Telematics
Computer Systems

Freie Universität Berlin

Computer Systems and Telematics

Distanzbasierte Indoorlokalisierung in mobilen Netzwerken

Dissertation zur Erlangung des akademischen Grades eines
Doktors der Naturwissenschaften (Dr. rer. nat.) im Fachbereich
Mathematik und Informatik der Freien Universität Berlin

vorgelegt von

Thomas Hillebrandt und **Heiko Will**

Datum der Disputation: 17. April 2014

Gutachter:
Prof. Dr. Marcel Kyas, Freie Universität Berlin
Prof. Dr. Kay Uwe Römer, Technische Universität Graz

Institut für Informatik, Freie Universität Berlin, Deutschland

2. Juli 2014

Ich versichere, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten Schriften entnommen wurden, sind als solche gekennzeichnet. Die Zeichnungen oder Abbildungen sind von mir selbst erstellt worden oder mit entsprechenden Quellennachweisen versehen. Diese Arbeit ist in gleicher oder ähnlicher Form noch bei keiner Prüfungsbehörde eingereicht worden.

Berlin, den 2. Juli 2014

(Thomas Hillebrandt)

(Heiko Will)

Zusammenfassung

Während die ausschließlich im Freien nutzbaren globalen Navigationssatellitensysteme (GNSS) wie z. B. das *Global Positioning System* (GPS) seit Jahren im Einsatz sind und ihren Durchbruch in den Massenmarkt spätestens durch die weltweite Nachfrage nach Smartphones und Autonavigationssystemen erreicht haben, steht dieser Meilenstein für die Indoorlokalisierung noch aus. Dies liegt zum einen an der fehlenden Anwendungsperspektive für einen Massenmarkt, aber im Wesentlichen an ungelösten Problemen aus dem Bereich der Informatik. Während sich erste Anwendungsszenarien im Bereich standortbezogener Werbung und der zivilen Sicherheit abzeichnen, zum Beispiel die Ortung von Rettungskräften, bleiben technische Fragen weiter ungelöst. Diese Probleme lassen sich in der Regel auf die zu erzielende Genauigkeit der Lokalisierung oder besondere Anforderung des Szenarios wie zum Beispiel Infrastrukturlosigkeit, ad hoc Vernetzung oder nicht zu erfüllende Hardwareanforderungen der existierenden Lokalisierungssysteme reduzieren.

In unserer Arbeit nehmen wir uns dem Problem der Verbesserung der Genauigkeit distanzbasierter Lokalisierungsalgorithmen an. Von besonderer Bedeutung ist die damit verbundene Robustheit eines Algorithmus gegenüber Ausreißern in den Distanzmessungen, die innerhalb von Gebäuden häufig auftreten. Wir zeigen drei aufeinander aufbauende Lösungsansätze auf, die zu einer wesentlichen Verbesserung in diesem Bereich beitragen:

1. In der Literatur existiert eine Vielzahl von Algorithmen die entweder für die Indoorlokalisierung entwickelt wurden oder zumindest in diesem Feld eingesetzt

werden können. Es existieren jedoch kaum breit akzeptierte Metriken, um die Qualität oder Eignung der unterschiedlichen Algorithmen zu bewerten oder zu vergleichen. Die meistens aus Simulationen gewonnen Erkenntnisse lassen sich nicht untereinander vergleichen und die Parameter sind oft nicht auf die Realitäten der Indoorlokalisierung abgestimmt. Wir präsentieren in unserer Arbeit eine Diskussion der verschiedenen Parameter und Modelle und leiten daraus ein Simulationstool ab, dass eine Simulation dieser Algorithmen unter vergleichbaren Parametern und Metriken erlaubt.

2. Experimente sind im gesamten Forschungsbereich der Indoorlokalisierung eher die Ausnahme als die Regel. Dies liegt zum einen in dem sehr hohen Aufwand und der benötigten Spezialhardware als auch in einer Spaltung der Community in Anwender und Algorithmiker. Die Algorithmiker setzen aufgrund der probabilistischen Eingaben fast vollständig auf Simulationen deren Übertragbarkeit auf die Praxis oft nur schwer möglich ist und die Anwender führen zwar Experimente durch, aber benchmarken damit nur das verwendete System gegenüber einem vorher definierten Ziel, ohne dass eine Vergleichbarkeit mit anderen Systemen besteht. Wir stellen in unserer Arbeit ein System vor, mit dem sich Experimente wesentlich vereinfachen lassen und auch eine Reproduzierbarkeit der einzelnen Experimente gewährleistet ist.

3. Aus den Erkenntnissen der ersten beiden Punkte leiten wir Anforderungen für präzisere und praxisrelevante Algorithmen ab und stellen zwei Algorithmen vor, deren Performance über den in der Literatur bekannten Algorithmen liegt. Wir präsentieren eine ausführliche Auswertung dieser Algorithmen und vergleichen diese nach den in 1) festgelegten Kriterien und mit den in 1) und 2) entwickelten Werkzeugen.

Die im Rahmen der Arbeit entwickelten Werkzeuge und Algorithmen wurden durch uns auf internationalen Konferenzen und in Journalen vorgestellt und diskutiert.

Über diese Arbeit

Diese vorliegende Arbeit ist in der Form einer Gruppendissertation der beiden Autoren abgefasst. Das bedeutet, dass die gesamte der Arbeit zugrundeliegende Forschungstätigkeit und alle zugrundeliegenden Publikationen von den Autoren gemeinsam und gleichwertig erstellt worden sind. Die Reihenfolge der Autorennennung bei gemeinsamen Veröffentlichungen richtet sich ausschließlich nach praktischen Erwägungen und es lässt sich keine Haupt- oder Nebenautorenschaft daraus ableiten. So wird bei Konferenzbeiträgen immer der Autor als erstes genannt, der den Beitrag auf der Konferenz vorgestellt hat. Dieses wiederum hat sich anhand eher profaner Parameter wie zeitliche Verfügbarkeit und den Möglichkeiten der Reisekostenerstattung ergeben. Journalbeiträge sind alphabetisch nach Nachnamen für die beiden Hauptautoren sortiert.

Dieser Arbeit liegen Forschungsbeiträge aus den Jahren 2009 bis 2013 zugrunde, die allesamt einem Peer-Review-Verfahren unterzogen worden und auf der entsprechenden Konferenz vorgestellt bzw. im entsprechenden Druckwerk veröffentlicht worden sind. Im Wesentlichen sind folgende Publikationen die wissenschaftliche Basis dieser Arbeit:

- Thomas Hillebrandt, Heiko Will und Marcel Kyas. The Membership Degree Min-Max Localisation Algorithm. Journal of Global Positioning Systems, „Im Erscheinen“.
- Heiko Will, Jakob Pfender, Stephan Adler, Thomas Hillebrandt und Marcel Kyas. A Greedy Approach to Cooperative Indoor Localization. In: Proceedings

of the 3rd International Conference on Indoor Positioning and Indoor Navigation (IPIN 2012), 2012.

- Heiko Will, Thomas Hillebrandt und Marcel Kyas. The Geo-n Localization Algorithm. In: Indoor Positioning and Indoor Navigation (IPIN), 2012 International Conference on, Seiten 1–10, 2012.
- Simon Schmitt, Heiko Will, Benjamin Aschenbrenner, Thomas Hillebrandt und Marcel Kyas. Demo: A Reference System for Indoor Localization Testbeds. IPIN 2012.
- Simon Schmitt, Heiko Will, Benjamin Aschenbrenner, Thomas Hillebrandt und Marcel Kyas. A Reference System for Indoor Localization Testbeds. In: Indoor Positioning and Indoor Navigation (IPIN), 2012 International Conference on, Seiten 1–8, 2012.
- Heiko Will, Thomas Hillebrandt, Yang Yuan, Zhao Yubin und Marcel Kyas. The Membership Degree Min-Max Localization Algorithm. In: Ubiquitous Positioning, Indoor Navigation, and Location Based Service (UPINLBS'12), Seiten 1–10, 2012.
- Thomas Hillebrandt, Heiko Will und Marcel Kyas. Quantitative and Spatial Evaluation of Distance-Based Localization Algorithms. In: J. M. Krisp (Hrsg.), Progress in Location-Based Services, Lecture Notes in Geoinformation and Cartography, Seiten 173–194. Springer Berlin Heidelberg, 2013.
- Heiko Will, Thomas Hillebrandt und Marcel Kyas. Wireless Sensor Networks in Emergency Scenarios: The FeuerWhere Deployment. In: Proceedings of the 1st ACM International Workshop on Sensor-Enhanced Safety and Security in Public Spaces, SESP '12, Seiten 9–14, New York, NY, USA, 2012. ACM.
- Heiko Will. Poster: Indoor Node Localization in Wireless Sensor Networks, The 11th ACM/IEEE Conference on Information Processing in Sensor Networks (IPSN), 2012.

- Heiko Will, Thomas Hillebrandt und Marcel Kyas. The FU Berlin Parallel Lateration-Algorithm Simulation and Visualization Engine. In: Positioning Navigation and Communication (WPNC), 2012 9th Workshop on, Seiten 131–136, 2012.
- Stephan Adler, Thomas Hillebrandt, Stefan Pfeiffer, Jochen Schiller und Heiko Will. Distance Measurement in Wireless Sensor Networks with low cost components. In: Proceedings of the 2nd International Conference on Indoor Positioning and Indoor Navigation (IPIN 2011), 2011.
- Stephan Adler, Thomas Hillebrandt, Stefan Pfeiffer, Jochen Schiller und Heiko Will. Demo: Distance Measurement in Wireless Sensor Networks with low cost components. In: Proceedings of the 2nd International Conference on Indoor Positioning and Indoor Navigation (IPIN 2011), 2011.
- Enrico Köppe, Heiko Will, Achim Liers und Jochen Schiller. Tracking Persons with an Autarkic Radio-Based Multi-Sensor System. International Conference on Indoor Positioning and Indoor Navigation (IPIN), Abstract Volume, 2010.
- Heiko Will, Norman Dziengel und Jochen Schiller. Distance-Based Distributed Multihop Localization in Mobile Wireless Sensor Networks. In: Proceedings of the 8th GI/ITG KuVS Fachgespräch "Drahtlose Sensornetze"(FGSN'09), 2009.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

AMCL . . . . . Adaptive Monte Carlo Localization

AML . . . . . Adapted Multi-Lateration

AOA . . . . . Angle of Arrival

AP . . . . . Access Point

AVX . . . . . Advanced Vector Extensions

BL . . . . . Bilateration

CDF . . . . . Cumulative Distribution Function

CluRoL . . . . . Clustering based Robust Localization

COO . . . . . Cell of Origin

CRLB . . . . . Cramér–Rao Lower Bound

CSS . . . . . Chirp Spread Spectrum

CSV . . . . . Comma-separated values

DOA . . . . . Direction of Arrival

E-Min-Max . . . . . Extended Min-Max

FHSS . . . . . Frequency Hopping Spread Spectrum

GNSS . . . . . Global Navigation Satellite System

GPS . . . . . Global Positioning System

GSM . . . . . Global System for Mobile Communications

GWT . . . . . Google Web Toolkit

ICLA . . . . . Iterative Clustering-based Localization Algorithm

ICM . . . . . Iterative Clustering Model

INS . . . . . Inertial Navigation System

JNA . . . . . Java Native Access

kNN . . . . . k-Nearest-Neighbor

LLS . . . . . Linear Least Squares

LMS . . . . . Least Median of Squares

LOS . . . . . line-of-sight

LS . . . . . Least Squares

LS^2 . . . . . The FU Berlin Parallel Lateration-Algorithm Simulation and Visualization Engine

LSE . . . . . Least Squares Estimator

MAE . . . . . Mean Absolute Error

MANET . . . . . Mobile Ad Hoc Network

MD-Min-Max . . . . Membership Degree Min-Max

MED . . . . . Median

MF . . . . . Membership Function

ML . . . . . Maximum Likelihood

MLE . . . . . Maximum Likelihood Estimation

MMX . . . . . MultiMedia eXtension

MSE . . . . . Mean Squared Error

NLLS . . . . . Nonlinear Least Squares

NLOS . . . . . non-line-of-sight

POA . . . . . Phase of Arrival

PTTU . . . . . Personal Tracking and Transmission Unit

RANSAC . . . . . Random Sample Consensus

REST . . . . . Representational State Transfer

RLSM . . . . . Robust Least-Squared Multilateration

RMSE . . . . . Root Mean Squared Error

ROS . . . . . Robot Operating System

RSS . . . . . Received Signal Strength

RSSI . . . . . Received Signal Strength Indicator

RTOF . . . . . roundtrip time-of-flight

RTT . . . . . Round-Trip Time

Rwgh . . . . . Residual weighting algorithm

SDS-TWR . . . . . Symmetrical Double-Sided Two Way Ranging

SIMD . . . . . Single instruction, multiple data

SISD . . . . . Single instruction, single data

SISR . . . . . Snap-Inducing Shaped Residuals

SSE . . . . . Streaming SIMD Extensions

TDOA . . . . . Time Difference of Arrival

TOA . . . . . Time of Arrival

TOF . . . . . time-of-flight

ToM . . . . . Trilaterate on Minima

UTM . . . . . Universal Transverse Mercator

UWB . . . . . Ultra-wideband

VBLE . . . . . Voting-Based Location Estimation

VBLE-O . . . . . Optimized Voting-Based Location Estimation

WDF . . . . . Wahrscheinlichkeitsdichtefunktion

WGS 84 . . . . . World Geodetic System 1984

WLAN . . . . . Wireless Local Area Network

WLLS . . . . . Weighted Linear Least Squares

WLS . . . . . Weighted Least Squares

WSN . . . . . Wireless Sensor Network

KAPITEL 1

Einleitung

Zwei Entwicklungen haben in den letzten Jahren zu einem wahren Boom bei ortsbezogenen Diensten geführt: Die erste Entwicklung war die Inbetriebnahme des global verfügbaren *Global Positioning System* (GPS) und die zweite Entwicklung war das Aufkommen von mobilen Endgeräten wie Smartphones und Tablet-PCs für die breite Masse [107, 162, 163]. Mit dem Zusammenführen beider Technologien sind eine Vielzahl verschiedener ortsbezogener Anwendungen entstanden. Die bekannteste ist sicherlich die automatische kartenbezogene Navigation für Kraftfahrzeuge. Mit der weiteren Entwicklung mobiler Rechner, insbesondere im Bereich der Smartphones, sind Navigationsanwendungen allerdings schon lange nicht mehr auf Kraftfahrzeuge mit spezialisierter Hardware begrenzt. Heutzutage ist eine kartengestützte Navigation auf Mobiltelefonen sowohl für Kraftfahrzeuge, für Radfahrer, für Wassersportler und für Fußgänger verfügbar. Mit der allgemeinen Verfügbarkeit und der weit verbreiteten Akzeptanz der Navigationsanwendung ist auch der Bedarf an der Verfügbarkeit einer Indoor-Lösung, die in geschlossenen Gebäuden beliebiger Größe zuverlässig funktioniert, gewachsen [107].

Satellitenbasierte Lösungen scheiden technologiebedingt für dieses Anwendungsgebiet aus [107], so dass spezialisierte Lösungen sowohl im Bereich der Hardware als auch im Bereich der Software benötigt werden. Durch die Verschiedenheit der Anwendungsszenarien, die von einer zentimetergenauen Lokalisierung von einzelnen Gabelstapler-Zinken für die automatische Containerumsetzung bis hin zur nur raumgenauen Lokalisierung für Navigationssysteme in Einkaufszentren reichen kann, wird schnell deutlich, dass es auch verschiedener Lösungen bedarf. Spielen bei der Containerumsetzung

technologischer Aufwand und Kosten nur eine untergeordnete Rolle, sind beide Punkte für eine akzeptierte smartphonebasierte Lösungen entscheidend. In der Literatur wird hier zwischen hoch präziser und minder präziser Lokalisierung unterschieden - auch wenn eine genaue Trennschärfe zwischen diesen beiden Klassen nicht gegeben ist.

Abhängig von gewünschter Genauigkeit und daraus folgendem technischen Aufwand, lässt sich die Lokalisierung auch nach der verwendeten Technologie einteilen. So existieren beispielsweise distanzbasierte Verfahren, Fingerprinting-Verfahren oder tomographische Verfahren. Diese Verfahren zeichnen sich in der Praxis sowohl durch unterschiedliche Genauigkeiten, als auch durch unterschiedliche Hardwareanforderungen, als auch durch die jeweils verwendeten Algorithmen aus. Kern dieser Arbeit ist die Untersuchung hoch präziser distanzbasierter Verfahren, wie sie bei einer Vielzahl von Szenarien der Indoorlokalisierung zum Einsatz kommen.

Nicht untersucht haben wir Verfahren, die auf anderen Technologien basieren, wie z. B. Fingerprinting oder Dead Reckoning. Diese Verfahren ermöglichen ebenfalls die Bestimmung einer Position, schränken aber durch ihre speziellen Vorbedingungen den allgemeinen Einsatz zum Teil stark ein. Distanzbasierte Systeme hingegen setzen neben einer geeigneten Hardware keinerlei weitere Einschränkungen wie das Vorhandensein von Karten oder das kontinuierliche Tracking bis hin zur gesuchten Position voraus.

1.1 Problemstellung und Zielsetzung

Distanzbasierte Verfahren basieren auf der hardwareseitigen Messung einer physikalischen Größe auf einem Gerät und der softwareseitigen Schätzung einer Entfernung zwischen dem Gerät und einem weiteren Punkt, basierend auf dieser Messung. Liegen für ein Gerät mehrere solcher Schätzungen vor, kann mithilfe geeigneter Algorithmen eine Schätzung der Position des Gerätes vorgenommen werden. Die Genauigkeit einer solchen Positionsschätzung ist von mehreren Faktoren abhängig. So wird die Messung der physikalischen Größe einem Messfehler unterliegen, die Schätzung der Entfernung kann diversen Modellfehlern unterliegen und die Schätzung der Position aus diesen Daten wiederum wird ebenfalls diversen Modellfehlern unterliegen. Beispielsweise wird eine Entfernungsbestimmung mittels der Laufzeitmessung von Audiosignalen einem Messfehler beim Bestimmen der Zeit unterliegen, aber auch dem Modellfeh-

ler, der von einer direkten Verbindung der beiden Signalquellen ausgeht - das Signal kann auch an einer Wand reflektiert worden sein. Kommt dann als Algorithmus die klassische Trilateration zum Einsatz, wird der Modellfehler darin bestehen, dass der Algorithmus davon ausgeht, dass die Entfernungen exakt bestimmt worden sind. Die Minimierung des letztgenannten Modellfehlers ist ein wesentliches Designziel bei der Entwicklung von Lokalisierungsalgorithmen.

Ein weiteres offenes Problem in der Indoorlokalisierung ist die Bewertung und der Vergleich verschiedener Algorithmen anhand von Gütekriterien. Sowohl diese Gütekriterien sind im Wesentlichen unbestimmt, als auch die Bedingungen, unter denen diese zu ermitteln sind. So zeichnen sich die wesentlichen Veröffentlichungen durch eine Vielzahl unterschiedlicher Simulationsansätze und -Parameter aus und auch die Bedingungen, unter denen diese ermittelt wurden, variieren stark. Abseits von Simulationen fällt auf, dass wirkliche Experimente die absolute Ausnahme darstellen. Dies lässt sich zum einen auf mangelnde Hardwareunterstützung für die jeweiligen Verfahren zurückführen, zum anderen sicherlich auf die Komplexität vieler solcher Experimente.

Die zwei Hauptfragen dieser Arbeit behandeln einerseits die Entwicklung geeigneter Algorithmen für die Indoorlokalisierung und andererseits die Herausarbeitung geeigneter Güteparameter und nachvollziehbarer Bedingungen - sowohl in der Simulation, als auch in Experimenten.

Als wesentlichen Beitrag führen wir die Betrachtung der räumlichen Fehlerverteilung als Analysekriterium für Algorithmen zur Indoorlokalisierung ein. Die grafische Darstellung dieser Metrik erlaubt einen für die Praxis aussagekräftigen Vergleich verschiedener Algorithmen unter Berücksichtigung der zu erwartenden Umgebungsparameter. Zur Visualisierung und Auswertung dieser Metrik haben wir geeignete Werkzeuge entwickelt und stellen sie im Rahmen dieser Arbeit vor. Diese Visualisierung erlaubt es im Gegensatz zu in Experimenten oder Simulationen ermittelten Durchschnittswerten für die Genauigkeit der verschiedenen Verfahren, die durch ihre sehr starke Kontextabhängigkeit immer nur eine Aussage über das jeweilige Experiment zulassen, eine allgemeinere Aussage über das Verhalten und die Performance des Verfahrens zu treffen. Das Auswählen geeigneter Algorithmen für eine bestehende Problemstellung und das gezielte Verbessern bestehender Algorithmen wird damit deutlich vereinfacht.

1.1.1 Algorithmen für die Indoorlokalisierung

Aufgabe eines Algorithmus in der distanzbasierten Indoorlokalisierung ist es, aus geschätzten Entfernungen zu Punkten mit bekannter Position die eigene bisher unbekannte Position möglichst genau zu schätzen. Wesentliche Kriterien für einen solchen Algorithmus könnten beispielsweise die Stabilität gegenüber Ausreißern in den gemessenen Entfernungen aber auch der aus redundanten Entfernungsmessungen gezogene Nutzen oder die Rechenkomplexität sein. In dieser Arbeit stellen wir zwei neuartige Algorithmen zur hochpräzisen, distanzbasierten Indoorlokalisierung vor und vergleichen sie mit bekannten und in der Praxis eingesetzten Algorithmen. Der von uns entwickelte Geo-n Algorithmus stellt eine wesentliche Verbesserung in Bezug auf die Genauigkeit der bisher bekannten Algorithmen dar und lässt sich auch auf Kleinstcomputern, wie sie beispielsweise in *Wireless Sensor Network*s (WSNs) genutzt werden, einsetzen. Durch die Nutzung dieses Algorithmus werden Anwendungen der Indoorlokalisierung deutlich robuster gegenüber Umgebungseffekten und unabhängiger gegenüber der verwendeten Hardware bzw. deren Genauigkeit und somit auch deren Kosten.

1.1.2 Simulationen und Experimente für die Indoorlokalisierung

Um zwei oder mehr Algorithmen zur Indoorlokalisierung zu vergleichen, bedarf es einem oder mehrerer Güteparameter, die eine für die Praxis relevante Aussage über die Güte dieser Algorithmen erlauben. Für verschiedene Einsatzzwecke können durchaus unterschiedliche Kriterien eine jeweils unterschiedliche Relevanz besitzen. Sind diese Kriterien festgelegt, ist noch die Art und Weise der Erhebung festzulegen. Die Art und Weise hat hierbei einen wesentlichen Einfluss auf das Ergebnis. Neben rein mathematischen Methoden kommen als wesentliche Methoden die Simulation und das Experiment in Betracht. Simulationsergebnisse sind aufgrund der hohen Anzahl an Umgebungseinflüssen und der äußerst komplexen Simulation einiger dieser Einflüsse nur sehr schwer auf die allgemeine Problemstellung zu übertragen und können daher oft nur als Indizien für eine allgemeinere Gütebestimmung herangezogen werden. Wesentlicher Nachteil von Experimenten ist der große Aufwand ihrer Durchführung und die schwierige Reproduzierbarkeit. Im Rahmen dieser Arbeit entwickeln wir Kriterien für die Vergleichbarkeit verschiedener Algorithmen anhand von Simulationen und Experimenten und stellen ein virtuelles Testbed als hybrides System aus Simulation

und Experiment für die Gütebestimmung von Lokalisierungsalgorithmen vor.

Durch das virtuelle Testbed ermöglichen wir es, Experimente zur Indoorlokalisierung ohne die Nutzung von teuren und aufwändigen Referenzsystemen durchzuführen. Mithilfe einer Datenbank, die eine große Anzahl von Distanzmessungen für jeden Ort eines Gebäudes enthält, lassen sich virtuelle Pfade in diesen Gebäuden konstruieren und mit verschiedenen Algorithmen vergleichbar und reproduzierbar auswerten. Durch die offene Struktur lassen sich einfach weitere Gebäude und verschiedene Messsysteme integrieren. Diese Datenbank ermöglicht es Forschern auf diesem Gebiet neuartige Algorithmen und Ansätze sofort mit einem Standarddatenset zu evaluieren und somit eine Vergleichbarkeit mit vorherigen Arbeiten zu gewährleisten.

1.2 Aufbau der Arbeit

Die Arbeit ist wie folgt aufgebaut: In Kapitel 2 widmen wir uns den gängigen Methoden zur Lokalisierung von Personen oder Objekten und klassifizieren die verschiedenen Arten der Lokalisierungsansätze, welche in der Literatur behandelt werden. Hierbei wird auf die Besonderheiten der Indoorlokalisierung eingegangen und abschließend werden die gebräuchlichsten Verfahren zur Distanzmessung kurz erläutert, da wir uns im Rahmen dieser Arbeit vor allem auf distanzbasierte Algorithmen konzentriert haben.

Kapitel 3 gibt einen Überblick über entfernungsbasierte Lokalisierungsalgorithmen, darunter sind bekannte Standardverfahren sowie neuere Algorithmen zu finden, die speziell für die Anwendung in Sensornetzen entwickelt worden sind. Anschließend werden drei von uns entwickelte Algorithmen speziell für die Indoorlokalisierung vorgestellt und schließlich die von uns entwickelte LatMath Bibliothek präsentiert, die unter anderem eine Referenzimplementierung sämtlicher Algorithmen enthält.

In Kapitel 4 diskutieren wir den Einfluss verschiedener Umgebungsparameter auf das Lokalisierungsergebnis und erörtern verschiedene Metriken zur Beurteilung der Lokalisierungsverfahren. Im Besonderen stellen wir die von uns eingeführte Metrik der räumlichen Fehlerverteilung vor.

In Kapitel 5 führen wir in Simulationen und Experimente im Bereich der distanzbasierten Indoorlokalisierung ein, indem wir eine ausführliche Auswertung der uns vorliegenden Literatur in diesem Bereich vornehmen, dabei Gemeinsamkeiten und Un-

terschiede herausstellen sowie vorhandene Schwächen aufzeigen. Anschließend stellen wir die von uns entwickelte Simulationsengine LS^2 vor, mithilfe derer die simulative Untersuchung der Algorithmen in Kapitel 6 erfolgt und präsentieren mit *LatViz* kurz ein von uns entwickeltes Werkzeug für die experimentelle Algorithmenauswertung. Abschließend präsentieren wir ein virtuelles Testbed, was die Durchführung und Auswertung von Experimenten im Bereich der Indoorlokalisierung erheblich vereinfacht.

In Kapitel 6 und 7 werden sämtliche der in Kapitel 3 eingeführten Algorithmen simulativ und experimentell untersucht. Hierbei finden im Wesentlichen die Bewertungsmetriken aus Kapitel 4 ihre Anwendung, wobei die Analyse der Algorithmen sowohl in der Simulation als auch in den Experimenten unter verschiedenen Bedingungen erfolgt, um die Stärken und Schwächen der einzelnen Algorithmen besser beurteilen zu können.

Kapitel 8 schließt die Arbeit mit der Zusammenfassung der Erkenntnisse und einem Ausblick auf Schwerpunkte zukünftiger Forschung ab.

KAPITEL 2

Lokalisierung

Unter Lokalisierung verstehen wir die Möglichkeit eines Systems unter Zuhilfenahme verschiedenster technischer Mittel den Standort von Personen oder Objekten zu bestimmen. Hierbei kann der Standort entweder relativ zu einer bekannten Position bestimmt werden oder innerhalb eines globalen Koordinatensystems. Häufig benutzte Koordinatensysteme sind z. B. das *World Geodetic System 1984* (WGS 84) und das UTM-System (von englisch *Universal Transverse Mercator*). Selbst die symbolische Angabe der Position (z. B. *im Büro* oder *in Laborraum 77*) reicht in manchen Anwendungsfällen als Positionsangabe aus. Im Rahmen dieser Arbeit wollen wir uns auf Computersysteme beschränken und schränken diese Computersysteme in der Regel weiter auf mobile vernetzte Kleincomputer ein, wie sie zum Beispiel in *Mobile Ad Hoc Network*s (MANETs) oder WSNs anzutreffen sind. Die verschiedenen Lokalisierungstechniken unterscheiden sich grundsätzlich in der Art der benötigten Umgebungsparameter und je nach Art der Parameter noch einmal in der Art und Weise, wie diese erhoben werden. Um in die Thematik einzuführen, geben wir im Folgenden einen kurzen Abriss über die grundlegenden Techniken zur Lokalisierung und führen in die Besonderheiten der Indoorlokalisierung ein, um anschließend die Grundlagen der Entfernungsmessung vorzustellen, da wir uns im Rahmen dieser Arbeit besonders auf die distanzbasierte Lokalisierung konzentriert haben.

2.1 Lokalisierungstechniken

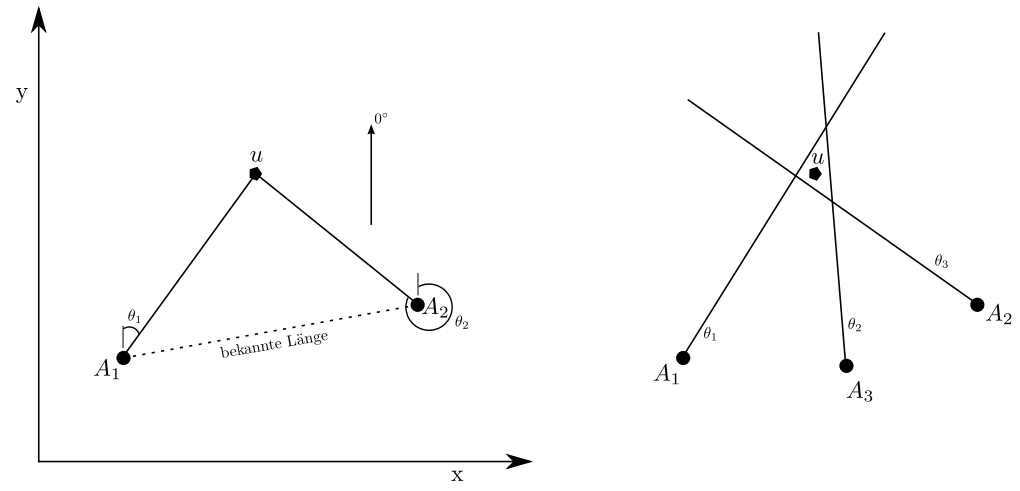
2.1.1 Nachbarschaftsbasierter Ansatz

Die rudimentärste Art der Positionsschätzung sind wohl Systeme, die keinerlei zusätzliche Hardware benötigen und die Positionsschätzung direkt aus der Nähe zu bekannten fixen Knoten ableiten. Dieses Verfahren ist in der Literatur bekannt als nachbarschaftsbasierte Lokalisierung (engl.: *proximity location sensing*) [51, 84, 44]. Hierbei wird die Anwesenheit eines zu lokalisierenden Objekts aus der Detektion eines physikalischen Phänomens mit begrenzter Reichweite abgeleitet. Dazu kann beispielsweise der direkte physische Kontakt mit einem Objekt bekannter Position verwendet werden, ermittelt z. B. durch Berührungs- oder Drucksensoren. Eine weitere Möglichkeit ist die Überwachung der Signale des mobilen Knotens durch ein flächendeckendes Netzwerk. Im einfachsten Fall wird dann z. B. die Position des fixen Knotens mit der größten Signalstärke in der Umgebung als eigene Position angenommen. Aufwändigere Verfahren berücksichtigen dabei die gesamte Umgebung aus fixen Knoten und rechnen beispielsweise den Schwerpunkt aller dieser Knoten als eigene Position aus [16]. Beispiele für solche Systeme sind das *Active Badge Location System* von Want et al. [137] und die *Zellortung* (*Cell of Origin* (COO)) in Mobilfunknetzen.

All diesen Verfahren ist die begrenzte Genauigkeit gemein, die oft direkt von der Dichte der ausgebrachten fixen Knoten abhängt. Besonders geeignet sind diese Verfahren für bereits existierende Netzwerke, die um eine Lokalisierungskomponente erweitert werden sollen, da keine Spezialhardware benötigt wird, die in herkömmlichen WSNs nicht vorhanden ist. Auch zur Einhaltung strikter Energiesparrichtlinien sind diese Verfahren geeignet, da bei einem Mindestmaß an Kommunikation innerhalb des Netzwerkes auch keinerlei zusätzlicher Netzwerkverkehr erzeugt werden muss.

2.1.2 Winkelbasierter Ansatz

Eine der ältesten Formen der Lokalisierung sind winkelbasierte Ansätze, auch Angulation genannt. Diese Methode geht vermutlich auf den Holländischen Mathematiker Gemma Frisius zurück, der sie um 1533 [102] zur Landvermessung entwickelte. Bei der Angulation wird über verschiedene technische Ansätze der eigene Winkel zu mehreren bekannten Signalquellen bestimmt und dann über Sätze aus der Trigonometrie



(a) (b)

Abbildung 2.1: Abbildung 2.1a: Illustration der 2D-Angulation am Beispiel der Lokalisierung von Knoten u mit zwei Winkeln relativ zu einem 0° Referenzvektor. Abbildung 2.1b: Angulation mit drei Ankern in der Gegenwart von Messfehlern.

die eigene Position errechnet. Für die Lokalisierung in der Ebene bedarf es zweier Signalquellen, zu denen der Winkel über ein eintreffendes Signal bestimmt werden kann und deren Positionen bekannt sind. Dies veranschaulicht Abbildung 2.1a mit einem sendenden Knoten u und zwei festen Knoten A_1 und A_2 mit Richtantennen. Die Koordinaten von A_1 und A_2 sind bekannt und die Ankunftswinkel θ_1 und θ_2 des Signals werden relativ zu einem 0° Referenzvektor (z. B. magnetischer Norden) gemessen. Mithilfe der beiden Winkel und der bekannten Länge der Seite zwischen den festen Knoten lassen sich die beiden anderen Seitenlängen des Dreiecks berechnen und damit auch die Position von u bestimmen. Im dreidimensionalen Raum wird zusätzlich eine Azimut-Messung benötigt [51]. Das Verfahren ist in der Literatur auch bekannt als *Angle of Arrival* (AOA) oder *Direction of Arrival* (DOA).

Bedeutung hatte diese Methode auch in der Seefahrt, wenn die genaue Position in küstennahen Gewässern bestimmt werden sollte. So konnte von einem Schiff mit einem Sextanten der Winkel zu zwei in der Seekarte eingezeichneten Landmarken angepeilt werden und aus diesen beiden Winkeln die Position auf dem Wasser sehr genau bestimmt werden. Ebenso hat das Verfahren auch andersrum funktioniert, sofern diese Landmarken (Leuchttürme) miteinander in Kontakt standen, um die gemessenen Winkel zu übermitteln. Die Triangulation war vor allem deshalb lange zur Land-

vermessung verbreitet, weil es in der Praxis und auf eventuell unwegsamen Terrain wesentlich einfacher war Winkel präzise zu ermitteln, als beispielsweise Entfernungen.

In der Praxis ergibt sich aus der Nutzung mehrerer Landmarken ($N > 2$) allerdings ein neues Problem: Wie Abbildung 2.1b zeigt, führen bereits kleine Messfehler in der Winkelmessung dazu, dass die gedachten Strahlen von den vermessenen Landmarken sich nicht mehr in einem Punkt kreuzen, sondern sich drei Schnittpunkte ergeben. Man kann nun entweder die Position der drei Schnittpunkte berechnen und dann den gewichteten Schwerpunkt als eigene Position bestimmen oder man nutzt einen probabilistischen Maximum Likelihood (ML) Schätzer, um die wahrscheinlichste Position zu bestimmen [85, S. 6 ff.]. Sei θ_i der Winkel zwischen der Signalquelle u mit Koordinaten (x, y) und i-tem Anker mit Koordinaten (x_i, y_i), dann gilt:

$$\tan(\theta_i) = \frac{y - y_i}{x - x_i}, i = 1, 2, \ldots, N; N \geq 2.$$

In der Gegenwart von Messfehlern kann der Vektor α von gemessenen Winkeln folgendermaßen ausgedrückt werden:

$$\alpha = \theta(u) + \eta.$$

Hierbei wird üblicherweise angenommen, dass die echten Winkel $\theta(u)$ durch additives Gaußsches Rauschen mit Erwartungswert Null und Kovarianzmatrix $S = \text{diag}\{\sigma_1^2, \ldots, \sigma_N^2\}$ verfälscht wurden. Die geschätzte Position $\hat{u}$ berechnet sich dann mittels ML-Schätzer wie folgt:

$$\hat{u} = \arg\min \left[\theta(\hat{u}) - \alpha\right]^T S^{-1} \left[\theta(\hat{u}) - \alpha\right].$$

Für die Lokalisierung innerhalb mobiler Netzwerke ist es praktisch allerdings sehr kompliziert und Hardware aufwändig, Winkel zu eintreffenden Signalen zu bestimmen. Römer hat mit dem Lighthouse System [111] ein optisches Verfahren vorgestellt, was auch für den Indoor-Einsatz praktikabel ist und die Winkel eines rotierenden Punktstrahlers mittels genauer Zeitsynchronisation bestimmen kann. Allerdings ist für die Angulation eine direkte Sichtverbindung (line-of-sight (LOS)) erforderlich und die benötigte Hardware ist für viele Einsatzgebiete in mobilen Netzwerken zu komplex und zu unhandlich, weshalb sie in allgemeinen Verfahren zur Lokalisierung kaum zum Einsatz kommt. Wesentlicher Vorteil dieses Verfahrens ist es aber, dass die

mobilen Knoten sehr einfach und klein gehalten werden können.

Auch elektromagnetische Verfahren wie *Beamforming* oder *Phased-Array-Antennen* spielen in mobilen Netzwerken aufgrund der hohen Hardware- und Umgebungsanforderungen nur eine untergeordnete Rolle - obgleich eine Vielzahl von Algorithmen existieren, deren Autoren die Hardwareanforderungen bzw. -verfügbarkeit nur am Rande behandeln [101, 68, 29]. Erkenntnisse, die über Simulationen hinausgehen, liegen nur in frühen Experimentierstadien vor und sind nur schwer auf mobile Netzwerke zu übertragen [122].

Der große Vorteil ist allerdings die fast beliebig hohe Genauigkeit, die sich mit diesen Systemen erzielen lässt, ohne dass der Hardwareaufwand wesentlich weiter als der grundlegend benötigte Aufwand zur Realisierung solcher Systeme steigt. Zusätzlich wird bei AOA-Systemen keine Zeitsynchronisierung benötigt, wie es bei anderen Verfahren, die später in diesem Kapitel vorgestellt werden, der Fall ist und auch keine Einschränkungen hinsichtlich Signalmodulation und Protokoll gemacht [9, S. 28].

2.1.3 Entfernungsbasierter Ansatz

Die am weitesten verbreitete Art von digitalen Lokalisierungssystemen stellen sicherlich die entfernungsbasierten Systeme dar. Diese Systeme funktionieren analog zu den auf Angulation basierenden Systemen, nur das statt eines Winkels zu einem fixen Punkt eine Entfernung zu diesem gemessen wird. Abbildung 2.2 zeigt dies beispielhaft mit drei Ankerknoten und den zu diesen ermittelten Distanzen, ausgehend von Knoten u. Wie man sehen kann, sind die Distanzen fehlerbehaftet und die Kreise schneiden sich nicht alle in einem gemeinsamen Punkt. Das Bestimmen einer Position aus den gemessenen Entfernungen wird auch als Lateration bezeichnet. Um aus den zu Fixpunkten gemessenen Entfernungen die Position eines unbekannten Knotens zu bestimmen, existieren diverse Ansätze - einige davon, die für mobile Netzwerke interessant sind, stellen wir in Kapitel 3 vor. Diese Systeme unterscheiden sich wesentlich in der Art und Weise, aus den Entfernungen eine Position zu errechnen.

Da sowohl seitens der zur Distanzmessung benutzten Hardware als auch seitens der zur Positionsbestimmung benutzen Algorithmen eine große Bandbreite an Lösungen existieren und einige dieser Lösungen besonders gut für mobile Netzwerke geeignet sind, werden wir uns im Rahmen dieser Arbeit auf die entfernungsbasierten Ansätze

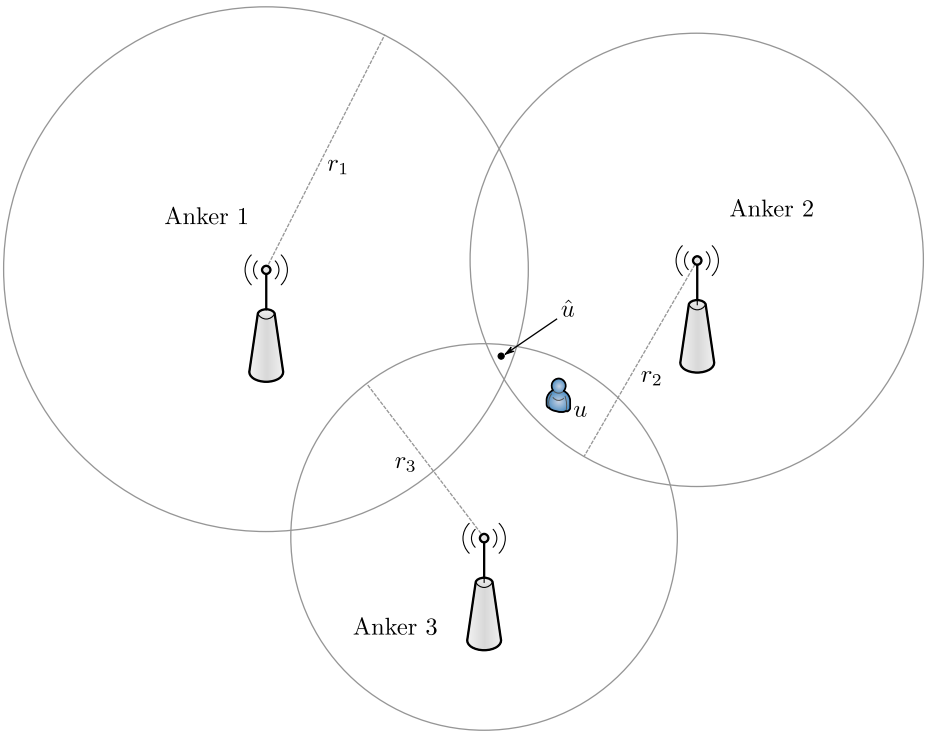


Abbildung 2.2: Ankerknoten, Distanzmessungen und nicht lokalisierter Knoten

konzentrieren. Besonders interessant ist, dass die meisten Algorithmen zur Lateration unabhängig von der Methode der Distanzbestimmung funktionieren und von einem Fortschritt seitens der Laterationsalgorithmen somit gleichzeitig eine Vielzahl von bereits eingesetzten Systemen profitieren können, ohne dass zwingend ein Austausch der Hardware nötig wäre.

Die bekannteste Anwendung der entfernungsbasierten Lokalisierung sind sicherlich *Global Navigation Satellite System*s (GNSSs) wie zum Beispiel das weltweit verfügbare und breit genutzte GPS. Aufgrund der in Abschnitt 2.2 beschriebenen Besonderheiten und Anforderungen an die Lokalisierung in mobilen Netzwerken werden für den Einsatz in diesen Systemen spezielle Algorithmen, die auf die besonderen Eigenschaften dieser Systeme optimiert sind, bevorzugt, da diese in der Regel einen deutlich niedrigeren Positionsfehler bieten.

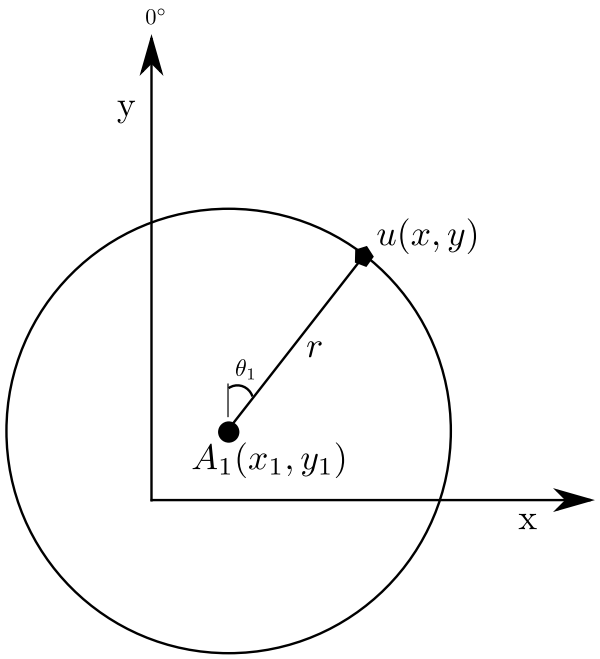


Abbildung 2.3: Lokalisierung mit Distanz und Winkel unter idealen Bedingungen

2.1.4 Kombination von AOA und TOA

Natürlich lassen sich die winkelbasierten und entfernungsbasierten Methoden auch kombinieren, falls die Hardware der ausgebrachten Knoten die Messung beider Größen erlaubt. Dies hat den Vorteil, dass nur noch ein fixer Knoten zur Positionsbestimmung des mobilen Knotens benötigt wird, wie Abbildung 2.3 zeigt. Dies kann zur Kostenersparnis beitragen sowie den Kommunikationsaufwand im Netzwerk senken und damit gleichzeitig die Gesamtlebensdauer erhöhen. Der gesuchte Knoten u befindet sich am Schnittpunkt des Kreises mit Radius r mit der Geraden durch A_1, die einen Winkel von θ_1 zu einem 0° Referenzvektor besitzt. Die Richtantenne, so wie in der Abbildung dargestellt, befindet sich bei dem fixen Knoten. Dies ist aber nicht zwingend notwendig. Die Position des Knotens u ergibt sich dann aus:

$$x = x_1 + r \cdot \sin(\theta_1)$$
$$y = y_1 + r \cdot \cos(\theta_1)$$

Dieses Verfahren besitzt - ähnlich wie der rein winkelbasierte Ansatz - den Nachteil, dass es für den allgemeinen Fall der Indoorlokalisierung nicht anwendbar ist.

2.1.5 Fingerprinting-Ansatz

Ein Verfahren, was gänzlich ohne Infrastruktur[1] und fixe Knoten auskommt, ist das sogenannte Fingerprinting. Dieses Verfahren wird in der Literatur der *scene analysis* zugeordnet, bei der die eigene Position anhand von markanten Merkmalen der direkten Umgebung zu bestimmen versucht wird. Basis ist ein Vergleich mit gemessenen Werten an verschiedenen Orten, wobei für den Vergleich visuelle (z. B. Bilder, die mit einer Kamera aufgenommen wurden oder davon abgeleitete und vereinfachte Merkmale, wie die Umrisslinie einer Landschaft) oder nicht-visuelle Parameter verwendet werden können. Diese Verfahren bestehen deshalb in der Regel aus zwei Phasen. In der ersten Phase werden beliebige Umgebungsparameter, denen eine geringe zeitliche Fluktuation unterstellt wird, aufgenommen und zusammen mit der Position, an der sie erhoben wurden, in eine Datenbank abgelegt. In der zweiten Phase findet die eigentliche Lokalisierung der mobilen Knoten statt. Diese erheben die selben Umweltparameter oder eine Teilmenge von ihnen, die in Phase Eins erhoben wurden und schlagen die gespeicherte Konstellation in der Datenbank aus der ersten Phase nach. Im besten Falle gibt es genau einen Treffer in der Datenbank und die nachgeschlagene Position entspricht der gesuchten Position. Natürlich können auch fortschrittlichere Techniken zur Mustererkennung angewandt werden. Dazu zählen beispielsweise probabilistische Methoden, die das Maximum einer Likelihood-Funktion bestimmen [84] oder die *k-nearest-neighbor* (kNN) Methode [9, S. 147 ff.]. Bei kNN werden die k besten Treffer ermittelt und deren arithmetisches Mittel als Position genommen. Die besten Treffer sind dabei diejenigen, die die Distanz im Signalraum zwischen den beobachteten Messungen und den vorher aufgenommen Messungen minimieren. Als Distanz-Metrik kann hierbei z. B. der euklidische Abstand oder die Manhattan-Distanz genommen werden.

In der Praxis stehen diesem einfachen Prinzip jedoch diverse Hürden entgegen. Die größte Herausforderung ist es, Umweltparameter zu finden, die sich räumlich fein aufgelöst unterscheiden und gleichzeitig zeitlich stabil sind, um eine lange Einsatzfähigkeit des Systems zu gewährleisten. Ein besonders oft verwendeter Umweltparameter sind beispielsweise die Signalstärken verschiedener *Wireless Local Area Network* (WLAN) *Access Point*s (APs) [31, 136] oder *Global System for Mobile Communications* (GSM) Basisstationen [98, 131] oder beider [15, 88] in der Umgebung des

[1] Für dieses Verfahren wird - wenn überhaupt - nur bereits vorhandene Infrastruktur benutzt. Somit benötigt die eigentliche Lokalisierung keine zusätzliche Infrastruktur.

mobilen Knotens. Die Signalstärken dieser Sender sind nicht nur von der Entfernung zum Empfänger abhängig, sondern auch von Eigenschaften des umgebenden Raumes wie Hindernissen, verbauten Materialien etc. Wenn sich nun einer dieser Parameter ändert, so kann u. U. keine Korrelation zwischen den Einträgen in der Datenbank und den aktuell erhobenen Daten mehr hergestellt werden. Dazu reichen mitunter schon wenige Personen, die sich durch das entsprechende Gebäude bewegen. Dieses Problem kann man durch eine Vermischung der beiden Phasen entgegenwirken: Die während der zweiten Phase zum Nachschlagen erhobenen Daten werden nach erfolgter Lokalisierung wieder in die Datenbank zurückgespielt. Trotzdem bleiben die Ergebnisse dieser Technik meistens relativ unscharf, so dass neben dem bloßen Nachschlagen der Werte meist Partikelfilter für eine kontinuierliche Lokalisierung (*Tracking*) eingesetzt werden [146].

Aktuelle Systeme erzielen Genauigkeiten im einstelligen Meterbereich [55] und kommen vor allem zur massenhaften Lokalisierung mittels Smartphones zum Einsatz, da auf Seiten der Smartphones die vorhandene Technik zum Erheben der Umgebungsparameter ausreichend ist und somit hier reine Softwarelösungen zum Einsatz kommen können. Eine wesentliche Hürde für eine noch präzisere Lokalisierung stellt das Problem dar, dass die erhobenen Daten nicht nur zeitlich schwanken, sondern mitunter erheblich vom Modell der jeweiligen Sensorik abhängen und in keinster Weise standardisiert sind. So liefern bisweilen sogar Smartphones aus verschiedenen Modellreihen des selben Herstellers vollkommen unterschiedliche Sensorwerte bei ansatzweise gleichen Umgebungsparametern.

Gegen den Einsatz in MANETs spricht vor allem die vorher zwingend benötigte Trainingsphase mittels eines Referenzsystems. Diese Vorgabe widerspricht dem Designziel von MANETs, ad hoc zum Einsatz zu kommen und nicht nur in vorher definierten und vermessenen Umgebungen.

2.1.6 Radiotomographie-Ansatz

Im Vergleich zu distanzbasierten Verfahren haben wir viel weniger Veröffentlichungen über bildgebende Verfahren aus dem Bereich der Radiotomographie [66, 64, 161] gefunden. Bei diesen Verfahren werden Signalquellen und Signalempfänger um den Einsatzort herum platziert und die Signalausbreitung paarweise für alle Signalquellen und Signalempfänger bestimmt. Aus den so gewonnenen Daten lässt sich ein

zweidimensionales Bild rekonstruieren, das die Signaldämpfung an den jeweiligen Koordinaten darstellt. Durch den Vergleich mehrerer dieser Bilder kann man sich im Raum bewegende Objekte erkennen. Wesentlicher Vorteil dieses Ansatzes gegenüber allen anderen Ansätzen ist, dass innerhalb des Einsatzszenarios keinerlei Technologie benötigt wird, also auch zu verfolgende Personen nicht mit Technik ausgestattet werden müssen.

Die Genauigkeit der letztendlich errechneten Position hängt stark von der Auflösung der gewonnen Bilder ab, die wiederum direkt von der Anzahl der verwendeten Knoten abhängt. Wesentliche Nachteile dieser Systeme sind die aufwändige Vorbereitung und das Einmessen der Umgebung. Entweder muss eine leere Umgebung eingemessen werden, um darin sicher ein Ziel zu lokalisieren oder das Ziel muss sich während der Einmessphase bewegen, um sicher erkannt zu werden. Problematisch ist auch die Lokalisierung mehrerer Ziele oder die Lokalisierung in Umgebungen, in denen Objekte vorkommen, die das betreffende Signal großflächig abschirmen (Wände, große Möbelstücke etc.). Für einen einzelnen Raum ohne solche Objekte erreichen diese Systeme bisher eine Genauigkeit von knapp unter einem Meter [64].

2.1.7 Koppelnavigation (Dead Reckoning)

Ein weitere Möglichkeit der Positionsbestimmung ohne Interaktion mit anderen Komponenten in der Umgebung ist die Koppelnavigation (engl.: *dead reckoning*). Bei dieser werden ausgehend von einer bekannten Position Bewegungsrichtung und Geschwindigkeit stetig fortgeschrieben und so über die vergangene Zeit jeweils eine neue geschätzte Position berechnet. Die Ermittelung der Bewegungsrichtung und Geschwindigkeit erfolgt in der Regel über spezielle Hardware wie Beschleunigungssensoren und Gyroskope, welche zusammen in sogenannten Inertialen Navigationssystemen (engl.: *Inertial Navigation System* (INS)) für diesen Zweck verbaut sind. Das Verfahren hat seinen Ursprung in der Seefahrt, bei dem der Kurs mittels Kompass und die Schiffsgeschwindigkeit mittels Log bestimmt wurde und wird heutzutage in verschiedenen Systemen eingesetzt. So wird es beispielsweise im Automobilbereich zur Navigation an Stellen benutzt, wo der GPS-Empfang nicht möglich oder eingeschränkt ist (z. B. in Tunneln oder großen Häuserschluchten). Hierfür werden z. B. spezielle GPS-Chips verwendet, die die Daten der Radsensoren und eines Gyroskops vom Daten-Bus des Fahrzeuges für diesen Zweck einlesen [130]. Auch zur Lokalisierung von Personen im Indoor-Bereich gibt es diverse Algorithmen und Systeme [30, 37, 33].

Die Genauigkeit dieser Methode hängt stark von der verwendeten Hardware ab. Vor allem sehr günstige Sensoren weisen einen großen Sensordrift auf, der sich bei ausschließlicher Benutzung der Koppelnavigation mit der Zeit kumulativ verstärkt, da durch den sogenannten Integrationsdrift z. B. kleine Fehler in der Beschleunigung stufenweise in größere Fehler in der Geschwindigkeit und noch größere Fehler in der Position integriert werden. Diese Fehler akkumulieren sich also proportional grob mit der vergangenen Zeit seit der initialen Positionsbestimmung. Deshalb wird dieses System häufig als unterstützendes Zweitsystem verwendet, um beispielsweise auftretende Lücken zwischen zwei GPS-Positionen oder Aufrufen eines Lokalisierungsalgorithmus im Indoor-Bereich zu interpolieren. Ist die berechnete Position mittels Zweitsystem über eine gewisse zeitlich begrenzte Periode stabil, kann durch das Absenken der Lokalisierungsrate zusätzlich die Netzwerklast und der Energiebedarf reduziert werden.

2.1.8 Probabilistische Trackingverfahren

Soll ein unbekannter mobiler Knoten nicht nur einmalig lokalisiert werden, sondern ein permanentes Tracking durchgeführt werden, so bieten sich auch probabilistische Ansätze zur Lokalisierung an. Diesen Ansätzen liegt ein physikalisches Modell über eine Verteilung von messbaren Parametern über einen bestimmten Raum zugrunde. Zusätzlich wird ein Modell, mit dem sich die Bewegung des mobilen Knotens vorhersagen lässt, benötigt. Mit dem physikalischen Modell und einer Sensorik, die die entsprechenden Parameter erfassen kann, lassen sich für alle Positionen im Raum die Wahrscheinlichkeiten berechnen, mit der sich der mobile Knoten unter den gemessenen Parametern an der getesteten Position befindet (ML). Setzt man dieses Verfahren iterativ ein, so kann über das Bewegungsmodell und der Berechnung von bedingten Wahrscheinlichkeiten nach einer gewissen Zeit eine Angabe über die wahrscheinlichste Position des mobilen Knotens gemacht werden. Mithilfe weiterer Randbedingungen der Modelle kann die Position des gesuchten Knotens zusätzlich eingeschränkt werden. Dazu zählen beispielsweise Einschränkungen der möglichen Positionen (z. B. nur auf Straßen) oder auch die Begrenzung der Bewegungsgeschwindigkeit eines Objekts oder einer Person. Ist die Position zu einem Zeitpunkt t_0 bekannt, können mögliche Positionen zum Zeitpunkt t_1 ($t_0 < t_1$) begrenzt werden. Diese Verfahren hängen jedoch im hohen Maße von der Plausibilität der beiden Modelle ab und schon kleine Fehler in den Annahmen dieser Modelle können zu sehr großen Fehlern bei der Lokalisierung führen.

Prominente Vertreter dieser Verfahren sind Kalman- oder Partikelfilter [42, 62, 3], die beide zur Lokalisierung eingesetzt werden können. Insbesondere Partikelfilter sind in letzter Zeit viel beachtet worden und werden in distanzbasierten Systemen anstelle klassischer Lokalisierungsalgorithmen eingesetzt. Vor allem in System wo der Messfehler der Distanzen sehr groß ist, sind diese Systeme aufgrund der Nutzung von Zustandsinformationen den klassischen Algorithmen oft ebenbürtig oder überlegen.

2.2 Spezielle Probleme in mobilen Indoor-Netzen

Für den Einsatz von Lokalisierungsverfahren in WSNs spielen neben rein technologischen und theoretischen Erwägungen auch eine Reihe an Einsatzszenario abhängigen Erwägungen eine Rolle. So kann bei gleicher geforderter Genauigkeit und selbem Budget für verschiedene Szenarien ein völlig unterschiedliches System zum Einsatz kommen. Im Folgenden stellen wir die wesentlichen Einschränkungen und Anforderungen kurz vor, mit denen Lokalisierungssysteme in der Praxis konfrontiert werden.

2.2.1 Indoor-Problem

Das Grundproblem der Indoorlokalisierung ist die Nichtnutzbarkeit der etablierten und vielfach erprobten GNSS-Systeme. Die Ausbreitung der durch die Satelliten erzeugten elektromagnetischen Wellen wird durch die Wände eines Gebäudes meistens komplett verhindert bzw. absorbiert. An den Stellen (Fenster, Türen etc.) wo diese Wellen ins Gebäude vordringen, sind sie nur in einem sehr beschränkten Rahmen unverfälscht verfügbar - nämlich dort wo ein direkter Sichtkontakt zum entsprechenden Satelliten besteht. Der Regelfall ist aber, dass Reflexionen der ursprünglichen Signale empfangen werden, aus denen dann aufgrund der längeren Wegstrecke des Signals falsche Positionen berechnet werden.

Aber auch ein einfaches Adaptieren von etablierten Systemen in den Indoor-Bereich ist wegen der strukturellen Eigenschaften von Gebäuden nicht möglich. So müssen alle Verfahren, die in ihren Modellen von einer direkten Sichtverbindung zwischen Anker und mobilem Knoten ausgehen, eine vollständige Installation in jedem Raum eines Gebäudes vornehmen, was die Installations- und Anschaffungskosten stark in die Höhe treibt. Und selbst wenn ein Sichtkontakt zu einer ausreichenden Anzahl an Ankerknoten für jeden Bereich des Gebäudes sichergestellt werden kann, ist es

keinesfalls sicher, dass Reflexionen nicht das Ursprungssignal auslöschen und eine noch weiter abgelenkte Reflexion schließlich als Signalquelle genutzt wird (siehe auch Kapitel 4.1.3).

Um diese Effekte zu verhindern, sind spezielle Algorithmen notwendig, die Redundanzen (z. B. in der Menge der verwendeten Ankerknoten) dazu nutzen die Ungenauigkeiten, die aus den strukturellen Gebäudeeigenschaften resultieren, auszugleichen bzw. zu eliminieren.

2.2.2 Infrastrukturbasierte Lokalisierung

Infrastrukturbasierte Lösungen können überall dort zum Einsatz kommen, wo das Gebiet, in dem die Knoten lokalisiert werden sollen, klar begrenzt und definiert ist. Das ganze Gebiet kann so mit einem fest installierten Netzwerk aus Ankerknoten überzogen werden, dass die mobilen Knoten ausschließlich mit den Knoten mit bekannter Position kommunizieren müssen, um ihre eigene Position zu bestimmen.

Ein weiterer Vorteil dieser Ansätze ist es, dass die Ankerknoten strategisch ausgebracht werden können, so dass an allen relevanten Positionen optimale Bedingungen für die Lokalisierung vorherrschen. Für eine Lokalisierung von Personen in einem Krankenhaus könnte man zum Beispiel über die Montage eines Ultraschall-*Beacons* an der Zimmerdecke jedes Raumes optimale Empfangsbedingungen sicherstellen und eine präzise Lokalisierung auf Raumebene mit einem Nachbarschaftsverfahren ermöglichen. Für Fingerprinting in einem Einkaufszentrum könnte z. B. komplett die bereits installierte Infrastruktur aus WLAN APs genutzt werden, die wahrscheinlich bereits strategisch montiert wurden, um eine vollständige Abdeckung der gesamten Verkaufsfläche zu ermöglichen.

Für eine Lokalisierung unter diesen Anforderungen kommen potenziell alle in Abschnitt 2.1 in Frage.

2.2.3 Infrastrukturlose Lokalisierung

Wesentlich komplexere Ansätze werden benötigt, wenn man die Forderung nach einer infrastrukturlosen Lösung stellt. Infrastrukturlos bedeutet in diesem Fall, dass in das betreffende Gebäude keinerlei Hardware a priori eingebracht werden kann und auch keinerlei Daten a priori erhoben werden dürfen. Sämtliche Hardware muss also

während der Nutzung des Systems ausgebracht und eventuell installiert werden. Infrastrukturlose Anwendungsfälle sind all die Anwendungsfälle, die potenziell in jedem Gebäude funktionieren sollen und nicht vom Nutzer des Gebäudes vorgesehen werden. Insbesondere bei Notfallsystemen für Rettungskräfte und ähnlichen Ansätzen spielen infrastrukturlose Systeme eine große Rolle.

Bei diesen Systemen kann nicht von der direkten Erreichbarkeit von bekannten Referenzknoten ausgegangen werden, da davon auszugehen ist, dass die unbekannten Gebäude einen potenziell größeren Durchmesser haben können, als die Reichweite des verwendeten Messsystemes ist, sofern dieses nicht sowieso auf Sichtkontakt beruht. Vielmehr basieren diese Ansätze meistens auf Rekursionen, die ein *Multi-Hop* Netzwerk aus mobilen Knoten zur Lokalisierung nutzen. Die Rekursion endet, wenn ein Knoten direkten Kontakt zu einem Ankerknoten mit fixer Position hat. Diese Systeme müssen nicht nur mit den ungenauen Messungen der Entfernung als Eingabefehler umgehen können, sondern auch die dazugehörigen Positionen als fehlerbehaftet ansehen.

Als Verfahren kommen hier nur die distanzbasierten Systeme und mit Einschränkungen auch die nachbarschaftsbasierten Verfahren in Frage.

2.2.4 Hardwarelimitierung

Die meisten Systeme dienen sicherlich dazu, Lebewesen und insbesondere Menschen zu lokalisieren. Um die Akzeptanz solcher Systeme zu erhöhen bzw. sie überhaupt erst praktikabel zu machen, sind somit die physischen Ausmaße und das Gesamtgewicht eines solchen Systemes stark begrenzt. Aus diesen Erwägungen sind es meistens mobile Klein- oder gar Kleinstcomputer, die als mobile Knoten eingesetzt werden. Auf diesen Geräten ist nicht nur die Rechenleistung und der Arbeitsspeicher durch die getroffene Hardwareauswahl begrenzt, sondern zusätzlich wird die real verfügbare Rechenleistung zusätzlich durch Anforderungen an die Batterielaufzeit beschränkt. Die gleichen Anforderungen können auch bei der Wahl der Hardware eine zusätzliche Beschränkung (neben dem Budget) erzwingen.

2.2.5 Die dritte Dimension

Potenziell können fast alle verfügbaren Technologien und Algorithmen für eine Lokalisierung im dreidimensionalen Raum eingesetzt werden. Jedoch ist es bei der Indoorlokalisierung in der Mehrzahl der Fälle sinnvoll, sich auf eine zweidimensionale Lokalisierung zu beschränken und die dritte Dimension - wo benötigt - eventuell mit einem Hilfssystem zu ermitteln.

Im Indoor-Bereich bedeutet die dritte Dimension in der Regel den Wechsel zwischen Stockwerken. Von Fahrstühlen und Treppenhäusern abgesehen findet die Lokalisierung in der Höhe also in einem stark diskretisierten Modell statt. Wenn wir von einer durchschnittlichen Stockwerkshöhe von ca. 3 Metern ausgehen würden, bräuchte man also ein System mit einer wesentlich höheren Genauigkeit als 1,50 m, um verlässliche Angaben über das Stockwerk machen zu können. In der zweidimensionalen Ebene wäre eine solche Genauigkeit für viele Einsatzzwecke mehr als ausreichend, da bei einem üblichen Gebäudelayout mit einem solchen System in der Mehrzahl der Fälle noch eine sichere Lokalisierung auf den einzelnen Raum möglich wäre.

Ein System, was eine derartige Genauigkeit für die Lokalisierung in der Ebene garantieren würde, würde aufgrund der üblichen Gebäudeeigenschaften für eine dreidimensionale Lokalisierung trotzdem mit hoher Wahrscheinlichkeit nicht funktionieren. Während in den meisten Gebäuden nur wenige Wände in Material und Dicke eine Durchdringbarkeit für die zur Lokalisierung genutzten Signale ausschließen und dort, wo das der Fall ist, oft eine ausreichende Menge an Ausschnitten wie Türen und Fenster vorhanden sind, so ist der Übergang zwischen zwei Stockwerken ein völlig anderer. Hier macht Material und Stärke ein Durchdringen so gut wie immer unmöglich und auch die Anzahl der Ausschnitte ist wesentlich geringer. Für das von uns zur Auswertung verwendete System in Kapitel 7 bestätigen die Experimente diese Vermutung. In einem Test, der ausschließlich die erfolgreichen Distanzmessungen in den verschiedenen Stockwerken zählte, ergab sich ein eindeutiges Bild. Hierbei waren 25 Anker im Obergeschoss verteilt und es wurde in allen Geschossen ein sehr ähnlicher Pfad abgeschritten (entlang der Flure). Hierbei lag die Erfolgsquote im Obergeschoss bei 33,7 %[2], im Erdgeschoss bei 6,3 % und im Untergeschoss bei 0,36 %. Es wurden in jedem Lauf ca. 10 000 Messungen durchgeführt. Die prinzipiell aus anderen Stockwerken erreichbaren Anker befanden sich in direkter Nähe zu den zwei großen Innenhöfen,

[2]In einem Messzyklus werden Distanzmessungen immer mit allen Ankern durchgeführt und nicht nur mit den Knoten in Reichweite. Details dazu finden sich in Kapitel 7.

welche in Abbildung 7.7 aus Abschnitt 7.4 in Grün hervorgehoben sind. Diese verbinden die einzelnen Stockwerke miteinander und ermöglichen im Erdgeschoss eine Lokalisierung in 23 % der Fälle (drei oder mehr Anker verfügbar), im Keller ist jedoch nie eine Lokalisierung möglich. In einem Gebäude mit durchgängiger Decke würde sich die Erfolgsquote noch einmal deutlich verschlechtern. Von daher wird sich das zum Lokalisieren benötigte Netz aus Ankerknoten in der Regel nach Stockwerken partitionieren und eine Ermittlung des Stockwerkes ist bereits durch die Kenntnis über die ausgebrachte Hardware möglich.

Für die infrastrukturlose Indoorlokalisierung tritt ebenfalls diese Partitionierung auf, aber ein Erkennen des Stockwerkes ist mangels fest installierter Anker auf diesem Wege nicht mehr möglich. Um eine Lokalisierung in der dritten Dimension trotzdem zu ermöglichen, kann auf Hilfssysteme ausgewichen werden, die direkt eine Ermittlung der Höhe zulassen - eine Nutzung des Ankersystems als dreidimensionaler Ansatz ist wegen der Partitionierung auch in diesem Falle nicht möglich. Als Hilfssystem bietet sich vor allem die Messung des Luftdruckes mittels barometrischer Sensorik an [135, 134]. Diese Sensoren sind hoch präzise und in der Praxis bewährt. Für heutige Smartphones gehören sie zunehmend zur Standardausstattung und die Software zur Bestimmung der Höhe bzw. einer Höhendifferenz ist bereits vorhanden. Auch für den Betrieb in WSNs sind diese Sensoren sowohl von den Kosten als auch von der Stromaufnahme her geeignet. Je nach Einsatzszenario kann auf die Bestimmung der absoluten Höhe verzichtet werden und nur der relative Höhenoffset ab einem bestimmten Zeitpunkt (zum Beispiel dem Betreten des Gebäudes) ist relevant. In solchen Szenarien kann die Höhendifferenz für die meisten besiedelten Gebiete der Erde direkt aus den von der Sensorik erfassten Werten ermittelt werden. Für den Einsatz in extremer Höhe oder wenn die absolute Höhe benötigt wird, kann die Höhe entweder initial von einem GPS-Empfänger beim Betreten eines Gebäudes ermittelt werden oder es wird der vor Ort vorherrschende Luftdruck bezogen auf Meeresspiegelhöhe benötigt. Diesen kann man z. B. über diverse Webservices aus dem Internet abfragen und an das mobile Netzwerk übertragen.

Eine wirkliche dreidimensionale Lokalisierung mit dem Hauptsystem kann dann eine Rolle spielen, wenn eine hoch präzise Lokalisierung benötigt wird und die beteiligten Knoten (mobile Knoten und Anker) auf unterschiedlichen Höhen in einem Stockwerk betrieben werden. Wenn z. B. die Ankerknoten aufgrund von architektonischen Besonderheiten mal an der Zimmerdecke und mal in Bodennähe angebracht werden müssen und auch die Höhe des mobilen Knotens nicht festgelegt werden kann, dann werden

in einem solchen System Lokalisierungsfehler entstehen, die aus der Abweichung vom zweidimensionalen Modell herrühren. Die Größe dieser Fehler hängt dabei von der Entfernung zu den abweichenden Ankerknoten und der absoluten Höhendifferenz ab. Um so größer die Höhendifferenz und umso dichter sich der mobile Knoten am jeweiligen Ankerknoten befindet, desto größer ist der Fehler in der bestimmten Position. Allerdings hat die Annäherung an einen Ankerknoten in einem gleichmäßigen Netzwerk die Entfernung von anderen Ankerknoten zur Folge, so dass dieser Fehler in der Praxis nur eine kleine Rolle spielt.

2.3 Distanzmessung zwischen zwei Knoten in mobilen Netzen

Um die Distanz zwischen zwei Knoten in einem Netzwerk funktechnisch zu bestimmen, gibt es drei grundlegende Methoden:

Messung der Signalstärke. Diesem Verfahren liegt ein Ausbreitungsmodell für das jeweilige Signal unter bestimmten Bedingungen zugrunde. Aufgrund dieses Modells wird aus der gemessenen Signalstärke die Entfernung zur Quelle des Signals geschätzt. Das gebräuchlichste Modell ist die Freifeldausbreitung von Funksignalen und die Verwendung des *Received Signal Strength Indicator* (RSSI) zur Abschätzung der Signalstärke.

Messung von Signallaufzeiten. Auch diesem Verfahren liegt ein Modell zugrunde, welches in der Regel wesentlich einfacher ist als die Modelle zur Distanzmessung über die Eingangssignalstärke. Dafür ist das Messen der Laufzeit eines Signals zwischen Quelle und Ziel in der Regel deutlich aufwändiger. Gebräuchliche Verfahren in mobilen Netzen messen beispielsweise die Laufzeit von Funksignalen, Ultraschallsignalen oder auch von optischen Signalen.

Messung der Signalphase. Bei diesem Verfahren wird die Messung der Phasendifferenz zwischen einem fortwährend ausgesandten Trägersignal mit einer festen Frequenz und dem wieder empfangenen Signal gemessen, um die Distanz zwischen zwei Positionen zu bestimmen. Das empfangene Signal kann entweder reflektiert worden sein (z. B. Laser-Entfernungsmesser) oder wurde erneut von einem anderen Knoten übertragen.

Allen Methoden ist gemein, dass Fehler sowohl durch die eigentliche Messung als auch durch das verwendete Modell entstehen können. So lassen sich *Multipath*-Effekte

nicht perfekt ohne weitere Information aus anderen Quellen in ein Modell integrieren. Gerade im Indoor-Bereich spielen diese Effekte jedoch eine große Rolle und die verschiedenen Systeme unterscheiden sich auch darin, wie mit diesen Effekten umgegangen wird. Im Folgenden beschreiben wir die häufig in mobilen Netzen eingesetzten Verfahren und diskutieren kurz die jeweiligen Vor- und Nachteile.

2.3.1 Entfernungsabschätzung über Signalabschwächung durch die Erhebung von RSSI-Werten

RSSI-basierende Verfahren nutzen zur Entfernungsbestimmung ein Modell, das die Signalabschwächung pro Entfernungseinheit bestimmt und messen für ein Signal mit bekannter Sendestärke die Empfangsfeldstärke (engl.: *Received Signal Strength* (RSS)). Indem die Differenz zwischen Sende- und Empfangsfeldstärke berechnet wird, kann über das Modell der Signalabschwächung die Entfernung zwischen Sender und Empfänger berechnet werden. Dieses Prinzip hat jedoch in der Praxis gravierende Nachteile in der Genauigkeit, da erstens das Modell nur eine stark vereinfachte Ableitung der Realität ist und zweitens die benötigten Parameter sich nicht immer eindeutig bestimmen lassen.

So verfügen so gut wie alle modernen Transceiver über Schaltungen zur adaptiven Sende- und Empfangsleistungsanpassung. Durch diese Anpassungen kann die reale Sendeleistung und die Empfangsfeldstärke nur noch näherungsweise abgeleitet und nicht mehr präzise bestimmt werden. Ein wesentlich größeres Problem stellen jedoch Schwächen im Abschwächungsmodell dar. Die Abschwächung eines Signals in geschlossenen Räumen unterliegt nicht nur dem quadratischen Verlust, der durch die gleichförmige Ausbreitung der Wellenfront herrührt, sondern auch diversen nicht im Modell vorhersagbaren Eigenschaften des umgebenden Raumes [144], den sogenannten *Multipath*-Effekten. Diese verursachen Schwankungen in der Empfangsfeldstärke (Fading). So wird ein Signal beim Treffen auf Materie zu einem gewissen Anteil reflektiert und absorbiert bzw. der verbleibende Anteil durchdringt u. U. die Materie. Durch diese Abschwächungen des Signals an z. B. Türen, Wänden, Möbelstücken oder sich im Gebäude bewegender Personen lässt sich aus dem reinen Verlust an Signalstärke keinerlei Aussage über die zurückgelegte Distanz mehr treffen. Zumal durch Reflexionen die Distanz, die das Signal überwunden hat, nicht dem euklidischen Abstand zwischen Sender und Empfänger entsprechen muss. Neben der Reflexion und

Abschwächung können auch noch Brechungen (Änderung der Ausbreitungsrichtung beim Übergang zwischen verschiedenen Medien), Beugungen (Ablenkung der Wellen an einem Hindernis) und Streuungen (z. B. an kleinen Objekten) der Signale auftreten. All diese Effekte führen zu einer unvorhersagbaren Abschwächung der RSS.

Das häufigste verwendete Modell [156] zur Berechnung des Pfadverlusts (PL) eines Signals über eine bestimmte Distanz ist das sog. *log-distance path loss model* [2]. Das Modell geht von gleichen Antennen für Senden und Empfangen aus und setzt die Kenntnis der Abschwächung (in dBm) eines Signals für eine Referenzentfernung d_0 voraus:

$$PL(d) = P_t(d) - P_r(d) = PL(d_0) + 10 \cdot n \cdot \log_{10}\left(\frac{d}{d_0}\right) + X_\sigma \tag{2.1}$$

Hierbei sind $P_r(d)$ die Empfangsfeldstärke und $P_t(d)$ die Sendeleistung, beide in dBm. Der Parameter n ist der Pfadverlust-Exponent (je größer dieser ist, desto stärker ist der Pfadverlust), $PL(d_0)$ ist der Pfadverlust der Referenzdistanz und X_σ ist eine Gaußsche Zufallsvariable mit einem Erwartungswert von Null und Standardabweichung von σ, die das Fading modelliert. Übliche Werte für die Referenzdistanz d_0 sind z. B. 1 km für große städtische Mobilfunksysteme und 1 m für Indoor-Systeme [2]. Der Pfadverlust-Exponent kann je nach Umgebung gewählt werden, im Freiraum gilt z. B. $n = 2$. Der Parameter d entspricht dann der Entfernung der beiden Knoten und ist die gesuchte Größe. Zur Berechnung der Entfernung muss die Sendeleistung des Knotens vorher bekannt sein oder mit übermittelt werden. Oft ist es jedoch besser die Parameter d_0, n und σ in einer spezifischen Umgebung bei einer bestimmten Frequenz selber zu bestimmen. Dazu werden eine Reihe von Signalstärkemessungen mit bekannten Entfernungen an verschiedenen Positionen zu verschiedenen Zeiten aufgenommen und dann ein Fitting der Daten mit der Methode der kleinsten Quadrate durchgeführt, um die Parameter für die Modellfunktion aus Gleichung (2.1) zu erhalten.

In der Praxis kommen RSSI-basierte Verfahren zur Distanzbestimmung vor allem dort zum Einsatz, wo eine Lokalisierung nur näherungsweise benötigt wird und die Benutzung vorhandener Transceiver vorausgesetzt wird [8]. Ein weiterer Vorteil dieser Methode ist es, dass keinerlei zusätzlicher Netzwerkverkehr benötigt wird, um die Entfernung zwischen zwei kommunizierenden Knoten zu bestimmen. Die Entfernungsmessung fällt also bei solchen Systemen zusätzlich zur Datenkommunikation

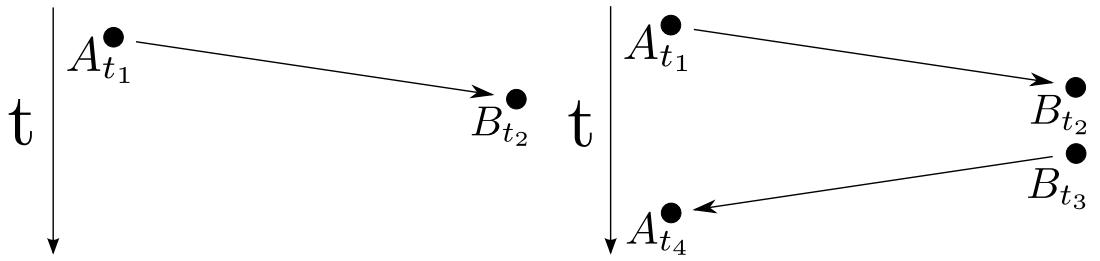
ohne weiteren Aufwand ab. Für die Nutzung in WSNs und in MANETs werden die RSSI-Werte der vorhandenen Transceiver in den meisten Projekten aus Leistungsgründen nicht direkt zur Entfernungsmessung verwendet, sondern, wo möglich, zur Bestimmung der Position mittels Fingerprinting (siehe Kapitel 2.1.5). Wie beschrieben, beziehen diese Verfahren im Gegensatz zu Pfadverlustmodellen die spezifische Signalausbreitung aufgrund von Gebäudeeigenschaften mit ein. Dies geschieht auf Kosten einer zwingend benötigten vorausgehenden Trainingsphase [144].

2.3.2 Entfernungsabschätzung mittels TOF- und RTOF-Verfahren

Die sicherlich naivste Methode eine Entfernung über eine Signalausbreitung zu bestimmen, ist es, die Laufzeit eines Signals als eine Differenz zwischen Start- und Ankunftszeit zu bestimmen (time-of-flight (TOF)) und dann mit der im Medium erwarteten Ausbreitungsgeschwindigkeit zu multiplizieren, üblicherweise der Lichtgeschwindigkeit c. Dies ist in Abbildung 2.4a dargestellt. Die Entfernung r zwischen Knoten A und B ergibt sich damit aus Gleichung (2.2).

$$r = (t_2 - t_1) \cdot c \tag{2.2}$$

Die größten Herausforderungen bei dieser Methode sind die Feststellung der Messpunkte innerhalb der Signallänge und die benötigte Genauigkeit bei der Synchronisation der Uhren. Wenn wir von Funksignalen ausgehen, ist es mit üblicherweise verwendeten Transceivern kaum möglich den genauen Sendezeitpunkt eines Signals zu bestimmen, da die letzte Möglichkeit zum Abgreifen meistens die noch voll-digitale Verarbeitungsstufe vor dem Mixer ist. Das Gleiche gilt für den Empfang eines Signals, meist ist es erst nach Verarbeitung einer Präambel und Identifikation des Signals möglich, einen Interrupt für den Abgriff zu generieren. Spezialhardware kann den durch diese Unschärfen entstehenden *Jitter* minimieren. Das zweite große Problem ist die benötigte Genauigkeit der Zeitsynchronisation. Wenn wir von einem Funksignal ausgehen, ist eine Gangabweichung der Uhren von nur einer Nanosekunde mit einem Messfehler von ca. 30 cm gleichzusetzen. Eine derartig genaue Zeitsynchronisierung ist in üblichen mobilen Netzwerken kaum möglich; zusätzlich können die üblicherweise verwendeten Uhren eine derartige Ganggenauigkeit nur für einen sehr kurzen Zeitraum halten.



(a) Entfernungsbestimmung mittels TOF-Verfahren. Knoten B kann die Entfernung aus $t_2 - t_1$ bestimmen. Da beide Zeitpunkte mit unterschiedlichen Uhren bestimmt wurden, müssen diese synchronisiert sein.

(b) Entfernungsbestimmung mittels RTOF-Verfahren. Knoten A kann die Entfernung aus $\frac{(t_4-t_1)-(t_3-t_2)}{2}$ bestimmen. Da auf Knoten A und B nur die jeweiligen Zeitspannen mit der gleichen Uhr bestimmt werden, ist keine Zeitsynchronisation nötig.

Abbildung 2.4: Entfernungsbestimmung mittels TOF und RTOF

Weiterhin hängt die Genauigkeit der bestimmten Zeitpunkte auch von der Auflösung des Systemtaktes des Transceivers ab, da ein eintreffendes Signal nur bei einer steigenden Taktflanke detektiert werden kann und so die Zeit und folglich auch die Distanz überschätzt wird. Bei einem angenommen Takt von 80 MHz (wie ihn die zur Auswertung benutzte Hardware in Kapitel 7 besitzt) kann sich im schlechtesten Fall eine Abweichung von 12,5 ns zwischen dem Eintreffen des Signals und dem feststellen dieses Zeitpunkts ergeben, falls die Ausbreitungsverzögerung aus einer einzelnen Messung berechnet wird. Dies entspricht einem Distanzfehler von ca. 3,75 m. Um diese Auflösung in einer einzelnen Messung zu erreichen, muss die Signalbandbreite mindestens so groß sein wie die Abtastfrequenz. Höhere Abtastfrequenzen können die Genauigkeit bei mehrfachen Messungen verbessern, bei denen die Distanz durch Durchschnittsbildung bestimmt wird. Eine umfassende Diskussion der Probleme und der Hardware-Anforderungen für ein solches Lokalisierungssystem wird z. B. von McCrady et al. [89] und Lanzisera et al. [74] durchgeführt.

Die populärste Anwendung von TOF-Verfahren sind sicherlich die GNSSs. Hier sind die Uhren in den Satelliten hoch genau und werden ständig synchronisiert. Eine hoch genaue Uhr beim Empfänger wird nicht benötigt, da dessen Drift durch einen redundant empfangenen Satelliten korrigiert werden kann. TOF-Verfahren werden in der Literatur oft auch als *Time of Arrival* (TOA) Verfahren bezeichnet.

Das Problem der Zeitsynchronisation kann man komplett durch die Benutzung von *roundtrip time-of-flight* (RTOF) Verfahren beseitigen. In diesen Verfahren wird beim

Aussenden eines Signals die Zeit beim Sender A genommen und das Signal beim Empfang durch den Empfänger B sofort durch ein weiteres Senden bestätigt. Beim Eintreffen des Bestätigungssignals beim ursprünglichen Sender A nimmt dieser erneut die Zeit. Von dieser Zeitdifferenz muss jetzt noch die Verarbeitungszeit beim Empfänger B abgezogen werden und man erhält die Zeit eines kompletten Umlaufes (engl.: *roundtrip*). Dies ist in Abbildung 2.4b dargestellt. Die Entfernung r zwischen Knoten A und B ergibt sich damit aus Gleichung (2.3).

$$r = \frac{(t_4 - t_1) - (t_3 - t_2)}{2} \cdot c \tag{2.3}$$

Als neue Fehlerquelle kommt hier jedoch ein *Jitter* in der Verarbeitungszeit beim Empfänger B hinzu. Dieser *Jitter* kann sowohl in der softwareseitigen Signalverarbeitung erzeugt werden, als auch im Hardwareteil der Signalverarbeitung. Den softwareseitigen Anteil kann man durch ein Messen dieser Zeit beim Empfänger B und Übertragen dieser Zeit zum Sender A korrigieren. Der hardwareseitige Anteil kann teilweise durch sogenannte Kalibrierungskonstanten korrigiert werden, die im Vorhinein ermittelt werden müssen. Auf dem Markt sind für mobile Netze geeignete Transceiver erhältlich, die all diese Schritte bereits hardwareseitig durchführen [95]. Ein weiteres Problem, welches bei RTOF eine Rolle spielen kann, sind Uhrenfehler (engl.: *clock drift*) der beteiligten Knoten, da die Messung unter Umständen über eine relativ lange Zeitspanne andauern kann. Nehmen wir einen relativen Drift von 40 ppm zwischen den Uhren von A und B im schlechtesten Fall an, dann ergibt sich für eine Zeitspanne $(t_3 - t_2)$ von 100 μs (vgl. Abbildung 2.4b) bereits ein Distanzfehler von 1,2 m. Einige Möglichkeiten mit diesem Problem umzugehen, werden durch A. Bensky [9, S. 99 ff.] diskutiert. Ein Ansatz ist das *back-to-back ranging*, bei dem das eine Mal Knoten A der Initiator der RTOF-Prozedur ist und das andere Mal Knoten B. Die beiden Ergebnisse werden gemittelt und der Fehler dadurch weitestgehend weggekürzt.

Der *Jitter* beim TOF- und RTOF-Verfahren hängt aber auch noch vom Kanalrauschen ab, welches ein zufälliger Fehler ist, dem jedes ankommende Signal unterliegt und das ein akkurates Bestimmen des absoluten Zeitpunktes innerhalb des ankommenden Signals verhindern kann. Dieser *Jitter* σ_τ wird durch Gleichung (2.4) approximiert und kann durch Durchschnittsbildung von N Messungen auf Kosten der Zeit um den Faktor $\sqrt{N}$ gedrückt werden [9, S. 101]. Hierbei ist f_0 die Signalbandbreite und S/N der Signal-Rausch-Abstand.

$$\sigma_\tau = \frac{1}{2 \cdot \pi \cdot f_0 \cdot (S/N)} \tag{2.4}$$

Wesentliche Probleme beider Verfahren sind darüber hinaus die in der Indoorlokalisierung zwangsweise auftretenden *Multipath*-Effekte, die unabhängig vom Messfehler dazu führen, dass anstelle der LOS-Strecke zwischen A und B eine durch Reflexion verlängerte Strecke gemessen wird. Diese Effekte können durch die Verkürzung der Signale und somit der Nutzung einer wesentlich höheren Bandbreite minimiert werden. Der Nutzung solcher *Ultra-wideband* (UWB) Systeme steht jedoch bei dem Einsatz in mobilen Netzen oft eine wesentlich höhere Stromaufnahme und die schlechtere Maximalreichweite entgegen [40, 115].

2.3.3 Entfernungsabschätzung mittels TDOA-Verfahren

Die in der Literatur beschriebenen *Time Difference of Arrival* (TDOA) Verfahren [160, 9, 104] teilen sich in zwei grundlegend verschiedene Ansätze auf:

1. Der Erste misst den Zeitunterschied eines an bekannten Zeitpunkten von bekannten Sendern (Basisstationen mit fester Position) ausgestrahlten Signals bei der Ankunft an einem Empfänger (mobiler Knoten), um daraus Aussagen über die jeweiligen Entfernungen zwischen Sendern und Empfänger abzuleiten. Das Verfahren funktioniert in beide Richtungen, d. h. auch der mobile Knoten kann das Signal aussenden und die Basisstationen sind dann die Empfänger. Die Richtung *Mobil-zu-Basis* wird auch als *Uplink*-Messung oder multilateraler Fall, die Richtung *Basis-zu-Mobil* als *Downlink*-Messung oder unilateraler Fall bezeichnet.
2. Der zweite Ansatz nutzt zwei unterschiedliche Signalquellen mit bekannter Ausbreitungsgeschwindigkeit die gleichzeitig oder zeitversetzt beim Sender ausgestrahlt werden. Der Zeitversatz beim Empfänger kann nun zur Entfernungsberechnung benutzt werden. Dieses Verfahren wird in der Literatur gelegentlich auch den TOF-Verfahren zugerechnet.

TDOA nach Ansatz 1

Bei diesem Verfahren handelt es sich nicht in erster Linie um ein direktes Verfahren zur Bestimmung der Entfernung zwischen zwei Knoten, welche dann zur weiteren Po-

sitionsbestimmung verwendet wird. Hier wird die Laufzeitdifferenz der Signale eines zu lokalisierenden Knotens und zweier fixer Referenzknoten gemessen und für die Bestimmung von Distanzdifferenzen benutzt. Da bei diesem Verfahren ebenfalls Signallaufzeiten gemessen werden, soll es an dieser Stelle im Vergleich zum TOF-Verfahren kurz beschrieben werden und die generelle Eignung für die Indoorlokalisierung betrachtet werden.

Das Verfahren lässt sich prinzipiell in beide Richtungen betreiben. Entweder sendet ein Sender ein Signal aus und mehrere Stationen mit bekannter Position empfangen es und kommunizieren die Empfangszeitpunkte zurück an den Sender oder an eine zentrale Senke, um die Position zu berechnen oder mehrere Sender senden ein Signal, dessen Laufzeitunterschiede ein Empfänger bestimmt. Falls das System nur eine Frequenz benutzt, senden mehrere Sender zu unterschiedlichen Zeitpunkten, damit keine Interferenzen auftreten. Der Empfänger kennt hierbei die Zeitdifferenz zwischen den nicht überlappenden Sendezeitpunkten und die Uhren der Sender sind auch hier synchronisiert. Für mobile Netzwerke bedeutet dieser Ansatz in der Regel, dass die Sender mit Kabeln vernetzt sein müssen, da eine derart genaue Synchronisation aufgrund der verwendeten Transceiver und Uhren über Funk in der Regel nicht möglich sein wird. Für mobile Netzwerke ist bei diesem Ansatz vor allem der ausschließlich durch die Anzahl der Basisstationen definierte Kommunikationsaufwand interessant, der nicht von der Anzahl der Empfänger abhängt. Damit sich die Funksignale auf keinen Fall beim Empfänger auslöschen, sind neben Zeitmultiplexverfahren auch Frequenzmultiplexverfahren möglich, die eine orthogonale Modulation der Signale verwenden, was in der Regel wieder die Beschaffung von aufwändiger Spezialhardware bedeutet. Möchte man darauf verzichten, kann man die Senderichtung umkehren und die Basisstationen empfangen lassen. Dann verliert man allerdings die Unabhängigkeit des Kommunikationsaufwands von der Anzahl der zu lokalisierenden Knoten, unter Beibehaltung des Synchronisierungsproblems.

Das geometrische Modell zur Berechnung der Position eines nicht lokalisierten Knotens $u = (x, y)$ aus den Differenzen der Ankunftszeiten eines bei u ausgesandten Signals, sind die Schnittpunkte von Hyperbeln (im zweidimensionalen Fall) bzw. Hyperboloiden (im dreidimensionalen Fall). Dies ist in Abbildung 2.5 für den zweidimensionalen Fall dargestellt. Sei $A_i = (x_i, y_i)$ die Position von Anker i, t_i der Empfangszeitpunkt des Signals bei Anker i und d_i die Distanz zwischen u und Anker i mit $i = 1, \ldots, N$. Ferner sei Δt_i die TOF zwischen Anker i und u, t_u der Sendezeitpunkt

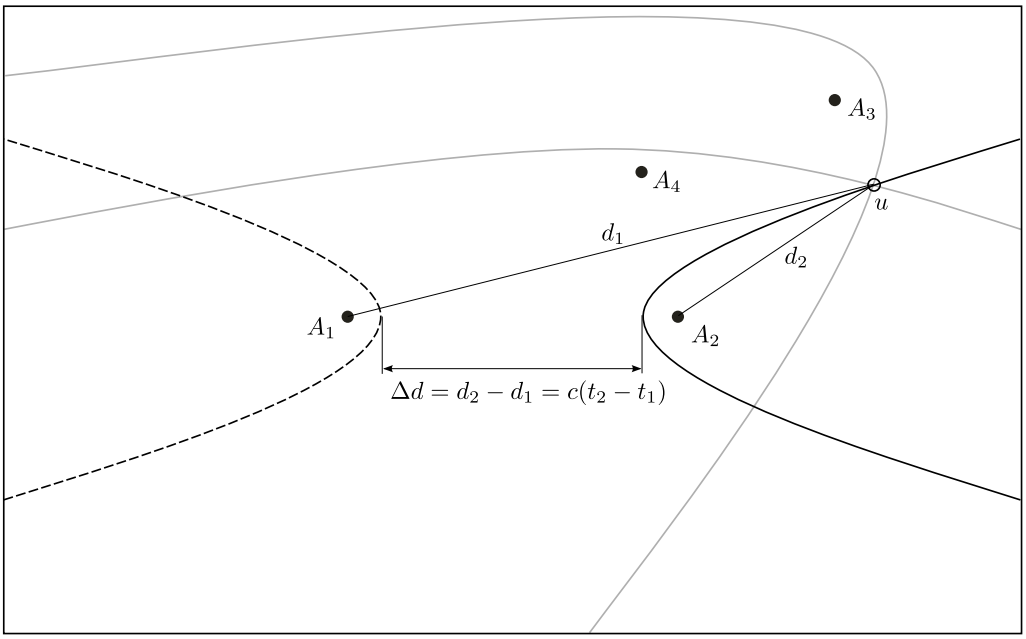


Abbildung 2.5: Prinzip von TDOA nach Ansatz 1

von Knoten u und c die Ausbreitungsgeschwindigkeit der benutzten Signalart. Dann lässt sich die Messung t_i und die Position von u folgendermaßen in Beziehung setzen:

$$\begin{aligned} t_i &= \Delta t_i + t_u \\ &= \frac{d_i}{c} + t_u \\ &= \frac{\sqrt{(x_i - x)^2 + (y_i - y)^2}}{c} + t_u \end{aligned} \tag{2.5}$$

Aus diesen Informationen lässt sich die Position von u noch nicht bestimmen, da wir nur eine Gleichung mit drei Unbekannten (x, y, t_u) haben. Durch Bilden der Differenz zweier Empfangszeitpunkte von verschiedenen Ankern i und j lässt sich jedoch t_u eliminieren:

$$\begin{aligned} t_i - t_j &= \Delta t_i + t_u - (\Delta t_j + t_u) \\ &= \Delta t_i - \Delta t_j \\ &= \frac{d_i}{c} - \frac{d_j}{c} \\ &= \frac{\sqrt{(x_i - x)^2 + (y_i - y)^2}}{c} - \frac{\sqrt{(x_j - x)^2 + (y_j - y)^2}}{c} \end{aligned} \tag{2.6}$$

Die Werte (x, y), die Gleichung (2.6) lösen, bilden eine Hyperbel, wenn sie gezeichnet werden. Dies ist in Abbildung 2.5 zu sehen. Die Differenz Δd der Abstände d_1 und d_2 der Punkte der Hyperbel mit den Ankern A_1 und A_2 ist konstant. Der spezifische Ast der Hyperbel, auf dem sich u befindet, ist derjenige, der am dichtesten bei dem Anker liegt, der das Signal zuerst gehört hat. In Abbildung 2.5 sind beide Hyperbeläste für das Paar $t_{1,2}$ eingezeichnet, wobei der unwesentliche Ast gestrichelt dargestellt ist. Wie man weiterhin sehen kann, liefert ein solcher Hyperbelast trotzdem eine unendliche Menge an möglichen Position für den Knoten u. Durch Bilden weiterer Zeitdifferenzen unabhängiger Messungen lässt sich die Position dann eindeutig bestimmen. Insgesamt gibt es $N-1$ solcher unabhängigen Messungen ($t_{1,2}$, $t_{1,3}$ und $t_{1,4}$ für Abbildung 2.5) und es werden im schlechtesten Fall vier Anker für eine eindeutige Lokalisierung in der Ebene benötigt (z. B. reicht eine weitere Messung mit A_3 nicht aus, um u eindeutig zu bestimmen, da es immer noch zwei Schnittpunkte gibt), wobei drei Anker oft ausreichend sind. In der Realität werden sich die Hyperbeln jedoch nicht in einem eindeutigen Punkt schneiden, da die Messungen zu einem gewissen Teil fehlerbehaftet sind und es sind oft auch mehr Gleichungen vorhanden, als es der Dimensionsgrad der Koordinaten von u vorgibt. In diesem Falle müssen wir die bestmögliche Lösung unter der Annahme finden, dass es keine gibt, die mit allen Messungen übereinstimmt. Dies wird üblicherweise mittels der Methode der kleinsten Fehlerquadrate getan:

$$\hat{u} = \underset{u}{\operatorname{argmin}} \sum_{i=1}^{N} \sum_{j \neq i} \left(c(\tilde{t}_i - \tilde{t}_j) - \sqrt{(x_i - x)^2 + (y_i - y)^2} + \sqrt{(x_j - x)^2 + (y_j - y)^2} \right)^2 \tag{2.7}$$

wobei $\tilde{t}_i$ die gemessenen Empfangszeitpunkte des Signals von u bei den Ankern sind ($\tilde{t}_i = t_i + n_i$, wobei n_i den Messfehler darstellt). Dies stellt ein nichtlineares Optimierungsproblem dar und kann mit den gleichen Methoden gelöst werden, die später noch ausführlich in Abschnitt 3.1.3 für das TOA-Verfahren erläutert werden. Eine Linearisierung des Problems aus Gleichung (2.7) ist ebenfalls im „Handbook of Position Location: Theory, Practise and Advances“ [160, S. 196] beschrieben.

Aus der vorangegangenen Diskussion und einem Vergleich mit TOA sollte klar sein, dass sich die beiden Ansätze deutlich voneinander unterscheiden und TOA zusätzliche Informationen im Vergleich mit TDOA ausnutzt und so mit weniger Informationen präzisere Aussagen liefert. Dies wird an der geometrischen Interpretation besonders deutlich. Ist nur eine Messung mit einem Anker vorhanden, so kann TDOA keine

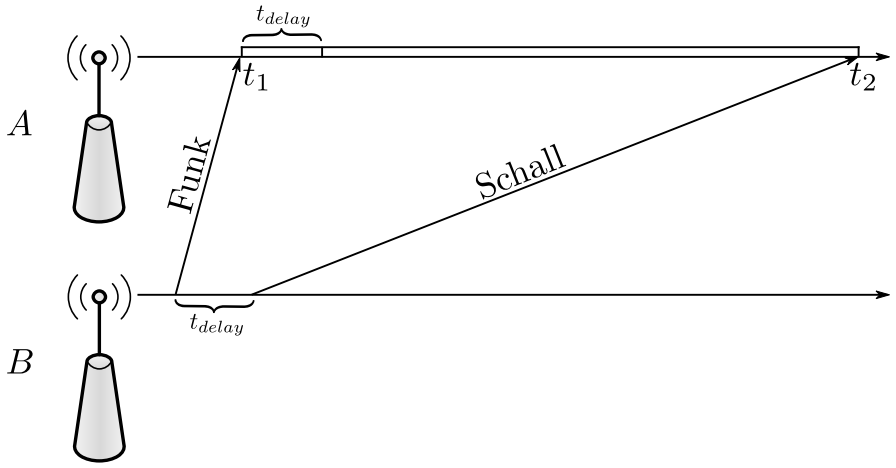


Abbildung 2.6: Prinzip von TDOA nach Ansatz 2

Aussage treffen, TOA die Position auf einen Kreis mit dem Anker als Mittelpunkt eingrenzen. Bei zwei Messungen liefert TDOA immer noch eine Aussage mit unendlich vielen Lösungen, TOA im Normalfall zwei Punkte als Lösung usw. TOA benötigt generell einen Anker weniger, um eine eindeutige Lösung zu finden (im Falle von exakten Messungen sind diese dann auch identisch).

Eine populäre Anwendung von TDOA-Verfahren findet innerhalb moderner GPS-Empfänger statt, die die Geschwindigkeit des Empfängers mittels Zeitversatz der Satellitensignale (Doppler-Effekt) berechnen. Ein weiteres Anwendungsbeispiel des unilateralen TDOA-Verfahrens ist LORAN-C (*Long Range Navigation*) [9, S. 38 ff.], welches vorwiegend zur Navigation in der See- und Luftfahrt verwendet wird bzw. wurde. Jedoch sind diese Ansätze wegen der benötigten Zeitsynchronisation für *Multi-Hop* Netzwerke derzeit nicht realisierbar.

TDOA nach Ansatz 2

Das zweite Verfahren ist eine Verbesserung des TOF-Verfahrens, wobei auf die Synchronisierung der Uhren verzichtet werden kann. Wie in Abbildung 2.6 zu sehen ist, sendet ein Knoten B mit kurzer Verzögerung t_{delay} (die Verzögerung kann auch Null sein, muss aber beiden Knoten vorher bekannt sein) zwei Signale mit unterschiedlicher Ausbreitungsgeschwindigkeit aus (z. B. Funk und Ultraschall). Die Ankunft des schnelleren Signals zum Zeitpunkt t_1 nutzt der Empfänger A, um eine Uhr zu starten, die bei Ankunft des langsameren Signals zum Zeitpunkt t_2 gestoppt wird. Aus

dieser Differenz lässt sich über die Ausbreitungsgeschwindigkeit c_s des langsameren Signals die Entfernung nach Gleichung (2.8) berechnen [104]. Es wird hier also die Tatsache ausgenutzt, dass sich Funkwellen substantiell schneller ausbreiten als Schall in der Luft. Unter LOS-Bedingungen ist diese Methode erstaunlich akkurat und erreicht im Indoor-Bereich bei Entfernungen bis zu 10 m eine Genauigkeit von wenigen Zentimetern.

$$r = (t_2 - t_1 - t_{delay}) \cdot c_s \tag{2.8}$$

Eine Fehlerquelle, die unter Umständen in Gleichung (2.8) eingeführt wird, ist die Temperaturabhängigkeit der Schallgeschwindigkeit in Luft. Bei einem Anstieg der Temperatur von 20°C auf 25°C beträgt der Unterschied beispielsweise 2,93 m/s. Durch Messung der Temperatur und entsprechende Anpassung der Geschwindigkeit in der Formel kann dieses Problem aber behoben werden. Der wesentliche Vorteil gegenüber RTOF-Verfahren ist hier der deutlich geringere Kommunikationsmehraufwand und die gleichzeitige Nutzung eines Lokalisierungssignals für alle benachbarten Knoten. Bei RTOF-Verfahren muss eine Distanzmessung prinzipbedingt immer paarweise stattfinden. Wesentliche Nachteile dieses Ansatzes sind die Notwendigkeit zweier Signalquellen und Empfänger auf den verwendeten Knoten und die oft schlechte Indoor-Eignung langsamer Signale (wie z. B. Schall) aufgrund von starker Dämpfung und Reflexionen. Um Störungen zu mindern ist es dann üblicherweise auch hier nötig, mehrfache Messungen durchzuführen und so Ausreißer zu entfernen und eine Durchschnittsbildung zu ermöglichen.

2.3.4 Entfernungsabschätzung mittels POA-Verfahren

Eine weitere Möglichkeit der Distanzbestimmung zwischen zwei Positionen ist die Messung der Phasendifferenz zwischen einem empfangenen Trägersignal mit fester Amplitude und Frequenz und einem Referenzsignal wie in Abbildung 2.7 dargestellt. Knoten A sendet hierbei dauerhaft (für mehrere Sekunden) ein kontinuierliches Trägersignal (mit fester Frequenz und Amplitude) aus. Knoten B stellt seinen Lokaloszillator auf die Phase des ankommenden Trägersignals ein und leitet es weiter. Knoten A empfängt dieses weitergeleitete Trägersignal und kann aus der Phasenverschiebung im Vergleich zum Referenzsignal die Distanz zwischen A und B aus Gleichung (2.9)

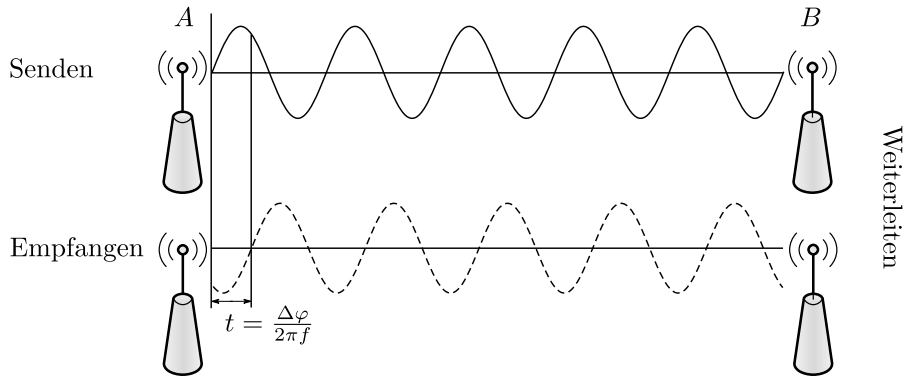


Abbildung 2.7: Prinzip von POA mit einer Trägerfrequenz f

ermitteln. Hierbei ist λ die Wellenlänge des Signals, $\Delta\varphi$ die Phasendifferenz und n eine Ganzzahl. Dieses Verfahren wird in der Literatur auch als *Phase of Arrival* (POA) bezeichnet [100].

$$r = \frac{\lambda}{2} \cdot \left(\frac{\Delta\varphi}{2\pi} + n \right) \tag{2.9}$$

Aus der Formel zur Berechnung der Distanz ist ersichtlich, dass die gemessene Distanz unter Umständen nicht eindeutig ist, da die Phase nur modulo 2π und nicht absolut gemessen werden kann. Somit wäre es notwendig die kompletten Perioden n, die vergangen sind, mitzuzählen, um Distanzen eines Vielfachen der halben Wellenlänge bestimmen zu können. Diese Mehrdeutigkeit kann behoben werden, indem mit zwei geringfügig unterschiedlichen Sendefrequenzen f_1 und f_2 unabhängig voneinander parallel gesendet wird und dann der Phasenunterschied beider empfangenen Signale gemessen wird, wobei jede Phase für sich mit einer Referenz verglichen wird [9, S. 107 ff.]. Für die Phasen der beiden Trägersignale gilt dann unter Benutzung der Frequenz anstatt der Wellenlänge:

$$\Delta\varphi_1 = 2\pi \cdot \left(\frac{2 \cdot r \cdot f_1}{c} - n \right) \tag{2.10}$$

$$\Delta\varphi_2 = 2\pi \cdot \left(\frac{2 \cdot r \cdot f_2}{c} - n \right) \tag{2.11}$$

wobei c die Lichtgeschwindigkeit ist. Durch Bilden von $\Delta\varphi_1 - \Delta\varphi_2$ und Umstellen nach r erhält man:

$$r = \frac{c}{4\pi} \cdot \frac{\Delta\varphi_1 - \Delta\varphi_2}{f_1 - f_2} \tag{2.12}$$

Der maximale Bereich für $\Delta\varphi$ ist 2π, deshalb ist die messbare Distanz r mit dieser Methode eingeschränkt durch die maximale Differenz beider Signalfrequenzen. Der Frequenzunterschied $(f_1 - f_2)$ muss unter Berücksichtigung der erwarteten maximalen Sendereichweite also so gewählt werden, dass die Phasendifferenz den Wert 2π nicht überschreitet. Für einen Frequenzunterschied von 1 MHz beträgt die maximal messbare Entfernung mittels Gleichung (2.12) beispielsweise 150 m [9, S. 108]. Hierbei ist zu beachten, dass kleinere Frequenzunterschiede in größeren Distanzfehlern resultieren. Weiterhin bestimmt die Auflösung des Phasendetektors die Genauigkeit der Distanzmessung. Entwickelte Systeme und deren Analyse für WSNs werden z. B. durch Maróti et al. [86] und Szilvasi et al. [123] beschrieben. Durch Verwendung von *Frequency Hopping Spread Spectrum* (FHSS) Techniken kann die Leistung der Phasendifferenzmethode zusätzlich gesteigert werden, da die Genauigkeit der Messung durch Verwendung mehrerer verschiedener Frequenzen erhöht wird und so auch die Redundanz erhöht wird, sollte z. B. ein bestimmter Kanal durch andere Sender gestört sein.

Ein wesentlicher Nachteil dieser Methode ist es, dass ein LOS-Signalpfad benötigt wird. Die massiven *Multipath*-Bedingungen einer Indoor-Umgebung führen generell zu substanziellen Fehlern in den Phasenmessungen und somit kann diese Methode auch keine akkuraten Distanzmessungen unter diesen Bedingungen liefern [100].

2.4 Fazit

Aufgrund der in diesem Kapitel dargelegten Überlegungen werden wir uns für diese Arbeit auf den allgemeinen Fall einer Indoorlokalisierung festlegen. D. h. wir gehen von einem Einsatz vollständig mobiler Knoten in einem unbekannten Gebäude aus. Als Lokalisierungsschema werden wir uns daher auf die distanzbasierte verteilte Lokalisierung beschränken und andere Verfahren nicht mehr betrachten. Die von uns betrachteten Algorithmen und Verfahren lassen sich sowohl für die infrastrukturbasierte als auch für die infrastrukturlose Lokalisierung verwenden. Für den infrastrukturlosen Fall muss natürlich eine geeignete Rekursion eingesetzt werden. Die von uns betrachteten Algorithmen lassen sich aber natürlich auch bei Beschränkung des allgemeinen

Falles anwenden - also beispielsweise für eine Lokalisierung nicht mobiler Knoten mit vorhandener Infrastruktur in einem Freifeldversuch. Ebenso beschränken wir uns aus oben genannten Gründen auf eine Lokalisierung im zweidimensionalen Raum und schlagen die Lokalisierung in der dritten Dimension mittels eines Hilfssystems vor, sofern diese benötigt wird.

Ein klassischer Einsatzfall für die hier beschriebenen Systeme ist das Nutzen von Technologien zur Indoorlokalisierung in Feuerwehreinsätzen, die wir im Forschungsprojekt „FeuerWhere“ ausführlich untersucht haben. In diesem Szenario trifft die Feuerwehr vor einem brennenden Gebäude ein und einige Feuerwehrleute beginnen sofort in das Gebäude vorzudringen. Ein Feuerwehrmann bleibt am Wagen zurück und kümmert sich um die Atemschutzüberwachung. Diese kann sowohl mittels digitaler Übertragung, als auch einfach zeitbasiert stattfinden. Während dessen überprüft er mittels Sprechfunkverbindung das Wohlergehen seiner Leute im Gebäude. Tritt ein Atemschutznotfall ein, weil der Atemluftvorrat einer Einsatzkraft im Gebäude zur Neige geht oder eine Einsatzkraft nicht mehr mittels Sprechfunk erreichbar ist, wird ein Rettungstrupp in das Gebäude geschickt. Dieser Rettungstrupp hat die Aufgabe zu den betreffenden Personen vorzurücken und ggf. weitere Maßnahmen einzuleiten. Dabei kommt es unter Umständen auf jede Sekunde an. Ein großes Problem für den Rettungstrupp ist das Auffinden der betreffenden Einsatzkräfte. Da in brennenden Gebäuden keinerlei Sicht möglich ist, orientiert sich der Rettungstrupp ausschließlich an dem durch die betreffenden Einsatzkräfte in das Gebäude gelegten Schlauch bis zu ihnen vor. Dieser Schlauch muss allerdings nicht zwangsweise den kürzesten Weg zu den Vermissten darstellen oder dieser Weg kann durch kollabierte Gebäudeteile gänzlich unerreichbar sein.

Um diese Probleme zu minimieren, können Systeme zur Indoorlokalisierung eingesetzt werden. Da eine Präparierung des Gebäudes mit Infrastruktur nicht möglich ist, kommen nur infrastrukturlose autonome Systeme in Frage. In unserem Falle waren das an der Einsatzkraft getragene Sensorknoten, die eine Entfernung zu anderen Knoten messen konnten. An den Einsatzfahrzeugen waren als Anker ebenfalls Knoten, die zusätzlich mit GPS ausgerüstet waren, angebracht. Diese Knoten haben zusätzlich mit Knoten an den Schlauchkupplungen ein Mesh-Netzwerk aufgespannt, in dem eine Lokalisierung der Einsatzkräfte möglich war. Die Positionen der Einsatzkräfte wurden über das selbe Netzwerk zu Computern in den Fahrzeugen übertragen, auf denen eine Visualisierung vorgenommen wurde. So konnten die Rettungstrupps präzise zu den Vermissten dirigiert werden.

Ausgehend von diesem Szenario haben wir uns intensiv mit den verschiedenen Algorithmen und Ansätzen, die eine distanzbasierte Lokalisierung ermöglichen, auseinandergesetzt. Auch wenn das oben beschriebene Szenario das Leitbild unserer Forschungsziele war, so sind die Einsatzmöglichkeiten der hier vorgestellten Erkenntnisse natürlich nicht auf dieses Szenario beschränkt. Dieses Szenario stellt vielmehr einen möglichst allgemeinen Fall für eine präzise infrastrukturlose Indoorlokalisierung dar, von dem sich weitere Spezialfälle, die unter Umständen einfacher zu lösen sind, leicht ableiten lassen.

KAPITEL 3

Distanzbasierte Lokalisierung

Die in diesem Kapitel vorgestellten Lokalisierungsalgorithmen haben alle gemein, dass sie auf Grundlage der Menge aller verfügbaren Ankerknoten und messtechnisch ermittelten oder geschätzten Distanzen eine Position für einen nicht lokalisierten Knoten bestimmen. Dies funktioniert über das Bestimmen der Schnittpunkte von Kreisen, die um die Ankerknoten mit dem Radius der bestimmten Entfernungen konstruiert werden. Meistens kommt für die Lokalisierung eine geschlossene Laterationsformel zur Anwendung, die als Parameter Tupel aus Positionen und Entfernungen bekommt und die eigene Position mathematisch als Lösung eines linearisierten Gleichungssystems bestimmt. Dieses Verfahren liefert bei einem geringen Fehler in der Position der angefragten Knoten und einem geringen Fehler bei der Distanzmessung sehr genaue Ergebnisse. Die Anzahl der Tupel hängt vom Dimensionsgrad der zu bestimmenden Position ab. Für eine zweidimensionale Positionsbestimmung sind für die meisten Algorithmen drei solcher Tupel notwendig.

Die Ankerknoten werden im Rahmen dieser Arbeit oft auch als *Referenzknoten* oder kurz als *Anker* bezeichnet. In Abbildung 2.2 auf Seite 12 sieht man beispielsweise die drei Anker A_1, A_2 und A_3 sowie die zu ihnen gemessenen Distanzen r_1, r_2 und r_3. Weiterhin ist der nicht lokalisierte Knoten u zu sehen, der oft auch als *Tag* oder *mobiler Knoten* bezeichnet wird. Die mittels eines Algorithmus geschätzte Position von u wird mit $\hat{u}$ bezeichnet. Eine Übersicht der wichtigsten Symbole, die in diesem Kapitel benutzt werden, ist in Tabelle 3.1 zu finden.

Sämtliche der hier vorgestellten Algorithmen sind nach der in Abschnitt 2.1 vorgestell-

Tabelle 3.1: Liste der Symbole

Symbol	Bedeutung
u	Position (x, y) des nicht lokalisierten Knotens in kartesischen Koordinaten
$\hat{u}$	Geschätzte Position $(\hat{x}, \hat{y})$ von Knoten u
A_u	$A_u = \{a_i\}$ für $i = 1, 2, \ldots, N$; Menge aller von u erreichbaren Ankerknoten
a_i	Position (x_i, y_i) des i-ten Ankers A_i in kartesischen Koordinaten
r_i	Wert der i-ten Distanzmessung
C_u	$C_u = \{C_i\}$ für $i = 1, 2, \ldots, N$
C_i	Kreis mit Mittelpunkt a_i und Radius r_i
$\overline{D}_i$	Abgeschlossene Kreisscheibe begrenzt durch C_i
A^{-1}	Inverse Matrix von A
A^T	Transponierte Matrix von A
$\mathrm{diag}(x)$	Durch Vektor x gebildete Diagonalmatrix
$\|x\|$	Euklidische Norm/L^2-Norm von x
$\oplus$	Verkettung von zwei Listen

ten Klassifizierung dem entfernungsbasierten Ansatz zuzuordnen. Weiterhin sind alle Algorithmen der *verteilten Lokalisierung* von Knoten zuzurechnen. Hierbei wird die eigene Position auf den unter Umständen ressourcenbeschränkten Knoten selber ausgerechnet und nicht an eine zentrale Komponente im Netzwerk mit leistungsstarker Hardware ausgelagert. Dies zieht ebenfalls den niedrigsten Kommunikationsaufwand nach sich, da nicht sämtliche Informationen durch das Netz an die am Rande gelegenen Infrastrukturknoten weitergeleitet werden müssen und hält somit die ohnehin limitierte Bandbreite vornehmlich für die Lokalisierung selbst offen.

Folglich werden zentralisierte Lokalisierungsansätze wie *Multidimensional Scaling* [120] oder *Convex Position Estimation* [26] nicht betrachtet. Ebenso ausgeschlossen sind Algorithmen aus dem Bereich der *kooperativen Lokalisierung*, bei denen Knoten sich bei der Lokalisierung gegenseitig helfen, sowie Algorithmen die auf Bayes-Filtern (z. B. Kalman- und Partikelfilter) [36] beruhen und eine Historie in den Lokalisierungsprozess einbeziehen.

Der Rest des Kapitels ist wie folgt aufgebaut. Abschnitt 3.1 stellt eine Auswahl der in der Literatur vorhanden Algorithmen für die verteilte, entfernungsbasierte Lokalisie-

rung vor. Darunter sind einige bekannte Standardverfahren sowie einige neuere Algorithmen, die in den letzten Jahren vorgestellt wurden, zu finden. In Abschnitt 3.2 wird mit Optimized Voting-Based Location Estimation (VBLE-O) kurz die von uns erarbeitete Verbesserung des in Abschnitt 3.1 vorgestellten Voting-Based Location Estimation (VBLE) Algorithmus beschrieben, bevor in Abschnitt 3.3 und Abschnitt 3.4 zwei von uns entwickelte Lokalisierungsalgorithmen detailliert vorgestellt werden. Die Auswertung aller vorgestellten Algorithmen erfolgt simulativ sowie anhand von Experimenten in Kapitel 6 und 7. Die Referenzknoten befinden sich dabei zuvörderst in der 1-hop Nachbarschaft des anfragenden Knotens, d. h. die primäre Fehlerquelle für die Algorithmen sind die ungenauen Distanzen und nicht die Position der Nachbarknoten. Dies stellt jedoch keine Einschränkung der Allgemeinheit dar, da sich die Entfernungen zu den Ankern im *Multi-Hop* Fall rekursiv ermitteln ließen.

3.1 Lokalisierungsalgorithmen für Sensornetze

In diesem Abschnitt werden einige distanzbasierte Lokalisierungsalgorithmen näher erläutert. Der Einsatz dieser Algorithmen ist keinesfalls auf den Bereich der Sensornetze beschränkt, jedoch eignen sich die vorgestellten Algorithmen von den Ressourcenanforderungen allesamt besonders für die Anwendung in diesen. Unter den Algorithmen befinden sich einfache Verfahren wie Min-Max sowie eine auf diesem aufbauende Verbesserung namens Extended Min-Max (E-Min-Max). Weiterhin werden bekannte Algorithmen, die die sich aus den Ankerpositionen und Distanzen ergebenden Gleichungen mathematisch zu lösen versuchen, vorgestellt. Zu nennen sind hier Multilateration mittels Linear Least Squares (LLS) und Nonlinear Least Squares (NLLS) sowie Maximum Likelihood Estimation (MLE). Diese Algorithmen lösen das Lokalisierungsproblem entweder mittels einer geschlossenen Formel oder durch iterative Optimierungsverfahren, die einen geeigneten Startpunkt als Ausgangsbasis verwenden.

Andere Algorithmen versuchen das Lokalisierungsproblem geometrisch mittels der Schnittpunkte der Distanzkreise zu lösen und verfolgen dabei unterschiedliche Ziele und Vorgehensweisen. Adapted Multi-Lateration (AML) hat beispielsweise das Ziel, die Geschwindigkeit der Lokalisierung bei gleichbleibender oder sogar leicht verbesserter Genauigkeit gegenüber Multilateration zu erhöhen. Algorithmen wie Bilateration (BL), Iterative Clustering-based Localization Algorithm (ICLA) und Clustering

based Robust Localization (CluRoL) basieren auf dem Clustering der Kreisschnittpunkte, um die relevanten Schnittpunkte für die Lokalisierung zu identifizieren. Diese Algorithmen beanspruchen gleichzeitig eine hohe Robustheit, da die Mehrzahl der Schnittpunkte dazu tendiert, um die echte Position des Tags verstreut zu liegen.

Ein ganz anderes Verfahren benutzt VBLE. Dieses ist auch geometrischer Natur, da die Überschneidung so genannter *Kandidatenringe* mit der in Zellen unterteilten Umgebung um die Anker bestimmt wird, benutzt aber ein Abstimmungsschema, um die Position des Tags zu ermitteln. Da Zellen in der Nähe des mobilen Knotens normalerweise mehr Stimmen erhalten, fällt dieses Verfahren ebenfalls in die Kategorie robuster Algorithmen. Andere Algorithmen versuchen beim Vorliegen redundanter Ankerknoten (es sind mehr Tupel als die minimale Anzahl für die Dimension vorhanden) daraus Vorteile für die Lokalisierung und die Robustheit des Algorithmus gegenüber Ausreißern in den Distanzmessungen zu ziehen. Dies ist besonders im Falle der Lokalisierung innerhalb von Gebäuden wichtig, da non-line-of-sight (NLOS) Fehler dort vermehrt auftreten. Größere Fehler könnten aber theoretisch auch durch bösartige Knoten verursacht werden, die die Distanzmessungen zufällig oder gezielt verfälschen. So berechnen Residual weighting algorithm (Rwgh) und Robust Least-Squared Multilateration (RLSM) sämtliche Kombinationen der vorhandenen Tupel mithilfe der Multilateration, um anschließend eine gewichtete oder gefilterte Linearkombination der Zwischenergebnisse zu bilden. Least Median of Squares (LMS) hingegen nutzt eine nach oben beschränkte Menge an Kombinationen, um eine Auswahl der idealen Anker für die Berechnung mittels LLS zu treffen.

Natürlich können die in diesem Abschnitt präsentierten Algorithmen nur einen gewissen Überblick der in der Literatur vorhandenen Verfahren geben. So gibt es z. B. noch eine Reihe weiterer Ansätze zur Verbesserung der Multilateration hinsichtlich der Robustheit [109, 69]. Zu nennen sind hier unter anderem *Trilaterate on Minima* (ToM), Trilateration mit der *Random Sample Consensus* (RANSAC) Methode sowie *Snap-Inducing Shaped Residuals* (SISR). Diese werden jedoch in dieser Arbeit nicht weiter untersucht.

3.1.1 Der Min-Max Algorithmus

Der Min-Max Algorithmus [143, 75, 73, 117], auch bekannt als Bounding Box Algorithmus, stellt eine sehr einfache Methode zur Lokalisierung von Knoten dar. Er

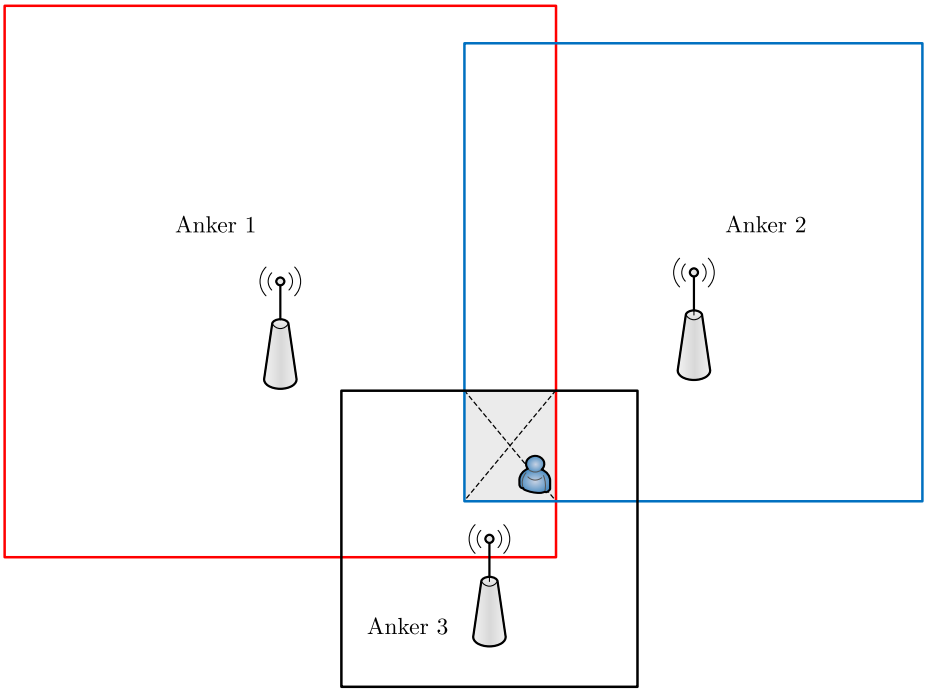


Abbildung 3.1: Konzept des Min-Max Algorithmus

enthält nur wenige arithmetische Operationen, insgesamt liegt die Laufzeitkomplexität in $\Theta(N)$, wobei N die Anzahl der beteiligten Anker ist. Min-Max konstruiert Boxen (Quadrate) um die Position der Anker, wobei sich die Ausdehnung der Boxen aus der gemessenen Distanz zwischen Anker und Knoten ergibt. Zum Beispiel hat die Box von Anker 1 in Abbildung 3.1 folgenden mathematischen Ausdruck:

$$[x_1 - r_1, y_1 - r_1] \times [x_1 + r_1, y_1 + r_1] \tag{3.1}$$

Der Mittelpunkt der Schnittfläche der einzelnen Boxen um die Anker stellt die Positionsabschätzung für den zu lokalisierenden Knoten dar. Zur Bestimmung der Schnittfläche wird das Maximum aller Minimum-Koordinaten sowie das Minimum aller Maximum-Koordinaten berechnet:

$$\left[\max_{i=1}^{N}(x_i - r_i), \max_{i=1}^{N}(y_i - r_i)\right] \times \left[\min_{i=1}^{N}(x_i + r_i), \min_{i=1}^{N}(y_i + r_i)\right] \tag{3.2}$$

Zum Beispiel ist die in Abbildung 3.1 grau dargestellte Fläche die Schnittfläche der drei Ankerknoten.

Die Genauigkeit der Methode hängt stark von der Position der Anker sowie der Position des zu lokalisierenden Knotens zu den Ankern ab. Die Positionsabschätzung ist genauer, falls sich die Anker an gegenüberliegenden Seiten der Schnittfläche befinden und sich der Knoten nicht außerhalb der konvexen Hülle der Anker bewegt. Selbst bei geringen Fehlern auf den gemessenen Distanzen können große Lokalisierungsfehler entstehen, besonders wenn sich der Knoten außerhalb der konvexen Hülle der Anker aufhält. Die Attraktivität des Algorithmus rührt jedoch von seinem äußerst geringen Rechenaufwand sowie Speicherplatzbedarf her, weshalb er besonders in ressourcenbeschränkten Sensornetzwerken präferiert wird.

3.1.2 Extended Min-Max

Der E-Min-Max Algorithmus von Robles et al. [110] benutzt eine Gewichtungsfunktion, um die Genauigkeit des Min-Max Algorithmus zu verbessern. Dieser bestimmt auf die gleiche Weise wie Min-Max die Schnittfläche der Boxen um die Anker, nur kann sich die Position des nicht lokalisierten Knotens an jedem Punkt innerhalb der Schnittfläche befinden und nicht nur in dessen Mittelpunkt. Hierfür weist E-Min-Max jedem Eckpunkt j der rechteckigen Schnittfläche ein Gewicht $W_a(j)$ zu. In der ursprünglichen Arbeit wird E-Min-Max mit vier unterschiedlichen Gewichtsfunktionen (W_1, W_2, W_3, W_4) evaluiert [110]. Wir limitieren unsere Auswertung in dieser Arbeit auf die beiden Gewichte W_2 und W_4, da diese die besten Ergebnisse in Hinsicht auf die Lokalisierungsgenauigkeit erzielt haben:

$$W_2(j) = \frac{1}{\sum_{i=1}^{N}(D_{i,j} - r_i)^2} \tag{3.3}$$

$$W_4(j) = \frac{1}{\sum_{i=1}^{N}\left|D_{i,j}^2 - r_i^2\right|} \tag{3.4}$$

Hierbei ist $D_{i,j}$ der euklidische Abstand zwischen Anker i und Eckpunkt j der rechteckigen Schnittfläche und r_i die gemessene Distanz zu Anker i.

Im Allgemeinen liefert W_4 bessere Ergebnisse innerhalb der konvexen Hülle der Ankerknoten und W_2 die durchschnittlich beste Leistung, sogar außerhalb der konvexen Hülle der Anker. Die finale Position des gesuchten Knotens wird dann durch den gewichteten geometrischen Schwerpunkt, mit den Gewichten und den Koordinaten der Eckpunkte wie in Gleichung (3.5) zu sehen, abgeschätzt.

$$\hat{u} = (\hat{x}, \hat{y}) = \left(\frac{\sum_{j=1}^{4} W_a(j) \cdot x_j}{\sum_{j=1}^{4} W_a(j)}, \frac{\sum_{j=1}^{4} W_a(j) \cdot y_j}{\sum_{j=1}^{4} W_a(j)} \right) \tag{3.5}$$

Im Vergleich zu dem ursprünglichen Min-Max Algorithmus benötigt E-Min-Max zusätzliche Operationen, um die Gewichte für die vier Eckpunkte der rechteckigen Schnittfläche zu berechnen. Besonders E-Min-Max (W_2) beinhaltet Quadratwurzeln, die teuer hinsichtlich des Rechenaufwandes sind, jedoch bleibt die Laufzeitkomplexität von E-Min-Max ebenfalls in $\Theta(N)$.

3.1.3 Multilateration

Multilateration ist im Grunde eine einfache Technik, die spezifische Mathematik zur Umsetzung des Verfahrens unterscheidet sich jedoch stark. Ziel der Multilateration ist es, aus der gegebenen Menge von Ankern A_u und den Distanzmessungen r_i die wahrscheinlichste Position von Knoten u zu bestimmen. Im Allgemeinen werden $n+1$ Referenzknoten für die Lokalisierung im n-dimensionalen Raum benötigt. Aus diesen Informationen lässt sich folgendes Gleichungssystem aufstellen:

$$\begin{aligned} (x - x_1)^2 + (y - y_1)^2 &= r_1^2 \\ (x - x_2)^2 + (y - y_2)^2 &= r_2^2 \\ &\vdots \\ (x - x_N)^2 + (y - y_N)^2 &= r_N^2 \end{aligned} \tag{3.6}$$

Die einzigen Unbekannten in den Gleichungen sind die Koordinaten (x, y) des gesuchten Knotens u. Da die Distanzmessungen in der echten Welt immer fehlerbehaftet sind und meistens auch mehr Messdaten als die minimal benötigte Anzahl zur Verfügung stehen (im 2-D Fall werden minimal drei Messdaten benötigt), kann keine eindeutige Lösung für das überbestimmte Gleichungssystem ermittelt werden. Eine Approximation für dieses Problem wird üblicherweise durch eine *Least Squares* (LS)[1] Methode [12] bestimmt. Hierbei wird typischerweise die Summe der Fehlerquadrate minimiert, welche sich aus der Differenz von den beobachteten Distanzen r_i und den

[1] Carl Friedrich Gauß wird die Entwicklung der Grundlagen des Verfahrens schon 1795 im Alter von 18 Jahren zugeschrieben. Der Franzose Adrien-Marie Legendre war aber der Erste, der die Methode 1805 publizierte.

vorhergesagten Distanzen $\|u - a_i\|$ ergeben:

$$\hat{u} = \underset{u}{\operatorname{argmin}}\ E(u) = \underset{u}{\operatorname{argmin}} \sum_{i=1}^{N} \big(\|u - a_i\| - r_i\big)^2 \tag{3.7}$$

Nonlinear Least Squares

Das Problem aus Gleichung (3.7) stellt ein nichtlineares Optimierungsproblem dar und ist allgemein bekannt als NLLS [96]. Grundlage dieses Verfahrens ist eine lokale Suche nach $\hat{u}$ mittels eines iterativen Algorithmus, der das nicht-lineare Gleichungssystem durch ein lineares approximiert. Hierbei wird ein Startpunkt u_0 benötigt, der in jedem Iterationsschritt sukzessive verbessert wird, bis ein lokales Minimum der Fehlerquadrate gefunden wurde. Liegt u_0 hinreichend nah bei u, so wird erwartet, dass $\hat{u}$ im iterativen Verfahren ermittelt werden kann. Einige bekannte Verfahren für die lokale Suche nach $\hat{u}$ sind das Gauß–Newton-Verfahren [25], die Newton–Raphson-Methode [103] oder der Levenberg–Marquardt-Algorithmus [76, 87, 97]. Ein geeigneter Startpunkt kann aufgrund vorheriger Informationen gewählt werden (z. B. die vorherige Position) oder durch Zuhilfenahme eines suboptimalen Verfahrens bestimmt werden (z. B. durch den Schwerpunkt aller Referenzknoten).

Es muss jedoch angemerkt werden, dass das Finden einer Positionsschätzung $\hat{u}$ mittels NLLS keine einfache Aufgabe ist, da lokale Minima auch abseits des globalen Minimums in der 2-D Fläche von $E(u)$ zu finden sind. Deshalb sollte der Algorithmus mehrfach mit unterschiedlichen Startpunkten ausgeführt werden, damit die Lösung mit hoher Wahrscheinlichkeit am globalen Optimum liegt. Dies macht das Verfahren natürlich insgesamt relativ rechenaufwändig. Unsere Referenzimplementierung des Gauß–Newton-Verfahrens nutzt beispielsweise zwei Startpunkte, die Lösung mittels LLS und den geometrischen Schwerpunkt aller Ankerkoordinaten.

Am Beispiel des Gauß–Newton-Verfahrens soll der Ablauf von NLLS gezeigt werden. Hier wird die Fehlerfunktion $e(u, a_i) = \|u - a_i\| - r_i$ aus Gleichung (3.7) mittels Näherung durch eine Taylorreihe erster Ordnung an der Stelle u_0 approximiert und stellt nun ein lineares Regressionsproblem dar:

$$
\begin{aligned}
e(u, a_i) &\approx e(u_0, a_i) + \nabla e(u_0, a_i)(u - u_0) \\
&= \nabla e(u_0, a_i)u - (-e(u_0, a_i) + \nabla e(u_0, a_i)u_0) \\
\nabla e(u, a_i) &= \frac{u - a_i}{\|u - a_i\|}
\end{aligned}
$$

Hierbei ist $\nabla e(u, a_i)$ die partielle Ableitung der Fehlerfunktion nach jedem Parameter. Durch Rückeinsetzen in Gleichung (3.7) erhält man

$$
\hat{u} \approx \underset{u}{\operatorname{argmin}} \sum_{i=1}^{N} \big(\nabla e(u_0, a_i)u - (-e(u_0, a_i) + \nabla e(u_0, a_i)u_0)\big)^2
$$

und in Matrix-Notation

$$
\hat{u} \approx \underset{u}{\operatorname{argmin}} \ \|Au - b\|^2 \tag{3.8}
$$

mit

$$
A = \begin{bmatrix} \nabla e(u_0, a_1) \\ \nabla e(u_0, a_2) \\ \vdots \\ \nabla e(u_0, a_N) \end{bmatrix}, \ b = \begin{bmatrix} -e(u_0, a_1) + \nabla e(u_0, a_1)u_0 \\ -e(u_0, a_2) + \nabla e(u_0, a_2)u_0 \\ \vdots \\ -e(u_0, a_N) + \nabla e(u_0, a_N)u_0 \end{bmatrix} \tag{3.9}
$$

Dabei ist A die Jacobi-Matrix des Fehlerfunktionsvektors $e(u_0, a_i)$. Jetzt kann die Position des gesuchten Knotens u mittels folgender iterativer Prozedur (NLLS) näherungsweise ermittelt werden:

1. Setze u_0 als den Startpunkt der Optimierung.
2. Berechne die Matrix A und den Vektor b mittels u_0 und Gleichung (3.9).
3. Berechne $\hat{u}_0 = \underset{u}{\operatorname{argmin}} \ \|Au - b\|^2$ mittels LLS.
4. Falls $E(u_0) - E(\hat{u}_0) < \epsilon$ ist, dann ist $\hat{u}_0$ die Lösung. Andernfalls setze $u_0 = \hat{u}_0$ und beginne erneut bei Schritt 2.

Hierbei ist ϵ eine hinreichend kleine positive Konstante, die als Abbruchkriterium benutzt wird, falls sich der Wert in einem Iterationsschritt nicht mehr bedeutend ändert.

Als weiteres Abbruchkriterium kann auch noch zusätzlich die Anzahl der Iterationen selber genommen werden, um die Laufzeit des Algorithmus fest nach oben zu beschränken. In der Praxis sind nur wenige Iterationsschritte notwendig. So verwendet unsere Implementierung einen Wert von 10. Die Laufzeit von NLLS wird daher im Wesentlichen von den Aufrufen von LLS festgelegt. Dieser hat - wie im folgenden Abschnitt gezeigt wird - eine Laufzeit in $\mathcal{O}(N^3)$. Im weiteren Verlauf bezeichnen wir NLLS mit Gauß–Newton als NLLS_{GN}. Dieser hat gegenüber Newton–Raphson den Vorteil, dass nicht die Hesse-Matrix berechnet werden muss, also die zweite partielle Ableitung von $e(u, a_i)$ und ist damit rechnerisch weniger aufwendig. Gleichermaßen bezeichnen wir die Methode nach Levenberg–Marquardt als NLLS_{LM}, welche unsere zweite Referenzimplementierung ist. Diese ist deutlich robuster als das Gauß–Newton-Verfahren, da sie auch bei schlechten Startbedingungen mit hoher Wahrscheinlichkeit konvergiert, ist dafür aber auch etwas langsamer [21]. Dafür zeigte sich, dass ein Startpunkt (der geometrische Schwerpunkt der Anker) bei dieser Methode ausreichend ist.

Linear Least Squares

Durch Subtraktion einer beliebigen Gleichung von den übrigen $N-1$ Gleichungen kann das System aus Gleichung (3.6) aber auch linearisiert werden. Durch Subtraktion der letzten Gleichung von den anderen und anschließendem Vereinfachen erhält man beispielsweise Gleichung (3.10).

$$\begin{aligned}
(x_1 - x_N)x + (y_1 - y_N)y &= \frac{1}{2}(x_1^2 - x_N^2 + y_1^2 - y_N^2 + r_N^2 - r_1^2) \\
(x_2 - x_N)x + (y_2 - y_N)y &= \frac{1}{2}(x_2^2 - x_N^2 + y_2^2 - y_N^2 + r_N^2 - r_2^2) \\
&\vdots \\
(x_{N-1} - x_N)x + (y_{N-1} - y_N)y &= \frac{1}{2}(x_{N-1}^2 - x_N^2 + y_{N-1}^2 - y_N^2 + r_N^2 - r_{N-1}^2)
\end{aligned} \tag{3.10}$$

In Matrix-Notation kann dieses lineare Gleichungssystem dann wie folgt ausgedrückt werden:

$$Au = b \tag{3.11}$$

mit

$$A = \begin{bmatrix} x_1 - x_N & y_1 - y_N \\ x_1 - x_N & y_1 - y_N \\ \vdots & \vdots \\ x_{N-1} - x_N & y_{N-1} - y_N \end{bmatrix}, \; b = \frac{1}{2} \begin{bmatrix} x_1^2 - x_N^2 + y_1^2 - y_N^2 + r_N^2 - r_1^2 \\ x_2^2 - x_N^2 + y_2^2 - y_N^2 + r_N^2 - r_2^2 \\ \vdots \\ x_{N-1}^2 - x_N^2 + y_{N-1}^2 - y_N^2 + r_N^2 - r_{N-1}^2 \end{bmatrix}$$

Für ein überbestimmtes lineares Gleichungssystem der Form $Au = b$ bekommt man eine approximierte Lösung $\hat{u}$, wenn der Vektor der Residuen $\varphi = Au - b$ minimiert wird. Im Sinne der LS-Methode wird die Summe der quadrierten Residuen φ_i minimiert.

$$\hat{u} = \underset{u}{\operatorname{argmin}} \sum_{i=1}^{N-1} \varphi_i^2 = \underset{u}{\operatorname{argmin}} \; \|\varphi\|^2 = \underset{u}{\operatorname{argmin}} \; \|Au - b\|^2 \tag{3.12}$$

Hierbei ist $\|Au - b\|$ anschaulich die Distanz von Vektor b zu dem Vektor $y = Au$ und die LS-Lösung $\hat{u}$ die orthogonale Projektion von b auf den durch die Spalten von A aufgespannten Untervektorraum span(A). Damit ist $\hat{u}$ der dichteste Punkt zu b, der in der Ebene span(A) liegt.

Durch Anwenden von $\|v\|^2 = v^T v$ auf Gleichung (3.12) und weiteres Vereinfachen mit $(M + N)^T = M^T + N^T$, $(RS)^T = S^T R^T$ und $x^T y = y^T x$ erhält man:

$$\begin{aligned} \|Au - b\|^2 &= (Au - b)^T (Au - b) \\ &= (u^T A^T - b^T)(Au - b) \\ &= u^T A^T Au - b^T Au - u^T A^T b + b^T b \\ &= u^T A^T Au - u^T A^T b - u^T A^T b + b^T b \\ &= u^T A^T Au - 2u^T A^T b + b^T b \end{aligned} \tag{3.13}$$

Um Gleichung (3.13) zu minimieren, wird die partielle Ableitung nach u gebildet und gleich Null gesetzt:

$$\begin{aligned} A^TAu + u^T(A^TA) - 2A^Tb &= 0 \\ A^TAu + (A^TA)^Tu - 2A^Tb &= 0 \\ A^TAu + A^TAu - 2A^Tb &= 0 \\ 2A^TAu - 2A^Tb &= 0 \\ A^TAu &= A^Tb \end{aligned} \tag{3.14}$$

Dies ergibt ein lineares System von Normalengleichungen[2] $A^TAu = A^Tb$, welches die Lösung des Minimierungsproblems liefert und numerisch als

$$u = (A^TA)^{-1}A^Tb \tag{3.15}$$

gelöst werden kann, falls die Matrix A vollen Rang besitzt. In der Literatur ist diese Methode als LLS bekannt [96]. Die Zeitkomplexität wird durch die Matrixmultiplikationen bestimmt und ist damit in $\mathcal{O}(N^3)$ (bei einfacher Implementierung). Da die inverse Operation nur zur Auswertung von $(A^TA)^{-1}$ benutzt wird und (A^TA) im 2-D-Fall immer eine 2x2-Matrix ist, beeinflusst dieser Schritt nicht die Gesamtlaufzeit. Hier gilt:

$$(A^TA)^{-1} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{\det(A^TA)} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix} = \frac{1}{ad-bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

Hier sieht man noch einmal direkt, dass Gleichung (3.15) lösbar ist, falls die Determinante der Matrix (A^TA) ungleich Null ist, die Matrix also nicht singulär ist. In allen anderen Fällen kann die Lösung mittels QR-Faktorisierung ermittelt werden, welche ebenfalls eine Laufzeit in $\mathcal{O}(N^3)$ hat.

Umgang mit Ausreißern

Generell ist anzumerken, dass Multilateration dann gut funktioniert, wenn die Fehler auf den Distanzmessungen unabhängig und identisch verteilt sind. Sollte auch nur eine Distanzmessung einen Fehler beinhalten, der sich deutlich von den anderen Mess-

[2] Der Begriff Normalengleichungen soll daran erinnern, dass der Residuenvektor φ eine Normale auf span(A) ist.

werten unterscheidet, dann wird die berechnete Position mittels des Verfahrens womöglich ungenau sein [5]. Verglichen mit NLLS resultiert LLS normalerweise in einem etwas größeren Lokalisierungsfehler[3], ist dafür aber deutlich einfacher und schneller zu berechnen.

Eine Möglichkeit mit größeren Fehlern in den Distanzmessungen umzugehen, ist die Weighted Linear Least Squares (WLLS) [48] Methode. Nach dem Vorbild von Gleichung (3.7) kann eine Weighted Least Squares (WLS) Schätzung der Position des Knotens u mittels Gleichung (3.16) ausgedrückt werden, wobei das Gewicht w_i ein Maß für die Verlässlichkeit der Distanzmessung zu dem i-ten Anker darstellt.

$$\hat{u} = \underset{u}{\operatorname{argmin}} \sum_{i=1}^{N} w_i \big(\|u - a_i\| - r_i \big)^2 \tag{3.16}$$

Durch Linearisierung mit der gleichen Methode wie bei LLS erhalten wir ein Gleichungssystem der Form $WAu = Wb$, wobei $W = \operatorname{diag}(w_1, w_2, \ldots, w_{N-1})$ eine symmetrische Gewichtsmatrix ist. Die Lösung des Gleichungssystems ist durch Gleichung (3.17) gegeben.

$$u = (A^T W^2 A)^{-1} A^T W^2 b \tag{3.17}$$

Güvenç et al. stellen einige heuristische Techniken zur Bestimmung der Gewichte w_i vor [48], die auf einer statistischen Auswertung der Mehrwege-Signalkomponenten beruhen. Eine einfache Methode besteht beispielsweise aus dem Entfernen von allen Ankern mit NLOS-Messungen, indem die Gewichte von Ankern mit NLOS-Messungen auf Null gesetzt werden. Unter der Annahme, dass die NLOS-Messungen perfekt von den LOS-Messungen unterschieden werden können, wird hierbei ebenfalls die Cramér–Rao Lower Bound (CRLB) minimiert. In der Praxis besteht aber immer die Chance zu Fehlidentifizierungen, wodurch auch die Lokalisierungsgenauigkeit sinken kann. Venkatesh und Buehrer haben berichtet [132], dass durch das Ausnutzen der NLOS-Informationen die Genauigkeit der Lokalisierung bei bestimmten geometrischen Konstellationen sogar im Vergleich zur reinen Nutzung der LOS-Messungen verbessert werden konnte. So besteht eine weitere Methode darin, LOS- und NLOS-Messungen fixe Gewichte k_1 und k_2 zuzuordnen ($k_2 < k_1$). Dadurch haben die iden-

[3] Der Fehler bei der Linearisierung des Gleichungssystems durch Subtraktion der letzten Gleichung überträgt sich z. B. auf die anderen Gleichungen. Die Gleichung für die Subtraktion sollte deshalb mit Bedacht ausgewählt werden.

tifizierten NLOS-Messungen beschränkten Einfluss auf die WLS-Schätzung.

3.1.4 Maximum Likelihood Estimation

MLE ist ein parametrisches Schätzverfahren und kann angewandt werden, falls die Wahrscheinlichkeitsdichtefunktion (WDF) des Distanzfehlers, welcher in den gestörten Distanzmessungen enthalten ist, bekannt ist. Sei $p(r \mid u)$ die WDF, die die Wahrscheinlichkeit beschreibt, die Distanzmessungen r zu beobachten, falls sich der nicht lokalisierte Knoten an Position u befindet. Dann berechnet MLE die gesuchte Position als denjenigen Wert u, der die Likelihood-Funktion maximiert (vgl. Gleichung (3.18)) oder anders ausgedrückt, wählt diejenige Position als Schätzung aus, nach dessen Verteilung die Realisierung der beobachteten Distanzen am plausibelsten erscheint.

$$\hat{u}_{ML} = \underset{u}{\operatorname{argmax}}\ p(r \mid u) \tag{3.18}$$

Die Maximierung der Likelihood-Funktion erfolgt im Allgemeinen durch Bilden der ersten Ableitung nach u und Gleichsetzen mit Null. Aus Gründen der einfacheren Berechnung wird jedoch häufig die logarithmierte Likelihood-Funktion (kurz: Log-Likelihood-Funktion) verwendet, dessen Maximum sich an der selben Stelle wie das der nicht-logarithmierten Dichtefunktion befindet (wegen der Monotonie des Logarithmus).

Im Folgenden werden die beiden Algorithmen MLE-$\mathcal{N}$ und MLE-Γ vorgestellt [54], die die Position eines Knotens mittels Normalverteilung $\mathcal{N}(\mu, \sigma^2)$ bzw. Gammaverteilung $\Gamma(\alpha, \beta)$ bestimmen. Die Parameter der Wahrscheinlichkeitsverteilungen müssen anhand von Experimenten bestimmt werden und sind für die verwendete Hardware und Umgebung charakteristisch. Eine Beschreibung der Parameterherleitung wird vor der experimentellen Auswertung der vorgestellten Algorithmen in Abschnitt 7.3 näher erläutert.

Normalverteilung

Ausgehend von einer Normalverteilung mit Erwartungswert μ und Varianz σ^2 ist die Likelihood-Funktion gegeben durch [38]:

$$p(r \mid u) = \frac{1}{(2\pi)^{N/2}\,|C|^{1/2}} \exp\Big(-\frac{1}{2}(r - f(u) - \mu)^T C^{-1}(r - f(u) - \mu)\Big) \tag{3.19}$$

wobei $C = \mathrm{diag}(\sigma_1^2, \sigma_2^2, \ldots, \sigma_N^2)$ die Kovarianzmatrix für r im Falle von unkorrelierten Rauschanteilen ist und $f(u) = \|u - a\|$ den rauschfreien Distanzvektor darstellt. Die ML-Schätzung wird durch Maximierung von Gleichung (3.19) bestimmt. Aus Gründen der einfacheren Berechnung wird das Ergebnis jedoch durch Maximierung der Log-Likelihood-Funktion aus Gleichung (3.20) berechnet.

$$\ln p(r \mid u) = \ln\left(\frac{1}{(2\pi)^{N/2}\,|C|^{1/2}}\right) - \frac{1}{2}(r - f(u) - \mu)^T C^{-1}(r - f(u) - \mu) \tag{3.20}$$

Da der erste Term unabhängig von u ist, kommt eine Maximierung von Gleichung (3.20) einer Minimierung des zweiten Terms gleich. Die ML-Schätzung ist deshalb:

$$\begin{aligned} \hat{u}_{ML} &= \underset{u}{\mathrm{argmin}}\ (r - f(u) - \mu)^T C^{-1}(r - f(u) - \mu) \\ &= \underset{u}{\mathrm{argmin}} \sum_{i=1}^{N} \frac{\Big(r_i - \|u - a_i\| - \mu_i\Big)^2}{\sigma_i^2}. \end{aligned} \tag{3.21}$$

Gängige Verfahren zur Lösung von Gleichung (3.21) beinhalten – vergleichbar mit NLLS – numerische Verfahren wie die Newton–Raphson-Methode, das Gauß–Newton-Verfahren oder das Gradientenverfahren [160]. Für eine Rauschverteilung mit Erwartungswert Null entspricht die Minimierung von Gleichung (3.21) einer gewichteten Version des NLLS Algorithmus, wobei die Gewichte invers proportional zu der Varianz des Rauschens sind und so größere Varianzen in kleineren Gewichten resultieren. Generell stellt MLE mittels Normalverteilung (im Folgenden als MLE-$\mathcal{N}$ bezeichnet) eine Verallgemeinerung der NLLS Methode dar und wird genau dann auf diese reduziert, falls ein Erwartungswert von Null angenommen wird und wenn alle σ_i^2 identisch sind. Die Laufzeit beider Verfahren sind daher identisch.

Gammaverteilung

Bei der Betrachtung von Histogrammen des Distanzfehlers aus der Praxis (vgl. Abschnitt 7.3) zeigt sich, dass die Gammaverteilung eine bessere Wahl für die Berechnung der Likelihood darstellt. Wie das Histogramm des Distanzfehlers ist die Gammaverteilung asymmetrisch, liefert nur positive Ergebnisse und erlaubt dennoch beliebig große Distanzmessfehler mit abnehmender Wahrscheinlichkeit. Die Gammaverteilung ist parametrisiert durch einen Formparameter $\alpha \geq 0$ und den Parameter $\beta \geq 0$, der die Umkehrung des Skalierparameters darstellt. Die WDF ist gegeben durch:

$$\Gamma(\alpha, \beta)(x) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x} \tag{3.22}$$

wobei $\Gamma(\alpha) = \int_0^\infty t^{\alpha-1} e^{-t} dt$ für den Funktionswert der Gammafunktion steht, nach dem die Verteilung auch benannt ist. Da die Gammaverteilung für nicht-positive Werte nicht definiert ist und die Distanzmessung manchmal kürzere Werte als tatsächlich vorhanden liefert, wird ein Offset $\eta \geq 0$ auf alle Messungen addiert. Dieser wird als kleinster beobachteter Messfehler definiert und wird im Zusammenhang mit der Gammaverteilung auch oft als Lokalisierungsparameter μ bezeichnet. Die Likelihood, sich an Position u zu befinden während man die Distanz r_i mit Anker A_i misst, ist dann:

$$p_i(r_i \mid u) = \begin{cases} \Gamma(\alpha, \beta)(r_i + \eta - \|u - a_i\|) & \text{falls } r_i \geq \|u - a_i\| - \eta \\ 0 & \text{andernfalls} \end{cases} . \tag{3.23}$$

Eine Approximation der Likelihood-Funktion $p(r \mid u) = \prod_i p_i(r_i \mid u)$ für die Position u hinsichtlich der Messungen zu berechnen, stellt sich als knifflig heraus. Liegt die Position u zu nahe bei einem Anker, so ist die Likelihood-Funktion für u und für eine schmale Umgebung um u gleich 0, genauso wie der Gradient $\nabla \prod_i p_i(r_i \mid u)$. Folglich wird die Benutzung der Log-Likelihood-Funktion sowie das Gradientenverfahren scheitern [54]. In diesem Fall ist ein iteratives Verfahren, welches nicht die Berechnung des Gradienten von $p(r \mid u)$ voraussetzt, erfolgversprechender. Das nichtlineare Optimierungsproblem wird hier mittels eines direkten Suchverfahrens, das nur mithilfe von Funktionswertevergleichen arbeitet, gelöst. Im konkreten Fall wird ein

Simplex-basiertes, multidirektionales Suchverfahren aus der *Commons Math*[4] Bibliothek [124] benutzt, welches mit verschiedenen Startwerten mehrfach ausgeführt wird, um eine annähernd optimale Lösung zu erhalten. In ausgearteten Fällen wird hier die Lösung die initiale Schätzung sein, während die maximale Likelihood üblicherweise approximiert werden kann. Das Verfahren wird im Folgenden als MLE-Γ bezeichnet. Da die verwendete Bibliothek keine Angaben über die Laufzeit der konkreten Implementierung enthält, können wir für diesen Algorithmus keine Aussagen bezüglich der asymptotischen Laufzeit machen.

3.1.5 Adapted Multi-Lateration

Der AML Algorithmus von Kuruoglu et al. [70] schätzt die Position eines nicht lokalisierten Knotens mittels sich schneidender Kreise. Mithilfe von Distanzmessungen zwischen Knoten u und allen Ankern werden Kreise mit dem Radius des gemessenen Abstands um die Ankerknoten konstruiert. Da sich alle Kreise aufgrund von Messfehlern nicht exakt in einem Punkt schneiden werden, wird die finale Position mittels der Hilfe der Kreisschnittpunkte annäherungsweise bestimmt. Der AML Algorithmus besteht im Wesentlichen aus drei Schritten: *Schnittbildung und Elimination*, *erste Abschätzung* und *Verfeinerung*.

Während der *Schnittbildung und Elimination* werden zufällig zwei Anker A_i und A_j ausgewählt, deren Kreise sich schneiden. Gibt es zwei Schnittpunkte, wird mithilfe eines dritten Ankers A_k der Schnittpunkt eliminiert, der weiter von diesem entfernt ist. Welcher Schnittpunkt weiter entfernt ist, wird mithilfe der in Formel 3.24 definierten Distanz zwischen einem Punkt mit den Koordinaten (x_p, y_p) und einem Kreis mit dem Mittelpunkt (x_k, y_k) bestimmt. Dies ist in Abbildung 3.2 zu sehen. Da $\Delta R_i < \Delta R_j$ ist, wird Schnittpunkt P_2 ausgewählt, welcher auch näher bei u liegt. Falls es nur einen Schnittpunkt gibt, ist keine *Elimination* notwendig und es kann direkt mit der ersten Positionsabschätzung weitergemacht werden.

$$\Delta R = \left| \sqrt{(x_p - x_k)^2 - (y_p - y_k)^2} - r_k \right| \tag{3.24}$$

Bei der *ersten Abschätzung* wird der vorher ausgewählte Schnittpunkt (P_2 im Bei-

[4] *Commons Math* ist eine Mathematikbibliothek der Apache Software Foundation, welche in Java geschrieben ist. Als Suchverfahren wurde die Klasse *MultiDirectionalSimplex* verwendet.

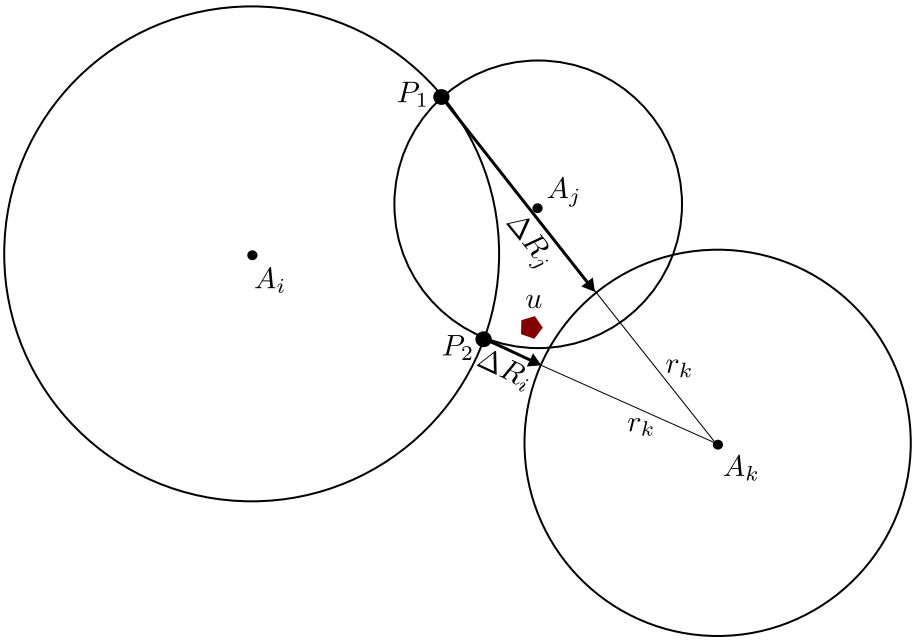


Abbildung 3.2: Elimination eines Kreisschnittpunktes durch einen dritten Anker

spielfall) in die Mitte der Linie, die ihn mit dem nächstgelegenen Punkt auf dem Kreis des aktuell betrachteten Ankers (in diesem Fall A_k) verbindet, verschoben. Dies ist in Abbildung 3.3 zu sehen. Hier wird Schnittpunkt P_2 zu $P_2{}'$ abgeschätzt. Ziel dieses Schrittes ist es, die Fehler in der Distanzmessung zu kompensieren [70].

Um die Position weiter zu verbessern, wird im *Verfeinerungschritt* die gleiche Prozedur wie in der *ersten Abschätzung* mit den verbleibenden Ankern durchgeführt (im Beispielfall kann $P_2{}'$ mithilfe eines vierten Ankers zu einem anderen Punkt $P_2{}''$ verschoben werden, was nicht in den Abbildungen dargestellt ist). Die Zeitkomplexität von AML wird durch den ersten Schritt, der Auswahl zweier sich schneidender Kreise bestimmt und ist damit $\mathcal{O}(N^2)$ [70]. Die durchgeführten Verschiebungsschritte benötigen nur arithmetische Operationen.

Kuruoglu et al. konstatieren eine leicht verbesserte Genauigkeit des AML Algorithmus gegenüber Multilateration mittels LLS unter Betrachtung des Mean Absolute Error (MAE). Entscheidend für die letztendliche Genauigkeit von AML ist aber, dass die Anzahl der verwendeten Anker proportional an die erwartete Anzahl von Distanzmessfehlern angepasst wird. So zeigte sich in den von Kuruoglu et al. durchgeführten Simulationen, dass ein Maximum von vier Ankern bessere Ergebnisse liefert, als alle verfügbaren Anker zu benutzen.

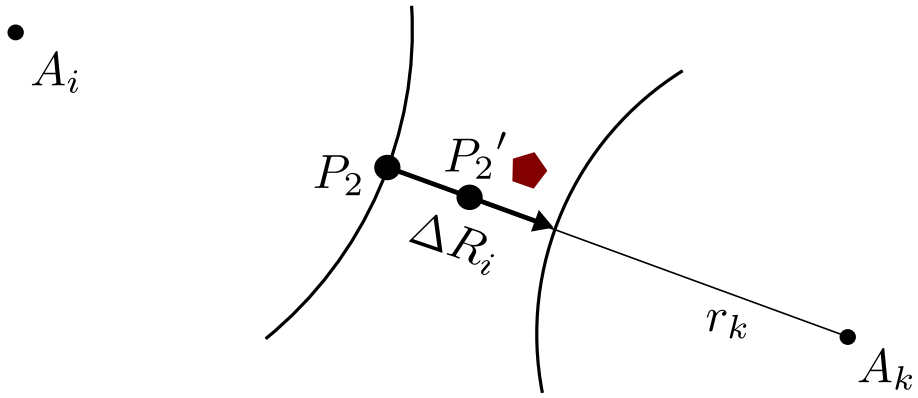


Abbildung 3.3: Erste Positionsabschätzung. Ausgewählter Schnittpunkt P_2 wird in Richtung A_k auf P_2' verschoben. Gleiches Verfahren wird mit den übrigen Ankern im *Verfeinerungsschritt* getan.

3.1.6 Residual weighting algorithm

Ziel des Rwgh Algorithmus von Chen [19] ist es, den Effekt von NLOS-Fehlern abzuschwächen, die z. B. bei NLLS voll einfließen. Dies ist wichtig, denn je mehr NLOS-Fehler auf den Distanzmessungen vorhanden sind, umso mehr wird die geschätzte Position mittels *Least Squares Estimator* (LSE) von der optimalen Position ohne Fehler abweichen. Rwgh besteht aus drei Schritten, wesentliche Voraussetzung ist aber, dass mehr Messdaten (Ankerkoordinaten und zugehörige Distanzmessungen) vorhanden sind, als minimal benötigt werden:

1. Bilde aus N ($N > 3$) verschiedenen Messdaten $M = \sum_{i=3}^{N} \binom{N}{i}$ mögliche Kombinationen. Jede Kombination wird durch eine Ankerindexmenge $\{A_{u,k} \,|\, k = 1, 2, \ldots, M\}$ repräsentiert.
2. Berechne für jede Kombination eine Zwischenposition $\hat{u}_k$ mittels NLLS und einen Restwert $R_{es}(\hat{u}_k, A_{u,k})$.

$$R_{es}(\hat{u}_k, A_{u,k}) = \frac{\sum_{i \in A_{u,k}} \Big[\|\hat{u}_k - a_i\| - r_i \Big]^2}{|A_{u,k}|} \tag{3.25}$$

3. Die finale Position wird durch eine gewichtete Linearkombination der Zwischenpositionen aus Schritt 2 ermittelt. Das Gewicht ist invers proportional zu $R_{es}(\hat{u}_k, A_{u,k})$.

$$\hat{u} = \frac{\sum_{k=1}^{M} \hat{u}_k \Big(R_{es}(\hat{u}_k, A_{u,k}) \Big)^{-1}}{\sum_{k=1}^{M} \Big(R_{es}(\hat{u}_k, A_{u,k}) \Big)^{-1}} \tag{3.26}$$

Gute Zwischenpositionen neigen dazu einen kleineren Restwert zu haben, deshalb wird das Gewicht invers proportional zu den Restwerten gewählt. Hierdurch sollen NLOS-Messungen in der finalen Positionsabschätzung unterdrückt werden. Ein großer Nachteil des Verfahrens ist der extrem hohe Rechenaufwand, da NLLS M-mal in Schritt 2 ausgeführt werden muss. Die Laufzeit kann also durch die Mächtigkeit der Potenzmenge abgeschätzt werden und liegt damit in $\mathcal{O}(N^3 \cdot 2^N)$. Bei einer Ankeranzahl von Sieben müsste NLLS beispielsweise schon 99-mal ausgeführt werden. Je nach Aktualisierungsintervall der eigenen Position kann dies auf ressourcenbeschränkten Sensorknoten schon kritisch sein. Da die Zwischenpositionen bei effizienter Implementierung nicht gespeichert werden müssen, liegt der Speicherplatzbedarf in der Eingabegröße von $\Theta(N)$.

3.1.7 Robust Least-Squared Multilateration

Die RLSM Methode von Bahillo et al. [6] hat einen ähnlichen Ablauf wie der Rwgh Algorithmus, benutzt zur Berechnung von Zwischenpositionen jedoch LLS:

1. Bilde aus N ($N > 3$) verschiedenen Messdaten $M = \sum_{i=3}^{N} \binom{N}{i}$ mögliche Kombinationen. Jede Kombination wird durch eine Ankerindexmenge $\{A_{u,k} \,|\, k = 1, 2, \ldots, M\}$ repräsentiert.
2. Berechne für jede Kombination eine Zwischenposition $\hat{u}_k$ mittels LLS.
3. Bilde Vektor $v = [v_1, v_2, \ldots, v_i, \ldots, v_C]$ von Distanzen zwischen allen Paaren von Zwischenpositionen, wobei $C = \binom{M}{2}$ und $v_i = \|\hat{u}_j - \hat{u}_k\|$ mit $j \neq k$ ist.
4. Berechne $MED(v)$, den Median des Vektors v und entferne alle Zwischenpositionen $\hat{u}_j$, die weiter als der zweifache Median von mehr als der Hälfte der anderen Zwischenpositionen $\hat{u}_k$ ($j \neq k$) entfernt liegen.
5. Die finale Positionsabschätzung ist durch die Position gegeben, die die Entfernung zu allen verbleibenden Zwischenpositionen P minimiert (vgl. Gleichung (3.27)).

$$\hat{u} = \underset{u}{\operatorname{argmin}} \sum_{k=1}^{|P|} \|u - \hat{u}_k\| \tag{3.27}$$

Die Lösung des in Gleichung (3.27) gezeigten Problems ist nicht trivial und ist in der Literatur als Fermat-Weber-Problem [67, 18] bekannt. Ein Verfahren zur Lösung dieses Problems ist der Weiszfeld-Algorithmus [139, 140], der ein iteratives Näherungsverfahren für das Fermat-Weber-Problem darstellt und eine lineare Konvergenz besitzt. Obwohl RLSM für die Bestimmung der Zwischenpositionen LLS benutzt, hat dieser Algorithmus mit steigender Anzahl an Messdaten eine noch höhere Laufzeit als Rwgh. Dies liegt im Wesentlichen an der durchgeführten Medianfilterung aus Schritt 4. Da in Schritt 3 die Anzahl der Einträge des Vektors quadratisch von M abhängt, liegt somit die Laufzeit- und Speicherkomplexität des Algorithmus in $\mathcal{O}(4^N)$.

3.1.8 Least Median of Squares

Multilateration mittels LLS oder NLLS ist nicht beständig gegen Ausreißer, da aufgrund der Summation in der Kostenfunktion von Gleichung (3.7) schon ein einzelner Ausreißer die Abschätzung ruinieren kann. Li et al. schlagen deshalb vor, den Median anstatt die Summe der Restquadrate basierend auf der LMS-Methode, eingeführt durch Rousseeuw und im Detail beschrieben in „Robust Regression and Outlier Detection“ [112], zu minimieren:

$$\hat{u} = \underset{u}{\operatorname{argmin}}\, \operatorname{med}_i \Big[\|u - a_i\| - r_i \Big]^2 \tag{3.28}$$

Hierdurch hat ein einzelner Ausreißer nur geringen Einfluss auf die Kostenfunktion und wird die Positionsabschätzung nicht bedeutend beeinflussen. Die exakte Lösung für dieses nichtlineare Optimierungsproblem zu finden, ist rechnerisch aufwändig, deshalb schlagen die Autoren folgendes Verfahren für die Implementierung eines robusten LMS Algorithmus vor [79]:

1. Setze $n = 4$ als die geeignete Teilmengengröße und $\lambda = 2{,}5$ als Schwellenwert.
2. Setze $M = \begin{cases} 20, & \text{wenn } N > 6 \\ \binom{N}{n}, & \text{andernfalls} \end{cases}$ als die Gesamtanzahl an Teilmengen.
3. Ziehe zufällig M Teilmengen der Größe n aus der Menge der verfügbaren An-

kerknoten A_u. Bestimme eine Position $\hat{u}_j$ für jede Teilmenge $j = 1, 2, \ldots, M$ mittels LLS und berechne für jedes $\hat{u}_j$ den Median der quadrierten Restwerte φ_{ij} zu jedem Ankerknoten $i = 1, 2, \ldots, N$. Hierbei ist $\varphi_{ij} = \|\hat{u}_j - a_i\| - r_i$.

4. Definiere $m = \operatorname{argmin}_j \operatorname{med}_i\{\varphi_{ij}^2\}$, dann ist $\hat{u}_m$ die Positionsabschätzung mit dem kleinsten Median der Mediane aller Teilmengen und $\{\varphi_{im}\}$ sind die dazugehörigen Restwerte.
5. Berechne $s_0 = 1{,}4826 \left(1 + \frac{5}{N-2}\right) \sqrt{\operatorname{med}_i\{\varphi_{im}^2\}}$.
6. Weise jedem Referenzknoten ein Gewicht w_i nach der Gleichung
$$w_i = \begin{cases} 1, & \left|\frac{\varphi_{im}}{s_0}\right| \leq \lambda \\ 0, & \text{andernfalls} \end{cases}$$
zu, wobei φ_{im} der Restwert des i-ten Ankerknotens für die Positionsabschätzung $\hat{u}_m$ ist.
7. Führe WLLS mit den Gewichten $\{w_i\}$ und allen Ankerknoten durch, um die finale Positionsabschätzung $\hat{u}$ zu bestimmen. Dies entspricht der Ausführung von LLS mit denjenigen Ankerknoten, die ein Gewicht von $w_i = 1$ haben.

Die Grundidee von LMS ist, dass mindestens eine Teilmenge unter allen Teilmengen nur geringe oder sogar keine Messfehler enthält. Obwohl kleinere Teilmengen die Wahrscheinlichkeit erhöhen, eine gute Teilmenge zu erhalten, wird $n = 4$ gewählt. Dadurch soll die Wahrscheinlichkeit verringert werden, dass die gezogenen Anker zu dicht beieinander liegen und so keine numerisch stabile Lösung für die Positionsabschätzung zustande kommt. Mit der Wahl von $n = 4$ und $M = 20$ ist das Verfahren mit einer Wahrscheinlichkeit $P \geq 0{,}99$ beständig gegenüber bis zu 30 % Ausreißern, d. h. P beziffert die Chance mindestens eine „gute“ Teilmenge zu erhalten.

Bei Nichtvorhandensein von gaußschem Rauschen toleriert LMS bis zu 50 % Ausreißer von N Distanzmessungen und liefert trotzdem noch die richtige Schätzung [112]. Aus obigem Verfahren ist weiterhin ersichtlich, dass für die robuste Positionsbestimmung eines mobilen Knotens bei Vorhandensein von fehlerbehafteten Messungen genügend Ankerknoten in Reichweite des Knotens sein müssen. In den von Li et al. [79] durchgeführten Simulationen beträgt die Anzahl der Ankerknoten, die von jedem mobilen Knoten gehört werden, beispielsweise 30, was unter realen Bedingungen sicher selten zutreffend ist. Ein weiterer Nachteil des Verfahrens ist der erhöhte Rechenmehraufwand, da LLS M-mal in Schritt 3 und einmal in Schritt 7 ausgeführt werden muss. Da M jedoch auf 20 nach oben beschränkt ist, liegt die Laufzeitkomplexität wie bei LLS in $\mathcal{O}(N^3)$.

3.1.9 Bilateration

Der BL Algorithmus von Cota-Ruiz et al. [21] basiert wie AML auf der Bildung von Kreisschnittpunkten und einer anschließenden Elimination von überflüssigen Schnittpunkten. Als Erstes werden die Schnittpunkte aller Kreispaare (C_i, C_j) mit $i \neq j$ berechnet, die sich aus den Ankerkoordinaten und gemessenen Distanzen ergeben. Von diesen gibt es $Q = \binom{N}{2}$ Stück. Jedes dieser Kreispaare generiert zwei Schnittpunkte g_k und $\bar{g}_k$, deshalb gibt es $2Q$ initiale Positionsabschätzungen, von denen die Hälfte als Spiegelbildlösungen betrachtet werden. Betrachtet man den Sachverhalt geometrisch, dann sollte die echte Position des gesuchten Knotens dort liegen, wo die Schnittpunkte einen Cluster bilden und die Spiegelbildlösungen dazu tendieren, isoliert zu sein. Um sich zwischen zwei Schnittpunkten zu entscheiden, wird die minimale quadrierte euklidische Summe für g_k zu allen anderen möglichen Positionen wie folgt berechnet:

$$\psi = \sum_{\substack{l=1,\\ l\neq k}}^{Q} \min\left(\|g_k - g_l\|^2, \|g_k - \bar{g}_l\|^2\right) \tag{3.29}$$

Gleiches wird ebenfalls für $\bar{g}_k$ durchgeführt:

$$\varphi = \sum_{\substack{l=1,\\ l\neq k}}^{Q} \min\left(\|\bar{g}_k - g_l\|^2, \|\bar{g}_k - \bar{g}_l\|^2\right) \tag{3.30}$$

Schließlich dient der kleinere Wert von ψ und φ dazu, sich zwischen g_k oder $\bar{g}_k$ zu entscheiden. Die gleiche Prozedur wird für alle weiteren der insgesamt Q Lösungspaare durchgeführt, um eine Menge von eindeutigen Positionen zu erhalten. Die finale Positionsabschätzung wird dann durch den geometrischen Schwerpunkt aller Punkte dieser Menge gebildet.

Ein Spezialfall des BL Algorithmus entsteht, wenn es keine zwei Schnittpunkte aus einem der Q Kreispaare gibt. In diesem Fall wird, falls nötig, ein Schnittpunkt zwischen den beiden Kreisen approximiert und es gilt $g_k = \bar{g}_k$. Cota-Ruiz et al. geben eine vergleichbare Genauigkeit des BL Algorithmus gegenüber Multilateration mittels NLLS an. Diese wurde in einer Simulation von 20 verschiedenen Netzwerken mit insgesamt vier Ankern und 96 nicht lokalisierten Knoten ermittelt [21]. Die Laufzeitkomplexität des Algorithmus liegt in $\mathcal{O}(N^4)$ und die Speicherkomplexität in $\Theta(N^2)$,

da alle Kreisschnittpunkte für den Auswahlschritt gespeichert werden müssen [21].

3.1.10 Iterative Clustering-based Localization Algorithm

Der ICLA Algorithmus von Haiyong et al. [49] wandelt die Aufgabe der Lokalisierung eines Knotens zum Clustering von Kreisschnittpunkten um und hat einen ähnlichen Grundaufbau wie der BL Algorithmus:

1. Berechne alle Schnittpunkte aller Kreispaare (C_i, C_j) mit $i \neq j$, die sich aus den Ankerkoordinaten und gemessenen Distanzen ergeben.
2. Wende das Iterative Clustering Model (ICM) an, um die maßgeblichen Schnittpunkte für die Lokalisierung zu erhalten.
3. Berechne die finale Position des gesuchten Knotens als den geometrischen Schwerpunkt aller ursprünglichen Schnittpunkte des größten Clusters, das ICM erzeugt hat.

Auch ICLA geht von der Grundannahme aus, dass sich die Mehrzahl der Schnittpunkte um den nicht lokalisierten Knoten clustert. ICM nimmt dabei die zentrale Stellung des Algorithmus ein. Hier werden alle Schnittpunkte schrittweise in Richtung ihrer jeweiligen Bewegungsrichtung verschoben und verschmolzen, falls eine Kollision von zwei Schnittpunkten auftritt. Die ursprüngliche Position der Knoten wird dabei im neuen Knoten gespeichert, dessen neue Position das Zentrum der Kollisionsfläche ist. Der Kollisionsbereich ist hierbei eine kreisförmige Fläche um den Schnittpunkt mit dem Radius der Schrittweite eines Bewegungsschrittes. Die Schrittweite λ ist ein Parameter des Algorithmus und legt indirekt die Anzahl der Iterationen fest. Größere Schrittweiten lassen den Algorithmus schneller konvergieren, liefern aber schlechtere Ergebnisse [49].

Die Bewegungsrichtung eines Schnittpunktes A wird in jeder Iteration neu festgelegt. Dabei üben alle Knoten, die sich innerhalb eines kreisförmigen Anziehungsbereichs um A befinden, eine Anziehungskraft auf A aus. Die Richtung der Anziehung von einem anderen Punkt B auf Punkt A wird aus Gründen der Einfachheit dabei durch Gleichung (3.31) festgelegt.

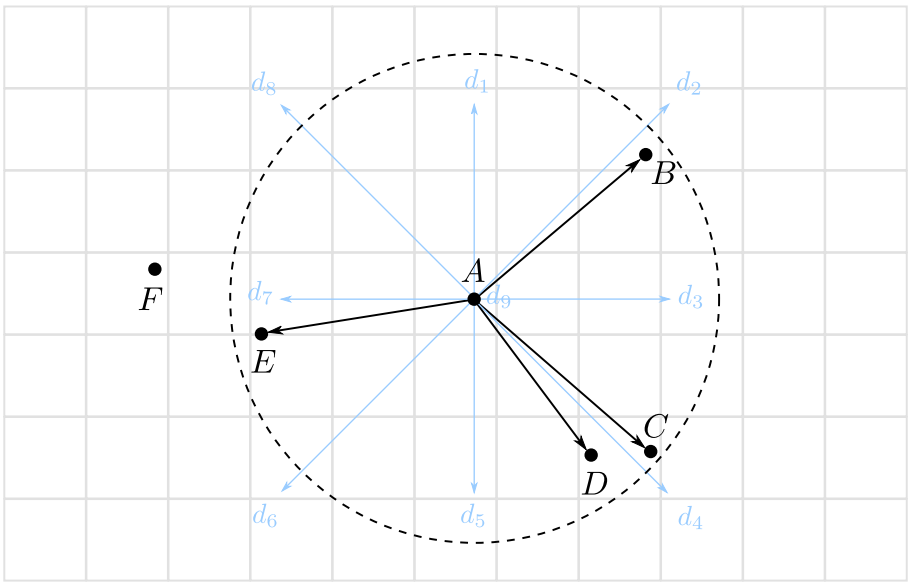


Abbildung 3.4: Bestimmung der Bewegungsrichtung beim ICLA Algorithmus

$$\begin{cases} x = \text{round}\left(\frac{B.x - A.x}{\|B-A\|}\right) \\ y = \text{round}\left(\frac{B.y - A.y}{\|B-A\|}\right) \end{cases} \tag{3.31}$$

Es gibt also insgesamt nur die neun Bewegungs- und Anziehungsrichtungen d_1 bis d_9, welche in Abbildung 3.4 hellblau dargestellt sind. Die Stärke der Anziehung eines Punktes wird durch dessen Gewicht w bestimmt, wobei alle Punkte initial das gleiche Gewicht von Eins haben. Bei Kollisionen wird das Gewicht des neu entstehenden Punktes auf die Gewichtssumme der beteiligten Punkte gesetzt. Die letztendliche Bewegungsrichtung ergibt sich aus der stärksten Einzelrichtung, nachdem die Gewichte für alle Punkte und deren Richtung addiert wurden. Im Fall von Abbildung 3.4 (mit initialem Gewicht aller Punkte) wird für Punkt A die Bewegungsrichtung d_4 bestimmt. Diese hat ein Gesamtgewicht von Zwei, alle anderen Richtungen haben nur ein Gewicht von Eins, denn Punkt F liegt nicht mehr im Anziehungsbereich (gestrichelter Kreis) von Punkt A. Sollten doch einmal zwei Richtungen das gleiche Gesamtgewicht aufweisen, wird die Richtung mit dem kleinsten Index genommen.

Der Radius des Anziehungsbereichs eines Punktes A wird anfangs auf die maximale Distanz zu allen anderen Punkten gesetzt. Nach jeder Bewegungsrunde wird der Radius mittels Gleichung (3.32) neu bestimmt.

$$R = d_{min} + \frac{d_{max} - d_{min}}{\alpha} \ (\alpha > 1) \tag{3.32}$$

Hierbei sind d_{min} und d_{max} die minimale und maximale Distanz zwischen A und anderen Punkten in dessen Anziehungsbereich und α ein weiterer Parameter des Algorithmus. Dieser kontrolliert ebenfalls dessen Konvergenzgeschwindigkeit und die Clustergranularität.

Der Algorithmus bricht ab, falls der Radius des Anziehungsbereichs aller Punkte Null ist. Der Radius wird Null, falls ein Punkt A nur noch einen anderen Punkt B im Anziehungsbereich hat, B aber A nicht in seinem Anziehungsbereich hat. Punkte mit Nullradius bewegen sich auch nicht weiter. Am Ende der Prozedur befinden sich die Schnittpunkte in einigen verschiedenen Clustern, die den übriggebliebenen Punkten entsprechen. Schritt 3 legt dann die finale Position des Algorithmus fest. Die Laufzeitkomplexität des Algorithmus liegt in $\mathcal{O}(N^4)$, die Speicherkomplexität in $\Theta(N^2)$, da alle Kreisschnittpunkte für die Anwendung des ICM gespeichert werden müssen [49].

3.1.11 Clustering based Robust Localization

Der CluRoL Algorithmus von Misra und Xue [92] hat – ähnlich wie Rwgh und RLSM – das Ziel, eine robuste Lokalisierung zu gewährleisten. Große Fehler können entweder durch böswillige Knoten entstehen, die gemeinsam und gezielt Messwerte verfälschen oder durch NLOS-Messungen. Im Gegensatz zu Rwgh, RLSM und LMS bildet CluRoL nicht alle möglichen Kombinationen von Messdaten, um Ausreißer herauszufiltern, sondern bildet wie ICLA sämtliche Schnittpunkte aller Kreispaare:

1. Berechne alle Schnittpunkte aller Kreispaare (C_i, C_j) mit $i \neq j$, die sich aus den Ankerkoordinaten und gemessenen Distanzen ergeben.
2. Berechne die paarweisen Distanzen $d(x, y)$ zwischen allen Schnittpunkten x und y und speichere diese zusammen mit den Punkten in einer Liste D.
3. Sortiere diese Liste D aufsteigend nach den Distanzen.
4. Wende das Clustering-Verfahren *findMaxCluster* auf D an, um das größte Cluster C_{max} zu bestimmen. Der Pseudocode dieses Verfahrens ist in Algorithmus 1 zu finden. Hierbei stellt d_{th} einen Distanzschwellenwert für das Clustering dar [92].

5. Treffe anhand von C_{max} eine Auswahl über die Anker, die für die finale Positionsabschätzung mittels LLS verwendet werden sollen. Diese erfolgt mittels Gleichung (3.33).

Algorithmus 1 *findMaxCluster*

Eingabe: $D = \{D_1, D_2, \ldots, D_Q\}, Q = |D|, d_{th} > 0$;
Ausgabe: C_{max}

$k \leftarrow 1$; ▷ Cluster-Zähler
$C_1 \leftarrow \{x, y\}$; ▷ $(x, y, d(x, y)) \in D_1$
for $i \leftarrow 2$ to Q **do**
  $x, y \in D_i$;
  if $(x \notin C_p) \wedge (y \notin C_p)$, für $p = 1, \ldots, k$ **then**
    $k \leftarrow k + 1$;
    $C_k \leftarrow \{x, y\}$; ▷ x und y sind bisher in keinem Cluster, füge in neues Cluster ein
  else if $(x \in C_p) \wedge (y \notin C_q)$, für $p, q = 1, \ldots, k$ **then** ▷ y bisher in keinem Cluster
    if $d(x, y) \leq d_{th}$ **then**
      $C_p \leftarrow C_p \cup \{y\}$;
    end if
  else if $(x \notin C_p) \wedge (y \in C_q)$, für $p, q = 1, \ldots, k$ **then** ▷ x bisher in keinem Cluster
    if $d(x, y) \leq d_{th}$ **then**
      $C_q \leftarrow C_q \cup \{x\}$;
    end if
  else if $\{(x \in C_p) \wedge (y \in C_q)\} \wedge \{C_p \neq C_q\}$, für $p, q = 1, \ldots, k$ **then** ▷ Prüfen, ob Cluster verschmolzen werden können
    $M_{C_p} \leftarrow \{\sum_{j=1}^{|C_p|} p\} / |C_p|$; ▷ geometrischer Schwerpunkt von C_p
    $M_{C_q} \leftarrow \{\sum_{j=1}^{|C_q|} q\} / |C_q|$; ▷ geometrischer Schwerpunkt von C_q
    if $\|M_{C_p} - M_{C_q}\| \leq d_{th}$ **then**
      Verschmelze C_p und C_q;
      $k \leftarrow k - 1$;
    end if
  end if
end for
$C_{max} \leftarrow \{C_i \mid |C_i| = \max\{|C_i|, i = 1, \ldots, k\}\}$;
return C_{max}

Die Schnittpunkte von C_{max} liegen unter der Annahme, dass die Messfehler die korrekten Messungen nicht übertreffen, um den gesuchten Knoten. Wie in Schritt 5 erwähnt, werden mithilfe der Schnittpunkte in C_{max} und des Parameters δ_{max} die Anker be-

stimmt, welche für den Aufruf von LLS[5] verwendet werden. Hierbei wird für jeden Anker geprüft, ob es einen Punkt x in C_{max} gibt, für den gilt:

$$\|x - a_i\| \leq \left(r_i(1 + \delta_{max})^2\right) \wedge \|x - a_i\| \geq \left(r_i/(1 + \delta_{max})^2\right). \tag{3.33}$$

Ist dies der Fall, dann wird der Anker und die Distanzmessung i für die finale Positionsabschätzung verwendet. Hierbei stellt δ_{max} den maximal erwarteten Distanzmessfehler dar.

Die Laufzeitkomplexität des Algorithmus liegt in $\mathcal{O}(N^4 \log N)$ und der Speicherplatzbedarf hängt im Wesentlichen von der Größe der Liste D ab, liegt also in $\mathcal{O}(N^4)$ [92]. Die Laufzeit ist damit insbesondere im Vergleich zu Rwgh und RLSM deutlich geringer. Diese müssen eine exponentielle Anzahl von Teilmengen der Messdaten aufzählen, um alle Zwischenpositionen zu bestimmen. Bei LMS wird hingegen ja die Anzahl der Teilmengen auf Kosten der Genauigkeit mit 20 künstlich nach oben beschränkt. Misra et al. vergleichen CluRoL in ihrer Arbeit simulativ mit LMS und LLS. LLS schneidet dabei am schlechtesten ab. CluRoL ist in der Lage, die Ergebnisse von LMS noch einmal zu verbessern und den Lokalisierungsfehler (MAE) um ungefähr die Hälfte zu senken.

3.1.12 Voting-Based Location Estimation

Beim VBLE Algorithmus von Liu et al. [82] handelt es sich um ein Verfahren, bei dem über die möglichen Positionen des zu lokalisierenden Knotens „abgestimmt“ wird. Um den Abstimmungsprozess zu vereinfachen, wird das *Zielfeld* (das Feld, in dem sich ein Knoten befinden kann) in Zellen gleicher Größe unterteilt. Dann wird mithilfe der gemessenen Distanzen zu allen Ankerknoten für jede Zelle bestimmt, wie wahrscheinlich es ist, dass der Knoten sich dort befindet. Am Ende werden die Zellen mit der höchsten Stimmenanzahl ausgewählt und deren geometrischer Schwerpunkt als finale Position genutzt.

Als erster Schritt des Algorithmus muss jedoch das *Zielfeld* bestimmt werden. Hierfür

[5] Die Verwendung von LLS anstatt NLLS zur Minimierung der Summe der Fehlerquadrate ergibt sich nur aus dem Kontext, d. h. es wird nicht eindeutig in der Quelle erwähnt. Jedoch deutet der Vergleich mit LMS darauf hin. Auch zeigen unsere Untersuchungen, dass der Algorithmus in Kombination mit NLLS keine Verbesserung darstellt, in manchen Situationen verschlechtern sich eher die Ergebnisse.

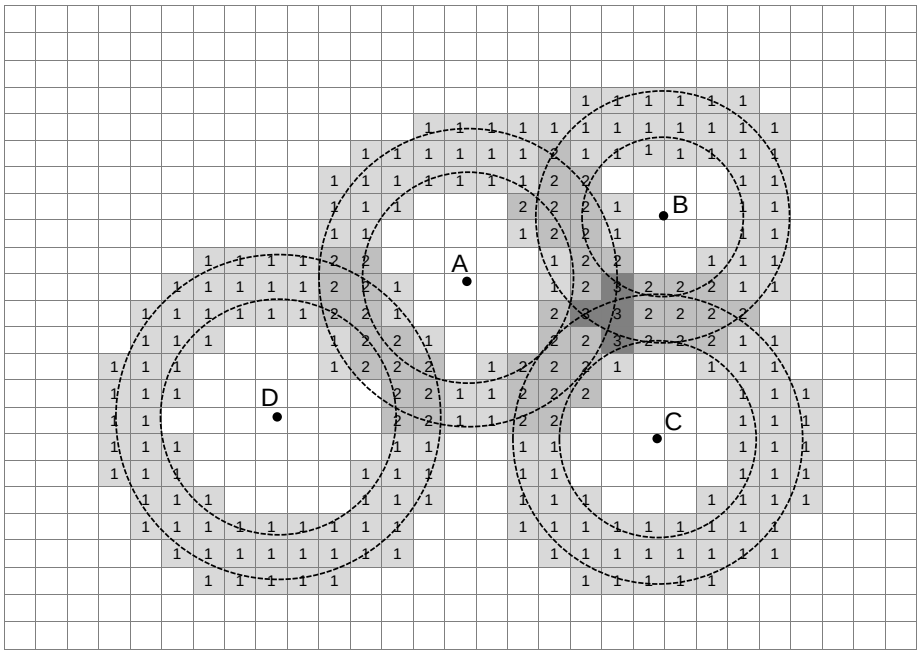


Abbildung 3.5: Konzept des VBLE Algorithmus

wird das minimale Rechteck bestimmt, dass alle verfügbaren Ankerkoordinaten umfasst und schließlich um die maximale Sendereichweite eines Knotens in alle Richtungen erweitert. Damit deckt das Feld alle möglichen Positionen des zu lokalisierenden Knotens ab. Dann wird das Rechteck in M kleine Quadrate (Zellen) der Seitenlänge L unterteilt, wie in Abbildung 3.5 zu sehen ist, wobei für jede Zelle die Stimmenanzahl gespeichert wird. Initial hat jede Zelle den Wert Null.

Betrachtet man alle Messdaten $\langle a_i, r_i \rangle$ bestehend aus Ankerkoordinaten und Distanzen, dann muss sich ein gesuchter Knoten innerhalb eines Ringes befinden, dessen Mittelpunkt a_i ist und welcher den inneren Radius $\max\{r_i - \epsilon, 0\}$ und den äußeren Radius $r_i + \epsilon$ besitzt. Ein solcher Ring wird als *Kandidatenring* bezeichnet, wobei ϵ der maximale Distanzmessfehler ist. Für alle Messdaten $\langle a_i, r_i \rangle$ werden dann alle Zellen ermittelt, die sich mit dem entsprechenden Kandidatenring schneiden und deren Stimmenanzahl um Eins erhöht. Abbildung 3.6 zeigt beispielhaft die Überlappung von Kandidatenring und Zelle. Für den Test auf Überlappung werden die Distanzen d_{min} und d_{max} berechnet und mit dem inneren und äußeren Radius des Kandidatenringes verglichen (siehe Algorithmus 2, Zeile 15). Liu et al. unterteilen die Berechnung von d_{min} und d_{max} in neun unterschiedliche Fälle, je nachdem in welchem der neun Sektoren sich der Anker befindet (bei Abbildung 3.6 befindet sich Anker A

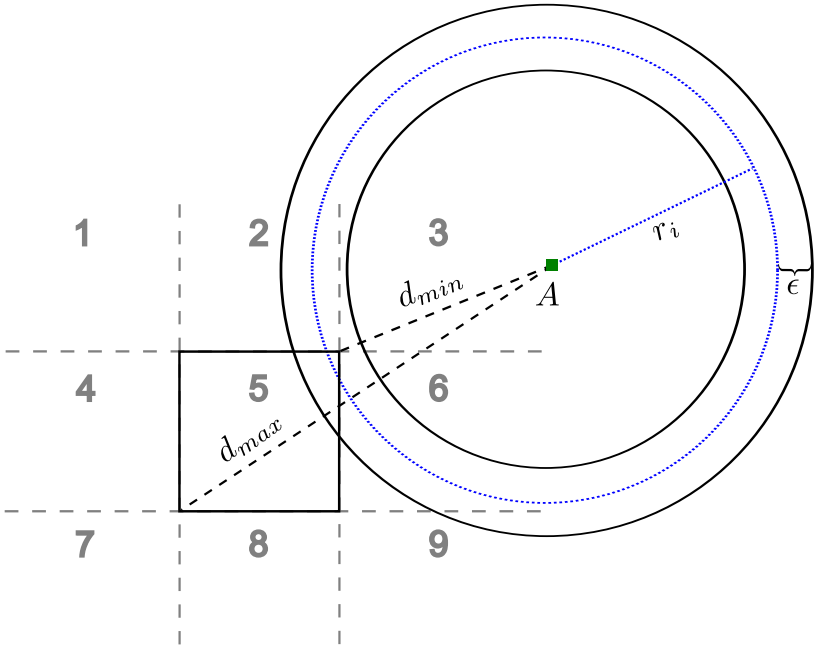


Abbildung 3.6: Überlappung von Kandidatenring und Zelle

in Sektor 3).

Zum Schluss werden alle Zellen mit der höchsten Stimmenanzahl ausgewählt und wie eingangs beschrieben deren geometrischer Schwerpunkt als finale Position des gesuchten Knotens genutzt. In dem Beispiel aus Abbildung 3.5 ergibt sich die Position des Knotens aus den vier Zellen, die alle ein Gewicht von Drei haben (gebildet aus den Kandidatenringen von Anker A, B und C). Anker D mit einer deutlich zu kurzen Messung, wie sie z. B. bei verfälschten Messwerten durch bösartige Knoten (Angreifer) vorkommen kann, hat keinen Einfluss auf die Lokalisierung. Gleiches würde für zu lange Messungen bei NLOS-Bedingungen gelten. Algorithmus 2 zeigt noch einmal den Pseudocode von VBLE. Die Laufzeit von VBLE wird durch die drei Schleifen bestimmt, die jede der M Zellen einmal betrachtet und dann eine Überlappung mit allen Ankern testet. Für die Berechnungen in Zeile 14 werden nur 10 Vergleiche und einige arithmetische Operationen benötigt [82]. Die Laufzeitkomplexität von VBLE liegt damit in $\mathcal{O}(N \cdot M)$. Wie man in Algorithmus 2 sehen kann, müssen nicht sämtliche Stimmen der Zellen zwischengespeichert werden, um die Position des Knotens zu berechnen. Der Speicherplatzbedarf liegt damit in der Eingabegröße von $\Theta(N)$.

Um mit der Ressourcenbeschränktheit von Sensorknoten umzugehen, gibt es eine ite-

Algorithmus 2 Voting-Based Location Estimation

Eingabe: Distanzmessungen r_i, Ankerpositionen $a_i, i = 1, \ldots, N, \forall i, j : a_i \neq a_j \vee i = j$, Seitenlänge L und maximal erwarteter Distanzfehler ϵ;
Ausgabe: die geschätzte Position $\hat{u}$

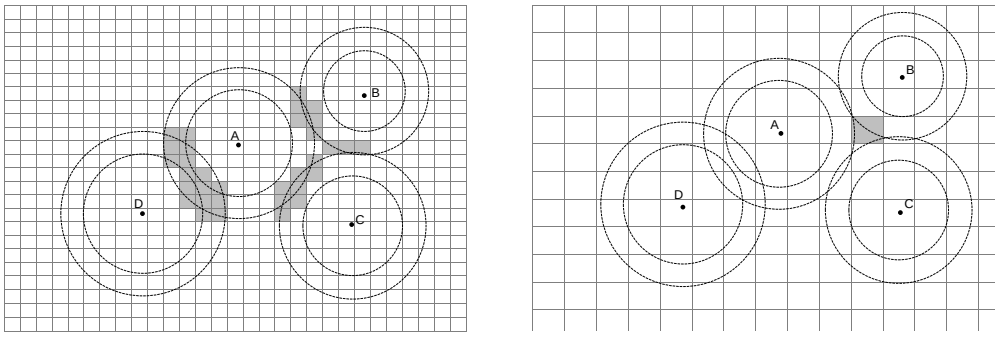
Berechne Zielfeld $ZF = [minX, maxX, minY, maxY]$;
$\hat{x} \leftarrow 0$;
$\hat{y} \leftarrow 0$;
$maxScore \leftarrow 0$;
$maxScoreCount \leftarrow 0$;
$x \leftarrow minX$;
$y \leftarrow minY$;
while $x < maxX$ **do**
    while $y < maxY$ **do**
        $score \leftarrow 0$;
        for $i \leftarrow 1$ to N **do**
            $r_{inner} \leftarrow \max\{r_i - \epsilon, 0\}$;
            $r_{outer} \leftarrow r_i + \epsilon$;
            Berechne d_{min} und d_{max} zwischen aktueller Zelle (x, y) und Anker a_i;
            if $d_{min} \leq r_{outer} \wedge d_{max} \geq r_{inner}$ **then** ▷ Test auf Überlappung von Kandidatenring und Zelle
                $score \leftarrow score + 1$;
            end if
        end for
        if $score \geq maxScore$ **then**
            if $score > maxScore$ **then**
                $maxScore \leftarrow score$;
                $\hat{x} \leftarrow 0$;
                $\hat{y} \leftarrow 0$;
                $maxScoreCount \leftarrow 0$;
            end if
            if $maxScore > 0$ **then**
                $\hat{x} \leftarrow \hat{x} + x + L/2$;
                $\hat{y} \leftarrow \hat{y} + y + L/2$;
                $maxScoreCount \leftarrow maxScoreCount + 1$;
            end if
        end if
        $y \leftarrow y + L$;
    end while
    $x \leftarrow x + L$;
end while
$\hat{u} \leftarrow (\hat{x}/maxScoreCount, \hat{y}/maxScoreCount)$;
return $\hat{u}$

rative Verfeinerung des oben beschriebenen Basisalgorithmus, der den Speicherbedarf reduziert und gleichzeitig eine höhere Genauigkeit der Positionsabschätzung bietet. Die Anzahl der Zellen M ist ein kritischer Parameter, da sie den Speicherbedarf festlegt und gleichzeitig die Genauigkeit der Lokalisierung, denn mit steigendem M werden die Ausmaße jeder einzelnen Zelle kleiner. So wird M in dieser Version des Algorithmus entsprechend des zur Verfügung stehenden Speichers auf dem Sensorknoten gewählt. Nach dem ersten Durchlauf werden dann alle Zellen mit der höchsten Stimmenanzahl ausgewählt und das minimale umschreibende Rechteck um diese Zellen berechnet. In Abbildung 3.5 würde sich ein Rechteck aus sechs Zellen ergeben, das alle Zellen mit Stimmanzahl 3 umschließt. Mit diesem Rechteck wird der Algorithmus erneut aufgerufen, bis eine gewünschte Genauigkeit erreicht wurde oder sich keine Verbesserung mehr ergibt. Darüber hinaus werden NLOS-Messungen sowie verfälschte Messwerte durch Angreifer höchst wahrscheinlich aussortiert, da sich deren Kandidatenringe normalerweise nicht mit denen von normalen Messungen überlappen werden. So wird z. B. Anker D in der zweiten Iteration nicht weiter verwendet, da dieser keine Stimmen für den neu gewählten Bereich in der aktuellen Iteration beiträgt.

VBLE ist fähig, den Einfluss von verfälschten oder grob fehlerhaften Messungen auf die geschätzte Position zu eliminieren, solange es mehr korrekte Messungen als fehlerhafte gibt, also die Zelle mit der wirklichen Position des Knotens mehr Stimmen erhält als beispielsweise die durch einen Angreifer bevorzugte.

3.2 Optimized Voting-Based Location Estimation

Eine Betrachtung der Verteilung des Distanzmessfehlers führte uns zu einer einfachen, aber substanziellen Verbesserungen für den VBLE Algorithmus. Liu et al. [82] nehmen an, dass der Distanzmessfehler gleichverteilt zwischen $-\epsilon$ und ϵ liegt. Die Annahme ist speziell für den Bereich der Indoorlokalisierung jedoch nicht korrekt, da zu kurze Messungen systembedingt nur eher selten vorkommen und im maximalen Fehler beschränkt sind. Dieser maximale Fehler wird bei gängigen Systemen deutlich unter dem Fehler liegen, der bei zu langen Messungen auftritt. Als Beispiel sei hier auf die Analyse des von uns zur Distanzbestimmung verwendeten Systems in Abschnitt 7.2 verwiesen. Deshalb ist es eine einfache aber effektive Maßnahme, den inneren Radius der Kandidatenringe als $\max\{r_i - \epsilon_{max}, 0\}$ und den äußeren Radius als $r_i + \epsilon_{min}$ zu



(a) (b)

Abbildung 3.7: Einfluss der Seitenlänge L auf die Genauigkeit der Lokalisierung

definieren, wobei ϵ_{min} deutlich geringer als ϵ_{max} ist. Ist die Fehlerverteilung bekannt, kann ϵ_{min} z. B. als 3 % Quantil definiert werden. Sollte die Fehlerverteilung im Voraus nicht bekannt sein, kann im Allgemeinen ein Wert von 0 angenommen werden und damit ebenfalls eine gute Verbesserung erzielt werden.

Eine weitere von uns eingeführte Verbesserung von VBLE kann durch eine passende Wahl der initialen Seitenlänge L erreicht werden. Zwar sieht VBLE die Anpassung der Seitenlänge an den zur Verfügung stehenden Speicherplatz vor und damit auch eine feinere Unterteilung der Zellen mit gesteigerter Lokalisierungsgenauigkeit, jedoch ist dieses Vorgehen nicht in allen Fällen optimal. Betrachtet man Abbildung 3.7, so zeigt sich, dass eine anfangs gröbere Unterteilung mit schrittweiser Halbierung von L robuster ist. Im Fall von Abbildung 3.7a werden aufgrund der feinen Unterteilung des Zielfeldes die grau gefärbten Zellen im ersten Schritt ausgewählt, da sie alle ein Gewicht von 2 haben. Auch im weiteren Verlauf der Berechnung des minimal umschreibenden Rechtecks um diese Zellen und der erneuten Anwendung der Prozedur auf das neue Rechteck mit einem kleineren L werden Zellen, die im Schnittbereich von Anker A und Anker D liegen, nicht komplett entfernt. Damit setzt sich eine zu kurz gemessene Distanz (oder in diesem Fall wahrscheinlicher ein Angreifer) leichter durch und kann die finale Position beeinflussen bzw. gezielt verfälschen. Im Fall von Abbildung 3.7b wird bei einer gröberen Unterteilung des Zielfeldes nur eine Zelle mit einem Gewicht von 3 für den nächsten Schritt ausgewählt und Anker D hat nun keinen Einfluss auf das Ergebnis.

Wir setzen die initiale Seitenlänge auf 40 % der Länge der kürzeren Seite des *Zielfeldes* und halbieren dann in jedem Verfeinerungsschritt diese Seitenlänge. Jedoch

wird der 5 %-Wert von VBLE in den Simulationen und Experimenten von uns nicht unterschritten, um eine Vergleichbarkeit zu gewährleisten. Dieser dient somit als Abbruchbedingung.

Eine erste Analyse der beiden Verbesserungen zeigte eine gesteigerte Genauigkeit der Positionierung um ungefähr 27 % bei Verwendung echter Messdaten von einigen Testläufen aus Kapitel 7 und noch ca. 23 %, falls $\epsilon_{min} = 0\,\text{m}$ gewählt wird. Eine ausführlichere Auswertung ist in Abschnitt 7.4 zu finden.

Im Folgenden wird dieser von uns optimierte Algorithmus als VBLE-O bezeichnet. Wie aus den oben genannten Änderungen leicht herzuleiten ist, hat sich die asymptotische Laufzeit im Vergleich zu VBLE nicht geändert, da nur eine konstante Anzahl an Iterationen zusätzlich durchgeführt werden muss, auch wenn sich das Zielfeld im *Worst Case* nicht verkleinern sollte. Im Erwartungswert ist die Laufzeit durch die schrittweise Verkleinerung und ausschließliche Betrachtung der relevanten Bereiche (Zellen mit höchster Stimmenanzahl) deutlich reduziert.

3.3 Der MD-Min-Max Algorithmus

Der Membership Degree Min-Max (MD-Min-Max) Algorithmus [151] wurde von uns entwickelt und basiert, wie der Name vermuten lässt, auf dem Min-Max Algorithmus. Dieser hat jedoch den Nachteil, dass sich die Position des zu lokalisierenden Knotens immer im Mittelpunkt der Bounding Box befindet (vgl. Abbildung 3.1) und damit im Besonderen eine präzise Lokalisierung außerhalb der konvexen Hülle der Anker nicht möglich ist. Ferner ist die durch Min-Max gebildete graue Schnittfläche größer als die grüne Schnittfläche, welche durch die Distanzkreise C_i gebildet wird. Dies ist aus Abbildung 3.8 ersichtlich. Ein potentieller Ansatz zur Verbesserung der Genauigkeit der Lokalisierung besteht deshalb darin, eine sinnvollere Position innerhalb der grauen Schnittfläche zu finden. MD-Min-Max versucht diese Nachteile zu umgehen, indem eine Gewichtung der Eckpunkte der Bounding Box mithilfe einer Zugehörigkeitsfunktion (engl.: Membership Function (MF)) durchgeführt wird, welche auf einer Verteilung des Distanzmessfehlers in einer echten Ausbringung basiert. Hierdurch ist die Position nicht auf den Mittelpunkt der Bounding Box beschränkt, was in einer Steigerung der Lokalisierungsgenauigkeit resultiert. Um den Algorithmus zu nutzen, muss die Fehlerverteilung im Voraus bekannt sein. Eine Anpassung des Algorithmus

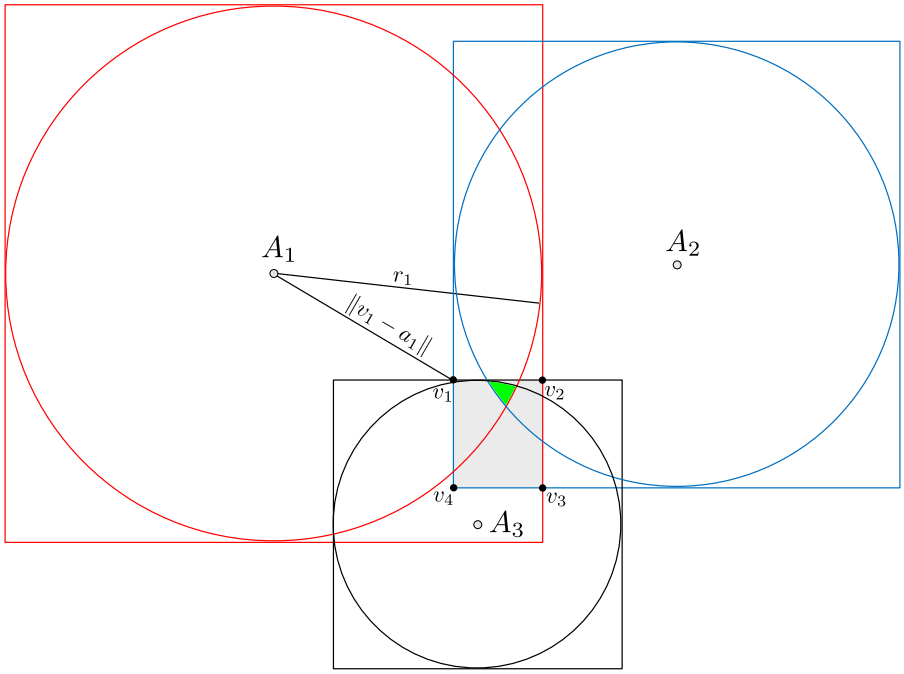


Abbildung 3.8: Geometrische Darstellung der Schnittflächen von Min-Max (grau) und entsprechenden Distanzkreisen (grün) sowie die Beziehung zwischen den Distanzen und den Eckpunkten $\mathbf{V} = \{v_1, v_2, v_3, v_4\}$ von Min-Max.

an eine Fehlerverteilung ist nicht aufwändig und sollte besonders für statische Ausbringungen einfach zu bewerkstelligen sein. Speziell im Bereich der Indoorlokalisierung zeigt unser Algorithmus eine wesentliche Verbesserung des Positionsfehlers auf, da er den Effekt der Mehrwegeausbreitung und der Signalreflexion abschwächt. Selbst im Fall von unbekannten Fehlerverteilungen in dynamischen Umgebungen zeigt unser Algorithmus eine recht gute Leistung mit einer allgemeingültigen Verteilungsfunktion. Desgleichen ist die Laufzeit von MD-Min-Max sowie der Speicherplatzbedarf nur unwesentlich höher als bei Min-Max. Im Folgenden werden die einzelnen Schritte des von uns entwickelten MD-Min-Max Algorithmus im Detail beschrieben.

3.3.1 MD-Min-Max im Detail

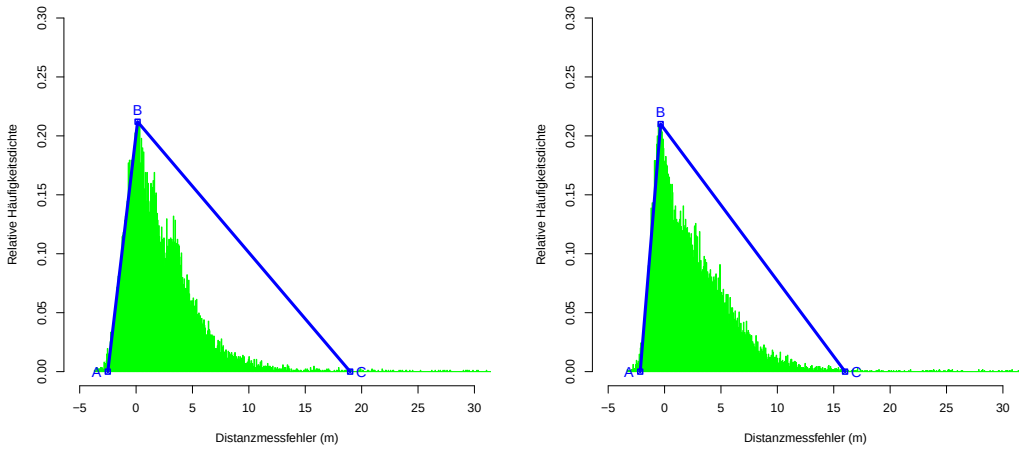
Da die Distanzmessungen r in Indoor-Szenarien im Allgemeinen zweifelhaft und ungenau sind, ist eine Abschätzung der Position auf Basis von r nicht eindeutig möglich. Wahrscheinlichkeitstheorie ist der gebräuchlichste Weg mit der Unbestimmtheit umzugehen, jedoch wird die WDF der Messungen benötigt und zieht zudem aufwendige Berechnungen nach sich. Deshalb verwendet MD-Min-Max Konzepte der Fuzzylogik

[158], die auf unscharfen Mengen (engl.: *fuzzy sets*) und sogenannten Zugehörigkeitsfunktionen basiert. Im Gegensatz zu traditionellen Mengen kann ein Element in einer unscharfen Menge auch *ein bisschen* oder *stark* enthalten sein. Damit lässt sich die Unschärfe von Angaben wie *ein wenig*, *sehr* oder *stark* in mathematischen Modellen erfassen. Eine Zugehörigkeitsfunktion μ weist jedem Element x einer Menge eine reelle Zahl zwischen 0 und 1 zu, die den Grad an Zugehörigkeit (engl.: *membership degree*) $\mu(x)$ von x zur unscharfen Menge angibt. MD-Min-Max benutzt eine solche Zugehörigkeitsfunktion, um die Distanzmessungen in einen Grad an Zugehörigkeit für die einzelnen Eckpunkte $\mathbf{V} = \{v_j \mid j = 1, \ldots, 4\}$ zu überführen, welche durch Min-Max bestimmt wurden. Somit ist im Kontext der Lokalisierung eine Beschreibung der Nähe (zum Beispiel *nah* oder *fern*) einer geschätzten Position $\hat{u}$ zu $\mathbf{V}$ unter Verwendung einer unscharfen Menge und den Distanzmessungen als Indizien möglich. Um den Algorithmus einfach und schnell zu halten, besitzt MD-Min-Max nur eine einzige Regel:

$$\begin{aligned} &\textbf{Falls } \|v_j - a_i\| \ r_i \text{ approximiert,} \\ &\textbf{dann } \text{liegt } \hat{u} \text{ nahe } v_j. \end{aligned} \tag{3.34}$$

Ein numerischer Wert im Intervall $[0, 1]$ steht für den Grad an Übereinstimmung in Gleichung (3.34) und wird durch die MF berechnet. Je höher der Wert ist, desto höher ist das Einvernehmen in Gleichung (3.34). Um dies zu veranschaulichen, kann Abbildung 3.8 betrachtet werden. Gleichung (3.34) sollte in einem höheren *membership degree* für v_1 und v_2 resultieren als für v_3 und v_4.

Üblicherweise werden Zugehörigkeitsfunktionen von Experten festgelegt oder anhand von Statistiken generiert. Wir nehmen an, dass die Fehlerverteilung der Distanzmessungen in gleichen Szenarien ähnlich ist und die MF deshalb auf Grundlage von empirischen Werten aus vorherigen Experimenten in gleichen Szenarien konfiguriert werden kann. Dieses empirische Wissen, welches in der MF enthalten ist, hilft dem Algorithmus in Situationen mit ungenauen Distanzmessungen anpassungsfähig zu sein. Der erste Schritt für die Berechnung der MF ist es, eine große Anzahl an Distanzmessungen und Referenzdistanzen mittels eines Referenzsystems zu ermitteln. In Abbildung 3.9 sind die Ergebnisse von zwei Experimenten, gekennzeichnet mit Testlauf 1 und Testlauf 2, dargestellt. Diese beinhalten jeweils einige zehntausend Messdaten und fanden im selben Bürogebäude mit unterschiedlichen Pfaden bei gleicher Ankerkonstellation statt. Aus diesen Messdaten kann ein Histogramm der absoluten



(a) Testlauf 1 (b) Testlauf 2

Abbildung 3.9: Relative Häufigkeitsdichte des Distanzmessfehlers $(r - \overline{r})$ von zwei exemplarischen Testläufen. Hierbei ist $\overline{r}$ die tatsächliche Distanz und r die gemessene Distanz. Die Dreiecksprofile, festgelegt durch die drei Punkte A, B und C, bilden die MF und werden durch die Parameter $[MF_{\text{low}}, MF_{\text{mode}}, MF_{\text{up}}]$ bestimmt. Der Distanzmessfehler wurde aus Darstellungsgründen bei 30 m abgeschnitten.

Häufigkeiten oder der relativen Häufigkeitsdichte erstellt werden und dazu benutzt werden, um die MF zu bestimmen und zu konfigurieren. Das Histogramm enthält den Distanzmessfehler $(r - \overline{r})$, wobei $\overline{r}$ die Distanz darstellt, die mit dem Referenzsystem ermittelt wurde und r die gemessene Distanz ist.

Abbildung 3.9a sowie Abbildung 3.9b zeigen eine dreieckige MF für Testlauf 1 und Testlauf 2. Diese Dreiecksgestalt ist allerdings keineswegs zwingend und somit nicht die einzige MF, die für unseren MD-Min-Max Algorithmus benutzt werden kann. Ebenso kann eine trapezförmige, rechteckige, auf einer quadratischen Funktion basierende MF oder irgendeine andere theoretische Verteilung der statistischen Ergebnisse verwendet werden. Aufgrund der konzeptionellen Einfachheit und der einfachen Berechenbarkeit haben wir uns für eine dreieckige MF entschieden. Für andere Szenarien sollte die MF entsprechend der Beschaffenheit der dortigen Distanzmessungen gewählt werden, so dass die Häufigkeitsverteilung so gut wie möglich approximiert wird.

Die dreieckige MF wird durch die drei Parameter $[MF_{\text{low}}, MF_{\text{mode}}, MF_{\text{up}}]$ festgelegt, wobei $[A = (MF_{\text{low}}, 0)$, B$=(MF_{\text{mode}}, 1)$, C$=(MF_{\text{up}}, 0)]$ die drei Eckpunkte des Dreiecks bilden (vgl. Abbildung 3.9). Wir berechnen die drei Parameter der MF wie folgt:

1. Ermittle eine große Anzahl von Distanzmessungen r und zugehörigen tatsächlichen Distanzen $\overline{r}$;
2. Berechne den Modus von allen $r - \overline{r}$, bezeichnet als MF_{mode};
3. Berechne MF_{up} als das 99,5 % Quantil und MF_{low} als das 0,5 % Quantil;
4. Konfiguriere die dreieckige MF mit den drei Parametern A, B und C.

Hierbei ist anzumerken, dass die angegebenen Werte für die Quantile empirisch ermittelt wurden und im Vergleich zu anderen Werten (1 % und 3 % Quantil bzw. 97 % und 99 % Quantil) die besten Ergebnisse lieferten. Die Werte sind gerade so gewählt, dass die gröbsten Ausreißer entfernt werden.

Die MF $\mu(x)$ aus Gleichung (3.35) kann dann mit den berechneten Parametern benutzt werden, um den *membership degree* $\mu(\bar{d}_{ij})$ zu bestimmen. Hierbei ist $\bar{d}_{ij} = r_i - d_{ij} = r_i - \|v_j - a_i\|$ für $i = 1, \ldots, N$ und $j = 1, \ldots, 4$ die Differenz von Distanzmessung und Entfernung zwischen Anker und Eckpunkt, drückt also den in Gleichung (3.34) dargestellten Sachverhalt aus. Dieser kann als Eingabe der MF benutzt werden, da diese ebenfalls auf absoluten Messfehlern basiert.

$$\mu(x) = \begin{cases} \frac{x - MF_{\text{up}}}{MF_{\text{mode}} - MF_{\text{up}}} & \text{falls } MF_{\text{mode}} \leq x < MF_{\text{up}} \\ \frac{x - MF_{\text{low}}}{MF_{\text{mode}} - MF_{\text{low}}} & \text{falls } MF_{\text{low}} < x < MF_{\text{mode}} \\ 0 & \text{andernfalls} \end{cases} \tag{3.35}$$

Gleichung (3.35) beschreibt, dass der *membership degree* $\mu_{ij} = \mu(\bar{d}_{ij})$ von 1 nach 0 abnimmt, wenn sich $\bar{d}_{ij}$ von MF_{mode} wegbewegt. Genauer gesagt ist μ_{ij} gleich 0, falls sich $\bar{d}_{ij}$ außerhalb des Intervalls $[MF_{\text{low}}, MF_{\text{up}}]$ befindet. Falls eine Distanzmessung stark fehlerhaft ist, befindet sich $\bar{d}_{ij}$ weit entfernt von MF_{mode} und μ_{ij} ist mit hoher Wahrscheinlichkeit 0. Dies ist immer dann der Fall, wenn eine Distanzmessung als Ausreißer betrachtet wird. Ein großer Fehler unter mehreren Messungen ist eher selten, wie aus den Statistiken in Abbildung 3.9 zu sehen ist und in verschiedenen Publikationen [133, 132, 47] beschrieben wurde. Deshalb kann der *membership degree* solche Distanzmessungen im Durchschnitt abschwächen. Die in Abbildung 3.9a und Abbildung 3.9b gezeigten Daten von Testlauf 1 und Testlauf 2 sollen vorläufig nur zur Veranschaulichung des MD-Min-Max Algorithmus dienen, eine genaue Analyse der Fehlerverteilung während der Experimente wird in Kapitel 7 durchgeführt. Die Berechnung der exakten Werte für die drei Parameter $[MF_{\text{low}}, MF_{\text{mode}}, MF_{\text{up}}]$ für

MD-Min-Max ist in Abschnitt 7.3 im Zusammenhang der Parameterbestimmung aller verteilungsabhängigen Algorithmen zu finden, die Überprüfung der Übertragbarkeit auf andere Szenarien wird während der Auswertung in Abschnitt 7.4 beschrieben.

Da mehrere Distanzmessungen eine Positionsabschätzung gemeinsam bestimmen, wird eine Kombinationsregel benötigt, die mehrere *membership degrees* in ein Gewicht $dw_j, j = 1, \ldots, 4$ für einen Eckpunkt v_j zusammenfasst. Sprachlich kann eine solche Regel folgendermaßen ausgedrückt werden:

$$\begin{aligned} &\textbf{Falls} \text{ alle } \mu_{ij} \text{ vollständig zu } v_j \text{ zustimmen}, i = 1, \ldots, N \\ &\textbf{dann} \text{ unterstützt } dw_j \text{ gänzlich } v_j = \hat{u}. \end{aligned} \tag{3.36}$$

Das Signal-Rausch-Verhältnis aus Gleichung (3.37), definiert als Kehrwert des Variationskoeffizienten von mehreren Zugehörigkeitsgraden, wird als Gewicht für jeden Eckpunkt genommen. Das Signal-Rausch-Verhältnis kann als Maß für die Homogenität der Distanzmessungen betrachtet werden sowie als Grad an Übereinstimmung.

$$dw_j = \begin{cases} \frac{\text{mean}(\bigcup_{i=1}^{N} \mu_{ij})}{\sqrt{\text{var}(\bigcup_{i=1}^{N} \mu_{ij})}} & \text{falls } \text{var}(\bigcup_{i=1}^{N} \mu_{ij}) > 0 \\ \infty & \text{falls } \text{var}(\bigcup_{i=1}^{N} \mu_{ij}) = 0 \end{cases} \tag{3.37}$$

Je größer dw_j ist (je größer die durchschnittliche Übereinstimmung und je kleiner die Varianz der Übereinstimmung sind), desto größer ist demnach auch die Wahrscheinlichkeit, dass sich der nicht lokalisierte Knoten an Eckpunkt v_j oder in dessen Nähe befindet. Mehrere Zugehörigkeitsgrade in dieser Weise zu kombinieren ist nicht nur einfach, sondern spiegelt auch das gemeinsame Urteil aller Distanzen wider. Der Pseudocode von MD-Min-Max ist noch einmal zusammenfassend in Algorithmus 3 beschrieben.

Die finale Positionsabschätzung wird durch den gewichteten geometrischen Schwerpunkt der vier Eckpunkte nach Gleichung (3.38) gebildet. Da Eckpunkte mit höherem Gewicht die Schätzung stärker beeinflussen, ist die so ermittelte Position eine wahrscheinliche Position innerhalb der vier Eckpunkte von Min-Max, auch aufgrund des guten Verständnisses der Distanzfehler, welches aus dem empirischen Wissen abgeleitet wurde.

Algorithmus 3 MD-Min-Max

Eingabe: Distanzmessungen r_i, Ankerpositionen a_i, $i = 1, \ldots, N$ und Eckpunkte $v_j, j = 1, \ldots, 4$ berechnet durch Min-Max;
Ausgabe: die geschätzte Position $\hat{u}$

1: **for** $i \leftarrow 1$ to N **do** ▷ Berechne *membership degrees*
2: **for** $j \leftarrow 1$ to 4 **do**
3: $\bar{d}_{ij} \leftarrow r_i - \|v_j - a_i\|$;
4: Berechne *membership degree* $\mu_{ij} = \mu(\bar{d}_{ij})$ durch Gleichung (3.35);
5: **end for**
6: **end for**
7: **for** $j \leftarrow 1$ to 4 **do** ▷ Kombiniere *membership degrees*
8: Berechne Eckpunktgewicht dw_j durch Gleichung (3.37);
9: **end for**
10: $\hat{u} \leftarrow$ gewichteter geometrischer Schwerpunkt durch Gleichung (3.38);
11: **return** $\hat{u}$

$$\hat{u} = (\hat{x}, \hat{y}) = \sum_{i=1}^{4} \frac{dw_i}{\sum_{j=1}^{4} dw_j} \cdot (v_{xi}, v_{yi}) \tag{3.38}$$

3.3.2 Laufzeitanalyse

Die Laufzeit von MD-Min-Max ist durch $\Theta(N)$ beschränkt.

Beweis. Die asymptotische Laufzeit von MD-Min-Max wird klar durch die Schleife in Zeile 1 dominiert. Die Berechnung der Distanz und des *membership degree* können in konstanter Zeit durchgeführt werden. Der Schleifenkörper wird nur viermal für jeden Anker ausgeführt. Die Kombination der Zugehörigkeitsgrade durch die Berechnung von Mittelwert und Standardabweichung kann ebenfalls in konstanter Zeit durchgeführt werden, wie auch die finale Gewichtung der vier Eckpunkte. □

3.3.3 Speicherplatzbedarf

Der Speicherplatzbedarf von MD-Min-Max ist durch $\Theta(N)$ beschränkt. Der Speicherplatzbedarf für die Koordinaten der Ankerknoten sowie für die gemessenen Distanzen ist linear, nämlich $3N$ Register. Zusätzlicher Speicherplatz wird für die Speicherung der Schleifenvariablen, der drei Parameter der Zugehörigkeitsfunktion sowie für die vier Eckpunkte von Min-Max und deren Gewichte benötigt. Dieser ist konstant. Der

Speicherplatz für die Berechnung der Gewichte aus μ_{ij} ist linear, kann jedoch ebenfalls konstant gehalten werden, falls eine Online-Methode wie Welfords [141] benutzt wird. Zusammenfassend kann gesagt werden, dass die asymptotische Laufzeit und Platzkomplexität von MD-Min-Max vergleichbar mit der des ursprünglichen Min-Max Algorithmus ist.

3.4 Der Geo-n Algorithmus

Der Geo-n Algorithmus [149] wurde von uns entwickelt, um speziell in schwierigen Umgebungen eine genaue Lokalisierung zu ermöglichen. Dies gilt insbesondere für den Indoor-Bereich, wo die geschätzten Entfernungen zwischen Knoten und Anker größere Fehler aufgrund der Mehrwege-Signalausbreitung beinhalten können. Hauptentwicklungsziel war eine hohe Beständigkeit gegen Ausreißer in den Distanzmessfehlern und ein breites Spektrum an geometrischen Konstellationen der beteiligten Knoten. Dies wird erreicht, indem der resultierende Positionsfehler räumlich möglichst einheitlich verteilt ist. Darüber hinaus ist die Laufzeitkomplexität des Algorithmus niedrig genug, um eine Echtzeit-Lokalisierung mit ressourcenbeschränkten Sensorknoten zu ermöglichen. Da im Gegensatz zu MD-Min-Max keine vorausgehenden Informationen wie die Fehlerverteilung der verwendeten Distanzmessmethode bekannt sein muss, kann Geo-n als Universalalgorithmus im Bereich der Indoorlokalisierung eingesetzt werden.

Wie viele der anderen in diesem Kapitel vorgestellten Algorithmen benutzt Geo-n ebenfalls die Schnittpunkte aller Kreispaare (C_i, C_j) mit $i \neq j$, um die Position eines nicht lokalisierten Knotens u zu berechnen. Auf die Menge der Schnittpunkte wird dann ein zweistufiger Filtermechanismus angewandt, um die maßgeblichen Schnittpunkte für die finale Positionsbestimmung zu erhalten. Dieser nutzt die Tatsache aus, dass die relevanten Schnittpunkte in einem Bereich einen Cluster bilden. Im Folgenden soll der Ansatz noch einmal genauer motiviert werden.

3.4.1 Motivation

Zur Motivation des gewählten Ansatzes kann Abbildung 3.10 betrachtet werden. Abbildung 3.10 zeigt den Knoten u sowie die Menge $A_u = \{A, B, C, D, E\}$ aller Ankerknoten, die von u erreicht werden können. Die schwarzen Kreise besitzen einen

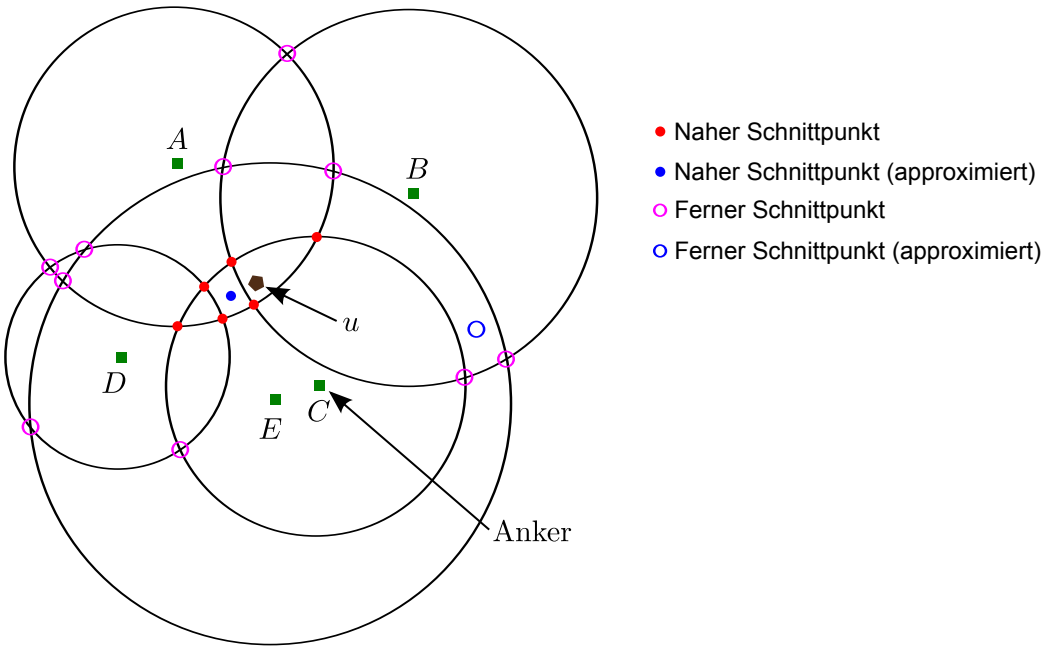


Abbildung 3.10: Motivation für einen Clustering ähnlichen Ansatz

Radius, der den fehlerbehafteten Distanzmessungen entspricht. Die Distanz zu Anker D wird beispielsweise ein bisschen zu kurz gemessen, während der Rest der Anker eine zu lange Distanzmessung aufweist (die geschätzte Entfernung ist größer als die echte Entfernung zu Knoten u). Im Besonderen weist Anker E einen sehr großen positiven Distanzmessfehler auf. Ohne jeglichen Messfehler würden sich alle Kreise an der Position von Knoten u schneiden. In der Realität werden jedoch immer Messfehler entstehen, die zum einen durch die Messung in einem verrauschten Kanal begründet sind und zum anderen durch NLOS Fehler entstehen, weil die Signale aufgrund von Hindernissen nicht den direkten Weg nehmen können und reflektiert empfangen werden. Folglich ist es möglich, dass kein Kreis durch die Position von u verläuft.

Die maximale Anzahl der Kreisschnittpunkte beträgt $2 \cdot \binom{N}{2}$, da sich zwei verschiedene Kreise höchstens in zwei Punkten schneiden können. Gehen wir von der Tatsache aus, dass die Mehrzahl der Messungen genau ist oder nur geringfügige Fehler aufweist, dann wird sich wegen der Geometrie in der Mehrzahl der Fälle einer der beiden Schnittpunkte dicht an der gesuchten Position von u befinden. Vereinzelt kann es vorkommen, dass beide Schnittpunkte den gleichen Abstand zu u aufweisen (z. B. die Schnittpunkte von Kreis A und C, unter idealen Bedingungen läge die Hälfte der Schnittpunkte an der gesuchten Position von u). Als Konsequenz ist die Dichte der Schnittpunkte um den gesuchten Knoten u am höchsten, formt also ein Clus-

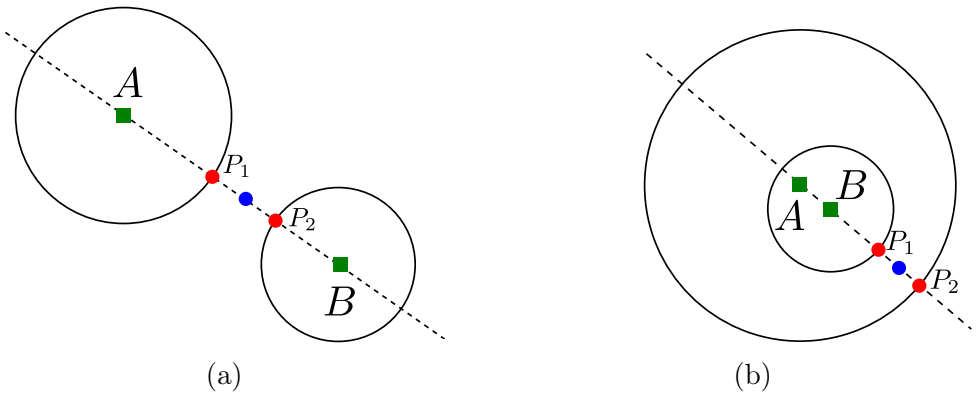


Abbildung 3.11: Approximation eines Schnittpunktes (in Blau gezeichnet), falls sich die Kreise nicht schneiden.

ter. Eine gute Approximation für dieses Cluster zu finden und dieses dann für die Positionsbestimmung zu nutzen, ist das wesentliche Problem, das von Geo-n gelöst wird.

Dieser Sachverhalt wird durch die sieben rot und blau gefärbten Punkte in Abbildung 3.10 angedeutet, die sich in der Nähe von u befinden und das größte Cluster bilden. Bei diesem Beispiel ist ein größerer Messfehler vorhanden. Die zweitgrößten Cluster bestehen jeweils nur aus drei Schnittpunkten und sind in Abbildung 3.10 mittels blauer und magentafarbener Kreise dargestellt. Jadliwala et al. haben bewiesen [59], dass die Anzahl der stark fehlerbehafteten Messungen im Allgemeinen nicht größer als $\frac{N-3}{2}$ sein darf, um eine feste obere Schranke für den Positionsfehler zu garantieren. Ähnliche Untersuchen wurden auch von Misra et al. durchgeführt [91].

3.4.2 Geo-n im Detail

Geo-n benutzt verschiedene Filtermethoden, um Schnittpunkte zu entfernen, die nicht zur Positionsbestimmung beitragen oder unter Verdacht stehen, den Positionsfehler zu erhöhen. Deshalb kann Geo-n den Standort von u mit hoher Genauigkeit bestimmen und ist gegen eine Vielzahl von Messfehlern beständig. Der Pseudocode von Geo-n ist in Algorithmus 4 beschrieben.

In den Zeilen 1 bis 2 werden die Variablen *IP* (die Liste aller Schnittpunkte) und *AIP* (die Liste aller approximierten Schnittpunkte) als leere Liste initialisiert. Zeilen 3 bis 12 befüllen *IP* mit allen Schnittpunkten aller Kreispaare, die sich aus den N An-

Algorithmus 4 Geo-n

Eingabe: Distanzmessungen r_i, Ankerpositionen a_i, $i = 1, \ldots, N$, $N \geq 3$, $\forall i, j : a_i \neq a_j \vee i = j$ und die Gewichte W_{IP} und W_{AIP};
Ausgabe: die geschätzte Position $\hat{u}$

$IP \leftarrow nil$; ▷ Schnittpunkte
$AIP \leftarrow nil$; ▷ Approximierte Schnittpunkte
for $i \leftarrow 1$ to $N - 1$ **do**
  for $j \leftarrow i + 1$ to N **do**
    $I \leftarrow C_i \cap C_j$;
    if $I \neq nil$ **then**
      $IP \leftarrow IP \oplus I$;
    else
      $AIP \leftarrow AIP \oplus$ [Approximiere einen Schnittpunkt zwischen C_i und C_j];
    end if
  end for
end for
$IP \leftarrow [x \in IP \mid \left|\{i \mid x \in \overline{D}_i\}\right| \geq N - 2]$;
$WIP \leftarrow nil$; ▷ Gewichtete Schnittpunkte
for all $x \in IP$ **do**
  $WIP \leftarrow WIP \oplus [(x, W_{IP})]$;
end for
for all $x \in AIP$ **do**
  $WIP \leftarrow WIP \oplus [(x, W_{AIP})]$;
end for
if $|WIP| \geq 3$ **then**
  $distance[\,|WIP|\,] \leftarrow 0$; ▷ Setze alle Distanzen gleich Null
  for $i \leftarrow 1$ to $|WIP|$ **do**
    for $j \leftarrow 1$ to $|WIP|$ **do**
      if $i \neq j$ **then**
        $distance[i] \leftarrow distance[i] + \| WIP_i - WIP_j \|$;
      end if
    end for
  end for
  $median \leftarrow$ berechne Median von $distance$;
  $WIP \leftarrow [x \in WIP \mid distance$ von $x \leq median]$;
end if
$\hat{u} \leftarrow$ gewichteter geometrischer Schwerpunkt von WIP;
return $\hat{u}$

kerkoordinaten a_i und gemessenen Distanzen r_i ergeben und AIP mit approximierten Schnittpunkten, falls sich die Kreise nicht schneiden. In Zeile 9 wird ein Schnittpunkt approximiert, falls es keinen echten Schnittpunkt gibt. Die zwei möglichen Fälle in diesem Zusammenhang sind in Abbildung 3.11a und Abbildung 3.11b dargestellt. In beiden Fällen ist es immer noch möglich, einen Schnittpunkt zu approximieren. Dieser Punkt (in den Abbildungen in Blau hervorgehoben) ist der Mittelpunkt der Linie, der die Punkte P_1 und P_2 verbindet. Hierbei sind P_1 und P_2 die am nächsten beieinander liegenden Schnittpunkte der beiden Kreise mit einer Geraden durch die Position der Anker A und B, wobei P_1 und P_2 auf verschiedenen Kreisen liegen müssen. Als Ergebnis der Zeilen 3 bis 12 ist die Anzahl der paarweisen Schnittpunkte zwischen den entsprechenden Kreisen nicht größer als N^2. Für jedes Kreispaar existieren höchstens zwei, mindestens jedoch ein (approximierter) Schnittpunkt, demzufolge gilt: $\frac{N\cdot(N-1)}{2} \leq |IP \oplus AIP| \leq N \cdot (N-1)$.

Zeile 13 und Zeilen 22 bis 31 spezifizieren einen zweistufigen Filtermechanismus auf den erzeugten Schnittpunkten des vorhergehenden Schritts. Zeile 13 behält nur solche Punkte in IP, die in mindestens $N-2$ geschlossenen Kreisscheiben $\overline{D}_i$ ($i \in \{1, \ldots, N\}$) enthalten sind. Von jetzt an bezeichnen wir diesen Schritt als Filter 1 (kurz: F1). Die meisten der fernen Schnittpunkte werden als Resultat von F1 aussortiert (vgl. Abbildung 3.10) und damit einhergehend die Gesamtlaufzeit des Algorithmus unter Umständen deutlich reduziert. Vorherige Ergebnisse von uns haben gezeigt [149], dass durchschnittlich ungefähr 60 % der Schnittpunkte entfernt werden. Dies führt zu einer deutlichen Reduzierung der Ausführungszeit, die zwischen 20 % und 60 % (abhängig von der Anzahl der Anker N) liegen kann. Es sei hier angemerkt, dass approximierte Schnittpunkte nicht durch diesen Filter tangiert werden. Diese zu behalten, führt so gut wie immer zu einer erhöhten Lokalisierungsgenauigkeit, besonders dann, wenn die Anzahl an echten Schnittpunkten dazu neigt, gering zu sein.

Bevor der zweite Filter angewandt werden kann, fügen die Zeilen 14 bis 20 die beiden Listen IP und AIP zu einer neuen Liste WIP zusammen, die gewichtete Schnittpunkte für den finalen Schritt von Geo-n enthält.

Zeilen 22 bis 31 implementieren einen Medianfilter basierend auf den Abständen zwischen den Punkten, um nochmalig Schnittpunkte zu entfernen, die das Ergebnis höchstwahrscheinlich negativ beeinflussen. Zuerst wird für jeden Punkt in WIP die Summe der Abstände zu allen anderen Punkten berechnet und in dem Feld *distance* gespeichert. Als nächstes berechnet Zeile 30 den Median des *distance* Feldes und Zei-

le 31 eliminiert die Punkte in *WIP*, dessen Wert in *distance* größer als der Median ist. Die Wahl des Medians erfolgte empirisch, indem auch kleinere Quantile und der Durchschnitt untersucht wurden. Im Schnitt erwies sich der Median jedoch als beste Metrik für die Filterung. Es gilt zu beachten, dass dieser Filter nur angewandt wird, falls die Anzahl der Punkte in *WIP* nicht kleiner als 3 ist. Andernfalls würde die Filterung nur einen gewichteten Schnittpunkt übrig lassen, der eventuell weiter von der echten Position entfernt ist, als das Ergebnis unter Verwendung der ganzen Liste zu nehmen. Von jetzt an bezeichnen wir diesen Schritt als Filter 2 (kurz: F2). In Summe ist die kombinierte Anwendung von beiden Filtern in der gegebenen Reihenfolge nutzbringend für die Steigerung der Lokalisierungsgenauigkeit. Eine Verwendung von F1 steigert das Ergebnis der Positionierung um 13 % bis 20 % in unseren Experimenten und Simulationen verglichen mit der ausschließlichen Benutzung von F2. Für eine noch genauere Analyse der Filter und einen tieferen Überblick über die Entwurfsentscheidungen, die Geo-n letztendlich seine endgültige Form gegeben haben, sei der Leser auf die Originalveröffentlichung [149] verwiesen.

Schlussendlich wird die Position des nicht lokalisierten Knotens als gewichteter geometrischer Schwerpunkt der übriggebliebenen Schnittpunkte abgeschätzt, so wie es Gleichung (3.39) zu entnehmen ist. Echte Schnittpunkte werden mit Gewicht W_{IP} gewichtet, approximiere Schnittpunkte mit Gewicht W_{AIP}. In den Experimenten stellte sich ein gleiches Gewicht als optimal heraus, in Simulationen hingegen zeigte eine dreifach höhere Gewichtung von echten Schnittpunkten die beste Genauigkeit [149].

$$\hat{u} = (\hat{x}, \hat{y}) = \left(\frac{\sum_{i=1}^{|WIP|} w_i \cdot x_i}{\sum_{i=1}^{|WIP|} w_i}, \frac{\sum_{i=1}^{|WIP|} w_i \cdot y_i}{\sum_{i=1}^{|WIP|} w_i} \right) \tag{3.39}$$

3.4.3 Laufzeitanalyse

Die Laufzeit von Geo-n ist durch $\mathcal{O}(N^4)$ beschränkt.

Beweis. Die Berechnung aller Schnittpunkte in den Zeilen 3 bis 12 hat eine asymptotische Laufzeit von $\mathcal{O}(N^2)$. Zeile 13, die die weiter entfernt liegenden Schnittpunkte aussortiert, besitzt eine asymptotische Laufzeit von $\mathcal{O}(N^3)$. Die Vereinigung der beiden Schnittpunktlisten in den Zeilen 14 bis 20 hat ebenfalls eine Laufzeit von $\mathcal{O}(N^2)$, da es wie vorher beschrieben maximal $N^2 - N$ Schnittpunkte geben kann. Die Medianfilterung der übriggebliebenen Schnittpunkte dominiert die Laufzeit von Geo-n

klar. Die Berechnung der Distanzsummen für alle Punkte in den Zeilen 22 bis 29 hat eine Laufzeit von $\mathcal{O}(N^4)$. Die Laufzeit der Bestimmung des Medians in Zeile 30 ist durch $\mathcal{O}(N^2)$ beschränkt, falls eine Methode wie BFPRT [13] benutzt wird. Die Elimination der Punkte, deren Distanzsumme größer als der Median ist (Zeile 31), hat erneut eine asymptotische Laufzeit von $\mathcal{O}(N^2)$. Die finale Positionsabschätzung mittels Gleichung (3.39) hat ebenfalls eine Laufzeit von $\mathcal{O}(N^2)$. □

Generell wird die Zahl der Anker N, von denen ein nicht lokalisierter Knoten Referenzpositionen empfängt, aufgrund der limitierten Reichweite der drahtlosen Kommunikation und durchschnittlichen Knotendichte in den meisten Anwendungsfällen gering sein. Deshalb ist die Laufzeit von Geo-n annehmbar und selbst auf limitierter Hardware eine Echtzeitlokalisierung möglich. Ein genauerer Vergleich konkreter Laufzeiten mit konkreten Ankeranzahlen wird für Geo-n sowie die anderen Algorithmen während der Auswertung in Kapitel 6 und 7 gegeben.

3.4.4 Speicherplatzbedarf

Der Speicherplatzbedarf von Geo-n ist durch $\Theta(N^2)$ beschränkt. Der Speicherplatzbedarf für die Koordinaten der Ankerknoten sowie für die gemessenen Distanzen ist linear, nämlich $3N$ Register. Der meiste Speicherplatz wird für die erzeugten Kreisschnittpunkte benötigt, von denen, wie vorhin beschrieben, höchstens $N \cdot (N-1)$ existieren. Das *distance* Feld hat die gleiche maximale Größe. Demnach ist Geo-n hinsichtlich des Gesamtbedarfs an Speicher noch als genügsam einzustufen.

3.4.5 Anpassung an eine Fehlerverteilung

Auch Geo-n lässt sich unter Beibehaltung der asymptotischen Laufzeit und des Speicherplatzbedarfs an eine konkrete Fehlerverteilung der Distanzmessung eines ausgewählten Szenarios anpassen. Damit verliert Geo-n in gewisser Weise seine Eigenschaft, universell einsetzbar zu sein, da Anpassungen des Algorithmus notwendig sind, gewinnt aber zusätzlich an Genauigkeit. Wie dies zu bewerkstelligen ist, soll im Folgenden kurz beschrieben werden. Im Gegensatz zu dem in Abschnitt 3.4.2 vorgestellten Basisalgorithmus wurde diese Anpassung bisher nicht publiziert.

Für eine Anpassung des Algorithmus an eine Verteilung der Distanzfehler können die übriggebliebenen Schnittpunkte in *WIP*, die zur finalen Positionsbestimmung in

Zeile 33 in Algorithmus 4 verwendet werden, genommen werden. Das Gewicht dieser eventuell bereits aus vorhergehenden Schritten gewichteten Schnittpunkte wird mittels einer frei wählbaren Gewichtungsfunktion erneut angepasst, bevor der gewichtete geometrische Schwerpunkt berechnet wird. Wie im Falle der MLE benutzen wir eine Normalverteilung und eine Gammaverteilung für die Approximation der Fehlerverteilung. Die Herleitung der Parameter dieser Verteilungen wird in Abschnitt 7.3 durchgeführt. Die angepassten Geo-n Algorithmen werden entsprechend als Geo-n-$\mathcal{N}$ und Geo-n-Γ bezeichnet und besitzen die gleichen Parameter und ermittelten Werte wie MLE-$\mathcal{N}$ und MLE-Γ. Die Gewichtung eines einzelnen Schnittpunktes u erfolgt dann mit den entsprechenden Likelihood-Funktionen sowie den gemessenen Distanzen und den Ankerpositionen. Für die Normalverteilung erfolgt dies nach Gleichung (3.41). Gleichung (3.40) zeigt noch einmal die dabei verwendete Dichtefunktion der Normalverteilung.

$$\mathcal{N}(\mu, \sigma^2)(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\} \tag{3.40}$$

$$w_u = p(r \mid u) = \prod_{i=1}^{N} p_i(r_i \mid u) = \prod_{i=1}^{N} \mathcal{N}(\mu, \sigma^2)(r_i - \|u - a_i\|) \tag{3.41}$$

Für den Fall der Gammaverteilung ergibt sich das neue Gewicht w_u eines Schnittpunktes analog aus der Likelihood-Funktion von Gleichung (3.23). Diese Berechnung muss natürlich für alle Schnittpunkte in *WIP* durchgeführt werden, was zu einer Laufzeit von $\mathcal{O}(N^3)$ führt, da es maximal quadratisch viele Schnittpunkte geben kann. Die Laufzeit des angepassten Geo-n bleibt also durch $\mathcal{O}(N^4)$ beschränkt. Zusätzlicher Speicherplatz wird ebenfalls nicht benötigt, da die neuen Gewichte den Speicherplatz der alten Gewichte wiederverwenden, nachdem diese miteinander multipliziert wurden.

3.5 Die LatMath Bibliothek

Sämtliche der hier vorgestellten Algorithmen wurden von uns anhand der Originalveröffentlichung in der Programmiersprache Java implementiert und in einer frei verfügbaren Bibliothek namens *LatMath* zusammengetragen. Diese Bibliothek wird an

der FU Berlin weiterentwickelt und gepflegt [52].

Im Wesentlichen implementiert jeder Algorithmus die in Quellcode 3.1 gezeigte Schnittstelle für den einheitlichen Aufruf aus anderen Programmen. Da es sich um distanzbasierte Verfahren handelt, werden der Lokalisierungmethode die Ankerpositionen und die gemessenen Distanzen als Parameter übergeben. Das Fehlermodell und die Maße des Spielfeldes sind nur als optionale Parameter für den Einsatz in Simulationen bestimmt.

```
1       String getName();
2
3       Point2d localize(Point2d[] anchors, double[] ranges,
            Point2d actualPosition, ErrorModel errorModel, int
            width, int height);
```

Quellcode 3.1: Lokalisierungsschnittstelle für Algorithmen (LaterationAlgorithm.java)

Darüber hinaus implementieren die Algorithmen noch weitere Schnittstellen, welche z. B. die Konfiguration der Parameter eines Algorithmus ermöglichen oder deren Speicherung für eine erneute Auswertung zu späteren Zeitpunkten. Auch Methoden zur statistischen Auswertung der Genauigkeit und Präzision der Algorithmen sind vorhanden. Hierfür lassen sich nach Aufruf des Algorithmus z. B. Metriken wie der MAE, der Root Mean Squared Error (RMSE) oder der maximale Fehler abfragen.

Neben den Lokalisierungsalgorithmen sind noch Schnittstellen für folgende Elemente definiert: Fehlermodelle für die Simulation eines Distanzmessfehlers, Filter für die gemessenen Distanzen und berechneten Positionen sowie Ankerselektionsalgorithmen. Die Fehlermodelle finden in erster Linie ihre Anwendung bei der simulativen Auswertung der Algorithmen, während die Filter z. B. im Werkzeug *LatViz* eingesetzt werden können. Dieses wurde von uns zur Experimentauswertung entwickelt, eine genauere Beschreibung von *LatViz* befindet sich in Abschnitt 5.2.2. Die vorhandenen Fehlermodelle sind identisch mit den Fehlermodellen, die in Abschnitt 5.1.2 bei der Betrachtung der Simulation der räumlichen Fehlerverteilung vorgestellt werden. Da wir uns im Rahmen dieser Arbeit auf die Evaluation von Lokalisierungsalgorithmen konzentriert haben, ist die vorhandene Zahl der Filter und Ankerauswahlalgorithmen überschaubar. Zu nennen wären hier beispielsweise ein Kalman-Filter für die Positionen und ein Medianfilter für die Glättung der gemessenen Distanzen.

Schließlich gibt es in der Bibliothek diverse Methoden zum automatischen ermitteln

Tabelle 3.2: Laufzeit- und Speicherkomplexität der Algorithmen

Algorithmus	asymptotische Laufzeit	Speicherplatzbedarf
AML	$\mathcal{O}(N^2)$	$\Theta(N)$
BL	$\mathcal{O}(N^4)$	$\Theta(N^2)$
CluRoL	$\mathcal{O}(N^4 \log N)$	$\mathcal{O}(N^4)$
E-Min-Max	$\Theta(N)$	$\Theta(N)$
Geo-n, Geo-n-$\mathcal{N}$, Geo-n-Γ	$\mathcal{O}(N^4)$	$\Theta(N^2)$
ICLA	$\mathcal{O}(N^4)$	$\Theta(N^2)$
LLS	$\mathcal{O}(N^3)$	$\Theta(N)$
LMS	$\mathcal{O}(N^3)$	$\Theta(N)$
MD-Min-Max	$\Theta(N)$	$\Theta(N)$
Min-Max	$\Theta(N)$	$\Theta(N)$
MLE-$\mathcal{N}$	$\mathcal{O}(N^3)$	$\Theta(N)$
NLLS	$\mathcal{O}(N^3)$	$\Theta(N)$
RLSM	$\mathcal{O}(4^N)$	$\mathcal{O}(4^N)$
Rwgh	$\mathcal{O}(N^3 \cdot 2^N)$	$\Theta(N)$
VBLE	$\mathcal{O}(N \cdot M)$	$\Theta(N)$
VBLE-O	$\mathcal{O}(N \cdot M)$	$\Theta(N)$

einer Liste aller verfügbaren Elemente eines Typs (Algorithmus, Filter etc.). Dies ist über die *Java Reflection API* gelöst. Hierdurch reicht z. B. das einfache Implementieren eines neuen Algorithmus und Bauen der Bibliothek aus, um diesen anderen Programmen automatisch zur Verfügung zu stellen.

3.6 Zusammenfassung

Tabelle 3.2 fasst noch einmal die Laufzeit- und Speicherkomplexität aller Algorithmen aus diesem Kapitel zusammen. Hierbei steht N für die Anzahl der Ankerknoten zu denen Distanzmessungen vorliegen und M für die Anzahl der Zellen beim VBLE und VBLE-O Algorithmus. Der Speicherplatzbedarf der meisten Algorithmen ist moderat und liegt nicht höher als quadratisch in der Anzahl der Anker (bei der Mehrzahl der Algorithmen in der Eingabegröße von $\Theta(N)$). Ausnahmen bilden CluRoL und RLSM. Letzterer hat enorm hohe Speicheranforderungen, worauf bei der Auswertung in Kapitel 7 auch noch einmal eingegangen wird. Bei den Laufzeiten zeigen sich größere Unterschiede zwischen den einzelnen Algorithmen. Simple Algorithmen wie die Min-Max Varianten weisen lineare Laufzeiten auf, während Optimierungsverfahren

wie LLS, NLLS und MLE-$\mathcal{N}$ kubische Laufzeiten besitzen. Die Laufzeit der meisten geometrischen Algorithmen, welche Kreisschnittpunkte bilden und ein Clustering durchführen, ist durch $\mathcal{O}(N^4)$ beschränkt. Algorithmen wie Rwgh und RLSM, die die Robustheit gegenüber Messausreißern zu erhöhen versuchen, weisen sogar eine exponentielle Laufzeit auf.

Da die Anzahl der verfügbaren Ankerknoten in den meisten realen Szenarien durch die beschränkte Kommunikationsreichweite im Indoor-Bereich gering sein wird (weniger als 10 Anker; vgl. Abschnitt 7.4), fallen die teils groß erscheinenden asymptotischen Laufzeiten jedoch in der Praxis nur selten ins Gewicht. Viel entscheidender sind die zum Teil vorhandenen versteckten Konstanten bei den einzelnen Algorithmen, die bei kleinen Ankerzahlen ins Gewicht fallen können. Deshalb ist das Benchmarking der wirklichen Laufzeit eines Lokalisierungsaufrufs neben der theoretischen Laufzeitbetrachtung ebenso von entscheidender Bedeutung für eine eventuelle Algorithmenauswahl. Ergebnisse in diesem Zusammenhang werden vornehmlich in Abschnitt 5.1.2 und Kapitel 6 vorgestellt, aber auch bei der Auswertung der Experimente in Kapitel 7 wird an der einen oder anderen Stelle darauf eingegangen.

KAPITEL 4

Umgebungsparameter und Bewertungsmetriken

Um verschiedene Ansätze der hochpräzisen distanzbasierten Indoorlokalisierung vergleichen zu können, braucht es verschiedene Metriken anhand derer diese Vergleiche ausgewertet werden können. Neben diesen Metriken bedarf es aber auch wohldefinierter Umgebungsbedingungen, die Ergebnisse eines Vergleichs verschiedener Systeme möglichst allgemeingültig erscheinen lassen. Während sich die Vergleichsmetriken relativ einfach bestimmen lassen, sind allgemeine und reproduzierbare Umgebungsbedingungen nur schwer zu definieren und wesentlich schwerer herzustellen. Mitunter sind die konkreten Anforderungen so verschieden, dass die Umgebungsbedingungen eines Experimentes keinerlei Rückschluss auf das allgemeine Verhalten des getesteten Systems zulassen. Insbesondere spielen hier die Positionen der Ankerknoten für die Entfernungsmessung, das Bewegungsmodell, sowie die strukturellen Eigenschaften eines Gebäudes die größte Rolle. Selbst bei fixer Ankerposition in ein und demselben Gebäude kann der Vergleich diverser Ansätze bei einem leicht abgewandelten Weg durch das Gebäude völlig unterschiedlich ausfallen. Auch statistische Methoden, die eine Lokalisierung stichpunktartig für das gesamte Gebäude testen, können für ein Einsatzszenario, das auf bestimmte Abläufe in diesem Gebäude ausgerichtet ist (z. B. ausschließliche Benutzung des Gebäudes bei Nacht), Ergebnisse liefern, die allgemeine Aussagen nicht mehr zulassen.

Noch schwieriger ist eine Einbeziehung dieser Umgebungsparameter in rein analytische Ansätze. Deshalb beschränken sich analytische Bewertungsmethoden oft darauf,

das Potenzial eines Algorithmus oder Verfahrens abzuschätzen und einen Grenzwert zu bestimmen. Diese Probleme besprechen wir beispielhaft anhand des einzigen uns aus der Literatur im Zusammenhang mit der Indoorlokalisierung bekannten Verfahrens, der Cramér-Rao-Ungleichung (CRLB).

Trotz der offensichtlichen Probleme bei der Aufstellung dieser Kriterien und der fraglichen Übertragbarkeit auf die Allgemeinheit sind derartige Metriken und Settings eine große Hilfe bei der Auswahl geeigneter Verfahren. Insbesondere wenn diese Verfahren parallel zum restlichen Aufbau eines Lokalisierungssystems (Hardware zur Distanzmessung, Computerhardware etc.) ausgewählt werden müssen oder konkrete Testläufe mangels verfügbarer Hardware überhaupt noch nicht möglich sind.

4.1 Fehlerquellen der distanzbasierten Indoorlokalisierung

Neben diversen Quellen für Messfehler in den gemessenen Entfernungen spielen bei der distanzbasierten Indoorlokalisierung auch Fehler in dem Algorithmus zugrunde liegenden Modell eine große Rolle für den letztendlich resultierenden Positionsfehler. Lassen sich Messfehler durch eine bessere und hochwertigere Komponentenauswahl recht einfach reduzieren, so sind die Modellfehler deutlich schwerer zu reduzieren, da sie aus einer Reihe schwer abzuschätzender oder gänzlich unbekannter Umgebungsparameter resultieren. Diese Fehlerquelle, die aus der Abweichung zwischen dem Algorithmus zugrunde liegenden Modell und der realen Umgebung beschrieben werden kann, nennen wir im Folgenden algorithmusinhärente Fehler. Beispielsweise geht die naive Trilateration im zugrunde liegenden mathematischen Modell davon aus, dass alle gemessenen Entfernungen keinerlei Messfehler enthalten. Ist dies der Fall, so wird eine absolut genaue Position berechnet. Für diesen Fall weist dieser Algorithmus keinerlei inhärenten Fehler auf. Andere Algorithmen wie z. B. VBLE werden trotz eines Eingabefehlers von Null keine absolute Genauigkeit zurückliefern. Kommt bei ansonsten gleich bleibender Umgebung ein Messfehler auf die ermittelten Distanzen, so wird sich das Verhältnis schnell umkehren. Je nach geometrischer Konstellation lässt sich z. B. VBLE nur unterproportional durch den Messfehler beeinflussen, während der Fehler der naiven Trilateration extrem ansteigt.

Wir definieren in unserer Arbeit den Schätzfehler der Distanzen als gerichteten Wert. Schätzungen, die zu kurz ausfallen, kennzeichnen wir durch ein negatives Vorzeichen. Schätzungen, die zu lang ausfallen, entsprechend durch ein positives Vorzeichen. Bei

allen statistischen Aussagen über den Distanzfehler sind somit vor deren Berechnung die Beträge der Werte zu bilden, um Aussagen über die absoluten Fehler treffen zu können!

4.1.1 Fehlerursachen in der verwendeten Hardware

Die Entfernungsmessung basiert im Wesentlichen darauf, eine physikalische Größe auf einem Knoten zu messen und mithilfe eines Modells daraus die Entfernung zu einem weiteren Knoten abzuschätzen. Gebräuchliche Verfahren sind das Messen von Signallaufzeiten und das Messen von Signaldämpfungen zwischen zwei Knoten. Für den Hochpräzisionsbereich kommt in erster Linie das Messen von Signallaufzeiten in Betracht. Die Signale sind dabei meistens elektromagnetische Signale, es kommen aber auch optische und akustische Verfahren zum Einsatz.

Die erste Fehlerquelle liegt dabei im Messen selber begründet, da sich keine physikalische Größe fehlerfrei messen lässt. Art und Betrag des Messfehlers lassen sich durch eine geeignete Hardwareauswahl oft erheblich reduzieren, jedoch verbunden mit Nachteilen wie höheren Kosten oder höherer Stromaufnahme. Beim Messen von Signallaufzeiten wird der Messfehler gleich aus mehreren Quellen gespeist. Als erstes sind hier ungenaue Uhren zu nennen [80]. Die in Computersystemen verwendeten Methoden zur Zeitmessung fußen meistens auf Quarzen. Diese können je nach Qualität, Alter und Umgebungstemperatur eine Abweichung von ihrer Sollfrequenz haben und somit zu kurze oder zu lange Perioden messen. Allerdings ist bei der Verwendung von hochgenauen Quarzen oder noch präzisieren, nicht Quarz basierten Uhren, eine sehr große Genauigkeit möglich. Der Aufwand hierfür steigt jedoch sehr schnell an - so kommen beim GPS-System in den Satelliten CS- oder RB-Atomuhren zum Einsatz, die eine Ganggenauigkeit von 10^{-13} aufweisen, mindestens 1500 USD kosten und eine Stromaufnahme von mind. 150 mW besitzen. Ein wesentlich größeres Problem als das präzise Messen einer Zeitdifferenz stellt jedoch die genaue Bestimmung der Eingangszeit eines (digitalen) Signals dar. Handelt es sich bei dem Signal um ein Funksignal, so werden in der Regel eine große Menge an Bauteilen und Algorithmen benötigt, um ein digitales Signal als ein solches zu erkennen. Hierbei findet ein Teil der Signalverarbeitung analog und der andere Teil digital statt. Beim Wechsel von analoger auf digitaler Verarbeitung sind die benötigten Zeitspannen für die Signalverarbeitung nicht immer deterministisch, so dass der Zeitpunkt des Messens nicht immer der Zeitpunkt des Eintreffens des Signals sein muss. Hinzu kommt in jedem

Fall der Jitter, der sich direkt aus der Abtastrate des digitalen Signals ergibt [152]. Eine vollständig analoges Signal würde bei entsprechend hoher Abtastung theoretisch zwar eine fast beliebige Genauigkeit bieten, allerdings lässt ein solches Signal die Zuordnung zu einem bestimmten Knoten nur über Zeitmultiplexing zu und ist zusätzlich noch extrem anfällig für Störungen. Spezialisierte Hardware nimmt daher eine solche Messung möglichst früh in der Signalverarbeitungskette vor oder moduliert ein Signal derartig, dass eine Vielzahl von Messpunkten existiert und so ein zuverlässiges statistisches Mittel gebildet werden kann. Ein Beispiel für eine solche Modulation ist die Chirp-Technologie [56, 72].

Wesentliche Eigenschaft dieser Fehler ist, dass insbesondere durch den Uhrenfehler auch negative Messfehler (zu kurze Strecken) auftreten können - die Fehlerverteilung unterliegt meistens der Normalverteilung.

4.1.2 Fehlerursachen im Bewegungsmodell des Knotens

Alle uns bekannten Algorithmen für die distanzbasierte Indoorlokalisierung gehen davon aus, dass die gegebenen Distanzen quasi gleichzeitig gemessen wurden und somit den Zustand des Gesamtsystems an genau einer Position des Knotens und der Anker widerspiegeln. Da mit der verfügbaren Hardware eine Distanzmessung zu mehreren Ankerknoten nur schwer möglich ist, bedeutet dies für das Modell des Algorithmus, dass sich der Knoten nicht bewegen kann. Wird ein mobiler Knoten mit einem solchen System trotzdem benutzt, so entsteht durch die zurückgelegte Strecke während der sequentiellen Entfernungsmessungen zu den Ankerknoten ein zusätzlicher Fehler, der umso größer ist, desto schneller die Bewegungsgeschwindigkeit des Knotens ist und desto länger die Messung zu einem Ankerknoten zurückliegt. Bei der Lokalisierung von Menschen spielt diese Fehlerquelle aufgrund der im Verhältnis zur Dauer einer einzelnen Messung sehr niedrigen Bewegungsgeschwindigkeit kaum eine Rolle.

Dieses Problem kann durch die Verwendung von TDOA Verfahren, wie sie auch GNSS verwenden, verhindert werden. Allerdings sind TDOA Systeme nur in wenigen speziellen Szenarien einsetzbar und sie erfordern einen hohen Synchronisierungsaufwand für die Uhren der Ankerknoten [45]. In kooperativen Systemen sind sie prinzipbedingt nicht zu verwenden.

In der kooperativen Indoorlokalisierung spielt bei sich bewegenden Knoten der Aspekt

der Lokalisierungsfrequenz eine große Rolle, da alle Knoten gleichzeitig auch als Anker fungieren. Sind diese Anker nun ebenfalls mobil, so kommt zum Messfehler der Distanz zusätzlich noch ein Fehler in der eigenen Position zum Tragen, da die an den Knoten übermittelte Position zwangsläufig nicht mehr aktuell ist. Ein solcher Fehler propagiert sich schnell durch das gesamte Netzwerk und führt zu bisher nur schwer zu beherrschenden Problemen in solchen Ansätzen [153].

4.1.3 Fehlerursache in strukturellen Gebäudeeigenschaften

Die wesentliche Fehlerquelle bei der distanzbasierten Indoorlokalisierung sind Modellfehler die eine falsche Aussage über die gemessene Entfernung implizieren. Selbst bei einer exakten Abschätzung eines Signalweges muss die abgeschätzte Wegstrecke eben nicht der euklidische Abstand der beiden involvierten Knoten sein! Sogenannte *Multipath*-Effekte und vor allem Reflexionen führen dazu, dass der Signalweg nicht dem euklidischen Abstand entspricht. Diese Effekte sind bei der Indoorlokalisierung im Durchschnitt wesentlich größer, als die Messfehler der Distanzen oder die Fehler, die aus der Mobilität der Knoten resultieren [121, 17].

Eine Reflexion tritt auf, wenn ein Signal auf feste Materie trifft. Je nach Materialeigenschaften wird ein Teil des Signals absorbiert und das Signal setzt seinen Weg abgeschwächt durch die Materie fort und ein anderer Teil des Signals wird abgeschwächt reflektiert. Dieser Effekt tritt in Gebäuden vor allem an Wänden und Einrichtungsgegenständen auf. Von einem *Multipath*-Effekt spricht man dann, wenn beim Empfänger sich mehrere Signale der gleichen Form zeitversetzt überlagern und der ursprüngliche Inhalt nicht mehr rekonstruiert werden kann. So kann es passieren, dass das Originalsignal den Empfänger zwar erreicht, aber durch eine Reflexion, die nur einen geringfügig längeren Weg genommen hat, überlagert und ausgelöscht wird. Nun kommt das Signal entweder gar nicht mehr an oder zu einem noch späteren Zeitpunkt erreicht eine weitere Reflexion des Ursprungssignals, das einen wesentlich längeren Weg genommen hat, den Empfänger und kann erfolgreich verarbeitet werden [58].

Da diese Fehler durch ein fehlerhaftes Modell bzw. durch die strukturellen Eigenschaften eines Gebäudes hervorgerufen werden, lassen sie sich statistisch nur sehr schwer beherrschen. Ansätze, diese durch eine Anpassung der verwendeten Hardware zu minimieren, sind oft mit starken Nachteilen verbunden. So lassen sich bei der Laufzeit-

messung von Funkpaketen die Fehler durch Reflexionen fast vollständig minimieren, wenn man Systeme mit extrem hoher Bandbreite (UWB-Systeme) verwendet. Bei dieses Systemen überlagern Reflexionen, die eine deutlich längere Wegstrecke genommen haben, nicht das Signal, das direkt die Luftlinie (also den euklidischen Abstand) genommen hat. Das liegt daran, dass diese Signale durch die hohe Bandbreite zeitlich extrem kurz sind. Diese Systeme haben allerdings einen sehr hohen Energiebedarf und eine sehr geringe Reichweite, so dass eine Vielzahl von Ankerknoten benötigt wird. Außerdem sind diese Signale sehr hochfrequent und können auch dünnste Wände oder Einrichtungsgegenstände kaum durchdringen, so dass das Originalsignal oft gar nicht die Eingangsempfindlichkeit des Empfängers übersteigt und somit das reflektierte Signal das einzig Verwertbare bleibt. UWB-Systeme kommen vor allem in Umgebungen, in denen LOS-Bedingungen herrschen, zum Einsatz und sind dort äußerst zuverlässig [40].

Wesentliche Eigenschaften dieser Modellfehler sind, dass sie in der Regel deutlich größer sind als die hardwarebedingten Messfehler und die Verteilung keiner bekannten statistischen Verteilung unterliegt, da sie stark von externen Faktoren beeinflusst sind [116]. Diese Art von Fehlern sind immer positiv, d. h. die gemessene Entfernung ist größer als die tatsächliche, da es keine kürzeren Signalwege als den im Modell angenommenen euklidischen Abstand gibt [155].

Der Umgang mit diesen Modellfehlern stellt die wesentliche Herausforderung an Algorithmen für die Indoorlokalisierung dar und die aus der Literatur bekannten Algorithmen unterscheiden sich wesentlich in der Behandlung dieser sog. NLOS-Fehler [19, 132, 46].

Spezialisierte Ansätze [39], die versuchen NLOS-Anker zu erkennen und somit nur LOS-Anker für eine Lokalisierung zu nutzen, sind in der Indoorlokalisierung wenig zielführend, da oft nur wenige Anker überhaupt zur Verfügung stehen oder alle Anker NLOS-Anker sind.

4.1.4 Algorithmusinhärente Fehler

Zusätzlich zu den Messfehlern und den modellbasierten Fehlern in der Entfernungsmessung, fügen manche Algorithmen der Messung einen situationsabhängigen *Bias* hinzu, der ebenfalls in einem Modellfehler begründet liegt. So erwartet die klassische Trilateration drei exakt bestimmte Entfernungen und drei dazugehörige, exakt

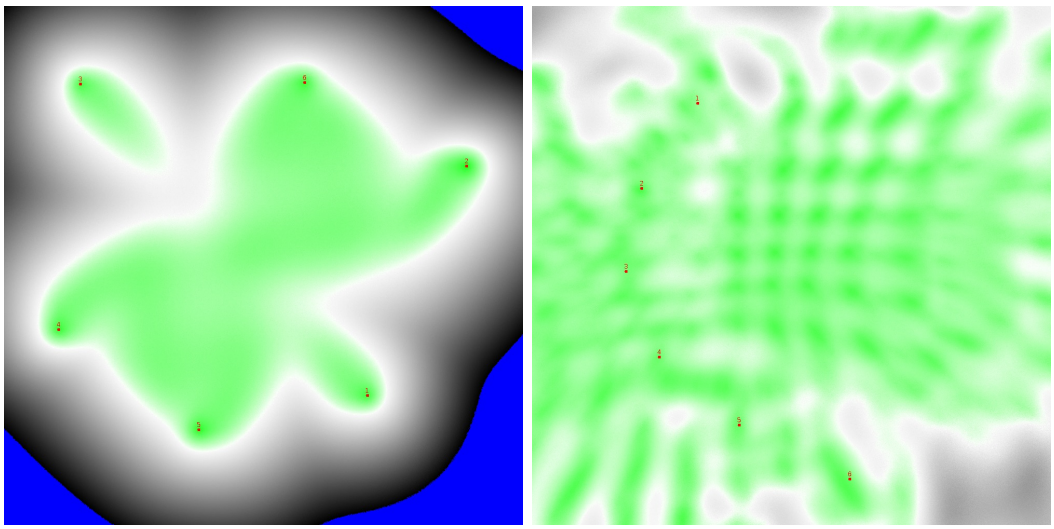
bestimmte Positionen. Werden diese Bedingungen eingehalten, so wird in jedem Fall (Ausnahme Kollinearität aller Anker) die exakte Position des Knotens zurückgegeben. Enthalten die Messungen minimale Fehler, so hängt der resultierende Positionsfehler nur unwesentlich vom Betrag der Messfehler und in einem wesentlich höheren Ausmaß von der Position der Ankerknoten ab. Bilden die Ankerknoten z. B. ein gleichseitiges Dreieck und der Knoten befindet sich am Schwerpunkt dieses Dreiecks, so wird der Positionsfehler ungefähr dem mittleren Betrag der Messfehler entsprechen. Entfernt sich der Knoten vom Schwerpunkt des Dreiecks, so wird der Fehler - bei gleichem mittleren Messfehler - stark ansteigen und nach kürzester Zeit wird das Ergebnis völlig unbrauchbar. Dieses Verhalten eines Algorithmus ist ein wesentliches Kriterium für die Robustheit gegenüber Fehlern in der Distanzmessung. Als Analysewerkzeug für dieses Verhalten eignet sich insbesondere die Analyse der räumlichen Fehlerverteilung [148].

Für die kooperative Indoorlokalisierung ist hierbei entscheidend, ob der jeweilige Algorithmus im Erwartungswert den Positionsfehler unterhalb des Messfehlers hält oder ob er darüber liegt. Letzteres führt zu einer starken Vergrößerung des Positionsfehlers nach wenigen Iterationen. Da in der kooperativen Indoorlokalisierung keine wesentlichen Annahmen über die Ankerpositionen getroffen werden können - Anker sind ebenfalls mobile Knoten - kann anders als im Beispiel von Abschnitt 4.2.4 dargestellt, nicht davon ausgegangen werden, dass sich ein Knoten immer in der Hülle der Anker befindet. Min-Max wäre hier also ein denkbar ungeeigneter Algorithmus. Für dieses Szenario ist ein Algorithmus mit einer möglichst homogenen räumlichen Verteilung des Fehlers geeigneter als einer mit einem niedrigeren Durchschnittsfehler hoher Varianz. Der Vergleich der räumlichen Fehlerverteilung von Min-Max und VBLE in Abbildung 4.1 zeigt deutlich die Unterschiede zwischen den beiden Algorithmen, obwohl der durchschnittliche Fehler bei beiden in etwa gleich ist.

4.2 Bewertungsmetriken

4.2.1 Rechenkomplexität

Wie für alle Computeralgorithmen bietet sich die Rechenkomplexität als eine Metrik zum Vergleich verschiedener Algorithmen an. Für eine hohe Gewichtung dieser Metrik



(a) Min-Max (b) VBLE-O

Abbildung 4.1: Vergleich der räumlichen Fehlerverteilung zwischen Min-Max und VBLE-O

spricht vor allem die in der Regel zur Indoorlokalisierung verwendete Hardware. Da der zu lokalisierende Knoten in der Regel direkt von einem Menschen getragen wird bzw. in dessen Kleidung oder Ausrüstung zu integrieren ist, spielen Eigenschaften wie Gewicht und Abmessungen oft eine zentrale Rolle für die Nutzung und Akzeptanz in der Praxis. Die für diese Zwecke verwendete Hardware kommt oft aus dem Bereich der mobilen Endgeräte oder gar aus dem Bereich der WSNs. Dementsprechend gering ist die Rechenleistung der einzelnen Knoten verglichen mit heutigen Desktop-Rechensystemen oder gar Großrechnern.

Als Kostenmetrik wird meistens die Anzahl der Anker herangezogen, in seltenen Fällen (VBLE, siehe 3.1.12) auch die Größe einer diskreten Umgebungsmatrix entsprechender Auflösung. Als Größe der Umgebung wird hier meistens die rechteckige Fläche angenommen mit der sich alle Anker umschreiben lassen, erweitert um die angenommene maximale Signalreichweite. Im Falle der Ankeranzahl als Metrik wird auch bei hoher Komplexität des Algorithmus die eigentliche Rechenzeit in vielen Fällen eine untergeordnete Rolle spielen. Es sind kaum Szenarien in der Literatur beschrieben wo eine mehr als zweistellige Anzahl von Ankern gleichzeitig empfangen werden konnte und auch auf minimaler Hardware stellt diese Ankeranzahl für die meisten Algorithmen keine wirkliche Herausforderung für den verwendeten Rechner dar. Allerdings kann die Rechenkomplexität für Anwendungen, die eine zeitlich in kurzer Periode wiederholte Lokalisierung benötigen - z. B. aufgrund von hoher Bewegungsgeschwindigkeit des Knotens - eine größere Rolle spielen. Für größere Simulationen oder das

zentrale Berechnen der Positionen gleich mehrerer Endgeräte spielt die unterschiedliche Rechenkomplexität der verschiedenen Algorithmen eine entscheidende Rolle. Je nach Einsatzszenario können Algorithmen mit einer Laufzeit von $\mathcal{O}(2^n)$ auch bei nur einer einzelnen Positionsberechnung schon eine zu lange Laufzeit haben. In vielen Szenarien wird analog zum GPS-System eine Updatefrequenz der Positionen von ein Mal pro Sekunde angestrebt. Für zweistellige Ankerzahlen ist dieses bei einigen der in Kapitel 3 getesten Algorithmen selbst auf leistungsfähigen Personalcomputern nicht mehr möglich. Für denkbare Anwendung in Szenarien wie Smart-Dust [63, 138] können selbst Laufzeiten von $\mathcal{O}(n^2)$, die einige der getesteten Algorithmen aufweisen, durchaus problematisch werden, da die verwendeten Rechner nochmals leistungsschwächer sind und die Anzahl der potenziellen Anker wesentlich höher sein kann. Für diese Probleme existieren Verfahren zur Auswahl einer kleineren Anzahl von Ankern aus der Menge der zur Verfügung stehenden Anker [20].

Algorithmen, die zur Positionsabschätzung eine diskrete Umgebungsmatrix verwenden, können aufgrund des quadratischen oder sogar kubischen (bei 3-D Lokalisierung) Wachstums der Matrix schnell in Bereiche geraten, bei denen die Rechenzeit nicht mehr tolerierbar erscheint. Diese Algorithmen lassen sich jedoch häufig derart optimieren, dass nur vorher abgeschätzte Teile der Matrix durchlaufen werden müssen und sich die eigentliche Komplexität dadurch wesentlich verringert. Oft stoßen derartige Algorithmen bereits vorher an die Speicherlimitierung der verwendeten Hardware als an die Grenzen der tolerierbaren Rechenzeit.

Eine weitere Möglichkeit mit hoher Rechenkomplexität umzugehen, kann es sein, die Messdaten an eine zentrale Senke zu übertragen und dort die Berechnungen durchführen zu lassen und ggf. das Ergebnis danach wieder an den Knoten zu übertragen. Gerade in Sensornetzen kommt diese Option unter Umständen schon vor erreichen der kritischen Rechenzeit aus energetischen Gründen in Betracht.

4.2.2 Speicherkomplexität

Für die Speicherkomplexität gelten im Wesentlichen die Aussagen zur Rechenkomplexität. Algorithmen deren Speicherkomplexität von der Anzahl der Anker abhängen, sind in der Regel völlig unkritisch, da die Anzahl der Anker in der Praxis stark limitiert ist und der pro Anker erforderliche Speicherbedarf sehr gering ist, verglichen mit der Menge an Speicher den das System zur Verfügung hat. Wesentlich gewich-

tiger als bei der Betrachtung der Rechenzeit fällt diese Metrik für Algorithmen aus, die eine diskrete Umgebungsmatrix zur Abschätzung der unbekannten Position benutzen. Je nach Granularität der Diskretisierung stoßen auch aktuelle Mobilrechner schnell an ihre Grenzen. So braucht der VBLE Algorithmus bei einer Gebäudegröße von 100x100 m^2 und einem Diskretisierungsintervall von 10 cm bereits 8 MB Speicher zur Berechnung eines einzigen Durchganges im 2-D-Fall. Für Sensornetze ist dieser Wert bereits weit über dem gesamt verfügbaren Speicher aktuell eingesetzter Hardware [94]. Auch das Auslagern der Berechnungen auf eine zentrale Senke kann bei derartigen Algorithmen, je nach Anzahl der Knoten, einen erhöhten Hardwarebedarf nach sich ziehen.

Ebenso wie bei der Rechenkomplexität gilt jedoch auch hier, dass sich die Mehrheit solcher Algorithmen, auch in Hinblick auf ihre Speicherkomplexität, mit gängigen Verfahren erheblich optimieren lassen.

4.2.3 Statistische Betrachtungen der Genauigkeit als Bewertungsmetrik

Vorüberlegungen

Um eine statistische Aussage über die Güte eines Lokalisierungssystems zu treffen, kann über die Auswertung einer bestimmten Menge von Lokalisierungen versucht werden, eine Aussage über das Verfahren an sich zu treffen. Die Güte definieren wir im Allgemeinen als erzielte Genauigkeit der Lokalisierung. Hierbei lässt sich eine absolute Angabe machen oder eine relative, die den resultierenden Fehler mit dem Fehler des Eingabesystems ins Verhältnis setzt. Vergleicht man alle Verfahren anhand desselben Eingabesystems, sind beide Ansätze somit äquivalent. Die Menge der Lokalisierungen kann man als Stichprobe betrachten und den Lokalisierungsalgorithmus demzufolge als Schätzer. Analog gilt das auch für das zugrunde liegende System zur Bestimmung der Entfernungen.

Ein Schätzer versucht also mithilfe einer Stichprobe, die als eine Reihe von Zufallsvariablen

$$X_1, \ldots, X_n$$

betrachtet werden kann, auf eine Eigenschaft γ aus der Grundgesamtheit zu schließen. Der Schätzer besteht dabei aus einer Schätzfunktion f, die die Schätzung $f[X_1, \ldots, X_n]$ abgibt. Der Schätzer f kann dabei wiederum als Zufallsvariable

betrachtet werden.

Die Qualität eines Schätzers beurteilt man üblicherweise danach, inwieweit der Schätzer aus der gegebenen Stichprobe auf Eigenschaften der Grundgesamtheit schließen kann. Dabei gibt es verschiedene Metriken, um die Abweichungen von der Grundgesamtheit zu bewerten. Im Wesentlichen gibt es zwei Arten von Schätzern, bei denen die einzelnen Bewertungsmetriken grundsätzlich verschiedene Aussagen haben bzw. unterschiedlich zu betrachten sind.

Die erste Art von Schätzern sind sog. erwartungstreue Schätzer. Bei erwartungstreuen Schätzern ist der Erwartungswert des Schätzers gleich der unbekannten Eigenschaft γ. Wollen wir mit einem solchen Schätzer beispielsweise den Erwartungswert der Grundgesamtheit γ schätzen und nehmen als Schätzfunktion f das Stichprobenmittel μ, so ist die Schätzfunktion als

$$f(X_1, \ldots, X_n) = \overline{X}_n = \frac{1}{n} \sum_{i=1}^{n} X_i$$

definiert. Der Erwartungswert dieser Schätzung ist also

$$E(\overline{X}_n) = E\left(\frac{1}{n} \sum_{i=1}^{n} X_i\right) = \frac{1}{n} \sum_{i=1}^{n} E(X_i) = \frac{1}{n} \cdot n \cdot \mu = \mu$$

sofern die zugrunde liegende Stichprobe zufällig aus der Grundgesamtheit gezogen wurde und der Erwartungswert einer Ziehung also als

$$E(X_i) = \mu$$

beschrieben ist. Somit ist die Schätzung des Erwartungswertes einer Grundgesamtheit mithilfe des Stichprobenmittels auf einer zufälligen Stichprobe ein erwartungstreuer Schätzer.

Neben erwartungstreuen Schätzern gibt es demzufolge auch Schätzer, die diese Eigenschaft nicht erfüllen - diese werden auch als verzerrt bezeichnet. Wichtig ist hierbei zu beachten, dass immer nur der Schätzer, der aus Schätzfunktion f und der Stichprobe $[X_1, \ldots, X_n]$ besteht und nicht die Schätzfunktion alleine erwartungstreu oder verzerrt sein kann. Eine Schätzfunktion kann also bei einer bestimmten Stichprobe erwartungstreu sein und bei einer anderen hingegen nicht.

Für die Indoorlokalisierung können wir den Algorithmus als einen Schätzer anse-

hen, der versucht die unbekannte Position eines Knotens abzuschätzen. Ein solcher Algorithmus ist also genau für solche Fälle als erwartungstreu anzusehen, in denen der Erwartungswert der gesuchten Position entspricht. Dies kann wieder über ein Stichprobenmittel getestet werden. Wenn wir also bei konstanter Position γ und variierendem Eingangsfehler wiederholt den Algorithmus anwenden, ist zu prüfen, ob der Erwartungswert dieser Stichprobe genau der Position γ entspricht.

Um den Erwartungswert einer Stichprobe aus zweidimensionalen Koordinaten zu berechnen, kann man die x- und y-Werte jeweils getrennt betrachten und den Mittelwert berechnen. Dieses Verfahren entspricht dem gewichteten Schwerpunkt aller Punkte der Menge, wobei alle Punkte das Gewicht Eins bekommen. Sofern dieser Erwartungswert der zu schätzenden Position entspricht, ist der Algorithmus **in dieser Konstellation** erwartungstreu.

Für erwartungstreue Schätzer ist das Maß zu Beurteilung eines Schätzers die Effizienz, die der Varianz entspricht. Dabei gilt eine kleinere Varianz als effektiver als eine größere Varianz.

Für nicht erwartungstreue Schätzer ist die Effizienz nicht über die Varianz zu beschreiben - ein nicht erwartungstreuer Schätzer mit einer kleineren Varianz als die eines erwartungstreuer Schätzers kann trotzdem weniger Effektiv sein!

Da ein und derselbe Algorithmus je nach Ankerkonstellation und Eingabefehler mal erwartungstreu und mal verzerrt sein kann, ist die Varianz ein schlechtes Kriterium zum Vergleich der Qualität eines solchen Algorithmus. Bei nicht erwartungstreuen Schätzern gibt die Varianz nur das Maß der Streuung um den Erwartungswert an, der aber eben nicht der gesuchten Position entspricht. Die Differenz zwischen dem Erwartungswert der Stichprobe eines nicht erwartungstreuen Schätzers und dem wahren Wert nennt man *Bias* oder Verzerrtheit:

$$\mathrm{Bias}(f(X)) = \mathrm{E}[f(X) - \gamma].$$

Das diese beiden Metriken nur eine geringe Aussagekraft haben lässt sich an einem Beispiel erklären: Wir nehmen einen Schätzer an, dessen Positionsschätzung immer genau auf einem Kreis mit Radius r liegt, dessen Mittelpunkt dem wahren Wert γ der zu schätzenden Variablen entspricht. Im Erwartungswert würde dieser Schätzer also immer genau die gesuchte Position treffen, obwohl jede einzelne Schätzung genau

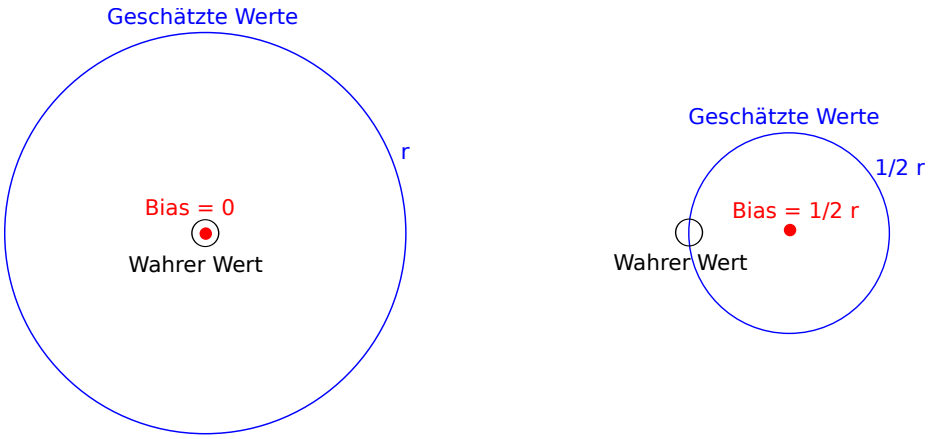


Abbildung 4.2: Beispiel für den Unterschied zwischen erwartungstreuen und nicht erwartungstreuen Schätzern in der Indoorlokalisierung. Links ist der erwartungstreue Schätzer abgebildet und rechts der nicht erwartungstreue Schätzer.

einen Fehler von r aufweist und der gesuchte Wert niemals getroffen wird. Die Varianz dieses Schätzers wäre somit r^2 und der *Bias* 0. Ein weiterer Schätzer möge wiederum auf einen Kreis mit dem Radius

$$\frac{r}{2}$$

und dem Mittelpunkt

$$\vec{\gamma} + \begin{pmatrix} \frac{r}{2} \\ 0 \end{pmatrix}$$

schätzen. Dieser hat demzufolge einen *Bias* von

$$\begin{pmatrix} \frac{r}{2} \\ 0 \end{pmatrix}$$

und eine Varianz von

$$\frac{r^2}{4}.$$

Es sind anhand dieser Parameter nur sehr schwer Aussagen über die Performance der beiden Algorithmen zu treffen (siehe Abbildung 4.2).

Eine weitere sehr problematische Metrik ist es, vom Erwartungswert des Schätzers den wahren Wert γ abzuziehen. Das so entstehende Residuum hat in Anwendungsszenarien, in denen der Fehler eine Richtung hat, eine sehr verminderte Aussage, da sie nur den Betrag des *Bias* darstellt und keinerlei Aussagen über die Varianz enthält. Bezugnehmend auf das erste Beispiel des vorherigen Absatzes (siehe Abbildung 4.2) gleichen also Schätzungen die im Mittel um den Betrag r daneben liegen, Schät-

zungen vollständig aus, die auf dem Kreis genau gegenüber liegen und ebenfalls den Betrag r aufweisen. Bei erwartungstreuen Schätzern könnte man sich diese Beobachtung zunutze machen, indem man jede Position häufiger schätzen würde und dann den Mittelwert als endgültige Schätzung ausgeben würde. Da aber so gut wie alle Algorithmen nicht in jeder Konstellation erwartungstreu sind (siehe Kapitel 6) und in der Praxis sich die zu lokalisierenden Ziele in der Regel bewegen, lässt sich hieraus kaum ein Nutzen ziehen und diese Metriken sind für einen Performancevergleich in der Regel ungeeignet.

Geeignetere Metriken treffen also keine alleinige Aussage über das Ergebnis des Schätzers und dessen Verteilung, sondern über den Fehler des Schätzers und dessen Verteilung.

Statistische Bewertungsmetriken

Eine der gängigsten Metriken für die Vergleichbarkeit von Systemen zur Indoorlokalisierung stellt sicherlich die Betrachtung des durchschnittlichen Fehlers (MAE) dar. Neben der bekannten Problematik, dass der Durchschnitt nichts über die Größe und Anzahl von Ausreißern aussagt, ist dieser Wert sehr stark von der Art und Weise der Erhebung abhängig.

Der MAE in der Indoorlokalisierung ist definiert als:

$$\mathrm{MAE}(f, \gamma) = \frac{1}{n} \sum_{1}^{n} \|f(X) - \gamma\|$$

wobei γ die wahre Position des Knotens ist und $f(x)$ dem Schätzer für die Zufallsvariable entspricht. Der Betrag der Differenz ist dabei als der euklidische Abstand definiert. $\|f(X) - \gamma\|$ ist also der Positionsfehler, der auch als Residuum bezeichnet wird.

Die rein numerischen Durchschnittsfehler verschiedener Algorithmen lassen sich nur vergleichen, wenn auch das zugrunde liegende Experiment oder die zugrunde liegende Simulation exakt die gleichen sind.

So hängt bei allen Algorithmen der resultierende Positionsfehler mindestens genauso stark vom Eingangsfehler der Entfernungsmessungen, wie auch der Positionen der Ankerknoten und des unbekannten Knotens, als auch von Algorithmus selber ab. Für einen Vergleich verschiedener aus der Literatur bekannter Algorithmen müssen also

alle zu vergleichenden Algorithmen implementiert und unter gleichen Bedingungen gebenchmarkt werden. Für Experimente stellt das einen erheblichen Aufwand dar, da es oft nicht mal annäherungsweise möglich ist, ein Experiment unter wenigstens ähnlichen Bedingungen zu wiederholen. So kann schon der zufällige Aufenthalt einer Person im Nachbarzimmer der Experimentierumgebung zu völlig anderen Ergebnissen führen.

Aus diesem Grunde ist es notwendig, für einen Vergleich verschiedener Algorithmen, die Algorithmen unter exakt gleichen Bedingungen zu simulieren und anschließend die Simulationsergebnisse in der Praxis mittels eines Experimentes zu verifizieren. Verhalten sich einzelne Algorithmen unter Simulationsbedingungen und im Experiment widersprüchlich, ist dieses auf einen Fehler im Simulationsmodell oder nicht berücksichtigte Einflüsse der Experimentierumgebung zurückzuführen. Eine weitere Möglichkeit besteht darin, in einem Experiment nur die Entfernungen zu den Ankerknoten zu messen und auf diesen Werten die verschiedenen Algorithmen rechnen zu lassen. So ist eine Reproduzierbarkeit auch für zukünftige Algorithmen gegeben.

Will man die Durchschnittsfehler verschiedener Algorithmen und verschiedener Experimente vergleichen, so bietet es sich an den Durchschnittsfehler zum Betrag des durchschnittlichen Eingabefehlers (hier der Fehler in den Distanzmessungen) des jeweiligen Experimentes ins Verhältnis zu setzen (siehe auch Kapitel 4.2.4).

Eine ebenfalls gängige Metrik zum Vergleich und zur Bewertung von Lokalisierungsalgorithmen sind statistische Mittelwerte, die das Vorhandensein von Ausreißern stärker berücksichtigen als der bloße Durchschnitt. In der Literatur haben sich hier der Mean Squared Error (MSE) bzw. der RMSE als praktikabel erwiesen.

Der MSE ist in der Lokalisierung definiert als:

$$\mathrm{MSE}(f, \gamma) = \mathrm{E}[\|f(X) - \gamma\|^2]$$

wobei γ die wahre Position des Knotens ist und $f(x)$ dem Schätzer für die Zufallsvariable entspricht. Der Betrag der Differenz ist dabei als der euklidische Abstand definiert. $\|f(X) - \gamma\|$ ist also der Positionsfehler, der auch als Residuum bezeichnet wird. Da diese Metrik nicht auf der durchschnittlichen Schätzung, sondern auf dem durchschnittlichen Schätzfehler beruht, ist diese Metrik deutlich besser geeignet als die Varianz, die nur für erwartungstreue Schätzer eine sinnvolle Aussage hat.

Der RMSE ist somit definiert als:

$$\mathrm{RMSE}(f, \gamma) = \sqrt{\mathrm{E}[\|f(X) - \gamma\|^2]}$$

Wichtig ist zu beachten, dass der RMSE immer nur in Relation zu den Residuen selber zu sehen ist und nicht als Absolutwert.

Ein kleiner RMSE weist im Fall der Indoorlokalisierung, wo wir in der Regel von nicht erwartungstreuen Schätzern ausgehen müssen, auf einen niedrige Verzerrtheit und eine geringe Varianz der Stichproben hin. Ein großer RMSE ist ohne die Kenntnis der Varianz jedoch schwer zu interpretieren und ist somit als Bewertungsmetrik alleine nicht zu gebrauchen. Zum Vergleich verschiedener Algorithmen unter konstanten Bedingungen stellt der RMSE jedoch eine brauchbare Metrik dar [154].

Kritik an der Verwendung des RMSE wird durch James Berger [10] geäußert. Berger weist darauf hin, dass der Betrag des RMSE vom abzuschätzenden Grundgesamtheitsparameter abhängt und somit als Benchmarkmetrik nur bedingt geeignet ist. Kritisiert wird außerdem, dass der RMSE betragsmäßig große Ausreißer durch die Quadrierung wesentlich stärker gewichtet als betragsmäßig kleinere [11]. Für die Anwendung in der Indoorlokalisierung kann dieses Verhalten auch eine durchaus gewünschte Metrik darstellen (siehe auch Kapitel 4.2.3).

Hat man die Stichprobenvarianz des Schätzers berechnet, kann man mithilfe des MSE die Verzerrtheit des Schätzers berechnen, denn es gilt:

$$\mathrm{MSE}(f, \gamma) = \mathrm{Bias}^2(f(X)) + \mathrm{Var}(f(X))$$

Woraus folgt, dass die Verzerrung folgendermaßen definiert werden kann:

$$\mathrm{Bias}^2(f(X)) = \mathrm{MSE}(f, \gamma) - \mathrm{Var}(f(X))$$

woraus sich wiederum ableiten lässt, dass ein Schätzer genau dann erwartungstreu ist, wenn keine Verzerrung vorhanden ist und somit MSE und Varianz gleich sind. Weiterhin zu beachten ist hier, dass zwischen der Varianz des Schätzers und der Varianz der Residuen des Schätzers ein erheblicher Unterschied in der Aussage besteht und diese nicht zu vermischen sind! So hatten wir im grundlegenden Beispiel des vorherigen Abschnittes (siehe Abbildung 4.2) eine Varianz von r^2 beobachtet. Da aber alle Messwerte die gleichen Residuen aufweisen, nämlich r, beträgt deren Varianz 0.

Maximaler Fehler

Eine garantierbare obere Schranke ist für diverse Einsatzbereiche, vor allem aus dem Sicherheitsbereich, eine Forderung, an der viele Praxiseinsätze von Indoorlokalisierung heutzutage noch scheitern [150]. Wie bereits dargestellt, hängt die Genauigkeit nicht nur von den Beschaffenheiten des technischen Systems ab, sondern kann auch massiv von der eingesetzten Umgebung abhängen. Ist diese Umgebung nicht Teil der Spezifikation, geht diese als Unbekannte in alle Abschätzungen ein und eine obere Schranke für den Fehler kann nicht garantiert werden. Es gibt jedoch Ansätze die Umgebung, in der das System zum Einsatz kommen soll, genau zu vermessen und aus den Vermessungsdaten ein hoch präzises Modell mittels Raytracings zu errechnen [71]. In diesen Fällen können - abhängig von der Güte des Modells - auch obere Schranken für den Fehler garantiert werden.

Im Allgemeinen ist die Garantie harter Schranken aufgrund der nur schwer zu kontrollierenden Umgebungsparameter allerdings kaum möglich. So können kleinste Anomalien in einem Gebäude, wie z. B. ein in eine Wand eingebautes Stahlteil oder ähnliches, größere Ausreißer bei der Genauigkeit der zugrunde liegenden Entfernungsmessung zur Folge haben. Wesentlich praxisnäher dürfte von daher die Betrachtung eines hohen Quantiles wie zum Beispiel des 95 % Quantils sein.

4.2.4 Räumliche Fehlerverteilung

Neben der rein statistischen Fehlerverteilung mittels Mittelwert und Streuung ist in der Indoorlokalisierung die Betrachtung der räumliche Fehlerverteilung oft aussagekräftiger. So kann es vorkommen, dass zwei Algorithmen einen identischen Durchschnittsfehler und eine ähnliche Stichprobenvarianz aufweisen, die räumliche Fehlerverteilung aber einen Algorithmus als sehr geeignet und den anderen Algorithmus als ungeeignet für ein gegebenes Szenario erscheinen lässt.

Zur Erhebung der räumlichen Fehlerverteilung wird das Betrachtungsgebiet diskretisiert und für jeden diskreten Punkt in diesem Gebiet der Positionsfehler erhoben. Je nach Algorithmus und Art der Erhebung, bietet sich eine mehrfache Erhebung mit anschließender Bildung des Mittelwertes an, bis der Wert konvergiert. Nach dieser Erhebung lässt sich die räumliche Fehlerverteilung grafisch darstellen und bewerten [148, 53].

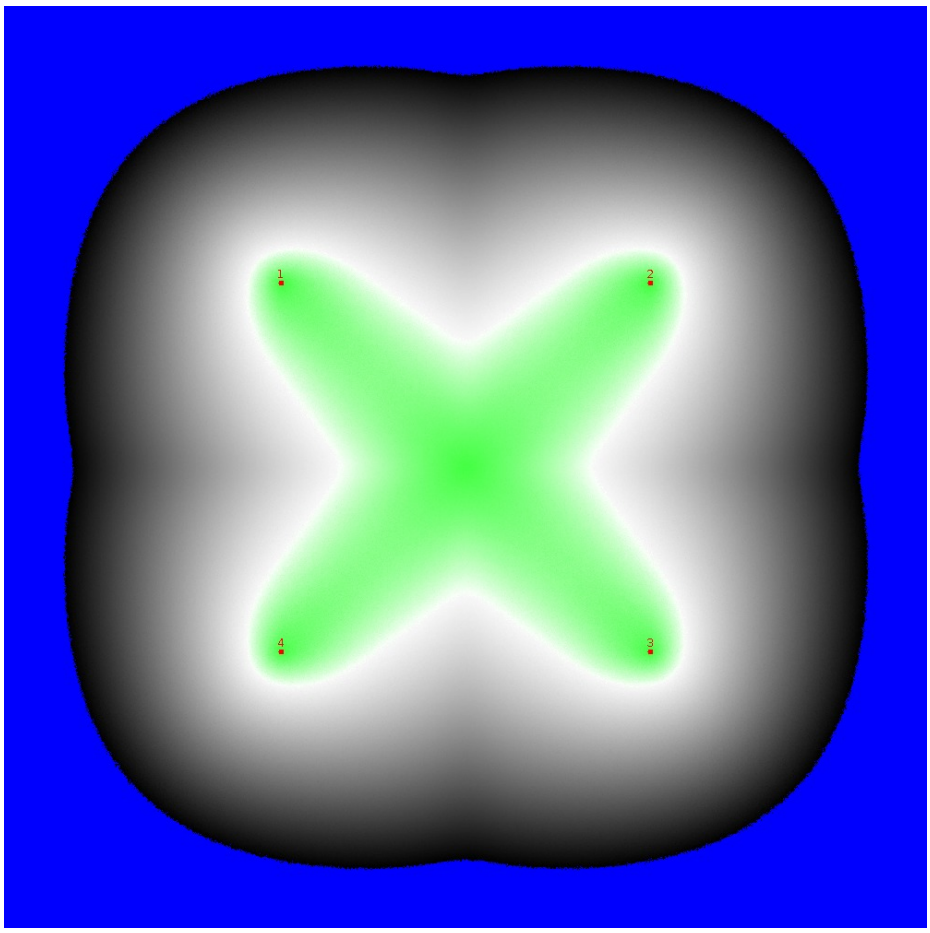


Abbildung 4.3: Beispiel für die Darstellung der räumlichen Fehlerverteilung

Abbildung 4.3 zeigt Beispielhaft die Analyse des Min-Max Algorithmus (siehe Kapitel 3.1.1) für ein Setup aus vier Ankernoten und einem normalverteilten Distanzfehler. Die Farbe der einzelnen Bildpunkte repräsentiert dabei den durchschnittlichen Positionsfehler des jeweiligen Algorithmus nach 1000 Durchläufen. Für grau eingefärbte Bereiche gilt: der Positionsfehler liegt zwischen 100 % und 250 % des mittleren Distanzmessfehlers; je dunkler die Farbe, desto größer ist dabei der Fehler. Grün eingefärbte Bereiche kennzeichnen einen Positionsfehler von weniger als 100 % des Distanzfehlers - hier ist der resultierende Fehler also geringer als der Erwartungswert des Eingabefehlers. Blaue Bereiche weisen einen größeren Fehler als 250 % des erwarteten Distanzfehlers auf und haben aus Gründen des Bildkontrastes keinen Gradienten.

Am gezeigten Beispiel kann man sehen, dass die bloße Analyse des durchschnittlichen Positionsfehlers für den Min-Max Algorithmus nur eine begrenzte Aussagekraft hat. Ein Betrachten der Varianz würde zwar auf eine große Streuung der Positions-

fehler hinweisen, würde aber nichts über die räumlichen Zusammenhänge der Fehler aussagen. In der räumlichen Fehlerverteilung lässt sich erkennen, dass der Min-Max Algorithmus innerhalb der konvexen Hülle der Anker sehr gute Ergebnisse liefert und außerhalb der konvexen Hülle stark abfällt und bereits bei geringer Entfernung zu dieser einen sehr großen Fehler aufweist. Auf den Diagonalen zwischen den im Quadrat positionierten Ankern liefert Min-Max sogar außergewöhnlich gute Ergebnisse. Für den Min-Max Algorithmus ist dies auch in der Analyse des Algorithmus sehr einfach zu begründen, da dieser Algorithmus konzeptbedingt nur Positionen zurückgeben kann, die innerhalb der konvexen Hülle aller Ankerpunkte liegen.

Die Analyse der räumlichen Fehlerverteilung zeigt auf den ersten Blick Stärken und Schwächen eines Algorithmus auf und erklärt oft das Streuungsverhalten eines bestimmten Algorithmus in der Auswertung eines Experimentes oder einer Simulation.

Insbesondere zur Beurteilung von Systemen in der Indoorlokalisierung ist diese Metrik ein sehr wertvolles Hilfsmittel, da sie mehrere andere Metriken in einer Übersicht zusammenfasst und spezielle Eigenschaften eines Algorithmus, die nur innerhalb von Gebäuden wirklich relevant sind, sichtbar macht. Diese Eigenschaften hängen insbesondere mit Fragen der Ankerplatzierung und des für eine Lokalisierung relevanten Bereiches zusammen. In der Indoorlokalisierung werden die Anker in der Regel nicht frei platziert und beispielsweise gleichverteilt in der gesamten Fläche ausgebracht. Vielmehr muss bei der Ankerplatzierung, aber auch bei den erwarteten Positionen des zu lokalisierenden Knotens, auf die spezifischen Eigenschaften des Gebäudes Rücksicht genommen werden. In vielen Fällen werden Anker direkt an oder in der Nähe von Wänden platziert, da andere Platzierungen die weitere Nutzung des Gebäudes einschränken würden. So wäre für den Min-Max Algorithmus beispielsweise unter diesen Bedingungen sichergestellt, dass ein Verlassen der konvexen Hülle der Anker ausgeschlossen ist. In diesem Fall würde sich Min-Max nach der Analyse der räumlichen Fehlerverteilung, trotz des sehr großen durchschnittlichen Fehlers, als geeignet erweisen.

Bei der Entwicklung von Algorithmen ist die Analyse der räumlichen Fehlerverteilung eine sehr wertvolle Metrik um den Fortschritt zwischen zwei verschiedenen Versionen eines Algorithmus zu beurteilen oder den Fokus bei der Entwicklung gezielt auf bestimmte Bereiche zu legen. Für das obige Beispiel hatte die Analyse der räumlichen Fehlerverteilung von Min-Max zur Folge, dass ein erweiterter Min-Max Algorithmus (siehe Kapitel 3.3) mit wesentlich besseren Eigenschaften jenseits der Diagonalen zwi-

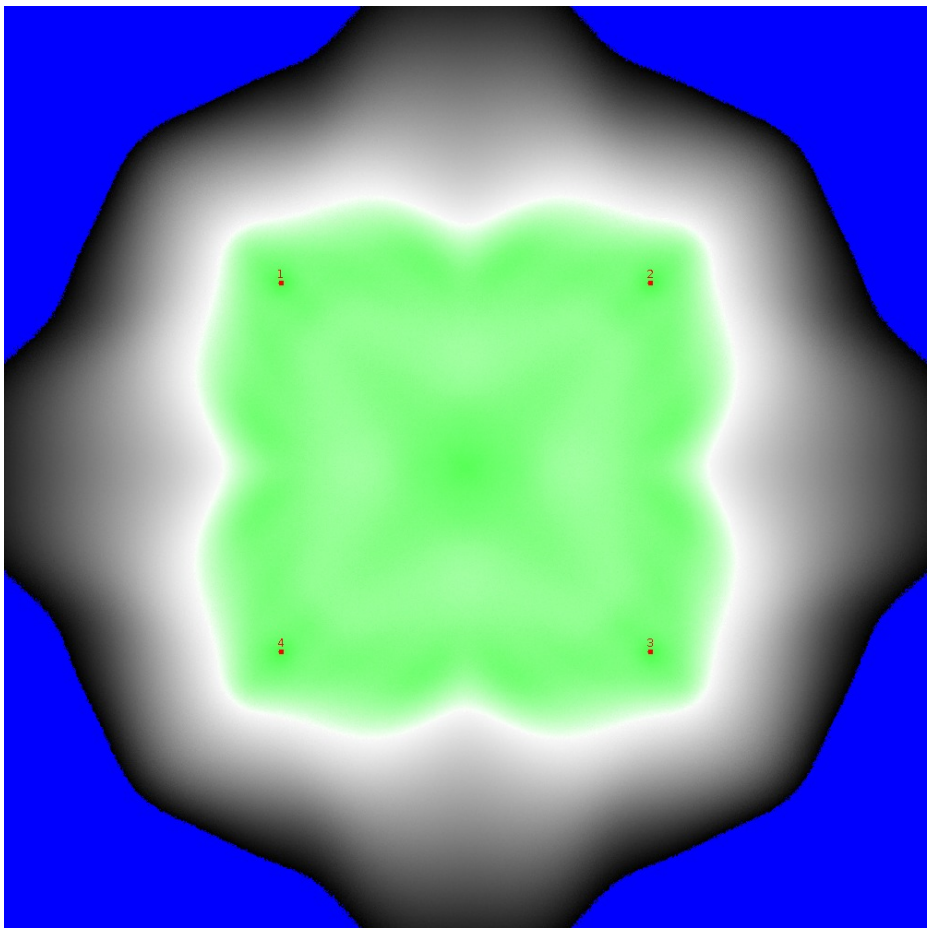


Abbildung 4.4: Verbesserung von Min-Max nach Analyse der räumlichen Fehlerverteilung

schen den Ankern entwickelt werden konnte (siehe Abbildung 4.4).

Nachteil dieser Metrik ist, dass die Erhebung extrem rechenaufwändig werden kann und die Analyse nicht immer offensichtlich ist. Eine numerische Vergleichbarkeit wie bei Durchschnittsfehler und Varianz ist nicht gegeben. Da die Positionen für die Erhebung der Stichproben genau bekannt sein müssen und die Experimente bis zur Konvergenz der Durchschnitte wiederholt werden sollten, bietet sich hauptsächlich eine Simulation für die Erhebung der Stichproben an. In Kapitel 5.1.2 stellen wir ein von uns entwickeltes Werkzeug vor, mithilfe dessen die räumliche Fehlerverteilung von Lokalisierungsalgorithmen effizient berechnet und grafisch dargestellt werden kann.

4.2.5 Analyse der Cramér-Rao-Ungleichung

In vielen Veröffentlichungen zum Thema Indoorlokalisierung wird anstelle einer Analyse der räumlichen Fehlerverteilung ein Vergleich mit den Ergebnissen der CRLB geliefert. Die CRLB stellt für alle erwartungstreuen Schätzer eine untere Schranke für die Varianz und somit die Effizienz dar. Bei einer Nutzung der CRLB im Bereich der Indoorlokalisierung ergeben sich unserer Ansicht nach drei wesentliche Probleme:

1. Das wesentliche Problem bei der Nutzung der CRLB ist es, dass die CRLB, wie sie in der Literatur angewendet wird, das Rauschen der Eingabeparameter als einziges Kriterium für die Berechnung einer Schranke auf den Lokalisierungsfehler verwendet. Wie wir mit der Analyse der räumlichen Fehlerverteilung gezeigt haben, ist die Geometrie aus Ankern und unbekannten Knoten eine viel stärkere Fehlerquelle, deren Einfluss bei der Anwendung der CRLB nicht richtig berücksichtigt wird. Daher wird die CRLB von diversen Algorithmen unter bestimmten geometrischen Konstellationen regelmäßig unterschritten und taugt somit nicht als Schranke [27, 28].
2. So gut wie alle von uns evaluierten Algorithmen sind keine erwartungstreuen Schätzer und die CRLB stellt für diese Klasse von Schätzern ausdrücklich keine untere Schranke für die Effizienz dar. Für nicht erwartungstreue Schätzer lässt sich mithilfe der CRLB eine untere Schranke für den RMSE berechnen. Die dazu nötigen Ableitungen lassen sich nicht für alle Algorithmen und Fehlermodelle berechnen.
3. Ein drittes Problem mit der CRLB ist, dass sie nur eine Aussage über Schätzer trifft, die exakt diese Schranke treffen. Über Schätzer, die von der Schranke entfernt sind, ist kaum eine Aussage möglich, da nicht gesagt werden kann, ob überhaupt ein Schätzer existiert, der die Schranke erreichen kann.

Aus diesen Gründen werden wir unseren folgenden Evaluierungen nicht auf die CRLB eingehen und uns auf die Analyse und den direkten Vergleich der räumlichen Verteilung beschränken.

4.3 Fazit

Die in diesem Kapitel vorgestellten Metriken eignen sich unter bestimmten Voraussetzungen und Zielsetzungen alle für die Bewertung und den Vergleich verschiedener Algorithmen und Verfahren. So macht eine Bewertung der Speicher- und Rechenkomplexität Sinn, wenn für einen Einsatz in der Praxis konkrete Hardwarelimitierungen vorliegen oder eine bestimmte Akkulaufzeit garantiert werden soll. Auch für größere Simulationen bestimmter Szenarien kann die Rechenzeit ein entscheidender Faktor sein, der über die Eignung eines Algorithmus entscheiden kann. Ein in der Praxis wesentlich gewichtigerer Faktor ist die zu erwartende Genauigkeit der durch einen Algorithmus geschätzten Position. Leider ist die Genauigkeit nicht nur vom Algorithmus selber, sondern vielmehr von der Fähigkeit des Algorithmus mit bestimmten Umgebungsparametern umzugehen, abhängig. Will man verschiedene Algorithmen in einem festgelegten Szenario experimentell vergleichen, so bietet sich je nach Aussage eine statistische Metrik wie der MAE an. Die Aussagen aus einem solchen Szenario lassen sich allerdings nicht ohne Weiteres auf die Allgemeinheit übertragen - in anderen Szenarien könnten die Ergebnisse konträr ausfallen.

Mit der Vorstellung der Analyse der räumlichen Fehlerverteilung haben wir eine Metrik eingeführt, die allgemeinere Aussagen zulässt und simulativ ermittelt werden kann. Insbesondere der Vergleich von bestimmten Algorithmen für bestimmte wohldefinierte Szenarien und Einsatzgebiete lässt sich durch diese Metrik wesentlich einfacher und objektiver durchführen.

KAPITEL 5

Datenerhebung mittels Simulationen und Experimenten

Sind die für die Bewertung verschiedener Ansätze zu nutzenden Bewertungsmetriken festgelegt, stellt sich als nächstes die Frage der Datenerhebung zum Gewinn dieser Metriken. Grundsätzlich werden in der Informatik neben rein mathematisch formalistischen Verfahren als wesentliche Mittel zur Datenerhebung die Simulation oder das Experiment eingesetzt. Für die distanzbasierte Indoorlokalisierung spielen beide Methoden eine wichtige Rolle, bei deren Einsatz jedoch diverse Fallstricke und Fehlerquellen zu beachten sind.

5.1 Simulationen

Um die Eignung der in Kapitel 3 vorgestellten Algorithmen für die Indoorlokalisierung zu überprüfen und die verschiedenen Algorithmen untereinander zu vergleichen, haben wir verschiedene Simulationen durchgeführt. Bei der Auswahl der Szenarien haben wir besonderen Wert darauf gelegt, solche Szenarien zu wählen, aus denen sich möglichst aussagekräftige Rückschlüsse auf einen Einsatz in der Praxis ziehen lassen. Wir diskutieren ausführlich die spezifischen räumlichen Fehlerverteilungen und zeigen, anhand aus der Praxis abgeleiteter Spezifika, die Stärken und Schwächen der jeweiligen Algorithmen auf.

Wir geben jedoch zu beachten, dass die Aussagekraft von Simulationen generell kritisch zu hinterfragen ist. Gerade im Bereich der Signalausbreitung können schon

kleinste Änderungen von Parametern einen sehr großen Einfluss haben. Durch die bloße Positionierung der Ankerknoten kann ein Algorithmus von einem niedrigen zu einen extrem hohen MAE verschoben werden. Da die Anzahl der Simulationen die hier und auch in der Literatur präsentiert werden können jedoch endlich ist, kann eine Auswahl der Parameter immer nur nach subjektiven Kriterien geschehen. Wären Simulationsergebnisse Objektiv, wäre das Finden eines optimalen Algorithmus deutlich leichter: Da alle neuen Algorithmen aus der Literatur in einer Simulation bewiesen haben, besser zu performen als die bestehenden Algorithmen und die Korrektheit der jeweiligen Methodik durch fachkundige Peer-Reviewer bestätigt wurde, sollte eine einfache Sortierung nach Veröffentlichungsdatum genügen, um den besten Algorithmus zu finden. Die Erfahrung der meisten Wissenschaftler und auch unsere im Folgenden präsentierten Simulationen und Experimente werden jedoch zeigen, dass diese einfache Methode natürlich nicht zielführend ist.

5.1.1 Verschiedene Simulationsansätze in der Literatur

In der Literatur werden eine Vielzahl verschiedener Ansätze zum Simulieren und Evaluieren von Algorithmen aus der Indoorlokalisierung verwendet. Die wesentlichen Parameter einer solchen Simulation sind: Das Fehlermodell für die Distanzen, aus denen die Position des unbekannten Knotens berechnet werden soll, die Positionen der Ankerknoten, das Bewegungsmodell, aus dem sich die Position des Knotens ableitet und schließlich die Menge der Ankerknoten.

Wir hatten vor im Rahmen dieser Arbeit alle uns vorliegenden Veröffentlichungen zur Indoorlokalisierung nach diesen Parametern auszuwerten und jeweils nach bestimmten Gemeinsamkeiten zu gruppieren. Leider sind in der überwiegenden Mehrzahl der Fälle die Simulation in keinster Weise vergleichbar bzw. genug Parameter für eine Reproduzierbarkeit der Ergebnisse angegeben. Die Auswahl der Szenarien, insbesondere der Anzahl und Position der Ankerknoten, erscheint oft willkürlich bzw. darauf bedacht, ein bestimmtes Ergebnis zu erzielen. Über die Geometrie der Ankerknoten, die einen mitunter größeren Einfluss auf den resultierenden Fehler als der Eingabefehler hat, wird nur in wenigen Ausnahmen überhaupt eingegangen. Auch die Algorithmen die zum Vergleich herangezogen werden, scheinen oft willkürlich ausgewählt bzw. ebenfalls um ein bestimmtes Ergebnis zu erzielen. So haben wir Simulationen evaluiert, in der eine große Anzahl von Lokalisierungen außerhalb der konvexen Hülle der Anker stattfand und trotzdem Min-Max als Vergleichsalgorithmus gewählt wurde -

obwohl dieser Algorithmus unter diesen Bedingungen bekanntlich keine Lokalisierung ermöglicht.

Auffallend ist, dass sich eine ganze Reihe von Veröffentlichungen mit dem Szenario eines Angreifers auf ein Lokalisierungssystem auseinandersetzt. Ein Angreifer übermittelt dem zu lokalisierenden Knoten entweder eine falsche Entfernung oder eine falsche eigene Position. Wir haben dieses Szenario der falschen Entfernungen mit dem der NLOS-Fehler und im Falle der falschen Positionen mit dem Fall der kooperativen Lokalisierung gleichgesetzt. In beiden Szenarien treten diese Fehler ohne das Vorhandensein eines Angreifers auf und lassen die Simulation dieser Fehlerquellen als Zufallsverteilung realistisch erscheinen. In den betreffenden Papern werden diese Fehlerquellen ebenfalls als zufallsverteilt angenommen und eine Motivation warum ein Angreifer zufällige Fehler einstreuen sollte, obwohl er ein bestimmtes Ziel verfolgt und über erhebliches Insiderwissen zur Durchführung dieser Angriffe verfügen muss, wird nicht geliefert.

Von den über 30 ausgewerteten Simulationen konnten wir nur die Ergebnisse von Zweien reproduzieren. Die wenigen Autoren, die überhaupt genug Parameter veröffentlicht haben um eine Reproduktion theoretisch zu erlauben, lieferten in unseren Versuchen meist andere Ergebnisse als in der jeweiligen Veröffentlichung. Ebenfalls ist auffällig, dass in nur sehr wenigen Fällen das Tool mit dem simuliert wurde, überhaupt genannt wurde - in allen Fällen wo dies geschah, handelte es sich um MatLab. Die überwiegende Mehrheit der Simulationen wurde mit unbekannten, oder mit selbst programmierten Tools durchgeführt, die nicht zusammen mit dem jeweiligen Paper veröffentlicht wurden. Auch haben die Autoren der jeweiligen Algorithmen in keinem uns bekannten Fall eine Version ihres Algorithmus im Quelltext zur weiteren Überprüfung veröffentlicht. Dieser Fakt führt ebenfalls dazu, dass als Vergleichsalgorithmen oft nur besonders simple und einfach zu implementierende Algorithmen herangezogen werden und nicht der jeweils aktuelle Stand der Forschung. Unserer Auffassung von wissenschaftlichen Veröffentlichungen nach hätte eine Mehrzahl der von uns ausgewerteten Veröffentlichungen ein Peer-Review nicht bestehen dürfen, da eine Beurteilung über den Grad der Neuerung mit den vorhandenen Angaben und den verglichenen Algorithmen schlicht und einfach nicht möglich ist.

Nachfolgend präsentieren wir eine Auswahl der Simulationen die einer Veröffentlichung zugrunde liegen, in der ein Algorithmus vorgestellt wurde, dessen Eigenschaften mit der jeweiligen Simulation gezeigt werden sollten. Wir haben uns bewusst für

eine Auswahl dieser Simulationen entschieden, da diese ganz klar das Ziel hatten, einen neuen Algorithmus anhand von Simulationen mit den Bestehenden zu vergleichen. Die anderen von uns ausgewählten Simulationen haben verschiedene andere Grundlagen gehabt und damit auch zum Teil einen anderen Aufbau oder völlig andere Verfahren. Dies hätte eine Vergleichbarkeit weiter erschwert und es wäre nötig gewesen, jede dieser Simulationen einzeln genau zu besprechen. Das Ergebnis unserer Untersuchung wäre freilich das Gleiche geblieben und so haben wir uns, auch um den jeweiligen Autoren gerecht zu werden, gegen eine Nennung der weiter betrachteten Simulationen entschieden. Die Gesamtaussage dieses Abschnittes ist dadurch allerdings nicht beeinträchtigt, da sich die wesentlichen Kritikpunkte auch anhand der folgenden Veröffentlichungen zeigen lassen. Wir haben uns dabei gegen eine Bewertung der konkreten Simulationen entschieden und fassen hier lediglich die erkennbaren Parameter zusammen, um das Dilemma der Objektivität und Vergleichbarkeit qualitativ darzulegen:

X. Ji und H. Zha, 2004 [60] Es werden 400 Knoten auf einer 100×100 Fläche zufällig positioniert - über die Verteilung wird dabei nichts gesagt. Der Fehler wird als maximal 50 % der Reichweite angenommen und ist dabei gleichverteilt. Die 10 Anker werden ebenfalls zufällig positioniert. Metrik ist dabei der MAE über alle Knoten. Die Knoten sind dabei unbeweglich.

J. Cota-Ruiz et al., 2012 [21] Es werden 96 Sensorknoten in einer 100×100 m^2 Simulationsfläche zufällig verteilt und dazu noch vier zufällig, aber nicht kollinear verteilte Ankerknoten ausgebracht. Das Experiment wird 20-mal wiederholt. Die Reichweite der Entfernungsmessung wird dabei als nicht beschränkt angenommen. Der Distanzfehler wird über ein entfernungsabhängiges *log-distance* Fehlermodell angenommen. Es werden also ausschließlich LOS-Fehler simuliert. Durch das zufällige Positionieren der vier Ankerknoten entsteht im Erwartungswert eine konvexe Hülle mit einem Flächeninhalt von ca. 10 % der Gesamtfläche, d. h. das 90 % der zu lokalisierenden Knoten sich außerhalb der konvexen Hülle der Anker befinden. Die Vergleichsalgorithmen hier sind LLS, NLLS und Min-Max.

L. Haiyong et al., 2011 [49] Es werden auf einer 50×50 m^2 Fläche 10 Ankerknoten und 16 zu lokalisierende Knoten platziert. Die Reichweite für die Distanzmessung wurde mit 15 m angenommen. Die Ankerknoten werden dabei auf einem regelmäßigen Gitter platziert und die zu lokalisierenden Knoten in einer un-

regelmäßigen Anordnung. Drei Knoten werden dabei außerhalb der konvexen Hülle der Ankerknoten platziert. Dieses Setup wird für alle 100 Simulationsläufe so beibehalten. Der Fehler wird als entfernungsabhängiger *log-normal* Fehler mit einem Maximum von 10 % und für weitere 100 Läufe mit einem Maximum von 60 % der wahren Entfernung addiert. Verglichen wird mit einem *Centroid* Algorithmus, der als Schätzung der Position den geometrischen Schwerpunkt aller Schnittpunkte zurück liefert und mit Min-Max.

P. C. Chen, 1999 [19] Es gibt es keine klaren Angaben über die Abmessungen der Simulationsfläche, zur Ankeranzahl oder zu den Entfernungen zwischen den Knoten. Es werden drei Simulationsreihen mit einem Viertel NLOS-Ankern, einem Fünftel NLOS-Ankern und zwei Fünfteln NLOS-Ankern durchgeführt. Der Grundfehler für alle Distanzmessungen ist ein normalverteilter Fehler mit einem Erwartungswert von 60 m. Der NLOS-Fehler variiert zwischen 100 m und 1300 m. Als Vergleichsalgorithmus wird LLS herangezogen. Über die Geometrie der Knoten in den Experimenten werden keine verwertbaren Angaben gemacht.

X. Li et al., 2008 [78] Es werden auf einer Fläche von 200×200 m^2 die Knoten und Anker gleichverteilt zufällig verteilt. Der Fehler ist normalverteilt mit Werten zwischen 0 und 50 Metern. Die Reichweite der Distanzmessung beträgt 50 m. Die Ankeranzahl, die ein zu lokalisierender Knoten in Reichweite hat, wird für Zahlen zwischen fünf und 25 simuliert. Vergleichsalgorithmen sind LLS und LMS. Eine Besprechung der geometrischen Konstellationen bzw. genauere Angaben zu diesen finden sich nicht.

D. Liu et al., 2008 [83] Es werden eine unbekannte Anzahl Ankerknoten auf einem 30×30 m^2 großen Spielfeld gleichmäßig (sic!) verteilt und der zu lokalisierende Knoten genau in der Mitte der Fläche positioniert. Eine Begrenzung der Reichweite findet de facto nicht statt. Der Grundfehler in der Distanzmessung beträgt -4 m bis 4 m. Hinzu kommt eine unbekannte Zahl an Ankerknoten die einen zusätzlichen Fehler zwischen 0 und 100 Metern addieren.

S. Misra und G. Xue, 2007 [92] Es werden auf einem 100×100 m^2 Feld zwischen vier und zwölf Anker ausgebracht, unter denen sich jeweils zwischen null und vier Angreifer befinden. Diese Angreifer arbeiten auf unbekannte Art und Weise zusammen, um den Knoten 30 m in eine unbekannte Richtung zu verschieben. Über jedwede Geometrien und den evtl. Wechsel der selbigen gibt es keinerlei Informationen. Die Anzahl der Durchläufe einer Simulation ist mit 50 angege-

ben. Es finden sich widersprüchliche Angaben darüber, ob der MAE oder der RMSE als Metrik benutzt wurden.

G. Kuruoglu et al., 2009 [70] In einem „C++ based event driven simulator“ wird die verteilte Lokalisierung von 200 normalverteilten Knoten, auf einer Fläche von 200×200 m^2, simuliert. 40 dieser Knoten kennen ihre Position initial und fungieren vom Start an als Anker. Der Distanzfehler wird als ein normalverteilter Fehler mit einer Varianz von 10 % der wirklichen Entfernung simuliert. Als Vergleichsalgorithmus wird Min-Max simuliert. Als weiterer Vergleichsalgorithmus wird LLS simuliert. Als Metrik wird der MAE aus 10 Simulationsläufen angegeben. Wie lange ein einzelner Lauf gedauert hat, ist nicht präzisiert.

J. J. Robles et al., 2012 [110] In dieser in Matlab durchgeführten Simulation wurden vier Ankerknoten auf einem 100×100 m^2 großen Feld platziert. Die Ankerknoten bilden dabei ein Quadrat in der Mitte des Spielfeldes, welches 10 % der Fläche einnimmt - 90 % der Fläche befinden sich also außerhalb der konvexen Hülle der Ankerknoten. Für die Auswertung wurden 10 000 verschiedene Positionen für den zu lokalisierenden Knoten zufällig bestimmt und für jede dieser Positionen 50-mal eine Lokalisierung vorgenommen. Der Fehler wird als ein normalverteilter Fehler mit einer Begrenzung von 0 % bis 20 % der wahren Distanz addiert. Als Metrik wird der MAE genommen und mit Min-Max, LLS und einem *R-LS* genannten Ansatz, zur brute-force basierten Lösung des nichtlinearen Gleichungssystems, verglichen.

S. Nawaz und N. Trigoni, 2010 [96] In dieser Simulation wird der zu lokalisierende Knoten „ungefähr“ in der Mitte einer 10×10 Einheiten^2 großen Fläche ausgebracht und 10 Ankerknoten zufällig in dieser verteilt. Eine zufällig gewählte Distanz erhält einen aus einer Gleichverteilung gezogenen NLOS-Fehler zu dem auf alle Entfernungen addierten normalverteilten Fehler hinzu addiert. Verglichen wird mit LLS und NLLS und der MAE wird als Metrik genommen. Der NLOS-Fehler variiert in Häufigkeit und maximalem Wert zwischen den einzelnen Simulationen. Jede Simulation wurde zur Bildung des MAE 100-mal wiederholt. Basierend auf diesen Grundparametern werden eine Reihe weiterer Simulationen durchgeführt und analysiert.

Z. Li et al., 2005 [79] In einer Fläche von 500×500 m^2 werden 10 Ankerknoten zufällig verteilt. Der Messfehler für alle Distanzen wird als normalverteilt angenommen. Die Wahrscheinlichkeit, dass ein Ankerknoten einen Ausreißer erzeugt,

soll unter 50 % liegen und die Richtung dieser Ausreißer soll abgestimmt sein, so dass alle Ausreißer einen Drift in die selbe Richtung erzeugen. Die Position des Knotens wird nicht beschrieben. Verglichen werden die Resultate mit LLS und als Metrik wird der RMSE verwendet.

M. Jadliwala et al., 2010 [59] Die Fläche der Simulation beträgt hier 500×500 m^2, auf der die 43 Ankerknoten zufällig verteilt werden, ebenso wie der zu lokalisierende Knoten. Der Fehler für die Distanzmessung wird einmal aus einer Gleichverteilung und einmal aus einer Normalverteilung mit Maximalfehlern zwischen -5 m bis 5 m gezogen. Ebenso werden für die einzelnen Simulationen zwischen 0 bis 20 Anker als Angreifer ausgewählt, dies mit einem nicht näher angegebenen Fehler. Verglichen wird mit dem VBLE Algorithmus und der MAE als Metrik genommen.

Gemeinsamkeiten

Auch wenn allein schon aufgrund der fehlenden Angaben eine Vergleichbarkeit der einzelnen Ansätze nicht gegeben ist, so fallen einige Gemeinsamkeiten auf: Die Simulationen finden grundsätzlich in zweidimensionalen rechteckigen Ebenen statt. Die Ankerknoten werden oft zufällig platziert, leider ohne eine Diskussion der sich daraus ergebenen geometrischen Implikationen, die für die meisten Algorithmen von großer Bedeutung wären. Auch wird oft (im Vergleich zu Experimenten in der Praxis) eine sehr große Zahl von Ankerknoten simuliert.

Bei der Simulation der Distanzfehler fällt auf, dass in vielen Fällen ein normalverteilter Grundfehler mit Erwartungswert Null simuliert wird. Mit den meisten Messmethoden in der Praxis wird der Erwartungswert allerdings weit in den positiven Bereich verschoben sein, da ein zu kurzes Messen höchstens durch schlecht kalibrierte Systeme auftreten kann, nicht aber durch physikalische Effekte der Signalausbreitung. So wird ein auf Laufzeitmessung basierendes System nur dann eine zu kurze Entfernung liefern, wenn eine der beteiligten Uhren zu langsam läuft. Nie aber durch Effekte auf das Signal selbst. Diese Beobachtung spielt für eine Vielzahl von Algorithmen eine große Rolle, da zu kurze Messungen im nicht Schneiden der gedachten Kreise um die jeweiligen Ankerknoten resultieren können. Andererseits lässt ein Zulassen dieser perfekt normalverteilten Fehler mit dem Erwartungswert Null viele Algorithmen, insbesondere den NLLS Algorithmus, extrem gute Resultate erzielen, die sie in der Praxis so nicht erzielen würden.

Ebenfalls auffällig ist, dass die Mehrheit der von uns ausgewerteten Simulationen als Vergleichsalgorithmen LLS oder Min-Max benutzen. Diese gehören jeweils zu den Algorithmen mit der schlechtesten Performance ihrer Art, sind dafür aber extrem einfach zu implementieren. Da nicht mit den jeweils besten bekannten Algorithmen verglichen wird, haben die Simulationsergebnisse oft nur einen rein theoretischen Nutzen. Aussagen über die Eignung eines Algorithmus für die Praxis lassen sich daraus leider nicht ableiten, da ja schon die Vergleichsalgorithmen nur eine sehr eingeschränkte Eignung aufweisen.

Fazit

Ein wesentliches Problem der in der Literatur durchgeführten Simulationen sehen wir in der mangelnden Vergleichbarkeit. Keine uns aus dem Bereich der Indoorlokalisierung bekannte Veröffentlichung versucht, die Simulationsparameter und Umgebung eines bereits veröffentlichten Algorithmus zu reproduzieren, sondern es werden grundsätzlich diverse Parameter nach eigenem Gusto gewählt. Sehr häufig ist die Wahl der Parameter - wenn überhaupt angegeben - nur äußerst dünn motiviert. Dabei hätte das Wiederholen vorhandener Simulationen neben der Vergleichbarkeit auch den wesentlichen Vorteil, dass gezwungenermaßen eine Diskussion der Parameter und Szenarien auf Konferenzen und Workshops erfolgen würde und eine ständige Überprüfung der publizierten Ergebnisse vorgenommen würde. Das dieses unterbleibt, liegt nicht nur an den publizierenden Wissenschaftlern selber, sondern auch an dem simplen Fakt, dass es keine von der Community anerkannten Simulationstools gibt. Die meisten Veröffentlichungen schweigen sich gänzlich über die verwendete Software aus bzw. geben nur einen generischen Hinweis auf Matlab oder ähnliche Tools.

Die letzten Absätze sind jedoch nicht als Plädoyer gegen Simulationen und für die ausschließliche Durchführung von Experimenten zu verstehen. Gegenüber Experimenten bieten Simulationen immer noch den Vorteil der Reproduzierbarkeit - sofern die eingestellten Parameter bekannt sind. Ergebnisse aus Praxisexperimenten sind so gut wie nicht reproduzierbar und oft können nicht beachtete Eigenschaften der Umgebung einen riesigen Einfluss auf die Ergebnisse haben. So haben wir in unseren Experimenten immer wieder festgestellt, dass beispielsweise das kurzzeitige Öffnen und Schließen von Türen einen großen Einfluss auf nahe Ankerknoten bzw. die zu diesen Knoten gemessenen Entfernungen haben kann. Ein anderes Beispiel sind Ausreißer an einer bestimmten Stelle des Gebäudes, die wir uns nur schwer erklären konnten.

Nach der Veröffentlichung entsprechender Ergebnisse hat sich dann herausgestellt, dass wahrscheinlich mit Metall beschichtete Fensterscheiben (zur Thermoisolierung) eine extreme Reflexion der Funksignale zur Folge hatte.

Unserer Meinung nach kann nur eine gewissenhafte Durchführung von Experimenten und Simulationen in möglichst standardisierten Szenarien und mit in der Community akzeptierten Tools einen wirklichen Fortschritt in diesem Gebiet erzielen. Ein wesentlicher Schritt wird es sein, die Ergebnisse von Simulationen durch Experimente so gut es geht zu verifizieren und möglichst alle Daten dieser Experimente der Community zur Verfügung zu stellen. Für entfernungsbasierte Verfahren könnte z. B. das Vorhandensein einer großen Menge an Entfernungsdatensätzen eine Reproduzierbarkeit und Vergleichbarkeit von Praxisexperimenten gewährleisten. Für Simulationen würde eine diskutierte und akzeptierte Sammlung von Tools und Szenarien diese beiden Punkte ebenfalls stark fördern.

Mit dem LS^2 Simulationsframework haben wir in diversen Veröffentlichungen und Diskussionen auf Konferenzen und in Journalen einen ersten Schritt in diese Richtung getan. Im Folgenden präsentieren wir für alle uns bekannten entfernungsbasierten Lokalisierungsalgorithmen eine ausführliche Diskussion der Ergebnisse und überprüfen anhand der ausgewählten Parameter und Szenarien die Algorithmen auf die Eignung für die Indoorlokalisierung bzw. für spezielle Problemstellungen innerhalb der Indoorlokalisierung.

5.1.2 Simulation der räumlichen Fehlerverteilung mittels LS^2

Um die in Kapitel 4.2.4 vorgestellte räumliche Fehlerverteilung zu analysieren, haben wir das Simulationswerkzeug *The FU Berlin parallel lateration-algorithm simulation and visualization engine (LS^2)* entworfen und implementiert. Ziel von LS^2 ist es, die räumliche Fehlerverteilung von distanzbasierten Lokalisierungsalgorithmen zu visualisieren. LS^2 ist ein offenes Werkzeug, das an der FU Berlin weiter entwickelt und gepflegt wird [148, 147].

Die wesentlichen Entwicklungsziele für LS^2 waren neben einer individualisierbaren Visualisierung, die einfache Erweiterbarkeit und der modulare Aufbau. Ein Hauptaugenmerk bei der Implementierung lag auf einer Minimierung der Simulationslaufzeit, um die Akzeptanz eines solchen Tools wesentlich zu steigern. Grundlage des Tools ist es, einen Lokalisierungsalgorithmus für jede Position einer diskreten Simulationsfläche

mit einer bestimmten Anzahl an Wiederholungen für ein bestimmtes Fehlermodell zu simulieren. Die daraus für eine Position resultierenden Fehler können nach bestimmten Metriken ausgewertet und visualisiert werden. Dabei soll das Tool einfach um neue Algorithmen und Fehlermodelle zu erweitern sein.

Softwarearchitektur

Kern von LS^2 (siehe Algorithmus 5) ist eine in C implementierte Bibliothek, in der sowohl der eigentliche Simulator als auch alle Algorithmen und alle Fehlermodelle enthalten sind. Der Simulator iteriert nach dem Starten über jede diskrete Position auf einer quadratischen Simulationsfläche und rechnet zunächst für diese Position die Entfernungen zu allen Ankern aus (Zeile 3 bis 6). Anschließend übergibt er die Positionen der Anker und die berechneten wahren Entfernungen an das, durch den User festgelegte, Fehlermodell (Zeile 10). Dort wird der Fehler nach den Maßgaben des Fehlermodells berechnet und auf die jeweiligen Entfernungen addiert und an den Simulator zurückgegeben. Der Simulator ruft nun für jede Position den ausgewählten Lokalisierungsalgorithmus auf und übergibt, neben den Positionen der Anker, die fehlerbehafteten Entfernungen. Der Lokalisierungsalgorithmus schätzt nun anhand dieser Daten die gesuchte Position des Knotens und gibt diese an den Simulator zurück. Dieser berechnet aus der zurückgegebenen Position und der wahren Position den resultierenden Fehler als euklidischen Abstand aus beiden und trägt diesen in eine Matrix an der Stelle der wahren Position ein (Zeile 12). Die letzten beiden Schritte werden je nach Maßgabe des Benutzers wiederholt, um eine konvergierende Verteilung zu erhalten (Zeile 8). Dabei wird je nach Metrik (RMSE, MAE etc.) die Verteilung berechnet. Ist jede Position der Matrix so oft durchgelaufen, wie es der Benutzer eingestellt hat, wird die Matrix mit den jeweiligen Verteilungen für jede Position zurückgegeben (Zeile 18).

Um die Bibliothek nutzen zu können, existiert ein einfacher Kommandozeilenwrapper, der die Parameter der Simulation entgegennimmt und an den eigentlichen Simulator übergibt. Aus der resultierenden Matrix berechnet dieser Wrapper eine grafische Darstellung in Form einer Bilddatei.

Algorithmus 5 LS²-Kern

Eingabe: Durchläufe n, Seitenlänge k Fehlermodel ERM, Algorithmus ALG, Metrik M, und Ankerpositionen A_i, $i = 1, \ldots, N$, $N \geq 3$, $\forall i, j : A_i \neq A_j \vee i = j$;
Ausgabe: die räumliche Fehlerverteilung SD, $SD \leftarrow k \times k$

1: $SD_{temp} \leftarrow k \times k \times n$ ▷ Enthält alle Residuen
2: $distance \leftarrow nil$ ▷ Fehlerbehaftete Distanzen
3: **for** $x \leftarrow 1$ to k **do**
4: **for** $y \leftarrow 1$ to k **do**
5: **for** $a \leftarrow 1$ to $N - 1$ **do**
6: $distance[a] \leftarrow$ euklidischer Abstand zu $A[a]$ von Punkt (x, y)
7: **end for**
8: **for** $i \leftarrow 1$ to $n - 1$ **do**
9: **for** $a \leftarrow 1$ to $N - 1$ **do**
10: $distance[a] \leftarrow ERM(A[a], distance[a])$ ▷ Fehlermodell anwenden
11: **end for**
12: $SDtemp[x, y, i] \leftarrow ALG(distance, A)$ ▷ Lokalisierungsalgorithmus aufrufen
13: **end for**
14: **end for**
15: **end for**
16: **for** $x \leftarrow 1$ to k **do**
17: **for** $y \leftarrow 1$ to k **do**
18: $SD \leftarrow M(SD_{temp}[x, y])$
19: **end for**
20: **end for**
21: **return** SD

Implementierung

In der Analyse des Pseudocodes (Algorithmus 5) sieht man, dass der eigentliche Algorithmus $n \times x^2$-mal aufgerufen wird und das Fehlermodell $n \times x^2 \times N$-mal. Wobei

$n \leftarrow$ der Anzahl der Durchläufe,

$x \leftarrow$ der Seitenlänge der quadratischen Simulationsfläche und

$N \leftarrow$ der Anzahl der Anker

entspricht. Für gängige Simulationsszenarien von 8 Ankern, 1000 Durchläufen und einer Feldbreite von 1000 Einheiten, sind das alleine eine Milliarde Aufrufe des Algorithmus und acht Milliarden Aufrufe des Fehlermodells, welches pro Durchlauf mindestens einmal eine Pseudozufallszahl benötigt - anspruchsvollere Fehlermodelle brauchen deutlich mehr Pseudozufallszahlen.

Um die Laufzeit des Simulationskernes wesentlich zu reduzieren, soll auf jegliche Funktionsaufrufe verzichtet werden, da diese einen erheblichen Overhead darstellen und sich durch das jeweilige Anlegen eines Stackrahmens sehr negativ auf das Cacheverhalten auswirken. Hierzu soll sowohl der Aufruf des Algorithmus, als auch der Aufruf des Fehlermodells als *inline* Aufruf erfolgen. Um dieses Ziel erreichen zu können, müssen in ANSI C alle Funktionen die *inline* aufgerufen werden, in der selben Datei stehen. Da dieses einer einfachen Erweiterbarkeit und der Übersichtlichkeit erheblich im Wege stehen würde, wurde hier durch uns ein anderer Weg gegangen. Alle Algorithmen und Fehlermodelle werden durch den Simulationskern nicht per *Linker* eingebunden, sondern als direkter *include* durch den Präprozessor. Dies hat eine sehr übersichtliche Dateistruktur zur Folge und steht dem *inlinen* von Funktionen nicht im Wege. Damit innerhalb der eigentlichen Algorithmen und Fehlermodelle ebenfalls keinerlei Funktionsaufrufe stattfinden, stellt die Simulationsumgebung eine Bibliothek mit diversen Funktionen bereit, die üblicherweise bei der Entwicklung von Lokalisierungsalgorithmen benötigt werden. Diese Bibliothek besteht ebenfalls nur aus *inline* Funktionen.

Durch diese Vorgabe konnte die Laufzeit des eigentlichen Simulationskerns de facto auf Null reduziert werden, da dieser so gut wie nur noch aus Schleifen besteht und diese in modernen Computersystemen mit Pipelineverarbeitung und mehreren ALUs keinen zusätzlichen Rechenaufwand gegenüber dem Schleifeninhalt bedeuten.

Da alle Aufrufe der Algorithmen und der Fehlermodelle voneinander unabhängig

sind, werden die Berechnungen dabei in eine, von der CPU-Anzahl abhängige Anzahl von Threads aufgeteilt, um die Berechnung parallelisieren zu können. Hierzu kommt die POSIX-Bibliothek *pthread* zum Einsatz. Die Laufzeit skaliert dabei fast genau antiproportional zur Anzahl der CPUs. Bei aktuellen Systemen (Intel Ivy Bridge) reicht bei manchen Algorithmen für mehr als drei parallele Berechnungen mit LS^2 die Speicherbandbreite der CPU nicht mehr aus und die Rechenzeit sinkt nicht mehr weiter.

Um die Parallelität weiter zu erhöhen wurde der gesamte Simulationskern für die Berechnung in *Single instruction, multiple data* (SIMD) Vektorpipelines ausgelegt. Unterstützt werden hierbei sowohl die Berechnungen in *Streaming SIMD Extensions* (SSE) [57, Seite 116] und *Advanced Vector Extensions* (AVX) [57, Seite 130] Vektorpipelines - *MultiMedia eXtension* (MMX) [57, Seite 259] Pipelines werden nicht unterstützt, da kaum noch Systeme auf dem Markt sind die nicht mindestens eine SSE Implementierung bieten. SIMD Pipelines bieten für eine Vielzahl von Rechenoperationen Befehle an, die diese Operationen anstatt auf zwei Skalaren, auf zwei Vektoren paarweise durchführen und das Ergebnis ebenfalls in einen Vektor gleicher Größe liefern. Diese ursprünglich für die 3D-Grafikberechnung entwickelte Technologie wurde in den letzten Jahren von den Prozessorherstellern kontinuierlich für die Nutzung in einem breiteren Anwendungsspektrum ausgebaut. SSE unterstützt dabei Vektoren der Größe 128 Bit und AVX Vektoren der Größe 256 Bit. Um diese Pipelines optimal nutzen zu können, haben wir uns dafür entschieden alle Fließkommaberechnungen in einfacher Genauigkeit durchzuführen, so dass in einen SSE Vektor vier Fließkommazahlen und in einen AVX Vektor acht Fließkommazahlen passen. Durch die Auslegung des Simulationskerns und aller angebotenen Hilfsfunktionen auf diese Vektoren, können immer gleich mehrere Berechnungen auf einmal durchgeführt werden und die Anzahl der Durchläufe kann entsprechend um den Faktor vier oder gar acht reduziert werden. Ein entsprechender Geschwindigkeitszuwachs stellt sich aber nur ein, wenn der Algorithmus und das Fehlermodell möglichst wenig Fallunterscheidungen enthalten, da diese natürlich auf den Vektoroperationen nur über Umwege implementiert werden können und der Geschwindigkeitszuwachs schnell durch diesen Overhead aufgezehrt wird. Implementierungen von Algorithmen mit komplexen Fallunterscheidungen packen in einer Schleife sogar den ganzen Vektor aus und rechnen mit den herkömmlichen *Single instruction, single data* (SISD) ALUs. Einen kleinen Geschwindigkeitszuwachs erzielen aber auch diese Algorithmen, da die SIMD Einheiten und die SISD ALUs komplett parallel betrieben werden können. Auch bei der Benutzung von

AVX tritt das Problem auf, dass aktuelle Rechner bei den meisten Algorithmen hier an die Grenzen ihrer Speicherbandbreite stoßen und der Geschwindigkeitszuwachs nicht mehr linear ausfällt. Zu dem eher moderaten Geschwindigkeitszuwachs bei der Nutzung von AVX kommt auch dazu, dass AVX aktuell (Ivy Bridge) keine Integer Operationen unterstützt und somit das Generieren der Zufallszahlen nicht wesentlich beschleunigt werden konnte. Zusätzlich sind einige der wesentlichen arithmetischen Operationen mit der Einführung von AVX zwar in der Bandbreite von 128 Bit auf 256 Bit verdoppelt worden, aber auch die Latenz (und damit die in der Praxis benötigte Rechenzeit) hat sich genau verdoppelt.

Als Zufallszahlengenerator kommt eine Implementierung des Fast-Rand Algorithmus von Intel zum Einsatz, welche auf die Benutzung von Integer-SSE optimiert ist und einen kompletten Vektor mit vier Zufallszahlen gleichzeitig füllt [99].

Fehlermodelle

Um die verschiedenen Algorithmen eingehend untersuchen zu können und die Algorithmen untereinander und mit den Angaben in der entsprechenden Literatur vergleichen zu können, haben wir eine Vielzahl von Fehlermodellen implementiert. Die einzelnen Fehlermodelle können dabei frei mit diversen Parametern konfiguriert werden. Im Nachfolgenden stellen wir die Fehlermodelle vor:

const Dieses Fehlermodell dient der Entwicklung von Algorithmen und Benchmarkzwecken - es gibt direkt die Eingabewerte wieder zurück, addiert also keinen Fehler auf und zieht auch keine Zufallszahlen.

eq-noise Dieses Fehlermodell addiert einen gleichverteilten Fehler auf - der Maximalfehler und der Minimalfehler sind konfigurierbar. Es wird eine Zufallszahl pro Aufruf gezogen.

gamma-noise Dieses Fehlermodell addiert einen gammaverteilten Fehler auf. Es lassen sich *shape*, *rate* und ein additiver *Offset* (Lokalisierungsparameter der Gammaverteilung) konfigurieren. Je nach Größe von *shape* werden mehrere Zufallszahlen gezogen. Die Dichtefunktion ist folgendermaßen definiert:

$$\Gamma(\alpha,\beta)(x) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x} \quad \text{für } x \geq 0 \text{ und } \alpha,\beta > 0$$

wobei α dem *shape*-Parameter und β dem *rate*-Parameter entspricht (siehe auch

Abschnitt 3.1.4). Die Gammaverteilung lässt sich recht gut an viele in der Praxis auftretende Mischverteilungen aus Messfehler und Modellfehler anpassen.

erlang-noise Dieses Fehlermodell addiert einen Erlang-verteilten Fehler auf. Die Erlang-Verteilung stellt einen Spezialfall der Gammaverteilung dar. Mittels eines von P. Reinecke [108] vorgestellten Tools, lassen sich Erlangverteilungen aus Histogrammen von gemessenen Daten generieren und sehr genau fitten. Die Dichtefunktion der Erlang-Verteilung unterscheidet sich von der Gammaverteilung durch die Beschränkung auf natürliche Zahlen im ersten Parameter und ist wie folgt definiert:

$$f(x;k,\lambda) = \frac{\lambda^k x^{k-1} e^{-\lambda x}}{(k-1)!} \quad \text{für } x, \lambda \geq 0$$

wobei k der *shape*-Parameter und λ der *rate*-Parameter ist. Ebenso kann wieder ein additiver *Offset*-Parameter angegeben werden. Mittels eines multiplikativen Skalierfaktors lässt sich der Erwartungswert des Fehlermodells auf einen bestimmten Wert festlegen.

nd-noise Dieses Fehlermodell addiert einen normalverteilten Fehler auf. Dieses Modell dient der Simulation von bloßen Messfehlern, da die Verteilung üblicher Modellfehler nicht berücksichtigt wird. Die Dichtefunktion ist definiert als:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

wobei μ dem Erwartungswert der Verteilung entspricht und σ der Standardabweichung. Es werden 25 Zufallszahlen in unserer Implementierung benötigt.

nlosp Dieses Fehlermodell addiert einen normalverteilten Messfehler (siehe oben) und zusätzlich mit einer konfigurierbaren Wahrscheinlichkeit einen exponentialverteilten Modellfehler. Für die Exponentialverteilung ist der *rate*-Parameter konfigurierbar. Die Dichtefunktion der Exponentialverteilung ist definiert als:

$$f(x;\lambda) = \begin{cases} \lambda e^{-\lambda x}, & x \geq 0, \\ 0, & x < 0. \end{cases}$$

wobei λ der *rate*-Parameter ist. Es werden 27 Zufallszahlen benötigt.

bahillo Dieses Fehlermodell wird von Bahillo et al. [6] beschrieben und soll ebenfalls

die Kombination aus Messfehler und Modellfehler berücksichtigen. Zur Simulation des Messfehlers wird ebenfalls ein normalverteilter Fehler angenommen, der einen Erwartungswert von Null hat und eine Standardabweichung von 2,3 m besitzt. Auf diesen Fehler wird nun immer ein exponentialverteilter Fehler mit einem zufällig bestimmten $\lambda \in (0, 3)$ aufaddiert. Im Erwartungswert benötigt dieses Modell 28 Zufallszahlen.

ab-nlos Während *nlosp* und *bahillo* geeignete Fehlermodelle für die Auswertung eines Durchschnittsfehlers sind, sind sie für die Auswertung der räumlichen Fehlerverteilung ungeeignet. Bei beiden letztgenannten Fehlermodellen werden die Anker, die einen NLOS-Fehler produzieren, pro Lokalisierung zufällig ausgewählt. Betrachtet man eine große Anzahl an Simulationsläufen, ist das Auftreten des NLOS-Fehlers über die Anker gleichverteilt. Dies führt in der Auswertung der räumlichen Fehlerverteilung zu einer Mittelung des Ergebnisses und zur Ausbildung von geometrischen Strukturen, die den Einfluss von NLOS-Fehlern auf die räumliche Fehlerverteilung nicht mehr erkennen lassen. Das *ab-nlos*-Fehlermodell simuliert NLOS- und LOS-Fehler mit der gleichen Verteilung wie das *nlosp*-Fehlermodell, mit dem einzigen Unterschied, dass der NLOS-Fehler immer auf die selben Anker angewendet wird. Daraus resultiert exakt derselbe Durchschnittsfehler, der auch beim *nlosp*-Fehlermodell entsteht, jedoch bei der Ausbildung von geometrischen Strukturen, die den Einfluss von NLOS-Fehlern auf den jeweiligen Algorithmus besser visualisieren.

Grafische Benutzeroberfläche

Um die Simulationen benutzerfreundlicher zu gestalten, haben wir eine grafische Benutzeroberfläche für LS^2 entwickelt (LS^2-GUI). Diese ist in Abbildung 5.1 zu sehen. Das Programm wurde in Java implementiert und benutzt Java Native Access (JNA) [32], um LS^2 als dynamische Programmbibliothek zu laden und für die Berechnung zu verwenden. Der dabei auftretende Overhead ist minimal. Wie in Abbildung 5.1 gezeigt, lassen sich die Ankerknoten auf einer in der Größe änderbaren Simulationsfläche platzieren und anschließend die Simulation mit einem Algorithmus und einem Fehlermodell starten. Verfügbare Algorithmen und Fehlermodelle werden aus der LS^2 Bibliothek ermittelt. Weiterhin lassen sich die Anzahl der Durchläufe pro Position sowie die Anzahl der zu verwendenden Threads festlegen. Die Anzeige des Fortschritts

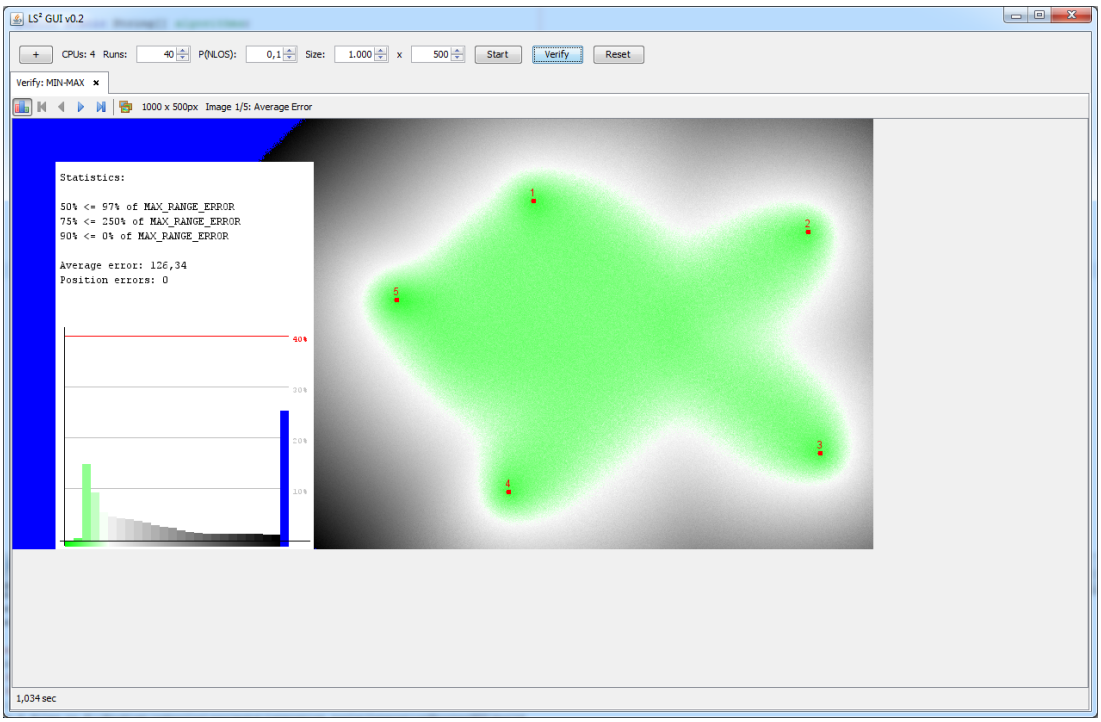


Abbildung 5.1: Grafische Benutzeroberfläche von LS^2. Zu sehen ist die räumliche Fehlerverteilung von Min-Max anhand der Durchschnittswerte.

einer Simulation sowie deren vorzeitiger Abbruch werden ebenso unterstützt. Als Ergebnis werden sämtliche von LS^2 unterstützten Ausgabemetriken dargestellt. Im Beispiel sieht man die räumliche Fehlerverteilung des Min-Max Algorithmus anhand der Durchschnittswerte.

Jedoch dient das Programm nicht allein als GUI-Wrapper für LS^2, sondern ebenso als Referenzimplementierung des LS^2 Simulators. Hierfür verwendet LS^2-GUI die in Abschnitt 3.5 vorgestellte LatMath Bibliothek und die in dieser enthaltenen Algorithmen und Fehlermodelle, welche anstelle der LS^2 Implementierung verwendet werden. Einziger Mehraufwand stellt ein minimaler Simulationskern dar, der analog zu LS^2 nach dem in Algorithmus 5 gezeigten Prinzip arbeitet. Dieser verwendet analog zu LS^2 eine einstellbare Anzahl an *Worker Threads*, um die Berechnungen möglichst effizient zu verteilen.

Mithilfe dieser Referenzimplementierung ließ sich z. B. die korrekte Portierung der in Java implementierten Algorithmen aus der LatMath Referenzbibliothek nach dem in C programmierten LS^2 leicht verifizieren und diente somit als Debugging-Werkzeug.

Tabelle 5.1: Multithreading Skalierung (AVX-Version)

Algorithmus	1 Thread	2 Threads	4 Threads	8 Threads
Const	8,92 s (100%)	4,50 s (100%)	2,40 s (107%)	1,67 s (149%)
Trilateration	13,19 s (100%)	6,69 s (101%)	3,51 s (106%)	2,03 s (123%)
LLS	17,57 s (100%)	8,98 s (102%)	4,66 s (106%)	3,18 s (144%)
AML	25,33 s (100%)	13,12 s (104%)	7,02 s (111%)	4,85 s (153%)
NLLS	176,58 s (100%)	99,09 s (112%)	59,59 s (136%)	44,70 s (200%)

Die Tabelle zeigt die Fähigkeit des Simulationskerns, die Simulation auf mehrere CPU-Kerne zu skalieren. Die Experimente wurden auf einem 4-Kern Rechner von Intel mit Hyperthreading durchgeführt. In Klammern ist das Verhältnis von insgesamt benötigter Rechenzeit über alle CPUs und der Single-Thread Variante angegeben - Werte um 100 % bedeuten also eine fast lineare Skalierung. Für alle Algorithmen wurde ein Setup aus drei Ankern und ein gleichverteilter Fehler benutzt. Die Umgebung ist 1000×1000 Einheiten groß und es wurden 1000 Durchläufe gemittelt.

Performanceanalyse

Wie Tabelle 5.1 zeigt, spielt im Wesentlichen die Implementierung des Algorithmus eine Rolle für die Laufzeit, der Overhead der Simulationsengine ist dabei sehr gering. Der *Const* Algorithmus ist eine Implementierung, die immer eine fixe Position zurückgibt, ohne irgendetwas zu berechnen. Bei einer Milliarde Durchläufen ist der Overhead für Simulator und Fehlermodell bei der Nutzung von 4 CPUs[1] knapp eine viertel Sekunde. Auch die Fähigkeit des Simulators auf mehrere CPUs zu skalieren, lässt sich aus der Tabelle leicht ableiten, auch wenn diese, wie für NLLS leicht zu sehen ist, von der Implementierung des Algorithmus abhängt. Obwohl NLLS komplett vektorisiert wurde, treten für diesen Algorithmus schon bei der Skalierung auf 4 CPUs Speicherengpässe auf und die Cachenutzung ist nicht mehr optimal. Die Skalierung auf 8 Threads zeigt, das der Simulator von Intels Hyperthreading nur mäßig profitieren kann, was aber daran liegt, dass Hyperthreading ausschließlich Taskwechsel beschleunigt, diese aber auf unserem ansonsten nicht ausgelasteten Testsystem nicht in großer Zahl vorkommen.

Als Referenzimplementierung haben wir LS^2-GUI mit der LatMath Bibliothek verwendet, welche - mit den Möglichkeiten die Java bietet - auf Performance hin optimiert wurde. Einen Vergleich präsentieren wir in Tabelle 5.2. Auch hier zeigt sich

[1] Alle Simulationen in diesem Abschnitt wurden auf einem Intel(R) Xeon(R) CPU E3-1240 V2 @ 3.40GHz mit 32 GB RAM durchgeführt.

Tabelle 5.2: Vergleich zwischen Java und C Implementierung (AVX-Version)

Algorithmus	Laufzeit Java	Laufzeit AVX	Beschleunigungsfaktor
Trilateration	4,57 s	0,27 s	16,5
LLS	80,68 s	0,68 s	117,8
NLLS	449,25 s	11,97 s	37,5
AML	10,61 s	5,55 s	1,8

Für alle Algorithmen wurde ein Setup aus drei Ankern und ein gleichverteilter Fehler benutzt. Die Umgebung ist 1000×1000 Einheiten groß und es wurden 1000 Durchläufe gemittelt und auf 8 Threads verteilt.

eine starke Abhängigkeit von der Implementierung der Algorithmen. Während voll vektorisierte Algorithmen erheblich von der optimierten Pipelinenutzung und der cacheoptimierten Programmierung der C-Version profitieren, läuft ein nicht optimal vektorisierbarer Algorithmus wie AML, nur ungefähr doppelt so schnell wie die Referenzimplementierung.

In Tabelle 5.3 sind die Laufzeiten für verschiedene Fehlermodelle angegeben. Zieht man die Laufzeit des const-Modells, was keinerlei Berechnungen durchführt, von den anderen Zeiten ab, so ergibt sich eine ungefähr lineare Abhängigkeit von der Menge der gezogenen Zufallszahlen. Neben der Implementierung des Algorithmus, ist also das Ziehen der Zufallszahlen der kritische Code im LS^2 Simulator. Beispielsweise müssen in der Beispielsimulation für *nlosp* immerhin 112 Milliarden Zufallszahlen gezogen werden. Ein schnellerer Generator mit ausreichender Verteilung würde den Simulator wesentlich beschleunigen. Experimente mit in CPUs der neuesten Generation (Intel Ivy-Bridge) enthaltenen Hardware-Generatoren haben gezeigt, dass diese langsamer als die *Fastrand*-Implementierung sind. Die Unterstützung für den Hardware-Generator wurde dementsprechend aus dem Simulator entfernt.

Ausgabemetriken

Über den Simulator können eine Vielzahl von Ausgabemetriken ausgewählt werden, um die räumliche Verteilung zu visualisieren. Durch die modulare Struktur der Software können leicht auch eigene Metriken hinzugefügt werden. Die Rechenzeit wird auch durch das Ausgeben mehrerer Ausgabemetriken auf einmal nur unwesentlich verlängert, weshalb auf einen Performancevergleich der verschiedenen Metriken an

Tabelle 5.3: Vergleich der Fehlermodelle

Fehlermodell	Laufzeit	Zufallszahlen
const	3,92 s	0
eq-noise	5,43 s	1
gamma-noise	8,53 s	3
erlang-noise	11,21 s	5
nd-noise	26,47 s	25
nlosp	34,96 s	27
bahillo	38,14 s	28

Für alle Durchläufe wurde ein Setup aus vier Ankern und der Algorithmus LLS benutzt. Die Umgebung ist 1000×1000 Einheiten groß und es wurden 1000 Durchläufe gemittelt und auf 4 Threads verteilt.

dieser Stelle verzichtet werden soll.

Die im Simulator enthaltenen Metriken sind die Folgenden:

Average Das Resultat enthält die Durchschnittswerte an der jeweiligen Position aus allen Durchläufen.

Minimum Das Resultat enthält den Minimalfehler an der jeweiligen Position aus allen Durchläufen.

Maximum Das Resultat enthält den Maximalfehler an der jeweiligen Position über alle Durchläufe.

Varianz Das Resultat enthält die Stichprobenvarianz der Residuen für eine Position aus allen Durchläufen. Zu beachten ist hier, dass aufgrund der geänderten Maßeinheiten die Standardfarbschemen eine andere Aussage bekommen. Da die Varianz hier die Streuung um den Mittelwert der Residuen beschreibt und nicht um den Mittelwert der geschätzten Positionen, kann aus der Varianz nur eine Aussage über die Streuung des Fehlers getroffen werden und nicht über die Effizienz des Schätzers!

MSE Das Resultat enthält den MSE für eine Position aus allen Durchläufen. Zu beachten ist hier, dass aufgrund der geänderten Maßeinheiten die Standardfarbschemen eine andere Aussage bekommen.

RMSE Das Resultat enthält den RMSE für eine Position aus allen Durchläufen.

Neben einer numerischen Ausgabe der Ergebnisse wird über den Wrapper auch eine grafische Repräsentation der Ergebnisse unterstützt.

Visualisierung der Ergebnisse

Wird eine Visualisierung der Ergebnisse gewünscht, ist die Rückgabe des Simulators jeweils eine Grafik pro Ausgabemetrik. Die standardmäßige Farbgebung ist vor allem auf eine absolute Vergleichbarkeit der Ergebnisse ausgelegt. Somit entsprechen alle visualisierten Ergebnisse Relativwerten, deren Bezugsgröße der Erwartungswert des Eingabefehlers (Distanzfehler) ist.

In den standardmäßig generierten Grafiken treten drei Farbschattierungen auf:

grün Der Fehler der visualisierten Position ist geringer als der Erwartungswert des Eingabefehlers. Eine dunklere Schattierung der Farbe entspricht einem niedrigeren Relativfehler.

grau Der Fehler der visualisierten Position ist größer als der Erwartungswert des Eingabefehlers. Eine dunklere Schattierung der Farbe entspricht einem größeren Relativfehler.

blau Blau visualisierte Positionen entsprechen einem Relativfehler, der größer als 250 % des Eingabefehlers ist. Aufgrund des begrenzten Kontrastumfangs von Computergrafiken wird der Fehler ab diesem Relativwert abgeschnitten.

Für eine Analyse ist zu beachten, dass die Ausgabegrafik also immer in Relation zum Erwartungswert des Eingabefehlers steht! Dies kann bedeuten, dass zwei Szenarien die sich nur im Erwartungswert des Eingabefehlers unterscheiden in der Ausgabe komplett gleich aussehen können, obwohl ein Eingabefehler einen sehr niedrigen und der andere Eingabefehler einen sehr hohen Erwartungswert aufweist. Im Falle der Gleichheit der Ausgabegrafiken würde das bedeuten, dass der Algorithmus den Ausgabefehler linear mit dem Eingabefehler skaliert - sind die Ausgaben unterschiedlich, weist das auf ein nicht lineares Verhalten hin.

Insbesondere ist dieses Verhalten für Fehlermodelle mit auf die Ankerknoten ungleich verteilten Fehlern zu beachten. Wird beispielsweise ein Algorithmus mit 5 Ankerknoten simuliert, von denen vier einen kleinen normalverteilten Fehler aufweisen und einer einen sehr großen exponential-verteilten Fehler aufweist und man senkt jetzt langsam den Erwartungswert des Ausreißerknotens ab, so kann es vorkommen, dass

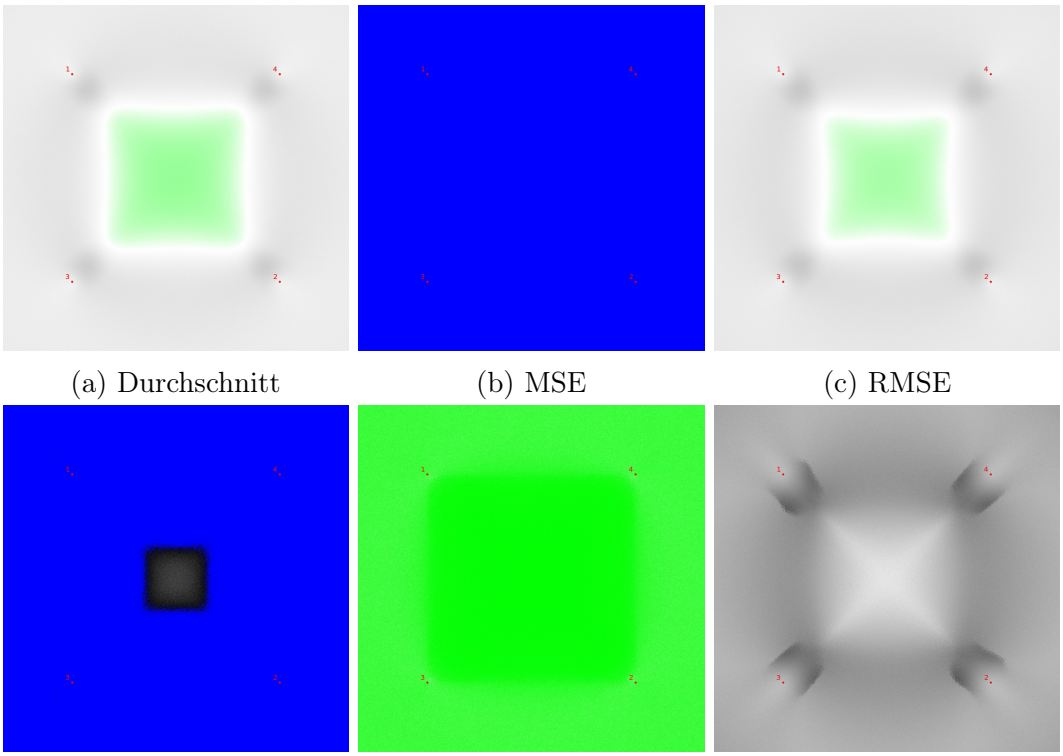
die Ergebnisse scheinbar schlechter und nicht wie erwartet besser werden. Dieses Verhalten liegt ebenfalls an der Bezugsgröße - dem Erwartungswert des Eingabefehlers. Senkt man den Fehler eines Ausreißerknotens ab, so sinkt auch der Erwartungswert der gesamten Fehlerverteilung aller Ankerknoten. Dies führt dann logischerweise zu einem größerem Gewicht der vier Knoten, deren Fehler überhaupt nicht verändert wurden. Insbesondere Algorithmen die den Ausreißer zuverlässig vor der Berechnung der Position erkennen und aussortieren, werden von diesem Effekt betroffen sein. Eine uneingeschränkte Vergleichbarkeit der Ausgaben liegt also nur bei Simulationen mit gleichen Fehlermodellen vor.

Ist dieses Verhalten nicht erwünscht und sollen absolute und nicht relative Fehler verglichen werden, so lässt sich dieses auf zwei Arten einstellen. Erstens kann im Fehlermodell ein Referenzerwartungswert und nicht der wirkliche Erwartungswert eingestellt werden. Auf diese Art und Weise sind mittels der Ausgabe absolute Änderungen der Genauigkeit für verschiedene Fehlermodelle anzugeben. Das eignet sich besonders für die Analyse eines Algorithmus mit verschiedenen Fehlermodellen. Die zweite Möglichkeit besteht darin, die Farbskalen für die verschiedenen Szenarien individuell zu kodieren oder gleich die Analyse mittels HDF5 Ausgabe vorzunehmen.

Ein Beispiel für die Ausgabe der einzelnen Metriken ist in Abbildung 5.2 zu sehen.

Sollen die Daten flexibel weiterverarbeitet werden können und nicht sofort visualisiert werden, bieten sich die Ausgabe im HDF5-Format [125] an, dass von einer Reihe von Statistikprogrammen (R, SPSS etc.) weiterverarbeitet werden kann.

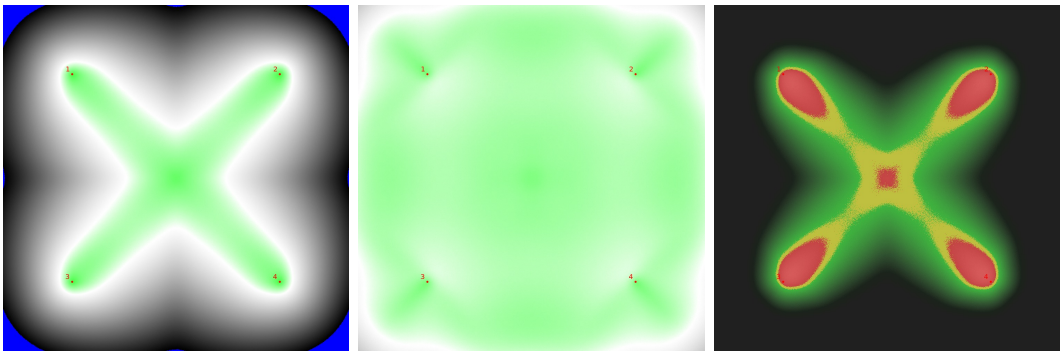
Neben der Ausgabe der Metriken für einen einzelnen Algorithmus bietet der Simulator auch eine Differenzauswertung für zwei verschiedene Algorithmen bei gleicher Metrik an. Dieses Option ist vor allem dann nützlich, wenn man präzise die Unterschiede im räumlichen Verhalten zweier Algorithmen analysieren möchte. Vor allem bei der Entwicklung und Verfeinerung verschiedener Ansätze lässt sich diese Methode sehr gut nutzen um Fort- oder auch Rückschritte in der Entwicklung auf einen Blick zu visualisieren. Ein Beispiel für diese Differenzauswertung zeigen wir in Abbildung 5.3. Hier haben wir bei gleichen Parametern einmal den klassischen Min-Max Algorithmus und einmal den optimierten E-Min-Max Algorithmus simuliert. Da das Designziel von E-Min-Max war, eine bessere Performance außerhalb der konvexen Hülle der Anker zu erzielen, kann man in der Differenzauswertung einfach sehen, ob dieses Ziel erreicht wurde und ob darunter die Performance innerhalb der Hülle der Anker gelitten hat.



(a) Durchschnitt (b) MSE (c) RMSE

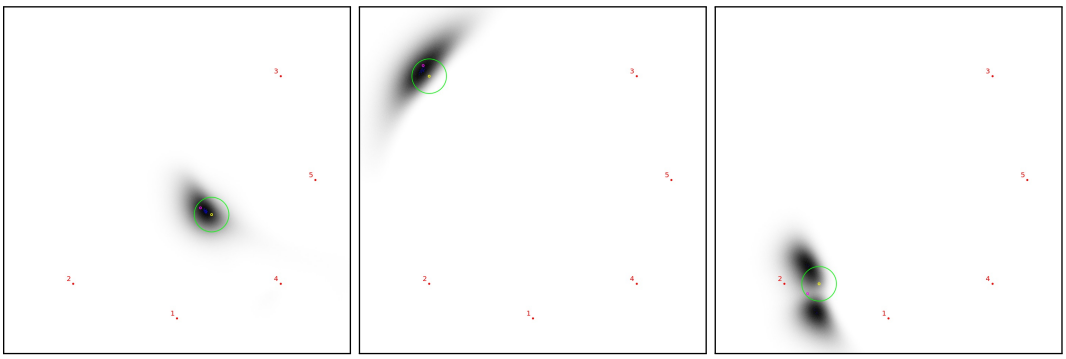
(d) Varianz (e) Minimum (f) Maximum

Abbildung 5.2: Beispielausgabe von LS^2 für verschiedene Ausgabemetriken für den Algorithmus NLLS und Gauß-verteilten Fehler.



(a) Durchschnittliche Min-Max Performance (b) Durchschnittliche E-Min-Max Performance (c) Farbcodierte Differenz aus den vorherigen beiden Bildern.

Abbildung 5.3: Beispielausgabe von LS^2 für eine Differenzauswertung.



(a) AML (b) AML (c) AML

Abbildung 5.4: Beispielausgabe von LS^2 für eine inverse Auswertung. Fehlermodell ist *erlang-noise* bei 500 000 000 Durchläufen.

Die Farbcodierung der Differenzauswertung stellt sich dabei wie folgt dar:

Blau Der Fehler an dieser Position war in beiden Algorithmen ungefähr gleich.

Grün Der Fehler im ersten Bild war größer als im Zweiten. Der zweite Algorithmus stellt für diese Positionen also eine Verbesserung dar.

Rot Der Fehler im ersten Bild war kleiner als im Zweiten. Der zweite Algorithmus stellt für diese Positionen also eine Verschlechterung dar.

Schwarz Die Differenz ist kleiner als der größtmögliche Kontrast. Der zweite Algorithmus ist an diesen Stellen also wesentlich besser geeignet.

Weiß Die Differenz ist größer als der größtmögliche Kontrast. Der zweite Algorithmus ist an diesen Stellen also wesentlich schlechter geeignet.

Inverser Modus

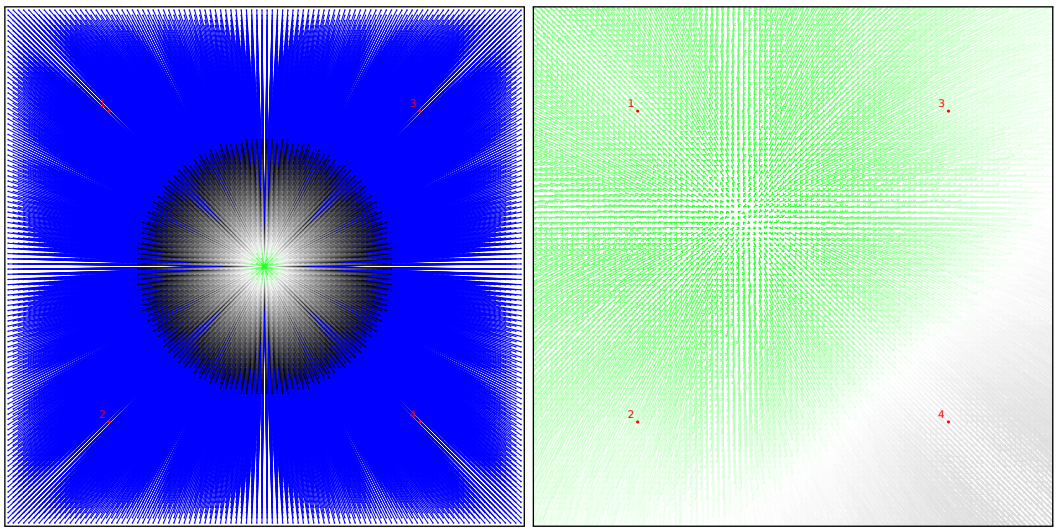
Neben der räumlichen Verteilung des Fehlers kann LS^2 auch dafür genutzt werden, eine invertierte Auswertung vorzunehmen. Anstelle einer Verteilung der Residuen für alle Positionen eines unbekannten Knotens, kann in diesem Modus die Verteilung der Schätzungen für genau eine Position des unbekannten Knotens ausgegeben werden. Diese Auswertung kann dann als ein zweidimensionales Histogramm visualisiert werden, in welchem die Frequenz als Farbton kodiert ist. Diese Form der Analyse hilft vor allem um genauere Aussagen über die Erwartungstreue bzw. über die Verzerrtheit einer Messreihe treffen zu können. Eine Beispielhafte Auswertung für den AML Algorithmus findet sich in Abbildung 5.4.

Die grünen Kreise in den Visualisierungen geben dabei die Entfernung an, die dem Erwartungswert des Eingabefehlers entspricht, in der also die Schätzungen liegen, die im normalen Modus grün eingefärbt werden würden. Im Vergleich der drei Grafiken, in denen alle Parameter der Simulation konstant blieben, sieht man leicht, dass die Verzerrtheit je nach Position des unbekannten Knotens stark variiert. Der Durchschnitt, der mit der Häufigkeit des Auftretens gewichteten Distanz einer Position zum wahren Wert, entspricht folglich wieder genau dem Durchschnitt des Fehlers für die Position des wahren Wertes in der normalen Darstellung. Anhand dieser Darstellung lassen sich sowohl die Varianz als auch die Verzerrtheit des Schätzers für eine gegebene Position diskutieren.

Phasendiagramme

Nutzt man die Ausgabe der Daten im HDF5 Format, kann man sich aus den HDF5 Daten ein Phasendiagramm generieren lassen. In diesem Phasendiagramm ist die durchschnittlich getroffene Position visualisiert. Die Visualisierung ist durch einen Pfeil von der wahren Position zur durchschnittlich geschätzten Position dargestellt. Der Pfeil ist entsprechend seiner Länge nach dem oben beschriebenen Farbschema eingefärbt. Wichtig ist hier zu bedenken, dass die Länge des Pfeils die Entfernung der durchschnittlichen Position zum wahren Wert angibt und nicht die durchschnittliche Entfernung der geschätzten Positionen von der wahren Position. So lässt sich mittels des Phasendiagramms der Grad der Verzerrtheit und die Richtung der Verzerrtheit des Schätzers beurteilen, nicht aber seine Varianz. Die Varianz lässt sich am besten über den inversen Modus beurteilen oder im Vergleich zwischen Phasendiagramm und normaler Darstellung.

In Abbildung 5.5 ist als erstes Beispiel ein Phasendiagramm für einen Schätzer, der unabhängig von Eingabefehlern und Ankern immer den Mittelpunkt der Simulationsfläche zurückgibt, zu sehen. Hier sieht man deutlich die Symmetrie der Darstellung, die auf eine Verzerrtheit abhängig von der wahren Position hinweist. Auch zu sehen ist, dass dieser Schätzer für den Mittelpunkt der Simulationsfläche erwartungstreu ist. In der rechten Abbildung sieht man den Einfluss von Ausreißern auf den LLS Algorithmus. Während der Schätzer um den Mittelpunkt der Simulationsfläche noch fast erwartungstreu ist, verschieben sich die durchschnittlich geschätzten Positionen immer weiter konzentrisch vom Mittelpunkt her nach außen. In der dem Ausreißer



(a) Mittelpunkt als fixe Schätzung (b) LLS

Abbildung 5.5: Beispielausgabe von LS^2 für ein Phasendiagramm. Fehlermodell ist *ab-nlos*, bei 500 Durchläufen pro Position.

produzierenden Anker #1 gegenüberliegenden Ecke steigt die Verzerrtheit sogar noch stärker an. Anker #1 verstärkt also den Effekt des Eingabefehlers, die Schätzung immer von der Mehrheit der Anker „weg zu schieben“, noch weiter.

5.2 Experimente

Neben Simulationen sind Experimente eines der wesentlichen Hilfsmittel zur Evaluierung von Lokalisationssystemen. In der uns vorliegenden Literatur zur Indoorlokalisierung sind Evaluationen mithilfe von Experimenten wesentlich weniger anzutreffen, als Evaluationen mittels Simulationen. Dies hat vor allem drei Gründe:

Aufwand Experimente sind extrem Aufwändig in der Planung und Durchführung. So muss für ein Experiment zur distanzbasierten Indoorlokalisierung ein Gebäude oder der Teil eines Gebäudes mit Ankerknoten ausgerüstet werden und dann muss der zu lokalisierende Knoten durch dieses Gebäude bewegt werden und dessen wahrer Pfad protokolliert werden. Grundvoraussetzung ist hierfür natürlich das Vorhandensein einer entsprechenden Hardware in großer Zahl. Ein Hinweis darauf gibt die Tatsache, dass es in der uns vorliegenden Literatur mehr Ansätze für die nicht RSSI-basierte Lokalisierung gibt als für die RSSI-basierte Lokalisierung. Gleichzeitig gibt es aber mehr RSSI-basierte Experimente als an-

dere Experimente. Wir folgern daraus, dass mit entsprechender Spezialhardware ausgerüstete Knoten zur Entfernungsschätzung noch relativ selten sind. Dazu kommt für große Experimente noch ein erheblicher Kommunikationsaufwand zwischen den Knoten, der ab einer gewissen Gebäudegröße das Vorhandensein eines *Multi-Hop*-fähigen Sensornetzes voraussetzt.

Reproduzierbarkeit Werden bei der Auswertung eines Experimentes Fehler festgestellt, muss das Experiment mit dem entsprechenden Aufwand wiederholt werden. Allerdings sind Experimente nicht einfach zu reproduzieren und gerade im Bereich der funkbasierten Entfernungsschätzung ist es so gut wie unmöglich, ein Experiment innerhalb eines üblichen Gebäudes nur ansatzweise unter den selben Bedingungen stattfinden zu lassen. So haben in unseren Experimenten irgendwo im Gebäude geöffnete oder geschlossene Fenster und Türen zum Teil erhebliche Abweichungen zwischen zwei Experimenten gezeigt und auch das Bewegen von Personen in Räumen, die nicht Teil des Experimentes sind, kann großen Einfluss haben.

Referenzsystem Um ein Experiment evaluieren zu können, reicht es oft nicht aus, den geschätzten Pfad auf eine Karte zu plotten und rein visuell zu beurteilen. Gerade im hoch präzisen Bereich bedarf es einer exakten numerischen Auswertung, was die Nutzung eines Referenzsystems, das präzise *Groundtruth*-Daten liefern kann, als logische Konsequenz nach sich zieht. Solche Systeme müssen eine wesentlich höhere Genauigkeit als das zu evaluierende System besitzen und dürfen die Messungen des zu evaluierenden Systems nicht beeinflussen. Solche Systeme sind meistens teuer und erhöhen den Aufwand des Experimentes mitunter nochmals erheblich.

Gleichzeitig weichen die Ergebnisse von Simulationen regelmäßig stark von den Ergebnissen von Experimenten ab, da die Qualität von Simulationen stark von den Modellen der zu simulierenden Umgebungsparameter abhängt. In Experimenten lassen sich die meisten dieser Parameter nur über die Auswahl eines Gebäudes beeinflussen, aber kaum gezielt steuern, weshalb die Ergebnisse eines Experiments oft eine allgemeinere Gültigkeit haben.

5.2.1 Verschiedene Experimente in der Literatur

Zieht man die in Kapitel 3 vorgestellten Algorithmen und die damit zusammenhängenden Veröffentlichungen als Stichprobe heran, so zeigt sich, dass die Auswertung von Lokalisierungsalgorithmen größtenteils auf den Ergebnissen von Simulationen beruht. Nur zwei der vorgestellten Algorithmen (VBLE und RLSM) werden anhand von Experimenten mit echter Hardware untersucht. In diesem Abschnitt soll ein Überblick der uns vorliegenden Literatur über solche Experimente gegeben werden. Da die Literatur im Gebiet der Lokalisierung recht umfangreich ist und wir uns in dieser Arbeit auf die Analyse entfernungsbasierter Algorithmen beschränkt haben, werden bei der Auswertung auch nur Experimente mit distanzbasierten Methoden in Betracht gezogen. Dazu wurden vor allem die Berichtsbände wichtiger Konferenzen der letzten drei Jahre im Bereich der Indoorlokalisierung durchgegangen, zu nennen sind hier beispielsweise die *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, die *International Conference and Exhibition on Ubiquitous Positioning, Indoor Navigation, and Location Based Service (UPINLBS)* und der *Workshop on Positioning, Navigation and Communication (WPNC)*. Zudem wurden einige digitale Online-Bibliotheken (vor allem *IEEE Xplore* und *ACM*) durchsucht. Betrachtet werden soll bei der Auswertung vor allem die eingesetzte Hardware und damit verbunden die Art der Distanzmessung sowie der Aufbau und die Durchführung des Experiments. Bei dem Aufbau des Experiments ist vor allem die Anzahl der verwendeten Anker und deren Verteilung von Interesse, da hieraus die durchschnittliche Ankerdichte resultiert, was wiederum erheblichen Einfluss auf die Ergebnisse haben kann. Dies ist vor allem bei Experimenten innerhalb von Gebäuden von besonderer Bedeutung. Experimente die sich auf einen einzelnen Raum beziehen, werden sicher andere Ergebnisse liefern als solche, die sich über mehrere Räume erstrecken oder sich gar über ein ganzes Gebäudestockwerk ausbreiten. Entscheidend ist hierbei das Verhältnis von LOS- zu NLOS-Distanzmessungen, welches sich durch die Ausbringung der Knoten ergibt, da dieses besonderen Einfluss auf die Genauigkeit der untersuchten Algorithmen haben kann.

Bei der Durchführung von Experimenten ist neben den vorhandenen Umgebungsparametern (bewegen sich Personen im Messfeld, öffnen und schließen sich Türen, wird parallel andere Funktechnologie im selben Frequenzband verwendet) zuallererst der Grad der Mobilität von Bedeutung. Im einfachsten Fall ist das komplette Sensornetz statisch, d. h. der mobile Knoten wird auf eine oder mehrere bekannte Positionen ge-

stellt und anschließend werden einige Messungen mit den Ankerknoten durchgeführt. Ein weitaus realistischerer Fall ist es, dass sich der mobile Knoten durch das Testfeld bewegt, während mit konstanter Rate die Entfernungen zu den Ankerknoten bestimmt werden. Hierzu bedarf es allerdings eines Referenzsystems, das eine deutlich höhere Genauigkeit aufweist, als das zu testende System. Bewegen sich mehrere mobile Knoten durch das Testfeld, könnten diese selbst als Referenzknoten dienen, sobald sie eine Position bestimmt haben. Der höchste Grad an Mobilität ist erreicht, falls sich alle Knoten bewegen und man keine fest installierten Ankerknoten mehr hat. Solche Art an Experimenten sind besonders für *Multi-Hop*-Algorithmen interessant. Allerdings bedeutet die Durchführung eines solchen Experiments einen enormen zeitlichen und personellen Aufwand oder hohe finanzielle Kosten, falls jeder Knoten mittels Referenzsystem bewegt und überwacht werden soll. Im Allgemeinen liegen die Kosten für ein gutes Referenzsystem deutlich über den Kosten des zu testenden Systems.

Ein weiterer zu berücksichtigender Punkt sind die gewählten Positionen des mobilen Knotens bzw. der abgeschrittene Pfad des mobilen Knotens bei vorhandener Mobilität. Hierbei sollte neben der durchschnittlichen Ankerdichte auf dem Pfad und dem Verhältnis von LOS- zu NLOS-Ankern auch die eigene Position zu den erreichbaren Ankern betrachtet werden. Befindet sich der mobile Knoten fortwährend innerhalb der konvexen Hülle der Ankerknoten, was bei einer gängigen Ausbringung der Anker an Wänden von Räumen und Fluren vorzufinden sein sollte, so wird die Lokalisierungsgenauigkeit höher sein. Deshalb ist die Betrachtung von ungünstigen geometrischen Ankerkonstellation und die Performance der Algorithmen unter diesen Bedingungen ebenfalls von Bedeutung, um deren Stärken und Schwächen besser zu beleuchten und die Leistung im schlechtesten Fall offenzulegen. Als letztes haben wir für jedes Experiment - sofern vorhanden - die Auswertungsmetrik angegeben und verglichen.

Um die Genauigkeit von VBLE außerhalb von Simulationen zu bestimmen, wurde von Liu et al. [82] ein Freifeldtest auf einem Feld von 12 m $\times$ 12 m durchgeführt. Es wurden 8 MICA2 [23] Sensorknoten auf dem Zielfeld gleichmäßig verteilt. Der zu lokalisierende Knoten befand sich dabei in der Mitte des Feldes und damit zentral in der konvexen Hülle der Anker, wobei kein Knoten seine Position geändert hat. Als Betriebssystem kam TinyOS [77, 129] zum Einsatz, das das Auslesen des RSSI für jedes empfangene Paket ermöglicht und somit die Berechnung der empfangenen Signalstärke erlaubt. Über vorher durchgeführte Messungen dieses Wertes bei verschiedenen Distanzen und Durchschnittsbildung über 20 Messwerte wurde eine Ta-

belle mit Signalstärken und Distanzen aufgebaut, die während des Experiments zur Distanzbestimmung genutzt wurde. War ein Tabellenwert nicht vorhanden, wurde dieser durch Interpolation der benachbarten Tabellenwerte berechnet. Als Auswertungsmetrik für die Güte der Lokalisierung wurde der MAE benutzt. Ein ähnlicher Aufbau wird von Blumenthal et al. [14] für die Evaluation von *Weighted Centroid Localization* in einem Outdoor-Test verwendet. Hier werden vier Chipcon CC2420DK an die Ecken eines 43 m × 43 m großen Feldes platziert und die Distanzen mittels Signalstärke zu einem weiteren Knoten an verschiedenen Rasterpunkten innerhalb des Feldes gemessen. Weitere Experimente dieser Art mit variierenden Feldgrößen und Ankeranordnungen finden sich zahlreich in der Literatur [16, 145, 22].

Nawaz und Trigoni verwenden ein Netzwerk aus 8 MIT Cricket [104] Knoten, welches innerhalb eines Raumes aufgebaut ist und eine sehr kleine Fläche von ca. 2 m × 2 m aufspannt. Der zu lokalisierende Knoten befindet sich zentral umringt von den übrigen Knoten, wobei alle Knoten statisch sind. Die Distanzen werden mittels TDOA eines kontinuierlich ausgesendeten Funk- und Ultraschallsignals ermittelt. Die Untersuchung der vorgestellten Algorithmen wurde mittels MAE durchgeführt, wobei einige Ultraschallsender so ausgerichtet wurden, dass sich mehrere NLOS-Verbindungen ergaben, um die Robustheit der Algorithmen zu testen. Ähnliche Experimente mit gleicher Hardware wurden von Moore et al. [93] durchgeführt, dort mit 16 und dann mit 40 zufällig ausgebrachten Knoten. Die Fläche war jedoch ebenfalls nicht sehr groß, maximal 5×6 m^2, die Position der Ankerknoten wurde manuell mittels eines aufgebrachten Bodenrasters bestimmt. Ein letzter Test umfasste die Lokalisierung eines mobilen Knotens, der auf einem autonomen Roboter angebracht war. Dieser bewegte sich innerhalb einer rechteckigen Fläche von 2×2 m^2, wobei sechs Ankerknoten kreisförmig um die rechteckige Fläche angeordnet waren. Dies war also ein ziemlich optimaler Aufbau, da es nur LOS-Verbindungen gab. Die Position des mobilen Knotens wurde mittels Videoaufzeichnung im Nachhinein manuell ermittelt und mit den gespeicherten Lokalisierungsdaten synchronisiert. Als Auswertungsmetrik wurde in allen Fällen der MSE benutzt. Wendeberg et al. [142] lokalisieren ebenfalls einen mobilen Tag mittels TDOA von Ultraschallsignalen. Dafür wurden auf einer Testfläche von 12×8 m^2 11 Ankerknoten in ovaler Ausbringung angeordnet. Der mobile Knoten bewegte sich mithilfe eines Modellautos innerhalb der Anker, die tatsächliche Position wurde mittels eines Videosystems bestimmt, welches an der Decke angebracht war. Eine Besonderheit war, dass die Position der Anker während einer Initialisierungsphase zur Laufzeit ermittelt wurde und nicht fest konfiguriert war.

Bahillo et al. führen zur Evaluation von RLSM einen Test unter realen Bedingungen in einem Bürogebäude durch, der schon eher einem realistischen Anwendungsfall für die Lokalisierung innerhalb von Gebäuden entspricht. Für den Test wurden 8 WLAN APs im Gebäude auf einer Fläche von ungefähr 50 m × 25 m verteilt, die sich über mehrere Räume und Flure erstreckt. Die APs wurden in verschiedene Räume gestellt und ein gerader Pfad über verschiedene Flure von ungefähr 68 m abgeschritten, eine direkte Sichtverbindung mit den APs bestand also nicht, jedoch befand sich der mobile Knoten immer in der konvexen Hülle der APs. Eine genaue Angabe über die Anzahl an Messungen bzw. Positionsabschätzungen ist der Veröffentlichung von Bahillo et al. nicht zu entnehmen [6], eine gezeigte Abbildung vom Versuchsaufbau lässt auf ungefähr 100 Positionsabschätzungen schließen. Wie die tatsächliche Position an den Messpunkten ermittelt wurde, ist nicht erwähnt. Zur Bestimmung der Distanzen wurden RSS und *Round Trip Time* (RTT) Messungen verwendet, die von einem IEEE 802.11 b WLAN-Adapter zu den APs durchgeführt wurden. Um genauere RTTs zu messen, wurde eine zusätzliche Platine [7] entwickelt und mit dem WLAN-Adapter verbunden. Ähnliche Versuche werden in [8, 157, 24, 41, 90] und [65] durchgeführt, um die dort vorgestellten Lokalisierungssysteme und Algorithmen zu testen. Diese beruhen auf dem Fingerprinting der RSS-Werte, die mit verschiedenen WLAN APs gemessen wurden. Bahl und Padmanabhan untersuchen unter anderem auch, wie sich die Mobilität des zu lokalisierenden Knotens auf die Genauigkeit auswirkt [8]. Hierzu wurden vier Signalstärkemessungen pro Sekunde während eines Laufes durch das Gebäude aufgenommen und mit den Messungen an ausschließlich fixen Punkten verglichen. Hierbei wurde eine leichte Verschlechterung der Genauigkeit der Lokalisierung von 16 % festgestellt.

Von Zanca et al. werden entfernungsbasierte Lokalisierungsalgorithmen in einem Testbed verglichen [159], um insbesondere die Performance der Algorithmen unter echten Bedingungen mit verschiedenen Ankeranzahlen zu testen. Bei den Algorithmen handelt es sich um Min-Max, Multilateration, MLE und ROCRSSI [81], der Vergleich erfolgt über den MAE. Das Testbed befand sich in einem Raum mit einer Dimension von 10×10 m^2 und bestand aus 48 fixen Knoten, die gitterförmig angeordnet waren. Die Ankeranzahl variierte von 3 bis 25, die Distanzen wurden mithilfe eines Pfadverlust-Modells aus den RSSI-Werten errechnet. Eine ähnliche Analyse wird von Kung et al. [69] mit einem C-förmig angeordneten WSN, bestehend aus 37 fixen Knoten, in einem Outdoor-Test unter Freifeldbedingungen durchgeführt. Auch hier werden die Distanzen aus den RSSI-Werten errechnet. Von Rallapalli et al. [106] wird ein

vorgestellter Lokalisierungsalgorithmus in einem *Multi-Hop*-Sensornetz mit 25 bzw. 36 MICA2 Knoten in einem Gebäude untersucht. Es werden jeweils zwei Experimente durchgeführt, eines mit den Knoten auf einer quadratischen Fläche von $5{,}5 \times 5{,}5$ m^2 und zufälliger Anordnung, das andere auf einer Fläche von $6{,}5 \times 6{,}5$ m^2 mit C-förmiger Platzierung der Knoten. Die Sendereichweite der Knoten wurde auf 2,3 m justiert und in jedem Experiment 6 Anker zufällig unter den Knoten ausgewählt. Die Distanzen wurden auf Basis der durchschnittlichen Signalstärke über mehrere Messungen ermittelt. Eine Besonderheit bei diesem Experiment war, dass die Knoten einen gewissen Grad an Mobilität besaßen, indem die Knoten in bestimmten Intervallen auf vorher errechnete Positionen per Hand platziert wurden. Hierzu wurde ein *Random Waypoint* Mobilitätsmodell verwendet, insgesamt gab es 15 Zeitintervalle. Die Auswertung erfolgte dann über diese 15 Intervalle und umfasste somit verschiedene Knoten- und Ankerkonstellationen.

Johnson et al. stellen ein WSN Testbed vor [61], das neben 25 fest installierten Knoten an der Decke und einigen Knoten in Bodennähe auch 6 Knoten besitzt, die auf Robotern befestigt sind und somit frei in dem 60 m^2 großen, L-förmigen Gebäudeteil bewegt werden können. Bei allen Knoten handelt es sich um MICA2 Sensorknoten. Die Positionierung der Roboter kann interaktiv durch den Benutzer erfolgen oder geskriptet werden. Die Genauigkeit der Positionierung des Robotersystems liegt im unteren einstelligen Zentimeterbereich. Auf allen Knoten kann beliebige Software programmiert werden, Experimente können auch aus der Ferne aufgesetzt und ferngesteuert werden. Dies ermöglicht natürlich auch Lokalisierungsexperimente. Um die Möglichkeiten des Testbeds aufzuzeigen, wird ein Distanzmessungsexperiment mit einigen fest installierten Knoten beschrieben. Obwohl schon seit 2006 in Betrieb, sind uns unter den bisherigen Veröffentlichungen, welche auf einer Webseite zusammengefasst sind, keine Lokalisierungsexperimente bekannt, die insbesondere die gebotene Mobilität mehrerer Knoten nutzen.

Von He et al. wird ein anderer Ansatz zur experimentellen Auswertung der TOA-basierten Indoorlokalisierung verfolgt [50]. Dort wird ein Testbed vorgestellt, mit dem sich die Effekte der Mehrwegeausbreitung wiederholt unter gleichen Bedingungen simulieren lassen, aber trotzdem reale Sensorknoten für die Auswertung benutzt werden. Herz des Testbeds ist ein Funkkanalemulator, mit dem acht Kanäle simultan emuliert werden können. Zur Auswertung werden fünf Nanotron Technologies *nanoLOC Development Kits* verwendet, die die Entfernungen mittels TOF bestimmen und die gleichen Funkchips wie in unserer Auswertung in Kapitel 7 besitzen. Untersucht

wird die Genauigkeit der Distanzmessung und der Lokalisierung mit einem festen Algorithmus für das Testbed und abschließend verglichen mit einigen tausend empirischen Messungen im gleichen Testgebäude. Dafür werden vier verschiedene Mehrwegebedingungen untersucht: Freiraum, LOS und zwei NLOS-Situationen. Für den Vergleich wird die kumulative Verteilungsfunktion des Distanzfehlers verglichen sowie der durchschnittliche Fehler und die Standardabweichung. Die Auswertung der beiden Ansätze zeigte dabei eine starke Übereinstimmung der Ergebnisse. Die Lokalisierung wird jedoch nur im LOS-Fall mit vier Ankern in den Ecken eines 14×7 m^2 großen Raumes untersucht. Der Tag nahm dabei 25 feste, gitterförmig verteilte Positionen ein.

Tabelle 5.4 zeigt noch einmal eine Zusammenfassung der ausgewählten Lokalisierungsexperimente geordnet nach Jahr und Autor. Auch wenn die gewählten Experimente nur eine Stichprobe darstellen, so zeigt sich dennoch, dass die meisten Experimente zur Lokalisierung die Auswertung der Signalstärke (RSS) oder der Zeitdifferenz eines ausgesendeten Ultraschall- und Funksignals (TDOA) benutzen. Als Hardware werden hierbei meistens WLAN APs verwendet oder frei verfügbare Sensorknoten wie MICA2 oder Cricket. Nur selten wird Spezialhardware zur Messung der TOF oder RTOF verwendet. Dies mag zum einen an den Kosten liegen, zum anderen aber auch an der bereits vorhanden Infrastruktur an WLAN APs. Als Auswertungsmetrik für die Güte der Lokalisierung wird meistens der MAE verwendet, häufig in Verbindung mit dem minimalen oder maximalen Fehler, nur selten wir der Median (MED) oder MSE benutzt.

Die durchgeführten Experimente wurden in der Mehrzahl der Fälle auf freier Fläche, z. B. auf Parkplätzen im Outdoor-Fall bzw. in einem Raum auf kleiner Fläche im Indoor-Fall durchgeführt. Nur bei Fingerprinting-Verfahren wurden häufig Versuche auf größerer Fläche durchgeführt, die sich auch über mehrere Räume erstreckte. Als Folge ergeben sich dadurch vermehrt bzw. teilweise ausschließlich LOS-Verbindungen, was natürlich die Genauigkeit der Lokalisierung positiv beeinflusst und die Schwächen mancher Algorithmen ungenügend offenlegt. Dies kann man auch an der unterschiedlichen Ankeranzahl zur verwendeten Versuchsfläche sehen, explizit angegeben durch die Fläche, die ein einzelner Anker einnimmt. Je kleiner dieser Wert, desto höher ist die Ankerdichte und damit folglich auch die zu erwartende Genauigkeit der Lokalisierung. Dies macht es natürlich schwierig, die Ergebnisse der Experimente für

Tabelle 5.4: Vergleich ausgewählter Lokalisierungsexperimente

Quelle	**Jahr**	**Indoor**	**Outdoor**	**Mobilität**	**#Anker, N**	**Fläche [m^2], A**	**Ein Anker in A/N m^2**	**Technologie**	**Metrik**
Bahl und Padmanabhan [8]	2000	✓		◐	3	43,5 × 22,5	326	RSS	MAE
Bulusu et al. [16]	2000		✓	○	4	10 × 10	25	-	MAE, MAX
Moore et al. [93]	2004	✓			16	2 × 2	0,25	TDOA	MSE
					40	5 × 6	0,75		
				◐	6	2 × 2	0,67		
Liu et al. [82]	2005		✓		7	12 × 12	20,6	RSS	MAE
Whitehouse et al. [145]	2005		✓		25	13 × 13	6,8	RSS, TDOA	MED
					49	50 × 50	51		
Crepaldi et al. [22]	2006	✓	✓		48	9 × 10	1,9	RSS	MAE
Johnson et al. [61]	2006	✓			6	60	10	TDOA	MIN, MAE
King et al. [65]	2006	✓		○	6	36 × 15	90	RSS	MAE
Blumenthal et al. [14]	2007		✓	○	4	43 × 43	462	RSS	MAE, MAX
Youssef und Agrawala [157]	2008	✓		○	21	68,2 × 25,9	84,1	RSS	MAE
					6	11,8 × 35,9	70,6		
Zanca et al. [159]	2008	✓		○	3-25	10 × 10	33-4	RSS	MAE
Kung et al. [69]	2009		✓		4	25 × 25	156	RSS	MAE
Bahillo et al. [6]	2010	✓		◐	8	50 × 25	156	RSS, RTT	MAE

○ Zu lokalisierende Knoten nahmen verschiedene feste Positionen ein
◐ Zu lokalisierende Knoten waren mobil
● Alle Knoten waren mobil

Tabelle 5.4: Vergleich ausgewählter Lokalisierungsexperimente (Fortsetzung)

Quelle	**Jahr**	**Indoor**	**Outdoor**	**Mobilität**	**#Anker, N**	**Fläche [m^2], A**	**Ein Anker in A/N m^2**	**Technologie**	**Metrik**
Cypriani et al. [24]	2010	✓		○	4	30×10	75	RSS	MAE
Nawaz und Trigoni [96]	2010	✓			7	2×2	0,57	TDOA	MAE
Rallapalli et al. [106]	2010	✓		○	6	$6{,}5 \times 6{,}5$	7	RSS	MAE
Giorgetti et al. [41]	2011	✓		○	18	2000	111	RSS	MAE
Meng et al. [90]	2011	✓		○	6	-	-	RSS	MAE
Wendeberg et al. [142]	2012	✓		◐	11	12×8	8,7	TDOA	MAE
He et al. [50]	2013	✓		○	4	14×7	24,5	TOA	CDF, MAE

○ Zu lokalisierende Knoten nahmen verschiedene feste Positionen ein
◐ Zu lokalisierende Knoten waren mobil
● Alle Knoten waren mobil

verschiedene Algorithmen miteinander zu vergleichen. In der Mehrzahl der Fälle war das untersuchte Netzwerk komplett statisch oder der zu lokalisierende Knoten nahm nur verschiedene feste Positionen ein. Dies könnte ebenfalls Einfluss auf das Endergebnis haben, da eventuell nicht genügend Positionen oder nur ausgewählt gute Positionen untersucht werden. Bei einer kontinuierlichen Bewegung mit eingestellter Samplerate ist die Wahrscheinlichkeit hierfür geringer. In allen Experimenten mit vorhandener Mobilität wurde genau ein Tag verwendet. Experimente, bei denen alle Knoten mobil sind oder mehrere Tags verwendet werden, die dann ebenfalls als Anker dienen, wurden von uns in der Literatur nicht gefunden. Dies mag vor allem an den eingangs beschriebenen hohen Kosten bzw. dem hohen Aufwand zur Durchführung solcher Versuche liegen.

Weiterhin hatten alle Experimente gemein, dass die untersuchten Positionen bzw. der abgeschrittene Pfad immer innerhalb der konvexen Hülle der ausgebrachten Ankerknoten lag. Dies begünstigt unter Umständen einige Algorithmen überproportional stark, da hierbei keine ungünstigen geometrischen Konstellationen betrachtet werden. Wie von Langendoen und Reijers untersucht [73], hängt z. B. die Genauigkeit von Min-Max von der Dichte des Netzwerks und der eigenen Position zu den Ankerknoten extrem ab. Außerhalb der konvexen Hülle der Anker fällt die Genauigkeit von Min-Max stetig ab. Deshalb sollten bei einer experimentellen Betrachtung von Algorithmen auch Extremfälle untersucht werden und bei einem Vergleich der Algorithmen berücksichtigt werden.

5.2.2 LatViz als Werkzeug für die experimentelle Algorithmenauswertung

Um die von uns durchgeführten Experimente effizient auswerten zu können, haben wir das Auswertungswerkzeug *LatViz* entworfen und implementiert. Das Programm ist in Java geschrieben und benutzt als Basis die in Abschnitt 3.5 vorgestellte Bibliothek *LatMath*. Abbildung 5.6 zeigt die grafische Benutzeroberfläche des Programms. Ziel von *LatViz* ist es, die durchgeführten und aufgezeichneten Lokalisierungsexperimente zum einen visuell darzustellen (vgl. Abbildung 5.6) und zum anderen numerisch auszuwerten (vgl. Abbildung 5.7).

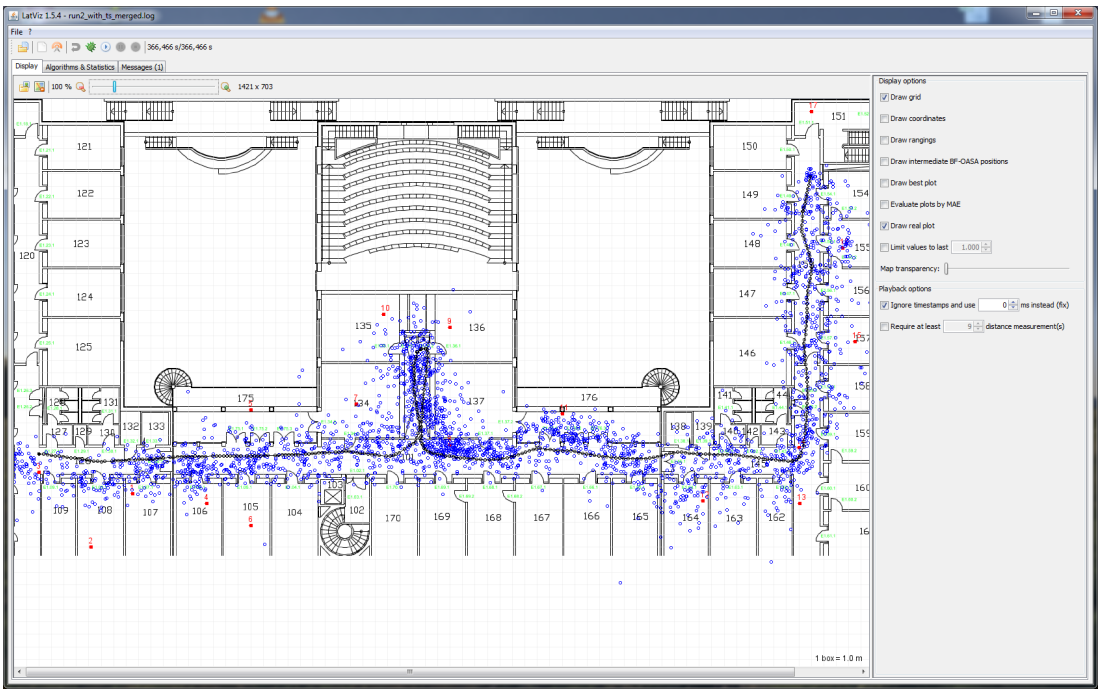


Abbildung 5.6: Grafische Benutzeroberfläche von *LatViz*. Zu sehen ist ein Experimentaufbau und die errechneten Positionen eines Algorithmus.

Genereller Auswertungsablauf

Als Programmeingabe dienen Textdateien, die die Ankerkonfiguration des Experiments (Position der Anker) sowie die gemessenen Distanzen von einem mobilen Knoten zu den Ankern enthalten. Soll neben der visuellen Auswertung auch eine numerische Auswertung erfolgen, muss die Datei zusätzlich die Referenzpositionen für die einzelnen Messzeitpunkte enthalten. Eine Zeile der Datei entspricht einem Messzyklus und hat den Aufbau $[ts; r_1, \ldots, r_n; x_m, y_m]$, wobei ts der Messzeitpunkt in Millisekunden, r_i die gemessene Distanz zum i-ten Anker und (x_m, y_m) die optionale Referenzposition des mobilen Knotens ist.

Als weitere Eingabemöglichkeit wurde eine *Representational State Transfer* (REST) Bibliothek entwickelt, die den direkten Zugriff auf die im virtuellen Testbed gespeicherten Experimente ermöglicht. Eine ausführliche Betrachtung dieses Testbeds erfolgt in Abschnitt 5.3.

Abbildung 5.7 zeigt die Konfigurationsmöglichkeiten für eine Algorithmenauswertung mittels *LatViz*. Algorithmen, die ausgewertet werden sollen, lassen sich zu einer Liste hinzufügen und anschließend konfigurieren. *LatViz* stellt automatisch alle Algorith-

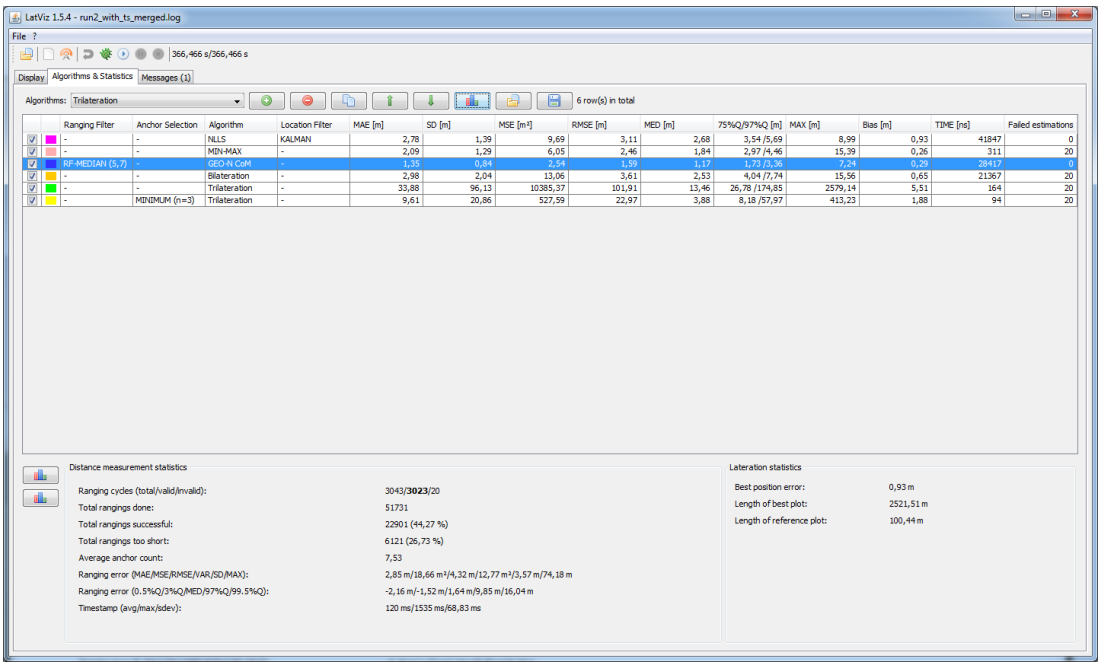


Abbildung 5.7: Konfiguration und Statistiken von *LatViz*

men zur Auswahl zur Verfügung, welche in der *LatMath* Bibliothek enthalten sind. Diese Algorithmen implementieren verschiedene Schnittstellen, damit beispielsweise eine nachträgliche Anpassung der Parameter eines Algorithmus oder die Speicherung aller getätigten Einstellungen möglich ist. Neben der reinen Auswertung der Algorithmen lassen sich Algorithmen auch in Kombination mit Filtern für die gemessenen Distanzen und geschätzten Positionen sowie Ankerauswahlalgorithmen auswerten. In Abbildung 5.7 kann man z. B. sehen, dass der NLLS Algorithmus einen Kalman-Filter auf den berechneten Positionen nachgeschaltet hat, während der Geo-n Algorithmus einen Medianfilter einer gewissen Fenstergröße auf den gemessenen Distanzen besitzt. Im Falle der Trilateration kann man einen einfachen Ankerauswahlalgorithmus sehen, der in diesem Fall immer die drei dichtesten Anker auswählt und damit deutlich bessere Ergebnisse erzielt als der reine Algorithmus an sich.

Jede Zeile der Tabelle aus Abbildung 5.7 bildet somit eine Verarbeitungskette, die von links nach rechts ausgewertet wird. Ausgaben einer Stufe bilden die Eingaben der folgenden Stufe, wobei nur der Algorithmus obligatorisch ist. Dieser erhält im Wesentlichen die Position der Anker und die gemessenen Distanzen aus der Eingabedatei oder ggf. von der Ankerauswahl als Parameter. Die Filter und die Ankerauswahl können wie bei den Algorithmen ebenfalls vom Benutzer konfiguriert werden, sofern diese dies unterstützen, wobei auch hier alle Einstellungen gesichert werden können.

Filter und Ankerauswahlalgorithmen werden bei Programmstart aus der *LatMath* Bibliothek ermittelt und geladen.

Als Endergebnis der Verarbeitungskette liefert der Algorithmus oder ggf. der Positionsfilter eine finale Position für die Auswertungs- und Visualisierungskomponente des Programms zurück.

Weitere Programmfeatures

Eine geladene Datei lässt sich abspielen, das Tempo des Fortschritts richtet sich dabei nach den Zeitstempeln in der Datei, d. h. nach der echten Aufnahmegeschwindigkeit während der Datenerfassung. Es lässt sich jedoch auch eine frei definierbare Verzögerung zwischen zwei Messzyklen einstellen, so dass die Auswertung allein durch die Komplexität der Algorithmen und Filter beschränkt wird. Eine Wiedergabe kann pausiert werden und die Wiedergabe ist auch in Einzelschritten möglich. Da die Ergebnisse in der grafischen Anzeige des Netzwerkes in Echtzeit dargestellt werden, lassen sich so einzelne Lokalisierungssituationen analysieren und das Verhalten der Algorithmen erkennen und evtl. optimieren. Sämtliche Statistiken lassen sich zurücksetzen während die Wiedergabe pausiert ist, so dass auch bestimmte Teilstücke eines aufgenommen Pfades für sich analysiert werden können. Auch das Öffnen und Wiedergeben von mehreren Dateien ist möglich. Die numerische Auswertung bildet dann den Durchschnitt über alle geöffneten Dateien. Weitere wesentliche Merkmale sind in folgender Auflistung zu finden:

- Die berechneten Positionen einer Verarbeitungskette werden als farbige Punkte in der Netzwerkanzeige dargestellt. Die Farbe kann pro Tabellenzeile frei festgelegt werden. Einzelne Algorithmen können auch komplett ausgeblendet werden sowie die Anzahl der einzuzeichnenden Positionen auf eine bestimmte Zahl limitiert werden.
- Die Anker werden als rote Quadrate mit entsprechender Nummer eingezeichnet. Die Referenzpositionen werden als schwarze Punkte eingezeichnet (falls vorhanden). Zusätzlich können die Distanzmessungen als grüne Kreise um die Anker eingezeichnet werden. Entsprechende Anker werden dann ebenfalls in Grün dargestellt.
- Zur besseren Auswertung lassen sich Karten in gängigen Grafikformaten einbinden (vgl. Abbildung 5.6) sowie ein Meterraster einblenden. Die Anzeige lässt

sich durch die Einstellung eines Zoomfaktors vergrößern oder verkleinern.

- Die in den Dateien vorhandenen Anker lassen sich vor Beginn der Wiedergabe filtern, d. h. bestimmte Anker können aus der Betrachtung ausgeklammert werden.
- Die numerischen Werte zur Beurteilung der Genauigkeit und Präzision der Algorithmen umfassen den MAE, die Standardabweichung, den MSE, den RMSE, den Median, das 75 % und 97 % Quantil sowie den maximalen Fehler. Diese Metriken werden während der Wiedergabe berechnet und in der Tabelle angezeigt. Zusätzlich lässt sich das Histogramm des Positionsfehlers grafisch darstellen.
- Die Tabelle enthält weiterhin die durchschnittliche Ausführungszeit eines Algorithmenaufrufs in Nanosekunden sowie die Anzahl der fehlgeschlagenen Lokalisierungen. Eine Lokalisierung kann z. B. fehlschlagen, wenn nicht genügend Ankerknoten in Reichweite waren oder der Algorithmus selber keine Position anhand der Eingabe berechnen kann.
- Es werden ebenfalls eine Reihe von Metriken über die Distanzmessungen berechnet (MAE, Standardabweichung, Varianz, MSE, RMSE, Maximalfehler, 0,5 % Quantil, 3 % Quantil, 50 % Quantil, 97 % Quantil und 99,5 % Quantil). Zusätzlich lassen sich beispielsweise das Histogramm des Distanzmessfehlers, der durchschnittliche Distanzmessfehler pro Anker sowie die Rohdaten der Distanzmessungen pro Anker grafisch darstellen.
- Die berechneten Histogrammdaten sowie Positionen der Algorithmen lassen sich für die Weiterverarbeitung in anderen Programmen als CSV-Dateien exportieren.
- Alle getätigten Einstellungen zur Konfiguration der Algorithmen und Filter lassen sich für eine erneute Verwendung abspeichern.

Bewertung von Fehlermodellen für die Nachbildung von Distanzmessfehlern

Eine weitere nützliche Eigenschaft von *LatViz* ist dessen generelle Eignung, Fehlermodelle in Filtern für die Distanzmessungen einzusetzen, wie sie beispielsweise in LS^2 zur Simulation der Distanzmessfehler verwendet werden. Hierbei wird die gemessene Distanz aus der Datei durch eine neu berechnete Distanz ersetzt, die sich aus echter Distanz zwischen Anker und mobilem Knoten zuzüglich eines Offsets aus dem verwendeten Fehlermodell ergibt. Dies ermöglicht neben der Evaluierung der Algorithmen

unter anderen Umweltbedingungen die Bewertung der Güte der Fehlermodelle, welche z. B. für die Nachbildung eines bestimmten realen Szenarios entwickelt wurden.

So haben wir mithilfe eines Werkzeugs von Reinecke [108] ein Fehlermodell entwickelt, das die Fehlerverteilung des Funkchips unserer Experimenthardware aus Kapitel 7 möglichst genau abbildet. Hierfür wurde eine Erlang-Verteilung benutzt. Hierbei zeigte sich, dass trotz nahezu identischer Kenngrößen (MAE, MSE etc.) bei der Distanzmessung die resultierenden Positionen in einigen Bereichen des abgeschrittenen Pfades im Quervergleich merklich abwichen. Dies war vor allem in Bereichen der Fall, die starke NLOS-Bedingungen aufwiesen (z. B. in der Nähe von Brandschutztüren und -wänden).

5.3 Ein virtuelles Testbed für die Indoorlokalisierung

Um die Schwelle für die Nutzung von aussagekräftigen Experimenten wesentlich zu senken, haben wir das Konzept eines virtuellen Testbeds entwickelt. Das virtuelle Testbed ist eine offene Plattform, die im Wesentlichen der Speicherung und Veröffentlichung von Messdaten mit den dazugehörigen *Groundtruth*-Daten dient. Grundidee dabei ist es, in einem Gebäude ein Experiment unter Nutzung eines Referenzsystems durchzuführen und zu protokollieren, nur dass anders als bei einem herkömmlichen Experiment kein einzelner Pfad durch das Gebäude aufgezeichnet wird, sondern möglichst Daten aus dem ganzen Gebäude erfasst werden. Dazu wird das Gebäude in gleichverteilte Rasterpunkte mit einer bestimmten Rasterweite aufgeteilt und dann an jedem dieser Punkte mehrere Messungen durchgeführt. Nach dem Einpflegen dieser Daten in eine Datenbank, können auf Basis dieser Daten später virtuelle Pfade durch das Gebäude berechnet werden, welche dann mit echten Messdaten und den dazugehörigen *Groundtruth*-Daten ausgeliefert werden.

Um das System zu testen und ein Benchmark der hier besprochenen Algorithmen durchzuführen, haben wir das virtuelle Testbed mit Messdaten aus dem Gebäude der Informatik an der FU Berlin gefüllt. Ziel war es, das Gebäude in ein 10 cm Raster zu unterteilen und von jedem dieser Rasterpunkte mindestens 100 Messwerte von allen erreichbaren Ankerknoten aufzunehmen. Einschränkend muss allerdings gesagt werden, dass diverse Positionen für das Referenzsystem nicht erreichbar waren, so dass eine Vollabdeckung des gesamten Gebäudes nicht möglich war. Allerdings ist dies keine wirkliche Einschränkung, da diese Punkte aufgrund von Möbeln, technischen

Anlagen etc. auch durch eine zu lokalisierende Person nicht erreichbar gewesen wären. Die so erlangten Daten bieten den Grundstock für alle weiteren Experimente und für eine tiefere Analyse des zugrunde liegenden Distanzmesssystems.

5.3.1 Das Referenzsystem

Eine wesentliche Schwierigkeit bei der Durchführung und Bewertung von Experimenten zur Indoorlokalisierung ist die Gewinnung von *Groundtruth*-Daten. Stattet man eine Testperson mit einem Lokalisierungssystem aus, welches es zu testen gilt, so kann man die durch das System ermittelten Positionen zwar mittels einer Karte visualisieren und grob die Genauigkeit des Systems abschätzen - für einen Vergleich mehrerer Systeme ist dieser Ansatz aber viel zu ungenau. Der naive Ansatz, den Probanden einen vorher genau definierten Pfad ablaufen zu lassen, scheitert an der nur mit großem Aufwand einzuhaltenden Geschwindigkeitsvorgabe. Ein solcher Ansatz lässt zwar zu, Positionsfehler, die orthogonal zum festgelegten Pfad liegen, ziemlich genau zu ermitteln, eine Aussage über Positionsfehler, die dieses Orthogonalitätskriterium nicht erfüllen, ist ohne eine exaktes Zuordnen von Positionen zu Zeitpunkten jedoch kaum möglich.

Um dieses Problem zu lösen, haben wir uns für einen anderen Ansatz entschieden: Wir setzen für die Gewinnung von *Groundtruth*-Daten ein weiteres Lokalisierungssystem ein, welches in der Genauigkeit wesentlich über dem zu evaluierenden System liegt. Als Referenzsystem haben wir uns für einen Hybriden aus optischen und odometrischen System entschieden. Wir haben einen mobilen Roboter mit mehreren Tiefenbildkameras ausgestattet, welche die Konstruktion eines dreidimensionalen Bildes der Umgebung ermöglichen. In Verbindung mit einem genauen Lageplan des Gebäudes lässt sich mit diesen Informationen und dem *Adaptive Monte Carlo Localization* (AMCL) Algorithmus [35] eine sehr genaue Lokalisierung vornehmen. Für die Lösung des allgemeinen Problems der Indoorlokalisierung ist dieser Ansatz natürlich aufgrund der speziellen Hardwareanforderungen nicht geeignet. Als Roboterplattform kommt dabei ein *Turtlebot* von Willow Garage und das Robot Operating System (ROS) zum Einsatz [105].

Das Referenzsystem erzielte bei allen Versuchen in unserem Gebäude einen mittleren Fehler von ca. 7 cm und liegt damit weit über der Genauigkeit der Systeme, die wir mittels des virtuellen Testbeds testen wollen. Diese 7 cm sind jedoch bei allen Ver-

gleichstests, die auf diesem Testbed basieren, als Grundrauschen anzunehmen. Eine detaillierte technische Beschreibung dieses Systems haben Schmitt et al. beschrieben [118, 119].

Eine weitere Einschränkung dieses Systems ist die nicht zu 100 % gegebene Übertragbarkeit der gewonnen Messwerte auf den Fall eines sich durch ein Gebäude bewegenden Menschen. Während ein Mensch ein Gerät zur Entferungsbestimmung immer zu ca. 50 % durch den eigenen Körper dämpfen würde, trifft dieses auf den verwendeten Roboter nur eingeschränkt zu. Zwar dämpft auch dieser das Signal etwas ab, jedoch wahrscheinlich nicht im selben Maße wie der Körper eines erwachsenen Menschen. Wir haben bei der Aufnahme der Testdaten diesen Fall dahingehend berücksichtigt, dass wir ein Subset der Sampledaten aufgenommen haben, während eine Person stets hinter dem Roboter hergelaufen ist. Diese Daten sind in unserer Datenbank besonders gekennzeichnet.

5.3.2 Die Datenbank

Da auch bei kleineren Gebäuden oder Räumen eine große Zahl an Messdaten anfällt, ist es nötig die Daten so abzulegen, dass ein schneller Zugriff und eine Filterung nach den später benötigten Kriterien im Vordergrund stehen. Wir haben uns daher für eine SQL-Datenbank entschieden. Anstatt durch verschiedene Benchmarks das für unsere Zwecke am besten geeignete Datenbanksystem zu ermitteln, haben wir uns für das Datenbanksystem entschieden, das die beste Unterstützung für zweidimensionale Punktdaten bietet - PostgreSQL (Version 9.1.9)[127] mit der Erweiterung PostGIS (Version 2.0.3)[126].

Beim Datenbankdesign wurde ebenfalls auf eine möglichst performante Abfragemöglichkeit der eigentlichen Messdaten Rücksicht genommen. So wurde die Tabelle mit den Messdaten komplett denormalisiert, so dass bei allen Abfragen auf die eigentlichen Pfaddaten keinerlei *Joins* nötig sind. Für statistische Zwecke wurden neben den gemessenen Entfernungen und den *Groundtruth*-Daten zusätzlich noch der sich daraus ergebende Messfehler, die MAC-Adressen der beiden beteiligten Knoten und eine Experimentkennung sowie eine Gebäudenummer mit erhoben. Mit diesen Daten ist es möglich, in den vorhandenen Daten nach Knoten zu suchen, die einen bestimmten *Bias* aufweisen oder zwei verschiedene Messsysteme zu vergleichen.

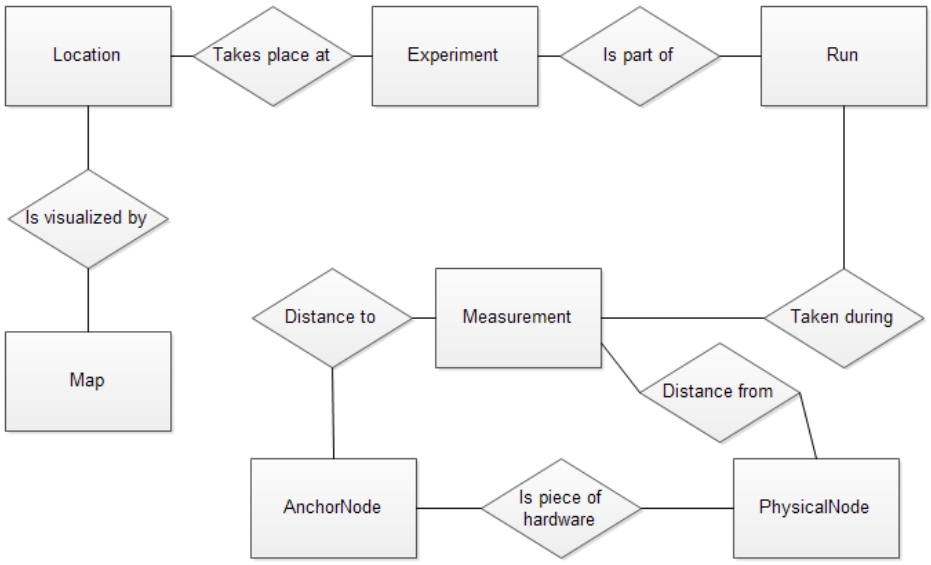


Abbildung 5.8: Datenbankrelationen des virtuellen Testbeds

Die wesentlichen Entitäten haben dabei die folgenden Hintergründe:

location Der Ort der Messung mit Bezeichnung und Textbeschreibung. Dient hauptsächlich zur Filterung in einem Frontend.

map Eine Karte als Bitmap. Die konkrete Bitmap ist dabei als Pfad des lokalen Dateisystems angegeben. Zusätzlich sind diverse Kartenparameter, die für eine grafische Darstellung benötigt werden, gespeichert - insbesondere sind dies Skalierungsfaktoren, Kartenursprung und Drehung. Die Drehung wird benötigt, wenn die Bitmap ein Gebäudeplan ist, der den Kartennorden nicht an der Oberseite hat.

experiment Ein Experiment ist die grundlegende Entität für eine Datenerhebung in einem Gebäude (einer *location*) mit feststehender Ankerkonfiguration.

run Ein Lauf ist die Entität für eine kontinuierliche Aufnahme von Messdaten.

physical_node Die Bezeichnung eines konkreten Knotens inkl. verwendeter Hardware und MAC-Adresse.

anchor_node Ein Ankerknoten mit seiner Position.

measurement Die wesentliche Entität, in welcher neben der gemessenen Entfernung

zu genau einem Ankerknoten, der Verweis auf diesen Knoten, der *Groundtruth*-Position und der konkrete Knoten gespeichert werden.

Die Relationen sind in Abbildung 5.8 dargestellt.

5.3.3 Das Frontend

Um Daten abzufragen und einzufügen, existieren zwei Frontends, die zum Teil aufeinander aufsetzen. Das erste Frontend ist eine einfache REST-API [34], mithilfe derer auf alle Entitäten der Datenbank lesend zugegriffen werden kann. Ebenfalls ist über diese API eine Authentifizierung möglich. Authentifizierte Benutzer dürfen alle Entitäten auch schreiben. Über diese API ist es möglich weitere auf der Datenbank aufsetzende Tools zu entwickeln und eine Vielzahl an Lokalisierungssystemen direkt in die Datenbank schreiben zu lassen, da REST Bibliotheken für eine Vielzahl von Programmiersprachen und Betriebssystemen verfügbar sind. So haben wir für die Implementierung des Referenzsystems eine Software in Java geschrieben, die direkt mit dem ROS und den Sensorknoten kommuniziert, die Referenzdaten mit den Messwerten synchronisiert und direkt über eine drahtlose Verbindung in die Datenbank schreibt.

Das eigentliche Frontend ist eine mittels *Google Web Toolkit* (GWT) [43] implementierte Webanwendung, die im Wesentlichen die Aufgabe hat es dem Benutzer zu ermöglichen, virtuelle Experimente durchzuführen und auf die gesammelten Daten zuzugreifen. Neben der grafischen Konstruktion der benötigten Pfade, an einem der in der Datenbank existierenden Orte, kann sich über dieses Frontend der Zustand der Datenbank angesehen werden und einfache statistische Auswertungen auf der Datenbasis vorgenommen werden. Ebenfalls können hier Karten hochgeladen und eingenordet werden sowie Experimente erstellt und bearbeitet werden. Sofern das System, mit dem neue Messdaten erhoben werden, direkt in die Datenbank schreibt, kann hier auch live der Fortschritt dieser Arbeiten beobachtet werden. Mittels Heatmap-Karten kann die Abdeckung eines Gebäudes mit Messdaten hier schnell und effizient ausgewertet werden. Alle Zugriffe dieses Frontends auf die Datenbank finden dabei über die REST-API statt.

Tabelle 5.5: Reales vs. virtuelles Experiment

	Real	Virtuell
Entfernungsmessungen	4977	3981
Durchschnittliche Ankeranzahl	8.69	8.01
Durchschnittlicher Distanzfehler	1,85 m	1,94 m

5.3.4 Evaluation

Um das generelle Funktionsprinzip und die Validität der Daten aus dem virtuellen Testbed zu untersuchen, haben wir einige Fahrten durch das Gebäude mit dem Referenzsystem, mit den Ergebnissen desselben Pfades - diesmal aus dem Testbed generiert - verglichen. Die in Tabelle 5.5 gezeigten Ergebnisse zeigen deutlich, dass der Ansatz funktioniert und sich keine wesentlichen Unterschiede zwischen der Lokalisierungsqualität der aus dem virtuellen Testbed gewonnenen Daten und denen der „echten“ Experimente ergibt. Dies ist umso erstaunlicher, als dass zwischen den Experimenten ein gewisser Zeitraum lag und nicht auszuschließen ist, dass sich diverse Einflussfaktoren (Menschen, Türen, Fenster) in dem Gebäude sich derweil geändert haben. Eine Beispielauswertung der Experimente mit dem NLLS Algorithmus hat eine Abweichung der Positionsgenauigkeit von unter einem Prozent ergeben. Der Ausreißern gegenüber wesentlich anfälligere LLS Algorithmus zeigt eine Abweichung von knapp drei Prozent. Diese Abweichungen liegen alleine schon im Bereich der Standardabweichung der Genauigkeit des Referenzsystems und sind daher als nicht signifikant zu betrachten.

5.3.5 Ausblick

Das gesamte Projekt des virtuellen Testbeds ist aktuell noch einem steten Entwicklungsprozess unterworfen. Auch wenn sämtliche Experimente dieser Arbeit unter Zuhilfenahme des Testbeds und des Referenzsystems entstanden sind, gibt es viele Punkte in diesem Projekt, die noch nicht fertig gestellt sind bzw. einer Verbesserung bedürfen. Von daher führen wir dieses Projekt im Rahmen dieser Arbeit nur so weit ein, wie es zum Verständnis der Arbeit und zur Nachvollziehbarkeit der Experimente benötigt wird. Die Qualität der Daten und die darauf basierenden Aussagen sind jedoch nicht von dem frühen Stadium des Testbeds beeinträchtigt, so dass wir sie im

Rahmen dieser Arbeit mit gutem Gewissen nutzen konnten.

Für die Zukunft wird sich das virtuelle Testbed nicht nur zum Benchmarken von Indoorlokalisierungssystemen eignen, sondern hoffentlich auch einen weiteren Beitrag zu einem besseren Verständnis spezifischer Fragestellungen aus dem Bereich der Distanzmessung liefern. Alleine durch die immense Menge an Messdaten in Verbindung mit präzisen Karten und *Groundtruth*-Werten lassen sich durch Ansätze wie Datamining oder anderen statistischen Verfahren weitere Forschungsarbeiten durchführen. Das virtuelle Testbed wird aktuell durch S. Adler et al. an der Freien Universität Berlin weiterentwickelt [119, 1].

KAPITEL 6

Evaluation der Algorithmen anhand der räumlichen Fehlerverteilung

6.1 Simulationssetup

Für die Simulation der räumlichen Fehlerverteilung haben wir vier Szenarien mit jeweils zwei verschiedenen Fehlermodellen simuliert. Als Simulationsumgebung kam der LS^2-Simulator zum Einsatz. Für alle Simulationen im normalen Modus (siehe Kapitel 5.1.2) haben wir 320 Durchläufe simuliert und den Durchschnittswert des Positionsfehlers visualisiert. Für einige Algorithmen mit sehr hohem Rechenaufwand stellte diese Anzahl von Durchläufen die obere Grenze der von uns tolerierten Laufzeit dar und ab ca. 300 Durchläufen haben wir für keine Simulation eine weitere Veränderung der räumlichen Fehlerverteilung bemerkt.

Als Fehlermodelle kamen *nd-noise* und *ab-nlos* zum Einsatz (siehe Kapitel 5.1.2). Mittels des *nd-noise* Fehlermodells zeigen wir, wie sich der jeweilige Algorithmus bei einem normalverteilten Fehler verhält, ohne den für eine Indoorlokalisierung spezifischen vereinzelten großen Ausreißern durch NLOS-Fehler ausgesetzt zu sein. Diese Auswertungen eignen sich besonders dazu, das grundsätzliche Potenzial eines Algorithmus aufzuzeigen und den Einfluss der Geometrie der Ankerknoten auf das Ergebnis zu visualisieren. Als zweites Fehlermodell haben wir *ab-nlos* simuliert. Hier wird besonders im Vergleich zur Simulation mit *nd-noise* deutlich, wie der simulierte Algorithmus mit Ausreißern bei der Entfernungsmessung umgehen kann. Um die Visualisierungen jeweils vergleichbar zu machen, haben wir - wie in Kapitel 5.1.2 dis-

kutiert - den Erwartungswert des Eingabefehlers nicht normiert. D. h. die Kolorierung der Ergebnisse bezieht sich auf den Erwartungswert der *nd-noise* Simulation. Die mit *ab-nlos* simulierten Ergebnisse können also nur auf Basis von Absolutwerten mit den anderen Simulationen unter jeweils gleicher Ankerkonstellation verglichen werden. Ein Vergleich dieser Ergebnisse mit Auswertungen in anderen Veröffentlichungen ist für diese Simulation nicht möglich!

Die Verteilungen der einzelnen Fehler werden in Abbildung 6.1 gezeigt. Demzufolge entspricht der Erwartungswert des *nd-noise* Fehlers 5 % der Kantenlänge der Simulationsfläche und der des *ab-nlos* Fehlers etwa 10 % der Kantenlänge der Simulationsfläche.

Zu den klassischen Visualisierungen der räumlichen Fehlerverteilung haben wir für ein Szenario den inversen Modus (siehe Kapitel 5.1.2) visualisiert. Durch diese Simulation kann der Algorithmus besonders bezüglich seiner Varianz und seiner Verzerrtheit für dieses Szenario bewertet werden. Die Auswahl des Szenarios und der Position ist von uns hauptsächlich nach dem Kriterium der Aussagekraft der Ergebnisse ausgewählt worden. Damit die Ergebnisse vergleichbar bleiben, haben wir für jeden Algorithmus dasselbe Szenario gewählt - wohl wissend, dass die gewählte Position für einige Algorithmen einen besonders schwierigen Fall und für andere evtl. einen Idealfall darstellen wird. Die Performance des jeweiligen Algorithmus in diesem Szenario lässt sich nicht ohne Weiteres auf den Algorithmus im allgemeinen übertragen und in der Diskussion der Ergebnisse gehen wir auf die besonderen Spezifika der jeweiligen Algorithmen für dieses Szenario noch einmal ausdrücklich ein.

Als weitere Auswertung haben wir für eines der Szenarien ein Phasendiagramm (siehe Kapitel 5.1.2) generiert, um diskutieren zu können, ob die Fehlerverteilung eher durch die Ankergeometrie oder durch die Varianz des Eingabefehlers zu Stande kommt. Für die Vergleichbarkeit der Simulation mit *ab-nlos* Fehlermodell zu anderen Veröffentlichungen gilt analog das im zweiten Absatz geschriebene.

Die von uns gewählten vier Szenarien sind die Folgenden:

quad In diesem Szenario haben wir vier Ankerknoten mittig in einem Quadrat angeordnet. Die Kantenlänge des Quadrats beträgt 40 % der Länge der Simulationsfläche. Dieses Szenario eignet sich besonders um zu beurteilen, wie ein Algorithmus mit wenigen Ankern in Reichweite umgehen kann. Auch wird hier die räumliche Fehlerverteilung innerhalb und außerhalb der konvexen Hülle der Anker besonders deutlich. Für die Praxis der Indoorlokalisierung spielt dieses

Szenario eine Rolle für Ausbringungen, in denen die Lokalisierung auf einen Raum oder ein Gebäude beschränkt ist und die Anker an den jeweiligen Ecken angebracht sind. In diesem Falle ist insbesondere die Fehlerverteilung innerhalb der Hülle der Anker von Interesse.

Im Falle des *ab-nlos* Fehlermodells ist der Anker oben links der NLOS-Anker.

kite Dieses Szenario ist für die meisten Algorithmen ein sehr extremes Szenario. Erneut sind wieder nur vier Ankerknoten ausgebracht, diesmal allerdings in fast kollinearer Formation. Im Vergleich zum vorherigen Versuch, der ebenfalls mit vier Ankern simuliert wurde, wird die Abhängigkeit der Algorithmen von der geometrischen Konstellation der Ankerknoten besonders deutlich, da der Eingabefehler in der Verteilung der gleiche geblieben ist. Für die Indoorlokalisierung ist diese Konstellation durchaus von praktischer Relevanz, da diese Ankergeometrie in größeren Gebäuden entlang von Fluren durchaus zu erwarten ist.

Im Falle des *ab-nlos* Fehlermodells ist der Anker oben links der NLOS-Anker.

grid Dieses Szenario ist für die meisten Algorithmen das günstigste Szenario. Mit neun Ankerknoten ist eine große Redundanz vorhanden und die Positionierung erfolgt in einem quadratischen Gitter, das ebenso wie im *quad* Szenario 16 % der Simulationsfläche einnimmt. Gerade im Vergleich zum letztgenannten Szenario kann man leicht ablesen inwiefern der jeweilige Algorithmus von einem Mehr an Ankerknoten profitiert und somit die entstandene Redundanz nutzen kann. In der Praxis wird man ein ähnlich optimales Szenario wahrscheinlich nicht vorfinden.

Im Falle des *ab-nlos* Fehlermodells sind der Anker oben links, der Anker oben rechts und der Anker unten rechts die NLOS-Anker.

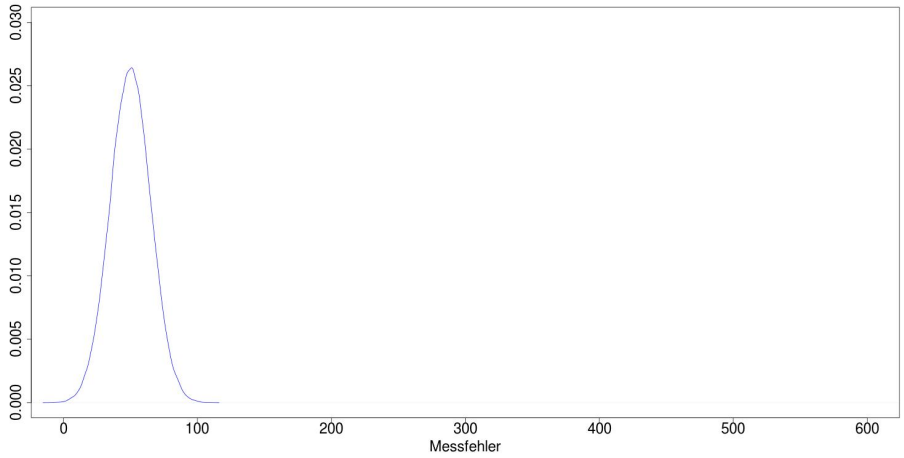
Für den RLSM Algorithmus konnte dieses Szenario nur mit sieben Ankern gerechnet werden, da dieser Algorithmus mit mehr Ankern eine derart lange Laufzeit hat, dass eine Beendigung der Simulation in einem vertretbaren Zeitrahmen nicht möglich gewesen wäre. Da die beiden Anker rechts unten weggelassen worden sind, gibt es für diesen Algorithmus im *ab-nlos* Fehlermodell demzufolge auch nur zwei NLOS-Anker.

tube In diesem Szenario haben wir ebenfalls neun Anker simuliert. Die Positionen sind jedoch weniger symmetrisch, sondern entlang eines imaginären Flures oder Ganges angeordnet. Dieses Szenario entspricht am ehesten einem üblichen Sze-

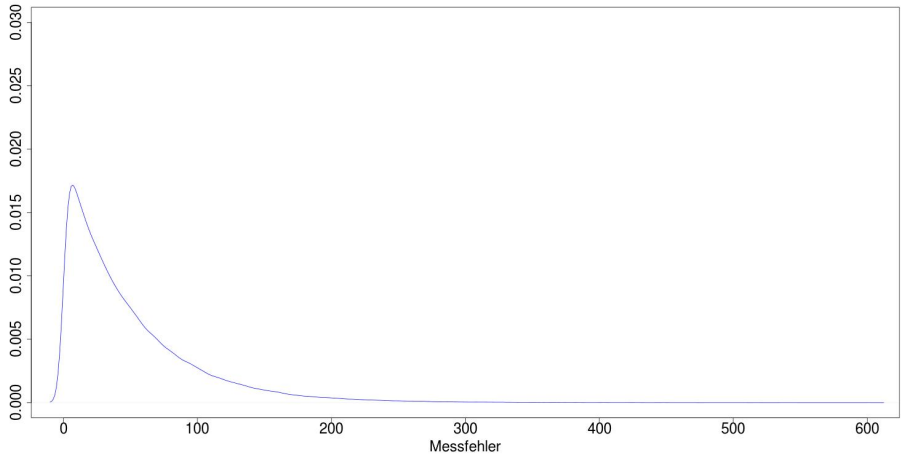
nario, wie man es in vielen Einsatzgebieten der Indoorlokalisierung finden dürfte.

Im Falle des *ab-nlos* Fehlermodells sind die ersten drei Anker der unteren Reihe die NLOS-Anker.

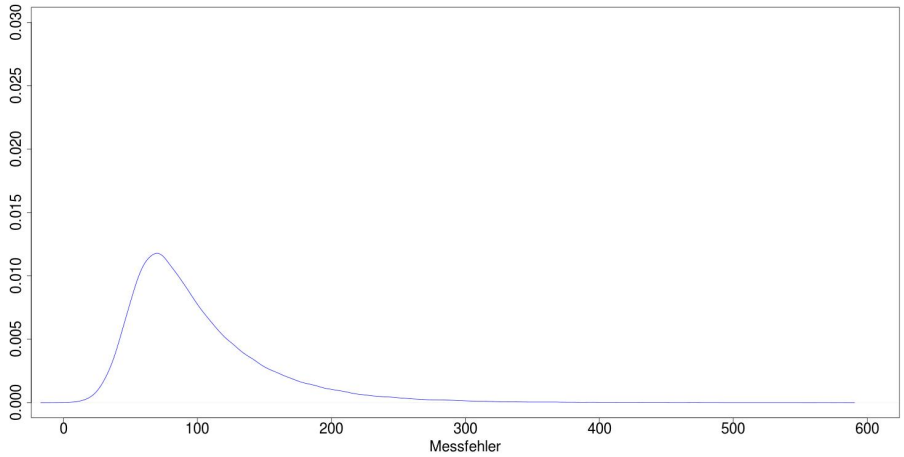
Für den RLSM Algorithmus konnte dieses Szenario nur mit sieben Ankern gerechnet werden, da dieser Algorithmus mit mehr Ankern eine derart lange Laufzeit hat, dass eine Beendigung der Simulation in einem vertretbaren Zeitrahmen nicht möglich gewesen wäre. Auch hier sind nach der Entfernung zweier Anker nur noch zwei Anker NLOS-Anker.



(a) Dichtefunktion des *nd-noise* Fehlermodells.



(b) Dichtefunktion des NLOS Anteils des *ab-nlos* Fehlermodells.



(c) Dichtefunktion des *ab-nlos* Fehlermodells.

Abbildung 6.1: Verteilung des Distanzfehlers für die einzelnen Fehlermodelle. Die Größe des Fehlers ist jeweils in Promille der Spielfeldlänge aufgetragen.

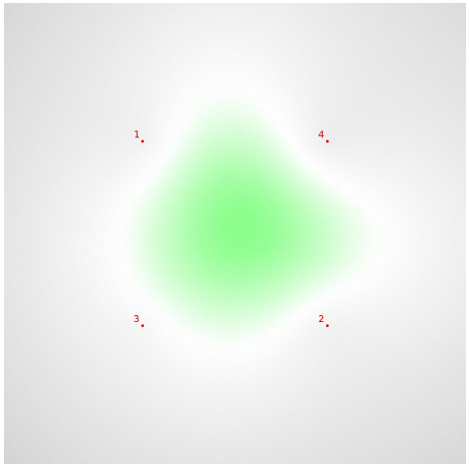
6.2 LS^2 Ergebnisse

LLS

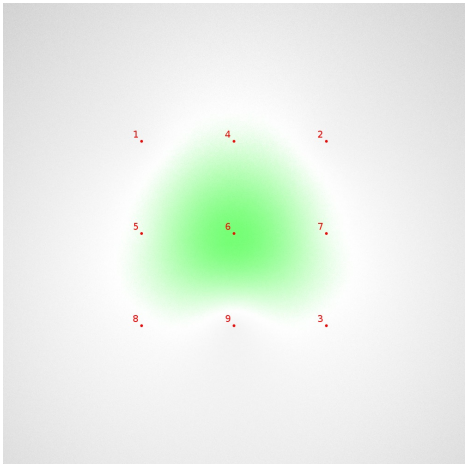
Das Lokalisierungsproblem als ein lineares Gleichungssystem zu beschreiben und mit dem LLS Algorithmus zu lösen, ist sicherlich einer der ältesten und einer der meist verwendeten Ansätze. In Abbildung 6.2 ist die räumliche Fehlerverteilung des LLS Algorithmus für das Fehlermodell *nd-noise* visualisiert. Bei der Analyse der Simulationsergebnisse aus der *kite* Konstellation ist schnell ersichtlich, dass fast in der gesamten für die Indoorlokalisierung relevanten Fläche innerhalb der Anker durchweg sehr gute Ergebnisse erzielt werden. Auch ein Abfall der Performance zu den Rändern der Simulationsfläche hin ist als moderat zu beschreiben.

In Abbildung 6.2b wurde die vorherige Konstellation um weitere fünf Anker erweitert. Da die Fehlerverteilung für alle Anker die Gleiche ist und sich die grundlegende Geometrie nicht verändert hat, kann dieser Lösungsansatz wie erwartet nicht von der gestiegenen Ankeranzahl profitieren. Dieser Lösungsansatz profitiert maximal von der symmetrischen Konstellation der Anker, da jede Position im Innenbereich für jede Ankerentfernung, die um einen bestimmten Betrag zu lang ausfällt, genau einen „Gegenspieler“ mit dem Anker gegenüber hat, der diesen Betrag durch eine Verschiebung in die Gegenrichtung wieder ausgleicht. Entfernt man sich vom Mittelpunkt der Anker, so treten immer häufiger Fälle ein, bei denen sich die Beträge nicht aufheben, sondern summieren. Dieser Effekt tritt insbesondere auf den beiden Diagonalen in großer Entfernung zum Mittelpunkt ein.

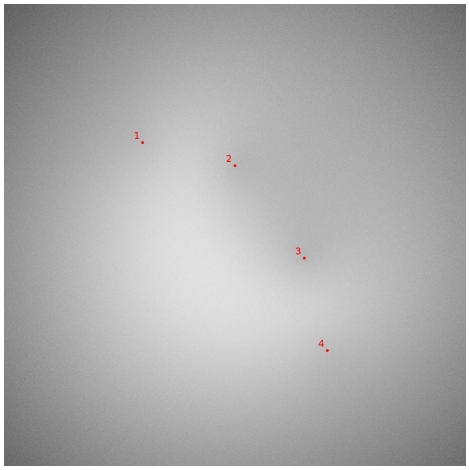
Diese absolute Symmetrie wird im *kite* Setup, welches in Abbildung 6.2c ausgewertet wird, aufgebrochen. Der Effekt ist deutlich sichtbar: Trotz gleicher Ankeranzahl und gleicher Fehlerverteilung bricht die Performance deutlich ein und der durchschnittliche Fehler steigt um fast das Doppelte an und auf der gesamten Fläche ist der Positionsfehler größer als der Fehler in den Distanzmessungen. Auch wenn sich der Verlust der Symmetrie nicht durch mehr Anker ausgleichen lässt, so lässt er sich zumindest abschwächen. In Abbildung 6.2d ist die räumliche Fehlerverteilung für die *tube* Konstellation dargestellt und auch wenn immer noch jede Position einen größeren Positionsfehler als den Eingangsfehler aufweist, so hat sich das Ergebnis doch stark verbessert. Insbesondere an Positionen, wo sich in ähnlicher Entfernung mehrere



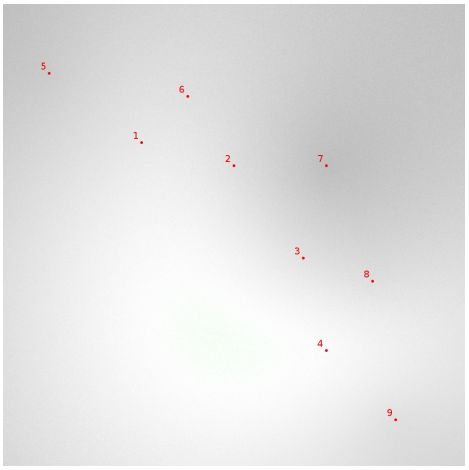
(a) Fehlerverteilung von LLS in der *quad* Simulation. Der MAE beträgt 119 %.



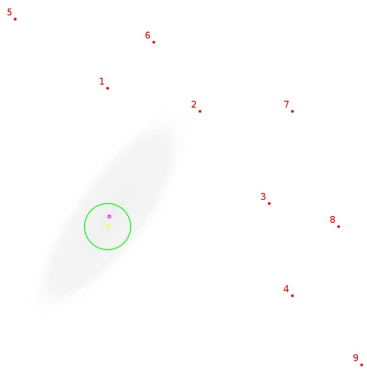
(b) Fehlerverteilung von LLS in der *grid* Simulation. Der MAE beträgt 119 %.



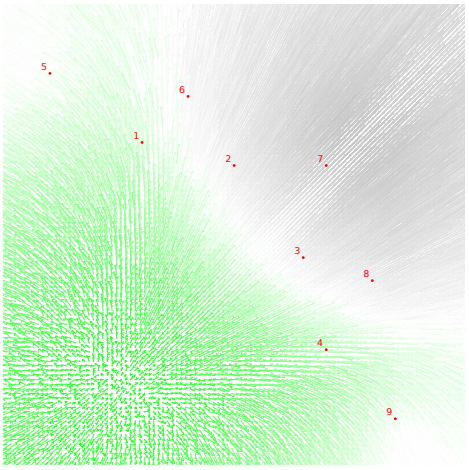
(c) Fehlerverteilung von LLS in der *kite* Simulation. Der MAE beträgt 231 %.



(d) Fehlerverteilung von LLS in der *tube* Simulation. Der MAE beträgt 145 %.

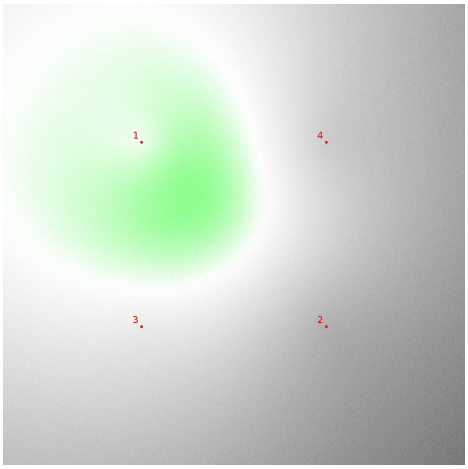


(e) Inversdarstellung von LLS in der *tube* Simulation. Der MAE beträgt 44 %.

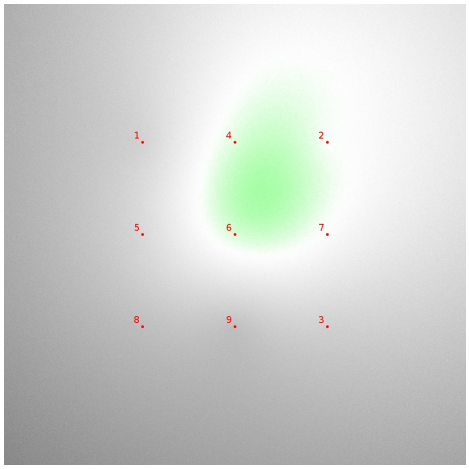


(f) Phasendiagramm von LLS in der *tube* Simulation.

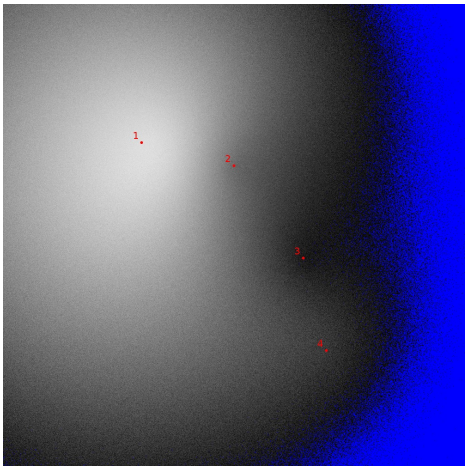
Abbildung 6.2: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den LLS Algorithmus mit nd-noise Fehlermodell.



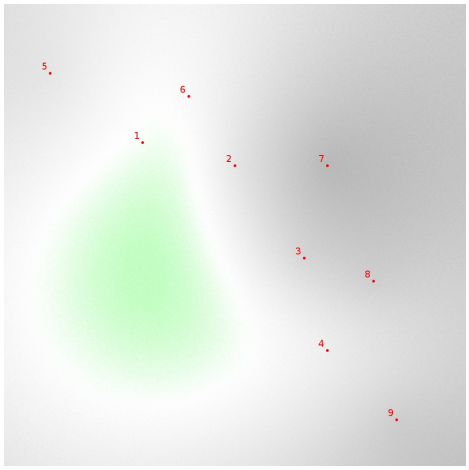
(a) Fehlerverteilung von LLS in der *quad* Simulation. Der MAE beträgt 159 %.



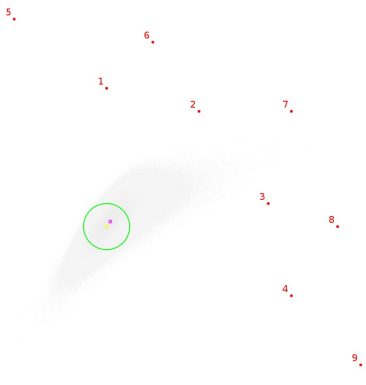
(b) Fehlerverteilung von LLS in der *grid* Simulation. Der MAE beträgt 169 %.



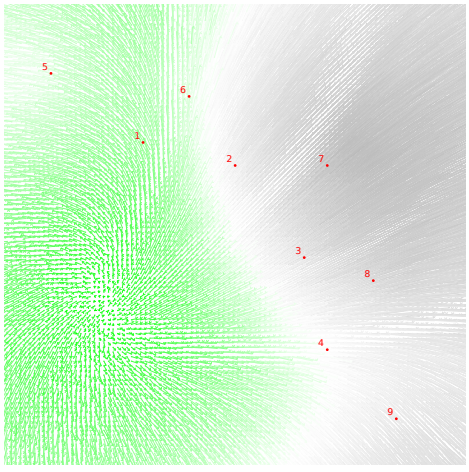
(c) Fehlerverteilung von LLS in der *kite* Simulation. Der MAE beträgt 371 %.



(d) Fehlerverteilung von LLS in der *tube* Simulation. Der MAE beträgt 135 %.



(e) Inversdarstellung von LLS in der *tube* Simulation. Der MAE beträgt 27 %.



(f) Phasendiagramm von LLS in der *tube* Simulation.

Abbildung 6.3: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den LLS Algorithmus mit ab-nlos Fehlermodell. Der Referenzbezug ist der Erwartungswert des Eingabefehlers aus der vorherigen Simulation.

Anker befinden, tritt der oben beschriebene Effekt des Fehlerausgleichs wieder ein, wenn auch dadurch abgeschwächt, dass der Winkel dieser „Gegenspieler“ nicht an die 180° aus der *quad* Konstellation herankommt.

Besonders deutlich wird dieses Phänomen in Abbildung 6.2e. Hier ist einfach zu sehen, dass die Position in der Richtung der fehlenden „Gegenspieler“ deutlich stärker streut als in der orthogonalen dazu. Auch ist sehr schön zu sehen, dass die Varianz der Schätzung ein wesentlich größeres Problem ist, als die Verzerrtheit. Wie in Abbildung 6.2f dargestellt, ist dieser Zustand aber auf die Fläche beschränkt, in der ein Ausgleich der Fehler überhaupt nur stattfinden kann. Auf der anderen Seite der Anker kann dieser Ausgleich nicht stattfinden und fast alle Fehler summieren sich auf. Da die Eingabefehler überwiegend positiv sind, kommt zur Varianz noch eine starke Verzerrtheit hinzu.

Die wichtigste Feststellung für dieses Fehlermodell bleibt jedoch, dass LLS wesentlich durch die Geometrie der Anker beeinflusst wird und bei entsprechender Ankerkonstellation die Varianz des Eingabefehlers nur eine untergeordnete Rolle spielt.

In Abbildung 6.3 ist die räumliche Fehlerverteilung des LLS Algorithmus für das Fehlermodell *ab-nlos* visualisiert. Hier zeigt sich auf den ersten Blick, dass der Algorithmus sehr stark an Performance verliert, sobald der Fehler im Erwartungswert nicht mehr gleichmäßig auf die Anker verteilt wird. Das ausgleichende Moment bleibt zwar noch etwas erhalten, wenn wie in Abbildung 6.3a visualisiert, nur ein Anker Ausreißer in der Entfernungsmessung produziert, verschiebt sich aber zu diesem Anker hin und die restliche Fläche fällt natürlich aufgrund des insgesamt größeren Fehlers stark ab. In Abbildung 6.3b wird gezeigt, dass das Einfügen von drei NLOS-Ankern die Performance weiter stark absenkt und sie sogar noch deutlich schlechter ausfällt, als mit nur vier Ankern. Allerdings ist hier anzumerken, dass die Auswahl der Ausreißer in drei von vier Ecken eindeutig den Worst Case für diesen Algorithmus darstellt. Hätte man an die vierten Ecke ebenfalls Ausreißer simuliert, wäre das Ergebnis insgesamt besser ausgefallen, trotz höheren Fehlers.

In Abbildung 6.3c kommen in der *kite* Konstellation eine ungünstige Ankerposition und NLOS-Anker zusammen und der LLS Ansatz versagt komplett. Eine wirkliche Lokalisierung findet nur noch an wenigen Positionen statt. Zum Vergleich sei hier erwähnt, dass eine bloße Schätzung jeder Position als Mitte der Simulationsfläche, einen MAE von 380 % zur Folge hätte. LLS ist hier also nur geringfügig besser als eine „Lokalisierung“ für die es keinerlei Hardware bzw. irgendwelche Technik geschweige

denn Berechnungen bedarf. Auf den ersten Blick etwas überraschend ist die Analyse des *tube* Szenarios, das in Abbildung 6.3d gezeigt wird. Im Gegensatz zur *grid* Konstellation kommt hier die Positionierung der Ausreißer dem Algorithmus sehr zu Gute. Durch die Positionierung der Ausreißer direkt nebeneinander in der Inneren der beiden Reihen, gleichen sich die Ausreißer hier zum Teil gegenseitig aus.

Auch gleichen sich aufgrund der Ankerpositionen hier gleich mehrere Anker mit „normalem“ Fehler aus, so dass das Ergebnis auf fast jeder Position besser ist als ohne die Ausreißer. In der Praxis wird es aber äußerst schwer sein, von diesem Effekt zu profitieren, da NLOS-Effekte bei der Ausbringung der Anker nur schwer abzuschätzen sind.

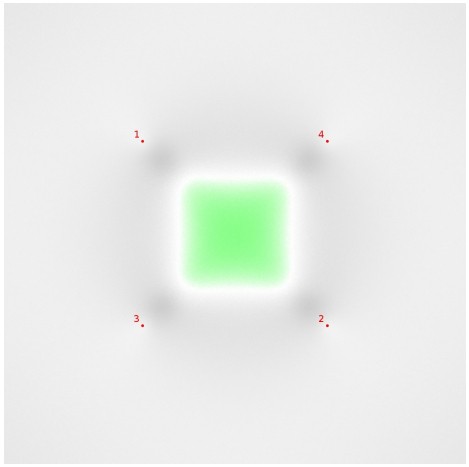
Abbildung 6.3e und 6.3f zeigen, dass im Vergleich zum rein normalverteilten Fehler die Verzerrtheit fast gleich geblieben ist und durch die spezielle Ankerkonstellation im *ab-nlos* Fehlermodell lediglich die Varianz der Schätzungen abgenommen hat. An manchen Positionen hat die Verzerrtheit sogar zugenommen, aber die Varianz in einem stärkeren Ausmaß abgenommen, so dass der resultierende Fehler sogar noch gesunken ist.

Zusammenfassend bleibt festzuhalten, dass der LLS Ansatz vor allem unter seiner starken Abhängigkeit von der Ankergeometrie leidet. Da die für eine gute Performance benötigte Symmetrie nur in kleineren Ausbringungen zu erreichen ist, ist dieser Ansatz für die Praxis nicht geeignet. Zu beachten ist allerdings die in Tabelle 6.1 dargestellte Laufzeit des Algorithmus. Diese liegt im absolut unteren Bereich aller Algorithmen und insb. für die Lokalisierung von Sensorknoten in einzelnen Räumen ist dieser Algorithmus eventuell eine mögliche Auswahl.

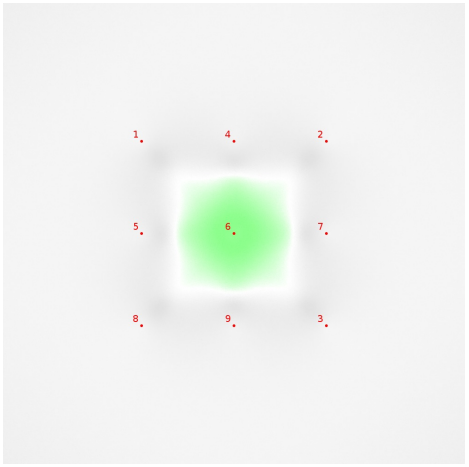
NLLS

In Abbildung 6.4 ist die räumliche Fehlerverteilung des NLLS Algorithmus für das Fehlermodell *nd-noise* visualisiert. Dieser Ansatz konstruiert aus den vorliegenden Daten ein nichtlineares Gleichungssystem und löst dieses mit dem Gauß–Newton-Verfahren.

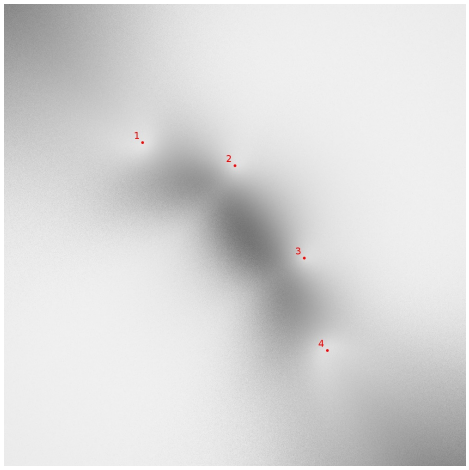
In Abbildung 6.4a ist die räumliche Fehlerverteilung des NLLS Algorithmus für die *quad* Konstellation visualisiert. Ähnlich wie auch der LLS Ansatz profitiert dieser Algorithmus stark von der Symmetrie der Ankerknoten und erzielt innerhalb der Hülle



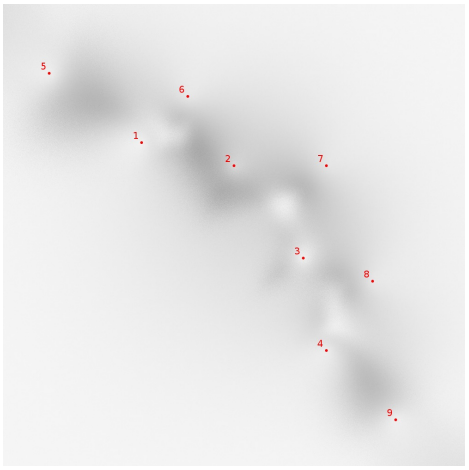
(a) Fehlerverteilung von NLLS in der *quad* Simulation. Der MAE beträgt 127 %.



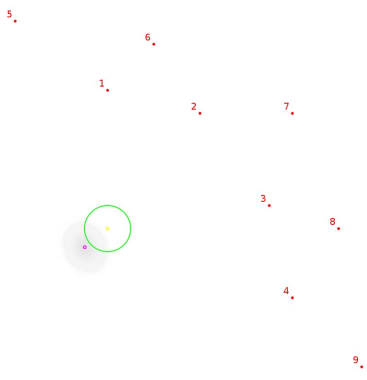
(b) Fehlerverteilung von NLLS in der *grid* Simulation. Der MAE beträgt 117 %.



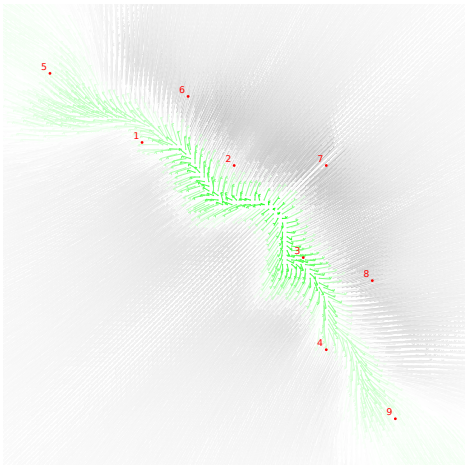
(c) Fehlerverteilung von NLLS in der *kite* Simulation. Der MAE beträgt 161 %.



(d) Fehlerverteilung von NLLS in der *tube* Simulation. Der MAE beträgt 138 %.

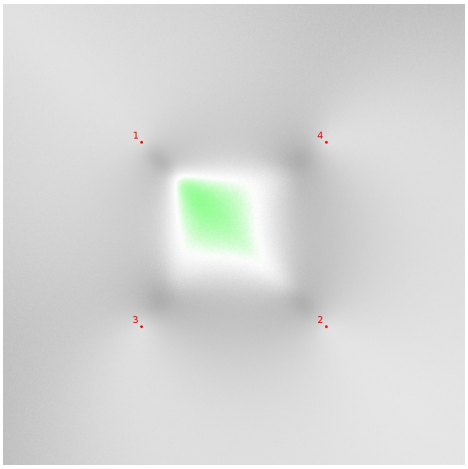


(e) Inversdarstellung von NLLS in der *tube* Simulation. Der MAE beträgt 128 %.

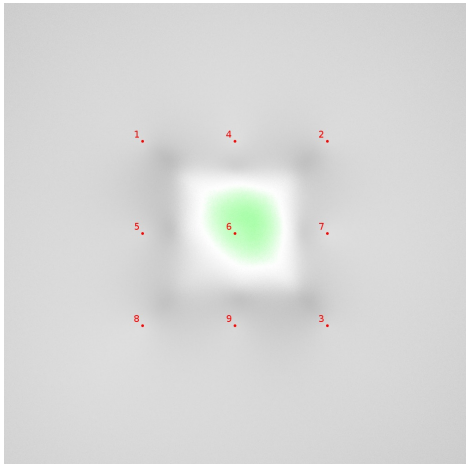


(f) Phasendiagramm von NLLS in der *tube* Simulation.

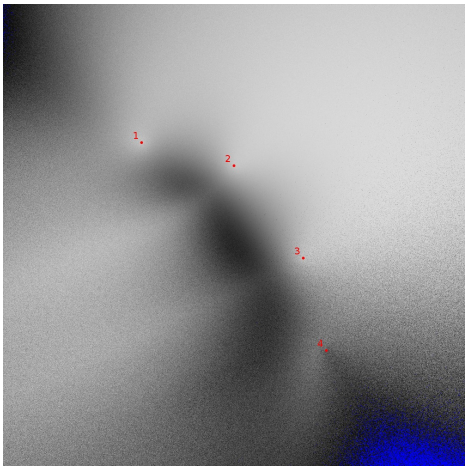
Abbildung 6.4: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den NLLS Algorithmus mit nd-noise Fehlermodell.



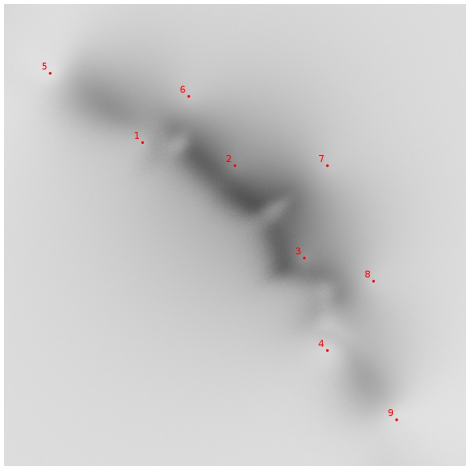
(a) Fehlerverteilung von NLLS in der *quad* Simulation. Der MAE beträgt 162 %.



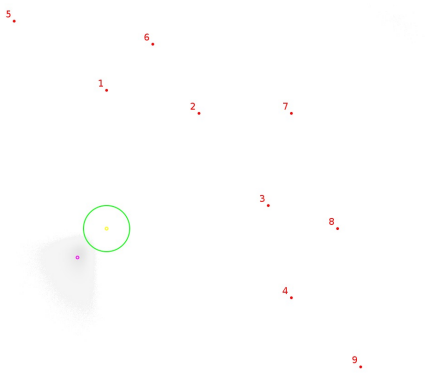
(b) Fehlerverteilung von NLLS in der *grid* Simulation. Der MAE beträgt 160 %.



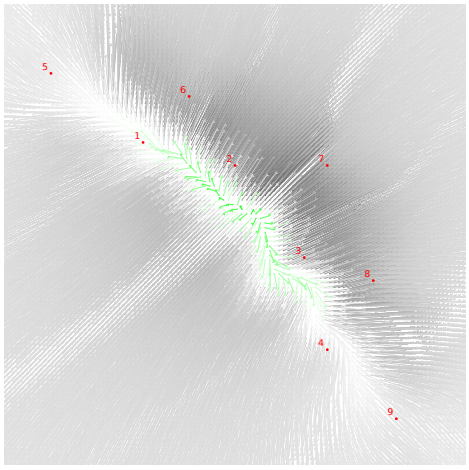
(c) Fehlerverteilung von NLLS in der *kite* Simulation. Der MAE beträgt 273 %.



(d) Fehlerverteilung von NLLS in der *tube* Simulation. Der MAE beträgt 186 %.



(e) Inversdarstellung von NLLS in der *tube* Simulation. Der MAE beträgt 178 %.



(f) Phasendiagramm von NLLS in der *tube* Simulation.

Abbildung 6.5: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den NLLS Algorithmus mit ab-nlos Fehlermodell. Der Referenzbezug ist der Erwartungswert des Eingabefehlers aus der vorherigen Simulation.

der Ankerknoten sehr gute Ergebnisse, wenn auch geringfügig schlechter als der verwandte LLS Ansatz. Da NLLS versucht, die Summe der quadrierten Residuen zu minimieren, fällt der Übergang vom grünen in den grauen Bereich besonders steil aus. Sobald der Abstand zu einem Anker kleiner als der Erwartungswert des Eingabefehlers wird, wird der Einfluss der anderen Anker zu groß, als das sich der einzelne Anker „durchsetzen“ könnte. Allerdings wirkt sich der Effekt des Aufaddierens der Fehler zweier oder mehrerer Anker außerhalb der inneren Zone nur schwach und absolut gleichmäßig aus. Dies liegt offensichtlich daran, dass der Algorithmus durch die quadratische Funktion dazu tendiert, das verbleibende Residuum gleichmäßig auf alle Anker zu verteilen, da ein einzelner Ausreißer ja quadratisch in die zu minimierende Summe eingehen würde. Das führt dazu, dass die Varianz des Positionsfehlers im Bereich des Eingabefehlers liegen sollte und der wesentliche Anteil des Positionsfehlers durch einen Ankergeometrie abhängigen Bias hinzugefügt wird.

Dementsprechend profitiert auch NLLS nur wenig von der Erhöhung der Ankerzahl um fünf weitere Anker im *grid* Szenario, dessen Auswertung in Abbildung 6.4b abgebildet ist. Die kleine Verbesserung in der inneren grünen Zone ist hauptsächlich auf eine größere Menge an Eingabefehlern zurückzuführen. Hier sollten einzelne Ausreißer besser durch die anderen Anker ausgeglichen werden.

In Abbildung 6.4c ist die räumliche Fehlerverteilung des NLLS Algorithmus für das *kite* Setup visualisiert. Ebenso wie LLS zeigt der Algorithmus hier eine sehr niedrige Performance, wenn auch nicht im selben Maße. Leider ist hier für die Indoorlokalisierung besonders störend, dass ausgerechnet die Fläche innerhalb der Hülle der Anker, die in der Praxis eine große Rolle spielt, eine sehr schlechte Performance zeigt. Dafür ist der Fehler in der restlichen Fläche besonders gleichmäßig verteilt und von mittlerer Ausprägung. Davon dürfte besonders die kooperative Lokalisierung profitieren, da hier in der Regel aufgrund der Mobilität keinerlei Annahmen über die Ankerkonstellationen getroffen werden können. Wie in Abbildung 6.4d zu sehen, profitiert NLLS relativ stark von der Erhöhung der Ankeranzahl bei einer ungünstigen Geometrie. Auch hier wird in der für die Indoorlokalisierung mit fixen Ankerknoten wichtigen Fläche innerhalb der Hülle der Anker nur eine mäßige Performance mit recht großen Schwankungen erzielt. Die wesentlich größere übrige Fläche erzielt jedoch Ergebnisse, die nur unwesentlich über dem Erwartungswert des Eingabefehlers liegen.

In Abbildung 6.4e ist die Streuung der Position des NLLS Algorithmus für das *tube* Setup visualisiert und die Ergebnisse entsprechen dem oben schon vermuteten Ver-

halten. Die Varianz der geschätzten Positionen entspricht fast exakt der Varianz des Eingabefehlers und auch der Gradient der Punktwolke lässt auf eine ähnliche Verteilung schließen. In Abbildung 6.4f wird diese Vermutung weiter untermauert und es ist ersichtlich, dass fast auf der gesamten Fläche der Positionsfehler überwiegend durch den geometrieabhängigen Bias zustande kommt. Interessanterweise ist dieser Bias ausgerechnet innerhalb der Hülle der Anker besonders niedrig. Da dort aber der Positionsfehler besonders hoch ist, lässt das auf eine hohe Varianz der Positionsschätzungen schließen. Hier gleichen sich die jeweiligen Eingabefehler auf der einen Achse wieder vollständig aus, aber auf der orthogonalen Achse addieren sich die Eingabefehler und führen zu einer hohen Varianz der Positionsschätzungen. Aus dieser Beobachtung könnten sich verbesserte Algorithmen, insbesondere für die kooperative Lokalisierung, ableiten lassen, indem je nach grober Schätzung der eigenen Position in Bezug auf die Ankerknoten, zwischen NLLS und anderen Algorithmen gewechselt wird.

In Abbildung 6.5 ist die räumliche Fehlerverteilung des NLLS Algorithmus für das Fehlermodell *ab-nlos* visualisiert. Die mathematische Funktion von NLLS lässt schon erahnen, dass der Algorithmus gerade mit einzelnen Ausreißern große Probleme haben wird. So wird um die Summe der Quadrate zu minimieren, die geschätzte Position vom Anker, der den Ausreißer produziert, weggedrückt werden, um dessen Residuum herabzusenken, obwohl drei andere dafür angehoben werden müssen. Dieses Verhalten ist sehr schön in Abbildung 6.5a zu sehen. In der hier visualisierten *quad* Konstellation treffen der oben beschriebene Effekt der inneren Zone und das Verschieben der Position in der Nähe der Anker ohne Ausreißer aufeinander und gleichen sich zum Teil aus. Außerhalb der konvexen Hülle spielt nur letzterer Effekt eine Rolle und führt zu einer gleichmäßigen Abschwächung der Performance im Vergleich zum *nd-noise* Fehlermodell. In Abbildung 6.5b ist die räumliche Fehlerverteilung für die *grid* Konstellation visualisiert. Die hier auftretenden Effekte sind dieselben wie in der vorherigen Simulation und auch hier fällt die Performance insbesondere in der Nähe von Anker #8 besonders stark ab, da der Anker die größeren Fehler der Anker an den anderen Eckpositionen nicht ausgleicht.

Mit nur vier Ankerknoten in ungünstiger Konstellation ist NLLS für die Indoorlokalisierung nicht zu empfehlen, wie in Abbildung 6.5c visualisiert ist. Auch durch das Hinzufügen weiterer fünf Anker wird die Eignung für den Indoorbereich nicht wesentlich erhöht, wie aus Abbildung 6.5d ersichtlich ist. Lediglich die sehr gleichmäßige Performance auf der gesamten Fläche außerhalb der Hülle der Anker lässt unter Um-

ständen eine Nutzung in der kooperativen Lokalisierung als möglich erscheinen. Aber auch hier ist der Positionsfehler im Durchschnitt dreimal so hoch, wie der nicht von NLOS-Effekten betroffene Eingabefehler. In der Praxis bedeutet das, dass man mit einem System zur Entfernungsbestimmung mit einem recht niedrigen durchschnittlichen Fehler von einem Meter in der Indoornutzung auf einen Positionsfehler von rund drei Metern kommen wird - und das auch nur, wenn viele Anker in Reichweite sind.

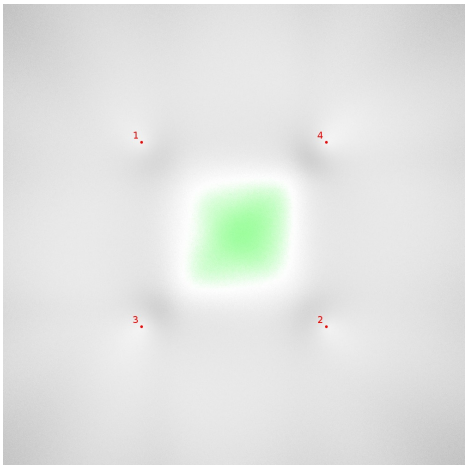
Der oben beschriebene Effekt, des „Wegdrückens“ von der Position der Ausreißer wird in Abbildung 6.5e bestätigt. Während die Varianz der Positionsschätzungen in etwa gleich bleibt, schlägt sich der zusätzliche Eingabefehler durch Ausreißer direkt in einem höheren Bias nieder, wie man im Vergleich der Abbildungen 6.5f und 6.5d sehr deutlich sehen kann.

In der Zusammenfassung der Ergebnisse hinterlässt auch NLLS einen eher negativen Eindruck. Während eine gewisse Eignung für die Lokalisierung im Freien ohne große Ausreißer im Eingabefehler besteht, so sind im Indoorbereich nur wenige Szenarien denkbar in denen sich die Performance von NLLS als befriedigend erweisen dürfte. Aufgrund des relativ gleichmäßigen Fehlerbildes bei großen Ankerzahlen könnte der Algorithmus eine gewisse Eignung für die Lokalisierung in sehr dichten, vollständig mobilen Netzen haben. Die Rechenzeit bleibt auch bei größeren Ankerzahlen in einem Bereich, der für Sensornetzhardware geeignet sein dürfte (siehe Tabelle 6.1).

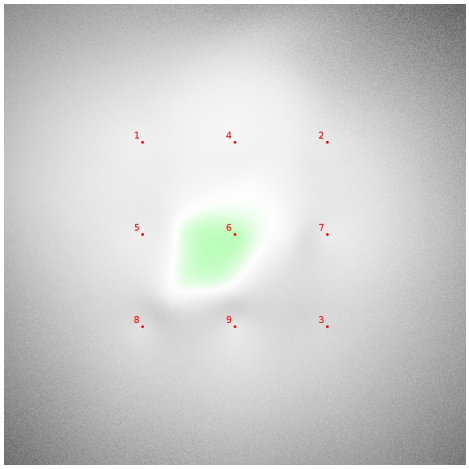
AML

In Abbildung 6.6 ist die räumliche Fehlerverteilung des AML Algorithmus für das Fehlermodell *nd-noise* visualisiert. Offensichtlich auffallend sind die nur durchwachsene Performance über alle Konstellationen und die leichte Asymmetrie, auch bei symmetrischen Ankerkonstellationen. In Abbildung 6.6a ist die räumliche Fehlerverteilung des AML Algorithmus für ein Setup aus nur vier Ankern, die an den vier Ecken der Simulationsfläche stehen, visualisiert. Der AML Algorithmus zeigt innerhalb des Quadrates aus Ankern nur in einem kleinen Bereich eine gute Performance und bleibt außerhalb dieses Quadrates auf einem durchschnittlichen Niveau. Ein starkes Abfallen der Performance in größerer Entfernung zu den Ankerknoten kann nicht beobachtet werden.

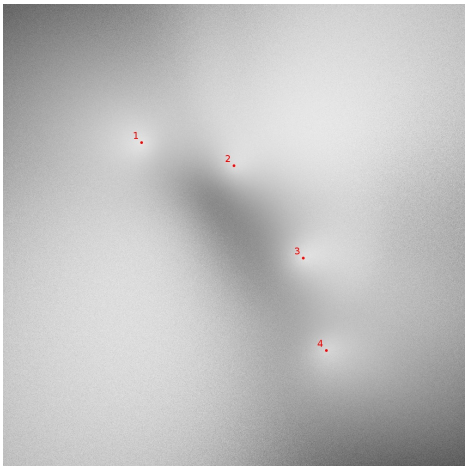
In Abbildung 6.6b ist die räumliche Fehlerverteilung des AML Algorithmus für ein



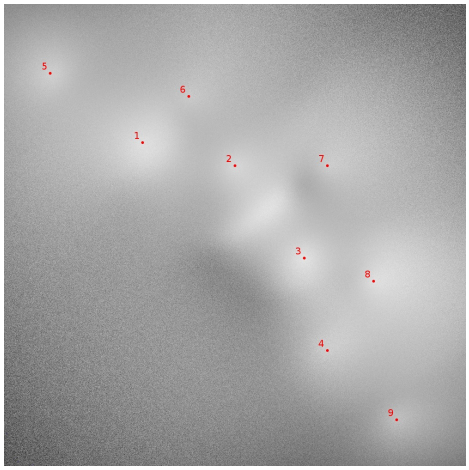
(a) Fehlerverteilung von AML in der *quad* Simulation. Der MAE beträgt 141 %.



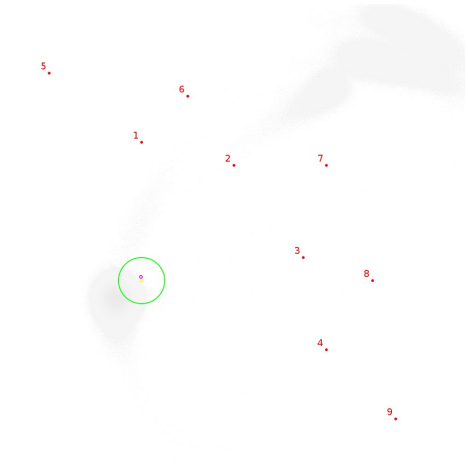
(b) Fehlerverteilung von AML in der *grid* Simulation. Der MAE beträgt 180 %.



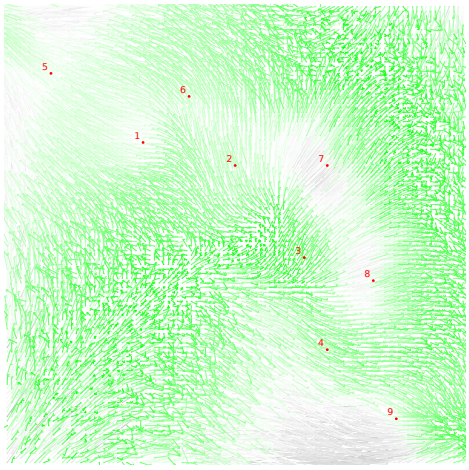
(c) Fehlerverteilung von AML in der *kite* Simulation. Der MAE beträgt 201 %.



(d) Fehlerverteilung von AML in der *tube* Simulation. Der MAE beträgt 238 %.

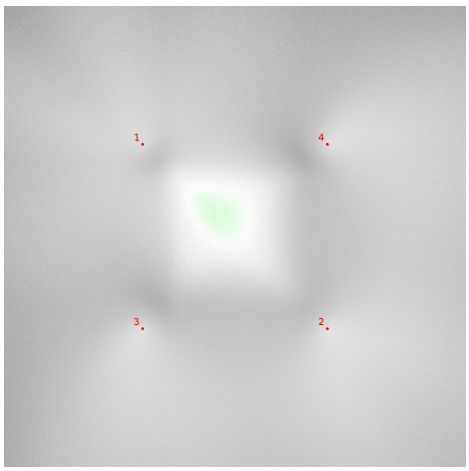


(e) Inversdarstellung von AML in der *tube* Simulation. Der MAE beträgt 16 %.

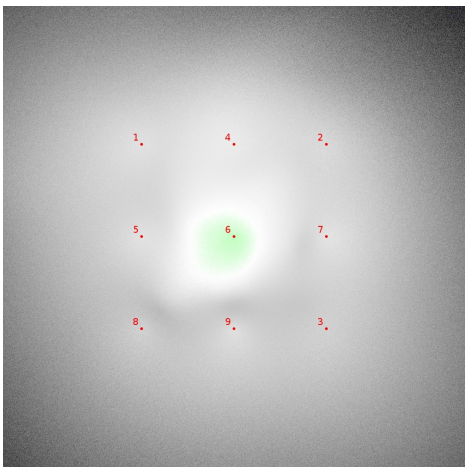


(f) Phasendiagramm von AML in der *tube* Simulation.

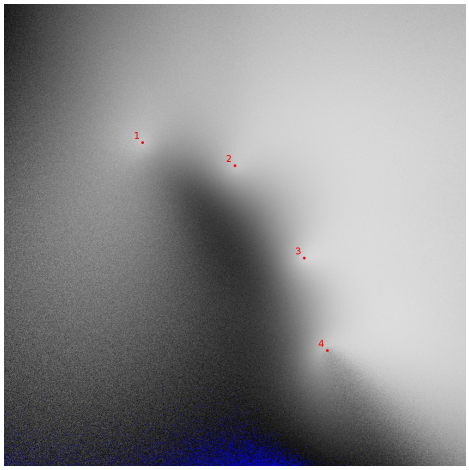
Abbildung 6.6: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den AML Algorithmus mit nd-noise Fehlermodell.



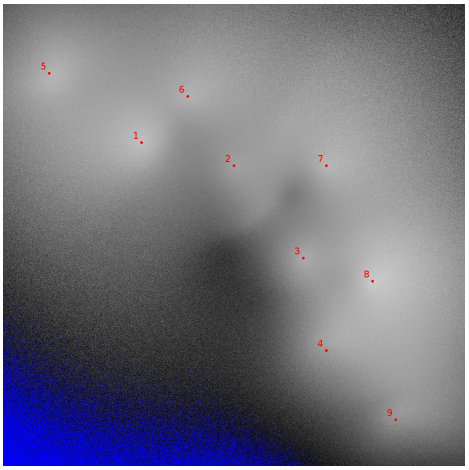
(a) Fehlerverteilung von AML in der *quad* Simulation. Der MAE beträgt 181 %.



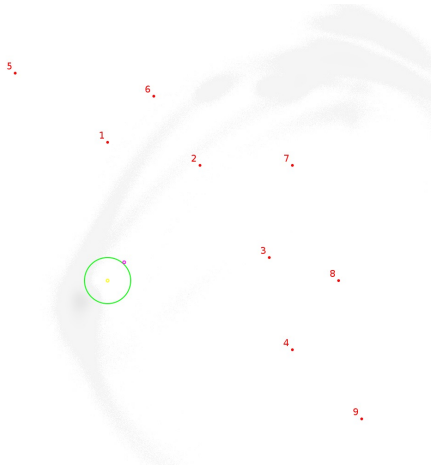
(b) Fehlerverteilung von AML in der *grid* Simulation. Der MAE beträgt 220 %.



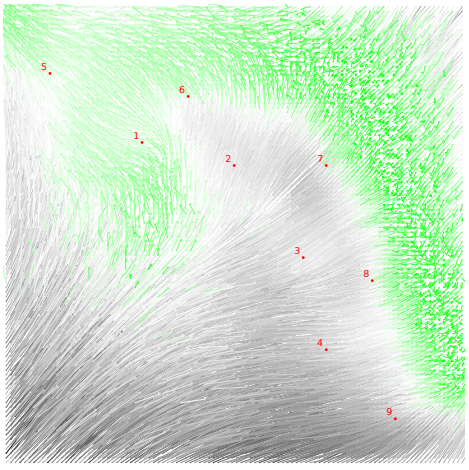
(c) Fehlerverteilung von AML in der *kite* Simulation. Der MAE beträgt 284 %.



(d) Fehlerverteilung von AML in der *tube* Simulation. Der MAE beträgt 326 %.



(e) Inversdarstellung von AML in der *tube* Simulation. Der MAE beträgt 107 %.



(f) Phasendiagramm von AML in der *tube* Simulation.

Abbildung 6.7: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den AML Algorithmus mit ab-nlos Fehlermodell. Der Referenzbezug ist der Erwartungswert des Eingabefehlers aus der vorherigen Simulation.

Setup aus neun Ankern in gleichmäßiger Verteilung über die Simulationsfläche visualisiert. Neben der schon erwähnten Asymmetrie ist vor allem auffallend, dass der Algorithmus nicht nur gar nicht von der durch immerhin fünf zusätzliche Anker erzeugten Redundanz profitiert, sondern, ganz im Gegenteil, seine Performance noch verschlechtert. Nicht nur im inneren der Anker, sondern gerade am Rand der Simulationsfläche werden die Ergebnisse nun deutlich schlechter. In der *grid* Konstellation ist AML einer der am schlechtesten performenden Algorithmen.

In Abbildung 6.6c ist die räumliche Fehlerverteilung des AML Algorithmus für ein Setup aus vier Ankern, die in einer denkbar ungünstigen, fast kollinearen Anordnung auf der Simulationsfläche stehen, visualisiert. Ausgerechnet im für die Indoorlokalisierung wichtigen Bereich innerhalb der konvexen Hülle der Anker zeigt AML hier deutliche Schwächen, dafür ist das Ergebnis im direkten Nahfeld um die Anker sehr gut.

Das Ergebnis der in Abbildung 6.6d visualisierten *tube* Konstellation überrascht erneut. Wieder verschlechtert sich das Ergebnis auf fast der gesamten Simulationsfläche zum Teil stark. Nur innerhalb der konvexen Hülle und in großer Nähe zu den Ankerknoten kann AML die Ergebnisse des vorherigen Experiments halten oder minimal ausbauen. Der für die verteilte Lokalisierung wichtige sehr gute Bereich kommt bei den letzten beiden Experimente allerdings überhaupt nicht mehr vor.

In Abbildung 6.6e ist die Streuung der Position des AML Algorithmus für das *tube* Setup visualisiert. In dieser Visualisierung kann man sehr leicht das Hauptproblem von AML erkennen. Während die durchschnittlich geschätzte Position fast genau der gesuchten Position entspricht, ist die Varianz der Schätzungen immens und bewegt sich über die volle Breite der Simulationsfläche, obwohl neun Ankerknoten mit nur einem geringen normalverteilten Fehler zur Verfügung stehen. Die große Erwartungstreue deutet allerdings daraufhin, dass AML nur wenig durch die geometrische Konstellation der Anker zu beeinflussen ist, sondern hauptsächlich durch die Varianz des Eingabefehlers zu beeinflussen ist. Diese Vermutung wird durch die Analyse des Phasendiagramms mit derselben Ankerkonstellation bestätigt. In Abbildung 6.6f zeigt sich, dass fast über die gesamte Simulationsfläche nur eine äußerst geringe Verzerrtheit festzustellen ist und AML fast durchgängig als erwartungstreu bezeichnet werden kann.

Für die Indoorlokalisierung bedeutender sind jedoch die Ergebnisse der Analyse für das *ab-nlos* Fehlermodell, die in Abbildung 6.7 gezeigt werden. Hier bestätigen sich

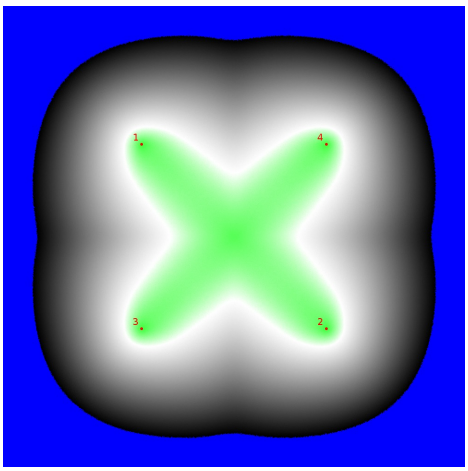
aber im Wesentlichen die Ergebnisse der Analyse des *nd-noise* Fehlermodells. In Abbildung 6.7a zeigt sich, dass das Hinzufügen eines NLOS-Fehlers auf die zu Anker #1 gemessenen Entfernungen die Ergebnisse gleichmäßig verschlechtert. Auch für die *grid* Konstellation tritt eine gleichmäßige Verschlechterung der Ergebnisse ein und AML performt erneut schlechter, wenn die Redundanz durch das Hinzufügen weiterer Anker erhöht wird. Das gleiche Verhalten ist auch für die *kite* Konstellation in Abbildung 6.7c und für die *tube* Konstellation in Abbildung 6.7d zu beobachten.

Für die inverse Darstellung in Abbildung 6.7e verschlechtert sich das Ergebnis nur geringfügig. Erneut zeigt sich, dass AML im Erwartungswert sehr Nahe an der gesuchten Position liegt, aber die Varianz wieder sehr hoch ist - die eigentlich Position also nur sehr selten getroffen wird. Wie Abbildung 6.7f zeigt, kann dieses Verhalten jedoch nun nicht mehr für alle Positionen der Simulationsfläche gehalten werden und für große Teile der Fläche kommt nun noch eine recht starke Verzerrtheit der Ergebnisse hinzu. Der größere Anteil des Fehlers entsteht aber auch in diesen Bereichen nach wie vor durch die hohe Varianz der Schätzungen, wie der Vergleich zwischen Abbildung 6.7f und Abbildung 6.7d zeigt.

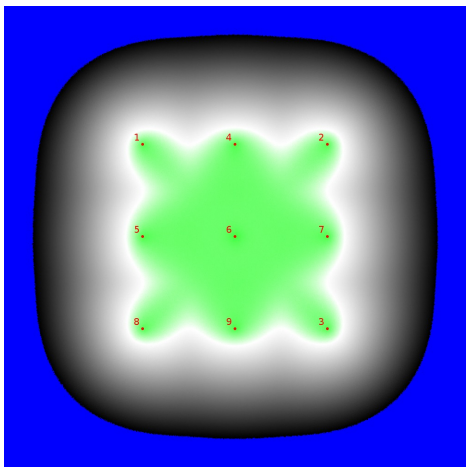
Abschließend kann zusammengefasst werden, dass der AML Algorithmus für die Indoorlokalisierung von bewegten Objekten unter üblichen Ankerkonstellationen nur bedingt geeignet scheint. Wesentlicher Nachteil sind hier die schlechte Performance bei steigender Ankerzahl. Aufgrund der strukturellen Eigenschaften der meisten Gebäude ist die vom unbekannten Knoten tatsächlich erreichte Zahl an Ankern in der Regel stark schwankend. Lediglich für sich nicht bewegende Objekte sind von AML aufgrund der hohen Erwartungstreue gute Ergebnisse zu erwarten, die durch das wiederholte Lokalisieren des stehenden Objekts und anschließende Mittelwertbildung erzielt werden können.

Min-Max

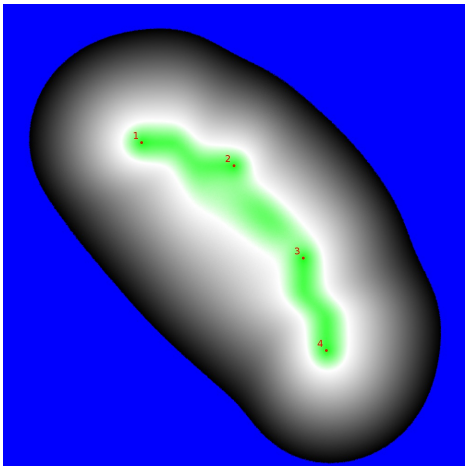
In Abbildung 6.8 ist die räumliche Fehlerverteilung für den Min-Max Algorithmus und das *nd-noise* Fehlermodell visualisiert. Min-Max ist einer der am einfachsten zu berechnenden Algorithmen, der auf den ersten Blick jedoch einen offensichtlichen Nachteil zeigt: Wie in Abbildung 6.8a für das *quad* Szenario zu sehen ist, kann der Algorithmus nur Positionen schätzen, die innerhalb der konvexen Hülle der Ankerkno-



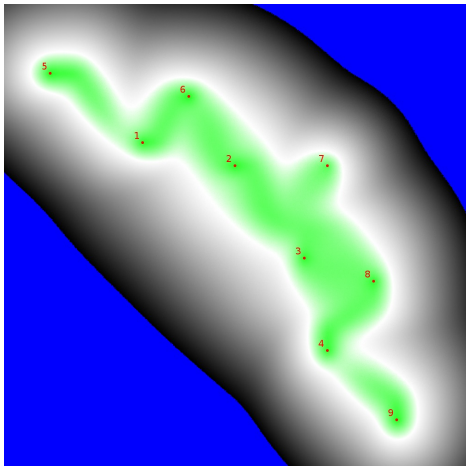
(a) Fehlerverteilung von Min-Max in der *quad* Simulation. Der MAE beträgt 365 %.



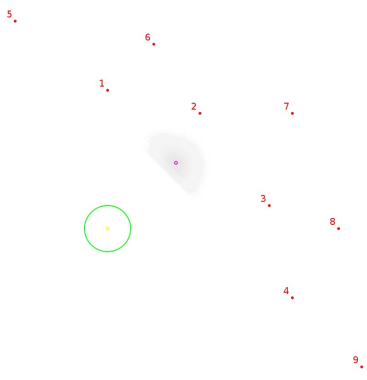
(b) Fehlerverteilung von Min-Max in der *grid* Simulation. Der MAE beträgt 361 %.



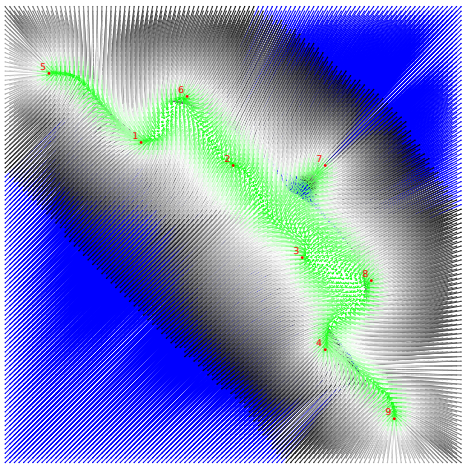
(c) Fehlerverteilung von Min-Max in der *kite* Simulation. Der MAE beträgt 498 %.



(d) Fehlerverteilung von Min-Max in der *tube* Simulation. Der MAE beträgt 368 %.

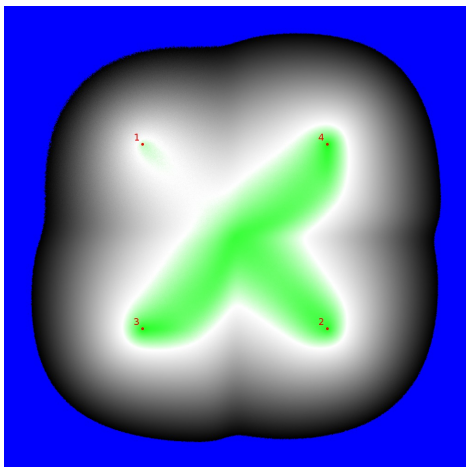


(e) Inversdarstellung von Min-Max in der *tube* Simulation. Der MAE beträgt 411 %.

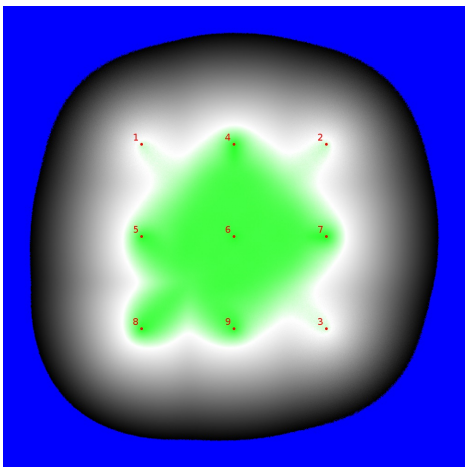


(f) Phasendiagramm von Min-Max in der *tube* Simulation.

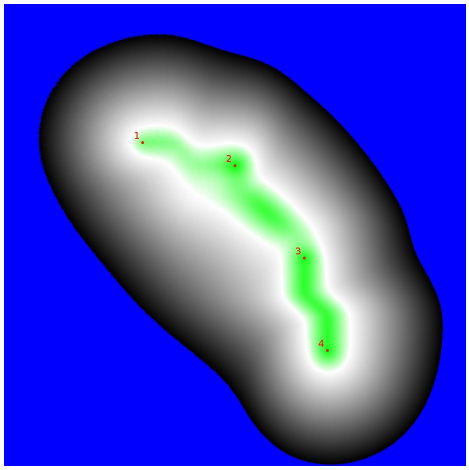
Abbildung 6.8: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den Min-Max Algorithmus mit nd-noise Fehlermodell.



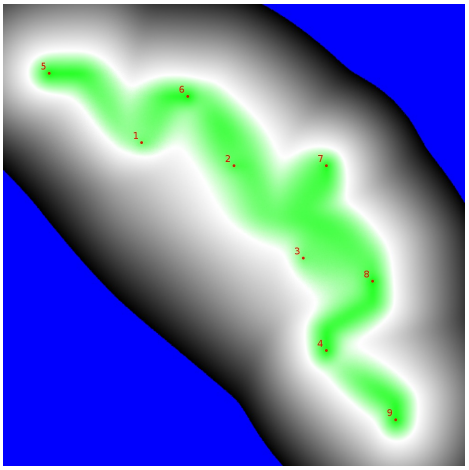
(a) Fehlerverteilung von Min-Max in der *quad* Simulation. Der MAE beträgt 369 %.



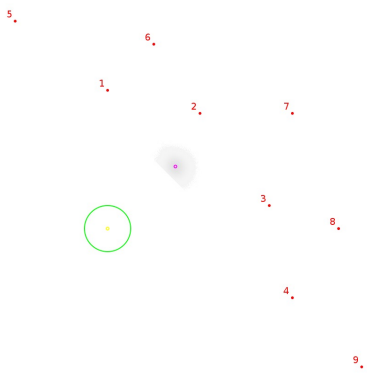
(b) Fehlerverteilung von Min-Max in der *grid* Simulation. Der MAE beträgt 373 %.



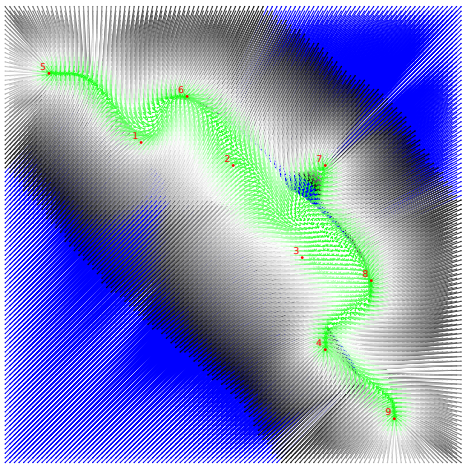
(c) Fehlerverteilung von Min-Max in der *kite* Simulation. Der MAE beträgt 504 %.



(d) Fehlerverteilung von Min-Max in der *tube* Simulation. Der MAE beträgt 360 %.



(e) Inversdarstellung von Min-Max in der *tube* Simulation. Der MAE beträgt 398 %.



(f) Phasendiagramm von Min-Max in der *tube* Simulation.

Abbildung 6.9: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den Min-Max Algorithmus mit ab-nlos Fehlermodell. Der Referenzbezug ist der Erwartungswert des Eingabefehlers aus der vorherigen Simulation.

ten liegen. Somit fällt die Performance außerhalb der Hülle linear ab, da im optimalen Fall die der gesuchten Position am nächsten auf der Hülle gelegene Position geschätzt werden kann. Dafür ist die geschätzte Position innerhalb der konvexen Hülle außerordentlich genau, sofern der Abstand zwischen den Ankerknoten nicht zu groß wird. Vergrößert man den Abstand, fällt der sehr gute Bereich nur noch auf die Diagonalen zwischen den Ankern. In der *quad* Konstellation zeigt der Min-Max Algorithmus im Bereich innerhalb der Ankerknoten die beste Performance der bisher besprochenen Algorithmen, obwohl der MAE der gesamten Simulation auf keine gute Performance schließen lässt.

Wie in Abbildung 6.8b gezeigt, profitiert der Algorithmus in der *grid* Simulation im Bereich zwischen den Ankern recht stark von den zusätzlichen Ankern und kann nun in der gesamten Fläche sehr gute Ergebnisse erzielen. Das die Performance außerhalb dieser Fläche noch schlechter als im vorherigen Fall ist, sei nur der Vollständigkeit halber erwähnt - in der Praxis ist der Algorithmus außerhalb der Hülle der Anker in keinem dieser Szenarien zu gebrauchen.

Ein vergleichbares Bild zeigt sich in Abbildung 6.8c für das *kite* Szenario. Auch hier werden innerhalb der nur sehr kleinen konvexen Hülle sehr gute Ergebnisse erzielt, während die Ergebnisse außerhalb stark abfallen. Das gleiche Bild zeigt sich in Abbildung 6.8d: Hier profitiert die Performance hauptsächlich von der Vergrößerung der konvexen Hülle durch die zusätzliche Platzierung von Ankerknoten.

In Abbildung 6.8e ist die Streuung der Position des Min-Max Algorithmus für das *tube* Setup visualisiert. Hier zeigt sich das Verhalten von Min-Max nochmals sehr deutlich: Die konvexe Hülle ist eine harte Schranke, die nicht überschritten werden kann. Die Varianz der Schätzungen entspricht hingegen in etwa der Varianz des Eingangsfehlers. Über weitere Fehlerquellen gibt auch das folgende Phasendiagramm in Abbildung 6.8f nur eine vage Auskunft, da der große Bias, der durch die besprochene Eigenheit des Algorithmus entsteht, eventuelle Schwankungen in der Varianz überlagert.

Fügt man den eben diskutierten Bedingungen mittels des *ab-nlos* Fehlermodells Ausreißer hinzu, so entstehen die in Abbildung 6.9 visualisierten Ergebnisse. Besonders beim *quad* Setup, welches in Abbildung 6.9a visualisiert ist, zeigen sich die Auswirkungen von Ausreißern sehr deutlich. Während in der direkten Umgebung des NLOS-Ankerknotens die Performance deutlich einbricht, ist dies in den restlichen Bereichen nicht der Fall. In Abbildung 6.9b kann man für das *tube* Setup die NLOS-Anker, im Vergleich mit der Simulation ohne Ausreißer, sofort erkennen. Bei genauerem Hinse-

hen fällt sogar auf, dass sich die Werte in den restlichen sehr guten Bereichen durch den Ausreißer sogar verbessert haben. Dieses Verhalten erklärt sich damit, dass beispielsweise im *quad* Setup für jede Kante der *Bounding-Box* in der Regel nur zwei Ankerknoten relevant sind und jeweils die Kante mit dem niedrigeren Fehler relevant wird. Innerhalb der konvexen Hülle sind aufgrund dieser Feststellung also nur Messungen ein „Problem“, die niedriger als der Erwartungswert ausfallen - längere Messungen fallen ja nur dann ins Gewicht, wenn keine einen größeren Fehler hat. Da die Wahrscheinlichkeit bei zwei Ziehungen größer ist, als bei nur einer Ziehung, werden die Werte, die bisher durch „zu präzise“ Messungen durch den Centroid vom eigentlichen Punkt weggedrückt wurden, nun weniger häufig auftreten. Diese Effekte werden immer auftreten, wenn mindestens drei Anker in etwa gleicher Entfernung zum gesuchten Punkt liegen - ein Szenario das innerhalb der konvexen Hülle von dicht stehenden Ankerknoten recht häufig vorkommen wird.

In der *kite* Simulation wird dieser Effekt ebenfalls deutlich: In Abbildung 6.9c ist zu sehen, dass sich das Ergebnis gegenüber der Simulation ohne Ausreißer in den Entfernungsmessungen kaum verändert hat. Durch die fehlende Symmetrie kann kaum vorhergesagt werden, welcher Anker einen potenziell größeren Einfluss auf welche Kante der *Bounding-Box* haben wird und auch „zu präzise“ Messungen sind hier nur an wenigen Positionen ein Problem. So wird auch hier der Ausreißer in so gut wie keine Schätzung einfließen und die für eine Normalverteilung mit niedriger Varianz sehr gut funktionierende *Bounding-Box* Konstruktion liefert auch mit nur drei Ankern fast dieselben Ergebnisse.

Dies gilt ebenfalls für die in Abbildung 6.9d visualisierte *tube* Simulation. Hier zeigt sich an vielen Orten sogar wieder der Symmetrie-Effekt, so dass sich das Ergebnis innerhalb der konvexen Hülle trotz drei Ausreißern sogar noch verbessert. In Abbildung 6.9e zeigt sich dieser Symmetrie-Effekt noch einmal sehr eindeutig. Auch wenn die letztendlich getroffene Schätzung weit von der gesuchten Position entfernt ist, so sind die geschätzten Positionen genau in einer solchen Fläche, in der eine relative Symmetrie herrscht und somit ist die Varianz ein gutes Stück kleiner als in der Simulation desselben Szenarios mit *nd-noise* Fehler.

Da diese Verbesserungseffekte aber nur partiell durch eine kleinere Varianz auftreten, ist das in Abbildung 6.9f gezeigte Phasendiagramm so gut wie unverändert.

Auch wenn Min-Max ein extrem simples und auch sehr naives Verfahren ist und die simulierten MAE Werte am absolut unteren Ende der Skala liegen, so ergibt die

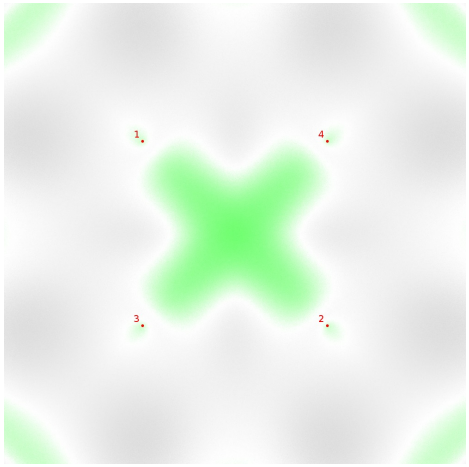
Analyse der räumlichen Fehlerverteilung, dass dieser Algorithmus für eine Vielzahl an Szenarien der Indoorlokalisierung sehr gut geeignet ist. Gerade wenn die Ankerknoten an Wänden montiert werden, befinden sich die zu lokalisierenden Knoten so gut wie immer innerhalb der konvexen Hülle. Das dieser Bereich in so gut wie jeder Konstellation auch sehr robust gegenüber Ausreißern ist, ist ein weiterer Pluspunkt. Von den oben diskutierten Symmetrie-Effekten kann man jedoch nicht gezielt profitieren. Sie weisen lediglich darauf hin, dass es in bestimmten Konstellationen besser sein kann, weniger als mehr Anker zu nutzen - die Voraussetzung für eine Verbesserung ist dann allerdings, dass keiner der verbleibenden Anker Ausreißer produziert!

Obwohl Min-Max auf dem, in vielen Bereichen anwendbaren, *Bounding-Box* Verfahren beruht, welches nicht einmal speziell für die Indoorlokalisierung entworfen wurde, so performen verschiedene Algorithmen, die für die Indoorlokalisierung entworfen wurden, zum Teil wesentlich schlechter als dieser Algorithmus.

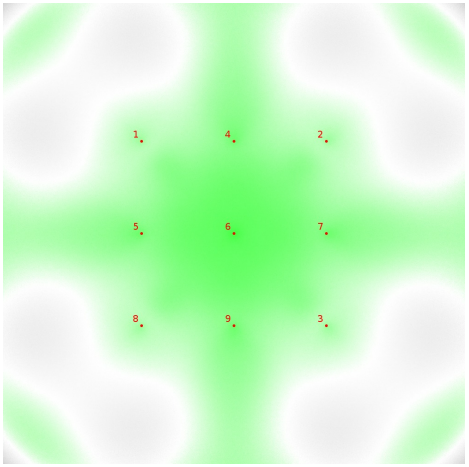
E-Min-Max (W2)

Der E-Min-Max (W2) Algorithmus wurde entwickelt, um unter Beibehaltung des Prinzips und der Stärken von Min-Max, dessen Schwächen zu minimieren. So sollten insbesondere Bereiche außerhalb der konvexen Hülle in der Performance deutlich verbessert werden. In Abbildung 6.10 ist die räumliche Fehlerverteilung des E-Min-Max (W2) Algorithmus für das Fehlermodell *nd-noise* visualisiert und die Verbesserungen gegenüber Min-Max sind deutlich in allen Grafiken zu sehen.

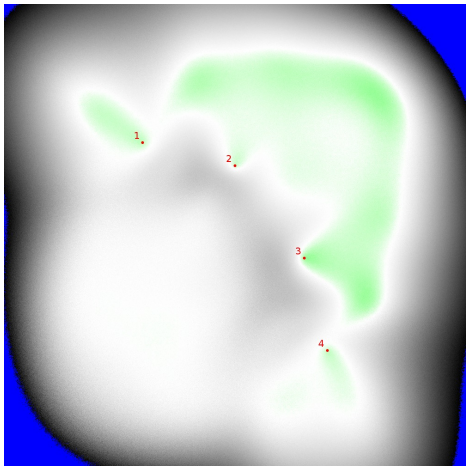
In Abbildung 6.10a ist die räumliche Fehlerverteilung des E-Min-Max (W2) Algorithmus für das *quad* Szenario visualisiert und die Verbesserung der Performance erstrecken sich fast über die gesamte Simulationsfläche. Die Performance innerhalb der konvexen Hülle ist in etwa gleich geblieben, wenn auch leichte Abfälle in direkter Umgebung der Anker zu sehen sind. Dafür ist der sehr gute Bereich innerhalb der Hülle jetzt etwas größer. Auch wenn die Bereiche außerhalb der Hülle für viele Szenarien der Indoorlokalisierung nicht von großer Bedeutung sind, sind hier dennoch sehr große Verbesserungen zu sehen und die Performance verteilt sich sehr homogen über die gesamte Fläche. Eine weitere wesentliche Verbesserung ist in Abbildung 6.10b für das *grid* Szenario zu beobachten: Im Gegensatz zum klassischen Min-Max kann dieser Algorithmus von der erhöhten Ankeranzahl in der gesamten Fläche profitieren.



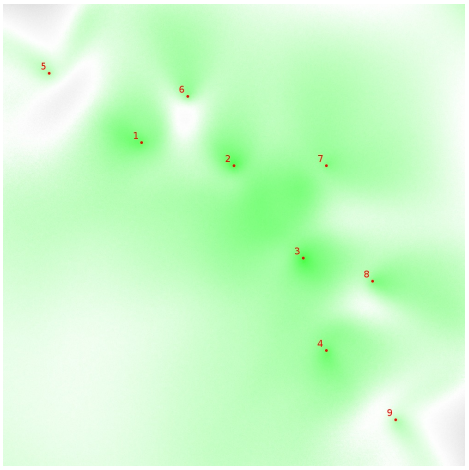
(a) Fehlerverteilung von E-Min-Max (W2) in der *quad* Simulation. Der MAE beträgt 115 %.



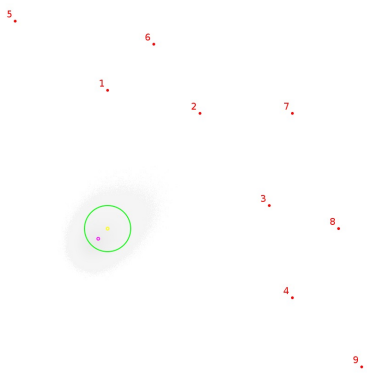
(b) Fehlerverteilung von E-Min-Max (W2) in der *grid* Simulation. Der MAE beträgt 86 %.



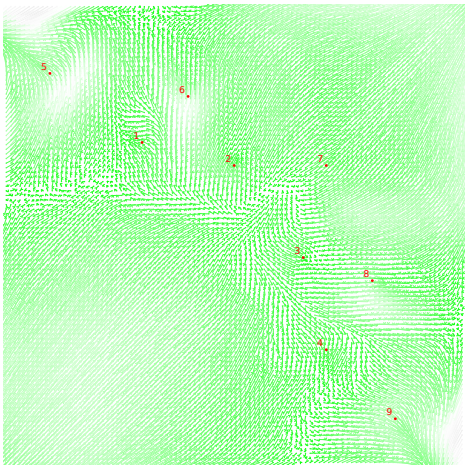
(c) Fehlerverteilung von E-Min-Max (W2) in der *kite* Simulation. Der MAE beträgt 195 %.



(d) Fehlerverteilung von E-Min-Max (W2) in der *tube* Simulation. Der MAE beträgt 80 %.

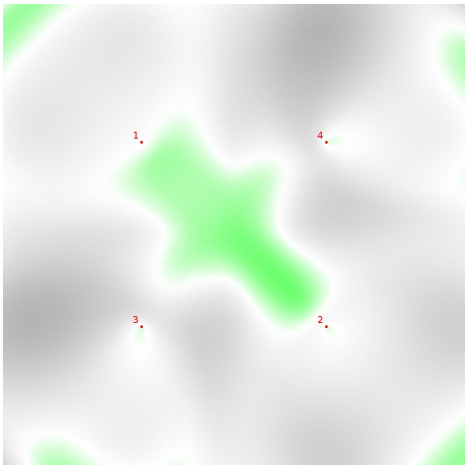


(e) Inversdarstellung von E-Min-Max (W2) in der *tube* Simulation. Der MAE beträgt 60 %.

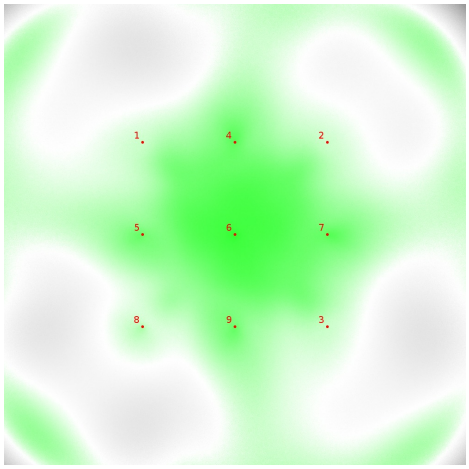


(f) Phasendiagramm von E-Min-Max (W2) in der *tube* Simulation.

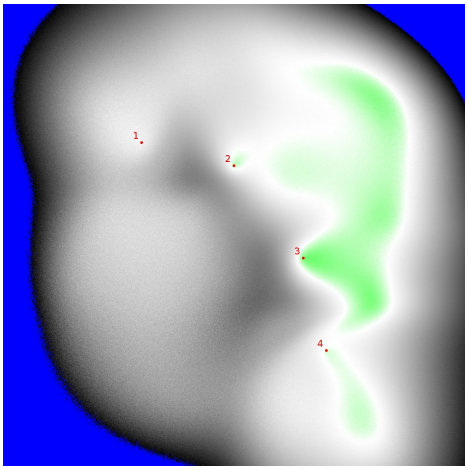
Abbildung 6.10: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den E-Min-Max (W2) Algorithmus mit nd-noise Fehlermodell.



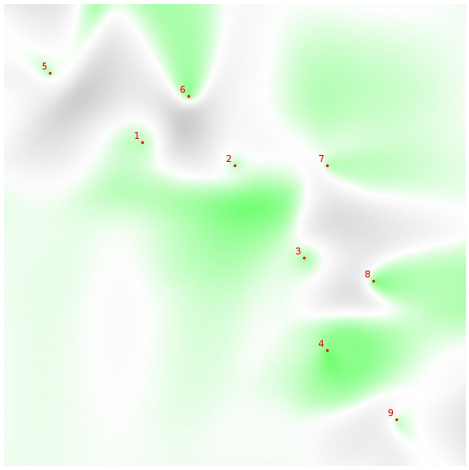
(a) Fehlerverteilung von E-Min-Max (W2) in der *quad* Simulation. Der MAE beträgt 129 %.



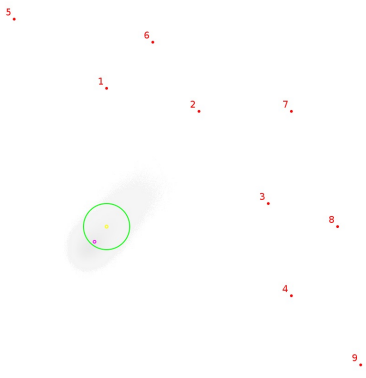
(b) Fehlerverteilung von E-Min-Max (W2) in der *grid* Simulation. Der MAE beträgt 88 %.



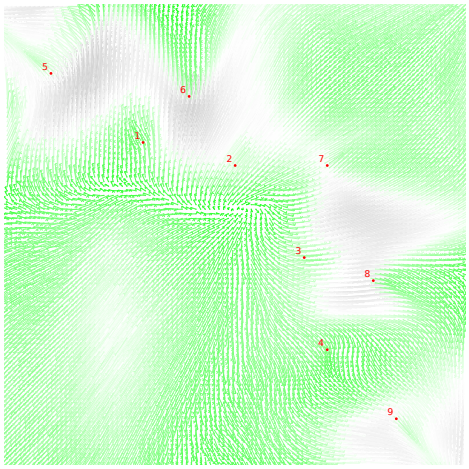
(c) Fehlerverteilung von E-Min-Max (W2) in der *kite* Simulation. Der MAE beträgt 271 %.



(d) Fehlerverteilung von E-Min-Max (W2) in der *tube* Simulation. Der MAE beträgt 96 %.



(e) Inversdarstellung von E-Min-Max (W2) in der *tube* Simulation. Der MAE beträgt 84 %.



(f) Phasendiagramm von E-Min-Max (W2) in der *tube* Simulation.

Abbildung 6.11: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den E-Min-Max (W2) Algorithmus mit ab-nlos Fehlermodell. Der Referenzbezug ist der Erwartungswert des Eingabefehlers aus der vorherigen Simulation.

Ein gewisser Rückschritt für die Indoorlokalisierung zeigt sich in Abbildung 6.10c im *kite* Szenario: Hier konnte zwar das durchschnittliche Ergebnis sehr stark verbessert werden, aber ausgerechnet innerhalb der für die Indoorlokalisierung so wichtigen Hülle der Anker zeigen sich nun deutliche Schwächen. Die starken Bereiche verteilen sich zu inhomogen, als dass bewusst von diesen profitiert werden könnte. Dieses Szenario stellt mit seinen wenigen Ankern allerdings ein Extremszenario dar, das in der Praxis höchstens partiell in einer größeren Installation vorkommen dürfte und der Einfluss dieses Verhaltens auf die Gesamtperformance somit beschränkt bleiben dürfte.

Einen wesentlichen Einfluss auf die Gesamtperformance dürften in der Praxis jedoch die Ergebnisse aus der *tube* Simulation haben, die in Abbildung 6.10d gezeigt werden. Hier ist die Performance über die gesamte Fläche gut bis sehr gut und weist keinerlei Schwachpubkte auf und auch die Varianz ist deutlich niedriger als beispielsweise im *grid* Szenario.

Abbildung 6.10e und Abbildung 6.10f zeigen deutlich, dass für die guten Schätzungen der, über die gesamte Fläche kaum vorhandene, Bias verantwortlich ist und der vorhandene Fehler fast ausschließlich von der Varianz der Schätzungen herrührt. Somit wird der Algorithmus bei der Lokalisierung (temporär) unbeweglicher Objekte eine noch wesentlich höhere Performance erreichen, indem mehrere Messungen von derselben Position aus gemittelt werden.

In der in Abbildung 6.11 visualisierten Simulation des Algorithmus mit *ab-nlos* Fehlermodell zeigt sich ein eher durchwachsenes Bild der Performanceentwicklung bei hinzukommen einzelner Ausreißer. So fällt die Performance in Szenarien mit wenig Ankerknoten, die in Abbildung 6.11a und Abbildung 6.11c gezeigt werden, vor allem in den wichtigen Bereichen sehr stark ab.

Auch in den beiden in Abbildung 6.11b und Abbildung 6.11d visualisierten Simulationsergebnissen mit deutlich mehr Ankern bleibt das Ergebnis durchwachsen. Während im symmetrischen *grid* Szenario die Performance fast gehalten werden kann, bricht die Performance in der *tube* Konstellation wiederum gerade in den wichtigen Bereichen ein. Beide Szenarien erzielen dennoch Ergebnisse, die eine generelle Eignung des Algorithmus für die Indoorlokalisierung nicht in Frage stellen.

Eine Analyse des in Abbildung 6.11f gezeigten Phasendiagramms verdeutlicht diese Feststellung. Während der Bias fast über die gesamte Fläche nahezu unverändert ist, hat er ausgerechnet innerhalb der konvexen Hülle stark zugenommen. Die in Abbildung 6.11e gesondert analysierte Position zeigt dementsprechend folgerichtig,

dass der Bias durch die hinzugekommenen Ausreißer an dieser Stelle nicht vergrößert wurde, aber die Varianz sich orthogonal zur Hauptachse der Anker stark vergrößert hat.

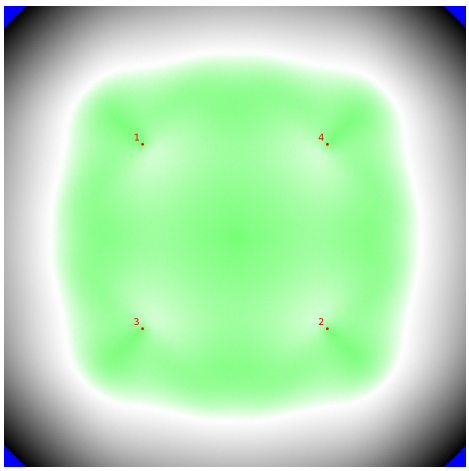
Zusammenfassend hinterlässt der E-Min-Max (W2) Algorithmus einen guten Eindruck bezüglich der Eignung für die Indoorlokalisierung. Die Stärken des originalen Algorithmus wurden beibehalten und die Schwächen zum Teil deutlich reduziert. Leider ist die Robustheit gegenüber Ausreißern noch nicht optimal und gerade in nicht symmetrischen Ankerkonstellationen kann die Performance das sehr gute Niveau nicht halten. In der Praxis dürfte dieser Algorithmus jedoch für die meisten Szenarien sehr gute Ergebnisse erzielen und die klassischen Verfahren deutlich in der Performance abhängen. Hinzu kommt die sehr niedrige Rechenzeit, die sich nur unwesentlich gegenüber des originalen Min-Max verändert hat und auch für Sensornetzhardware kein Problem darstellt.

E-Min-Max (W4)

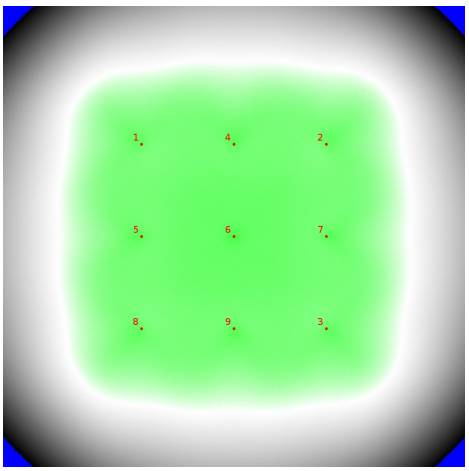
Während der E-Min-Max (W2) Algorithmus die Stärken von Min-Max beibehalten und die Schwächen abgemildert hat, setzt der in Abbildung 6.12 analysierte E-Min-Max (W4) Algorithmus einen gegenteiligen Schwerpunkt: Während die Schwächen von Min-Max nur mäßig verbessert wurden, wurden die Stärken des originalen Algorithmus weiter, zum Teil stark, ausgebaut.

In Abbildung 6.12a ist die räumliche Fehlerverteilung des E-Min-Max (W4) Algorithmus für das *quad* Setup visualisiert. Schon hier zeigt sich deutlich, dass vor allem der Bereich innerhalb der konvexen Hülle und die direkt außerhalb liegende Fläche deutliche Performancegewinne erzielen kann. So hat sich der sehr gute Bereich von den Diagonalen der Ankerknoten auf beinahe die gesamte Fläche ausgeweitet und das gesamte Ergebnis innerhalb dieser, für die Indoorlokalisierung entscheidenden Fläche, ist nun sehr homogen. Wie beim originalen Min-Max ändert sich das Ergebnis bei der Hinzunahme weiterer fünf Ankerknoten, deren Ergebnis in Abbildung 6.12b visualisiert ist, kaum, da sich die Fläche innerhalb der konvexen Hülle nicht verändert hat.

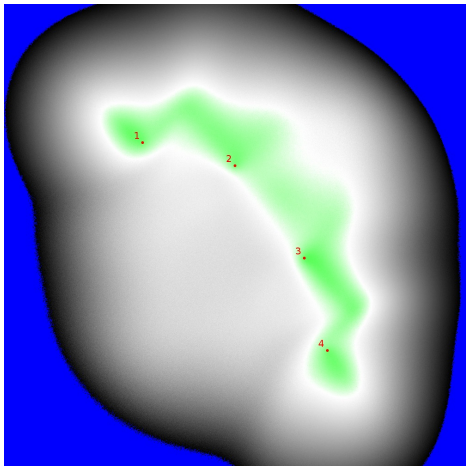
Für das schwierige *kite* Szenario werden, wie in Abbildung 6.12c gezeigt, recht gute Ergebnisse in den wichtigen Bereichen erzielt. Leider hat sich der sehr gute Bereich



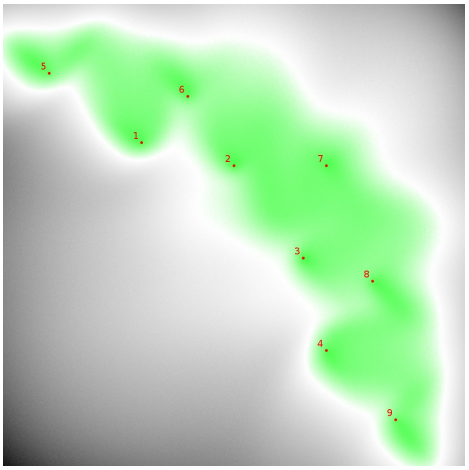
(a) Fehlerverteilung von E-Min-Max (W4) in der *quad* Simulation. Der MAE beträgt 131 %.



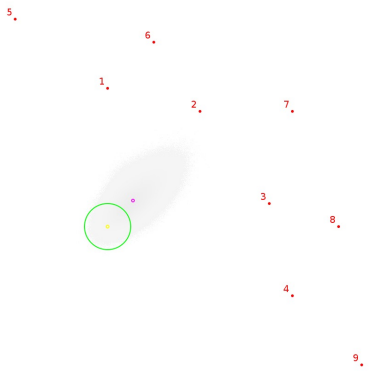
(b) Fehlerverteilung von E-Min-Max (W4) in der *grid* Simulation. Der MAE beträgt 129 %.



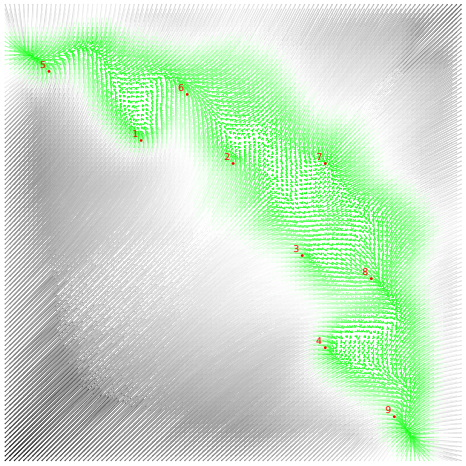
(c) Fehlerverteilung von E-Min-Max (W4) in der *kite* Simulation. Der MAE beträgt 315 %.



(d) Fehlerverteilung von E-Min-Max (W4) in der *tube* Simulation. Der MAE beträgt 141 %.

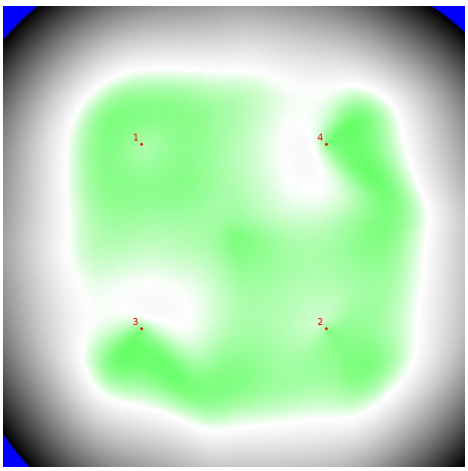


(e) Inversdarstellung von E-Min-Max (W4) in der *tube* Simulation. Der MAE beträgt 158 %.

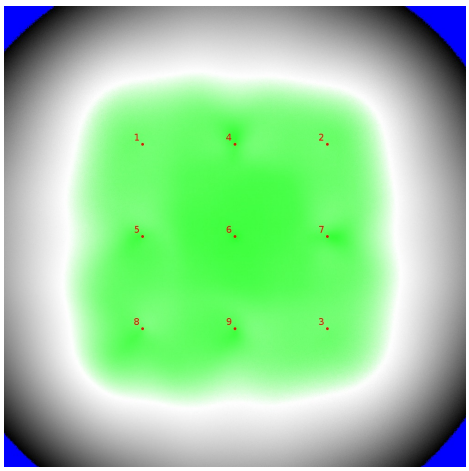


(f) Phasendiagramm von E-Min-Max (W4) in der *tube* Simulation.

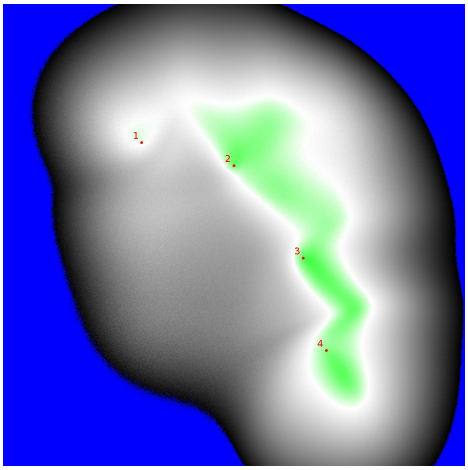
Abbildung 6.12: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den E-Min-Max (W4) Algorithmus mit nd-noise Fehlermodell.



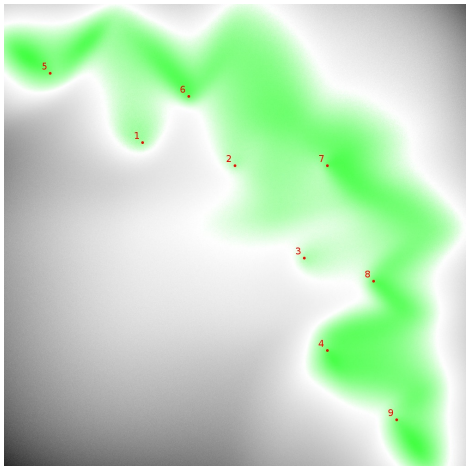
(a) Fehlerverteilung von E-Min-Max (W4) in der *quad* Simulation. Der MAE beträgt 140 %.



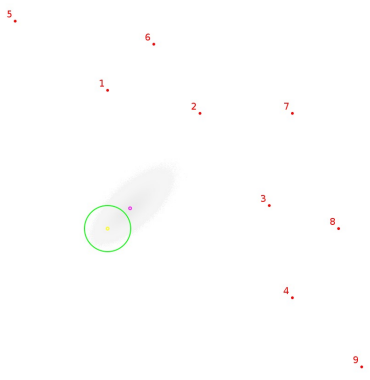
(b) Fehlerverteilung von E-Min-Max (W4) in der *grid* Simulation. Der MAE beträgt 150 %.



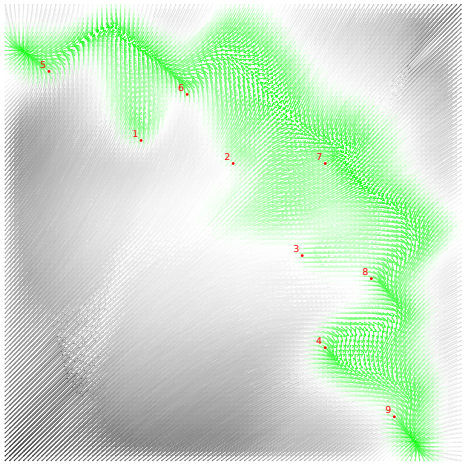
(c) Fehlerverteilung von E-Min-Max (W4) in der *kite* Simulation. Der MAE beträgt 378 %.



(d) Fehlerverteilung von E-Min-Max (W4) in der *tube* Simulation. Der MAE beträgt 135 %.



(e) Inversdarstellung von E-Min-Max (W4) in der *tube* Simulation. Der MAE beträgt 131 %.



(f) Phasendiagramm von E-Min-Max (W4) in der *tube* Simulation.

Abbildung 6.13: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den E-Min-Max (W4) Algorithmus mit ab-nlos Fehlermodell. Der Referenzbezug ist der Erwartungswert des Eingabefehlers aus der vorherigen Simulation.

im Gegensatz zu Min-Max nun etwas aus der konvexen Hülle hinaus geschoben, wenn auch der Bereich innerhalb der konvexen Hülle immer noch gute Ergebnisse erzielt. Dass die weit von den Ankerknoten entfernten Bereiche unbrauchbare Ergebnisse erzielen, dürfte in der Praxis kaum ins Gewicht fallen. Lediglich eine Eignung für die kooperative Lokalisierung in recht losen Netzwerken darf bezweifelt werden. In Abbildung 6.12d ist die räumliche Fehlerverteilung des E-Min-Max (W4) Algorithmus für das *tube* Setup mit fünf zusätzlichen Ankern visualisiert. Hier vergrößert sich der sehr gute Bereich recht stark und nimmt beinahe die gesamte konvexe Hülle ein.

Die in den Abbildungen 6.12e und 6.12f visualisierten Eigenschaften, Varianz und Bias, erinnern wiederum sehr stark an den originalen Min-Max. Innerhalb der konvexen Hülle sind die Schätzungen fast erwartungstreu und außerhalb kommt ein starker Bias hinzu, wobei die Testposition eine größere Varianz des Schätzers als die des Eingabefehlers zeigt.

Der für die Indoorloaklisierung wichtige Fall mit zusätzlichen Ausreißern in den Eingabefehlern, mittels *ab-nlos* Fehlermodell, ist in Abbildung 6.13 visualisiert. Zeigt der Algorithmus bei einem betroffenen Anker in Abbildung 6.13a schon leichte Schwächen in der Fehlerverteilung, kann er wie in Abbildung 6.13b gezeigt, bei neun Ankern in symmetrischer Aufstellung die drei Ausreißer etwas schlechter kompensieren. Verglichen mit anderen Algorithmen, ist dieses Ergebnis allerdings als sehr gut zu bezeichnen.

Noch besser stellt sich die Performance in den weniger symmetrischen *kite* und *tube* Szenarien dar, deren Ergebnisse in Abbildung 6.13c und Abbildung 6.13d gezeigt werden. Im *kite* Szenario fällt Min-Max typisch die Performance in direkter Nähe zum NLOS-Anker recht stark ab, während andere Bereiche kaum betroffen sind. In der für die Praxis aussagekräftigsten *tube* Simulation zeigen sich nur minimale Performanceunterschiede. In einigen zentralen Bereichen wurde die Performance sogar gesteigert - dieser Effekt wurde bei der Diskussion des Min-Max Algorithmus ausführlich beschrieben.

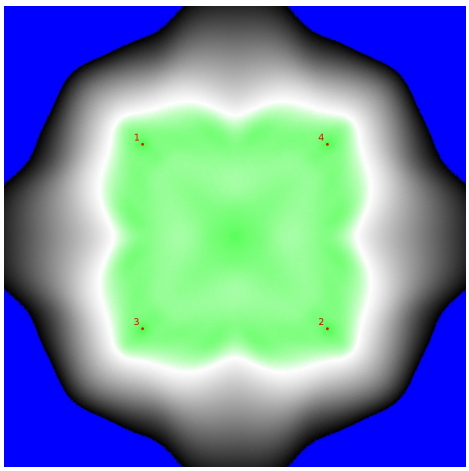
Sowohl der Bias als auch die Varianz der Schätzungen, die in Abbildung 6.13e und Abbildung 6.13f gezeigt werden, haben sich durch die Ausreißer kaum verändert. Der Bias ist in den Randbereichen der konvexen Hülle minimal gestiegen, in den Bereichen um die NLOS-Anker herum jedoch etwas zurückgegangen. Insgesamt kann weiterhin von einem fast erwartungstreuen Verhalten innerhalb der konvexen Hülle ausgegangen werden.

Der E-Min-Max (W4) Algorithmus zeigt zwar in allen durchgeführten Simulation im MAE wesentlich schlechtere Ergebnisse als der E-Min-Max (W2) Algorithmus, doch die Analyse der räumlichen Fehlerverteilungen ergibt, dass er in der Praxis mindestens genauso gut, in Einzelfällen sogar besser als der Andere abschneiden dürfte. Insbesondere das Ausweiten der sehr guten Bereiche innerhalb der konvexen Hülle der Ankerknoten dürfte in der Praxis eine hohe Relevanz haben, der starke Bias an zunehmender Entfernung zu den Ankerknoten nur in Spezialfällen. Lediglich für die kooperative Indoorlokalisierung dürfte E-Min-Max (W2) einen Vorteil aufgrund der sehr homogenen räumlichen Fehlerverteilung haben.

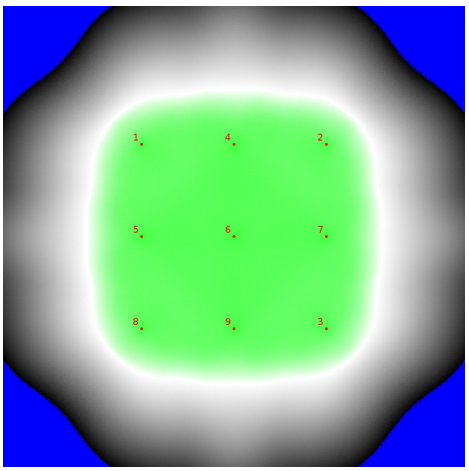
MD-Min-Max

Der MD-Min-Max Algorithmus ist ein Hybrid aus ML Algorithmus und Min-Max Algorithmus. Den Punkten der *Bounding-Box* werden anhand einer MF Funktion Gewichte zugewiesen. Wie für alle verteilungsabhängigen Algorithmen ist die Simulation der räumlichen Fehlerverteilung recht schwierig. Hauptproblem ist das für diese Algorithmen benötigte Fitting auf die zu erwartenden Fehler. In der Praxis wird man dieses Fitting anhand einer Stichprobe von Testdaten vornehmen. Dabei kann natürlich nicht garantiert werden, dass in der Praxis genau diese Verteilung wieder auftreten wird. Auch gehen diese Algorithmen von einer bestimmten Verteilung aus und ermitteln durch das Fitting nur die für diese Verteilung charakteristischen Parameter. Auch ist in der Praxis das Auftreten von NLOS-Fehlern nicht durch eine Zufallsverteilung zu beschreiben, da diese Ereignisse weder zufällig, noch voneinander unabhängig sind. So sind beim kontinuierlichen Bewegen durch ein Gebäude weder die aktuelle Position, noch das Auftreten von NLOS-Fehlern zufällig, sondern in hohem Maße von der vorherigen Position bzw. den strukturellen Eigenschaften des Gebäudes abhängig.

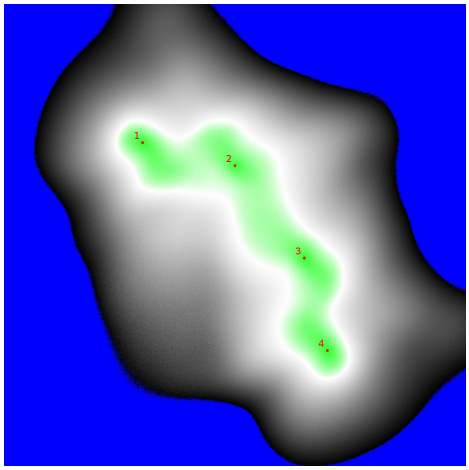
Für diese Simulation gelten diese Einschränkungen nicht. Die auftretenden Fehler sind als unabhängige Ereignisse mit einer festgelegten Zufallsverteilung definiert. Es ist also zu erwarten, dass diese Klasse von Algorithmen in der Simulation wesentlich besser abschneiden wird, als in der Praxis. Dies hat im Wesentlichen zwei Gründe: Erstens wird beim Anpassen der Algorithmen auf eine konkrete Verteilung das Fitting auf genau den Daten vorgenommen, die im anschließenden Simulationsdurchlauf auch zu erwarten sind. In der Praxis müsste man, um diesen Effekt zu erreichen, den



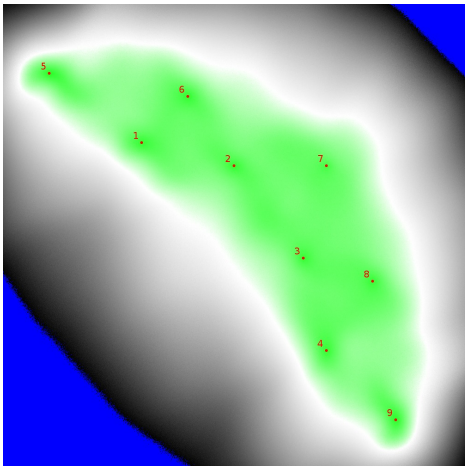
(a) Fehlerverteilung von MD-Min-Max in der *quad* Simulation. Der MAE beträgt 316 %.



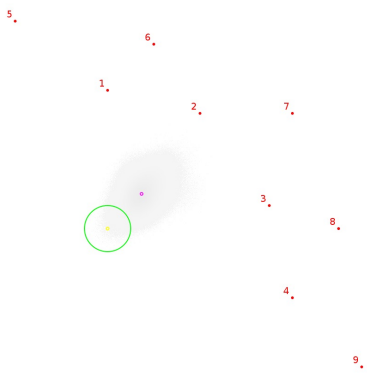
(b) Fehlerverteilung von MD-Min-Max in der *grid* Simulation. Der MAE beträgt 230 %.



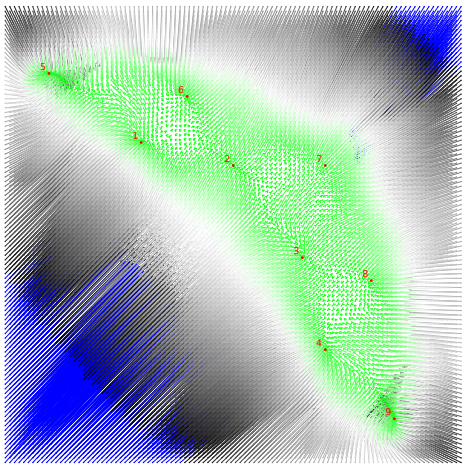
(c) Fehlerverteilung von MD-Min-Max in der *kite* Simulation. Der MAE beträgt 453 %.



(d) Fehlerverteilung von MD-Min-Max in der *tube* Simulation. Der MAE beträgt 212 %.

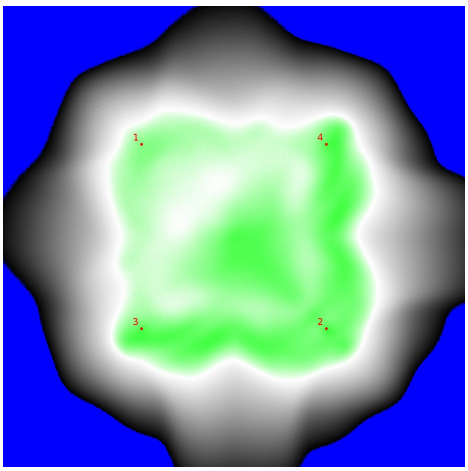


(e) Inversdarstellung von MD-Min-Max in der *tube* Simulation. Der MAE beträgt 210 %.

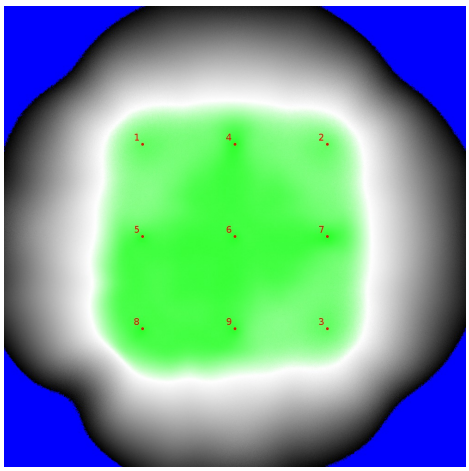


(f) Phasendiagramm von MD-Min-Max in der *tube* Simulation.

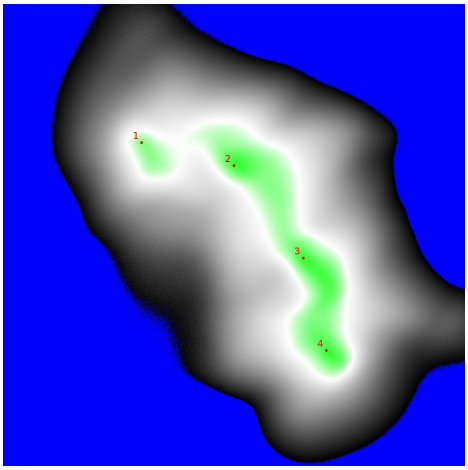
Abbildung 6.14: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den MD-Min-Max Algorithmus mit nd-noise Fehlermodell.



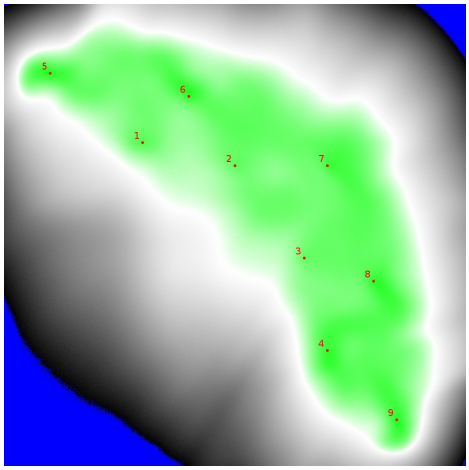
(a) Fehlerverteilung von MD-Min-Max in der *quad* Simulation. Der MAE beträgt 330 %.



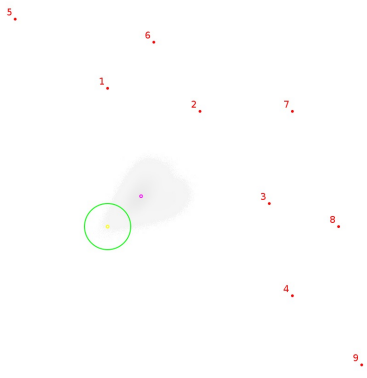
(b) Fehlerverteilung von MD-Min-Max in der *grid* Simulation. Der MAE beträgt 266 %.



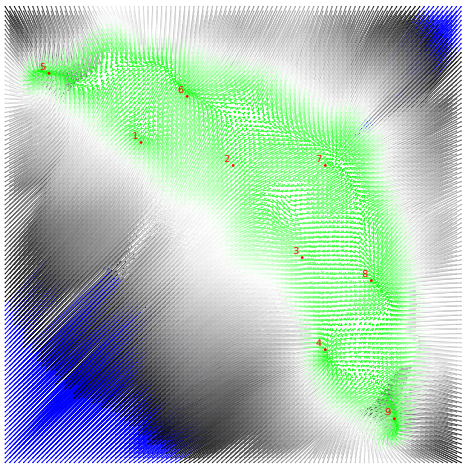
(c) Fehlerverteilung von MD-Min-Max in der *kite* Simulation. Der MAE beträgt 503 %.



(d) Fehlerverteilung von MD-Min-Max in der *tube* Simulation. Der MAE beträgt 194 %.



(e) Inversdarstellung von MD-Min-Max in der *tube* Simulation. Der MAE beträgt 196 %.



(f) Phasendiagramm von MD-Min-Max in der *tube* Simulation.

Abbildung 6.15: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den MD-Min-Max Algorithmus mit ab-nlos Fehlermodell. Der Referenzbezug ist der Erwartungswert des Eingabefehlers aus der vorherigen Simulation.

Durchlauf durch ein Gebäude mittels eines Referenzsystems aufzeichnen und dann ein Fehlerhistogramm bestimmen. Nun würde man das Fitting durchführen und dann auf denselben Daten den Algorithmus rechnen lassen. Dies ist natürlich kein realistisches Szenario, da die Daten des Referenzsystems alleine schon für eine Lokalisierung ausreichen würden. Zweitens sind die Varianzen der einzelnen Verteilungen in dieser Simulation bewusst klein gehalten, um nicht einen zu großen Kontrast in den Ausgabegrafiken zu erzeugen. Würde man hier größere Varianzen im Eingabefehler zulassen, so hätte man mit sehr großen blauen Flächen zu kämpfen und die Grafiken hätten kaum Aussagekraft. Durch die kleinen Varianzen profitieren auch ML Algorithmen von der zufälligen Verteilung, die an sich auf eine ganz andere Verteilung fitten.

Dennoch haben wir uns entschieden, die ML Algorithmen hier ebenfalls zu besprechen. Das Fitting wurde nur auf den *nd-noise* Daten vorgenommen und nicht auf den *ab-nlos* Daten, um zu zeigen, wie ein solcher Algorithmus mit Ausreißern von der erwarteten Verteilung umgeht.

In Abbildung 6.14 ist die räumliche Fehlerverteilung des MD-Min-Max Algorithmus für das *nd-noise* Fehlermodell visualisiert. Da MD-Min-Max keine festgelegte Verteilung für die ML Komponente zugrunde legt, sondern nur charakteristische Punkte aus dem Histogramm der Fehlerverteilung ermittelt, dürften diese Werte für Aussagen bezüglich des Verhaltens in der Praxis taugen. Die in den Abbildungen 6.14a und 6.14b gezeigten räumlichen Fehlerverteilungen für die beiden Szenarien mit symmetrischer Ankerverteilung zeigen eine sehr ähnliche Fehlerverteilung innerhalb der konvexen Hülle wie sie auch der E-Min-Max (W4) Algorithmus gezeigt hat. Außerhalb fällt die Performance allerdings wesentlich stärker ab und die größere algorithmische Nähe zu Min-Max wird deutlich. Auch von einer Erhöhung der Ankeranzahl auf neun Anker profitiert dieser Algorithmus innerhalb der Hülle nur mäßig. Außerhalb steigt die Performance zwar stark an, jedoch hat dies nur theoretische Bedeutung, da die Werte immer noch viel zu schlecht sind um einen Nutzen für die Praxis abzuleiten.

Ein ähnliches Bild ergibt sich für die in Abbildung 6.14c und Abbildung 6.14d gezeigten asymmetrischen Verteilungen der Anker. Der für die Praxis relevante Bereich erzielt im *kite* Setup eine ähnliche Performance wie der E-Min-Max (W4) Algorithmus, wenn auch der sehr gute Bereich deutlich günstiger positioniert ist. Dieses Verhalten wird in der *tube* Simulation besonders deutlich. Hier ist die gesamte konvexe Hülle und ein recht breiter Bereich direkt außerhalb dieser Hülle durchgängig als sehr gut zu bezeichnen.

Die in Abbildung 6.14e und Abbildung 6.14f dargestellten Metriken, Varianz und Bias, der Schätzungen zeigen ebenfalls die typische Min-Max Charakteristik. Die Varianz liegt in der Beispielauswertung nur knapp über der Eingangsvarianz des Fehlers, ist aber durch den Min-Max typischen starken Bias außerhalb der Ankerhülle stark verschoben. Innerhalb der Hülle zeigt sich in großen Bereichen eine fast absolute Erwartungstreue.

Entscheidend für die Praxis wird sein, inwiefern dieser Algorithmus mit Ausreißern umgehen kann bzw. sich durch diese beeinflussen lässt. Nochmals sei an dieser Stelle angemerkt, dass das Fitting zum vorherigen Experiment mit *nd-noise* Fehler nicht verändert wurde! Die Ergebnisse der Simulationen mit *ab-nlos* Fehler sind in Abbildung 6.15 abgebildet.

Wird die Performance des Algorithmus in Szenarien mit nur wenigen Ankern, wie in den Abbildungen 6.15a und 6.15c gezeigt, nur mäßig verschlechtert, so zeigt sich in Abbildung 6.15b und Abbildung 6.15d die Stärke dieses Algorithmus. In beiden Szenarien mit neun Ankern und drei Ausreißern kann die sowieso schon sehr gute Performance der vorherigen Simulation fast durchgängig gehalten werden und ein wirklicher Effekt der Ausreißer auf die Lokalisierungsperformance kann in den relevanten Bereichen kaum ausgemacht werden.

Dies bestätigt sich auch in den, in Abbildung 6.15e und Abbildung 6.15f gezeigten, Analysen von Varianz und Bias. In beiden Auswertungen sind ebenfalls nur marginale Unterschiede festzustellen. Die für die Auswertung der Varianz gewählte Position profitiert sogar von den Ausreißern in einer etwas erhöhten Performance, die jedoch erneut auf den in der Auswertung von Min-Max besprochenen Effekt zurückzuführen ist, der sich in der Praxis nicht gezielt ausnutzen lassen wird und an anderen Positionen nicht vorhanden sein muss.

Eine abschließende Bewertung der Ergebnisse der Simulationen für diesen Algorithmus ist schwer. Auf der einen Seite hat der Algorithmus unter denselben Bedingungen unter denen auch die anderen Algorithmen getestet wurden, eine sehr gut Performance gezeigt, allerdings sind aus den eingangs besprochenen Gründen diese Ergebnisse nicht unbedingt auf die Praxis zu übertragen. Zu groß sind die Einflüsse von Testdaten und das anschließende Fitting auf die Gesamtergebnisse. So kann in der Praxis bereits eine einzige geöffnete Tür eine völlig andere Fehlerverteilung der Entfernungsmessungen zur Folge haben und die Fehlerverteilung deutlich von den Daten, auf die gefittet wurde, abweichen lassen. Andererseits haben wir das Fitting zwischen den

beiden Simulationen nicht verändert und die Performance wurde dadurch kaum beeinflusst. Davon ausgehend dürfte der MD-Min-Max Algorithmus in der Praxis nahe bei den anderen beiden optimierten Min-Max Vertretern liegen bzw. diese sogar leicht in der Performance übertreffen. Insbesondere in Situationen wo viele Ankerknoten in Reichweite sind.

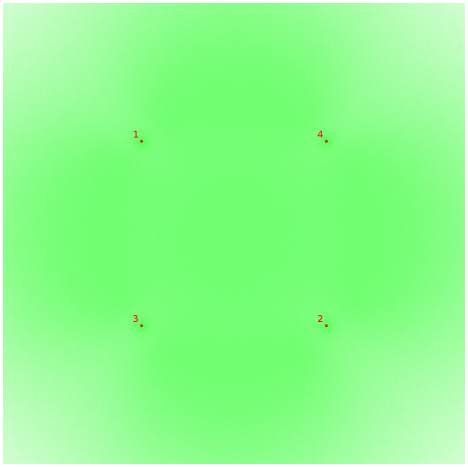
MLE-$\mathcal{N}$

In Abbildung 6.16 ist die räumliche Fehlerverteilung des MLE-$\mathcal{N}$ Algorithmus für das *nd-noise* Fehlermodell visualisiert. Für diesen Algorithmus gilt ebenfalls das schon einleitend zu MD-Min-Max Geschriebene: Da dieser Algorithmus eine ML Funktion auf eine Gauß-Verteilung ist und die Funktion genau auf die hier genutzte Verteilung gefittet wurde, sind in allen Szenarien die bestmöglichen Ergebnisse zu erwarten. Eine Übertragbarkeit in die Praxis ist ohne Weiteres aber nicht möglich, da weder ein so genaues Fitting auf die (in der Zukunft!) zu erwartenden Daten möglich ist, noch eine wirkliche Gauß-Verteilung überhaupt zu erwarten ist.

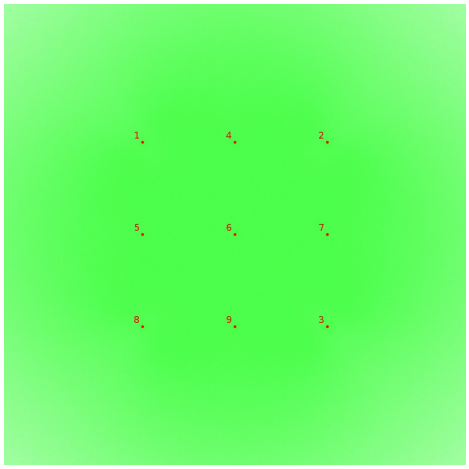
Die folgenden Grafiken sind daher nur der Vollständigkeit halber dokumentiert und eine Einschätzung für den Praxisnutzen unterbleibt ganz. Für eine Einschätzung des Praxis-Nutzens sei auf die von uns durchgeführten Experimente in Kapitel 7.4 verwiesen.

Für die Simulationen des *nd-noise* Fehlermodells sind insb. die Abbildungen 6.16e und 6.16f interessant. Hier ist sehr schön sichtbar, dass der Algorithmus auf diese Eingabedaten perfekt gefittet ist und an jeder Position der Simulationsfläche erwartungstreu ist. Auch die Varianz der geschätzten Positionen ist durch die redundanten Anker sogar geringer als die der Eingangsfehler.

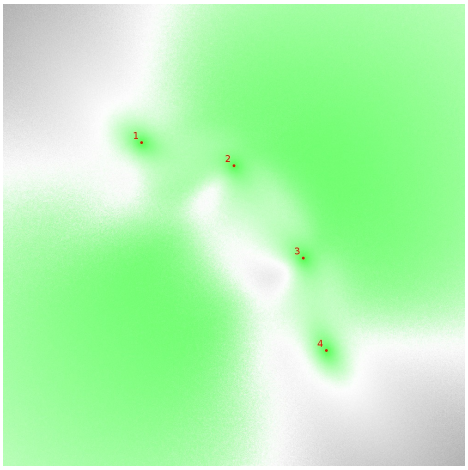
Interessanter für die Charakteristik dieses Algorithmus ist die Visualisierung des Verhaltens beim zusätzlichen Auftreten von Ausreißern in den Eingabefehlern. In Abbildung 6.17 ist die räumliche Fehlerverteilung des MLE-$\mathcal{N}$ Algorithmus für das *ab-nlos* Fehlermodell visualisiert. Erneut sei darauf hingewiesen, dass kein erneutes Fitting auf diese Fehlerverteilung durchgeführt wurde und die Fehlerverteilung auch keine Gauß-Verteilung mehr ist. Ebenfalls sei nochmals darauf hingewiesen, dass die Ausreißer auf den normalverteilten Grundfehler aufaddiert wurden und exponentialverteilte Fehler mit der Rate zwei sind. Diese Rate wurde gewählt, um die technischen



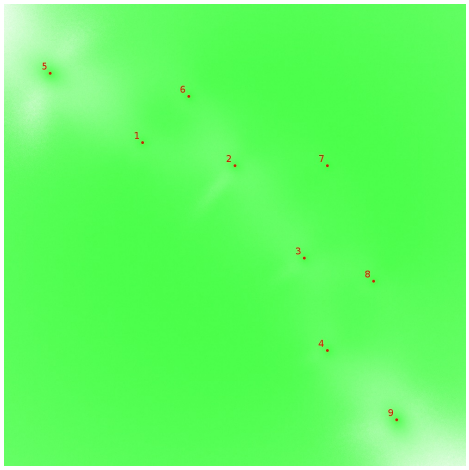
(a) Fehlerverteilung von MLE-$\mathcal{N}$ in der *quad* Simulation. Der MAE beträgt 54 %.



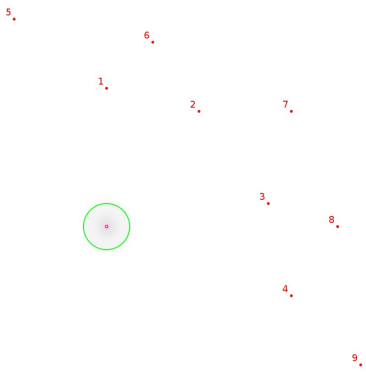
(b) Fehlerverteilung von MLE-$\mathcal{N}$ in der *grid* Simulation. Der MAE beträgt 39 %.



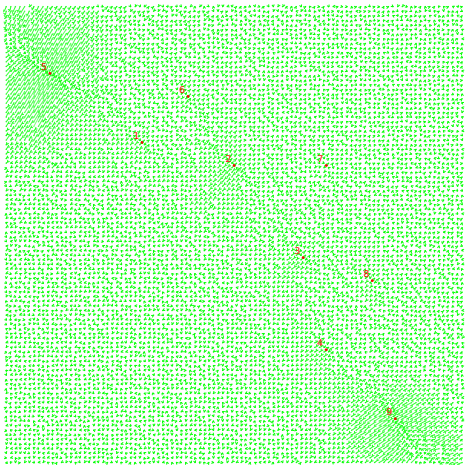
(c) Fehlerverteilung von MLE-$\mathcal{N}$ in der *kite* Simulation. Der MAE beträgt 80 %.



(d) Fehlerverteilung von MLE-$\mathcal{N}$ in der *tube* Simulation. Der MAE beträgt 37 %.

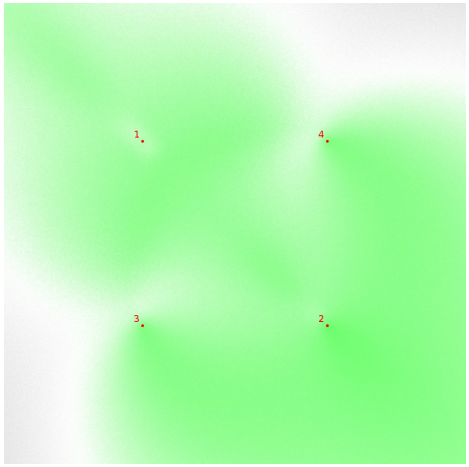


(e) Inversdarstellung von MLE-$\mathcal{N}$ in der *tube* Simulation. Der MAE beträgt 0 %.

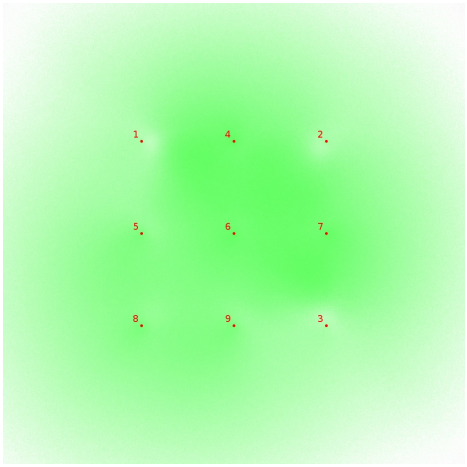


(f) Phasendiagramm von MLE-$\mathcal{N}$ in der *tube* Simulation.

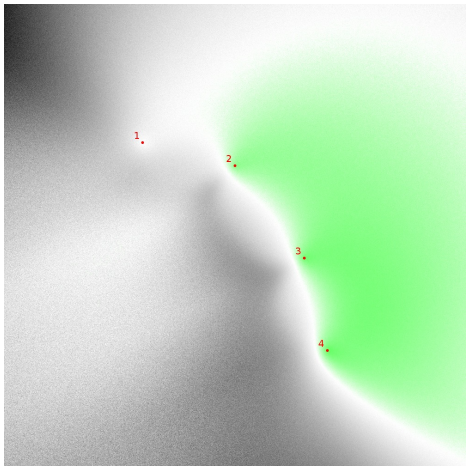
Abbildung 6.16: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den MLE-$\mathcal{N}$ Algorithmus mit nd-noise Fehlermodell.



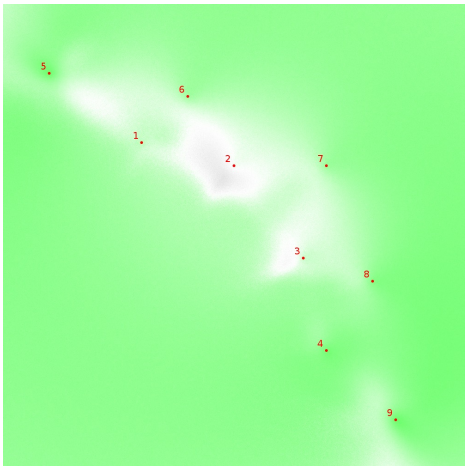
(a) Fehlerverteilung von MLE-$\mathcal{N}$ in der *quad* Simulation. Der MAE beträgt 72 %.



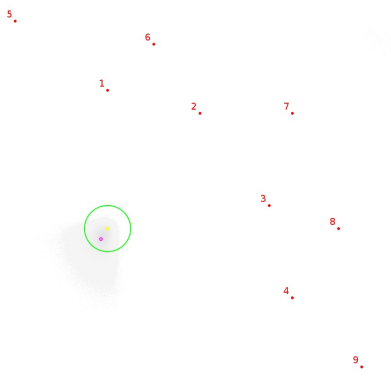
(b) Fehlerverteilung von MLE-$\mathcal{N}$ in der *grid* Simulation. Der MAE beträgt 69 %.



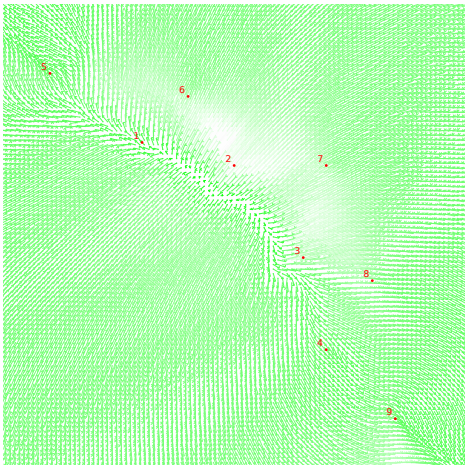
(c) Fehlerverteilung von MLE-$\mathcal{N}$ in der *kite* Simulation. Der MAE beträgt 149 %.



(d) Fehlerverteilung von MLE-$\mathcal{N}$ in der *tube* Simulation. Der MAE beträgt 62 %.



(e) Inversdarstellung von MLE-$\mathcal{N}$ in der *tube* Simulation. Der MAE beträgt 54 %.



(f) Phasendiagramm von MLE-$\mathcal{N}$ in der *tube* Simulation.

Abbildung 6.17: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den MLE-$\mathcal{N}$ Algorithmus mit ab-nlos Fehlermodell. Der Referenzbezug ist der Erwartungswert des Eingabefehlers aus der vorherigen Simulation.

Möglichkeiten der Visualisierung bestmöglich auszunutzen - wäre eine kleinere Rate genutzt worden, so wären Großteile der Grafiken sehr dunkel eingefärbt gewesen und hätten einen deutlich niedrigeren Kontrastumfang gezeigt.

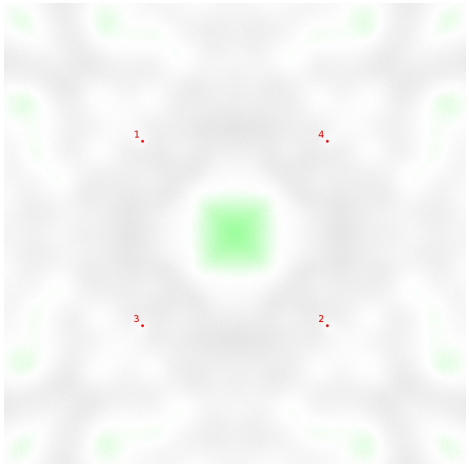
In den ersten beiden Auswertungen mit symmetrischer Ankerplatzierung, die in den Abbildungen 6.17a und 6.17b gezeigt sind, ist zu erkennen, dass die Performance an einigen Stellen relativ stark einbricht, obwohl sich der Erwartungswert des Distanzfehlers über alle Anker nur mäßig verändert hat. In den Abbildungen 6.17c und 6.17d zeigt sich, dass diese Einbrüche für den asymmetrischen Fall noch deutlicher sind und zusätzlich in den für eine Indoorlokalisierung bedeutsamen Flächen auftreten. Vor direkten Rückschlüssen auf ein Verhalten in der Praxis sei hier jedoch noch einmal gewarnt!

Die Analyse des inversen Falles in Abbildung 6.17e und des Phasendiagramms in Abbildung 6.17f zeigen, dass durch die Ausreißer ein leichter Bias hinzugefügt wurde und die Verteilung um den Mittelwert der Schätzungen auch nicht mehr symmetrisch ist.

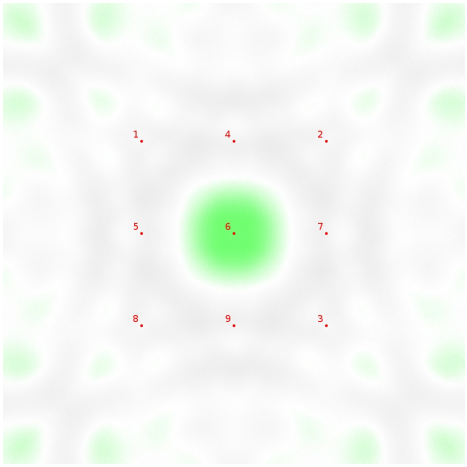
Als Ergebnis dieser Simulation bleibt lediglich festzuhalten, was für alle ML Algorithmen gilt: die Performance ist direkt von der Güte des Fittings abhängig, sowohl in Bezug auf die Stichprobe als auch in Bezug auf die Verteilfunktion. In der Praxis wird es vom Einzelfall abhängen, ob diese beiden Kriterien erfüllt werden können oder nicht. Zumindest für diesen Algorithmus dürfte es in der Praxis jedoch schwer sein, die hier gezeigte Performance zu halten, da unabhängig von der Güte und Reproduzierbarkeit der Stichprobe, keine Gauß-Verteilung des Eingabefehlers zu erwarten ist. ML Algorithmen für andere Verteilungen sind uns aus der Literatur nicht bekannt.

VBLE

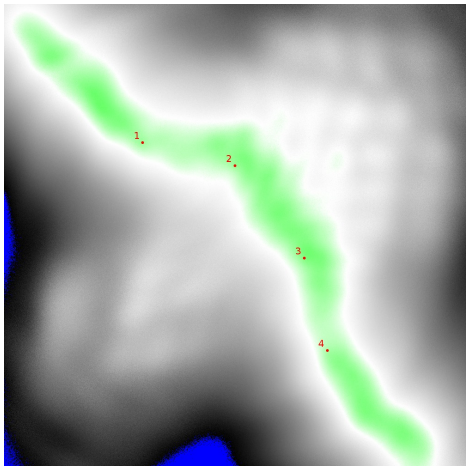
In Abbildung 6.18 ist die räumliche Fehlerverteilung des VBLE Algorithmus für das *nd-noise* Fehlermodell visualisiert. Der Algorithmus ist der erste hier diskutierte Algorithmus, der das Problem der Lokalisierung als ein geometrisches Problem, dass über die Schnittpunkte von Kreisscheiben definiert ist, auffasst. Anders als die später diskutierten Algorithmen, versucht dieser Algorithmus die Schnittpunkte nicht anhand fester Beziehungen oder Grenzwerte zu clustern, sondern stellt iterativ ein zweidimensionales Histogramm dieser Schnittpunkte auf. Der iterative Verfeinerungs-



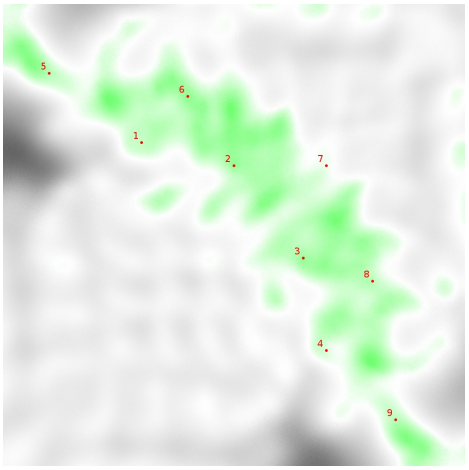
(a) Fehlerverteilung von VBLE in der *quad* Simulation. Der MAE beträgt 114 %.



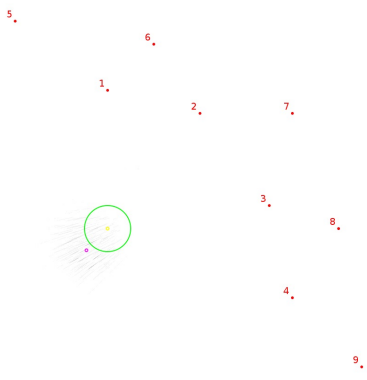
(b) Fehlerverteilung von VBLE in der *grid* Simulation. Der MAE beträgt 107 %.



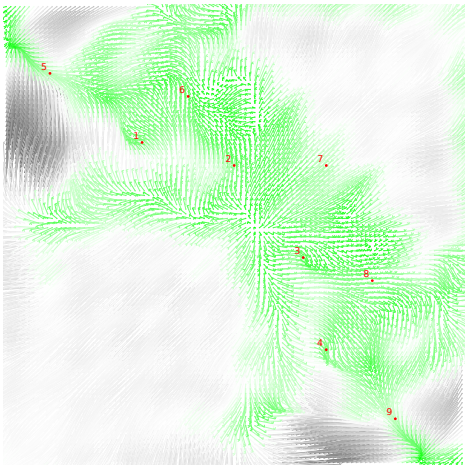
(c) Fehlerverteilung von VBLE in der *kite* Simulation. Der MAE beträgt 220 %.



(d) Fehlerverteilung von VBLE in der *tube* Simulation. Der MAE beträgt 119 %.

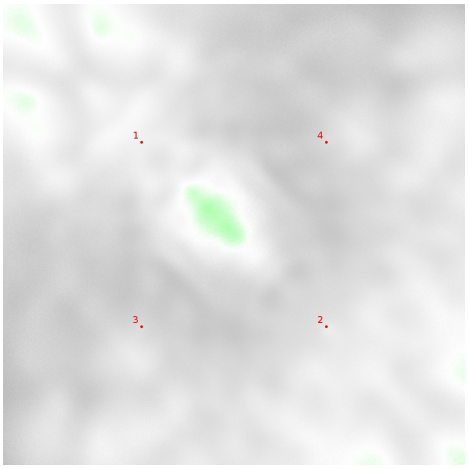


(e) Inversdarstellung von VBLE in der *tube* Simulation. Der MAE beträgt 131 %.

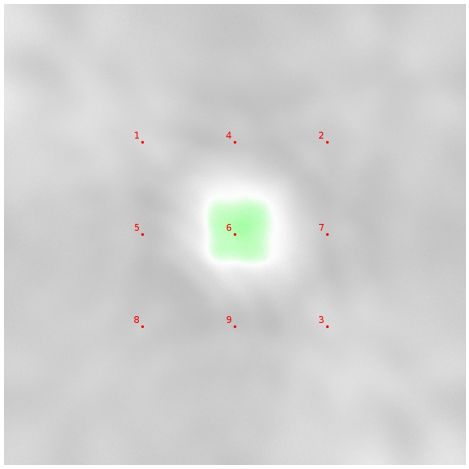


(f) Phasendiagramm von VBLE in der *tube* Simulation.

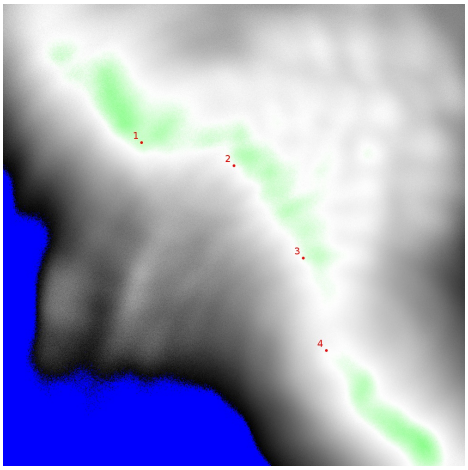
Abbildung 6.18: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den VBLE Algorithmus mit nd-noise Fehlermodell.



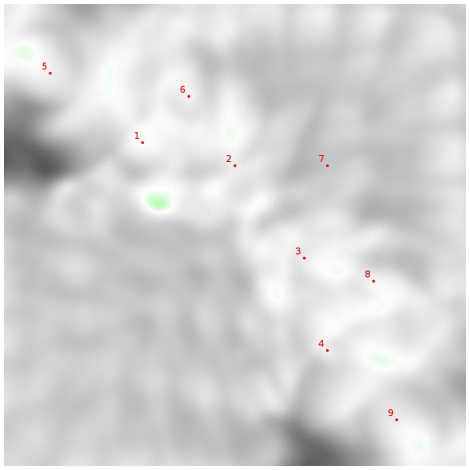
(a) Fehlerverteilung von VBLE in der *quad* Simulation. Der MAE beträgt 147 %.



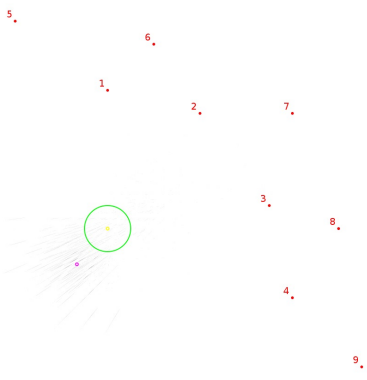
(b) Fehlerverteilung von VBLE in der *grid* Simulation. Der MAE beträgt 171 %.



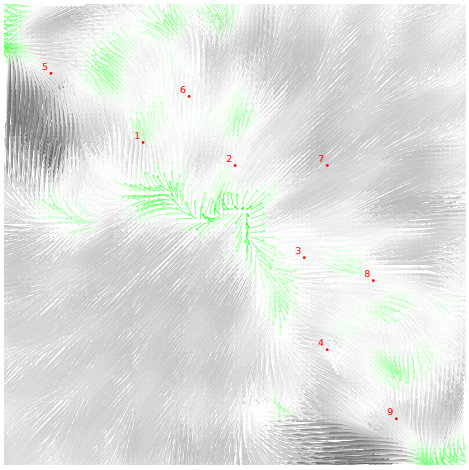
(c) Fehlerverteilung von VBLE in der *kite* Simulation. Der MAE beträgt 261 %.



(d) Fehlerverteilung von VBLE in der *tube* Simulation. Der MAE beträgt 169 %.



(e) Inversdarstellung von VBLE in der *tube* Simulation. Der MAE beträgt 204 %.



(f) Phasendiagramm von VBLE in der *tube* Simulation.

Abbildung 6.19: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den VBLE Algorithmus mit ab-nlos Fehlermodell. Der Referenzbezug ist der Erwartungswert des Eingabefehlers aus der vorherigen Simulation.

prozess ist dabei nicht durch absolute Grenzwerte in der zu erwartenden Qualität der Schätzung begrenzt, sondern durch Speicher- und Laufzeitanforderungen.

Die in den Abbildungen 6.18a und 6.18b visualisierten Simulationsergebnisse für Szenarien mit symmetrischer Ankerverteilung unterscheiden sich dann auch deutlich von den bisher diskutierten Algorithmen. Während das Ergebnis über die gesamte Fläche als gut zu bezeichnen ist und überhaupt nicht zu den Rändern hin abzufallen scheint, so sind die Bereiche mit sehr guten Ergebnissen vergleichsweise klein. Auch wenn das Clustering durch die grafische Musterung der Ergebnisse gut zu erkennen ist, so sind die Ergebnisse dieses Algorithmus gerade in Bezug auf die Gesamtfläche als sehr homogen zu bezeichnen.

Bei einem sehr asymmetrischen Szenario mit nur vier Ankern bricht die Performance wie in Abbildung 6.18c visualisiert ist zwar relativ stark ein, zeigt jedoch in der für die meisten Indoor-Szenarien besonders wichtigen Fläche um die konvexe Hülle herum ein gutes bis sehr gutes Verhalten. In dem in Abbildung 6.18d visualisierten *tube* Szenario, kommen weitere fünf Anker hinzu und die Fehlerverteilung zeigt sich fast auf der gesamten Fläche erneut als sehr homogen. Wenngleich der sehr gute Bereich innerhalb der konvexen Hülle schwächer als bei einigen Min-Max Algorithmen ausfällt.

In der inversen Darstellung, die in Abbildung 6.18e gezeigt wird, ist der Einfluss der minimalen Clustergröße besonders gut zu erkennen. Da letztendlich als Schätzung nur ein Clustermittelpunkt gewählt werden kann, ist die Verteilung der Schätzungen keine homogene Fläche, sondern eine Sammlung eben dieser Clustermittelpunkte. Diese sind zwar recht symmetrisch und mit einer Varianz, die in etwa der des Eingabefehlers entspricht, um den Mittelwert angeordnet, weisen aber zusätzlich noch einen mittelstarken Bias auf. Das Phasendiagramm in Abbildung 6.18f zeigt nochmal deutlich, dass der resultierende Fehler hauptsächlich auf einen durch die Geometrie verursachten Bias zurückzuführen ist und die Varianz der Schätzungen nicht wesentlich im Vergleich zum Eingabefehler angestiegen ist.

Die in Abbildung 6.19 visualisierten Ergebnisse, die unter Verwendung des *ab-nlos* Fehlermodells entstanden sind, bestätigen die Erkenntnisse aus der vorherigen Simulation. Kein Szenario hat seine Geometrie und Performanceverteilung wesentlich verändert und alle Fehler sind linear um etwa den Faktor angestiegen, um den auch der Erwartungswert des Distanzfehlers über alle Anker angestiegen ist. Für die in den Abbildungen 6.19a und 6.19b visualisierten symmetrischen Setups bedeutet dies eine Beibehaltung der sehr homogenen Verteilung, aber im Durchschnitt auch nur

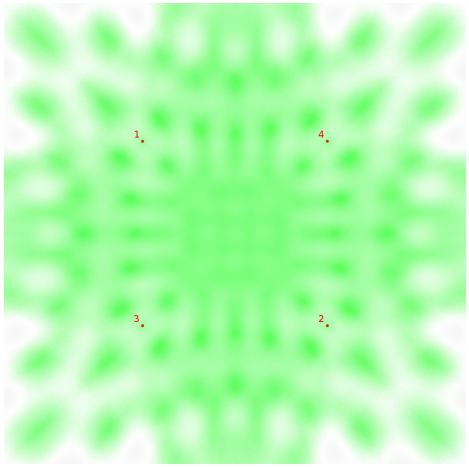
noch befriedigende Werte. Der jeweils im inneren Zentrum verbliebene sehr gute Bereich dürfte sich in der Praxis von den wenigsten Installationen effektiv ausnutzen lassen. Dieses Verhalten kann analog für die weniger symmetrischen *kite* und *tube* Szenarien in den Abbildungen 6.19c und 6.19d beobachtet werden. Auffallend im Vergleich der vier Szenarien ist, dass der Algorithmus mit zunehmender Ankerzahl seine Abhängigkeit der Performance von der Ankeranzahl zu verlieren scheint - oder andersherum: ab einer gewissen Ankeranzahl spielt die geometrische Konstellation der Anker nur noch eine untergeordnete Rolle. Diese Beobachtung, gepaart mit der beobachteten sehr homogenen Fehlerverteilung lässt auf eine hohe Eignung für die kooperative Indoorlokalisierung schließen - sofern der Rechenzeitaufwand und vor allem der Speicherbedarf einer Nutzung hier nicht entgegenstehen.

Abbildung 6.19e und insbesondere Abbildung 6.19f zeigen, dass der höhere Erwartungswert des Eingabefehlers sich über die gesamte Fläche als ein vergrößerter Bias niederschlägt.

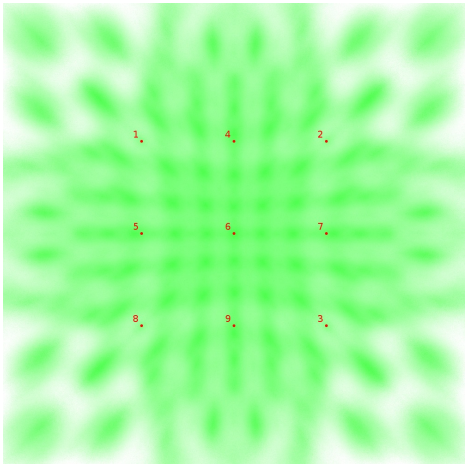
Der VBLE Algorithmus basiert nicht nur auf einem für die Indoorlokalisierung außergewöhnlichen Ansatz, sondern zeigt auch eine außergewöhnliche räumliche Fehlerverteilung. Kann er in der in der Praxis, mit der zu erwartenden Performance nicht ganz mit den bisher stärksten Algorithmen mithalten, so lässt ihn die Analyse der räumlichen Fehlerverteilung für eine Reihe von Spezialfällen sehr gut geeignet erscheinen. Insbesondere die kooperative Lokalisierung oder Szenarien in denen Ankerknoten ohne Planung ad hoc ausgebracht werden müssen, sind aufgrund der Unabhängigkeit von der Geometrie und die homogene Fehlerverteilung, die Stärken dieses Algorithmus. Lediglich der hohe Speicherbedarf und der gesteigerte Rechenaufwand lassen eine Nutzung auf Kleinstcomputern wie beispielsweise Sensorknoten zumindest fraglich erscheinen.

VBLE-O

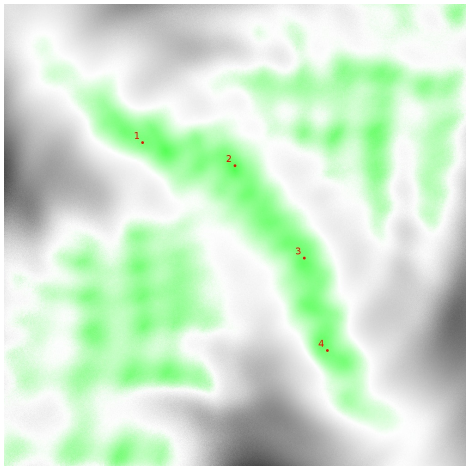
In Abbildung 6.20 ist die räumliche Fehlerverteilung des VBLE-O Algorithmus für das *nd-noise* Fehlermodell visualisiert. Der Algorithmus ist eine Weiterentwicklung des vorher besprochenen VBLE Algorithmus durch uns. Insbesondere wurde der Algorithmus dem zu erwartenden Verhalten des Distanzfehlers in der Indoorlokalisierung angepasst.



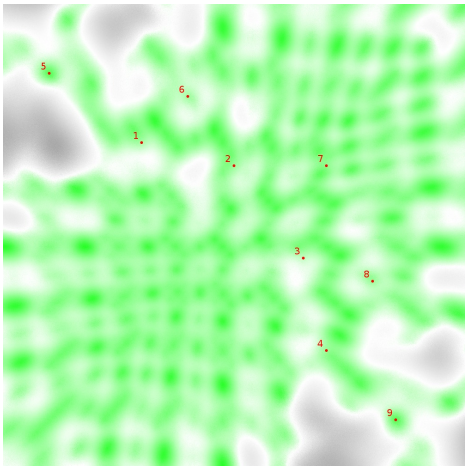
(a) Fehlerverteilung von VBLE-O in der *quad* Simulation. Der MAE beträgt 69 %.



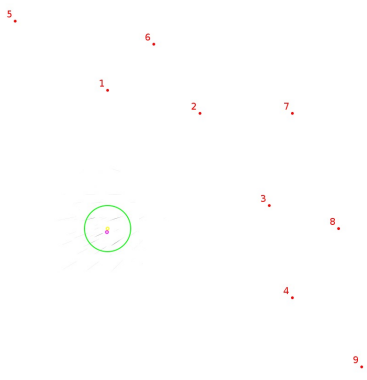
(b) Fehlerverteilung von VBLE-O in der *grid* Simulation. Der MAE beträgt 63 %.



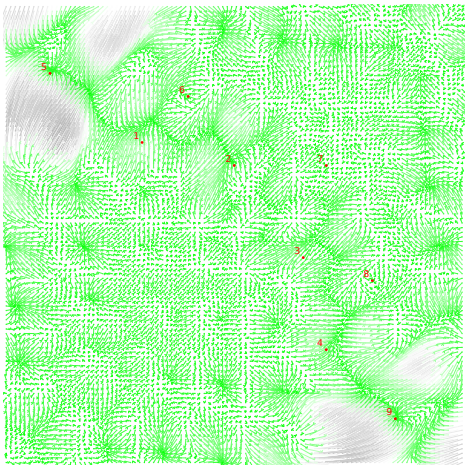
(c) Fehlerverteilung von VBLE-O in der *kite* Simulation. Der MAE beträgt 122 %.



(d) Fehlerverteilung von VBLE-O in der *tube* Simulation. Der MAE beträgt 78 %.

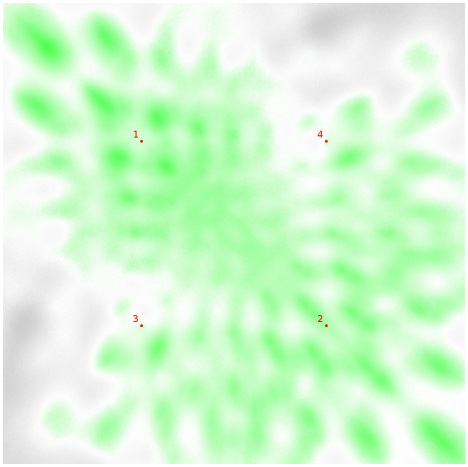


(e) Inversdarstellung von VBLE-O in der *tube* Simulation. Der MAE beträgt 15 %.

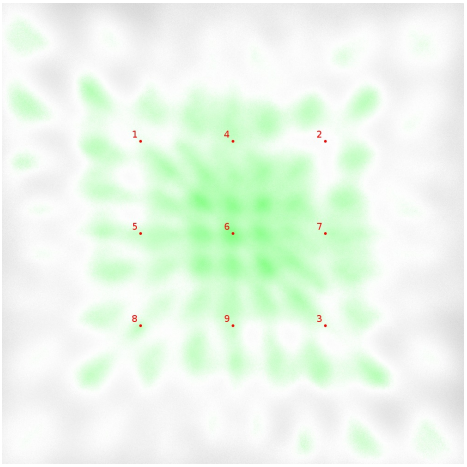


(f) Phasendiagramm von VBLE-O in der *tube* Simulation.

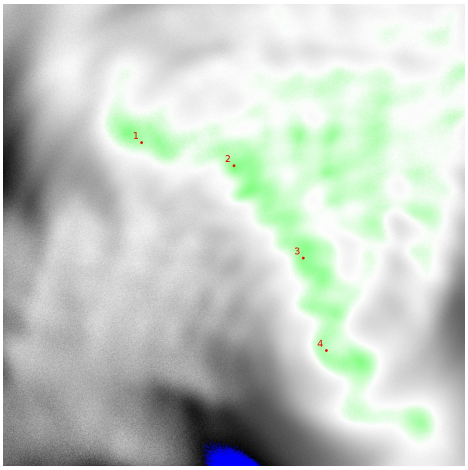
Abbildung 6.20: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den VBLE-O Algorithmus mit nd-noise Fehlermodell.



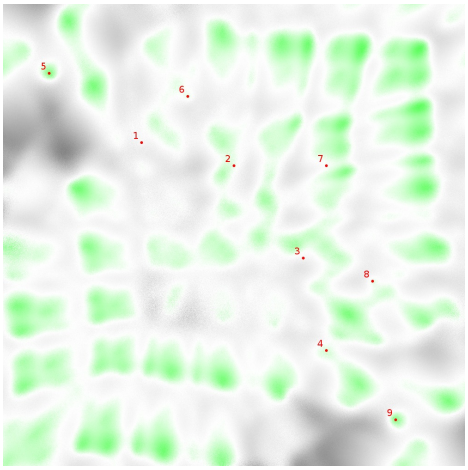
(a) Fehlerverteilung von VBLE-O in der *quad* Simulation. Der MAE beträgt 89 %.



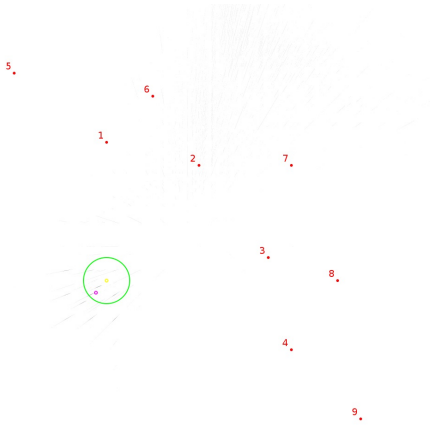
(b) Fehlerverteilung von VBLE-O in der *grid* Simulation. Der MAE beträgt 103 %.



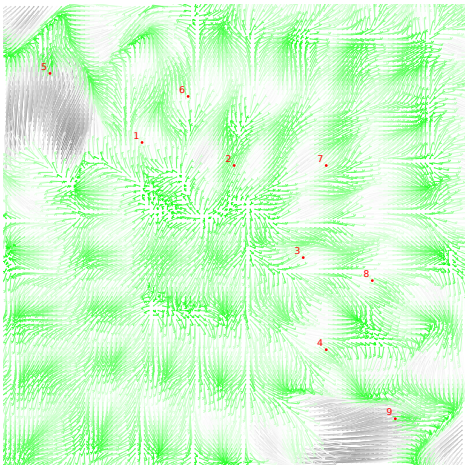
(c) Fehlerverteilung von VBLE-O in der *kite* Simulation. Der MAE beträgt 174 %.



(d) Fehlerverteilung von VBLE-O in der *tube* Simulation. Der MAE beträgt 110 %.



(e) Inversdarstellung von VBLE-O in der *tube* Simulation. Der MAE beträgt 70 %.



(f) Phasendiagramm von VBLE-O in der *tube* Simulation.

Abbildung 6.21: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den VBLE-O Algorithmus mit ab-nlos Fehlermodell. Der Referenzbezug ist der Erwartungswert des Eingabefehlers aus der vorherigen Simulation.

Für die in Abbildung 6.20a und Abbildung 6.20b gezeigten symmetrischen Ankerkonstellationen wird sofort deutlich, dass VBLE-O in diesen Szenarien eine wesentliche Verbesserung gegenüber dem Vorgänger darstellt. Über die gesamte Fläche ist die Performance als gut bis sehr gut zu bezeichnen und innerhalb der konvexen Hülle der Ankerknoten ist die Performance in beiden Szenarien sehr gut.

Auch in den weniger symmetrischen *kite* und *tube* Konstellationen sind die, in den Abbildungen 6.20c und 6.20d gezeigten, Ergebnisse eine wesentliche Verbesserung gegenüber dem Original. Die Performance ist fast in der gesamten Fläche gut bis sehr gut und auch die Homogenität bleibt über wesentliche Bereiche erhalten. Vor allem das für die meisten Algorithmen schwierige *kite* Szenario erzielt nun wesentlich bessere Ergebnisse - vor allem in den relevanten Bereichen.

In der inversen Darstellung und dem Phasendiagramm, welche in Abbildung 6.20e und Abbildung 6.20f gezeigt werden, wird deutlich, dass sowohl die Varianz der Schätzung als auch der Bias, zum Teil deutlich, verringert wurden. Im Großteil der Simulationsfläche ist gar kein Bias mehr vorhanden und der Schätzer somit als erwartungstreu zu bezeichnen.

In Abbildung 6.21 ist die räumliche Fehlerverteilung des VBLE-O Algorithmus für das, für die Beurteilung der Indoor-Performance, wichtigere *ab-nlos* Fehlermodell visualisiert. Hier haben die Veränderungen durch die hinzugekommenen Ausreißer bei den Distanzmessungen denselben Einfluss wie der, der auch beim originalen Algorithmus schon besprochen wurde. Die Performance verschlechtert sich annäherungsweise linear über die gesamte Fläche, ohne das wesentliche Änderungen der eigentlichen Geometrie der räumlichen Verteilung zu beobachten sind. Für die symmetrischen Ankerkonstellationen in den Abbildungen 6.21a und 6.21b sind lediglich leichte Verschiebungen der bisher symmetrischen Fehlerverteilung durch die Ausreißer erkennbar.

Diese Beobachtungen gelten analog für die in den Abbildungen 6.21c und 6.21d dargestellte *kite* bzw. *tube* Konstellationen. In der *tube* Konstellation fällt auf, dass der für die Praxis wichtige Bereich innerhalb der konvexen Hülle etwas stärker an Performance verloren hat, als die restliche Fläche.

Inverse Darstellung und Phasendiagramm in den Abbildungen 6.21e und 6.21f zeigen, wie auch schon beim originalen Algorithmus besprochen, dass der zusätzliche Fehler in den Distanzmessungen sich im Wesentlichen in einem erhöhten Bias und nicht in einer vergrößerten Varianz niedergeschlagen hat.

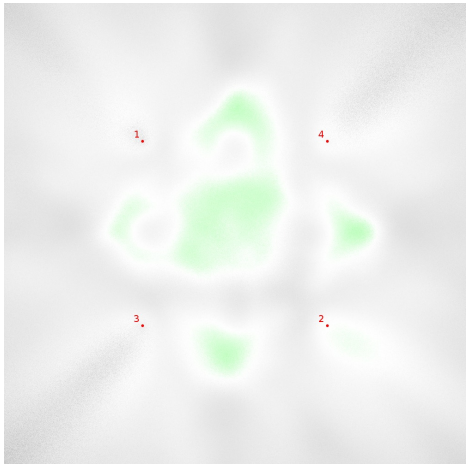
Alles für den VBLE Algorithmus geschriebene hat auch hier Bestand. Durch die deutlich verbesserte Performance ist dieser Algorithmus jetzt offensichtlich auch für die Indoorlokalisierung in allgemeinen Szenarien gut geeignet, auch wenn eine noch homogenere Performance innerhalb der konvexen Hülle wünschenswert wäre. Insbesondere durch die Absenkung der Speicher- und Rechenzeitanforderungen dürfte dieser Algorithmus für einen Einsatz in der kooperativen Indoorlokalisierung sehr gut geeignet sein.

ICLA

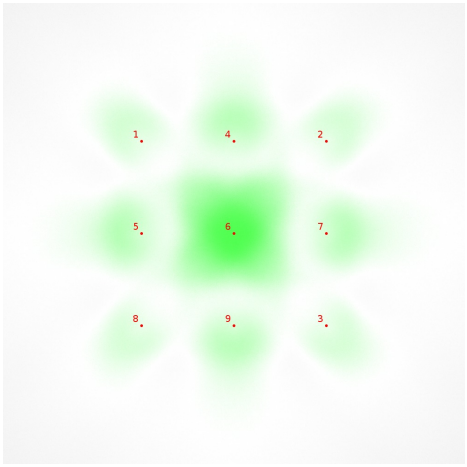
In Abbildung 6.22 ist die räumliche Fehlerverteilung des ICLA Algorithmus für das Fehlermodell *nd-noise* visualisiert. Der ICLA Algorithmus basiert wie die folgenden Algorithmen auch, auf der Berechnung von Schnittpunkten, der um die Ankerknoten, mit den gemessenen Entfernungen als Radien, gebildeten Kreisen. Ziel dieser Algorithmen ist das Clustering dieser Schnittpunkte mit verschiedenen Ansätzen und die anschließende Bildung von Mittelwerten über die verbleibenden Schnittpunkte. Aufgrund des Clusteransatzes sollten diese Algorithmen relativ stabil gegen Ausreißer sein und stark von einer Erhöhung der Ankerzahl profitieren.

In Abbildung 6.22a ist die räumliche Fehlerverteilung des ICLA Algorithmus für das *quad* Szenario visualisiert. Wie zu erwarten, ist das Ergebnis bei nur vier Ankern eher durchschnittlich, auch wenn es keine wirklichen Schwächen gibt. Wird die Ankeranzahl, wie in Abbildung 6.22b gezeigt, um weitere fünf Anker erhöht, so steigert sich die Performance über die gesamte Fläche recht stark. Insbesondere außerhalb der konvexen Hülle ist das Ergebnis im Vergleich zu den bisher diskutierten Algorithmen gut.

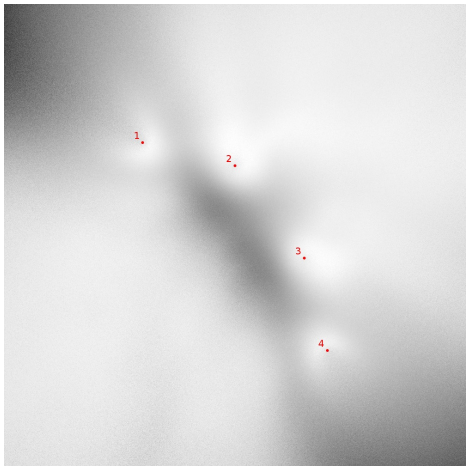
Wird die Anordnung der Anker verändert und die Symmetrie der Ankerknoten abgeschwächt, wie es bei den beiden in Abbildung 6.22c und Abbildung 6.22d gezeigten Szenarien der Fall ist, so bricht für das *kite* Szenario mit nur vier Ankern die Performance sehr stark ein. Ausgerechnet der wichtige Bereich um die Ankerknoten herum ist besonders betroffen und schon hier wird deutlich, dass der Algorithmus für Indoor-Szenarien, mit nur wenig Ankern in Reichweite, nicht die erste Wahl sein dürfte. Im günstigeren *tube* Szenario fällt auf, dass die Performance gegenüber dem symmetrischen *grid* Szenario, kaum eingebrochen ist und zusätzlich das bisher homogenste



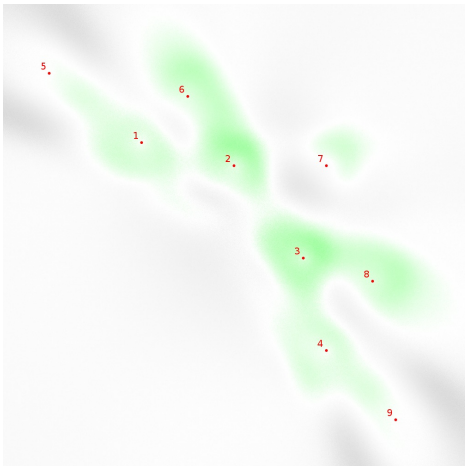
(a) Fehlerverteilung von ICLA in der *quad* Simulation. Der MAE beträgt 123 %.



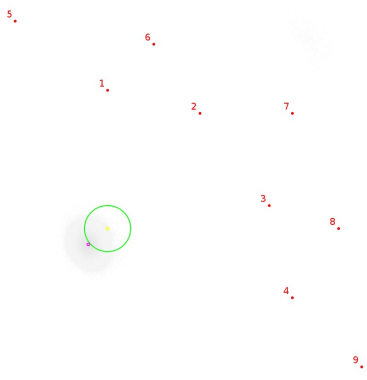
(b) Fehlerverteilung von ICLA in der *grid* Simulation. Der MAE beträgt 98 %.



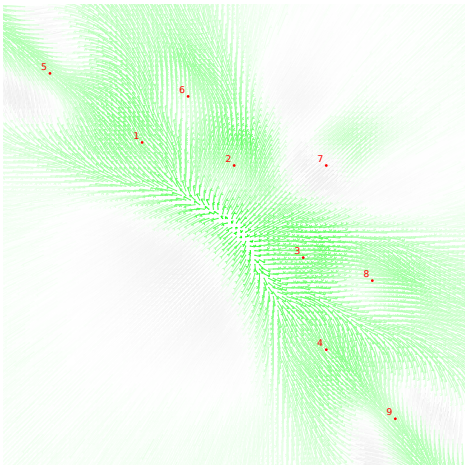
(c) Fehlerverteilung von ICLA in der *kite* Simulation. Der MAE beträgt 167 %.



(d) Fehlerverteilung von ICLA in der *tube* Simulation. Der MAE beträgt 105 %.

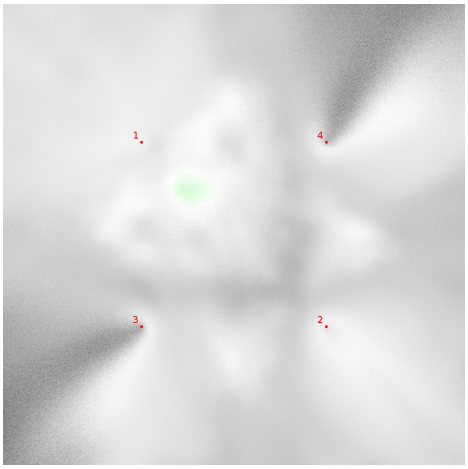


(e) Inversdarstellung von ICLA in der *tube* Simulation. Der MAE beträgt 108 %.

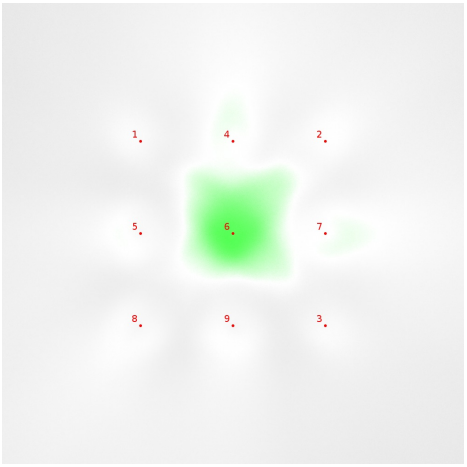


(f) Phasendiagramm von ICLA in der *tube* Simulation.

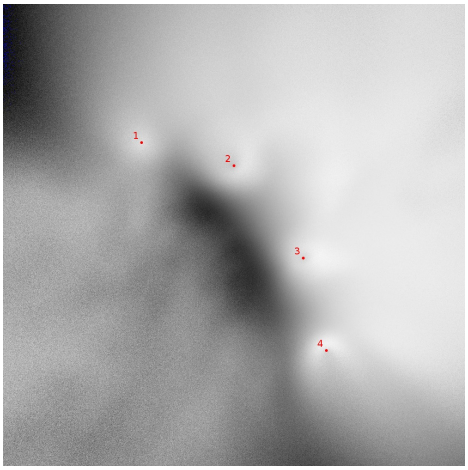
Abbildung 6.22: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den ICLA Algorithmus mit nd-noise Fehlermodell.



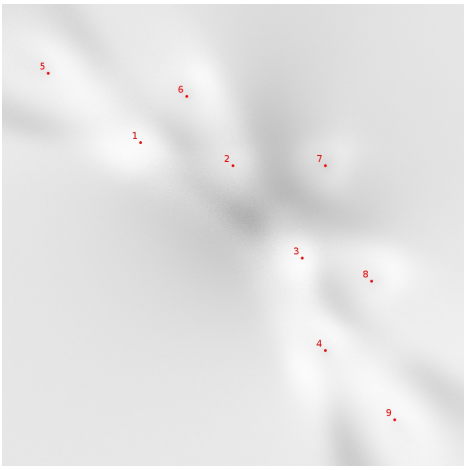
(a) Fehlerverteilung von ICLA in der *quad* Simulation. Der MAE beträgt 160 %.



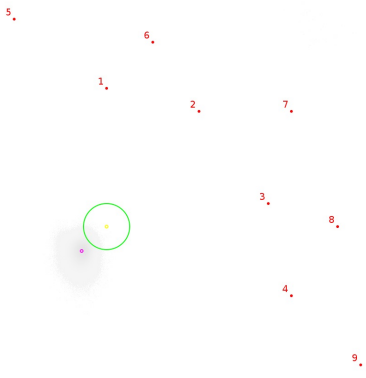
(b) Fehlerverteilung von ICLA in der *grid* Simulation. Der MAE beträgt 116 %.



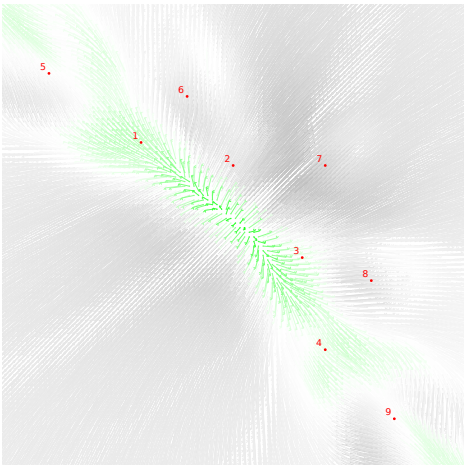
(c) Fehlerverteilung von ICLA in der *kite* Simulation. Der MAE beträgt 229 %.



(d) Fehlerverteilung von ICLA in der *tube* Simulation. Der MAE beträgt 144 %.



(e) Inversdarstellung von ICLA in der *tube* Simulation. Der MAE beträgt 151 %.



(f) Phasendiagramm von ICLA in der *tube* Simulation.

Abbildung 6.23: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den ICLA Algorithmus mit ab-nlos Fehlermodell. Der Referenzbezug ist der Erwartungswert des Eingabefehlers aus der vorherigen Simulation.

Ergebnis für dieses aussagekräftigste Szenario erzielt wurde.

Die Analyse der inversen Darstellung und die des Phasendiagramms, in den Abbildungen 6.22e und 6.22f, zeigen, dass der Algorithmus die Varianz des Eingabefehlers, trotz des Ansatzes die Schnittpunkte zu clustern, auch bei neun Ankerknoten nicht wirklich mindern kann, dafür aber auch nur einen sehr geringen Bias erzeugt.

In Abbildung 6.23 ist die räumliche Fehlerverteilung des ICLA Algorithmus für das Fehlermodell *ab-nlos* visualisiert. Für die symmetrischen Szenarien aus den Abbildungen 6.23a und 6.23b bestätigen sich Ergebnisse der vorherigen Simulationen in Bezug auf die Abhängigkeit der Performance von der Ankeranzahl. Leider bricht die Performance insgesamt relativ stark ein, obwohl das aufgrund der Funktionsweise des Algorithmus nicht unbedingt zu erwarten gewesen wäre. Auch die für die Praxis relevanteren *kite* und *tube* Szenarien aus den Abbildungen 6.23c und 6.23d zeigen ähnliche Ergebnisse. Vor allem die Auswertung des *kite* Szenarios zeigt noch einmal deutlich die Abhängigkeit von der Ankeranzahl. So hat sich hier die Performance durch das Auftreten von Ausreißern in den Messwerten von nur einem Anker wesentlich verschlechtert. Eine Lokalisierung innerhalb der konvexen Hülle der Ankerknoten ist in diesem Szenario kaum noch möglich. Im für die Praxis wohl relevantesten *tube* Szenario fällt die Verschlechterung der Performance wesentlich schwächer aus, so dass eine Lokalisierung über die gesamte Fläche mit guten Ergebnissen möglich scheint. Vor allem die sehr große Homogenität der Verteilung bleibt erhalten.

Die in Abbildung 6.23e gezeigte inverse Darstellung für eine Beispielposition zeigt, dass die zusätzlichen Ausreißer an dieser Position zu einer Verzerrung in der Symmetrie der Schätzungen und zu einem leicht höheren Bias geführt haben. Die Varianz der Schätzungen hingegen scheint kaum betroffen. Dieses Ergebnis wird in der Phasendarstellung für das gesamte Szenario bestätigt. Die in Abbildung 6.23f präsentierte Auswertung zeigt eine Zunahme des Bias über die gesamte Fläche.

Für die Anwendung in der Praxis hinterlässt der ICLA Algorithmus einen zwiespältigen Eindruck: Einerseits sind diverse Szenarien denkbar, in denen der Algorithmus durchweg gute Ergebnisse erwarten lässt, andererseits dürfte in einer großen Anzahl an Installationen ein Algorithmus aus der Min-Max Gruppe eine deutlich bessere Performance als dieser Algorithmus zeigen. Dies fällt vor allem ins Gewicht, wenn zusätzlich noch die Laufzeit beider Algorithmen vergleichen wird - hier ist Min-Max wesentlich günstiger. Für den generischen Einsatz in ad hoc Szenarien, in denen Anker nicht strategisch platziert werden können, sollte ICLA jedoch eine überdurchschnitt-

liche Performance erwarten lassen.

Bilateration

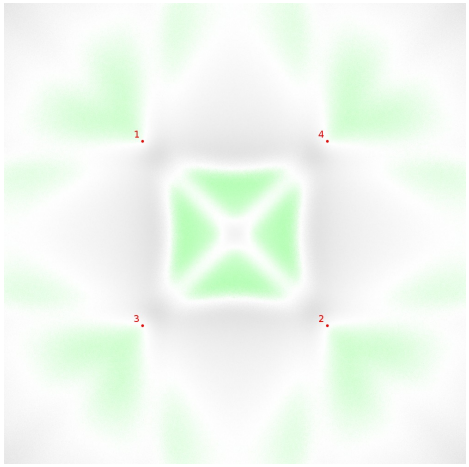
In Abbildung 6.24 ist die räumliche Fehlerverteilung des BL Algorithmus für das *nd-noise* Fehlermodell visualisiert. Dieser Algorithmus verfolgt ebenfalls einen geometrischen Ansatz über das Clustern von Kreisschnittpunkten.

Für die in den Abbildungen 6.24a und 6.24b dargestellten Verteilungen für symmetrische Szenarien fällt erneut die Abhängigkeit der Performance von der Ankeranzahl ins Auge. Die Performance ist für beide Szenarien durchgängig gut bis sehr gut. Vor allem an den Rändern der Simulationsfläche ist kein wirklicher Abfall der Performance zu erkennen und das, obwohl die Hülle der Ankerknoten nur 16 % der Fläche belegt.

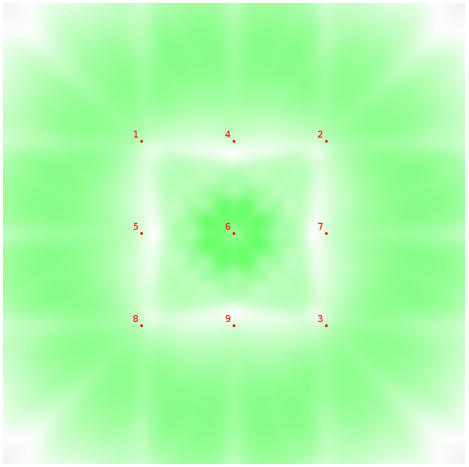
Wie auch schon beim ICLA Algorithmus fällt die Performance durch eine Aufhebung der Symmetrie bei der Positionierung der Ankerknoten verhältnismäßig stark ab, wie für die *kite* und *tube* Szenarien in Abbildung 6.24c und Abbildung 6.24d visualisiert ist. Auch wenn im *kite* Szenario die Performance erneut ausgerechnet in der konvexen Hülle der Ankerknoten nicht optimal ist, so ist eine Lokalisierung im Großteil der Fläche mit einer guten Performance möglich. Im *tube* Szenario wird sogar fast durchweg eine sehr gute Performance erzielt.

Die Verteilung der Schätzungen, verglichen mit der Verteilung der Fehler, ähnelt sehr stark dem schon bei ICLA beobachteten Schema. Wie in den Abbildungen 6.24e und 6.24f gezeigt, entsteht der Fehler im Erwartungswert zu etwa gleichen Teilen durch einen orthogonal zur Hauptachse der Ankerknoten ausgerichteten Bias und einer Varianz, die in etwa der des Eingangsfehlers entspricht. Die einzelnen Schätzungen sind für die Testpositionen sehr symmetrisch um das Mittel verteilt.

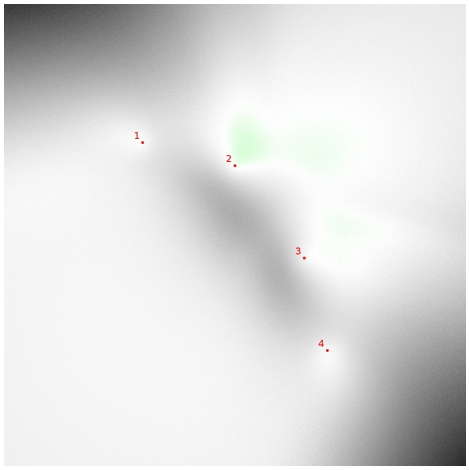
In Abbildung 6.25 ist die räumliche Fehlerverteilung des BL Algorithmus für das *ab-nlos* Fehlermodell visualisiert. Die Simulation des *quad* Szenarios, deren Ergebnisse in Abbildung 6.25a gezeigt werden, zeigt, dass Ausreißer bei den gemessenen Distanzen einen je nach Position sehr unterschiedlichen Einfluss haben. Leider ist ein großer Teil der in der konvexen Hülle der Ankerknoten liegenden Positionen von diesem Effekt betroffen und der Algorithmus kann für diese Szenarien nicht mit einfacheren Algorithmen, wie beispielsweise E-Min-Max (W2), mithalten. Bei Hinzunahme weiterer fünf Anker fällt der Performanceabfall, wie in Abbildung 6.25b gezeigt, geringer



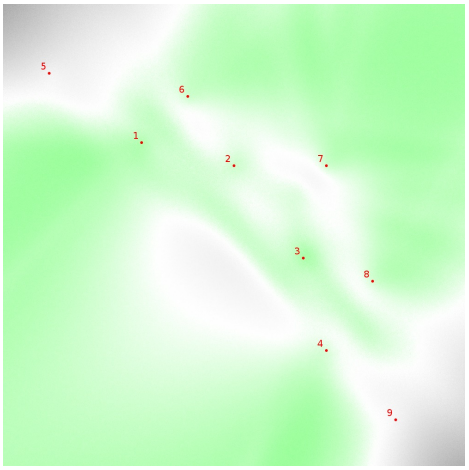
(a) Fehlerverteilung von BL in der *quad* Simulation. Der MAE beträgt 104 %.



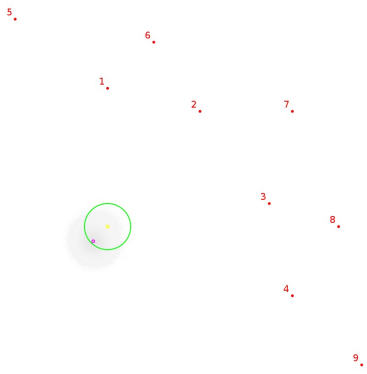
(b) Fehlerverteilung von BL in der *grid* Simulation. Der MAE beträgt 68 %.



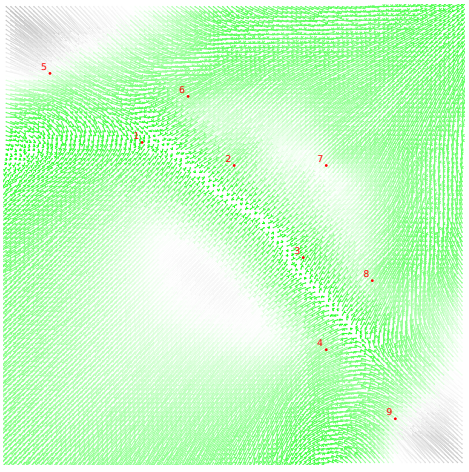
(c) Fehlerverteilung von BL in der *kite* Simulation. Der MAE beträgt 152 %.



(d) Fehlerverteilung von BL in der *tube* Simulation. Der MAE beträgt 87 %.

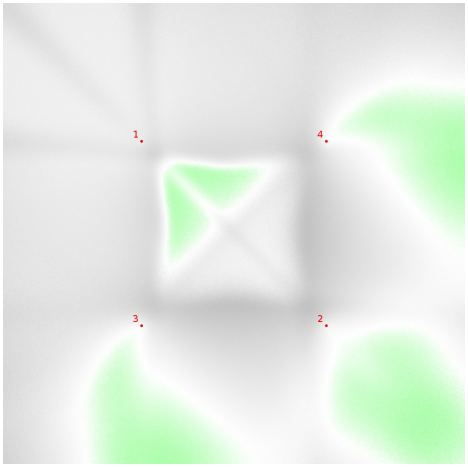


(e) Inversdarstellung von BL in der *tube* Simulation. Der MAE beträgt 89 %.

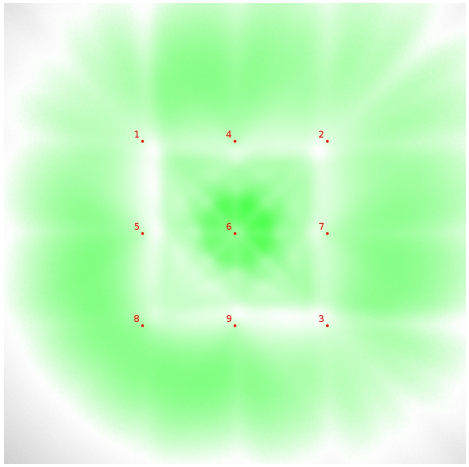


(f) Phasendiagramm von BL in der *tube* Simulation.

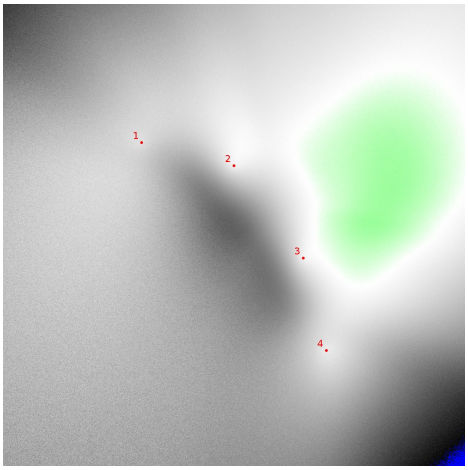
Abbildung 6.24: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den BL Algorithmus mit nd-noise Fehlermodell.



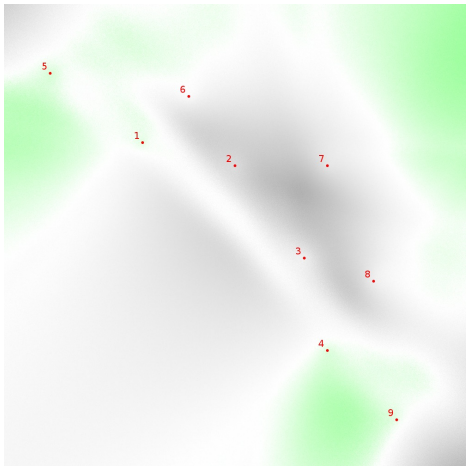
(a) Fehlerverteilung von BL in der *quad* Simulation. Der MAE beträgt 125 %.



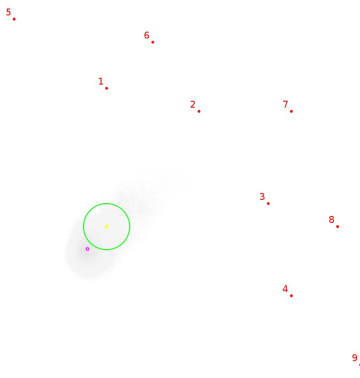
(b) Fehlerverteilung von BL in der *grid* Simulation. Der MAE beträgt 77 %.



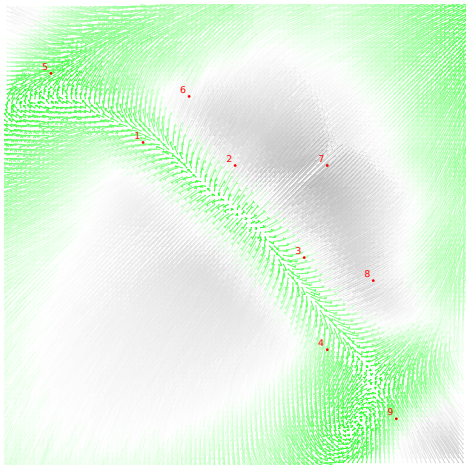
(c) Fehlerverteilung von BL in der *kite* Simulation. Der MAE beträgt 197 %.



(d) Fehlerverteilung von BL in der *tube* Simulation. Der MAE beträgt 113 %.



(e) Inversdarstellung von BL in der *tube* Simulation. Der MAE beträgt 127 %.



(f) Phasendiagramm von BL in der *tube* Simulation.

Abbildung 6.25: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den BL Algorithmus mit ab-nlos Fehlermodell. Der Referenzbezug ist der Erwartungswert des Eingabefehlers aus der vorherigen Simulation.

aus und innerhalb der konvexen Hülle werden fast durchgängig sehr gute Fehlerwerte erreicht.

In den asymmetrischen Szenarien, deren Ergebnisse in Abbildung 6.25c und Abbildung 6.25d gezeigt werden, zeigen sich erneut Schwächen in den für die Indoorlokalisierung wichtigen Bereichen innerhalb der konvexen Hülle. Das *tube* Szenario erzielt in diesem Bereich trotz Vorhandensein von neun Ankern nicht überall gute Ergebnisse.

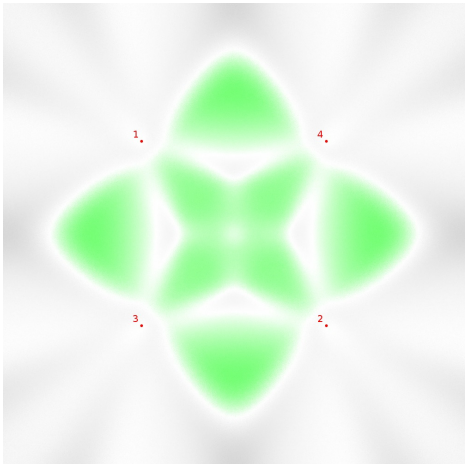
Die Auswertung der inverser Darstellung in Abbildung 6.25e und des Phasendiagramms in Abbildung 6.25f zeigen deutlich, dass in diesem Algorithmus die Ausreißer in den Distanzmessungen über die gesamte Fläche in einer höheren Varianz der Schätzungen münden. Da diese Varianz im Verhältnis zum Bias sehr groß ist, wirkt sie sich negativ auf dem durchschnittlichen Lokalisierungsfehler aus.

Für die Praxis hinterlässt der BL Algorithmus, ebenso wie ICLA, einen durchwachsenen Eindruck. Während die reinen Durchschnittswerte des Fehlers in den meisten Szenarien sehr niedrig ausfallen, ist die räumliche Verteilung der Fehler für die Indoorlokalisierung nicht optimal. In der Praxis dürfte dieser Algorithmus in vielen Szenarien eine wesentlich schlechtere Performance zeigen, als die reinen MAE Werte vermuten lassen. Verglichen mit ICLA ist die Rechenzeit allerdings wesentlich niedriger und die Eignung für die meisten Anwendungen ist insgesamt wesentlich höher als die von ICLA einzuschätzen.

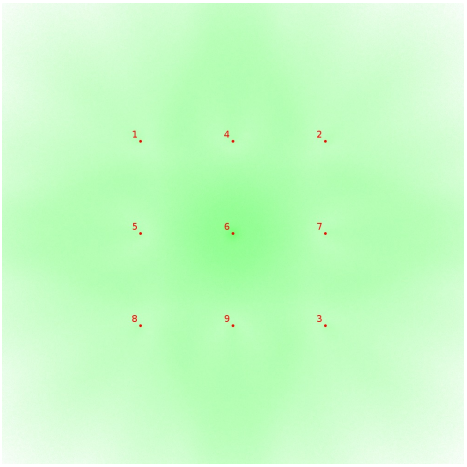
Geo-n

In Abbildung 6.26 ist die räumliche Fehlerverteilung des Geo-n Algorithmus für das *nd-noise* Fehlermodell visualisiert. Der von uns im Rahmen dieser Arbeit entwickelte Algorithmus ist ebenfalls ein Vertreter der geometriebasierten Klasse von Algorithmen, die auf den Schnittpunkten um die Anker gedachter Kreise arbeiten.

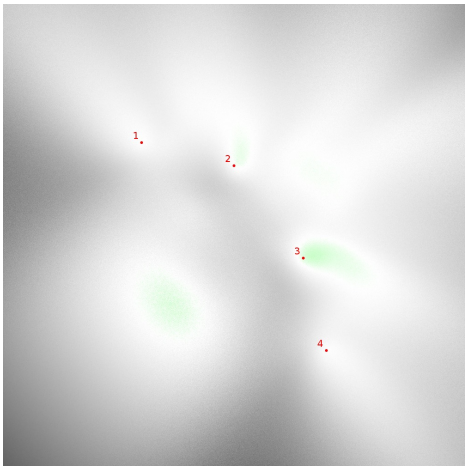
Im Vergleich mit den anderen Algorithmen dieser Klasse zeigt die Auswertung der symmetrischen Szenarien in den Abbildungen 6.26a und 6.26b keine großen Überraschungen. Die Auswertung der *quad* Simulation zeigt einen vergleichbaren MAE wie die Auswertung von BL, wenngleich auch die sehr guten Bereiche stärker im Zentrum konzentriert sind, was der Indoorlokalisierung zu Gute kommt. Die Auswertung des *grid* Szenarios ergibt eine sehr große Homogenität der durchweg sehr guten Ergebnisse. Auf der gesamten Fläche werden hier Positionsfehler erzielt, die niedriger sind



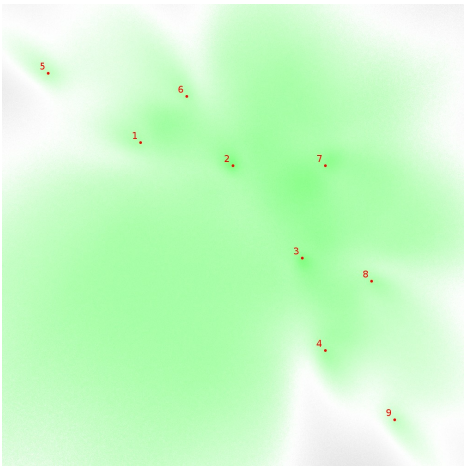
(a) Fehlerverteilung von Geo-n in der *quad* Simulation. Der MAE beträgt 104 %.



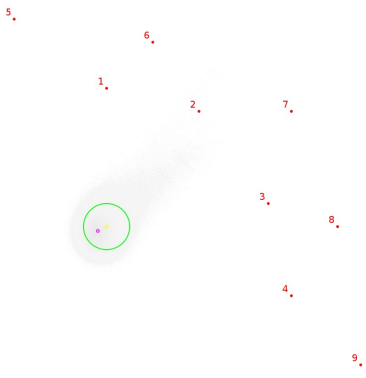
(b) Fehlerverteilung von Geo-n in der *grid* Simulation. Der MAE beträgt 77 %.



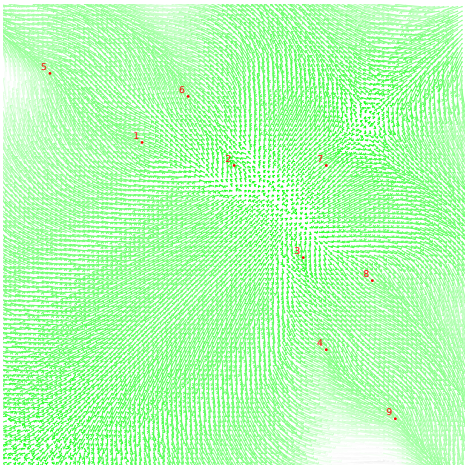
(c) Fehlerverteilung von Geo-n in der *kite* Simulation. Der MAE beträgt 162 %.



(d) Fehlerverteilung von Geo-n in der *tube* Simulation. Der MAE beträgt 80 %.

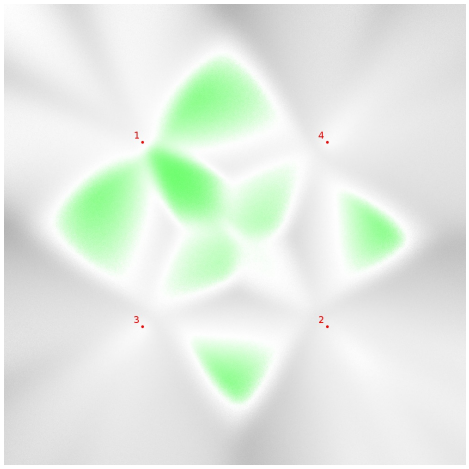


(e) Inversdarstellung von Geo-n in der *tube* Simulation. Der MAE beträgt 42 %.

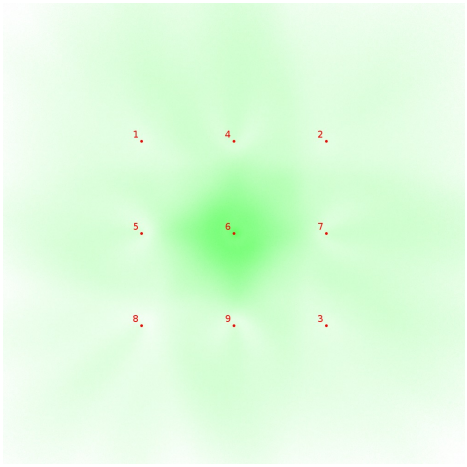


(f) Phasendiagramm von Geo-n in der *tube* Simulation.

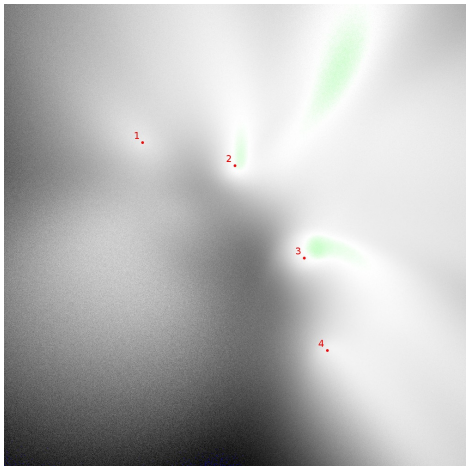
Abbildung 6.26: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den Geo-n Algorithmus mit nd-noise Fehlermodell.



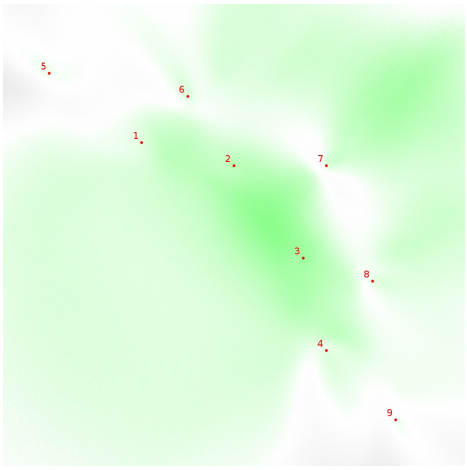
(a) Fehlerverteilung von Geo-n in der *quad* Simulation. Der MAE beträgt 126 %.



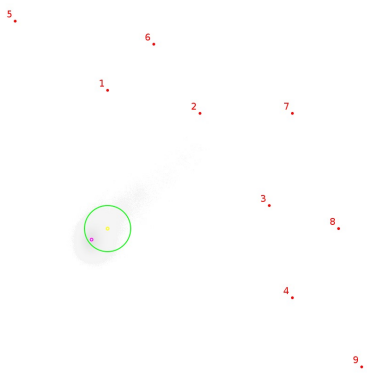
(b) Fehlerverteilung von Geo-n in der *grid* Simulation. Der MAE beträgt 87 %.



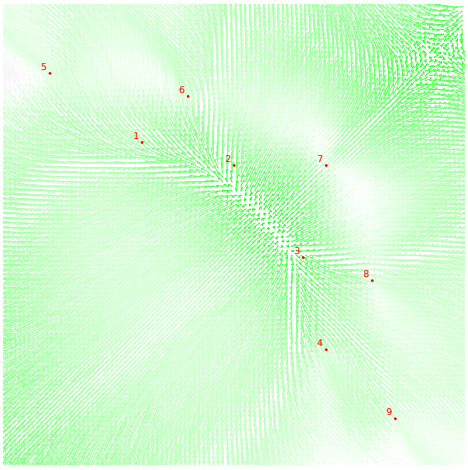
(c) Fehlerverteilung von Geo-n in der *kite* Simulation. Der MAE beträgt 211 %.



(d) Fehlerverteilung von Geo-n in der *tube* Simulation. Der MAE beträgt 89 %.



(e) Inversdarstellung von Geo-n in der *tube* Simulation. Der MAE beträgt 84 %.



(f) Phasendiagramm von Geo-n in der *tube* Simulation.

Abbildung 6.27: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den Geo-n Algorithmus mit ab-nlos Fehlermodell. Der Referenzbezug ist der Erwartungswert des Eingabefehlers aus der vorherigen Simulation.

als der erwartete Distanzfehler der Entfernungsmessungen.

Für die in Abbildung 6.26c und Abbildung 6.26d ausgewerteten Fehlerverteilung der asymmetrischen Szenarien zeigt sich zuerst die prinzipbedingte Abhängigkeit von der Ankeranzahl, die in den weniger symmetrischen Szenarien deutlich ausgeprägter ist. Im *kite* Szenario werden nun kaum noch sehr gute Ergebnisse erzielt und die Performance innerhalb der konvexen Hülle ist außerhalb des guten Bereiches. Unter Hinzunahme fünf weiterer Anker im *quad* Szenario kann der Algorithmus seine wahre Stärke zeigen und erzielt erneut fast durchgehend eine sehr gute Performance.

Überraschend, auch im Vergleich mit den anderen Algorithmen dieser Klasse, ist die Auswertung der Varianz und des Bias der Schätzungen in den Abbildungen 6.26e und 6.26f. Hier zeigt sich, dass der durchschnittliche Positionsfehler fast ausschließlich durch eine hohe Varianz und nicht durch einen Bias in den Schätzungen zustande kommt. Somit lässt sich die Performance des Algorithmus in Szenarien, in denen an die Lokalisierung von nicht mobilen Objekten höhere Anforderungen als an die von bewegten Objekten (beispielsweise Personenüberwachung in Rettungseinsätzen) gestellt werden, noch weiter steigern.

In Abbildung 6.27 ist die räumliche Fehlerverteilung des Geo-n Algorithmus für das *ab-nlos* Fehlermodell visualisiert.

Ähnlich wie schon beim BL Algorithmus beobachtet, fällt die Performance des Geo-n Algorithmus in der in Abbildung 6.27a ausgewerteten *quad* Simulation partiell auffallend stark ab. Auch hier zeigt sich die durch die Ausreißer gestiegene Abhängigkeit der Performance von der Anzahl der Ankerknoten. Obwohl in dem in Abbildung 6.27b gezeigten *grid* Szenario verhältnismäßig mehr Anker Ausreißer in der Entfernungsmessung generieren, kann hier die größere Redundanz besser genutzt werden und die Performance bricht nur unwesentlich ein - der Clusteralgorithmus scheint hier besonders gut zu greifen.

Dieses Beobachtung wird in den beiden asymmetrischen Szenarien, die in den Abbildungen 6.27c und 6.27d gezeigt werden, noch wesentlich deutlicher. So bricht die Performance bei nur minimaler Redundanz in den Ankern erneut stark ein, sobald Ausreißer hinzukommen und in den Szenarien mit größerer Redundanz aufgrund der hohen Ankeranzahl, ist nur ein unwesentlicher Unterschied festzustellen. Besonders im *tube* Szenario bricht die durchschnittliche Performance kaum ein, wenngleich die räumliche Verteilung sich etwas verschlechtert. Trotzdem sind die Ergebnisse der *tube* Simulation in großen Bereichen sehr gut und kommen in den wichtigen Bereichen an

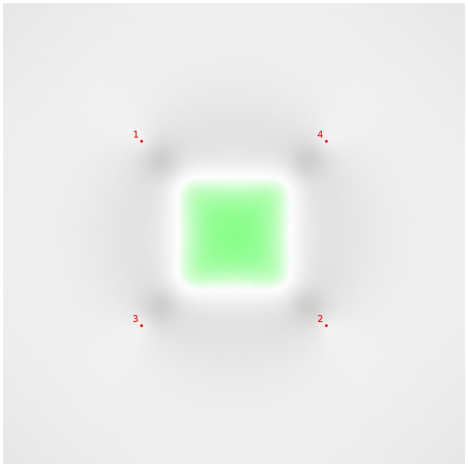
die Performance der Min-Max Algorithmen ran, wobei diese in den übrigen Bereichen, außerhalb der Hülle der Ankerknoten, deutlich übertroffen werden.

Auch mit diesem Fehlermodell liegt der Bias der Schätzungen über die gesamte Fläche sehr niedrig, obgleich der Bias durch die Ausreißer deutlich angestiegen ist. Interessanterweise ist in dem, im Phasendiagramm ausgewerteten, Szenario (siehe Abbildung 6.27e) die Varianz der Schätzungen gegenüber dem selben Szenario mit *nd-noise* Fehlermodell gesunken, was zu dem kaum veränderten Durchschnittsfehler führt. Dieser Effekt dürfte allerdings nur eine zufällige Beobachtung für genau diese Parameter sein. Erhöht man den Erwartungswert der Ausreißer weiter, so stabilisieren sich sowohl Bias und Varianz, da die Ausreißer zuverlässig durch die Clusterphase ausgeschlossen werden und in der weiteren Berechnung keine Rolle mehr spielen. Allerdings lassen sich durch diesen Effekt die oben beschriebenen Eigenschaften für die Lokalisierung von unbewegten Objekten durch den hinzugekommenen Bias nicht mehr im gleichen Ausmaß nutzen. Besonders deutlich wird das in der inversen Darstellung, die in Abbildung 6.27f gezeigt wird.

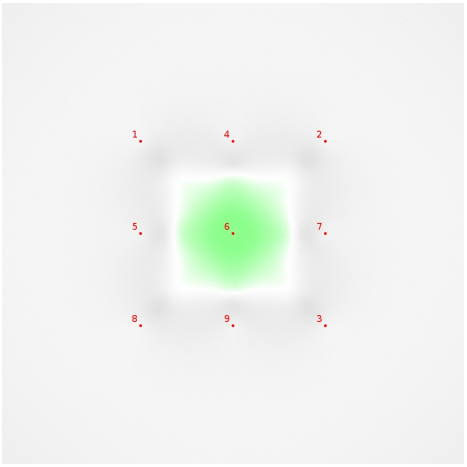
Für die praktische Indoorlokalisierung dürfte dieser Algorithmus für alle bisher diskutierten Szenarien einsetzbar sein. Für Szenarien mit wenig Ankern, in denen die geometrische Konstellation nicht beschränkt ist, dürften ICLA und BL besser geeignet sein. In Szenarien mit mehr Redundanz zeigt dieser Algorithmus die bisher mit Abstand beste Performance, insbesondere wenn keine strenge Beschränkung der geometrischen Verhältnisse aus Ankern und gesuchter Position garantiert werden kann. In Szenarien in denen dieses möglich ist, dürften die beiden E-Min-Max Algorithmen mit diesem Algorithmus in der Performance der Lokalisierung konkurrieren können, während sie ihn in der Rechenzeit unterbieten. Die Rechenzeit von Geo-n ist allerdings auch noch durch Kleinstrechner zu handhaben und wird auch bei hohen Updateraten der Position in der Regel kein Hindernis darstellen.

CluRoL

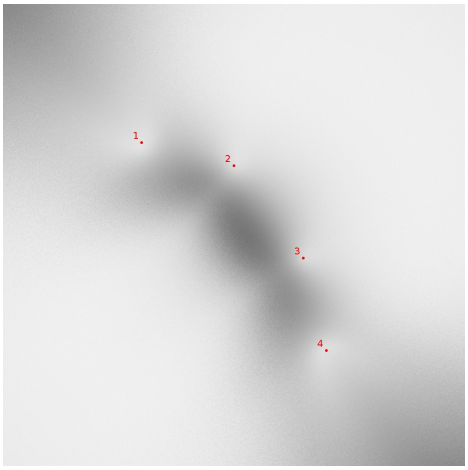
In Abbildung 6.28 ist die räumliche Fehlerverteilung des CluRoL Algorithmus für das Fehlermodell *nd-noise* visualisiert. Dieser Algorithmus ist genau genommen kein eigener Ansatz, sondern versucht anhand von Schnittpunktberechnungen eine optimale Auswahl an Ankern zu treffen und das Lokalisierungsproblem dann mittels der ver-



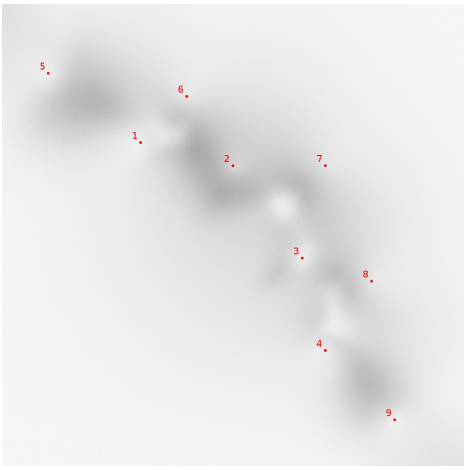
(a) Fehlerverteilung von CluRoL in der *quad* Simulation. Der MAE beträgt 127 %.



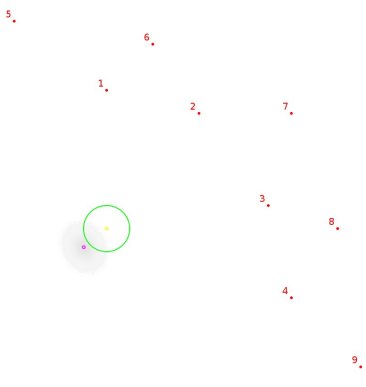
(b) Fehlerverteilung von CluRoL in der *grid* Simulation. Der MAE beträgt 117 %.



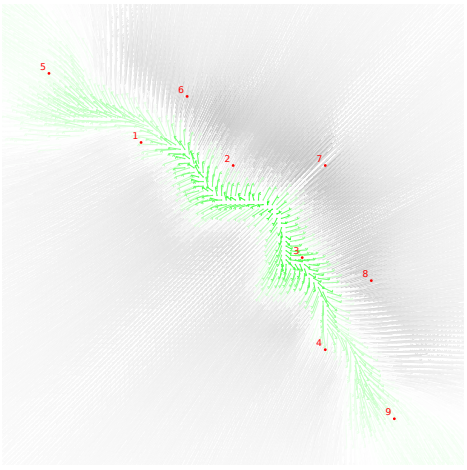
(c) Fehlerverteilung von CluRoL in der *kite* Simulation. Der MAE beträgt 161 %.



(d) Fehlerverteilung von CluRoL in der *tube* Simulation. Der MAE beträgt 138 %.

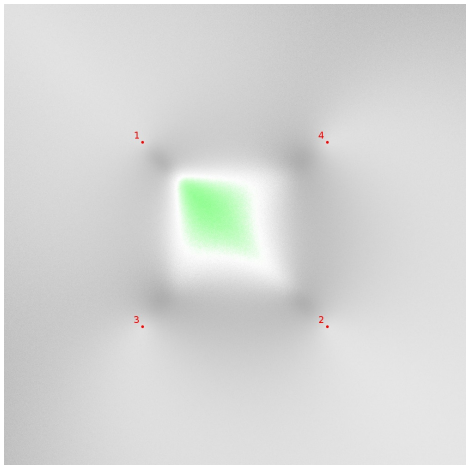


(e) Inversdarstellung von CluRoL in der *tube* Simulation. Der MAE beträgt 128 %.

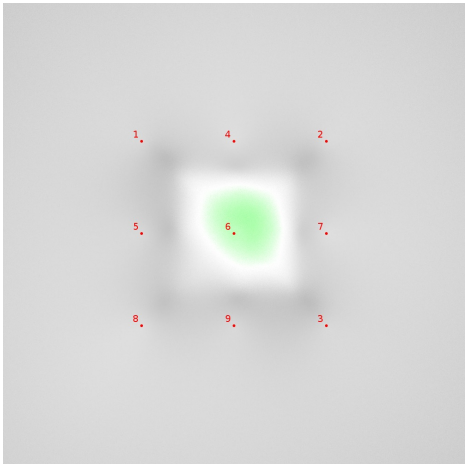


(f) Phasendiagramm von CluRoL in der *tube* Simulation.

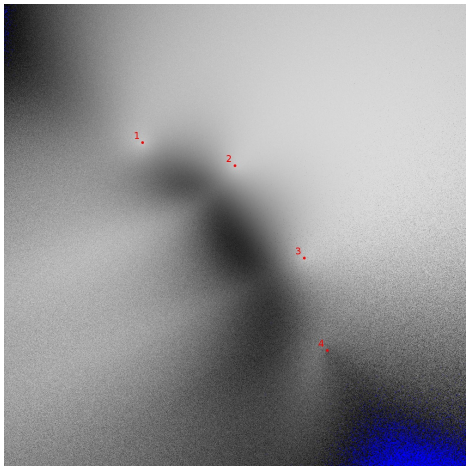
Abbildung 6.28: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den CluRoL Algorithmus mit nd-noise Fehlermodell.



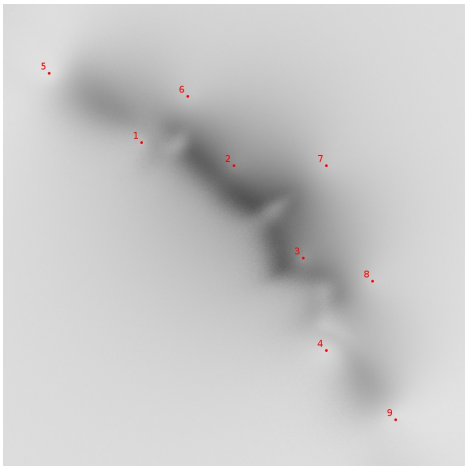
(a) Fehlerverteilung von CluRoL in der *quad* Simulation. Der MAE beträgt 162 %.



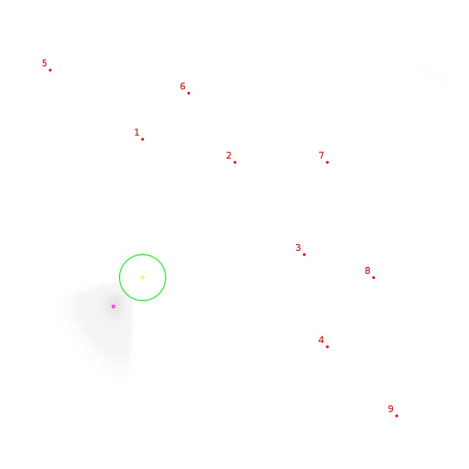
(b) Fehlerverteilung von CluRoL in der *grid* Simulation. Der MAE beträgt 160 %.



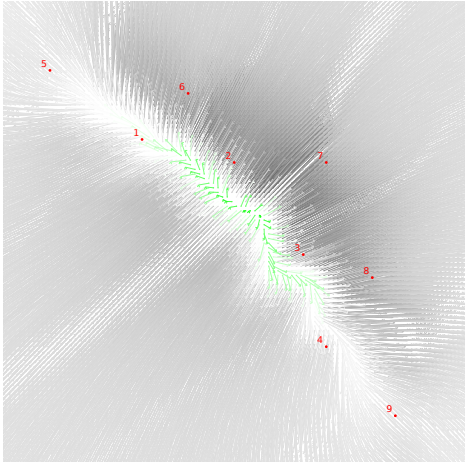
(c) Fehlerverteilung von CluRoL in der *kite* Simulation. Der MAE beträgt 273 %.



(d) Fehlerverteilung von CluRoL in der *tube* Simulation. Der MAE beträgt 186 %.



(e) Inversdarstellung von CluRoL in der *tube* Simulation. Der MAE beträgt 178 %.



(f) Phasendiagramm von CluRoL in der *tube* Simulation.

Abbildung 6.29: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den CluRoL Algorithmus mit ab-nlos Fehlermodell. Der Referenzbezug ist der Erwartungswert des Eingabefehlers aus der vorherigen Simulation.

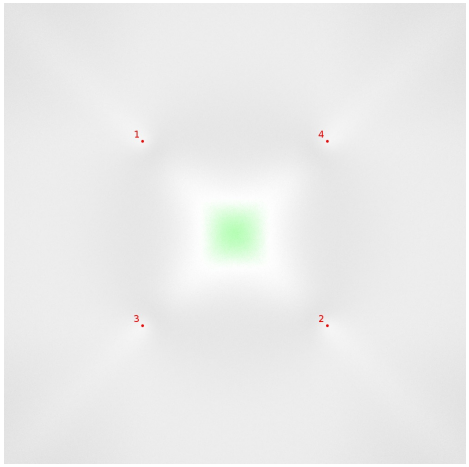
bleibenden Daten über den NLLS Algorithmus zu lösen. Es ist also für das *nd-noise* Fehlermodell keine Verbesserung gegenüber dem NLLS Algorithmus zu erwarten, da dieses Fehlermodell keine wesentlichen Ausreißer produziert. So sind auch folgerichtig keinerlei Unterschiede zwischen CluRoL und NLLS in den vier Simulationen zu erkennen.

Andere Ergebnisse hätten wir für das *ab-nlos* Fehlermodell erwartet, da hier über eine Erkennung und Filterung der NLOS-Anker eventuell eine verbesserte Performance zu erwarten gewesen wäre. Die in Abbildung 6.29 gezeigten Visualisierungen zeigen jedoch, dass auch für dieses Fehlermodell keinerlei Unterschiede zu NLLS zu erkennen sind. Um zu überprüfen, ob dieses Verhalten an den gewählten Fehlermodellen liegt, haben wir weitere Simulationen durchgeführt, in denen wir die NLOS-Anker einfach weggelassen haben. Die Ergebnisse zeigten eine signifikant bessere Performance als die Ergebnisse der hier abgebildeten Simulation. Wir gehen also davon aus, dass die Ankerauswahl zumindest für die hier simulierten Fehlermodelle nicht optimal funktioniert und man etwaige Performanceverbesserungen nur in den Praxisexperimenten wird beurteilen können, wo der NLOS-Fehler evtl. einen größeren Betrag annimmt.

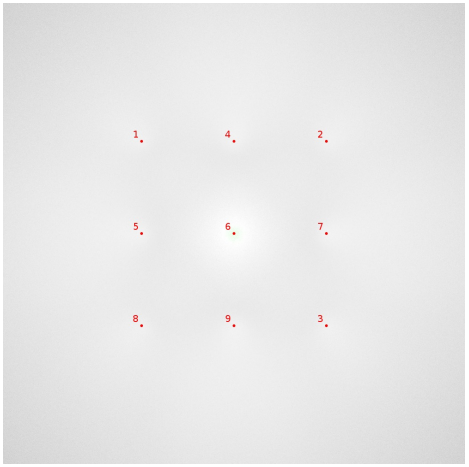
Eine wirkliche Schlussfolgerung kann aufgrund der Ergebnislage nicht gezogen werden. Generell kann jedoch gesagt werden, dass die im Bestfall zu erwartenden Performancesteigerungen in einem Bereich liegen, der verglichen mit anderen Algorithmen, in keinem von uns denkbaren Fall den Einsatz von CluRoL in der Praxis rechtfertigen würde. Lokalisierungen auf Kleinstcomputern dürften mit diesem Algorithmus ab einer gewissen Ankermenge, aufgrund der immensen Rechenzeitanforderungen, nicht möglich sein.

Rwgh

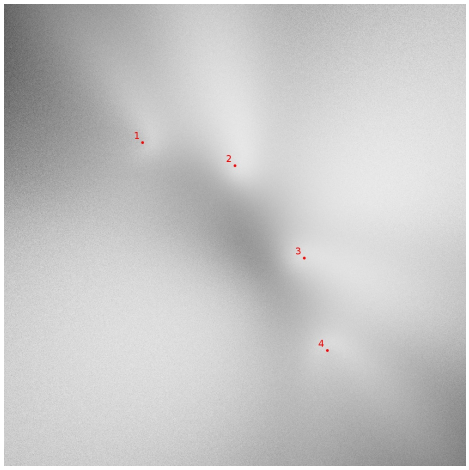
Ähnlich wie der CluRoL Algorithmus versucht auch dieser Algorithmus die Performance von NLLS zu verbessern. Der Algorithmus berechnet aus allen Kombinationen von Ankerknoten eine Position mithilfe von NLLS und gewichtet diese anhand der Restwerte, um die zurückgegebene Position dann mittels Centroid zu schätzen. In Abbildung 6.30 ist die räumliche Fehlerverteilung des Rwgh Algorithmus für das Fehlermodell *nd-noise* visualisiert. Da eine Verbesserung bei diesem Fehlermodell durch das Weglassen von Ankern im Durchschnitt nicht möglich sein wird, sollte die Per-



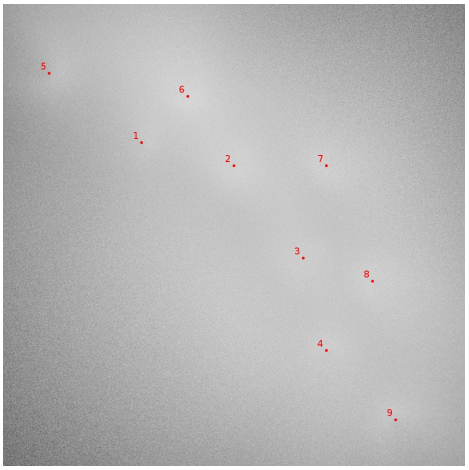
(a) Fehlerverteilung von Rwgh in der *quad* Simulation. Der MAE beträgt 130 %.



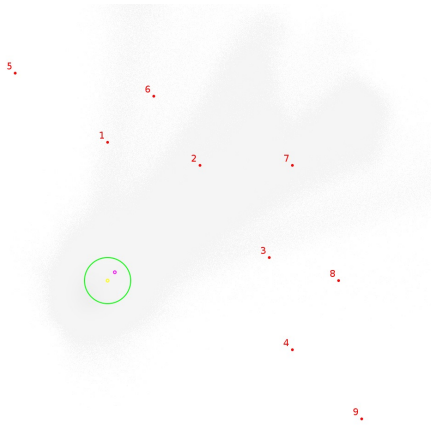
(b) Fehlerverteilung von Rwgh in der *grid* Simulation. Der MAE beträgt 136 %.



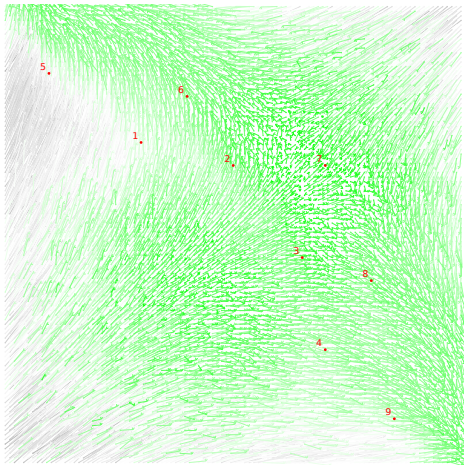
(c) Fehlerverteilung von Rwgh in der *kite* Simulation. Der MAE beträgt 191 %.



(d) Fehlerverteilung von Rwgh in der *tube* Simulation. Der MAE beträgt 214 %.

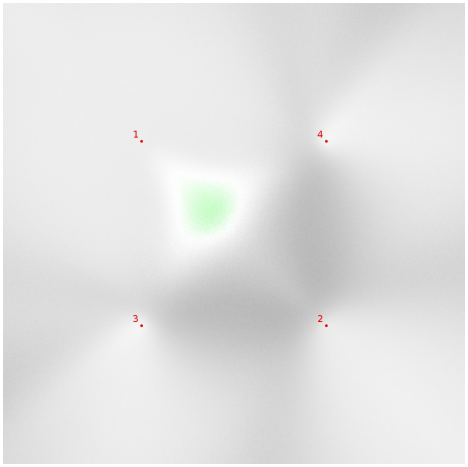


(e) Inversdarstellung von Rwgh in der *tube* Simulation. Der MAE beträgt 48 %.

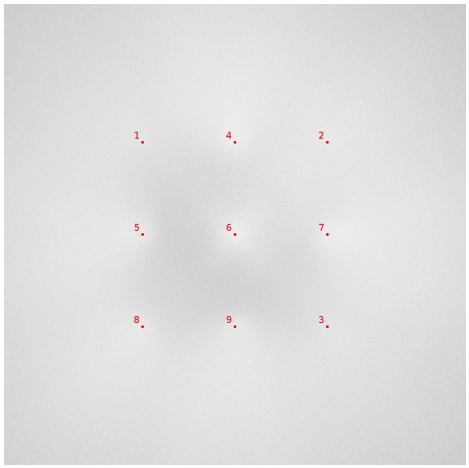


(f) Phasendiagramm von Rwgh in der *tube* Simulation.

Abbildung 6.30: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den Rwgh Algorithmus mit nd-noise Fehlermodell.

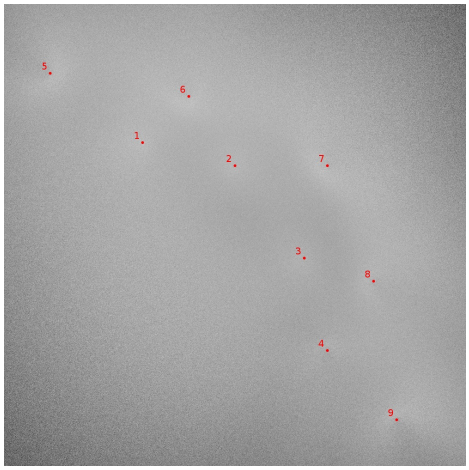


(a) Fehlerverteilung von Rwgh in der *quad* Simulation. Der MAE beträgt 144 %.

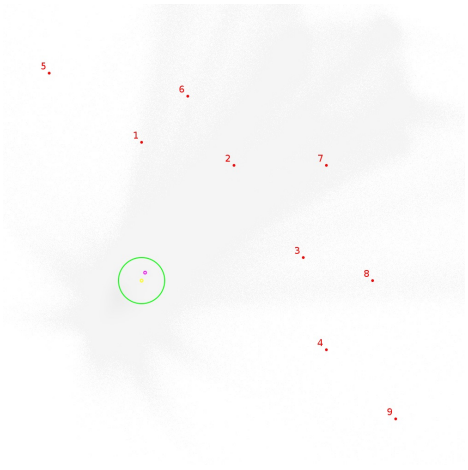


(b) Fehlerverteilung von Rwgh in der *grid* Simulation. Der MAE beträgt 150 %.

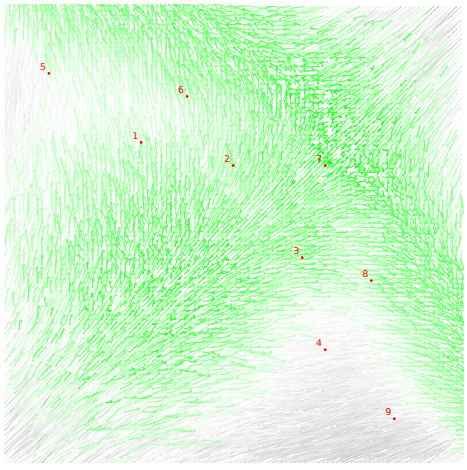
(c) Fehlerverteilung von Rwgh in der *kite* Simulation. Der MAE beträgt 197 %.



(d) Fehlerverteilung von Rwgh in der *tube* Simulation. Der MAE beträgt 244 %.



(e) Inversdarstellung von Rwgh in der *tube* Simulation. Der MAE beträgt 38 %.



(f) Phasendiagramm von Rwgh in der *tube* Simulation.

Abbildung 6.31: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den Rwgh Algorithmus mit ab-nlos Fehlermodell. Der Referenzbezug ist der Erwartungswert des Eingabefehlers aus der vorherigen Simulation.

formance des Algorithmus nahe bei der des NLLS Algorithmus liegen. Wie in den Abbildungen 6.30a und 6.30c zu sehen ist, gelingt das für die Setups mit nur wenigen Ankern recht gut. Der leichte Performanceabfall gibt allerdings einen Hinweis darauf, dass in der Mehrzahl der Fälle genau ein Anker aus dem Setup entfernt wurde - da die Minimalzahl von drei Ankern nicht unterschritten werden darf.

Dieser Eindruck wird in den Abbildungen 6.30b und 6.30d dann auch bestätigt. Bei neun Ankern scheint die Auswahlkomponente deutlich mehr Anker auszuschließen und die Performance fällt sehr stark ab. Da aber in diesem Fehlermodell im Erwartungswert keine großen Ausreißer vorkommen, lassen sich hieraus noch keine Rückschlüsse für die Praxis ziehen, in der immer mit solchen Ausreißern zu rechnen ist. In den beiden Abbildungen 6.30e und 6.30f sieht man dann auch sehr deutlich die Arbeitsweise dieses Algorithmus. Während der Bias der Schätzungen wesentlich reduziert wird, wird die Varianz sehr stark vergrößert. Die Schätzungen für die Beispielpositionen belegen die komplette Breite der Simulationsfläche und eine Lokalisierung ist nur für unbewegte Objekte denkbar.

In Abbildung 6.31 ist die räumliche Fehlerverteilung des Rwgh Algorithmus für das Fehlermodell *ab-nlos* visualisiert. Die Auswertungen der vier Simulationsdurchläufe hinterlassen einen geteilten Eindruck. In den Szenarien mit nur wenigen Ankern werden zum Teil beachtliche Performancegewinne erzielt und weitere Probesimulationen zeigen, dass zuverlässig der Anker, der für die Ausreißer verantwortlich ist, gefiltert wird. Die Ergebnisse für diese Szenarien finden sich in den Abbildungen 6.31a und 6.31c.

Für die Szenarien mit mehr Ankern offenbart sich eine Schwäche dieses Algorithmus: Bringt das Filtern der Anker mit Ausreißern im symmetrischen *grid* Setup noch einen kleinen Performancevorteil (siehe Abbildung 6.31b), so schlägt das Filtern dieser Anker im *tube* Szenario ins Gegenteil um (siehe Abbildung 6.31d) und die Performance bricht stark ein. Dieses Verhalten ist eine generelle Schwäche des Ansatzes: Selbst wenn Entfernungsmessungen mit großen Fehlern zuverlässig erkannt werden, macht es nicht immer Sinn, diese unberücksichtigt zu lassen. Werden die drei Anker im *tube* Szenario entfernt, so ist die verbleibende Geometrie der Ankerknoten so ungünstig, dass auch ohne Ausreißer eine schlechtere Performance erzielt wird.

Bias und Varianz haben sich durch das Hinzukommen der Ausreißer kaum verändert. Wie in den Abbildungen 6.31e und 6.31f gezeigt wird, wurde der Bias erfolgreich minimiert, die Varianz ist allerdings weiter sehr groß.

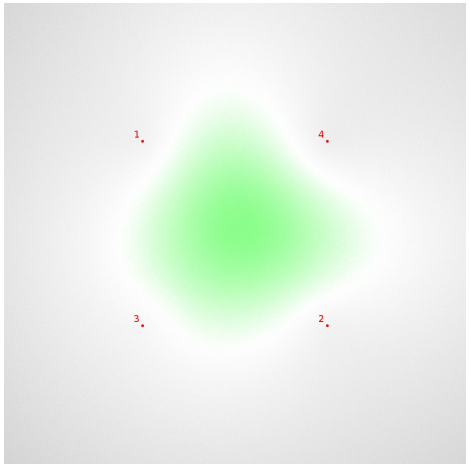
Auch dieser Algorithmus hat eher einen theoretischen Nutzen. Wie für den zuvor diskutierten Algorithmus gilt auch hier, dass selbst beim perfekten Gewichten von Ausreißern der Ansatz nur in wenigen Fällen mit anderen Algorithmen mithalten dürfte, da die Grundperformance von NLLS in der Praxis einfach zu niedrig ist. Auch hier ist der Rechenaufwand sehr hoch, die erzielte Performance ist zu durchwachsen. Insbesondere die hohe Varianz der Lokalisierungen macht für bewegte Objekte starke Glättungsfilter erforderlich. Nur für die Lokalisierung von nicht mobilen Objekten erzielt dieser Algorithmus unter Umständen eine gute Performance, die allerdings mit einer hohen Rechenzeit einhergehen würde.

LMS

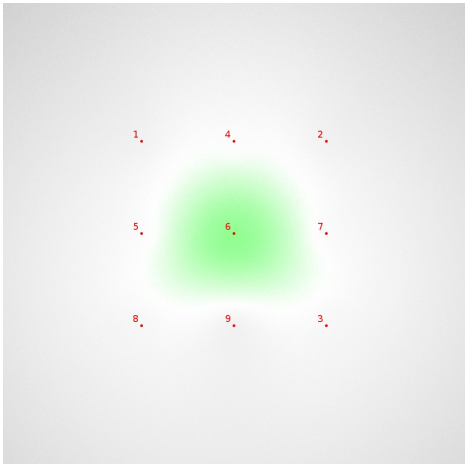
In Abbildung 6.32 ist die räumliche Fehlerverteilung des LMS Algorithmus für das Fehlermodell *nd-noise* visualisiert. Ähnlich wie der Rwgh und der CluRoL Algorithmus versucht auch dieser Algorithmus, über eine Ankerauswahl einen bestehenden Algorithmus zu verbessern. In diesem Fall wird für die endgültige Lokalisierung der LLS Ansatz benutzt. Für das *nd-noise* Fehlermodell fällt auf, dass die Ankerauswahl deutlich besser als bei den beiden ersten Vertretern dieser Klasse funktioniert. Eine Verschlechterung findet so gut wie nirgends statt und die Ergebnisse entsprechen denen von LLS.

In Abbildung 6.33 ist die räumliche Fehlerverteilung des LMS Algorithmus für das Fehlermodell *ab-nlos* visualisiert. Für die Setups mit großer Ankeranzahl, die in den Abbildungen 6.33b und 6.33d gezeigt werden, erzielt der Algorithmus eine leichte Verbesserung der Performance gegenüber LLS. Das Filtern von Ausreißern scheint generell zuverlässig zu funktionieren und auch einen leicht positiven Effekt zu haben. Die anderen beiden Szenarien profitieren hingegen nicht von der Ankerauswahl und die Ergebnisse sind die des LLS Algorithmus über alle Anker.

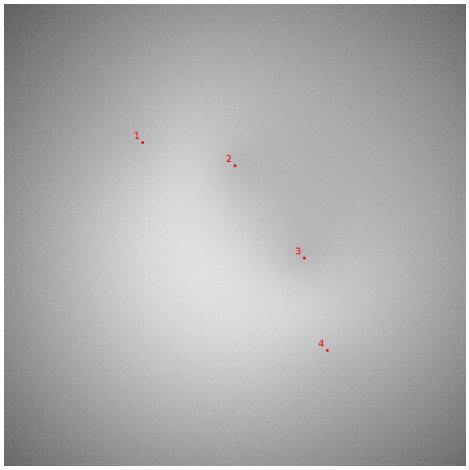
Für die Praxis sehen wir auch für diesen Algorithmus kein wirkliches Einsatzfeld. Es wurde eine leichte Verbesserung für den Algorithmus erzielt, der in der Gesamtbetrachtung der Performance unter allen Gesichtspunkten sehr schlecht abschneidet und die Rechenzeit wurde zusätzlich noch wesentlich erhöht. Allerdings könnte es ein Ansatz für weitere Untersuchungen sein, die Ankerauswahlkomponente von LMS mit anderen Algorithmen als mit LLS zu kombinieren. Auf Untersuchungen, ob der



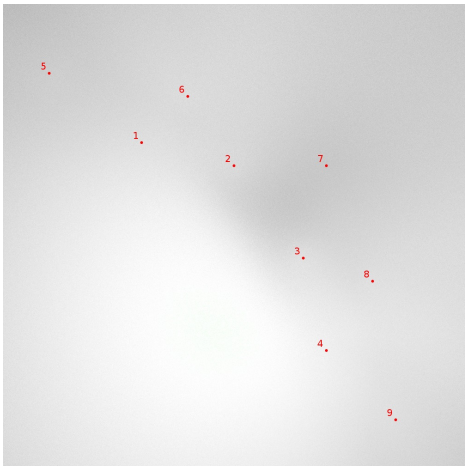
(a) Fehlerverteilung von LMS in der *quad* Simulation. Der MAE beträgt 119 %.



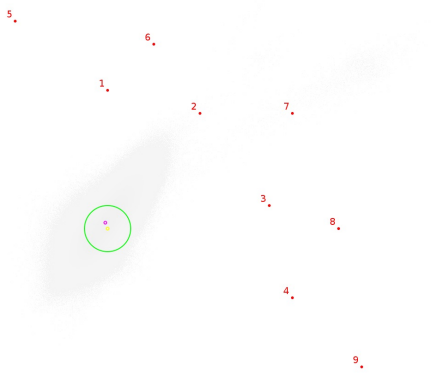
(b) Fehlerverteilung von LMS in der *grid* Simulation. Der MAE beträgt 123 %.



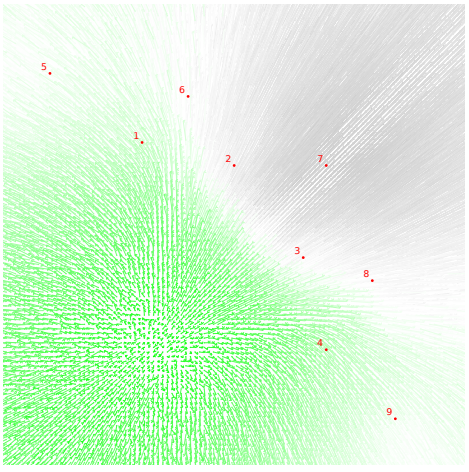
(c) Fehlerverteilung von LMS in der *kite* Simulation. Der MAE beträgt 232 %.



(d) Fehlerverteilung von LMS in der *tube* Simulation. Der MAE beträgt 148 %.

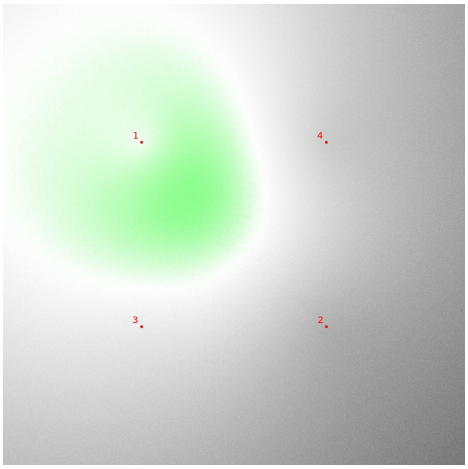


(e) Inversdarstellung von LMS in der *tube* Simulation. Der MAE beträgt 27 %.

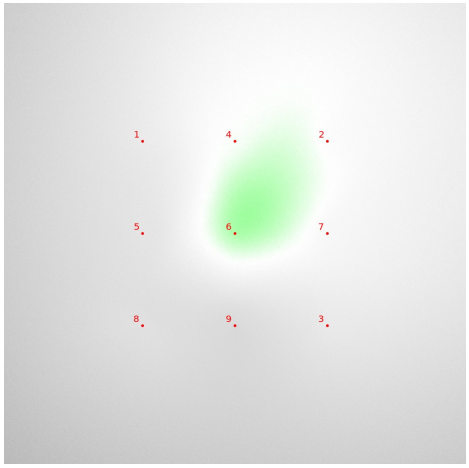


(f) Phasendiagramm von LMS in der *tube* Simulation.

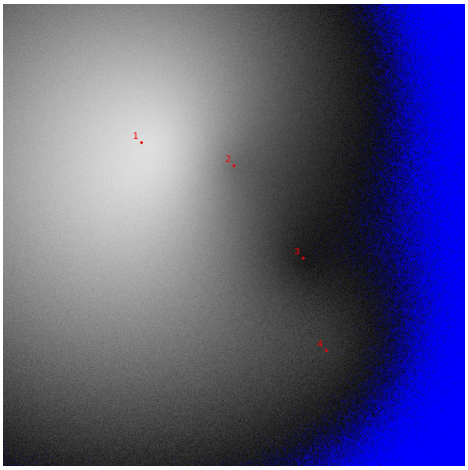
Abbildung 6.32: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den LMS Algorithmus mit nd-noise Fehlermodell.



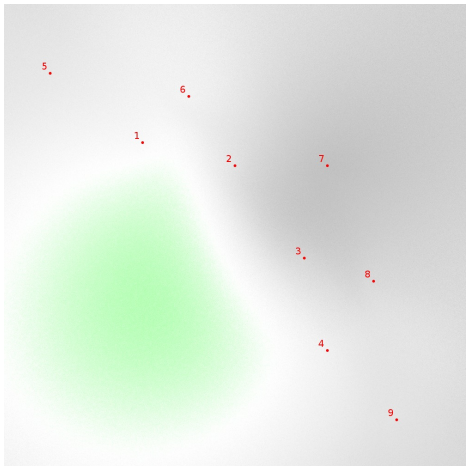
(a) Fehlerverteilung von LMS in der *quad* Simulation. Der MAE beträgt 158 %.



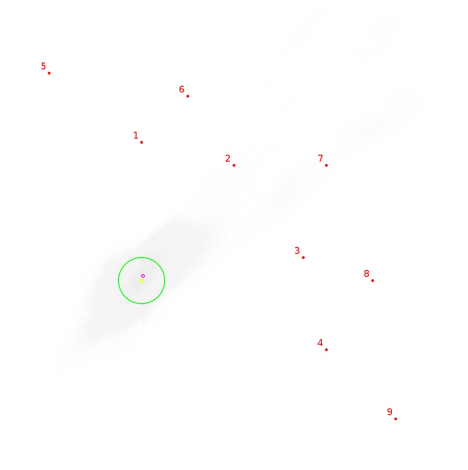
(b) Fehlerverteilung von LMS in der *grid* Simulation. Der MAE beträgt 138 %.



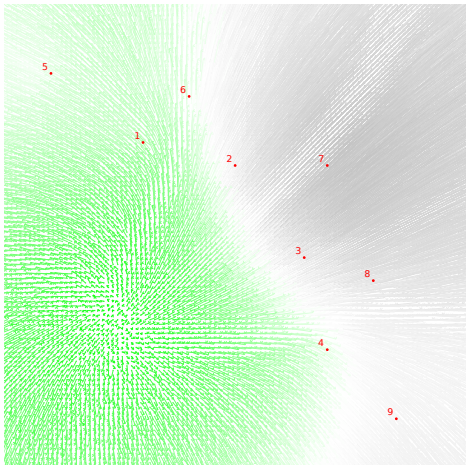
(c) Fehlerverteilung von LMS in der *kite* Simulation. Der MAE beträgt 371 %.



(d) Fehlerverteilung von LMS in der *tube* Simulation. Der MAE beträgt 129 %.



(e) Inversdarstellung von LMS in der *tube* Simulation. Der MAE beträgt 21 %.



(f) Phasendiagramm von LMS in der *tube* Simulation.

Abbildung 6.33: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den LMS Algorithmus mit ab-nlos Fehlermodell. Der Referenzbezug ist der Erwartungswert des Eingabefehlers aus der vorherigen Simulation.

Ansatz für Szenarien mit noch größeren Ausreißern eine wesentliche Performancesteigerung bringen würde, haben wir verzichtet, da die Grundperformance von LLS nicht ausreichen wird, um zu einer wesentlich anderen Einschätzung zu kommen.

RLSM

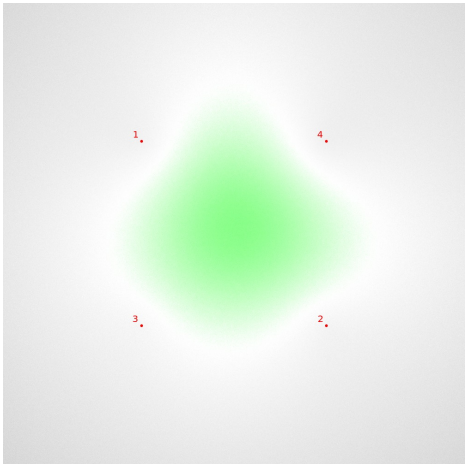
In Abbildung 6.34 ist die räumliche Fehlerverteilung des RLSM Algorithmus für das Fehlermodell *nd-noise* visualisiert. Dieser Algorithmus versucht aus allen Permutationen von Ankerknoten, eine Optimale zu finden und das Problem der Lokalisierung dann mittels *Point Geometric Median* zu lösen. Aufgrund der exponentiell mit der Ankerzahl steigenden Laufzeit dieses Algorithmus war es uns nicht möglich, die Simulationen mit mehr als sieben Ankerknoten durchzuführen. Die Laufzeit der Simulation wäre ansonsten unverhältnismäßig in die Höhe gestiegen. Die Ergebnisse der Simulationen mit sieben Ankern sind also nicht direkt mit den Ergebnissen der anderen Algorithmen zu vergleichen.

Die Ergebnisse für die beiden symmetrischeren Fälle, die in den Abbildungen 6.34a und 6.34b gezeigt werden, erzielen durchweg gute bis sehr gute Performancewerte. Die sehr gute Performance wird zudem in der besonders wichtigen Region innerhalb der konvexen Hülle erzielt. Allerdings reichen die Ergebnisse dieser Szenarien nicht an die besten der bisher besprochenen Algorithmen heran.

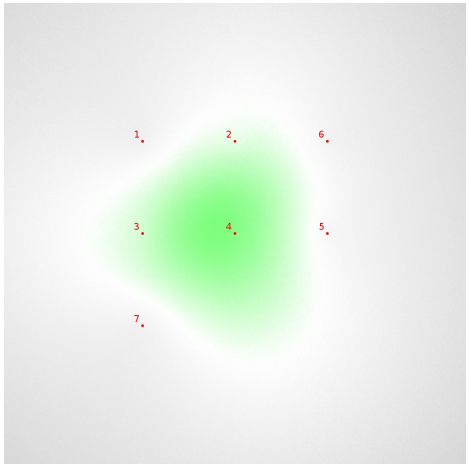
Im asymmetrischen *kite* Szenario, welches in Abbildung 6.34c gezeigt wird, bricht die Performance sehr stark ein und eine zuverlässige Lokalisierung ist kaum möglich. Der Fehler ist dabei allerdings auffallend homogen über die Fläche verteilt. Im ebenfalls asymmetrischen *tube* Szenario, dessen Ergebnisse in Abbildung 6.34d zu sehen sind, fällt diese Homogenität ebenfalls auf. Hier erreichen die Ergebnisse insgesamt aber ein gutes Niveau.

Bei der Analyse der Verteilung der Schätzungen, die in den Abbildungen 6.34e und 6.34f visualisiert sind, fällt vor allem die große Streuung der über große Flächen erwartungstreuen Schätzungen auf. Für das zuverlässige Lokalisieren bewegter Objekte ist diese Streuung für die meisten Anforderungsszenarien zu groß.

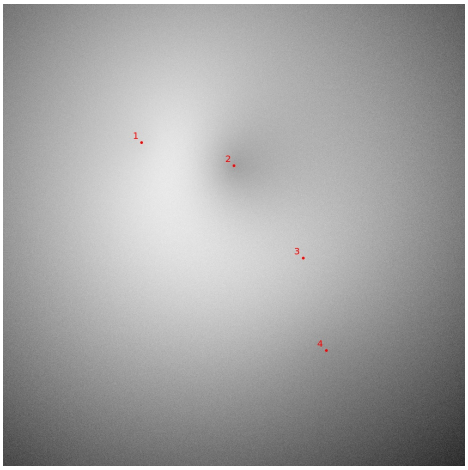
In Abbildung 6.35 ist die räumliche Fehlerverteilung des RLSM Algorithmus für das Fehlermodell *ab-nlos* visualisiert. Kommen zu den normalverteilten Fehlern in der Distanzmessung noch größere Ausreißer hinzu, bricht die Performance dieses Algo-



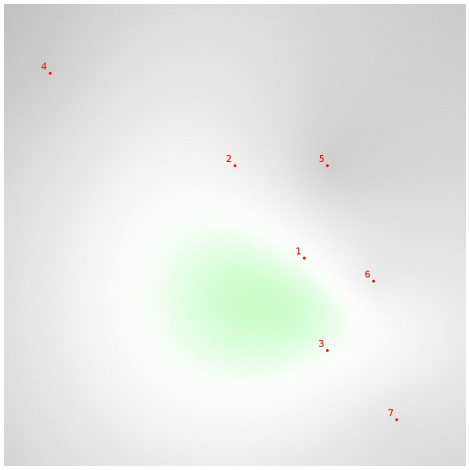
(a) Fehlerverteilung von RLSM in der *quad* Simulation. Der MAE beträgt 117 %.



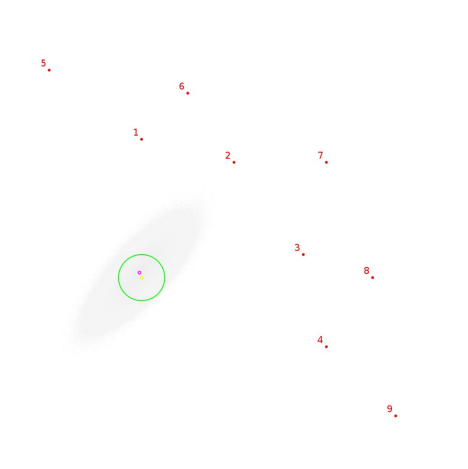
(b) Fehlerverteilung von RLSM in der *grid* Simulation. Der MAE beträgt 119 %.



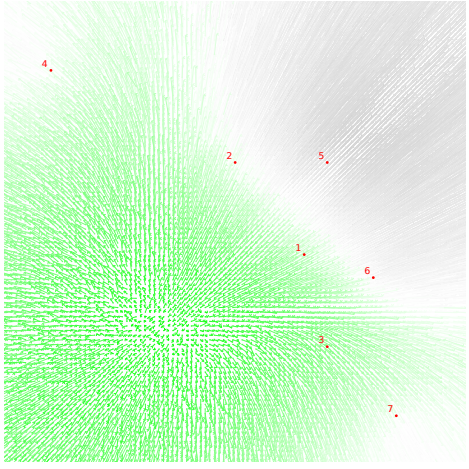
(c) Fehlerverteilung von RLSM in der *kite* Simulation. Der MAE beträgt 247 %.



(d) Fehlerverteilung von RLSM in der *tube* Simulation. Der MAE beträgt 134 %.

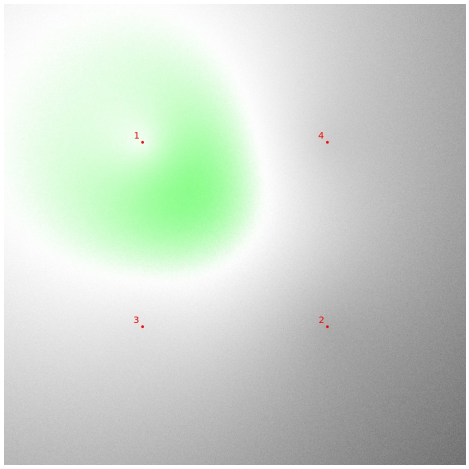


(e) Inversdarstellung von RLSM in der *tube* Simulation. Der MAE beträgt 23 %.

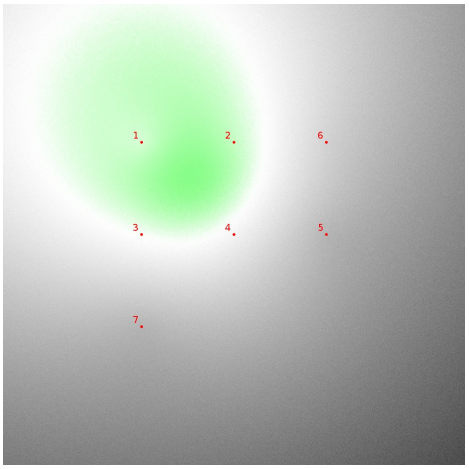


(f) Phasendiagramm von RLSM in der *tube* Simulation.

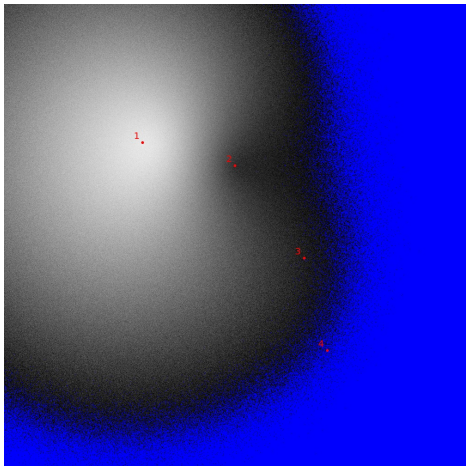
Abbildung 6.34: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den RLSM Algorithmus mit nd-noise Fehlermodell.



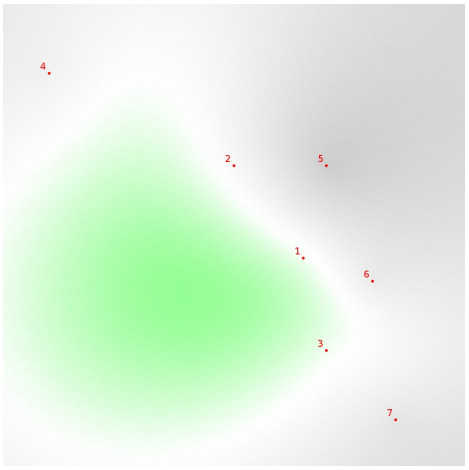
(a) Fehlerverteilung von RLSM in der *quad* Simulation. Der MAE beträgt 162 %.



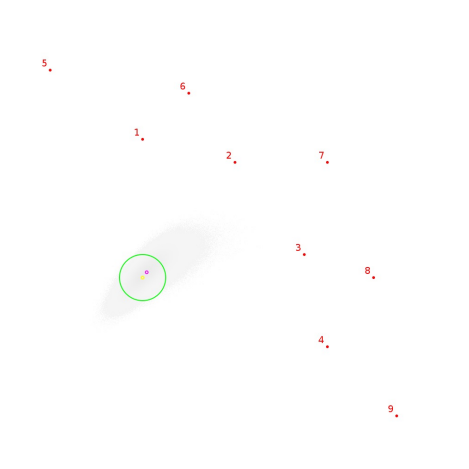
(b) Fehlerverteilung von RLSM in der *grid* Simulation. Der MAE beträgt 192 %.



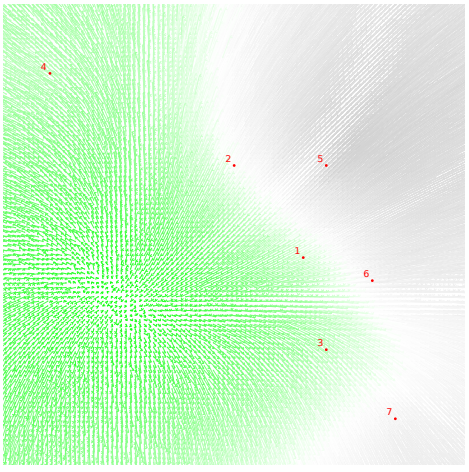
(c) Fehlerverteilung von RLSM in der *kite* Simulation. Der MAE beträgt 438 %.



(d) Fehlerverteilung von RLSM in der *tube* Simulation. Der MAE beträgt 111 %.



(e) Inversdarstellung von RLSM in der *tube* Simulation. Der MAE beträgt 29 %.



(f) Phasendiagramm von RLSM in der *tube* Simulation.

Abbildung 6.35: Räumliche Verteilung des Distanzmessfehlers für verschiedenen Szenarien für den RLSM Algorithmus mit ab-nlos Fehlermodell. Der Referenzbezug ist der Erwartungswert des Eingabefehlers aus der vorherigen Simulation.

rithmus in fast allen Szenarien sehr stark ein und ein Einsatz in der Praxis erscheint nur schwer vorstellbar. Lediglich im *tube* Szenario, dessen Ergebnisse in Abbildung 6.35d präsentiert werden, findet sogar eine Steigerung der Performance statt. Allerdings dürfte dieser Effekt auf spezifische geometrische Besonderheiten der simulierten Konstellation zurückzuführen sein, die auch schon bei anderen Algorithmen beobachtet und diskutiert wurden. Eine Ausnutzung dieser Effekte in der Praxis dürfte auch hier nicht möglich sein. Trotzdem ist die Performance in diesem Szenario durchweg gut bis sehr gut.

Für die Analyse der inversen Darstellung und des Phasendiagramms gilt im Wesentlichen das schon für das *nd-noise* Fehlermodell beobachtete.

Zusammenfassend darf auch die Eignung dieses Algorithmus, der wie die Vorherigen im Wesentlichen auf einer Ankerauswahl beruht, für praktische Einsätze stark bezweifelt werden. Insbesondere steht hier erneut das sehr ungünstige Verhältnis zwischen erzielter Performance und der sehr langen Laufzeit, einer positiven Bewertung im Wege. Gerade in größeren Szenarien, wo die Anzahl der gleichzeitig erreichbaren Ankerknoten nicht schon während der Ausbringung der Ankerknoten genau bestimmt werden kann, dürfte dieser Algorithmus auch für relativ rechenstarke Computer kaum in Frage kommen.

In Tabelle 6.1 sind die Laufzeiten der ausgewerteten Algorithmen dargestellt. Für RLSM sind in den Szenarien *grid* und *tube* allerdings nur sieben Ankerknoten, anstatt neun, simuliert worden. Daher ist das Ergebnis für diese beiden Fälle nicht mit den anderen Ergebnissen zu vergleichen.

In Tabelle 6.2 ist der MAE der ausgewerteten Algorithmen für das jeweilige Fehlermodell dargestellt.

6.3 Fazit

Wie in den Ergebnissen ausführlich beschrieben, ergibt sich aus der bloßen Auswertung der statistischen Fehlerverteilung (siehe Tabelle 6.2) ein völlig anderes Bild, als aus der Auswertung der räumlichen Fehlerverteilung bezüglich der Praxiseignung des jeweiligen Algorithmus. Die Analyse der räumlichen Fehlerverteilung hat für den allgemeinen Fall der Indoorlokalisierung ergeben, dass sich die verschiedenen Min-Max

Tabelle 6.1: LS^2 Laufzeiten

Algorithmus	*kite*	*quad*	*tube*	*grid*
AML	1369 s	1258 s	1333 s	3140 s
BL	479 s	486 s	8340 s	8111 s
E-Min-Max (W2)	144 s	134 s	322 s	309 s
E-Min-Max (W4)	147 s	136 s	326 s	313 s
Geo-n	1512 s	1522 s	7417 s	8076 s
LLS	138 s	138 s	313 s	303 s
LMS	422 s	419 s	26771 s	4795 s
MD-Min-Max	167 s	167 s	365 s	351 s
Min-Max	125 s	124 s	286 s	272 s
MLE-$\mathcal{N}$	10503 s	9694 s	10425 s	10425 s
NLLS	419 s	415 s	844 s	837 s
VBLE	16997 s	15436 s	33600 s	34042 s
VBLE-O	10516 s	6164 s	11820 s	12465 s
ICLA	11117 s	13025 s	348054 s	333060 s
CluRoL	21796 s	21786 s	802777 s	692874 s
Rwgh	3764 s	3712 s	165353 s	225813 s
RLSM	2649 s	2873 s	255971 s	237169 s

Laufzeiten von jeweils 320000000 Durchläufen des jeweiligen Algorithmus im jeweiligen Szenario.

Tabelle 6.2: LS^2 Durchschnittsfehler

Algorithmus	*nd-noise*				*ab-nlos*			
	kite	*quad*	*tube*	*grid*	*kite*	*quad*	*tube*	*grid*
AML	201	141	238	180	284	181	134	119
BL	152	104	87	68	197	125	113	77
E-Min-Max (W2)	195	115	80	86	271	129	96	88
E-Min-Max (W4)	315	131	141	129	378	140	135	150
Geo-n	162	104	80	77	211	126	89	87
LLS	231	119	145	119	371	159	135	169
LMS	232	119	148	123	185	69	129	79
MD-Min-Max	453	316	212	230	503	330	194	266
Min-Max	498	365	368	361	504	369	360	373
MLE-$\mathcal{N}$	80	54	37	39	149	72	62	69
NLLS	161	127	138	117	273	162	186	160
VBLE	220	114	119	107	261	147	169	171
VBLE-O	122	69	78	63	174	89	110	103
ICLA	167	123	105	98	229	160	144	116
CluRoL	161	127	138	117	273	162	186	160
Rwgh	191	130	214	136	197	144	244	150
RLSM	119	117	134	119	192	162	111	192

MAE von jeweils 320000000 Durchläufen des jeweiligen Algorithmus im jeweiligen Szenario.

Derivate, VBLE-O und der Geo-n Algorithmus besonders gut für dieses Einsatzgebiet eignen. Legt man für die Ausbringung der Ankerknoten keine klassischen Gebäudestrukturen zu Grunde und verteilt sie quasi zufällig im Gebäude bzw. besteht durch die Ankerausbringung keine deutlich erhöhte Wahrscheinlichkeit, dass sich der zu lokalisierende Knoten innerhalb der konvexen Hülle der Ankerknoten aufhält, so bleiben nur noch VBLE-O und Geo-n übrig.

Besonders deutlich ist das Ergebnis auch für die vier Algorithmen, die durch eine Vorauswahl der Anker nur eine Teilmenge der Messdaten in den eigentlichen Algorithmus geben: Diese Algorithmen können in Bezug auf die Lokalisierungsperformance mit den Algorithmen, die alle Messwerte verarbeiten, in keinster Weise mithalten und benötigen mit Ausnahme von LMS zusätzlich noch erhebliche Rechenkapazitäten. Wie die Analyse ergeben hat, kann, je nach geometrischer Konstellation, das Filtern eines Ankers mit großen Ausreißern schädlicher sein, als das Einbeziehen dieser stark fehlerbehafteten Werte. Das große Problem dieser Algorithmen ist es, dass sie beim Filtern nur die Messwerte an sich berücksichtigen, nicht jedoch die Geometrie der Ankerknoten.

Eine zweite Erkenntnis aus der Analyse der räumlichen Fehlerverteilung ist es, dass die geometrische Konstellation der Ankerknoten und die Position des unbekannten Knotens einen wesentlich höheren Einfluss haben, als der Messfehler der Distanzen. Dieses hat erhebliche Konsequenzen für die Ausbringung von Systemen zur Indoorlokalisierung in der Praxis. Eine gut geplante Positionierung der Anker in Bezug auf die zu erwartenden Positionen der zu lokalisierenden Objekte kann die Präzision des Gesamtsystems stärker beeinflussen, als die Auswahl und damit der Preis der Hardware. Trotzdem lassen sich die Ergebnisse dieser Simulationen nicht generell auf andere Szenarien übertragen, sondern maximal Prognosen erstellen. Zu groß sind die Unterschiede zwischen Simulation und Praxis: Das Auftreten von NLOS-Fehlern ist nicht zufällig, sondern gebäudeabhängig. Durch das Öffnen oder Schließen von Türen sowie durch das Bewegen von Personen im Gebäude können sich diese Eigenschaften verändern und durch mangelnde Kalibrierung der Hardware können Entfernungsmessungen auftreten, die kürzer als die direkte Entfernung sind.

Alle diese Effekte machen es notwendig, dass für eine umfassende Bewertung der Algorithmen Experimente unter möglichst realistischen Bedingungen folgen müssen. Aber auch hier gilt natürlich, dass eine Übertragbarkeit solcher Ergebnisse nur im begrenzten Rahmen möglich ist, da eine bloße Neupositionierung der Ankerknoten wiederum

völlig andere Ergebnisse liefern könnte. Zusammen mit der ausführlichen Analyse der räumlichen Verteilung sollten die folgenden Experimente aber eine ausreichend große Datenbasis bilden, um die Bewertung der hier vorgestellten Algorithmen wesentlich zu objektivieren und eine Auswahlentscheidung eines bestimmten Algorithmus für einen bestimmten Einsatzzweck wesentlich zu vereinfachen.

KAPITEL 7

Evaluation und Vergleich der Algorithmen anhand von Experimenten

In diesem Kapitel werden alle durchgeführten Experimente vorgestellt und ausgewertet. Hierfür wurde das in Abschnitt 5.3 vorgestellte virtuelle Testbed unter Verwendung der Sensorknoten aus Abschnitt 7.1 benutzt. Dieses speichert alle aufgenommenen Messdaten zur allgemeinen Verfügbarkeit in einer Datenbank ab, so dass eine bequeme Auswertung und Optimierung der Algorithmen auch im Nachhinein möglich ist. Bevor wir die einzelnen Testszenarien genauer vorstellen und auswerten, soll zuerst die für die Distanzmessung eingesetzte Hardware vorgestellt und deren Genauigkeit ausführlich untersucht werden.

7.1 Experimenthardware

Für die Experimente wurde eine modifizierte Version des *Modular Sensor Board* (MSB) A2 [4] eingesetzt. Abbildung 7.1 zeigt einen solchen Sensorknoten, der während des FeuerWhere Projekts an der Freien Universität Berlin entwickelt wurde [150] und als *Personal Tracking and Transmission Unit* (PTTU) bezeichnet wird. Jede PTTU ist mit einem Nanotron Technologies nanoPAN 5375 [95] Transceiver ausgestattet. Dieser ermöglicht den Sensorknoten die Abstände zu benachbarten Knoten mittels TOF im 2,4 GHz Frequenzband zu bestimmen. Das eingesetzte Protokoll ist standardisiert unter ISO/IEC 24730-5:2010 und verwendet als Modulationsverfahren ein

Abbildung 7.1: Eine PTTU mit Gehäuse

Chirp Spread Spectrum (CSS) mit einer Bandbreite von 80 MHz. Dies soll robust gegen schmal- und breitbandige Signalstörungen sowie gegen Mehrwegeempfang sein. Die exakte Methode, die zur Ermittlung einer Distanz eingesetzt wird, nennt sich *Symmetrical Double-Sided Two Way Ranging* (SDS-TWR). Bei diesem Verfahren ermittelt jeder der beteiligten Knoten mittels Umlaufzeitmessung eine Distanz zum jeweiligen anderen Knoten, die finale Distanz ergibt sich durch Durchschnittsbildung beider Messungen. Diese Methode ist einfach in der Implementierung, erhöht aber die nötige Signalverarbeitung, da ein Signal von Knoten A zu Knoten B und zurück gesendet werden muss und ebenso ein Signal von Knoten B zu Knoten A und wieder zurück zu B. Für diesen Vorgang werden minimal 5 Pakete benötigt, falls nur der initiierende Knoten eine Distanz berechnen soll. Dafür ist aufgrund der beidseitigen Auswertung keine Synchronisation der Uhren nötig und es können kostengünstige Uhrenquarze verwendet werden (vgl. auch Abschnitt 2.3.2). Unsere Messungen haben ergeben, dass eine Distanzbestimmung zwischen zwei Knoten im Schnitt 2,98 Millisekunden benötigt, also 335 Distanzmessungen pro Sekunde im besten Fall möglich sind. Ist einer der beteiligten Knoten außer Reichweite, erhöht sich die Dauer für

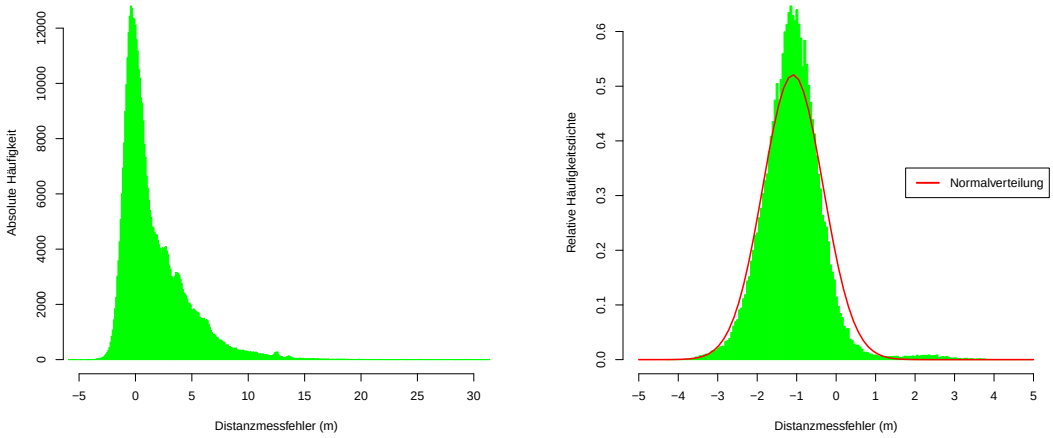
Tabelle 7.1: Genauigkeit der Distanzmessung

Metrik	Indoor-Wert	Outdoor-Wert
Durchschnittsfehler*	2,12 m	1,17 m
Varianz	8,62 m^2	0,59 m^2
MSE	11,18 m^2	1,78 m^2
RMSE	3,34 m	1,33 m
Maximalfehler*	74,18 m	4,37 m
0,5 % Quantil	−2,18 m	−3,08 m
Modus	−0,38 m	−1,08 m
99,5 % Quantil	13,50 m	2,35 m

* absoluter Fehlerwert

eine Messung auf durchschnittlich 10,16 Millisekunden, im schlechtesten Fall auf 13 Millisekunden.

Zu Beginn jeder Datenerhebung für das virtuelle Testbed wird dem mobilen Knoten, der mittels Roboter einen festgelegten Pfad abfährt, eine Liste von Ankerknoten gegeben, die zu einem Experiment gehören. Nach erfolgreicher Initialisierung startet der Sensorknoten in zyklischen Intervallen von 4 Hz eine Distanz zu allen Knoten in der Liste zu ermitteln und die Distanzen über die serielle Schnittstelle an das virtuelle Testbed zu übermitteln. Ein kompletter Zyklus mit der maximalen Anzahl an vorhandenen Sensorknoten (25 Stück) dauert 74,5 Millisekunden, falls alle Knoten erreichbar sind und 254 Millisekunden im entgegengesetzten Fall. In der Praxis werden sich abhängig von der Dichte des Sensornetzes und der Länge der Liste Zeiten ergeben, die zwischen den entsprechenden Extrema liegen. Bei einem Beispiellauf mit 25 Ankern und einem Knotenabstand von 5 bis 10 Metern ergab sich eine durchschnittliche Anzahl an erreichbaren Ankern von 7,78. Dies entspricht einer durchschnittlichen Zeit von 198,14 Millisekunden, um die Liste einmal abzuarbeiten. In dieser Zeit bewegt sich der Roboter mit konstanter Geschwindigkeit von 10 cm/s (maximal 50 cm/s) fort. Die resultierende Differenz zwischen erster und letzter Messung beträgt daher 1,98 cm (9,91 cm) und ist somit im Verhältnis zur Genauigkeit der Distanzmessungen (vgl. Abschnitt 7.2) und des Referenzsystems (vgl. Abschnitt 5.3.1) zu vernachlässigen.



(a) Indoor (b) Outdoor

Abbildung 7.2: Verteilung des Distanzmessfehlers nachdem Ausreißer entfernt wurden. Negative Werte sind das Ergebnis von zu kurzen Messungen, positive Werte von zu langen Messungen.

7.2 Genauigkeit der Distanzmessung

Tabelle 7.1 zeigt einige Metriken zur Genauigkeit der Distanzmessung über alle Experimente dieser Arbeit. Die angegebenen Werte sind somit Durchschnittswerte und geben somit einen Eindruck von der Genauigkeit des Systems unter verschiedenen Bedingungen. Die Tabelle enthält ebenfalls die drei Parameter der dreieckigen MF für MD-Min-Max. Bei der Auswertung der einzelnen Experimente werden die jeweiligen Werte für das Experiment noch einmal genau aufgelistet. Während der Tests innerhalb von Gebäuden wurden 450 376 von insgesamt 674 055 Distanzmessungen erfolgreich durchgeführt. Dies entspricht einer Erfolgsquote von fast 67 %. Diese Quote hat für die Praxis aber nur eine geringe Aussagekraft, da aus zwei Gründen nicht immer die momentan erreichbaren Anker für die Distanzmessung verwendet wurden, sondern wie bereits beschrieben eine festgelegte Ankerliste abgearbeitet wird:

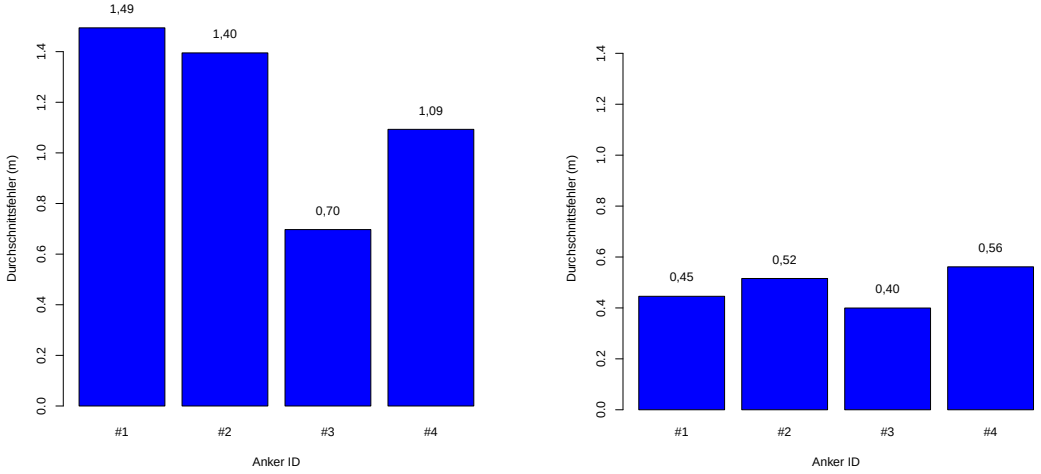
1. Soll der mobile Knoten nur Anker verwenden, die sich in seiner Reichweite befinden, müssten diese in regelmäßigen Abständen Pakete broadcasten, damit der mobile Knoten eine Nachbarschaftstabelle pflegen kann. Dies reduziert aber zugleich die pro Sekunde durchführbaren Messungen, da nun ein Medienzugriffsverfahren eingesetzt werden müsste. Da wir für die Evaluation der Algorithmen

nur einen mobilen Knoten benötigen, ist ein Pollingverfahren in diesem Fall effizienter, auch wenn öfters Anker verwendet werden, die gar nicht in Reichweite sind. In diesem Fall liefert der verwendete Treiber von Nanotron Technologies, wie im Falle einer fehlgeschlagenen Distanzmessung, −1,0 als Ergebnis zurück.

2. Ein ganz profaner Grund für dieses Vorgehen war die Instabilität des Nanotron Technologies Treibers im gemischten Betrieb von durchgeführten Distanzmessungen und gleichzeitigem „normalen“ Datenverkehr. Dies führte regelmäßig zu Abstürzen, im schlimmsten Fall dazu, dass manche Knoten gar keine Pakete mehr verarbeiten und somit nicht mehr erreichbar sind.

Von den erfolgreichen Messungen waren 157 691 oder gut 35 % zu kurz gemessen, wobei die maximal zu kurz gemessene Distanz einen Fehler von 5,95 m besaß. Der durchschnittliche Distanzfehler betrug 2,12 m bei einem RMSE von 3,34 m. Dennoch kann der Distanzfehler des öfteren bis zu 30 m betragen. In sehr seltenen Fällen kann der Fehler sogar bis auf ca. 75 m ansteigen. Abbildung 7.2a zeigt die Verteilung des Distanzfehlers über alle durchgeführten Messungen innerhalb von Gebäuden. Innerhalb von Gebäuden sind weniger als 3 % der Messfehler kleiner als −1,54 m und ebenso sind weniger als 3 % größer als 8,66 m.

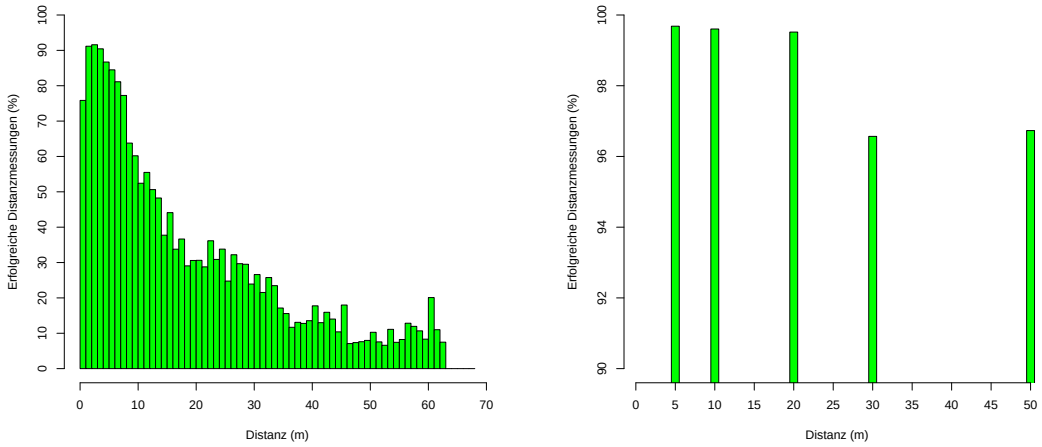
Zur Bestimmung der Genauigkeit der Distanzmessung unter Idealbedingungen haben wir Freifeldtests auf dem Tempelhofer Feld in Berlin durchgeführt. Dies ist das Gelände des ehemaligen Flughafens Berlin Tempelhof und bietet ausreichend Platz, um Tests unter LOS-Bedingungen ohne jegliche Störeinflüsse wie Personen oder andere Funktechnologien im selben Frequenzbereich durchführen zu können. Wir haben Entfernungsmessungen für Distanzen von 5, 10, 20, 30 und 50 Metern durchgeführt. Größere Entfernungen als 50 m sollten in Gebäuden selten vorkommen, weshalb wir die Freifeldbetrachtung dort abgebrochen haben. Pro Entfernung haben wir einige Tausend Messungen aufgezeichnet. Der mobile Knoten war hierfür mit einem Laptop verbunden und hat Messungen zu vier Ankerknoten durchgeführt. Die Anker sowie der mobile Knoten waren in einer Höhe von 1,80 Metern angebracht und damit außerhalb der ersten Fresnelzone. In der Mitte zwischen Sender und Empfänger ergibt sich der maximale Radius der ersten Fresnelzone durch $r_{\max} = \frac{\sqrt{\lambda \cdot d}}{2}$. Hierbei ist λ die Wellenlänge des Signals und d die Funkfeldlänge des Richtfunkfeldes. Für unser System ergibt sich somit ein Radius von 1,25 Metern bei einer maximalen Entfernung von 50 Metern. Dies ist wichtig, damit keine vom Erdboden reflektierten Signale die ursprünglichen Signale beim Empfänger auslöschen.



(a) Nicht korrigiert (b) *Bias* korrigiert

Abbildung 7.3: Der durchschnittliche Distanzfehler während des Freifeldtests in Abhängigkeit des verwendeten Ankers über alle Distanzen. In der rechten Grafik wurden die Distanzfehler um den *Bias* des jeweiligen Ankers korrigiert.

Insgesamt haben wir 129 680 Distanzmessungen durchgeführt, von denen 127 270 erfolgreich waren. Dies entspricht einer Erfolgsquote von gut 98 %. Von den erfolgreichen Messungen waren 120 602 oder fast 95 % zu kurz gemessen, wobei die maximal zu kurz gemessene Distanz einen Fehler von 4,32 m besaß. Der durchschnittliche absolute Distanzfehler betrug 1,17 m bei einem RMSE von 1,33 m. Abbildung 7.2b zeigt die Verteilung des Distanzfehlers über alle durchgeführten Messungen bei dem Freifeldtest. Hier sind weniger als 3 % der Messfehler kleiner als −2,45 m und ebenso sind weniger als 3 % größer als 0,27 m. Die Verteilung lässt sich wie in Abbildung 7.2b gezeigt durch eine Normalverteilung mit einem Mittelwert von −1,09 m und einer Standardabweichung von 0,77 m approximieren. Der Standardfehler beträgt 0,002 m und zeigt, dass die Normalverteilung eine gute Abschätzung für die Fehlerverteilung bildet. Der Vergleich der Varianz des Fehlers (0,59 m^2) mit dem MSE (1,78 m^2) sowie auch die reine Betrachtung des Histogramms aus Abbildung 7.2b zeigen, dass die Messung einen *Bias* beinhaltet. Eine mögliche Ursache für dieses Verhalten ist eine ungenaue Kalibrierung der Quarze der beteiligten Knoten. Die Genauigkeit ließe sich steigern, falls alle gemessenen Distanzen um den Mittelwert des Distanzfehlers verlängert würden. Abbildung 7.3 zeigt in der Tat, dass die erzielte Genauigkeit sogar von der verwendeten Ankerkombination abhängig ist. In Abbildung 7.3a sieht man, dass Distanzmessungen zu Anker 3 beispielsweise im Schnitt deutlich besser als Messun-

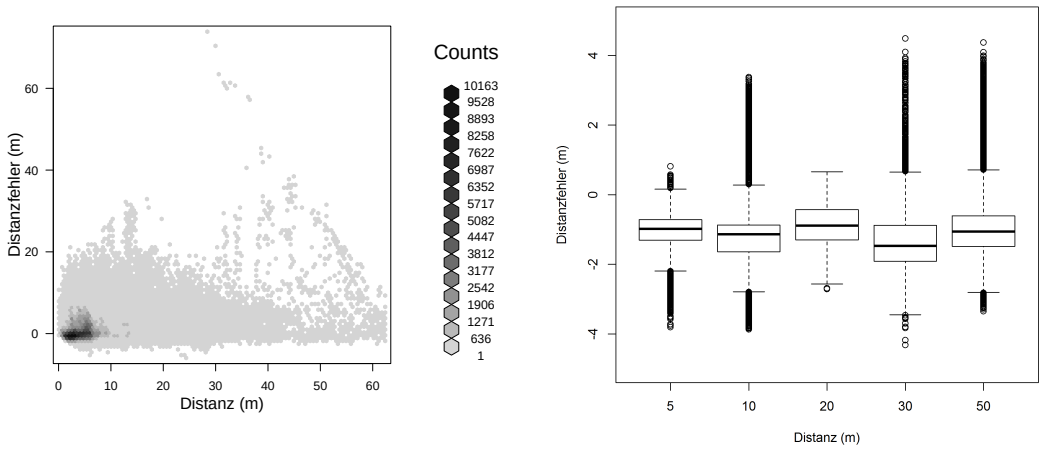


(a) Indoor (b) Outdoor

Abbildung 7.4: Relative Anzahl der erfolgreichen Distanzmessungen bei verschiedenen Entfernungen während der Experimente.

gen mit Anker 1 sind. Korrigiert man die Messungen für die einzelnen Anker um den *Bias* der Verteilung des jeweiligen Ankers, so sind die durchschnittlichen Fehler deutlich niedriger und liegen auch deutlich näher beieinander (vgl. Abbildung 7.3b). Die maximale Abweichung beträgt nur noch 16 cm (40 %) im Vergleich zu 79 cm (113 %) bevor der Korrektur. Der Durchschnittsfehler über alle Anker sinkt damit ebenfalls deutlich ab und beträgt nur noch 0,48 m mit einer Varianz von 0,475 m^2.

Abbildung 7.4a zeigt die Erfolgsquote der Distanzmessung bei verschiedenen Entfernungen in Prozent während der Testläufe in Gebäuden an. Ein Balken entspricht einem Distanzbereich von einem Meter. In einer Entfernung von 1 bis 4 Metern liegt die Wahrscheinlichkeit einer erfolgreichen Messung bei über 90 %. Danach sinkt die Wahrscheinlichkeit einer erfolgreichen Messung bis 35 m mit zunehmender Geschwindigkeit ab. Danach erhält man nur noch mit einer Wahrscheinlichkeit um 10 % eine valide Messung. Generell lässt sich festhalten, dass man mit der eingesetzten Hardware im Indoor-Bereich noch bis zu einer Reichweite von ca. 15 m eine tolerierbare Fehlerrate erzielt. Weiterhin fällt auf, dass die Fehlerrate im sehr nahen Bereich zwischen 0 m und 1 m mit 25 % merklich höher ist als bei Entfernungen zwischen 1 m bis 4 m. Wie eingangs beschrieben, werden bei dieser Statistik bedingt durch die Rangingprozedur alle Anker betrachtet, auch die, die außer Reichweite liegen. Somit dient diese Statistik in erster Linie einer Einschätzung der durchschnittlichen



(a) Indoor (b) Outdoor

Abbildung 7.5: Verteilung des Distanzmessfehlers bei verschiedenen Entfernungen während der Experimente.

Indoor-Sendereichweite der eingesetzten Hardware.

Abbildung 7.4b gibt die Erfolgsquote der Distanzmessung bei den Freifeldtests für die fünf festen Distanzen wieder. Für die Distanzen von 5, 10 und 20 Metern liegt die Wahrscheinlichkeit einer erfolgreichen Messung bei über 99 %. Bei 30 Metern sowie 50 Metern fällt die Wahrscheinlichkeit einer erfolgreichen Messung leicht ab, beträgt aber immer noch mehr als 96 %. Um die theoretische Reichweite des Transceivers unter den gegebenen Idealbedingungen zu ermitteln, haben wir zum Abschluss die Distanz sukzessive erhöht, indem wir uns mit konstanter Geschwindigkeit von den vier Ankern entfernt haben. Selbst bei einer Entfernung von 160 Metern lieferte der Funkchip noch konstant Entfernungsmessungen. Hier haben wir den Versuch abgebrochen.

In Abbildung 7.5a sieht man die Verteilung des Distanzmessfehlers bei verschiedenen Entfernungen während der Testläufe in Gebäuden. Wie man sehen kann, ist der Distanzfehler unabhängig von der Distanz, an der gemessen wurde. Wenige größere Ausreißer wurden bei Distanzen von 30 bis 40 Metern gemessen. Sonst verteilt sich der Fehler gleichmäßig über die Distanz, d. h. dass sich z. B. bei größeren Distanzen kein größerer Distanzfehler ergibt. Die geringe Anzahl an Werten bei Distanzen größer als 15 m erklärt sich durch die erhöhte Wahrscheinlichkeit von fehlerhaften Messungen. Messungen bei Distanzen von 20 bis 30 Metern gelingen nur in 25 % bis 35 % der Versuche, bei 30 Metern und größer nur in 10 % bis 20 % der Versuche

(vgl. Abbildung 7.4a). Der Korrelationskoeffizient der beiden Daten beträgt zudem 0,12. Somit ist der Fehler unkorreliert zur tatsächlichen Distanz. Dies ist ein starker Hinweis darauf, dass beide Größen voneinander unabhängig sind.

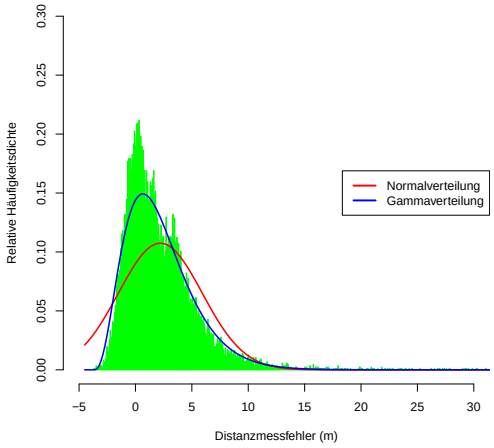
Abbildung 7.5b zeigt die Verteilung des Distanzmessfehlers für die fünf Distanzen des Freifeldtests als Boxplot. Auch hier bestätigt sich, dass der Distanzfehler unabhängig von der Distanz ist, an der gemessen wurde. Die Mediane der Boxen liegen sehr eng beieinander mit einer maximalen Abweichung von 58 cm und auch die Interquartilsabstände sind ähnlich groß. Ausreißer in die positive Richtung sowie in die negative Richtung finden sich für fast alle Distanzen in der gleichen Größenordnung. Der Korrelationskoeffizient der beiden Daten beträgt hier sogar nur 0,044, was auf eine Unabhängigkeit der beiden Größen schließen lässt.

7.3 Parameterbestimmung für verteilungsabhängige Algorithmen

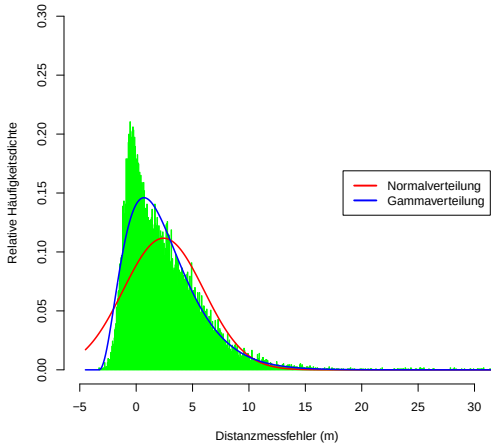
Wie in Kapitel 3 beschrieben, müssen MLE-$\mathcal{N}$, MLE-Γ sowie MD-Min-Max parametrisiert werden. Im Falle von Geo-n kann eine erweiterte Version des Algorithmus verwendet werden, die ebenfalls parametrisiert werden muss. Die Parameter werden anhand der echten Fehlerverteilung des Distanzmessfehlers in einem Testszenario ermittelt. Die Übertragbarkeit auf andere Experimente wird in Abschnitt 7.4 untersucht.

Abbildung 7.6 zeigt die Verteilung des Distanzmessfehlers für zwei ausgewählte Testläufe von Experiment 4 aus Abschnitt 7.4, welche im zweiten Stock unseres Bürogebäudes stattfanden und mit Testlauf 1 und Testlauf 2 bezeichnet sind. In Testlauf 1 wurde ein Pfad von ungefähr 72 m abgefahren, in Testlauf 2 ein Pfad von gut 100 m und dabei jeweils mit insgesamt 17 Ankern 17 277 bzw. 22 901 erfolgreiche TOF-Messungen durchgeführt. Falls die Distanzmessung kürzer als die echte Distanz ist, wird der Fehler als negativer Fehler definiert, andernfalls als positiver Fehler. Aus Darstellungsgründen sind Ausreißer mit Werten größer als 30 m nicht abgebildet.

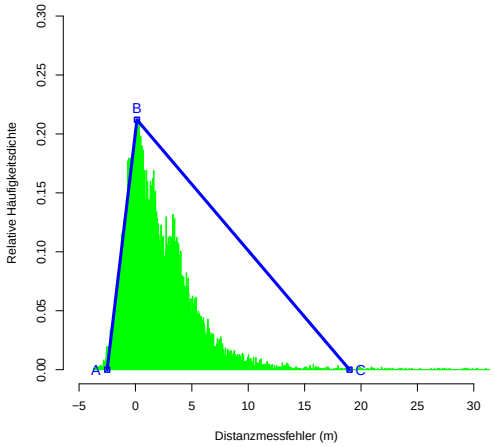
Abbildungen 7.6a und 7.6b zeigen das Fitting der Messdaten mittels Normalverteilung und Gammaverteilung. Für diesen Zweck wurde die Programmiersprache R [128] und dessen Funktion *fitdistr(x, densfun, start, ...)* aus dem *MASS*-Paket verwendet.



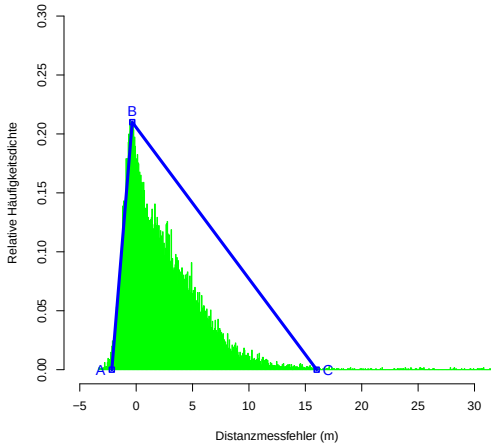
(a) Testlauf 1 (MLE und Geo-n)



(b) Testlauf 2 (MLE und Geo-n)



(c) Testlauf 1 (MD-Min-Max)



(d) Testlauf 2 (MD-Min-Max)

Abbildung 7.6: Verteilung des Distanzmessfehlers zweier ausgewählter Testläufe mit 17 277 und 22 901 erfolgreichen TOF Messungen. Werte größer als 30 m werden als Ausreißer betrachtet und sind deshalb nicht dargestellt. In Abbildung 7.6a und Abbildung 7.6b wurde die Verteilung mittels Normal- und Gammaverteilung für die Algorithmen MLE-($\mathcal{N}$, Γ) und Geo-n-($\mathcal{N}$, Γ) approximiert, in Abbildung 7.6c und Abbildung 7.6d mittels der in Abschnitt 3.3 beschriebenen Dreiecksmethode für MD-Min-Max.

Tabelle 7.2: Verteilungsparameter der Testläufe

	Testlauf 1	Testlauf 2
Normalverteilung	$\mathcal{N}(2.205, 3.715^2)$	$\mathcal{N}(2.426, 3.572^2)$
Gammaverteilung	$\Gamma(3.824, 0.647), \mu = 3.70$	$\Gamma(3.306, 0.576), \mu = 3.31$
Dreieckige MF	$[-2.508, 0.074, 18.981]$	$[-2.16, -0.349, 16.005]$

Diese ermöglicht das Maximum-Likelihood-Fitting von univariaten Verteilungen. Als Resultat ergeben sich die in den Abbildungen dargestellten roten und blauen Kurven mit den in Tabelle 7.2 dargestellten Parametern. Der Parameter μ bezeichnet den Lokalisierungsparameter der Gammaverteilung.

Abbildungen 7.6c und 7.6d zeigen das Fitting der Messdaten mit der in Abschnitt 3.3 beschriebenen Dreiecksmethode zur Parameterbestimmung für den MD-Min-Max Algorithmus. So ergeben sich für das Fitting von Testlauf 2 mit dem 0,005 Quantil, dem Modus und dem 0,995 Quantil beispielsweise $[-2.16, -0.349, 16.005]$ als Parameter der MF, welche die x-Koordinaten der Punkte A, B und C bilden. Die Parameter für beide Testläufe sind ebenfalls noch einmal in Tabelle 7.2 abgebildet. Falls nicht anderweitig erwähnt, werden für die Auswertung der Experimente im Falle von MD-Min-Max sowie für alle anderen verteilungsabhängigen Algorithmen die Parameter von Testlauf 2 verwendet. Für einen Test der Übertragbarkeit der Parameter auf andere Szenarien sollten entsprechend immer die Werte benutzt werden, die auf den Messdaten eines anderen Szenarios beruhen. Die Ähnlichkeit der Parameter von Testlauf 1 und Testlauf 2 legt bereits nahe, dass die Distanzfehler im gleichen oder ähnlichen Szenario ein ähnliches Verhalten zeigen und die erwähnten Algorithmen somit einfach in der Praxis anwendbar sind.

7.4 Auswertung

Tabelle 7.3 zeigt eine Auflistung der in diesem Abschnitt untersuchten Algorithmen und deren Parametrisierung, falls vorhanden. Die Bedeutung und Bezeichnung der Parameter ist der Einführung der Algorithmen in Kapitel 3 entnommen. Eventuell abweichende Parametrisierungen werden bei den Auswertungen explizit angegeben, ansonsten gelten experimentübergreifend diese Werte.

Tabelle 7.3: Algorithmen und Parameter

Algorithmus	Parameter
NLLS_{GN}	Max. Iterationen: 10; $\epsilon = 0{,}001\,\text{m}$
NLLS_{LM}	Max. Iterationen: 10; $\epsilon = 0{,}001\,\text{m}$
ICLA	$\lambda = 0{,}5\,\text{m}$; $\alpha = 1{,}5$
Geo-n	$W_{IP} = 1$, $W_{AIP} = 1$
Verteilungsabhängige Algorithmen:	
VBLE	$L = 5\,\%$ der kürzeren Zielfeldseite; $\epsilon = 2{,}85\,\text{m}$
VBLE-O	$L = 40\,\%$ bis $5\,\%$ der kürzeren Zielfeldseite; $\epsilon_{min} = 0\,\text{m}$; $\epsilon_{max} = 2{,}85\,\text{m}$
CluRoL	$\delta_{max} = 0{,}05$
Geo-n-$\mathcal{N}$, MLE-$\mathcal{N}$	$\mu = 2{,}426\,\text{m}$; $\sigma = 3{,}572\,\text{m}$
Geo-n-Γ, MLE-Γ	$\alpha = 3{,}306\,\text{m}$; $\beta = 0{,}576\,\text{m}$; $\mu = 3{,}31\,\text{m}$
MD-Min-Max	$MF_{\text{low}} = -2{,}16\,\text{m}$; $MF_{\text{mode}} = -0{,}349\,\text{m}$; $MF_{\text{up}} = 16{,}005\,\text{m}$
Weitere untersuchte Algorithmen ohne Parameter:	
Min-Max, E-Min-Max (W_2), E-Min-Max (W_4), LLS, AML, Rwgh, RLSM, LMS, BL	

Der Vollständigkeit halber sind die Parameter der beiden NLLS Algorithmen aufgeführt. Dies sind die maximale Anzahl der Iterationen und das Abbruchkriterium ϵ. Für ICLA werden die Werte aus der Publikation von Haiyong et al. verwendet. Geo-n gewichtet echte und approximierte Kreisschnittpunkte gleich. Die Parameter dieser Algorithmen sind von keinem spezifischen Szenario und entsprechender Fehlerverteilung abhängig und können somit als universelle Algorithmen für die Lokalisierung angesehen werden. Gleiches gilt ebenfalls für die parameterlosen Algorithmen in Tabelle 7.3. Der Algorithmus Rwgh verwendet zur Berechnung von Zwischenpositionen NLLS_{GN} mit den dargestellten Parametern.

Der Parameter ϵ von VBLE muss anhand des erwarteten maximalen Fehlers bestimmt werden und benötigt somit vorheriges Wissen, mindestens Wissen über das eingesetzte Distanzmesssystem und dessen Fehler. In der Veröffentlichung von Liu et al. wird in den Simulationen ein gleichverteilter Fehler zwischen $-\epsilon$ und ϵ benutzt ($\epsilon = 4\,\text{m}$). Während dies bei einer Gleichverteilung Sinn ergibt, ist es bei Verteilungen, die in der Realität vorkommen, sinnvoller den Wert auf den Erwartungswert des Distanzmessfehlers (MAE) zu setzen. Hierfür wurde der gleiche Testlauf wie für die

anderen verteilungsabhängigen Algorithmen verwendet. Die Seitenlänge L der Zellen wird bei VBLE auf 5 % der Länge der kürzeren Seite des *Zielfeldes* gesetzt. Liu et al. machen keine konkreten Angaben zur Bestimmung der Seitenlänge und der daraus resultierenden Anzahl an Zellen M. Diese sollte nur an den verfügbaren Speicher der Sensorknoten angepasst werden und wird in den Simulationen auf $M = 100$ gesetzt. Dies entspricht einem Wert von 10 % der Länge einer Seite, falls das *Zielfeld* quadratisch ist. Für VBLE-O wird die initiale Seitenlänge auf 40 % der Länge der kürzeren Seite des *Zielfeldes* gesetzt und dann in jedem Verfeinerungsschritt halbiert, jedoch der 5 %-Wert von VBLE nicht unterschritten, um eine Vergleichbarkeit zu gewährleisten. Für CluRoL muss der Parameter δ_{max} bestimmt werden. Dieser hängt laut Misra und Xue in gewisser Weise vom maximalen Distanzfehler ab [92]. Da keine genaue Beschreibung in der Veröffentlichung vorhanden ist, haben wir den Wert bei 0,05 belassen.

Die Analyse der verschiedenen Algorithmen erfolgte mittels des in Java geschriebenen Programms *LatViz*, welches in Abschnitt 5.2.2 vorgestellt wurde. Dieses Werkzeug unterstützt das Einlesen der aus der Datenbank extrahierten Lokalisierungsdaten und nimmt eine automatische Auswertung ausgewählter Algorithmen nach verschiedenen Metriken vor. Bei den Metriken handelt es sich im Wesentlichen um die in Kapitel 4 eingeführten Kenngrößen wie MAE, RMSE, Maximalfehler oder *Bias*. Die erzeugten Daten lassen sich visualisieren und ebenfalls zur weiteren Verarbeitung exportieren. Obwohl sich die Genauigkeit der Lokalisierung natürlich durch Verwendung von Filtertechniken steigern ließe, z. B. durch die Nutzung von Kalman- oder Partikelfiltern, so geht es uns bei der Auswertung um die reine Leistung und den Vergleich der Algorithmen unter Ausschluss jeglicher dieser Filtertechniken. Somit zeigen sämtliche Abbildungen der geschätzten Positionen die bloße Performance der Algorithmen und lassen Rückschlüsse auf deren Arbeitsweise zu. Die Lokalisierungsdaten bestehen aus den verwendeten Ankern und deren Position sowie den eigentlichen Messdaten. Ein einzelnes Messdatum besteht aus einem Unix-Zeitstempel, einer Reihe von gemessenen Distanzen in Metern und der Referenzposition. Jedes in der Datenbank gespeicherte und im Folgenden untersuchte Experiment besteht aus einer festen Ankerausbringung und kann verschiedene Testläufe enthalten. Ein solcher Testlauf kann aus einer einfachen Fahrt durch Flure und Räume bestehen, aus einer Aufzeichnung an einer festen Position oder aber aus einer kompletten Kartierung eines Gebäudeteils für die eigentliche Nutzung als virtuelles Testbed.

7.4.1 Experimententwurf

In den von uns durchgeführten Experimenten findet man Instanzen der Simulationssetups aus Abschnitt 6.1. So ist Experiment 1 an das *grid* Szenario aus Abschnitt 6.1 angelehnt (9 Anker, gitterförmige Anordnung). Für die Indoor-Lokalisierung stellt dies ein sehr optimales Szenario dar, an dem sich gut ablesen lässt, inwiefern der jeweilige Algorithmus von einem Mehr an Ankerknoten profitiert und somit die entstandene Redundanz nutzen kann. Die restlichen Experimente ähneln einer Kombination aus *quad*, *kite* und *tube* Konstellation aus Abschnitt 6.1, wobei die einzelnen Szenarien in verschiedenen Abschnitten eines Experiments anzutreffen sind. So ist das *quad* Szenario (4 Anker in den Raumecken) beispielsweise auf Abschnitten des Pfades von Experiment 2 (Seminarraum 046 und 049) anzutreffen. Experiment 2 dient im Wesentlichen zur Beurteilung der Algorithmen bei unterschiedlichen Ankerdichten und Vorhandensein von soliden Stahlbetonwänden. In Experiment 3 sind größere Ankerabstände vorhanden und die Ankeranordnung ist weniger symmetrisch. Testlauf 2 und 3 von Experiment 3 stellen Extremfälle für die Indoorlokalisierung dar, da sich der mobile Knoten hier zunehmend außerhalb der konvexen Hülle der Ankerknoten bewegt. Dies ist beispielsweise besonders für den Min-Max Algorithmus eine ungünstige Konstellation (vgl. Abschnitt 6.2) und soll die Stärken und Schwächen der einzelnen Algorithmen unter solchen Bedingungen offenlegen. Experiment 4 dient zur Beurteilung der Algorithmen in Gängen. Dies entspricht insgesamt am ehesten einem üblichen Szenario, wie man es in vielen Einsatzgebieten der Indoorlokalisierung finden dürfte und wird in den Simulationen durch das *tube* Szenario abgebildet. Durch die Anlehnung der Experimententwürfe und Simulationssetups soll insgesamt eine Vergleichbarkeit der jeweiligen Ergebnisse ermöglicht werden und sich so die Simulationssettings und deren Schlüsse in der Realität verifizieren lassen. Ferner wird hiermit eine breite und solide Basis für die Beurteilung der einzelnen Algorithmen geschaffen, die eine allgemeingültige Aussagekraft über die Grenzen der einzelnen Experimente hinaus hat.

Für die numerische Auswertung der Algorithmen haben wir uns für den MAE, den RMSE und den maximalen Fehler als Metrik entschieden. Eine Diskussion dieser und weiterer Metriken inklusive der Vor- und Nachteile haben wir in Abschnitt 4.2.3 vorgenommen. Zusätzlich haben wir für jedes Experiment die kumulative Verteilungsfunktion (CDF) des Positionsfehlers als Maß für die Präzision angegeben, mit deren Hilfe sich auch die verschiedenen Quantile ablesen und vergleichen lassen. Die Prä-

zision der Lokalisierung gibt an, wie konsistent ein Algorithmus arbeitet, ist also ein Maß für die Robustheit eines Algorithmus, da er die Schwankung der Leistung eines Algorithmus über mehrere Versuche offenlegt. Beim Vergleich von zwei Algorithmen mit gleichem MAE ist im Allgemeinen der Algorithmus zu bevorzugen, dessen CDF-Kurve große Wahrscheinlichkeitswerte schneller erreicht (der Positionsfehler ist also vornehmlich in kleinen Werten konzentriert).

Bei unserer Auswertung haben wir generell den MAE als Maß für die Genauigkeit und für den Vergleich der einzelnen Algorithmen verwendet. Dies gilt im Besonderen, wo wir die Leistung der Algorithmen eines Experiments mit Prozentwerten gegenübergestellt haben. Hierbei wird die Hypothese, dass ein Algorithmus A besser als ein Algorithmus B lokalisiert, als wahr angenommen, wenn der erzielte MAE von Algorithmus A kleiner als der von Algorithmus B ist. Um Abweichungen aus dem Messapparat zu berücksichtigen, wird dem MAE eine Unsicherheit zugeschrieben, da dieser durch die endliche Anzahl an Messwerten ebenfalls noch zufälligen Fehlern unterliegt. Je größer die Anzahl der Messwerte ist, desto kleiner wird diese Unsicherheit. Für die experimentübergreifenden Ergebnisse aus Abschnitt 7.5 und Testlauf 1 von Experiment 1 haben wir daher eine Abschwächung von 5 cm für den Vergleich gewählt (ca. 2,5 % des mittleren Indoor-Messfehlers; mehr als 10 000 Lokalisierungen). Ein Verfahren eignet sich hier also besser als ein anderes Verfahren, wenn der MAE um mindestens 5 cm kleiner ist. Für die anderen Testläufe wird eine Abschwächung von 15 cm verwendet, da hier weniger Messwerte vorhanden sind (ca. 7,5 % des mittleren Indoor-Messfehlers; weniger als 5000 Lokalisierungen. Der Fall 5000 bis 10 000 Lokalisierungen mit einer Abschwächung von 10 cm (5 %) tritt nicht auf). Die Robustheit der Algorithmen vergleichen wir anhand der CDFs und dient als Maß für die Präzision der Algorithmen. Auch hier verwenden wir die gleiche Abschwächung beim Vergleich von zwei Algorithmen. Verläuft die CDF-Kurve eines Algorithmus zu jeder Zeit über der Kurve eines Vergleichsalgorithmus, so ist dieser Algorithmus definitiv besser und dem anderen Algorithmus vorzuziehen. Auch hier muss der Abstand mindestens 5 cm bzw. 15 cm betragen, ansonsten kann keine definitive Aussage getroffen werden, ob die Algorithmen gleichwertig sind oder nicht.

Die Bewertung und die Entscheidung für oder wider einen Algorithmus hängt in erster Linie aber auch vom konkreten Einsatzszenario ab. So ist es zum Beispiel für das *FeuerWhere*-Szenario aus Abschnitt 2.4 besser, einen Algorithmus mit größerem MAE und kleinerem Maximalfehler (oder großem Quantil, z. B. 99 %) zu verwenden, als den Algorithmus zu nehmen, der im Durchschnitt die besten Werte erzielt, jedoch

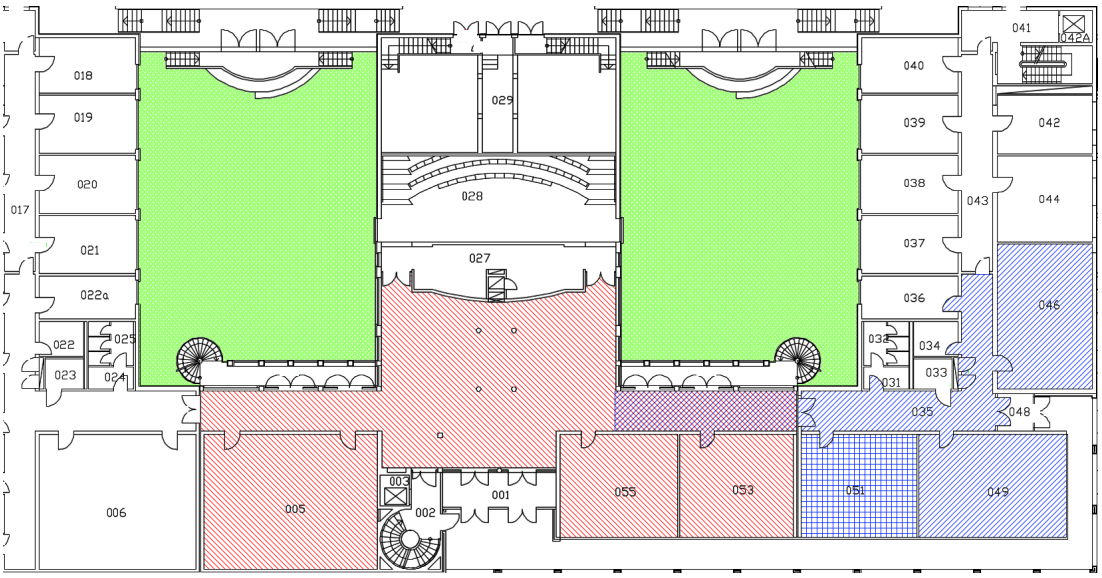


Abbildung 7.7: Gebäudeplan vom Erdgeschoss des Instituts für Informatik der Freien Universität Berlin. Bereiche von Experiment 1 und Experiment 2 sind in Blau hervorgehoben. Der blaue Bereich spannt eine rechteckige Fläche von ungefähr 32 m × 20 m auf. Der Bereich von Experiment 3 ist in Rot hervorgehoben und spannt eine rechteckige Fläche von ungefähr 38,5 m × 20 m auf. Die Innenhöfe sind in Grün hervorgehoben.

erheblich große Ausreißer besitzt. Dies ist relevant, da Personen jederzeit raumgenau geortet werden müssen. Durch die Angabe der oben aufgeführten Metriken soll daher ferner eine Algorithmenauswahl für andere Einsatzzwecke ermöglicht werden, die einen anderen Fokus bei der Bewertung der Algorithmen legen.

7.4.2 Experiment 1

Als erstes Experiment haben wir einen mittelgroßen Aufbau in einem geschlossenen Raum mit den Maßen 7,50 m × 6,80 m gewählt, da dies ein verbreitetes Setup ist (vgl. Tabelle 5.4, S. 146). In Abbildung 7.7 ist dieser Seminarraum 051 blau kariert hervorgehoben. Das Experiment wird im Folgenden als Experiment 1 bezeichnet und bestand aus mehreren Testläufen. Diese sind in Tabelle 7.4 aufgeführt. Alle Testläufe haben neun Anker zur Verfügung gehabt, die nahezu gitterförmig angeordnet und bis auf den zentralen Knoten nahe der Wände auf Tischen platziert waren. Die innere Fläche des Raumes war somit frei von Hindernissen und der Roboter konnte diese freie Fläche für das virtuelle Testbed komplett kartieren. Die Referenzdaten dieses

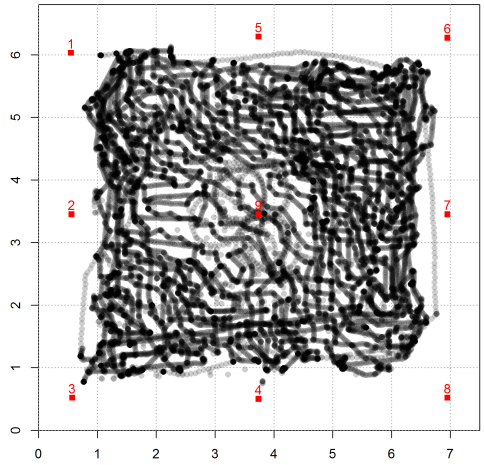
Tabelle 7.4: Testläufe und Metriken von Experiment 1

Metrik	Testlauf			
	1	2	3	4
Referenzpositionen	29 109	387	736	3306
Distanzmessungen	261 981	3483	6624	29 754
Erfolgsquote	88,67 %	91,47 %	91,14 %	90,44 %
Durchschnittliche Ankeranzahl	7,98	8,23	8,20	8,14
Pfadlänge	-	13,48 m	27,53 m	-
Durchschnittsfehler*	1,87 m	1,64 m	1,79 m	2,23 m
Varianz	6,60 m^2	5,59 m^2	6,07 m^2	9,33 m^2
MSE	8,37 m^2	6,52 m^2	7,60 m^2	13,04 m^2
RMSE	2,89 m	2,55 m	2,76 m	3,61 m
Maximalfehler*	21,75 m	13,79 m	15,84 m	22,02 m
0,5 % Quantil	−2,15 m	−2,14 m	−2,17 m	−1,90 m
Modus	−0,37 m	−0,49 m	−0,51 m	0,71 m
99,5 % Quantil	11,37 m	10,55 m	10,73 m	13,02 m

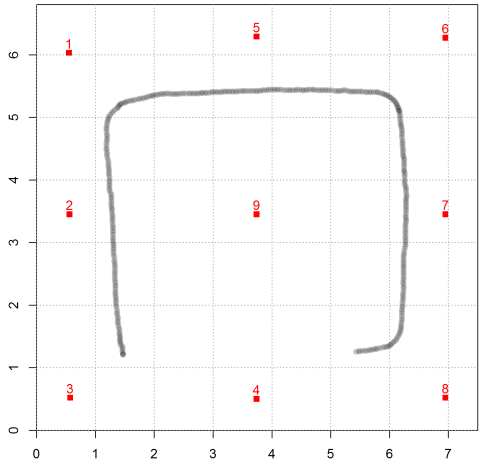
* absoluter Fehlerwert

Testlaufes (Testlauf 1) sind in Abbildung 7.8a dargestellt. Je schwärzer die Punkte, desto mehr Referenzdaten wurden an dieser Stelle aufgenommen. Insgesamt wurden 29 109 Referenzpositionen aufgezeichnet. Testlauf 2 und 3 sind kürzere Roboterfahrten und dienen als Vergleichsläufe. Diese wurden auch zur einfachen Erzeugung von virtuellen Pfaden zur initialen Evaluierung des virtuellen Testbeds genutzt, um mit diesen Pfaden als Eingabe virtuelle Pfade aus den Daten von Testlauf 1 zu erstellen. Testlauf 4 besteht aus den vier festen Positionen A, B, C und D. Der Roboter wurde hier nicht bewegt und jeweils gut 800 Messungen aufgezeichnet. Dies soll einer Analyse der Varianz und des *Bias* der Algorithmen dienen, wie dies auch in Abschnitt 6.2 bei den Simulationen der Fall gewesen ist.

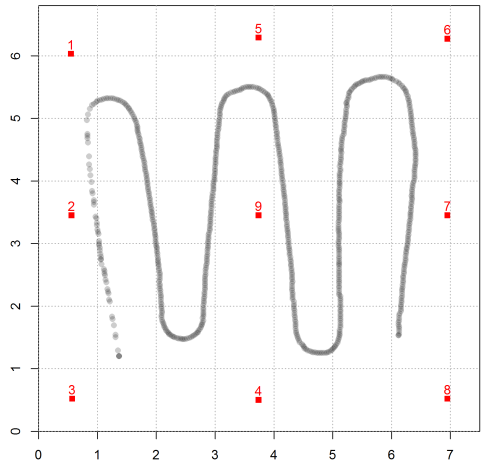
Tabelle 7.4 zeigt noch einmal die wichtigsten Daten zu den einzelnen Testläufen. Neben der Anzahl der Distanzmessungen ist auch deren Erfolgsquote angegeben und die resultierende durchschnittliche Anzahl der Ankerknoten, die für eine Lokalisierung zur Verfügung stehen. Bei mobilen Roboterfahrten, die keine Komplettkartierung vornehmen, ist zudem die Länge des abgefahrenen Pfades aufgeführt. Die Erfolgsquote ist erwartungsgemäß mit durchschnittlich 90 % hoch, da prinzipiell eine direkte Sichtverbindung bestand und alle Knoten sich in der theoretischen Sendereichweite befanden.



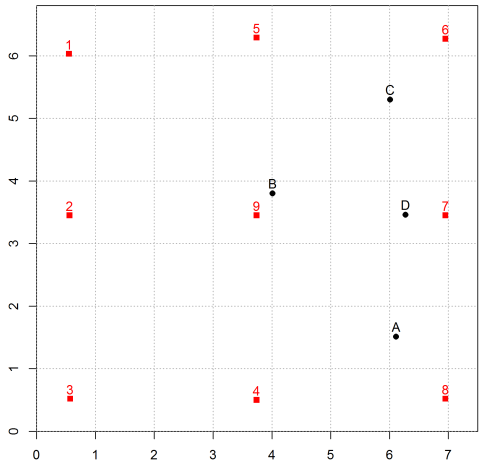
(a) Testlauf 1



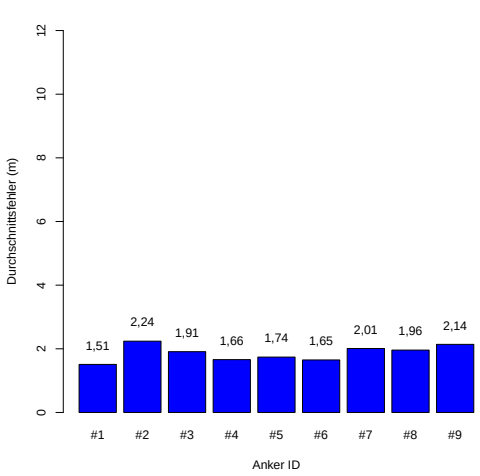
(b) Testlauf 2



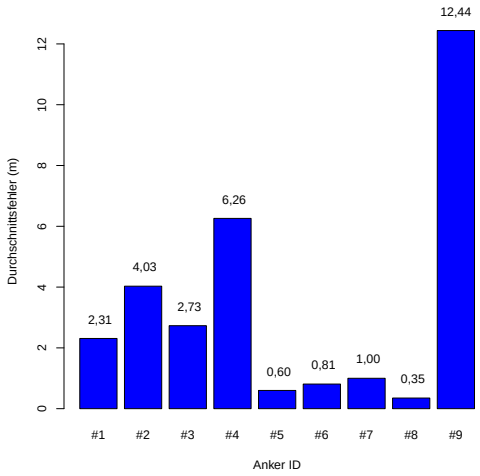
(c) Testlauf 3



(d) Testlauf 4



(e) Distanzmessfehler pro Anker (Testlauf 1)



(f) Distanzmessfehler pro Anker (Testlauf 4, Position D)

Abbildung 7.8: Abbildungen 7.8a bis 7.8d zeigen die einzelnen Testläufe von Experiment 1 (alle Angaben in Metern), Abbildungen 7.8e und 7.8f den MAE der Distanzmessungen für die einzelnen Anker für zwei ausgewählte Tests.

Daraus resultiert auch eine sehr hohe durchschnittliche Ankeranzahl von ca. 8.

Ebenfalls zu sehen sind die Metriken zur Genauigkeit der Distanzmessung. Testlauf 1 und 3 zeigen ähnliche Ergebnisse, bei Testlauf 2 fällt der Distanzmessfehler etwas niedriger aus. Da hier nur Bereiche am Rand des Raumes abgefahren wurden, scheint die Messgenauigkeit im Schnitt dort leicht besser zu sein. Mit einem MAE von 2,23 m ist der Distanzmessfehler der festen Positionen am größten. Für die einzelnen Positionen A, B, C und D ergibt sich ein durchschnittlicher Fehler von 2,22 m, 1,53 m, 1,77 m bzw. 3,33 m. Besonders Position D weicht von allen anderen Messungen stark ab. Dies verdeutlichen Abbildung 7.8e und Abbildung 7.8f noch einmal. Während die Durchschnittsfehler der Anker bei Testlauf 1 recht nahe beieinander liegen, so ergibt sich für die festen Positionen ein anderes Bild. Dies sieht man besonders bei Position D von Testlauf 4. Hier gibt es 4 Anker, die relativ genau messen und mehrere Anker, die größere Ausreißer produzieren. Insbesondere Messungen zu Anker 9 sind konstant zu lang und resultieren in einem MAE von 12,44 m. Ähnliche Ergebnisse liefern die anderen festen Positionen, nur dass die Anzahl der Ausreißer und deren Fehlergröße variieren. Somit kann Testlauf 4 zudem als guter Benchmark für die Robustheit der Lokalisierungsalgorithmen in einem solchen kompakten Aufbau dienen.

Tabelle 7.5 zeigt die Genauigkeit der Lokalisierung aller Algorithmen anhand des MAE und des maximalen Fehlers für alle Testläufe. Die Präzision der Lokalisierung ist in Abbildung 7.9a bis 7.9c für Testlauf 1 dargestellt. Diese zeigen die CDF des Positionsfehlers der Algorithmen. Der besseren Lesbarkeit halber sind maximal sieben Algorithmen pro Abbildung eingezeichnet. Zudem sind die Algorithmen absteigend nach dem MAE geordnet, d. h. Abbildung 7.9a enthält die sieben Algorithmen mit den größten MAEs. Wegen der nahezu identischen Ergebnisse beider NLLS Algorithmen ist nur NLLS_{GN} abgebildet. Weiterhin ist in allen Abbildungen der durchschnittliche Distanzmessfehler als gestrichelte schwarze Linie zum Vergleich eingezeichnet.

Betrachtet man Testlauf 1 mit der größten Anzahl an Messdaten, so schneiden LLS mit 4,03 m und NLLS mit 2,45 m sowie die darauf fußenden Verbesserungen (LMS mit 3,23 m, RLSM mit 3,47 m, CluRoL mit 3,68 m und Rwgh mit 2,37 m) am schlechtesten ab. Dies gilt ebenfalls für AML mit einem Fehler von 2,65 m, welcher von Kuruoglu et al. als laufzeitverbesserte Alternative zu LLS vorgestellt wurde [70]. Der Positionsfehler liegt bei allen Algorithmen über dem durchschnittlichen Fehler der Distanzmessungen von 1,87 m. Jedenfalls zeigen alle verbesserten Algorithmen eine gewisse Steigerung der Genauigkeit gegenüber ihrem Basisalgorithmus (CluRoL mit

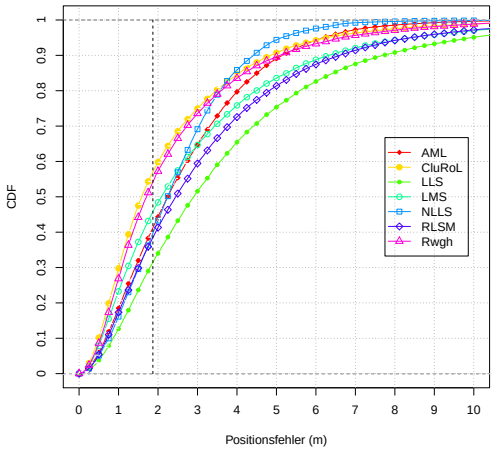
Tabelle 7.5: Ergebnisse von Experiment 1

Algorithmus	Testlauf 1		Testlauf 2		Testlauf 3		Testlauf 4	
	MAE	MAX	MAE	MAX	MAE	MAX	MAE	MAX
AML	2,65	15,50	2,41	9,61	2,52	11,43	3,59	13,50
BL	1,16	12,20	0,91	3,74	1,14	6,05	1,12	5,26
CluRoL	3,68	9661	1,95	12,59	8,77	4717	2,21	30,89
E-Min-Max (W2)	0,91	11,62	0,71	2,56	0,88	4,33	0,93	5,08
E-Min-Max (W4)	0,91	8,94	0,72	2,35	0,87	3,68	0,84	4,05
Geo-n	1,08	12,51	0,84	3,81	1,02	7,88	0,98	7,47
ICLA	1,67	14,94	1,31	9,68	1,56	9,50	1,40	9,79
LLS	4,03	6605	3,70	15,92	3,84	84,75	4,73	56,30
LMS	3,23	6605	2,86	21,79	2,92	84,75	3,72	56,30
Min-Max	0,93	6,20	0,73	2,31	0,88	3,71	0,78	3,36
NLLS_{GN}	2,45	13,06	1,89	6,79	2,40	10,78	2,98	10,86
NLLS_{LM}	2,48	13,75	1,89	6,24	2,40	10,78	2,98	10,94
RLSM	3,47	6605	3,11	13,05	3,31	84,75	3,81	56,30
Rwgh	2,37	15,92	1,96	12,42	2,25	12,96	2,45	14,16
Geo-n-$\mathcal{N}$	1,03	12,82	0,77	2,95	0,97	7,10	1,18	6,58
Geo-n-Γ	0,93	12,82	0,70	2,85	0,91	5,26	1,15	6,34
MD-Min-Max	0,90	8,03	0,72	2,29	0,86	4,12	0,78	4,61
MLE-$\mathcal{N}$	1,32	10,51	1,13	3,67	1,20	8,27	1,47	10,02
MLE-Γ	1,03	12,19	0,72	3,92	1,03	7,70	1,37	6,38
VBLE	2,13	14,38	1,73	6,24	2,23	10,09	2,76	9,68
VBLE-O	1,36	16,73	1,17	7,77	1,30	10,67	1,27	8,70

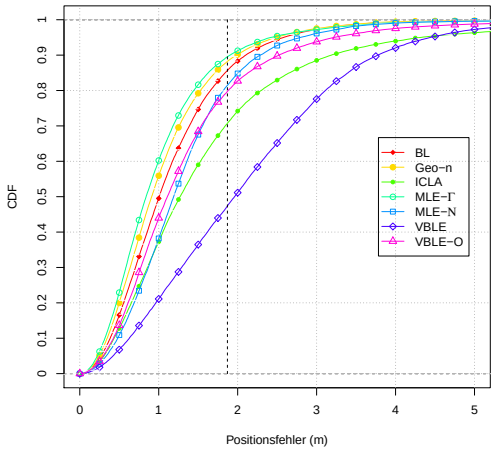
(alle Angaben in Metern)

8,7 %, RLSM mit 13,9 %, LMS mit 19,9 % gegenüber LLS und Rwgh mit 3,3 % gegenüber NLLS). Die beiden NLLS Algorithmen, NLLS_{GN} und NLLS_{LM}, zeigen nahezu identische Performance und liegen nur minimal von dem MAE entfernt, falls beide Algorithmen die echte Position als Startwert der Optimierung erhalten. Dieser beträgt in diesem Fall 2,40 m. Betrachtet man die CDFs der Algorithmen, so fällt besonders bei CluRoL und Rwgh auf, dass die Fehlerverteilung beider Algorithmen rechtsschief ist. Diese besitzen die anfangs am stärksten steigenden CDF-Kurven, fallen dann aber letztendlich gegenüber NLLS und sogar AML ab. Zudem leiden sämtliche auf LLS basierenden Algorithmen an zum Teil extrem großen Ausreißern. Der Maximalfehler beträgt bei LLS, LMS und RLSM beispielsweise 6605 m. Dieser entsteht aufgrund der nahezu gitterförmigen Ankeranordnung bei dreielementigen Ankerteilmengen, die eine schlechte Geometrie aufweisen, z. B. die fast kollinearen Anker 1, 5 und 6. Diesen Effekt scheint CluRoL durch seine eingebaute Ankerauswahl noch zu verstärken und so einige große Ausreißer mehr zu produzieren, denn hier ist der Maximalfehler sogar 9661 m groß. Dies führt letztendlich zu dem relativ großen MAE von 3,68 m. Aufgrund der geringeren Anzahl an Messungen fällt dieser Effekt bei Testlauf 3 mit 8,77 m besonders deutlich aus. Betrachtet man hingegen Testlauf 2, bei dem immer mehr als drei Anker zur Verfügung standen und auch sehr selten vier und fünf Anker vorhanden waren, so übertrifft CluRoL hier sehr deutlich die Leistung von LLS (ist fast doppelt so genau) und ist nach NLLS der zweitbeste der sieben Algorithmen.

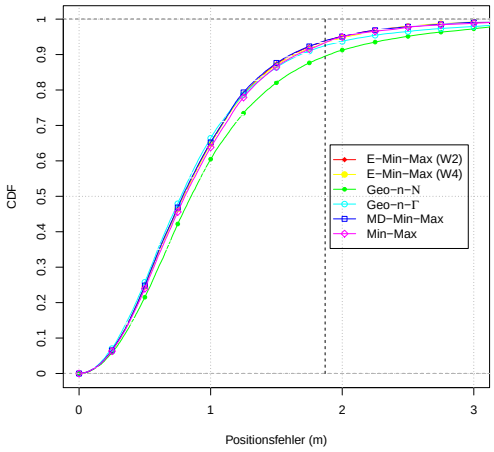
Als letzter Algorithmus liegt der MAE von VBLE mit 2,13 m über dem der Distanzmessungen. Die von uns verbesserte Variante VBLE-O erreicht ein deutlich besseres Ergebnis mit einem Fehler von 1,36 m (36,2 %). Die Fehlerverteilung von VBLE-O ist auch rechtsschief. Da der MAE von VBLE-O und MLE-$\mathcal{N}$ im Grenzbereich von 5 cm liegt, ist keine Aussage beim Vergleich dieser Algorithmen möglich. Von den geometrischen Algorithmen, die auf Schnittpunktbildung und Clustering basieren, schneiden BL (1,16 m) und Geo-n (1,08 m) am besten ab. ICLA erreicht mit 1,67 m nur einen durchschnittlichen Fehler, der ungefähr in der Größenordnung der Eingabe liegt. MLE-Γ hat mit 1,03 m eine Genauigkeit, die keinen sicheren Schluss zulässt, ob dieser Algorithmus besser ist als Geo-n. Abbildung 7.9b zeigt die CDFs dieser Algorithmen. Die Leistungsunterschiede von Geo-n und MLE-Γ sind auch hier nicht klar abgrenzbar. Die Anzahl der Positionsfehler, die kleiner als der Eingabefehler sind, liegt bei beiden deutlich über 85 %. Auf jeden Fall ist MLE-Γ im Gegensatz zu Geo-n wegen seiner Abhängigkeit von der Fehlerverteilung der Distanzmessungen



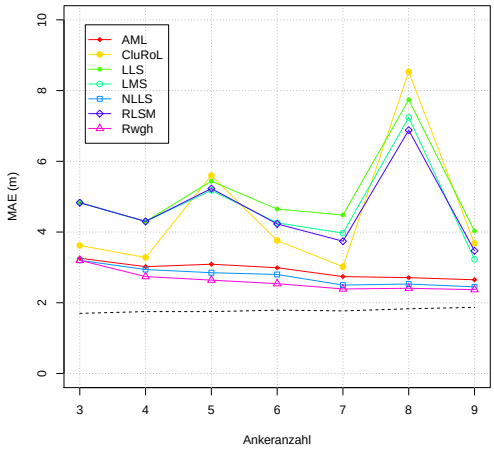
(a) CDF



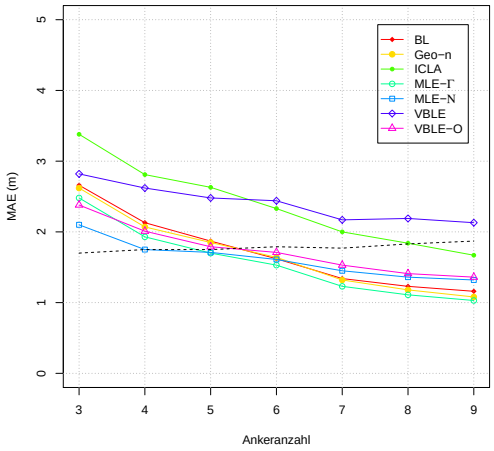
(b) CDF



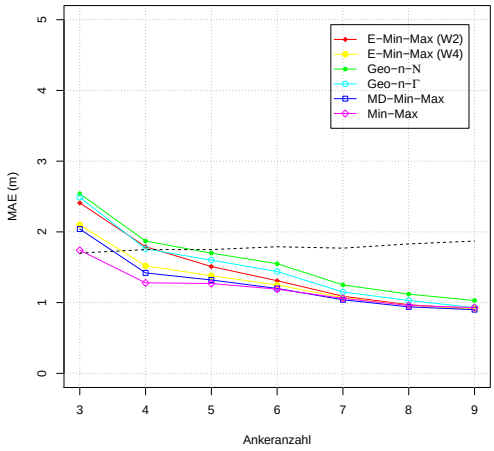
(c) CDF



(d) MAE per Ankeranzahl



(e) MAE per Ankeranzahl



(f) MAE per Ankeranzahl

Abbildung 7.9: Abbildungen 7.9a bis 7.9c zeigen die kumulative Verteilungsfunktion des Positionsfehlers der Algorithmen von Testlauf 1, Abbildungen 7.9d bis 7.9f den MAE der Algorithmen in Abhängigkeit der Ankeranzahl von Testlauf 1.

nicht universell einsetzbar.

Nur Min-Max mit 0,93 m sowie die auf diesem aufbauenden Verbesserungen E-Min-Max (W2) und E-Min-Max (W4) mit jeweils 0,91 m sowie MD-Min-Max mit 0,90 m zeigen noch bessere Genauigkeiten. Aufgrund der Anordnung der Anker in diesem Experiment war dies auch zu erwarten. Die Ankerabstände sind relativ gering und der mobile Knoten befand sich innerhalb der konvexen Hülle dieser Anker. Aus den Ergebnissen der simulativen Untersuchung mittels LS^2 in Kapitel 6 wissen wir, dass dies ein optimales Szenario für diese Algorithmen darstellt. Aber auch Geo-n-Γ, der ebenfalls die Fehlerverteilung der Distanzmessung in Betracht zieht, erreicht mit 0,93 m eine ebenbürtige Genauigkeit. Da die MAEs sich alle im Grenzbereich befinden, ist jedoch keine definitive Aussage möglich, welcher dieser fünf Algorithmen am besten ist. Betrachtet man die CDFs aus Abbildung 7.9c, so lässt sich auch hier kein Unterschied zwischen den einzelnen Algorithmen anhand der vorher festgelegten Kriterien ausmachen. Einzig Geo-n-$\mathcal{N}$ hat eine leicht schlechtere Fehlerverteilung von den sechs abgebildeten Algorithmen.

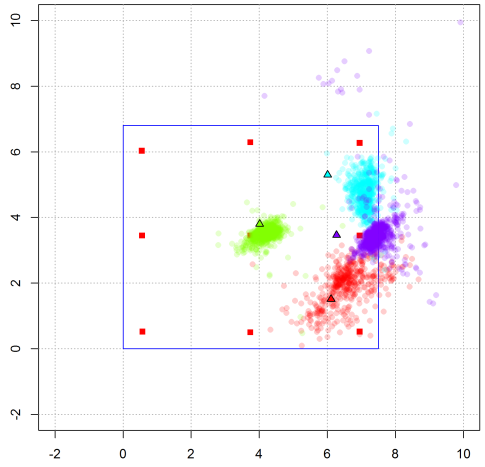
Der Vergleich der Positionsfehler von Testlauf 1 mit denen der Vergleichsläufe 2 und 3 bestätigt die bisherige Analyse. Die Ergebnisse bleiben für fast alle Algorithmen im Verhältnis identisch, werden in Testlauf 2 mit dem kleineren durchschnittlichen Distanzmessfehler von 1,64 m etwas besser und bei Testlauf 3 erneut schwächer. Einzige Ausnahme stellt CluRoL dar, dessen Verhalten bereits geschildert wurde. Darüber hinaus erwähnenswert ist die Anzahl der Situationen, in denen die Algorithmen keine Position berechnen konnten. Für LLS, LMS und RLSM ist diese Anzahl mit 6 im Vergleich mit der Gesamtanzahl an Lokalisierungen von Testlauf 1 von 29 109 verschwindend gering, für CluRoL steigt diese Anzahl jedoch auf 101 an. Dies lässt auf eine suboptimale Ankerauswahl schließen. AML und ICLA weisen ebenfalls jeweils 20 gescheiterte Lokalisierungen auf. Dies ist immer dann der Fall, wenn es keine Schnittpunkte zweier Distanzkreise gibt, z. B. wenn viele Distanzen zu kurz gemessen wurden.

Aufgrund der hohen Anzahl an Messungen und der durchschnittlich hohen Erfolgsquote von fast 90 % eignet sich Testlauf 1 gut, um das Verhalten der Algorithmen in Abhängigkeit der maximal verfügbaren Ankeranzahl zu untersuchen. Dazu wurde die Ankeranzahl aus Testlauf 1 schrittweise um einen Anker von drei auf neun Anker erhöht und der MAE der Algorithmen bestimmt. Das Ganze erfolgte in der Ankerreihenfolge 1, 6, 8, 3, 5, 7, 4, 2 und 9 entsprechend der Nummerierung aus

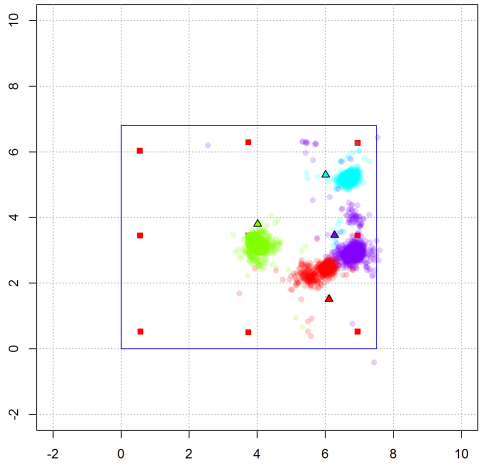
Abbildung 7.8a, damit ungünstige Geometrien durch die Reihenfolge ausgeschlossen sind. Wie die bisherige Untersuchung zeigte, entstehen dennoch ungünstige Geometrien, da nicht immer Messungen zu allen Ankern vorhanden sind. Abbildungen 7.9d bis 7.9f zeigen die Ergebnisse in der gleichen Algorithmenunterteilung wie bei den CDFs, damit die Abbildungen besser lesbar sind. Auch hier wurde der durchschnittliche Distanzmessfehler als gestrichelte schwarze Linie zum Vergleich eingezeichnet.

In Abbildung 7.9d sieht man noch einmal besonders gut, dass CluRoL am stärksten unter ungünstigen Ankerkonstellationen leidet, bei Nichtvorhandensein aber das beste Verbesserungspotenzial besitzt. So hat dieser bei drei und vier Ankern den niedrigsten MAE verglichen mit den anderen LLS Algorithmen, steigt aber sprunghaft bei Zunahme von Anker 5 an. LMS und RLSM liegen konstant unter den Ergebnissen von LLS, können diesen also immer moderat verbessern, wobei LMS erst bei neun Ankern an RLSM vorbeiziehen kann. AML, NLLS und Rwgh zeigen die größte Steigerung der Performance bei dem Wechsel von drei zu vier Ankern, danach ist nur noch eine langsame Leistungssteigerung um 12,3 %, 16,6 % bzw. 13,5 % möglich. Abbildung 7.9e zeigt, dass die geometrischen Algorithmen BL, Geo-n und ICLA sowie MLE-Γ die besten Performancezuwächse haben, wobei der Abstand zwischen diesen fast gleich bleibt. ICLA erzielt eine Steigerung von 50,6 %, BL eine von 56,4 %, MLE-Γ eine von 58,5 % und Geo-n die größte Steigerung von 58,8 %. Auffällig ist, dass MLE-$\mathcal{N}$ bei niedrigen Ankerzahlen von drei und vier der beste Algorithmus ist und schon bei vier Ankern einen MAE in Größe des Eingabefehlers besitzt. Danach ist aber keine vergleichbare Steigerung mehr möglich. BL, Geo-n, VBLE-O und MLE-Γ erreichen erst ab sechs Ankern eine bessere Leistung als der Eingabefehler. VBLE-O zeigt unter allen Bedingungen eine bessere Leistung als VBLE, wobei sich die Differenz mit steigender Ankeranzahl vergrößert. Abbildung 7.9f zeigt die Min-Max Algorithmen sowie die verbesserten Geo-n Varianten. Geo-n-Γ hat die insgesamt beste Leistungssteigerung mit 62,7 %. Alle dort gezeigten Algorithmen erreichen schon ab fünf Ankern eine bessere Leistung als der Eingabefehler, wobei die Differenz zwischen den Algorithmen bei großen Ankerzahlen (ca. ab sieben Ankern) verschwindend gering wird. Bis auf Geo-n-$\mathcal{N}$ erreichen alle Algorithmen sehr gute Genauigkeiten, der MAE der Positionen ist weniger als halb so groß wie der Eingabefehler.

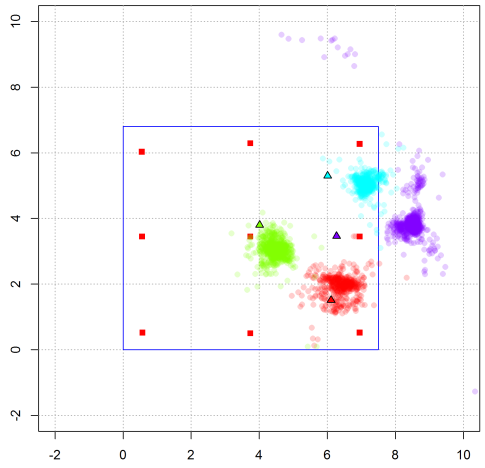
Abbildung 7.10 zeigt die berechneten Positionen für die vier festen Positionen aus Testlauf 4 für sechs ausgewählte Algorithmen. Zu sehen sind die Positionen der Anker in dem in Blau eingezeichneten Raum und die vier festen Positionen als kleine farbige



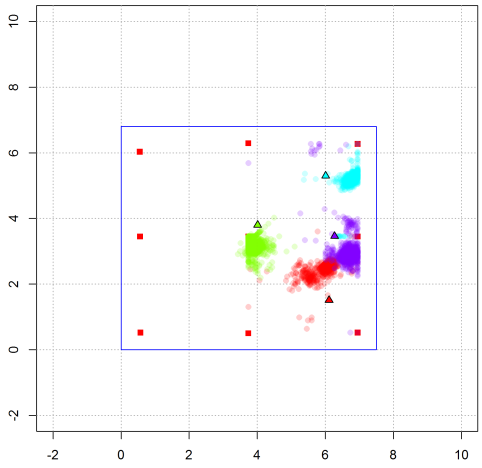
(a) Geo-n



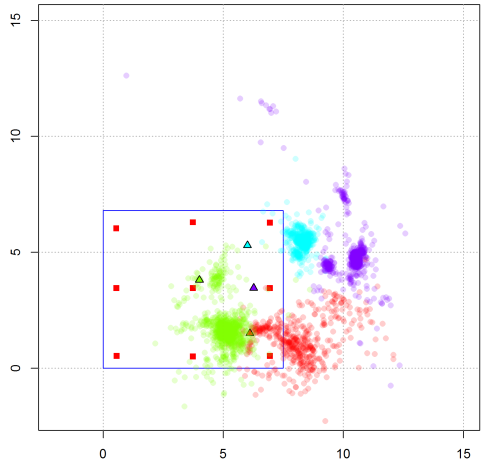
(b) MD-Min-Max



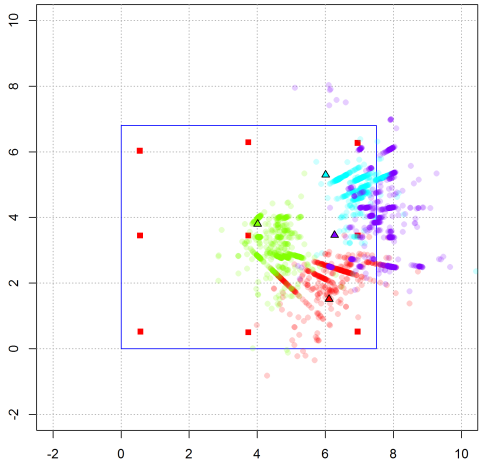
(c) MLE-Γ



(d) Min-Max



(e) NLLS



(f) VBLE-O

Abbildung 7.10: Berechnete Positionen für sechs ausgewählte Algorithmen für die vier festen Punkte aus Testlauf 4. Die echten Positionen sind als kleine Dreiecke dargestellt, die geschätzten Positionen als Punkte in der zum Dreieck passenden Farbe (alle Angaben in Metern).

Tabelle 7.6: Verzerrung und Varianz von Testlauf 4

Algorithmus	*A*		*B*		*C*		*D*	
	Bias	*Var*	*Bias*	*Var*	*Bias*	*Var*	*Bias*	*Var*
Geo-n	0,62	0,81	0,34	0,12	1,26	0,33	1,17	0,72
Min-Max	0,93	0,23	0,64	0,07	0,76	0,11	0,63	0,30
NLLS_{GN}	2,04	2,45	2,31	1,17	2,29	0,21	4,32	1,65
MD-Min-Max	0,89	0,21	0,63	0,09	0,69	0,10	0,70	0,35
MLE-Γ	0,49	0,28	0,89	0,13	1,18	0,10	2,25	0,76
VBLE-O	0,63	0,97	0,74	0,60	1,03	0,40	1,42	1,96

(Einheiten: *Bias* - m, *Var* - m^2)

Dreiecke. Hierbei sind Position A und die geschätzten Positionen in Rot dargestellt, Position B entsprechend in Grün, C in Hellblau und D in Violett. Die Ergebnisse für Testlauf 4 aus Tabelle 7.5 zeigen die durchschnittlichen Werte über die vier Positionen. In diesen Szenarien bleiben die Messfehler für die einzelnen Anker gesehen relativ konstant, was zum Teil zu großen Ausreißern führt und damit ein guter Test für die Robustheit der Algorithmen ist (vgl. Abbildung 7.8f). Die numerischen Ergebnisse fallen für die meisten Algorithmen wie erwartet aus - aufgrund des höheren durchschnittlichen Distanzmessfehlers steigen die MAEs der meisten Algorithmen entsprechend an, die Verhältnisse der Algorithmen untereinander bleiben aber größtenteils gleich. Einzig AML und die vier Algorithmen, die eine Normalverteilung oder Gammaverteilung nutzen, schneiden in manchen Szenarien schlechter ab, was deren MAEs aufgrund der geringen Anzahl an untersuchten Punkten nach oben steigen lässt. Dies gilt besonders für die Position D. Betrachtet man Abbildung 7.10c, so liegen die Positionen von MLE-Γ weiter rechts als z. B. bei Geo-n. Gleiches gilt - auch wenn nicht dargestellt - für MLE-$\mathcal{N}$ sowie Geo-n-$\mathcal{N}$ und Geo-n-Γ. Der normale Geo-n schneidet hier deutlich besser ab, die Positionen liegen näher an der echten Position. Min-Max erzielt die insgesamt besten Ergebnisse aller Algorithmen in diesem Test. Den Grund hierfür sieht man in Abbildung 7.10d. Min-Max kann algorithmenbedingt keine Positionen außerhalb der konvexen Hülle der Anker erzeugen, was für Position C und D mit den hellblauen und violetten Punkten deutlich wird. Das Muster der Streuung von VBLE-O und ebenfalls VBLE ist identisch mit den Ergebnissen der inversen Darstellung aus den Simulationen. Die Verzerrung von VBLE ist im Allgemeinen hierbei deutlich größer als die von VBLE-O.

Tabelle 7.6 zeigt diesen *Bias* für alle abgebildeten Algorithmen für die einzelnen Positionen. Der *Bias* wurde nach der in Abschnitt 4.2.3 eingeführten Definition berechnet, ist also die Differenz zwischen dem Erwartungswert der einzelnen Positionsschätzungen und der realen Position des Knotens. Alle Algorithmen haben je nach Position und Fehlerverteilung einen mehr oder weniger großen *Bias*, sind also in diesem Sinne keine erwartungstreuen Schätzer. Dies gilt auch für die restlichen Algorithmen, die nicht in der Tabelle vorhanden sind. Die kleinste Verzerrung weisen im Durchschnitt über alle vier Punkte sämtliche Min-Max Algorithmen auf, diese liegt hier zwischen 0,73 m und 0,86 m. Ebenfalls geringe durchschnittliche Verzerrungen erzielen Geo-n (0,85 m), VBLE-O (0,96 m) und BL (0,97 m). Auffällig ist noch Rwgh, der die Verzerrung von NLLS mit 2,74 m auf 1,41 m deutlich senken kann. Während BL und Geo-n geringe Streuungen aufweisen und damit MAEs von 1,12 m bzw. 0,98 m erzielen, so ergibt sich für Rwgh jedoch ein komplett anderes Verhalten. Hier ist die Varianz der Schätzungen stark vergrößert und damit fällt der MAE auf 2,45 m ab. Diese Funktionsweise des Algorithmus stimmt mit den Erkenntnissen aus den Simulationen überein. ICLA (1,11 m) und CluRoL (1,52 m) besitzen mittelgroße Verzerrungen, während AML (2,06 m), VBLE (2,37 m), LMS (2,72 m), NLLS_{LM} (2,73 m), NLLS_{GN} (2,74 m), RLSM (3,78 m) und LLS (4,42 m) große Verzerrungen aufweisen.

Im Allgemeinen deckt sich das in Abbildung 7.10 und Tabelle 7.6 gezeigte mit den Resultaten der inversen Darstellung von LS^2. So hat auch dort NLLS einen großen *Bias*, Geo-n weist geringe Verzerrungen auf und Min-Max sowie MD-Min-Max besitzen auch dort sehr geringe Verzerrungen innerhalb der konvexen Hülle der Anker. Für VBLE-O ist es schwer eine Aussage zu treffen. Die experimentellen Ergebnisse zeigen einen größeren *Bias* als die inverse Darstellung vermuten lässt. Allerdings fällt dort die untersuchte Position in einen der grünen Bereiche des Phasendiagramms, also in einen Bereich mit niedriger Verzerrung. Dies lässt sich für die Punkte aus dem Experiment nicht sagen, wenn auch der *Bias* hier noch als gering einzustufen ist.

7.4.3 Experiment 2

Als zweites Experiment haben wir einen Aufbau mit 21 Ankerknoten verwendet, der sich über eine Fläche von 32 m × 20 m erstreckte. In Abbildung 7.7 ist dies der gesamte blaue Bereich der Karte. Das Experiment wird im Folgenden als Experiment 2 bezeichnet und bestand aus zwei Testläufen. Abbildung 7.11 zeigt den zweiten, länge-

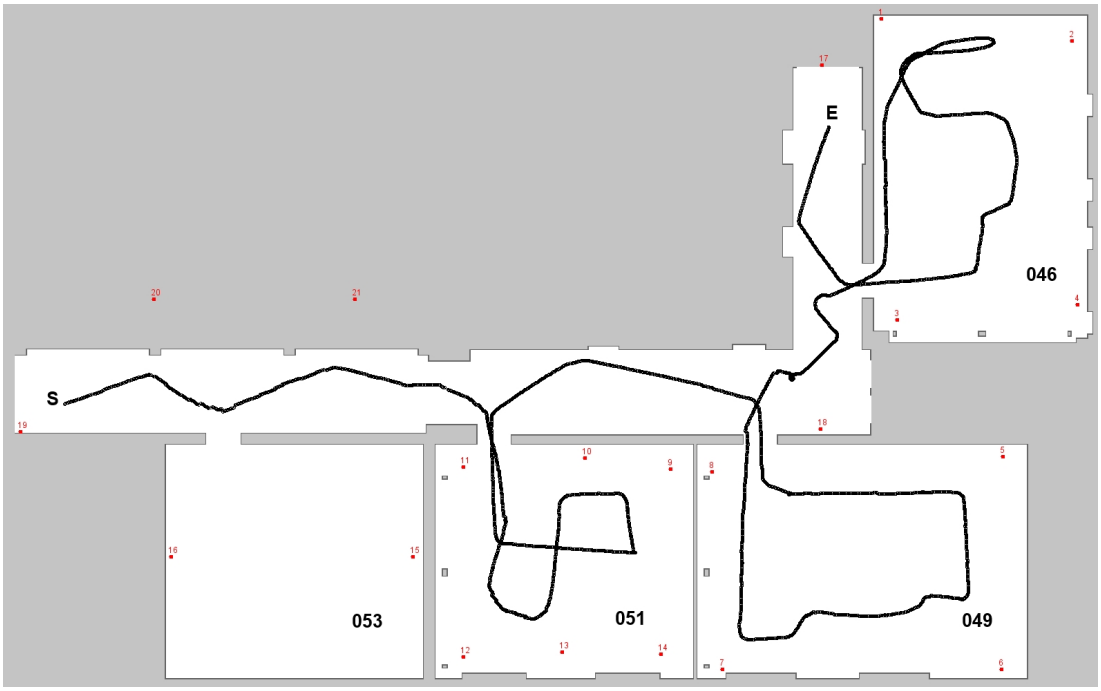


Abbildung 7.11: Testlauf 2 von Experiment 2 durch die Seminarräume 051, 049, 046 und den Flur mit insgesamt 21 Ankern (rote Punkte).

ren Testlauf durch das komplette Gelände. Start- und Endposition sind mit einem **S** bzw. **E** gekennzeichnet. Wie man sehen kann, ist die Ankerdichte in manchen Bereichen größer (z. B. um Raum 051) und in manchen Bereichen kleiner. Die Bewegung durch die Räume orientierte sich an den örtlichen Gegebenheiten, indem der Roboter Hindernissen wie Tischen und Stühlen ausweichen musste, was zu dem abgebildeten Pfad führte. Dieser Test entspricht also realen Bedingungen, es wurde in den Räumen nichts verändert. Der erste Testlauf fand mit der gleichen Ankerausbringung, jedoch ausschließlich auf dem Flur statt. Tabelle 7.7 zeigt die wichtigsten Daten zu diesen Testläufen.

Bei der Betrachtung der Daten fällt die deutlich niedrigere Erfolgsquote im Vergleich zu Experiment 1 auf und die daraus resultierende niedrigere durchschnittliche Ankeranzahl von 6,95 bzw. 6,85 und dies trotz der großen Ankerzahl und relativ geringer Ankerabstände. Dies liegt vor allem an den soliden Stahlbetonwänden der Räume im Erdgeschoss, welche eine Dicke von ca. 40 cm aufweisen. Hierdurch ist es kaum möglich, dass der mobile Knoten entfernt liegende Knoten aus Nachbarräumen empfängt. Insgesamt steigt der Durchschnittsfehler der Distanzmessungen um ca. einen halben Meter im Vergleich zu Experiment 1 an. Auch die Streuung des Fehlers ist deutlich

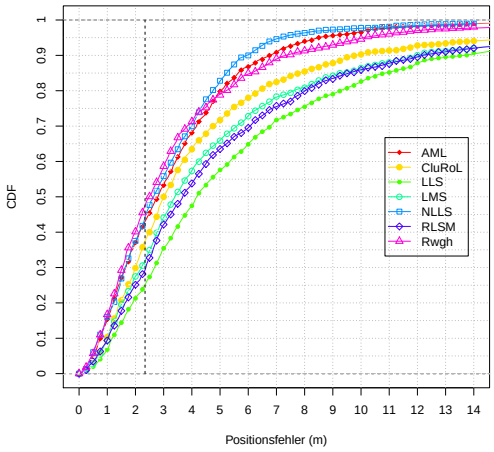
Tabelle 7.7: Testläufe und Metriken von Experiment 2

Metrik	Testlauf	
	1	2
Referenzpositionen	1441	4510
Distanzmessungen	30 261	94 710
Erfolgsquote	33,08 %	32,64 %
Durchschnittliche Ankeranzahl	6,95	6,85
Pfadlänge	33,74 m	107,68 m
Durchschnittsfehler*	2,34 m	2,22 m
Varianz	10,59 m^2	8,92 m^2
MSE	13,54 m^2	11,53 m^2
RMSE	3,68 m	3,40 m
Maximalfehler*	22,63 m	21,77 m
0,5 % Quantil	−2,38 m	−2,28 m
Modus	−0,03 m	−0,43 m
99,5 % Quantil	15,07 m	13,16 m

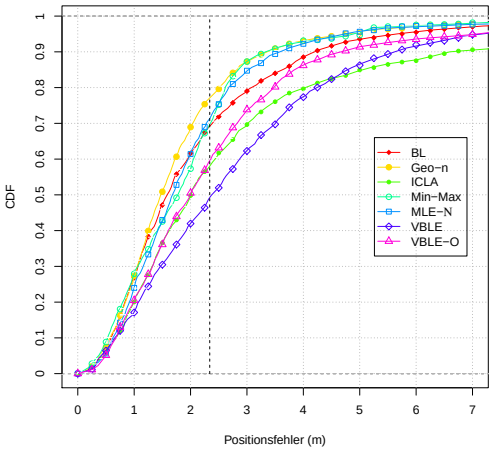
* absoluter Fehlerwert

größer, wobei Testlauf 1 aufgrund des potentiell höheren NLOS-Anteils (es gibt nur drei Anker, die direkt auf dem Flur stehen) auch die schlechtesten Werte aufweist.

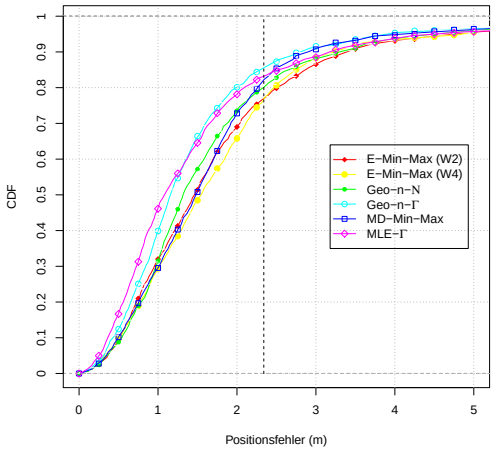
Betrachtet man die Ergebnisse beider Testläufe, welche in Tabelle 7.8 und Abbildung 7.12 dargestellt sind, so zeigt sich, dass sich die Verhältnisse zwischen den einzelnen Algorithmen nicht wesentlich verschoben haben. Die Ergebnisse fallen insgesamt aufgrund des höheren Distanzmessfehlers und der geringeren durchschnittlichen Ankerzahl für alle Algorithmen schlechter aus. Die Algorithmen sind in den Abbildungen auch hier absteigend nach ihrem MAE geordnet und Tabelle 7.8 enthält in diesem Fall noch zusätzlich den RMSE, um einen Eindruck über die Streuung der Positionsfehler zu erhalten. Unter den sieben Algorithmen mit dem größten MAE gab es keine Veränderung. All diese Algorithmen schneiden schlechter ab als der Durchschnittsfehler der Distanzmessungen mit 2,34 m bzw. 2,22 m. AML erzielt noch die besten Werte mit einem Fehler von 3,44 m bzw. 3,31 m. Im Vergleich mit Experiment 1 erzielt in beiden Testläufen CluRoL die größte Verbesserung von LLS, wobei LMS und RLSM je nach Testlauf den zweiten bzw. dritten Platz belegen. Rwgh ist unter den gegebenen Bedingungen zu keiner Zeit in der Lage, $NLLS_{GN}$ zu verbessern, verschlechtert



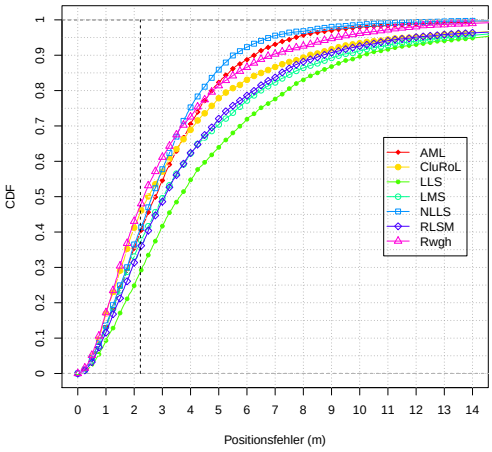
(a) CDF



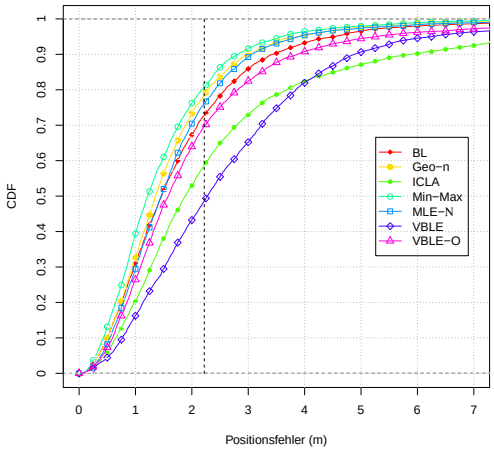
(b) CDF



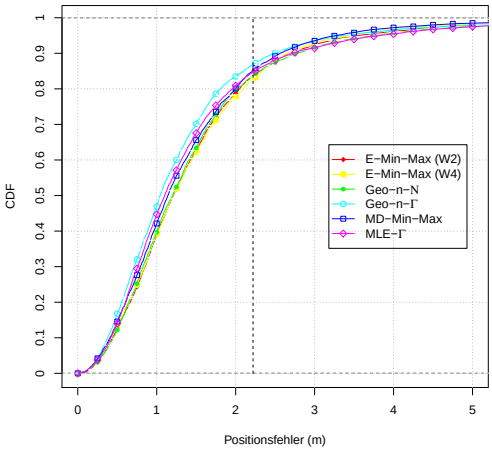
(c) CDF



(d) CDF



(e) CDF



(f) CDF

Abbildung 7.12: Abbildungen 7.12a bis 7.12c zeigen die kumulative Verteilungsfunktion des Positionsfehlers der Algorithmen von Testlauf 1, Abbildungen 7.12d bis 7.12f die Verteilung von Testlauf 2.

Tabelle 7.8: Ergebnisse von Experiment 2

Algorithmus	Testlauf 1			Testlauf 2		
	MAE	RMSE	MAX	MAE	RMSE	MAX
AML	3,44	4,46	26,68	3,31	4,16	24,99
BL	2,16	2,93	19,17	1,82	2,36	19,20
CluRoL	6,55	26,29	579	4,62	18,65	961
E-Min-Max (W2)	1,82	2,41	16,13	1,48	1,90	13,07
E-Min-Max (W4)	1,83	2,33	12,85	1,46	1,83	11,69
Geo-n	1,89	2,56	17,53	1,65	2,12	18,41
ICLA	3,07	4,56	27,45	2,74	3,88	23,55
LLS	7,40	20,40	434	5,79	17,46	792
LMS	6,59	20,11	434	5,22	17,28	792
Min-Max	1,96	2,45	13,70	1,51	1,90	11,32
NLLS_{GN}	3,19	4,07	20,42	3,03	3,72	20,74
NLLS_{LM}	3,21	4,10	18,08	2,99	3,67	20,80
RLSM	6,81	20,16	434	5,06	17,12	792
Rwgh	3,50	4,88	26,52	3,24	4,33	23,14
Geo-n-$\mathcal{N}$	1,79	2,55	19,11	1,51	2,01	18,70
Geo-n-Γ	1,57	2,28	18,84	1,36	1,87	18,85
MD-Min-Max	1,71	2,18	12,95	1,40	1,79	12,69
MLE-$\mathcal{N}$	2,08	2,83	16,96	1,74	2,23	18,73
MLE-Γ	1,64	2,57	18,61	1,46	2,06	18,56
VBLE	2,93	3,83	19,53	2,69	3,34	18,92
VBLE-O	2,55	3,44	17,40	2,06	2,77	23,72

(alle Angaben in Metern)

die Genauigkeit sogar von 3,19 m auf 3,50 m bzw. 3,03 m auf 3,24 m. In Prozenten beträgt die Verbesserung von CluRoL 15,5 % bzw. 20,2 %, für LMS 10,9 % bzw. 9,8 % und für RLSM 7,9 % bzw. 12,6 %. Vergleicht man MAE und RMSE, so leidet die Genauigkeit aller LLS Algorithmen unter der starken Streuung des Positionsfehlers. Auch bei Rwgh ist diese im Verhältnis zu NLLS_{GN} erhöht.

Zu Rwgh und RLSM muss noch angemerkt werden, dass die Anzahl der Anker und Distanzmessungen, die an den Algorithmus übergeben wurden, aufgrund der hohen Laufzeit künstlich beschränkt werden musste. Für Rwgh wurden 15 Anker und die gemessenen Distanzen zufällig ausgewählt, für RLSM musste die Anzahl sogar auf 12 herabgesetzt werden. Situationen mit mehr als 15 Ankern treten zwar nur in vernachlässigbarer Größe auf, so dass das durchschnittliche Ergebnis davon relativ unberührt

bleiben sollte, in der Realität wird man bei diesen Algorithmen um eine Beschränkung aber auch nicht herumkommen. So braucht ein einzelner Aufruf von Rwgh mit 19 Ankern auf einer modernen Workstation[1] beispielsweise ungefähr 25 Sekunden, für RLSM liegt eine Berechnung aufgrund des benötigten Speicherbedarfs völlig außerhalb einer sinnvollen Anwendung. Die Begrenzung wurde für beide Algorithmen letztlich so eingestellt, dass ein Aufruf maximal eine Sekunde benötigt.

Von den geometrischen Algorithmen die Schnittpunkte clustern, schneidet Geo-n mit 1,89 m bzw. 1,65 m am besten ab, gefolgt von BL mit 2,16 m bzw. 1,82 m und ICLA mit 3,07 m bzw. 2,74 m. Nur der MAE von ICLA liegt in beiden Fällen über dem Eingabefehler. Dies stimmt mit den Ergebnissen aus Abbildung 7.9e überein. Für Ankeranzahlen von sechs und sieben liegt die Genauigkeit von ICLA noch deutlich über der gestrichelten schwarzen Linie, die den Eingabefehler zeigt. VBLE-O ist immer noch in der Lage, VBLE zu verbessern, auch wenn die Steigerung der Performance nicht mehr ganz so groß ist wie in Experiment 1. In Testlauf 2 beträgt diese noch 23,4 % während diese in Testlauf 1 nur noch bei 12,9 % liegt. In Abbildung 7.12b sieht man auch, dass die CDF-Kurven dichter beieinander liegen als in Abbildung 7.12e. Eine mögliche Erklärung hierfür liefert die räumliche Fehlerverteilung aus Kapitel 6. Diese hat gezeigt, dass VBLE-O außerhalb der konvexen Hülle der Anker schlechtere Ergebnisse liefert als innerhalb. Dieser Umstand ist erstmals in Experiment 2 der Fall und zwar stärker bei Testlauf 1 als bei Testlauf 2, da dieser nur auf dem Flur stattfand. Betrachtet man Abbildung 7.11, so befinden sich z. B. vor Raum 051 auf dem Flur Bereiche, die außerhalb der sich in Reichweite befindenden Anker liegen (Anker aus Raum 046 und davor werden selten bis gar nicht empfangen). Unter der gleichen Tatsache leidet auch Min-Max. Auch wenn der Aufbau in Experiment 2 noch kein schlechtes Szenario für Min-Max darstellt, so nimmt die Leistung doch ab und liegt in Testlauf 2 mit einem MAE von 1,51 m zum ersten Mal am äußersten Ende des Grenzbereichs von 15 cm, an dem Min-Max als schlechter einzustufen wäre als Geo-n.

E-Min-Max (W2) und E-Min-Max (W4) können die Leistung von Min-Max zwar noch steigern (7,1 % bzw. 6,6 %), sind aber in Testlauf 1 mit sechs bzw. sieben Zentimetern nicht klar besser als Geo-n. D. h. auch hier scheint die Lokalisierung außerhalb der konvexen Hülle der Anker trotz Verbesserung problematisch zu bleiben. MD-Min-Max erzielt mit 1,71 m (12,8 %) erneut die besten Ergebnisse der Algorithmen dieser Gruppe, gleiches gilt für den zweiten Testlauf. Von den restlichen verteilungsabhängigen

[1] Intel(R) Core(TM) i5-3570K CPU mit 3,40 GHz und 16 GB Arbeitsspeicher.

Algorithmen weisen MLE-Γ mit 1,64 m bzw. 1,46 m und Geo-n-Γ mit 1,57 m bzw. 1,36 m die höchste Genauigkeit der Lokalisierung auf, wobei das Niveau in beiden Testläufen keine klare Benennung eines überlegenen Algorithmus zulässt. Geo-n-Γ ist in der Lage die Performance von Geo-n noch einmal um 16,9 % bzw. 17,6 % zu steigern. Dies unterstützt die aufgestellte Vermutung aus Abschnitt 7.3, dass die verteilungsabhängigen Algorithmen leicht auf andere Szenarien übertragbar sind und diese eine effektive Verbesserung erzielen. Während die CDF-Kurven der sechs besten Algorithmen von Experiment 1 (vgl. Abbildung 7.9c) kaum zu unterscheiden waren, so zeigen sich hier bei Abbildung 7.12f und besonders bei Abbildung 7.12c schon wahrnehmbare Unterschiede. Geo-n-Γ weist neben MLE-Γ die insgesamt größte Präzision auf, wobei die Fehlerverteilung von MLE-Γ rechtsschief ist.

Auch in diesem Experiment haben wir das Verhalten der Algorithmen an festen Positionen untersucht. Hierfür wurden fünf Postionen verwendet, welche sich im Flur und in den Räumen befanden und jeweils 480 Messungen ausgeführt. Der Distanzmessfehler betrug im Schnitt 2,30 m. Wir möchten uns hier auf eine kurze Beschreibung der Beobachtungen beschränken, da sich die Ergebnisse im Wesentlichen mit denen aus Experiment 1 decken. Alle Algorithmen haben auch hier je nach Position und Fehlerverteilung der Eingabe einen mehr oder weniger großen *Bias*, sind also in diesem Sinne keine erwartungstreuen Schätzer. Die kleinste durchschnittliche Verzerrung weist MLE-$\mathcal{N}$ mit 0,89 m auf. Darauf folgen sämtliche Min-Max Varianten: E-Min-Max (W2) mit 1,10 m, Min-Max mit 1,11 m, MD-Min-Max mit 1,14 m und E-Min-Max (W4) mit 1,15 m. Auch niedrige Verzerrungen haben noch VBLE-O mit 1,25 m, BL mit 1,36 m und Geo-n mit 1,57 m. Beide Geo-n Varianten schaffen es diesmal den *Bias* auf 1,26 m zu senken und so den MAE von 1,77 m auf 1,48 m zu verbessern. Auch Rwgh kann erneut die Verzerrung von NLLS von 2,33 m auf 1,85 m absenken, führt dabei aber wieder eine große Varianz ein und produziert hierdurch einen um einen Meter größeren MAE. Wie in Experiment 1 kann RLSM die Verzerrung und die Varianz von LLS simultan absenken und so einen 1 Meter kleineren MAE erzielen. Die größten Verzerrungen besitzen ICLA (2,38 m), CluRoL (2,74 m) und LMS (2,91 m). LLS hat mit 2,18 m nicht mehr die größte Verzerrung, in diesem Szenario aber eine deutlich höhere Varianz und landet mit einem MAE von 4,09 m wieder am hinteren Ende des Feldes.

Die Ergebnisse zeigen, dass eine endgültige Beurteilung der Algorithmen anhand der durchgeführten Untersuchungen mit festen Positionen schwierig ist. Zum einen müsste die Anzahl der Positionen deutlich gesteigert werden, um belastbarere Aussagen

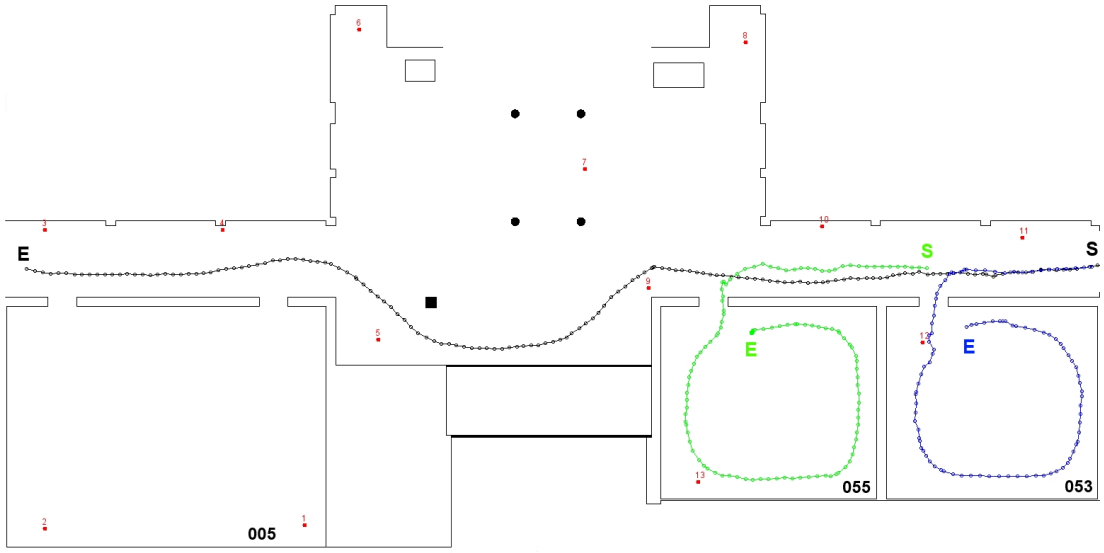


Abbildung 7.13: Testlauf 1 (schwarz), Testlauf 2 (grün) und Testlauf 3 (blau) von Experiment 3 durch die Seminarräume 053, 055, den Flur und das Foyer mit insgesamt 13 Ankern (rot).

für den durchschnittlichen Fall geben zu können - so es diesen gibt. Denn die bisherigen Resultate haben gezeigt, dass für eine bestimmte Konstellation aus Ankern und Knotenposition sich immer unterschiedliche Reihenfolgen der Algorithmen für die kleinste Verzerrung ergeben haben. Dennoch lassen sich aus den Untersuchungen allgemeine Erkenntnisse über die Arbeitsweise der Algorithmen ziehen und Tendenzen ablesen. So sind sämtliche evaluierten Algorithmen unter den realen Bedingungen in so gut wie allen Fällen keine erwartungstreuen Schätzer, was einen Vergleich mit der CRLB obsolet macht, da diese für diese Klasse von Schätzern ausdrücklich keine untere Schranke für die Effizienz darstellt.

7.4.4 Experiment 3

Auch das dritte Experiment fand im Erdgeschoss unseres Bürogebäudes statt. Bei diesem Aufbau, der sich über eine Fläche von 38,5 m × 20 m erstreckte, wurden 13 Ankerknoten verwendet. In Abbildung 7.7 ist dies der gesamte rote Bereich der Karte. Die Fläche ist also leicht größer als in Experiment 2 und die Ankerzahl fast halbiert. Folglich ergeben sich auch größere durchschnittliche Abstände zwischen den Ankern. Das Experiment wird im Folgenden als Experiment 3 bezeichnet und bestand aus

Tabelle 7.9: Testläufe und Metriken von Experiment 3

Metrik	Testlauf		
	1	2	3
Referenzpositionen	3667	2394	2025
Distanzmessungen	47 671	31 122	26 325
Erfolgsquote	82,04 %	74,49 %	61,55 %
Durchschnittliche Ankeranzahl	10,66	9,68	8,00
Pfadlänge	40,03 m	27,76 m	25,23 m
Durchschnittsfehler*	2,36 m	2,44 m	2,45 m
Varianz	12,93 m^2	9,90 m^2	11,36 m^2
MSE	15,62 m^2	14,02 m^2	15,73 m^2
RMSE	3,95 m	3,74 m	3,97 m
Maximalfehler*	31,74 m	25,96 m	30,37 m
0,5 % Quantil	−2,57 m	−2,02 m	−1,87 m
Modus	−0,38 m	−0,45 m	−0,26 m
99,5 % Quantil	18,22 m	14,61 m	17,62 m

* absoluter Fehlerwert

drei Testläufen. Abbildung 7.13 zeigt diese Testläufe, wobei Testlauf 1 in Schwarz, Testlauf 2 in Grün und Testlauf 3 in Blau gekennzeichnet sind. Start- und Endposition sind mit einem **S** bzw. **E** gekennzeichnet (Testlauf 1 und 3 haben den gleichen Startpunkt) und die Anker in Rot. Tabelle 7.9 zeigt die wichtigsten Daten zu diesen Testläufen.

Die Erfolgsquote der Distanzmessungen ist bei diesem Experiment wieder deutlich höher als beim vorherigen Experiment. Besonders der erste Testlauf erreicht mit 82,04 % sehr gute Werte. Sämtliche Anker in den Fluren und im Foyer werden in über 90 % der Fälle empfangen. Die anderen Testläufe fallen mit 74,49 % und 61,55 % sukzessive ab, da in den Räumen der Empfang von entfernt liegenden Ankern eingeschränkt ist. So werden bei Testlauf 3 beispielsweise Anker 1 und 2 gar nicht empfangen und Anker 3 und 4 nur in 45 % der Fälle. Hieraus ergeben sich durchschnittliche Ankeranzahlen zwischen 8,00 und 10,66. Der Durchschnittsfehler der Distanzmessungen ist mit 2,36 m, 2,44 m und 2,45 m nur unwesentlich größer als bei Experiment 2. Dafür ist die Streuung des Fehlers noch einmal größer geworden sowie auch der Maximalfehler mit über 30 Metern. Bei diesem Experiment war die Abtastrate nicht auf 4 Hz begrenzt und somit ergeben sich trotz kürzerer Pfade eine größere Anzahl an Messungen im

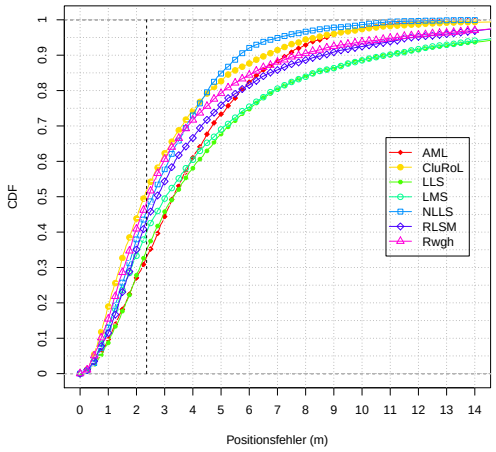
Tabelle 7.10: Ergebnisse von Experiment 3

Algorithmus	Testlauf 1		Testlauf 2		Testlauf 3	
	MAE	MAX	MAE	MAX	MAE	MAX
AML	3,82	35,00	4,11	32,71	4,66	32,87
BL	2,51	32,88	1,83	32,59	2,98	32,79
CluRoL	3,07	19,48	3,42	30,68	4,36	266
E-Min-Max (W2)	1,51	23,26	1,50	12,76	2,00	14,46
E-Min-Max (W4)	1,50	19,09	1,56	10,90	2,46	14,81
Geo-n	1,37	16,45	1,55	30,51	1,70	24,06
ICLA	1,89	27,19	2,37	32,59	3,01	41,65
LLS	5,06	57,09	3,88	38,73	6,37	66,66
LMS	4,87	57,09	3,54	38,73	6,21	66,66
Min-Max	1,56	14,40	1,72	13,53	3,35	15,40
NLLS_{GN}	3,05	30,73	3,20	32,70	3,47	32,55
NLLS_{LM}	3,00	17,74	3,24	23,79	3,51	32,55
RLSM	3,97	36,33	3,18	35,56	5,67	66,16
Rwgh	3,63	34,19	3,88	40,43	4,39	41,82
Geo-n-$\mathcal{N}$	1,49	27,68	1,48	31,39	1,69	21,20
Geo-n-Γ	1,35	26,72	1,33	31,99	1,53	22,59
MD-Min-Max	1,48	18,04	1,55	17,49	2,58	15,60
MLE-$\mathcal{N}$	2,05	30,09	1,82	30,31	2,26	30,09
MLE-Γ	1,49	28,28	1,38	32,05	1,64	19,01
VBLE	2,79	24,42	3,04	24,88	3,17	15,87
VBLE-O	2,37	28,00	2,14	30,14	2,64	28,23

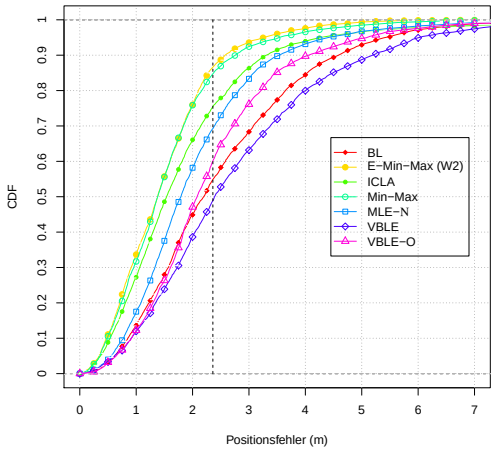
(alle Angaben in Metern)

Vergleich mit beispielsweise Experiment 2. Da es nur 13 Anker bei diesem Experiment gab, musste die Ankerzahl nur für RLSM auf zwölf beschränkt werden, was aber sehr selten der Fall war.

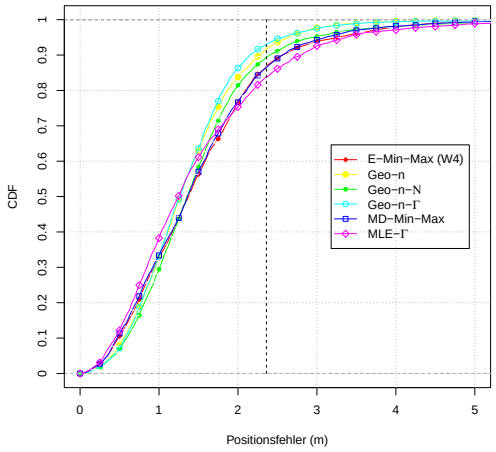
Betrachtet man die Ergebnisse der Testläufe, welche in Tabelle 7.10 und Abbildung 7.14 dargestellt sind, so zeigt sich, dass sich die Verhältnisse zwischen den Algorithmen bei diesem Experiment schon teilweise verschoben haben. Dies liegt zum Hauptteil an der irregulären Geometrie der Ankerpositionen, die am ehesten mit den *kite* und *tube* Simulationen aus Kapitel 6 zu vergleichen sind. Der Durchschnittsfehler der Distanzmessungen ist nur leicht größer als in Experiment 2, dafür sind aber im Schnitt auch zwei bis drei Anker mehr vorhanden, was für alle Algorithmen eigentlich



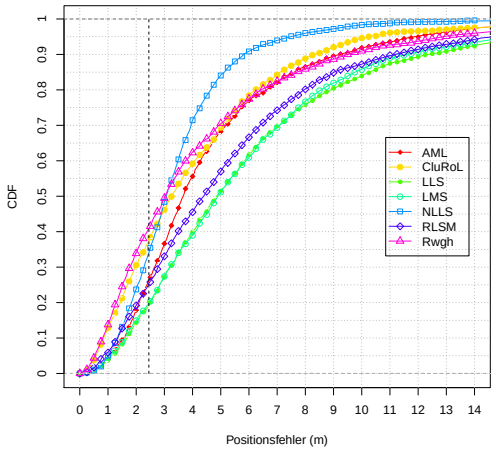
(a) CDF



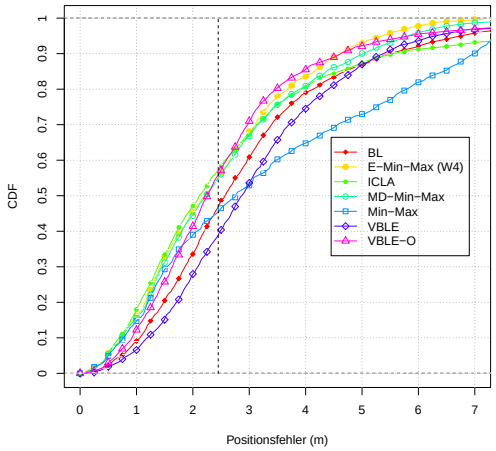
(b) CDF



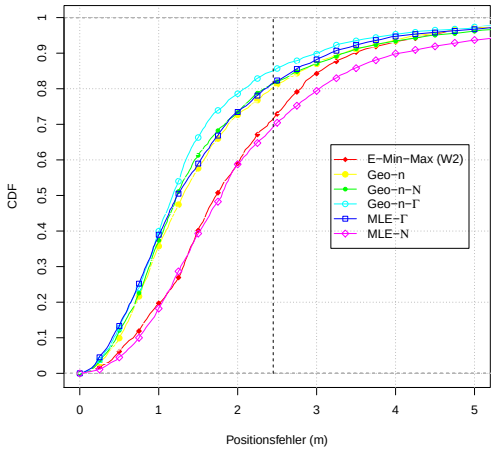
(c) CDF



(d) CDF



(e) CDF



(f) CDF

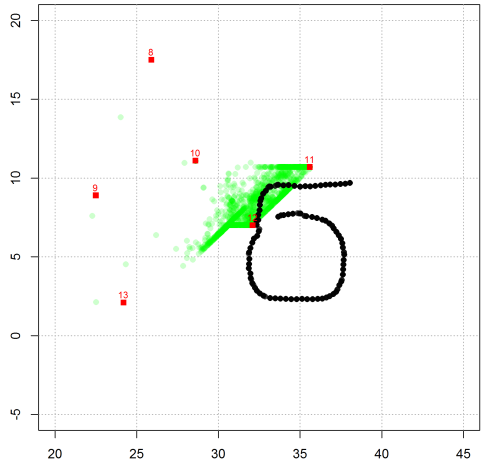
Abbildung 7.14: Abbildungen 7.14a bis 7.14c zeigen die kumulative Verteilungsfunktion des Positionsfehlers der Algorithmen von Testlauf 1, Abbildungen 7.14d bis 7.14f die Verteilung von Testlauf 3.

eine Verbesserung nach sich ziehen sollte, zieht man Abbildungen 7.9d bis 7.9f in Betracht. Vergleicht man Testlauf 1 von Experiment 2 und Testlauf 1 von Experiment 3, so ergibt sich aber ein geteiltes Bild: viele Algorithmen verbessern ihre Leistung, besonders zu erwähnen sind hier die LLS Algorithmen, doch Algorithmen wie AML, Rwgh und BL schneiden schlechter ab. Für die auf LLS basierenden Algorithmen ist die Erklärung einfach. Durch die irreguläre Ankerkonstellation bei gleichzeitig hoher Erfolgsquote sinkt die Wahrscheinlichkeit einer schlechter Ankerauswahl für diese Algorithmen stark ab und so werden weniger Ausreißer produziert, was sich in einem niedrigeren MAE niederschlägt. Dies kann man an den CDF-Kurven der ersten sieben Algorithmen in Abbildung 7.14a sehen. Diese liegen hier deutlich näher beieinander, d. h. die Kurve von beispielsweise LLS steigt hier steiler an. Auch die maximalen Fehler der Algorithmen sind durchgängig besser (maximal 57 m im Vergleich zu 579 m). Rwgh (3,63 m, 3,88 m, 4,39 m) verschlechtert auch hier in allen Testläufen die Ergebnisse von NLLS_{GN} (3,05 m, 3,20 m, 3,47 m), anstatt sie zu verbessern. Besonders stark in Testlauf 3, wo sich der mobile Knoten mehrheitlich außerhalb der konvexen Hülle der Anker bewegte. Insgesamt leidet der Algorithmus an einer deutlich höheren Streuung der Positionsfehler als NLLS. Die Performance beider NLLS Varianten ist auch hier nahezu identisch, das Ergebnis der Optimierung mit der echten Position als Startwert erreicht einen Fehler von 2,95 m, 3,09 m bzw. 3,36 m und ist damit nur minimal besser. CluRoL, LMS und RLSM steigern die Genauigkeit von LLS auch hier und zwar um 39,3 %, 3,6 % und 21,5 % für Testlauf 1 und unter den extremeren Bedingungen von Testlauf 3 noch um 31,6 %, 2,5 % und 10,9 %. Die Leistungssteigerung von CluRoL reicht bei Testlauf 1 sogar an die Genauigkeit von NLLS heran, RLSM ist in Testlauf 2 ebenbürtig, so dass für beide Algorithmen im Vergleich mit NLLS hier kein eindeutiger Sieger hervorgeht.

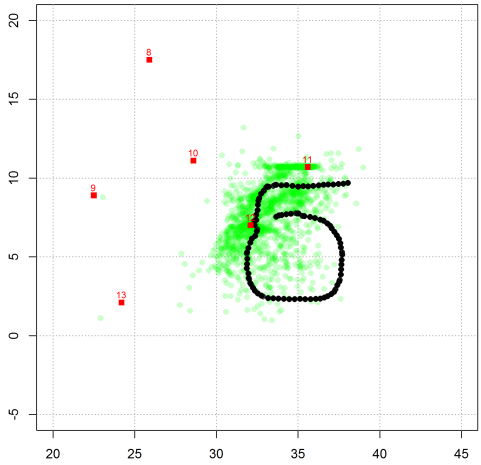
Für die Beurteilung von AML, BL und Rwgh muss die räumliche Fehlerverteilung betrachtet werden. Alle Algorithmen verschlechtern ihre Leistung beim Übergang von regulären Aufbauten wie dem *grid* Szenario zu ungleichmäßigen Anordnungen, besonders im Bereich der konvexen Hülle der Ankerknoten. Dies wird vor allem bei BL deutlich, der mit 1,14 m einen merklich größeren Abstand zu Geo-n aufweist als bisher. Außerhalb der Anker bzw. im Übergangsbereich ist das Verhalten von BL zweigeteilt. Wie die räumliche Fehlerverteilung beim *tube* Szenario mit *ab-nlos* Fehlermodell zeigt, gibt es überwiegend graue Bereiche, aber auch kleinere grüne, also gute Stellen. Dies lässt sich auch in der Realität nachvollziehen. Während BL in Testlauf 2 mit 1,83 m besser als der Eingabefehler abschneidet und seinen bisher üblichen

Abstand zu Geo-n mit ca. 30 cm einnimmt, so ergibt sich für Testlauf 3 das Gegenteil. Hier erzielt BL nur eine Genauigkeit von 2,98 m und liegt 1,28 m hinter Geo-n und damit deutlich über dem Eingabefehler. ICLA verhält sich bei diesem Experiment besser als die Simulationen erwarten lassen. Dieser erzielt mit 1,89 m eine bessere Genauigkeit als der Eingabefehler. Betrachtet man die räumliche Fehlerverteilung, dann sind die Bereiche um und zwischen den Ankern zwar besser als außerhalb, jedoch nicht grün und damit größer als der Eingabefehler. Mit zunehmender Ankerentfernung von Testlauf 2 und schließlich Testlauf 3 fällt die Performance sukzessive ab und gleicht den Ergebnissen von LS^2 wieder.

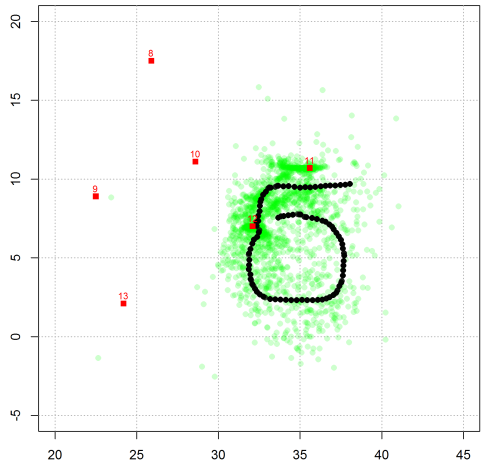
Während Min-Max von der günstigen Ankerkonstellation der vorherigen Experimente überproportional profitieren konnte, schlägt sich dies nun ins Gegenteil um. In Testlauf 1 wirken sich die größeren durchschnittlichen Ankerabstände bereits negativ aus und Min-Max fällt mit 1,56 m messbar hinter Geo-n mit 1,37 m zurück. Besonders bei Testlauf 3 werden die Schwächen von Min-Max deutlich, da sich der mobile Knoten in dem Raum zunehmend von der konvexen Hülle der Anker entfernt. Dies sieht man auch in Abbildung 7.14e. Dort steigt die CDF-Kurve ab ca. 3,50 m linear an, weist also eine extrem schlechte Fehlerverteilung auf. Die anderen Min-Max Varianten können die Ergebnisse von Min-Max zum Teil deutlich verbessern. In diesem Experiment kann man die Stärken und Schwächen der einzelnen Varianten auch insgesamt besser erkennen als bisher. Während E-Min-Max (W4) und im Besonderen MD-Min-Max ihre Stärken innerhalb der konvexen Hülle der Anker haben, so erzielt E-Min-Max (W2) die insgesamt besten Ergebnisse, auch an Positionen außerhalb der Anker. Dies spiegeln zum einen die numerischen Werte aus Tabelle 7.10 wider, zum anderen wird dies aber an der visuellen Darstellung der geschätzten Positionen in Abbildung 7.15 für Testlauf 3 deutlich. Die einzelnen Abbildungen zeigen die wirklich gefahrene Strecke des mobilen Knotens in Schwarz und die geschätzten Positionen für die einzelnen Messpunkte in Grün. Zusätzlich sind noch die in dem Ausschnitt vorhanden Anker inklusive Nummer in Rot dargestellt. Jedes Bild entspricht einem Algorithmus. So kann man die Arbeitsweise der vier Min-Max Algorithmen gut erkennen. NLLS und Geo-n wurden zum Vergleich ebenfalls hinzugefügt. Die Beschränkung der Positionen auf die konvexe Hülle kann man für Min-Max schön erkennen. E-Min-Max (W4) und MD-Min-Max können diesen Nachteil schon sichtbar mindern, wobei E-Min-Max (W2) am ehesten der Verteilung von Geo-n gleicht. Diese liegen mit einem MAE von 2,00 m (E-Min-Max (W2)) und 1,70 m (Geo-n) auch nicht sehr



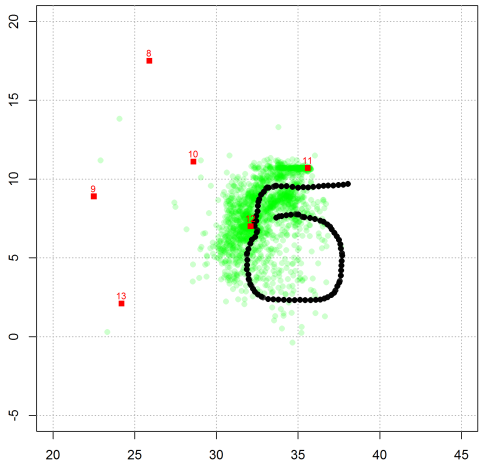
(a) Min-Max



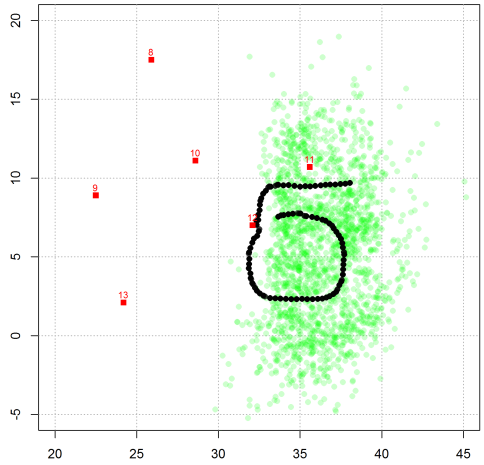
(b) E-Min-Max (W4)



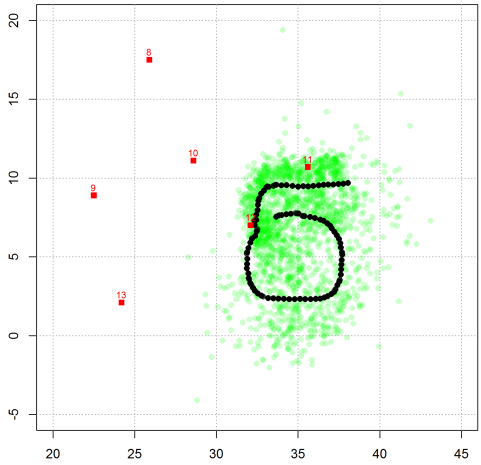
(c) E-Min-Max (W2)



(d) MD-Min-Max



(e) NLLS



(f) Geo-n

Abbildung 7.15: Berechnete Positionen für sechs ausgewählte Algorithmen für den dritten Testlauf. Die echten Positionen sind als schwarze Punkte dargestellt, die geschätzten Positionen als grüne Punkte und Anker in Rot (alle Angaben in Metern).

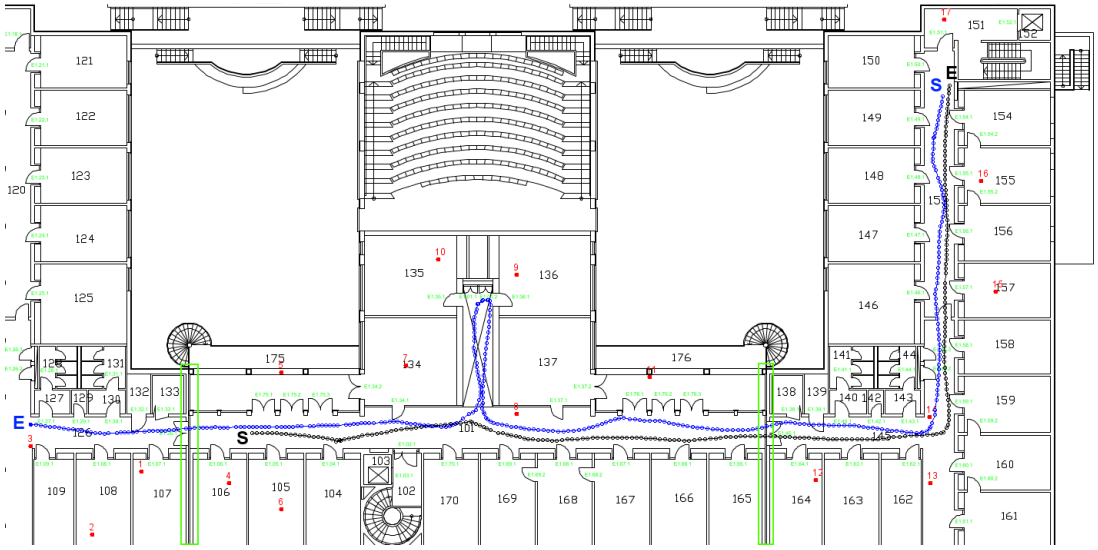


Abbildung 7.16: Gebäudeplan vom Obergeschoss des Instituts für Informatik der Freien Universität Berlin. Die gezeigte Karte spannt eine rechteckige Fläche von ungefähr 71,5 m × 37 m auf. Testlauf 1 von Experiment 4 ist in Schwarz dargestellt, Testlauf 2 in Blau. Insgesamt sind 17 Anker (rot) vorhanden.

weit auseinander.

VBLE-O ist in allen Testläufen in der Lage, die Genauigkeit von VBLE zu steigern (15,1 %, 29,6 % bzw. 16,7 %). Während die Leistung von VBLE kontinuierlich mit steigender Testlaufnummer abnimmt, so schwanken die Ergebnisse bei VBLE-O. Beides deckt sich mit den Ergebnissen von LS^2. Je weiter man von den Ankern entfernt liegt, desto schlechter wird VBLE, während das Bild von VBLE-O unterschiedlich große grüne „Inseln“ aufweist. Somit hängen die Ergebnisse davon ab, in welchen Bereich man gerade fällt. Geo-n erzielt in Testlauf 1 und 3 mit 1,37 m und 1,70 m die besten Genauigkeiten aller Universalalgorithmen. Besonders die Algorithmen, die eine Gammaverteilung verwenden, haben das größte Verbesserungspotenzial. So liefert Geo-n-Γ vor allem in den Grenzbereichen (zweiter und dritter Testlauf) die größte Verbesserung mit 14,2 % und 10,0 % sowie die höchste Genauigkeit der Lokalisierung. Die Anzahl der Positionsfehler, die kleiner als der Eingabefehler sind, liegt bei allen Geo-n Varianten bei Testlauf 1 zwischen 88 % und 92 %, bei Testlauf 3 zwischen 81 % und 85 %, was für eine sehr hohe Präzision dieser Algorithmen spricht (vgl. Abbildungen 7.14c und 7.14f).

Tabelle 7.11: Testläufe und Metriken von Experiment 4

Metrik	Testlauf	
	1	2
Referenzpositionen	2334	3043
Distanzmessungen	39 678	51 731
Erfolgsquote	43,54 %	44,27 %
Durchschnittliche Ankeranzahl	7,40	7,53
Pfadlänge	71,61 m	100,44 m
Durchschnittsfehler*	2,65 m	2,85 m
Varianz	13,80 m^2	12,77 m^2
MSE	18,66 m^2	18,66 m^2
RMSE	4,32 m	4,32 m
Maximalfehler*	63,39 m	74,18 m
0,5 % Quantil	−2,51 m	−2,16 m
Modus	0,074 m	−0,349 m
99,5 % Quantil	18,98 m	16,005 m

* absoluter Fehlerwert

7.4.5 Experiment 4

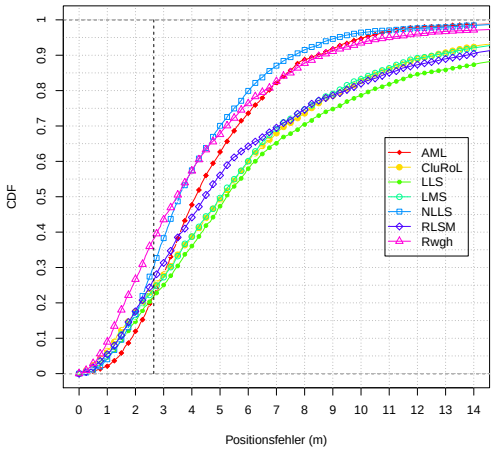
Das vierte und letzte Experiment fand im Obergeschoss unseres Bürogebäudes statt. Dieses weist eine andere Gebäudestruktur als bisher auf. Hier sind vor allem viele kleinere Büroräume vorhanden, welche im Gegensatz zu den massiven Wänden im Erdgeschoss hauptsächlich aus Trockenbauwänden konstruiert sind. Einzig die beiden in Abbildung 7.16 grün umrandeten Feuerschutzwände sind massiver. Der Aufbau erstreckte sich fast über die gesamte Geschossfläche von 71,5 m × 37 m und besaß 17 Ankerknoten. Die Fläche ist also die bisher größte aller Experimente und teilt sich in zwei Bereiche auf, die unterschiedliche Ankerdichten aufweisen. So ist im linken Teil der Karte die Ankerdichte höher als im rechten Teil. Fünf Anker wurden direkt auf dem Flur platziert, der Rest wurde in die uns zugänglichen Räume des Stockwerks verteilt. Das Experiment bestand diesmal aus zwei Testläufen, welche in Abbildung 7.16 dargestellt sind. Testlauf 1 ist in Schwarz und Testlauf 2 in Blau gekennzeichnet. Start- und Endposition sind mit einem **S** bzw. **E** markiert. Tabelle 7.11 zeigt die wichtigsten Daten zu diesen Testläufen.

Die Erfolgsquote ist bei diesem Experiment wieder abgesunken, liegt aber ungefähr

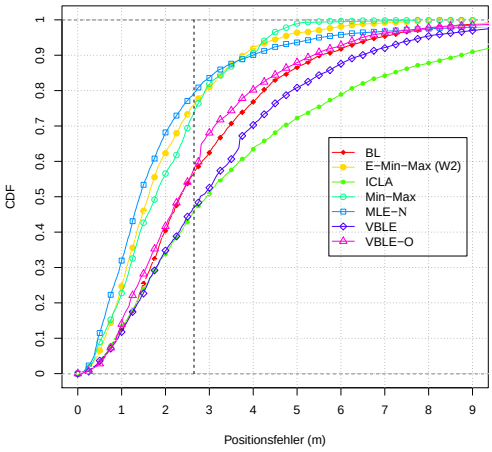
10 % über den Werten von Experiment 2, welches einen vergleichbaren Aufbau hatte. So ergeben sich trotz größerer Fläche und geringer Ankeranzahl im Vergleich mit Experiment 2 doch mit 7,40 und 7,53 sogar noch leicht höhere durchschnittliche Ankeranzahlen. Dies liegt in erster Linie an den zwei zusätzlichen Ankern auf dem Flur, die für einen solch großen Aufbau benötigt werden. Diese sind auf den jeweiligen Fluren trotz großer Entfernungen von bis zu 70 m konstant erreichbar, auch wenn die Wahrscheinlichkeit für Reflexionen ansteigt. Dies sieht man an den großen Maximalfehlern von 63,39 m und 74,18 m. Entfernt man alle Anker auf dem Flur, so sinkt die durchschnittliche Ankeranzahl auf 4,97 ab und der Maximalfehler fällt unter 40 m. Der Durchschnittsfehler der Distanzmessungen ist mit 2,65 m und 2,85 m noch einmal größer als bei allen vorherigen Experimenten. Bei beiden Testläufen gab es 13-mal 13 Anker und dreimal 14 Anker als die beiden größten verfügbaren Ankeranzahlen. So musste die Ankerzahl auch hier nur für RLSM auf zwölf beschränkt werden.

Betrachtet man die Ergebnisse der Testläufe, welche in Tabelle 7.12 und Abbildung 7.17 dargestellt sind, so zeigt sich, dass sich die Verhältnisse zwischen den Algorithmen bei diesem Experiment nicht verändert haben. Die Reihenfolge der Algorithmen sortiert nach ihrem MAE ist im Vergleich mit Experiment 3 unverändert, wenn man Testlauf 1 als Basis nimmt. Dieser besitzt ein ähnliches Bewegungsmuster, da sich alle Positionen innerhalb der Ankerfläche befanden. Die niedrige durchschnittliche Ankeranzahl und der größere Fehler bei den Distanzmessungen lassen die Algorithmen jedoch insgesamt schlechter abschneiden. Bei vielen Algorithmen steigt der Positionsfehler an, größtenteils auch zwischen Testlauf 1 und Testlauf 2, da dieser einen leicht größeren Fehler besitzt. Nur die LLS Varianten können ihre Genauigkeit im messbaren Bereich verbessern. Die Genauigkeit von NLLS sowie aller Geo-n Varianten bleibt im nicht unterscheidbaren Bereich, während sich Algorithmen wie AML oder Rwgh merklich verschlechtern. Im Falle von Geo-n-Γ bestätigen die Resultate die Anwendbarkeit des Fittings mittels Gammaverteilung in ähnlichen Szenarien. Gleiches gilt für MD-Min-Max, der seine Leistung nahezu halten kann.

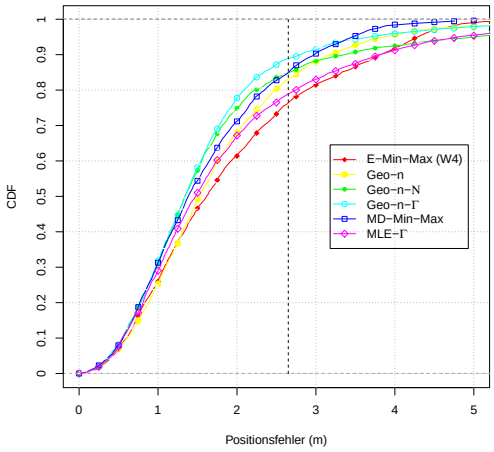
Die Verbesserung von LLS, LMS, CluRoL und RLSM liegt an der geringeren Anzahl größerer Messfehler. Vergleicht man Abbildung 7.17a und Abbildung 7.17d, so fallen die CDF-Kurven der vier Algorithmen bei Abbildung 7.17a stärker ab, auch der Maximalfehler ist bei Testlauf 1 deutlich größer. Alle LLS Varianten schaffen es in beiden Testläufen auch hier den Basisalgorithmus zu verbessern, wobei die Rangfolge wieder



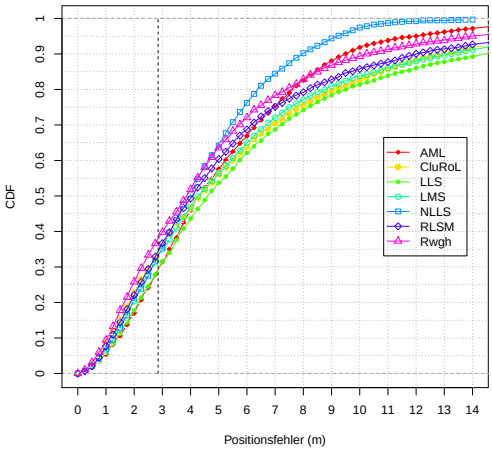
(a) CDF



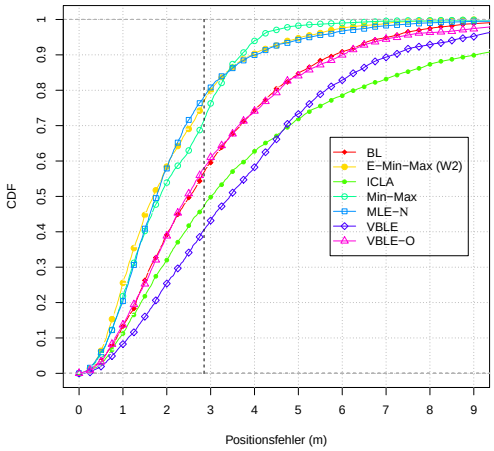
(b) CDF



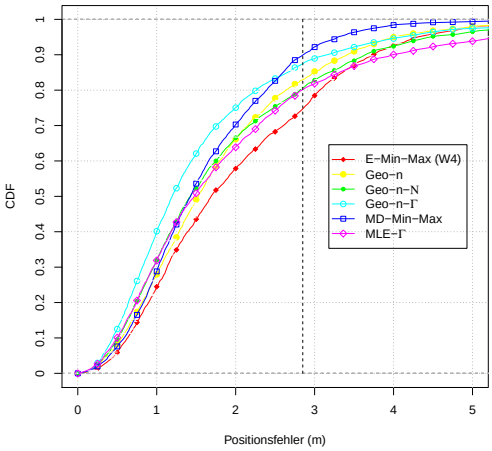
(c) CDF



(d) CDF



(e) CDF



(f) CDF

Abbildung 7.17: Abbildungen 7.17a bis 7.17c zeigen die kumulative Verteilungsfunktion des Positionsfehlers der Algorithmen von Testlauf 1, Abbildungen 7.17d bis 7.17f die Verteilung von Testlauf 2.

Tabelle 7.12: Ergebnisse von Experiment 4

Algorithmus	Testlauf 1			Testlauf 2		
	MAE	RMSE	MAX	MAE	RMSE	MAX
AML	4,77	5,55	22,56	5,14	6,32	36,91
BL	2,89	3,52	21,86	2,98	3,61	15,56
CluRoL	7,70	18,88	256	6,12	9,15	126
E-Min-Max (W2)	1,98	2,44	17,91	2,06	2,53	11,10
E-Min-Max (W4)	1,91	2,29	16,48	2,01	2,38	9,00
Geo-n	1,77	2,14	16,38	1,80	2,18	12,39
ICLA	4,09	5,66	37,06	4,34	6,18	45,52
LLS	10,52	28,21	461	7,70	14,44	195
LMS	8,26	23,21	449	7,00	13,36	195
Min-Max	2,00	2,36	15,26	2,09	2,46	15,39
NLLS_{GN}	4,42	5,33	25,55	4,43	5,17	23,39
NLLS_{LM}	4,29	5,22	27,43	4,34	5,11	34,85
RLSM	8,74	25,23	449	5,93	9,60	195
Rwgh	4,43	5,82	28,89	5,03	6,84	41,26
Geo-n-$\mathcal{N}$	1,80	2,54	20,97	1,84	2,35	17,27
Geo-n-Γ	1,59	2,01	19,12	1,57	2,04	13,00
MD-Min-Max	1,60	1,90	17,86	1,62	1,88	12,67
MLE-$\mathcal{N}$	2,02	2,99	22,81	2,15	2,72	20,87
MLE-Γ	1,90	2,48	27,05	1,95	2,55	13,44
VBLE	3,32	4,07	18,55	3,88	4,64	21,05
VBLE-O	2,76	3,40	20,35	3,07	3,86	25,35

(alle Angaben in Metern)

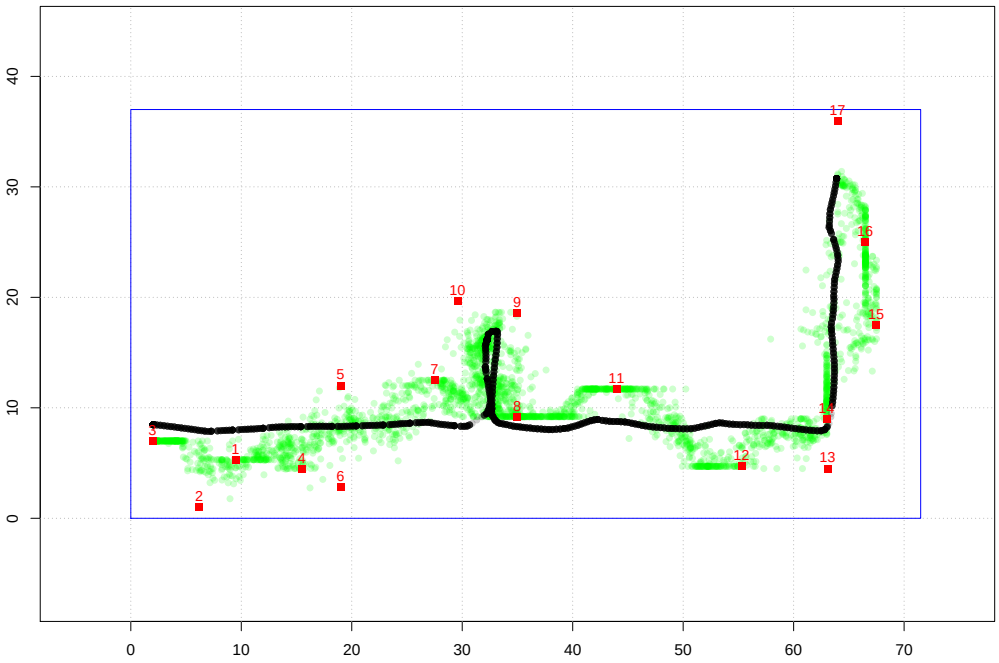
uneinheitlich ausfällt. Im ersten Testlauf ist diese LLS (10,52 m), RLSM (16,9 %), LMS (21,5 %) und CluRoL (26,8 %). Im zweiten Testlauf ist diese LLS (7,70 m), LMS (9,1 %), CluRoL (20,5 %) und RLSM (23,0 %). Eine größere Anzahl von schlechten Ankerkonstellationen scheint die Performance von RLSM am größten negativ zu beeinflussen, wenn man dieses und auch die anderen Experimente betrachtet.

Auch die anderen optimierten Algorithmen erzielen jeweils eine Verbesserung gegenüber ihrem Basisalgorithmus: VBLE-O 16,9 % bzw. 20,9 %, Geo-n-Γ 10,2 % bzw. 12,8 %, E-Min-Max (W2) 1,0 % bzw. 1,4 %, E-Min-Max (W4) 4,5 % bzw. 3,8 % und MD-Min-Max 20,0 % bzw. 22,5 %. Die Verbesserung von MD-Min-Max fällt diesmal im Vergleich zu den beiden E-Min-Max Algorithmen besonders stark aus, da MD-Min-Max aufgrund des exakten Fittings bei diesen Testläufen sein volles Poten-

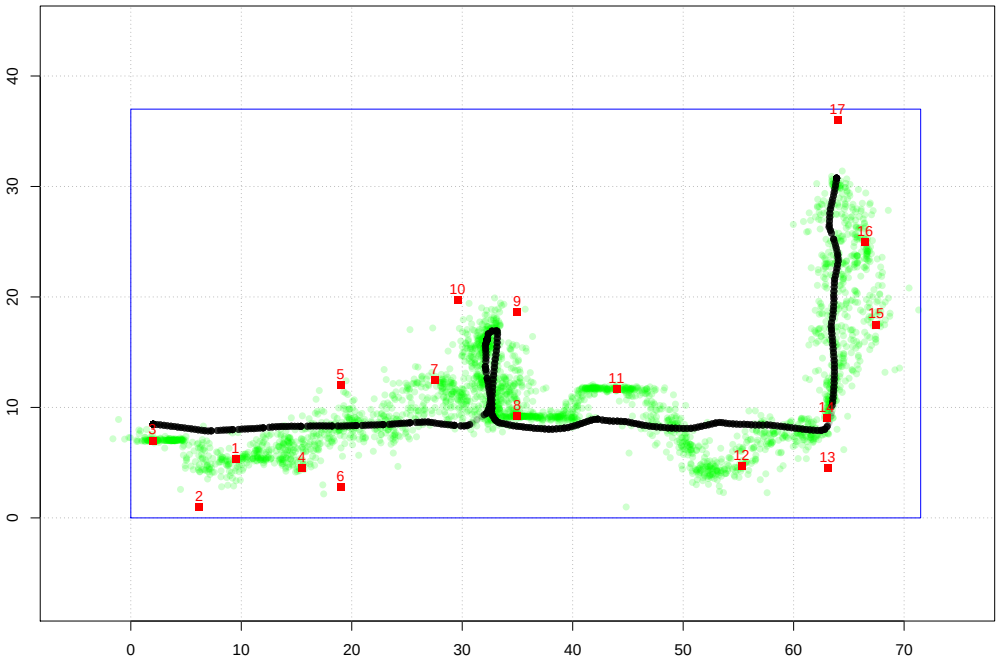
zial ausspielen kann. Rwgh kann die Leistung von $NLLS_{GN}$ erneut nicht verbessern, im zweiten Lauf tritt sogar wieder eine Verschlechterung von 4,43 m auf 5,03 m auf. Die Leistung von $NLLS_{LM}$ ist bei diesem Experiment in beiden Testläufen leicht besser als die von $NLLS_{GN}$, weshalb diesmal $NLLS_{LM}$ für die Darstellung in Abbildung 7.17 verwendet wurde. Die Optimierung mit der echten Position als Startwert erreicht Genauigkeiten von 4,01 m und 4,00 m und liegt hier zum ersten Mal merklich über den Ergebnissen beider NLLS Varianten. Auch bei diesem Experiment erzielt Geo-n mit 1,77 m bzw. 1,80 m die insgesamt beste Genauigkeit aller Universalalgorithmen und auch eine hohe Präzision. In beiden Testläufen liegt die Anzahl der Positionsfehler, die kleiner als der Eingabefehler sind, bei deutlich über 80 % (vgl. Abbildungen 7.17c und 7.17f).

Abbildung 7.18 zeigt die geschätzten Positionen für Min-Max und die beiden Verbesserungen E-Min-Max (W4) und MD-Min-Max sowie für die drei geometrischen Algorithmen BL, Geo-n und ICLA, die die Schnittpunkte der Kreise clustern. Die einzelnen Abbildungen zeigen zudem die wirklich gefahrene Strecke des mobilen Knotens in Schwarz, wobei die geschätzten Positionen für die einzelnen Messpunkte in Grün dargestellt sind. Zusätzlich sind noch die vorhanden Anker inklusive Nummer in Rot und der Grundriss der Karte von Abbildung 7.16 in Blau eingezeichnet. Jedes Bild entspricht einem Algorithmus.

Min-Max eignet sich in solchen Szenarien besonders, da die geschätzte Position nicht außerhalb der Eingrenzung durch die Anker liegen kann. Dies sieht man um die Anker 11 und 12 sowie auf dem letzten Teilstück des vertikalen Flures sehr gut. Hier schätzt Min-Max die Position relativ konstant falsch zu einer Seite, da die Ankerabstände aber nur gering sind und der echte Pfad nur unweit der Anker verläuft, ergeben sich auch nur Fehler von ca. 2,5 m bis 3 m. Dies resultiert in einem guten MAE von 2,09 m, welcher noch deutlich unter dem Eingabefehler von 2,85 m liegt und geringer Streuung, betrachtet man zusätzlich den RMSE von 2,46 m oder Abbildung 7.18a. Von den beiden E-Min-Max Algorithmen kann in erster Linie E-Min-Max (W4) die Nachteile von Min-Max in diesem Szenario mildern, da dieser auf die Fläche innerhalb der Anker optimiert ist. Dies kann man vor allem im Vergleich der letzten Teilstücke des vertikalen Flures sehen. Hier liegen die Positionen viel gleichmäßiger verteilt. Natürlich können jetzt auch Positionen außerhalb der Anker geschätzt werden, wie Abbildung 7.18b zeigt. Dies egalisiert die Vorteile zum Teil wieder, insgesamt kommt

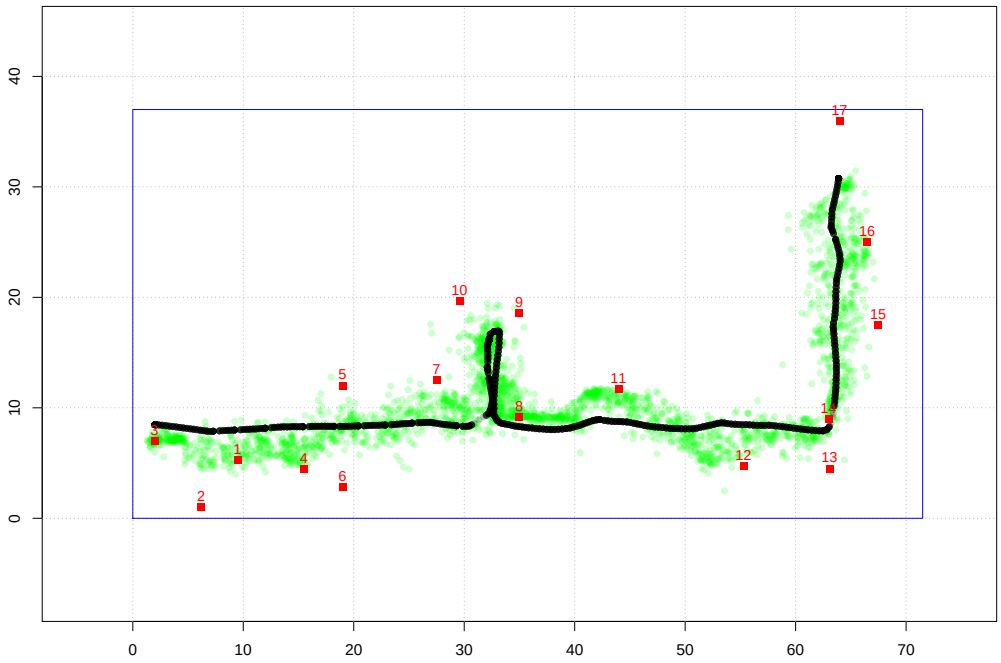


(a) Min-Max

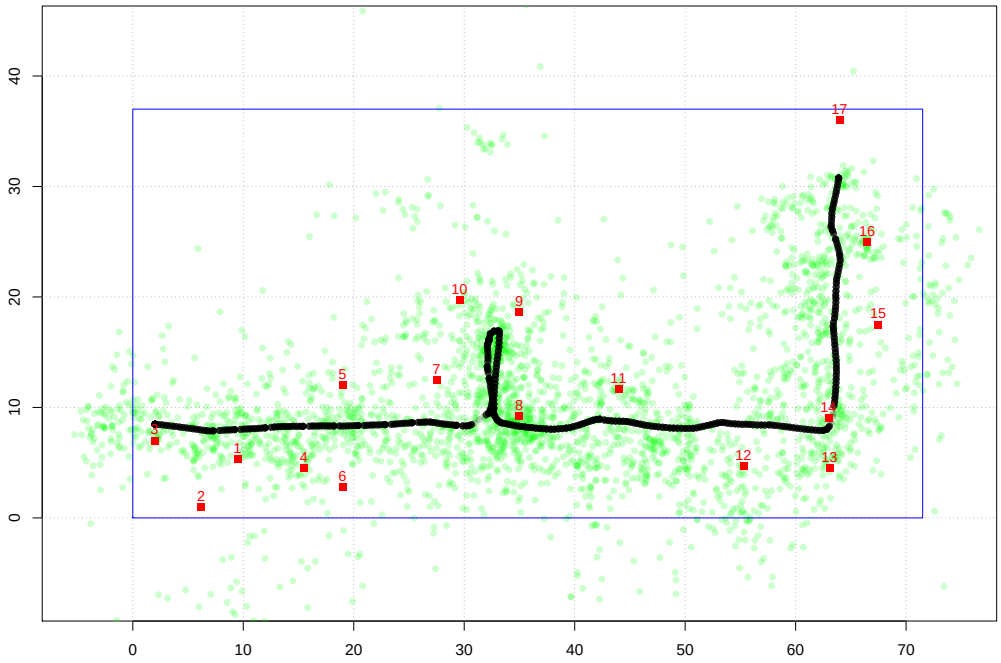


(b) E-Min-Max (W4)

Abbildung 7.18: Berechnete Positionen für sechs ausgewählte Algorithmen für Testlauf 2. Die echten Positionen sind als schwarze Punkte dargestellt, die geschätzten Positionen als grüne Punkte und Anker in Rot (alle Angaben in Metern).

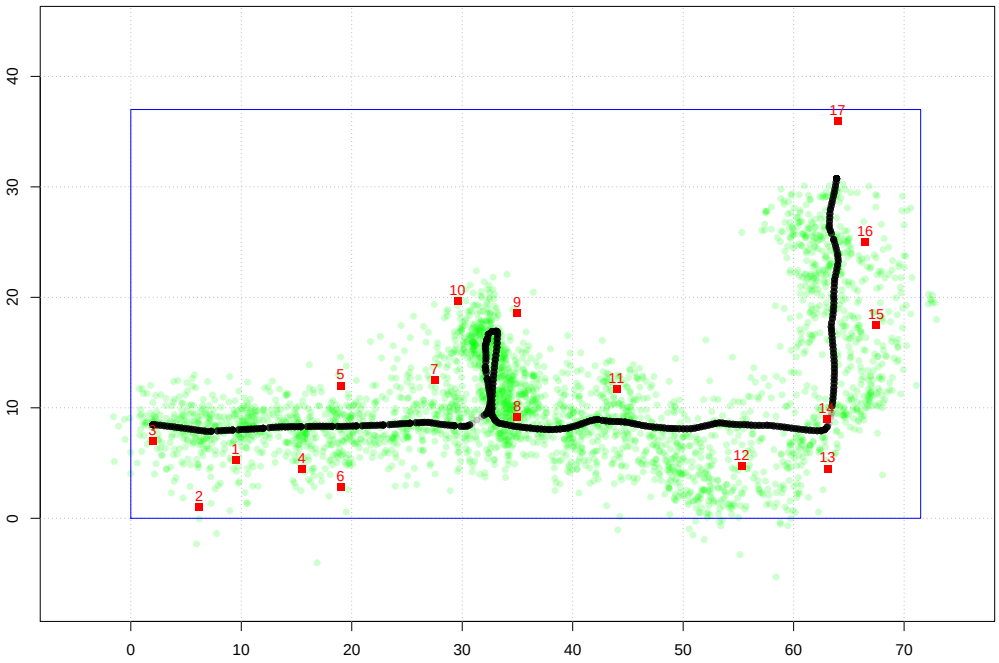


(c) MD-Min-Max

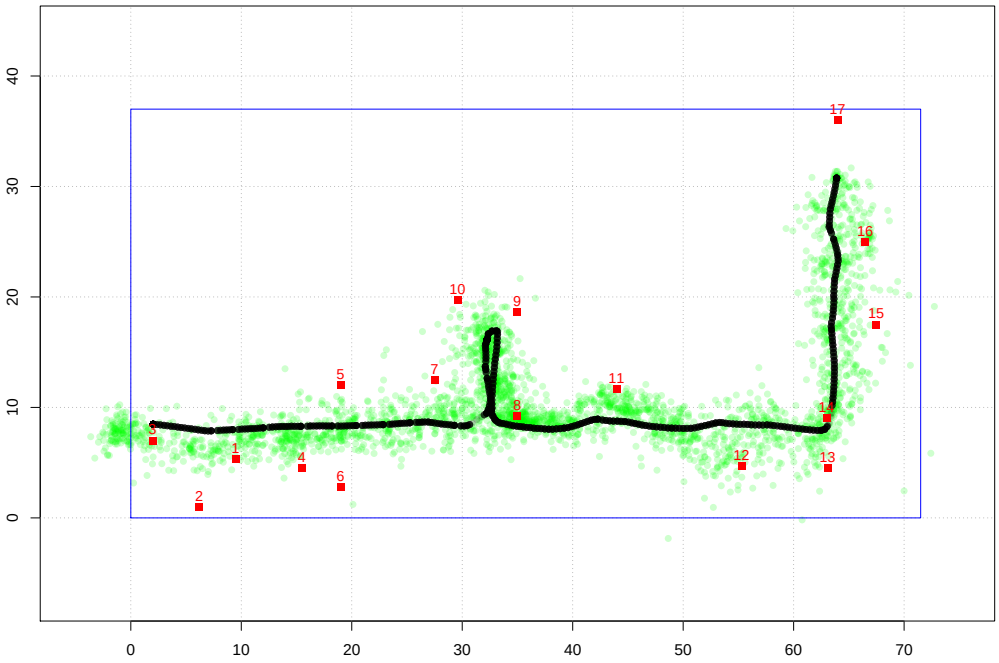


(d) ICLA

Abbildung 7.18: Berechnete Positionen für sechs ausgewählte Algorithmen für Testlauf 2. Die echten Positionen sind als schwarze Punkte dargestellt, die geschätzten Positionen als grüne Punkte und Anker in Rot (alle Angaben in Metern).



(e) BL



(f) Geo-n

Abbildung 7.18: Berechnete Positionen für sechs ausgewählte Algorithmen für Testlauf 2. Die echten Positionen sind als schwarze Punkte dargestellt, die geschätzten Positionen als grüne Punkte und Anker in Rot (alle Angaben in Metern).

ein MAE von 2,01 m heraus. Der Unterschied beträgt also nur 8 cm und liegt damit im nicht exakt unterscheidbaren Bereich. In Fällen wo die Nachteile von Min-Max noch stärker zum Tragen kommen (z. B. Testlauf 3 von Experiment 3), werden die Unterschiede entsprechend deutlicher ausfallen. Für MD-Min-Max fällt die Verbesserung noch stärker aus, was man beispielsweise gut in den Bereichen um und zwischen Anker 11 und 12 in Abbildung 7.18c sehen kann. Neben dem deutlich kleineren durchschnittlichen Positionsfehler (1,62 m), ist auch die Streuung des Fehlers noch einmal minimal reduziert worden.

Im Gegensatz zu den Min-Max Algorithmen, weisen die geometrischen Algorithmen eine größere aber annähernd gleichmäßige Streuung der Positionen um den echten Pfad auf. Während die Streuung bei ICLA doppelt so groß ist wie bei BL, die Standardabweichungen betragen 2,04 m und 4,40 m, wird die Streuung bei Geo-n noch einmal fast halbiert. Die Standardabweichung beträgt 1,23 m. Dies lassen auch die Abbildungen 7.18d bis 7.18f erahnen. Während die Streuung bei ICLA überall gleich groß ist, ist die Streuung bei Geo-n im rechten Teil der Karte leicht und bei BL merklich größer, was aus einer getrennten Auswertung für beide Bereiche, aber auch beim reinen Betrachten der Abbildungen deutlich wird. Im rechten Kartenabschnitt ist der Informationsgehalt aufgrund der geringeren Ankerdichte herabgesetzt. Für BL bestätigen sich hier erneut die Ergebnisse der LS^2 Auswertung: Die hohe Varianz der Schätzungen im Verhältnis zum kleineren *Bias* wirkt sich negativ auf den durchschnittlichen Lokalisierungsfehler aus, welcher auch nur bei 2,98 m liegt und sich damit über dem Eingabefehler befindet. Bei Geo-n wird der Positionsfehler hauptsächlich aus dem *Bias* des Algorithmus bestimmt, die Streuung ist relativ gering. Dies kann man ebenfalls in der Abbildung gut erkennen. Am Ende des Pfades bei Anker 3 kann man noch ein deutlich unterschiedliches Verhalten von BL gegenüber Geo-n erkennen, was jedoch nur in der Minderheit der Fälle positiv ausfällt. Hier produziert Geo-n größere Ausreißer auf der x-Achse (zwischen 2 m und 5 m), die sich alle der letzten Position zuordnen lassen. Der mobile Roboter hat hier noch ca. 11 Sekunden weiter Daten aufgezeichnet, ohne sich zu bewegen. Da sämtliche Anker inkl. Anker 3 in der Mehrzahl zu lang messen und es keine ausgleichenden Anker (links von Anker 3 oder oberhalb) gibt, liegen die meisten Schnittpunkte der Ankerkreise hinter Anker 3. Dies lässt den MAE für Geo-n von 1,75 m auf 1,80 m ansteigen. BL kann mit dieser Situation besser umgehen, da dieser Algorithmus eine andere Methode zur Approximation von Kreisschnittpunkten verwendet, welche die approximierten Schnittpunkte weiter nach innen zieht. Insgesamt schneiden dann beide Algorithmen in diesem für

Tabelle 7.13: Ergebnisse über alle Experimente

Algorithmus	MAE	90 % Quantil	99 % Quantil	Laufzeit ($N = 9$)
AML	3,92 m	7,65 m	13,68 m	4495 ns
BL	2,14 m	4,16 m	7,90 m	38 840 ns
CluRoL	4,39 m	8,26 m	21,47 m	651 013 ns
E-Min-Max (W2)	1,52 m	2,85 m	5,92 m	3981 ns
E-Min-Max (W4)	1,53 m	2,99 m	5,52 m	1571 ns
Geo-n	1,54 m	2,84 m	5,75 m	30 563 ns
ICLA	2,70 m	5,61 m	15,29 m	492 901 ns
LLS	5,96 m	11,46 m	34,19 m	5778 ns
LMS	5,30 m	10,63 m	30,02 m	50 853 ns
Min-Max	1,64 m	3,18 m	7,05 m	820 ns
NLLS_{GN}	3,40 m	6,10 m	10,72 m	38 835 ns
NLLS_{LM}	3,36 m	5,99 m	10,69 m	34 034 ns
RLSM	4,86 m	9,22 m	25,69 m	5 417 744 ns
Rwgh	3,81 m	8,06 m	18,94 m	10 398 962 ns
Geo-n-$\mathcal{N}$	1,54 m	2,81 m	6,32 m	46 424 ns
Geo-n-Γ	1,37 m	2,44 m	5,67 m	51 671 ns
MD-Min-Max	1,45 m	2,69 m	5,66 m	1624 ns
MLE-$\mathcal{N}$	1,84 m	3,31 m	7,35 m	49 585 ns
MLE-Γ	1,52 m	2,90 m	6,63 m	369 709 ns
VBLE	3,01 m	5,37 m	10,18 m	67 231 ns
VBLE-O	2,22 m	4,18 m	9,16 m	14 931 ns

sich betrachteten Kartenabschnitt gleich gut bzw. schlecht ab.

7.5 Experimentübergreifende Ergebnisse

Um einen abschließenden und kompakten Gesamtüberblick über die Genauigkeit und Präzision der Algorithmen zu geben, sind in Tabelle 7.13 noch einmal die experimentübergreifenden Ergebnisse aller Algorithmen dargestellt. Enthalten sind alle Experimente und Testläufe ausgenommen Testlauf 1 von Experiment 1, da es sich hier um eine Komplettkartierung im Vergleich zu den anderen Testläufen handelte. Neben dem MAE als Maß für die Genauigkeit benutzen wir die Verteilung der absoluten Positionsfehler als Maß für die Präzision und verwenden hier das 90 % und 99 % Quantil. Zum Beispiel hat AML eine Lokalisierungspräzision von 90 % innerhalb von 7,65 m (die CDF des Positionsfehlers von 7,65 m ist 0,9). Zusätzlich ist noch die durchschnitt-

liche Laufzeit einer Lokalisierung mit dem jeweiligen Algorithmus in Nanosekunden angegeben[2]. Dies soll einen Eindruck über die Laufzeit der Algorithmen unter Praxisbedingungen geben, da die mit LS[2] ermittelten Laufzeiten die nebenläufige und hoch optimierte (vektorisierte) Ausführung von Millionen von Durchläufen der Algorithmen darstellen. Nicht alle Algorithmen ließen sich hierbei gleich gut vektorisieren. Da nicht alle Messdaten die gleichen Ankeranzahlen enthalten, wurde die gerundete durchschnittliche Ankeranzahl von 9 genommen und nur Messungen mit exakt dieser Anzahl für die hier gegebene Laufzeitbetrachtung verwendet, um so eine Vergleichbarkeit zu gewährleisten. So weisen die NLLS Algorithmen für eine Lokalisierung von bis zu neun Ankern unter vergleichbaren Praxisbedingungen sogar eine leicht höhere Laufzeit als Geo-n auf, obwohl die asymptotische Laufzeit von Geo-n in $\mathcal{O}(N^4)$ und die von NLLS nur in $\mathcal{O}(N^3)$ liegt.

7.6 Vergleichbarkeit der Ergebnisse von Experiment und räumlicher Fehlerverteilung

Ein Vergleich der Ergebnisse aus den durchgeführten Simulationen und Experimenten zeigt eine große Übereinstimmung in der Beurteilung der Algorithmen und damit die generelle Eignung der räumlichen Fehlerverteilung als Bewertungsmetrik für die Performance einzelner Algorithmen. Zwar treten im Detail Unterschiede auf, die das Durchführen von Experimenten erforderlich machen, doch überwiegen in der Mehrzahl der Fälle die Übereinstimmungen in den getroffenen Bewertungen. Besonders große Übereinstimmung liefert die Analyse der Abhängigkeit zwischen Ankerzahl und erzielter Genauigkeit. Auch die Evaluation von *Bias* und Varianz der Algorithmen stimmte größtenteils überein. Besonders für Algorithmen die in der räumlichen Fehleranalyse starke Schwankungen auf kleinster Fläche aufweisen (z. B. VBLE-O), ist die Übertragbarkeit der Simulationsergebnisse auf die Praxis schwierig (vgl. Abschnitt 7.4.2, Abbildung 7.10).

Für den allgemeinen Fall der Indoor-Lokalisierung haben sich sowohl in der Analyse der räumlichen Fehlerverteilung als auch in den durchgeführten Experimenten die selben Algorithmen als geeignet bzw. ungeeignet erwiesen. Insbesondere Geo-n, VBLE-O, BL und die Min-Max Varianten zeigen in unseren Experimenten Lokalisie-

[2]Die Berechnung erfolgte auf einer modernen Workstation (Intel(R) Core(TM) i5-3570K CPU mit 3,40 GHz und 16 GB Arbeitsspeicher).

rungsgenauigkeiten in Größe des Erwartungswertes des Eingabefehlers. Genau dieses Verhalten haben wir in Abschnitt 6.2 für den praktischen Einsatz dieser Verfahren prognostiziert.

7.7 Fazit

In dieser experimentellen Auswertung wurde eine große Anzahl von Lokalisierungsalgorithmen unter einheitlichen Bedingungen in unterschiedlichen Szenarien begutachtet. Hierbei variierte die Größe und Regularität des Aufbaus, die strukturellen Gebäudeeigenschaften sowie die durchschnittliche Ankeranzahl und Ankerdichte. Insgesamt ergaben sich für die einzelnen Experimente unterschiedliche Verhältnisse von der Anzahl der Anker zur Experimentfläche (A/N), die jedoch im oberen Bereich liegen, zieht man Tabelle 5.4 aus Abschnitt 5.2 als Referenz heran. So kam für Experiment 1 ein Anker auf 5,7 m^2, bei Experiment 2 auf 30,5 m^2, bei Experiment 3 auf 59,2 m^2 und bei Experiment 4 auf 155,6 m^2. Damit weisen unsere Experimente deutlich realistischere Werte als die anderen entfernungsbasierten Experimente aus der Literatur auf. Nur die RSS-basierten Fingerprinting-Verfahren erzielen gleiche oder noch höhere Werte. Auch der durchschnittliche Distanzmessfehler und dessen Streuung veränderte sich in den einzelnen Experimenten, was eine Analyse der Algorithmen unter unterschiedlich starken NLOS-Bedingungen möglich machte. Alle Experimente fanden unter realistischen Bedingungen statt, d. h. es bewegten sich Personen während der Messphase durch den Aufbau, es wurden von diesen Personen Türen geöffnet etc. Zudem war andere Funktechnologie (WLAN) parallel in Betrieb. Somit besitzen sämtliche Ergebnisse und Schlussfolgerungen hohe Praxisrelevanz.

LLS zeigte die durchgängig schlechteste Performance von allen untersuchten Algorithmen. Die darauf aufbauenden Optimierungen LMS, RLSM und CluRoL können die Leistung im Schnitt zwar zwischen 10 % und 40 % steigern, jedoch reicht die Genauigkeit nur in seltenen Fällen an die Leistung von NLLS heran. Schon beim Vorhandensein von kleinem Messrauschen leiden alle Algorithmen unter extrem starken Ausreißern, falls eine schlechte geometrische Konstellation für die Lokalisierung vorliegt (vgl. Zekavat und Buehrer [160, S. 195]). Besonders CluRoL verstärkt diesen Effekt durch eine eventuell ungünstige Ankerauswahl für die finale Positionsabschätzung. LMS begegnet ungünstigen Geometrien durch die Auswahl von 4-elementigen Teilmengen für die Berechnung der Zwischenpositionen, um die Wahrscheinlichkeit

für das Auftreten dieser zu reduzieren. Daher scheint der Mechanismus der Ankerauswahl von CluRoL stark verbesserungsbedürftig, zumal im ungünstigsten Fall Anker entfernt werden, so dass keine Lokalisierung mehr möglich ist.

Rwgh konnte nur in dem einfachen Szenario von Experiment 1 eine minimale Verbesserung von NLLS erreichen und verschlechterte die Performance in den meisten Fällen sogar. AML erzielte die über alle Experimente gesehen beste Leistung im Vergleich mit allen LLS Varianten und liegt mit einem MAE von 3,92 m sogar nur knapp 0,5 m hinter NLLS (ebenfalls bei Betrachtung aller Experimente). Anhand der Ergebnisse von Hillebrandt et al. muss noch angemerkt werden [53], dass AML generell am besten arbeitet, wenn die Ankerzahl zwischen vier und fünf liegt. Dies entspricht den Resultaten von Kuruoglu et al. [70] und ist der Grund, warum sie die Ankerzahl beim Vergleich mit LLS und Min-Max auf vier beschränkt haben. Sie haben die Verfeinerungsphase ihres Algorithmus als Ursache für das Verhalten ausgemacht. Beachtet werden muss hierbei, dass die asymptotische Laufzeit von AML niedrig ist und AML in unseren Experimenten ungefähr 31-mal schneller als NLLS und 2,8-mal schneller als LLS war (durchschnittliche Laufzeit über alle Ankeranzahlen in den Messdaten, d. h. keine Beschränkung auf 9 Anker). Dies macht AML für den Praxiseinsatz auf leistungsschwachen Sensorknoten unter Umständen besonders attraktiv. Insbesondere im Vergleich mit Rwgh und RLSM schneidet AML hier deutlich besser ab, da diese Algorithmen enorme Rechenkapazitäten in Anspruch nehmen, die erzielte Genauigkeit bzw. Verbesserung dies aber in keinster Weise rechtfertigt.

MLE-$\mathcal{N}$ und MLE-Γ können die Leistung von NLLS_{GN} deutlich steigern. Um eine Vergleichbarkeit zu gewährleisten, benutzen diese Algorithmen alle die gleichen Startpunkte für ihre Optimierungsprozedur. MLE-$\mathcal{N}$ schlägt NLLS_{GN} um 35 % bis 54 % und MLE-Γ übertrifft NLLS_{GN} in der Regel um 48 % bis 58 % in der Genauigkeit. Beide Algorithmen zeigen eine signifikant bessere Leistung, da sie die positive Verzerrung des Distanzmessfehlers berücksichtigen und NLLS leicht durch NLOS-Fehler fehlgeleitet werden kann. Jedoch können die überdurchschnittlichen Ergebnisse aus Kapitel 6 in den Experimenten nicht reproduziert werden. Der Grund hierfür liegt in dem Fehlermodell der Simulation, das sich von der Realität unterscheidet, auch wenn die Verteilungen sehr ähnlich sind. Während in der Simulation die Fehler in völlig zufälliger Reihenfolge auftreten, so hängt der Fehler in der Realität stark von der eigenen Position im Gebäude ab. Auftretende Reflexionen und *Multipath*-Fehler sind daher nicht zufällig und hängen von der physikalischen Struktur des Gebäudes und der aktuellen Position ab. Aufeinanderfolgende Positionen eines Pfades haben

mit hoher Wahrscheinlichkeit ähnliche Eigenschaften, da die Bewegungsgeschwindigkeit in der Regel limitiert ist. Dies macht die Lokalisierung in der Praxis für diese Algorithmen schwieriger und führt zu einem anderen Verhalten.

In der Praxis haben sich sämtliche Min-Max Algorithmen als sehr geeignet erwiesen. In den meisten Aufbauten erzielen diese gute Leistungen mit Durchschnittsfehlern, die deutlich unter dem Eingabefehler liegen. In schwierigen Situationen fällt die Leistung der meisten Algorithmen aber ab, vor allen Dingen außerhalb der Anker. Hier bleibt noch E-Min-Max (W2) als Alternative übrig. Zudem weisen all diese Algorithmen die durchgängig kleinsten Maximalfehler auf. Diese Eigenschaften sollten bei der Auswahl eines passenden Algorithmus berücksichtigt werden. MD-Min-Max liefert die beste Performance aller Min-Max Varianten innerhalb der konvexen Hülle der Anker, speziell bei einem exakten Fitting der Parameter auf die Fehlerverteilung. Die Zugehörigkeitsfunktion, definiert durch die drei Parameter $[MF_{\text{low}}, MF_{\text{mode}}, MF_{\text{up}}]$, ist also charakteristisch für eine Ausbringung in einem Gebäude oder einem Stockwerk des Gebäudes und die Ergebnisse sind von ähnlicher Güte bei mehreren Testläufen in einer solchen Ausbringung. MD-Min-Max ist wohlgemerkt durchaus empfindlich gegenüber den Parametern der Zugehörigkeitsfunktion. Falls wir eine Normalverteilung auf unseren statistischen Daten (Experiment 4, Testlauf 2) annehmen und die Drei-Sigma-Regel anwenden, so wird die Zugehörigkeitsfunktion charakterisiert durch [−8,3; 2,4; 13,1]. Mit dieser Funktion geht der MAE von 1,62 m auf 1,89 m zurück. MD-Min-Max kann auch der schlechteste Algorithmus werden, falls die Zugehörigkeitsfunktion die Daten nicht fittet. Die Wahl von [6; 12; 18] führt beispielsweise zu einem MAE von 2,19 m. Eine sorgfältige Analyse der statistischen Daten ist also notwendig für gute Resultate.

Geo-n erzielte die durchgängig beste Leistung der getesteten geometrischen Algorithmen und auch die besten Ergebnisse aller Universalalgorithmen in irregulären und großen Ausbringungen. Besonders in schwierigen Situationen, beispielsweise außerhalb der konvexen Hülle der Anker, schneidet Geo-n im Vergleich mit den ebenfalls guten Min-Max Algorithmen besser ab. Dies liegt an der homogenen räumlichen Fehlerverteilung, welche aus den LS^2-Simulationen ersichtlich wurde. Um diese zu erzielen, werden jedoch mindestens Messungen zu sechs Ankern benötigt. Doch auch in einfachen bzw. kleineren Installationen schneidet Geo-n noch sehr gut ab, was die obere Platzierung im Testfeld bei Experiment 1 und 2 zeigte. Damit eignet sich dieser Algorithmus besonders gut für eine initiale Positionsabschätzung im Indoor-Bereich, die gegebenenfalls danach noch verfeinert werden kann. Von den verbesserten Vari-

anten taugt in erster Linie Geo-n-Γ für Installationen, für die die Fehlerverteilung bekannt ist. Die Gammaverteilung approximiert die reale Verteilung deutlich besser und erzielt folglich auch die besten Genauigkeiten in fast allen Experimenten. Ist diese einmal für eine bestimmte Kombination aus eingesetzter Messhardware und Gebäudetyp ermittelt, so lässt sich die Leistung von Geo-n ohne großen Aufwand in einer Vielzahl der Szenarien im Schnitt zwischen 10 % und 17,5 % steigern.

Weiterhin zeigen alle Algorithmen in Abhängigkeit der Ankerkonstellation, der eigenen Position zu den Ankern sowie der Varianz und dem *Bias* der Distanzmessungen einen mehr oder weniger großen resultierenden *Bias* in den geschätzten Positionen, falls mehrere Messungen an einer Position vorhanden sind. Dies stimmt mit den Ergebnissen von Rui und Ho überein [113], die die Auswirkung des *Bias* auf die Genauigkeit der Lokalisierung für MLE-$\mathcal{N}$ untersuchen. Bei einem normalverteilten Rauschen steigt dort der Beitrag des *Bias* zum MSE der Lokalisierung mit zunehmender Varianz an. Von Sadler et al. wird in ähnlicher Weise die Genauigkeit von WLS untersucht [114], welcher im Allgemeinen ebenfalls verzerrte Ergebnisse produziert und dabei der Einfluss des *Bias* und der Varianz der Messungen sowie die Positionen der Sensorknoten berücksichtigt. Im Gegensatz hat hier bei steigender Varianz der Messungen ein zusätzlicher *Bias* auf den Messungen weniger Einfluss auf den RMSE der Lokalisierung. Hier wurde zudem noch gezeigt, dass der *Bias* bei der Lokalisierung auch ausschließlich durch die Geometrie der Sensorknoten erzeugt werden kann. Somit stellt sich eine Analyse der Verzerrung der Algorithmen als schwierig heraus. In der Mehrzahl der Fälle ist keiner der Algorithmen ein erwartungstreuer Schätzer und daher der Genauigkeitsgewinn durch mehrfaches Messen und Mittelwertbildung der geschätzten Positionen schwer einzuschätzen.

KAPITEL 8

Fazit und Ausblick

8.1 Fazit

Wir haben im Rahmen dieser Arbeit eine umfassende Betrachtung über den Stand der Forschung in distanzbasierten Verfahren zur Indoorlokalisierung vorgelegt und verschiedenste Algorithmen für die Durchführung einer Analyse implementiert. Als Ergebnis haben wir jeweils eine C- und eine Java-Bibliothek zur Verfügung gestellt, die sämtliche Algorithmen enthält.

Neben den aus der Literatur bekannten Algorithmen haben wir drei Algorithmen für verschiedene Szenarien selber entworfen. MD-Min-Max ist ein Algorithmus, der nur wenig Rechenzeit und Speicherplatz benötigt und auf Szenarien optimiert ist, wo möglichst genaue Annahmen über die Verteilung des Distanzfehlers getroffen werden können. Hierdurch verbessert er die Leistung von Min-Max zum Teil erheblich, bleibt aber leichtgewichtig genug, um für die verteilte Lokalisierung und das Tracking in WSNs eine gute Wahl zu sein, wo jeder Knoten die eigene Position oft und schnell berechnen muss. Die Performance des VBLE-O Algorithmus ist weitgehend unabhängig von der geometrischen Konstellation der Ankerknoten und zeigt ein besonders homogenes Bild in der räumlichen Fehlerverteilung, selbst bei niedriger Ankeranzahl. Dieser Algorithmus ist besonders für Szenarien entwickelt worden, in denen keine strategische Planung der Ankerknoten möglich ist - hier ist in erster Linie die kooperative Indoorlokalisierung zu nennen. Der Geo-n Algorithmus ist ein universell einsetzbarer Algorithmus zur Indoorlokalisierung. In allen von uns simulierten und experimentell untersuchten Fällen zeigt er die beste Performance, wobei eine Mindestanzahl von

sechs Ankern erforderlich ist, damit der Algorithmus sein volles Potenzial ausschöpfen kann. Die Genauigkeit lag in den meisten realistischen Szenarien weit über der der Vergleichsalgorithmen aus der Literatur, sofern es sich ebenfalls um Universalalgorithmen handelt, die keine Annahmen über die Fehlerverteilung machen. Wir empfehlen Geo-n als Standardalgorithmus für das allgemeine Problem der Indoorlokalisierung einzusetzen, da dieser in der Praxis nur eine leicht höhere Laufzeit als VBLE-O erzielt und selbst auf ressourcenbeschränkten Sensorknoten noch problemlos laufen sollte. In Einsatzbereichen, in denen die Verteilung des Distanzfehlers bekannt ist, steht mit Geo-n-Γ ein Algorithmus zur Verfügung, der die Leistung von Geo-n nochmals verbessert und auch die durchgängig beste Leistung aller Algorithmen zeigte. Jedoch ist die Laufzeit z. B. im Vergleich mit MD-Min-Max erheblich größer und eine Kosten-Nutzen-Abwägung sollte hier im Einzelfall getroffen werden.

Zur Bewertung der Eignung von verschiedenen Algorithmen zur Indoorlokalisierung haben wir verschiedene Metriken diskutiert und deren Eignung gegenübergestellt. Als Ergebnis haben wir für einen simulativen Vergleich die Metrik der räumlichen Fehlerverteilung entwickelt und ausführlich diskutiert. Zur Analyse der räumlichen Fehlerverteilung haben wir das von uns entwickelte Simulationsframework LS^2 vorgestellt. Mit diesem Framework ist eine einfache Simulation verschiedener Szenarien der Indoorlokalisierung möglich, wobei die Ergebnisse speziell für eine Analyse der räumlichen Fehlerverteilung aufbereitet werden. Die Analyse wird anhand verschiedener unterstützter Metriken erleichtert. Wir empfehlen die Nutzung von LS^2 als Standardwerkzeug zur simulativen Analyse verschiedener Algorithmen zur Indoorlokalisierung.

Für die Durchführung und Analyse von Experimenten zur Indoorlokalisierung haben wir ein virtuelles Testbed vorgestellt, mittels dessen sich Experimente reproduzieren und vergleichen lassen. Für dieses Testbed haben wir Millionen von Datensätzen mit einem von uns entwickelten Referenzsystem erhoben und stellen sie der Öffentlichkeit zur Verfügung. Alle Daten dieses Testbeds wurden mit auf dem Markt erhältlicher Hardware in einem normalen Bürogebäude aufgenommen, so dass möglichst realistische Bedingungen für die zu testenden Algorithmen geschaffen wurden. Wir empfehlen die Nutzung und Erweiterung dieser Datensätze, um zukünftige Experimente reproduzierbar und vergleichbar zu gestalten.

Die von uns vorgestellten Algorithmen haben wir umfassend simuliert und nach den von uns festgelegten Metriken verglichen und bewertet. Ebenso haben wir eine Reihe von Experimenten durchgeführt und diese Ergebnisse ebenfalls verglichen und be-

wertet. Dabei haben wir gezeigt, dass die resultierende räumliche Fehlerverteilung nicht nur von der Fehlerverteilung der Distanzmessung abhängt, sondern auch von der geometrischen Konstellation des Knotens und der Anker und den Eigenschaften des Algorithmus. Algorithmen mit niedrigen Durchschnittsfehlern in der Simulation müssen in der Praxis nicht zwangsweise gut abschneiden, da die guten Ergebnisse in Regionen erzielt worden sein können, die für die Praxis nicht maßgeblich sind. Umgekehrt können Algorithmen mit hohen Durchschnittsfehlern in der Praxis besser abschneiden, da ihr Fehler in den relevanten Bereichen niedrig sein kann. Mithilfe der Analyse der räumlichen Fehlerverteilung lassen sich diese Sachverhalte leichter beurteilen und die Auswahlentscheidung eines bestimmten Algorithmus für einen bestimmten Einsatzzweck erheblich vereinfachen. Die im Rahmen dieser Arbeit entstandene Auswertung von Algorithmen zur Indoorlokalisierung ist die umfassendste uns bekannte Betrachtung dieser Art.

8.2 Ausblick

Die in dieser Arbeit betrachteten Algorithmen und Verfahren zielen alle auf eine distanzbasierte Positionsbestimmung im Verhältnis zu direkt erreichbaren Nachbarn ab, deren Position genau bekannt ist. Diese Verfahren lassen sich natürlich auch in Szenarien anwenden, in denen die direkten Nachbarn ihre Position ebenfalls mit den besprochenen Verfahren gewonnen haben und die ebenfalls mobil sind. Diese Verfahren werden unter dem Begriff kooperative Lokalisierung zusammengefasst und sind in der Literatur nur verhältnismäßig wenig untersucht. Diese Verfahren stellen jedoch ganz besondere Ansprüche an die zugrunde liegenden Algorithmen. Auf diese Ansprüche sind wir nur am Rande und als Prognose eingegangen. Eine genaue Untersuchung dieser Verfahren und der daraus entstehenden neuen Probleme und Anforderungen wären im Rahmen zukünftiger Arbeiten sehr zu begrüßen.

Ebenfalls bedarf es für eine solche Analyse neuer Ansätze in der Simulation und vor allem für praktische Experimente. Das Problem großer, vollständig mobiler Netzwerke erfordert einen ganz anderen Typus von Testbeds und eine genaue Protokollierung für die Auswertung. Somit scheint eine präzise Auswertung unter heutigen Bedingungen nur schwer vorstellbar bzw. als extrem aufwändig.

Ein weiteres Feld zukünftiger Forschung, welches sich direkt aus den Ergebnissen der räumlichen Fehlerverteilung ableitet, sehen wir in der Kombination verschiede-

ner effizienter Algorithmen zur weiteren Verbesserung der Genauigkeit der Lokalisierung. Hierzu müssen Auswahlalgorithmen entwickelt werden, die für jede einzelne Lokalisierung die aktuelle Situation aus Geometrie und Messdaten analysieren und entsprechend die beste Methode zur Erzielung des kleinstmöglichen Positionsfehlers anwenden. Eine einfaches Verfahren in diesem Zusammenhang könnte beispielsweise die Auswahl von MD-Min-Max innerhalb der konvexen Hülle der Ankerknoten sein und Geo-n entsprechend im umgekehrten Fall. Hierfür müsste eine initiale Positionsabschätzung mit einem Algorithmus erfolgen, der einen möglichst gleichverteilten räumlichen Positionsfehler aufweist. Hierfür käme ebenfalls Geo-n in Frage. Wie eine Berechnung des MAE aus der Position des jeweils besten Algorithmus während der Experimente gezeigt hat, ist das Verbesserungspotenzial erheblich. So lässt sich der durchschnittliche Positionsfehler im Vergleich zum jeweils besten Einzelalgorithmus im Schnitt noch einmal halbieren, selbst eine nur partielle Ausnutzung erscheint daher lohnenswert.

Für die einfache Bewertung verschiedener Algorithmen und Verfahren sollten zusammen mit der Forschungsgemeinschaft verschiedene Standardszenarien definiert und im virtuellen Testbed abgebildet werden. Anhand dieser Szenarien könnten neue oder verbesserte Algorithmen sofort bewertet und verglichen werden. Diese Standardszenarien könnte man ebenfalls für eine Analyse der räumlichen Fehlerverteilung nutzen und würde somit zu einem umfassenden Analysewerkzeug für die Bewertung verschiedener Verfahren in Abhängigkeit vom Einsatzszenario gelangen. Auch könnte das virtuelle Testbed einfach um andere Verfahren, wie beispielsweise Fingerprinting-Verfahren, erweitert werden. So ließen sich verschiedene Algorithmen aus unterschiedlichen Klassen untereinander vergleichen und bewerten.

In zukünftigen Arbeiten zum Thema der distanzbasierten Algorithmen zur Indoorlokalisierung hoffen wir auf eine Weiterentwicklung der Ansätze aus den drei von uns entwickelten Algorithmen und auf einen Vergleich nach den hier vorgestellten Metriken und mit den hier vorgestellten Werkzeugen. Eine kontinuierliche Weiterentwicklung dieser Werkzeuge und Diskussion dieser Metriken wäre ebenfalls wünschenswert.

Literaturverzeichnis

[1] S. Adler, H. Will, T. Hillebrandt, S. Schmitt und M. Kyas. Virtual Testbed for Indoor Localization. In: *Indoor Positioning and Indoor Navigation (IPIN), Proceedings of 2013 International Conference on*, Montbeliard, France, Okt. 2013. *(Zitiert auf Seite 159)*

[2] J. Andersen, T. Rappaport und S. Yoshida. Propagation measurements and models for wireless communications channels. *Communications Magazine, IEEE*, 33(1):42–49, 1995. doi:10.1109/35.339880. *(Zitiert auf Seite 25)*

[3] J. Ansari, J. Riihijarvi und P. Mahonen. Combining Particle Filtering with Cricket System for Indoor Localization and Tracking Services. In: *Personal, Indoor and Mobile Radio Communications, 2007. PIMRC 2007. IEEE 18th International Symposium on*, Seiten 1–5, 2007. doi:10.1109/PIMRC.2007.4394578. *(Zitiert auf Seite 18)*

[4] M. Baar, H. Will, B. Blywis, T. Hillebrandt, A. Liers, G. Wittenburg und J. Schiller. The ScatterWeb MSB-A2 Platform for Wireless Sensor Networks. Technischer Bericht TR-B-08-15, Freie Universität Berlin, Department of Mathematics and Computer Science, Institute for Computer Science, Telematics and Computer Systems group, Takustraße 9, 14195 Berlin, Germany, Sep. 2008. URL: ftp://ftp.inf.fu-berlin.de/pub/reports/tr-b-08-15.pdf. *(Zitiert auf Seite 237)*

[5] J. Bachrach und C. Taylor. Localization in Sensor Networks. In: *Handbook of Sensor Networks*, Seiten 277–310. John Wiley & Sons, Inc., 2005. URL: http://dx.doi.org/10.1002/047174414X.ch9. *(Zitiert auf Seite 51)*

[6] A. Bahillo, S. Mazuelas, R. M. Lorenzo, P. Fernández, J. Prieto, R. J. Durán und E. J. Abril. Hybrid RSS-RTT Localization Scheme for Indoor Wireless Networks. *EURASIP Journal on Advances in Signal Processing*, 2010:17:1–

17:12, Feb. 2010. URL: http://asp.eurasipjournals.com/content/2010/1/126082. *(Zitiert auf Seiten 58, 127, 143 und 146)*

[7] A. Bahillo, J. Prieto, S. Mazuelas, R. Lorenzo, J. Blas und P. Fernandez. IEEE 802.11 Distance Estimation Based on RTS/CTS Two-Frame Exchange Mechanism. In: *Vehicular Technology Conference, 2009. VTC Spring 2009. IEEE 69th*, Seiten 1–5, 2009. doi:10.1109/VETECS.2009.5073583. *(Zitiert auf Seite 143)*

[8] P. Bahl und V. N. Padmanabhan. RADAR: An In-Building RF-Based User Location and Tracking System. In: *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, Band 2, Seiten 775–784. IEEE, 2000. *(Zitiert auf Seiten 25, 143 und 146)*

[9] A. Bensky. *Wireless positioning technologies and applications*. The GNSS Technology and Applications Series. Artech House, Incorporated, 2008. URL: http://books.google.de/books?id=nEHbAAAAMAAJ. *(Zitiert auf Seiten 11, 14, 28, 29, 33, 35 und 36)*

[10] J. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer Series in Statistics. Springer, 1985. URL: http://books.google.de/books?id=oY_x7dE15_AC. *(Zitiert auf Seite 106)*

[11] S. Bermejo und J. Cabestany. Oriented principal component analysis for large margin classifiers. *Neural Networks*, 14(10):1447–1461, 2001. URL: http://dx.doi.org/10.1016/S0893-6080(01)00106-X. *(Zitiert auf Seite 106)*

[12] Å. Björck. *Numerical Methods for Least Squares Problems*. Society for Industrial and Applied Mathematics, Philadelphia, 1996. URL: http://epubs.siam.org/doi/abs/10.1137/1.9781611971484. *(Zitiert auf Seite 45)*

[13] M. Blum, R. W. Floyd, V. Pratt, R. L. Rivest und R. E. Tarjan. Time Bounds for Selection. *Journal of Computer and System Sciences*, 7(4):448–461, Aug. 1973. doi:10.1016/S0022-0000(73)80033-9. *(Zitiert auf Seite 85)*

[14] J. Blumenthal, R. Grossmann, F. Golatowski und D. Timmermann. Weighted Centroid Localization in Zigbee-based Sensor Networks. In: *Intelligent Signal Processing, 2007. WISP 2007. IEEE International Symposium on*, Seiten 1–6, 2007. doi:10.1109/WISP.2007.4447528. *(Zitiert auf Seiten 142 und 146)*

[15] P. Bolliger. Redpin - Adaptive, Zero-Configuration Indoor Localization through User Collaboration. In: *Proceedings of the First ACM International Workshop*

on Mobile Entity Localization and Tracking in GPS-less Environments, MELT '08, Seiten 55–60, New York, NY, USA, 2008. ACM. doi:10.1145/1410012.1410025. *(Zitiert auf Seite 14)*

[16] N. Bulusu, J. Heidemann und D. Estrin. GPS-less Low Cost Outdoor Localization For Very Small Devices. *Personal Communications, IEEE*, 7(5):28–34, Okt. 2000. doi:10.1109/98.878533. *(Zitiert auf Seiten 8, 142 und 146)*

[17] J. J. Caffery und G. L. Stuber. Radio Location in Urban CDMA Microcells. In: *Personal, Indoor and Mobile Radio Communications, 1995. PIMRC'95.'Wireless: Merging onto the Information Superhighway'., Sixth IEEE International Symposium on*, Band 2, Seiten 858–862. IEEE, 1995. *(Zitiert auf Seite 95)*

[18] R. Chandrasekaran und A. Tamir. Open questions concerning Weiszfeld's algorithm for the Fermat-Weber location problem. *Mathematical Programming*, 44:293–295, 1989. URL: http://dx.doi.org/10.1007/BF01587094. *(Zitiert auf Seite 59)*

[19] P.-C. Chen. A Non-Line-of-Sight Error Mitigation Algorithm in Location Estimation. In: *Wireless Communications and Networking Conference, 1999. WCNC. 1999 IEEE*, Band 1, Seiten 316–320, 1999. doi:10.1109/WCNC.1999.797838. *(Zitiert auf Seiten 57, 96 und 117)*

[20] K.-Y. Cheng, V. Tam und K.-S. Lui. Improving APS with Anchor Selection in Anisotropic Sensor Networks. In: *Autonomic and Autonomous Systems and International Conference on Networking and Services, 2005. ICAS-ICNS 2005. Joint International Conference on*, Seiten 49–49, 2005. doi:10.1109/ICAS-ICNS.2005.55. *(Zitiert auf Seite 99)*

[21] J. Cota-Ruiz, J.-G. Rosiles, E. Sifuentes und P. Rivas-Perea. A Low-Complexity Geometric Bilateration Method for Localization in Wireless Sensor Networks and Its Comparison with Least-Squares Methods. *Sensors*, 12(1):839–862, 2012. doi:10.3390/s120100839. *(Zitiert auf Seiten 48, 61, 62 und 116)*

[22] R. Crepaldi, P. Casari, A. Zanella und M. Zorzi. Testbed Implementation and Refinement of a Range-based Localization Algorithm for Wireless Sensor Networks. In: *Proceedings of the 3rd International Conference on Mobile Technology, Applications & Systems*, Mobility '06, New York, NY, USA, 2006. ACM. doi:10.1145/1292331.1292401. *(Zitiert auf Seiten 142 und 146)*

[23] Crossbow Technology Inc. MICA2 Datasheet, Mai 2013. URL: `http://bullseye.xbow.com:81/Products/Product_pdf_files/Wireless_pdf/MICA2_Datasheet.pdf` [Stand: 31.05.2013]. *(Zitiert auf Seite 141)*

[24] M. Cypriani, F. Lassabe, P. Canalda und F. Spies. Wi-Fi-based indoor positioning: Basic techniques, hybrid algorithms and open software platform. In: *Indoor Positioning and Indoor Navigation (IPIN), 2010 International Conference on*, Seiten 1–10, 2010. `doi:10.1109/IPIN.2010.5648232`. *(Zitiert auf Seiten 143 und 147)*

[25] J. E. Dennis, Jr. und R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Society for Industrial and Applied Mathematics, 1996. `doi:10.1137/1.9781611971200`. *(Zitiert auf Seite 46)*

[26] L. Doherty, K. Pister und L. El Ghaoui. Convex Position Estimation in Wireless Sensor Networks. In: *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, Band 3, Seiten 1655–1663, 2001. `doi:10.1109/INFCOM.2001.916662`. *(Zitiert auf Seite 40)*

[27] S. O. Dulman, A. Baggio, P. J. Havinga und K. G. Langendoen. A Geometrical Perspective on Localization. In: *Proceedings of the First ACM International Workshop on Mobile Entity Localization and Tracking in GPS-less Environments*, MELT '08, Seiten 85–90, New York, NY, USA, 2008. ACM. `doi:10.1145/1410012.1410032`. *(Zitiert auf Seite 111)*

[28] S. O. Dulman, P. J. Havinga, A. Baggio und K. G. Langendoen. Revisiting the Cramer-Rao Bound for Localization Algorithms. *4th IEEE/ACM DCOSS Work-in-progress paper*, 2008. *(Zitiert auf Seite 111)*

[29] R. B. Ertel und J. H. Reed. Angle and Time of Arrival Statistics for Circular and Elliptical Scattering Models. *Selected Areas in Communications, IEEE Journal on*, 17(11):1829–1840, Nov. 1999. *(Zitiert auf Seite 11)*

[30] L. Fang, P. Antsaklis, L. Montestruque, M. McMickell, M. Lemmon, Y. Sun, H. Fang, I. Koutroulis, M. Haenggi, M. Xie und X. Xie. Design of a Wireless Assisted Pedestrian Dead Reckoning System - The NavMote Experience. *Instrumentation and Measurement, IEEE Transactions on*, 54(6):2342–2358, 2005. `doi:10.1109/TIM.2005.858557`. *(Zitiert auf Seite 16)*

[31] S.-H. Fang, T.-N. Lin und K.-C. Lee. A Novel Algorithm for Multipath Fin-

gerprinting in Indoor WLAN Environments. *Wireless Communications, IEEE Transactions on*, 7(9):3579–3588, 2008. doi:10.1109/TWC.2008.070373. *(Zitiert auf Seite 14)*

[32] T. Fast, T. Wall und L. Chen. Java Native Access. URL: https://github.com/twall/jna/ [Stand: 31.05.2013]. *(Zitiert auf Seite 128)*

[33] R. Feliz, E. Zalama und J. Gómez. Pedestrian tracking using inertial sensors. *Journal of Physical Agents*, 3(1), 2009. *(Zitiert auf Seite 16)*

[34] R. T. Fielding und R. N. Taylor. Principled design of the modern Web architecture. *ACM Transactions on Internet Technology*, 2(2):115–150, Mai 2002. doi:10.1145/514183.514185. *(Zitiert auf Seite 157)*

[35] D. Fox, W. Burgard, F. Dellaert und S. Thrun. Monte Carlo Localization: Efficient Position Estimation for Mobile Robots. In: *Proc. of the National Conference on Artificial Intelligence*, AAAI '99/IAAI '99, Seiten 343–349, 1999. URL: http://dl.acm.org/citation.cfm?id=315149.315322. *(Zitiert auf Seite 154)*

[36] D. Fox, J. Hightower, L. Liao, D. Schulz und G. Borriello. Bayesian Techniques for Location Estimation. *Pervasive Computing, IEEE*, 2(3):24–33, 2003. doi:10.1109/MPRV.2003.1228524. *(Zitiert auf Seite 40)*

[37] E. Foxlin. Pedestrian Tracking with Shoe-Mounted Inertial Sensors. *Computer Graphics and Applications, IEEE*, 25(6):38–46, 2005. doi:10.1109/MCG.2005.140. *(Zitiert auf Seite 16)*

[38] S. Gezici. A Survey on Wireless Position Estimation. *Wireless Personal Communications*, 44(3):263–282, Feb. 2008. doi:10.1007/s11277-007-9375-z. *(Zitiert auf Seite 52)*

[39] S. Gezici, H. Kobayashi und H. V. Poor. Nonparametric nonline-of-sight identification. In: *Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th*, Band 4, Seiten 2544–2548. IEEE, 2003. doi:10.1109/VETECF.2003.1285996. *(Zitiert auf Seite 96)*

[40] S. Gezici, Z. Tian, G. B. Giannakis, H. Kobayashi, A. F. Molisch, H. V. Poor und Z. Sahinoglu. Localization via ultra-wideband radios: a look at positioning aspects for future sensor networks. *Signal Processing Magazine, IEEE*, 22(4):70–84, 2005. doi:10.1109/MSP.2005.1458289. *(Zitiert auf Seiten 29 und 96)*

[41] G. Giorgetti, R. Farley, K. Chikkappa, J. Ellis und T. Kaleas. Cortina: Collaborative indoor positioning using low-power sensor networks. In: *Indoor Positioning and Indoor Navigation (IPIN), 2011 International Conference on*, Seiten 1–10, 2011. doi:10.1109/IPIN.2011.6071938. *(Zitiert auf Seiten 143 und 147)*

[42] J. González, J. L. Blanco, C. Galindo, A. Ortiz-de Galisteo, J. A. Fernández-Madrigal, F. A. Moreno und J. L. Martínez. Mobile Robot Localization based on Ultra-Wide-Band Ranging: A Particle Filter Approach. *Robotics and Autonomous Systems*, 57(5):496–507, 2009. doi:10.1016/j.robot.2008.10.022. *(Zitiert auf Seite 18)*

[43] Google. Google Web Toolkit Website, Mai 2013. URL: https://developers.google.com/web-toolkit/ [Stand: 30.06.2013]. *(Zitiert auf Seite 157)*

[44] Y. Gu, A. Lo und I. Niemegeers. A Survey of Indoor Positioning Systems for Wireless Personal Networks. *Communications Surveys & Tutorials, IEEE*, 11(1):13–32, 2009. doi:10.1109/SURV.2009.090103. *(Zitiert auf Seite 8)*

[45] F. Gustafsson und F. Gunnarsson. POSITIONING USING TIME-DIFFERENCE OF ARRIVAL MEASUREMENTS. In: *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on*, Band 6, Seiten VI–553–6. IEEE, 2003. doi:10.1109/ICASSP.2003.1201741. *(Zitiert auf Seite 94)*

[46] I. Güvenç und C.-C. Chong. A Survey on TOA Based Wireless Localization and NLOS Mitigation Techniques. *Communications Surveys & Tutorials, IEEE*, 11(3):107–124, 2009. doi:10.1109/SURV.2009.090308. *(Zitiert auf Seite 96)*

[47] I. Güvenç, C.-C. Chong und F. Watanabe. Analysis of a Linear Least-Squares Localization Technique in LOS and NLOS Environments. In: *Vehicular Technology Conference, 2007. VTC2007-Spring. IEEE 65th*, Seiten 1886–1890, 2007. doi:10.1109/VETECS.2007.391. *(Zitiert auf Seite 76)*

[48] I. Güvenç, C.-C. Chong, F. Watanabe und H. Inamura. NLOS identification and weighted least-squares localization for UWB systems using multipath channel statistics. *EURASIP Journal on Advances in Signal Processing*, 2008, Jan. 2008. doi:10.1155/2008/271984. *(Zitiert auf Seite 51)*

[49] L. Haiyong, L. Hui, Z. Fang und P. Jinghua. An Iterative Clustering-Based Localization Algorithm for Wireless Sensor Networks. *China Communica-*

tions, 8(1):58–64, 2011. URL: http://www.chinacommunications.cn/EN/abstract/article_7324.shtml. *(Zitiert auf Seiten 62, 64 und 116)*

[50] J. He, K. Pahlavan, S. Li und Q. Wang. A Testbed for Evaluation of the Effects of Multipath on Performance of TOA-based Indoor Geolocation. *Instrumentation and Measurement, IEEE Transactions on*, 62(8):2237–2247, 2013. doi:10.1109/TIM.2013.2255976. *(Zitiert auf Seiten 144 und 147)*

[51] J. Hightower und G. Borriello. A Survey and Taxonomy of Location Systems for Ubiquitous Computing. Technischer Bericht UW-CSE 01-08-03, IEEE Computer, Aug. 2001. *(Zitiert auf Seiten 8 und 9)*

[52] T. Hillebrandt, H. Will und M. Kyas. LatMath Bibliothek. URL: http://inf.fu-berlin.de/groups/ag-tech/projects/ls2 [Stand: 27.02.2013]. *(Zitiert auf Seite 87)*

[53] T. Hillebrandt, H. Will und M. Kyas. Quantitative and Spatial Evaluation of Distance-Based Localization Algorithms. In: J. M. Krisp (Hrsg.), *Progress in Location-Based Services*, Lecture Notes in Geoinformation and Cartography, Seiten 173–194. Springer Berlin Heidelberg, 2013. doi:10.1007/978-3-642-34203-5_10. *(Zitiert auf Seiten 107 und 290)*

[54] T. Hillebrandt, H. Will und M. Kyas. The Membership Degree Min-Max Localisation Algorithm. *Journal of Global Positioning Systems*, „Im Erscheinen“. *(Zitiert auf Seiten 52 und 54)*

[55] A. Hossain, H. N. Van, Y. Jin und W.-S. Soh. Indoor Localization Using Multiple Wireless Technologies. In: *Mobile Adhoc and Sensor Systems, 2007. MASS 2007. IEEE Internatonal Conference on*, Seiten 1–8, 2007. doi:10.1109/MOBHOC.2007.4428622. *(Zitiert auf Seite 15)*

[56] H. Hur und H.-S. Ahn. A Circuit Design for Ranging Measurement Using Chirp Spread Spectrum Waveform. *Sensors Journal, IEEE*, 10(11):1774–1778, 2010. doi:10.1109/JSEN.2010.2049488. *(Zitiert auf Seite 94)*

[57] Intel Corporation. *Intel® 64 and IA-32 Architectures Software Developer's Manual Combined Volumes: 1, 2A, 2B, 2C, 3A, 3B and 3C*. Intel Corporation, März 2013. *(Zitiert auf Seite 125)*

[58] M. Irsigler. *Multipath Propagation, Mitigation and Monitoring in the Light of Galileo and the Modernized GPS*. Doktorarbeit, Universität der Bundeswehr München, Deutschland, Juli 2008. *(Zitiert auf Seite 95)*

[59] M. Jadliwala, S. Zhong, S. Upadhyaya, C. Qiao und J.-P. Hubaux. Secure Distance-Based Localization in the Presence of Cheating Beacon Nodes. *Mobile Computing, IEEE Transactions on*, 9(6):810–823, 2010. doi:10.1109/TMC.2010.20. *(Zitiert auf Seiten 81 und 119)*

[60] X. Ji und H. Zha. Sensor positioning in wireless ad-hoc sensor networks using multidimensional scaling. In: *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, Band 4, Seiten 2652–2661, 2004. doi:10.1109/INFCOM.2004.1354684. *(Zitiert auf Seite 116)*

[61] D. Johnson, T. Stack, R. Fish, D. M. Flickinger, L. Stoller, R. Ricci und J. Lepreau. Mobile Emulab: A Robotic Wireless and Sensor Network Testbed. In: *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, Seiten 1–12, 2006. doi:10.1109/INFOCOM.2006.182. *(Zitiert auf Seiten 144 und 146)*

[62] D. Jourdan, J. Deyst, Jr., M. Win und N. Roy. Monte Carlo localization in dense multipath environments using UWB ranging. In: *Ultra-Wideband, 2005. ICU 2005. 2005 IEEE International Conference on*, Seiten 314–319, 2005. doi:10.1109/ICU.2005.1570005. *(Zitiert auf Seite 18)*

[63] J. M. Kahn, R. H. Katz und K. S. J. Pister. Next Century Challenges: Mobile Networking for „Smart Dust“. In: *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, MobiCom '99, Seiten 271–278, New York, NY, USA, 1999. ACM. doi:10.1145/313451.313558. *(Zitiert auf Seite 99)*

[64] O. Kaltiokallio, M. Bocca und N. Patwari. Enhancing the Accuracy of Radio Tomographic Imaging Using Channel Diversity. In: *the 9th IEEE International Conference on Mobile Ad hoc and Sensor Systems*, Seiten 254–262, Las Vegas, USA, Okt. 2012. IEEE. *(Zitiert auf Seiten 15 und 16)*

[65] T. King, S. Kopf, T. Haenselmann, C. Lubberger und W. Effelsberg. COMPASS: A probabilistic indoor positioning system based on 802.11 and digital compasses. In: *Proceedings of the 1st international workshop on Wireless network testbeds, experimental evaluation & characterization*, WiNTECH '06, Seiten 34–40, New York, NY, USA, 2006. ACM. doi:10.1145/1160987.1160995. *(Zitiert auf Seiten 143 und 146)*

[66] R. Kuang, H. Song und G. Wang. Target localization via correlated link in-

ference. In: *Mechatronics and Automation (ICMA), 2011 International Conference on*, Seiten 1010–1014. IEEE, 2011. doi:10.1109/ICMA.2011.5985798. *(Zitiert auf Seite 15)*

[67] H. W. Kuhn. A note on Fermat's problem. *Mathematical Programming*, 4:98–107, 1973. doi:10.1007/BF01584648. *(Zitiert auf Seite 59)*

[68] P. Kułakowski, J. Vales-Alonso, E. Egea-López, W. Ludwin und J. García-Haro. Angle-of-arrival localization based on antenna arrays for wireless sensor networks. *Computers & Electrical Engineering*, 36(6):1181–1186, 2010. doi:10.1016/j.compeleceng.2010.03.007. *(Zitiert auf Seite 11)*

[69] H. T. Kung, C.-K. Lin, T.-H. Lin und D. Vlah. Localization with snap-inducing shaped residuals (SISR): coping with errors in measurement. In: *Proceedings of the 15th annual international conference on Mobile computing and networking*, MobiCom '09, Seiten 333–344, New York, NY, USA, 2009. ACM. doi:10.1145/1614320.1614357. *(Zitiert auf Seiten 42, 143 und 146)*

[70] G. Kuruoglu, M. Erol und S. Oktug. Localization in Wireless Sensor Networks with Range Measurement Errors. In: *Telecommunications, 2009. AICT '09. Fifth Advanced International Conference on*, Seiten 261–266, 2009. doi:10.1109/AICT.2009.51. *(Zitiert auf Seiten 55, 56, 118, 255 und 290)*

[71] M. Laaraiedh, B. Uguen, J. Stephan, Y. Corre, Y. Lostanlen, M. Raspopoulos und S. Stavrou. Ray tracing-based radio propagation modeling for indoor localization purposes. In: *Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), 2012 IEEE 17th International Workshop on*, Seiten 276–280, 2012. doi:10.1109/CAMAD.2012.6335350. *(Zitiert auf Seite 107)*

[72] J. Lampe und Z. Ianelli. Introduction to Chirp Spread Spectrum (CSS) Technology. *Nanotron Technologies*, Juli 2003. *(Zitiert auf Seite 94)*

[73] K. Langendoen und N. Reijers. Distributed localization in wireless sensor networks: a quantitative comparison. *Computer Networks*, 43(4):499–518, Nov. 2003. doi:10.1016/S1389-1286(03)00356-6. *(Zitiert auf Seiten 42 und 148)*

[74] S. Lanzisera, D. Zats und K. Pister. Radio Frequency Time-of-Flight Distance Measurement for Low-Cost Wireless Sensor Localization. *Sensors Journal, IEEE*, 11(3):837–845, 2011. doi:10.1109/JSEN.2010.2072496. *(Zitiert auf Seite 27)*

[75] L. Lazos und R. Poovendran. SeRLoc: Robust Localization for Wireless Sensor Networks. *ACM Transactions on Sensor Networks (TOSN)*, 1(1):73–100, Aug. 2005. *(Zitiert auf Seite 42)*

[76] K. Levenberg. A Method for the Solution of Certain Non-Linear Problems in Least Squares. *The Quarterly of Applied Mathematics*, 2:164–168, 1944. *(Zitiert auf Seite 46)*

[77] P. Levis, S. Madden, J. Polastre, R. Szewczyk, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer und D. Culler. TinyOS: An Operating System for Sensor Networks. In: W. Weber, J. Rabaey und E. Aarts (Hrsg.), *Ambient Intelligence*, Seiten 115–148. Springer Berlin Heidelberg, 2005. doi:10.1007/3-540-27139-2_7. *(Zitiert auf Seite 141)*

[78] X. Li, B. Hua, Y. Shang und Y. Xiong. A robust localization algorithm in wireless sensor networks. *Frontiers of Computer Science in China*, 2(4):438–450, 2008. doi:10.1007/s11704-008-0018-7. *(Zitiert auf Seite 117)*

[79] Z. Li, W. Trappe, Y. Zhang und B. Nath. Robust statistical methods for securing wireless localization in sensor networks. In: *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*, Seiten 91–98, 2005. doi:10.1109/IPSN.2005.1440903. *(Zitiert auf Seiten 59, 60 und 118)*

[80] H. Linde. *On Aspects of Indoor Localization.* Doktorarbeit, Universität Dortmund, Deutschland, Aug. 2006. *(Zitiert auf Seite 93)*

[81] C. Liu, K. Wu und T. He. Sensor localization with Ring Overlapping based on Comparison of Received Signal Strength Indicator. In: *Mobile Ad-hoc and Sensor Systems, 2004 IEEE International Conference on*, Seiten 516–518, 2004. doi:10.1109/MAHSS.2004.1392193. *(Zitiert auf Seite 143)*

[82] D. Liu, P. Ning und W. K. Du. Attack-Resistant Location Estimation in Sensor Networks. In: *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*, IPSN '05, Seiten 99–106, Piscataway, NJ, USA, 2005. IEEE Press. doi:10.1109/IPSN.2005.1440904. *(Zitiert auf Seiten 66, 68, 70, 141 und 146)*

[83] D. Liu, P. Ning, A. Liu, C. Wang und W. K. Du. Attack-Resistant Location Estimation in Wireless Sensor Networks. *ACM Transactions on Information and System Security (TISSEC)*, 11(4):22:1–22:39, Juli 2008. doi:10.1145/

1380564.1380570. *(Zitiert auf Seite 117)*

[84] H. Liu, H. Darabi, P. Banerjee und J. Liu. Survey of Wireless Indoor Positioning Techniques and Systems. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 37(6):1067–1080, 2007. doi:10.1109/TSMCC.2007.905750. *(Zitiert auf Seiten 8 und 14)*

[85] G. Mao und B. Fidan. *Localization Algorithms and Strategies for Wireless Sensor Networks: Monitoring and Surveillance Techniques for Target Tracking*. Information Science Reference - Imprint of: IGI Publishing, Hershey, PA, USA, 2009. doi:10.4018/978-1-60566-396-8. *(Zitiert auf Seite 10)*

[86] M. Maróti, P. Völgyesi, S. Dóra, B. Kusý, A. Nádas, A. Lédeczi, G. Balogh und K. Molnár. Radio interferometric geolocation. In: *Proceedings of the 3rd international conference on Embedded networked sensor systems*, SenSys '05, Seiten 1–12, New York, NY, USA, 2005. ACM. doi:10.1145/1098918.1098920. *(Zitiert auf Seite 36)*

[87] D. W. Marquardt. An Algorithm for Least-Squares Estimation of Nonlinear Parameters. *SIAM Journal on Applied Mathematics*, 11(2):431–441, 1963. URL: http://www.jstor.org/stable/2098941. *(Zitiert auf Seite 46)*

[88] E. Martin, O. Vinyals, G. Friedland und R. Bajcsy. Precise Indoor Localization Using Smart Phones. In: *Proceedings of the International Conference on Multimedia*, MM'10, Seiten 787–790, New York, NY, USA, 2010. ACM. doi:10.1145/1873951.1874078. *(Zitiert auf Seite 14)*

[89] D. McCrady, L. Doyle, H. Forstrom, T. Dempsey und M. Martorana. Mobile ranging using low-accuracy clocks. *Microwave Theory and Techniques, IEEE Transactions on*, 48(6):951–958, 2000. doi:10.1109/22.846721. *(Zitiert auf Seite 27)*

[90] W. Meng, W. Xiao, W. Ni und L. Xie. Secure and robust Wi-Fi fingerprinting indoor localization. In: *Indoor Positioning and Indoor Navigation (IPIN), 2011 International Conference on*, Seiten 1–7, 2011. doi:10.1109/IPIN.2011.6071908. *(Zitiert auf Seiten 143 und 147)*

[91] S. Misra, S. Bhardwaj und G. Xue. ROSETTA: Robust And Secure Mobile Target Tracking In A Wireless Ad Hoc Environment. In: *Military Communications Conference, 2006. MILCOM 2006. IEEE*, Seiten 1–7, 2006. doi:10.1109/MILCOM.2006.302534. *(Zitiert auf Seite 81)*

[92] S. Misra und G. Xue. CluRoL: Clustering based Robust Localization in Wireless Sensor Networks. In: *Military Communications Conference, 2007. MILCOM 2007. IEEE*, Seiten 1–7, 2007. doi:10.1109/MILCOM.2007.4454815. *(Zitiert auf Seiten 64, 66, 117 und 249)*

[93] D. Moore, J. Leonard, D. Rus und S. Teller. Robust Distributed Network Localization with Noisy Range Measurements. In: *Proceedings of the 2Nd International Conference on Embedded Networked Sensor Systems*, SenSys '04, Seiten 50–61, New York, NY, USA, 2004. ACM. doi:10.1145/1031495.1031502. *(Zitiert auf Seiten 142 und 146)*

[94] L. Mottola und G. P. Picco. Programming Wireless Sensor Networks: Fundamental Concepts and State of the Art. *ACM Computing Surveys*, 43(3):19:1–19:51, Apr. 2011. doi:10.1145/1922649.1922656. *(Zitiert auf Seite 100)*

[95] Nanotron Technologies GmbH. Nanopan 5375 RF module datasheet, 2009. URL: http://www.nanotron.com. *(Zitiert auf Seiten 28 und 237)*

[96] S. Nawaz und N. Trigoni. Robust localization in cluttered environments with NLOS propagation. In: *Mobile Adhoc and Sensor Systems (MASS), 2010 IEEE 7th International Conference on*, Seiten 166–175, Nov. 2010. doi:10.1109/MASS.2010.5663983. *(Zitiert auf Seiten 46, 50, 118 und 147)*

[97] J. Nocedal und S. J. Wright. *Numerical Optimization*. Springer Science+Business Media, LLC, New York, NY, USA, 2. Aufl., 2006. *(Zitiert auf Seite 46)*

[98] V. Otsason, A. Varshavsky, A. LaMarca und E. Lara. Accurate GSM Indoor Localization. In: M. Beigl, S. Intille, J. Rekimoto und H. Tokuda (Hrsg.), *UbiComp 2005: Ubiquitous Computing*, Band 3660 d. Reihe *Lecture Notes in Computer Science*, Seiten 141–158. Springer Berlin Heidelberg, 2005. doi:10.1007/11551201_9. *(Zitiert auf Seite 14)*

[99] K. Owens und R. D. Parikh. Fast Random Number Generator on the Intel® Pentium® 4 Processor, März 2012. URL: http://software.intel.com/en-us/articles/fast-random-number-generator-on-the-intel-pentiumr-4-processor [Stand: 30.06.2013]. *(Zitiert auf Seite 126)*

[100] K. Pahlavan, X. Li und J.-P. Makela. Indoor Geolocation Science and Technology. *Communications Magazine, IEEE*, 40(2):112–118, Feb. 2002. doi:

10.1109/35.983917. *(Zitiert auf Seiten 35 und 36)*

[101] R. Peng und M. Sichitiu. Angle of Arrival Localization for Wireless Sensor Networks. In: *Sensor and Ad Hoc Communications and Networks, 2006. SECON '06. 2006 3rd Annual IEEE Communications Society on*, Band 1, Seiten 374–382, 2006. doi:10.1109/SAHCN.2006.288442. *(Zitiert auf Seite 11)*

[102] A. Pogo. Gemma Frisius, His Method of Determining Differences of Longitude by Transporting Timepieces (1530), and His Treatise on Triangulation (1533). *Isis*, 22(2):469–506, Feb. 1935. URL: http://www.jstor.org/stable/225130. *(Zitiert auf Seite 8)*

[103] W. H. Press, S. A. Teukolsky, W. T. Vetterling und B. P. Flannery. *Numerical Recipes: The Art of Scientific Computing*. Cambridge University Press, 3. Aufl., 2007. *(Zitiert auf Seite 46)*

[104] N. B. Priyantha, A. Chakraborty und H. Balakrishnan. The Cricket location-support system. In: *Proceedings of the 6th annual international conference on Mobile computing and networking*, MobiCom '00, Seiten 32–43, New York, NY, USA, 2000. ACM. doi:10.1145/345910.345917. *(Zitiert auf Seiten 29, 34 und 142)*

[105] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler und A. Y. Ng. ROS: an open-source Robot Operating System. In: *ICRA Workshop on Open Source Software*, 2009. *(Zitiert auf Seite 154)*

[106] S. Rallapalli, L. Qiu, Y. Zhang und Y.-C. Chen. Exploiting Temporal Stability and Low-Rank Structure for Localization in Mobile Networks. In: *Proceedings of the sixteenth annual international conference on Mobile computing and networking*, MobiCom '10, Seiten 161–172, New York, NY, USA, 2010. ACM. doi:10.1145/1859995.1860015. *(Zitiert auf Seiten 143 und 147)*

[107] B. Rao und L. Minakakis. Evolution of Mobile Location-based Services. *Commun. ACM*, 46(12):61–65, Dez. 2003. doi:10.1145/953460.953490. *(Zitiert auf Seite 1)*

[108] P. Reinecke. *Efficient System Evaluation Using Stochastic Models*. Doktorarbeit, Freie Universität Berlin, Deutschland, 2012. *(Zitiert auf Seiten 127 und 153)*

[109] A. Rice und R. Harle. Evaluating Lateration-Based Positioning Algorithms for Fine-Grained Tracking. In: *Proceedings of the 2005 joint workshop on Founda-*

tions of mobile computing, DIALM-POMC '05, Seiten 54–61, New York, NY, USA, 2005. ACM. doi:10.1145/1080810.1080820. *(Zitiert auf Seite 42)*

[110] J. J. Robles, J. S. Pola und R. Lehnert. Extended Min-Max algorithm for position estimation in sensor networks. In: *Positioning Navigation and Communication (WPNC), 2012 9th Workshop on*, Seiten 47–52, 2012. doi:10.1109/WPNC.2012.6268737. *(Zitiert auf Seiten 44 und 118)*

[111] K. Römer. The Lighthouse Location System for Smart Dust. In: *Proceedings of the 1st International Conference on Mobile Systems, Applications and Services*, MobiSys '03, Seiten 15–30, New York, NY, USA, 2003. ACM. doi:10.1145/1066116.1189036. *(Zitiert auf Seite 10)*

[112] P. J. Rousseeuw und A. M. Leroy. *Robust Regression and Outlier Detection.* John Wiley & Sons, Inc., New York, NY, USA, 1987. *(Zitiert auf Seiten 59 und 60)*

[113] L. Rui und K. C. Ho. Bias analysis of source localization using the maximum likelihood estimator. In: *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, Seiten 2605–2608, 2012. doi:10.1109/ICASSP.2012.6288450. *(Zitiert auf Seite 292)*

[114] B. Sadler, N. Liu, Z. Xu und R. Kozick. Range-Based Geolocation in Fading Environments. In: *Communication, Control, and Computing, 2008 46th Annual Allerton Conference on*, Seiten 15–20, 2008. doi:10.1109/ALLERTON.2008.4797529. *(Zitiert auf Seite 292)*

[115] Z. Sahinoglu, S. Gezici und I. Güvenc. *Ultra-wideband Positioning Systems: Theoretical Limits, Ranging Algorithms, and Protocols.* Cambridge University Press, 2008. *(Zitiert auf Seite 29)*

[116] A. Saleh und R. Valenzuela. A Statistical Model for Indoor Multipath Propagation. *Selected Areas in Communications, IEEE Journal on*, 5(2):128–137, 1987. doi:10.1109/JSAC.1987.1146527. *(Zitiert auf Seite 96)*

[117] A. Savvides, H. Park und M. B. Srivastava. The Bits and Flops of the N-hop Multilateration Primitive for Node Localization Problems. In: *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, WSNA '02, Seiten 112–121, New York, NY, USA, 2002. ACM. doi:10.1145/570738.570755. *(Zitiert auf Seite 42)*

[118] S. Schmitt, H. Will, B. Aschenbrenner, T. Hillebrandt und M. Kyas. A Re-

ference System for Indoor Localization Testbeds. In: *Indoor Positioning and Indoor Navigation (IPIN), 2012 International Conference on*, Seiten 1–8, 2012. doi:10.1109/IPIN.2012.6418865. *(Zitiert auf Seite 155)*

[119] S. Schmitt, H. Will, T. Hillebrandt und M. Kyas. A Virtual Indoor Localization Testbed for Wireless Sensor Networks. In: *Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2013 10th Annual IEEE Communications Society Conference on*, Seiten 239–241, New Orleans, LA, USA, Juni 2013. doi:10.1109/SAHCN.2013.6644985. *(Zitiert auf Seiten 155 und 159)*

[120] Y. Shang, W. Ruml, Y. Zhang und M. P. J. Fromherz. Localization from Mere Connectivity. In: *Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking &Amp; Computing*, MobiHoc '03, Seiten 201–212, New York, NY, USA, 2003. ACM. doi:10.1145/778415.778439. *(Zitiert auf Seite 40)*

[121] M. Silventoinen und T. Rantalainen. Mobile station emergency locating in GSM. In: *Personal Wireless Communications, 1996., IEEE International Conference on*, Seiten 232–238, 1996. doi:10.1109/ICPWC.1996.494274. *(Zitiert auf Seite 95)*

[122] Q. H. Spencer, B. D. Jeffs, M. A. Jensen und A. L. Swindlehurst. Modeling the Statistical Time and Angle of Arrival Characteristics of an Indoor Multipath Channel. *Selected Areas in Communications, IEEE Journal on*, 18(3):347–360, 2000. doi:10.1109/49.840194. *(Zitiert auf Seite 11)*

[123] S. Szilvasi, P. Volgyesi, J. Sallai, A. Ledeczi und M. Maroti. *Interferometry - Research and Applications in Science and Technology*, Kapitel: Interferometry in Wireless Sensor Networks, Seiten 437–462. InTech, 2012. doi:10.5772/36492. *(Zitiert auf Seite 36)*

[124] The Apache Software Foundation. Commons Math: The Apache Commons Mathematics Library, Mai 2013. URL: http://commons.apache.org/proper/commons-math/ [Stand: 31.05.2013]. *(Zitiert auf Seite 55)*

[125] The HDF Group. What is HDF5?, Mai 2013. URL: http://www.hdfgroup.org/HDF5/whatishdf5.html [Stand: 30.06.2013]. *(Zitiert auf Seite 134)*

[126] The PostGIS development team. PostGIS Website, Mai 2013. URL: http://www.postgis.net/ [Stand: 30.06.2013]. *(Zitiert auf Seite 155)*

[127] The PostgreSQL Global Development Group. PostgreSQL Website, Mai 2013.

URL: http://www.postgresql.org/ [Stand: 30.06.2013]. *(Zitiert auf Seite 155)*

[128] The R Foundation for Statistical Computing. The R Project for Statistical Computing, Mai 2013. URL: http://www.r-project.org/ [Stand: 31.05.2013]. *(Zitiert auf Seite 245)*

[129] TinyOS Alliance. TinyOS Website, Mai 2013. URL: http://www.tinyos.net/ [Stand: 31.05.2013]. *(Zitiert auf Seite 141)*

[130] u-blox AG. u-blox Automotive Dead Reckoning technology, Juni 2013. URL: http://www.u-blox.de/de/dead-reckoning.html [Stand: 28.11.2013]. *(Zitiert auf Seite 16)*

[131] A. Varshavsky, E. de Lara, J. Hightower, A. LaMarca und V. Otsason. GSM Indoor Localization. *Pervasive and Mobile Computing*, 3(6):698–720, Dez. 2007. doi:10.1016/j.pmcj.2007.07.004. *(Zitiert auf Seite 14)*

[132] S. Venkatesh und R. M. Buehrer. A Linear Programming Approach to NLOS Error Mitigation in Sensor Networks. In: *Proceedings of the 5th International Conference on Information Processing in Sensor Networks*, IPSN '06, Seiten 301–308, New York, NY, USA, 2006. ACM. doi:10.1145/1127777.1127823. *(Zitiert auf Seiten 51, 76 und 96)*

[133] S. Venkatraman, J. Caffery Jr. und H.-R. You. A novel ToA location algorithm using LoS range estimation for NLoS environments. *Vehicular Technology, IEEE Transactions on*, 53(5):1515–1524, 2004. doi:10.1109/TVT.2004.832384. *(Zitiert auf Seite 76)*

[134] H. Wang, H. Lenz, A. Szabo, J. Bamberger und U. Hanebeck. WLAN-Based Pedestrian Tracking Using Particle Filters and Low-Cost MEMS Sensors. In: *Positioning, Navigation and Communication, 2007. WPNC '07. 4th Workshop on*, Seiten 1–7, 2007. doi:10.1109/WPNC.2007.353604. *(Zitiert auf Seite 22)*

[135] H. Wang, H. Lenz, A. Szabo, U. D. Hanebeck und J. Bamberger. Fusion of Barometric Sensors, WLAN Signals and Building Information for 3-D Indoor/-Campus Localization. In: *Proceedings of International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI 2006)*, Seiten 426–432, 2006. *(Zitiert auf Seite 22)*

[136] Y. Wang, X. Jia, H. Lee und G. Li. An indoors wireless positioning system based on wireless local area network infrastructure. In: *6th International Symposium*

on Satellite Navigation Technology Including Mobile Positioning & Location Services, Juli 2003. *(Zitiert auf Seite 14)*

[137] R. Want, A. Hopper, V. Falcão und J. Gibbons. The active badge location system. *ACM Transactions on Information Systems*, 10(1):91–102, Jan. 1992. doi:10.1145/128756.128759. *(Zitiert auf Seite 8)*

[138] B. Warneke, M. Last, B. Liebowitz und K. Pister. Smart Dust: communicating with a cubic-millimeter computer. *Computer*, 34(1):44–51, 2001. doi:10.1109/2.895117. *(Zitiert auf Seite 99)*

[139] E. Weiszfeld. Sur le point pour lequel la somme des distances de n points donnés est minimum. *Tôhoku Mathematical Journal*, 43:355–386, 1937. *(Zitiert auf Seite 59)*

[140] E. Weiszfeld und F. Plastria. On the point for which the sum of the distances to n given points is minimum. *Annals of Operations Research*, 167(1):7–41, 2009. doi:10.1007/s10479-008-0352-z. *(Zitiert auf Seite 59)*

[141] B. Welford. Note on a Method for Calculating Correct Sums of Squares and Products. *Technometrics*, 4(3):419–420, Aug. 1962. *(Zitiert auf Seite 79)*

[142] J. Wendeberg, J. Muller, C. Schindelhauer und W. Burgard. Robust Tracking of a Mobile Beacon using Time Differences of Arrival with Simultaneous Calibration of Receiver Positions. In: *Indoor Positioning and Indoor Navigation (IPIN), 2012 International Conference on*, Seiten 1–10, 2012. doi:10.1109/IPIN.2012.6418919. *(Zitiert auf Seiten 142 und 147)*

[143] K. Whitehouse und D. Culler. A Robustness Analysis of Multi-hop Ranging-based Localization Approximations. In: *Proceedings of the 5th International Conference on Information Processing in Sensor Networks*, IPSN '06, Seiten 317–325, New York, NY, USA, 2006. ACM. doi:10.1145/1127777.1127825. *(Zitiert auf Seite 42)*

[144] K. Whitehouse, C. Karlof und D. Culler. A Practical Evaluation of Radio Signal Strength for Ranging-based Localization. *SIGMOBILE Mobile Computing and Communications Review*, 11(1):41–52, Jan. 2007. doi:10.1145/1234822.1234829. *(Zitiert auf Seiten 24 und 26)*

[145] K. Whitehouse, C. Karlof, A. Woo, F. Jiang und D. Culler. The Effects of Ranging Noise on Multihop Localization: An Empirical Study. In: *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks*,

IPSN '05, Piscataway, NJ, USA, 2005. IEEE Press. *(Zitiert auf Seiten 142 und 146)*

[146] Widyawan, M. Klepal und S. Beauregard. A Novel Backtracking Particle Filter for Pattern Matching Indoor Localization. In: *Proceedings of the First ACM International Workshop on Mobile Entity Localization and Tracking in GPS-less Environments*, MELT '08, Seiten 79–84, New York, NY, USA, 2008. ACM. doi:10.1145/1410012.1410031. *(Zitiert auf Seite 15)*

[147] H. Will, T. Hillebrandt und M. Kyas. LS² - Lateration Simulator, Feb. 2012. URL: http://inf.fu-berlin.de/groups/ag-tech/projects/ls2 [Stand: 27.02.2013]. *(Zitiert auf Seite 121)*

[148] H. Will, T. Hillebrandt und M. Kyas. The FU Berlin Parallel Lateration-Algorithm Simulation and Visualization Engine. In: *Positioning Navigation and Communication (WPNC), 2012 9th Workshop on*, Seiten 131–136, 2012. doi:10.1109/WPNC.2012.6268752. *(Zitiert auf Seiten 97, 107 und 121)*

[149] H. Will, T. Hillebrandt und M. Kyas. The Geo-n Localization Algorithm. In: *Indoor Positioning and Indoor Navigation (IPIN), 2012 International Conference on*, Seiten 1–10, 2012. doi:10.1109/IPIN.2012.6418867. *(Zitiert auf Seiten 79, 83 und 84)*

[150] H. Will, T. Hillebrandt und M. Kyas. Wireless Sensor Networks in Emergency Scenarios: The FeuerWhere Deployment. In: *Proceedings of the 1st ACM International Workshop on Sensor-Enhanced Safety and Security in Public Spaces*, SESP '12, Seiten 9–14, New York, NY, USA, 2012. ACM. doi:10.1145/2248356.2248360. *(Zitiert auf Seiten 107 und 237)*

[151] H. Will, T. Hillebrandt, Y. Yuan, Z. Yubin und M. Kyas. The Membership Degree Min-Max Localization Algorithm. In: *Ubiquitous Positioning, Indoor Navigation, and Location Based Service (UPINLBS'12)*, Seiten 1–10, 2012. doi:10.1109/UPINLBS.2012.6409781. *(Zitiert auf Seite 72)*

[152] H. Will, S. Pfeiffer, S. Adler, T. Hillebrandt und J. Schiller. Distance Measurement in Wireless Sensor Networks with Low Cost Components. In: *Proceedings of the 2nd International Conference on Indoor Positioning and Indor Navigation (IPIN 2011)*, 2011. *(Zitiert auf Seite 94)*

[153] H. Will, J. Pfender, S. Adler, T. Hillebrandt und M. Kyas. A Greedy Approach to Cooperative Indoor Localization. In: *Proceedings of the 3rd International*

Conference on Indoor Positioning and Indoor Navigation (IPIN 2012), 2012. *(Zitiert auf Seite 95)*

[154] C. J. Willmott und K. Matsuura. Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. *Climate Research*, 30(1):79–82, 2005. doi:10.3354/cr030079. *(Zitiert auf Seite 106)*

[155] M. Wylie und J. Holtzman. The Non-Line of Sight Problem in Mobile Location Estimation. In: *Universal Personal Communications, 1996. Record., 1996 5th IEEE International Conference on*, Band 2, Seiten 827–831, 1996. doi:10.1109/ICUPC.1996.562692. *(Zitiert auf Seite 96)*

[156] K. Yedavalli, B. Krishnamachari, S. Ravula und B. Srinivasan. Ecolocation: A Sequence Based Technique for RF Localization in Wireless Sensor Networks. In: *Proceedings of the 4th international symposium on Information processing in sensor networks*, IPSN '05, Piscataway, NJ, USA, 2005. IEEE Press. URL: http://dl.acm.org/citation.cfm?id=1147685.1147733. *(Zitiert auf Seite 25)*

[157] M. Youssef und A. Agrawala. The Horus location determination system. *Wireless Networks*, 14(3):357–374, Juni 2008. doi:10.1007/s11276-006-0725-7. *(Zitiert auf Seiten 143 und 146)*

[158] L. Zadeh. Fuzzy Sets. *Information and Control*, 8(3):338–353, 1965. *(Zitiert auf Seite 74)*

[159] G. Zanca, F. Zorzi, A. Zanella und M. Zorzi. Experimental comparison of RSSI-based localization algorithms for indoor wireless sensor networks. In: *Proceedings of the workshop on Real-world wireless sensor networks*, REALWSN '08, Seiten 1–5, New York, NY, USA, 2008. ACM. doi:10.1145/1435473.1435475. *(Zitiert auf Seiten 143 und 146)*

[160] R. Zekavat und R. Buehrer. *Handbook of Position Location: Theory, Practice and Advances*. IEEE Series on Digital & Mobile Communication. Wiley-IEEE Press, 2011. doi:10.1002/9781118104750. *(Zitiert auf Seiten 29, 32, 53 und 289)*

[161] Y. Zhao und N. Patwari. Histogram Distance-based Radio Tomographic Localization. In: *Proceedings of the 11th International Conference on Information Processing in Sensor Networks*, IPSN '12, Seiten 129–130, New York, NY, USA,

2012. ACM. doi:10.1145/2185677.2185712. *(Zitiert auf Seite 15)*

[162] T. Zhou. An empirical examination of user adoption of location-based services. *Electronic Commerce Research*, 13(1):25–39, 2013. doi:10.1007/s10660-013-9106-3. *(Zitiert auf Seite 1)*

[163] K. Zickuhr. Three-quarters of smartphone owners use location-based services. *Pew Internet & American Life Project*, 2012. *(Zitiert auf Seite 1)*