Diplomarbeit

Ein probabilistischer Ansatz für robuste Lokalisierung eines mobilen Robotersystems mittels omnidirektionaler Sichtsysteme

Björn Gaworski

bg@elien.de

Universität Hamburg - Fachbereich Informatik

Arbeitsbereich Technische Aspekte Multimodaler Systeme

Juli 2005

Betreut von:

Prof. Dr. Jianwei Zhang

Prof. Dr. Bernd Neumann

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig verfasst habe und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Hamburg, den 28. Juli 2005

Björn Gaworski

Inhaltsverzeichnis

Danksagung

Vielen Dank an Daniel, für seine immer offene Tür
und seine Hilfe meine Gedanken zu sortieren.

Danke an Martin für seine Unterstützung
mit dem Featureextractor.

Ich bedanke mich bei Alexander für die
nette Gesellschaft beim Schreiben.

Ganz Besonders möchte ich mich bei meinem Vater
und bei Dieter bedanken, die mit detektivischem
Eifer Rechtschreib- und Grammatikfehler aufspürten.

Danke Mama für den Kaffee, der mir die Kraft
gab immer weiter zu schreiben.

Und natürlich bedanke ich mich bei meinen beiden
Betreuern, Prof. Dr. Jianwei Zhang und
Prof. Dr. Bernd Neumann, für ihre Unterstützung.

1 Einleitung

Diese Diplomarbeit beschäftigt sich mit der Lokalisierung eines mobilen Serviceroboters mit Hilfe einer omnidirektionalen Kamera. Die Problemstellung und Fragen hierzu werden in Kapitel 1.1 erläutert. Danach wird ein Einblick in den Stand der Forschung gegeben, der für die bearbeiteten Fragen relevant ist. Es wird dargestellt, wie an das Problem der Lokalisierung mit optischen Sensoren und mit omnidirektionaler Technik im Speziellen zur Zeit herangegangen wird. Zum Schluss des ersten Kapitels wird die für die Diplomarbeit zur Verfügung stehende Hardware vorgestellt und deren Einfluss auf den gewählten Lösungsweg beschrieben.

Kapitel 2 beschreibt die für die folgenden Kapitel notwendigen Verfahren zur Positionsschätzung mittels Odometrie, Zustandsschätzung durch Partikelfilter und Verarbeitung von omnidirektionalen Bildern. In Kapitel 3 wird dann im Einzelnen auf die Realisierung der Lokalisierung eingegangen. Die in den Experimenten gesammelten Daten werden in Kapitel 4 analysiert. Es folgt abschließend in Kapitel 5 die Zusammenfassung der Ergebnisse und ein Ausblick.

1.1 Problemstellung

Eine der wichtigsten Fragen in der mobilen Robotik lautet „Wo bin ich?“. Die genaue Position des Roboters zu kennen, ist von großer Bedeutung. Dies gilt insbesondere für die Pfadplanung und das Abfahren eines geplanten Pfades. An engen Stellen wie Türen oder in der Nähe von Hindernissen ist es zudem oft notwendig, eine bestimmte Position genau zu erreichen, damit der Roboter seine eigentliche Aufgabe durchführen kann. Schon eine kleine Abweichung der Orientierung von wenigen Grad oder ein Positionierungsfehler von wenigen Zentimetern können dazu führen, dass ein vorher berechneter Greifvorgang nicht mehr korrekt ausgeführt wird.

In der aktuellen Forschung gibt es verschiedene Ansätze, eine robuste Lokalisierung und Navigation für ein mobiles System zu realisieren. Viele Arbeiten stützen sich dabei auf Lasersensoren. Diese haben den Vorteil, dass die Entfernungen der erkannten Merkmale vorliegen und ohne weiteren Aufwand zur Berechnung genutzt werden können. Allerdings haben sie den entscheidenden Nachteil, dass sie nur eine horizontale Ebene der Umgebung abtasten. Dies kann zu einer Kollision führen, da ein Tischbein als Hindernis erkannt, die Tischplatte aber übersehen wird. Es gibt Verfahren, die mehrere Ebenen der Umgebung per Lasersensoren abtasten. Dies geschieht sukzessiv, indem die Sensoren geschwenkt oder die Laserstrahlen über Spiegel umgeleitet werden. [1] Dadurch dauert das Erfassen eines Scans der Umgebung länger, und

[1] Ein Beispiel ist der 3D-Laserscanner am Fraunhofer Institut für Autonome Intelligente Systeme

der Roboter hält während des Messvorgangs an. Diese Verfahren sind daher zurzeit eher zur dreidimensionalen Kartierung geeignet und weniger für Serviceaufgaben.

Optische Sensoren haben den Vorteil, dass sie - je nach Öffnungswinkel - die Umgebung direkt in mehreren Ebenen erfassen. Die für diese Diplomarbeit verwendete omnidirektionale Aufnahmetechnik (siehe Kapitel 1.3.2 und 2.3) ermöglicht es, eine Rundumsicht mit nur einer Aufnahme zu erfassen. Der Nachteil optischer Sensoren ist, dass sie keine Entfernungsinformationen zur Verfügung stellen. Außerdem entsteht bei der Bildverarbeitung aufgabenabhängig leicht ein hoher Rechenaufwand, der mit heutiger Hardware für einen Serviceroboter nicht geeignet ist, da er nicht in zufrieden stellender Zeit gelöst werden kann.

In dieser Diplomarbeit wird der Frage nachgegangen, inwieweit Lokalisierungsverfahren, die nur auf Basis optischer Sensoren arbeiten, in der Lage sind, eine robuste Positionsbestimmung durchzuführen. Dazu wurden drei verschiedene Algorithmen ausgewählt, die mit Hilfe omnidirektionaler Bilder die Positionsänderungen eines mobilen Robotersystems verfolgen. Jedes der Verfahren ist in der Lage, aus den gelieferten Bildern eine Positionsschätzung abzuleiten, die an einen Partikelfilter weiter geleitet werden. Der Partikelfilter trifft Vorhersagen über die Änderungen der Position des Roboters und integriert die Schätzungen seiner Quellen, um daraus den momentanen Aufenthaltsort des Roboters zu bestimmen. Die Grundlagen des Partikelfilters werden in Kapitel 2.2 aufgeführt.

In Kapitel 1.2 werden aktuelle Verfahren zur Positionsbestimmung mittels optischer Sensoren vorgestellt. Für die Verwendung in dieser Arbeit wurden ein Fingerprinting- und ein Tracking-Algorithmus sowie ein Verfahren zur Bestimmung der Rotationsänderung anhand vertikaler Kanten ausgewählt. Der Lösungsansatz wird in Kapitel 1.4 detailliert vorgestellt.

1.2 Forschungsstand

Es ist heutzutage üblich, eine robuste Lokalisierung auf Basis von Lasersensoren zu implementieren. Es gibt aber weiter gehende Ansätze, autonome Systeme mit Hilfe optischer Systeme zu lokalisieren. Im Folgenden werden einige Verfahren der letzten Jahre vorgestellt, um einen Rahmen für diese Arbeit festzulegen.

1.2.1 Fingerprinting

Mit Fingerprinting sind Verfahren gemeint, die in einer Trainingsphase Signaturen für Bilder an verschiedenen Orten bestimmen und in einer Datenbank ablegen. In der Lokalisierungsphase werden anhand der Signatur der neu aufgenommenen Bilder Übereinstimmungen in der Datenbank gesucht, um schon einmal besuchte Positionen wieder zu erkennen.

Lamon *et al.* extrahieren in [1] und [2] Merkmale aus Panoramabildern und generieren daraus einen String, der die erkannten Merkmale repräsentiert. Es können

(AIS), http://www.ais.fhg.de/ARC/3D/scanner/index.html

alle möglichen Arten von Merkmalen benutzt werden. In ihren Veröffentlichungen werden vertikale Kanten und Farbflächen genutzt. In der Bildsignatur werden vertikale Kanten durch ein v repräsentiert, während die Buchstaben $A,B,C,\ldots,P$ für verschieden farbige Flächen stehen. Um zwei Signaturstrings miteinander zu vergleichen, wird ein *Minimum energy algorithm* vorgestellt.

In [3] wird Fingerprinting zur groberen topologischen Lokalisierung eingesetzt. In verschiedenen topologisch wichtigen Bereichen werden Bilder aufgenommen. Dies können zum Beispiel Positionen bei Türen oder an anderen prägnanten Stellen sein. Zu jedem Bereich wird aus den Bildern der *prominenteste* Eigenvektor gelernt und in einer Datenbank abgelegt, so dass zu jedem Bereich ein charakterisierender Eigenvektor bekannt ist. Zur Lokalisierung werden die Eigenvektoren der neu aufgenommenen Bilder bestimmt und mit der Datenbank verglichen, um einen vorher gelernten Bereich wieder zu erkennen.

Gross *et al.* bestimmen in [4] und [5] die Bildsignatur über Farbwerte in Panoramabildern. An den Referenzpunkten werden diese aufgenommen und in vertikale Sektoren eingeteilt. Für jeden Sektor wird der durchschnittliche Farbwert bestimmt. Um zwei Signaturen miteinander zu vergleichen, wird der Winkel der normalisierten Signatur-Vektoren berechnet.

Weitere Verfahren zur Signaturbestimmung von Panoramabildern und zum Signaturvergleich sind in [6], [7] und [8] beschrieben.

Fingerprintingverfahren bilden in der Literatur den Schwerpunkt zur Lokalisierung mittels optischer Sensoren. Es lassen sich dort viele Algorithmen zur Signaturerzeugung und zum Signaturvergleich finden. Allerdings lokalisieren die meisten Verfahren topologisch, so dass die Ergebnisse bestenfalls so gut sind wie zwei Referenzpunkte auseinander liegen.

Für diese Diplomarbeit wird ein Fingerprinting-Verfahren eingesetzt (vgl. Kapitel 3.2), das in [9] vorgestellt wird, dort aber nur in einer kleinen künstlichen Umgebung getestet wurde.

1.2.2 Fusionierung mit Lasersensordaten

In [10], [11], [12] und [13] werden Sensordaten von optischen Sensoren und Laserscannern fusioniert und zur Lokalisierung verwendet. Die gewonnenen Daten über die Umgebung enthalten neben den Entfernungsinformationen der Laserscanner auch die Farbinformationen der Sichtsysteme. Man versucht nun beide Datensätze zu kombinieren, um daraus eine (eventuell dreidimensionale) Repräsentation der Umwelt zu generieren. Da diese Verfahren Lasersensoren zur Lokalisierung einsetzen, sind sie für diese Arbeit nicht in Betracht gezogen worden.

1.2.3 Stereovision

Bei Einsatz mehrerer Kameras, deren Positionierung zueinander bekannt ist, können durch die Kamerabilder dreidimensionale Repräsentationen der Umgebung geschaffen werden (Stereomessung, *Photogrammetrie*).

In [14] wird ein *Simultaneous Localization And Map Building*-Verfahren (SLAM) vorgestellt, das lediglich mit optischen Sensoren arbeitet. Mit drei Kameras wird während der Fahrt eine 3D-Karte der Umwelt aufgebaut und ständig erweitert. Gleichzeitig wird der Roboter innerhalb dieser Karte lokalisiert.

1.2.4 Tracking

Für das Tracking werden in den aufgenommenen Bildern Merkmale bestimmt. Diese können jegliche Form von Primitiven wie Ecken oder Kanten sein, aber auch komplexere Muster wie Farb- oder Textureigenschaften von Bildbereichen. Innerhalb des Videostreams werden diese Eigenschaften verfolgt und über deren Positionsänderung im Stream Rückschlüsse auf die Bewegung des Roboters gemacht [15].

In [16] werden künstliche Landmarken erkannt und verfolgt. Zugleich sind die Landmarken geometrisch so modelliert, dass auch ihre Orientierung im Raum aus den Bildern erkannt werden kann.

Andreasson, Treptow und Duckett verwenden in [17] eine angepasste Form von Lowes SIFT-Algorithmus [18], um lokale Merkmale in Panoramabildern zu erkennen. SIFT-Merkmale sind invariant gegenüber Skalierung, Translation und Rotation und zum Teil invariant gegenüber Beleuchtungsänderungen und affiner oder 3D-Projektion. SIFT-Algorithmen sind patentrechtlich geschützt und werden bereits kommerziell eingesetzt [2] [3].

Für diese Diplomarbeit wird ein Tracking-Verfahren implementiert, das ohne künstliche Landmarken arbeitet, um ein Verfahren zu haben, das auch in einer unbekannten Umgebung eingesetzt werden kann.

1.3 Hardware und Referenzlokalisierung

Für diese Arbeit stand ein mobiler Serviceroboter mit einem omnidirektionalen Sichtsystem zur Verfügung (siehe Abbildung 1.1). Details zu den Geräten und der verwendeten Referenzlokalisierung werden in den folgenden drei Unterkapiteln gegeben.

1.3.1 Der mobile Serviceroboter der Arbeitsgruppe TAMS

Die mobile Serviceplattform der Arbeitsgruppe TAMS wurde von der Firma Neobotix hergestellt. An der Plattform ist ein Roboterarm [4] befestigt, der für den weiteren Verlauf dieser Arbeit keinerlei Bedeutung hat. Der Roboter hat eine Höhe von knapp

[2] SIFT im Internet: http://www.cs.ubc.ca/ lowe/keypoints/

[3] SIFT wird unter anderem von Evolution Robotics, Inc. verwendet, deren Software unter anderem in Sonys AIBO eingesetzt wird. http://www.evolution.com, http://www.eu.aibo.com/

[4] MHI PA10-6C (6-Achsen Modell), http://www.mhi.co.jp/kobe/mhikobe-e/products/mechatronic/e_index.html

zwei Meter und wiegt ca. 200 kg. In der Aufsicht hat der Roboter eine quadratische Form mit ca. 65 cm Kantenlänge. Je nachdem, ob der Roboter in einer zwei- oder dreidimensionalen Repräsentation der Umwelt lokalisiert wird, ist es üblich, die Form zu einem Kreis oder Zylinder zu vereinfachen.

In der Regel halten Roboter bei der Fahrt einen Sicherheitsabstand zu Wänden und anderen Hindernissen ein, der durch die Kreisform und einen entsprechend gewählten Radius dargestellt wird. Außerdem sind x- und y-Koordinate des Mittelpunkts zur Kollisionserkennung ausreichend, die Orientierung braucht nicht berücksichtigt zu werden. Die mobile Serviceplattform lokalisiert sich innerhalb einer zweidimensionalen Karte. Aus diesem Grund wird für den weiteren Verlauf dieser Diplomarbeit die Form des Roboters als ein Kreis mit einem Radius von 70 cm behandelt (siehe Abbildung 1.2).

Der Roboter wird von einem Differentialantrieb bewegt, der ihn bis auf eine Geschwindigkeit von $1\frac{m}{s}$ beschleunigen kann. Für alle Fahrten, die in dieser Arbeit untersucht wurden, ist die Höchstgeschwindigkeit jedoch auf $\frac{1}{2}\frac{m}{s}$ beschränkt, weil die Fahrten dann wesentlich ruhiger verlaufen und die Kamerabilder trotz der Aufbauhöhe des omnidirektionalen Sichtsystems nicht verwackeln. Der Antrieb besteht aus zwei Servomotoren, die jeweils ein Antriebsrad auf der linken bzw. rechten Seite steuern. Die Motoren sind unabhängig voneinander, so dass kein weiteres Rad zur Steuerung notwendig ist. Die Richtung wird allein durch den Antrieb des linken und rechten Rades beeinflusst.

Die omnidirektionale Kamera bildet den höchsten Punkt und ist über eine Metallkonstruktion mit dem Roboter verbunden. Dadurch ist die Kamera zwar seitlich zentriert, befindet sich aber ca. 43 cm hinter dem Mittelpunkt des Roboters (siehe Abbildung 1.2).

1.3.2 Das omnidirektionale Sichtsystem

Für die Aufnahmen der Bilder wird ein omnidirektionales Sichtsystem, bestehend aus einer Firewire-Kamera und einem omnidirektionalen Spiegel (Panorama Eye® von Seiwapro) [5], verwendet.

Die für die Arbeit verwendete Firewire-Kamera ist das Modell Sony DFW-SX900 und hat eine Auflösung von maximal 1280 mal 960 Pixeln bei einer Bildrate von höchstens 7,5 Bildern pro Sekunde. Sie ist nach oben ausgerichtet und filmt den darüber befestigten Spiegel.

Der Spiegel hat eine hyperboloide Form, die es ermöglicht, die Umgebung horizontal in 360° abzubilden. Der vertikale Öffnungswinkel beträgt nach oben 15° und wird nach unten durch den Rand der Kamera begrenzt (siehe Abbildung 1.3). Dadurch entstehen so genannte omnidirektionale Bilder (siehe Abbildung 1.4). Wie diese in Panoramabilder umgerechnet werden können, wird in Kapitel 2.3 gezeigt. Die auf dem Scheitelpunkt des Spiegels befestigte schwarze Nadel verhindert, dass Reflektionen von der gegenüber liegenden Plexiglaswand auf den Spiegel fallen (vgl. [19], Seite 31 ff.).

[5] http://www.accowle.com

Abbildung 1.1: Der für die Experimente dieser Arbeit eingesetzte Serviceroboter des Arbeitsbereichs TAMS.

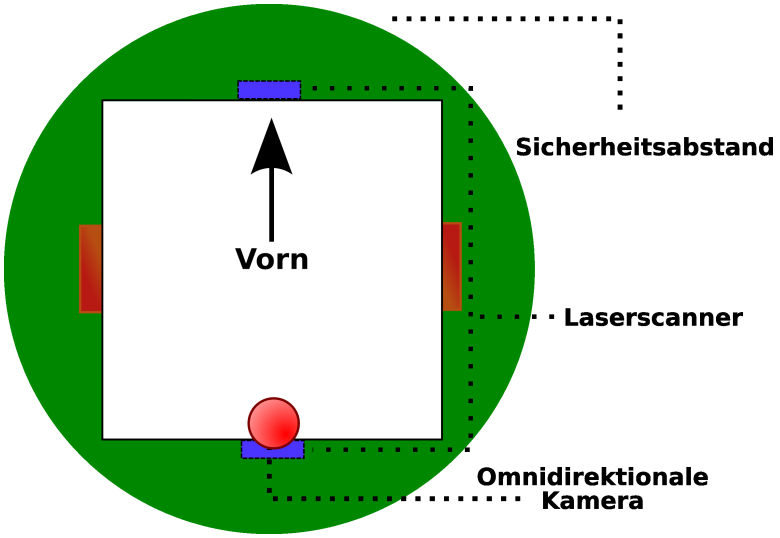

Abbildung 1.2: Schematische Darstellung des Roboters, Sicht von oben.

Der Vorteil der omnidirektionalen Aufnahmetechnik liegt in der Möglichkeit, eine Rundumsicht der Umgebung mit einer Aufnahme zu erfassen. Es müssen keine einzelnen Bilder aus verschiedenen Winkeln aufgenommen und später mosaikartig verschmolzen werden. Ein Nachteil sind die hohen Herstellungskosten der Spiegel. Sie müssen sehr präzise gefertigt werden, damit sich die gespiegelten Lichtstrahlen in genau einem Brennpunkt treffen (siehe auch [19] und [20]). Damit die Kamera und der Spiegel exakt zueinander ausgerichtet sind, ist der Spiegel in einer Plexiglasröhre eingebettet, die am unteren Ende mit einem Gewinde ausgestattet ist und vor die Kamera geschraubt wird.

1.3.3 Referenzlokalisierung

Auf dem Roboter läuft ein Lokalisierungsalgorithmus. Dieser bestimmt die aktuelle Position mit Hilfe der zwei Laserscanner, die vorn und hinten am Serviceroboter angebracht sind. Scherer zeigt in [21], dass diese Lokalisierung bis auf 1 cm genau ist. Die 25 cm über dem Boden am Roboter befestigten Laserscanner bedeuten für die Laserlokalisierung, dass Merkmale und eventuelle Hindernisse nur auf dieser Ebene erkannt werden.

Die Lokalisierung arbeitet mit Reflektormarken, die in der Umgebung des Roboters verklebt sind. Der Algorithmus wird mit initialen Positionen der Marken und einer manuellen Kalibrierung gestartet, d.h. eine Schätzung der Roboterposition und -orientierung muss zu Beginn der Lokalisierung vorgegeben werden. Im weiteren Verlauf der Positionsschätzung werden die Positionen der Reflektormarken verbessert. Zur Positionsbestimmung werden die Winkel- und Entfernungsmessungen der erkannten Marken und die Messungen der Odometrie mittels Kalman-Filter vereint.

Abbildung 1.3: Für diese Arbeit verwendete Kamera und Spiegel. Rechts unten eine schmeatische Darstellung des Öffnungswinkels vom Spiegel.

Abbildung 1.4: Omnidirektionales Bild

Andere durch die Laserscanner erkannte Merkmale werden lediglich zur Kollisionsvermeidung ausgewertet.

Für den weiteren Verlauf dieser Arbeit wird angenommen, dass die von der Laserlokalisierung bestimmte Position zu jedem Zeitpunkt die fehlerfreie Position des Roboters darstellt. Sie wird im Folgenden als Referenzlokalisierung bezeichnet. Durch die Referenzlokalisierung ist es möglich, die in dieser Arbeit implementierten Verfahren quantitativ zu bewerten und Messfehler zu vermeiden, die durch Bestimmen der Position per Hand - zum Beispiel mit einem Zollstock - entstünden. Außerdem können so leicht viele Wiederholungen einer Fahrt berechnet werden, da Zeit für das manuelle Ausmessen nicht anfällt. Da die Laserlokalisierung höher getaktet ist als die auf Seite 16 gezeigte Implementierung der Lokalisierung dieser Arbeit, steht zu jedem Zeitpunkt eine Ist-Position zur Verfügung.

Die Ausgaben des Laserlokalisierungsalgorithmus dienen direkt als Eingabe für den Positionsregler der Robotersteuerung. An einigen Stellen, die der Roboter aufsucht, kommt es zu Fehlern in der Lokalisierung. Dies führt zu heftigen Kurskorrekturen (vgl. Kapitel 4 und Abbildung 4.1).

1.4 Lösungsansatz

Für diese Diplomarbeit sollen drei Verfahren zur Lokalisierung mittels optischer Sensoren miteinander verglichen werden. Dabei wird nicht nur die Fähigkeit der Verfahren unter optimalen Bedingungen getestet, sondern besonders auf deren Leistung nach längeren Fahrten eingegangen. Im realen Einsatz sind mobile Systeme durchaus mehrere Stunden in Betrieb, und das hat Auswirkungen auf die Leistungsfähigkeit.

Die Betriebsspannung der Batterien lässt nach und es kommt zu Ermüdungserscheinungen der mechanischen Teile, z.B. zum Erhitzen der Bremsen oder Motoren. Aus diesem Grund werden solche Eigenheiten bewusst mit untersucht. Dem mobilen System wird zwischen den Aufnahmen der Testfahrten keine Pause gelassen, um den realen Betrieb zu simulieren.

Zum Vergleich der Lokalisierungsverfahren wird die vorhandene Laserlokalisierung als *ground truth* (Referenzlokalisierung) angenommen. Neben der omnidirektionalen Kamera steht die Odometrie als zusätzliche Sensorquelle zur Verfügung. Folgende Verfahren werden miteinander verglichen:

- Korrektur der Orientierung durch Bestimmung der Rotation durch vertikale Kanten (Kapitel 3.1)
- Wiedererkennen von gelernten Positionen durch Fingerprinting (Kapitel 3.2)
- Korrektur der Positionsänderung durch das Tracking von Merkmalen in der Umgebung (Kapitel 3.3)

Die von der Odometrie und den Bildverarbeitungsverfahren gesammelten Daten werden durch einen Partikelfilter integriert (siehe Kapitel 2.2). Der Partikelfilter arbeitet in diskreten Zeitabschnitten, die von den Sensorquellen bestimmt werden. Wenn eine Aktualisierung der Odometriedaten stattfindet oder ein neues Bild zur Verfügung steht, beginnt ein neuer Zeitabschnitt. Im Folgenden werden die einzelnen Zeitabschnitte als Berechnungsrunden oder einfach nur als Runden bezeichnet.

Der Vergleich der Verfahren findet auf vorher aufgezeichneten Fahrten statt. Der Entschluss, auf Live-Fahrten zu verzichten, hat mehrere Gründe. Zum einen sind die Bildverarbeitungsalgorithmen für diese Arbeit nicht so umgesetzt worden, dass sie in Echtzeit ausgeführt werden können, da hier die Verfahren und nicht die Qualität der Implementierung zum Vergleich steht. Folglich würden bei Live-Fahrten die Auswertungen der langsamen Verfahren überhaupt nicht berücksichtigt, da sie erst zu spät zur Verfügung stünden. Zum anderen können einmal aufgezeichnete Fahrten immer wieder berechnet werden. Die Änderungen in der Genauigkeit der Lokalisierung müssen dann Parameteränderungen zugeschrieben werden, da die Fahrt dieselbe ist (siehe auch Kapitel 4). Wie es zu Schwankungen in der Ausführung einer Fahrt kommt, wird genauer in Kapitel 2.1 erläutert.

In den Fahrtaufzeichnungen stellt die Odometrie ungefähr alle 30 Millisekunden eine neue Position zur Verfügung. Demgegenüber steht die Bildrate der Kamera, die in den Aufzeichnungen ungefähr alle 600 ms ein neues Bild liefert. Die Differenz zur auf Seite 13 angegebenen Bildrate entsteht, weil jedes Bild ungefähr 3,5 MB groß ist und der Speichervorgang auf einer Notebookfestplatte stattfindet und dementsprechend Zeit benötigt. Die Konvertierung der Bilder vor dem Speichern in ein platzsparenderes Format führt zu einer noch niedrigeren Bildrate, da der Roboter durch die Fahrtausführung und Referenzlokalisierung so ausgelastet ist, dass für die Konvertierung lediglich ca. 20% der CPU-Zeit zur Verfügung stehen.

Da die Kamerabildrate im Vergleich zur Odometrie wesentlich niedriger getaktet ist, beginnt vereinfacht ungefähr alle 30 ms eine neue Runde. Zu jeder Runde werden

die berechneten Daten zusammengefasst und die aktuelle Position bestimmt (siehe auch Kapitel 2.2). Zum Vergleich wurde eine Testfahrt bestimmt, die zehnmal aufgezeichnet wurde. Für jedes der drei Verfahren wurde dann die Lokalisierung der aufgezeichneten Fahrten berechnet, die im Kapitel Analyse ab Seite 43 mit den tatsächlichen Aufenthaltsorten der Referenzlokalisierung verglichen wird. Außerdem werden in dem Kapitel noch weitere Partikelfilterkonfigurationen untersucht, um zum Beispiel die Genauigkeit des Partikelfilters anhand einer perfekten Quelle zu bestimmen.

2 Grundlagen

In den folgenden drei Unterkapiteln werden die für die weitere Diplomarbeit notwendigen Verfahren zur Positionsschätzung mittels Odometrie, Zustandsschätzung durch Partikelfilter und Verarbeitung von omnidirektionalen Bildern beschrieben.

2.1 Odometrie

Als Odometrie wird die Berechnung der Position aufgrund des zurück gelegten Weges während einer Zeiteinheit bezeichnet. Die Wegberechnung geschieht hierbei durch Analyse der gemessenen Wegstrecke des linken und rechten Antriebsrades. Im Laufe des Kapitels wird gezeigt, warum dieses Verfahren zwar grundlegend ist, aber aufgrund seiner Fehlerhaftigkeit nicht zuverlässig sein kann.

2.1.1 Berechnung der Wegstrecke

Der Roboter ändert beim Fahren seine Position (x, y) sowie seine Orientierung θ in Bezug zu einem Weltkoordinatensystem. Dabei repräsentiert θ den Winkel zur x-Achse des Weltkoordinatensystems. Das Tripel (x, y, θ) definiert eindeutig die Position des Roboters und wird *Pose* genannt.

Zur Berechnung der Änderung der *Pose* über die Zeit liefern Inkremental- oder Winkelgeber in den Antrieben die nötigen Daten. Sie geben die Winkelgeschwindigkeiten der Räder zu einem Zeitpunkt t an. Es wird hier nur auf den diskreten Fall eingegangen, in dem zwischen zwei aufeinander folgenden Odometriedaten die Zeit δt liegt.

Es seien v_r und v_l die Winkelgeschwindigkeiten des rechten und linken Rades. Wenn diese die Einheit Grad haben, müssen sie durch die Formel 2.1 in Rad umgerechnet werden. Des weiteren sei d_w der Durchmesser der beiden Räder (d_w sei für beide Räder gleich).

Durch die Formel 2.2 wird die Strecke berechnet, die das rechte bzw. linke Rad in δt mit den gegebenen Geschwindigkeiten zurückgelegt hat.

$$rad = \frac{grad}{360} \cdot \pi \tag{2.1}$$

$$\begin{aligned} s_r &= v_r \cdot d_w \cdot \delta t \\ s_l &= v_l \cdot d_w \cdot \delta t \end{aligned} \tag{2.2}$$

Wenn der Abstand der beiden Räder d_r (Durchmesser der Achse) bekannt ist, berechnet sich der Winkel α, um den sich der Roboter gegenüber seiner alten *Pose* gedreht hat, nach

$$\alpha = \frac{s_r - s_l}{d_r}$$

Die Distanz s, die der Roboter im Mittel zurückgelegt hat, ist für kleine α

$$s = \frac{v_r + v_l}{2}$$

ansonsten

$$s = \frac{v_r + v_l}{\alpha} \cdot sin\left(\frac{\alpha}{2}\right)$$

Damit ergibt sich die Veränderung der *Pose* von (x, y, θ) nach $(\delta x, \delta y, \delta\theta)$ mit

$$\begin{aligned} \delta x &= s \cdot cos\left(\theta + \frac{\alpha}{2}\right) \\ \delta y &= s \cdot sin\left(\theta + \frac{\alpha}{2}\right) \\ \delta\theta &= \theta + \alpha \end{aligned}$$

In der Regel ist es sinnvoll, den Winkel nach der Berechnung zu normalisieren, so dass er im Intervall $[0; 2\pi]$ liegt.

2.1.2 Aufsummierung der Fehler

Dieser Berechnung der Odometrie liegt das Modell einer perfekten Fahrbahn zu Grunde, die völlig eben ist und eine unendliche Reibung besitzt, die Räder also nicht durchdrehen. Dieses Modell spiegelt natürlich nicht die Realität wider. Kleine Wellen im Boden, Hindernisse oder Schlupf führen zu einem Fehler der Odometrie in jedem Zeitintervall δt.

Angenommen, die Odometrie misst für beide Motoren dieselbe zurück gelegte Strecke, dann ergibt sich aus den oben aufgeführten Formeln, dass der Roboter geradeaus gefahren ist, ohne seine Orientierung zu ändern. In Wirklichkeit ist aber ein Rad, z.B. das Linke, über eine kleine Welle gefahren. Diese Welle verändert die Entfernung, die das linke Rad zu seinem Startpunkt zurücklegt hat. Zwar sind beide Räder die gleiche Strecke gefahren, aber da das linke Rad etwas bergauf fahren musste (und eventuell auch wieder etwas bergab), hat es auf die Ebene projiziert nicht die gleiche

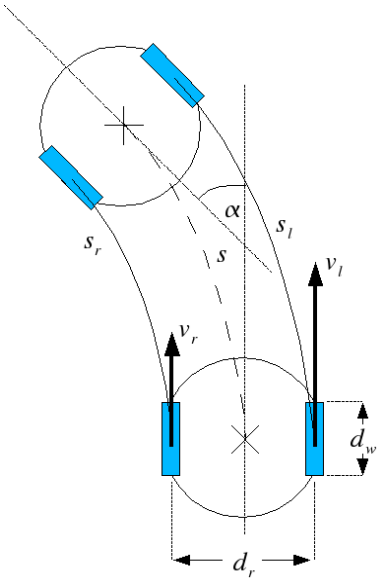

Abbildung 2.1: Skizze zur Vorwärtskinematik

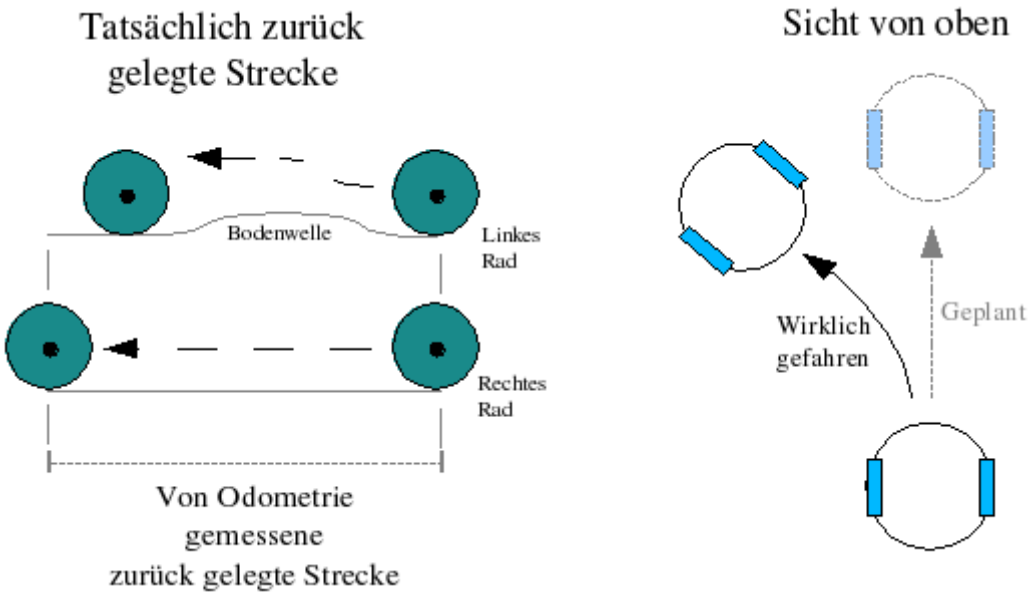

Abbildung 2.2: Fehler bei einer Bodenwelle

Entfernung zurückgelegt. Der Roboter ist also nicht geradeaus gefahren, sondern eine leichte Kurve, die auch noch zu einer Änderung in der Orientierung führte (siehe Abbildung 2.2).

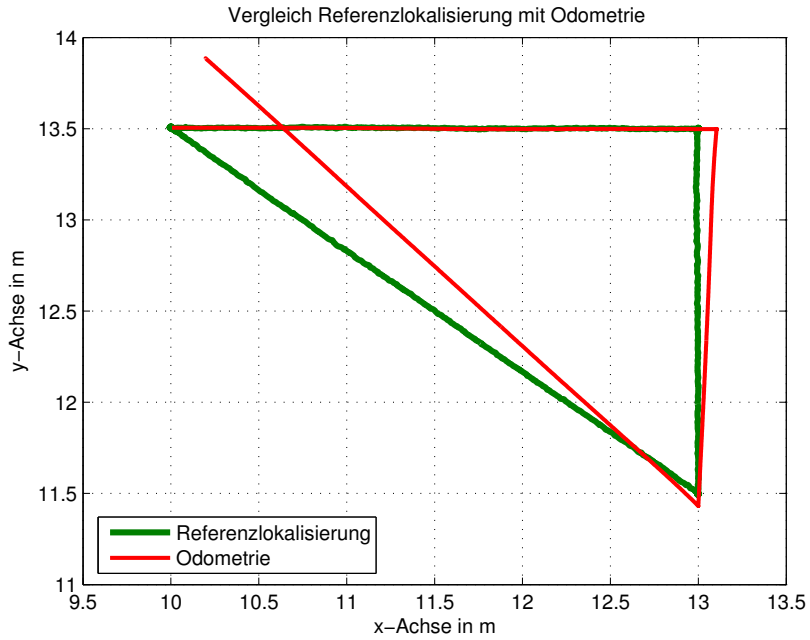

Abbildung 2.3: Summierung der Odometriefehler

Dieser Fehler mag für ein Zeitintervall δt unbedeutend klein erscheinen, aber die Fehler über alle Zeitintervalle addieren sich. Außerdem führen auch kleine Fehler in der Orientierung zu immer größeren Abweichungen bei steigender Distanz, selbst wenn weitere Fehler ausblieben. (siehe Abbildung 2.3).

2.2 Partikelfilter

Der Partikelfilter wird dazu genutzt, die Position des mobilen Roboters zu schätzen. Dazu integriert er mehrere Eingabequellen, um die Qualität der Schätzung zu verbessern. In diesem Abschnitt wird ein Überblick über die Funktionsweise eines Partikelfilters gegeben. Die in dieser Arbeit benutzten Quellen werden im Kapitel 3 ab Seite 30 vorgestellt.

Die Hauptaufgabe ist zu beobachten, wie sich die *Pose* über die Zeit entwickelt. Typischerweise besitzt diese eine nicht-Gauss'sche Verteilungsfunktion, die mit Hilfe von Samples (Partikeln) dargestellt wird. Es wird eine Menge von Kopien der *Pose* erzeugt und jeder Kopie ein Gewicht zugewiesen, das die Qualität des Partikels bestimmt. Jede Aktion des Roboters hat nun zur Folge, dass sich die *Pose* und das Gewicht der Partikel verändern. Die Funktionsweise des Partikelfilters ist rekursiv und lässt sich in zwei Phasen aufteilen: *Vorhersage* und *Update*. Von Zeit zu Zeit wird ein *Resampling* durchgeführt, das die Partikelpopulation aussagekräftig hält.

2.2.1 Vorhersage

Formell ausgedrückt ist die *Pose* ($p^t = [x^t, y^t, \theta^t]^T$) zu einer Zeit t also eine Menge M von Partikeln ($P^t = [p_j^t, w_j^t] : j = 1 \ldots |M|$), wobei j den Partikel bezeichnet. Jeder Partikel besteht aus einer Kopie der *Pose* p_j^t und einem Gewicht w_j^t.

In der Vorhersagephase wird anhand eines Modells die Auswirkung simuliert, die eine Aktion auf die Menge der Partikel hat. Dieser Arbeit liegt kein theoretisches Modell zugrunde, statt dessen werden die Daten der Odometrie benutzt, um die Partikel zu verschieben. Die Formel 2.2 wird dahin gehend verändert, dass ein Rauschen zu der gefahrenen Strecke des linken bzw. rechten Rades hinzu gefügt wird:

$$\begin{aligned} s_r &= (v_r \cdot d_w \cdot \delta t) * N(0, \sigma) \\ s_l &= (v_l \cdot d_w \cdot \delta t) * N(0, \sigma) \end{aligned} \tag{2.3}$$

Es hat sich gezeigt, dass die Nutzung der Odometrie als Grundlage des Partikelfilters ein ausreichendes Modell für die robuste Lokalisierung ist. Die Odometrie liefert in sehr kurzen Abständen neue Daten, in denen der Roboter naturgemäß nicht sehr weit fahren konnte (selbst mit maximaler Geschwindigkeit nur wenige Zentimeter). Außerdem interessiert für das Modell des Partikelfilters nur die relative Veränderung zum letzten Ort, an dem Odometriedaten geliefert wurden. Dadurch bleibt das Modell durch das Phänomen der Aufsummierung der Odometriefehler unbeeinflusst. Zudem ergeben sich zwei weitere Vorteile:

1. Die Partikel werden nur so verschoben, wie es die Odometrie gemessen hat. Der Fehler der Odometrie wird durch Hinzufügen von Rauschen ausreichend simuliert (siehe Kapitel 3). Dadurch ergibt sich eine Einengung des Suchraums für die Position, an dem sich der Roboter befindet. Dies ist ein entscheidender Vorteil, wenn man viele Partikelfilterquellen hat oder Quellen nutzt, die für jede Positionsschätzung viel Rechenzeit benötigen.

2. Das Rauschen wird nur über einen freien Parameter σ simuliert (Formel 2.3), der leicht im Experiment bestimmt werden kann. Für die Modellierung des gesamten Fahrverhaltens eines Roboters bei entsprechenden Aktionen müssten sonst umfangreiche Experimente durchgeführt und viele Parameter bestimmt werden. In [22] wird ein Modell vorgestellt, welches die Translation und den Drift bei der Bewegung simuliert. Hierzu werden sechs Parameter benötigt.

2.2.2 Update

In der Update-Phase werden die Gewichte der Partikel bestimmt. Dazu wird jede Partikelfilterquelle um eine Schätzung für die *Pose* jedes Partikels gebeten. Mit anderen Worten, für wie wahrscheinlich sie es halten, dass sich der Roboter an der *Pose* des Partikels befindet.

Von den einzelnen Quellen wird nicht verlangt, dass die Summe ihrer Wahrscheinlichkeiten über alle Partikel Eins ergeben. Es ist wesentlich einfacher, dieses später zu normalisieren. Die Wahrscheinlichkeiten für einen Partikel von den verschiedenen Quellen werden multipliziert, um das Gewicht des Partikels zu bestimmen. Allerdings werden nur Quellen befragt, die eine Aussage treffen können. Es wäre zum Beispiel sinnlos, eine Quelle um eine Gewichtung zu bitten, die auf Bildverarbeitung beruht, obwohl kein aktuelles Bild zur Verfügung steht. Das Gewicht eines Partikels ist für die Menge der Partikelfilterquellen S^t, die eine Aussage zum Zeitpunkt t über die *Pose* der Partikel treffen können, definiert als

$$w_j^t = \prod_{\forall s_i \in S^t} s_i$$

Zum Normalisieren wird die Summe der Gewichte w_t zum Zeitpunkt t bestimmt. Der normalisierte Partikel m_i^t ergibt sich dann aus dem Produkt seines Gewichtes mit w_t^{-1}:

$$\begin{aligned} w_t &= \sum_{\forall w_i \in M^t} w_i^t \\ m_i^t &= [p_i^t, w_i^t \cdot w_t^{-1}] \end{aligned}$$

2.2.3 Resampling

Einige der Partikel bekommen mit der Zeit sehr kleine Gewichte zugewiesen und verlieren immer mehr an Bedeutung für die Wahrscheinlichkeitsverteilung. Sei $S_t = \{p_i^t, w_i^t\}$ die diskrete Verteilungsfunktion der *Pose* des mobilen Roboters, dann wird eine neue Darstellung $S'_t = \{p'^t_i, w'^t_i\}$ gesucht, so dass $p_i^k = p'^l_i$ für k, l aus $[1, M]$, dieselbe Verteilungsfunktion darstellt.

Dazu wird eine untere Schranke definiert, die Partikel mit kleineren Gewichten als unbedeutend markiert. Zusätzlich werden zwei weitere Schranken gesetzt. Eine definiert die Anzahl der Partikel, die mindestens vorhanden sein müssen. Fällt die Gesamtpopulation unter diesen Wert, werden neue Partikel erzeugt. Die andere Schranke bestimmt, wie viele Partikel maximal in einer Runde gelöscht werden dürfen. Sollte die Menge der unbedeutenden Partikel größer sein als die Anzahl der Partikel, die gelöscht werden dürfen, werden aus der Menge genau so viele Partikel gelöscht wie höchstens erlaubt ist. Die Auswahl der Partikel wird zufällig getroffen.

Beispiel: In der Implementierung sind die Schranken zur Populationserhaltung prozentual definiert. Dadurch ändern sich die Schranken automatisch, wenn die Gesamtzahl der Partikel verändert wird. Nehmen wir an, die maximale Partikelpopulation ist auf 100 begrenzt und beim Resampling dürfen maximal 60% aller Partikel auf einmal gelöscht werden, es müssen aber gleichzeitig immer mindestens 70% der Gesamtpopulation vorhanden sein. Sobald also weniger als 70 Partikel vorhanden sind,

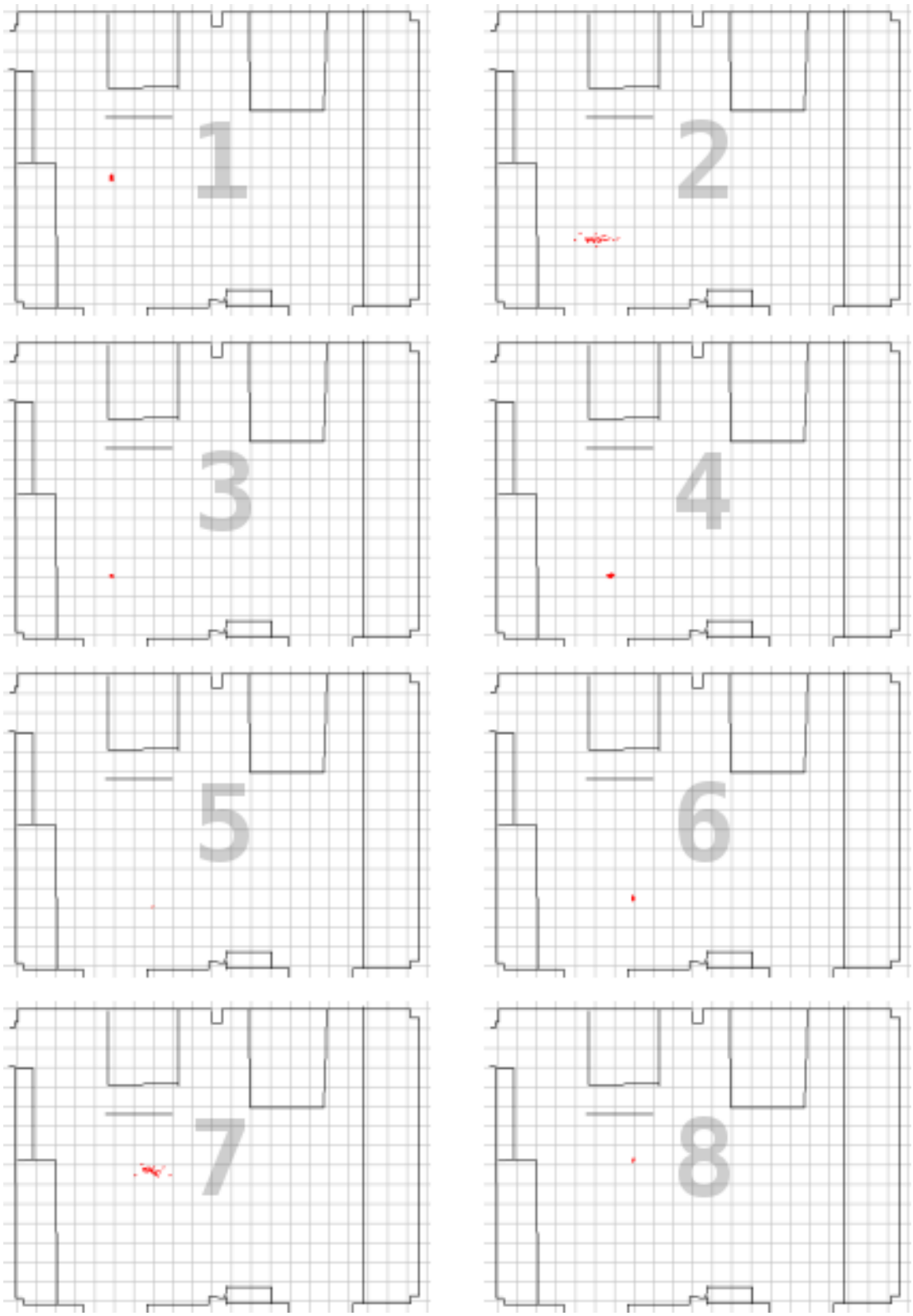

Abbildung 2.4: Simulation einer Fahrt mit 100 Partikeln. Man kann sehen, wie sich die Partikel während der Fahrt verbreiten. Zu weit entfernte Partikel werden gelöscht und bei Bedarf werden neue erzeugt. Sonst würden sich die Partikel immer weiter voneinander entfernen.

werden neue erstellt. Die Strategie hierbei ist es, auf der momentan geschätzten *Pose*, die als Aufenthaltsort des Roboters angenommen wird, die Anzahl Partikel zu erzeugen, die notwendig sind, die Gesamtanzahl auf die maximale Partikelpopulation aufzustocken. Sollten mehr als 60 Partikel zum Löschen markiert sein, werden aus der Menge der zu löschenden Partikel per Zufall 60 ausgewählt, die gelöscht werden. Dürfen alle Partikel auf einmal gelöscht werden, wird in derselben Runde beim Resampling die maximale Anzahl von Partikeln auf der momentan geschätzten *Pose* erzeugt. Da alle neu erzeugten Partikel ein Gewicht von Eins erhalten, ist diese Situation gleich dem Startzustand, in dem auch alle Partikel mit einem Gewicht von Eins an derselben Stelle liegen.

2.2.4 Bestimmung der Position

Aus der Menge der Partikel muss eine Schätzung der *Pose* abgeleitet werden. Dazu werden üblich die folgenden Techniken benutzt: Das gewichtete Mittel ($P = \sum_{j=1}^{M} w_j p_j$), der beste Partikel ($p_j$, so dass $w_j = max(w_k) : k = 1 \ldots |M|$) und das gewichtete Mittel in einem kleinen Fenster um den besten Partikel (auch robustes Mittel genannt).

Bei der Methode des besten Partikels wird der Partikel mit der größten Wahrscheinlichkeit gewählt und seine *Pose* als aktuelle Position angenommen. Sollten mehrere Partikel die größte Wahrscheinlichkeit besitzen, wird der zuerstgefundene gewählt.

Das gewichtete Mittel berechnet die aktuelle Position als den Schwerpunkt der Partikelfilterpopulation. Bei der Schwerpunktsberechnung wird die Wahrscheinlichkeitsschätzung jedes Partikels als Gewicht für die *Pose* des Partikels berücksichtigt.

Das robuste Mittel wird genauso bestimmt wie das gewichtete Mittel. Es werden zur Schwerpunktberechnung aber nur Partikel berücksichtigt, die einen maximalen Abstand zum besten Partikel nicht überschreiten. Der beste Partikel wird wie oben beschrieben bestimmt.

Abbildung 2.4 zeigt eine Folge von acht Grafiken, die illustriert, wie sich die *Pose* der Partikel und die Partikelpopulation über die Zeit verändern.

2.3 Omnidirektionale Bilder in Panoramabilder umwandeln

Für die meisten Bildverarbeitungsalgorithmen ist es einfacher, Panoramabilder zu verwenden. In Kapitel 3.1 wird zum Beispiel ein Verfahren vorgestellt, das mit vertikalen Kanten arbeitet. In einem Panoramabild können die bekannten Kantendetektoren verwendet werden [23]. In einem omnidirektionalen Bild müssten die Algorithmen angepasst werden, da hier vertikale Kanten strahlenförmig von innen nach außen abgebildet werden. Einen solchen radialen Kantendetektor zu konstruieren ist aufwendiger, als das Bild in ein Panoramabild umzurechnen, in dem vertikale Kanten auch als solche abgebildet werden.

Die Umrechnung von omnidirektionalen Bildern in Panoramabilder wird in [24] beschrieben. Die dort behandelte, perspektivisch korrekte Umrechnung benötigt die

Abbildung 2.5: Erzeugtes Panoramabild aus einem omnidirektionalen Bild

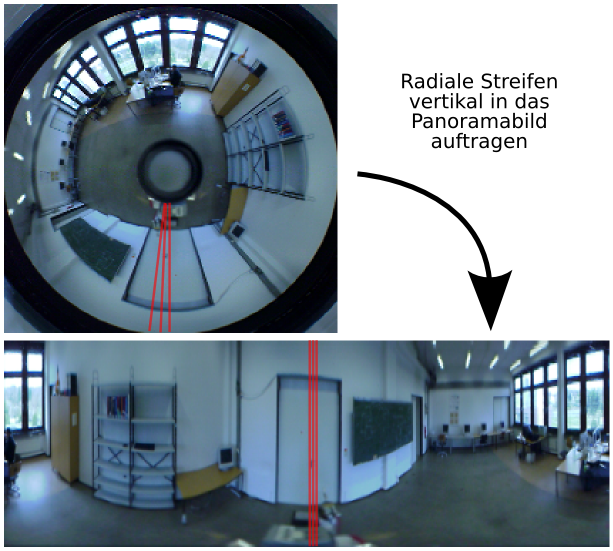

Abbildung 2.6: Abtragen des omnidirektionalen Bildes auf einen Panoramastreifen

Kamera- und Spiegelparameter. Da für die in dieser Arbeit behandelten Bildverarbeitungsalgorithmen keine perspektivisch korrekte Umrechnung notwendig ist, wird die Umrechnung etwas einfacher durchgeführt. Dazu wird das omnidirektionale Bild vom Mittelpunkt ausgehend strahlenförmig Spalte für Spalte in ein Panoramabild aufgetragen (siehe Abbildung 2.6).

In der Praxis ist es effizienter, den Algorithmus so zu gestalten, dass vom Panoramabild ausgehend eine Abbildung geschaffen wird, die jedem Pixel im Panoramabild genau ein Pixel im omnidirektionalen Bild zuordnet. Diese Abbildung muss dann für jede Umrechnung „Omnidirektionales Bild einer bestimmten Auflösung zu Panoramabild einer gewünschten Auflösung“ nur einmal berechnet werden. Die daraus entstehende Abbildung wird in einer Art *Karte* gespeichert. Jede weitere gleichartige Transformation kann dann schnell mit der entsprechenden Abbildung der Karte durchgeführt werden. Die Umrechnung wird so von einer aufwendigen Rechnung in einfaches Pixelkopieren umgewandelt.

Dass die Bildverarbeitungsalgorithmen keine perspektivisch korrekte Umrechnung benötigen, lässt sich recht einfach erläutern. Das Fingerprintingverfahren (ab Seite 34) vergleicht Bilder anhand einer abstrakten Struktur. Dafür ist es nur notwendig, dass Bilder an gleichen Orten möglichst ähnliche Signaturen besitzen. Ob die Bilder perspektivisch korrekt umgerechnet wurden, ist dafür unerheblich. Die beiden anderen Verfahren - Rotationsverschiebung durch vertikale Kanten (ab Seite 30) und Tracking (ab Seite 37) - arbeiten mit horizontalen Bildeigenschaften, die durch das radiale Auftragen erhalten bleiben.

3 Implementierung

In diesem Kapitel wird die Umsetzung des Partikelfilters (siehe Kapitel 2.2). und seiner Quellen beschrieben. Die dabei verwendete Programmiersprache ist Java™ [1]. Für die Implementierung verwendete Hilfsmittel sind auf Seite 69 angegeben.

Die *Pose*-Schätzungen der verschiedenen Quellen werden durch einen Partikelfilter zu einer stärkeren Schätzung vereint. Später werden in Kapitel 4 die Lokalisierungsergebnisse der einzelnen Quellen analysiert. Dort wird auch die Genauigkeit des implementierten Partikelfilters für den *best case* gezeigt - nämlich mit den den Daten der Referenzlokalisierung als Eingabequelle.

Um die Referenzlokalisierung mit der Lokalisierung dieser Arbeit zu vergleichen, wird jede Berechnung durch den Partikelfilter mit der Start-*Pose* der Referenzlokalisierung initialisiert. Die Partikel liegen zu Beginn alle an dem Ort, an dem sich der Roboter zum Startzeitpunkt aufgehalten hat. Für den Verlauf der Fahrt wird dann die Abweichung der Lokalisierung zur Referenzlokalisierung untersucht.

Es wurde bereits erwähnt, dass der Partikelfilter rundenbasiert aufgebaut ist. Zu Beginn einer Runde wird jede zur Verfügung stehende Quelle gefragt, ob sie für die anstehende Runde eine Schätzung abgeben kann.

Für die Runde trifft der Partikelfilter eine Vorhersage der Verschiebung für die *Poses* der Partikel. Dazu nimmt er als Grundlage die Veränderung in der *Pose*, die durch die Odometrie berechnet wurde (siehe Seite 19). Das Tripel der Veränderung sei $p_\delta = (x_\delta, y_\delta, \theta_\delta)$. Die neue *Pose* für jeden Partikel ergibt sich aus der Summe von p_δ und der alten *Pose* des Partikels. In Kapitel 2.1.2 wurde gezeigt, dass die Odometrie fehlerhaft ist. Um dies zu berücksichtigen, wird p_δ zusätzlich für jeden Partikel verrauscht. Dazu werden drei normal verteilte Zufallswerte generiert, die dann auf die *Pose* addiert werden.

Nun wird jede Quelle, die eine Schätzung abgeben kann, für die neue *Pose* von jedem Partikel um eine Bewertung gebeten. Die Antwort liegt im Intervall $[0, 1]$ und wird mit der aktuellen Wahrscheinlichkeit des Partikels multipliziert (Update). Die Bildverarbeitungsquellen werden eine Bewertung der Partikel verweigern, wenn die Kamera kein neues Bild geliefert hat. Ansonsten wird jede Quelle je nach Verfahren versuchen, eine Schätzung zu berechnen.

So entsteht eine Wahrscheinlichkeitsverteilung über alle Partikel, die dann normiert werden muss, damit die Summe über alle Wahrscheinlichkeiten einer Runde Eins ergibt. Aus diesen nicht gaussverteilten Wahrscheinlichkeiten wird nun wie in Kapitel 2.2.4 beschrieben die *Pose* bestimmt, an der sich der Roboter vermutlich aufhält. Je

[1] *Java* ist ein eingetragenes Warenzeichen von Sun Microsystems, Inc., http://java.sun.com

nach Notwendigkeit wird die Partikelmenge nun noch dem Resampling unterworfen, bevor mit einer neuen Runde begonnen wird.

Zu jedem Rundenende steht der berechneten *Pose* eine Referenz-*Pose* der Referenzlokalisierung gegenüber. In Kapitel 4 wird der Fehler zwischen beiden *Poses* bestimmt und die Entwicklung des Fehlers über die Zeit analysiert.

In den folgenden Unterkapiteln werden die bildverarbeitenden Verfahren erläutert, die für den Partikelfilter zur Verfügung gestellt werden.

3.1 Rotationsbestimmung durch vertikale Kanten

Dieses Verfahren wird im folgenden *RotationEdgeDetect* oder kurz RED genannt.

Ein Panoramabild stellt eine vollständige Rundumsicht dar. Läuft man über die erste oder letzte Spalte hinaus, beginnt man wieder auf der gegenüberliegenden Seite. Das hat zur Folge, dass Bilder, die an der gleichen Position, aber unterschiedlicher Orientierung, aufgenommen werden, zueinander lediglich horizontal verschoben sind. Abweichungen ergeben sich nur durch das Rauschen der Kamerabilder. Aus der horizontalen Verschiebung kann man bestimmen, wie sich die Orientierung geändert hat. Genau dieses wird anhand der Verschiebung von vertikalen Kanten versucht.

Das Verfahren funktioniert nur dann erwartungsgemäß, wenn der Roboter zwischen zwei Vergleichsbildern lediglich die Orientierung ändert. Wurde eine Strecke zurückgelegt, gibt es zwischen den Bildern kompliziertere Transformationen, mit denen dieses Verfahren nicht umgehen kann. Daher liefert diese Quelle nur Daten an den Partikelfilter, wenn zwischen zwei Aufnahmen lediglich eine Rotation statt gefunden hat. Dieses wird anhand der Veränderung der Position durch die Odometrie fest gestellt, da es bei der vorhandenen Robotersteuerung nicht möglich ist, in Erfahrung zu bringen, welche Bewegung vom Roboter gerade ausgeführt wird.

Leider ist es mit der für diese Arbeit verwendeten Hardware nicht möglich, eine Drehung auszuführen, so dass die Kamera ihre Position nicht verändert. Wie in Abbildung 1.2 dargestellt befindet sich die Kamera hinter dem Robotermittelpunkt, so dass eine Drehung des Roboters immer eine Translation der Kamera zur Folge hat. Es ist für den Roboteraufbau, der für diese Diplomarbeit zur Verfügung stand, nicht möglich, den Kameraursprung über dem Robotermittelpunkt zu platzieren, da die Kamera dann im Arbeitsbereich des Roboterarmes läge, was vermieden werden muss. Bei kleinen Rotationen sind die Transformationen der Bilder nach einer Drehung trotzdem klein genug, dass der RED-Algorithmus zufrieden stellende Ergebnisse liefert.

Bei der Bestimmung der ausgeführten Bewegung über die Odometriedaten muss berücksichtigt werden, dass sich die Position des Roboters auch bei einer reinen Rotation verändert. Um diesem Phänomen Rechnung zu tragen, wird eine Schranke fest gelegt. Liegt eine Positionsverschiebung unterhalb der Schranke, wird angenommen, dass nur eine Rotation statt gefunden hat. Die Bestimmung einer guten Schranke ist von großer Bedeutung. Rotationsversuche haben gezeigt, dass große Orientierungsänderungen zum Teil große Positionsverschiebungen bewirken (siehe Tabelle 3.1).

Eine perfekte Schranke - d.h. alle Rotationen werden als solche erkannt, während keine Translation für eine Rotation gehalten wird - kann es nicht geben.

Zur Erläuterung sei eine Schranke von 145 Millimetern gewählt, so dass alle Rotationen aus Tabelle 3.1 als solche klassifiziert werden. Die Kamera liefere alle 1,6 Sekunden ein neues Bild. Angenommen der Roboter setzt sich 1 Sekunde, nachdem das letzte Bild von der Kamera geliefert wurde, in Bewegung, um geradeaus zu fahren. Er müsste in 0,6 Sekunden mindestens 146 Millimeter zurücklegen, damit die Bewegung nicht als Rotation klassifiziert wird. Bei Höchstgeschwindigkeit würde der Roboter in der Zeit ca. 300 mm zurücklegen. Während der Beschleunigungsphase zu Fahrtbeginn auf die Fahrtgeschwindigkeit von $\frac{1}{2}\frac{m}{s}$ fährt er in 0,6 Sekunden jedoch nicht einmal die notwendigen 146 Millimeter, so dass die Translationsbewegung falsch als Rotation erkannt wird.

Um die Verschiebung zweier Panoramabilder P_1 und P_2 zu ermitteln, werden diese in vertikale Kantenbilder transformiert. Dazu wird jedes Bild zuerst mit einem Median gefiltert, um das Rauschen in den Kameradaten zu minimieren. Dann werden mit einer vertikalen Sobelmaske (siehe [23]) die vertikalen Kanten im Bild bestimmt. In den resultierenden Kantenbildern bedeutet schwarz keine vertikale Kante, und je heller der Pixel ist, um so größer ist die vertikale Kante an der Stelle.

Daraus wird ein vertikales Kantenprofil erstellt, für das die Kantenwerte jeder Spalte aufsummiert und anschließend normiert werden. Aus den normierten Kantenprofilen zweier Panoramabilder wird ein Verschiebungsprofil errechnet. Dafür werden die beiden Kantenprofile übereinander gelegt und der mittlere quadratische Fehler errechnet:

Es sei n die Anzahl der Spalten eines Panoramabildes, und es seien K^1 und K^2 die beiden vertikalen Kantenprofile der zu vergleichenden Panoramabilder. K_i^1 und K_i^2 bezeichnen die i-te Spalte in den Profilen. Dann berechnet sich der mittlere quadratische Fehler ϵ der Profile durch

$$\epsilon = \frac{1}{n} \cdot \sum_{i=1}^{n} (K_i^1 - K_i^2)^2$$

Man erhält das Verschiebungsprofil, wenn man diese Berechnung n-mal durchführt und die Profile für jede Berechnung gegeneinander verschiebt. Das Verschiebungsprofil enthält also in Spalte i den mittleren quadratischen Fehler der Profile, die um i Spalten gegeneinander verschoben wurden. Deutlich wird das aus Abbildung 3.1. Würde man Kantenprofil A zu Kantenprofil B um 327 (oder -33) Spalten verschieben, sind sie annähernd deckungsgleich. Bei Spalte 327 ist auch der kleinste quadratische Fehler im Verschiebungsprofil.

Nach dieser Methode wird die Verschiebung zweier Panoramabilder zueinander bestimmt. Das Minimum im Verschiebungsprofil bestimmt die Anzahl der Spalten, um die die Panoramabilder zueinander verschoben sind. Sollte es mehrere gleiche Minima geben, wird das zuerst gefundene gewählt. Man könnte auch das Minimum wählen, das der Bewegung, die durch die Odometrie bestimmt wurde, am nächsten

Winkel	Verschiebung in mm
0.07°	46.10
95.88°	54.34
-94.95°	56.04
-0.96°	130.86
30.06°	43.42
-30.08°	17.26
66.01°	56.22
28.04°	96.01
44.89°	31.02
-59.97°	77.06
-44.98°	8.54
-33.98°	86.01
0.16°	10.20
-0.18°	61.03
105.90°	40.11
-105.80°	77.01
6.91°	95.01
-6.92°	66.01
177.69°	67.90
-79.80°	60.30
29.90°	38.21
-17.94°	108.23
-109.96°	24.17
0.03°	144.50
14.96°	9.49
140.01°	31.32
-87.92°	79.91
11.91°	129.31
-78.97°	56.32
172.05°	80.40
-171.06°	85.48
146.08°	52.63
-147.04°	12.00
-0.04°	65.03
0.07°	59.14

Tabelle 3.1: Zufällig hintereinander ausgeführte Rotationen und dabei entstehende Positionsverschiebung

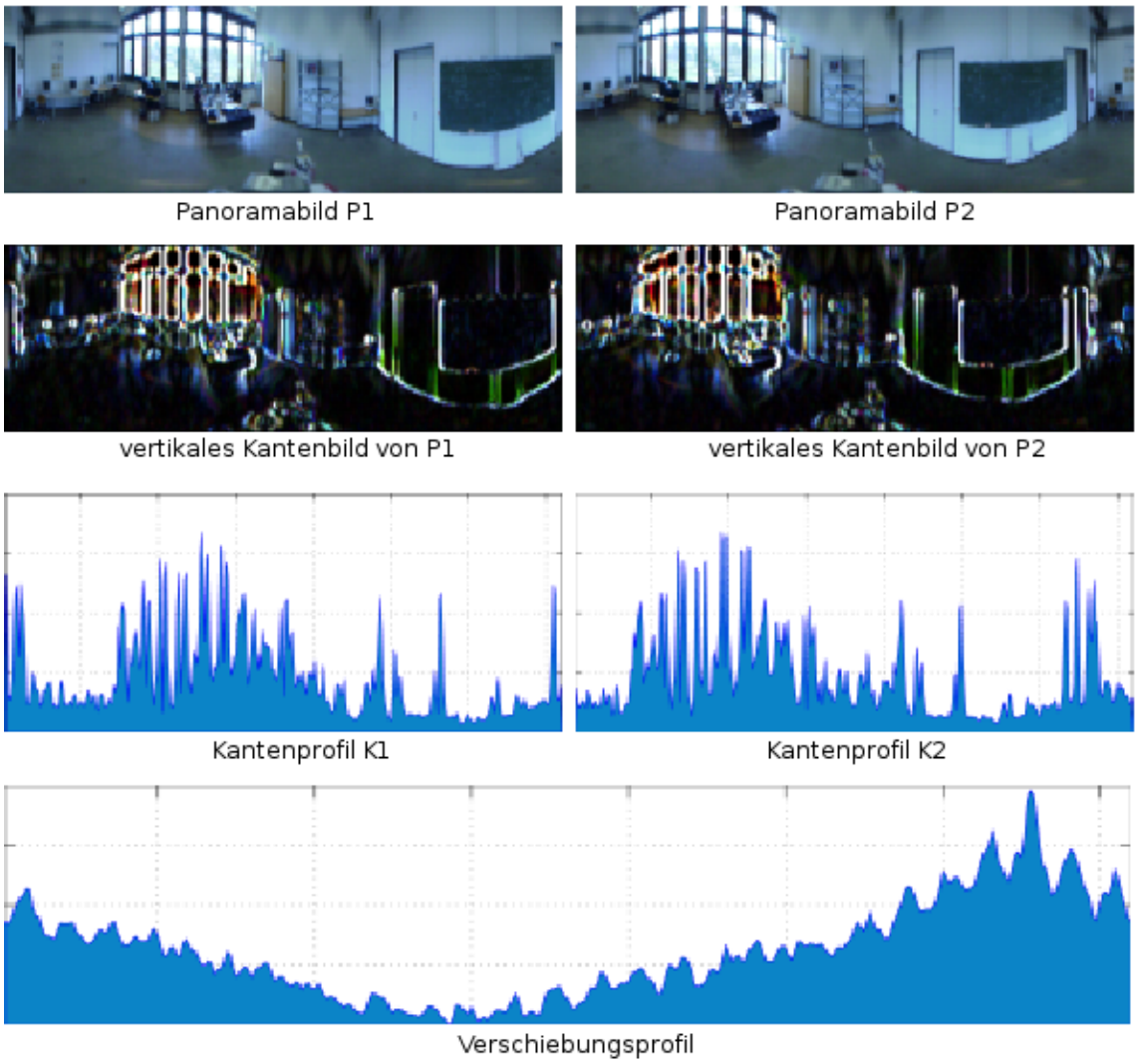

Abbildung 3.1: Panoramabilder, Kantenbilder, Kantenhistogramme und Verschiebungsprofil. Das Maximum im Verschiebungsprofil liegt bei Spalte 327. Das ergibt eine Verschiebung von 33° zwischen den beiden Bildern. Das Verfahren funktioniert, obwohl der Roboterarm am unteren Bildrand zu sehen ist, d.h. das Verfahren ist robust gegenüber kleinen Veränderungen im Raum.

ist. Allerdings gibt es nur in seltenen Fällen mehrere gleiche Minima - in den durchgeführten Experimenten trat der Fall kein einziges mal auf. Die Bestimmung der Bewegung durch die Odometrie gestaltet sich aufwendig, da es zwischen den zwei zu vergleichenden Bildern mehrere Odometriewerte gibt, die zusammen gefasst werden müssen. Wegen des zusätzlichen Aufwands und der niedrigen Auftrittswahrscheinlichkeit des Sonderfalls, dass es mehrere gleiche Minima gibt, wurde auf die Implementierung der aufwendigeren Minimum-Bestimmung verzichtet. Trotzdem sollte dies in zukünftigen Versionen berücksichtigt werden, da es eine Verbesserung der Rotationserkennung bedeutet.

Mit dem gefundenen Minimum an der Stelle i und der Annahme, dass ein Panoramabild die Umgebung im Intervall $[0, 360]$ Grad abbildet, lässt sich der Winkel der Verschiebung α durch

$$\alpha = \frac{360}{n \cdot i}$$

berechnen.

Um eine *Pose* P schätzen zu können, bestimmt die Quelle die Rotation θ_δ aus dem aktuellen Bild und dem vorherigen Bild. Es wird noch die *Pose* P_{alt} bestimmt, an der sich der Roboter zum Zeitpunkt aufgehalten hat, als das vorherige Bild aufgenommen wurde. Die Orientierungsänderung zwischen alter *Pose* und zu schätzender *Pose* ist $\theta_\epsilon = |P(\theta) - P_{alt}(\theta)|$.

Weil dieses Verfahren nur Aussagen über Orientierungsänderungen machen kann, wird bei der Bestimmung der Schätzung nur die Orientierung berücksichtigt. Die Schätzung ist der Wert der Normalverteilung $\mathcal{N}(\theta_\delta; 1, 5)$ für θ_ϵ multipliziert mit einem Gewicht, so dass das Maximum der Normalverteilung Eins ist, weil alle Partikelfilterquellen Schätzungen zwischen Null und Eins zurückgeben sollen.

Die Verteilung der Schätzung verläuft dann wie in Abbildung 3.2 dargestellt. Nur nicht mit dem Erwartungswert Null, sondern θ_δ. Die Verteilung der Schätzung ist so gewählt, dass Abweichungen des Erwartungswertes größer als Vier sehr klein bewertet werden. Dass die Verteilung ausreicht, die Lokalisierung zu verbessern, und zudem zur Genauigkeit des RED-Verfahrens passt, wird in Kapitel 4.3 erläutert, in welchem die Ergebnisse des RED-Verfahrens für den Partikelfilter analysiert werden.

3.2 Fingerprinting

Dieser Abschnitt richtet sich im wesentlichen nach [9]. In dem Paper wurde das Verfahren in einer künstlichen Umgebung und mit einem sehr kleinen Roboter getestet. Diese Arbeit überträgt das Verfahren in eine Realumgebung.

Linåker und Ishikawa haben ein Verfahren vorgestellt, das mit Hilfe von Autokorrelations-Masken Signaturen von Grauwertbildern erzeugt, die für Menschen nicht wahrnehmbare Merkmale beschreiben. Die Signatur eines Bildes ist ein 35-Tupel aus den Ergebnissen der Anwendung der Masken auf das Bild (siehe Abbildung 3.3).

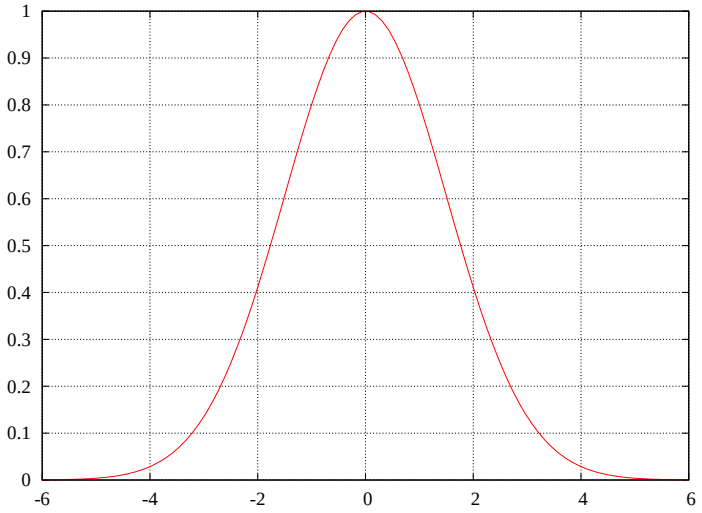

Abbildung 3.2: Normalverteilung mit $\sigma = 1,5$ und einem Faktor mulipliziert, so dass das Maximum bei Eins liegt.

Jede Maske wird an jeder möglichen Position im Bild angewendet, d.h. die in den Matrizen angegebenen Werte mit den Pixelwerten des Bildes multipliziert. Die 35 Summen der Anwendungen der Matrizen ergeben den Merkmalsvektor (Signatur). Damit die Anwendung der Masken auf Panoramabilder rotationsunabhängig ist, wird das Panoramabild wie ein Zylinder behandelt; linker und rechter Rand werden miteinander verknüpft. Da eine Rotation nur eine horizontale Verschiebung der Panoramabilder zur Folge hat, ist die Bildsignatur zweier gleicher Panoramabilder, die lediglich horizontal zueinander verschoben sind, gleich. Unterschiede entstehen nur durch das Rauschen bei der Bildaufnahme.

In Abbildung 3.4 ist dargestellt, wie die Masken radial direkt auf omnidirektionale Bilder angewendet werden können, um die Rotationsunabhängigkeit zu erhalten. Für die korrekte Anwendung müssen die Spiegelparameter bekannt sein und berücksichtigt werden. Für die Anwendung der Masken für diese Diplomarbeit wurden die omnidirektionalen Bilder in kleine grauwertige Panoramabilder umgerechnet, um auf die kompliziertere Anwendung der Autokorrelations-Masken auf omnidirektionale Bilder zu verzichten.

Man nimmt nun an, dass Bilder, die an verschiedenen Positionen aufgenommen wurden, eine für diese Position eindeutige Signatur bekommen. Zwei Bilder, die in unmittelbarer Umgebung zueinander gemacht werden, unterscheiden sich in ihrer Signatur idealerweise nur gering voneinander.

In einer anfänglichen Lernphase muss der Roboter seine Umgebung referenzieren. Die Testfahrten fanden in einer sechs mal drei Meter großen Fläche statt, die mit einem Netz aus Referenzpunkten überzogen wurde (Grid), die 20 cm auseinander liegen. Während der Trainigsphase wurde jeder der Gridpunkte angefahren, um dort ein Referenzbild aufzunehmen und dessen Signatur zu bestimmen. Die Entfernung der Gridpunkte muss wohl überlegt sein. Eine Halbierung der Entfernung hätte eine quadratische Zunahme der Gridpunkte zur Folge. Dieses bedeutet zum einen einen

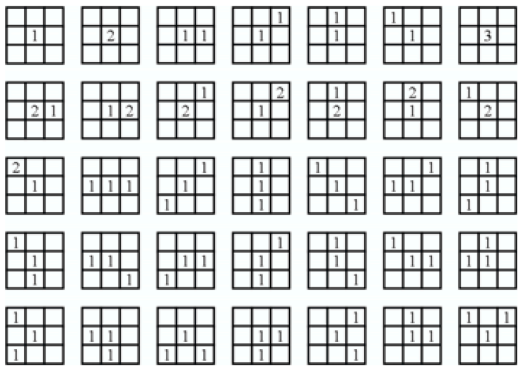

Abbildung 3.3: Autokorrelations-Masken zum Berechnen der 35 Eigenschaftswerte

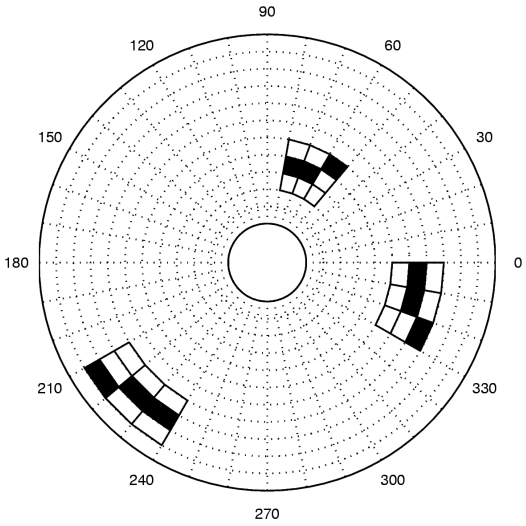

Abbildung 3.4: Anwendung der Masken auf ein omnidirektionales Bild. Die Form muss auf die Spiegelparameter abgestimmt werden. Der dargestellte Spiegel hat eine gleichmäßige Dichte von Innen nach Außen.

erhöhten Aufwand in der Bestimmung der nächsten Signatur aus der Referenzdatenbank. Zum anderen hat dieses speziell für die Anwendung des Verfahrens in dieser Arbeit zur Folge, dass sich die Anzahl der Ausgänge des neuronalen Netzes (siehe unten) quadriert, was erhebliche Auswirkung auf Trainingsdauer und Abfrage hat. Lägen die Gridpunkte zu weit auseinander, dann verlören sie ihren Nutzen.

Sollte sich dieses Verfahren tatsächlich als robust genug erweisen, dem Roboter die nächste Position im Grid zu seinem Standort zu geben, wären 20 cm ausreichend zur Initialisierung des Kalman-Filters der Referenzlokalisierung. Wodurch ebenso das manuelle Kalibirieren entfällt, sowie eine Möglichkeit der Überwachung entsteht, um zum Beispiel das *Kidnapped-Robot*-Problem [2] zu lösen, wozu ein Kalman-Filter alleine nicht in der Lage wäre.

Nachdem alle Gridpunkte angefahren und die Signaturen berechnet wurden, kann mit der Lokalisierungs-Phase begonnen werden. Von der Kamera gelieferte Bilder werden wie die Referenzbilder in Panorama-Grauwertbilder umgewandelt und ihre Signatur mit den Signaturen der Referenzbilder verglichen. Der Fehler zwischen zwei Signaturen sei definiert als der euklidische Abstand der Signaturen. Es wird vermutet, dass der tatsächliche Aufenthaltsort in der Nähe der Position des Referenzbildes mit dem kleinsten Fehler liegt.

Um die Ergebnisse des Algorithmus in den Partikelfilter zu integrieren, wird hier der

[2] *Kidnapped robot* bedeutet, dass die *Pose* des Roboters verändert wird, ohne dass seine Sensoren etwas bemerken. Beispiel: Ein kleiner Roboter hat sich erfolgreich lokalisiert, dann wird er aufgehoben und an einen anderen Ort gestellt. Seine Umwelt befindet sich plötzlich in einer unbekannten Konfiguration, die nicht zu den vorherigen Sensoreindrücken passt.

gleiche Ansatz verfolgt wie in [9]. Während der Lernphase wurde ein *Radial-Basis-Function-Net* (RBF-Netz) mit 35 Eingängen, 60 unsichtbaren Knoten und 441 Ausgängen trainiert. Dabei steht jeder Ausgang für eine Referenzposition im Grid. Das Netz wird so trainiert, dass es nur an dem Ausgang einer Referenzsignatur Eins ausgibt, wenn die entsprechende Referenzsignatur an den Eingängen gegeben wird. Um das RBF-Netz robuster zu trainieren, sollten die Signaturen verschiedener Bilder an einer Gridposition verwendet werden. Durch die Rotationsinvarianz ist die Orientierung bei der Bildaufnahme zwar vernachlässigbar, wohingegen unterschiedliche Beleuchtung und Raumkonfigurationen die Bildsignatur beeinflussen.

Die Verwendung eines neuronalen Netzes, um zu einer Bildsignatur die korrekte Referenzposition zu finden, verbessert dieses Verfahren. Das neuronale Netz lernt während des Trainings eine Gewichtung des Merkmalverktors. Es kann zum Beispiel möglich sein, dass die vierte Stelle mehr Aussagekraft besitzt als die achte. Das neuronale Netz ist in der Lage, diese Zusammenhänge zu lernen und bei der Referenzpositionssuche zu berücksichtigen.

Die Schätzung einer *Pose* für den Partikelfilter findet nur statt, wenn ein neues Bild der Kamera zur Verfügung steht. Für die Schätzung wird die nächste Referenzposition im Grid zur *Pose* des gefragten Partikels gefunden. Als Bewertung für den Partikel wird die Ausgabe des RBF-Netzes für die Signatur des aktuellen Bildes am Ausgang der Referenzposition zurück gegeben.

3.3 Tracking

Das folgende Verfahren wurde in [25] erarbeitet und für diese Arbeit adaptiert. Es versucht, Farb- und Struktureigenschaften in aufeinander folgenden Bildern zu verfolgen und aufgrund ihrer Verschiebung im Bild die Positionsänderung des Roboters mittels Triangulation zu bestimmen. Für die Triangulation ist nur der Winkel, in dem die Eigenschaften erkannt wurden, von Bedeutung. Daher beschränkt sich dieses Verfahren darauf, ganze Bildspalten in aufeinander folgenden Bildern einander zuzuordnen. Die Verschiebung der Spalten ist gleichbedeutend mit einer Veränderung des Sichtwinkels. Durch Triangulation kann die Veränderung des Winkels zur Bestimmung der Positionsveränderung des Roboters genutzt werden.

3.3.1 Berechnung der Bildsignatur

Für die Bildsignatur wird jede Spalte vollständig und überschneidungsfrei in m einzelne Blöcke zerlegt, die jeweils eine Länge von 20 Pixeln haben. Für jeden Block wird ein Eigenschaftsvektor c_n berechnet:

$$c_n = \begin{pmatrix} r_\sigma \\ g_\sigma \\ b_\sigma \\ r_\delta \\ g_\delta \\ b_\delta \end{pmatrix}$$

r_σ, g_σ, b_σ sind die Summen der Farbwerte der drei Farbkanäle in einem Block. Die drei Komponenten $r_\delta, g_\delta, b_\delta$ bestehen aus der Summe der Absolutwerte der Differenzen von zwei benachbarten Pixeln in einem Block, ebenfalls für je einen Farbanteil. Sei P die Menge der Pixel eines Blockes, dann bezeichnet P_n^r den Rot-, P_n^g den Grün- und P_n^b den Blauanteil des n-ten Pixels. Für die Anzahl k der Pixel eines Blockes seien

$$\begin{aligned}
r_\sigma &= \sum_{n=1}^{k} P_n^r \\
g_\sigma &= \sum_{n=1}^{k} P_n^g \\
b_\sigma &= \sum_{n=1}^{k} P_n^b \\
r_\delta &= \sum_{n=1}^{k-1} |P_n^r - P_{n+1}^r| + |P_k^r - P_{k-1}^r| \\
g_\delta &= \sum_{n=1}^{k-1} |P_n^g - P_{n+1}^g| + |P_k^g - P_{k-1}^g| \\
b_\delta &= \sum_{n=1}^{k-1} |P_n^b - P_{n+1}^b| + |P_k^b - P_{k-1}^b|
\end{aligned}$$

Das erste Tripel ist ein Maß für die Farbgebung, das zweite ein Maß für die Textur jedes Blockes. Die Folge von $c_1 \ldots c_m$ der so generierten Vektoren für eine Bildspalte wird als charakterisierend für deren Farbgebung und Textur angenommen:

$$C = \begin{pmatrix} c_1^T \\ c_2^T \\ \vdots \\ c_m^T \end{pmatrix} = \begin{pmatrix} r_\sigma^1 & g_\sigma^1 & b_\sigma^1 & r_\delta^1 & g_\delta^1 & b_\delta^1 \\ r_\sigma^2 & g_\sigma^2 & b_\sigma^2 & r_\delta^2 & g_\delta^2 & b_\delta^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ r_\sigma^m & g_\sigma^m & b_\sigma^m & r_\delta^m & g_\delta^m & b_\delta^m \end{pmatrix}$$

Um die Ähnlichkeiten zwischen zwei Vektoren c_n zu bestimmen, wird eine Metrik d definiert:

$$\begin{aligned} d(c,c') &= 0.3 \cdot |r_\sigma - r'_\sigma| + 0.59 \cdot |g_\sigma - g'_\sigma| + 0.11 \cdot |b_\sigma - b'_\sigma| + \\ &\quad 20 \cdot (|r_\delta - r'_\delta| + |g_\delta - r'_\delta| + |b_\delta - b'_\delta|) \end{aligned}$$

Die Gewichtungen der ersten Zeile sind an die Umrechnung von Farb- in Schwarzweißbilder angelehnt und nähern sich der menschlichen Farbwahrnehmung an. Die Gewichtung der zweiten Zeile sorgt dafür, dass Fehler in der Textur stärkeren Einfluss auf den Gesamtfehler $d(c, c')$ haben.

Die Ähnlichkeit D zweier Bildspalten ist definiert als die Summe der mit dieser Metrik berechneten Abstände zwischen je zwei charakterisierenden Vektoren zweier sich entsprechender Blöcke. Für zwei Bildspalten mit den Matrizen C und C' also:

$$D(C, C') = \sum_{n=1}^{m} d(c_n \cdot c'_n) \tag{3.1}$$

Man kann natürlich auch andere merkmalextrahierende Funktionen, die auf einer Bildspalte definiert sind, zur Berechnung der Ähnlichkeit heran ziehen, wie etwa die Auswertung eines Histogramms über die einzelnen Farbkanäle. Die hier gewählten Funktionen haben den Vorteil, aussagekräftig zu sein und lassen sich schnell berechnen.

3.3.2 Interpretation der Ähnlichkeitsmatrix

Aus den Bildsignaturen C und C' zweier zu vergleichender Bilder wird eine Ähnlichkeitsmatrix gebildet. Dafür wird der Abstand für jede Kombination aus Spaltenpaaren der beiden Bilder mit Gleichung 3.1 berechnet. Sollte keine Bewegung des Roboters stattgefunden haben, dann sind C und C' gleich. Die Diagonale besteht in dem Fall aus großen Werten, während der Rest größtenteils verschwindet. Jedoch nur, wenn die Bilder genügend Textur enthalten, ansonsten ist die ganze Matrix mit großen Werten belegt.

Hat zwischen den beiden Bildern eine leichte Drehung statt gefunden, dann ist die Diagonale weiterhin sichtbar, jedoch nach links oder rechts - je nach Drehrichtung - verschoben (siehe Abbildung 3.5).

Bei einer Vorwärts- oder Rückwärtsbewegung haben die Spalten bei 0° und 180° ihr Maximum immer noch auf der Diagonalen. Die Bildspalten, die den Bereich seitlich des Roboters abbilden, werden sich jetzt aber nach hinten bzw. vorne bewegen. Die ursprüngliche Diagonale wird also eine leichte S-Form einnehmen (siehe Abbildung 3.6).

Natürlich verfälschen Objekte, die sich im Bild bewegen, ohne dass eine Roboterbewegung statt gefunden hat, die Ähnlichkeitsmatrix. Dadurch jedoch, dass sehr viele Merkmale - verbunden mit dem großen Blickwinkel der Panoramabilder - berücksichtigt werden, wird vermieden, dass solche Objekte die Berechnung zu stark beeinflussen. Dennoch wäre ein Roboter, der dieses Verfahren primär zur Lokalisierung

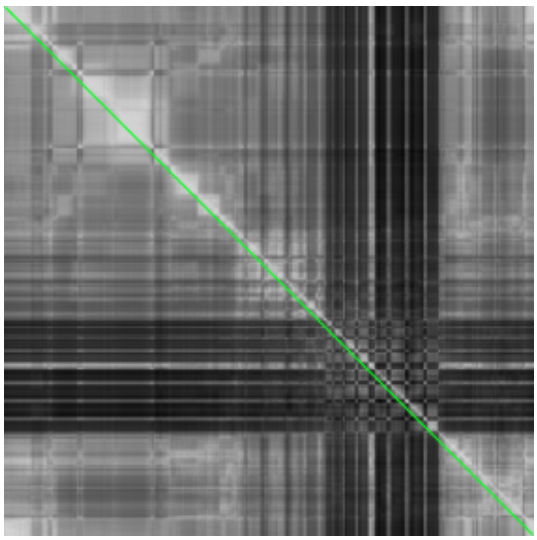

Abbildung 3.5: Ähnlichkeitsmatrix: zwischen den zu vergleichenden Bildern hat eine Rotation statt gefunden. Die Diagonale ist grün eingezeichnet.

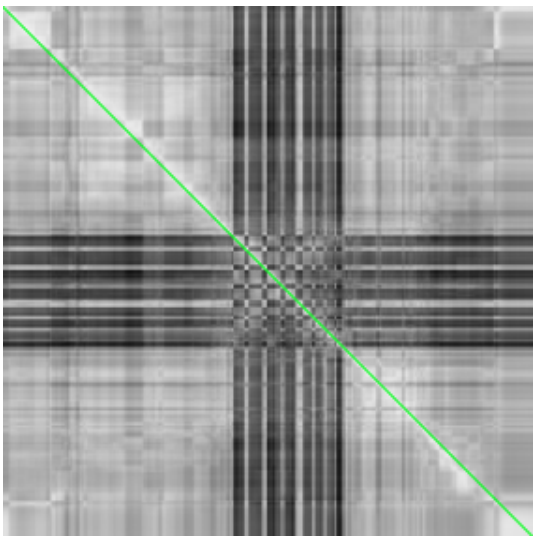

Abbildung 3.6: Ähnlichkeitsmatrix: zwischen den zu vergleichenden Bildern hat eine Translation statt gefunden. Die Diagonale ist grün eingezeichnet.

benutzt, in einer sehr unruhigen Umwelt nicht in der Lage, seine *Pose* verlässlich zu bestimmen.

3.3.3 Berechnung des Übersetzungsvektors

Für die spätere Bestimmung der Positionsverschiebung durch Triangulation sind nur die Änderungen der Winkel der einzelnen Spalten von Interesse. Deswegen wird aus der Ähnlichkeitsmatrix ein Übersetzungsvektor erzeugt, der diese Winkeländerung abbildet. Die Länge des Übersetzungsvektors entspricht der Anzahl der Spalten im Panoramabild, dessen n-tes Element einer Spalte im neueren der zu vergleichenden Panoramabilder entspricht. Im Element wird dabei die Spaltennummer der ähnlichsten Spalte im vorhergehenden Bild gespeichert. Aus k aufeinander folgenden Bildern lassen sich also $k-1$ Übersetzungsvektoren generieren, die aussagen, unter welchen Sichtwinkeln ein Merkmal über die Zeit wieder erkannt werden konnte.

Ein Übersetzungsvektor wird wie folgt gewonnen: für jede Spalte der Matrix wird das Maximum der darin gespeicherten Elemente gesucht. Liegt dieses über einem Schwellwert, dann wird die betreffende Zeilennummer im Übersetzungsvektor an der Stelle der überprüften Spalte gespeichert. Fällt das Maximum unter den Schwellwert, wird ein Ausnahmewert gespeichert, der bedeutet, dass ein Merkmal nicht deutlich genug wieder erkannt werden konnte.

3.3.4 Schätzung für den Partikelfilter

Für die Schätzung werden das aktuelle Bild und die letzten zwei Bilder mit in die Berechnung einbezogen. Zwischen den drei Bildern werden wie oben beschrieben die

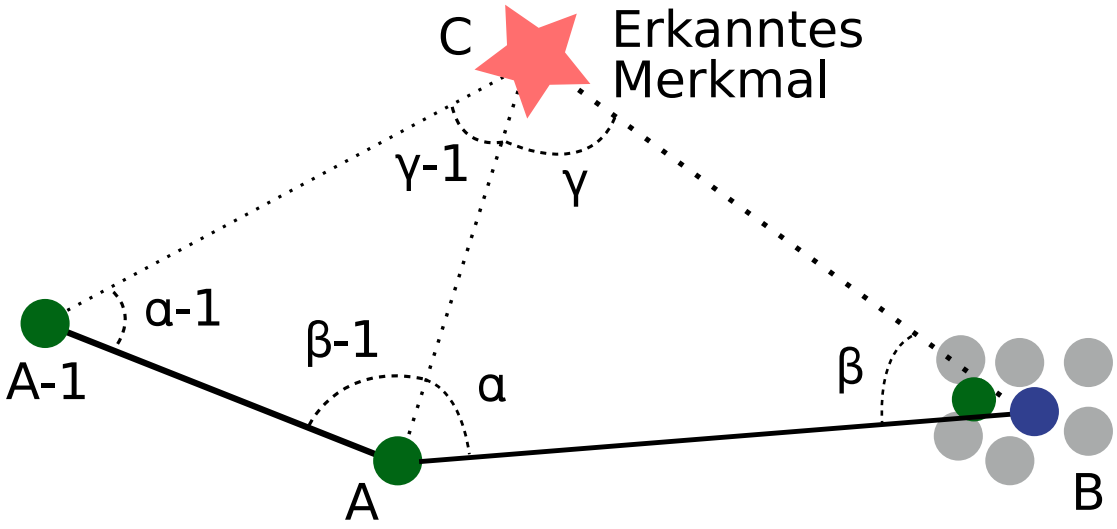

Abbildung 3.7: A: *Pose*, die als Aufenthaltsort der letzten Runde bestimmt wurde. B: Grün der Punkt, der angefahren wurde. Blau der Partikel, dessen Gewichtung geschätzt wird. Grau deutet die anderen Partikel der Partikelwolke an. α, α_{-1}, β und β_{-1} sind die Winkel, in denen das Merkmal im Bild erkannt wurde.

Übersetzungsvektoren bestimmt. Mithilfe der zu den beiden alten Bildern bestimmten *Poses* A und A_{-1} (siehe Abbildung 3.7) kann nun per Triangulation die Position des erkannten Merkmals bestimmt werden. Die Strecken $\overline{A_{-1}A}$ und $\overline{AB}$ ergeben sich aus den euklidischen Abständen der Positionen. Für das aktuelle Bild wird immer die *Pose* des zu schätzenden Partikels benutzt.

Die unbekannten Winkel ergeben sich aus

$$\begin{aligned} \gamma &= 180 - (\alpha + \beta) \\ \gamma_{-1} &= 180 - (\alpha_{-1} + \beta_{-1}) \end{aligned}$$

Dann sind die Strecken

$$\begin{aligned} \overline{AC} &= \frac{\overline{AB} \cdot sin\beta}{sin\gamma} \\ \overline{A_{-1}C} &= \frac{\overline{A_{-1}A} \cdot sin\beta_{-1}}{sin\gamma_{-1}} \end{aligned}$$

Aus den Orientierungen θ der Positionen A_{-1}, A und B im Weltkoordinatensystem kann man nun die x- und y-Verschiebung in Weltkoordinaten für das erkannte Merkmal bestimmen. So erhält man zwei Positionen $pose^c$ und $pose^c_{-1}$, an denen das erkannte Merkmal durch die Triangulation vermutet wird. Jeweils eine für die

Berechnung durch die beiden älteren Bilder und durch das aktuelle und das Bild davor.

Für diese beiden Positionen wird der Fehler als der euklidische Abstand zwischen beiden bestimmt. Dieser Fehler wird für alle Merkmale, die in allen drei Bildern wieder erkannt wurden, berechnet und gemittelt. Dadurch ergibt sich ein gemittelter Abstand, der angibt, wie gut die *Pose* des zu schätzenden Partikels zu dem aufgenommenen Bild passt.

Dieser Fehler wird linear umgerechnet, so dass ein Abstand von null Zentimetern auf eine Wahrscheinlichkeit von Eins und ein Abstand von zehn Zentimetern auf eine Wahrscheinlichkeit von Null abgebildet wird.

3.4 Zusammenfassung

In diesem Kapitel wurde die Implementierung des Partikelfilters und der Partikelfilterquellen vorgestellt, die im nächsten Kapitel in den Experimenten untersucht werden. Folgende Konfigurationen der Implementierung werden untersucht:

- Berechnung nur mit der Odometrie
- Berechnung nur mit der Referenzlokalisierung (*best case*)
- Berechnung nur mit dem Fingerprinting-Verfahren
- Berechnung nur mit dem RED-Verfahren
- Berechnung nur mit dem Tracking-Verfahren
- und Berechnung mit Fingerprinting-, RED- und Trackingverfahren

Die ersten beiden Konfigurationen zeigen die Genauigkeit der Lokalisierung für den schlimmsten bzw. besten Fall. Mit den anderen Konfigurationen wird die Genauigkeit der verschiedenen Verfahren zur Lokalisierung mittels optischer Sensoren getestet. Im letzten Fall werden alle drei Verfahren gleichzeitig eingesetzt, mehr dazu im nächsten Kapitel.

4 Analyse

In diesem Kapitel werden die verschiedenen Partikelfilterkonfigurationen untersucht und miteinander verglichen. An entsprechenden Stellen werden Begründungen für festgelegte Schranken und Schwellwerte gegeben. Zu Beginn des Kapitels werden die Parameter erläutert, die für die Berechnungen des Partikelfilters verwendet wurden. Insgesamt werden sechs verschiedene Programmkonfigurationen untersucht und ihre Berechnungsabläufe beschrieben.

Zuerst wird in Kapitel 4.1 die Genauigkeit der Lokalisierung ausschließlich mit Odometriedaten diskutiert. Dieses stellt den schlechtesten Fall dar, in dem die *Pose*-Schätzung nicht durch die Verarbeitung weiterer Sensormessungen unterstützt wird. Danach wird in Kapitel 4.2 das genaue Gegenteil untersucht: die Referenzlokalisierung als alleinige Partikelfilterquelle. Dieses stellt den *best case* dar, in dem die einzige Partikelfilterquelle die genaue *Pose* kennt und dementsprechende Bewertungen für Partikel-*Poses* abgibt. In den darauf folgenden drei Kapiteln wird der Einfluss verschiedenen Verfahren zur Lokalisierung mittels optischer Sensoren als Verbesserung der reinen Odometrie-Lokalisierung analysiert. Abschließend werden in Kapitel 4.6 die Ergebnisse der Berechnungen miteinander verglichen. Zusätzlich findet die Berechnung einer weiteren Konfiguration statt, in der alle drei behandelten Lokalisierungs-Verfahren als Partikelfilterquelle eingesetzt wurden. Diese wird in Kapitel 4.6 mit in den Vergleich einbezogen.

Es gibt zwei verschiedene Fälle der Lokalisierung, die sich darin unterscheiden, an welchen Punkten eine genaue Lokalisierung notwendig ist. In einem Fall wird nur ein Zielpunkt vorgegeben, zu dem das mobile System fahren muss. Auf welchem Pfad das Ziel erreicht wird, ist nicht weiter definiert. Das mobile System muss ausreichend mit lokalen Sensoren und Algorithmen ausgestattet sein, um den Weg zum Ziel zu finden und während der Fahrt Kollisionen zu vermeiden. Erst am Zielpunkt wird die genaue Lokalisierung wieder wichtig, um die vorgegebene Position möglichst exakt zu erreichen.

Im zweiten Lokalisierungsfall wird dem Roboter ein geplanter Pfad vorgegeben, dem er folgen muss. In diesem Fall ist das Wissen der Position zu jedem Zeitpunkt wichtig, um zu erkennen, wenn vom geplanten Pfad abgewichen wird. In dieser Diplomarbeit wird dieser zweite Lokalisierungsfall behandelt. Um vergleichbare Daten zu produzieren, wurde eine Standardfahrt festgelegt, die aus mehreren „U“s besteht (siehe Abbildung 4.1). Die Gesamtlänge der Fahrt beträgt ungefähr 25 m.

Zum Vergleich der verschiedenen Konfigurationen des Partikelfilters, muss ein Maß der Genauigkeit eingeführt werden. Der Fehler zwischen zwei Positionen sei definiert als der euklidische Abstand zwischen ihnen. Der Fehler in der Orientierung

sei definiert als der Absolutwert der Differenz zweier Orientierungen. So kann man zu jeder Runde die vom Partikelfilter bestimmte *Pose* mit der Referenz-*Pose* der Runde von der Referenzlokalisierung vergleichen und einen Fehler bestimmen. Fehler in der Position und der Orientierung werden aus Gründen der Anschaulichkeit getrennt voneinander betrachtet.

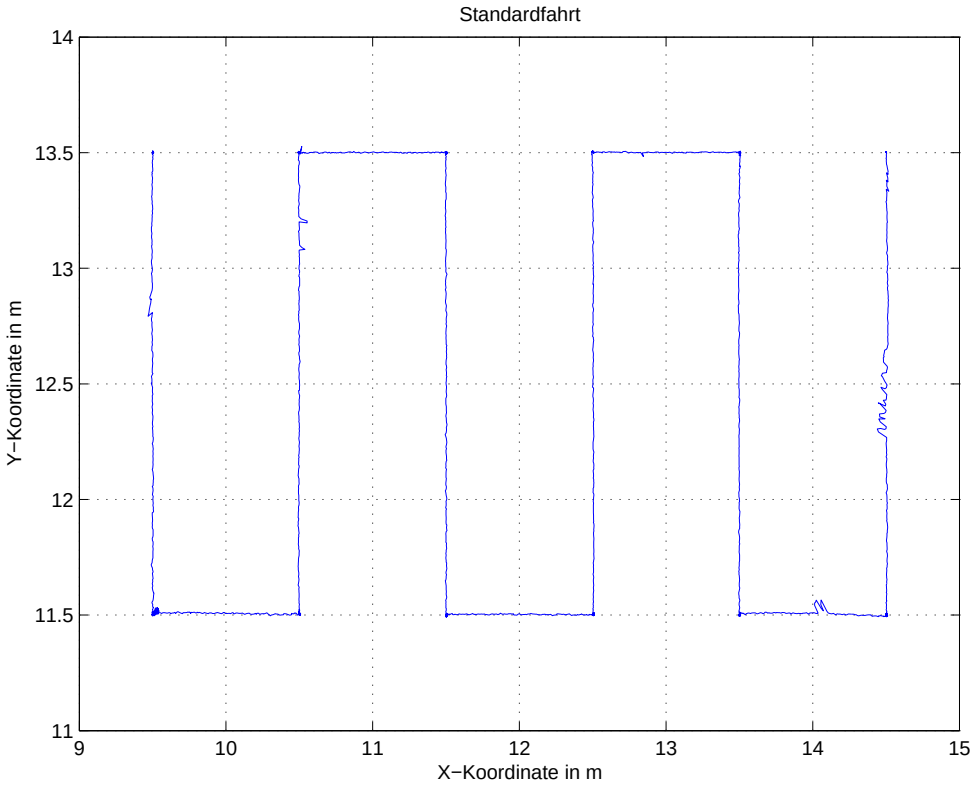

Abbildung 4.1: Eine der 10 durch die Referenzlokalisierung aufgezeichneten Standardfahrten. Man kann erkennen, dass die Referenzlokalisierung zum Teil Schwankungen unterliegt. Zum Beispiel die starke Abweichung unten rechts - oder ganz rechts auf halber Höhe - sind Fehler in der Referenzlokalisierung, die in allen 10 Aufnahmen zu finden sind. Der Roboter ist an den Stellen der geraden, gegbenen Trajektorie gefolgt. Die Fehler entstehen durch Verdeckung von Landmarken, die für die Referenzlokalisierung notwendig sind, durch Tischbeine, Mülleimer oder andere dauerhafte Hindernisse. Die verbleibenden erkannten Landmarken reichen an diesen Stellen zur robusten Lokalisierung nicht aus.

Alle Berechnungen wurden offline auf vorher aufgezeichneten Fahrten durchgeführt. Der Roboter hat die Standardfahrt zehnmal direkt hintereinander ausgeführt, wobei die Daten der Odometrie, die Daten der Referenzlokalisierung und die Bilder der Kamera aufgezeichnet wurden. Damit ist sichergestellt, dass die Daten einer Fahrt nicht zufällig ein Verfahren zur Lokalisierung begünstigen und das Ergebnis verzerren. Außerdem wurde für jede Partikelfilterkonfiguration jede Fahrt 50 mal berechnet, um einen repräsentativen Mittelwert der Genauigkeit zu ermitteln. Insgesamt setzt sich

also das Ergebnis jeder Parameterkonfiguration aus 500 Berechnungen zusammen. Außerdem werden diese Berechnungen je einmal für jede Art der Partikelwahl (siehe Seite 26) ausgeführt und deren Ergebnisse miteinander verglichen.

In den Berechnungen wurden maximal 80 Partikeln benutzt. Pro Runde durften höchstens 4 Partikel gelöscht werden, sobald weniger als 64 Partikel aktiv waren, wurden neue erzeugt, so dass insgesamt wieder 80 Partikel aktiv waren. Die Anzahl der Gesamtpartikel hat direkte Auswirkung auf die Berechnungsdauer. Bei Versuchen mit verschiedenen Werten für die maximale Anzahl von Partikeln hat sich ergeben, dass 80 ein guter Kompromiss zwischen Berechnungsdauer und Genauigkeit der Lokalisierung ist. Zumal für den speziellen Lokalisierungsfall in dieser Diplomarbeit, in dem Multimodalität und das *Kidnapped-Robot*-Problem nicht untersucht werden, eine deutlich höhere Anzahl von Partikeln nur wenig Auswirkung auf die Genauigkeit hat (vgl. Kapitel 5).

Die Standardabweichung der eindimensionalen Gauss-Verteilung zum Verrauschen der Odometrie für die Vorhersage im Partikelfilter (siehe Seite 29) wurde auf 0,3 festgelegt. Die Versuche haben gezeigt, dass Änderungen der Standardabweichung den Fehler der Lokalisierung nur für die „Bester-Partikel"-Auswahlstrategie signifikant beeinflusst haben (siehe Seite 56 ff.).

4.1 Odometrie als Partikelfilterquelle

In der ersten getesteten Konfiguration berechnet der Partikelfilter die Fahrten nur auf Basis der Odometrie, es stehen keine weiteren Quellen zur Verfügung. Die Ergebnisse sollen zeigen, wie sich der Roboter ohne weitere Sensordatenverarbeitung lokalisiert. Sie dienen später als Vergleichbasis für weiterer Programmkonfigurationen.

Wie zu Beginn des Kapitels beschrieben, wurde die Referenzfahrt zehnmal vom Roboter ausgeführt und aufgezeichnet. Um den Einfluss zufälliger Ereignisse zu minimieren, die das Ergebnis der Lokalisierung beeinflussen, wird jede Fahrt 50 mal berechnet und der Mittelwert bestimmt. Die so erhaltenen Daten können dann mit der tatsächlichen Position aus der Referenzlokalisierung verglichen werden.

Dazu wird die *Pose*, die der Partikelfilter in jeder Runde bestimmt, mit der tatsächlichen *Pose* des Roboters verglichen. In Abbildung 4.2 ist deutlich zu erkennen, wie der Fehler mit fortschreitender Fahrt immer größer wird. Es wird deutlich, wie schnell eine Lokalisierung nur auf Basis der Odometrie zu großen Fehlern führt. Diese Ergebnisse sind nicht überraschend. Die Ursachen werden ausführlich in [26] beschrieben.

Abbildung 4.2 zeigt, dass die Länge der Fahrtausführungen schwankt. Dies ist eine direkte Folge des fünfstündigen Dauereinsatzes, während diesem die zehn Aufzeichnungen der Standardfahrt gemacht wurden. Der Dauereinsatz bedingt die Schwächung der Bordbatterie und die Erhitzung der Motoren und Bremsen. Beides führt dazu, dass sich die Bremsen teilweise nicht mehr richtig lösen. Der Roboter schlingert dann unter lauter Geräuschentwicklung und wird durch die Fahrtplanung immer

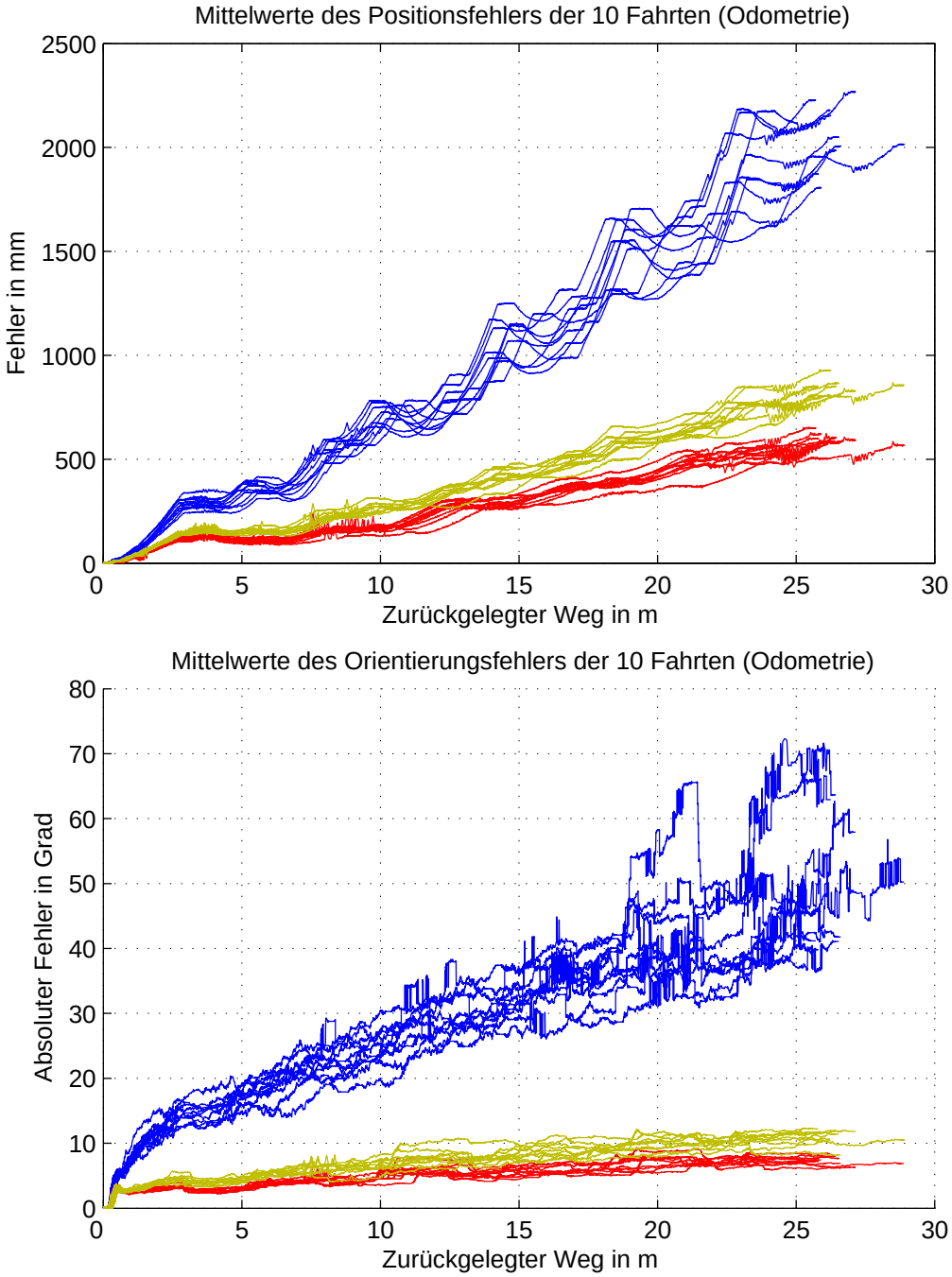

Abbildung 4.2: Mittelwert des Positionierungsfehlers der 10 Fahrten. Farblich gekennzeichnet sind die Ergebnisse der verschiedenen Partikelauswahlalgorithmen. Blau: Bester Partikel, Rot: Schwerpunkt, Orange: Schwerpunkt im Radius um den besten Partikel.

wieder zu Kurskorrekturen gezwungen. Die Auswirkungen werden im nächsten Abschnitt 4.2 deutlich, in dem es um Millimetergenauigkeit geht. Fahrten, die später während des Dauerbetriebes aufgenommen wurden, produzieren dann aufgrund der Ermüdungserscheinungen größere Fehler.

Abbildung 4.2 zeigt einen deutlichen Unterschied in der Genauigkeit der unterschiedlichen Auswahlalgorithmen. Besonders der beste Partikel schneidet schlechter ab. Er ist direkt von der Standardabweichung der Normalverteilung abhängig, die zum Verrauschen der Odometriedaten dient (vgl. Seite 56). Je größer die Standardabweichung, desto größer ist der Fehler des besten Partikels. Außerdem unterliegen die Partikel bei der Lokalisierung ausschließlich mit Odometriedaten keiner weiteren Schätzung, darum haben alle dasselbe Gewicht und der Auswahlalgorithmus für den besten Partikel wählt immer den ersten Partikel aus der Gesamtmenge der Partikel als momentane *Pose* aus.

Da die Partikel in der Vorhersage-Phase mit einem normalverteilten Fehler verschoben werden und die Gewichte der Partikel alle gleich sind, bestimmt das gewichtete Mittel den Mittelpunkt der Partikelwolke. Das ist die *Pose*, die die Odometrie ohne künstliches Rauschen gemessen hat. Das robuste Mittel zeigt ebenfalls Tendenzen, den Wolkenmittelpunkt zu bestimmen, ist aber immer noch schlechter, da nicht alle Partikel bei der Schwerpunktsberechnung berücksichtigt werden.

4.2 Referenzlokalisierung als Partikelfilterquelle

Um zu zeigen, dass der Partikelfilter korrekt implementiert wurde und im besten Fall optimale Ergebnisse liefert, wurde eine Quelle für den Partikelfilter erzeugt, die die Daten der Referenzlokalisierung bereit stellt. Arbeitet der Partikelfilter korrekt, sollte er die gleiche *Pose* bestimmen wie die Referenzlokalisierung oder dieser sehr nahe liegen.

Dem Partikelfilter standen zur Berechnung der Fahrten nur Daten der Referenzlokalisierung als Eingabequelle zur Verfügung. Abbildung 4.3 zeigt den Fehler zwischen Partikelfilterlokalisierung und Referenzlokalisierung. Der Fehler konvergiert, abgesehen von einigen Ausreißern, gegen einen Wert. Der mittlere Fehler über die gesamte Fahrt ist demnach aussagekräftig, weil man davon ausgehen kann, dass er auch bei einer längeren Fahrt gegen den gleichen Wert konvergiert.

In Abbildung 4.4 sind die mittleren Fehler aller Berechnungen der Fahrten aufgetragen. Jeweils 50 gehören immer zu der Berechnung einer Aufzeichnung. Es ist deutlich zu erkennen, wie der Fehler bei späteren Fahrten aufgrund der Dauerbelastung größer ist.

Tabelle 4.1 listet die Mittelwerte und Standardabweichungen für die Fehler der Position und Orientierung für die verschiedenen Partikelauswahlalgorithmen auf. Der Fehler ist zufrieden stellend klein und der Partikelfilter zeigt das erwartete Verhalten.

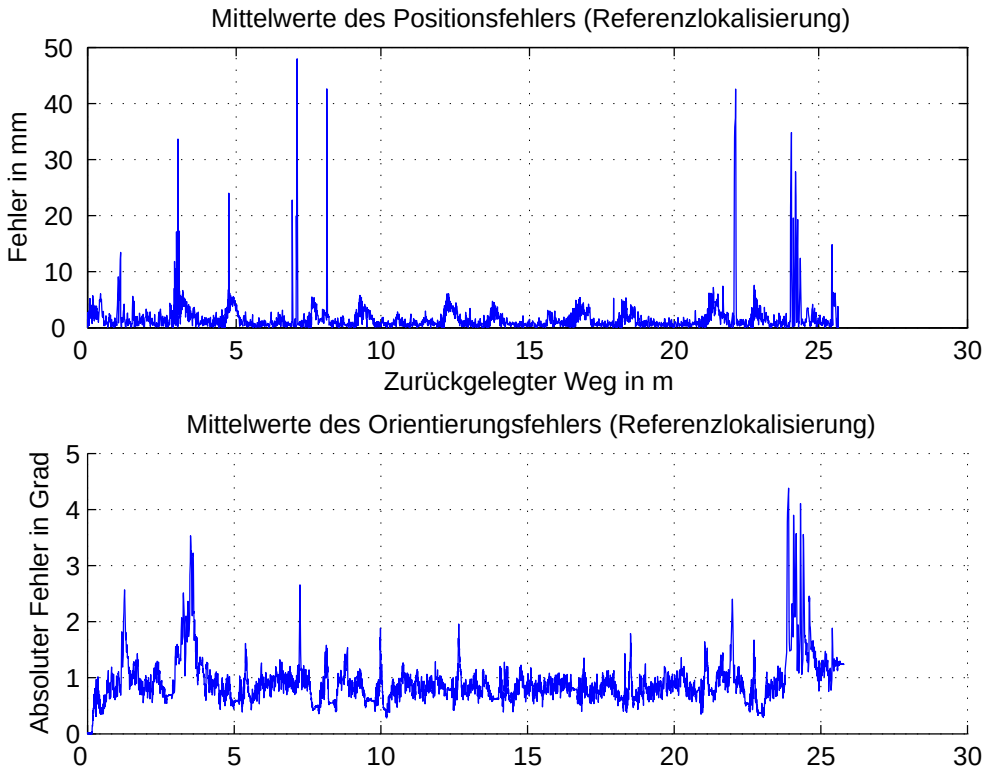

Abbildung 4.3: Mittelwert des Positionierungsfehlers exemplarisch für die Berechnung einer Fahrt. Alle Berechnungen zeigen einen ähnlichen Verlauf, daher wurde zugunsten der Übersichtlichkeit auf deren Darstellung verzichtet.

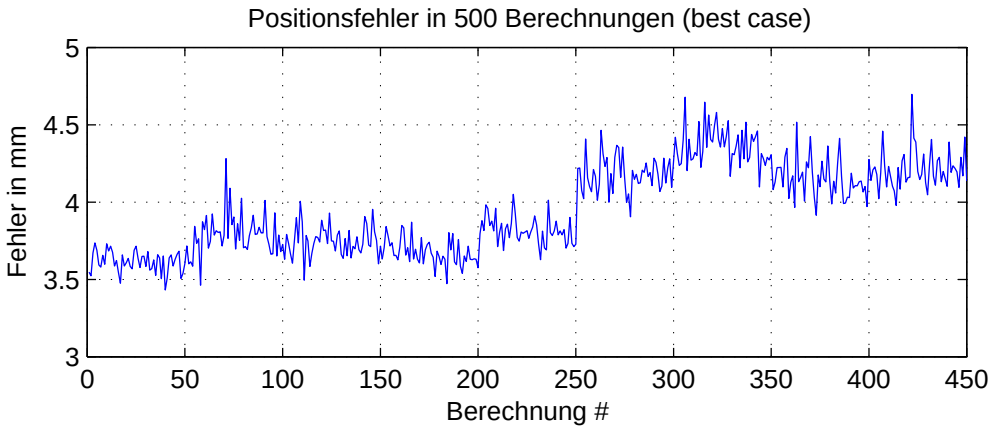

Abbildung 4.4: Mittlerer Fehler jeder Berechnung. Blöcke von je 50 Berechnungen gehören zu einer Fahrtaufzeichnung.

Fehler bei	Best	Schwerpunkt	Schwerpunktradius
Position in mm	3.9633 ± 21.9088	3.4762 ± 37.2501	3.1535 ± 20.6868
Orientierung in Grad	0.8450 ± 0.6942	0.8900 ± 0.8327	0.8798 ± 0.8896

Tabelle 4.1: Mittelwerte der Positions- und Orientierungsfehler über alle 500 Berechnungen für jede Auswahlstrategie.

4.3 Rotationsbestimmung durch vertikale Kanten als Partikelfilterquelle

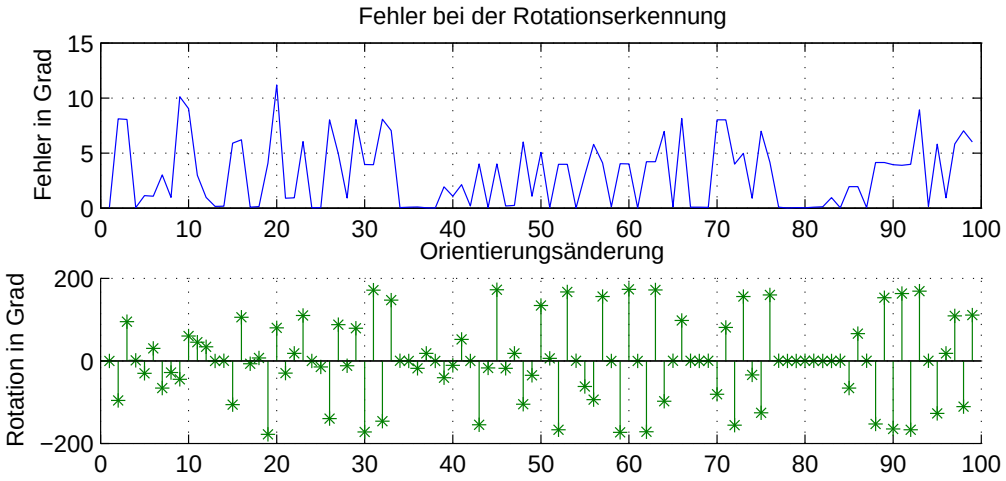

Abbildung 4.5: Testlauf des Algorithmus. 100 zufällige Rotationen und der Fehler bei der Rotationserkennung durch Auswertung der omnidirektionalen Bilder vor und nach der Drehung.

Zur Bestimmung der Genauigkeit des RED-Verfahrens, wurden 100 zufällige Rotationen nacheinander ausgeführt. Jeweils vor und nach der Rotation wurde ein Bild aufgenommen. Abbildung 4.5 stellt den Fehler der Rotationserkennung aus den Bildern im Vergleich zur tatsächlichen Orientierungsänderung dar. In Anbetracht der einfachen Mittel des Algorithmus liefert er gute Ergebnisse. Bei dem Testlauf konnte er die Drehung auf $3,0566 \pm 9,2071$ Grad genau bestimmen.

Als Quelle für den Partikelfilter ist der Algorithmus zwar in der Lage, die Lokalisierung stellenweise zu verbessern, wie man aber Abbildung 4.6 entnehmen kann, ist der Fehler immer noch recht groß.

Dass die Ergebnisse schlecht sind, liegt zum einen daran, dass dieser Algorithmus zu selten Schätzungen liefert. In der Testfahrt gibt es nur zehn Stellen, an denen sich der Roboter dreht. Zu den Fahrtabschnitten dazwischen kann keine Aussage getroffen werden. Eine andere Fehlerquelle liegt in der Positionierung des omnidirektionalen Sichtsystems am Roboter. Es liegt nicht auf dem Mittelpunkt des Roboters, sondern

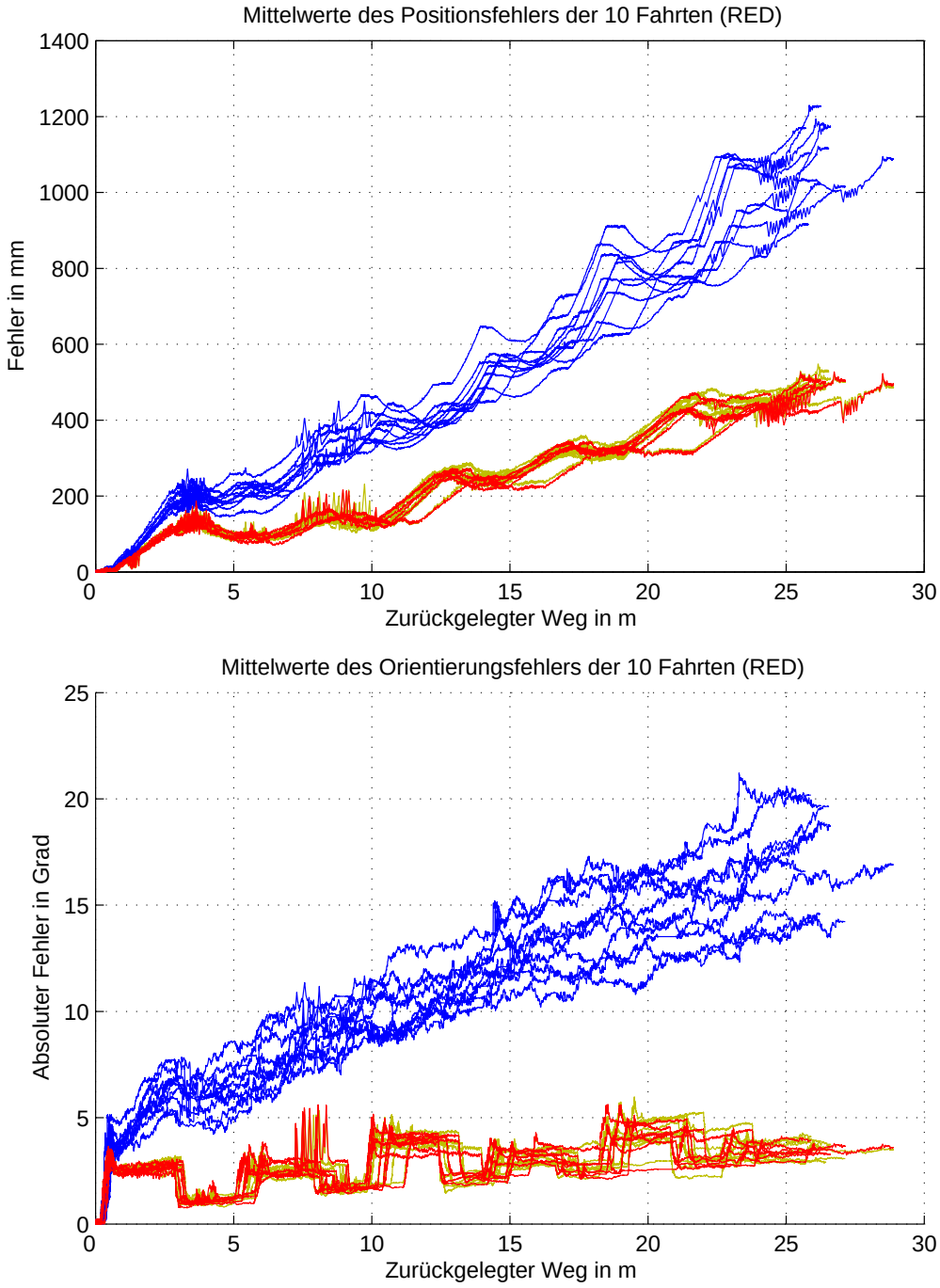

Abbildung 4.6: Positionierungsfehler für den Partikelfilter mit dem *RotationEdgeDetect* Algorithmus. Jeweils die Mittelwerte für die 10 Fahrten. Farblich gekennzeichnet die Ergebnisse der verschiedenen Partikelauswahlalgorithmen. Blau: Bester Partikel, Rot: Schwerpunkt, Orange: Schwerpunkt im Radius um den besten Partikel.

ein Stück dahinter (vgl. Abbildung 1.2). Dadurch führt eine Rotation immer zu einer Positionsänderung des Sichtsystems. Das erklärt, dass bei großen Rotationen in Abbildung 4.5 große Fehler auftreten und bei kleinen Rotationen die Fehler deutlich geringer bleiben. Zum anderen hängt das schlechte Ergebnis mit der Erkennung einer reinen Orientierungsänderung zusammen. Wie bereits in Kapitel 3.1 erläutert, liefert dieses Verfahren nur verwertbare Ergebnisse, wenn zwischen zwei Bildern lediglich eine Rotation stattgefunden hat. Dazu wird ein Schwellwert festgelegt. Liegt die Positionsänderung zwischen zwei Bildern unter diesem, dann wird angenommen, dass sich der Roboter nur gedreht hat. Allerdings ist es möglich, dass die Positionsänderung unterhalb des Schwellwerts liegt und der Roboter doch eine Translation ausgeführt hat. Dieses geschieht insbesondere zu Beginn einer Bewegung, während der Beschleunigung.

Für die Berechnungen mit dem Partikelfilter wurden die gesammelten Daten untersucht und für diese der optimale Schwellwert bestimmt. Sollte eine Positionsveränderungen kleiner als 150 mm sein, wird sie als Rotationen eingestuft. Diese Art der Schwellwertbestimmung ist für die Praxis untauglich. Der optimale Wert für die auszuführende Fahrt kann nicht vorher bestimmt werden. So ist allerdings für den besonderen Fall dieser Arbeit gesichert, dass das Verfahren unter optimalen Bedingungen eingesetzt wird. Das Ergebnis ist dennoch nicht zufrieden stellend, da ungefähr 18% der Bewegungen, die für Rotationen gehalten werden, in Wirklichkeit Translationen sind.

Um diese Fehleinschätzung niedriger zu halten oder ganz auszuschliessen, muss eine andere Methode gefunden werden, die Art der ausgeführten Bewegung zu bestimmen. Am besten wird direkt die Befehlskette des Roboters nach der Bewegung abgefragt, die er zurzeit ausführt. Diese Informationen standen jedoch auf dem mobilen Serviceroboter der Arbeitsgruppe TAMS nicht zur Verfügung. Die dort eingesetzte Pfadplanung empfängt nur Zielkoordinaten und berechnet selbstständig, wie das Ziel angefahren wird. Es gibt keine Möglichkeit abzufragen, welche Bewegung gerade ausgeführt wird.

In Abbildung 4.6 ist der Orientierungsfehler jeder Runde in Grad dargestellt. Für das gewichtete und robuste Mittel ergibt sich eine Art Rechteckverlauf. Die Ecken im Rechteckverlauf sind Stellen der Fahrt, an denen der Roboter eine Drehung ausgeführt hat. In der Standardfahrt (siehe Abbildung 4.1) gibt es zehn solcher Stellen, an denen eine Richtungsänderung stattfindet. Das sind die einzigen Stellen, an denen der RED-Algorithmus - im Idealfall - eine Schätzung abgibt. Bei jeder Drehung werden die Partikel mit der besseren Orientierung θ höher bewertet. Bei der folgenden Geradeausfahrt bleibt dann der Orientierungsfehler annähernd gleich oder steigt bis zur nächsten Drehung - bis auf einige Ausreißer - nur flach an.

4.4 Fingerprinting als Partikelfilterquelle

In [9] wird Fingerprinting als Partikelfilterquelle in einer kleinen künstlichen Umgebung getestet, in der es gute Ergebnisse liefert. In der Laborumgebung, in der die Fahrten dieser Arbeit statt gefunden haben, sind die Resultate deutlich schlechter.

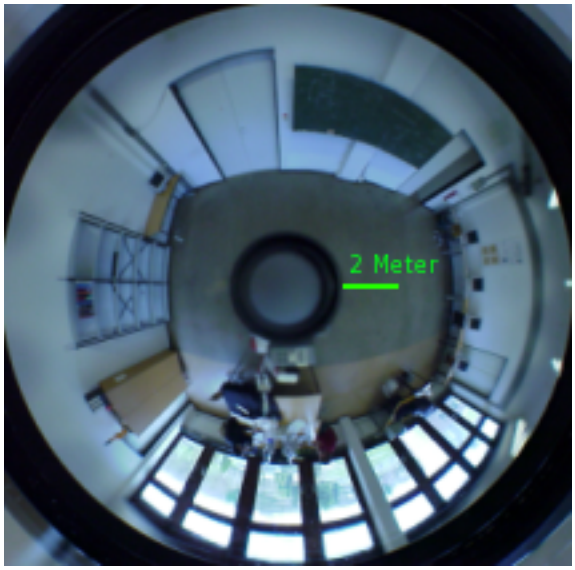

Abbildung 4.7: Omnidirektionales Bild des Laborraumes, in dem die Testfahrten aufgezeichnet wurden.

Abbildung 4.8: Beispielhafte Umrechnung in ein Panoramabild, für das mittels der Autokorrelations-Masken eine Signatur bestimmt wird.

Abbildung 4.7 gibt einen Eindruck vom Labor. Es ist ungefähr neun mal sechs Meter groß und relativ leer. Das führt dazu, dass sich die Bilder sehr ähnlich sehen, die in der Mitte des Raumes aufgenommen werden. Offensichtlich produziert der Algorithmus keine ausreichenden Bildsignaturen, um diese Bilder deutlich genug von einander zu unterscheiden. Um das zu testen, wurden an verschiedenen Positionen im gelernten Grid Aufnahmen gemacht und der Fehler der Signaturen dieser Bilder zu jedem Gridpunkt bestimmt. In Abbildung 4.8 ist ein typisches Panoramabild abgebildet, für das die Bildsignatur mittels der Autokorrelations-Masken bestimmt wurde. Die Bilder sind in schwarz-weiß und nur 60 mal 40 Pixel groß. Bei den Tests hat sich gezeigt, dass kleinere Bilder ein besseres Ergebnis produzieren. Abbildung 4.9 stellt vier der Testergebnisse grafisch dar.

Die vier Testpositionen liegen in der Mitte des Grids. Man kann sehen, dass alle vier Grafiken eine ähnliche Grundstruktur gemeinsam haben. Dieses *Rauschen* folgt aus der Tatsache, dass sich die Aufnahmen in der Mitte des Raumes gleichen. Erst nahe den Wänden, an denen eine auffällige Fensterfront auf der einen und eine große Tafel auf der anderen Seite sind, und in den Ecken, wird der Fehler größer. Es ist zwar zu beobachten, dass das Gebirge in Richtung tatsächlicher Position fällt, aber nicht so eindeutig wie in den Experimenten von [9].

Um diesem Phänomen entgegen zu wirken und einen Mechanismus zu haben, der Positionen im Grid anhand eines Bildes schätzen kann, wurde ein Neuronales Netz trainiert. Dafür wurde wie in [9] ein *Radial-Basis-Function-Net* (RBF-Netz) mit 35 Eingangsknoten, 441 Ausgangsknoten und 60 unsichtbaren Knoten aufgebaut. Die Anzahl der unsichtbaren und Ausgangsknoten wurde angepasst, weil das Grid mehr

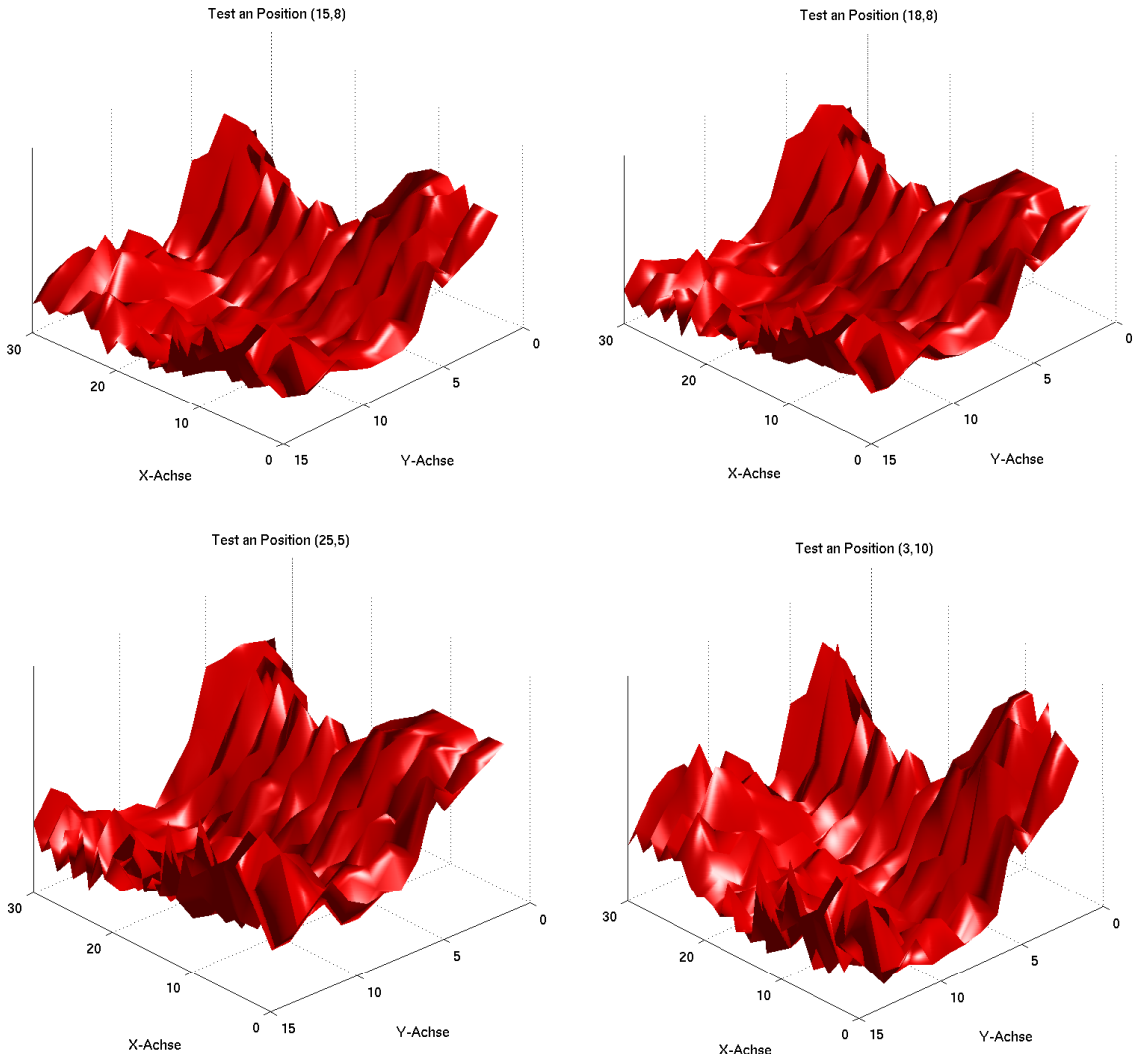

Abbildung 4.9: Fehler zwischen den Bildern der Datenbank und je einem aktuellen Kamerabild an vier verschiedenen Positionen.

Knoten hat als das Grid in [9]. Die Referenzpunkte im Grid wurden nacheinander angefahren und dort Bilder aufgenommen, die im Folgenden Referenzbilder genannt werden. Mit den Signaturen der Referenzbilder wurde das RBF-Netz trainiert, nur am entsprechenden Ausgang des jeweiligen Gridpunkts mit Eins zu antworten.

Während der Trainingsphase, als das Grid abgefahren und die Referenzbilder aufgenommen wurden, kam es zu unregelmäßigen Abstürzen des Kameraservers, deren Ursache noch nicht gefunden und beseitigt werden konnte. Ein Absturz hat zur Folge, dass das mobile System neu gestartet werden muss. Dadurch erzeugt die Aufnahme des gesamten Grids einen hohen Zeitaufwand, weswegen es für diese Arbeit nicht möglich war, Serien von Referenzaufnahmen durchzuführen, die unter verschiedenen Beleuchtungszenarien oder Raumkonfigurationen stattgefunden hätten. Bei den Experimenten wurde deshalb darauf geachtet, dass Beleuchtung und Raumkonfiguration gleich den Bedingungen waren, unter denen die Referenzaufnahmen stattfanden.

In Abbildung 4.10 sind die Fehler der Lokalisierung der verschiedenen Partikelauswahlalgorithmen gegenüber gestellt. Beim Orientierungsfehler zeigt sich kein Rechteckmuster, wie etwa beim RED-Verfahren (siehe Seite 50). Das überrascht nicht, sondern bestätigt, dass die Berechnung der Bildsignaturen durch die Autokorrelations-Masken rotationsunabhängig ist. Dieses Fingerprinting-Verfahren, kann daher keine Aussage über die Qualität der Orientierung der Partikel treffen. Ansonsten zeigt der Verlauf der Kurven keinen signifikanten Unterschied zu den anderen Konfigurationen.

4.5 Tracking als Partikelfilterquelle

Das Tracking hat leider keine guten Ergebnisse geliefert. Die Komplexität des Algorithmus liegt bei $O(2^n)$, wodurch es notwendig war, die Berechnung der Bildeigenschaften auf Panoramabilder der Größe 360 mal 122 Pixel zu beschränken. Dadurch gehen viele Informationen verloren, besonders wenn nur kleine Bewegungen zwischen den Bildern statt gefunden haben. Dann wird die Triangulation mit wenigen Parametern durchgeführt und der Fehler ist entsprechend hoch.

In Abbildung 4.11 wird der Verlauf einer typischen Fahrtberechnung dieses Verfahrens dargestellt. Es ist auffällig, dass entweder viele (über 300) oder deutlich weniger (unter 100) Spalten wieder erkannt werden. Im Mittelfeld finden sich nur einige Ausnahmen. Das hängt mit der Anzahl der Spalten zusammen, die in allen drei Bildern eindeutig wiedergefunden wurden. Zur Erinnerung: Jede Spalte wird in n überschneidungsfreie Blöcke zerlegt, die je 20 Zeilen haben (siehe Kapitel 3.3). Für diese Blöcke wird ein Vektor generiert, der die Eigenschaften des Blocks repräsentiert. Objekte, denen sich der Roboter nähert oder von denen er sich entfernt, wandern in den Bildern hoch bzw. runter. Prägnant gefärbte Flächen können so im ersten Bild im ersten Block zu finden sein, im zweiten in den ersten beiden und im dritten Bild vielleicht nur noch im zweiten Block. Dadurch kann diese Spalte nicht eindeutig genug einer Spalte in den darauf folgenden Bildern zugeordnet werden und fällt aus der Menge, der in allen drei Bildern wiedererkannten Spalten, heraus.

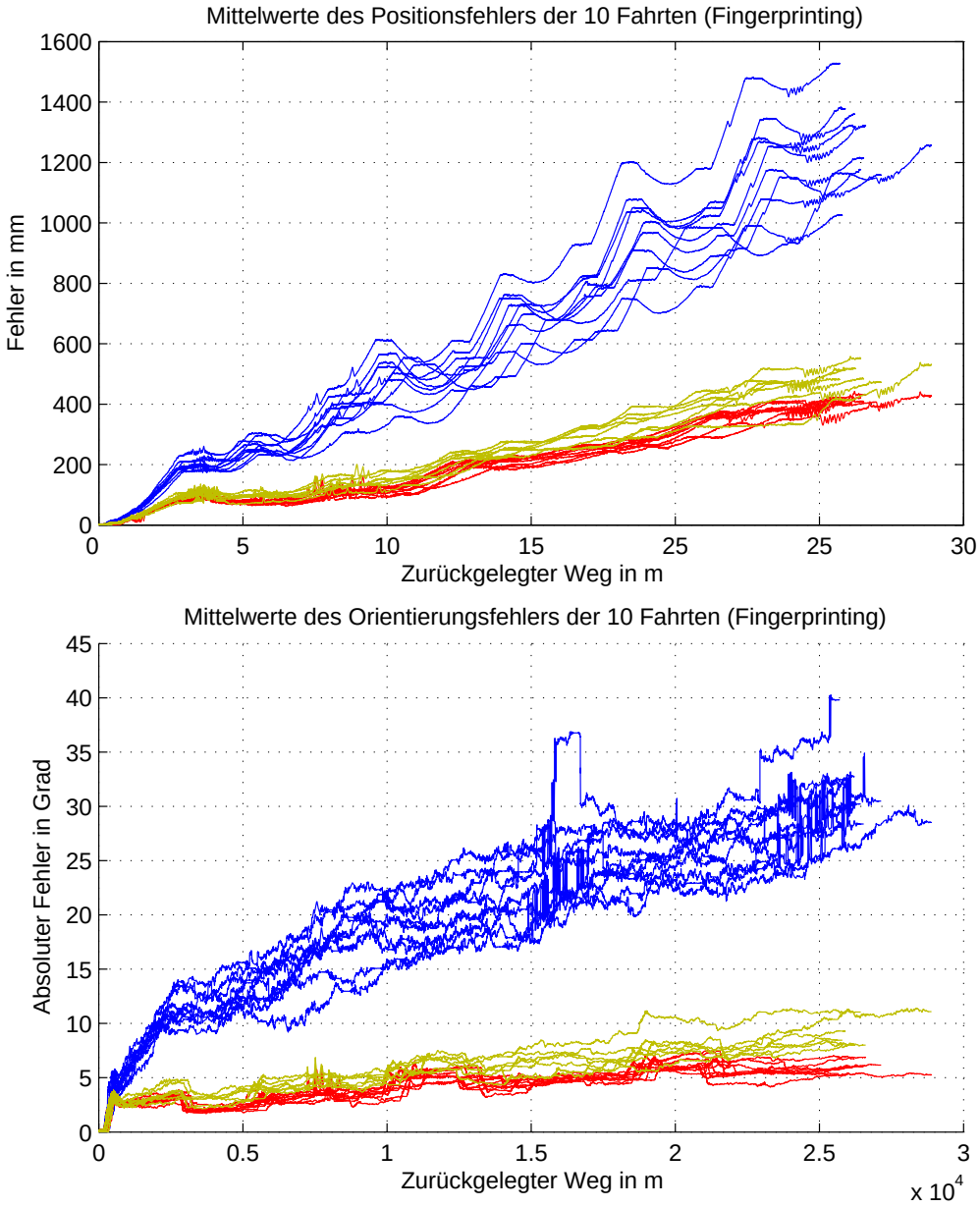

Abbildung 4.10: Mittelwert des Positionierungsfehler der 10 Fahrten. Farblich gekennzeichnet die Ergebnisse der verschiedenen Partikelauswahlalgorithmen. Blau: Bester Partikel, Rot: Schwerpunkt, Orange: Schwerpunkt im Radius um den besten Partikel.

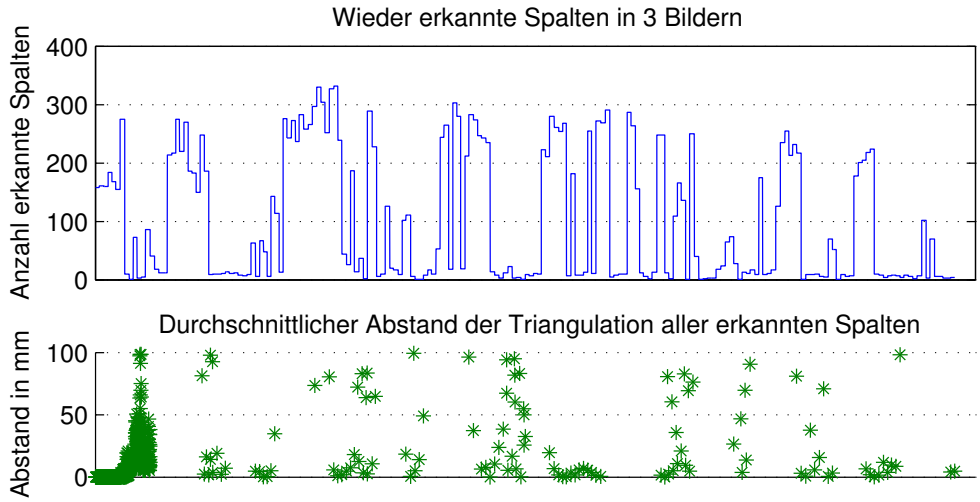

Abbildung 4.11: Oben die Anzahl der wieder erkannten Spalten in drei Bildern in einer Runde. Unten der berechnete Fehler zwischen der durch Triangulation berechneten Position und dem zu schätzenden Partikel. Die Grafik ist bei 100 mm abgeschnitten, da dieses das Maximum ist, das noch zur Schätzung genutzt wird. Das Maximum des Abstands in der Berechnung liegt bei ca. 50000 mm (5 m). Deshalb sind einige Werte nicht abgebildet.

Zu Beginn dieses Abschnitts wurde behauptet, dass dieses Verfahren einen großen Fehler produziert, wenn nur eine kleine Bewegung zwischen zwei Bildern statt gefunden hat. Der Verlauf des Orientierungsfehlers in Abbildung 4.12 zeigt ein ähnliches Rechteckmuster im Verlauf wie der Orientierungsfehler beim RED-Verfahren (siehe Seite 50). Die signifikanten Stellen des Anstiegs bzw. Abfalls eines Rechtecks liegen genau wie beim RED-Verfahren an den Stellen, wo der Roboter eine Drehung ausgeführt hat. Das lässt den Schluss zu, dass dieses Verfahren tatsächlich die besten Schätzungen liefert, wenn sich die Bilder deutlicher von einander unterscheiden. Da dieses bei hochauflösenderen Bildern auch für kleinere Positionsänderungen möglich ist, liegt die Vermutung nahe, dass die Qualität dieses Tracking-Verfahrens mit der Auflösung der Kamerabilder steigt.

4.6 Ergebnisse

In den vorherigen Abschnitten dieses Kapitels ist deutlich geworden, dass das gewichtete Mittel für die *Pose*-Schätzung des Partikelfilters den kleinsten Fehler produziert. Der beste Partikel ist mit Abstand das schlechteste Auswahlverfahren. Seine *Pose* ist offensichtlich nicht repräsentativ für den Partikelschwarm; jedenfalls nicht für unsichere Quellen. Bei der Berechnung der Konfiguration mit der Referenzlokalisierung als Quelle für den Partikelfilter zeigt sich, dass der beste Partikel hier nahe den anderen Auswahlverfahren liegt (vgl. Tabelle 4.1). Man könnte das Ergebnis der *Pose*-Schätzung durch die Wahl des besten Partikels als Maß für die Sicherheit einer Partikelfilterquelle verstehen. Unter dem Begrifft *Sicherheit* ist zu verstehen,

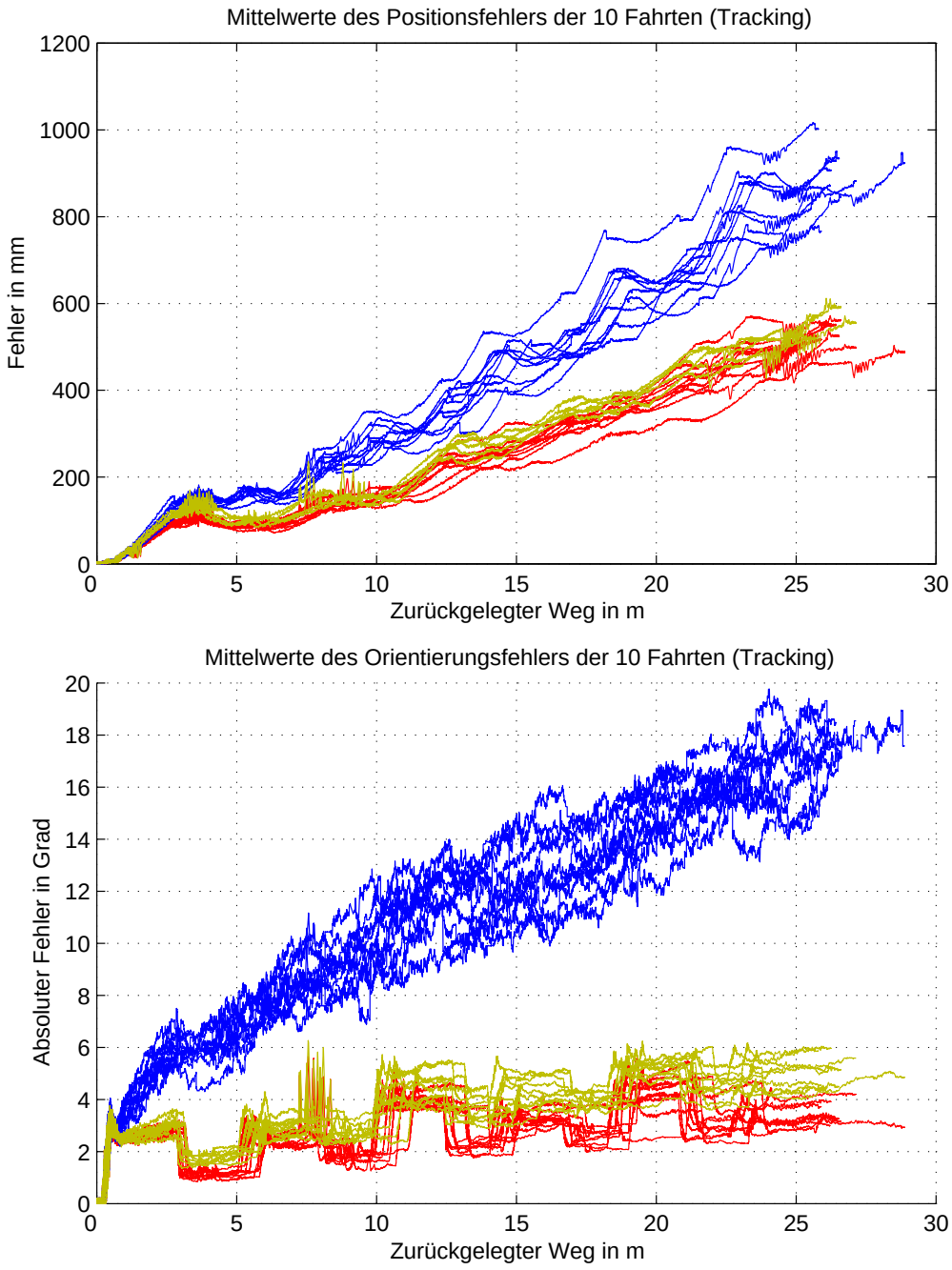

Abbildung 4.12: Mittelwert des Positionierungsfehlers der 10 Fahrten. Farblich gekennzeichnet die Ergebnisse der verschiedenen Partikelauswahlalgorithmen. Blau: Bester Partikel, Rot: Schwerpunkt, Orange: Schwerpunkt im Radius um den besten Partikel.

wie robust und genau eine Quelle die aktuelle *Pose* bestimmen kann.

Unterstützt wird diese Behauptung durch Lokalisierungsergebnisse mit verschiedenen Standardabweichungen für die Normalverteilung, die zum Verrauschen der *Pose* in der Vorhersage-Phase dient. Wird die Standardabweichung dieser Normalverteilung verändert, zeigt sich, dass sich die *Pose*-Schätzungen durch das gewichtete oder robuste Mittel kaum beeinflussen lassen. Lediglich der beste Partikel zeigt entweder einen größeren Fehler bei größerer Standardabweichung oder einen kleineren Fehler bei kleinerer Standardabweichung (siehe Abbildung 4.13). Dies gilt jedoch nicht für den Fehler, wenn die Referenzlokalisierung als Quelle für den Partikelfilter genutzt wird. Dann veränderte sich der Fehler für den besten Parikel ebenso unmerklich wie für die anderen beiden Auswahlverfahren. Einzige Ausnahme bildet eine zu klein gewählte Standardabweichung. Nach einem zu großen Fehler in der Referenzlokalisierung (vgl. auch Abbildung 4.1) werden die *Poses* der Partikel in der Vorhersage-Phase des Partikelfilters nicht stark genug verrauscht, um sich der tatsächlichen *Pose* wieder zu nähern. Die Tatsache, dass die Referenzlokalisierung die sicherste aller Quellen ist und der Verlauf in Abbildung 4.13 zeigen, dass der Fehler der „bester Partikel"-Auswahlstrategie als ein Maß der Sicherheit einer Partikelfilterquelle begriffen werden kann.

Das robuste Mittel liegt immer nahe dem Fehler des gewichteten Mittels. Das liegt daran, dass der Grenzwert für den Radius um den besten Partikel im Verhältnis zum Grenzwert für Partikel, die gelöscht werden sollen, so gewählt ist, dass eine große Menge der Partikelwolke zur Schwerpunktsberechnung mit einbezogen wird. Das robuste Mittel ist für Umgebungen gedacht, an denen der Roboter an engen Stellen an zwei Seiten um ein Hindernis herumfahren kann. Die Partikelwolke spaltet sich dann auf, so dass auf jeder Seite des Hindernisses ein Teil der Wolke liegt. Das gewichtete Mittel würde in dem Fall genau zwischen den beiden Wolken, also im Hindernis, liegen. Das robuste Mittel würde für jede Wolke den besten Partikel bestimmen und in einem Radius herum den Schwerpunkt bestimmen. Dieser Fall trat in den untersuchten Testfahrten nicht auf und es gab keine Stelle der Fahrt, an der das robuste Mittel im Vorteil gewesen wäre.

Für den folgenden Vergleich der verschiedenen Partikelfilterkonfigurationen wird nur das gewichtete Mittel untersucht. Und um das Ganze etwas übersichtlicher zu gestalten, wird von jeder Konfiguration nur die Berechnung der fünften aufgezeichneten Fahrt analysiert. In den vorherigen Analysen konnte man sehen, dass die Resultate der verschiedenen Fahrtaufzeichnungen für das gewichtete Mittel nahe zusammenliegen, die Einschränkung nur eine der Aufzeichnungen zu untersuchen, ist also ausreichend.

Zusätzlich wurden alle zehn Fahrten noch einmal mit allen drei Partikelfilterquellen zusammen berechnet. Da die Mittelwerte der Positions- und Orientierungsfehler für die Berechnungen den Ergebnissen der anderen Berechnungen gleichen, wird das Ergebnis nicht ausführlich grafisch dargestellt, sondern im folgenden nur mit den anderen Ergebnissen verglichen.

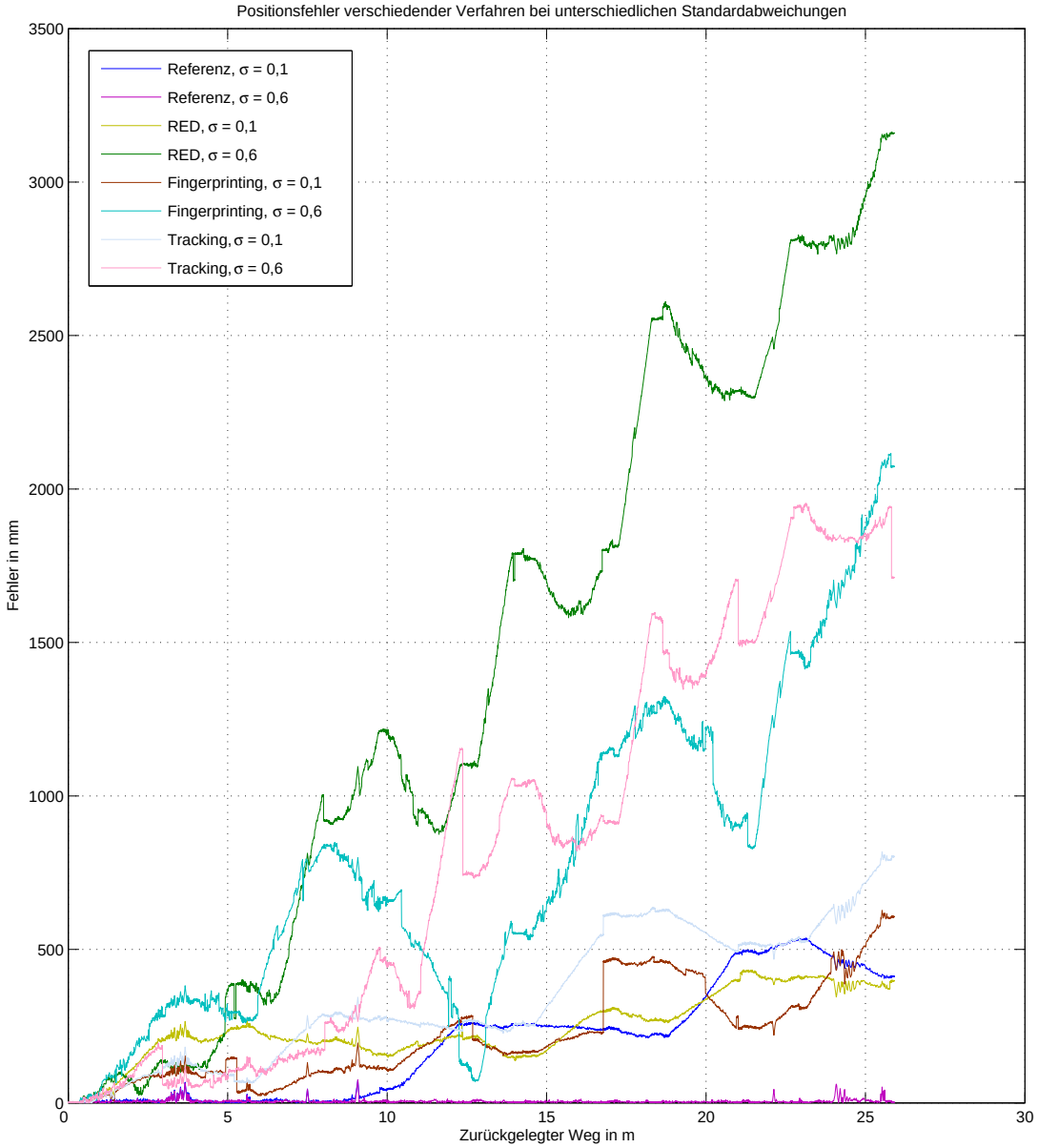

Abbildung 4.13: Entwicklung des Positionsfehlers für den besten Partikel bei unterschiedlicher Standardabweichung σ zum Verrauschen der *Pose*-Vorhersage. Bei den optischen Verfahren wird der Fehler mit größerem σ größer. Für die Berechnung mit der Referenzlokalisierung wird der Fehler bei größerem σ sogar kleiner. Eine Standardabweichung von 0,1 ist demnach zu klein, um die Partikel nach einem Fehler in der Referenzlokalisierung so vorherzusagen, dass sie sich wieder der korrekten *Pose* nähern.

4.6.1 Vergleich der verschiedenen Partikelfilterquellen

Nachdem in den vorherigen Abschnitten dieses Kapitels jede Partikelquelle einzeln analysiert wurde, werden sie jetzt miteinander verglichen. In Abbildung 4.14 sind jeweils die Positions- und Orientierungsfehler der Berechnung der fünften Fahrt für die einzelnen Partikelfilterkonfigurationen aufgezeichnet.

Offensichtlich waren alle Verfahren in der Lage, den Fehler des Partikelfilters im Vergleich zur reinen Odometrie zu verbessern. Dennoch ist die Steigung des Fehlers immer noch zu steil, um für die Praxis brauchbare Lokalisierung zu liefern. Wie Eingangs beschrieben ist für die meisten Aufgaben eines mobilen Robotersystems eine sehr genaue Positionsbestimmung notwendig. Interessant ist, dass alle Kurven einen ähnlichen Verlauf haben. Anscheinend ist keines der optischen Verfahren in der Lage, robust global zu lokalisieren. Sie können lediglich den Fehler der Odometrie verbessern, wodurch der grundsätzliche Verlauf der Fehlerkurven erhalten bleibt.

Der RED-Algorithmus verbessert den Fehler für die Orientierung trotz seiner einfachen Mittel am besten. Wenn man bedenkt, dass dieses Verfahren eigentlich nur an zehn Stellen der Fahrt eine Schätzung der Partikel liefert, ist auch die Wirkung auf den Positionsfehler bemerkenswert. Allerdings ist nicht zu erwarten, dass dieses Verfahren durch Optimierung der Parameter wesentlich bessere Ergebnisse liefert. Denn selbst wenn die Genauigkeit der Rotationserkennung von $3,0566° \pm 9,2071°$ verbessert würde, wird das nur wenig Ausschlag auf den Positionsfehler haben. Der Verlauf der Kurve zeigt, dass die Fehler auf geraden Strecken dominanter sind.

In den aufgenommenen Fahrten steht nur alle 600 ms ein neues Bild zur Verfügung. Das heißt, der Roboter legt bei voller Fahrt ($\frac{1}{2}\frac{m}{s}$) zwischen zwei Bildern ungefähr 30 cm zurück, bevor ein neues Bild zur Verfügung steht und die optischen Verfahren Partikel bewerten können. Bei dem RED- und Tracking-Verfahren liegen zum Teil noch größere Entfernungen zwischen zwei Bewertungsrunden.

Der RED-Algorithmus wird aber gut in der Lage sein, die Orientierung bei Erreichen einer gewünschten Position zu korrigieren. Stellen wir uns den Fall vor, dass der Roboter eine vorher berechnete *Pose* vor einem Regal anfahren und daraus ein Buch greifen soll. Bei Erreichen der (x, y)-Koordinate findet vor dem Regaln nun noch die Drehung zur gewünschten Ausrichtung statt. Bei einer Genauigkeit von $3,0566° \pm 9,2071°$ bei der Orientierung und einer Entfernung des zu greifenden Objektes von einem Meter liegt der Fehler durchschnittlich bei ca. 5 cm. Dieser Fehler muss dann durch andere Sensoren wie z.B. eine Handkamera für die Greifoperationen korrigiert werden.

Das Fingerprinting zeigt, dass es funktionsbedingt durch die Rotationsunabhängigkeit zwar den Positionsfehler gut korrigiert, die Orientierung aber nur wenig beeinflusst. Abgesehen von der starken Fehlerkorrektur zu Beginn (bei ca. 3 m) ist die Steigung des weiteren Verlaufs annähernd dem der Odometriekurve gleich, während die anderen drei Kurven einen flacheren Verlauf zeigen.

Erstaunlich ist, dass der Fehler der Berechnungen für den Partikelfilter mit allen drei Quellen größer ist als der einzelner Konfigurationen. Das war eigentlich nicht zu erwarten, weil der Partikelfilter die Schätzungen der einzelnen Quellen integrieren sollte, um daraus eine bessere Schätzung abzuleiten. Dieser merkwürdige Effekt

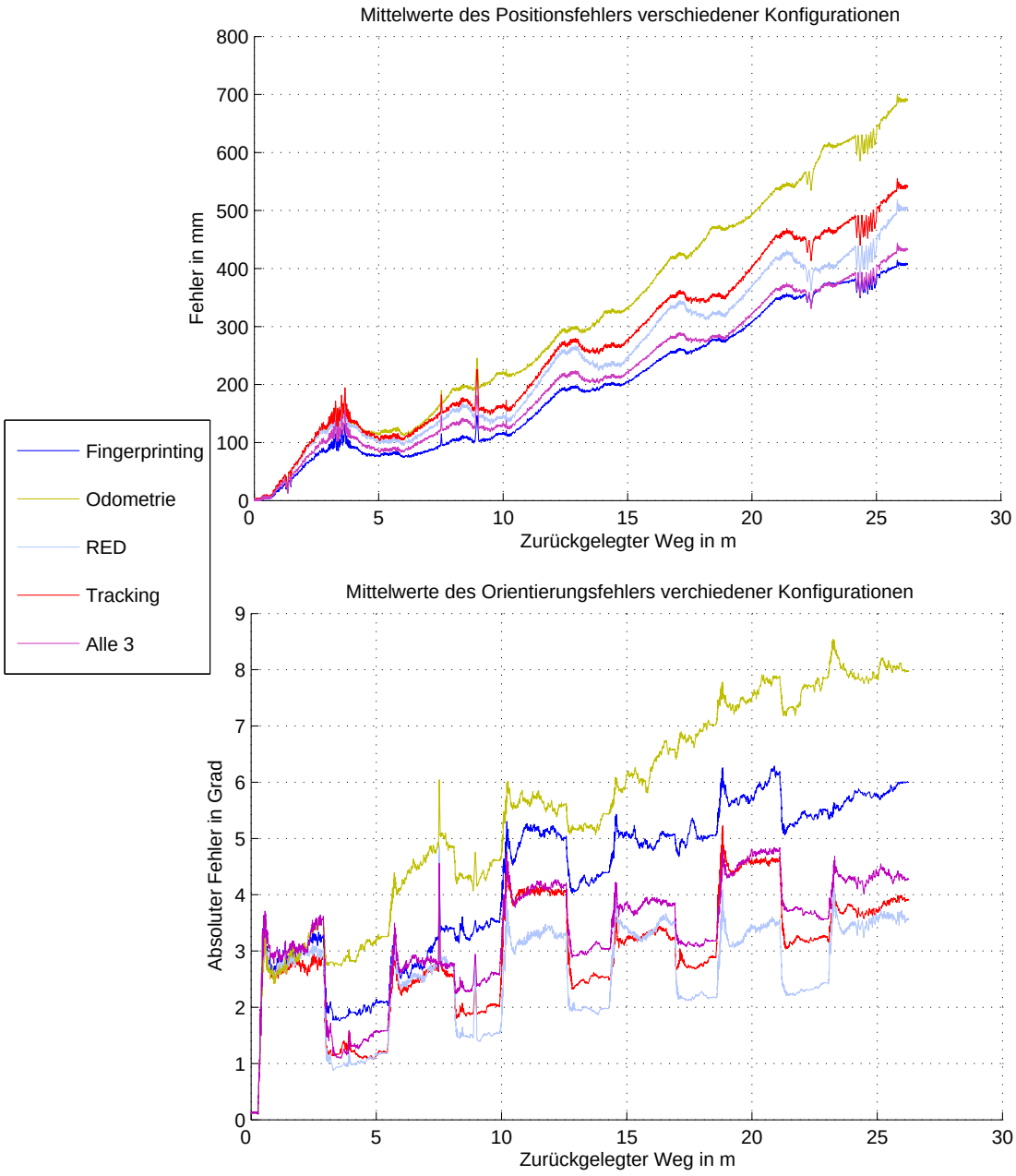

Abbildung 4.14: Mittelwert des Positionierungsfehlers der 10 Fahrten. Farblich gekennzeichnet die Ergebnisse der verschiedenen Partikelauswahlalgorithmen. Blau: Fingerprinting, Rot: Tracking, Orange: Nur Odometrie, Grün: Laserlokalisation, Hellblau: vertikale Kantenerkennung.

folgt aus der Tatsache, dass alle Partikelfilterquellen eine Bewertung eines Partikels abgeben sollen, die im Intervall $[0, 1]$ liegt (siehe Seite 29). Die RED- und Trackingverfahren berechnen ihre Bewertungen so, dass für sehr gute *Poses* Schätzungen nahe Eins gegeben werden. Dies gilt für die Werte an den Ausgängen des RBF-Netzes des Fingerprinting-Verfahrens leider nicht. Durch seine innere Struktur ist zwar garantiert, dass die Werte an seinen Ausgängen im Intervall $[0, 1]$ liegen, aber große Werte, die nahe bei Eins liegen werden im Vergleich zu den beiden anderen Verfahren selten geschätzt. Da die Bewertungen der Quellen durch den Partikelfilter lediglich miteinander multipliziert werden, fällt die Bewertung des Fingerprinting-Verfahrens im Vergleich zu den Bewertungen der anderen beiden Verfahren schwächer ins Gewicht. Das ist in diesem Fall besonders tragisch, da das Fingerprinting-Verfahren die beste Fehlerkorrektur leisten konnte, seinen Bewertungen aber nicht genügend Bedeutung bei der Verschmelzung mit den Bewertungen der anderen Verfahren gegeben wird.

Um dem entgegen zu wirken, sollte eine Gewichtung der Schätzungen der Partikelfilterquellen statt finden, bevor sie zusammengefasst werden, um ihrer Genauigkeit Rechnung zu tragen. Für den Orientierungsfehler ist das Ergebnis aller drei Verfahren sogar schlechter als das Ergebnis der RED- und Tracking-Verfahren alleine. Daraus lässt sich schliessen, dass für zukünftige und robuste Anwendungen der Verfahren für die Orientierungs- und die Positionsschätzung unterschiedliche Gewichtungen gefunden werden müssen, um das Ergebnis zu optimieren.

Das Tracking zeigt zwar im Gesamtvergleich bei der Positionsschätzung den größten Fehler. Durch sein gutes Ergebnis bei der Rotationsschätzung und wegen der Optimierungsmöglichkeiten, lässt sich jedoch vermuten, dass Tracking-Verfahren durchaus in der Lage sind, eine robuste Positionsschätzung zu liefern. Optimierungen ergeben sich durch robustere Techniken zur Merkmalserkennung in der Umwelt und dem Zuweisen der gleichen, in verschiedenen Bildern gefundenen, Merkmale. Das Tracking-Verfahren, das für diese Diplomarbeit eingesetzt wurde, hat im Verhältnis zu seiner einfachen Merkmalserkennung gute Ergebnisse geliefert. Fortgeschrittenere Techniken zur Merkmalserkennung wie zum Beispiel SIFT (vgl. Kapitel 1.2.4) sollten den Fehler noch einmal deutlich verkleinern können. Zumal durch den Einsatz des Partikelfilters auch mehrere unterschiedliche Tracking-Algorithmen parallel eingesetzt werden können.

4.6.2 Laufzeiten

Obwohl in dieser Arbeit die Verfahren verglichen wurden und nicht ihre Implementierung, werden nun einige Informationen über die beobachteten Berechnungslaufzeiten der verschiedenen Partikelfilterkonfigurationen gegeben.

Weil die Berechnungen mit aufgezeichneten Fahrten durchgeführt wurden, war es möglich, Bildtransformationen, die in jeder Berechnung notwendig waren, vorher durchzuführen, um die dafür benötigte Zeit bei den Berechnungen zu sparen. Die Bilder der aufgezeichneten Fahrten wurden aus diesem Grund bereits vor den Berechnungen in Panoramabilder verschiedener Größen umgerechnet, mit einem Median gefiltert und in Kantenbilder transformiert. Ebenso wurden die Merkmalsvek-

toren für das Fingerprinting-Verfahren (vgl. Kapitel 3.2) für die Bilder bereits vor den Berechnungen bestimmt.

Dadurch ist bei den Berechnungen der verschiedenen Partikelfilterkonfigurationen, in denen jeweils nur ein Verfahren als Partikelfilterquelle eingesetzt wird, nur ein kleiner Unterschied in den Laufzeiten zu bemerken. Daraus lässt sich schließen, dass Laufzeitunterschiede im wesentlichen aus dem Aufwand der Vorverarbeitung der Bilder resultieren. Erst bei der Verwendung aller drei Verfahren als Quellen für den Partikelfilter ließ sich eine deutliche Verlängerung der Laufzeiten wahrnehmen, die ungefähr bei 300% lag. D.h. für Quellen ähnlicher Komplexität ändert sich die Laufzeit des Partikelfilters linear zur Anzahl der eingesetzten Quellen.

Für das RED-Verfahren wurden keine Vorverarbeitungsschritte berechnet. Mit der Implementierung in dieser Arbeit und der Suche nach Spalten in den letzten drei Bildern beträgt die Laufzeit des Verfahrens für die Berechnung einer Fahrt ungefähr eine Minute. Das Fingerprinting-Verfahren hat zwar bereits die Signaturen der Bilder der Fahrten vorberechnet, muss aber für jede Schätzung das RBF-Netz abfragen. Die Berechnung einer Fahrt dauert so ca. 50 Sekunden. Das RED-Verfahren hat mit ca. 40 Sekunden die kürzeste Laufzeit.

Die Ausführung einer Fahrt durch das mobile Robotersystem dauerte knapp unter vier Minuten. Für die Implementierung dieser Arbeit könnte demnach eine Live-Fahrt ohne Unterbrechungen berechnet werden, vorausgesetzt die Vorverarbeitungsschritte können in vernachlässigbarer Zeit durchgeführt werden. Entweder indem spezielle Hardware diese Vorverarbeitung übernimmt, oder die eingesetzte Hardware ohne Spezialisierung schnell genug ist. Eine solche Hardware stand aber für diese Diplomarbeit nicht zur Verfügung, so dass eine Berechnung der Lokalisierung inklusive Vorverarbeitung der Bilder deutlich länger dauert, als die Fahrtausführung durch den Roboter.

Für die Analyse der Verfahren in dieser Arbeit wurden jeweils 500 Berechnungen pro Partikelauswahlstrategie durchgeführt (vgl. Kapitel 4). Für die Partikelfilterkonfigurationen mit jeweils einer Partikelfilterquelle gilt also, dass die 500 Berechnungen für jeweils eine der Strategien ca. sechs Stunden benötigten. Der komplette Durchlauf für alle Strategien braucht demnach ca. 18 Stunden, für die Partikelfilterkonfiguration mit allen drei Verfahren als Quellen verlängert sich die Berechnungszeit auf ungefähr 54 Stunden.

5 Zusammenfassung und Ausblick

Für diese Diplomarbeit wurden verschiedene optische Verfahren untersucht, die heutzutage üblicherweise zur Lokalisierung eines mobilen Robotersytems eingesetzt werden. Dabei wurde insbesondere darauf Wert gelegt, die Experimente unter möglichst realen Bedingungen durchzuführen. Falls sich Einschränkungen der Experimente nicht vermeiden liessen, wurde an entsprechender Stelle darauf hingewiesen. Diese Arbeit hatte nicht als Ziel, die Verfahren optimal zu implementieren, sondern vielmehr, sie miteinander zu vergleichen. Aus diesem Grund sind die Ergebnisse der Lokalisierungsexperimente dieser Arbeit nicht repräsentativ für die Genauigkeit optischer Lokalisierungsverfahren des aktuellen Forschungsstands. An vielen Stellen lassen sich die Implementierungen dieser Arbeit noch optimieren. Dennoch lassen sich Aussagen über die Fähigkeiten der angewandten Verfahren treffen.

Die Ergebnisse zeigen, dass die Verfahren zwar in der Lage sind, die *Pose*-Veränderung des Roboters im Vergleich zur Odometrie zu verbessern, aber keines der Verfahren konnte den Fehler im Verlauf der Fahrt begrenzen. Aufgrund der Auswertungen muss davon ausgegangen werden, dass der Fehler mit Länge der Fahrt immer größer wird. Einige Algorithmen haben jedoch das Potential gezeigt, bei einer genügend langen Fahrt eine Fehlerkonvergenz leisten zu können. Insbesondere das Tracking (Seite 37) und das Fingerprinting (Seite 34). Es bleibt aber die Frage, ob der Fehler, gegen den die Lokalisierung dann konvergiert, für einen praktischen Einsatz klein genug ist. Vor allen Dingen im Servicebereich muss die Bestimmung der aktuellen *Pose* eine hohe Genauigkeit haben, darf aber auch nicht zu viel Rechenzeit in Anspruch nehmen.

Beim Fingerprinting hängt der Erfolg stark vom Training des eingesetzten neuronalen Netzes und der Struktur der Umwelt ab. Die Gewinnng der Trainingsdaten, insbesondere zu unterschiedlichen Lichtverhältnissen und Raumkonfigurationen (Türen offen, geschlossen, Stühle verschoben, Personen im Raum, ...), muss sehr aufwendig gestaltet werden, um Robustheit zu erlangen. Für den realen Einsatz zur Lokalisierung eines Serviceroboters spielt der Aufwand der Trainingsphase keine besondere Rolle, da diese nur einmal durchgeführt werden muss. Die Komplexität der anschliessenden Lokalisierung ist lediglich abhängig von der Anzahl der Referenzpunkte.

Das Tracking wird dann besser, wenn mehr Positionen in der Vergangenheit berücksichtigt werden, die Erkennung der Bildeigenschaften eindeutiger und die Auflösung der Kamerabilder höher wird. Dann wird der Rechenaufwand im Gegenzug schnell so gross, dass er nicht einmal mehr annähernd in Echtzeit - d.h. in einem vorgegebenen Zeitintervall von oft nur wenigen Millisekunden- berechnet werden kann, so dass ein realer Einsatz zum Lokalisieren eines mobilen Roboters unwahrscheinlich

wird. Das spricht aber nur für die schlechte Qualität des für diese Arbeit eingesetzten Tracking-Verfahrens. Es gibt andere Tracking-Verfahren, die sich besser zur optischen Lokalisierung eignen [1].

Wie bereits in Kapitel 4.6.1 (vgl. Seite 60) erläutert, ist vom RED-Verfahren, wie es für diese Arbeit implementiert wurde, keine großartige Verbesserung mehr zu erwarten. Die Ergebnisse des Verfahrens, trotz seiner einfachen Methoden, die horizontale Verschiebung zweier omnidirektionaler Bilder auf wenige Grad zu bestimmen, zeigen, dass der Ansatz gut gewählt ist. Man könnte mehrere Verfahren verwenden, die per horizontaler Verschiebung wieder erkannter Merkmale eine Orientierungsänderung ableiten, z.B. mit Symmetriepunkten [27].

Angenommenm, es wäre eine optische Lokalisierung erfolgreich in Echtzeit durchzuführen und der Fehler ginge gegen einen Grenzwert, dann ist aufgrund der Testergebnisse zu erwarten, dass der Fehler deutlich über dem der derzeit eingesetzten Lokalisierung mit Lasersensoren liegt. Allerdings hat sich in der Informatik immer wieder gezeigt, dass einst unmögliche Verfahren durch den Fortschritt der Computertechnik (Moor'sches Gesetz) einsetzbar wurden. Techniken wie der Kalman- oder Partikelfilter wären vor zwanzig Jahren undenkbar gewesen. Heute finden sie vielerorts Anwendung.

Zusammenfassend läßt sich sagen, dass es mittelfristig unwahrscheinlich ist, dass Lokalisierungen auf Basis von Lasersensoren im Servicebereich außerhalb der Forschung großflächig durch rein optische Lokalisierungen ersetzt werden. Es muss trotzdem weiter in diese Richtung geforscht werden, weil optische Sensoren auf lange Sicht Lasersensoren überlegen sein werden. Sie sammeln mehr Informationen über ihre Umwelt, die auch für andere Verfahren, die ein mobiles System zur Bewältigung seiner Aufgaben benötigt, genutzt werden. Falls ein Serviceroboter Lasersensoren nur noch zur Lokalisierung besitzt, seine übrigen Aufgaben aber mittels optischer Sensoren bewältigt, wäre es effizienter, robuste optische Lokalisierungsverfahren zu besitzen, die den Einsatz der Laserscanner überflüssig machen. Zudem ist der Verbrauchermarkt für optische Technik größer als der für Lasersensoren. Es ist deswegen zu erwarten, dass die Herstellungskosten für optische Sensoren stärker fallen werden als für Laserscanner.

Für die Übergangszeit scheinen Verfahren viel versprechend zu sein, die Daten von Laser- und optischen Sensoren vereinen. Sie generieren eine dreidimensionale Karte ihrer Umgebung, mit Tiefen- und Farbinformationen. Die gesammelten Informationen können nicht nur für die Lokalisierung, sondern auch für Algorithmen genutzt werden, die die eigentlichen Aufgaben des Roboters ausführen.

Ein Vorteil des Partikelfilters im Vergleich zum Kalman-Filter ist es, die Wahrscheinlichkeitsverteilung der aktuellen *Pose* des Roboters in mehrere Bereiche zu zerteilen. Dieser sogenannte multimodale Fall wurde in dieser Arbeit nicht berücksichtigt. Im Falle einer solchen Zerteilung wird die Schätzung über die Zeit, wenn mehr Sensordaten der Umwelt zur Verfügung stehen, verfeinert, bis eine *Pose* sicher bestimmt werden kann. Das ist in Fällen hilfreich, wenn der Roboter ohne anfängliche Positionsschätzung initialisiert wird oder aufgrund unzureichender Sensorwerte

[1] Vgl. SIFT, Seite 12

oder eines *Kidnaps* die aktuelle *Pose* nicht mehr eindeutig bestimmt werden kann. In einem solchen Fall können bestimmte Routinen zur vorsichtigen Erkundung der Umwelt ausgeführt werden, um wieder eine eindeutige Schätzung für die aktuelle Position herzuleiten. Eine solche Positionsbestimmung mit optischen Sensoren und einem Partikelfilter wird zum Beispiel in [4] und [5] durchgeführt.

Um direkt an diese Diplomarbeit anzuknüpfen, wäre es interessant, Quellen für den Partikelfilter einzusetzen, die mit künstlichen Landmarken arbeiten. Die eingesetzte Referenzlokalisierung auf der mobilen Serviceplattform arbeitet ebenfalls mit künstlichen Landmarken (Reflektoren). Der Roboter könnte sich zwar nicht mehr innerhalb einer unbekannten Umgebung lokalisieren, aber das gilt ebenso für den Fall, dass Fingerprinting-Verfahren eingesetzt werden. Künstliche Landmarken in der Umwelt aufzustellen, könnte im Aufwand sogar unter der Trainingsphase von Fingerprinting-Verfahren liegen. Durch den Einsatz von künstlichen Landmarken kann eine Konvergenz des Fehlers bei der Lokalisierung herbei geführt werden. Der Aufwand wäre hierbei geringer als bei Verfahren, die nur mit natürlichen Landmarken arbeiten.

Hinweise auf übernommene Hilfsmittel

Die Implementierung der Programme für diese Diplomarbeit wurde in der Programmiersprache Java™ durchgeführt. Für die Kontrolle und Steuerung der mobilen Serviceplattform wurden am Arbeitsbereich TAMS zur Verfügung stehende Programme auf der Basis des genRob®-Rahmenwerkes [2] genutzt.

Die Laserlokalisierung, die als Referenzlokalisierung genutzt wurde, haben Axel Schneider, Daniel Westoff [28] und Thorsten Scherer [21] implementiert und als Java™-Bibliothek zur Verfügung gestellt.

Martin Weser hat den Auto-Korrelations Algorithmus aus [9] implementiert. Dieser wurde für die Fingerprinting-Quelle benutzt (Seite 34).

[2] *genRob* und *Roblet* sind eingetragene Warenzeichen von Hagen Stanek, Deutschland, http://www.genrob.com

Literaturverzeichnis

[1] Lamon, Pierre ; Nourbakhsh, Illah ; Jensen, Björn ; Siegwart, Roland: Deriving and matching image fingerprint sequences for mobile robot localization. In: *International Conference on Robotics & Automation*. Seoul, Korea, 2001

[2] Lamon, Pierre ; Tapus, Adriana ; Glauser, Etienne ; Tomatis, Nicola ; Siegwart, Roland: Environmental Modeling with Fingerprint Sequences fpr Topological Global Localization. In: *International Conference on Intelligent Robots and Systems*. Las vegas, Nevada, U.S.A., 2003

[3] Paletta, Lucas ; Frintrop, Simone ; Hertzberg, Joachim: Robust Localization Using Context in Omnidirectional Imaging. In: *International Conference on Robotics & Automation*. Seoul, Korea, 2001

[4] Gross, Horst-Michael ; Koenig, Alexander ; Schroeter, Christof ; Boehme, Hans-Joachim: Omnivision-based Probabilistic Self-localization for a Mobile Shopping Assistant Continued. In: *International Conference on Robotics & Automation*. Las Vegas, Nevada, U.S.A., 2002

[5] Gross, Horst-Michael ; Koenig, Alexander ; Boehme, Hans-Joachim ; Schroeter, Christof: Vision-based Monte Carlo Self-localization for a Mobile Service Robot Acting as Shopping Assistant in a Home Store. In: *International Conference on Robotics & Automation*. Lausanne, Schweiz, 2002

[6] Artač, Matej ; Jogan, Matjaž ; Leonardis, Aleš: Mobile Robot Localization Using Incremental Eigenspace Model. In: *International Conference on Robotics & Automation*. Washington, DC, U.S.A., 2002

[7] Tang, Lixin ; Yuta, Shin'ichi: Indoor Navigation for Mobile Robots Using Memorized Omni-directional Images and Robot's Motion. In: *International Conference on Intelligent Robots and Systems*. Lausanne, Schweiz, 2002

[8] Yuen, David C. ; MacDonald, Bruce A.: Natural landmark based localisation system using panoramic images. In: *International Conference on Robotics & Automation*. Washington, DC, U.S.A., 2002

[9] Linåker, Fredrik ; Ishikawa, Masumi: Rotation Invariant Freatures from Omnidirectional Camera Images using a Polar Higher-oder Local Autocorrelation Feature Extractor. In: *International Conference on Intelligent Robots and Systems*. Sendai, Japan, 2004

[10] Clerentin, Arnaud ; Delahoche, Laurent ; Brassard, Eric ; Cauchois, Cyril: Moble robot localization based on multi target tracking. In: *Proceedings of the IEEE International Conference on Robotics & Automation.* Washington, DC, U.S.A., 2002

[11] Jeong, In S. ; Cho, Hyung S.: Self-localization for Mobile Robota by Matching of Two Consecutive Environmenal Range Data. In: *International Conference on Robotics & Automation.* Seoul, Korea, 2001

[12] Miura, Jun ; Negishi, Yoshiro ; Shirai, Yoshiaki: Mobile Robot Map Generation by Integrating Omnidirectional Stereo and Laser Range Finder. In: *International Conference on Intelligent Robots and Systems.* Lausanne, Schweiz, 2002

[13] Schulenburg, Erik ; Weigel, Thilo ; Kleiner, Alexander: Self-localization in Dynamic Environments based on Laser and Vision Data. In: *International Conference on Intelligent Robots and Systems.* Las Vegas, Nevada, U.S.A., 2003

[14] Se, Stephen ; Lowe, David ; Little, Jim: Vision-based Mobile Robot Localization And mapping using Scale-Invariant Features. In: *International Conference on Robotics & Automation.* Seoul, Korea, 2001

[15] Dao, Nguyen X. ; You, Bum-Jae ; Oh, Sang-Rok ; Hwangbo, Myung: Visual Self-Localization for Indoor Mobile Robots Using Natural lines. In: *International Conference on Intelligent Robots and Systems.* Las Vegas, Nevada, U.S.A., 2003

[16] Jang, Gijeong ; Kim, Sungho ; Lee, Wangheon ; Kweon, Inso: Colo Landmark Based Self-localization for Indoor Mobile Robots. In: *International Conference on Robotics & Automation.* Washington, DC, U.S.A., 2002

[17] Andreasson, Henrik ; Treptow, Andre ; Ducket, Tom: Localization for Mobile Robots using Panoramic Vision, Local Features and Particle Filter / Örebro University, Dept. of Technology University of Tübingen, Dept. of Computer Science WSI-RA. Örbero, Sweden; Tübingen, Deutschland, 2005. – Forschungsbericht

[18] Lowe, David G.: Object Recognition from Local Scale-Invariant Features. In: *Proc. of the International Conference on Computer Vision ICCV, Corfu.* Corfu, Greece, 1999

[19] Benosman, Ryad (Hrsg.) ; Kang, Sing B. (Hrsg.): *Panoramic Vision.* New York, U.S.A. : Springer, 2001

[20] Nayar, Shree K.: Omnidirectional Vision. In: *The Eighth International Symposium of Robotics Research.* Hayama, Japan, 1997

[21] Scherer, Torsten: *A Mobile Service Robot for Automisation of Sample Taking and Sample Management in a Biotechnological Pilot Laboratory.* Bielefeld

/ München, Deutschland, Dept. of Cell Culture Technology, University of Bielefeld Dept. of Robotics and Embedded Systems, Technical University of Munich, Diss., 2004

[22] Rekleitis, Ioannis M.: A Particle Filter Tutorial for Mobile Robot Localization / Centre for Intelligent Machines, McGill University, 3480 University St., Montreal, Québec, Canada. Québec, Kanada, 2003. – Forschungsbericht

[23] Jähne, Bernd: *Digitale Bildverarbeitung.* Berlin, Deutschland : Springer, 2002

[24] Yamazawa, Kazumase ; Yagi, Yasushi ; Yachida, Masahiko: Omnidirectional Imaging with Hyperboloidal Projection. In: *Proceedings of the IEEE/RSJ International Conference in Intelligent Robots and Systems.* Yokohama, Japan, 1993

[25] Poettering, Lennart: *Ein Selbstlokalisierungsalgorithmus für mobile Roboter mit Bewegtbild-Panoramakameras.* 2005. – Universität Hamburg, Fachbereich Informatik, Arbeitsgruppe TAMS. Projektarbeit Wintersemester 2003/2004

[26] Borenstein, J. ; Everett, H. R. ; L., Feng: Where am I? Sensors and Methods for Mobile Robot Positioning / University of Michigan. Michigan, U.S.A., 1996. – Forschungsbericht

[27] Westhoff, Daniel ; Hübner, Kai ; Zhang, Jianwei: Robust Illumination-Invariant Features by Quantitative Bilateral Symmetry Detection. In: *Proceedings of the IEEE International Conference on Information Acquisition.* Warsaw, Poland, 2005

[28] Schneider, Axel ; Westhoff, Daniel: *Autonomous Navigation and Control of a Mobile Robot in a Cell Culture Laboratory.* Bielefeld, Deutschland, University of Bielefeld, Diplomarbeit, 2002