[Xiao 11] W. Xiao, W. Ni, Y. Toh. “Integrated Wi-Fi fingerprinting and inertial sensing for indoor positioning”. In: *International Conference on Indoor Positioning and Indoor Navigation*, IEEE, 2011.

[Zehn 05] M. L. Zehner, K. Bannicke, R. Bill. “Positionierungsansätze mittels WLAN-Ausbreitungsmodellen”. 2005.

[Zinn 09] A. Zinnen, C. Wojek, B. Schiele. “Multi Activity Recognition Based on Bodymodel-Derived Primitives”. In: *Proceedings of the 4th International Symposium on Location and Context Awareness*, pp. 1–18, 2009.

Quellcode-Beispiele

Octave-Skript zur Berechnung des Laterationsbeispiels

Das folgende Skript führt die notwendigen Berechnungen aus, wie im Beispiel zur Lateration in Beispiel 2.1.1 beschrieben:

```
% Basisstationen
B=[0,0;10,0;15,10;0,12]
%Korrekter Ort
Pk=[2,2]
%Initiale Schätzung
Ps=[20,20];
% Distanzen zum Beispiel:
d=[2.92,8.14,15.46,9.89]

function retval=Step(Ps,B,d)
    alpha=[];
    for i=1:size(B)(1)
      alpha(i) = sqrt((B(i,1)-Ps(1))*(B(i,1)-Ps(1))
                      +(B(i,2)-Ps(2))*(B(i,2)-Ps(2)));
    endfor
    % Setze Matrix A zusammen
    A=[];b=[];
    for i=1:size(B)(1)
       A=[A;
       (-B(i,1)+Ps(1))/alpha(i),(-B(i,2)+Ps(2))/alpha(i);
         ];
       b=[b;d(i)-alpha(i)];
    endfor
    retval=transpose(A\b)  % Löse mit Ausgleichsrechnung
endfunction

S=[Ps];R=[];K=[];
for i=2:10
  K=[K;Step(S(i-1,:),B,d)];    # Korrekturvektor
  S(i,:)=S(i-1,:)+K(i-1,:);    # Schätzung
  R= [R; norm(S (i,:) - Pk)]; # Residuum
endfor
```