Technische Universität Ilmenau
Fakultät für Informatik und Automatisierung
Fachgebiet Neuroinformatik und Kognitive Robotik

Graphbasierte Lokalisation und 3D-Mapping für mobile Roboter

Masterarbeit zur Erlangung des akademischen Grades Master of Science

Christian Reuther

Betreuer: Dipl.-Inf. Erik Einhorn,
FG Neuroinformatik und Kognitive Robotik
Verantwortlicher Hochschullehrer:
Prof. Dr. H.-M. Groß,
FG Neuroinformatik und Kognitive Robotik

Die Masterarbeit wurde am 04.02.2013 bei der Fakultät für Informatik und Automatisierung der Technischen Universität Ilmenau eingereicht.

Erklärung: „Hiermit versichere ich, dass ich diese Masterarbeit selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel verwendet habe. Alle von mir aus anderen Veröffentlichungen übernommenen Passagen sind als solche gekennzeichnet.“

Ilmenau, 04.02.2013 .....

Christian Reuther

Inhaltsverzeichnis

KAPITEL 1

Einleitung

> *Cartography [is] the art and science of graphically representing a geographical area, usually on a flat surface such as a map or chart.*
>
> – Encyclopaedia Britannica, *Cartography*

Seit frühen Phasen der menschlichen Geschichte erstellen Menschen Karten, um ihr gesammeltes Wissen über geographische Merkmale der Umwelt festzuhalten. Wo einst die relative Lage markanter Landmarken als Höhlenmalereien beschrieben wurde, existiert heute eine genaue, flächendeckende Abbildung der tatsächlichen Erdoberfläche durch Satellitenaufnahmen [Britannica, 2012]. Auch die Werkzeuge zur Lokalisation und Navigation haben eine derartige Entwicklung erfahren. Begonnen bei astronomischer Navigation bis hin zu GPS, konnte die Menschheit ihre Position anhand von Karten immer genauer bestimmen. Die Verfügbarkeit guter Karten und Lokalisationswerkzeuge ist daher für viele Menschen heutzutage selbstverständlich.

Im aufstrebenden Gebiet der mobilen Robotik wird versucht, dieses weitreichende menschliche Wissen über Kartierung, Lokalisation und Navigation auf mobile Roboter zu übertragen. Autonomes Verhalten wie beispielsweise das trivial erscheinende Navigieren von Punkt A zu Punkt B erfordert Kenntnis über die eigene Position in der Einsatzumgebung und deren Beschaffenheit. Fehlerbehaftete Sensoren und eingeschränkte Fähigkeiten zur semantischen Interpretation der Messwerte erschweren jedoch das autonome Erstellen topologisch korrekter Karten, sowie die Lokalisation in der kartierten Umgebung. Ein wichtiger Forschungsbereich der mobilen Robotik versucht daher, durch gleichzeitige Lokalisation und Kartierung eine möglichst plausible Erklärung der unsicheren beobachteten Umgebungsmerkmale zu finden. In den letzten

20 Jahren wurden viele potentielle Lösungsansätze für dieses sog. SLAM-Problem (engl. *Simultaneous Localization and Mapping*) präsentiert, häufig basierend auf der rekursiven Schätzung multivariater Wahrscheinlichkeitsverteilungen über den Roboter- und Umweltzustand. Ein neuerer Ansatz speichert die gesamten erhaltenen Messwerte als beschränkende Bedingungen in einem topologisch korrekten Graphen, um die bestmögliche Schätzung über Roboter- und Umweltzustand durch nichtlineare Optimierungsmethoden zu ermitteln. Derartige Verfahren weisen gegenüber der rekursiven Zustandsschätzung deutliche Vorteile auf und gelten daher als die momentan geeignetsten Kandidaten zur Lösung des SLAM-Problems.

1.1 Kurzfassung

In dieser Arbeit soll ein solcher graphbasierter SLAM-Algorithmus zur robusten und möglichst echtzeitfähigen Erstellung dreidimensionaler Karten erarbeitet werden. Derartige Verfahren lassen sich grob in zwei Bestandteile unterteilen: das SLAM-Frontend und das SLAM-Backend. Im Fokus dieser Arbeit steht das SLAM-Frontend, das für das Erzeugen eines topologisch korrekten Posengraphen aus den erhaltenen Sensormessungen verantwortlich ist. Eine erste Schätzung der Struktur dieses Posengraphen wird durch die Odometriedaten des Roboters ermittelt, die jedoch einem schnell akkumulierenden Fehler unterliegen. Aus diesem Grund werden im Frontend Messungen dreidimensionaler Tiefensensoren verglichen, um durch die sog. Registrierung redundante und meist genauere Informationen über den geometrischen Versatz zweier Posen zu erhalten. Zur Registrierung von Messungen mit relativ genau bekannten Aufnahmeposen wird ein Levenberg-Marquardt-Algorithmus implementiert, der robuste Konvergenzeigenschaften und geringe Laufzeit bietet. Zur Registrierung von Beobachtungen mit nur sehr ungenau bekannten Aufnahmeposen, beispielsweise beim Schleifenschluss, wird ein partikelschwarmbasiertes Optimierungsverfahren implementiert, das gut mit dieser hohen Unsicherheit umgehen kann. Zum effizienten Umgang mit den dreidimensionalen Sensordaten nutzt das Frontend den sog. Normal Distributions Transform, der eine speicherplatzeffiziente Beschreibung dreidimensionaler Oberflächen ermöglicht. Das erarbeitete SLAM-Frontend wird im Anschluss an das frei verfügbare SLAM-Backend

g2o angebunden, um die Qualität des erzeugten Posengraphen zu evaluieren. Anhand von vier Testdatensätzen wird eine Leistungsbewertung der entwickelten Verfahren durchgeführt und gezeigt, dass selbst für Roboter mit relativ schlechter Odometrie konsistente Ergebnisse erreicht werden.

1.2 Gliederung

Nach dieser Einführung in Kapitel 1 werden in Kapitel 2 zunächst wichtige Konzepte und Entwicklungen der SLAM-Forschung vorgestellt und sowohl Notwendigkeit als auch Eigenschaften der graphbasierten SLAM-Verfahren etabliert. Kapitel 3 beschreibt die für diese Arbeit gewählte Datenstruktur zur Darstellung von Tiefendaten, auf der die nachfolgend vorgestellten Verfahren basieren. Kapitel 4 behandelt das algorithmische Schätzen der relativen Transformation zweier überlappender Sensormessungen durch zwei verschiedene Optimierungsverfahren für scharfe und unscharfe Startschätzungen. Diese Verfahren werden in Kapitel 5 zum Erzeugen eines topologisch korrekten Posengraphen verknüpft, der darauf an das freie SLAM-Backend *g2o* angebunden wird. In Kapitel 6 wird die Leistungsfähigkeit der Registrierungsalgorithmen und des entwickelten SLAM-Verfahrens anhand von Testdatensätzen und Messreihen evaluiert. In Kapitel 7 werden letztendlich die Ergebnisse dieser Arbeit beschrieben, und etwaige Punkte für weiterführende Arbeiten identifiziert.

KAPITEL 2

Grundlagen und Theorie

Um die theoretische Grundlage für diese Arbeit zu legen, soll in diesem Kapitel zunächst das Problem der gleichzeitigen Lokalisation und Kartierung besprochen werden. Dabei werden wichtige Konzepte und Methoden, die in dieser Arbeit eine zentrale Rolle spielen, hervorgehoben. Nach einem Überblick wichtiger Spielarten des SLAM-Problems, vor allem graphbasierter Verfahren, wird eine Übersicht über den aktuellen Stand der Forschung in diesem Gebiet gegeben. Soweit nicht anders gekennzeichnet, wird in dieser Arbeit die Formulierung sowie das Verständnis des SLAM-Problems von Thrun, Burgard und Fox in ihrem Buch *Probabilistic Robotics* aufgegriffen [Thrun et al., 2005]. Etwaige Abweichungen der Notation dienen dem Erhalt der Konsistenz in allen Teilen dieser Arbeit und werden als solche gekennzeichnet.

2.1 Simultaneous Localization and Mapping

Das SLAM-Problem vereint den Aspekt der Kartierung einer unbekannten Umgebung mit der Lokalisation in einer bekannten Umgebung bei eingeschränkter Sensorgenauigkeit und Berechnungskapazität. Während die Lokalisation bei bekannter Umgebungskarte sowie die Kartierung bei bekannter Roboterpose in der Theorie als gelöst gelten, ist das Problem der gleichzeitigen Kartierung und Lokalisation noch immer Gegenstand aktiver Forschung.

Bei der Erkundung der unbekannten Umgebung verfolgt der mobile Roboter einen Pfad aus Posen $\mathbf{x}$, zwischen denen die relative Bewegung durch Odometriedaten $\mathbf{u}$ gemessen wird. An jedem Punkt $\mathbf{x}$ des Pfads wird die Beschaffenheit der Umgebung durch geeignete Sensoren abgetastet, so dass für jede Pose $\mathbf{x}$ eine sog. Beobachtung

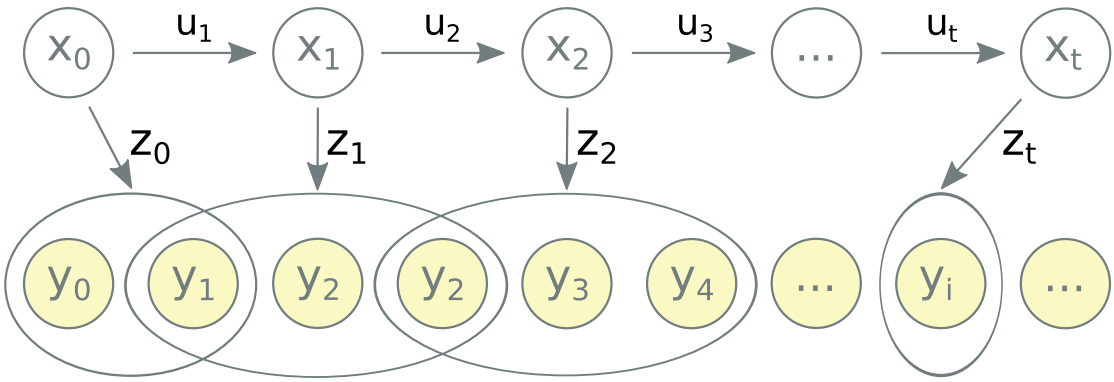


Abbildung 2.1: Veranschaulichung des stochastischen SLAM-Prozesses als gerichtetes Netz. Obwohl die Zustandsübergangsfunktion $f(\mathbf{x}_{t-1}) = \mathbf{x}_t$ nicht bekannt ist, können durch die beobachteten Steuerdaten $\mathbf{u}_t$ Rückschlüsse auf den potentiellen aktuellen Zustand $\mathbf{x}_t$ gezogen werden. Sensormessungen z_t beobachten den ebenfalls nicht bekannten Zustand der Umwelt, repräsentiert durch Umgebungsmerkmale $\mathbf{y}$.

$\mathbf{z}$ existiert (siehe Abbildung 2.1). Durch die Ungenauigkeit der Sensoren ist es nicht möglich, alle Beobachtungen $\mathbf{z}$ zu einer korrekten Umgebungskarte m zu integrieren, oder die Steuerdaten $\mathbf{u}$ mit einer Startpose $\mathbf{x}_0$ zu einem korrekten Pfad zusammenzusetzen. Aus diesem Grund wird das SLAM-Problem häufig als Henne-Ei-Problem formuliert: Um die Aufnahmepose einer Beobachtung zu bestimmen, wird eine korrekte Karte der Umgebung benötigt, deren Erstellung jedoch die korrekte Aufnahmepose aller Beobachtungen benötigt. Als Lösung dieser Zwickmühle hat sich die gleichzeitige probabilistische Schätzung der Roboterpose $\mathbf{x}$ und der Umgebungskarte m auf Basis der Beobachtungen $\mathbf{z}$ und Odometriedaten $\mathbf{u}$ etabliert [Thrun et al., 2005].

2.1.1 SLAM als stochastischer Prozess

Zur Herleitung einer mathematischen Berechnungsvorschrift wird das SLAM-Problem häufig als *Hidden Markov Model* beschrieben. Man betrachtet die Bewegung des Roboters und das Aufnehmen von Sensordaten als beobachtbare Komponenten eines stochastischen Prozesses, dessen Zustandsvariablen nicht gemessen werden können und dessen Zustandsübergange nur vom aktuellen Zustand abhängen. Abbildung 2.1 illustriert diesen Prozess als vereinfachtes dynamisches Bayes'sches Netz, das häufig

zur Beschreibung von Hidden Markov Modellen verwendet wird. Die Zufallsvariablen $\mathbf{X} = \bigcup \mathbf{x}_i$ beschreiben die unbekannten Posen des Roboters, während die statischen Zufallsvariablen $\mathbf{Y} = \bigcup \mathbf{y}_i$ die unbekannten Positionen diskreter oder kontinuierlicher Umgebungsmerkmale beschreiben. Diese Zufallsvariablen sind nicht direkt messbar, erzeugen jedoch messbare Beobachtungen. Die Odometriedaten $\mathbf{u}$ repräsentieren dabei die Beobachtungen der Zufallsvariable $\mathbf{X}$, wohingegen die Sensormessungen $\mathbf{z}$ die Beobachtungen der Zufallsvariable $\mathbf{Y}$ repräsentieren, also Messungen von Umgebungsmerkmalen durch externe Sensoren wie Kameras oder Tiefensensoren.
In dieser Darstellung entspricht das SLAM-Problem der Schätzung des Zustands $(\mathbf{x}_t, m = \bigcup \mathbf{y})$ durch die generierten Beobachtungen $(\mathbf{u}_t, \mathbf{z}_t)$, ausgehend vom aktuell geschätzten Zustand $\mathbf{x}_{t-1}$. Diese rekursive Iterationsvorschrift stellt die Realisierung einer der beiden mathematischen Formulierungen des SLAM-Problems dar, die im Folgenden vorgestellt werden sollen.

2.1.2 Probabilistische Formulierung

Die als *online SLAM* bekannte Formulierung des SLAM-Problems bedient sich der Markowannahme und des Bayestheorems, um eine Wahrscheinlichkeitsverteilung über die Roboterpose $\mathbf{x}_t$ und die Umgebungskarte m, bedingt durch die Menge der Beobachtungen $\mathbf{u}_{1:t}$ und $\mathbf{z}_{1:t}$ zu schätzen:

$$p\left(\mathbf{x}_t, m | \mathbf{z}_{1:t}, \mathbf{u}_{1:t}, \mathbf{x}_0\right) \tag{2.1}$$

Basierend auf der aktuellen Schätzung der Roboterpose $\mathbf{x}_{t-1}$ und der Umgebungskarte m wird bei derartigen SLAM-Algorithmen der neue Roboterzustand durch Integration der Steuerdaten $\mathbf{u}_t$ und der Beobachtungen $\mathbf{z}_t$ berechnet. Im Gegensatz dazu betrachtet die als *full SLAM* (oft auch *offline SLAM*) bezeichnete Berechnungsvorschrift die gesamte Abfolge der Roboterposen, also den Pfad bzw. die Trajektorie des Roboters während der Erkundung der Umgebung:

$$p\left(\mathbf{x}_{1:t}, m | \mathbf{z}_{1:t}, \mathbf{u}_{1:t}, \mathbf{x}_0\right) \tag{2.2}$$

Ein Großteil der momentan veröffentlichten SLAM-Algorithmen lässt sich einer dieser beiden Formulierungen zuordnen. Dieses probabilistischen Grundgerüst der Schätzung

einer Wahrscheinlichkeitsverteilung über Roboterpose und Umgebungskarte hat den entscheidenden Vorteil, dass sich die durch Sensorfehler in die Schätzung eingeführte Unsicherheit mathematisch solide beschreiben und integrieren lässt. Ohne diese systematischen und nicht-systematischen Sensorfehler wäre das SLAM-Problem trivial: Der exakte Zustand $(\mathbf{x}_t, m)$ könnte zu jeder Zeit aus den fehlerfreien Beobachtungen $(\mathbf{x}_0, \mathbf{u}_1, \ldots, \mathbf{u}_t, \mathbf{z}_1, \ldots, \mathbf{z}_t)$ berechnet werden. Die Verwaltung und Reduktion der durch Sensoren eingeführten Unsicherheiten sind daher elementare Ziele für SLAM-Algorithmen. Eine grundlegende Voraussetzung hierfür ist zunächst eine mathematische Darstellungsform dieser Unsicherheiten.

2.1.2.1 Modellierung von Unsicherheiten

Durch frühe Beiträge von Smith und Cheeseman sowie Leonard und Durrant-Whyte, die einfache Parametrisierbarkeit und den starken Einfluss von Gauß'schen Filtern hat sich die Normalverteilung als de facto Standard zur mathematischen Beschreibung von Unsicherheiten in der mobilen Robotik etabliert. Smith und Cheeseman entwickelten in ihrer Arbeit ein probabilistisches Framework zur Schätzung relativer Transformationen unter Bedingung normalverteilter Unsicherheiten, auf dem bis heute die Darstellung räumlicher Unsicherheiten des SLAM-Problems basiert [Smith und Cheeseman, 1986]. Leonard und Durrant-Whyte veröffentlichten in ihrem grundlegenden Paper eine der ersten Implementierungen des SLAM-Problems auf Basis von Gauß'schen Filtern, indem sie den Zustandsraum über den Roboterposen und Umgebungslandmarken durch ein Extended Kalman-Filter schätzten [Leonard und Durrant-Whyte, 1991]. Dabei verwendeten sie zwei wichtige Modelle zur Beschreibung der Systemdynamik des SLAM-Problems: das Bewegungsmodell und das Beobachtungsmodell. Diese Modelle beschreiben das Verhalten der modellierten Sensoren für die vorgegebene Situation, und berücksichtigen dabei durch sog. Sensormodelle auch die durch interne und externe Einflüsse entstehenden Unsicherheiten.

2.1.2.2 Sensormodelle

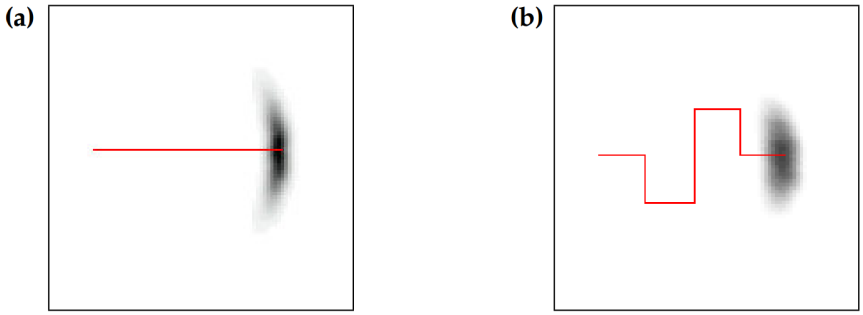
Ein wichtiges Konzept zum Umgang mit den systematischen und nicht-systematischen Fehlerquellen von Sensoren sind Sensormodelle. Um die Ungenauigkeit eines Sensors abzuschätzen, betrachtet man den Fehler jeder Messung als Ziehung aus einer unbekannten Wahrscheinlichkeitsverteilung, die für das dynamische generative Modell des Sensorfehlers steht. Würde man bei Annahme statischer Umweltbedingungen dieselbe Größe oft genug messen, so könnte aus der Gesamtheit aller gemessenen Werte die Verteilungsdichte des Fehlers geschätzt werden. Durch den starken Einfluss dynamischer Umweltbedingungen ist dies jedoch häufig nicht praktikabel; stattdessen wird diese Fehlerverteilung meist durch eine geeignet parametrisierte Normalverteilung approximiert. Zusammen mit einer Modellierung der Funktionsweise des Sensors lassen sich so Aussagen über die für eine Situation zu erwartenden Messwerte unter Berücksichtigung der zu erwartenden Unsicherheiten treffen. Im Kontext des SLAM-Problems wird sowohl ein Modell für das fehlerbehaftete Verhalten der Odometrie, als auch ein Modell für das Messverhalten der verwendeten externen Sensoren benötigt.

2.1.2.3 Bewegungsmodell

Das Bewegungsmodell beschreibt die Bewegungsdynamik eines mobilen Roboters anhand der Antriebskinematik und eines geeigneten Modells des Sensorfehlers. Im hier verwendeten probabilistischen Framework wird dieses Modell meist als die durch die vorherige Pose $\mathbf{x}_{t-1}$ und die Steuerdaten $\mathbf{u}_t$ bedingte Wahrscheinlichkeitsdichte über der Roboterpose $\mathbf{x}_t$ beschrieben:

$$p\left(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_t\right) \tag{2.3}$$

Neben einer kinematischen Funktion $f\left(\mathbf{x}, \mathbf{u}\right)$ zur Abbildung der Odometriedaten auf die Roboterpose ist ein Sensormodell ν nötig, das die Einflüsse systematischer Fehlerquellen wie ungerade Achsen oder unterschiedliche Raddurchmesser, sowie nicht-systematischer Fehlerquellen wie durchdrehende oder abhebende Räder beschreibt. Das Bewegungsmo-



Probabilistic Robotics, Thrun et al., p. 135

Abbildung 2.2: Veranschaulichung des Bewegungsmodells nach Thrun et al. [Thrun et al., 2005]. Durch Monte Carlo-Simulation wurde die a posteriori Wahrscheinlichkeit über die Roboterpose für zwei unterschiedliche Pfade approximiert, um den Einfluss der unterschiedlichen Steuerkommandos auf die Posenunsicherheit zu veranschaulichen.

dell verhält sich dann wie folgt [Leonard und Durrant-Whyte, 1991]:

$$p\left(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_t\right) \Leftrightarrow \mathbf{x}_t = f\left(\mathbf{x}_{t-1}, \mathbf{u}_t\right) + \nu \tag{2.4}$$

$$\nu \sim \mathcal{N}\left(\mu_M, \Sigma_M|\, \mathbf{u}_t\right) \tag{2.5}$$

Die Funktion $f\left(\mathbf{x}_{t-1}, \mathbf{u}_t\right)$ beschreibt die Kinematik des mobilen Roboters, während ν den Sensorfehler als additiv normalverteilt beschreibt. Für die Parametrisierung dieser Normalverteilung gilt meist $\mu_M = 0$, so dass dieses Modell allein durch eine Kovarianzmatrix Σ_M beschrieben wird. Diese Parametrisierung ist abhängig von der gemessenen Bewegung $\mathbf{u}_t$, da Rotationen und Translationen meist unterschiedliche Auswirkungen auf den Odometriefehler haben. Abbildung 2.2 zeigt diesen Unterschied anhand zweier Pfade mit gleichen Ausgangs- und Endposen, die jedoch durch verschiedene Steuerkommandos erreicht wurden.

Durch ein korrektes Bewegungsmodell kann folglich die Wahrscheinlichkeit jeder potentiellen Folgepose $\mathbf{x}_{t-1} \rightarrow \mathbf{x}_t$ unter Bedingung der aktuellen Odometriedaten $\mathbf{u}_t$ berechnet werden. Für die probabilistische SLAM-Formulierung bedeutet dies, dass neue Odometriedaten unter Berücksichtigung der Bewegungsunsicherheit in die bestehende Wahrscheinlichkeitsdichte $p\left(\mathbf{x}_{t-1}, m|\, \mathbf{u}_{1:t-1}, \mathbf{z}_{1:t-1}, \mathbf{x}_0\right)$ integriert werden können.

2.1.2.4 Beobachtungsmodell

Im Vergleich zum Bewegungsmodell beschreibt das Beobachtungsmodell nicht den Einfluss einer Messgröße auf den aktuell geschätzten Zustand, sondern die für den aktuellen Zustand zu erwartende Messgröße. Beschrieben wird dies durch die bedingte Wahrscheinlichkeit, eine Beobachtung $\mathbf{z}_t$ unter Annahme der momentanen Schätzung $\mathbf{x}_t$ und m zu messen:

$$p\left(\mathbf{z}_t|\mathbf{x}_t, m\right) \tag{2.6}$$

Wie für das Bewegungsmodell ist es auch hier erforderlich, sowohl die Funktionsweise des Sensors $h\left(\mathbf{x}_t, m\right)$ als auch dessen systematische und nicht-systematische Fehlerquellen durch ein Sensormodell ω zu modellieren. Für externe Sensoren wie Kameras oder Tiefensensoren müssen also neben dem typischen Sensorrauschen auch dynamische Umwelteinflüsse wie Reflexionen oder physikalische Eigenschaften des Übertragungsmediums in Betracht gezogen werden. Das Beobachtungsmodell verhält sich wie folgt [Leonard und Durrant-Whyte, 1991]:

$$p\left(\mathbf{z}_t|\mathbf{x}_t, m\right) \Leftrightarrow \mathbf{z}_t = h\left(\mathbf{x}_t, m\right) + \omega \tag{2.7}$$

$$\omega \sim \mathcal{N}\left(\mu_O, \Sigma_O\right) \tag{2.8}$$

Die Funktion $h\left(\mathbf{x}_t, m\right)$ berechnet die unter Annahme der Roboterpose $\mathbf{x}_t$ und Umgebungskarte m zu erwartende Beobachtung, während der Term ω für den additiv normalverteilten Fehler des verwendeten Sensors steht. Wie schon beim Bewegungsmodell wird μ_O meist auf 0 gesetzt, um dieses Rauschen durch Σ_O parametrisieren zu können. Im Unterschied zum Bewegungsmodell ist die Modellierung des Fehlers hier nicht abhängig von anderen Faktoren, sondern wird stets gleich modelliert.

Ein korrektes Beobachtungsmodell ist folglich in der Lage, eine Vorhersage über die unsichere Position von Umgebungsmerkmalen auf Basis der aktuellen Schätzung $\mathbf{x}_t$ und m zu treffen. Die bedingte Wahrscheinlichkeitsdichte $p\left(\mathbf{x}_t, m|\mathbf{u}_{1:t}, \mathbf{z}_{1:t-1}, \mathbf{x}_0\right)$ kann somit für alle $\mathbf{x}_t$ anhand der Diskrepanz $h\left(\mathbf{x}_t, m\right) \leftrightarrow \mathbf{z}_t$ angepasst werden. Auf diese Weise kann die Beobachtung $\mathbf{z}_t$ unter Berücksichtigung der Unsicherheit in die aktuelle Schätzung integriert werden, um die durch das Bewegungsmodell vorbereitete Zustandshypothese zu verifizieren und anzupassen. Ein rekursiver SLAM-Algorithmus iteriert

die Anwendung des Bewegungs- und des Beobachtungsmodells für eine Initialschätzung $(\mathbf{x}_0, m_0)$, um eine Schätzung über den aktuellen Zustand $(\mathbf{x}_t, m)$ zu erhalten.

2.1.3 Schleifenschluss

Ein elementares Problem eines SLAM-Verfahrens ist die Etablierung von Korrespondenzen zwischen verschiedenen Beobachtungen. In der Praxis ist es durch die Unsicherheit der Sensoren und den meist hohen Ähnlichkeitsgrad der Umgebung nicht möglich, eine exakte Korrespondenz zwischen Beobachtungen und abgebildeten Umgebungsmerkmalen herzustellen. Um eine konsistente Karte zu erzeugen ist es jedoch erforderlich, unterschiedliche Beobachtungen derselben Umgebungsmerkmale zu erkennen und zu vereinen. Für jede neue Beobachtung $\mathbf{z}_t$ stellt sich folglich die Frage: Wurden die durch $\mathbf{z}_t$ beobachteten Umgebungsmerkmale bereits durch frühere Beobachtungen $\mathbf{z}_i$ gemessen? Dieses Problem ist als Schleifenschluss- bzw. Korrespondenzproblem bekannt und stellt das wichtigste Werkzeug eines SLAM-Verfahrens dar, um eine konsistente Zustandsschätzung durch Reduktion der Unsicherheiten zu erzeugen.
Formell wird dieses Problem von Thrun et al. durch Ergänzung des zu schätzenden Zustands um Korrespondenzvariablen $\mathbf{c}_t$ beschrieben [Thrun et al., 2005]. Die probabilistische Formulierung dieses Kapitels wird daher wie folgt erweitert:

$$\text{Online SLAM: } p\left(\mathbf{x}_t, m, \mathbf{c}_t | \mathbf{z}_{1:t}, \mathbf{u}_{1:t}, \mathbf{x}_0\right) \tag{2.9}$$

$$\text{Full SLAM: } p\left(\mathbf{x}_{1:t}, m, \mathbf{c}_{1:t} | \mathbf{z}_{1:t}, \mathbf{u}_{1:t}, \mathbf{x}_0\right) \tag{2.10}$$

Eine potentielle Darstellung der Korrespondenzen $\mathbf{c}_t$ ist eine $t \times t$-Matrix über allen bekannten Beobachtungen $\mathbf{z}_{1:t}$. Die Matrizeneinträge c_{ij} stehen dabei für die Wahrscheinlichkeit der Korrespondenz der beiden betrachteten Beobachtungen $\mathbf{z}_i$ und $\mathbf{z}_j$. Da $\mathbf{c}_t$ in normalen Anwendungen nicht bekannt ist, müssen diese Korrespondenzen entweder geschätzt oder explizit etabliert werden. Neben der Ungenauigkeit der Sensoren und der hohen Ähnlichkeit vieler Beobachtungen müssen Methoden zur Korrespondenzetablierung auch damit umgehen, dass die kontinuierliche Umwelt durch diskrete Beobachtungen abgebildet wird. Es ist daher höchst unwahrscheinlich, dass es zwei Beobachtungen gibt, die genau denselben Abschnitt der Umgebung abbilden. Die

Korrespondenzetablierung von Beobachtungen beschränkt sich daher meist darauf, den Anteil der gemeinsam beobachteten Strukturen zu maximieren.

2.2 Filterbasiertes SLAM

Die bereits angesprochene Familie der *online SLAM*-Algorithmen basiert üblicherweise auf einem rekursiven Schätzverfahren, dem Bayes-Filter. Dieses Filter berechnet eine bedingte Wahrscheinlichkeitsdichte über den diskreten Zustandsraum eines Markow'schen Prozesses auf Basis der messbaren Steuer- und Beobachtungsdaten. In diesem Kapitel wurde das SLAM-Problem bereits als derartiger stochastischer Prozess dargestellt (siehe Abbildung 2.1), so dass sich die Anwendung dieser Filtermethode für dieses Problem anbietet.

2.2.1 Bayes-Filter

Das Bayes-Filter basiert auf dem namensgebenden Theorem der bedingten Wahrscheinlichkeit nach Thomas Bayes:

$$p\left(x|y\right) = p\left(y|x\right) p\left(x\right) \cdot p\left(y\right)^{-1} \tag{2.11}$$

Im Kontext dieses Filters wird in der Literatur zumeist vom sog. Belief $bel\left(\mathbf{x}_t\right)$ gesprochen, also vom Überzeugungsgrad über den internen Zustand des Markow'schen Prozesses. Dieser Überzeugungsgrad wird anhand der messbaren Steuerdaten $\mathbf{u}_t$ und der Beobachtungen $\mathbf{z}_t$ in zwei Schritten berechnet: Vorhersage und Korrektur [Thrun et al., 2005].

Bei der Vorhersage wird der momentane Überzeugungsgrad $bel\left(\mathbf{x}_{t-1}\right)$ für alle möglichen Zustände $\mathbf{x}_{t-1}$ in einen hypothetischen Überzeugungsgrad $\overline{bel}\left(\mathbf{x}_t\right)$ überführt, der die Auswirkung der Steuerdaten $\mathbf{u}_t$ vorhersagt:

$$\overline{bel}\left(\mathbf{x}_t\right) = \int p\left(\mathbf{x}_t|\mathbf{u}_t, \mathbf{x}_{t-1}\right) bel\left(\mathbf{x}_{t-1}\right) d\mathbf{x}_{t-1} \tag{2.12}$$

Hierbei wird häufig auch von der Zustandsübergangsfunktion gesprochen. Der durch die aktuelle Beobachtung $\mathbf{z}_t$ bedingte Überzeugungsgrad $bel\left(\mathbf{x}_t|\mathbf{z}_t\right)$ entsteht aus dieser

Hypothese durch das Bayestheorem:

$$bel\left(\mathbf{x}_t\right) = bel\left(\mathbf{x}_t|\mathbf{z}_t\right) = p\left(\mathbf{z}_t|\mathbf{x}_t\right)\overline{bel}\left(\mathbf{x}_t\right)p\left(\mathbf{z}_t\right)^{-1} \tag{2.13}$$

Ausgehend von einer initialen Schätzung der gesuchten Wahrscheinlichkeitsverteilung, repräsentiert durch den Überzeugungsgrad $bel\left(\mathbf{x}_0\right)$, kann die bedingte Wahrscheinlichkeitsdichte über $\mathbf{x}$ somit rekursiv durch das Integrieren der Steuer- und Beobachtungsdaten $(\mathbf{u}_t, \mathbf{z}_t)$ für alle Zeitschritte geschätzt werden. Für anspruchsvolle Probleme explodiert die Berechnungskomplexität des Bayes-Filters jedoch schnell, zudem existiert für Probleme mit kontinuierlichen Zustandsräumen keine analytisch geschlossen berechenbare Form. Aus diesem Grund ist das Bayes-Filter hauptsächlich von theoretischer Bedeutung; für Probleme wie SLAM werden stattdessen fortgeschrittene Filterverfahren eingesetzt.

2.2.2 Extended Kalman-Filter

Obwohl das Bayes-Filter für komplexe Probleme wie das SLAM-Problem meist ungeeignet ist, bildet dieses Filter die Grundlage für eine Familie kontinuierlicher Zustandsschätzer, die weitläufig eingesetzt werden: sog. *Gauß'sche Filter* [Thrun et al., 2005]. Im Gegensatz zum diskreten Bayes-Filter schätzen Gauß'sche Filter die Überzeugungsverteilung eines kontinuierlichen Zustandsraums als multivariate Normalverteilung.

2.2.2.1 Kalman-Filter

Das sog. Kalman-Filter ist die bekannteste Implementierung eines Gauß'schen Filters zur Schätzung und Vorhersage des Zustands eines stochastischen Prozesses [Kalman, 1960]. Die normalverteilte Überzeugungsverteilung über den Systemzustand wird durch den Mittelwert μ_t und die Kovarianzmatrix Σ_t beschrieben. Um zu garantieren, dass der a posteriori Zustand (μ_t, Σ_t) stets eine Normalverteilung beschreibt, muss der zugrundeliegende stochastische Prozess strikte Linearitätsbedingungen erfüllen:

$$\mathbf{x}_t = A_t\mathbf{x}_{t-1} + B_t\mathbf{u}_t + \varepsilon_t \qquad \left(\varepsilon_t \sim \mathcal{N}\left(0, R_t\right)\right) \tag{2.14}$$

$$\mathbf{z}_t = C_t\mathbf{x}_t + \delta_t \qquad \left(\delta_t \sim \mathcal{N}\left(0, Q_t\right)\right) \tag{2.15}$$

$$bel\left(\mathbf{x}_0\right) \sim N\left(\mu_0, \Sigma_0\right) \tag{2.16}$$

Sowohl die Beobachtungsfunktion in Gleichung (2.15) als auch die Zustandsübergangsfunktion in Gleichung (2.14) müssen folglich linear sein, mit zeitlich unkorreliertem normalverteiltem Rauschen zur Modellierung der Unsicherheit. Des Weiteren muss die initiale Überzeugungsverteilung normalverteilt sein (Gleichung (2.16)). Derartige Bedingungen existieren jedoch meist nur bei relativ trivialen Problemen.

2.2.2.2 Nichtlineare Systeme

Die Annahme einer linearen Systemdynamik ist sehr restriktiv, da viele komplexere Probleme, wie das SLAM-Problem, nichtlineares Verhalten besitzen. Um das Bayes'sche Filterprinzip dennoch auf nichtlineare Probleme anwenden zu können, linearisiert das sog. Extended Kalman-Filter (EKF) die nichtlinearen Beobachtungs- und Zustandsübergangsfunktionen im ersten Moment der geschätzten Wahrscheinlichkeitsdichte [Thrun et al., 2005]. Die Anforderungen an die Systemdynamik sind im Vergleich zum Kalman-Filter erheblich toleranter:

$$\mathbf{x}_t = f\left(\mathbf{x}_{t-1}, \mathbf{u}_t\right) + \varepsilon_t \qquad \left(\varepsilon_t \sim \mathcal{N}\left(0, R_t\right)\right) \tag{2.17a}$$

$$\mathbf{z}_t = h\left(\mathbf{x}_t\right) + \delta_t \qquad \left(\delta_t \sim \mathcal{N}\left(0, Q_t\right)\right) \tag{2.17b}$$

$$bel\left(\mathbf{x}_0\right) \sim N\left(\mu_0, \Sigma_0\right) \tag{2.17c}$$

Die Zustandsübergangsfunktion $f(\cdot)$ und die Beobachtungsfunktion $h(\cdot)$ können im EKF durch nichtlineare Funktionen beschrieben werden, die jedoch differenzierbar sein müssen. Weiterhin muss die initiale Schätzung der Überzeugungsverteilung $bel\left(\mathbf{x}_0\right)$ normalverteilt sein.

2.2.2.3 Linearisierung

Um trotz nichtlinearer Funktionen $f(\cdot)$ und $h(\cdot)$ die Normalverteiltheit der geschätzten Überzeugungsverteilung (μ_t, Σ_t) zu garantieren, werden diese Funktionen zur Berechnung durch linearisierte Approximationen ersetzt (siehe Abbildung 2.3). Der EKF nutzt hierzu eine Taylorreihenentwicklung erster Ordnung, bei der anhand der ersten Ableitung und einer Stützstelle eine Linearisierung der Funktion berechnet wird [Thrun

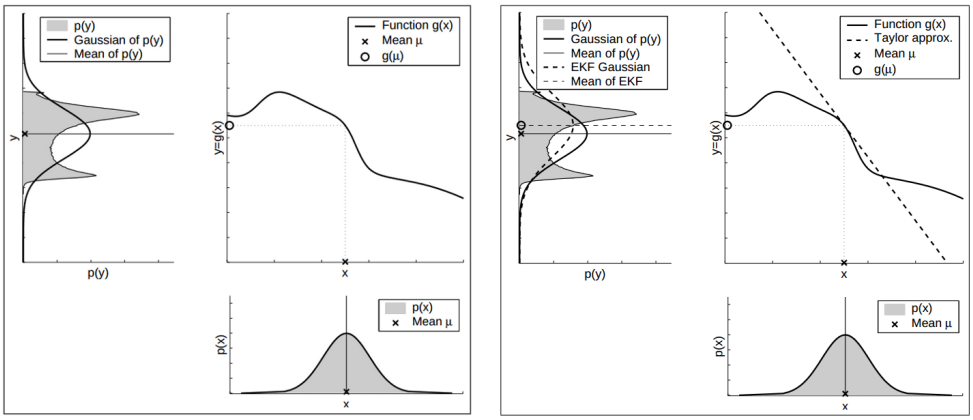


Abbildung 2.3: Veranschaulichung der Linearisierung der Zustandsübergangs- und Beobachtungsfunktion des Extended Kalman-Filters [Thrun et al., 2005]. Die linke Abbildung zeigt die zugrundeliegende Problematik: Die normalverteilte Zufallsvariable x wird durch eine nichtlineare Funktion $g(x) = y$ transformiert, so dass die resultierende Wahrscheinlichkeitsdichte $p(y)$ nicht mehr normalverteilt ist. Die rechte Abbildung zeigt die Linearisierung von $g(x)$ durch Taylorreihenentwicklung. Die durch diese Linearisierung transformierte Wahrscheinlichkeitsdichte ist weiterhin normalverteilt und stellt eine ausreichende Approximation der tatsächlichen Verteilung dar.

et al., 2005]. Es werden folglich die partiellen Ableitungen der Zustandsübergangsfunktion und der Beobachtungsfunktion benötigt:

$$f'(\mathbf{x}_{t-1}, \mathbf{u}_t) = \frac{\partial f(\mathbf{x}_{t-1}, \mathbf{u}_t)}{\partial \mathbf{x}_{t-1}} \tag{2.18}$$

$$h'(\mathbf{x}_{t-1}) = \frac{\partial h(\mathbf{x}_{t-1})}{\partial \mathbf{x}_{t-1}} \tag{2.19}$$

Da die Schätzung üblicherweise in einem mehrdimensionalen Zustandsraum stattfindet, werden die partiellen Ableitungen im Folgenden durch die Jacobimatrizen F_t und H_t dargestellt.

2.2.2.4 Vorhersage

Mithilfe der Jacobimatrizen kann nun der Vorhersageschritt des Extended Kalman-Filters für die Parameter μ_t und Σ_t der zu schätzenden Normalverteilung beschrieben

werden:

$$\bar{\mu}_t = f\left(\mu_{t-1}, \mathbf{u}_t\right) \tag{2.20}$$

$$\bar{\Sigma}_t = F_t \Sigma_{t-1} F_t^T + R_t \tag{2.21}$$

Der vorhergesagte Mittelwert $\bar{\mu}_t$ ergibt sich direkt aus der Anwendung der Zustandsübergangsfunktion auf die momentane Schätzung μ_{t-1}. Ebenso wird die vorhergesagte Kovarianz $\bar{\Sigma}_t$ durch die Anwendung der approximierten linearisierten Systemdynamik F_t berechnet, und um die Unsicherheit R_t der Zustandsübergangsfunktion verrauscht. Die Schätzung $\left(\bar{\mu}_t, \bar{\Sigma}_t\right)$ entspricht folglich der Anwendung der Steuerdaten $\mathbf{u}_t$ auf den Systemzustand des vorherigen Zeitschritts. Im Korrekturschritt soll nun die a posteriori Schätzung (μ_t, Σ_t) durch Integration der Beobachtung $\mathbf{z}_t$ berechnet werden.

2.2.2.5 Korrektur

Um den Korrekturschritt durchzuführen, wird ein Maß für die Qualität der aktuellen Schätzung $\left(\bar{\mu}_t, \bar{\Sigma}_t\right)$ im Hinblick auf die Beobachtung $\mathbf{z}_t$ benötigt. Eine wichtige Metrik zur Bewertung der Schätzung ist die als Innovation oder Residuum bekannte Diskrepanz y_t zwischen Vorhersage und Beobachtung:

$$y_t = \bar{\mu}_t - h\left(\bar{\mu}_t\right) \tag{2.22}$$

Die zu erwartende Unsicherheit kann nun als Varianz des Residuums geschätzt werden:

$$\hat{\Sigma}_t = E[y_t y_t^T] \tag{2.23}$$

Um die durch das Integrieren der Beobachtung in das System eingeführte Unsicherheit gering zu halten, muss die Gleichung (2.23) minimiert werden [Welch und Bishop, 1995]. Der dabei entstehende Ausdruck beschreibt den als annähernd optimaler (engl. *near-optimal*) Kalman-Gain bezeichneten Faktor K_t:

$$K_t = \bar{\Sigma}_t H_t^T \left(H_t \bar{\Sigma}_t H_t^T + Q_t\right)^{-1} \tag{2.24}$$

Dieser Term beschreibt den möglichen Informationsgewinn unter Berücksichtigung der momentanen Schätzungenauigkeit und der Ungenauigkeit der Beobachtung, und kann

somit zur Skalierung der Korrektur des Zustands durch die Beobachtung verwendet werden. Wäre die Integration der Beobachtung nur unter großer Unsicherheit möglich, so würde die Korrektur entsprechend vorsichtiger durchgeführt werden. Hieraus ergibt sich der Korrekturschritt zur Berechnung der a posteriori Schätzung (μ_t, Σ_t):

$$\mu_t = \bar{\mu}_t + K_t y_t \tag{2.25}$$

$$\Sigma_t = (I - K_t H_t)\,\bar{\Sigma}_t \tag{2.26}$$

Der Mittelwert μ_t wird also um die Diskrepanz zwischen Beobachtung und Vorhersage korrigiert, skaliert um den potentiellen Informationsgewinn bei Integration der Beobachtung $\mathbf{z}_t$. Die Kovarianzmatrix Σ_t wird ebenfalls entsprechend dieses Informationsgewinns skaliert, um die Auswirkungen der Korrektur als proportionale Verbesserung oder Verschlechterung der Schätzgenauigkeit zu beschreiben.

2.2.2.6 Evaluation

Eine Vielzahl früher, aber auch aktueller SLAM-Algorithmen basiert auf der rekursiven probabilistischen Schätzung des Zustandsraums $(\mathbf{x}, m, \mathbf{c})$ durch das Extended Kalman-Filter. Durch die hohe Berechnungskomplexität (quadratisch in der Dimensionszahl n [Thrun et al., 2005]) skaliert der EKF jedoch schlecht mit der Anzahl der zu schätzenden Landmarken und Korrespondenzen. Des Weiteren ist es bei komplexen Problemen mit hoher Ambiguität häufig nötig, mehrere potentielle Hypothesen zu verfolgen. Dies ist für den EKF durch die Modellierung der Überzeugungsverteilung als inhärent unimodale Normalverteilung nicht möglich. Die größte potentielle Schwachstelle des Extended Kalman-Filters ist jedoch die Linearisierung der Beobachtungs- und Zustandsübergangsfunktion. Wie von Thrun et al. bemerkt, basiert die verwendete Taylorreihenentwicklung nur auf dem Mittelwert der Schätzung, so dass die Genauigkeit der berechneten Schätzung stark von der Kovarianz der zu transformierenden Größe abhängig ist. Des Weiteren hängt die Genauigkeit der Schätzung stark von der lokalen Nichtlinearität der zu transformierenden Funktion an der gewählten Stützstelle ab. Durch diese hohe Abhängigkeit des Leistungsverhaltens vom aktuell geschätzten Zustand kann es durchaus zur Divergenz der Schätzung kommen.

Abbildung 2.4: Veranschaulichung der Approximation der Wahrscheinlichkeitsdichte durch ein Partikelfilter am Beispiel der Lokalisation. Links wird die initiale, zufällige Verteilung der Partikelpopulation gezeigt, die durch die Integration von Beobachtungen zu einer genaueren Approximation der Dichte des Zustandsraums wird (rechte Abbildung). Die mittlere Abbildung zeigt zudem die Fähigkeit des Partikelfilters, mehrere disjunkte Hypothesen gleichzeitig modellieren zu können [Dellaert et al., 1999].

2.2.3 Partikelfilter

Eine im SLAM-Bereich sehr verbreitete Variante der rekursiven Zustandsschätzung ist das sog. Partikelfilter, auch sequentielle Monte Carlo-Methode genannt. Im Vergleich zu Gauß'schen Filtern schätzt dieses nichtparametrische Filter die a posteriori Verteilung der Überzeugung durch Monte Carlo-Simulation. Dabei wird eine endliche Anzahl von systematisch verteilten Zustandshypothesen (die sog. Partikeln) simuliert, die als Ziehungen aus der unbekannten, zu schätzenden Zustandsverteilungsfunktion betrachtet werden. Unterzieht man eine initial geeignet verteilte Generation von Partikeln nun fortlaufend der Systemdynamik, so lässt sich die zu schätzende Verteilung durch die Partikelpopulation approximieren. Je dichter ein lokaler Unterraum mit Partikeln besiedelt ist, desto wahrscheinlicher fällt der tatsächliche Zustand auch in diesen Unterraum [Thrun et al., 2005].

2.2.3.1 Vorhersage

Die Überzeugungsverteilung wird in der beschriebenen Monte Carlo-Methode durch eine Population $\mathcal{X}_t$ aus Partikeln $\mathbf{x}_t^i$ parametrisiert. Würde für jedes Partikel $\mathbf{x}_t^i$ dieselbe Systemdynamik durch die Zustandsübergangsfunktion $\mathbf{x}_t^i = f\left(\mathbf{x}_{t-1}^i, \mathbf{u}_t\right)$ angewendet,

so würden sich alle Partikeln bis auf ihre Anfangsposition $\mathbf{x}_0^i$ gleich verhalten. Um den Zustandsraum im Rahmen der gegebenen Unsicherheiten zu erkunden, wird die neue Position daher für jedes Partikel individuell aus einer unsicheren stochastischen Modellierung der Zustandsübergangsfunktion gezogen:

$$\bar{\mathbf{x}}_t{}^i \sim p\left(\mathbf{x}_t | \mathbf{x}_{t-1}^i, \mathbf{u}_t\right) \tag{2.27}$$

Ein einfaches Modell würde beispielsweise für jedes Partikel aus einer Normalverteilung ziehen, deren Mittelwert dem durch die Zustandsübergangsfunktion bestimmten Wert $f\left(\mathbf{x}_{t-1}^i, \mathbf{u}_t\right)$ entspricht, mit der zugehörigen Unsicherheit R_t als Kovarianz:

$$\bar{\mathbf{x}}_t{}^i \sim \mathcal{N}\left(f\left(\mathbf{x}_{t-1}^i, \mathbf{u}_t\right), R_t\right) \tag{2.28}$$

Die Anwendung einer solchen randomisierten Zustandsübergangsfunktion entspricht dem Vorhersageschritt des Partikelfilters. Die durch $\bar{\mathcal{X}}_t$ beschriebene aktuelle Schätzung der Wahrscheinlichkeitsdichte des Zustandsraums reflektiert nun die Integration der unsicheren Steuerdaten $\mathbf{u}_t$.

2.2.3.2 Korrektur

In diese Population $\bar{\mathcal{X}}_t$ soll nun im Korrekturschritt die Beobachtung $\mathbf{z}_t$ integriert werden. Im Vergleich zum Kalman-Filter sind die Zustandshypothesen hier nicht mit Unsicherheiten behaftet, die anhand der Diskrepanz zwischen Vorhersage und Beobachtung verändert werden könnten. Stattdessen ordnet das Partikelfilter jedem Partikel $\bar{\mathbf{x}}_t{}^i$ basierend auf dieser Diskrepanz einen Gewichtungsfaktor w_t^i zu. Dieser Faktor beschreibt die Wahrscheinlichkeit, dass die Beobachtung $\mathbf{z}_t$ unter Bedingung der Zustandshypothese $\bar{\mathbf{x}}_t{}^i$ wahrgenommen werden könnte:

$$w_t^i = p\left(\mathbf{z}_t | \bar{\mathbf{x}}_t{}^i\right) \tag{2.29}$$

Partikel mit höherem Gewichtungsfaktor bieten somit eine bessere Hypothese zur Erklärung der Beobachtungen und sind folglich näher am zu schätzenden Systemzustand. Da die a posteriori Überzeugungsverteilung jedoch nur durch die Verteilung der Partikeln $\bar{\mathbf{x}}_t{}^i$ im Zustandsraum geschätzt wird, muss dieses Wissen über die Güte der einzelnen Hypothesen auf die Partikelpopulation übertragen werden.

2.2.3.3 Resampling

Dieser Vorgang heißt Resampling und bedient sich der Eigenschaft von Wahrscheinlichkeitsverteilungen, dass ein zufällige Ziehung wahrscheinlich in einen Unterraum mit hoher Wahrscheinlichkeitsdichte fällt. Da ein Partikel mit einem hohen Gewichtungsfaktor für eine gute Zustandsschätzung steht, sollte die a posteriori Überzeugungsverteilung an diesem Punkt eine hohe Dichte aufweisen.
Die Population der Partikeln wird nun bedingt durch die anhand der Gewichtungsfaktoren geschätzte Wahrscheinlichkeitsdichte neu im Zustandsraum verteilt: Ein Partikel $\mathbf{x}_t^j \in \mathcal{X}_t$ wird mit Wahrscheinlichkeit w_t^i aus $\bar{\mathbf{x}}_t{}^i$ gezogen (d.h. als Kopie von $\bar{\mathbf{x}}_t{}^i$ angelegt). Dieser Vorgang wird für alle Partikeln so lange wiederholt, bis die Population $\mathcal{X}_t$ wieder die erforderliche Größe besitzt. Auf diese Weise korreliert die Bevölkerungsdichte der neuen Generation $\mathcal{X}_t$ mit der durch die Gewichtungsfaktoren approximierten Dichte des Zustandsraums. Unterräume guter Zustandshypothesen werden somit dichter besiedelt und verstärkt exploriert, während schlechte Hypothesen zunehmend aussortiert werden. Bei ausreichender Partikelzahl und guter Systemmodellierung konvergiert die geschätzte Überzeugungsverteilung somit gegen die tatsächliche Wahrscheinlichkeitsdichte des Zustandsraums.

2.2.3.4 Evaluation

Aufgrund der recht präzisen Schätzung der tatsächlichen Wahrscheinlichkeitsverteilung des Zustandsraums gilt das Partikelfilter als gutes Werkzeug zur Zustandsschätzung komplexer Probleme, wie beispielsweise dem Lokalisationsproblem. Die simple Berechnungsvorschrift ermöglicht eine einfache Implementierung mit einer überschaubaren Anzahl an Parametern. Im Gegensatz zu beispielsweise Kalman-Filtern ist jedoch ein enormer Berechnungsaufwand nötig, denn die Komplexität steigt exponentiell mit der Dimension des Zustandsraums [Thrun et al., 2005]. Komplexe Probleme benötigen zur präzisen Schätzung zudem oft eine hohe Anzahl an Partikeln, so dass der Berechnungsaufwand eines Partikelfilters meist mehrere Größenordnungen höher als bei anderen Filtern ist. Eine weitere Einschränkung ist das Fehlen der expliziten Wahrscheinlichkeitsdichte über den Zustandsraum, die bei Bedarf beispielsweise durch

den k-means-Algorithmus approximiert werden muss.
Ein weiteres Problem des Partikelfilters ergibt sich aus den stochastischen Operationen Sampling und Resampling. Das Problem der Stichprobenvarianz besagt, dass eine endliche Menge von Stichproben aus einer Verteilung nie exakt die gleiche Verteilung approximieren kann. Durch das häufige Ziehen aus Verteilungen erhält das Partikelfilter also eine nichtdeterministische Komponente, so dass bei gleicher Abfolge von $\mathbf{u}_{1:t}$ und $\mathbf{z}_{1:t}$ für die gleiche Startpopulation $\mathcal{X}_0$ möglicherweise gravierend unterschiedliche Überzeugungsverteilungen geschätzt werden können. Ebenso bemerken Thrun et al., dass durch das Resampling zwar die Varianz der Partikelpopulation verringert wird, die Varianz der Partikelpopulation als Schätzer durch den Verlust von Vielfalt jedoch steigt [Thrun et al., 2005].

2.3 Graphbasiertes SLAM

Eine erhebliche Einschränkung der filterbasierten SLAM-Verfahren ergibt sich aus der rekursiven Berechnung der aktuellen Überzeugungsverteilung in Gleichung (2.9). Durch die Markowannahme werden alle Informationen der Vergangenheit zu einer als korrekt angenommenen Schätzung des Systemzustands integriert. Dies bewirkt, dass vergangene Korrespondenz- oder Schätzfehler nicht oder nur sehr aufwändig korrigiert werden können, da deren Auswirkungen auf die Zustandsschätzung nicht mehr nachvollziehbar sind. In Gleichung (2.10) dagegen werden alle etablierten Korrespondenzen und ermittelten Informationen explizit aufrecht gehalten, so dass durch neue Erkenntnisse potentielle Fehler der Vergangenheit erkannt und korrigiert werden können.
Eine beliebte Interpretation des *full SLAM* ist die Formulierung als hochdimensionales nichtlineares Optimierungsproblem. Der gefahrene Pfad wird häufig als gerichteter Graph aus Posen betrachtet, dessen Kanten einschränkende Bedingungen für die relative Transformation der verbundenen Knoten beschreiben. Auf dieser Graphstruktur können Optimierungsverfahren durchgeführt werden, um eine bezüglich der Kanten optimale Knotenkonfiguration zu finden. Da eine derartige Konfiguration die bestmögliche Schätzung des tatsächlich gefahrenen Roboterpfads darstellt, kann mit dieser Information eine korrekte Karte gebaut werden. Dieses Optimierungsproblem über einem Graphen

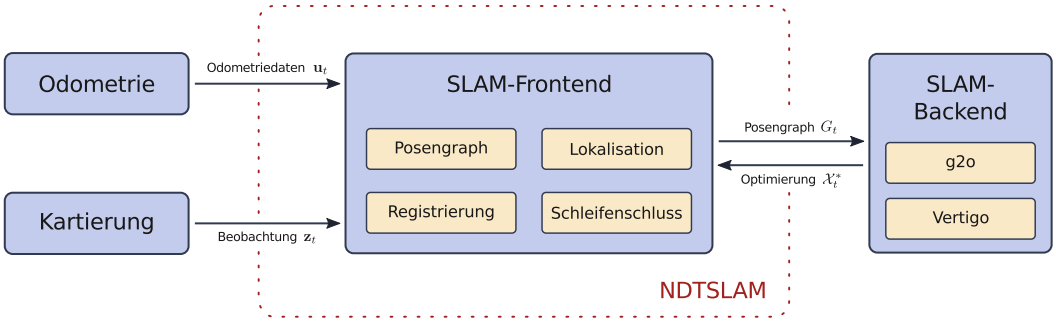


Abbildung 2.5: Abstrakte Darstellung der Komponenten eines graphbasierten SLAM-Verfahrens. Der rot markierte Teil stellt den Fokus dieser Arbeit und der entwickelten Implementierung **NDTSLAM** dar.

aus Posen wird aus diesem Grund gemeinhin als graphbasiertes SLAM bezeichnet (siehe Abbildung 2.5 für eine abstrakte Veranschaulichung) [Grisetti et al., 2010a]. Graphbasierte SLAM-Algorithmen benötigen einen sensor- und anwendungsspezifischen Mechanismus zur Erzeugung dieses Posengraphen, der meist SLAM-Frontend genannt wird. Die Bestimmung einer optimalen Knotenkonfiguration eines Posengraphen wird meist als SLAM-Backend bezeichnet. In diesem Abschnitt soll nur eine kurze Einführung gegeben werden, für mehr Details sei auf Kapitel 5 verwiesen.

2.3.1 Posengraph

Der Posengraph repräsentiert die ermittelten Informationen über Roboterposen und Beobachtungen in einer topologisch korrekten Struktur aus Knoten und Kanten. Jeder Knoten steht für eine Pose $\mathbf{x}_t$ und wird mit weiteren Knoten durch Kanten verbunden, die eine Einschränkung der jeweiligen relativen Transformation beschreiben (siehe Abbildung 2.6). Jede Kante beschreibt sowohl die vermutete relative Transformation zwischen zwei Knoten, als auch die Sicherheit bzw. Unsicherheit dieser Information. Odometriedaten $\mathbf{u}_t$ beschreiben eine Odometriekante o_t, die für den von der fehlerbehafteten Odometrie gemessenen relativen Versatz der Posen $\mathbf{x}_{t-1}$ und $\mathbf{x}_t$ steht. Zwei Beobachtungen $(\mathbf{z}_{t-1}, \mathbf{z}_t)$ hingegen ermöglichen das Einfügen von Registrierungskanten r_t, welche die relative Transformation $\mathbf{x}_{t-1} \leftrightarrow \mathbf{x}_t$ aus den überlappenden Beobachtungen schätzen. Existiert für die aktuelle Beobachtung $\mathbf{z}_t$ eine weitere Korrespondenz

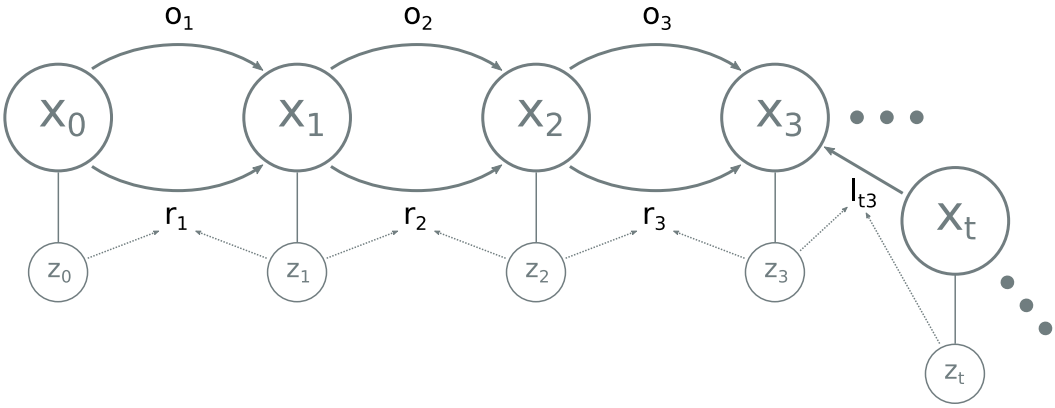


Abbildung 2.6: Veranschaulichung des Posengraphen, der durch das SLAM-Frontend erstellt wird. Zwei aufeinanderfolgende Roboterposen $\mathbf{x}$ werden jeweils durch eine Odometriekante o sowie eine Registrierungskante r verbunden. Im Falle eines Schleifenschlusses werden als korrespondierend festgestellte Roboterposen durch eine Schleifenschlusskante l verbunden.

$\mathbf{z}_j \neq \mathbf{z}_{t-1}$, so liegt ein Schleifenschluss vor. Der relative Versatz dieser ebenfalls überlappenden Beobachtungen wird bestimmt und als Schleifenschlusskante l_{tj} in den Posengraphen eingefügt. Das Einfügen von neuen Knoten sowie das Erzeugen von Odometrie-, Registrierungs- und Schleifenschlusskanten ist die Aufgabe des SLAM-Frontends.

2.3.2 Frontend

Die Aufgabe des SLAM-Frontends ist es, aus den Daten $(\mathbf{x}_0, \mathbf{u}_1, \ldots, \mathbf{u}_t, \mathbf{z}_1, \ldots, \mathbf{z}_t)$ einen topologisch korrekten Posengraph zu erzeugen. Beginnend mit $\mathbf{x}_0$ wird für jeden diskreten Zeitschritt t ein Knoten mit zugehöriger Pose $\mathbf{x}_t$ in den Posengraphen eingefügt. Um eine grobe Schätzung für diese Pose zu erhalten, wird meist die von der fehlerbehafteten Odometrie geschätzte Position zum Zeitpunkt t verwendet. Um diese Information explizit zu repräsentieren, werden zwei aufeinanderfolgende Knoten durch Odometriekanten mit dem jeweiligen relativen Versatz verbunden. Durch die sog. Registrierung der überlappenden Beobachtungen $\mathbf{z}_{t-1}$ und $\mathbf{z}_t$ dieser Knoten ist es möglich, eine redundante und meist ziemlich genaue Schätzung der relativen Transformation der Knoten $\mathbf{x}_{t-1}$ und $\mathbf{x}_t$ zu bestimmen (siehe Kapitel 4). Diese Information wird zusätzlich

zur Odometriekante als Registrierungskante zwischen diese Knoten eingefügt.
Der bislang vom SLAM-Frontend erzeugte Posengraph ist höchstwahrscheinlich nicht topologisch korrekt, da das Korrespondenzproblem für die Beobachtungen $\mathbf{z}_t$ nicht gelöst würde. Ein elementarer Bestandteil des Frontends ist daher eine Methode zum Entdecken und Lösen von Schleifenschlüssen, die beim Wiedereintritt in bereits bekanntes Gebiet auftreten. Für die aktuelle Beobachtung $\mathbf{z}_t$ müssen dabei alle potentiell korrespondierenden Beobachtungen $\mathbf{z}_k$ ermittelt und registriert werden. Die so bestimmten relativen Transformationen werden als Schleifenschlusskanten zwischen die Knoten $\mathbf{x}_t$ und $\mathbf{z}_k$ geschrieben. Der erzeugte Posengraph ist nun vermutlich topologisch korrekt und beinhaltet alle bekannten Informationen, die zur Bestimmung einer optimalen Konfiguration der Posen und somit einer optimalen Karte verwendet werden können.

2.3.3 Backend

Das SLAM-Backend verarbeitet die einschränkenden relativen Transformationen der Odometrie-, Registrierungs- und Schleifenschlusskanten, um eine bestmögliche Knotenkonfiguration des Posengraphen zu finden. Hierzu werden die von den Kanten $(i \to j)$ vorhergesagten Knotenposen $\hat{\mathbf{x}}_j$ mit der aktuellen Schätzung $\mathbf{x}_j$ verglichen, und zur Minimierung der Diskrepanz schrittweise eine Anpassung der geschätzten Knotenkonfiguration vorgenommen. Mathematisch gesehen ist dies ein Ausgleichsproblem, das mit der Methode der kleinsten Quadrate gelöst werden kann (siehe Kapitel 5 für Details). Wichtig ist, dass für jede Vorhersage $\hat{\mathbf{x}}_j$ durch eine Kante die zugehörige Unsicherheit Ω in das Fehlermaß eingeht, so dass Abweichungen für sichere Vorhersagen einen höheren Fehler als für unsichere Vorhersagen verursachen. Durch typische numerische Lösungsverfahren wie das Gauß-Newton-Verfahren lässt sich folglich eine Knotenkonfiguration $\mathcal{X}$ finden, die die bestmögliche Zustandsschätzung auf Basis der vorhandenen Informationen und deren Unsicherheiten darstellt. Eine derartige Optimierung eines topologisch korrekten Posengraphen ist folglich eine Lösung für die *full SLAM*-Formulierung des SLAM-Problems aus Gleichung (2.10).

2.4 Forschungsstand

In diesem Abschnitt soll der aktuelle Stand der Forschung im SLAM-Bereich vorgestellt werden. Zur Recherche wurden zunächst die relevanten bibliographischen Bemerkungen in [Thrun et al., 2005] analysiert. Auf dieser Basis wurde gezielt nach relevanten Einträgen in der *IEEE/IET Electronic Library* (IEL) sowie der *ACM Digital Library* gesucht. Soweit möglich, wurden Publikationen von weiteren Verlagen (z.B. Springer) über alternative Datenbanken, beispielsweise *Google Scholar*, gesucht. Die Quellenangaben gefundener Publikationen wurden gegebenenfalls nach weiteren relevanten Arbeiten durchsucht, um eine breite Spanne des Gebiets abzudecken. Aus Platz- und Relevanzgründen sollen hier jedoch nur grundlegende und wegweisende Verfahren vorgestellt werden, sowie für diese Arbeit besonders relevante Publikationen.

2.4.1 Filterbasiertes SLAM

Der Grundstein filterbasierter SLAM-Verfahren wurde 1986 durch das probabilistische Framework von Smith und Cheeseman gelegt [Smith und Cheeseman, 1986], welches die Basis für EKF-SLAM-Implementierungen wie von Leonard und Durrant-Whyte darstellte [Leonard und Durrant-Whyte, 1991]. Eine wichtige Einschränkung von EKF-SLAM-Verfahren ist der Berechnungsaufwand des Korrekturschritts durch die Inversion der Kovarianzmatrix, deren Größe quadratisch von der Zahl der Landmarken abhängt [Thrun et al., 2005]. Zur eingeschränkten Entkopplung dieser Abhängigkeit werden häufig lokale Teilkarten [Williams et al., 2002] oder hierarchische Strukturen [Paz et al., 2007] verwendet. Das *Extended Information Filter* (EIF) betrachtet stattdessen die filterbasierte Schätzung der Fischer-Informationsmatrix des Systems, und reduziert somit die Inversion der Kovarianzmatrix im Korrekturschritt auf eine Addition zweier Informationsmatrizen [Thrun et al., 2005]. Ansätze wie das *Sparse Extended Information Filter* (SEIF) von Thrun et al. modellieren den Systemzustand explizit als dünn besetzte Informationsmatrix durch topologische Landmarkennetze, so dass der Korrekturschritt des Filters in konstanter Zeit durchgeführt werden kann [Thrun et al., 2005]. Inkonsistenzen dieser Filter durch die Linearisierung mit Taylorreihenent-

wicklung werden beispielsweise durch das *Unscented Kalman Filter* (UKF) adressiert, welches die Linearisierung der Zustandsübergangs- und Beobachtungsfunktion durch die Transformation statistisch signifikanter Punkte approximiert [Julier et al., 1995]. Der beschriebene Partikelfilter-Algorithmus wurde durch Dellaert et al. im Gebiet der mobilen Robotik zum ersten Mal zur Lösung des Lokalisationsproblems vorgestellt [Dellaert et al., 1999]. Für das SLAM-Problem ist ein Standard-Partikelfilter jedoch meist nicht praktikabel, da ein viel größerer Zustandsraum geschätzt werden muss. Eine der bedeutendsten Erweiterungen ist der *FastSLAM*-Algorithmus von Montemerlo et al. [Montemerlo et al., 2002]. Dieser nutzt die als *Rao-Blackwellisierung* [Murphy, 1999] bekannte Entkopplung von Landmarkenhypothesen und Pfadhypothesen, um den Roboterpfad durch die Monte Carlo-Methode zu schätzen, während die einzelnen Partikeln eigene Schätzungen über die Landmarkenposen aufrecht halten. Neben dieser filterbasierten Schätzung der Umgebungskarte werden häufig auch *occupancy grid*-basierte Kartenhypothesen verwendet [Grisetti et al., 2005]. Zur Reduktion des Speicheraufwands werden oft nur lokale Teilkarten in den Partikeln gespeichert, die mit einer globalen Kartenhypothese abgeglichen und somit auch zur Gewichtung der Partikeln genutzt werden [Schroeter und Gross, 2008].

2.4.2 Graphbasiertes SLAM

Im Gegensatz zu filterbasierten SLAM-Verfahren lösen graphbasierte SLAM-Algorithmen das *full SLAM*-Problem. Diese Formulierung reiht sich in eine Vielzahl von Lösungsansätzen ein, beispielsweise *Incremental Smoothing And Mapping* (iSAM) durch iterative Faktorisierung der dünn besetzten Fischer-Informationsmatrix [Kaess et al., 2008]. Die graphbasierte Formulierung ermöglicht eine intuitive Darstellung des SLAM-Problems als Posengraph aus nichtlinearen Beschränkungen, der mit numerischen Lösungsverfahren optimiert werden kann [Grisetti et al., 2010a]. Während das SLAM-Frontend zum Erstellen des Posengraphen sehr sensorspezifisch ist, kann das SLAM-Backend zur Optimierung generischer gehalten werden. Grisetti et al. beschreiben mit *g2o* eine effiziente Methode, das hochdimensionale Optimierungsproblem über allen Roboter- und Landmarkenposen durch hierarchische Zerlegung des Zustandsraums

in mehrere Abstraktionsebenen effizient zu lösen [Grisetti et al., 2010b]. Durch eine Transformation des Zustandsraums in eine Mannigfaltigkeit werden zudem Singularitäten in der Rotationsparametrisierung vermieden, so dass die Optimierung erheblich robuster wird. Der Ansatz von Grisetti et al. geht von korrekter Korrespondenzetablierung aus, so dass topologische Fehler im Posengraphen durch falsche Schleifenschlüsse zu fehlerhaften Lösungen führen. Sünderhauf und Protzel lockern daher in ihrer Publikation die Annahme eines steifen Posengraphen und Integrieren die etablierten Korrespondenzen in den Optimierungsvorgang [Sunderhauf und Protzel, 2012]. Ihre Erweiterung *Vertigo* ergänzt die Graphstruktur von *g2o* um Kanten mit Schaltvariablen, deren Wert den binären Beitrag der Kante zur Optimierung beschreibt. Auf diese Weise kann der Optimierungsvorgang etwaige Kanten, die zur Verletzung anderer Bedingungen führen würden, dynamisch aus- und einschalten.
Das SLAM-Problem in der verbreiteten *least squares*-Formulierung zeichnet sich durch hohe Dimensionalität, Nichtlinearität und Nichtkonvexität aus. Die gute Lösbarkeit durch Linearisierungen weist jedoch auf eine spezielle zugrunde liegende Struktur hin, deren Ausnutzung noch effizientere SLAM-Algorithmen ermöglichen könnte [Wang et al., 2012]. Huang et al. untersuchen diese Struktur und zeigen, dass die Nichtlinearität des Problems durch geschickte Wahl des Zustandsraums sowie lokales Vereinen von Karten erheblich reduziert werden kann [Huang et al., 2010]. Wang et al. überführen die Optimierung eines Posengraphen mit zwei Ankerknoten und sphärischen Kovarianzmatrizen in ein eindimensionales Optimierungsproblem, dessen globales Minimum in analytisch geschlossener Form berechnet werden kann [Wang et al., 2012]. Weitere Forschungsarbeit an der Struktur graphbasierter SLAM-Verfahren könnte das bereits jetzt sehr gute Leistungsverhalten dieser Verfahren folglich noch um einiges verbessern.

2.4.3 Dreidimensionales SLAM

Die Übertragung des SLAM-Problems aus dem zweidimensionalen Raum in den dreidimensionalen Raum bedeutet einen erheblichen Zuwachs an Komplexität. Die Zahl der Freiheitsgrade des Bewegungsmodells steigt von 3DOF auf 6DOF, ebenso steigt die Komplexität der Verarbeitung von Sensormessungen. Nach Bailey und Durrant-Whyte

lassen sich 3D-SLAM Algorithmen durch ihr Verständnis des Adjektivs „dreidimensional“ unterscheiden [Bailey und Durrant-Whyte, 2006]. Die einfachste Variante schätzt den Systemzustand weiterhin im x-y-φ-Raum, nutzt jedoch zusätzliche Sensoren, um die erstellte Karte an den bestimmten Roboterposen in die dritte Raumdimension zu erweitern [Mahon und Williams, 2003].
Derartige Implementierungen basieren jedoch noch immer auf einem Bewegungsmodell mit drei Freiheitsgraden, sie lösen nicht das wesentlich anspruchsvollere 6DOF-SLAM-Problem. Dieses entsteht beispielsweise durch eine in der Hand getragene Kamera oder einen Roboter im Outdoor-Einsatz. Viele filterbasierte 6DOF-SLAM Implementierungen setzen folglich auf zweidimensionale Kameras und spärlich besiedelte Landmarkenkarten, um die hohe Komplexität der Schätzung im sechsdimensionalen Zustandsraum zu beschränken [Davison et al., 2004] [Diaz et al., 2012]. Für viele Anwendungen der Robotik sind derartige topologische Karten jedoch nicht detailliert genug; häufig werden fein aufgelöste metrische Belegtheitskarten des dreidimensionalen Raums bevorzugt. Algorithmen zur Erzeugung derartiger Karten unter Annahme eines Bewegungsmodell mit sechs Freiheitsgraden werden von Nüchter et al. als *Full 6D-SLAM* bezeichnet. Solche Algorithmen nutzen meist den *Iterative Closest Points* (ICP)-Algorithmus zur Registrierung dreidimensionaler Punktewolken [Nuechter et al., 2007].

2.5 Zusammenfassung

In diesem Kapitel wurde das Problem der gleichzeitigen Lokalisation und Kartierung beschrieben. Durch fehlerbehaftete Sensoren ist es nicht möglich, lediglich die Gesamtheit aller Roboterposen und Beobachtungen zu kombinieren, um eine korrekte Karte und den tatsächlich gefahrenen Pfad zu ermitteln. Als Lösung hat sich die gleichzeitige Schätzung der Roboterpose und der Umgebungskarte mittels probabilistischer Methoden etabliert. Meist steht die Schätzung der Robotertrajektorie im Vordergrund, da dies für einen niedrigdimensionalen Zustandsraum sorgt und die Umgebungskarte aus einer korrekten Trajektorie extrahiert werden kann.
Eine der ersten SLAM-Spielarten basiert auf erweiterten Kalman-Filtern, welche die Zustandshypothese als multivariate Normalverteilung modellieren. Bereits bei einfachen

Problemen ist diese Einschränkung auf eine Positionshypothese jedoch oft zu restriktiv, um die tatsächliche Wahrscheinlichkeitsdichte des Zustandsraums zu approximieren. Aus diesem Grund greift eine der populärsten SLAM-Spielarten auf Filter zurück, die auf der sequentiellen Monte Carlo-Methode basieren. Mithilfe sog. Partikeln werden viele Hypothesen abgetastet und evaluiert, um so die tatsächliche, beliebig geartete Wahrscheinlichkeitsdichte des Zustandsraums zu approximieren. Diesen Partikelfiltern ist es daher möglich, mehrere Hypothesen gleichzeitig zu verfolgen, bis neue Sensormessungen eine eindeutigere Aussage ermöglichen. Derartige Verfahren sind durch die explizite Abbildung der Zustandshypothesen für jedes Partikel jedoch sehr berechnungs- und speicherintensiv. Eine neuere Spielart nutzt im Vergleich zu den bisherigen filterbasierten SLAM-Verfahren keine rekursive Zustandsschätzung, sondern repräsentiert die Gesamtheit aller Sensormessungen in einem Posengraphen, in dem durch konventionelle Optimierungsmethoden die gesamte Robotertrajektorie optimiert wird. Diese graphbasierte SLAM-Spielart ist berechnungseffizient und ermöglicht bei geeigneter Konstruktion des Frontends eine robuste Lösung des SLAM-Problems.
Ein solches Verfahren soll in dieser Arbeit entwickelt werden. Zur Konstruktion des Posengraphen durch das SLAM-Frontend ist eine geeignete Darstellungsform der dreidimensionalen Sensormessungen nötig, sowie eine Methode zur Schätzung der relativen Transformation zweier überlappender Beobachtungen. Ebenso muss das Korrespondenzproblem für die betrachteten Beobachtungen gelöst werden, um durch das Entdecken sog. Schleifenschlüsse die topologische Korrektheit des Posengraphen sicherzustellen. Ein topologisch korrekter Posengraph kann durch das SLAM-Backend optimiert werden, um eine konsistente Schätzung der Robotertrajektorie und somit der Umgebungskarte zu erzeugen. In den folgenden Kapiteln sollen die verwendete Darstellungsform und die Registrierungsmethoden, das Entdecken von Schleifenschlüssen und die Konstruktion des Posengraphen näher beschrieben werden.

KAPITEL 3

Erzeugung und Darstellung von Tiefendaten

Bis vor kurzer Zeit stellten zweidimensionale Sensoren wie Sonarsensoren, Lasersensoren oder Kameras die einzigen praktikablen Möglichkeit dar, die Umwelt eines mobilen Roboters wahrzunehmen. Vorhandene dreidimensionale Tiefensensoren waren meist zu langsam oder störanfällig für einen echtzeitfähigen Einsatz, oder die vorhandene Rechenkapazität der Prozessoren reichte nicht aus, um die entstehende Datenmenge effizient zu verarbeiten. Erst neuere Entwicklungen im Bereich der Prozessortechnik, der dreidimensionalen Sensoren und der entsprechenden Datenstrukturen ermöglichten es, Tiefendaten in einer für autonome Roboter ausreichenden Geschwindigkeit aufzunehmen und zu verarbeiten. Trotz dieser Fortschritte stellen dreidimensionale Verfahren aufgrund der inhärent hohen Datendichte noch immer hohe Anforderungen an die Rechenleistung, so dass eine Integration in komplexe echtzeitfähige Algorithmen weiterhin Rücksichtnahme erfordert. Effiziente Datenstrukturen zur Darstellung und Verarbeitung von Tiefendaten stellen daher einen wichtigen Teil dreidimensionaler Verfahren dar.

In diesem Kapitel sollen zunächst übliche Tiefensensoren und Darstellungsformen von Tiefendaten vorgestellt werden. Auf dieser Basis soll dann das in dieser Arbeit genutzte Datenformat zur Darstellung dreidimensionaler Oberflächen genauer beschrieben und evaluiert werden.

3.1 Tiefensensoren

Zur Abtastung einer dreidimensionalen Oberfläche ist ein Sensor nötig, der die Entfernung z eines Punktes (x, y) im Sensorkoordinatensystem bestimmen kann. Häufig werden hier aktive Sensoren verwendet, die die zu messenden Signale selbst erzeugen. Durch bewusste Manipulation der Umgebung ist es diesen Sensoren möglich, zusätzliche Informationen über die Struktur der Umwelt zu gewinnen. In der mobilen Robotik werden häufig sog. Time of Flight (ToF)-Sensoren verwendet, die durch Laufzeitmessung eines Signals die Entfernung des reflektierenden Punktes ermitteln. Eine andere, ebenfalls weit verbreitete Art von Tiefensensoren nutzt hingegen strukturiertes Licht, um durch die oberflächenspezifische Verzerrung eines Musters ein vollständiges Tiefenbild der gemessenen Oberfläche zu ermitteln. Da diese beiden Varianten einen Großteil der verwendeten Sensoren darstellen, sollen sie hier kurz vorgestellt werden.

3.1.1 Time of Flight

Time of Flight (ToF)-Sensoren nutzen die Flugdauer t ausgesendeter Impulse und deren Geschwindigkeit v im Übertragungsmedium, um die Entfernung d der reflektierenden Oberfläche zu bestimmen:

$$d = \frac{vt}{2} \tag{3.1}$$

Die physikalische Charakteristik des verwendeten Impulses wirkt sich stark auf die Präzision der Messung sowie die Anfälligkeit für den störenden Einfluss von Umweltfaktoren aus. Zwei häufig verwendete Signalarten sind Schall und Licht.

3.1.1.1 Ultraschall

Ultraschallsensoren nutzen die Flugdauer hochfrequenter akustischer Signale zur Distanzbestimmung. Durch die starke Abhängigkeit der Ausbreitungsgeschwindigkeit von den Eigenschaften des Übertragungsmediums und die hohe Anfälligkeit für schlechte Reflexionseigenschaften der abgetasteten Oberfläche, ist die Tiefenauflösung eines Sonarsensors meist recht eingeschränkt. Die kegelförmige Ausbreitungscharakteristik des

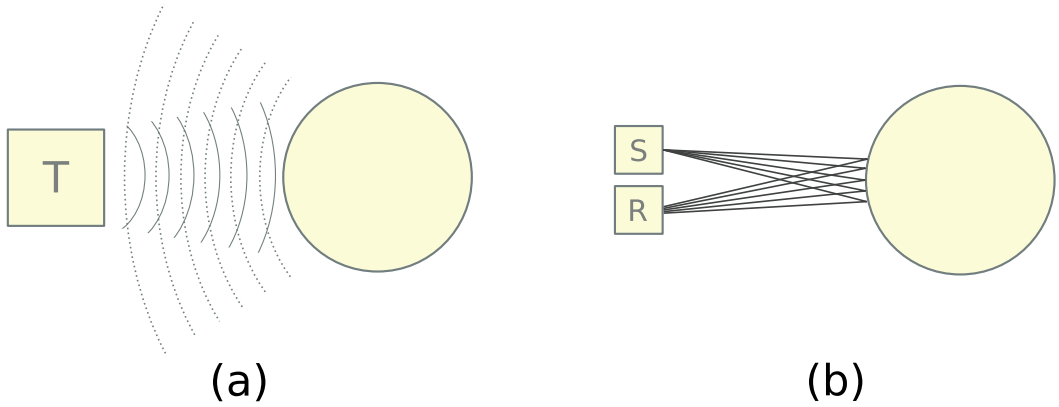


Abbildung 3.1: Illustration der Entfernungsmessung durch schall- (a) und lichtbasierte Impulse (b). Zur Illustration möglicher Konfigurationen wurden Sender und Empfänger des Impulses in Abbildung (b) getrennt, während Abbildung (a) eine kombinierte Einheit (Transceiver) nutzt.

Impulses (siehe Abbildung 3.1 (a)) verhindert zudem eine genaue Richtungsauflösung, so dass derartige Sensoren nur bedingt für die Modellierung dreidimensionaler Oberflächen durch Abtastung geeignet sind. Stattdessen werden Ultraschallsensoren in der mobilen Robotik meist zur groben Erkennung von Hindernispositionen während der autonomen Fahrt verwendet.

3.1.1.2 LIDAR

Erheblich präziser sind *Light Detection And Ranging* (LIDAR) Sensoren, die punktförmige Lichtimpulse im nahen Infrarotbereich aussenden. Diese Sensoren sind relativ unempfindlich für Störungen durch das Übertragungsmedium und erreichen durch gute Reflexionseigenschaften eine hohe Präzision. Da jede Messung die Entfernung eines einzigen Punktes liefert, werden sog. Scanning-Verfahren eingesetzt, um ein vollständiges Entfernungsprofil der gemessenen Oberfläche zu erhalten (siehe Abbildung 3.1 (b) für eine vereinfachte Illustration). Durch Ablenksysteme wie rotierende Spiegel wird dabei eine Menge von Einzelmessungen erzeugt, die als diskretisierter zweidimensionaler Schnitt des Entfernungsprofils der Oberfläche interpretiert werden kann. Dieses Verfahren kann mit ähnlichen Methoden zur Aufnahme eines dreidimensionalen Entfernungsprofils erweitert werden, jedoch sind derartige Sensoren in der Praxis

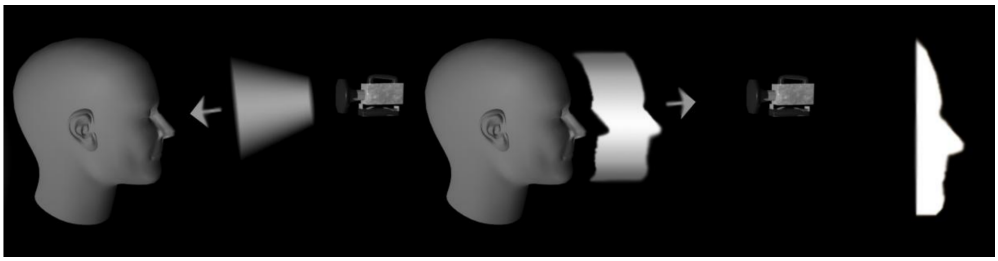


Abbildung 3.2: Illustration der Entfernungsmessung durch eine Time of Flight-Kamera mit intensitätsbasierter Laufzeitmessung [Iddan und Yahav, 2001]. Die auf das abzutastende Objekt gesendete Lichtwand wird durch das Objekt zurückgeworfen und durch die Kamera wieder aufgenommen. Je mehr Licht ein Pixel des Kamerachips wahrnimmt, desto geringer war die Distanz des zugehörigen reflektierenden Oberflächenpunktes.

häufig zu langsam für den Einsatz auf einem mobilen Roboter. Neben Time of Flight-Ansätzen existieren auch LIDAR-Sensoren, die Entfernungen durch Phasenmodulation oder Triangulation bestimmen. In der mobilen Robotik finden jedoch hauptsächlich ToF-LIDAR-Sensoren Einsatz.

3.1.1.3 Time of Flight-Kamera

Im Vergleich zu LIDAR- und Ultraschallsensoren liefern Time of Flight-Kameras mit einer Messung das komplette Entfernungsprofil der gemessenen Oberfläche aus der Perspektive des Sensors (2.5D). Hierzu wird ein kurzer, speziell modulierter Lichtimpuls ausgesendet, der genau die von der Kamera erfasste Szene beleuchtet. Für jeden Pixel des Lichtsensors wird nun die Entfernung durch die Laufzeit des reflektierten Lichtimpulses bestimmt. Eine alternative Methode verwendet schnellschaltende Blenden, um nur die Lichtwellen aufzunehmen, deren Laufzeit t im Intervall $[t_{\min} \dots t_{\max}]$ entsprechend des gewünschten Messintervalls $[d_{\min} \dots d_{\max}]$ liegt. Die Intensitäten der CCD-Pixel sind nun indirekt proportional zu den Laufzeiten des Lichts. Abbildung 3.2 illustriert dieses Prinzip. Um mit Interferenzen durch Hintergrundbeleuchtung umzugehen, werden beispielsweise redundante Aufnahmen ohne Lichtimpuls sowie Wellenlängenfilter benutzt. Durch das verhältnismäßig hohe Rauschverhalten und den konzeptuell bedingt eingeschränkten Sichtwinkel dieser Sensoren finden ToF-Kameras

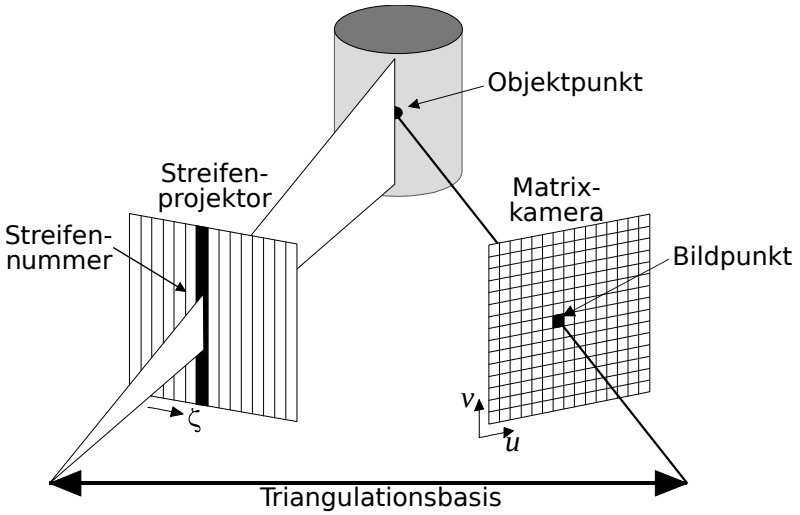


Abbildung 3.3: Illustration der Funktionsweise eines Sensors mit strukturiertem Licht. Ein Projektor (links) mit bekannter geometrischer Transformation zur Kamera (rechts) sendet eine Menge von zeitlich oder örtlich eindeutig kodierten Streifen auf das Objekt. Die Kamera kann nun für jeden Bildpunkt durch die eindeutige Kodierung den zugehörigen Streifen ς und somit den ausgestrahlten Winkel des Streifens bestimmen. Die Entfernung des Punktes ergibt sich nun durch Triangulation.

bislang jedoch nur begrenzt Einsatz auf mobilen Robotern.

3.1.2 Strukturiertes Licht

Eine vergleichsweise neue Art der aktiven Tiefensensoren sind Kameras mit Projektoren für strukturiertes Licht (engl. *structured light*). Im Vergleich zu ToF-Sensoren messen diese Sensoren nicht die Impulslaufzeit, sondern betrachten die charakteristische Verzerrung der Signale durch die beleuchtete Oberfläche. Derartige Sensoren besitzen einen fest ausgerichteten Projektor zur Beleuchtung der Szene mit charakteristischen Lichtmustern, sowie mindestens eine fest ausgerichtete Kamera zur Erfassung der beleuchteten Objekte (siehe Abbildung 3.3).

Ein häufig verwendetes Muster ist eine Menge vertikaler Streifen, die mehrmals mit unterschiedlichen Eigenschaften auf die Szene projiziert werden. Die charakteristische Verzerrung dieser Streifen durch die beleuchtete Oberfläche gibt Aufschluss über die

	Ultraschall	LIDAR	ToF-Kamera	Strukturiertes Licht
Auflösung	-	+	0	+
Distanzabhängigkeit	-	+	+	-
Interferenz	-	+	-	+
Geschwindigkeit	-	-	0	+

Abbildung 3.4: Evaluation der Tiefensensoren zur Modellierung dreidimensionaler Oberflächen.

Oberflächenstruktur im Bereich dieses Streifens. Da die Position des Projektors und der Kamera zueinander bekannt ist, muss zur Berechnung der Entfernung lediglich der zugehörige Streifen der Bildpunkte identifiziert werden.
Dies geschieht beispielsweise durch mehrmalige Projektion des Streifenmusters mit unterschiedlichen Helligkeiten, so dass die fortlaufende Identifikationsnummer jedes Streifens durch die Helligkeitswechsel der betroffenen Bildpunkte binär kodiert wird. Die Genauigkeit dieser Sensoren hängt folglich von der Breite dieser Streifen ab, die jedoch meist nach unten durch die Auflösung der Kamera sowie die Störanfälligkeit der Sensoren beschränkt ist. Derartige Sensoren, beispielsweise die Kinect™ von Microsoft, bieten eine hohe Informationsdichte bei geringen Kosten, und werden aus diesem Grund häufig in der mobilen Robotik eingesetzt.

3.1.3 Evaluation

Die vorgestellten Sensortypen sollen hier kurz unter dem Gesichtspunkt der Modellierung dreidimensionaler Oberflächen evaluiert werden. In Tabelle 3.4 werden die vorgestellten Methoden bezüglich vier wichtiger Kriterien bewertet: Auflösung der Orts- und Tiefendaten, Abhängigkeit der Tiefenauflösung von der Distanz der Objekte, Anfälligkeit für Interferenzen sowie Aufnahmegeschwindigkeit.
Aufgrund der hohen Genauigkeit und geringen Störanfälligkeit gelten LIDAR-Sensoren als de facto Standardsensoren zur Erfassung zweidimensionaler Entfernungsprofile. Die Erweiterung ins Dreidimensionale erfordert jedoch einen Kompromiss zwischen Auflösung und Geschwindigkeit, so dass echtzeitfähige 3D-LIDAR-Sensoren meist eine

recht geringe Auflösung besitzen. Die weiteste Verbreitung finden in diesem Bereich Sensoren mit strukturiertem Licht, die aufgrund der guten Auflösung, der geringen Störanfälligkeit und nicht zuletzt aufgrund der geringen Kosten häufig auf mobilen Robotern eingesetzt werden.

3.2 Darstellungsformen

Alle Tiefensensoren ermitteln für eine endliche Menge von Punkten im Sensorkoordinatensystem die Tiefenwerte der abgebildeten Oberflächenpunkte. Übliche Datenformate sind diskrete Mengen von dreidimensionalen Messpunkten (sog. Punktewolken) und Tiefenbilder, deren Pixel die Tiefeninformation der abgebildeten Punkte farblich kodieren. Häufig sind diese sensornahen Darstellungsformen zu ineffizient oder ungeeignet zur systematischen Verarbeitung durch Algorithmen, so dass die Wahl einer geeigneten Datenstruktur zur Darstellung der Sensordaten eine wichtige Grundlage vieler Verfahren darstellt. Geeignete Formate berücksichtigen die Unsicherheit des Sensors, die Abbildungsgenauigkeit sowie die Berechnungs- und Speicherkosten der Darstellung. In diesem Abschnitt sollen verschiedene Formen der Darstellung vorgestellt werden, um die Entscheidung für die in dieser Arbeit gewählte Datenstruktur fundieren zu können.

3.2.1 Punktewolken

Eine triviale und sensornahe Form der Beschreibung dreidimensionaler Strukturen entsteht direkt durch das Abtasten der Oberfläche in einem diskreten Raster. Die resultierende Menge von dreidimensionalen räumlichen Versatzpunkten (x, y, z) bezüglich des Koordinatensystems des Sensors nennt man Punktewolke. Abbildung 3.5 zeigt eine Menge von Objekten und deren als Punktewolke dargestellte Abtastung durch unterschiedlich aufgelöste Tiefensensoren. Die Vorteile der Punktewolke als Darstellungsform dreidimensionaler Tiefendaten liegen hauptsächlich in ihrer trivialen Berechnung und Speicherung, sowie in der exakten Abbildung der Sensormessung ohne Genauigkeitsverlust. Dies führt jedoch zu verhältnismäßig hohem Speicherbedarf; des Weiteren erschwert die Strukturlosigkeit des Formats eine effiziente algorithmische

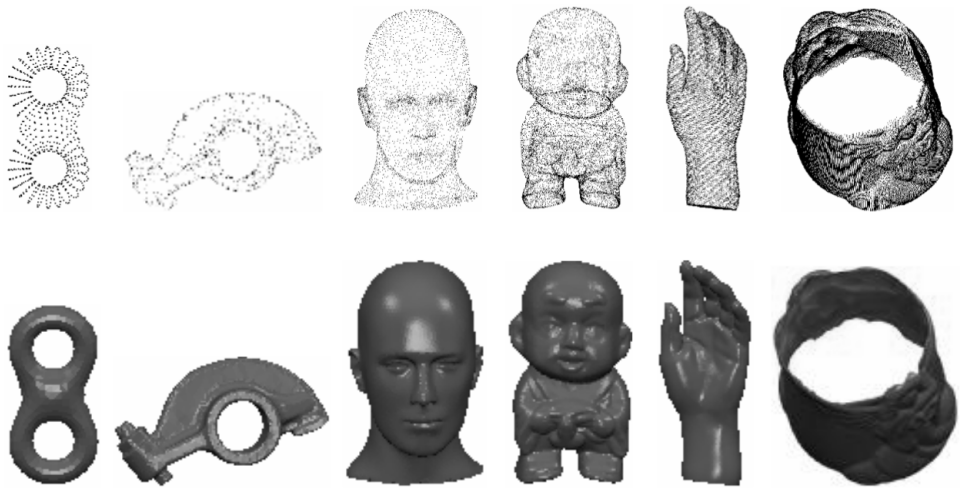


Abbildung 3.5: Verschiedene dreidimensionale Objekte, wahrgenommen durch einen Tiefensensor und dargestellt als Punktewolke (obere Reihe). Die untere Reihe zeigt eine Rekonstruktion der Oberflächenstrukturen durch mathematische Beschreibung mittels des IPD-Algorithmus' [Long et al., 2009].

Verarbeitung der Daten, ermöglicht jedoch die einfache geometrische Transformation der gesamten Datenstruktur. Trotz des hohen Speicherbedarfs werden Punktewolken häufig für dreidimensionale Verfahren verwendet, vor allem in Zusammenhang mit dem einflussreichen ICP-Algorithmus.

3.2.2 Mathematische Oberflächenrekonstruktion

In manchen Fällen ist die implizite Darstellung der Oberflächenstruktur einer Punktewolke durch die Punkte als Stützstellen nicht ausreichend. Um beispielsweise die Schnittflächen zweier Beobachtungen zu bestimmen, muss die Oberflächenstruktur explizit beschreibbar sein. Meist wird daher mit mathematischen Mitteln versucht, diese Information anhand der Messdaten zu approximieren. Eine übliche Methode ist die räumliche Vernetzung der Messpunkte durch Delaunay-ähnliche Triangulation. Fortgeschrittene Verfahren lösen die Triangulation als Approximationsproblem über der lokalen Nachbarschaft und Oberflächenzugehörigkeit der Punkte [Hoppe et al., 1992]. Häufig wird zur Reduktion des Berechnungsaufwands sowohl vor als auch nach

der Oberflächenrekonstruktion eine Reduktion der Datenmenge durch Unterabtastung der Punktewolke oder Zusammenfassen ähnlicher Regionen durchgeführt. Abbildung 3.5 zeigt die durch einen IPD-Algorithmus bestimmten Oberflächenstrukturen der abgebildeten Punktewolken, deren Teilflächen zur Darstellung als Polygone gerendert und anschließend geglättet wurden.
Ein solches mathematisches Format ermöglicht sowohl eine sensornahe Beschreibung der Oberflächenstruktur, als auch eine stark abstrahierte, grobe Approximation. Die Effizienz des Darstellungsverfahrens hängt stark von der Homogenität der Oberfläche und der gewählten Abstraktionsebene ab. Homogene Flächen, die ursprünglich durch tausende Punkte dargestellt wurden, können meist mit einer oder wenigen Formen beschrieben werden. Die Darstellung äußerst inhomogener Flächen benötigt im schlimmsten Fall jedoch sogar mehr Speicherplatz als die zugrundeliegende Punktewolke. Die hohe Berechnungskomplexität der verschiedenen Rekonstruktionsalgorithmen führt dazu, dass diese Datenstruktur eher in der Computergrafik Anwendung findet, als in der mobilen Robotik.

3.2.3 Diskretisierte Belegtheit des Raums

Eine häufig gewählte Form der Modellierung dreidimensionaler Oberflächen diskretisiert den Raum in gleich- oder verschieden große Zellen, in denen Informationen über die Belegtheitswahrscheinlichkeit dieser Zellen stehen. Die trivialste Form dieser Darstellung unterteilt den Raum um die Beobachtung in gleichgroße kubische Unterräume, die genau dann als belegt gelten, wenn sie mehr als k Messpunkte enthalten. Intelligentere Methoden nutzen beispielsweise eine Octree-Diskretisierung, um den Raum effizienter zu unterteilen. Enthält ein Unterraum mehr als m Punkte ($m > k$), so wird dieser wieder unterteilt. Auf diese Weise werden Bereiche mit hoher Punktedichte durch kleinere Unterräume modelliert, während leere oder beinahe leere Bereiche durch große Unterräume beschrieben werden können. Derartige Verfahren ermöglichen eine beliebig genaue und dabei speicherplatzeffiziente Darstellung der Beobachtung, verwerfen dabei jedoch häufig Informationen über die lokale Oberflächenstruktur der modellierten Objekte.

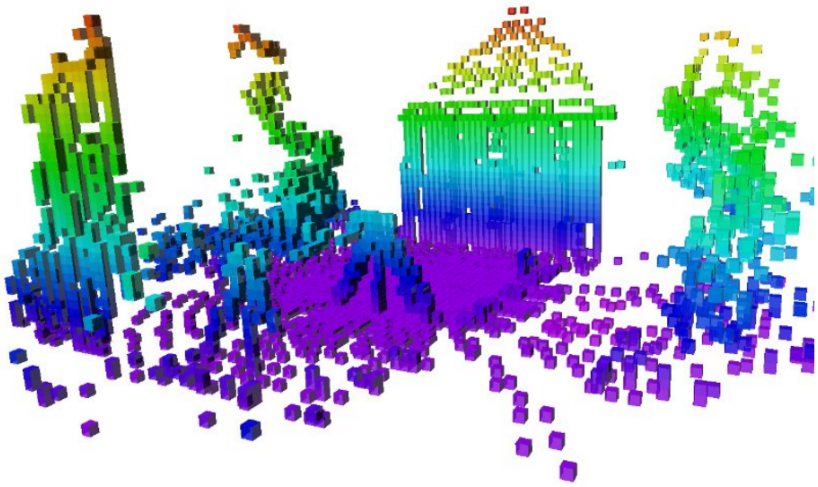


Abbildung 3.6: Dreidimensionale Belegtheitskarte eines Raums, konstruiert aus mehreren Beobachtungen [Stoyanov, 2012]. Die farbige Kodierung der einzelnen Voxel steht für die den Höhenwert z.

3.2.4 Evaluation

Wie schon die Tiefensensoren im vorherigen Abschnitt sollen auch die hier vorgestellten Darstellungsformen unter dem Gesichtspunkt der Modellierung dreidimensionaler Oberflächen evaluiert werden. In Tabelle 3.7 werden die vorgestellten Verfahren bezüglich drei wichtiger Kriterien bewertet: Genauigkeit der Abbildung, Komplexität des Berechnungsvorgangs und Aussagefähigkeit über die lokale Oberflächenstruktur.
Wie bereits erwähnt, werden Punktewolken durch ihre Simplizität häufig als Datenstruktur dreidimensionaler Verfahren eingesetzt. Wenig Verbreitung findet die mathematische Rekonstruktion der Oberflächenstruktur, die trotz präziser Ergebnisse für den Einsatz in echtzeitfähigen Verfahren zu komplex ist. Die Abbildung der Belegtheitsstruktur des Raums durch Octree-Diskretisierung bietet eine intelligente und effiziente Unterteilung des Raums, verwirft jedoch wichtige Informationen über die lokale Oberflächenstruktur. Da diese Information gerade für Korrespondenzprobleme wie die Registrierung hilfreich ist, soll im folgenden Abschnitt ein Verfahren vorgestellt werden, welches auch die lokale Oberflächenstruktur modelliert.

	Punktewolke	Rekonstruktion	Diskretisierte Belegtheit
Genauigkeit	+	+	+
Berechnungseffizienz	+	-	+/-
Oberflächenstruktur	-	+	-

Abbildung 3.7: Evaluation der Datenstrukturen zur Modellierung dreidimensionaler Oberflächen.

3.3 Normal Distributions Transform

Wie im vorherigen Abschnitt beschrieben, bietet die Modellierung der Belegtheit des Raums durch Diskretisierung eine effiziente und präzise Abbildung, verwirft jedoch die aus der ursprünglichen Beobachtung extrahierbaren Informationen über die lokale Oberflächenkrümmung und -orientierung. Es wurde angenommen, dass für jede diskrete Zelle die Belegtheit durch eine Belegtheitswahrscheinlichkeit $p \in [0, 1]$ beschrieben wird. Es ist möglich, diese triviale Beschreibung durch ein mathematisches Modell zu ersetzen, das sowohl die lokale Oberflächenstruktur als auch die Belegtheit der Zelle beschreiben kann. Dieses Prinzip nutzt der sog. *Normal Distributions Transform* (NDT), der eine Beobachtung in eine diskrete Menge multivariater Normalverteilungen transformiert.

3.3.1 Definition

Unter dem Normal Distributions Transform versteht man den Transformationsvorgang einer diskreten Menge von Punkten $\mathbf{x}$ in eine stückweise kontinuierliche und differenzierbare Wahrscheinlichkeitsverteilung [Biber und Strasser, 2003]. Hierzu wird eine Punktemenge S nach geeigneten Kriterien in räumlich disjunkte Teilmengen T_i zerlegt, deren Teilräume als (NDT-)Zellen bezeichnet werden. Die lokale Punktemenge einer Zelle kann als Ziehung aus deren unbekannter Oberflächenverteilungsfunktion interpretiert werden, so dass die Schätzung der lokalen Punkteverteilung einer stochastischen Approximation der Oberflächenstruktur entspricht. Zusammen mit einer intelligenten Diskretisierungsmethode des Raums lässt sich auf diese Weise ein präzises Modell der gesamten gemessenen Oberfläche erzeugen.

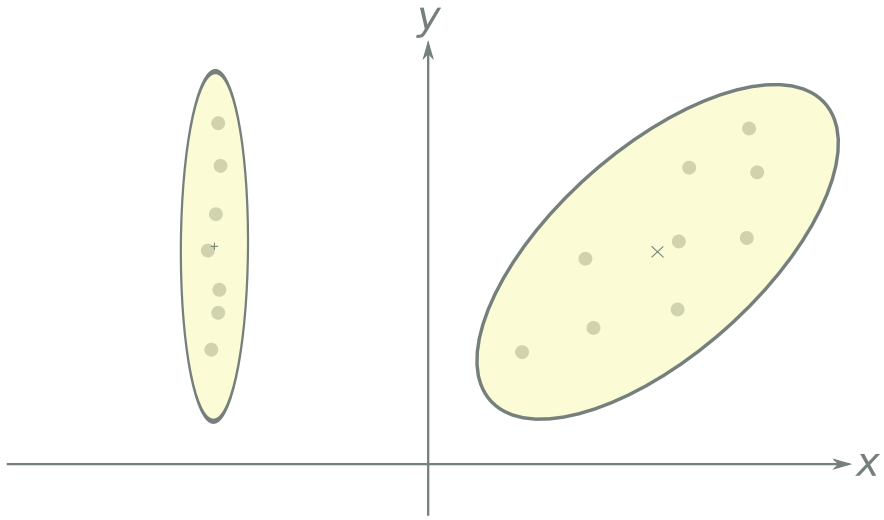


Abbildung 3.8: Veranschaulichung der Berechnung der Normalverteilungen für zwei Punktemengen. Zur Darstellung der berechneten Verteilungen wurden die 3σ-Fehlerellipsen aus den Parametern μ und Σ berechnet.

Aufgrund der einfachen Berechnung, Darstellung und Parametrisierbarkeit werden multivariate Normalverteilungen $N(\mu_i, \Sigma_i)$ eingesetzt, um die Punkteverteilungen der Teilmengen $T_i \subseteq S$ zu berechnen. Die Mittelwerte μ_i und Kovarianzmatrizen Σ_i einer Zelle i ergeben sich wie folgt:

$$\mu_i = \frac{1}{||T_i||} \sum_j \mathbf{x}_j, \quad \mathbf{x}_j \in T_i \tag{3.2}$$

$$\Sigma_i = \frac{1}{||T_i||} \sum_j (\mathbf{x}_j - \mu_i)(\mathbf{x}_j - \mu_i)^T, \quad \mathbf{x}_j \in T_i \tag{3.3}$$

Abbildung 3.8 zeigt am Beispiel eines zweidimensionalen Problems die für zwei Punktemengen approximierten Normalverteilungen anhand ihrer 3σ-Fehlerellipsen. Für jede Zelle i lässt sich nun die Wahrscheinlichkeitsdichte $p(\mathbf{x}|T_i)$ berechnen:

$$p(\mathbf{x}|T_i) = \frac{1}{(2\pi)^{\frac{3}{2}} |\Sigma_i|^{\frac{1}{2}}} \exp\left(-\frac{(\mathbf{x} - \mu_i)^T \Sigma_i^{-1} (\mathbf{x} - \mu_i)}{2}\right) \tag{3.4}$$

Für einen beliebigen Punkt $\mathbf{x}$ beschreibt diese Funktion die Wahrscheinlichkeit, dass $\mathbf{x}$ durch $N(\mu_i, \Sigma_i)$ verteilt ist, also mit welcher Wahrscheinlichkeit $\mathbf{x}$ zur modellierten Oberfläche der Zelle i gehört.

In der Literatur finden sich häufig konstante, empirisch bestimmte Regulierungsfaktoren, um die Berechnung der Wahrscheinlichkeitsdichte zu vereinfachen [Magnusson,

2009][Stoyanov et al., 2012]. Im Folgenden wird daher eine vereinfachte Form der Formel (3.4) verwendet:

$$p(\mathbf{x}|T_i) = d_1 \exp\left(-d_2 \frac{(\mathbf{x}-\mu_i)^T \Sigma_i^{-1} (\mathbf{x}-\mu_i)}{2}\right) \tag{3.5}$$

Die Menge $N(\mu_i, \Sigma_i)$ der sog. NDT-Komponenten aller Zellen i einer diskretisierten Beobachtung $\mathcal{S}$ nennt man den Normal Distributions Transform von $\mathcal{S}$. Da diese Art der Modellierung wie schon die mathematische Oberflächenrekonstruktion in der Lage ist, große homogene Teilmengen effizient zu beschreiben, ist eine intelligente Diskretisierungsmethode wünschenswert, die Rücksicht auf die Verteilung der Punkte nimmt.

3.3.2 Räumliche Diskretisierung

Zur Konstruktion eines Normal Distributions Transforms muss ein geeignetes Kriterium existieren, um die Messdaten $\mathcal{S}$ in räumlich disjunkte Teilmengen $\mathcal{S} = \bigcup_i T_i$ zu unterteilen. Biber und Magnusson wählen regelmäßige Zellen gleicher Größe, so dass jede Zelle entweder eine Teilmenge T_i oder Freiraum beinhaltet [Magnusson, 2009]. Die Wahl der Größe dieser Zellen stellt dabei eine Abwägung zwischen Genauigkeit der Darstellung und Speicherbedarf dar. Durch größere Zellen verringert sich die Zahl der benötigten Normalverteilungen, jedoch verschwimmen dabei kleinere Features zugunsten der Approximation der Oberflächenstruktur der gesamten Zelle. Kleinere Zellen führen im Umkehrschluss zu einer präziseren Modellierung bei höherem Speicherbedarf. Es existiert jedoch eine untere Grenze für die Zellgröße, die sich durch die Berechnung der Normalverteilungen begründet. Nach Magnusson werden mindestens 5 Punkte benötigt, um in Gleichung (3.3) eine zuverlässige Kovarianzmatrix zu erhalten. Die minimale Zellgröße lässt sich somit durch die Auflösung, Art und Reichweite des verwendeten Sensors schätzen.

3.3.2.1 Octree-Diskretisierung

Um die Vorteile großer und kleiner Zellen zu kombinieren, bietet sich eine variable Wahl der Zellgröße an. Magnusson vergleicht in seiner Arbeit verschiedene derartige

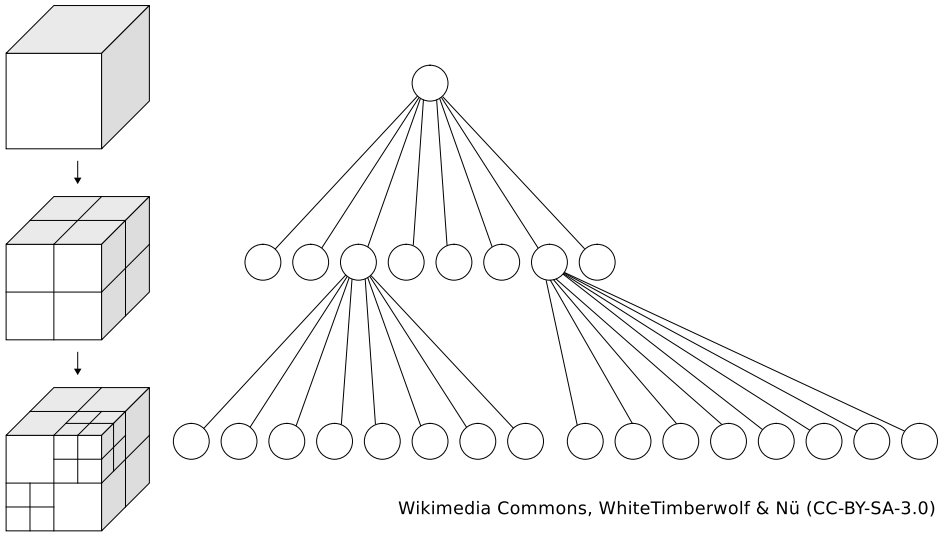


Abbildung 3.9: Veranschaulichung der Octree-Diskretisierung eines dreidimensionalen Raums und der entstehenden Baumstruktur. Begonnen mit einem allumfassenden Würfel wird der Raum bei Bedarf in acht gleichgroße Teilräume unterteilt, die die Kinder des unterteilten Würfels darstellen. Dieser Vorgang wird rekursiv wiederholt, bis die gewünschte Diskretisierung des Raums erreicht wird.

Verfahren, beispielsweise adaptives Clustering durch k-means Clustering oder Octree-Diskretisierung. Ein Octree ist ein Baum, in dem jeder Knoten entweder acht oder keine Kinder besitzt und einen bestimmten dreidimensionalen Teilraum repräsentiert [Meagher, 1982]. Die Wurzel des Baums deckt hierbei den gesamten Raum des Baums ab. Abbildung 3.9 zeigt schematisch die Unterteilung eines Raums in Teilräume unterschiedlicher Größen, sowie die dabei entstehende Baumstruktur.

In der von Magnusson vorgeschlagenen Variante der Octree-Diskretisierung wird ein Wald von Octrees erzeugt, indem der Raum zuerst in regelmäßige Zellen gleicher Größe unterteilt wird, welche darauf intern eine Octree-Diskretisierung ihres Teilraums durchführen. Als Teilungskriterium für die Octree-Knoten gilt die Anzahl der in den Knoten enthaltenen Messpunkte. Als Grund gegen einen einzigen, globalen Octree führt Magnusson die Notwendigkeit einer schnellen Zuordnung Punkt-zu-Zelle an [Magnusson, 2009].

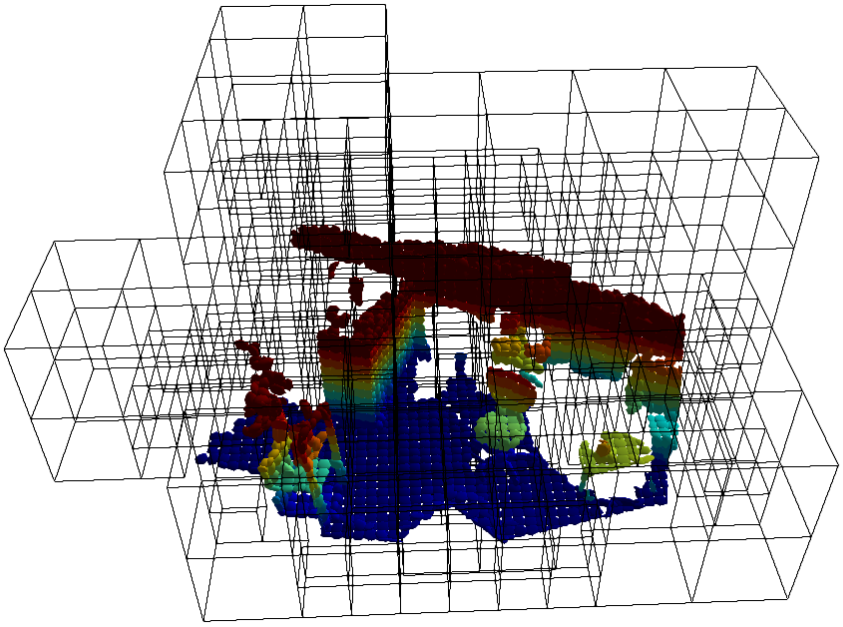


Abbildung 3.10: Demonstration eines Normal Distributions Transforms einer Beobachtung. Jede NDT-Zelle wird durch eine dreidimensionale 3σ-Fehlerellipse dargestellt, die farblich ihre Höhe z kodiert. Das schwarze Gitter beschreibt einen selektiven Ausschnitt der verwendeten Octree-Diskretisierung.

3.3.3 Beschreibung des gewählten Verfahrens

In dieser Arbeit werden Octrees genutzt, um den Normal Distributions Transform einer Beobachtung auf intelligent diskretisierten Zellen zu berechnen. Im Gegensatz zur Methode von Magnusson wird jedoch nur ein einziger Octree O zur Diskretisierung des Raums eingesetzt. Durch approximierende Nächste-Nachbarn-Suche (siehe Kapitel 4) über den Knoten des Octrees wird eine schnelle Punkt-zu-Zelle-Zuordnung erreicht, so dass die Geschwindigkeit eines Octree-Walds mit der Abbildungseffizienz eines einzelnen Octrees verbunden wird. Abbildung 3.10 zeigt einen auf diese Art berechneten Normal Distributions Transform einer umfangreichen Beobachtung, mit angedeuteter Struktur der räumlichen Octree-Diskretisierung.

Ein weiterer Unterschied zur Arbeit von Magnusson ist die Berechnung der Normalverteilungen für jeden Knoten der unterschiedlichen Diskretisierungsebenen, nicht nur

für die Blätter des Octrees. Durch eine geschickte rekursive Berechnungsvorschrift ist dies nicht viel aufwändiger als die Berechnung der Normalverteilungen der Blätter, ermöglicht jedoch die Extrahierung mehrerer unterschiedlich aufgelöster Normal Distributions Transforms. Abbildung 3.11 zeigt beispielhaft drei verschiedene derartige Diskretisierungsebenen anhand des aus Abbildung 3.10 bekannten Normal Distributions Transforms.

3.3.3.1 Algorithmische Beschreibung

Algorithmus 1 beschreibt den Vorgang der NDT-Berechnung mit Octree-Diskretisierung als Ablaufdiagramm. Ausgehend vom Wurzelknoten, der den gesamten Raum der Beobachtung abdeckt, werden die Knoten des Octrees so lange unterteilt, bis jedes Blatt maximal m Messpunkte beinhaltet. Gemäß der Bedingung für eine robuste Berechnung der Kovarianzmatrix wird die zugehörige Normalverteilung $N(\mu_k, \Sigma_k)$ nur berechnet, wenn der Knoten mindestens $l > 5$ Punkte beinhaltet. Für jeden Knoten k gelten nun folgenden Eigenschaften:

1. Der Knoten k deckt die Zelle k mit Punktemenge T_k ab.
2. Der Knoten k ist ein Blatt, wenn $|T_k| \leq m$.
3. Der Knoten k besitzt eine Normalverteilung $N(\mu_k, \Sigma_k)$, wenn $|T_k| \geq l$.
4. Die Belegtheitswahrscheinlichkeit des Knotens k berechnet sich über $|T_k|$.

Der Octree O beschreibt nun den Normal Distributions Transform der Beobachtung S durch die Menge der Normalverteilungen $\bigcup_k N(\mu_k, \Sigma_k)$, die durch intelligente Diskretisierung auf Zellen variabler Größe berechnet wurden.

3.3.4 Evaluation

Die intelligente Octree-Diskretisierung ermöglicht eine speicherplatzeffiziente Modellierung der abgebildeten Oberflächenstruktur mit dem Normal Distributions Transform. Diese Form der Darstellung benötigt bei gleicher Genauigkeit weniger Speicher als eine Diskretisierung mit fester Zellgröße, da beispielsweise Freiräume durch größere

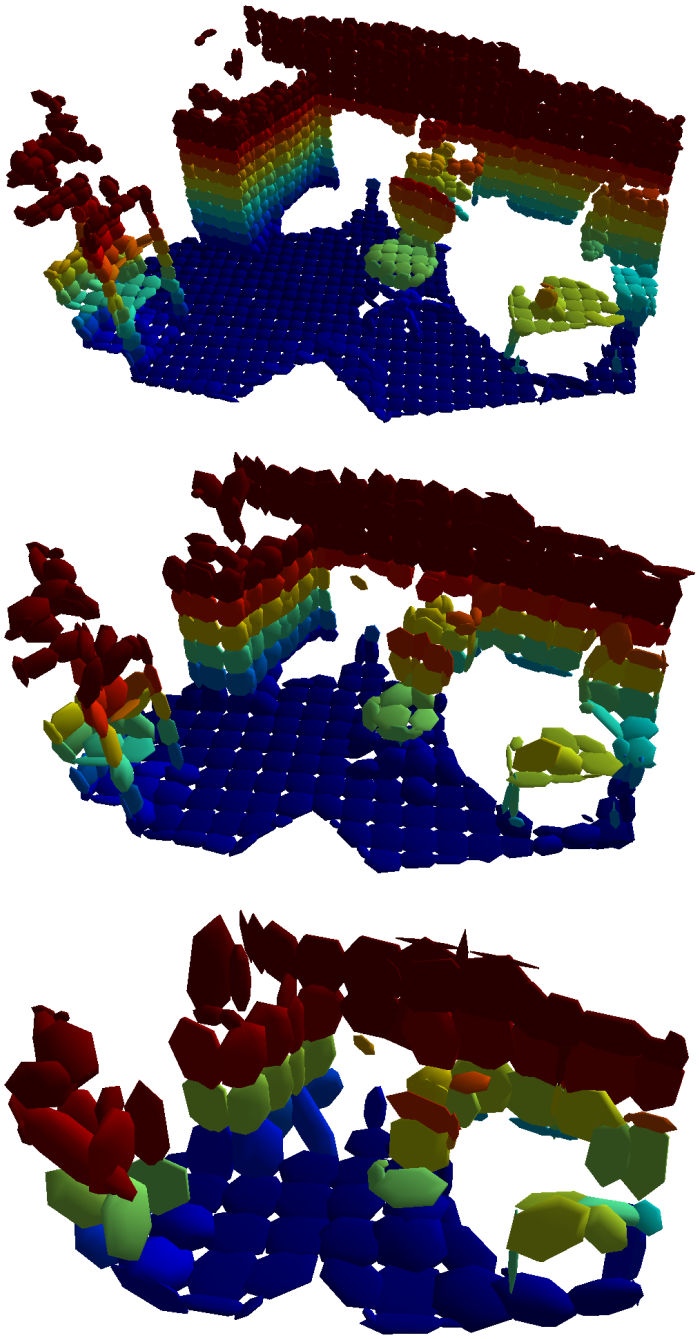


Abbildung 3.11: Abbildung des Normal Distributions Transforms einer Beobachtung. Jede NDT-Zelle wird durch eine dreidimensionale 3σ-Fehlerellipse dargestellt, die farblich durch ihre absolute Höhe kodiert wird. Gezeigt werden drei unterschiedlich aufgelöste Diskretisierungsebenen des Normal Distributions Transforms.

Input : Beobachtung $\mathcal{S}$, Teilungsgrenze m, Mindestanzahl l
Result : Octree O mit Normal Distributions Transform $\bigcup_k N(\mu_k, \Sigma_k)$

Initialisiere Wurzel des leeren Octrees O auf Raum der Messpunkte;
`// Octree-Diskretisierung`
while *Blatt $k \in O$ beinhaltet mehr als m Messpunkte* **do**
  Zerlege Knoten k in acht Kinder $c_1, \ldots, c_8$;
  Füge neue Knoten $c_1, \ldots, c_8$ in den Baum ein;
end
`// Berechnung des Normal Distributions Transforms`
for *Knoten $k \in O$* **do**
  Sei $T_k \subseteq \mathcal{S}$ die Menge der Messpunkte in k;
  if $|T_k| \leq l$ **then**
    Überspringe ungenügend belegten Knoten k;
  end
  Berechne μ_k und Σ_k aus T_k mit Gleichungen (3.2)(3.3);
  Speichere μ_k und Σ_k in Knoten k;
end

Algorithmus 1: Normal Distributions Transform mit Octree-Diskretisierung.

Zellen modelliert werden können. Des Weiteren ermöglicht diese Methode gegenüber statischer Diskretisierung eine höhere Abbildungsgenauigkeit, da Zellen mit hoher Varianz der Punkteverteilung weiter unterteilt werden können, bis die Oberfläche des Objekts möglichst genau approximiert wird. Ein Nachteil dieser Methode ist der im Vergleich zur statischen Diskretisierung höhere Aufwand, um die zugehörige Zelle für einen Punkt $\mathbf{x}$ zu finden. Durch die in dieser Arbeit verwendete approximierende Nächste-Nachbarn-Suche ist der hierfür notwendige Aufwand jedoch vernachlässigbar. Durch den Normal Distributions Transform approximiert jede Zelle bzw. jeder Knoten mithilfe einer multivariaten Normalverteilung die lokale Oberflächenstruktur des abgedeckten Teilraums. Die Wahrscheinlichkeit eines Punktes $\mathbf{x} \in \mathbb{R}^3$, in den durch Knoten k abgedeckten Raum zu fallen, lässt sich durch Formel (3.4) explizit berechnen. Durch

die Art der Berechnung hängt $p(\mathbf{x}|k)$ nicht nur von der Lage von $\mathbf{x}$ in Relation zu μ_k ab, sondern durch den Faktor Σ_k^{-1} auch von der in Knoten k modellierten Oberflächenstruktur. Die Aussagekraft über die mögliche Zugehörigkeit $\mathbf{x} \rightarrow k$ ist folglich höher als bei Verfahren, die die Wahrscheinlichkeit $p(\mathbf{x}|k)$ auf Basis der geometrischen Grenzen der Zelle k berechnen.

3.4 Zusammenfassung

In diesem Kapitel wurde die Erzeugung und Darstellung der zu verwendenden Tiefendaten näher vorgestellt. Durch Tiefensensoren wie Time of Flight-Kameras, LIDARs und Sensoren mit strukturiertem Licht werden Tiefendaten eines festgelegten Sichtbereichs gemessen, die explizit oder implizit als dreidimensionale Punktewolken dargestellt werden können. Zur effizienten Darstellung der durch diese Punktewolken beschriebenen Oberflächen existieren verschiedene Verfahren, von denen die Diskretisierung des Messraums in Zellen fester oder variabler Größe am verbreitetsten ist. Die statische Diskretisierung mit fester Zellgröße ermöglicht eine schnelle Zuordnung von Punkten zu Zellen, muss jedoch einen nicht-trivialen Kompromiss zwischen Abbildungsgenauigkeit und Speichereffizienz treffen. Variable Diskretisierung durch Octrees ermöglicht hingegen eine speicherplatzeffiziente Darstellung bei hoher Abbildungsgenauigkeit, da Zellen basierend auf der Struktur des abgebildeten Teilraums häufiger oder seltener geteilt werden können. Im Unterschied zur Speicherung einer Belegtheitswahrscheinlichkeit $p_k \sim [0, 1]$ beschreibt der Normal Distributions Transform die lokale Oberflächenstruktur des durch die Zelle k abgedeckten Teilraums anhand einer multivariaten Normalverteilung. Der Normal Distributions Transform mit Octree-Diskretisierung bietet folglich eine speicherplatzeffiziente Form der Darstellung dreidimensionaler Oberflächen, die zudem Informationen über die lokale Oberflächenstruktur bewahrt. Auf Basis dieser Datenstruktur sollen nun im nächsten Kapitel Verfahren vorgestellt werden, um durch den Vergleich zweier Beobachtungen deren relative Transformation zu bestimmen.

KAPITEL 4

Registrierung von Normal Distributions Transforms

Mit der im vorherigen Kapitel vorgestellten Darstellungsform von Tiefendaten lassen sich die während des SLAM-Vorgangs aufgenommenen Beobachtungen kompakt und effizient repräsentieren. Nachdem in den letzten Kapiteln stets einzelne, abstrakte Sensormessungen als Beobachtungen bezeichnet wurden, wird dieser Begriff nun enger definiert. Im Folgenden sei eine Beobachtung ein Normal Distributions Transform, der aus beliebig vielen Sensormessungen erzeugt wurde (eine Beobachtung ist folglich ebenfalls eine Karte). Der nächste Schritt ist nun der Vergleich zweier Beobachtungen $\mathbf{z}_i \leftrightarrow \mathbf{z}_j$, die zu einem bestimmten Anteil dieselbe Oberfläche gemessen haben. Durch diese Überlappung ist es prinzipiell möglich, Informationen über den tatsächlichen geometrischen Versatz der Aufnahmeposen $\mathbf{x}_i \leftrightarrow \mathbf{x}_j$ zu bestimmen. Das Ermitteln dieses Versatzes durch das algorithmische Einpassen der Beobachtungen wird als *Registrierung* bezeichnet. Meist wird die Beobachtung mit der geringeren Unsicherheit bezüglich ihrer Aufnahmepose als statisch betrachtet, um die Anzahl der Freiheitsgrade zu reduzieren. Unter Registrierung versteht man folglich das Ermitteln der relativen Transformation für eine bewegliche Beobachtung $\mathcal{S}$, die zu einer perfekten Übereinstimmung mit einer statischen Beobachtung $\mathcal{M}$ führt. Dieses Verfahren ist in den meisten SLAM-Algorithmen in zwei unterschiedlichen Varianten enthalten:

1. **Messungsüberdeckung** (engl. *scan matching*): Registrierung aufeinanderfolgender Beobachtungen, für die durch den Odometrieversatz eine ausreichend scharfe Schätzung der relativen Transformation vorliegt.

2. **Schleifenschluss** (engl. *loop closing*): Registrierung zweier Beobachtungen, für die durch den akkumulierten Odometriefehler keine ausreichend scharfe Schätzung der relativen Transformation mehr möglich ist.

Meist kann durch die Registrierung eine genauere relative Transformation bezüglich der Aufnahmeposen bestimmt werden, als von der Odometrie vorgegeben wurde. Gerade durch den Schleifenschluss trägt die Registrierung essentiell zum Leistungsverhalten eines SLAM-Verfahrens bei. Im folgenden Kapitel soll daher die Registrierung zweier als Normal Distributions Transforms vorliegender Beobachtungen betrachtet werden.

4.1 Transformation im $\mathbb{R}^3$

Das Ziel der Registrierung ist es, Aussagen über die relative Transformation zweier überlappender Normal Distributions Transforms $\mathcal{M}$ und $\mathcal{S}$ zu treffen. Im dreidimensionalen Raum $\mathbb{R}^3$ wird eine solche Transformation durch einen Translationsvektor $\mathbf{t} = (t_x, t_y, t_z)$ und eine Rotation aus der Drehgruppe $\mathbb{SO}(3)$ beschrieben werden. Die Darstellung dreidimensionaler Rotationen ist nicht-trivial, soll an dieser Stelle jedoch nicht weiter vertieft werden. Im Folgenden werden Rotationen durch die Eulerwinkel (ϕ_x, ϕ_y, ϕ_z) beschrieben, die im Vergleich zu beispielsweise Quaternionen das Differenzieren der Transformationsfunktion erleichtern und keine zusätzlichen Bedingungen im Optimierungsvorgang erfordern [Magnusson, 2009]. Zur Anwendung einer Rotation mit Eulerwinkeln werden diese meist in Drehmatrizen überführt, die im Folgenden mit $\mathbf{R}_{\phi_x}, \mathbf{R}_{\phi_y}, \mathbf{R}_{\phi_z}$ beschrieben werden. Eine Transformation kann folglich durch einen sechsdimensionalen Vektor $\mathbf{p}$ parametrisiert werden:

$$\mathbf{p} = (t_x, t_y, t_z, \phi_x, \phi_y, \phi_z) \tag{4.1}$$

Für diesen Parametervektor sei die Transformationsfunktion $T(\mathbf{x}, \mathbf{p})$ eines Punktes $\mathbf{x} \in \mathbb{R}^3$ wie folgt definiert:

$$T(\mathbf{x}, \mathbf{p}) = \mathbf{R}_{\phi_x}\mathbf{R}_{\phi_y}\mathbf{R}_{\phi_z}\mathbf{x} + \mathbf{t} \tag{4.2}$$

Im Folgenden wird die Parametrisierung einer Transformation durch den Vektor $\mathbf{p}$ zur Vereinfachung direkt als Transformation bezeichnet.

4.1.1 Transformation von Normal Distributions Transforms

Die Beobachtungen $\mathcal{M}$ und $\mathcal{S}$ überlappen sich; sie stellen also zu gewissen Anteilen dieselben Umgebungsmerkmale dar. Es existiert folglich eine räumliche Anordnung von $\mathcal{M}$ und $\mathcal{S}$, so dass die übereinstimmenden Anteile bestmöglich überlappen. Durch den Odometrieversatz ist meist eine akzeptable, aber dennoch fehlerbehaftete Schätzung dieser Transformation gegeben. Das Ziel der Registrierung ist es folglich, die relative Transformation $\mathbf{p}$ zu finden, für die die beiden Beobachtungen bestmöglich überlappen:

$$\mathcal{M} = T\left(\mathcal{S}, \mathbf{p}\right) \tag{4.3}$$

Die Anwendung der Transformationsfunktion aus Formel (4.3) auf den Normal Distributions Transform $\mathcal{S}$ bewirkt die Transformation einer Menge von Normalverteilungen $\mathcal{S} = \bigcup_i N\left(\mu_i, \Sigma_i\right)$. Wie in Kapitel 3 beschrieben, werden alle Komponenten eines Normal Distributions Transforms unabhängig voneinander berechnet, so dass auch die Transformationsfunktion für jede NDT-Komponente $N\left(\mu_i, \Sigma_i\right)$ separat angewendet wird:

$$T\left(\mu_i, \mathbf{p}\right) = \mathbf{R}_{\phi_x}\mathbf{R}_{\phi_y}\mathbf{R}_{\phi_z}\mu_i + \mathbf{t} \tag{4.4}$$

$$T\left(\Sigma_i, \mathbf{p}\right) = \left(\mathbf{R}_{\phi_x}\mathbf{R}_{\phi_y}\mathbf{R}_{\phi_z}\right)^T \Sigma_i \left(\mathbf{R}_{\phi_x}\mathbf{R}_{\phi_y}\mathbf{R}_{\phi_z}\right) \tag{4.5}$$

Die Aufgabe der Registrierung ist es folglich, eine Menge von disjunkten Normalverteilungen $\mathcal{S}$ derartig zu transformieren, dass sie bestmöglich mit der Ground Truth-Vorgabe $\mathcal{M}$ übereinstimmt.

4.2 Distanzmetrik

Die in Formel (4.3) beschriebene Gleichung setzt voraus, dass $\mathcal{M}$ auf $\mathcal{S}$ abgebildet werden kann. Eine exakte Abbildung existiert jedoch nur dann, wenn $\mathcal{M}$ und $\mathcal{S}$ identisch sind. Durch das inhärente Sensorrauschen und die unterschiedlichen Aufnahmeposen werden nicht-identische Beobachtungen jedoch nie perfekt übereinstimmen können. Aus

diesem Grund ist das Ziel der Registrierung das Finden einer relativen Transformation $\mathbf{p}$, die den Grad der Übereinstimmung maximiert. Um dies für eine bestimmte Transformationen numerisch bewerten zu können, ist eine Metrik zur Schätzung der Disparität bzw. Distanz der transformierten Normal Distributions Transforms nötig:

$$d\left(\mathcal{M}, T\left(\mathcal{S}, \mathbf{p}\right)\right) \rightarrow \mathbb{R} \tag{4.6}$$

Eine derartige Metrik ermöglicht das Vergleichen und qualitative Sortieren unterschiedlicher Transformationen $\mathbf{p}_i$ und $\mathbf{p}_j$:

$$\mathbf{p}_i < \mathbf{p}_j \Leftrightarrow d\left(\mathcal{M}, T\left(\mathcal{S}, \mathbf{p}_i\right)\right) < d\left(\mathcal{M}, T\left(\mathcal{S}, \mathbf{p}_j\right)\right) \tag{4.7}$$

Eine solche qualitative Metrik ist der Grundstein, um eine approximierte Lösung für die nicht direkt lösbare Gleichung (4.3) zu ermitteln.

4.2.1 Distanzmetrik für Normal Distributions Transforms

Stoyanov et al. haben mehrere Distanzmetriken für Normal Distributions Transforms betrachtet, beispielsweise die Kullback-Leibler-Divergenz, oder ein Distanzmaß auf Basis der Mutual Information. Derartige Metriken bieten jedoch keine analytisch berechenbaren Ableitungen, weshalb sie sich nicht zum Einsatz für die meisten Lösungsverfahren eignen. Aus diesem Grund schlagen Stoyanov et al. die L2-Distanz als Distanzmetrik für zwei Normal Distributions Transforms vor [Stoyanov et al., 2012].
Die L2-Distanz D_2 berechnet die L2-Norm zweier Wahrscheinlichkeitsdichten für alle möglichen Inputs. Für zwei Normal Distributions Transforms $\mathcal{M}$ und $\mathcal{S}$ berechnet sich D_2 unter Berücksichtigung der Transformation $\mathbf{p}$ über allen möglichen Inputs $\mathbf{x} \in \mathbb{R}^3$ wie folgt:

$$D_2\left(\mathcal{M}, \mathcal{S}, \mathbf{p}\right) = \int \left(p\left(\mathbf{x}|\mathcal{M}\right) - p\left(\mathbf{x}|T\left(\mathcal{S}, \mathbf{p}\right)\right)\right)^2 d\mathbf{x} \tag{4.8}$$

$$\begin{aligned} &= \int p\left(\mathbf{x}|\mathcal{M}\right)^2 d\mathbf{x} + \int p\left(\mathbf{x}|T\left(\mathcal{S}, \mathbf{p}\right)\right)^2 d\mathbf{x} \\ &\quad - 2\int p\left(\mathbf{x}|\mathcal{M}\right) p\left(\mathbf{x}|T\left(\mathcal{S}, \mathbf{p}\right)\right) d\mathbf{x} \end{aligned} \tag{4.9}$$

Die Terme der Form $p\left(x|\mathcal{N}\right)$ stehen für die Wahrscheinlichkeit, dass ein Punkt $\mathbf{x}$ von dem durch $\mathcal{N}$ beschriebenen Normal Distributions Transform generiert wurde. Über die

Gesamtheit aller Inputs wird folglich die Diskrepanz des generativen Verhaltens zweier durch $\mathbf{p}$ transformierter Modelle berechnet. Sind sich die Modelle ähnlich genug, so hängt diese Diskrepanz nur von der angewendeten Transformation ab. Da im Rahmen der Registrierung mit ähnlichen, überlappenden Beobachtungen gearbeitet wird, bietet sich D_2 als Metrik zur Bestimmung der unbekannten relativen Transformation zweier Normal Distributions Transforms $\mathcal{M}$ und $\mathcal{S}$ durch geeignete Lösungsverfahren an. Da Gleichung (4.8) in dieser Form nicht direkt berechnet werden kann, muss jedoch zuerst eine Diskretisierung erfolgen.

4.2.1.1 Diskretisierung der Berechnung

Zur Vereinfachung der Berechnung bedienen sich Stoyanov et al. der quadratischen Entropie von Renyi, durch die die quadratischen Terme der ausmultiplizierten Distanzfunktion in (4.9) als konstant über allen Transformationen angenommen werden können [Stoyanov et al., 2012]. Da diese Terme bei der Bewertung einer Transformation $\mathbf{p}$ lediglich einen konstanten Versatz C der Distanzmetrik darstellen, kann die transformationsabhängige L2-Distanz vereinfacht ausgedrückt werden:

$$\begin{aligned} D_2(\mathcal{M}, \mathcal{S}, \mathbf{p}) &= -2 \int p(\mathbf{x}|\mathcal{M})\, p(\mathbf{x}|T(\mathcal{S}, \mathbf{p}))\, d\mathbf{x} + C \\ &= -2 \int p(\mathbf{x}|\mathcal{M})\, p(\mathbf{x}|T(\mathcal{S}, \mathbf{p}))\, d\mathbf{x} \end{aligned} \tag{4.10}$$

Da die im Normal Distributions Transform verwendeten Wahrscheinlichkeitsdichten normalverteilt sind, kann durch die folgende Identität eine Diskretisierung der Berechnungsvorschrift vorgenommen werden:

$$\int N(\mathbf{x}|\mu_i, \Sigma_i)\, N(\mathbf{x}|\mu_j, \Sigma_j)\, d\mathbf{x} = N(0|\mu_i - \mu_j, \Sigma_i + \Sigma_j) \tag{4.11}$$

Die Berechnung ist folglich unabhängig von der unendlichen Menge aller Inputs und beschränkt sich stattdessen auf die Kreuzkorrelation aller Komponenten $N_i \in \mathcal{S}$ und $N_j \in \mathcal{M}$ der Normal Distributions Transforms für die Transformation $\mathbf{p}$:

$$D_2(\mathcal{M}, \mathcal{S}, \mathbf{p}) = \sum_{N_i \in \mathcal{S}} \sum_{N_j \in \mathcal{M}} N(0|T(\mu_i, \mathbf{p}) - \mu_j, T(\Sigma_i, \mathbf{p}) + \Sigma_j) \tag{4.12}$$

Stoyanov und Magnusson beschränken die Kreuzkorrelation in Gleichung (4.12) auf die

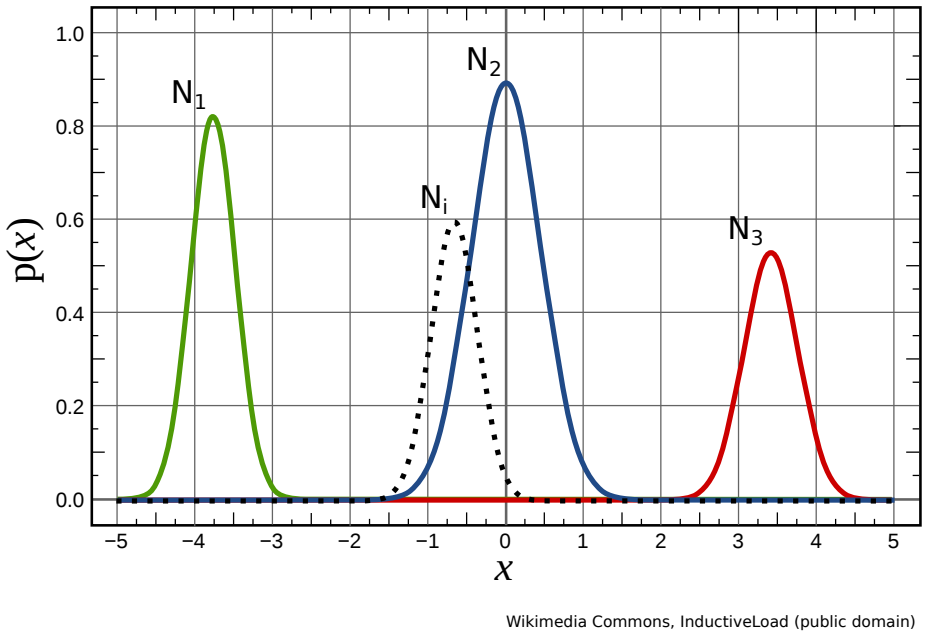


Abbildung 4.1: Veranschaulichung der Distanzberechnung auf Basis der besten Korrespondenz statt Berechnung aller Kreuzkorrespondenzen. Durch die inhärent räumlich disjunkte Berechnung der Normalverteilungen geht $\int p\left(\mathbf{x}|N_i\right) p\left(\mathbf{x}|N_j\right) d\mathbf{x}$ aus Gleichung (4.10) für alle nicht-optimalen Korrespondenzen $N_i \rightarrow N_j$ gegen 0. Nur für die beste Korrespondenz (hier $N_i \rightarrow N_2$) ist diese Berechnung signifikant ungleich 0.

Summation der besten Korrespondenzen [Magnusson, 2009][Stoyanov, 2012]. Diese Approximation ist ausreichend, da die normalverteilten Komponenten für räumlich disjunkte Punktemengen berechnet wurden, und somit für eine Komponente $N_i \in \mathcal{S}$ alle Terme bis auf die beste Korrespondenz $N_j \in \mathcal{M}$ gegen 0 gehen (siehe Abbildung 4.1 für eine Veranschaulichung im eindimensionalen Raum). Um den Berechnungsaufwand zu verringern, ist es also nötig, die Korrespondenzen der Komponenten zweier Normal Distributions Transforms zu bestimmen.

4.3 Korrespondenzetablierung

Eine möglichst exakte Berechnung der Gleichung (4.10) würde zur Diskretisierung des Integrals die Kreuzkorrelation aller Komponenten beider Normal Distributions

Transforms berechnen. Da dies jedoch zu quadratischer Berechnungskomplexität bezüglich der Anzahl der NDT-Komponenten führen würde, wird die Berechnung in Gleichung (4.12), wie bereits erwähnt, auf die besten Korrespondenzen beschränkt. Für jede NDT-Komponente $N_i \in \mathcal{S}$ muss folglich bestimmt werden, welche Komponente $N_j \in \mathcal{M}$ am wahrscheinlichsten dieselben Umgebungsmerkmale abbildet.

4.3.1 Euklidische Distanz

Eine einfache Methode zur Korrespondenzetablierung ist das Minimieren der euklidischen Norm zweier Datenpunkte $\mathbf{x} \in \mathbb{R}^n$ und $\mathbf{y} \in \mathbb{R}^n$:

$$d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2 = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{4.13}$$

Diese Metrik wird beispielsweise im ICP-Algorithmus genutzt, um die korrespondierenden Datenpunkte zweier zu registrierender Punktewolken zu ermitteln. Dabei wird davon ausgegangen, dass die Beobachtungen bereits relativ gut übereinstimmen und die unbekannte relative Transformation der Punktewolken gering ist. Unter dieser Bedingung wird angenommen, dass zwei geometrisch nächste Punkte $\mathbf{y} \in \mathcal{M}$ und $\mathbf{x} \in \mathcal{S}$ denselben reellen Datenpunkt beschreiben, und der geometrische Versatz somit nur der unbekannten Transformation geschuldet ist. Eine Korrespondenzfunktion auf Basis der euklidischen Distanz lässt sich also wie folgt beschreiben:

$$k(\mathbf{x}) = \underset{\mathbf{y}}{\operatorname{argmin}}\, d(\mathbf{x}, \mathbf{y}) \tag{4.14}$$

Die Korrespondenzfunktion (4.14) kann für die in dieser Arbeit verwendeten Normalverteilungen $N(\mu, \Sigma)$ eingesetzt werden, wenn nur das erste Moment $\mu \in \mathbb{R}^3$ betrachtet wird. In diesem Fall wird jede Komponente des Normal Distributions Transforms als Datenpunkt in $\mathbb{R}^3$ betrachtet, dessen Wert die gemittelte Position der Punkteverteilung der zugehörigen Zelle darstellt.

Offensichtlich wird hierbei das zweite Moment Σ der Normalverteilungen vernachlässigt, welches Aufschluss über die dreidimensionale Verteilung der Punkte in der entsprechenden Zelle gibt. Es können folglich korrespondierende Komponenten gefunden werden, die trotz geringem Abstand der Mittelwerte deutlich verschiedene Punkteverteilungen

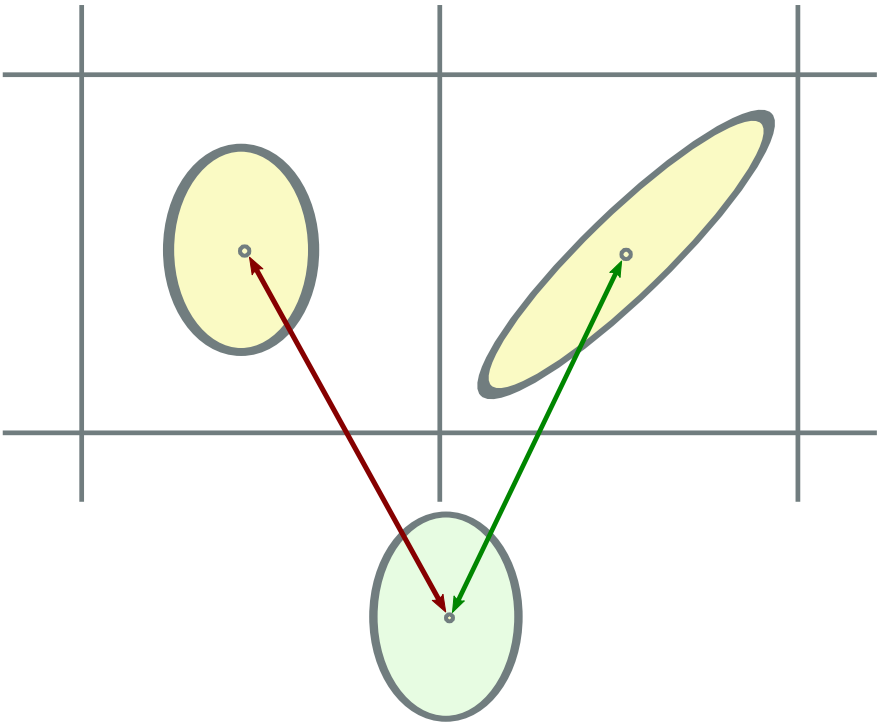


Abbildung 4.2: Illustration der Korrespondenzetablierung auf Basis des Mittelwerts der Normalverteilungen. Obwohl die Normalverteilung der rechten Zelle eine völlig andere Punkteverteilung modelliert, wird diese aufgrund des geringfügig geringeren Abstands der Mittelwerte als bessere Korrespondenz bestimmt.

modellieren. Abbildung 4.2 zeigt ein Beispiel für eine derartige, schlechte Korrespondenz. Aufgrund der geringeren euklidischen Distanz der Mittelwerte wird eine deutlich anders verteilte Komponente als Korrespondenz etabliert, obwohl eine sehr ähnlich verteilte Komponente nur minimal weiter entfernt wäre.

Um bessere Korrespondenzen zu ermitteln, müsste folglich die Ähnlichkeit der zweiten Momente Σ in der Distanzberechnung berücksichtigt werden. Statt der euklidischen Distanz könnte hierfür beispielsweise die erweiterte Mahalanobisdistanz verwendet werden (siehe Abschnitt A.3). Dies würde jedoch durch die hohe Anzahl notwendiger Matrizeninversionen einen erheblich höheren Berechnungsaufwand bedeuten. Aufgrund der effizienten Berechnung und der Tatsache, dass das zweite Moment in der Distanzfunktion (4.12) berücksichtigt wird, wird in dieser Arbeit wie auch bei Stoyanov die euklidische Distanz zur Etablierung der approximierend besten Korrespondenzen

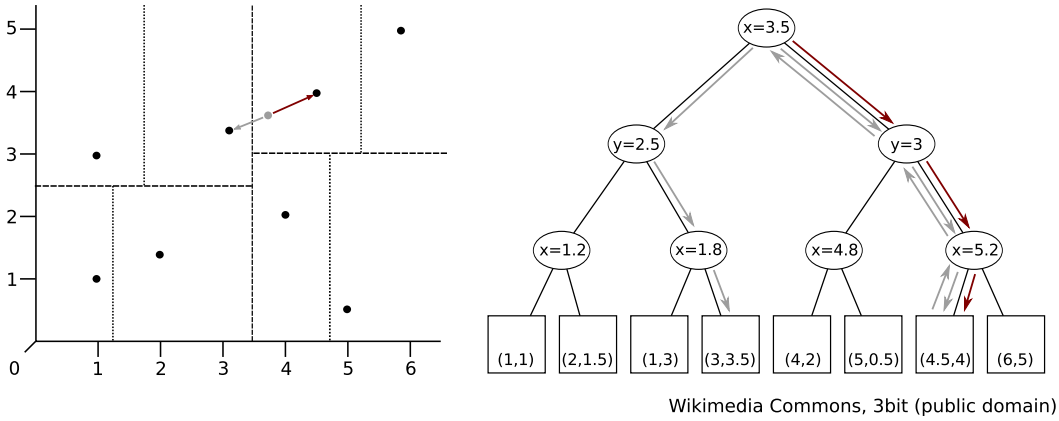


Abbildung 4.3: Veranschaulichung der Partitionierung (links) und Baumstruktur (rechts) eines k-d-Baumes, der für einen zweidimensionalen Datensatz konstruiert wurde. Für den grauen Punkt $(3.7, 3.5)$ wird der nächste Nachbar gesucht. In der Baumstruktur ist der Suchweg der exakten Nächste-Nachbarn-Suche grau gekennzeichnet, während der Suchweg einer relativ toleranten approximierenden Nächste-Nachbarn-Suche in rot dargestellt wird.

verwendet [Stoyanov, 2012].

4.3.2 Effiziente Korrespondenzetablierung

Um die approximiert beste Korrespondenz $k\left(N_i \in \mathcal{S}\right)$ nach Gleichung (4.14) zu finden, muss die euklidische Distanz zwischen N_i und jeder einzelnen Komponente $N_j \in \mathcal{M}$ bestimmt werden. Diese triviale Korrespondenzetablierung hat folglich eine Komplexität von $O\left(n\right)$ mit n gleich Anzahl der NDT-Komponenten in $\mathcal{M}$. Da diese Operation bei der Berechnung der L2-Distanz für jede Komponente in $\mathcal{S}$ durchgeführt wird, bedeutet dies einen signifikanten Berechnungsaufwand. Da die Laufzeit der Registrierung erheblich von den Evaluationen der Gleichung (4.12) bestimmt wird, ist eine effizientere Methode zur Korrespondenzetablierung wünschenswert.

4.3.2.1 Nächste-Nachbarn-Suche mit k-d-Bäumen

Zur effizienten Partitionierung und Durchsuchung von mehrdimensionalen Datensätzen werden häufig k-d-Bäume eingesetzt. Derartige Bäume erweitern das Prinzip binärer

Suchbäume auf den k-dimensionalen Raum und ermöglichen somit eine durchschnittliche Suchkomplexität von $O(\log n)$ mit n gleich Größe des Datensatzes [Bentley, 1975]. Beginnend mit der Wurzel des Baumes wird der Datensatz rekursiv entlang der Dimension mit der größten Varianz geteilt, mit dem Median der Punkteverteilung in dieser Dimension als Pivot. Diese Teilungsstrategie führt in der Regel zu sehr gut balancierten Bäumen, so dass die durchschnittliche Iterationszahl einer Suchoperation tatsächlich im Bereich $O(\log n)$ liegt [Omohundro, 1987]. Diese Partitionierung wird so lange durchgeführt, bis jedes Blatt des Baumes eine Teilmenge des Datensatzes mit maximal m Punkten beinhaltet. Abbildung 4.3 zeigt einen derartig konstruierten k-d-Baum für einen kleinen zweidimensionalen Datensatz.
Um nun den nächsten Datenpunkt $\mathbf{y}$ für einen Datenpunkt $\mathbf{x}$ zu finden, wird beginnend mit der Wurzel durch den k-d-Baum abgestiegen. Anhand der Pivotisierung eines Knotens wird jeweils entschieden, in welcher Partition der beiden Kinderknoten der gesuchte Punkt liegen muss. In einem Blatt angekommen, wird durch euklidische Distanz der nächste Nachbar $\mathbf{y}$ ermittelt. Im Durchschnitt werden hierfür $O(\log n)$ Schritte benötigt.

4.3.2.2 Approximierende Nächste-Nachbarn-Suche

Ein Problem der exakten Nächste-Nachbarn-Suche mit k-d-Bäumen ist die Notwendigkeit des sog. *ball-within-bounds* Tests [Bentley, 1975]. Liegt der Eingabe-Datenpunkt $\mathbf{x}$ näher an einer Partitionierungsgrenze als am nächsten enthaltenen Punkt der aktuellen Partition, so könnte potentiell ein Datenpunkt mit geringerer euklidischer Distanz auf der anderen Seite der Partitionierungsgrenze existieren. Abbildung 4.3 veranschaulicht diese Problematik anhand des grauen Datenpunktes $\mathbf{x} = (3.7, 3.5)$. Obwohl der Punkt $(4.5, 4)$ in die für $\mathbf{x}$ bestimmte Partition fällt, ist der Punkt $(3, 3.5)$ der korrekte nächste Nachbar. Wird eine potentielle derartige Situation durch den ball-within-bounds Test erkannt, so muss der Suchweg durch den k-d-Baum zurückverfolgt werden, um die andere Seite der Partitionierungsgrenze zu erforschen. In Abbildung 4.3 bedeutet dies beispielsweise, dass der Weg bis zur Wurzel zurückverfolgt werden muss.
Gerade in höheren Dimensionen ist ein solches *backtracking* durch die steigende Kan-

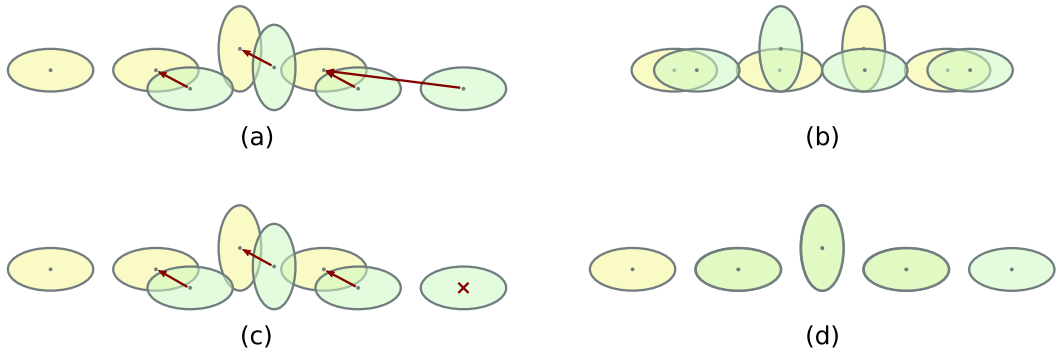


Abbildung 4.4: Veranschaulichung der Korrespondenzetablierung mit Freiräumen. Abbildung (a) zeigt die Nächste-Nachbarn-Korrespondenzetablierung für zwei Normal Distributions Transforms ohne Berücksichtigung der Freiräume, während Abbildung (b) das Ergebnis einer Registrierung mit diesen Korrespondenzen zeigt. Abbildung (c) zeigt hingegen die Korrespondenzetablierung mit Freiräumen, während Abbildung (d) das Ergebnis einer derartigen Registrierung zeigt. Im Vergleich zu Abbildung (a) wird in Abbildung (c) die rechte NDT-Komponente des grünen Normal Distributions Transforms durch Korrespondenz mit einer unbelegten Raumzelle „deaktiviert“, so dass die Distanzmetrik D_2 nicht verzerrt wird.

tenzahl der Partitionen (Hyperwürfel) häufig erforderlich, um das Finden des exakten nächsten Nachbarn zu garantieren. Approximierende Nächste-Nachbarn-Suchen verzichten auf die Bedingung des exakten nächsten Nachbarn, um die Anzahl der Backtracking-Operationen zu verringern und somit die Geschwindigkeit der Suchoperation zu erhöhen [Beis und Lowe, 1997]. Meist kann bei derartigen Verfahren ein maximaler Fehler ϵ zum tatsächlichen nächsten Nachbarn garantiert werden [Arya et al., 1998]. Ein solches Verfahren wird in dieser Arbeit genutzt, um eine schnelle und annähernd genaue Korrespondenzetablierung zu garantieren.

4.3.3 Korrespondenz mit Freiräumen

Wie in Kapitel 3 beschrieben, besteht jeder Normal Distributions Transform aus einer Menge von räumlich disjunkt berechneten Normalverteilungen. Des Weiteren kann für jede Raumzelle der verwendeten Octree-Diskretisierung die Belegtheitswahrscheinlichkeit über die Anzahl der enthaltenen Messpunkte berechnet werden. Mit dieser Informati-

on über etwaige Freiräume der Beobachtungen ist es möglich, die Distanzmetrik D_2 robuster gegen ungenügende Überlappung zu machen. Abbildung 4.4 veranschaulicht das inhärente Problem durch zwangsweise etablierte Korrespondenzen. Die rechte NDT-Komponente der grünen Beobachtung in Abbildung 4.4 (a) wird trotz der hohen Distanz zwangsweise mit der nächsten NDT-Komponente der gelben Beobachtungen in Korrespondenz gesetzt. Dies führt einen unnötigen Fehlerterm in die Distanzmetrik ein, der beispielsweise zu einer fehlerhaften Registrierung wie in Abbildung 4.4 (b) führen könnte.

Um die Berechnung der Distanzmetrik robuster gegen solche Fehler zu machen, werden die erwähnten Belegtheitswahrscheinlichkeiten in die Korrespondenzetablierung integriert. Sind die nächsten Raumzellen $k \in \mathcal{M}$ für eine NDT-Komponente $N_i \in \mathcal{S}$ mit hoher Wahrscheinlichkeit nicht belegt, so wird für N_i keine Korrespondenz etabliert und folglich kein Distanzwert zu (4.12) beigetragen. Dies begründet sich durch die Annahme, dass diese Komponente vermutlich im nicht-überlappenden Teil der Beobachtung liegt, und somit bei einer scharfen Startschätzung der relativen Transformation nur verzerrend auf die Distanzmetrik einwirken würde. Abbildung 4.4 (c) zeigt dieselbe Situation wie Abbildung 4.4 (a), diesmal jedoch unter Berücksichtigung der Freiräume. Durch die „Deaktivierung“ der rechten NDT-Komponente bleibt die Distanzmetrik unverzerrt, so dass eine etwaige Registrierung ein fehlerfreies Ergebnis wie in Abbildung 4.4 (d) erreichen könnte. Die Korrespondenz unter Berücksichtigung von Freiräumen führt folglich zu einer robusteren Distanzmetrik für Beobachtungen mit partieller Überlappung, die im Kontext von SLAM-Algorithmen oft vorliegen.

4.3.4 Beschreibung des gewählten Verfahrens

In dieser Arbeit wird eine approximierende Nächste-Nachbarn-Suche mit k-d-Bäumen verwendet, um die korrespondierenden Komponenten zweier Normal Distributions Transforms anhand der euklidischen Distanz der ersten Momente μ zu bestimmen. Hierzu wird für jede statische Beobachtung $\mathcal{M}$ ein k-d-Baum mit der beschriebenen Methode erzeugt. Da sich die globale Transformation und die Struktur von $\mathcal{M}$ nicht ändern, kann dieser k-d-Baum für alle folgenden Berechnungen zur Korrespondenzetablierung

verwendet werden. Zur Beschleunigung der Suchoperationen wird ein approximierendes Verfahren eingesetzt, das nur eine parametrisierbare Anzahl nach Distanz sortierter Blätter durchsucht und dann den bisher nächsten Nachbarn zurück gibt [Muja und Lowe, 2009]. Durch Erweiterung des k-d-Baumes um die freien Raumzellen des Normal Distributions Transforms kann die Korrespondenzetablierung zudem Komponenten erkennen, die potentiell in nicht-überlappenden Bereichen liegen und somit nachteilig für die Distanzberechnung wären.

4.4 Distanzminimierung

Mit der vorgestellten Methode zur Korrespondenzetablierung ist es möglich, eine relative Transformation $\mathbf{p}$ zweier Normal Distributions Transforms $\mathcal{M}$ und $\mathcal{S}$ durch die L2-Distanz D_2 numerisch zu bewerten. Auf dieser Grundlage kann nun ein Registrierungsalgorithmus zur Minimierung der Distanzfunktion (4.12) bezüglich $\mathbf{p}$ entwickelt werden. Durch die Nichtlinearität der Zielfunktion D_2 ist es allerdings nicht möglich, das Problem in eine analytisch geschlossen lösbare Form zu überführen. Derartige Probleme werden daher häufig als nichtlineare Ausgleichsprobleme formuliert und mit numerischen Optimierungsalgorithmen gelöst, beispielsweise mit dem Gauß-Newton-Verfahren oder dem Levenberg-Marquardt-Algorithmus. Die Konvergenz solcher Verfahren hängt jedoch stark von der Qualität der verwendeten Startschätzung ab, die nicht in allen Fällen vorhanden ist. Fehlt eine akzeptable Startschätzung, so werden meist Metaheuristiken wie beispielsweise die Partikelschwarmoptimierung oder die Simulierte Abkühlung eingesetzt.

4.4.1 Registrierung in SLAM-Algorithmen

In einem SLAM-Algorithmus muss die Registrierung sowohl für scharfe als auch für unscharfe Startschätzungen berechnet werden. Bei der Registrierung zweier aufeinanderfolgender, überlappender Beobachtungen ist durch den Odometrieversatz der Aufnahmeposen eine meist ausreichend scharfe Schätzung der relativen Transformation $\hat{\mathbf{p}}$ vorhanden. Im Falle eines Schleifenschlusses hat der mobile Roboter zwischen zwei

überlappenden Beobachtungen jedoch häufig eine erhebliche Fahrtstrecke zurückgelegt, so dass durch den akkumulierten Odometriefehler nur noch eine sehr unscharfe Schätzung der relativen Transformation $\hat{\mathbf{p}}$ bestimmt werden kann. Diese Schätzung ist meist so fehlerhaft, dass die Konvergenz eines numerischen Lösungsverfahrens nicht garantiert werden kann.
Theoretisch wäre es ausreichend, einen Registrierungsalgorithmus auf Basis einer Metaheuristik zu implementieren, da derartige Verfahren sowohl mit scharfen als auch mit unscharfen Startschätzungen funktionieren. Durch die üblicherweise höhere Berechnungskomplexität und schlechteren Konvergenzeigenschaften würde sich dies jedoch negativ auf das Leistungsverhalten und die Echtzeitfähigkeit des SLAM-Verfahrens auswirken. Im Folgenden soll daher sowohl ein numerischer Algorithmus, als auch ein metaheuristischer Algorithmus zur Registrierung entwickelt werden.

4.4.2 Registrierung mit scharfer Startschätzung

In diesem Abschnitt wird ein Registrierungsalgorithmus vorgestellt, der von einer scharfen Startschätzung $\hat{\mathbf{p}}$ der relativen Transformation der beiden Normal Distributions Transforms $\mathcal{M}$ und $\mathcal{S}$ ausgeht. Das Problem der Minimierung der L2-Distanz (4.12) wird dabei als nichtlineares Ausgleichsproblem betrachtet und durch den Levenberg-Marquardt-Algorithmus gelöst. Ein ähnlicher Ansatz wurde in [Ulaş und Temeltaş, 2012] verfolgt, jedoch wurde dort die Registrierung einer Punktewolke mit einem Normal Distributions Transform beschrieben. Dennoch bestätigen die Ergebnisse von Ulaş und Temeltaş die Eignung des Levenberg-Marquardt-Algorithmus' zur Minimierung einer geeigneten Distanzfunktion.
Ausgleichsprobleme suchen anhand einer Menge von m Eingaben x_i und zugehörigen Messwerten y_i nach dem unbekannten Parametervektor $\mathbf{p}$ einer Modellfunktion $f(x, \mathbf{p})$, mit dem die Messwerte bestmöglich durch die Modellfunktion beschrieben werden können. Häufig werden derartige Probleme mit der Methode der kleinsten Quadrate gelöst, die durch schrittweises Anpassen einer Schätzung $\bar{\mathbf{p}}$ den quadratischen Fehler zwischen Messwerten und Vorhersage der Modellfunktion minimiert:

$$\min_{\bar{\mathbf{p}}} \sum_{i=1}^{m} \left(f\left(x_i, \bar{\mathbf{p}}\right) - y_i\right)^2 \tag{4.15}$$

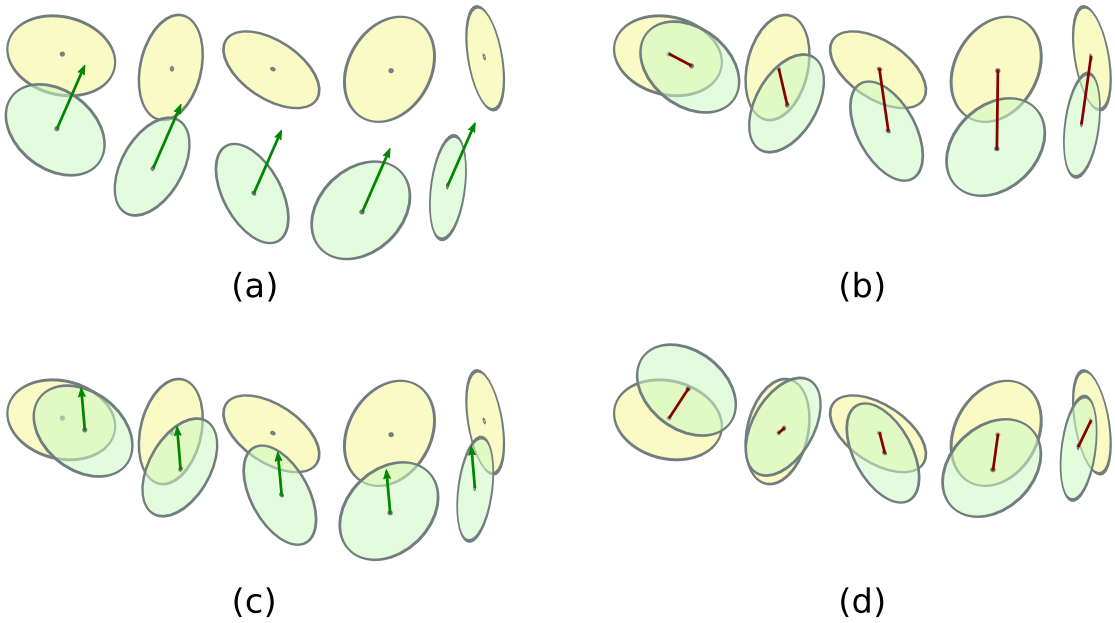


Abbildung 4.5: Veranschaulichung der Registrierung zweier Normal Distributions Transforms als Ausgleichsproblem. Abbildung (a) zeigt die initial geschätzte relative Transformation als grüne Pfeile. Die Anwendung dieser Transformation wird in Abbildung (b) zusammen mit den in rot angedeuteten Residuen gezeigt. Durch die Linearisierung der Zielfunktion D_2 ergibt sich der in Abbildung (c) vorgestellte Transformationsschritt. Abbildung (d) deutet wiederum die Residuen dieser Zwischenlösung an. Ein nächster, hier nicht dargestellter Schritt würde vermutlich eine leichte Rotation gegen den Uhrzeigersinn vornehmen.

Die Differenz zwischen Modellvorhersage und Messwert wird Residuum genannt. Um typische nichtlineare Optimierungsalgorithmen wie den Levenberg-Marquardt-Algorithmus auf die Registrierung zweier Normal Distributions Transforms anzuwenden, muss dieses Problem zunächst als Ausgleichsproblem formuliert werden.

4.4.2.1 Registrierung als Ausgleichsproblem

Da das Ziel der Registrierung die Bestimmung der relativen Transformation $\mathbf{p}$ zwischen $\mathcal{M}$ und $\mathcal{S}$ ist, kann die Transformationsfunktion $\mathcal{M} = T(\mathcal{S}, \mathbf{p})$ als Modellfunktion $f(x, \mathbf{p})$ eines Ausgleichsproblems betrachtet werden. Die Transformation $T(N_i, \mathbf{p})$ einer NDT-Komponente $N_i \in \mathcal{S}$ beschreibt daher die Vorhersage der Modellfunktion für eine

Parametrisierung $\mathbf{p}$, während die gefundene Korrespondenz $N_j \in \mathcal{M}$ den zugehörigen Messwert beschreibt. Die zu minimierende Gleichung ist folglich analog zu (4.15):

$$\min_{\mathbf{p}} \sum_{N_i \in \mathcal{S}} \left(T\left(N_i, \mathbf{p}\right) - N_j\right)^2 \tag{4.16}$$

Um die Residuen in Gleichung (4.16) zu berechnen, wird die erweiterte Mahalanobisdistanz (siehe A.3) der Komponenten $T\left(N_i, \mathbf{p}\right)$ und N_j verwendet:

$$\left(T\left(N_i, \mathbf{p}\right) - N_j\right) \equiv \sqrt{\left(T\left(\mu_i, \mathbf{p}\right) - \mu_j\right)^T \left(T\left(\Sigma_i, \mathbf{p}\right) + \Sigma_j\right)^{-1} \left(T\left(\mu_i, \mathbf{p}\right) - \mu_j\right)} \tag{4.17}$$

Um (4.16) nach $\mathbf{p}$ zu minimieren, werden ausgehend von einem Startwert $\mathbf{p}_0$ kleine Schritte gewählt, die jeweils zu einem geringeren summierten quadratischen Fehler führen. Abbildung 4.5 zeigt diesen Vorgang am zweidimensionalen Beispiel zweier Normal Distributions Transforms. In dieser Arbeit soll für diese Minimierung der Levenberg-Marquardt-Algorithmus eingesetzt werden.

4.4.2.2 Minimierung mit dem Levenberg-Marquardt-Algorithmus

Der Levenberg-Marquardt-Algorithmus ist ein robustes numerisches Optimierungsverfahren zur Lösung nichtlinearer Ausgleichsprobleme (siehe Abschnitt A.1 für eine nähere Beschreibung). Algorithmus 2 beschreibt die Minimierung des Ausgleichsproblems (4.16) mit diesem Verfahren. Für jeden Zeitschritt wird die Jacobimatrix der Transformationsfunktion für die einzelnen Komponenten des transformierten Normal Distributions Transforms $\mathcal{S}$ berechnet. Die Residuen ergeben sich als erweiterte Mahalanobisdistanz zwischen den transformierten Komponenten $N_i \in \mathcal{S}$ und ihren Korrespondenzen $N_j \in \mathcal{M}$ (siehe Gleichung (4.17)). Mit dieser stückweisen Jacobimatrix J_t und dem Residuenvektor $\mathbf{r}_t$ wird nun die Schrittweite $\Delta\left(\mathbf{p}\right)_t$ des aktuellen Zeitschritts berechnet und zur aktuellen Schätzung der relativen Transformation $\mathbf{p}_t$ addiert. Nach einer festen Anzahl an Schritten oder dem Unterschreiten eines Grenzwertes wird der Algorithmus beendet und der Parametervektor $\mathbf{p}_t$ als Schätzung der relativen Transformation ausgegeben. Der Levenberg-Marquardt-Algorithmus ermöglicht folglich die Registrierung zweier Normal Distributions Transforms, falls eine akzeptable Startschätzung vorliegt.

Input : Normal Distributions Transforms $\mathcal{M}$ und $\mathcal{S}$, Startschätzung $\mathbf{p}_0$

Result : Geschätzter Parametervektor $\mathbf{p}_t$

for *Iterationsschritt* t **do**

`// Berechne Jacobimatrix` J_t `und Residuenvektor` $\mathbf{r_t}$;

for *NDT-Komponenten* $N_i \in \mathcal{S}$ **do**

Bestimme Korrespondenz $N_j \in \mathcal{M}$ für $T(N_i, \mathbf{p}_t)$;

Berechne Residuum $r_t^i = d(N_i, N_j, \mathbf{p}_t)$ mit (4.17);

Berechne Gradient $J_t^i = \dfrac{\partial T(N_i, \mathbf{p}_t)}{\partial \mathbf{p}_t}$;

end

`// Aktualisierung der Schätzung mit` (A.4);

$\Delta(\mathbf{p})_t = -\left(J_t^T J_t + \lambda I\right)^{-1} J_t^T \mathbf{r}_t$;

$\mathbf{p}_{t+1} = \mathbf{p}_t + \Delta(\mathbf{p})_t$;

`// Ablaufsteuerung des Algorithmus'`;

Vergrößere ($\mathbf{r}_t \geq \mathbf{r}_{t-1}$) oder verkleinere ($\mathbf{r}_t < \mathbf{r}_{t-1}$) Regulierungsfaktor λ;

Falls zutreffend, ziehe schlechten Schritt zurück: $\mathbf{p}_{t+1} = \mathbf{p}_t$;

Überprüfe Faktoren t, λ und $\Delta(\mathbf{p})_t$ auf Abbruchkriterien;

end

Algorithmus 2: Registrierung mit dem Levenberg-Marquardt-Algorithmus.

4.4.3 Registrierung mit unscharfer Startschätzung

Wie bereits erwähnt, hängt das Konvergenzverhalten und die Robustheit numerischer Lösungsverfahren stark von einer geeigneten Startschätzung ab. Im Falle eines Schleifenschlusses lässt sich durch die Odometrie zwar eine relative Transformation zweier überlappender Beobachtungen ermitteln, die hohe Unsicherheit dieser Schätzung weist jedoch auf die vermutlich große Abweichung von der tatsächlichen relativen Transformation hin. Für derartige Startwerte ist das Fehlergebirge der Distanzfunktion meist extrem schlecht bestimmt (siehe äußerer Bereich der Abbildung 4.6). In diesen Fällen würde der vorgestellte Levenberg-Marquardt-Algorithmus mit hoher Wahrscheinlichkeit

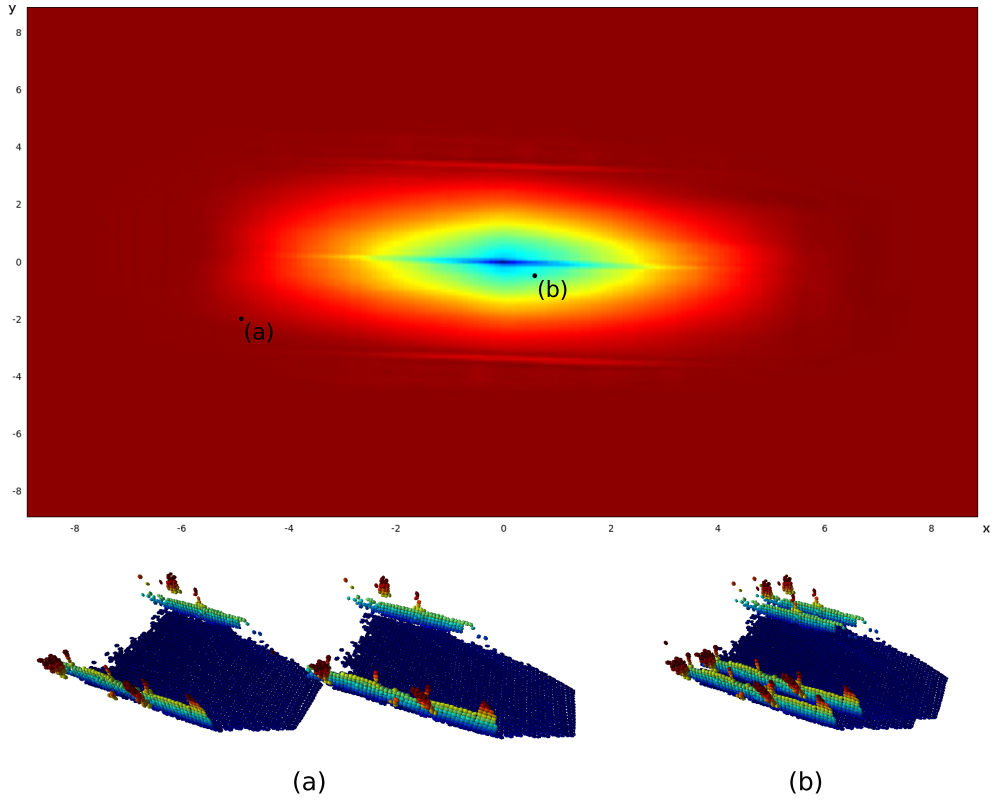


Abbildung 4.6: Veranschaulichung des x-y-Schnitts des Fehlergebirges der L2-Distanz für zwei identische Normal Distributions Transforms eines Korridors. In der verwendeten Jet-Palette werden niedrige Distanzwerte blau gezeichnet, hohe Distanzwerte dagegen rot. Hier ist gut zu sehen, dass das Fehlergebirge der Distanzfunktion außerhalb eines akzeptablen Parameterbereichs komplett homogen ist. Linearisierungs-basierte Verfahren wie der Levenberg-Marquardt-Algorithmus würden für derartige Parameterbereiche mit hoher Wahrscheinlichkeit divergieren. Ein metaheuristisches Verfahren könnte den Parameterraum dagegen so lange strategisch explorieren, bis eine Verbesserung der Distanzwerte erkennbar ist.

divergieren. Aus diesem Grund wird für die Registrierung mit unscharfer Startschätzung in dieser Arbeit die sog. Partikelschwarmoptimierung eingesetzt.

4.4.3.1 Partikelschwarmoptimierung

Die Partikelschwarmoptimierung wurde 1995 von Kennedy und Eberhardt als metaheuristisches Optimierungsverfahren entwickelt [Kennedy und Eberhart, 1995]. Das Verfahren orientiert sich an der Bewegungsdynamik von Tierschwärmen, die als soziales Gefüge nach Ressourcen suchen. Eine Population von Individuen, die durch ihre Position jeweils eine Hypothese im Suchraum beschreiben, werden zufällig in einem festgelegten Teilraum verteilt und mit einer zufälligen Geschwindigkeit und Bewegungsrichtung initialisiert. Jedes Individuum exploriert den Suchraum selbstständig, wird durch sein gieriges Verhalten jedoch stets zu der besten bisher gefundenen Position des Schwarms und der besten bisher selbst gefundenen Position gezogen. Im Laufe der Zeit konvergiert ein Großteil der Population auf die Position, die den besten bisher bekannten Ressourcenwert bietet. Für eine nähere Beschreibung der Partikelschwarmoptimierung siehe Abschnitt A.2.

4.4.3.2 Metaheuristische Registrierung

Mathematisch gesehen stellt die Partikelschwarmoptimierung (PSO) einen metaheuristischen Algorithmus zur Optimierung nichtlinearer Funktionen dar, der den zugehörigen Suchraum durch semi-deterministische Abtastung exploriert. Bei der Registrierung zweier Normal Distributions Transforms würde folglich der Parameterraum $\mathbf{p} \in \mathbb{R}^6$ der Transformationsfunktion $T(\mathbf{x}, \mathbf{p})$ durchsucht. Da ein Explorieren des kompletten Parameterraums weder möglich noch praktikabel ist, wird der zu durchsuchende Teilraum bei der Registrierung mit Partikelschwarmoptimierung durch die unsichere, von der Odometrie bestimmte relative Transformation $\hat{\mathbf{p}}$ der Aufnahmeposen ermittelt. Durch die Unsicherheit $\hat{\Sigma}$ dieser Transformation kann der Bereich abgeschätzt werden, in dem die tatsächliche Transformation liegen muss. Um die Partikelpopulation in diesem Teilraum zu verteilen, kann schlicht für jedes Partikel aus der Normalverteilung $N\left(\hat{\mathbf{p}}, \hat{\Sigma}\right)$ gezogen werden (siehe Abbildung 4.7).

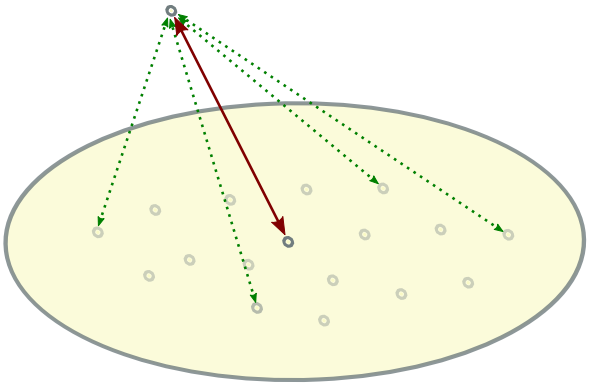


Abbildung 4.7: Veranschaulichung der Bestimmung des Suchraums für die Partikelschwarmoptimierung. Der rote Pfeil steht für die relative Transformation $\hat{\mathbf{p}}$, die durch die Odometrie zwischen den dunkelgrauen Posen bestimmt wurde. Die Unsicherheit dieser Transformation wird als 3σ-Fehlerellipse der zugehörigen Kovarianzmatrix $\hat{\Sigma}$ abgebildet. Die grünen Pfeile deuten die Transformationen an, die durch eine Auswahl in diesem Bereich initialisierter Individuen repräsentiert werden.

Eine so initialisierte Partikelpopulation führt nun iterativ die Bewegungsdynamik des Schwarms durch (siehe Abschnitt A.2 für Details), um die Distanzfunktion D_2 bezüglich $\mathbf{p}$ zu minimieren. In dieser Arbeit werden zwei Modifikationen dieser Bewegungsdynamik genutzt, um sowohl das Konvergenzverhalten als auch die Konvergenzgeschwindigkeit zu verbessern. Die Erweiterung *Euclidean PSO* entdeckt eine Stagnation des Bewegungsverhaltens der Partikelpopulation und zerstreut diese daraufhin in der lokalen Nachbarschaft des besten Partikels [Zhu et al., 2009]. Des Weiteren ist in dieser Arbeit die erste Ableitung der Zielfunktion bekannt (siehe vorheriger Abschnitt), so dass jedes Partikel zudem explizit in Richtung des Gradientenabstiegs gelenkt werden kann.
Algorithmus 3 beschreibt die Registrierung zweier Normal Distributions Transforms $\mathcal{M}$ und $\mathcal{S}$ durch Partikelschwarmoptimierung für eine unscharfe Schätzung der relativen Transformation $\hat{\mathbf{p}}$ und die zugehörige Unsicherheit $\hat{\Sigma}$. Ein Zeitschritt dieses Algorithmus' wird in Abbildung 4.8 veranschaulicht. Durch Partikelschwarmoptimierung ist es folglich möglich, die relative Transformation zweier Beobachtungen mit unscharfer Startschätzung zu bestimmen.

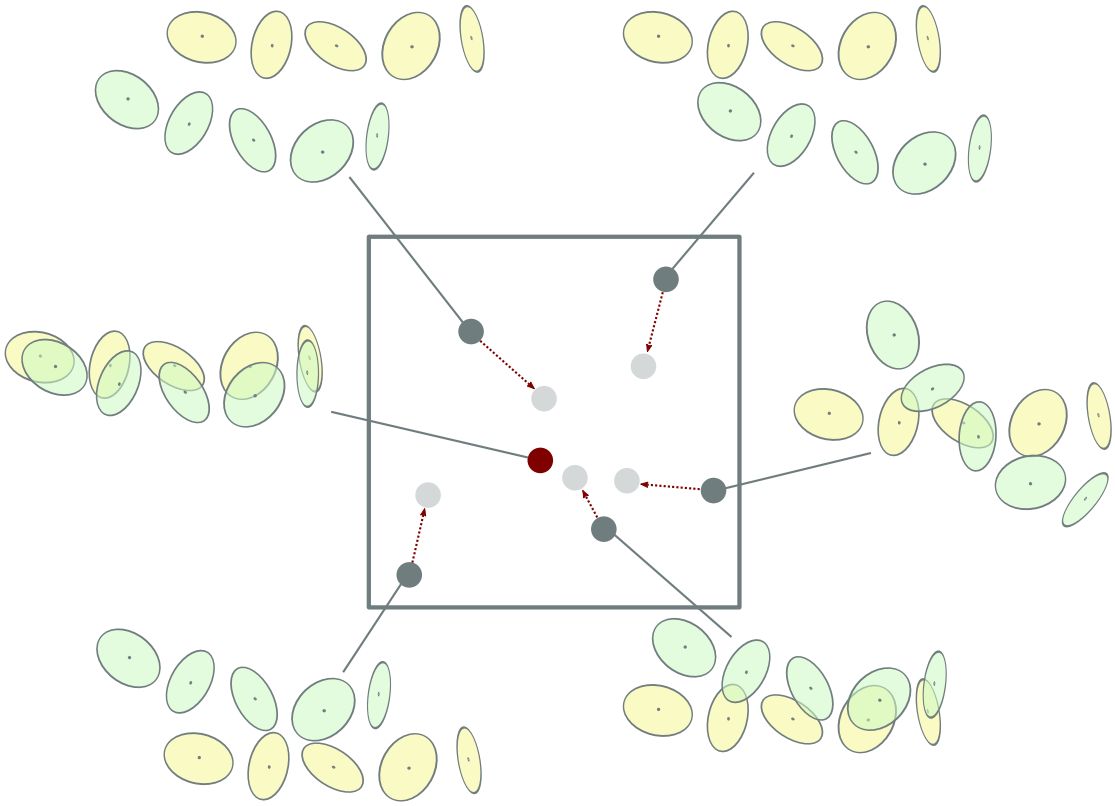


Abbildung 4.8: Veranschaulichung der Registrierung zweier Normal Distributions Transforms durch Partikelschwarmoptimierung im zweidimensionalen Raum. Der dreidimensionale Suchraum der Transformationen $(x, y, \phi) \in \mathbb{R}^3$ wird hier aus Gründen der Übersichtlichkeit als zweidimensional dargestellt. Das beste Partikel wird in dunkelrot dargestellt, während die restliche Population dunkelgrau dargestellt wird. Die Bewegungsdynamik der Partikeln wird hier vereinfacht durch einen roten Pfeil dargestellt, der auf die hellgrau angedeutete Position im nächsten Zeitschritt zeigt. In jedem Zeitschritt explorieren die Partikeln den Suchraum und konvergieren dabei auf die momentan beste gefundene Position (x^*, y^*, ϕ^*). Eine genauere Beschreibung der Bewegungsdynamik und der Funktionsweise der Partikelschwarmoptimierung findet sich in Kapitel A.2.

Input : Normal Distributions Transforms $\mathcal{M}$ und $\mathcal{S}$, Startschätzung $\hat{\mathbf{p}}$ mit zugehöriger Unsicherheit $\hat{\Sigma}$

Result : Geschätzter Parametervektor $\mathbf{p}_t$

```
// Initialisiere Partikelpopulation im Suchraum;
```

for *Alle Partikeln i* **do**

  Ziehe Position $\mathbf{x}_0^i$ des Partikels i: $\mathbf{x}_0^i \sim N\left(\hat{\mathbf{p}}, \hat{\Sigma}\right)$;

  Initialisiere Geschwindigkeit $\mathbf{v}_0^i$ zufällig;

end

```
// Iteration der Bewegungsdynamik;
```

for *Iterationsschritt t* **do**

  `// Bewegungsdynamik der Partikeln;`

  for *Alle Partikeln i* **do**

    Berechne neue Geschwindigkeit $\mathbf{v}_t^i$ nach Gleichung (A.8);

    Berücksichtige Krümmung des Fehlergebirges: $\mathbf{v}_t^i = \mathbf{v}_t^i + \dfrac{\partial T\left(\mathcal{S}, \mathbf{x}_{t-1}^i\right)}{\partial \mathbf{x}_{t-1}^i}$;

    Berechne neue Position: $\mathbf{x}_t^i = \mathbf{x}_{t-1}^i + \mathbf{v}_t^i$;

    Propagiere entdeckten Wert $D_2\left(\mathcal{M}, \mathcal{S}, \mathbf{x}_t^i\right)$ an den Schwarm;

  end

  `// Stagnationsvermeidung durch Euclidean PSO;`

  if *Bewegungsdynamik stagniert* **then**

    Zerstreue Partikeln in der Nachbarschaft des besten Partikels;

  end

  `// Ablaufsteuerung des Algorithmus';`

  Überprüfe t und gewisse Schwarmeigenschaften auf Abbruchkriterien;

end

```
// Ausgabe der bestimmten relativen Transformation;
```

Extrahiere Ergebnis $\mathbf{p}_t$ aus bester bisher gefundener Position;

Algorithmus 3: Registrierung mit Partikelschwarmoptimierung.

4.5 Schätzung der Unsicherheit

Um die durch die Registrierung ermittelte relative Transformation zweier Beobachtungen und somit ihrer Aufnahmeposen in ein probabilistisches SLAM-Framework einzubetten, wird eine Abschätzung der Unsicherheit dieser Information benötigt. Weder der Levenberg-Marquardt-Algorithmus noch die Partikelschwarmoptimierung garantieren, dass bei der Registrierung das globale Minimum gefunden wird. Aus diesen Gründen ist es nötig, ein Maß für die Genauigkeit bzw. Unsicherheit des Registrierungsergebnisses zu finden.
Betrachtet man das Ergebnis der Registrierung als Erwartungswert einer Zufallsgröße $\mathcal{R}$, so kann die Unsicherheit als Varianz der zugrundeliegenden Wahrscheinlichkeitsverteilung ausgedrückt werden [Stoyanov, 2012]. Um die ermittelte Schätzung in das probabilistische Framework integrieren zu können, wird diese Zufallsgröße hier als normalverteilt angenommen. Im Fall der Registrierung wird die ermittelte relative Transformation $\mathbf{p}$ als Mittelwert μ einer unbekannten Normalverteilung interpretiert, deren Dichte für Werte $\mathbf{x} \in \mathbb{R}^6$ durch die Distanzfunktion $D_2(\mathcal{M}, \mathcal{S}, \mathbf{x})$ berechnet werden kann. Offensichtlich lässt sich die zugehörige Kovarianzmatrix Σ nicht direkt aus Formel (4.12) herleiten, so dass alternative Berechnungsmethoden nötig sind. In diesem Abschnitt sollen daher zwei Methoden vorgestellt und evaluiert werden, die zur Schätzung der Kovarianzmatrix Σ des Registrierungsergebnisses $\mathbf{p}$ verwendet werden können.

4.5.1 Stichprobenvarianz

Eine statistisch motivierte Methode zur Schätzung dieser Kovarianz ist die gewichtete Stichprobenvarianz. Ziel ist die Schätzung der Varianz einer Zufallsgröße, für die ohne genaue Kenntnis der Verteilungsfunktion Stichproben erzeugt werden können. Durch eine Menge gewichteter Stichproben soll in diesem Verfahren der Parameter Σ der zugrundeliegenden Normalverteilung approximiert werden. Da während der Registrierung mit Partikelschwarmoptimierung eine Vielzahl solcher gewichteter Stichproben ermittelt wird, bietet sich die Schätzung der Unsicherheit durch gewichtete Stichprobenvarianz

für dieses Verfahren an.
Aus der Verteilungsfunktion werden n Stichproben $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ um den Mittelwert μ gezogen; im Rahmen der Registrierung entspricht dies der Bestimmung relativer Transformationen $\mathbf{x} \in \mathbb{R}^6$ in der Nachbarschaft des Registrierungsresultats $\mathbf{p}$. Für jede Stichprobe $\mathbf{x}_i$ ist auch die Dichte, also die Distanz der transformierten Normal Distributions Transforms bekannt:

$$w_i = D_2\left(\mathcal{M}, \mathcal{S}, \mathbf{x}_i\right) \tag{4.18}$$

Diese Dichte wird als Gewichtung w_i der Stichprobe $\mathbf{x}_i$ betrachtet. Im Folgenden werden diese Gewichte als normiert über allen Stichproben angenommen:

$$\sum_{i=1}^{n} w_i = 1 \tag{4.19}$$

Die Integration der Distanzfunktion D_2 zur Gewichtung ermöglicht die Berücksichtigung der charakteristischen Krümmung der Zielfunktion bei der statistischen Berechnung der Momente der Verteilungsfunktion.

4.5.1.1 Mittelwert

Durch das Ergebnis der Registrierung ist der Mittelwert $\mu_{\text{reg}} = \mathbf{p}$ der Normalverteilung bereits bekannt. Es ist wichtig zu beachten, dass dies nicht zwangsweise dem gewichteten statistischen Mittelwert entspricht:

$$\mu = \sum_{i=1}^{n} w_i \mathbf{x_i} \tag{4.20}$$

Der Grund liegt in einer Kombination aus der Struktur des Problems und der Natur der Stichprobenvarianz. Bei der Registrierung wurde nicht der komplette Zustandsraum betrachtet, sondern eine selektive, ungleichmäßige Unterabtastung vorgenommen, die letztendlich zum Registrierungsergebnis führte. Durch das Ziehen von Stichproben wird der lokale Unterraum jedoch regelmäßig und möglichst vollständig abgetastet, so dass dies zwei unterschiedliche Ausgangspositionen sind. Da die Kovarianz aber für das Registrierungsergebnis $\mathbf{p}$ berechnet werden soll, wird dieses im Folgenden als Mittelwert μ_{reg} betrachtet.

4.5.1.2 Kovarianz

Bei Kenntnis des Mittelwerts lässt sich über allen Stichproben die Kovarianz der Zufallsgröße schätzen. Im Fall der Registrierung ist diese Zufallsgröße eine Funktion des sechsdimensionalen Parametervektors $\mathbf{p} = (t_x, t_y, t_z, \phi_x, \phi_y, \phi_z)$, so dass die Kovarianzmatrix wie folgt berechnet wird [Pozzi et al., 2012]:

$$\Sigma_{\text{reg}} = \begin{pmatrix} \text{cov}\,(t_x, t_x) & \cdots & \text{cov}\,(t_x, \phi_z) \\ \vdots & \ddots & \vdots \\ \text{cov}\,(\phi_z, t_x) & \cdots & \text{cov}\,(\phi_z, \phi_z) \end{pmatrix} \in \mathbb{R}^{(6,6)} \tag{4.21}$$

Für alle Stichproben $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ und deren Gewichte $(w_1, \ldots, w_n)$ berechnet sich die Kovarianz zweier Zufallsvariablen durch gewichtetes Mitteln:

$$\text{cov}\left(x^i, x^j\right) = E[\left(\mathbf{x}^i - \mu^i\right)\left(\mathbf{x}^j - \mu^j\right)] = \sum_{k=1}^{n} w_i^2 \left(x_k^i - \mu^i\right)\left(x_k^j - \mu^j\right) \tag{4.22}$$

Die Schreibweise x_k^i bezeichnet hierbei die i-te Komponente $(1 \leq i \leq 6)$ des sechsdimensionalen Vektors $\mathbf{x}_k = (x_k^1, \ldots, x_k^6)$, der die k-te Stichprobe beschreibt.

4.5.1.3 Evaluation

Die Approximationsgenauigkeit dieses statistischen Schätzers steigt bis zu einem gewissen Punkt proportional zur Anzahl der verwendeten Stichproben, ist jedoch nach oben durch die Eignung der Normalverteilung zur Modellierung der tatsächlichen, unbekannten Wahrscheinlichkeitsverteilung beschränkt. Für ein gutes Leistungsverhalten dieses Schätzers müssen in der Praxis sowohl die Menge der Stichproben als auch deren Auswahlverfahren geeignet bestimmt werden. Da in der Partikelschwarmoptimierung eine strategische Abtastung des Parameterraums bezüglich der Wahrscheinlichkeitsdichte (also D_2) stattfindet, kann die Unsicherheit des Registrierungsergebnisses direkt als Stichprobenvarianz aller abgetasteten Positionen $\mathbf{x}$ berechnet werden.

4.5.2 Analytische Kovarianz

Eine andere Möglichkeit der Kovarianzschätzung ist Linearisierung der Zielfunktion D_2 durch Taylorreihenentwicklung, so dass die Kovarianzmatrix analytisch aus der

Hessematrix dieser Linearisierung geschätzt werden kann [Bengtsson und Baerveldt, 2001]. Die linearisierte Zielfunktion lässt sich vereinfacht durch die folgenden Gleichungen beschreiben:

$$\mathbf{Y} = M\mathbf{X} + \omega \tag{4.23}$$
$$\bar{\mathbf{Y}} = M\bar{\mathbf{X}}$$

Gleichung 4.23 modelliert die linearisierte Systemfunktion über dem Parametervektor $\mathbf{X}$, so dass $\bar{\mathbf{Y}}$ den Funktionswert für einen Parametervektor $\bar{\mathbf{X}}$ beschreibt. Die unbekannte Matrix M modelliert die linearisierte Systemdynamik, während ω für normalverteiltes Rauschen steht. Das Ziel der Registrierung in dieser Form ist die Minimierung des quadratischen Fehlerterms:

$$E\left(\bar{\mathbf{X}}\right) = \left(\mathbf{Y} - M\bar{\mathbf{X}}\right)^T \left(\mathbf{Y} - M\bar{\mathbf{X}}\right) \tag{4.24}$$

Bengtsson und Baerveldt leiten aus diesem Term nun eine Formulierung der optimalen Schätzung für die Kovarianzmatrix $\Sigma\left(\bar{\mathbf{X}}\right)$ ab:

$$\Sigma\left(\bar{\mathbf{X}}\right) = \sigma^2 \left(M^T M\right)^{-1} \tag{4.25}$$

Durch zweimaliges Differenzieren der Gleichung (4.24) erreichen sie es, die Kovarianzmatrix in Gleichung (4.25) ohne Kenntnis der Modellmatrix M zu berechnen:

$$H = \frac{\partial E^2\left(\bar{\mathbf{X}}\right)}{\partial \bar{\mathbf{X}}^2} = 2M^T M \tag{4.26}$$

Das Einsetzen in Gleichung (4.25) ergibt die Kovarianzformulierung auf Basis der Hessematrix der linearisierten Zielfunktion:

$$\Sigma\left(\bar{\mathbf{X}}\right) = \sigma^2 \left(\frac{1}{2}H\right)^{-1} \tag{4.27}$$

Die invertierte Hessematrix dieses Ausgleichsproblems approximiert durch ihre Struktur also die Kovarianzmatrix der Zielfunktion für den aktuellen Parametervektor $\bar{\mathbf{X}}$. Da bei der Registrierung mit dem Levenberg-Marquardt-Algorithmus stets die approximierte Hessematrix für den aktuellen Parametervektor berechnet wird, bietet sich die analytische Schätzung der Unsicherheit des Ergebnisses für die Levenberg-Marquardt-Registrierung an.

4.5.2.1 Skalierung um Varianz

Die Kovarianzmatrix aus Gleichung (4.27) hängt noch von einem Skalierungsfaktor σ^2 ab, der für die Varianz des Datensatzes steht. Diese Skalierung transformiert die Werte der invertierten Hessematrix in die korrekte metrische Größenordnung.
Eine übliche Approximation dieser Varianz ergibt sich nach Bengtsson und Baerveldt aus dem gemittelten quadratischen Fehler [Bengtsson und Baerveldt, 2001]:

$$\sigma^2 = \frac{E\left(\bar{\mathbf{X}}\right)}{n-k} \tag{4.28}$$

Hier steht n für die Anzahl der Datenpunkte $y \in \bar{\mathbf{Y}}$, während k die Dimension des Parameterraums $\bar{\mathbf{X}}$ beschreibt. Der Faktor $(n-k)$ berechnet die Anzahl der Freiheitsgrade, die in dem verwendeten Datensatz vorhanden sind. Gleichung (4.28) berechnet folglich den unverzerrten Mittelwert des quadratischen Fehlers, der als geschätzte Varianz des Datensatzes interpretiert werden kann.

4.5.3 Evaluation

In diesem Abschnitt sollen die beiden vorgestellten Verfahren kurz evaluiert werden. Die Kovarianzschätzung durch Berechnung der gewichteten Stichprobenvarianz bietet eine statistisch fundierte Methode, um die Varianz einer unbekannt verteilten, messbaren Zufallsgröße $\mathcal{R}$ zu berechnen. Für eine große Anzahl n an Stichproben approximiert diese Schätzung die tatsächliche Wahrscheinlichkeitsverteilung, bzw. deren bestmögliche Modellierung durch eine Normalverteilung. Durch die Berechnung der Distanzmetrik D_2 für jede Stichprobe ist diese Modellierung zwar sehr genau, führt jedoch zu einem sehr hohen Berechnungsaufwand. In der Praxis muss daher unter dem Gesichtspunkt der Echtzeittauglichkeit eine Stichprobengröße n^* gefunden werden, die bei ausreichender Genauigkeit noch schnell genug berechnet werden kann.
Die analytische Schätzung der Kovarianz leidet nicht unter dieser Problematik, erfordert dafür jedoch eine zweimalig differenzierbare Zielfunktion, sowie die Berechnung der partiellen Ableitungen zweiter Ordnung. Diese Berechnung ist ebenfalls aufwändig und benötigt spezielles Wissen über die mathematische Form der Zielfunktion. Censi bemerkt zudem, dass die durch die Hessematrix beschriebene Krümmungscharakteristik

der Zielfunktion für den bestimmten Parametervektor $\mathbf{p}$ nicht genügend Informationen enthält, um die Kovarianz des Registrierungsergebnisses ausreichend zu beschreiben [Censi, 2007].

4.6 Zusammenfassung

In diesem Kapitel wurden Methoden zur Registrierung zweier überlappender Normal Distributions Transforms vorgestellt, mit denen die bezüglich der Überlappung bestmögliche relative Transformation $\mathbf{p}$ der Beobachtungen bestimmt werden kann. Die Grundlage dieser Registrierungsverfahren ist eine Distanzfunktion D_2, welche die Diskrepanz der statischen Beobachtung $\mathcal{M}$ und der transformierten Beobachtung $T(\mathcal{S}, \mathbf{p})$ bezüglich der Transformation bestimmt. Diese Distanzmetrik berechnet sich als Summe der Übereinstimmungswahrscheinlichkeiten für alle Komponenten $N_i \in T(\mathcal{S}, \mathbf{p})$ mit deren korrespondierenden Komponenten $N_j \in \mathcal{M}$. Um diese Korrespondenzen zu bestimmen, wird eine approximierende Nächste-Nachbarn-Suche in einem k-d-Baum über der euklidischen Distanz der Mittelwerte μ durchgeführt. Mit diesem Fehlermaß und einer Startschätzung ist es nun möglich, durch geeignete Optimierungsverfahren eine bezüglich D_2 optimale relative Transformation $\mathbf{p}$ zu ermitteln.
Da nicht immer eine ausreichend scharfe Startschätzung der relativen Transformation von $\mathcal{M}$ und $\mathcal{S}$ bekannt ist, wurden zwei unterschiedliche Registrierungsalgorithmen zur Bestimmung der bestmöglichen Transformation $\mathbf{p}$ entwickelt. Für scharfe Startschätzungen wird ein Levenberg-Marquardt-Algorithmus eingesetzt, der iterativ Schritte $\Delta(\mathbf{p})_t$ wählt, welche die Distanzmetrik $D_2(\mathcal{M}, \mathcal{S}, \mathbf{p}_{t-1} + \Delta(\mathbf{p})_t)$ minimieren. Der Algorithmus konvergiert für scharfe Startschätzungen robust, divergiert jedoch ab einer gewissen Unschärfe der Startschätzung. In diesen Fällen wird ein Partikelschwarmoptimierungsverfahren eingesetzt, das mit derartigen unscharfen Startschätzungen umgehen kann. Eine Population von Partikeln exploriert in diesem Verfahren semi-deterministisch den Parameterraum der Transformationsfunktion, und sucht dabei nach dem Parametervektor $\mathbf{p}$, der die Distanz $D_2(\mathcal{M}, \mathcal{S}, \mathbf{p})$ minimiert. Durch Erkennen und Lösen von Bewegungsstagnation sowie Integration des Gradienten der Zielfunktion konvergiert das Verfahren robust gegen lokale Minima, bei genügend Iterationen auch mit hoher

Wahrscheinlichkeit gegen das globale Minimum. Um die Unsicherheit der mit diesen Methoden bestimmten Registrierungsergebnisse abschätzen zu können, werden statistische und analytische Schätzverfahren eingesetzt. Für zwei überlappende Beobachtungen kann also die relative Transformation $\mathbf{p}$ und die zugehörige Unsicherheit Σ bestimmt werden, sowohl für scharfe als auch für unscharfe Startschätzungen. Im folgenden Kapitel werden diese Verfahren im Rahmen des SLAM-Frontends dieser Arbeit genutzt, um einen topologisch korrekten Posengraphen zu erzeugen.

KAPITEL 5

Graphbasierte Optimierung

In den vorherigen Kapiteln wurde die Grundlage für ein graphbasiertes SLAM-Verfahren geschaffen. Auf Basis der effizienten Oberflächenmodellierung des Normal Distributions Transforms konnten Registrierungsalgorithmen entwickelt werden, welche die Bestimmung der relativen Transformation und deren Unsicherheit sowohl mit scharfer als auch mit unscharfer Startschätzung ermöglichen. Diese Mittel können nun vom SLAM-Frontend verwendet werden, um während der Fahrt einen topologisch korrekten Posengraphen zu erstellen. Mit nichtlinearen Optimierungsmethoden wird darauf im SLAM-Backend eine Posenkonfiguration $\mathcal{X}^* = (\mathbf{x}_1^*, \ldots, \mathbf{x}_t^*)$ gesucht, welche die Menge der nichtlinearen Beschränkungen im Posengraphen bestmöglich erfüllt. In diesem Kapitel soll beschrieben werden, wie dieser Posengraph durch das SLAM-Frontend aufgebaut wird, und wie die Optimierung des Posengraphen durch das SLAM-Backend funktioniert.

5.1 Konstruktion des Posengraphen

Das SLAM-Frontend verarbeitet alle Sensor- und Steuerdaten, die der mobile Roboter während der Fahrt misst, und erzeugt daraus einen Posengraphen. Für diese Arbeit wird angenommen, dass der Roboter zu regelmäßigen Zeitpunkten t Odometriedaten $\mathbf{u}_t$ sowie Beobachtungen $\mathbf{z}_t$ messen kann. Dieser Abschnitt soll erklären, wie im SLAM-Frontend dieser Arbeit aus diesen Daten ein topologisch korrekter Posengraph erzeugt werden kann. Dabei soll der Datenfluss innerhalb des Frontends, der in Abbildung 5.1 dargestellt wird, genauer beschrieben werden.

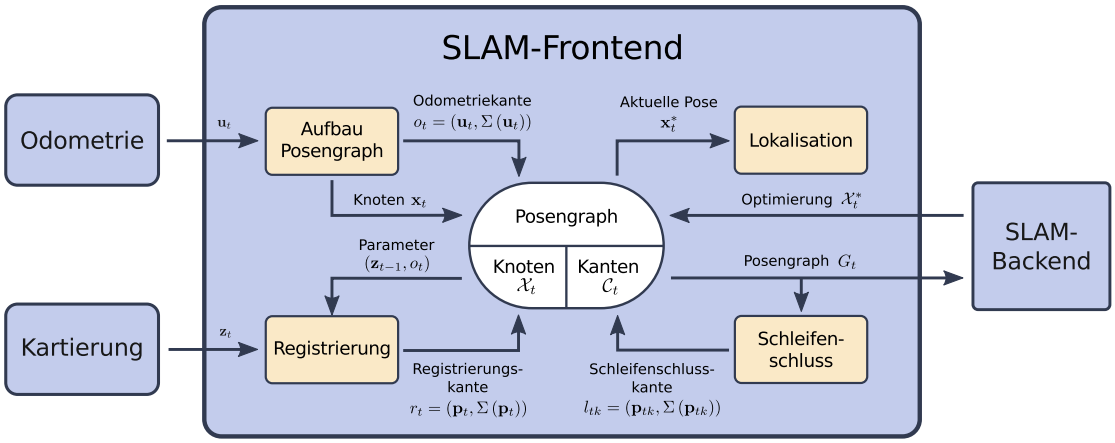


Abbildung 5.1: Detaillierte Ansicht des Datenflusses eines graphbasierten SLAM-Verfahrens, sowie der Hauptkomponenten eines SLAM-Frontends.

5.1.1 Knoten und Kanten

Jeder Graph $G = (V, E)$ besteht aus einer Menge von Knoten V und Kanten E (siehe Abbildung 5.1). Im Posengraphen dieser Arbeit beschreiben die Knoten $V = \bigcup \mathbf{x}_t$ jeweils eine Pose des mobilen Roboters zu einem bestimmten Zeitpunkt t. Die Kanten $E = \bigcup z_{ij}$ des Posengraphen beschreiben Beschränkungen über die relative Transformation der verbundenen Knoten $\mathbf{x}_i$ und $\mathbf{x}_j$, beispielsweise durch Odometriedaten oder Registrierungsergebnisse. Jede Kante beschreibt sowohl eine relative Transformation als auch die Unsicherheit dieser Transformation, so dass die Posen zweier Knoten durch die verbindenden Kanten beschränkt sind.

5.1.1.1 Einfügen von Knoten

Zu regelmäßigen Zeitpunkten t wird durch das Frontend ein neuer Knoten $\mathbf{x}_t$ mit zugehöriger Beobachtung $\mathbf{z}_t$ in den Posengraphen eingefügt. Die Pose dieses Knotens ist initial nicht exakt bestimmbar, es ist lediglich der von der Odometrie gemessene Versatz $\mathbf{u}_t$ seit dem letzten eingefügten Knoten $\mathbf{x}_{t-1}$ bekannt. Zur Initialisierung wird die Pose des neuen Knotens $\mathbf{x}_t$ also aus der Pose des vorherigen Knotens $\mathbf{x}_{t-1}$ und dem Odometrieversatz $\mathbf{u}_t$ berechnet:

$$\mathbf{x}_t = \mathbf{x}_{t-1} \oplus \mathbf{u}_t \tag{5.1}$$

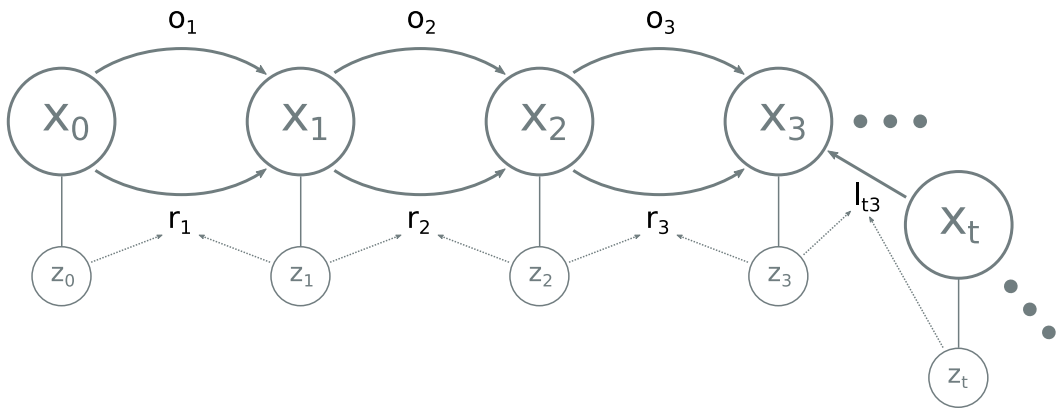


Abbildung 5.2: Veranschaulichung der Struktur eines Posengraphen. Knoten $\mathbf{x}_t$ sind durch Odometriekanten o_t und Registrierungskanten r_t mit ihren Vorgängerknoten $\mathbf{x}_{t-1}$ verbunden und besitzen jeweils eine Beobachtung $\mathbf{z}_t$, die an der Pose $\mathbf{x}_t$ aufgenommen wurde. Wird ein Schleifenschluss zwischen Knoten $\mathbf{x}_i$ und $\mathbf{x}_j$ entdeckt, so wird eine Schleifenschlusskante l_{ij} eingefügt.

Der Operator $\oplus$ ermöglicht hierbei die Transformation der Pose $\mathbf{x}_{t-1}$ anhand der Versatzdaten $\mathbf{u}_t$. Offensichtlich ist für diese rekursive Berechnungsvorschrift ein Startwert $\mathbf{x}_0$ nötig, der die Transformation des Posengraphen in das korrekte Koordinatensystem bewirkt [Grisetti et al., 2010a]. Die Knoten $(\mathbf{x}_0, \ldots, \mathbf{x}_t)$ des Posengraphen beschreiben folglich den gefahrenen Pfad des Roboters, jedoch unter dem Einfluss des akkumulierenden Odometriefehlers. Um die auf diese Weise eingeführte Unsicherheit explizit zu modellieren, werden vom Frontend Odometriekanten zwischen diese Knoten eingefügt.

5.1.1.2 Odometriekanten

In Kapitel 2 wurde im Rahmen des Bewegungsmodells $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_t)$ bereits über die Modellierung des Odometriefehlers als Normalverteilung gesprochen. Die in Gleichung (5.1) durch $\mathbf{u}_t$ eingeführte Posenunsicherheit kann folglich durch die Kovarianzmatrix $\Sigma(\mathbf{u}_t)$ parametrisiert werden. Die Knoten $\mathbf{x}_t$ und $\mathbf{x}_{t-1}$ stehen also durch die relative Transformation $\mathbf{u}_t$ und die zugehörige Unsicherheit $\Sigma(\mathbf{u}_t)$ in Beziehung. Diese Information wird im Posengraph explizit durch die sog. Odometriekanten repräsentiert. Sobald das Frontend also einen neuen Knoten einfügt, wird zugleich eine Odometriekante $o_t = (\mathbf{u}_t, \Sigma(\mathbf{u}_t))$ zwischen $\mathbf{x}_{t-1}$ und $\mathbf{x}_t$ eingefügt (siehe Kanten o_1, o_2 und o_3 in

Abbildung 5.2, sowie Block „Aufbau Posengraph“ in Abbildung 5.1).

5.1.1.3 Registrierungskanten

Die Registrierung der Beobachtungen zweier aufeinanderfolgender Knoten $\mathbf{x}_{t-1}$ und $\mathbf{x}_t$ durch die in Kapitel 4 vorgestellten Algorithmen ermöglicht eine genaue Schätzung der relativen Transformation $\mathbf{p}$ dieser Knoten und deren Unsicherheit $\Sigma(\mathbf{p})$. Diese Information wird als Registrierungskante $r_t = (\mathbf{p}, \Sigma(\mathbf{p}))$ zwischen $\mathbf{x}_{t-1}$ und $\mathbf{x}_t$ in den Posengraphen eingefügt, und stellt somit eine redundante und meist etwas genauere Schätzung der relativen Transformation gegenüber der Odometriekante o_t dar. Für jeden neuen Knoten $\mathbf{x}_t$ führt das Frontend den Levenberg-Marquardt-Registrierungsalgorithmus für $\mathbf{z}_t$ und $\mathbf{z}_{t-1}$ durch, mit der durch o_t gegebenen relativen Transformation als Startschätzung. Zwischen zwei Knoten $\mathbf{x}_{t-1}$ und $\mathbf{x}_t$ existiert folglich stets eine Odometriekante o_t und eine Registrierungskante r_t (siehe Kanten r_1, r_2 und r_3 in Abbildung 5.2, sowie Block „Registrierung“ in Abbildung 5.1).

5.1.2 Schleifenschlusskanten

Ein Spezialfall von Registrierungskanten sind die sog. Schleifenschlusskanten. Durch diese Kanten werden Redundanzen in den betrachteten Beobachtungen beschrieben, die durch das erneute Durchfahren eines bereits bekannten Gebiets entstehen. Durch Registrierung zweier solcher überlappender Beobachtungen $\mathbf{z}_t$ und $\mathbf{z}_k$ $(k \neq t-1)$ ist es möglich, die von der Odometrie geschätzte relative Transformation zwischen $\mathbf{x}_t$ und $\mathbf{x}_k$ erheblich zu verbessern. Um die topologische Korrektheit des Posengraphen zu gewährleisten, muss diese Information explizit als Kante zwischen diese Knoten geschrieben werden. Sobald ein Schleifenschluss entdeckt wird, werden die Beobachtungen der betroffenen Knoten $\mathbf{x}_t$ und $\mathbf{x}_k$ mit dem in Kapitel 4 vorgestellten Partikelschwarmverfahren registriert, und das Ergebnis als Schleifenschlusskante l_{tk} in den Posengraphen eingefügt (siehe Kante l_{t3} in Abbildung 5.2, sowie Block „Schleifenschluss“ in Abbildung 5.1). Das Entdecken und Auflösen von Schleifenschlüssen wird im nächsten Abschnitt genauer behandelt.

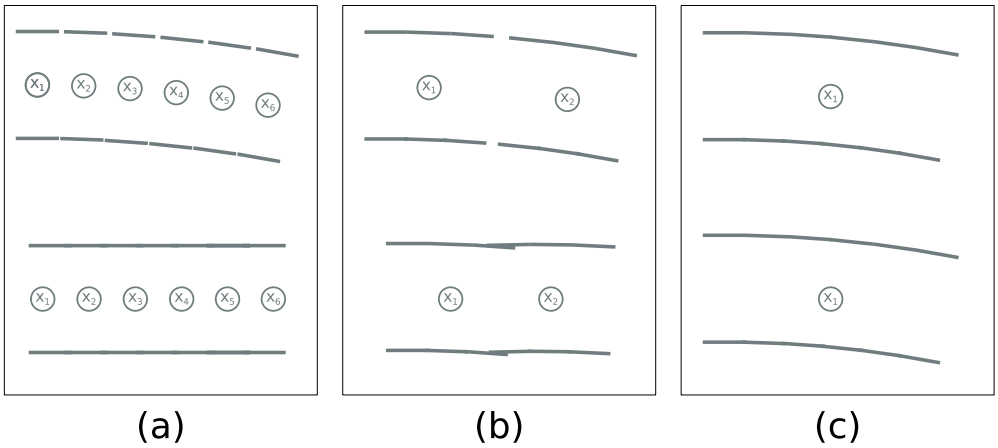


Abbildung 5.3: Veranschaulichung unterschiedlicher zeitlicher Diskretisierungsintervalle am Beispiel eines geraden Korridors. Die obere Reihe zeigt jeweils die aufeinanderfolgenden Beobachtungen der zeitlichen Diskretisierungsschritte (6 (a), 2 (b), 1 (c)). Die untere Reihe zeigt die durch Registrierung rekonstruierte Oberflächenstruktur, die aus den jeweiligen Beobachtungen ermittelt wurde. Größere Diskretisierungsschritte bedeuten folglich größere Modellierungsfehler, die nicht mehr durch Registrierung ausgeglichen werden können.

5.1.3 Zeitliche Diskretisierung

Ein bisher nicht betrachteter Faktor des Frontends ist das Diskretisierungsintervall der Messzeitpunkte t, an denen neue Knoten $\mathbf{x}_t$ mit Beobachtungen $\mathbf{z}_t$ in den Posengraphen eingefügt werden. Das kleinstmögliche akkurate Intervall ergibt sich direkt als kleinstes gemeinsames Vielfaches der Messgeschwindigkeiten der verwendeten Sensoren. Werden beispielsweise alle 25 ms Odometriedaten gemessen, während Tiefendaten nur alle 50 ms gemessen werden, so ist das kleinste mögliche Diskretisierungsintervall 50 ms. Würde man dieses Intervall nutzen, so würde der Posengraph mit einer Rate von 20 Knoten pro Sekunde wachsen. Bei einer Fahrtdauer von einer Stunde würde dies einen Posengraphen mit 72000 Knoten ergeben.

Bei begrenzt verfügbaren Ressourcen oder erforderter Echtzeitfähigkeit ist es daher oftmals wünschenswert, die Anzahl der Registrierungen und Schleifenschlussversuche sowie die Größe des Posengraphen durch ein größeres Diskretisierungsintervall zu reduzieren. Hierzu ist ein Kartierungsalgorithmus erforderlich, um mehrere Sensormessungen zu

einer Beobachtung zu vereinen. Die Wahl eines geeigneten Diskretisierungsintervalls ist nicht-trivial, wie Abbildung 5.3 zeigt. Schematisch werden Beobachtungen eines geraden Korridors illustriert, die mit drei unterschiedlich aufgelösten Diskretisierungsintervallen aufgenommen wurden. Im Vergleich der unteren Reihe zeigt sich deutlich, dass der Modellierungsfehler der durch die Registrierung rekonstruierten Oberflächenstruktur proportional mit der Größe des Diskretisierungsintervalls steigt. Eine intelligente Wahl der Diskretisierungsstrategie ist folglich nötig, um einen passenden Kompromiss zwischen Berechnungseffizienz und Modellierungsfehler zu finden. In der Praxis hängt dieses Intervall stark vom verwendeten Sensor und der Art der Anwendung ab.

5.1.4 Algorithmische Beschreibung

Algorithmus 4 beschreibt die Funktionsweise des SLAM-Frontends für passend gewählte Zeitschritte t. Zu jedem Zeitschritt wird die Pose des neuen Knotens durch die Odometriedaten bestimmt und der Knoten mit der zugehörigen Odometriekante in den Posengraphen eingefügt. Für die aktuelle und vorhergehende Beobachtung wird nun die in Kapitel 4 vorgestellte Levenberg-Marquardt-Registrierung durchgeführt und die bestimmte Registrierungskante zwischen den aktuellen und vorherigen Knoten eingefügt. Des Weiteren muss in jedem Zeitschritt nach potentiellen Schleifenschlüssen gesucht werden. Wird ein Schleifenschluss entdeckt, wird dieser gelöst und die zugehörige Schleifenschlusskante zwischen die betroffenen Knoten geschrieben. Nach jedem erfolgten Schleifenschluss wird die Optimierung durch das SLAM-Backend aufgerufen, um diese neue Information zur Reduktion der Posenunsicherheiten zu verarbeiten. Mit dieser optimierten Schätzung wird im nächsten Zeitschritt fortgefahren.

5.2 Schleifenschluss

Einer der wichtigsten Mechanismen eines SLAM-Algorithmus' ist das Entdecken und Auflösen von Schleifenschlüssen. Ein Schleifenschluss tritt auf, wenn der Roboter bereits bekanntes Gebiet durchfährt. In diesem Fall werden Umgebungsmerkmale beobachtet, die der Roboter in der Vergangenheit bereits gesehen hat. Durch das Abgleichen der

Data : Startpose $\mathbf{x}_0$, Beobachtungen $\mathbf{z}$, Odometriedaten $\mathbf{u}$

Result : Posengraph $G = (V, E)$

Erzeuge erste Beobachtung $\mathbf{z}_0$;

$V = (\mathbf{x}_0, \mathbf{z}_0)$;

for *Zeitschritt* t **do**

`// Einfügen des neuen Knotens`

Berechne $\mathbf{x}_t$ aus vorherigem Knoten $\mathbf{x}_{t-1}$ und Odometriedaten $\mathbf{u}_t$;

Erzeuge Beobachtung $\mathbf{z}_t$ aus den Sensormessungen;

$V = V \cup (\mathbf{x}_t, \mathbf{z}_t)$;

`// Einfügen der Odometriekante`

Berechne Odometrietransformation $\mathbf{o}_t$ mit Unsicherheit $\Sigma(\mathbf{o}_t)$;

$E = E \cup \left(\mathbf{x}_{t-1} \xrightarrow{(\mathbf{o}_t, \Sigma(\mathbf{o}_t))} \mathbf{x}_t\right)$;

`// Einfügen der Beobachtungskante`

Registriere $\mathbf{z}_t$ mit $\mathbf{z}_{t-1}$ für Startschätzung $\mathbf{o}_t$;

Extrahiere Registrierungstransformation $\mathbf{r}_t$ mit Unsicherheit $\Sigma(\mathbf{r}_t)$;

$E = E \cup \left(\mathbf{x}_{t-1} \xrightarrow{(\mathbf{r}_t, \Sigma(\mathbf{r}_t))} \mathbf{x}_t\right)$;

`// Schleifenschluss`

Bestimme Schleifenschlusskandidaten $\mathbf{x}_k \in C$;

Berechne relative Transformationen $\mathbf{p}_k$ und Unsicherheiten $\Sigma(\mathbf{p}_k)$;

$E = E \cup \left(\mathbf{x}_t \xrightarrow{(\mathbf{p}_k, \Sigma(\mathbf{p}_k))} \mathbf{x}_k\right)$;

`// Korrektur`

if *Schleifenschluss erfolgt* **then**

Führe Optimierung der Posen über aktuellen Graphen G_t durch;

Aktualisiere Odometrie mit neuer Information über aktuelle Pose $\mathbf{x}_t$;

end

end

Algorithmus 4: Erzeugung des Posengraphen durch das SLAM-Frontend.

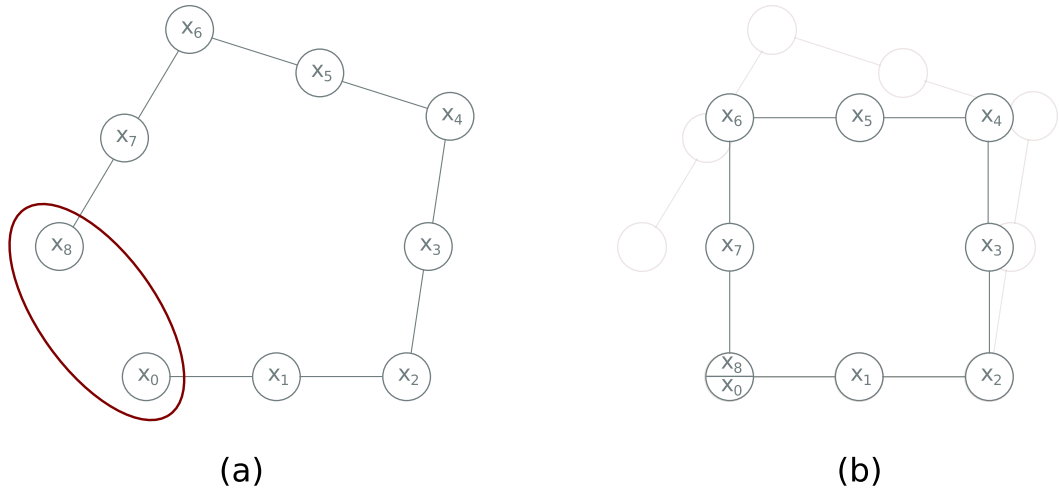


Abbildung 5.4: Veranschaulichung eines Schleifenschlusses für einen symbolischen Posengraphen. Jede Rotation ruft einen relativ großen Odometriefehler hervor, so dass die eigentlich übereinstimmenden Knoten $\mathbf{x}_0$ und $\mathbf{x}_8$ in (a) einen deutlichen Versatz aufweisen. Durch Vergleich der Beobachtungen $\mathbf{z}_8$ und $\mathbf{z}_0$ würde sich in diesem Beispiel erkennen lassen, dass der Roboter eigentlich auf seine Startposition zurückgefahren ist, und die zugehörigen Knoten somit relativ ähnliche Posen aufweisen müssen. Abbildung (b) zeigt die Anordnung der Knoten nach der Korrektur durch einen Schleifenschlussalgorithmus. Man erkennt deutlich die teils erhebliche Korrektur der Knotenposen.

Aufnahmeposen der neuen und alten Beobachtungen lässt sich die in der Zwischenzeit akkumulierte Unsicherheit bezüglich der Posen stark reduzieren. Abbildung 5.4 zeigt diesen Vorgang am Beispiel eines Posengraphen, der beim Fahren eines quadratischen Pfads aus den Odometriedaten erzeugt wurde. In diesem Beispiel kann durch das Entdecken und Auflösen des Schleifenschlusses eine erhebliche Korrektur der Knotenposen durchgeführt werden. Aus diesem Grund ist ein guter Schleifenschlussmechanismus essentiell für die topologische Korrektheit des Posengraphen.

5.2.1 Entdeckung von Schleifenschlüssen

Um einen Schleifenschluss zu entdecken, müssen diejenigen Beobachtungen der Vergangenheit bestimmt werden, die mit der aktuellen Beobachtung teilweise oder vollständig

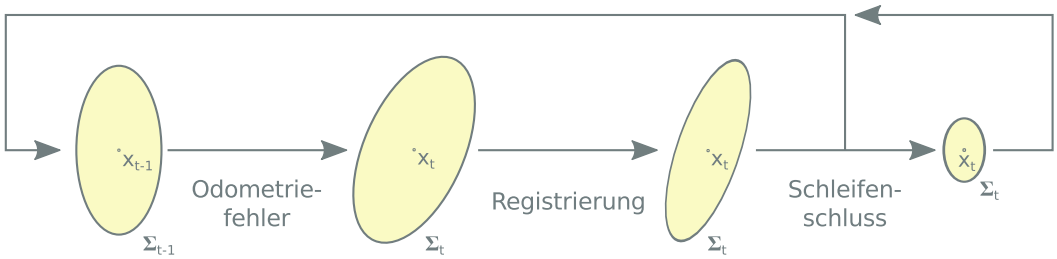


Abbildung 5.5: Veranschaulichung des Trackings der Posenunsicherheit. Die Unsicherheit des aktuellen diskreten Zeitschritts, hier dargestellt durch die in den x-y-Raum projizierte 3σ-Fehlerellipse, ergibt sich aus dem Odometriefehler und der vorherigen Posenunsicherheit. Durch Registrierung der Beobachtungen $\mathbf{z}_{t-1} \leftrightarrow \mathbf{z}_t$ ist es möglich, die aktuell geschätzte Unsicherheit etwas mehr auf die tatsächliche Unsicherheit zu reduzieren. Ein potentieller Schleifenschluss würde sogar eine ziemlich genaue Schätzung der tatsächlichen aktuellen Roboterpose $\mathbf{x}_t$ ermöglichen.

übereinstimmen könnten. Eine Grundvoraussetzung hierfür ist, dass die Unsicherheit über die einzelnen Aufnahmeposen korrekt geschätzt werden kann. Abbildung 5.5 zeigt das sog. *Tracking* der Posenunsicherheit, das für eine stets möglichst genaue Schätzung der Unsicherheit über die Posen sorgt. Die meist zu hoch eingeschätzte Unsicherheit der Odometrie kann durch Registrierung und eventuelle Schleifenschlüsse zu einer schärferen Hypothese über die Posenunsicherheit transformiert werden. Dieser Vorgang wird für jeden neu eingefügten Knoten iteriert, so dass für den Knoten $\mathbf{x}_t$ stets eine akkurate Schätzung der Posenunsicherheit vorliegt. Mit dieser Information kann nun die Wahrscheinlichkeit berechnet werden, dass zwei Posen $\mathbf{x}_t$ und $\mathbf{x}_k$ unter Berücksichtigung ihrer Unsicherheiten übereinstimmen könnten. Alle Knoten $\mathbf{x}_k$, die potentiell mit $\mathbf{x}_t$ übereinstimmen könnten, heißen Schleifenschlusskandidaten. Die Wahrscheinlichkeit ist hoch, dass die Beobachtung eines Schleifenschlusskandidaten mit der aktuellen Beobachtung überlappt.

5.2.2 Bestimmung der Schleifenschlusskandidaten

Um für zwei Knoten $\mathbf{x}_t$ und $\mathbf{x}_k$ mit zugehörigen relativen Unsicherheiten Σ_t und Σ_k die Wahrscheinlichkeit der physischen Übereinstimmung zu berechnen, wird eine

Distanzfunktion benötigt. Da die Pose eines Knotens als Mittelwert μ und die Posenunsicherheit als Kovarianz Σ einer Normalverteilung angesehen werden kann, bietet sich die erweiterte Mahalanobisdistanz als Distanzmaß an (siehe Abschnitt A.3). Für zwei Knoten lässt sich die erweiterte Mahalanobisdistanz wie folgt berechnen:

$$d\left(\mathbf{x}_t, \mathbf{x}_k\right) = \sqrt{\left(\mathbf{x}_t - \mathbf{x}_k\right)^T \left(\Sigma_t + \Sigma_k\right)^{-1} \left(\mathbf{x}_t - \mathbf{x}_k\right)} \tag{5.2}$$

Daraus ergibt sich direkt die Wahrscheinlichkeitsdichte der zugehörigen Normalverteilung $N\left(\mathbf{x}_t - \mathbf{x}_k, \Sigma_t + \Sigma_k\right)$:

$$p\left(\mathbf{x}_t, \mathbf{x}_k\right) = c \cdot \exp\left(-\frac{d\left(\mathbf{x}_t, \mathbf{x}_k\right)^2}{2}\right) \tag{5.3}$$

Mit Gleichung (5.3) lässt sich nun für alle Knoten $\mathbf{x}_k$ bestimmen, mit welcher Wahrscheinlichkeit sie mit $\mathbf{x}_t$ übereinstimmen könnten. Algorithmus 5 beschreibt den nun relativ trivialen Vorgang der Bestimmung der Schleifenschlusskandidaten C. Für alle Kandidatenknoten $\mathbf{x}_k \in C$ ist die Wahrscheinlichkeit hoch, dass die zugehörige Beobachtung $\mathbf{z}_k$ mit der Beobachtung $\mathbf{z}_t$ des aktuellen Knotens $\mathbf{x}_t$ überlappen könnte. Durch Registrierung dieser Beobachtungen kann die relative Unsicherheit zwischen den beiden Knoten folglich stark reduziert werden.

5.2.3 Auflösung potentieller Schleifenschlüsse

Für alle möglichen Schleifenschlusskandidaten $(\mathbf{x}_t, \mathbf{x}_k)$ mit $\mathbf{x}_k \in C$ kann nun eine Registrierung durchgeführt werden, um die bislang sehr unscharfe relative Transformation der Knoten genauer zu bestimmen. In Kapitel 4 wurde das Problem der Registrierung mit unscharfer Startschätzung bereits angesprochen, und ein Registrierungsalgorithmus auf Basis der Partikelschwarmoptimierung entwickelt. Dieser Algorithmus wird nun eingesetzt, um $\mathbf{z}_t$ mit $\mathbf{z}_k$ zu registrieren. Als Startschätzung dient die Aufnahmepose $\mathbf{x}_t$ und die relative Unsicherheit Σ_{tk} zwischen $\mathbf{x}_t$ und $\mathbf{x}_k$. In dem zugehörigen Konfidenzbereich muss sich die korrekte Aufnahmepose $\mathbf{x}_t^*$ befinden, die $\mathbf{z}_t$ bestmöglich mit $\mathbf{z}_k$ überlappen lassen würde. Algorithmus 6 beschreibt die Auflösung potentieller Schleifenschlüsse durch Registrierung. Für die erhaltenen relativen Transformationen $\mathbf{p}_k : \mathbf{x}_t \rightarrow \mathbf{x}_k$ und die zugehörigen Unsicherheiten Σ_k können nun Schleifenschlusskanten für die betroffenen Knoten $\mathbf{x}_t$ und $\mathbf{x}_k$ eingefügt werden.

Data : Aktuelle Pose $\mathbf{x}_t$, vergangene Posen $(\mathbf{x}_1, \ldots, \mathbf{x}_{t-1})$

Result : Schleifenschlusskandidaten C

$C = \emptyset$;

for *Alle vergangenen Posen* $\mathbf{x}_k$ **do**

  Berechne erweiterte Mahalanobisdistanz $d_k = d(\mathbf{x}_k, \mathbf{x}_t)$ nach (5.2);

  $d_k = \sqrt{(\mathbf{x}_k - \mathbf{x}_t)^T (\Sigma_t + \Sigma_k)^{-1} (\mathbf{x}_k - \mathbf{x}_t)}$;

  Berechne Wahrscheinlichkeit der Übereinstimmung nach (5.3);

  $p(\mathbf{x}_t, \mathbf{x}_k) = c \cdot \exp\left(-\frac{d(\mathbf{x}_t, \mathbf{x}_k)^2}{2}\right)$;

  if $p(\mathbf{x}_t, \mathbf{x}_k) \geq$ *geeigneter Schwellwert* **then**

    $C = C \cup \mathbf{x}_k$;

  end

end

Algorithmus 5: Bestimmung der Schleifenschlusskandidaten.

5.3 Optimierung des Posengraphen

In diesem Abschnitt soll die Optimierung des erzeugten Posengraphen näher ausgeführt werden. Durch den Posengraphen ist eine Menge Knoten bekannt, deren Posen als sog. Konfiguration $\mathcal{X} = (\mathbf{x}_0, \ldots, \mathbf{x}_t)$ bezeichnet werden. Diese Knoten sind durch eine Menge von Bedingungen $\mathcal{C} = \bigcup z_{ij}$ verbunden. Für zwei Knoten $\mathbf{x}_i$ und $\mathbf{x}_j$ stellt eine Bedingung z_{ij} mit zugehöriger Informationsmatrix Ω_{ij} eine Vorhersage der Pose des Knotens $\mathbf{x}_j$ relativ zu $\mathbf{x}_i$ dar:

$$h(\mathbf{x}_i, z_{ij}) = \hat{\mathbf{x}}_j \tag{5.4}$$

Das Bestimmen derartiger Bedingungen durch Odometrie, Registrierung sowie Schleifenschlüsse ist die Aufgabe des SLAM-Frontends. Für eine Knotenkonfiguration $\mathcal{X}$ und eine Bedingung z_{ij} kann anhand dieser Vorhersage berechnet werden, wie gut die Knotenkonfiguration die Bedingung erfüllt:

$$e(z_{ij}, \mathcal{X}) = \mathbf{x}_j - h(\mathbf{x}_i, z_{ij}) = \mathbf{x}_j - \hat{\mathbf{x}}_j \tag{5.5}$$

Data : Aktuelle Pose $\mathbf{x}_t$, Schleifenschlusskandidaten $C = \bigcup \mathbf{x}_k$

Result : Schleifenschlüsse $T = \{(\mathbf{x}_k, \mathbf{p}_k, \Sigma_k)\}$

$T = \emptyset$;

for *Jeden Kandidaten* $\mathbf{x}_k \in C$ **do**

  Berechne relative Unsicherheit Σ_{tk} zwischen $\mathbf{x}_t$ und $\mathbf{x}_k$;

  Registriere $\mathbf{z}_t$ und $\mathbf{z}_k$ durch PSO auf Suchbereich $(\mathbf{x}_t, \Sigma_{tk})$;

  Extrahiere relative Transformation $\mathbf{p}_k$ aus bester Position $\mathbf{g}$;

  Berechne Stichprobenvarianz Σ_k aus allen abgetasteten Positionen;

  $T = T \cup (\mathbf{x}_k, \mathbf{p}_k, \Sigma_k)$;

end

Algorithmus 6: Auflösung der Schleifenschlüsse durch Registrierung.

Abbildung 5.6 veranschaulicht diese Berechnung geometrisch. Um die bezüglich aller Bedingungen optimale Knotenkonfiguration $\mathcal{X}^*$ zu finden, muss der Fehler $e(z_{ij}, \mathcal{X})$ für alle Bedingungen $z_{ij} \in \mathcal{C}$ minimiert werden.

5.3.1 Methode der kleinsten Quadrate

Wie bereits die Registrierung zweier Beobachtungen in Kapitel 4 wird auch dieses nichtlineare Optimierungsproblem durch die Methode der kleinsten Quadrate gelöst. In diesem Fall stellt die Knotenkonfiguration $\mathcal{X}$ die Parametrisierung der Modellfunktion dar, während die Vorhersagen durch $z_{ij} \in \mathcal{C}$ als Messwerte betrachtet werden. Der gesamte zu minimierende quadratische Fehler ergibt sich mit Gleichung (5.5) wie folgt [Grisetti et al., 2010a]:

$$\begin{aligned} F(\mathcal{X}) &= \sum_{z_{ij} \in C} (\mathbf{x}_j - h(\mathbf{x}_i, z_{ij}))^T \Omega_{ij} (\mathbf{x}_j - h(\mathbf{x}_i, z_{ij})) \\ &= \sum_{z_{ij} \in C} (\mathbf{x}_j - \hat{\mathbf{x}}_j)^T \Omega_{ij} (\mathbf{x}_j - \hat{\mathbf{x}}_j) \end{aligned} \tag{5.6}$$

Für eine scharfe Startschätzung $\mathcal{X}_0$ kann zur Minimierung des Fehlers ein numerisches Lösungsverfahren wie das Gauß-Newton-Verfahren oder der Levenberg-Marquardt-Algorithmus eingesetzt werden. Analog zu Kapitel 4 dienen auch hier die Odometriewerte als akzeptable Startwerte. Durch eine Linearisierung der Fehlerfunktion durch

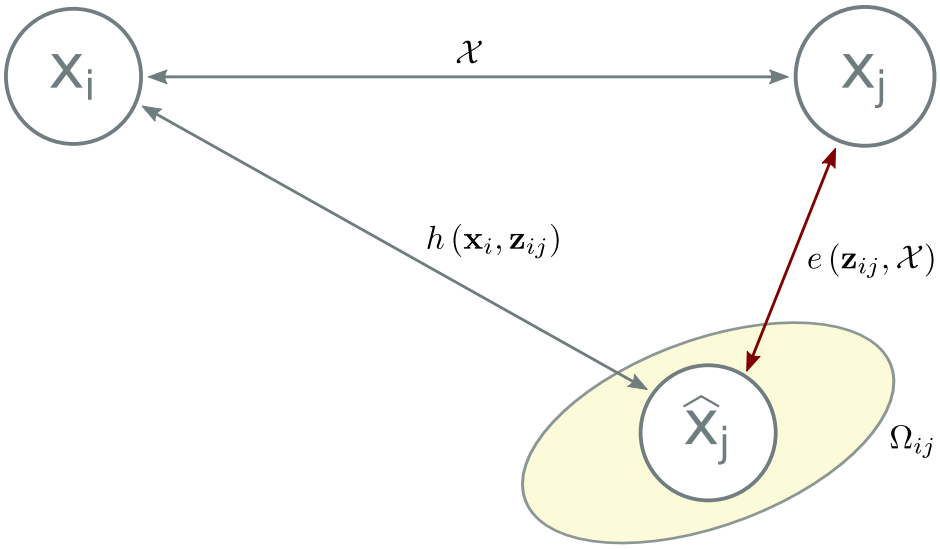


Abbildung 5.6: Veranschaulichung einer Bedingung z_{ij} und des zugehörigen Fehlers $e(z_{ij}, \mathcal{X})$ für eine Knotenkonfiguration $\mathcal{X}$. Die Informationsmatrix Ω_{ij} wird hier als 3σ-Fehlerellipse der zugehörigen Kovarianzmatrix dargestellt.

Taylorreihenentwicklung erster Ordnung können für eine derartige Knotenkonfiguration nun iterativ Schritte $\Delta(\mathcal{X})$ bestimmt werden, um Gleichung (5.6) zu minimieren.

5.3.2 Optimierungsframework *g2o*

Zur Optimierung des Posengraphen durch die Minimierung des quadratischen Fehlers (5.6) wird in dieser Arbeit das Open Source-Framework *g2o* von Grisetti et al. benutzt [Grisetti et al., 2010b]. Durch den Rotationsanteil der dreidimensionalen Transformationsfunktion ist der Zustandsraum des Optimierungsproblems nicht-euklidisch, so dass Grisetti et al. die Minimierung des quadratischen Fehlers auf einer Mannigfaltigkeit durchführen. Um mit den potentiellen Singularitäten des Rotationsanteils umzugehen, wird die aktuelle Schätzung $\hat{\mathcal{X}}$ stets in einer überparametrisierten Form dargestellt, beispielsweise durch Quaternionen. Da derartige Darstellungsformen gewissen Beschränkungen unterliegen, die von den verwendeten Lösungsverfahren nicht berücksichtigt werden, wird zur Berechnung der inkrementellen Schritte $\Delta(\mathcal{X})$ eine minimale, euklidische Darstellungsform des Rotationsanteils verwendet, beispielsweise die in dieser Arbeit verwendeten Eulerwinkel [Grisetti et al., 2010a]. Durch einen speziellen Ope-

rator $\boxplus$ wird ein derartiger Schritt in minimaler euklidischer Darstellungsform in den nicht-euklidischen Zustandsraum überführt:

$$\mathcal{X}_t = \mathcal{X}_{t-1} \boxplus \Delta\left(\mathcal{X}_t\right) \tag{5.7}$$

Durch die Überführung der minimalen, euklidischen Schritte in den globalen, überparametrisierten Zustandsraums bleibt die Lösung konsistent und frei von Fehlereinflüssen wie Rotationssingularitäten.
Um die Optimierung zu beschleunigen, nutzen Grisetti et al. zudem eine Hierarchie aus stufenweise abstrahierten Posengraphen, deren Knoten jeweils einen Teilgraphen der darunterliegenden Abstraktionsebene repräsentieren. Diese verschiedenen Abstraktionsebenen ermöglichen eine effizientere Optimierung durch selektives Aktualisieren von Teilbäumen.

5.3.3 Erweiterung Vertigo

Der größte Nachteil eines derartigen Optimierungsverfahrens ist die starke Abhängigkeit von einem topologisch konsistenten Posengraphen. Ein topologisch korrekter Posengraph enthält nur Kanten, die eine im Rahmen der assoziierten Unsicherheit korrekte relative Transformation zwischen zwei Knoten beschreiben. Während die topologische Korrektheit der Odometrie- und Registrierungskanten in den meisten Fällen gegeben ist, kann dies für Schleifenschlusskanten nur schwer garantiert werden. Da ein falsch-positiver Schleifenschluss die topologische Konsistenz des Posengraphen zerstören würde, muss die Wahrscheinlichkeit falsch-positiver Schleifenschlüsse so klein wie möglich gehalten werden [Magnusson, 2009].
Trotz möglichst robuster Schleifenschlussalgorithmen kann meist jedoch nicht ausgeschlossen werden, dass topologisch inkonsistente Kanten in den Posengraphen gelangen. Sünderhauf und Protzel stellen daher eine Methode vor, um das Ausgleichsproblem gegen solche Ausreißer robuster zu machen. Ihr Beitrag *Vertigo* erweitert das Optimierungsframework *g2o*, so dass die Topologie des Posengraphen in gewissem Maße Teil der Optimierung wird [Sunderhauf und Protzel, 2012]. Zu diesem Zweck führen sie sog. Schaltvariablen $s_{ij} \in \mathcal{S}$ ein, die auf die Fehlerterme der zugehörigen Kanten

multipliziert werden:

$$\begin{aligned} e\left(z_{ij}, \mathcal{X}, \mathcal{S}\right) &= s_{ij} \cdot \left(\mathbf{x}_j - h\left(\mathbf{x}_i, z_{ij}\right)\right) \\ &= s_{ij} \cdot \left(\mathbf{x}_j - \hat{\mathbf{x}}_j\right) \end{aligned} \tag{5.8}$$

Indem Sünderhauf und Protzel den Zustandsraum der Optimierung um die Menge aller Schaltvariablen $\mathcal{S}$ erweitern, ermöglichen sie dem Optimierungsverfahren das dynamische Aktivieren und Deaktivieren von Kanten durch das „Umschalten“ der Schaltvariablen $s_{ij} \in \{0, 1\}$. Ein falscher Schleifenschluss führt im Normalfall zu vielen verletzten Bedingungen z_{ij} und somit zu einem großen quadratischen Fehler in Gleichung (5.6). Das Deaktivieren der entsprechenden Schleifenschlusskante l_{ij} würde folglich zu einem erheblich geringeren Fehler führen, so dass der Gradient der Zielfunktion in der Dimension s_{ij} in Richtung 0 sehr groß wäre. Das Optimierungsverfahren würde folglich einen Schritt wählen, der zur „Abschaltung“ dieser Kante führt. Falsche Schleifenschlusskanten können folglich als Teil des Optimierungsprozesses erkannt und deaktiviert werden. Auf diese Weise kann trotz nicht-perfektem Verhalten des SLAM-Frontends eine konsistente Lösung berechnet werden.

5.4 Zusammenfassung

In diesem Kapitel wurde das graphbasierte SLAM-Verfahren beschrieben, das im Rahmen dieser Arbeit entwickelt wurde. Das SLAM-Frontend erzeugt einen topologisch korrekten Posengraphen, indem neue Knoten zunächst auf Basis der fehlerbehafteten Odometriedaten eingefügt werden. Durch Registrierung der Beobachtungen mit dem Levenberg-Marquardt-Registrierungsalgorithmus kann die relative Transformation zweier aufeinanderfolgender Knoten bereits etwas genauer bestimmt werden. Diese Informationen werden als Odometrie- und Registrierungskanten in den Posengraphen geschrieben. Um Schleifenschlüsse zu entdecken, wird jedes Mal nach dem Einfügen eines neuen Knotens mit der erweiterten Mahalanobisdistanz eine Liste von Knoten bestimmt, die aufgrund ihrer relativen Posenunsicherheit potentiell mit dem aktuellen Knoten übereinstimmen könnten. Für diese Schleifenschlusskandidaten wird der partikelschwarmbasierte Registrierungsalgorithmus durchgeführt, und die bestimmte

relative Transformation sowie die zugehörige Unsicherheit als Schleifenschlusskante in den Posengraphen eingefügt.
Zur Optimierung wird das freie SLAM-Backend *g2o* von Grisetti et al. verwendet, das die Knotenkonfiguration des Posengraphen bezüglich der durch die Kanten vorgegebenen Beschränkungen optimiert. Zur Optimierung wird ein numerisches Lösungsverfahren auf einer Mannigfaltigkeit verwendet, das iterativ den quadratischen Fehler zwischen geschätzter Knotenkonfiguration und Vorhersage durch die Kantenbedingungen minimiert. Um mit topologischen Defekten des Posengraphen umgehen zu können, wird eine Erweiterung von Sünderhauf und Protzel verwendet, welche die Topologie des Graphen zum Gegenstand der Optimierung erklärt. Durch Schaltvariablen können topologisch falsche Kanten ausgeschaltet werden, so dass sie nicht mehr verzerrend in die Berechnung des quadratischen Fehlers eingehen. Der resultierende Posengraph enthält folglich keine fälschlicherweise eingefügten topologischen Inkonsistenzen mehr.
Das Leistungsverhalten des entwickelten graphbasierten SLAM-Verfahrens und der darin enthaltenen Algorithmen soll im folgenden Kapitel evaluiert werden.

KAPITEL 6

Evaluation

In diesem Kapitel soll das in dieser Arbeit vorgestellte graphbasierte SLAM-Verfahren anhand der entwickelten Implementierung **NDTSLAM** evaluiert werden. Die Implementierung wurde in C++ für die Middleware *MIRA* [Einhorn et al., 2012] geschrieben und wurde zur Anfertigung aller Messungen in diesem Kapitel verwendet.

6.1 Leistungsverhalten des Frontends

Zunächst soll das Leistungsverhalten des entwickelten SLAM-Frontends evaluiert werden. Auf eine explizite Leistungsbewertung des Normal Distributions Transforms als Darstellungsform wird an dieser Stelle verzichtet, da dies keinen uneingeschränkten Eigenanteil dieser Arbeit darstellt. Eine hervorragende Analyse und Evaluation findet sich in [Magnusson, 2009]. Stattdessen soll in diesem Abschnitt das Leistungsverhalten der Registrierungsalgorithmen evaluiert werden, die in Kapitel 4 entwickelt wurden.

6.1.1 Registrierung mit scharfer Startschätzung

Zur Registrierung zweier überlappender Beobachtungen mit scharfer Startschätzung wurde ein Levenberg-Marquardt-Algorithmus implementiert. Ausgehend von einer Startschätzung wird die zu minimierende Zielfunktion D_2 iterativ linearisiert, um mit Hilfe des Gradienten einen Schritt in Richtung des gesuchten Minimums machen zu können. Abbildung 6.1 zeigt diesen Vorgang für die Registrierung zweier identischer Normal Distributions Transforms, die zur Evaluierung um $(dx = 1\text{m}, dy = 1\text{m})$ versetzt wurden. Über dem x-y-Schnitt des Fehlergebirges der zu minimierenden Distanzfunktion

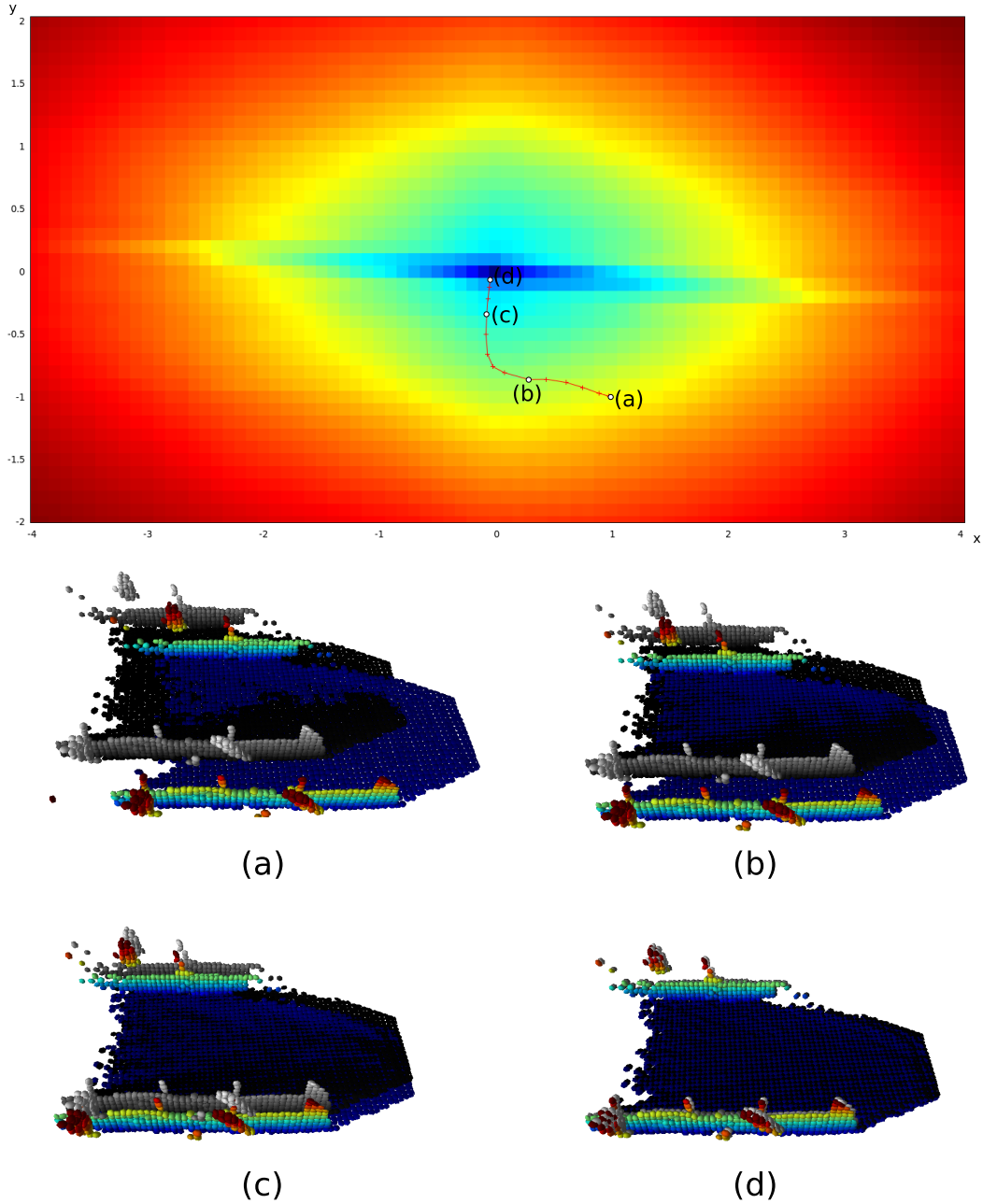


Abbildung 6.1: Veranschaulichung der ersten 20 Iterationen einer realen Registrierung mit dem Levenberg-Marquardt-Algorithmus über dem x-y-Schnitt des Fehlergebirges der Distanzfunktion. Abbildungen (a) bis (d) zeigen jeweils die aktuelle relative Transformation der beiden Normal Distributions Transforms für verschiedene Zeitpunkte während der Registrierung.

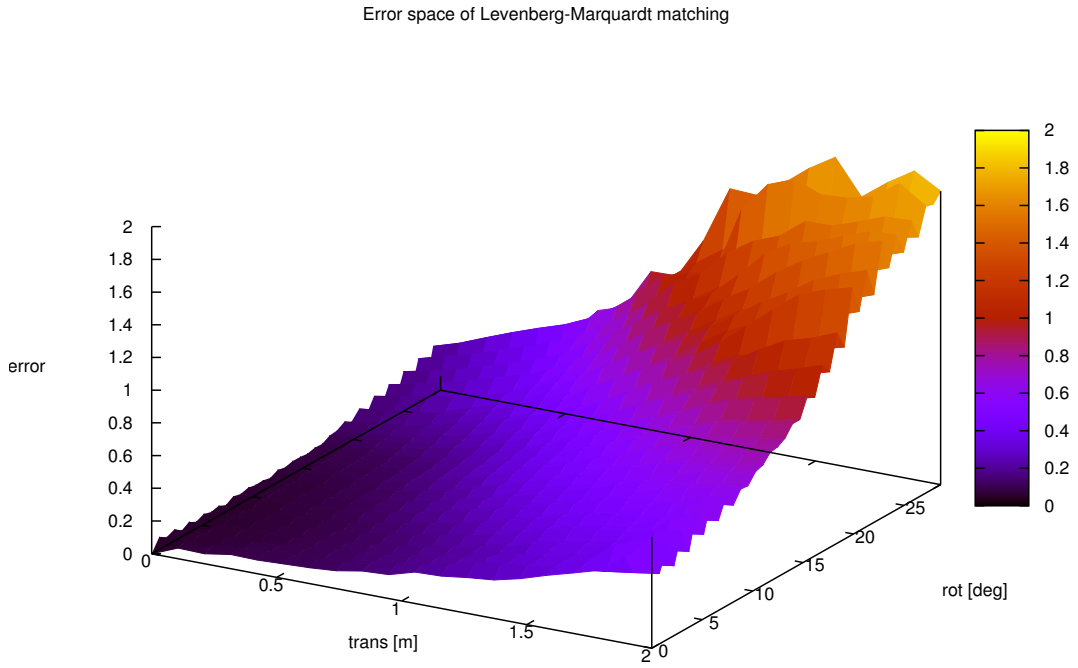


Abbildung 6.2: Veranschaulichung des Fehlers der Levenberg-Marquardt-Registrierung für die jeweils vorgegebenen relativen Transformationen. Gezeigt wird das Fehlergebirge über der Abweichung zwischen vorgegebener und ermittelter relativer Transformation im Bereich $[0\text{ m} \leq \sqrt{x^2+y^2} \leq 2\text{ m}]$ und $[0° \leq \phi_z \leq 29°]$. Die Farbgebung sowie Höhe beschreibt den durchschnittlichen summierten Fehler der Registrierung für die jeweilige Vorgabe.

ist der vom Algorithmus verfolgte Lösungsweg mit roten Kreuzen markiert. Dargestellt werden die ersten 20 von 30 Iterationen, nach denen der Algorithmus bereits eine gute Lösung von $(dx = 0.96\text{m}, dy = 0.94\text{m})$ ermittelt hat. Ab diesem Punkt kann es zur Oszillation mit minimalen Schrittweiten kommen, da die zu minimierende Distanzfunktion durch die minimale Zellgröße von hier 10 cm in diesem Größenbereich relativ homogen wird. Aus diesem Grund wird meist ein Abbruchkriterium bezüglich der Schrittgröße $\|\Delta(\mathbf{p})\|$ definiert, das in diesem Fall nach 20 Iterationen erreicht wurde.

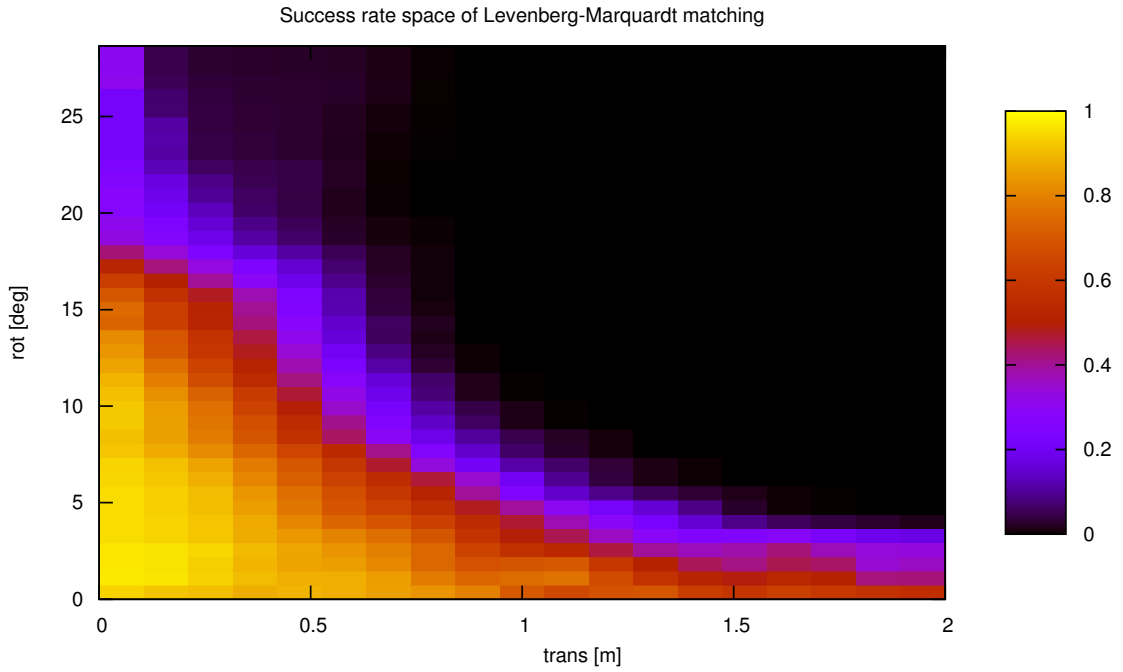


Abbildung 6.3: Veranschaulichung der Erfolgsrate der Levenberg-Marquardt-Registrierung für die jeweils vorgegebenen relativen Transformationen. Die Farbgebung beschreibt die durchschnittliche Erfolgsrate der Registrierung für die Vorgaben im Bereich $[0 \text{ m} \leq \sqrt{x^2 + y^2} \leq 2 \text{ m}]$ und $[0° \leq \phi_z \leq 29°]$. Eine Registrierung ist erfolgreich, wenn Rotations- und Translationsfehler kleiner als 3° bzw. 5 cm sind.

6.1.1.1 Fehler und Erfolgsrate

Wie bereits erwähnt, hängt das Leistungsverhalten des Levenberg-Marquardt-Verfahrens stark von der Qualität der verwendeten Startschätzung ab. Um die Abhängigkeit der Registrierungsqualität von der Güte der Startschätzung zu evaluieren, wurde der durchschnittliche Fehler sowie die Erfolgsrate für die Registrierung der in Abbildung 6.1 vorgestellten Normal Distributions Transforms ermittelt. Abbildung 6.2 zeigt den Registrierungsfehler, also die durchschnittliche summierte Abweichung des Ergebnisses von der vorgegebenen relativen Transformation, für insgesamt 12800 verschieden transformierte Registrierungsvorgänge. Um die ermittelten Daten dreidimensional darstellen zu können, wurden x und y zu einer Größe xy kombiniert. Für jeden diskreten Schritt

r auf der xy-Achse wurden x und y derart bestimmt, so dass gilt:

$$r = \sqrt{x^2 + y^2} \tag{6.1}$$

Diese Gleichung beschreibt einen Kreis in der x-y-Ebene, auf dem jeweils 16 Wertepaare (x, y) ermittelt wurden. Für alle betrachteten xy-Schritte mit $[0 \text{ m} \leq \sqrt{x^2 + y^2} \leq 2 \text{ m}]$ wurde der Rotationswinkel um die z-Achse (ϕ_z) im Bereich $[0° \leq \phi_z \leq 29°]$ gewählt. Für alle evaluierten Transformationen (x, y, ϕ_z) wurde der Fehler in allen sechs Dimensionen der Transformationsparametrisierung $\mathbf{p}$ summiert und als z-Wert in Abbildung 6.2 visualisiert. Das Ergebnis zeigt einen langsam ansteigenden Fehler, der aufgrund der Form der verwendeten Normal Distributions Transforms etwas anfälliger für Translationen ist. Ab einer gewissen Schwelle wird die relative Transformation zu groß, so dass der Levenberg-Marquardt-Algorithmus keine akzeptablen Lösungen mehr findet. Dies zeigt sich vor allem in der korrespondierenden Erfolgsrate in Abbildung 6.3. Eine Registrierung wurde als erfolgreich gewertet, wenn der summierte Fehler *aller* Translationskomponenten kleiner als 5 cm und der summierte Fehler *aller* Rotationskomponenten kleiner als 3° war.

6.1.1.2 Laufzeit

Die Levenberg-Marquardt-Registrierung zeichnet sich vor allem durch ihre robuste Konvergenzrate für scharfe Startschätzungen aus, bietet jedoch auch eine gute Laufzeit. Für die Abbildungen 6.2 und 6.3 wurde die Levenberg-Marquardt-Registrierung mit 30 Iterationen und einem Abbruchkriterium von $\|\Delta(\mathbf{p})_t\| \leq 0.5$ cm ausgeführt. In dieser Konfiguration benötigte eine Registrierung im Durchschnitt 170 ms mit einer Varianz von ±44 ms. Eine einzelne Iteration benötigte im Durchschnitt 8 ms mit einer Varianz von ±1.4 ms, so dass durchschnittlich 21 Iterationen durchgeführt wurden, bevor das Abbruchkriterium erreicht wurde. Die Registrierung mit dem Levenberg-Marquardt-Algorithmus ist folglich selbst in dieser strikten Konfiguration für bis zu fünf Beobachtungen pro Sekunde echtzeitfähig.

6.1.1.3 Evaluation

Insgesamt bietet die Levenberg-Marquardt-Registrierung eine gute Konvergenzrate bei hervorragender Laufzeit. Aufgrund der schnellen Divergenz bei zu großen relativen Transformationen eignet sich der Algorithmus jedoch nur für die Registrierung mit scharfen Startschätzungen, beispielsweise für zwei aufeinanderfolgende Beobachtungen.

6.1.2 Registrierung mit unscharfer Startschätzung

Zur Registrierung zweier überlappender Beobachtungen mit nur unscharf bekannter relativer Transformation wurde in Kapitel 4 ein partikelschwarmbasiertes Registrierungsverfahren entwickelt. Dieses exploriert den Parameterraum der Transformationen $\mathbf{p}$ semi-deterministisch und folgt dabei einer biologisch inspirierten Schwarmdynamik. Abbildung 6.4 veranschaulicht diesen Vorgang auf dem bereits bekannten Fehlergebirge mit den bekannten Normal Distributions Transforms (siehe Abbildung 6.1). Gezeigt werden die Iterationen 2, 4, 5, 7 und 9 der zehn Iterationen, die für die fünf verwendeten Partikeln durchgeführt wurden. Die Partikeln wurden für dieses Beispiel zufällig im Bereich $[-8\text{ m} \leq x, y \leq 8\text{ m}]$ eingestreut. In den zehn Iterationen erreichte der Schwarm mit der gefundenen besten Position $\mathbf{g} = (0.05\text{ m}, -0.02\text{ m})$ ein sehr gutes Ergebnis (siehe Abbildung 6.4 (e) für eine Visualisierung dieser Transformation).

6.1.2.1 Fehler und Erfolgsrate

Analog zur Bestimmung des Fehlers des Levenberg-Marquardt-Registrierungsverfahrens wurde auch für den partikelschwarmbasierten Registrierungsalgorithmus eine Auswertung des Fehlers für verschiedene Translationen und Rotationen vorgenommen. Auf einem Bereich von $[0\text{ m} \leq \sqrt{x^2 + y^2} \leq 10\text{ m}]$ und $[0° \leq \phi_z \leq 86°]$ wurden insgesamt 12800 Registrierungen durchgeführt und jeweils die summierte Abweichung von der perfekten Transformation (in diesem Fall $(0, 0, 0)$) als Fehler der Registrierung definiert. Abbildung 6.5 zeigt das entstandene Fehlergebirge und veranschaulicht gut, wie tolerant die Partikelschwarmoptimierung gegenüber großen Suchräumen ist. Weder hohe Rotations- noch Translationswerte bewirken einen signifikanten Fehler, was sich auch in der Abbildung der Erfolgsrate 6.6 zeigt. Der große Anstieg des Fehlers und der

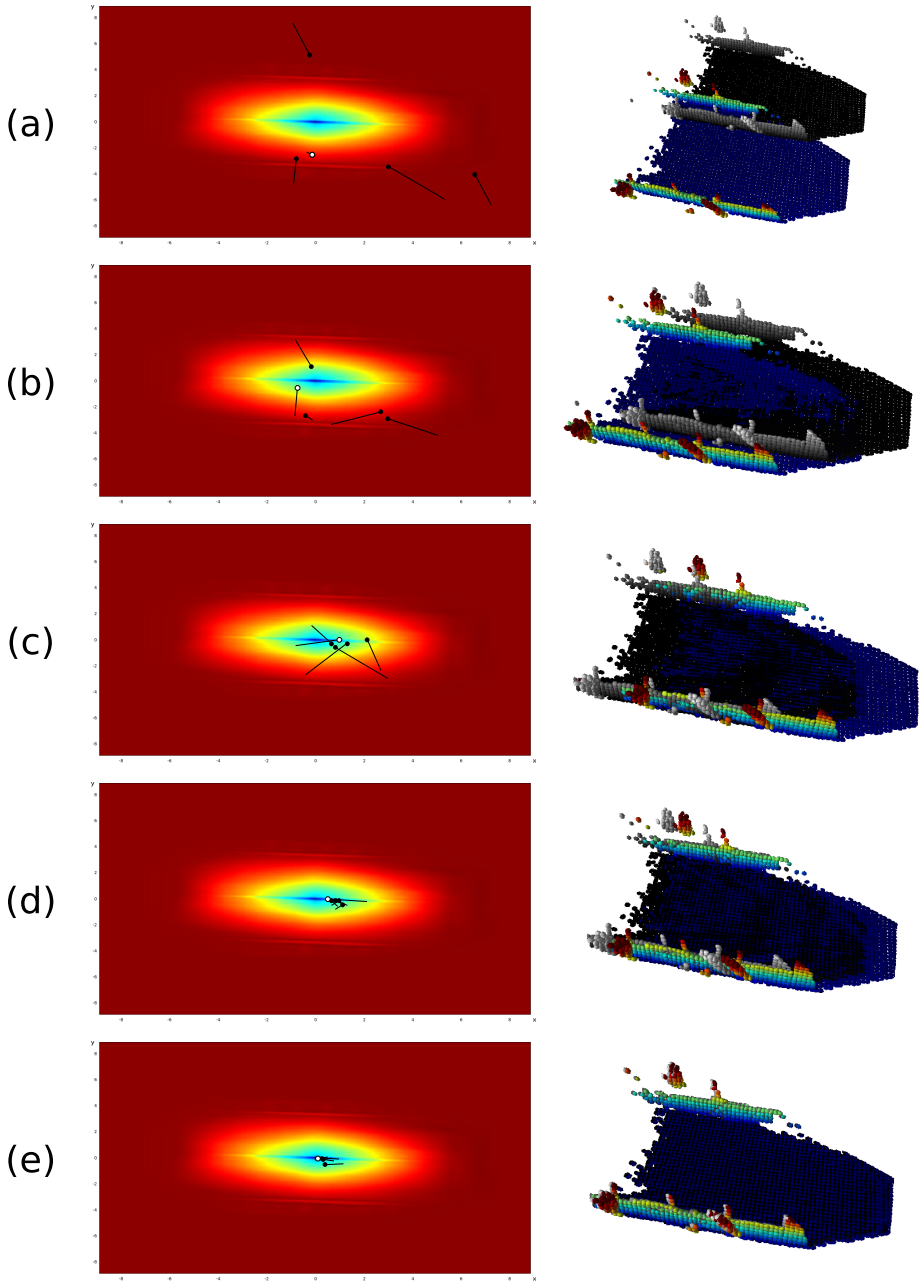


Abbildung 6.4: Beispiel einer realen Registrierung durch Partikelschwarmoptimierung mit fünf zufällig eingestreuten Partikeln und zehn Iterationen (aus Platzgründen werden nur fünf Iterationen dargestellt). Die Partikeln werden als schwarze Punkte zusammen mit ihrer jeweiligen Bewegungsvergangenheit dargestellt. Entdeckt ein Partikel eine neue global beste Position, wird diese in weiß markiert.

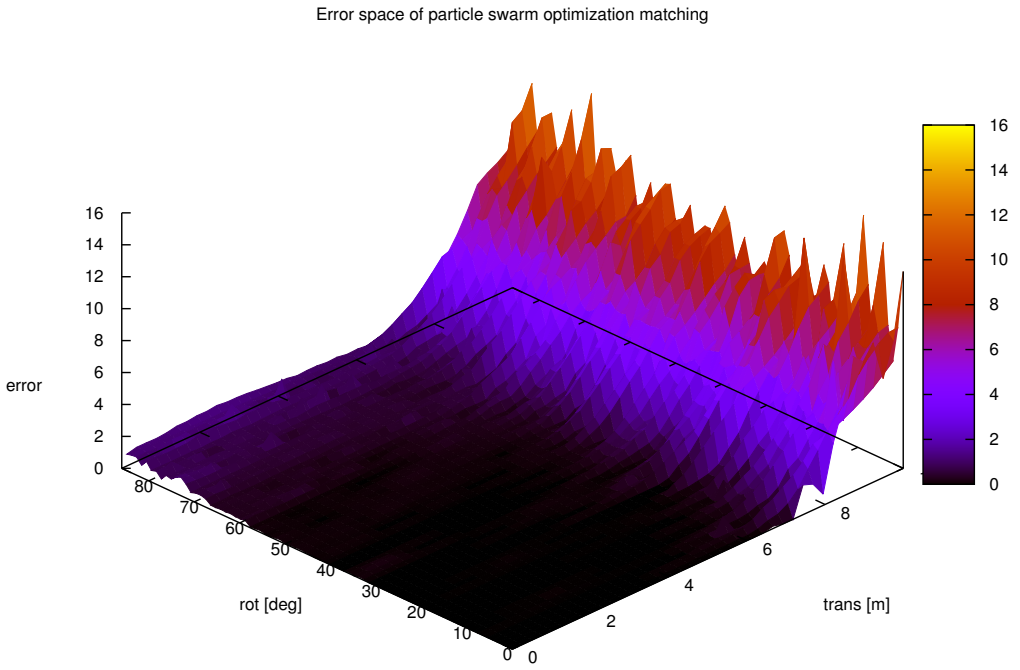


Abbildung 6.5: Veranschaulichung des Fehlers der Registrierung mit Partikelschwarmoptimierung für die jeweils vorgegebenen relativen Transformationen. Gezeigt wird das Fehlergebirge über der Abweichung zwischen ermittelter und perfekter relativer Transformation im Bereich $[0 \text{ m} \leq \sqrt{x^2 + y^2} \leq 10 \text{ m}]$ und $[0° \leq \phi_z \leq 86°]$. Die Farbgebung sowie Höhe beschreibt den durchschnittlichen summierten Fehler der Registrierung für die jeweilige Vorgabe.

Abfall der Erfolgsrate ab $xy = 8$ m entsteht durch die Einschränkung des Suchraums, der wie in Abbildung 6.4 auf $xy = 8$ m beschränkt wurde.

6.1.2.2 Laufzeit

Ein Nachteil des Partikelschwarmverfahrens ist der inhärent hohe Berechnungsaufwand durch die Berechnung der Distanzmetrik und des Gradienten für jede betrachtete Position $\mathbf{x}$. Abbildung 6.4 zeigt jedoch, dass die Partikelschwarmoptimierung auch mit wenigen Partikeln und einer geringen Iterationszahl konvergieren kann. Für schwerere Probleme sollten dennoch mehr Partikeln und Iterationen gewählt werden, um eine gute Konvergenzrate zu garantieren. Für die Abbildungen 6.5 und 6.6 wurde die Partikelschwarmoptimierung mit 100 Iterationen und 20 Partikeln ausgeführt. In dieser

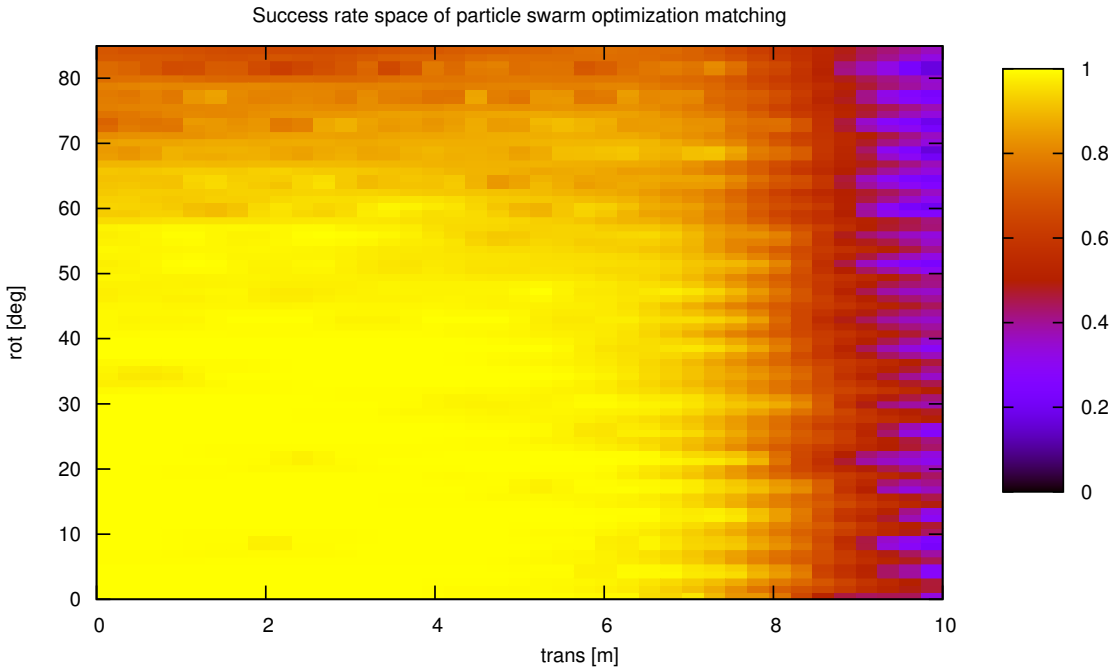


Abbildung 6.6: Veranschaulichung der Erfolgsrate der Registrierung mit Partikelschwarmoptimierung für die jeweils vorgegebenen relativen Transformationen. Die Farbgebung beschreibt die durchschnittliche Erfolgsrate der Registrierung für die Vorgaben im Bereich $[0\text{ m} \leq \sqrt{x^2+y^2} \leq 10\text{ m}]$ und $[0° \leq \phi_z \leq 86°]$. Eine Registrierung ist erfolgreich, wenn Rotations- und Translationsfehler kleiner als 3° bzw. 5 cm sind.

Konfiguration benötigte eine Registrierung im Durchschnitt 2449 ms mit einer Varianz von ±436 ms. Eine Iteration der Partikelschwarmoptimierung benötigte dabei im Durchschnitt 25 ms mit einer Varianz von ±4 ms. Da kein anderes Abbruchkriterium als die Zahl der Iterationen gegeben wurde, wurden stets die vollen 100 Iterationen durchgeführt, obwohl eine hinreichende Lösung meist schon nach einem Bruchteil der Iterationen gefunden wird. Mit einem geeigneten Abbruchkriterium hat folglich selbst die Registrierung mit Partikelschwarmoptimierung Potential zum echtzeitfähigen Einsatz.

6.1.2.3 Evaluation

Die Partikelschwarmoptimierung weist durch die Integration der Gradienteninformation eine hervorragende Konvergenzrate auf, bewahrt jedoch die inhärent gute Explorationsfähigkeit des Verfahrens zum Umgang mit großen Suchräumen. Für den realen Einsatz, beispielsweise zum Auflösen von Schleifenschlüssen in dieser Arbeit, muss ein Kompromiss zwischen Genauigkeit und Laufzeit getroffen werden. Wie in Beispiel 6.4 gezeigt, bietet die Partikelschwarmoptimierung jedoch auch mit schwächerer Parametrisierung ein gutes Leistungsverhalten, so dass beispielsweise ein Einsatz zum Schleifenschluss in echtzeitfähigen SLAM-Algorithmen möglich ist.

6.2 Leistungsverhalten des SLAM-Verfahrens

In diesem Abschnitt soll nun das Leistungsverhalten des entwickelten SLAM-Verfahrens evaluiert werden. Leider war bis zum aktuellen Zeitpunkt kein geeigneter Datensatz mit Ground Truth-Daten der Posen vorhanden, so dass keine metrische Evaluation der ermittelten Trajektorien durchgeführt werden konnte. Stattdessen soll die Performanz des Algorithmus' qualitativ anhand der erzeugten Karten gezeigt werden. Zur Evaluation wurden zwei Roboterplattformen (SCITOS G3 und SCITOS A5, jeweils MetraLabs GmbH) verwendet, die einen signifikanten Odometriefehler aufweisen. Es wurden vier Datensätze evaluiert:

1. **Kaffeeküche**: Drei aufeinanderfolgende Runden in einem Gang mit insgesamt 100 m Fahrstrecke.

2. **NI&KR**: Mehrere verschachtelte kleine Schleifen und eine große Schleife in den Gängen des Fachgebiets NI&KR mit einer Fahrstrecke von insgesamt 200 m.

3. **Labor**: Vier Runden durch das Robotiklabor des Fachgebiets NI&KR mit einer Fahrstrecke von insgesamt 60 m.

4. **Hufeisen**: Eine große Schleife und mehrere kleine Schleifen in zwei Büroflügeln des Zusebaus mit einer Fahrstrecke von insgesamt 375 m.

Im Folgenden sollen diese Datensätze durch den Vergleich zwischen unkorrigierter und korrigierter Karte evaluiert werden. Die korrigierte Karte wurde dabei mit dem in dieser Arbeit entwickelten SLAM-Verfahren erzeugt.

6.2.1 Datensatz Kaffeeküche

In diesem Datensatz wurden mit einem SCITOS G3 drei Runden um die Küchenzeile des Fachgebiets NI&KR mit einer Gesamtstrecke von 100 m gefahren. Abbildung 6.7 (a) zeigt die unkorrigierte Kartierung der gemessenen Tiefendaten, während Abbildung 6.7 (b) die von NDTSLAM erstellte Karte zeigt. Abbildung 6.8 zeigt diesen Vergleich aus zwei weiteren Perspektiven, sowie den Vergleich einer Detailaufnahme einer markanten Stelle. Das SLAM-Verfahren hat es erfolgreich geschafft, die während der drei Runden gemachten Beobachtungen der Gänge passend zu überlagern. Der Vergleich von Abbildung 6.8 (e) und (f) zeigt am Beispiel eines Tisches mit Stühlen, dass auch kleinere Details gut eingepasst werden.

6.2.2 Datensatz NI&KR

Der Datensatz „NI&KR“ wurde ebenfalls mit einem SCITOS G3 aufgenommen und demonstriert das Verhalten des SLAM-Verfahrens auf einem komplizierteren Datensatz. Über eine Gesamtstrecke von 200 m wurde der Flügel des Fachgebiets NI&KR kartiert, und dabei eine große Schleife mit mehreren verschachtelten kleinen Schleifen gefahren. Abbildung 6.9 zeigt den Vergleich zwischen der unkorrigierten und der korrigierten Karte für diesen Datensatz. Abbildungen 6.9 (b) und (d) zeigen die korrigierte Karte, während Abbildungen 6.9 (a) und (c) die unkorrigierte Karte aus denselben Perspektiven zeigen. Wie schon im Datensatz „Kaffeeküche“ konnten die Beobachtungen der verschiedenen Runden gut überlagert werden. Interessant ist vor allem, dass der lange Gang auf der rechten Seite der Karte 6.9 (b) trotz über 60 m Fahrt ohne Korrekturmöglichkeit bei Wiedereintritt in das bekannte Gebiet korrekt ausgerichtet wurde. Dies ist möglich, da die Kovarianzen der Registrierungs- und Odometriekanten der Knoten dieses Gangs die relative Ausrichtung korrekt widerspiegeln.

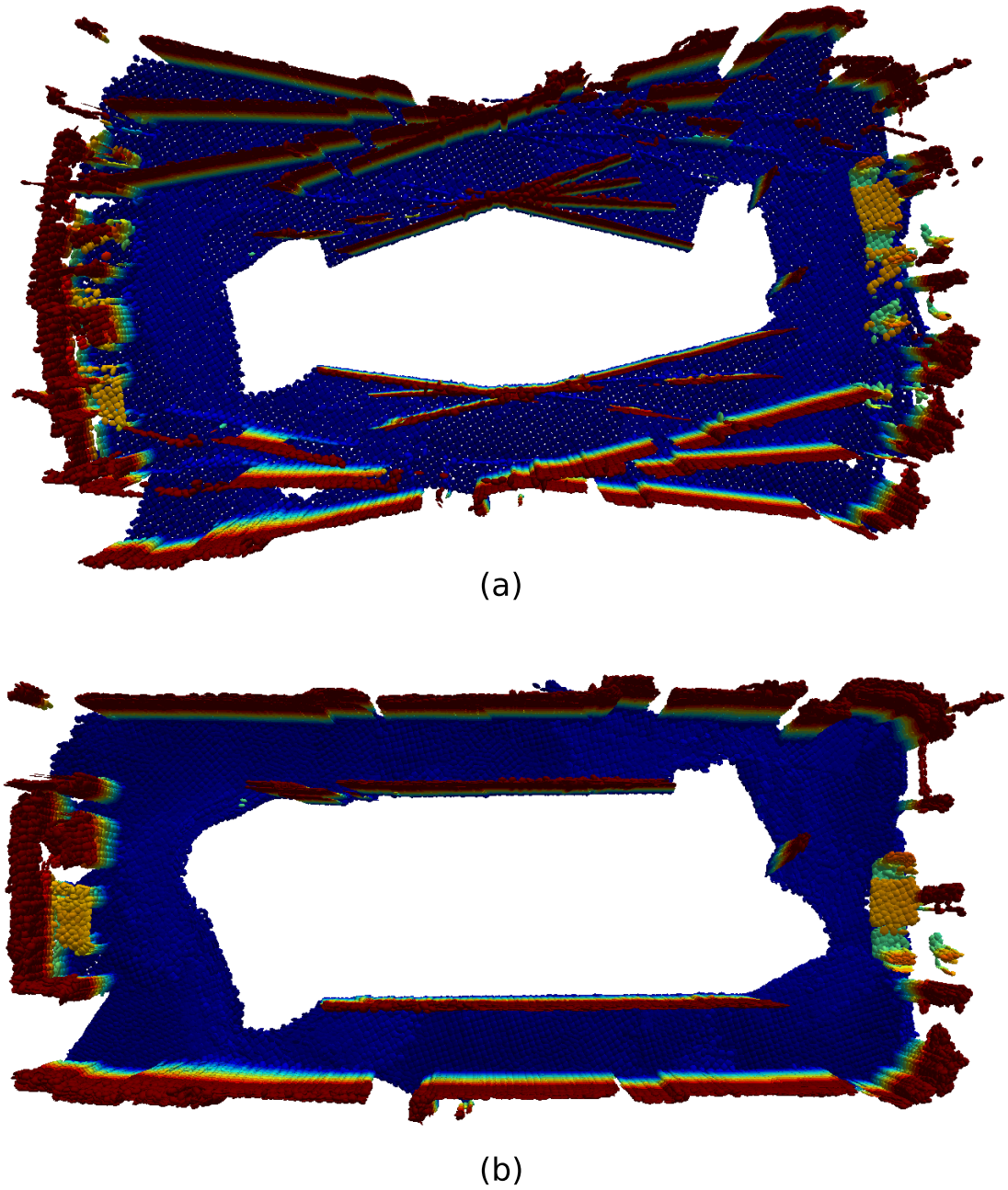


Abbildung 6.7: Vergleich der unkorrigierten Karte (a) mit der korrigierten Karte (b) für den Datensatz „Kaffeeküche“ aus einer orthogonalen Perspektive.

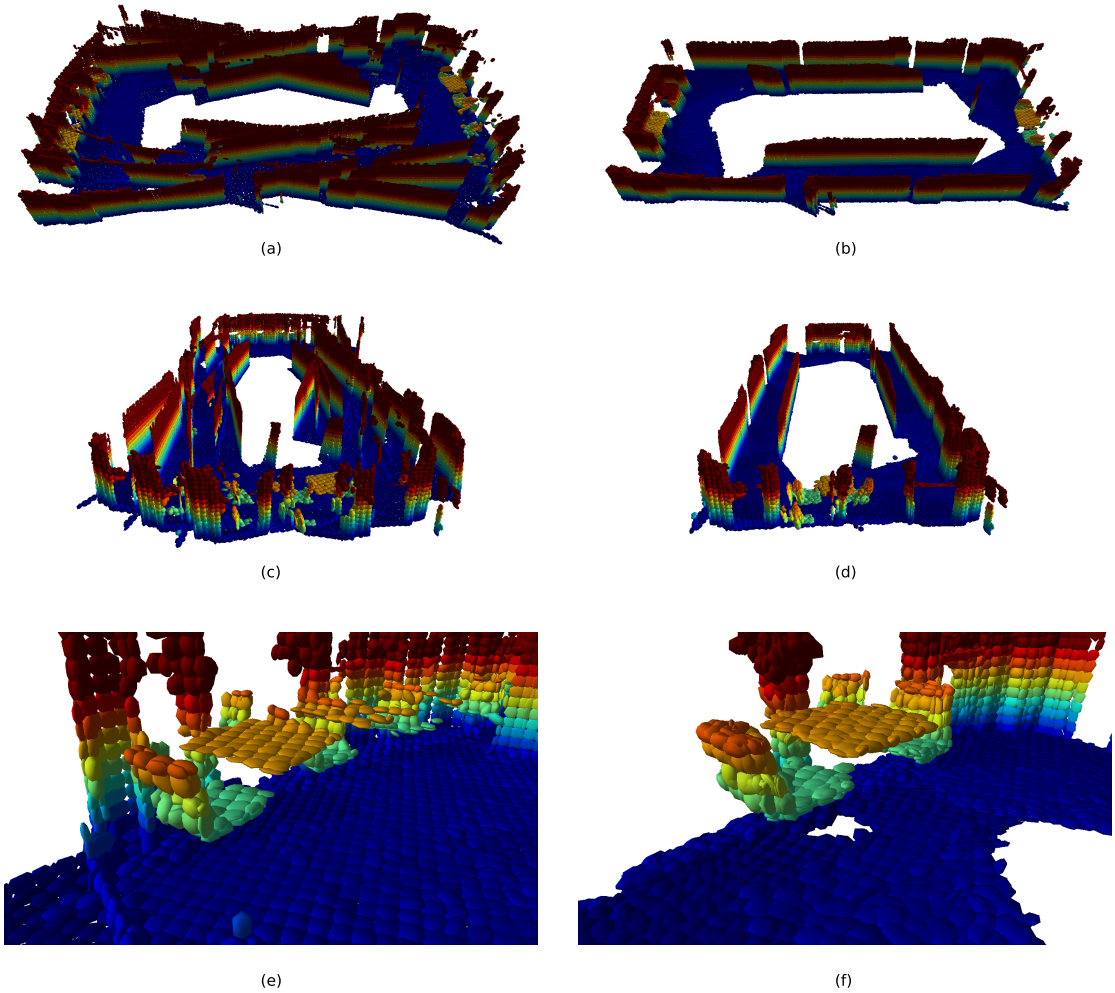


Abbildung 6.8: Weitere perspektivische Ansichten des Datensatzes „Kaffeeküche“. Abbildungen (a) und (c) zeigen die unkorrigierte Karte aus der Vogelperspektive, während Abbildung (e) eine Detailaufnahme eines markanten Punktes der unkorrigierten Karte zeigt. Abbildungen (b) und (d) zeigen zum Vergleich die korrigierte Karte aus den selben Perspektiven wie Abbildungen (a) und (c). Abbildung (f) zeigt dieselbe Detailaufnahme wie Abbildung (e), jedoch für die korrigierte Karte.

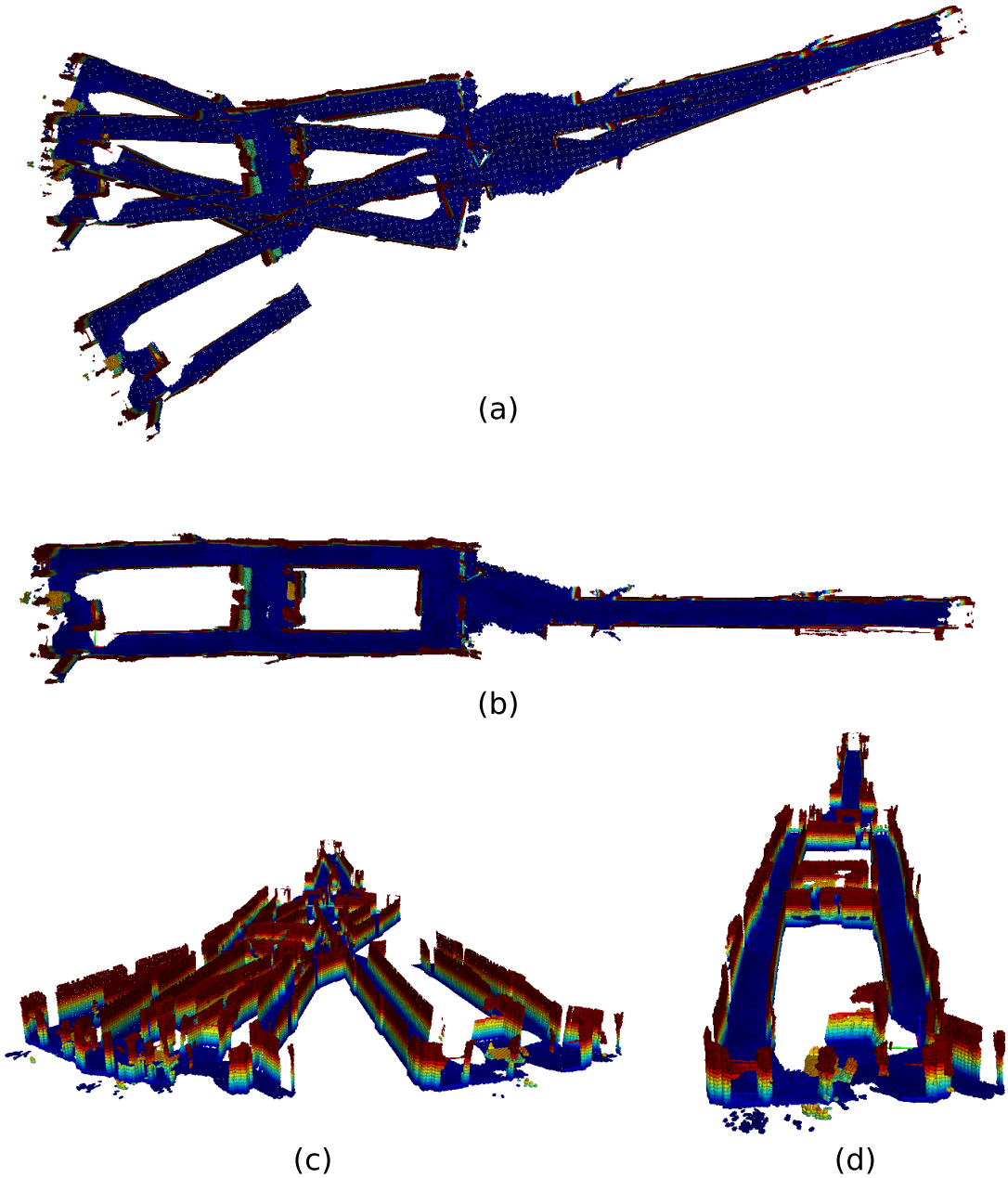


Abbildung 6.9: Vergleich der unkorrigierten Karte (a) und (c) mit der korrigierten Karte (b) und (d) für den Datensatz „NI&KR“ aus einer orthogonalen und einer schrägen Perspektive.

6.2.3 Datensatz Labor

Die beiden Datensätze „Kaffeeküche“ und „NI&KR“ demonstrieren die Leistungsfähigkeit des Algorithmus für ein typisches Bürogebäude-Szenario mit langen rechtwinkligen Gängen, und wenigen ungewöhnlichen Formen. Aus diesem Grund wurde der Datensatz „Labor“ im Robotiklabor des Fachgebiets NI&KR aufgenommen. In diesem Raum wird eine häusliche Umgebung nachgestellt, wie sie beispielsweise in einem Appartement vorgefunden werden kann. In dieser Umgebung wurden vier Runden mit einer Gesamtstrecke von 60 m mit einem SCITOS G3 gefahren, davon drei Mal gegen den Uhrzeigersinn und ein Mal im Uhrzeigersinn. Abbildung 6.10 zeigt den Vergleich der unkorrigierten mit der korrigierten Karte für diesen Datensatz. Abbildung 6.11 zeigt wieder zwei weitere Perspektiven und eine Detailaufnahme dieses Vergleichs.
Während für die vorherigen Datensätze anhand der Gänge schnell ersichtlich war, dass eine gute Korrektur stattgefunden hat, ist dies bei diesem Datensatz nicht direkt offensichtlich. Die im Vergleich zu den vorherigen Datensätzen „dicken“ Wände entstehen durch die höhere Aufnahmedistanz dieser Oberflächen. Durch die nicht-perfekte intrinsische Kalibrierung des Tiefensensors werden die stets relativ weit entfernten Wände leicht gekrümmt dargestellt, so dass es in der korrigierten Karte durch Überlagerung mehrere solcher Messungen zu den sichtbaren „Ausfransungen“ kommt (siehe Abbildung 6.10 (b)). In diesem Datensatz zeigt sich die Korrekturleistung des Verfahrens eher im Detail. Betrachtet man beispielsweise das Waschbecken in der oberen linken Ecke, die Sessel und den Tisch in der unteren linken Ecke, und vor allem das Sofa, den Couchtisch und den Stuhl in der unteren rechten Ecke (siehe Abbildungen 6.11 (e) und (f)), so zeigt sich die Korrekturleistung des Verfahrens deutlich. Das entwickelte SLAM-Verfahren bietet folglich auch für nicht-regelmäßige Umgebungen ein gutes Leistungsverhalten.

6.2.4 Datensatz Hufeisen

Der Datensatz „Hufeisen“ übertrifft die vorherigen Datensätze sowohl in der gefahrenen Distanz als auch in der Schwierigkeit für das SLAM-Verfahren. Über eine Distanz von 375 m wurde sowohl das Fachgebiet NI&KR als auch das Fachgebiet Telematik kartiert.

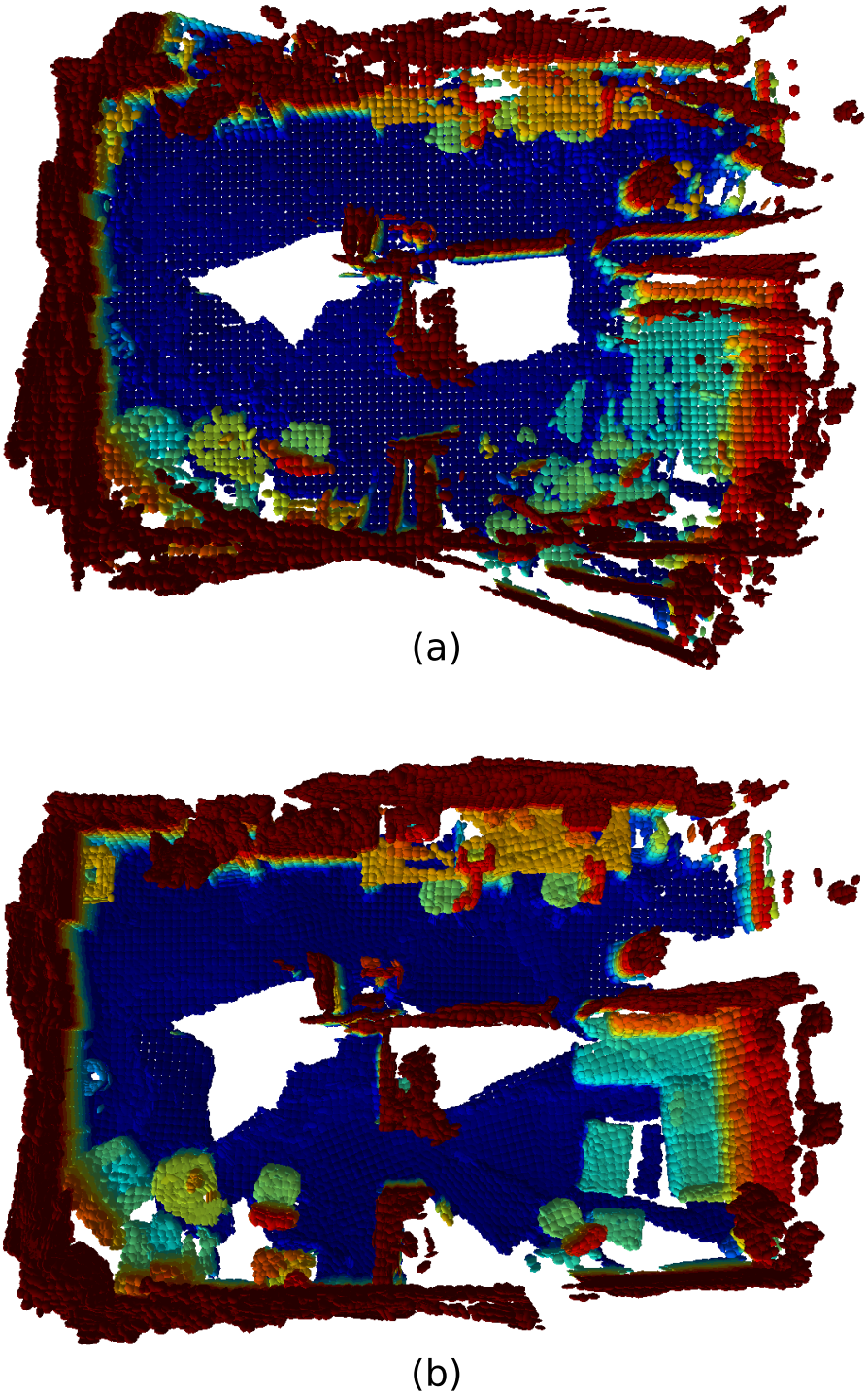


Abbildung 6.10: Vergleich der unkorrigierten Karte (a) mit der korrigierten Karte (b) für den Datensatz „Labor“ aus einer orthogonalen Perspektive.

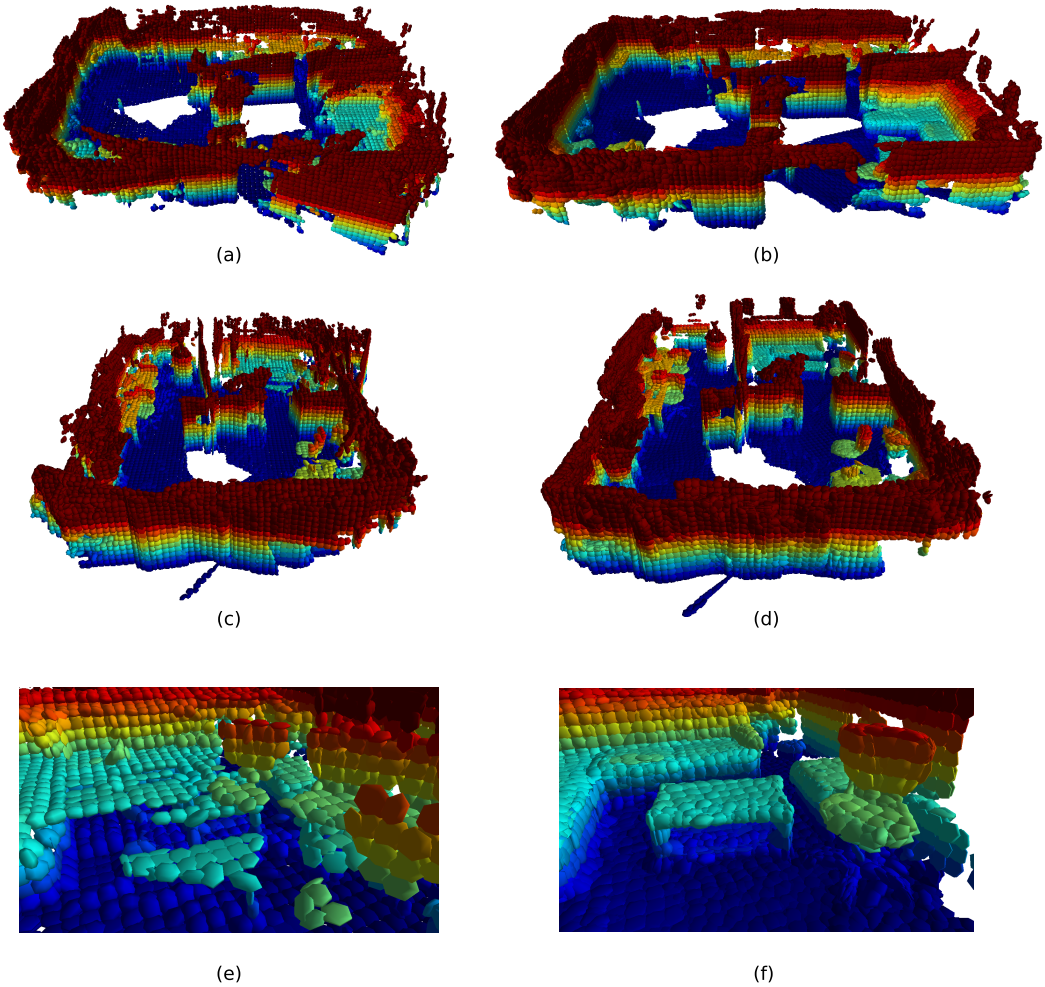


Abbildung 6.11: Weitere perspektivische Ansichten des Datensatzes „Labor“. Abbildungen (a) und (c) zeigen die unkorrigierte Karte aus der Vogelperspektive, während Abbildung (e) eine Detailaufnahme eines markanten Punktes der unkorrigierten Karte zeigt. Abbildungen (b) und (d) zeigen zum Vergleich die korrigierte Karte aus den selben Perspektiven wie Abbildungen (a) und (c). Abbildung (f) zeigt dieselbe Detailaufnahme wie Abbildung (e), jedoch für die korrigierte Karte.

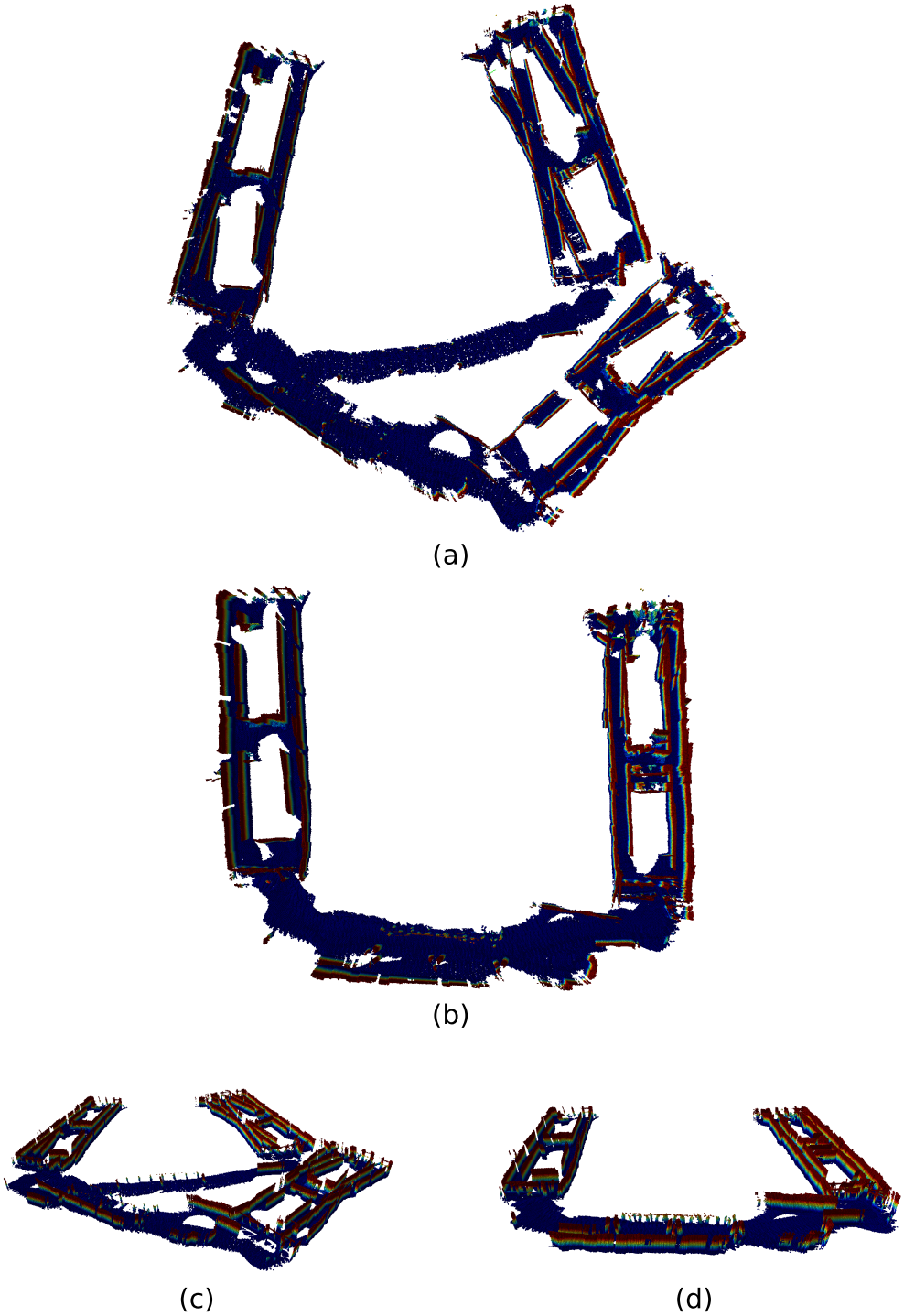


Abbildung 6.12: Vergleich der unkorrigierten Karte (a) und (c) mit der korrigierten Karte (b) und (d) für den Datensatz „Hufeisen“ aus einer orthogonalen und einer schrägen Perspektive.

Diese beiden Fachgebiete werden durch einen ca. 30 m langen Gang verbunden, für den durch den eingeschränkten Öffnungswinkel des Tiefensensors kaum relevante Details aufgenommen wurden. Abbildung 6.12 (a) veranschaulicht anhand der unkorrigierten Karte die Schwierigkeit dieses Datensatzes. Durch die fast 200 m lange Fahrt zur Kartierung des Fachgebiets Telematik wächst die Unsicherheit über die Roboterpose so sehr, dass bei Wiedereintritt in das bekannte Gebiet des Fachgebiets NI&KR nicht einmal mehr eine physische Überschneidung der Beobachtungen existiert. Dieser akkumulierte Fehler kann vom SLAM-Verfahren erst korrigiert werden, wenn wieder ein Eintritt in das bekannte Gebiet des Fachgebiets NI&KR stattfindet. Durch die hohe Unsicherheit stellt dies eine relativ schwere Operation dar. Dennoch zeigt Abbildung 6.12 (b) die relativ gut korrigierte Karte dieses Datensatzes. Die topologische Struktur der Karte ist korrekt, jedoch wurde durch die hohe Unsicherheit vor dem Wiedereintritt in das bekannte Gebiet eine Menge falscher Schleifenschlüsse etabliert, die durch die verbleibenden Beobachtungen nicht mehr alle korrigiert werden konnten. Abgesehen von diesem Fehler zeigt auch dieser Datensatz das gute Leistungsverhalten des entwickelten SLAM-Verfahrens, vor allem durch die korrekte orthogonale Ausrichtung der beiden kartierten Flügel.

6.3 Zusammenfassung

In diesem Kapitel wurden die Ergebnisse des in dieser Arbeit entwickelten SLAM-Verfahrens **NDTSLAM** vorgestellt und evaluiert. Es wurde gezeigt, dass die Registrierungsalgorithmen des Frontends für ihre jeweiligen Einsatzzwecke ein sehr gutes Leistungsverhalten aufweisen. Der Levenberg-Marquardt-Algorithmus ermöglicht eine robuste und sehr schnelle Registrierung für relative Transformationen, die den Rahmen des Odometriefehlers zwischen zwei Beobachtungen deutlich abdecken. Die Partikelschwarmoptimierung ermöglicht eine Registrierung selbst für sehr große relative Transformationen, bietet jedoch trotzdem eine gute Konvergenzrate und eine hohe Präzision. Sowohl typische Gebäude-Szenarien mit langen rechtwinkligen Gängen als auch unregelmäßige Umgebungen wie Appartements werden vom entwickelten SLAM-Verfahren robust verarbeitet. Die erzeugten Karten sind durchgehend konsistent, und

weisen auch im Detail eine meist hohe Korrektheit auf.

KAPITEL 7

Zusammenfassung und Ausblick

In dieser Arbeit wurde ein graphbasiertes SLAM-Verfahren vorgestellt, das mit effizient repräsentierten dreidimensionalen Tiefendaten arbeitet. Die graphbasierte Formulierung adressiert bekannte Defizite vieler filterbasierter SLAM-Algorithmen, beispielsweise die hohe Berechnungskomplexität bei einer großen Anzahl von Umgebungsmerkmalen oder etwaige Inkonsistenzen durch die zeitunabhängige rekursive Zustandsschätzung. Graphbasierte SLAM-Verfahren repräsentieren sämtliche Informationen über die Umgebung in einem topologisch korrekten Graphen aus Roboterposen und zugehörigen Beobachtungen. Dieser sog. Posengraph wird während der Erkundung der Umgebung kontinuierlich aufgebaut, und gegebenenfalls zur Reduktion der Unsicherheiten bezüglich der Roboterposen optimiert. Der Fokus dieser Arbeit lag auf dem Aufbau eines topologisch korrekten Posengraphen durch ein SLAM-Frontend, das an ein frei verfügbares SLAM-Backend angebunden werden sollte.

Datenstruktur und Registrierung. Das in dieser Arbeit entwickelte SLAM-Frontend nutzt den Normal Distributions Transform, um die mit einem Tiefensensor gemessenen Tiefendaten effizient darzustellen. Dabei wird eine Punktewolke in eine Menge räumlich disjunkter multivariater Normalverteilungen transformiert, die als stückweise stetige Wahrscheinlichkeitsverteilung die Struktur der gemessenen Oberfläche beschreibt. Auf zwei derartig dargestellten, aufeinanderfolgenden Beobachtungen wird ein Registrierungsalgorithmus durchgeführt, der die relative Transformation der Beobachtungen sowie die Unsicherheit des Ergebnisses bestimmen soll. Hierzu wird eine Mahalanobisdistanz-ähnliche Metrik genutzt, um die Bestimmung der relativen Transformation zweier Normal Distributions Transforms als Optimierungsproblem über

dieser Distanzmetrik zu formulieren. Zur Minimierung der Distanz zweier Beobachtungen wird der Levenberg-Marquardt-Algorithmus genutzt, der schnell und robust konvergiert, und zudem die analytische Berechnung der Unsicherheit des Ergebnisses ermöglicht. Dieser Algorithmus erfordert jedoch eine möglichst scharfe Startschätzung der vorliegenden relativen Transformation, die nicht immer vorhanden ist. Für diese Fälle wurde ein Registrierungsalgorithmus auf Basis der Partikelschwarmoptimierung entwickelt. Dieses Verfahren exploriert den Suchraum semi-deterministisch auf Basis einer schwarmähnlichen sozialen Bewegungsdynamik, um das Minimum der Distanzfunktion bezüglich der relativen Transformation zu finden. Die Partikelschwarmoptimierung konvergiert selbst auf großen Suchräumen robust, und ermöglicht durch die Abtastung des Parameterraums zudem die „kostenlose“ Berechnung der Unsicherheit durch die gewichtete Stichprobenvarianz aller betrachteten Hypothesen.

SLAM-Frontend und SLAM-Backend. Mit diesen Informationen wird der zunächst unkorrigierte Posengraph aufgebaut. Zu jedem diskreten Messzeitpunkt wird die aktuelle, von der Odometrie bestimmte Pose als Knoten in den Graphen eingefügt. Zwei aufeinanderfolgende Knoten werden stets durch eine Odometriekante und eine Registrierungskante verbunden, welche die jeweils bestimmten relativen Transformationen und deren Unsicherheiten beschreiben. Zur Reduktion der akkumulierten Posenunsicherheiten ist es erforderlich, Schleifenschlüsse zu entdecken. Das in dieser Arbeit vorgestellte Frontend bestimmt anhand der Posenunsicherheiten zweier Knoten, ob die zugehörigen Beobachtungen physisch überlappen könnten. Ist dies der Fall, wird eine Registrierung mit Partikelschwarmoptimierung durchgeführt, um die relative Transformation der Beobachtungen und die zugehörige Unsicherheit zu bestimmen. Für derartig entdeckte und gelöste Schleifenschlüsse wird die ermittelte relative Transformation und deren Unsicherheit als Schleifenschlusskante zwischen dem aktuellen Knoten und dem Schleifenschluss-Kandidaten in den Posengraph eingefügt. Schleifenschlüsse sorgen für die topologische Konsistenz des Posengraphen und ermöglichen durch Informationsredundanz eine Reduktion der Posenunsicherheiten, so dass nach jedem Schleifenschluss die Optimierung der Graphstruktur durch das freie SLAM-Backend *g2o* durchgeführt wird. Das Backend betrachtet die Optimierung des Posengraphen als Ausgleichspro-

blem zwischen aktueller Schätzung der Knotenkonfiguration und der „Vorhersage“ der Knotenposen durch die enthaltenen Kanten. Durch numerische Lösungsverfahren über einer geschickt konstruierten Mannigfaltigkeit wird der Posengraph bezüglich der Knotenposen optimiert. Eine Erweiterung des Lösungsverfahrens macht diese Optimierung zudem robust gegen topologische Inkonsistenzen im Posengraphen, so dass etwaige falsch-positive Schleifenschlüsse korrigiert werden können. Die anhand der optimierten Knotenkonfiguration bestimmte Information über die aktuelle Roboterpose wird an das Frontend zurückgegeben, um mit der bestmöglichen Lokalisation weiter zu erkunden.

Ergebnisse. Anhand von vier verschiedenen Datensätzen wurde das Leistungsverhalten des entwickelten SLAM-Verfahrens sowohl für typische Bürogebäude-Umgebungen als auch für häusliche Einsatzszenarien analysiert. Die erzeugten Karten weisen eine hohe topologische Konsistenz und Korrektheit auf, ebenso wie eine meist hohe Detailtreue. Die entwickelten Registrierungsverfahren zeichnen sich in ihren jeweiligen Einsatzgebieten ebenfalls durch ein sehr gutes Leistungsverhalten aus. Insgesamt entstand in dieser Arbeit ein graphbasiertes SLAM-Verfahren, das konsistente dreidimensionale Karten erzeugt, und durch effiziente Algorithmen zum echtzeitfähigen Einsatz geeignet ist.

7.1 Weiterführende Arbeiten

Aufgrund der begrenzten Bearbeitungszeit konnten einige Erweiterungen nicht umgesetzt werden, die das Leistungsverhalten des vorgestellten SLAM-Verfahrens potentiell steigern könnten. In diesem Abschnitt sollen die wichtigsten derartigen Erweiterungen, die während der Bearbeitung evaluiert wurden, kurz zusammengefasst werden.

7.1.1 Integration mehrerer NDT-Abstraktionsebenen

In Kapitel 3 wurde bereits beschrieben, dass bei der rekursiven Berechnung des Normal Distributions Transforms mehrere Auflösungsebenen ohne erheblichen Mehraufwand berechnet werden können (siehe Abbildung 3.11). Momentan werden diese verschiedenen Auflösungen hauptsächlich genutzt, um den Berechnungsaufwand der Distanzmetrik

für unterschiedliche Anforderungen zu skalieren. Die Registrierung zweier Beobachtungen wird beispielsweise auf der feinsten Auflösungsebene durchgeführt, während die Partikelschwarmoptimierung zur Berechnung auf eine gröbere, dafür performantere Auflösungsebene zurückgreift. Eine Erweiterung dieses Prinzips auf alle verwendeten Algorithmen würde sowohl deren Geschwindigkeit als auch deren Robustheit verbessern. Die Registrierung könnte zunächst auf einer hohen Auflösungsebene starten, bis eine gewisse Varianzschwelle unterschritten wird. Zu diesem Zeitpunkt wird die Registrierung mit dieser Zwischenlösung und einer feineren Auflösung fortgesetzt, bis ein globales Konvergenzkriterium unterschritten wird (siehe [Ulaş und Temeltaş, 2012] für einen derartigen Ansatz). Ein solcher Algorithmus würde eine grobe, aber schnelle Einpassung der Beobachtungen erreichen, die dann kontinuierlich verfeinert wird. Dies würde potentiell sowohl die Robustheit gegen Ausreißer als auch die Laufzeit des Verfahrens verbessern. Die Partikelschwarmoptimierung könnte ebenfalls von mehreren Auflösungsebenen profitieren, indem die Partikelpopulation hierarchisch gespalten wird. Ein Teil der Partikelpopulation würde den Suchraum in einer groben, dafür performanten Auflösungsebene durchsuchen. Jedes dieser Partikel besitzt eine ortsabhängige Anzahl Kinder, die den lokalen Teilraum in einer feineren Auflösung durchsuchen. Durch eine solche Kaskade könnte das globale Extremum schneller gefunden werden, ohne die Genauigkeit der Lösung zu opfern.

7.1.2 Akkurate Schätzung der Posenunsicherheit

In Kapitel 5 wurde eine Möglichkeit vorgestellt, auf Basis der Unsicherheiten der Knotenposen potentielle Schleifenschluss-Kandidaten zu bestimmen. Dabei wurde auf das Tracking der aktuellen Posenunsicherheit eingegangen, die dieses Verfahren überhaupt erst ermöglicht. Momentan wird diese Unsicherheit jedoch nur über die aktuell akkumulierte Odometrieunsicherheit, eventuell korrigiert durch das SLAM-Backend, geschätzt. Dies stellt eine zu pessimistische Schätzung dar, so dass der Schleifenschlussalgorithmus unbeteiligte Knoten potentiell als Schleifenschluss-Kandidaten identifizieren könnte. Ein explizites Tracking der aktuellen Posenunsicherheit durch einen Kalman-Filter würde beispielsweise erlauben, die durch die Registrierung ermittelten Unsicherheiten mit

den Odometrieunsicherheiten zu einer genaueren Schätzung zu verrechnen. Mit dieser akkuraten Schätzung der Posenunsicherheit könnte potentiell die Zahl der falschen Schleifenschlüsse reduziert werden.

7.1.3 Zusammenführung redundanter Beobachtungen

Ein Problem potentiell echtzeitfähiger graphbasierter SLAM-Verfahren ist die kontinuierlich wachsende Anzahl von Beobachtungen, aus denen sich die geschätzte globale Umgebungskarte zusammensetzt. Wie bereits erwähnt, sind Schleifenschlüsse unentbehrlich für die topologische Konsistenz und Korrektheit des Posengraphen und somit der geschätzten Umgebungskarte. Es ist daher unvermeidbar, dass physische Oberflächenstrukturen (beispielsweise an einer viel befahrenen Kreuzung) mehrfach gemessen werden. Aufgrund von Störfaktoren wie Sensorrauschen oder variablen Messabständen bei nicht optimaler Sensorkalibrierung ist keine dieser Beobachtungen identisch. Es gibt folglich mit zunehmender Fahrtzeit immer mehr Stellen im Posengraphen, an denen mehrere Knoten mit leicht unterschiedlichen Beobachtungen durch Schleifenschlüsse eng zusammengezogen liegen. Dies führt sowohl zu unschönen Artefakten in den fertigen Karten, als auch zu erhöhtem Berechnungsaufwand. Problematisch ist hierbei vor allem die Tatsache, dass an solchen Stellen alle diese beinahe identischen Beobachtungen als Schleifenschluss-Kandidaten in Frage kommen, und folglich mit der aktuellen Beobachtung registriert werden müssen. Zur Reduktion des Berechnungsaufwands ist es daher vorteilhaft, diese redundanten Beobachtungen und deren Knoten im Posengraphen nach geeigneten Kriterien zu vereinen. Ein geeignetes Verfahren würde beispielsweise den Informationsverlust der Zusammenführung zweier Beobachtungen berechnen und nur diejenigen Beobachtungen zusammenführen, die zum geringsten Informationsverlust führen würden. Gleichermaßen kann mithilfe dieses Informationswerts die erhöhte Unsicherheit des zusammengeführten Knotens berechnet werden.

KAPITEL A

Ergänzende Unterlagen

A.1 Levenberg-Marquardt-Algorithmus

Der Levenberg-Marquardt-Algorithmus ist eine Kombination des Gauß-Newton- und des Gradientenabstiegsverfahrens, zwischen denen fließend gewechselt werden kann. In der Nähe des Minimums der Zielfunktion ähnelt der Algorithmus im Verhalten dem Gauß-Newton-Verfahren, während er sich weiter entfernt mehr wie das Gradientenabstiegsverfahren verhält [Press et al., 1992].

A.1.1 Gauß-Newton-Verfahren

Das Gauß-Newton-Verfahren überführt ein nichtlineares *least squares*-Problem durch lokale Linearisierung der Zielfunktion in ein lineares Ausgleichsproblem, welches mit dem regulären Newton-Verfahren gelöst werden kann. Man betrachte zuerst, ohne dies näher auszuführen, die Iterationsvorschrift des Gauß-Newton-Verfahrens für den aktuell geschätzten Parametervektor $\mathbf{p}_t$:

$$\mathbf{p}_t = \mathbf{p}_{t-1} - \left(J_t^T J_t\right)^{-1} J_t^T \mathbf{r}_t \tag{A.1}$$

Der Vektor $\mathbf{r}_t$ entspricht hier den Residuen zwischen der Modellfunktion $f(\mathbf{x}, \mathbf{p}_t)$ und den Messwerten $\mathbf{y}$:

$$\mathbf{r}_t = \left(\left(f\left(x_1, \mathbf{p}_t\right) - y_1\right), \ldots, \left(f\left(x_m, \mathbf{p}_t\right) - y_m\right)\right)^T \tag{A.2}$$

Der Ausdruck $\left(J_t^T J_t\right)$ steht für die approximierte Hessematrix der Modellfunktion. Diese Approximation wird häufig verwendet, da das Berechnen der partiellen Ablei-

tungen zweiter Ordnung meist sehr aufwändig ist. Newton-Verfahren mit derartiger Approximation heißen *quasi*-Newton-Verfahren.

A.1.2 Levenberg-Marquardt-Algorithmus

Das Gauß-Newton-Verfahren wurde durch Levenberg und Marquardt um einen Regulierungsfaktor λ erweitert, der für die hohe Robustheit des Verfahrens verantwortlich ist. Die Iterationsvorschrift für den Parameter $\mathbf{p}_t$ lautet wie folgt:

$$\mathbf{p}_t = \mathbf{p}_{t-1} + \Delta\mathbf{p}_t \tag{A.3}$$

$$\Delta\mathbf{p}_t = -\left(J_t^T J_t + \lambda I\right)^{-1} J_t^T \mathbf{r}_t \tag{A.4}$$

Für große Werte λ wird der Term $\left(J_t^T J_t + \lambda I\right)^{-1}$ zunehmend kleiner und unabhängiger vom Einfluss der approximierten Hessematrix $J_t^T J_t$, da

$$\lambda I \gg J_t^T J_t \tag{A.5}$$

gilt. Der Levenberg-Marquardt-Algorithmus macht in diesem Fall also einen kleinen Schritt in Richtung des Gradientenabstiegs. Für kleine Werte λ gilt das Umgekehrte - der Einfluss des Regulierungsfaktors wird verschwindend gering, so dass sich die Iterationsvorschrift dem Verhalten des Gauß-Newton-Verfahrens in Gleichung A.1 annähert. Zur Wahl des Regulierungsfaktors existieren viele Möglichkeiten und Ansätze. Eine häufig verwendete Methode wird in der Beschreibung des Levenberg-Marquardt-Verfahrens in Algorithmus 7 vorgestellt. Ausgehend von einem Startwert wird nach jedem Iterationsschritt geprüft, ob sich das Residuum verschlechtert hat. Wenn ja, wird der Schritt rückgängig gemacht und λ vergrößert. Dies bewirkt, dass der Algorithmus bei schlechtem Leistungsverhalten des Gauß-Newton-Verfahrens fließend zum Gradientenabstiegsverfahren wechselt. Umgekehrt wird bei vermuteter Annäherung an das Minimum verstärkt das Gauß-Newton-Verfahren genutzt, um nicht über das Minimum hinauszuspringen.

Input : Modellfunktion $f(\mathbf{p}, \mathbf{x})$, Messwerte $\mathbf{y}$, Startwert $\mathbf{p}_0$

Result : Geschätzter Parametervektor $\mathbf{p}_t$

Initialisiere λ auf einen Startwert, beispielsweise $\lambda = 0.01$;

for *Alle Zeitschritte* t **do**

  Berechne Residuen $\mathbf{r_t} = \{f(\mathbf{p}, \mathbf{x}) - \mathbf{y}\}$;

  Berechne Jacobimatrix J_t;

  `// Iterationsschritt;`

  $\mathbf{p}_t = \mathbf{p}_{t-1} - \left(J_t^T J_t + \lambda I\right)^{-1} J_t^T \mathbf{r}$;

  if $\mathbf{r}_t > \mathbf{r}_{t-1}$ **then**

    `// Ziehe schlechten Schritt zurück;`

    $\mathbf{p}_t = \mathbf{p}_{t-1}$;

    `// Verschiebe Dynamik in Richtung Gradientenabstieg;`

    $\lambda = 10 \cdot \lambda$;

  else

    `// Verschiebe Dynamik in Richtung Gauß-Newton-Verfahren;`

    $\lambda = \frac{\lambda}{10}$;

  end

  Überprüfe Faktoren t, r_t und λ auf Abbruchkriterien;

end

Algorithmus 7: Levenberg-Marquardt-Algorithmus.

A.2 Partikelschwarmoptimierung

Die Partikelschwarmoptimierung wurde 1995 von Kennedy und Eberhardt als Verfahren zur Optimierung nichtlinearer Funktionen entwickelt, für die einzig die Funktionswerte berechnet werden können [Kennedy und Eberhart, 1995]. Die Grundlage des Verfahrens ist die Simulation der Schwarmbewegung von Tieren, die als soziales Gefüge gemeinsam nach Ressourcen wie Futter oder Unterschlupf suchen. Derartige Schwärme besitzen Methoden, um das individuelle Wissen über diese Ressourcen mit dem ganzen Schwarm zu teilen. Jedes Individuum verfolgt eine Explorationsstrategie,

Position $\mathbf{x}_t^i \in \mathbb{R}^n$	Aktuelle Position des Partikels im Suchraum.
Geschwindigkeit $\mathbf{v}_t^i \in \mathbb{R}^n$	Bewegungsrichtung und -geschwindigkeit des Partikels.
Eigenwissen $\mathbf{p}_t^i \in \mathbb{R}^n$	Beste persönlich gefundene Position bezüglich $f(\mathbf{x})$.
Schwarmwissen $\mathbf{g}_t \in \mathbb{R}^n$	Beste vom Schwarm gefundene Position bezüglich $f(\mathbf{x})$.

Tabelle A.1: Zustand eines Partikels i zum Zeitpunkt t.

wird jedoch durch die eigene Gier früher oder später zur besten bekannten Ressource gezogen. Mathematisch gesehen sucht der Schwarm das Optimum der meist nichtlinearen Ressourcenverteilungsfunktion, weshalb Kennedy und Eberhardt die Simulation des Schwarmverhaltens als Grundlage eines Verfahrens zur Optimierung nichtlinearer Funktion verwendeten.

A.2.1 Eigenschaften des Schwarms

Die Partikelschwarmoptimierung nach Kennedy und Eberhardt ist ein inkrementelles Verfahren zur Minimierung einer Funktion $f(x) \in \mathbb{R}$ durch einem Schwarm S aus sog. Partikeln. Der Zustand eines Partikels besteht aus vier Komponenten, die in Tabelle A.1 beschrieben sind. Somit besitzt jedes Partikel i eine Position $\mathbf{x}_t^i$ und eine Bewegungsrichtung $\mathbf{v}_t^i$, sowie eine Erinnerung über die bisher beste persönlich gefundene Position $\mathbf{p}_t^i$. Zusätzlich kennt jedes Partikel die bisher global beste Position $\mathbf{g}_t$ des Schwarms. Zum Zeitpunkt t gilt für alle Partikeln $i \in S$:

$$\mathbf{p}_t^i = \underset{\mathbf{x}_k^i}{\operatorname{argmin}}\, f\left(\mathbf{x}_k^i\right) \qquad k < t \tag{A.6}$$

$$\mathbf{g}_t = \underset{\mathbf{p}_t^i}{\operatorname{argmin}}\, f\left(\mathbf{p}_t^i\right) \qquad \forall i \in S \tag{A.7}$$

Gleichung (A.6) zeigt, dass das Eigenwissen $\mathbf{p}_t^i$ eines Partikels i die Position in dessen persönlicher Vergangenheit $(\mathbf{x}_0^i, \ldots, \mathbf{x}_t^i)$ mit dem geringsten Funktionswert ist. Gleichermaßen wird das Schwarmwissen $\mathbf{g}_t$ als das beste Eigenwissen $\mathbf{p}_t^i$ aller Partikeln definiert, und beschreibt somit die absolut beste bisher entdeckte Position. Auf Basis dieser Eigenschaften lässt sich nun die Schwarmbewegung definieren.

A.2.2 Bewegungsdynamik des Schwarms

Aus ihren Beobachtungen schlossen Kennedy und Eberhardt, dass ein Ressourcen-optimierendes Bewegungsmodell ab einem gewissen Zeitpunkt einen Anführer benötigt, der die restlichen Individuen zu einer vielversprechenden Ressourcenquelle führt. Um eine gewisse Bewegungsdynamik zu wahren, muss jedes Individuum auch selbst Entscheidungen basierend auf der persönlichen Erfahrung treffen. Die Bewegung eines Partikels wird daher sowohl von der global besten Position $\mathbf{g}_t$, als auch von der bisher persönlich besten Position $\mathbf{p}_t^i$ beeinflusst. Um eine physikalisch plausible Bewegungsdynamik zu erreichen, unterliegt jedes Partikel zudem der Trägheit durch die aktuelle Geschwindigkeit $\mathbf{v}_t^i$. Unter diesen Gesichtspunkten ergeben sich die Berechnungsvorschriften der neuen Geschwindigkeit $\mathbf{v}_{t+1}^i$ und Position $\mathbf{x}_{t+1}^i$ der Partikeln:

$$\mathbf{v}_{t+1}^i = \underbrace{\omega \mathbf{v}_t^i}_{\text{Trägheit}} + \underbrace{c_1 r_1 \left(\mathbf{p}_t^i - \mathbf{x}_t^i\right)}_{\text{Kognitive Komponente}} + \underbrace{c_2 r_2 \left(\mathbf{g}_t - \mathbf{x}_t^i\right)}_{\text{Soziale Komponente}} \qquad r_1, r_2 \sim U(0,1) \tag{A.8}$$

$$\mathbf{x}_{t+1}^i = \mathbf{x}_t^i + \mathbf{v}_{t+1}^i \tag{A.9}$$

Die neue Geschwindigkeit und somit Bewegungsrichtung eines Partikels setzt sich also aus drei Komponenten zusammen (Gleichung (A.8)) [Van Den Bergh, 2002]. Die Trägheit beschreibt den Einfluss der vorherigen Geschwindigkeit anhand des Trägheitskoeffizienten ω und verhindert somit unnatürliche Richtungswechsel. Die kognitive Komponente mit Koeffizient c_1 beschreibt den Einfluss der bisher besten entdeckten Position auf die Bewegungsrichtung, während die soziale Komponente mit Koeffizient c_2 den Einfluss der global besten Position beschreibt. Die gleichverteilten Zufallszahlen r_1 und r_2 sorgen für eine zufällige Gewichtung der Komponenten, so dass trotz klarer Richtungsvorgaben eine hohe Bewegungsdynamik gewährleistet ist. Abbildung A.1 illustriert diese Berechnungsvorschrift anhand zweier Partikeln.
Die neue Position $\mathbf{x}_{t+1}^i$ eines Partikels ergibt sich nun als Summe der letzten Position und der neuen Geschwindigkeit (Gleichung (A.9)). Iteriert man diese Gleichungen für zufällig initialisierte Startwerte und breit eingestreute Partikeln, so erhält man den Partikelschwarmalgorithmus zur nichtlinearen Optimierung. Algorithmus 8 beschreibt das Verfahren anhand der Minimierung einer nichtlinearen Kostenfunktion $f(\mathbf{x})$.

Data : Zielfunktion $f(\mathbf{x}) \in \mathbb{R}$, Partikelschwarm S, Suchraum A

Result : Position $\mathbf{g}_t \in \mathbb{R}^n$

```
1  for Alle Partikeln i ∈ S do    // Initialisierung der Partikeln für t = 0
2      x_0^i = Zufällige Position in A;
3      v_0^i = Zufällige Geschwindigkeit in [−v_max, v_max];
4      p_0^i = x_0^i, da dies bisher die einzige bekannte Position ist;
5      g_0 = Position mit dem bisher geringsten Funktionswert f(p_0^i);
6  end
7  for Iteration t + 1 do                    // Iteration des Algorithmus'
8      for Alle Partikeln i ∈ S do           // Update für jeden Partikel
9          r_1, r_2 ~ U(0, 1);
10         v_{t+1}^i = ωv_t^i + c_1 r_1 (p_t^i − x_t^i) + c_2 r_2 (g_t − x_t^i);
11         x_{t+1}^i = x_t^i + v_{t+1}^i;
12         if f(x_{t+1}^i) < f(p_t^i) then   // Update des Eigenwissens
13             p_{t+1}^i = x_{t+1}^i;
14             if f(p_{t+1}^i) < f(g_t) then // Update des Schwarmwissens
15                 g_{t+1} = p_{t+1}^i;
16             end
17         end
18     end
19 end
```

Algorithmus 8: Minimierung der Funktion $f(\mathbf{x})$ über dem Suchraum A durch Partikelschwarmoptimierung.

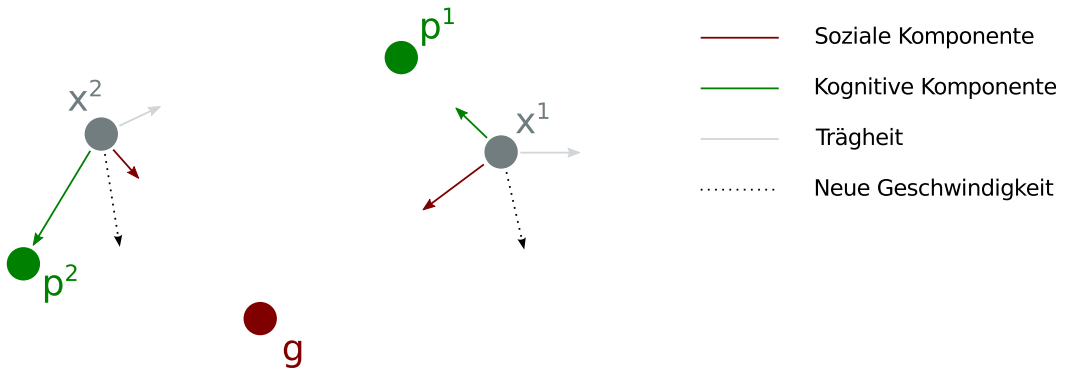


Abbildung A.1: Veranschaulichung der Bewegungsdynamik des Partikelschwarms, zur Übersichtlichkeit ohne Zeitindizes und nur für zwei Partikeln. Die momentane Bewegungsrichtung und Geschwindigkeit $\mathbf{v}_t^i$ der Partikeln wird durch graue Vektoren angegeben, während die Einflüsse durch das Eigenwissen $\mathbf{p}_t^i$ sowie das Schwarmwissen $\mathbf{g}_t$ als grüne bzw. rote Vektoren dargestellt werden. Die neue Geschwindigkeit $\mathbf{v}_{t+1}^i$ ergibt sich nach Gleichung (A.8) aus diesen drei Faktoren. Man beachte die zufällig bestimmte Gewichtung der kognitiven und sozialen Komponente der Partikeln 1 und 2. Während Partikel 1 stark zum globalen Optimum gezogen wird, erkundet Partikel 2 eher seinen lokalen Suchraum.

A.2.3 Evaluation

Die Partikelschwarmoptimierung stellt sehr geringe Anforderungen an das zu lösende Problem. Im Vergleich zu anderen Lösungsverfahren benötigt der Algorithmus keine analytisch oder numerisch berechneten Ableitungen, sondern lediglich die Möglichkeit zur Berechnung von Funktionswerten. Die simple Berechnungsvorschrift ermöglicht zudem eine einfache Implementierung, deren Berechnungsaufwand sich beinahe komplett auf die Funktionsauswertungen $f(\mathbf{x})$ beschränkt. Kennedy und Eberhardt bemerken zudem die gute Balance zwischen Erkundung des Suchraums (*exploration*) und Verfeinerung der aktuellen Lösung (*exploitation*), die zu einer guten Konvergenzrate führt. Durch die nichtdeterministische Natur der Erkundung konvergiert die Partikelschwarmoptimierung jedoch langsamer als deterministische Verfahren, zudem lässt sich die Konvergenz gegen globale oder auch lokale Extrema in der unmodifizierten Variante nicht beweisen [Van Den Bergh, 2002].

A.2.3.1 Konvergenzbetrachtung

Van den Bergh beschreibt in seiner Dissertation eine Eigenart der Berechnungsvorschrift, durch die die Partikelschwarmoptimierung zur vorzeitigen Konvergenz tendieren kann. Für das beste Partikel j stimmen Schwarmwissen $\mathbf{g}_t$ und Eigenwissen $\mathbf{p}_t^j$ überein, so dass dieses Partikel nach Gleichung (A.8) gegen g konvergiert. Durch Oszillation um diesen Punkt verringert sich dessen Geschwindigkeit $\mathbf{v}_t^j$ immer mehr, bis das Partikel einen infinitesimal kleinen Abstand zu $\mathbf{g}_t$ hat. Zu diesem Zeitpunkt gilt:

$$\mathbf{v}_t^j \approx 0 \tag{A.10}$$

$$\mathbf{x}_t^j = \mathbf{p}_t^j = \mathbf{g}_t \tag{A.11}$$

Daraus ergibt sich die Berechnungsvorschrift für Zeitpunkt $(t+1)$:

$$\begin{aligned} \mathbf{v}_{t+1}^j &= \omega \mathbf{v}_t^j + c_1 r_1 \left(\mathbf{p}_t^j - \mathbf{x}_t^j\right) + c_2 r_2 \left(\mathbf{g}_t - \mathbf{x}_t^j\right) \\ &= \omega 0 + c_1 r_1 0 + c_2 r_2 0 = 0 \\ \mathbf{x}_{t+1}^j &= \mathbf{x}_t^j + \mathbf{v}_{t+1}^j = \mathbf{x}_t^j \end{aligned} \tag{A.12}$$

In diesem Zustand bewegt sich das global beste Partikel nicht mehr, und der Schwarm konvergiert auf diese Position, solange auf dem Weg keine bessere Position gefunden wird. Ist der gesamte Schwarm auf eine kleine Stelle konvergiert, spricht man von Stagnation. In diesem Zustand findet keine Erkundung des Suchraums mehr statt, so dass keine bessere Lösung gefunden werden kann. Die Stelle der Konvergenz ist lediglich die beste bisher gefundene Position im Suchraum, und nicht notwendigerweise ein lokales oder globales Minimum. Um diese gravierende Einschränkung zu beheben, werden in der Literatur einige Erweiterungen vorgeschlagen.

A.2.4 Erweiterungen

In diesem Abschnitt soll zuerst einer der wichtigsten Beiträge zur lokalen Konvergenz beschrieben werden, und dann einige im Rahmen dieser Arbeit evaluierte Erweiterungen zur globalen Konvergenz vorgestellt werden.

A.2.4.1 Guaranteed Convergence PSO

Als Antwort auf das Problem der Stagnation entwarf Van den Bergh die Modifikation *Guaranteed Convergence Particle Swarm Optimization* (GCPSO), die die Konvergenz auf ein lokales Minimum garantiert [Van Den Bergh, 2002]. Die Neuerung dieser Methode ist eine separate Berechnungsvorschrift für das beste Partikel, um die Stagnation zu verhindern. Für das beste Partikel j gilt folgende Berechnungsvorschrift:

$$\mathbf{v}_{t+1}^j = \underbrace{-\mathbf{x}_t^j + \mathbf{g}_t}_{\text{Reset}} + \underbrace{\omega \mathbf{v}_t^j}_{\text{Suchrichtung}} + \underbrace{\rho_t(1-2r_1)}_{\text{Zufälliger Suchraum}} \qquad r_1 \sim U(0,1) \tag{A.13}$$

$$\begin{aligned} \mathbf{x}_{t+1}^j &= \mathbf{x}_t^j + \mathbf{v}_{t+1}^j \\ &= \mathbf{g}_t + \omega \mathbf{v}_t^j + \rho_t(1-2r_1) \end{aligned} \tag{A.14}$$

Der Parameter ρ_t bestimmt die dynamische Skalierung des zufälligen Suchraums um die global beste Position. Findet der Algorithmus oft genug in Folge ein neues $\mathbf{g}_t$, so wird der zufällige Suchraum durch ρ vergrößert, um größere Schritte zu ermöglichen. Analog wird ρ verkleinert, wenn sich $\mathbf{g}$ eine bestimmte Zeit lang nicht mehr geändert hat. In diesem Fall ist ein kleinerer Suchraum nötig, um das vermutete nahe lokale Minimum zu finden. Diese Erweiterung verhindert die Stagnation der Partikelbewegung und bietet zugleich eine adaptive Anpassung des lokalen Suchraums. Durch die somit konstant vorhandene Bewegungsdynamik kann für $t \to \infty$ die Konvergenz gegen lokale Extrema garantiert werden.

A.2.4.2 Multi-Start PSO

Um auch gegen das globale Minimum zu konvergieren, erweiterte Van den Bergh das Verfahren CGPSO um einen Mechanismus zur automatischen Detektion von Stagnation, in welchem Fall ein Neustart des Algorithmus' durchgeführt wird. Die Erweiterung *Multi-Start Particle Swarm Optimization* (MPSO) misst den Schwarmradius $\mathbf{r}$ um das momentan beste Partikel j mit

$$\mathbf{r} = \max_i \|\mathbf{x}_t^i - \mathbf{x}_t^j\|_2 \tag{A.15}$$

als euklidische Distanz im n-dimensionalen Suchraum. Fällt der Schwarmradius unter eine beliebig festgelegte Schwelle, so geht man von Stagnation aus. In diesem Fall wird

die bislang beste Position $\mathbf{g}_t$ zusammen mit $f(\mathbf{g}_t)$ gespeichert, und der Algorithmus neu gestartet. Am Ende der Iterationen wird das beste $\mathbf{g}_t$ aller Durchläufe als Ergebnis gewählt.

A.2.4.3 Regrouping PSO

Evers und Ben Ghalia [Evers und Ben Ghalia, 2009] bemerkten in ihrer Arbeit, dass im MPSO mit jedem Neustart Informationen über bereits erkundete Teilräume des Suchraums verworfen werden. Um diese Information zu nutzen, ersetzen sie in ihrer Modifikation *Regrouping Particle Swarm Optimization* (RegPSO) den Neustart des Algorithmus' durch eine lokale Neu-Gruppierung in der Nachbarschaft des Stagnationspunktes, um aus dem vermuteten lokalen Minimum zu fliehen. Der neue Suchraum wird durch die maximalen Abstände aller Partikeln entlang der n Dimensionen bestimmt. In ihren Untersuchungen erreichten Evers und Ben Ghalia im Durchschnitt einen um mehrere Größenordnungen kleineren Fehler als die unmodifizierte Partikelschwarmoptimierung, und einen um eine Größenordnung kleineren Fehler als die Modifikation MPSO.

A.2.4.4 Euclidean PSO

Ähnlich der Modifikation von Evers und Ben Ghalia nutzen auch Zhu et al. die vorhandene Information der vorhergehenden Suche, statt den Algorithmus neu zu starten. Ihre Modifikation *Euclidean Particle Swarm Optimization* (EPSO) erklärt den Schwarm für stagnierend, wenn mehr als k Iterationen lang keine Verbesserung von $\mathbf{g}_t$ stattgefunden hat [Zhu et al., 2009]. In diesem Fall erhält jedes Partikel einen sog. Interferenzimpuls, der den Schwarm zerstreuen soll. Die Stärke ε_t^i dieses Impulses ist eine Funktion der Entfernung des Partikels zur global besten Position, individuell für jede Dimension d:

$$\varepsilon_t^i|_d = \frac{2\mathbf{v}_{\max}|_d}{1 + \exp\left(\frac{-a}{\left(\mathbf{x}_t^i|_d - \mathbf{g}_t|_d\right)}\right)} - 1 \tag{A.16}$$

Diese Gleichung stellt eine Sigmoid-Funktion dar, skaliert anhand des Steigungsfaktors $a \in \mathbb{R}$ und der Distanz zur global besten Position. Die Größe dieses Interferenzimpulses

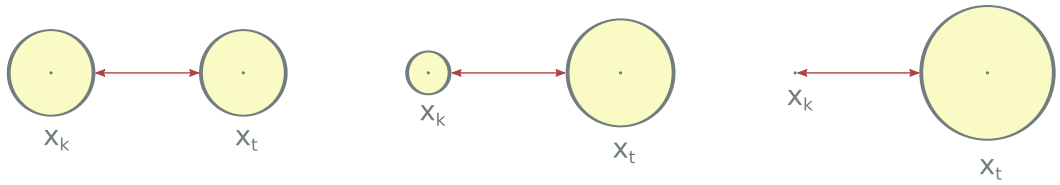


Abbildung A.2: Erweiterung der Mahalanobisdistanz zur Berücksichtigung der Unsicherheit des Datenpunktes $\mathbf{x}_k$. Hier wird schrittweise die Übertragung der Unsicherheit Σ_k auf Σ_t veranschaulicht, so dass letztendlich die Mahalanobisdistanz für den scharfen Datenpunkt $\mathbf{x}_k$ und die Normalverteilung $N(\mathbf{x}_t, \Sigma_t + \Sigma_k)$ berechnet werden kann.

wird durch die maximal mögliche Geschwindigkeit $\mathbf{v}_{\max}$ begrenzt. Wird eine Stagnation des Schwarms festgestellt, so wird eine modifizierte Berechnungsvorschrift angewendet:

$$\mathbf{v}_{t+1}^i = \omega \mathbf{v}_t^i + c_1 r_1 \left(\mathbf{p}_t^i - \mathbf{x}_t^i\right) + c_2 r_2 \left(\mathbf{g}_t - \mathbf{x}_t^i\right) + \varepsilon_t^i \tag{A.17}$$

Dies führt zu einer Art Explosion der Partikeln, zentriert über der global besten Position. Jedes Partikel wird indirekt proportional zu seiner Entfernung von $\mathbf{g}_t$ abgestoßen, so dass Partikeln kurz vor der Stagnation weit zurück in den Suchraum getrieben werden.

A.3 Erweiterte Mahalanobisdistanz

Ein übliches Distanzmaß für Normalverteilungen ist die Mahalanobisdistanz, die sich für eine Normalverteilung $N(\mu_i, \Sigma_i)$ und einen Datenpunkt $\mathbf{x}_k \in \mathbb{R}^3$ wie folgt berechnet:

$$d\left(\mathbf{x}_k, N\left(\mu_i, \Sigma_i\right)\right) = \sqrt{\left(\mathbf{x}_k - \mu_i\right)^T \Sigma_i^{-1} \left(\mathbf{x}_k - \mu_i\right)} \tag{A.18}$$

Die Mahalanobisdistanz entspricht der euklidischen Distanz, skaliert um die Varianz der Normalverteilung in Richtung des Datenpunktes. Dies stellt ein gutes Distanzmaß dar, wenn für den Datenpunkt $\mathbf{x}_k$ eine punktförmige Verteilungsfunktion angenommen wird. In diesem Abschnitt soll eine Erweiterung der Mahalanobisdistanz vorgestellt werden, die annimmt, dass auch der Datenpunkt $\mathbf{x}_k$ normalverteilt ist.

Gleichung (A.18) lässt sich einfach erweitern, um die Normalverteiltheit des Datenpunktes $N(\mu_k, \Sigma_k)$ zu berücksichtigen:

$$d\left(N\left(\mu_k, \Sigma_k\right), N\left(\mu_i, \Sigma_i\right)\right) = \sqrt{\left(\mu_k - \mu_i\right)^T \left(\Sigma_k + \Sigma_i\right)^{-1} \left(\mu_k - \mu_i\right)} \tag{A.19}$$

Zur Veranschaulichung dieser Erweiterung stelle man sich einen zeitlichen Prozess vor, der die Unsicherheit Σ_k kontinuierlich verringert und die Unsicherheit Σ_i zu gleichen Teilen erweitert (siehe Abbildung A.2). Während sich $N(\mu_k, \Sigma_k)$ immer weiter gegen eine punktförmige Verteilung bewegt, kompensiert $N(\mu_i, \Sigma_i)$ den Informationsverlust durch „Übernahme“ der verlorenen Unsicherheit. Am Ende dieses Prozesses kann die reguläre Mahalanobisdistanz für den Datenpunkt $\mu_k \in \mathbb{R}^3$ und die Normalverteilung $N(\mu_i, \Sigma_i + \Sigma_k)$ berechnet werden (siehe Gleichung (A.19)).

A.4 Betrachtungen zur numerischen Stabilität

Alle in dieser Arbeit vorgestellten Verfahren werden als Computerprogramme realisiert, die auf einem mobilen Roboter ausgeführt werden. Aufgrund des hohen Verbreitungsgrads besitzen die meisten mobilen Roboter einen handelsüblichen x86 oder x64-Prozessor. Unabhängig von der Architektur des Prozessors können reelle Zahlen von Computern nur mit beschränkter Genauigkeit dargestellt werden, so dass bei der Darstellung ein kleiner, aber nicht irrelevanter Diskretisierungsfehler gemacht wird. Dieser Fehler entsteht, da die reellen Zahlen durch eine endliche Anzahl von Bits repräsentiert werden müssen. Es existiert also nur eine endliche Anzahl an Diskretisierungspunkten im Spektrum der reellen Zahlen $\mathbb{R}$, die als Gleitkommazahlen dargestellt werden können. Für eine Zahl $x \in \mathbb{R}$ wird folglich der nächstmögliche Diskretisierungspunkt gewählt, um die Zahl möglichst genau darzustellen:

$$0.1 \xrightarrow{\text{24bit Genauigkeit}} 0.100000001490116119384765625 \tag{A.20}$$

Offensichtlich akkumuliert sich dieser Diskretisierungsfehler bei jeder arithmetischen Operation, so dass nach genügend Zeit eine signifikante Abweichung von der eigentlich darzustellenden Lösung $x^* \in \mathbb{R}$ vorliegen kann. Gerade iterative Verfahren leiden unter dieser numerischen Instabilität, da hier besonders viele fehlerbehaftete Operationen auf bereits fehlerhafte Zwischenlösungen angewendet werden.

Numerisch stabile Matrizeninversion

In dieser Arbeit wird mit großen Mengen mehrdimensionaler Daten gearbeitet, die alle mit beschränkter Genauigkeit dargestellt werden. Grundlage der meisten Operationen sind NDT-Komponenten $N \in \mathcal{N}$, die jeweils aus $|\mu| + |\Sigma| = 6 + 36 = 42$ Gleitkommazahlen bestehen. In Kapitel 4 werden die Kovarianzmatrizen Σ zur Berechnung der Distanzmetrik D_2 invertiert. Die Inversion einer Matrix ist ein numerisch relativ instabiles Verfahren, dessen Instabilität jedoch stark von der Kondition der Matrix abhängt. Für extrem schlecht konditionierte Matrizen A ($cond\,(A) \gg \varepsilon^{-1}$) führt eine Inversion durch numerische Algorithmen im Allgemeinen dazu, dass für A^{-1} keine einzige Stelle eines Koeffizienten a_{ij} als korrekt angenommen werden darf [Rump, 2009]. Dies ist im Rahmen dieser Arbeit sehr problematisch, da die Kovarianzmatrizen der NDT-Komponenten die geometrische Verteilung einer Punktemenge im dreidimensionalen Raum beschreiben. Da ein Großteil aller von Menschen geschaffenen Oberflächen aus ebenen Formen zusammengesetzt ist, werden durch NDT-Zellen häufig planare Punkteverteilungen modelliert. Derartige Punkteverteilungen besitzen eine hohe Varianz in mindestens einer Dimension, sowie eine sehr geringe Varianz in ebenfalls mindestens einer Dimension. Die entstehende Kovarianzmatrix ist daher meist relativ schlecht konditioniert, in manchen Fällen sogar beinahe singulär, so dass die invertierte Kovarianzmatrix Σ^{-1} häufig mit großen Fehlern behaftet ist. Diese Fehler wirken sich unmittelbar auf das berechnete Distanzmaß D_2 aus, so dass das Leistungsverhalten der entwickelten Registrierungsalgorithmen beeinflusst wird. Aus diesem Grund ist eine numerisch stabile Methode zur Inversion der Kovarianzmatrizen nötig.
Nach [DU CROZ und HIGHAM, 1992] bieten LU-Zerlegungen die besten Stabilitätskriterien für numerische Matrizeninversion. Aufgrund der hervorragenden Stabilitätseigenschaft [Higham, 2009] und der verhältnismäßig effizienten Berechnung wurde für diese Arbeit die Cholesky-Faktorisierung als Grundlage der Matrizeninversion gewählt.

Abbildungsverzeichnis

Literaturverzeichnis

[Arya et al., 1998] Arya, Sunil, D. M. Mount, N. S. Netanyahu, R. Silverman und A. Y. Wu (1998). *An optimal algorithm for approximate nearest neighbor searching fixed dimensions.* J. ACM, 45(6):891–923.

[Bailey und Durrant-Whyte, 2006] Bailey, T. und H. Durrant-Whyte (2006). *Simultaneous localization and mapping (SLAM): part II.* Robotics Automation Magazine, IEEE, 13(3):108 – 117.

[Beis und Lowe, 1997] Beis, J.S. und D. Lowe (1997). *Shape indexing using approximate nearest-neighbour search in high-dimensional spaces.* In: *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, S. 1000 – 1006.

[Bengtsson und Baerveldt, 2001] Bengtsson, O. und A. Baerveldt (2001). *Localization in changing environments - estimation of a covariance matrix for the IDC algorithm.* In: *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, Bd. 4, S. 1931 – 1937.

[Bentley, 1975] Bentley, Jon Louis (1975). *Multidimensional binary search trees used for associative searching.* Commun. ACM, 18(9):509–517.

[Biber und Strasser, 2003] Biber, P. und W. Strasser (2003). *The normal distributions transform: a new approach to laser scan matching.* In: *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, Bd. 3, S. 2743 – 2748.

[Britannica, 2012] Britannica, Encyclopaedia (2012). *Cartography.*

[Censi, 2007] Censi, A. (2007). *An accurate closed-form estimate of ICP's covariance.* In: *Robotics and Automation, 2007 IEEE International Conference on*, S. 3167 – 3172.

[Davison et al., 2004] Davison, A. J., Y. G. Cid und N. Kita (2004). *Real-time 3D SLAM with wide-angle vision.* In: *Proceedings of Intelligent Autonomous Vehicles, Portugal.*

[Dellaert et al., 1999] Dellaert, Frank, D. Fox, W. Burgard und S. Thrun (1999). *Monte Carlo Localization for Mobile Robots.* In: *IEEE International Conference on Robotics and Automation (ICRA99)*, S. 1322 – 1328.

[Diaz et al., 2012] Diaz, A., E. Caicedo, L. Paz und P. Pinies (2012). *A Real Time 6DOF Visual SLAM System Using a Monocular Camera.* In: *Robotics Symposium and Latin American Robotics Symposium (SBR-LARS), 2012 Brazilian*, S. 45 – 50.

[DU CROZ und HIGHAM, 1992] DU CROZ, JEREMY J. und N. J. HIGHAM (1992). *Stability of Methods for Matrix Inversion.* IMA Journal of Numerical Analysis, 12(1):1 – 19.

[Einhorn et al., 2012] Einhorn, E., T. Langner, R. Stricker, C. Martin und H. Gross (2012). *MIRA - middleware for robotic applications.* In: *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, S. 2591 – 2598.

[Evers und Ben Ghalia, 2009] Evers, G.I. und M. Ben Ghalia (2009). *Regrouping particle swarm optimization: A new global optimization algorithm with improved performance consistency across benchmarks.* In: *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*, S. 3901 – 3908.

[Grisetti et al., 2010a] Grisetti, G., R. Kuemmerle, C. Stachniss und W. Burgard (2010a). *A Tutorial on Graph-Based SLAM*. Intelligent Transportation Systems Magazine, IEEE, 2(4):31 – 43.

[Grisetti et al., 2010b] Grisetti, G., R. Kuemmerle, C. Stachniss, U. Frese und C. Hertzberg (2010b). *Hierarchical optimization on manifolds for online 2D*

and 3D mapping. In: *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, S. 273 – 278.

[Grisetti et al., 2005] Grisetti, G., C. Stachniss und W. Burgard (2005). *Improving Grid-based SLAM with Rao-Blackwellized Particle Filters by Adaptive Proposals and Selective Resampling.* In: *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, S. 2443 – 2448.

[Higham, 2009] Higham, Nicholas J. (2009). *Cholesky factorization.* Wiley Interdisciplinary Reviews: Computational Statistics, 1(2):251–254.

[Hoppe et al., 1992] Hoppe, Hugues, T. DeRose, T. Duchamp, J. McDonald und W. Stuetzle (1992). *Surface reconstruction from unorganized points.* SIGGRAPH Comput. Graph., 26(2):71–78.

[Huang et al., 2010] Huang, Shoudong, Y. Lai, U. Frese und G. Dissanayake (2010). *How far is SLAM from a linear least squares problem?.* In: *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, S. 3011 – 3016.

[Iddan und Yahav, 2001] Iddan, G.J. und G. Yahav (2001). *3D imaging in the studio (and elsewhere).* In: *Proceedings of the SPIE 4298: Videometrics and Optical Methods for 3D Shape Measurements.*, S. 48 – 55.

[Julier et al., 1995] Julier, S.J., J. Uhlmann und H. Durrant-Whyte (1995). *A new approach for filtering nonlinear systems.* In: *American Control Conference, Proceedings of the 1995*, Bd. 3, S. 1628 – 1632.

[Kaess et al., 2008] Kaess, M., A. Ranganathan und F. Dellaert (2008). *iSAM: Incremental Smoothing and Mapping.* IEEE Trans. on Robotics, TRO, 24(6):1365 – 1378.

[Kalman, 1960] Kalman, Rudolph Emil (1960). *A New Approach to Linear Filtering and Prediction Problems.* Transactions of the ASME–Journal of Basic Engineering, 82(Series D):35–45.

[Kennedy und Eberhart, 1995] Kennedy, J. und R. Eberhart (1995). *Particle swarm optimization.* In: *Neural Networks, 1995. Proceedings., IEEE International Conference on*, Bd. 4, S. 1942 – 1948.

[Leonard und Durrant-Whyte, 1991] Leonard, J.J. und H. Durrant-Whyte (1991). *Simultaneous map building and localization for an autonomous mobile robot.* In: *Intelligent Robots and Systems '91. 'Intelligence for Mechanical Systems, Proceedings IROS '91. IEEE/RSJ International Workshop on*, S. 1442 – 1447.

[Long et al., 2009] Long, Chengjiang, J. Zhao, Z. Yuan, Y. Ding, Y. Zhang, L. Xiong, G. Liang und X. Jiang (2009). *Improvements on IPD Algorithm for Triangular Mesh Reconstruction from 3D Point Cloud.* In: *Multimedia Information Networking and Security, 2009. MINES '09. International Conference on*, Bd. 1, S. 305 – 308.

[Magnusson, 2009] Magnusson, Martin (2009). *The Three-Dimensional Normal-Distributions Transform — an Efficient Representation for Registration, Surface Analysis, and Loop Detection.* Doktorarbeit, Örebro University. Örebro Studies in Technology 36.

[Mahon und Williams, 2003] Mahon, Ian und S. Williams (2003). *Three-Dimensional Robotic Mapping.* In: *In Proc. of the Australasian Conf. on Robotics and Automation.*

[Meagher, 1982] Meagher, D. (1982). *Geometric modeling using octree encoding.* Computer Graphics and Image Processing, 19(2):129 – 147.

[Montemerlo et al., 2002] Montemerlo, Michael, S. Thrun, D. Koller und B. Wegbreit (2002). *FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem.* In: *In Proceedings of the AAAI National Conference on Artificial Intelligence*, S. 593 – 598. AAAI.

[Muja und Lowe, 2009] Muja, Marius und D. G. Lowe (2009). *Fast approximate nearest neighbors with automatic algorithm configuration.* In: *In VISAPP International Conference on Computer Vision Theory and Applications*, S. 331 – 340.

[Murphy, 1999] Murphy, Kevin (1999). *Bayesian Map Learning in Dynamic Environments.* In: *In Neural Info. Proc. Systems (NIPS)*, S. 1015 – 1021. MIT Press.

[Nuechter et al., 2007] Nuechter, Andreas, K. Lingemann, J. Hertzberg und H. Surmann (2007). *6D SLAM - 3D mapping outdoor environments.* J. Field Robot., 24(8-9):699 – 722.

[Omohundro, 1987] Omohundro, S.M. (1987). *Efficient Algorithms with Neural Network Behaviour*. UIUCDCS-R. Department of Computer Science, University of Illinois at Urbana-Champaign.

[Paz et al., 2007] Paz, L.M., P. Jensfelt, J. Tardos und J. Neira (2007). *EKF SLAM updates in O(n) with Divide and Conquer SLAM*. In: *Robotics and Automation, 2007 IEEE International Conference on*, S. 1657 – 1663.

[Pozzi et al., 2012] Pozzi, F., T. Matteo und T. Aste (2012). *Exponential smoothing weighted correlations.* The European Physical Journal B, 85:1 – 21.

[Press et al., 1992] Press, William H., S. A. Teukolsky, W. T. Vetterling und B. P. Flannery (1992). *Numerical Recipes in C: The Art of Scientific Computing. Second Edition.*

[Rump, 2009] Rump, SiegfriedM. (2009). *Inversion of extremely Ill-conditioned matrices in floating-point.* Japan Journal of Industrial and Applied Mathematics, 26:249 – 277.

[Schroeter und Gross, 2008] Schroeter, C. und H.-M. Gross (2008). *A sensor-independent approach to RBPF SLAM - Map Match SLAM applied to visual mapping.* In: *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, S. 2078 – 2083.

[Smith und Cheeseman, 1986] Smith, Randall C. und P. Cheeseman (1986). *On the Representation and Estimation of Spatial Uncertainty.* Int. J. Rob. Res., 5(4):56–68.

[Stoyanov et al., 2012] Stoyanov, T., M. Magnusson und A. Lilienthal (2012). *Point set registration through minimization of the L2 distance between 3D-NDT models.* In: *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, S. 5196 – 5201.

[Stoyanov, 2012] Stoyanov, T.D. (2012). *Reliable Autonomous Navigation in Semi-Structured Environments using the Three-Dimensional Normal Distributions Transform (3D-NDT).* Doktorarbeit, Örebro University.

[Sunderhauf und Protzel, 2012] Sunderhauf, N. und P. Protzel (2012). *Towards a robust back-end for pose graph SLAM.* In: *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, S. 1254 – 1261.

[Thrun et al., 2005] Thrun, Sebastian, W. Burgard und D. Fox (2005). *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents).* The MIT Press.

[Ulaş und Temeltaş, 2012] Ulaş, Cihan und H. Temeltaş (2012). *3D Multi-Layered Normal Distribution Transform for Fast and Long Range Scan Matching.* Journal of Intelligent & Robotic Systems, S. 1 – 24.

[Van Den Bergh, 2002] Van Den Bergh, Frans (2002). *An analysis of particle swarm optimizers.* Doktorarbeit, Pretoria, South Africa, South Africa. AAI0804353.

[Wang et al., 2012] Wang, Heng, G. Hu, S. Huang und G. Dissanayake (2012). *On the Structure of Nonlinearities in Pose Graph SLAM.* In: *Proceedings of Robotics: Science and Systems VIII.*

[Welch und Bishop, 1995] Welch, Greg und G. Bishop (1995). *An Introduction to the Kalman Filter.*

[Williams et al., 2002] Williams, S., G. Dissanayake und H. Durrant-Whyte (2002). *An Efficient Approach to the Simultaneous Localisation and Mapping Problem*. In: *In Proc. IEEE Int. Conf. Robotics and Automation*, S. 406 – 411.

[Zhu et al., 2009] Zhu, Hongbing, C. Pu, K. Eguchi und J. Gu (2009). *Euclidean Particle Swarm Optimization*. In: *Intelligent Networks and Intelligent Systems, 2009. ICINIS '09. Second International Conference on*, S. 669 – 672.

Thesen zur Masterarbeit „Graphbasierte Lokalisation und 3D-Mapping für mobile Roboter“

i Die Kartierung und Lokalisation (SLAM) in einer unbekannten Umgebung ist eine grundlegende Voraussetzung für autonom agierende mobile Roboter.

ii Graphbasierte SLAM-Verfahren bieten ein besseres und konsistenteres Leistungsverhalten als momentan prävalente, filterbasierte SLAM-Algorithmen.

iii Der Normal Distributions Transform ist eine geeignete Datenstruktur zur detailgetreuen und dennoch effizienten Darstellung dreidimensionaler Tiefendaten.

iv Durch Registrierung kann die relative Transformation und Unsicherheit zwischen zwei Beobachtungen genauer als durch die Odometrie bestimmt werden.

v Der Levenberg-Marquardt Algorithmus und die Partikelschwarmoptimierung eignen sich zur robusten, echtzeitfähigen Registrierung zweier Beobachtungen.

vi Durch Registrierung und echtzeitfähige Schleifenschlusserkennung erzeugt das entwickelte SLAM-Frontend einen topologisch korrekten Posengraphen.

vii Die Optimierung durch *g2o* und *Vertigo* ermöglicht eine robuste Reduktion der Unsicherheiten und topologischen Inkonsistenzen des Posengraphen.

viii Es wurde experimentell gezeigt, dass das entwickelte graphbasierte SLAM-Verfahren auch für schwierige Datensätze konsistente Lösungen liefert.

Ilmenau, 04.02.2013 .....

Christian Reuther