# Autonome Erkundung und Kartographierung unbekannter Innenräume durch einen Flugroboter

Bei der Fakultät für Elektrotechnik der
Helmut-Schmidt-Universität / Universität der Bundeswehr Hamburg
zur Erlangung des akademischen Grades eines Doktor-Ingenieurs
genehmigte

**Dissertation**
vorgelegt von

Mirco Alpen
aus Rendsburg

Hamburg, 2013

| | |
|---|---|
| Tag der mündlichen Prüfung: | 11.09.2103 |
| Erstgutachter: | Prof. Dr.-Ing. Joachim Horn |
| Zweitgutachter: | Prof. Dr.-Ing. Gerd Scholl |

# Vorwort

Die vorliegende Dissertation wurde in der Professur für Regelungstechnik der Helmut Schmidt Universität, Universität der Bundeswehr Hamburg angefertigt.

Für die Unterstützung, die ich bei der Erstellung dieser Arbeit erhalten habe, danke ich allen Beteiligten. Mein besonderer Dank gilt dabei meinem Doktorvater Herrn Professor Dr.-Ing. Joachim Horn sowie Herr Dr.-Ing. Klaus Frick und Herrn Dipl.-Ing Reinhard Lindloff, die mir stets mit Rat und Tat zur Seite standen. Die hervorragende Betreuung hat ebenso wie die konstruktive Zusammenarbeit maßgeblich zum erfolgreichen Abschluss dieser Arbeit beigetragen.

Des Weiteren gilt mein Dank Herrn Professor Dr.-Ing. Gerd Scholl für die Übernahme des Korefferats und Herrn Professor Dr.-Ing. habil. Detlef Schulz für die Übernahme des Vorsitzes der Prüfungskommision.

An dieser Stelle danke ich ebenfalls meiner Familie und insbesondere meiner Frau für die Unterstützung während meiner wissenschaftlichen Tätigkeit. Ohne sie wäre die Fertigstellung dieser Arbeit nicht möglich gewesen.

Hamburg, im September 2013

Mirco Alpen

# Inhaltsverzeichnis

**1 Einleitung** **1**

**2 Einführung in die SLAM-Algorithmen und die Navigation von Robotern** **5**

2.1 Grundprinzip bestehender SLAM-Algorithmen . . . . . 6

2.2 Scan-Matching-Verfahren . . . . . 12

2.3 Navigation von Robotern . . . . . 18

**3 Echtzeitfähiger SLAM-Algorithmus** **25**

3.1 Schätzung der Ausrichtung des Roboters . . . . . 26

3.2 Von den Rohdaten zur orthogonalen Linie . . . . . 32

3.3 Aktualisierung und Optimierung der globalen Karte . . . 41

3.4 Validierung durch Simulation . . . . . 48

**4 Experimentelle Ergebnisse** **51**

4.1 Beschreibung und Modellierung des Quadrotors . . . . . 51

4.1.1 Funktionsprinzip eines Quadrotors . . . . . 52

4.1.2 Die Flugplattform AR100B® . . . . . 56

4.1.3 Dynamisches Modell der Flugplattform . . . . . 58

4.2 Die verwendete Nutzlast ICAM . . . . . 63

4.3 Experiment zur Parameteranpassung . . . . . 67

4.4 Autonome Erkundung und Kartographierung . . . . . 71

**5 Zusammenfassung und Ausblick** **81**

**A Modellierung des Quadrotors** **85**

A.1 Modellansatz aus der Literatur . . . . . 85

A.2 Translation und Rotation bezogen auf die Gierachse . . . 88

A.3 Translation in der $x_{\mathrm{E}} / y_{\mathrm{E}}$ - Ebene . . . . . 99

**B Intelligente Navigationsstrategie** **107**

B.1 Lokale und globale Navigation . . . . . 107

B.2 Ermittlung von Zielpunkten . . . . . 113

B.3 Validierung durch Experiment . . . . . 115

**C Symbolverzeichnis** **121**

C.1 Verwendete Abkürzungen . . . . . 121

C.2 Lateinische Buchstaben . . . . . 121

C.3 Griechische Buchstaben . . . . . 122

C.4 Tiefgestellte Indizes . . . . . 123

C.5 Hochgestellte Indizes . . . . . 124

**Literatur** **125**

# 1 Einleitung

Roboter übernehmen sowohl zivil als auch militärisch immer komplexere Aufgaben. Es wird versucht, sie überall dort einzusetzen, wo es auf ein hohes Maß an Präzision, Geschwindigkeit und Zuverlässigkeit ankommt. Ein weiteres Einsatzgebiet sind Umgebungen, deren Betreten für den Menschen große Gefahren darstellt. Dies können Regionen mit unzumutbaren Umweltbedingungen, zerstörte Gebäude oder auch Minenfelder sein. In solchen Fällen können Roboter als intelligente Werkzeuge des Menschen wertvolle Dienste leisten.

Der Fokus dieser Arbeit liegt auf der Erkundung und Kartographierung unbekannter Innenräume durch einen Flugroboter, ein sogenanntes ***U****nmanned* ***A****erial* ***V****ehicle* (kurz UAV). Über die Anwendung solcher kleinen auch für das Innere von Gebäuden einsetzbaren Flugplattformen wurde in der jüngeren Vergangenheit öfter in verschiedenen Medien berichtet. So setzt die Feuerwehr diese Geräte beispielsweise zur Unterstützung bei der Koordination der Löscharbeiten bei Großbränden ein [28]. Im militärischen Bereich werden mit Hilfe von kleinen UAVs unter anderem Sprengfallen gesucht oder eine urbane Umgebung aufgeklärt [46]. In diesen Szenarien ist das Fluggerät jeweils mit einer Kamera ausgerüstet, deren Bilder dem Anwender zur Verfügung gestellt werden. Der Anwender muss es jedoch per Hand mittels Funkfernsteuerung navigieren. Entgegen mancher Darstellungen von Herstellern solcher Vehikel ist dies insbesondere in engen Umgebungen keine leichte Aufgabe. Besonders schwierig wird es dann, wenn der Anwender das Fluggerät nur aufgrund der Bilder der an Bord befindlichen Kamera steuern muss, da es sich nicht mehr in seinem Sichtfeld befindet.

Im Rahmen dieser Arbeit wird beschrieben, wie sich der Anwendungsbereich der urbanen Aufklärung auch auf das Innere von Gebäuden erweitern lässt. Eine solche Möglichkeit kann zu einem Zuwachs an

wichtigen Informationen und damit zu detaillierteren Einsatzplanungen beitragen. Damit sich ein Roboter in einer unbekannten Umgebung eigenständig bewegen und die gewonnenen Informationen in einer dynamisch zu erstellenden Karte speichern kann, müssen zwei Aufgaben zeitgleich erfüllt werden. Er muss sich in seiner Umgebung zum einen eindeutig lokalisieren und zum anderen die Karte erstellen. Die Lokalisierung erfolgt üblicherweise anhand einer Karte, die im Falle unbekannter Innenräume jedoch erst während der Mission entsteht. Die Kartographierung erfordert wiederum die Kenntnis der aktuellen Position. Die Lösung dieses Problems bezeichnet man als SLAM (engl. ***S****imultaneous* ***L****ocalization* ***A****nd* ***M****apping*). Kapitel 2 gibt einen Überblick über bereits bekannte Verfahren auf diesem Gebiet. Der Fokus liegt dabei auf Methoden, die ohne GPS (engl. ***G****lobal* ***P****ositioning* ***S****ystem*) auskommen, da die dazu erforderliche Satellitenverbindung innerhalb von Gebäuden nicht sichergestellt werden kann.
Aufgrund der oben genannten Schwierigkeiten bei der Bedienung ohne Sichtverbindung ist es wichtig, dass das Fluggerät über einen gewissen Grad an Autonomie verfügt, der die Steuerung durch den Anwender erleichtert. Der Roboter sollte in der Lage sein, sich innerhalb des Gebäudes eigenständig, zielorientiert und kollisionsfrei zu bewegen. In Abschnitt 2.3 sind verschiedene Bewegungsstrategien aufgeführt, mit denen ein solches Verhalten erreicht werden kann. Um unabhängig von der dauerhaften Verbindung zu einer Bodenstation zu sein, wird bei der späteren Realisierung darauf Wert gelegt, dass alle zur Navigation erforderlichen Berechnungen auf einem an Bord befindlichen Computer durchgeführt werden können.

Aus den in den Abschnitten 2.1 und 2.2 angegeben Quellen wird deutlich, dass man insbesondere bei Verwendung von Flugrobotern bei den für den SLAM-Algorithmus erforderlichen Berechnungen auf eine Bodenstation zurückgreift. Diese Arbeit zeichnet sich durch die Entwicklung und Implementierung eines Verfahrens aus, bei dem diese Berechnungen an Bord des Fluggerätes durchgeführt werden können. In Kapitel 3 wird der im Rahmen dieser Arbeit verwendete echtzeitfähige SLAM-Algorithmus vorgestellt. Dazu wird zunächst erläutert, wie

sich die Position des Roboters im Raum nur aufgrund von optischen Messdaten bestimmen lässt. Durch die Kenntnis der Ausrichtung der Roboterplattform lassen sich Informationen einer neuen Messung besonders elegant und effizient in die bereits bestehende Karte integrieren. Diese Eingliederung der aktuellen Messung erfolgt im Gegensatz zu den bekannten Verfahren in nur einem Iterationsschritt. Die Validierung durch eine Simulation ist ebenfalls Bestandteil dieses Kapitels.
Bei der Entwicklung wurde darauf geachtet, dass der resultierende Algorithmus auf einem einfachen handelsüblichen Microcontroller lauffähig ist. Als Datenquelle für den Algorithmus wurde ein Laserscanner gewählt. Dieser optische Sensor liefert eine gut zu verarbeitende und überschaubare Datenmenge. Darüberhinaus ist er, verglichen mit einer Kamera, deutlich unempfindlicher gegenüber der in der untersuchten Umgebung vorherrschenden Beleuchtung. Die verwendeten Hardwarekomponenten werden in Abschnitt 4.2 im Detail vorgestellt.

Für die autonome Erkundung und Kartographierung von unbekannten Innenräumen eignen sich Drehflügler besonders gut. Sie gehören zur Gruppe der Fluggeräte, die senkrecht starten und landen (engl. ***V****ertical* ***T****ake-****O****ff and* ***L****anding*, kurz VTOL) können und die Fähigkeit des Schwebeflugs (engl. *hovering*) besitzen [6]. Diese Eigenschaft erfüllen neben den allgemein bekannten Helikoptern auch Quadrotoren. Letztgenannte Geräte haben den Namen durch ihren Aufbau, der sich durch vier Rotoren in einer Ebene auszeichnet. Im Abschnitt 4.1 wird die allgemeine Funktionsweise von Quadrotoren erläutert und ihre Vorteile in Bezug auf die im Rahmen dieser Arbeit behandelte Anwendung herausgearbeitet. Die hier verwendete Flugplattform AR100B® der in Deutschland ansässigen Firma AirRobot® wird ebenfalls eingeführt. Für einen autonomen Flugbetrieb müssen verschiedene Regler realisiert werden. Da der hier verwendete industrielle Quadrotor bereits über eine Lageregelung verfügt, deren Struktur und Parameter nicht bekannt sind, kann bei der Reglerauslegung nicht auf die aus der Literatur bekannten Ansätze zurückgegriffen werden. Die Parameter des hier präsentierten nichtlinearen Modells wurden auf Basis von mehreren Messungen ermittelt.

Im Anschluss an die Betrachtungen zum verwendeten Quadrotor wird im Kapitel 4 die Durchführung verschiedener Experimente beschrieben. Nach einem vorbereitenden Versuch zur Ermittlung der geeigneten Wahl der verschiedenen Entscheidungsvariablen des SLAM-Algorithmus und der für den autonomen Flug erforderlichen Reglerparameter wird anhand von drei Szenarien die Leistungsfähigkeit des hier präsentierten Verfahrens zur eigenständigen Erkundung und Kartographierung von Innenräumen demonstriert. Die Ergebnisse zeigen, dass das im Rahmen der Arbeit gesteckte Ziel erreicht wird. Die Arbeit schließt mit einer Zusammenfassung und einem Ausblick in Kapitel 5.

# 2 Einführung in die SLAM-Algorithmen und die Navigation von Robotern

Damit ein Roboter unbekannte Innenräume erkunden und kartographieren kann, muss er einerseits über die Fähigkeit verfügen, seine Position in der ihm zunächst unbekannten Umgebung zu bestimmen und andererseits die durch Sensoren erlangten Umgebungsinformationen zu einer maßstabsgetreuen Karte verarbeiten. Man kann sich leicht vorstellen, dass eine Positionsbestimmung in der Regel mit Hilfe einer Karte erfolgt, und die Kartenerstellung wiederum eine genaue Kenntnis der Position voraussetzt. Damit ergibt sich in einer unbekannten Umgebung ein sogenanntes Henne-Ei-Problem, dessen Lösung die simultane Lokalisierung und Kartenerstellung ist, die in der Literatur [50, 52, 57] kurz als SLAM bezeichnet wird.

Die Lokalisierung und die Kartenerstellung sind zwei wesentliche Bestandteile, die zu einer eigenständigen Erkundung unbekannter Innenräume erforderlich sind. Ein weiteres Element ist die Navigationsstrategie, durch die der Roboter bestimmte Verhaltensregeln für seine Bewegungen bekommt. Bild 2.1 zeigt die genannten Bausteine und deren Kombination.
Im folgenden Abschnitt werden zunächst bekannte SLAM-Algorithmen vorgestellt und anhand ihrer Kartographierungsmethode klassifiziert. Da ein unbekannter Innenraum aufgrund von Abschattungen oder der begrenzten Reichweite des verwendeten Sensors im Allgemeinen nicht in vollem Umfang durch eine Messung erfasst werden kann, sind zum Erhalt einer vollständigen Karte Messungen an unterschiedlichen Positionen nötig. Um die einzelnen Messungen zu einer konsistenten Karte zusammenzuführen, ist jeweils ein Abgleich (engl. *Scan-Matching*) erforderlich. Dabei wird durch automatisierte Algorithmen nach Gemeinsamkeiten zwischen bereits bekannten Daten und der aktuellen Messung gesucht. Derartige Verfahren werden im zweiten Abschnitt vorgestellt. Die Kombination aus SLAM und einer Bewegungsstrategie ist in Bild 2.1

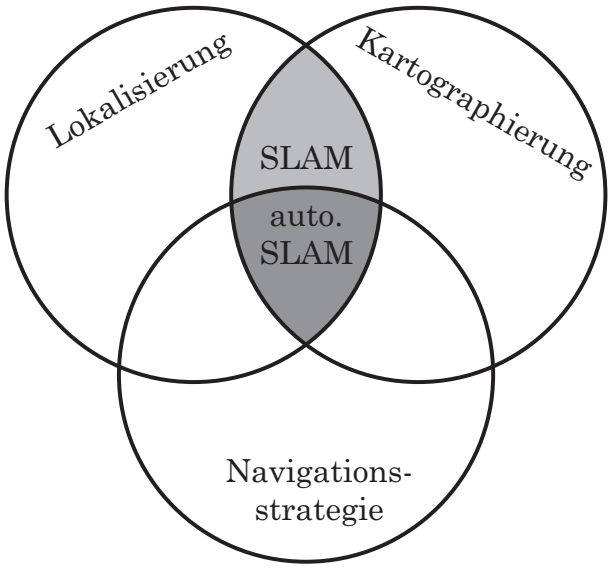



**Bild 2.1:** Entwicklung von SLAM aus Grundbausteinen nach [61]

mit „auto. SLAM“ bezeichnet. Entsprechende Algorithmen für die Navigation, die zu einer autonomen Erkundung und Kartographierung von unbekannten Innenräumen führen, werden im letzten Abschnitt dieses Kapitels beschrieben.

## 2.1 Grundprinzip bestehender SLAM-Algorithmen

An dem Thema SLAM wird seit fast zwei Jahrzehnten weltweit in vielen Instituten gearbeitet. Über die Jahre sind viele Lösungsansätze für dieses Problem publiziert worden. Einen guten Überblick über die bereits mit Bodenrobotern erzielten Ergebnisse zu Beginn dieser Arbeit im Jahr 2007 geben zwei Artikel aus 2006 [7, 22]. Die bekannten Verfahren lassen sich mathematisch in inkrementelle und vollständige SLAM-Verfahren unterteilen [34]. Bei inkrementellen Verfahren werden nach jeder Messung sowohl die Karte als auch die aktuelle Position ermittelt. Wird durch Messfehler einmal eine falsche Position geschätzt, pflanzt sich dieser Fehler im weiteren Verlauf des jeweiligen Algorithmus fort. Ein vollständiger SLAM-Algorithmus beinhaltet zusätzlich die Schätzung aller bisherigen Positionen des Roboters. Damit kann die eben genannte Fehlerfortpflanzung verhindert werden. Ein solcher Algorithmus ist beispielsweise der GraphSLAM [55], bei dem zur Lösung

des Problems dem Namen nach eine Graphentheorie verwendet wird. Voraussetzung für ein solches Verfahren ist jedoch, dass Vorkenntnisse über die Position bestimmter Objekte, sogenannter Landmarken, innerhalb des zu erkundenden Gebiets bekannt sind.
Da im Rahmen dieser Arbeit von unbekannten Innenräumen ausgegangen wird, werden vollständige SLAM-Algorithmen hier nur kurz betrachtet und der Fokus auf die inkrementellen Verfahren gelegt. In diesem Bereich sind die bisher publizierten Ansätze sehr vielfältig. Es existieren viele Namenskreationen, die hier nicht alle genannt werden können. Die bekanntesten werden in Abhängigkeit des jeweils verwendeten Kartentyps und mit Blick auf die im Rahmen dieser Arbeit verwendete Hardware im Folgenden kurz vorgestellt und bewertet.

*Ansätze mit topologischen Karten*

Topologische Karten basieren in diesem Zusammenhang auf Landmarken. Diese sind im Allgemeinen Objekte, deren Position und Form bekannt ist. Während der Erkundung hinzugewonnene Informationen werden in Abhängigkeit solcher Landmarken in Form eines Graphen mit Knoten und Kanten gespeichert. Damit lässt sich die zugehörige topologische Karte sehr effizient speichern.
Ein Verfahren, das diese Art der Kartographierung verwendet, ist der EKF-SLAM (engl. ***E**xtended **K**alman **F**ilter* - SLAM) [34]. Durch das erweiterte Kalman-Filter werden zu jedem Zeitschritt die Position und Ausrichtung des Roboters sowie die Position der zu erwartenden Landmarken geschätzt. Für diese Schätzung müssen Matrizen miteinander multipliziert werden. Dies macht das gesamte Verfahren sehr rechenaufwändig. Für die Anwendung auf dem im Rahmen dieser Arbeit verwendeten Microcontroller ist dieser Algorithmus daher nicht geeignet.
Ein weiterer Ansatz, der auf der Verwendung von topologischen Karten basiert, ist der FastSLAM, dessen Grundidee in [34] beschrieben ist. Es handelt sich hierbei um eine Abwandlung des EKF-SLAM, bei der Partikelfilter verwendet werden, die unter anderem auch nichtlineare Prozesse abbilden können. Durch sie wird erreicht, dass mehrere Hypothesen gleichzeitig verfolgt werden können. Dies führt nach [34] zu einer deut-

lich größeren Robustheit gegenüber fehlerhaften Positionsschätzungen. Auf eine ausführlichere Darstellung wird hier verzichtet, da auch dieser Ansatz auf der Kenntnis von Landmarken basiert. Eine detaillierte Beschreibung des Algorithmus liefern Montemerlo [45] und Hähnel [32]. Bei der Erkundung von unbekannten Innenräumen unter Verwendung eines Laserscanners ist die Abhängigkeit von Landmarken in zweierlei Hinsicht problematisch. Zum einen variieren die Abmessungen möglicher Marken wie Schränke, Tische und Stühle sehr stark und zum anderen müssen solche Gegenstände aus möglichst vielen Perspektiven eindeutig erkannt werden, um eine zuverlässige Lokalisierung und Kartographierung zu ermöglichen. Gerade im Hinblick auf die Verwendung eines Laserscanners ist der zweite Punkt ein wesentlicher Nachteil. Die durch den Sensor zur Verfügung gestellten Daten repräsentieren lediglich eine Ebene des untersuchten Raums mit einer überschaubaren Datenmenge. Aus solchen Messwerten bestimmte für Marken charakteristische Merkmale herauszufiltern führt zu großen Ungenauigkeiten und ist somit nicht praktikabel. Algorithmen, die auf Landmarken und topologischen Karten basieren, sind daher nicht geeignet, um das in dieser Arbeit gesteckte Ziel zu erreichen.

*Ansätze mit metrisch diskreten Karten*

Diese Art von SLAM-Algorithmen basieren auf Karten, die eine Gitterstruktur haben. Beispiele hierfür sind eine abgewandelte Form des bereits erwähnten FastSLAM [30], der Grid-based visual SLAM [19], sowie der DP-SLAM [23, 24]. Grundsätzlich wird dabei jedem Gitterelement eine Belegwahrscheinlichkeit zugeordnet. In einem einfachen Fall könnten das die Eigenschaften *belegt*, *frei* oder *offen* sein.
Ein Beispiel solch einer metrisch diskreten gitterbasierten Karte zeigt der in Bild 2.2 dargestellte Flurabschnitt. Die freien Gitterzellen sind weiß, die belegten dunkel gefüllt. Man kann in Bild 2.2 klar den Verlauf des Flurs erkennen. Die Ausbuchtungen repräsentieren geschlossene Türen. Über die grauen Felder ist aufgrund der fehlenden Sichtverbindung keine eindeutige Aussage möglich. Sie werden in diesem Zusammenhang als *offen* bezeichnet.

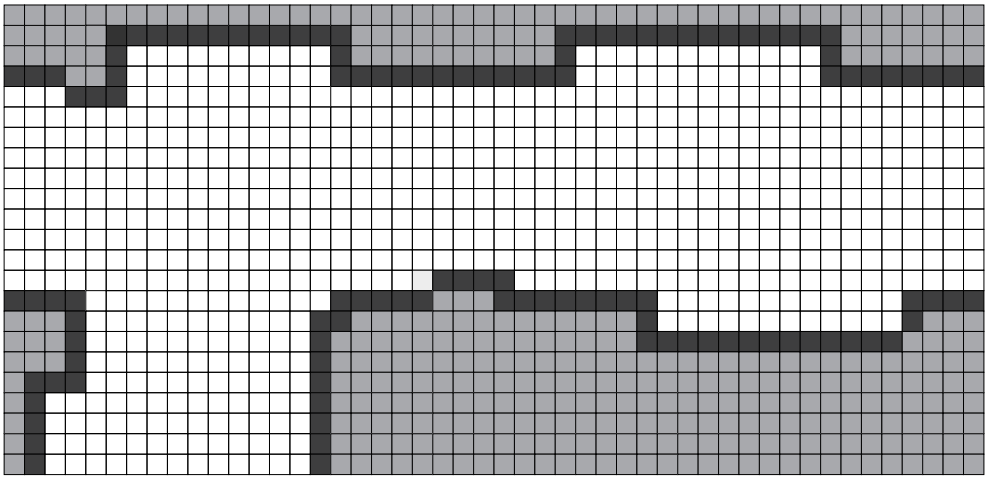


**Bild 2.2:** Gitterbasierte Karte mit drei Zuständen nach [61]

Ein Vorteil dieser Art der Algorithmen ist, dass durch die einheitliche und sehr einfache Datenstruktur die erforderlichen Rechenoperationen grundsätzlich sehr effizient durchgeführt werden können. Man kann sich leicht vorstellen, dass die Menge der zu verarbeitenden und zu speichernden Daten mit der Größe des gewählten Rasters zusammenhängt. Um eine genaue Karte zu bekommen, sollte das Raster möglichst fein sein. Dies hat jedoch negative Auswirkungen auf den Speicherbedarf und den Rechenaufwand, da pro Zeitschritt deutlich mehr Operationen ausgeführt werden müssen, als bei einem groben Raster. Bei einem solchen ist der benötigte Speicherplatz dementsprechend geringer. Es kommt jedoch zu vielen Teilüberdeckungen von einzelnen Zellen und detektierten Hindernissen. Damit steigt das Quantisierungsrauschen und die resultierende Karte wird ungenau.
Zur Lösung dieses Problems hat sich die Verwendung unterschiedlicher Rastergrößen durchgesetzt. Dabei werden Zellen mit gleichen Informationen in einer größeren Zelle zusammengefasst. Ein entsprechendes Beispiel zeigt Bild 2.3. Um eine solche Teilung zu erreichen, wird das gesamte Areal zunächst als eine Zelle aufgefasst. Enthält diese freie und belegte Bereiche, wird sie in vier gleich große Teile aufgespalten. Mit diesen vier wird jeweils nach gleicher Art und Weise verfahren. Enthält eine Zelle nur freie Bereiche, wird sie nicht weiter geteilt. Wird eine vorher definierte minimale Zellengröße erreicht, bricht der Algorithmus ebenfalls ab [34].

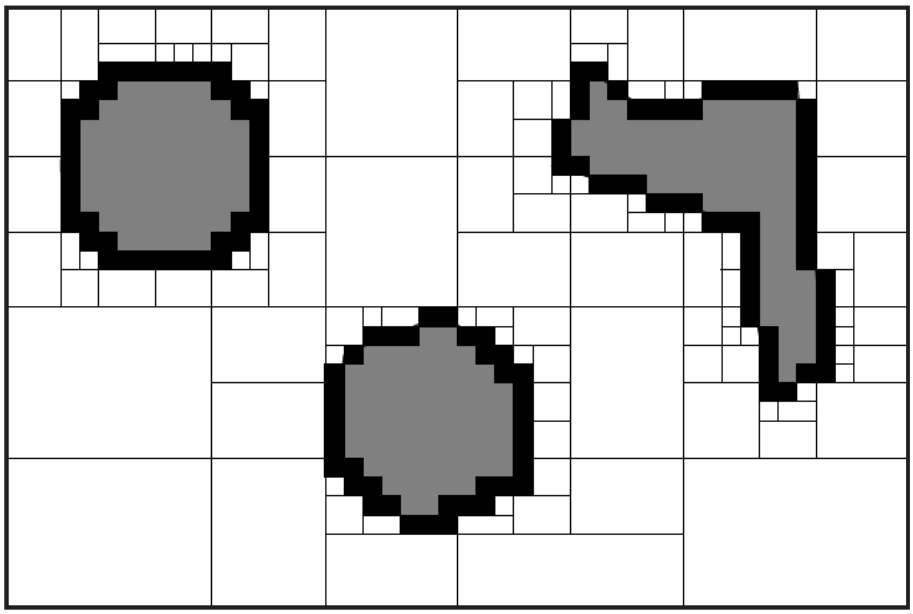


**Bild 2.3:** Karte mit angepasster Gittergröße [34]

Trotz einer solchen Optimierung der Größe der einzelnen Gitterzellen und der damit einhergehenden Reduktion des Speicherbedarfs und des Rechenaufwands bei gleichbleibender Genauigkeit bleibt ein entscheidender Nachteil dieser Kategorie von SLAM-Algorithmen. Die Größe und die grobe Struktur des zu erkundenden Areals sollten bekannt sein, damit die Karte geeignet initialisiert werden kann. Dies bezieht sich zum einen auf die Gesamtgröße und zum anderen auf die minimale Größe der einzelnen Zelle. Daher sind auch diese Verfahren zur Kartographierung unbekannter Innenräume nur bedingt geeignet.

*Ansätze mit metrisch kontinuierlichen Karten*

In metrisch kontinuierlichen Karten werden die Daten in Form von Koordinaten abgespeichert. Durch die Kontinuität der Achsen haben solche Karten theoretisch eine unendlich hohe Genauigkeit. Da die Informationen jedoch digital gespeichert werden, muss auch hier letztlich ein bestimmtes Raster gewählt werden. Im Gegensatz zu den vorher diskutierten Verfahren wirkt sich die gewählte Genauigkeit aber nur sehr gering auf die Laufzeit und den Speicherbedarf aus. Diese beiden Faktoren hängen in dem hier vorliegenden Fall hauptsächlich von der Menge der gespeicherten Koordinaten ab. Die zu speichernden Da-

tenmengen können beispielsweise durch die Approximation von Linien verringert werden. Ein solches Verfahren wird als line-feature-based SLAM bezeichnet [18]. Bild 2.4 zeigt beispielhaft eine zu dieser Methode gehörende Karte.

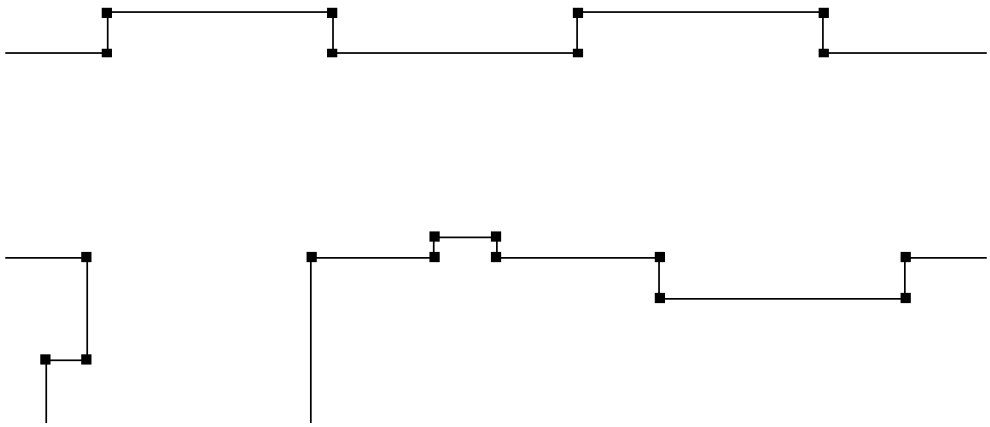


**Bild 2.4:** Linien basierte metrisch kontinuierliche Karte [61]

Die Bilder 2.2 und 2.4 zeigen den gleichen Flurabschnitt. Mit der Repräsentation durch Linien müssen jetzt nur noch die mit schwarzen Quadraten gekennzeichnete Anfangs- und Endpunkte der Linien gespeichert werden. Dies stellt eine deutliche Reduktion des Speicherbedarfs dar. Diesen Vorteil erkauft man sich jedoch mit dem Nachteil, dass die Originaldaten für die weitere Verwendung verloren gehen. Es gibt jedoch die Möglichkeit, über die zwei Koordinaten hinaus noch weitere Eigenschaften zusätzlich zu speichern. Im Verlauf des folgenden Kapitels 3 wird deutlich, dass beispielsweise die Anzahl der Punkte, die zu einer Linie geführt haben, eine hilfreiche Größe sein kann.

Da Innenräume häufig eine vorwiegend rechtwinklige Struktur aufweisen, bietet sich die Verwendung eines orthogonalen Verfahrens, dem sogenannten orthogonal SLAM an [48]. Dieser Algorithmus ist eine Sonderform des auf Linien basierenden Verfahrens, bei dem nur orthogonale Strukturen in die Karte übernommen werden. Dadurch wird der Rechenaufwand für die einzelnen Zeitschritte noch einmal deutlich reduziert, wie die ausführliche Beschreibung in Kapitel 3 zeigt. SLAM-Algorithmen,

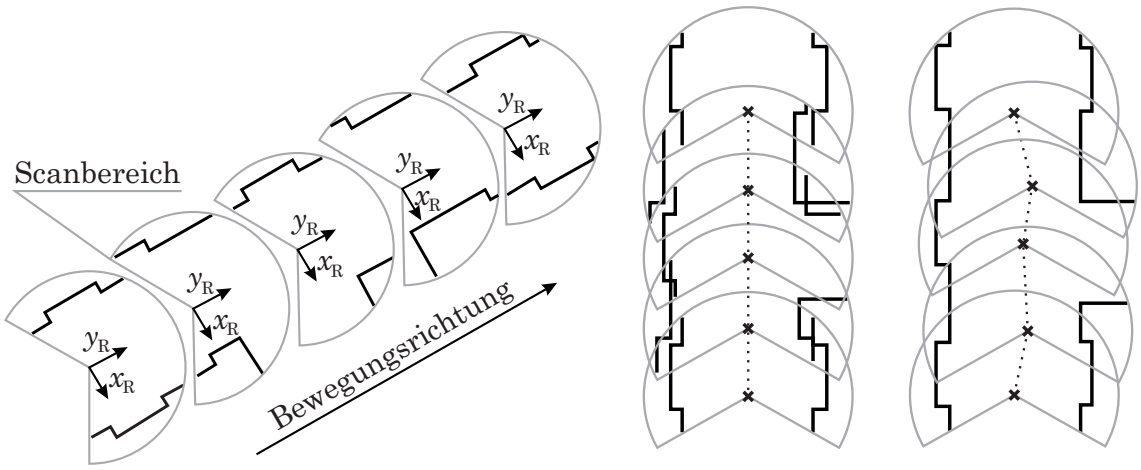
die auf dieser Art der Kartographierung basieren, benötigen keine Vorabinformationen über die Größe des zu erkundenden Areals und keine Kenntnis von Landmarken. Diese Tatsache und die bedarfsgerechte Genauigkeit machen diese Verfahren für die Erkundung von unbekannten Innenräumen sehr interessant. Der später implementierte Algorithmus gehört daher zu dieser Gruppe und wird in Kapitel 3 ausführlich beschrieben.

## 2.2 Scan-Matching-Verfahren

Zur vollständigen Kartographierung einer unbekannten Umgebung sind in der Regel Messungen an unterschiedlichen Orten erforderlich. Die beispielhaften Karten im vorherigen Abschnitt zeigen, dass aufgrund der Position des Roboters einige Teile des zu untersuchenden Raums verdeckt sein können. Des Weiteren ist aufgrund der begrenzten Reichweite des eingesetzten Sensors nicht sichergestellt, dass alle Teile des Raums durch eine Messung erfasst werden. Die Daten aus einer einzelnen Messung werden jeweils im roboterfesten Koordinatensystem, das durch die $x_{\mathrm{R}}$ - Achse und die $y_{\mathrm{R}}$ - Achse aufgespannt wird, zu einer lokalen Karte verarbeitet. Durch die Kombination der lokalen Karten ergibt sich schließlich die globale Karte. Diese wird im Weltkoordinatensystem gespeichert. Die zugehörigen Achsen werden hier durch den Index ‚E‘ gekennzeichnet. Für diesen Schritt sind Algorithmen erforderlich, die die Überlappung von zwei Messungen oder einer Messung und den bereits vorhandenen Daten ermitteln können. Diese werden in der Literatur als Scan-Matching-Verfahren bezeichnet [34, 50, 54]. Da sich aus der Überlappung auch die Position des Roboters bestimmen lässt, kommen diese Verfahren vorwiegend dort zum Einsatz, wo GPS-Empfang nicht sichergestellt ist.

Im Allgemeinen führt der Roboter zwischen zwei Messungen eine translatorische und eine rotatorische Bewegung aus. Im Falle der Bewegung in der $x_{\mathrm{E}} / y_{\mathrm{E}}$ - Ebene müssen somit die beiden Achsenabschnitte $x_{\mathrm{E}}$ und $y_{\mathrm{E}}$ sowie der Winkel $\Psi$, der die Ausrichtung des Roboters innerhalb des Weltkoordinatensystems beschreibt, durch ein Scan-Match-Verfahren

geschätzt werden. Mit Hilfe dieser Information lässt sich bei bekannten Startkoordinaten die absolute Position des Roboters in der betrachteten Ebene bestimmen und die Daten der aktuellen Messung können richtig in die globale Karte eingegliedert werden. Die Mehrzahl der bekannten Ansätze behandelt das 2D-Scan-Matching. Grundsätzlich ist auch deren Erweiterung auf eine dritte Dimension möglich. Diese Möglichkeit ist jedoch für den weiteren Verlauf dieser Arbeit nicht relevant und wird somit nicht weiter ausgeführt.
Für die erfolgreiche Anwendung aller Verfahren gilt, dass in den beiden verwendeten Datensätzen eine ausreichend große Überlappung bestehen muss. Dies ist in der Praxis durch die Anpassung von Abtastrate der Messungen und der Geschwindigkeit des Roboters sicher zu stellen. Innerhalb dieser Überlappung sollte eine eindeutige Struktur vorhanden sein, damit die Position eindeutig bestimmt werden kann. Bild 2.5 zeigt anhand eines kleinen Beispiels die Notwendigkeit solcher Verfahren.




(a) Einzelmessungen (b) ohne Scan-Match (c) mit Scan-Match

**Bild 2.5:** Kartenerstellung mit und ohne Scan-Match Verfahren nach [61]

In Bild 2.5(a) sind fünf bereits zu Linien verarbeitete Messungen im Roboterkoordinatensystem dargestellt, die auf einer Fahrt entlang eines Flures aufgenommen wurden. Die globale Karte soll aus diesen fünf lokalen Karten zusammengesetzt werden. Erfolgt diese Überlagerung

nur auf Basis der kommandierten Fahrbefehle, so ist Bild 2.5(b) ein mögliches Ergebnis. Die lokalen Karten wurden hier auf Basis einer idealen konstanten Geschwindigkeit übereinander gelegt. Da der Roboter während seiner Fahrt jedoch im Allgemeinen Störungen ausgesetzt ist, kann die Struktur des kartographierten Raumes in der resultierenden globalen Karte nicht eindeutig erkannt werden. Verwendet man wie in Bild 2.5(c) jedoch ein Scan-Match-Verfahren, so ergibt sich eine klar strukturierte Karte, aus der für den Anwender alle wesentlichen Merkmale des untersuchten Raumes erkennbar sind. Die schwarze gestrichelte Linie gibt den Weg des Roboters an. Vergleicht man diesen in den Bildern 2.5(b) und 2.5(c), so werden die Abweichungen von der in Bild 2.5(b) getroffenen idealen Annahme deutlich.

Die populärsten 2D-Scan-Match-Verfahren sind iterative Verfahren. Alle haben gemeinsam, dass man in beiden Datensätzen korrespondierende Punkte oder Punktmengen findet und diese iterativ zur Deckung bringt. Die drei gebräuchlichsten Heuristiken hierfür sind der ICP (engl. ***I**terative **C**losest **P**oints*), der IMR (engl. ***I**terative **M**atching **R**ange*) und der IDC (engl. ***I**terative **D**ual **C**orrespondance*).
Beim ICP-Algorithmus [10, 42] wird zu jedem Messwert $p$ des Datensatzes $P$ derjenige Punkt $p'$ aus der vorangegangenen Messung $P'$ gesucht, welcher einen minimalen euklidischen Abstand hat. Führt man eine maximale Schranke $D_{\max}$ ein, so dass $|p - p'| < D_{\max}$ gelten muss, damit ein Punktepaar $(p, p')$ zur Berechnung der gesuchten Transformation berücksichtigt wird, kann die Laufzeit verkürzt werden. Die jeweiligen Punkte $p$ und $p'$ werden dabei in kartesischen Koordinaten beschrieben.

Im Gegensatz dazu werden beim IMR-Verfahren [20, 42] die Punkte in Polarkoordinaten verarbeitet. Bild 2.6 zeigt das Prinzip dieses Verfahrens, bei dem die Abstände $|O\,p|$ und $|O\,p'|$ im Fokus stehen. Der Punkt $p$ der aktuellen Messung $P$ wird dem Punkt $p'$ zugeordnet, der innerhalb des Winkelintervalls $[\gamma - \nu, \gamma + \nu]$ liegt und für den der Betrag der Differenz $|O\,p| - |O\,p'|$ minimal wird. $\gamma$ ist dabei der eingeschlossene Winkel zwischen $|O\,p|$ und der $x$-Achse. Über den Parameter $\nu$ kann der zulässige Winkelbereich eingeschränkt werden, um die Konvergenz

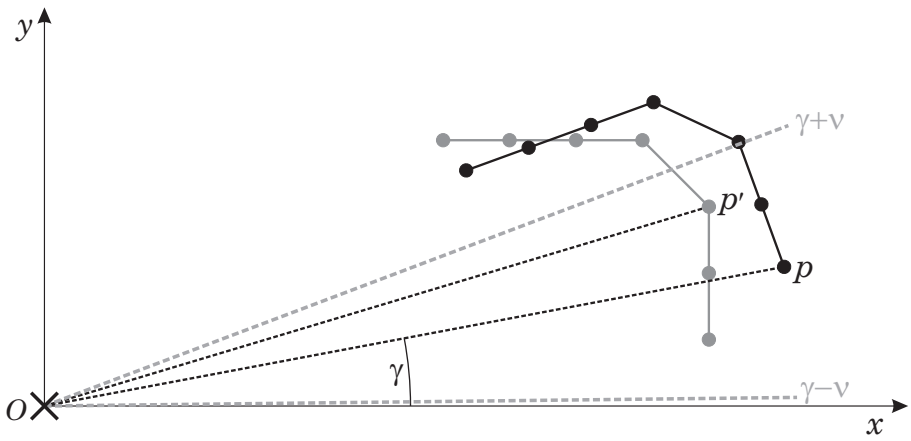
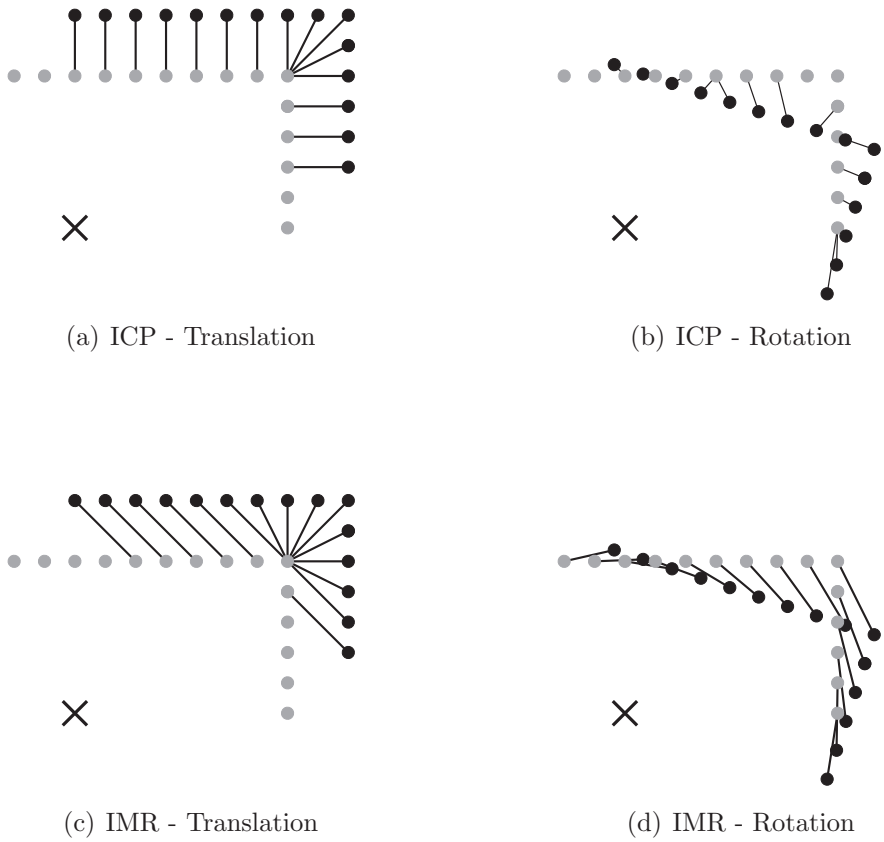



**Bild 2.6:** Prinzip des IMR-Verfahrens nach [42]

des Verfahrens zu beschleunigen. Diese Methode hat Vorteile bei der Ermittlung der Rotation. Translationen lassen sich hingegen besser mit dem erst genannten Verfahren bestimmen. Bild 2.7 stellt die beiden unterschiedlichen Punktkorrespondenz-Strategien gegenüber.



(a) ICP - Translation

(b) ICP - Rotation

(c) IMR - Translation

(d) IMR - Rotation

**Bild 2.7:** Gegenüberstellung von ICP und IMR nach [34]

Die schwarz gekennzeichneten Punkte repräsentieren jeweils die Punktemenge $P$ des aktuellen Scans. In grau ist die entsprechende Referenz oder die vorangegangene Messung $P'$ markiert. Die Position des Roboters ist jeweils durch das Kreuz markiert. Aus Bild 2.7(a) wird deutlich, wie bei der Bestimmung einer Translation durch den ICP-Algorithmus alle Punkte $p$ des aktuellen Scans $P$ einem Referenzpunkt $p'$ zugeordnet werden. Dabei können die Referenzpunkte mehrfach verwendet werden. Gleiches gilt für die Abschätzung der Rotation in Bild 2.7(b). In den jeweils darunter liegenden Abbildungen ist die entsprechende Zuordnung für das IMR-Verfahren zu sehen, bei dem $\nu = 25°$ gewählt wurde.
Vergleicht man die beiden linken Bilder, so wird der oben angesprochene Vorteil des ICP in Bezug auf die Translation deutlich. In Bild 2.7(c) sind deutlich weniger Punkte ihrem tatsächlichen Abbild zugeordnet. Bei der Rotation stellt es sich erwartungsgemäß genau anders herum dar. Hier liegen die Vorteile beim IMR-Verfahren. Eine Kombination aus beiden Ansätzen bildet der IDC-Algorithmus. Hier wird in jedem Iterationsschritt je eine Transformation auf Basis des ICP und des IMR berechnet und als Ergebnis für die Translation bzw. die Rotation übernommen [34].

Komplexere Scan-Matching-Methoden lassen sich grundsätzlich in unimodale und multimodale Ansätze unterteilen. Bei erst genannten wird jeweils eine Position ermittelt und für die weitere Verwendung gespeichert. Im anderen Fall werden Aufenthaltswahrscheinlichkeiten bestimmt. Diese bleiben über den gesamten Verlauf des Algorithmus erhalten und werden auf Basis der aktuellen Messung angepasst. So können zunächst fehlerhafte Positionen nachträglich korrigiert werden. Die bekanntesten Verfahren in diesem Bereich sind die Markow- und die Monte-Carlo-Lokalisierung. Für diese ist jedoch die Verwendung eines GraphSLAM erforderlich, welcher in dieser Arbeit nicht zum Einsatz kommt. Gleiches gilt damit für die genannten Scan-Match-Methoden. Die unimodalen Verfahren, die im weiteren Verlauf kurz vorgestellt werden, beinhalten meist den ICP. Da ähnlich wie bei den SLAM-Algorithmen über die Jahre sehr viele Ansätze publiziert wurden, sind hier nur die bekanntesten Verfahren genannt.

Bei der Kalman-Lokalisierung [54] wird ein EKF verwendet, das die Ergebnisse des ICP zusätzlich mit einer Abschätzung der Position durch Odometriedaten fusioniert. Diese Daten werden auf Basis des Vortriebssystems des Roboters ermittelt. Bei Bodenfahrzeugen ist dies beispielsweise die Anzahl der Radumdrehungen. Bei Flugplattformen werden diese Daten im Allgemeinen durch verschiedene Beschleunigungsaufnehmer, die zu einer IMU (engl. ***I**nertial **M**easurement **U**nit*) zusammengefasst sind, zur Verfügung gestellt. Aufgrund des eingeschränkten Zugriffs auf diese Daten bei dem später verwendeten industriellen Quadrotor, der in Kapitel 4.1 im Detail vorgestellt wird, ist das Verfahren für diese Arbeit nicht empfehlenswert.
Es existieren auch Methoden, bei denen aus den Messdaten zunächst gewisse Merkmale extrahiert werden [34]. Diese werden dann anschließend mit den genannten iterativen Verfahren zur Deckung gebracht. In diesem Fall müssen die zu erwartenden Merkmale vorher bekannt und in ausreichender Anzahl in der jeweiligen Messung vorhanden sein, damit ein gutes Ergebnis erzielt wird. Diese Art des Scan-Matching wird vorzugsweise in Verbindung mit dem entsprechenden Verfahren für die Kartographierung verwendet. Aus den obigen Ausführungen wird bereits deutlich, dass dieser Ansatz hier nicht weiter verfolgt wird.

Passend zu dem oben für diese Arbeit gewählten Kartentyp ist die Lokalisierung anhand von Linien. Hier wird üblicherweise eine bereits bekannte Linie durch eine Anzahl von Punkten repräsentiert, damit der ICP-Algorithmus verwendet werden kann. Somit ist auch dieses Verfahren, wie alle bisher genannten, iterativ. Damit geht einher, dass eine genaue Aussage in Bezug auf die Laufzeit sehr schwierig ist. Sie kann lediglich durch eine Begrenzung der Iterationsschritte nach oben abgeschätzt werden.

Ein Ansatz, der eine geschlossene Form der Lösung ermöglicht, ist das Hayai-Verfahren [41]. Hier werden ähnlich wie bei der Lokalisierung anhand von Merkmalen bestimmte Charakteristika aus den jeweiligen Scans miteinander verglichen. Das Ergebnis ist eine Transformation, die sowohl die Rotation als auch die Translation zwischen beiden

verglichenen Datensätzen beinhaltet. Vergleicht man zwei unmittelbar aufeinander folgende Scans, ist das Verfahren besonders geeignet, da es auf Basis der Rohdaten eines Laserscanners sehr effizient funktioniert. Sollen bei der Abschätzung der Transformation hingegen auch weiter zurückliegende Informationen verwendet werden, ist diese Methode nicht sehr attraktiv. Zudem scheitert das Verfahren, wenn keine zu erwartenden Charakteristika wie beispielsweise Ecken oder Sprünge in einem Datensatz des Laserscanners auftreten.

Alle genannten Ansätze haben Vor- und Nachteile. Für die im Rahmen dieser Arbeit betrachtete, auf orthogonalen Linien basierende Kartographierung ist eine entsprechende Lokalisierung zu bevorzugen. Mit Blick auf das Echtzeitverhalten ist jedoch eine geschlossene Lösung wünschenswert, die auch die Verwendung von Informationen aus zurückliegenden Messungen möglich macht. Daher wird eine Alternative zu dem Hayai-Algorithmus vorgestellt. Sie basiert auf Kenntnis der Orientierung $\Psi$ des Roboters, deren Abschätzung im Abschnitt 3.1 ausgeführt wird. Somit kann das Scan-Matching auf die Ermittlung der translatorischen Bewegung in der $x_\mathrm{E} / y_\mathrm{E}$ - Ebene beschränkt werden. Dabei ist aufgrund des gewählten Kartentyps ein Verfahren sinnvoll, das direkt auf der Basis von Linien arbeitet. Eine solche Methode wird in Abschnitt 3.3 detailliert beschrieben.

## 2.3 Navigation von Robotern

Damit sich ein Roboter eigenständig in einer unbekannten Umgebung bewegen kann, muss er über eine Navigations- oder Bewegungsstrategie verfügen. Ähnlich wie bei den zuvor behandelten Themen sind auch in diesem Bereich bereits viele Ansätze vorgestellt worden. Einen guten Überblick bieten die Bücher von Fujimura [27] und Ibrahim [37]. Der Großteil dieser Algorithmen ist in Zusammenhang mit Bodenrobotern behandelt worden. Die erzielten Ergebnisse lassen sich jedoch grundsätzlich auch auf Flugplattformen übertragen.
Die Navigation eines Roboters kann nach einer nahezu beliebig komplexen Strategie erfolgen. Viele Probleme lassen sich jedoch mit sehr

einfachen Verhaltensregeln lösen. Die hier vorgestellten Verfahren werden gemäß Bild 2.8 nach der Frequenz ihrer Kontrollzyklen und der Komplexität der erforderlichen Sensordatenverarbeitung kategorisiert. Wiederum können hier aufgrund der Vielzahl von Methoden nicht alle genannt werden.

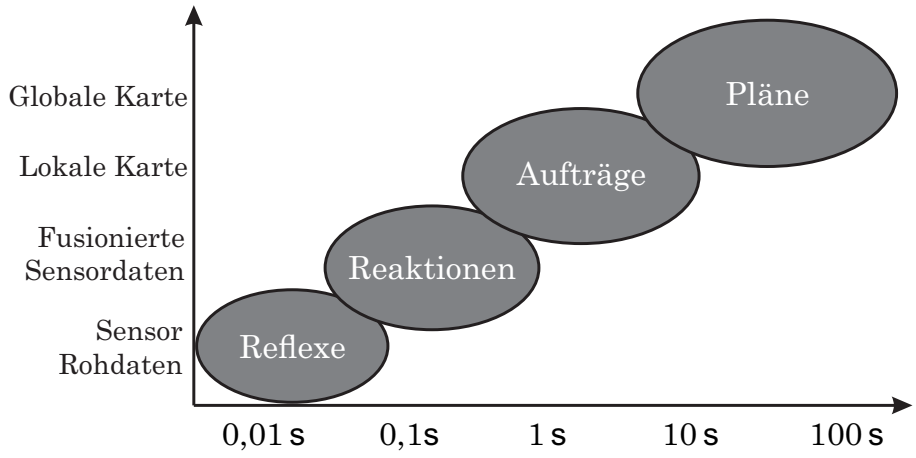



**Bild 2.8:** Darstellung verschiedener Navigationsebenen nach [34]

Auf der $x$-Achse des Diagramms in Bild 2.8 ist die Zykluszeit der zugehörigen Algorithmen der verschiedenen Navigationsebenen angegeben. Auf der $y$-Achse ist die erforderliche Verarbeitungsstufe für die Sensordaten aufgetragen. Die Bedeutung der einzelnen Begriffe wird im Zusammenhang der Erläuterung der vier Ebenen deutlich. Anhand dieser werden im Folgenden verschiedenen Verfahren zur Navigation von Robotern vorgestellt und charakterisiert.

*Reflexe*

Diese untere Navigationsebene hat ihren Namen aufgrund der Analogie zum menschlichen Verhalten. Berührt ein Mensch versehentlich eine heiße Herdplatte, so zuckt seine Hand im Normalfall sofort zurück. Diese Handlung bezeichnet man als Reflex. Übertragen auf den Roboter sind Reflexe unmittelbare Reaktionen auf äußere Einflüsse, die durch Sensoren oder Schalter erfasst werden. So kann etwa die Unterschreitung eines gewissen Sicherheitsabstands unmittelbar zum Anhalten des

Roboters führen. Charakteristisch ist, dass die Reflexe des Roboters aus den Rohdaten der entsprechenden Sensoren hervorgehen und keine Vorverarbeitung der Messdaten erforderlich ist. Damit können die in dieser Ebene angesiedelten Schutzfunktionen sehr schnell ausgeführt werden. Aufgrund dieser Reflexe, die im Allgemeinen zu sehr starken Änderungen der Stellsignale und damit zu harten Manövern führen, wird ein Roboter in hohem Maße mechanisch beansprucht. Daher ist eine Navigation, die ausschließlich auf das hier beschriebene Verhalten beruht, nicht sinnvoll. Im Hinblick auf eine Erkundung können in dieser Ebene auch keine nützlichen Ziele vorgegeben werden. Dies ist jedoch indirekt mit der nächst höheren möglich.

*Reaktionen*

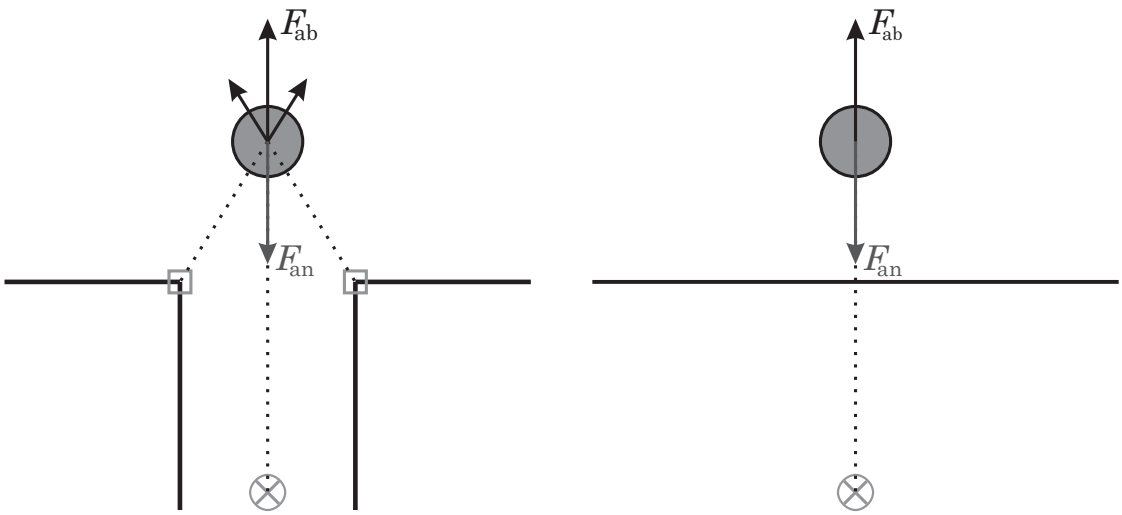
In dieser Ebene sind einfache Verhaltensmuster zusammengefasst. Solche sind zum Beispiel das Folgen einer Kontur oder die Orientierung in den freien Raum. Die Definition sehr einfacher Regeln kann dazu führen, dass ein Roboter in der Lage ist, vergleichsweise komplexe Aufgaben zu erledigen. So kann das Folgen einer Kontur beispielsweise ermöglichen, dass er ein ihm unbekanntes Labyrinth durchquert und sichergestellt ist, dass er zum Ausgangspunkt wieder zurückkehrt. Bild 2.8 zeigt, dass die für diese Ebene erforderliche Sensordatenvorverarbeitung nicht sehr aufwendig ist. Daher steht bei dieser Art der Navigation ein sehr großer Teil der vorhandenen Rechenkapazität für andere Aufgaben, wie beispielsweise die Kartographierung, zur Verfügung.
Auch die Anpassung der Geschwindigkeit an die Umgebung ist innerhalb dieser Navigationsebene möglich. Durch den sogenannten *dynamic window approach* [25] wird ausgehend von der aktuellen Position für verschiedene Geschwindigkeiten eine Prädiktion der potentiellen neuen Standorte durchgeführt. Die Informationen der jeweiligen Umgebungen werden in einem zweidimensionalen Histogramm gespeichert und ausgewertet. Die untersuchte Region wird als Fenster bezeichnet und gibt dem Verfahren seinen Namen. Auf Basis der verschiedenen Histogramme wird schließlich die Geschwindigkeit für den zugehörigen Abtastschritt ermittelt. Durch einen solchen Ansatz, der im Detail in [25] angegeben ist, werden abrupte Änderungen der Stellgröße, wie sie bei den Reflexen

auftreten, vermieden.
Neben den offensichtlichen Vorteilen bietet diese Form der Navigation auch Nachteile. Roboter, die im Innenbereich eingesetzt werden, verfügen im Allgemeinen über eine elektrische Energieversorgung. Die in den Akkumulatoren gespeicherte Energie ist begrenzt und sollte möglichst effizient genutzt werden. Im Hinblick auf diese Arbeit bedeutet dies, dass möglichst viele Informationen mit der zur Verfügung stehenden Energie gesammelt werden. Durch die Formulierung sehr einfacher Regeln kann man diesem Anspruch nicht gerecht werden. Folgt ein Roboter beispielsweise einer Kontur, so können bestimmte Bereiche eines Areals mehrfach durchquert werden, während andere noch unbekannt sind. Um weg- oder zeitoptimale Strategien zu ermöglich, werden die höheren Ebenen ‚Aufträge‘ und ‚Pläne‘ benötigt.

*Aufträge*

Diese Navigationsebene erfordert in der Regel die darüber liegende Planungsebene. Aus dieser werden prinzipiell Fahraufträge abgeleitet, die der Roboter ausführen soll. So ist zum Beispiel das Anfahren von Wegpunkten ein möglicher Auftrag. Damit beinhaltet diese Ebene die Verfahren, die sich auch unter dem Begriff „lokale Navigation“ subsummieren lassen. In diesem Bereich gibt es unter anderem die Potentialfeldmethode [38] oder auf Vektoren basierende Verfahren [12]. Diese beiden Ansätze sollen im weiteren Verlauf, stellvertretend für die zahlreich in der Literatur vorhandenen, kurz erläutert werden.
Bei der Methode, die auf Vektor-Histogrammen (engl. ***V****ector* ***F****ield* ***H****istogramm*, kurz VFH) basiert, wird die Umgebung des Roboters zunächst in einem zweidimensionalen Gitter in kartesischen Koordinaten gespeichert. Dabei wird jeder Gitterzelle ein Wert in Abhängigkeit der Wahrscheinlichkeit für ein Hindernis zugeordnet. Je höher der Wert einer Zelle ist, desto größer ist die Wahrscheinlichkeit für ein Hindernis. Im Anschluss werden aus dieser Darstellung unter Berücksichtigung der aktuellen Position des Roboters die Belegwahrscheinlichkeiten über die mögliche Fahrtrichtung aufgetragen. In dem resultierenden Kurvenverlauf wird nach einem Minimum gesucht, auf dessen Basis die neue Fahrtrichtung festgelegt wird. Einzelheiten dieses Verfahrens

können in der zugehörigen Quelle [12] nachgelesen werden.
Die Potentialfeldmethode (engl. ***A**rtificial **P**otential **F**ields Method*, kurz APF) modelliert Roboter als geladenes Teilchen. Die Umgebung wird entsprechend durch positive und negative Ladungen beschrieben und ist wie im Fall zuvor durch eine gitterbasierte Karte beschrieben. Da sich gleiche Ladungen abstoßen, werden die Hindernisse und der Roboter im Allgemeinen durch Punktladungen mit gleichem Vorzeichen belegt. Der Zielpunkt hingegen wird mit einer entgegengesetzten Ladung versehen. Somit ergeben sich anziehende (attraktive) und abstoßende (repulsive) Kräfte, die auf den Roboter wirken. Damit entspricht das Modell der Bewegung eines geladenen Teilchens in einem Potentialfeld. Um dieses Verfahren auch bei den im Rahmen dieser Arbeit verwendeten metrisch kontinuierlichen Karten anwenden zu können ist eine in Abschnitt B.1 beschriebene Abwandlung erforderlich.
Nachteil dieser Art der Navigation sind lokale Minima, die zu einer Bewegungsunfähigkeit des Roboters führen können. Bild 2.9 zeigt zwei solcher Situationen am Beispiel der Potentialfeldmethode.




(a) Minimum bei Durchfahrt (b) Zielpunkt jenseits des Hindernisses

**Bild 2.9:** Möglichkeiten für ein lokales Minimum bei lokaler Navigation [53]

In Bild 2.9(a) wird das grundsätzliche Prinzip des oben beschriebenen Verfahrens deutlich. Aus Gründen der Übersichtlichkeit ist hier nur die abstoßende Kraft $F_{\mathrm{ab}}$ je eines Wandelements dargestellt. Die Wirkung

des mit einem Kreuz markierten Zielpunkts auf den Roboter äußert sich durch eine anziehende Kraft $F_{\text{an}}$. In diesem Beispiel heben sich beide auf ($\|F_{\text{an}}\| = \|F_{\text{ab}}\|$) und der Roboter ist in der aktuellen Position „gefangen“. Bild 2.9(b) zeigt eine weitere Möglichkeit eines lokalen Minimums. Hier ist der Roboter durch eine durchgehende Wand vom Ziel getrennt und die jeweiligen Kräfte heben sich ebenfalls auf. In beiden gezeigten Beispielen kann unter der Annahme einer konstanten Umgebung der Roboter nur durch eine Änderung der Zielvorgabe aus der Situation befreit werden. Dafür wird die übergeordnete Planungsebene benötigt [51].

*Pläne*

In dieser obersten Navigationsebene werden übergeordnete Ziele formuliert. Für die vorliegende Arbeit könnte dies zum Beispiel die vollständige Erkundung des Raumes in möglichst kurzer Zeit sein. In dieser Ebene greifen die entsprechenden Verfahren in der Regel auf Karten zurück, die entweder vorher bekannt oder im Verlauf der Mission enstanden sind. Über Suchfunktionen wie den A*-Algorithmus [33] können Sichtbarkeitsgraphen [49], zu denen im Allgemeinen auch Karten zählen, effizient nach dem optimalen Weg durchsucht werden. Gemäß Bild 2.8 auf Seite 19 sind die Taktzyklen für diese Berechnungen vergleichsweise groß. Daher kann es sinnvoll sein, die dafür erforderlichen Algorithmen nicht an Bord des Roboters, sondern auf der zugehörigen Bodenstation zu implementieren. Damit wird der an Bord befindliche Rechner deutlich entlastet und kann kleiner dimensioniert werden. Dies spart in der Regel Kosten und Gewicht.

In dieser Ebene können auf Basis vorgegebener Kriterien auch eigenständig Zielpunkte gesucht werden, die zur Erfüllung des übergeordneten Auftrags sinnvoll sind. Das Anfahren solcher übergeordneten Zielpunkte kann durch Setzen einzelner Wegpunkte mit Hilfe der lokalen Navigation erreicht werden [47]. Eine solche Kombination beider Navigationsebenen bildet auch die Basis einer selbst entwickelten Strategie [4], die im Jahre 2010 im Rahmen eines Konferenzbeitrages veröffentlicht wurde und im Kapitel B ausführlich beschrieben ist. Hier ist die zentrale Eigenschaft des Roboters, dass er neugierig bleibt. Es soll immer das Ziel

angefahren werden, bei dem der größte Informationszuwachs zu erwarten ist. Realisiert wird dies durch ein Bewertungsschema, welches den offenen Enden einer Karte eine Wertigkeit in Bezug auf die zu erwartenden neuen Informationen zuordnet. Dieser Ansatz ist zur Erkundung unbekannter Innenräume besonders geeignet, da sinnvolle Zielpunkte auf Basis einer unvollständigen Karte ermittelt werden können.

# 3 Echtzeitfähiger SLAM-Algorithmus

In dieser Arbeit wird ein Laserscanner als zentraler optischer Sensor verwendet. Gegenüber einer Kamera hat er zwar eine höhere Masse, die zur Verfügung stehenden Daten sind jedoch durch die direkte Abstandsmessung deutlich einfacher in Echtzeit zu verarbeiten. Bei einer Kamera ist eine aufwendige Bildverarbeitung erforderlich, um die für die Kartographierung benötigten Abstände zu ermitteln. Zudem ist der Laserscanner nahezu unabhängig von den gegebenen Lichtverhältnissen. Den allgemeinen Nachteil, dass durchsichtige Materialien wie beispielsweise Glas nicht erkannt werden, besitzen alle optischen Methoden. Da Flugplattformen, die sich für eine Anwendung im Innenraum eignen, im Allgemeinen im Vergleich zur Raumhöhe nur eine geringe Höhe aufweisen, ist der beim Laserscanner auf eine Ebene reduzierte Sichtbereich kein gravierender Nachteil. Eine detaillierte Beschreibung des verwendeten Sensors erfolgt im Abschnitt 4.2.

Zur Kartographierung wird ein auf orthogonalen Linien basierender SLAM-Algorithmus verwendet, der auf einem Ansatz aus [48] beruht. Dieses Verfahren ist aus den in Abschnitt 2.1 genannten Gründen für überwiegend rechtwinklig strukturierte Innenräume in Bezug auf die Echtzeitfähigkeit besonders gut geeignet. Um eine möglichst geringe Laufzeit und damit ein gutes Echtzeitverhalten des Algorithmus sicher zu stellen, ist die Kenntnis über die Ausrichtung $\Psi$ des Roboters besonders wichtig. Daher wird im ersten Abschnitt dieses Kapitels zunächst die Schätzung des Winkels $\Psi$ auf Basis der optischen Messdaten des Laserscanners erläutert. Dieses Verfahren ist erforderlich, da in Innenräumen kein Empfang von GPS - Signalen sichergestellt werden kann und die Verwendung eines Magnetometers aufgrund von eventuell vorhandenen ferromagnetischen Stoffen wie Eisen keine verlässlichen Daten liefert.
Im Anschluss wird dann der verwendete SLAM-Algorithmus vorgestellt. Dieser ist dahingehend optimiert, dass er später auf einem 16bit Microcontroller, der im Abschnitt 4.2 detailliert beschrieben wird, mit einer

Frequenz von 1 Hz lauffähig ist. Damit ist im Gegensatz zum Ansatz von Grzonka, Grisetti und Burgard aus dem Jahr 2009 [31] keine Bodenstation für die Kartographierung erforderlich. Der hier beschriebene Algorithmus wurde in wesentlichen Teilen in einem eigenen Konferenzbeitrag im Jahr 2010 [5] präsentiert und beinhaltet fünf Schritte.

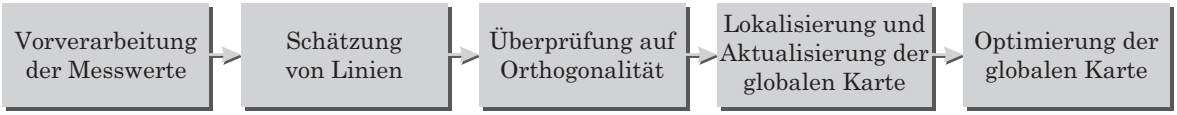



**Bild 3.1:** Schematischer Aufbau des SLAM-Algorithmus

In jedem der fünf Schritte sind einige Entscheidungsvariablen für die Qualität der resultierenden globalen Karte maßgeblich. Im Folgenden werden alle fünf Schritte des in Bild 3.1 dargestellten Algorithmus eingehend beschrieben und die zugehörigen Entscheidungsvariablen und deren Einfluss herausgearbeitet. Abschnitt 3.2 bezieht sich auf die ersten drei und Abschnitt 3.3 auf die letzten beiden Schritte. Der zentrale Punkt des Algorithmus ist der in Bild 3.1 durch den Block „Lokalisierung und Aktualisierung der globalen Karte“ repräsentierte Teil. Im Gegensatz zu den meisten im vorherigen Absatz 2.2 vorgestellten Scan-Match-Verfahren erfolgt diese Aktualisierung hier in einem Iterationsschritt. Abschließend wird der vorgestellte Algorithmus durch eine Simulation mit Hilfe der Software Matlab/Simulink® validiert.

## 3.1 Schätzung der Ausrichtung des Roboters

Die Kenntnis der Ausrichtung $\Psi$ des Sensors und damit des Roboters ist für den in Bild 3.1 dargestellten SLAM-Algorithmus unerlässlich. In diesem Abschnitt wird die Schätzung des Winkels $\Psi$ auf Basis der Daten eines Laserscanners beschrieben. Ein solches Verfahren, welches auf der Auswertung der Winkelhistogramme zweier aufeinanderfolgender Scans beruht, wurde bereits 1994 vorgestellt [58]. Die Histogramme geben die Häufigkeitsverteilung der in einem Scan auftretenden Winkel an. Wie die zugehörigen Winkel im einzelnen bestimmt werden, wird im weiteren Verlauf dieses Abschnitts beschrieben. In der genannten Arbeit werden

die aus den Scandaten bestimmten Winkelhistogramme miteinander korreliert und so die Rotation $\Delta\Psi$ zwischen beiden Scans ermittelt. Bei bekannter Ausgangssituation $\Psi_0$ lässt sich gemäß

$$\Psi_k = \Psi_0 + \sum_{i=1}^{k} \Delta\Psi_i \tag{3.1}$$

die aktuelle Ausrichtung $\Psi_k$ des Roboters aus der Summe aller bis dato ermittelten Rotationen $\Delta\Psi_i$ bestimmen.
Bei der hier vorgestellten Methode kann aufgrund der Tatsache, dass der SLAM-Algorithmus in einer überwiegend rechtwinklig strukturierten Umgebung angewendet werden soll, das Winkelhistogramm des aktuellen Scans mit einem normierten Histogramm verglichen und der zugehörige Winkel $\Delta\Psi$ ohne Berechnung der Kreuzkorrelierten ermittelt werden. Die Basis für dieses vereinfachte Verfahren bildet der enge Zusammenhang zwischen der Bestimmung der Korrelation

$$r_{f_1 f_2}(\tau) = \int_{-\infty}^{\infty} f_1(t) f_2(t+\tau) dt \tag{3.2}$$

und der Faltung

$$h(t) = f_1(t) * f_2(t) = \int_{-\infty}^{\infty} f_1(\tau) f_2(t-\tau) d\tau \tag{3.3}$$

zweier Funktionen $f_1(t)$ und $f_2(t)$ [8]. Die Unterschiede der Gleichungen (3.2) und (3.3) bestehen in der Integrationsvariablen und dem Vorzeichen im zweiten Faktor des jeweiligen Integrals. Aufgrund des positiven Vorzeichens liegt bei der Korrelation in Gleichung (3.2) keine Spiegelung der Funktion $f_2(t)$ in Bezug auf die $y$-Achse vor. Gemäß [8] ist der Dirac-Impuls $\delta(t)$ das neutrale Element der Faltung.

$$x(t) * \delta(t) = x(t) \tag{3.4}$$

Ist dieser wie in Gleichung (3.4) nicht verschoben, ist der beschriebene Unterschied in Bezug auf das Vorzeichen zu vernachlässigen. Damit stellt in diesem Fall der Dirac $\delta(t)$ ebenfalls das neutrale Element der Korrelation dar.

Das Winkelhistogramm einer strikt rechtwinklig strukturierten Umgebung weist in dem Intervall von $[0° \; 360°[$ vier Maxima bei 0°, 90°, 180° und 270° auf, die jeweils als Dirac-Impuls interpretiert werden können. Durch Reduktion dieses Intervalls auf einen Bereich von $[-45° \; 45°[$ um eines der vier Maxima erhält man ein ideales Histogramm, welches als nicht verschobener Dirac-Impuls betrachtet werden kann. Damit liefert ein Vergleich der Lage des Maximums in dem Winkelhistogramm des aktuellen Scans im vergleichbaren Intervall mit der Lage des Maximums im idealen Histogramm eine Abschätzung für die gesuchte Ausrichtung $\Psi$ der Plattform.
Die Reduktion des Intervalls von $[0° \; 360°[$ auf $[-45° \; 45°[$ um ein Maximum kann grundsätzlich auf zwei Arten erfolgen. Zum einen kann sie durch eine Transformation erreicht werden und zum anderen durch eine Prädiktion des zu erwartenden Winkels. Die zweite Variante ist im Rahmen dieser Arbeit realisiert worden. Aus der Prädiktion erhält man einen entsprechenden Offset, der neben dem Vergleich zwischen Referenz und Messdaten zur Abschätzung der Ausrichtung berücksichtigt werden muss. Bei der späteren Implementierung wird immer die Referenz gewählt, welche von den Linien, die parallel zur gewünschten Bewegungsrichtung der Flugplattform verlaufen, hervorgerufen wird. Somit werden durch den Vergleich der Histogramme Abweichungen $\Delta\Psi$ zur gewünschten Flugrichtung bestimmt. Aus der Kombination dieses Ergebnisses mit der vorgegebenen Richtung kann so die absolute Ausrichtung $\Psi$ des Roboters in Bezug auf das Weltkoordinatensystem bestimmt werden. Im Folgenden wird das implementierte Verfahren, das im Rahmen eines Konferenzbeitrages [3] veröffentlicht wurde, anhand eines Beispiels erläutert.

Bild 3.2 zeigt ein beispielhaftes Szenario. Die grauen Linien repräsentieren einen Ausschnitt der im Testfeld eines Labors aufgestellten Wände, die eine rechtwinklige Struktur bilden. Die Achsenbezeichnungen $x_{\mathrm{R}}$ und $y_{\mathrm{R}}$ verdeutlichen, dass die Bestimmumg der Ausrichtung im Roboterkoordinatensystem erfolgt. Der graue Punkt in der unteren rechten Bildecke symbolisiert den Standort des mit einem Laserscanner ausgerüsteten Roboters. Der Pfeil gibt die geforderte Bewegungsrichtung

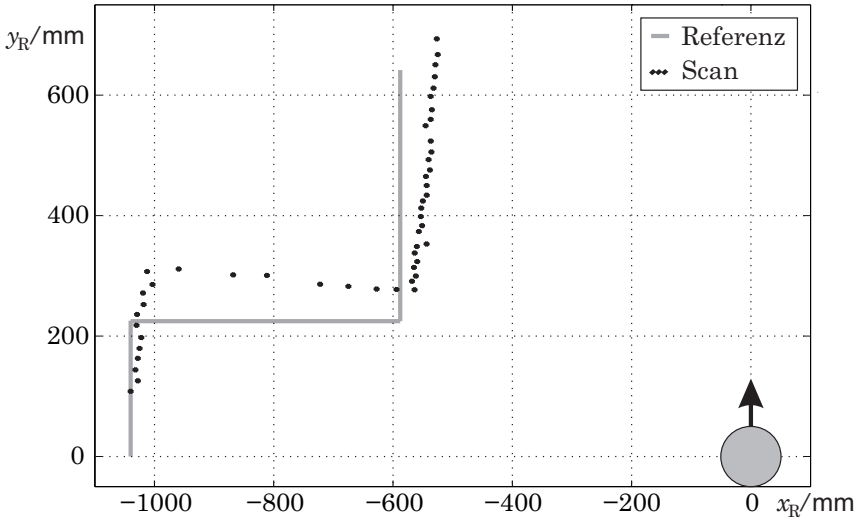



**Bild 3.2:** Beispielhafte Scandaten im Vergleich zur Referenz

des Roboters an. Die schwarzen Punkte sind die Abbildung der aktuellen Messung, die bereits in kartesische Koordinaten umgerechnet wurde. Man sieht, dass sich die Messung und Referenz durch eine Rotation um den Standpunkt des Laserscanners unterscheiden. Aufgrund der später verwendeten Navigationsstrategie wird in diesem Beispiel nur der Bereich links des Roboters betrachtet.

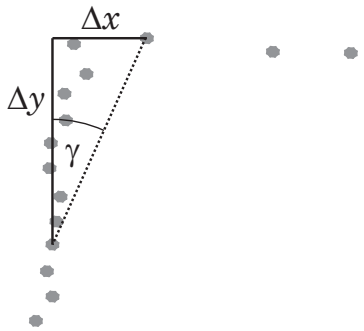



**Bild 3.3:** Bestimmung der Winkel für das Histogramm

Die für das Histogramm benötigten Winkel berechnet man, indem zunächst zwei Punkte der Messung mit einer Linie verbunden werden und ein zugehöriger Winkel $\gamma$ berechnet wird. Bild 3.3 zeigt anhand eines Ausschnitts der in diesem Beispiel verwendeten Messdaten anschaulich die Bestimmung des Winkels, der wie folgt berechnet wird:

$$\gamma = \arctan\left(\frac{\Delta x}{\Delta y}\right). \tag{3.5}$$

Auf dieses Weise werden alle möglichen Wertepaare der Messung beginnend mit dem Punkt im unteren linken Bereich von Bild 3.3 untersucht. Die Anzahl der Punkte, die man beim Einzeichnen der jeweiligen Verbindungslinie überspringt, ist dabei konstant. Es ist nicht empfehlenswert, zwei aufeinanderfolgende Messwerte zu verbinden. Zwar erhält man so die maximale Anzahl an Winkeln und damit eine breite Datenbasis für das Histogramm, jedoch führt dies auch zu einem großen Rauschen. Ist der Abstand zwischen zwei Punkten zu groß, nimmt die Genauigkeit des Verfahrens ab. Es gilt daher, einen geeigneten Kompromiss zu finden. Im hier vorliegenden Fall wurden beim Einzeichnen der Verbindungslinien jeweils neun Werte übersprungen.

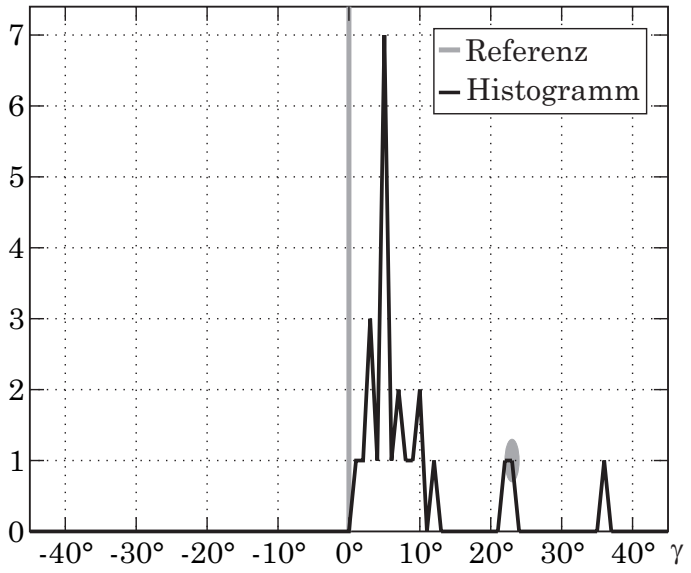



**Bild 3.4:** Zugehöriges Winkelhistogramm

Die so berechneten Winkel werden in ein Histogramm übertragen. Die resultierende Verteilung der Winkel zeigt die schwarze Linie in Bild 3.4. Die graue Elipse markiert den Winkel, der in Bild 3.3 exemplarisch hervorgehoben wurde. Die Darstellung ist auf ein Intervall von $[-45°\ 45°[$ um die in grau dargestellte Referenz begrenzt. Dieses bezieht sich gemäß

der obigen Ausführungen auf die Linien, die parallel zur gewünschten Bewegungsrichtung des Roboters verlaufen. Hier wird daher auf Basis der beiden senkrechten Linien in Bild 3.2 die relative Ausrichtung $\Delta\Psi$ zu diesen bestimmt.
Aus dem Versatz zwischen dem Maximum des Winkelhistogramms und der Referenz lässt sich direkt der gesuchte Winkel ablesen. In dem in Bild 3.2 gezeigten Beispiel beträgt der gesuchte Winkel $\Delta\Psi = 5°$. Wendet man das Inverse der ermittelten Rotation auf die Messdaten an, erhält man Bild 3.5. Es ist gut zu erkennen, dass sämtliche Messwerte nun in unmittelbarer Nähe der tatsächlichen Linie liegen. Die verbliebenen Ungenauigkeiten sind auf das Messrauschen des Laserscanners zurückzuführen.

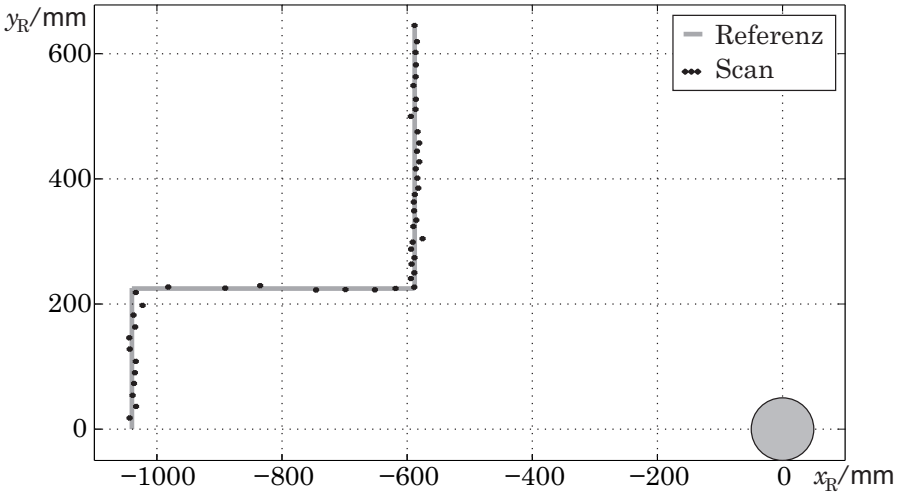



**Bild 3.5:** Ergebnis unter Berücksichtigung der aktuellen Rotation

Da ein einzelnes Beispiel keine belegbare Aussage über die Genauigkeit des Verfahrens ermöglicht, ist dazu eine gesonderte Messung durchgeführt worden. Bei dieser Messung wurde auch das Quantisierungsrauschen berücksichtigt, welches bei der hier verwendeten Auflösung von 1° bei ±0,5° liegt. Der Laserscanner wurde so ausgerichtet, dass seine Blickrichtung um 12,5° gegenüber einer geraden Wand zu seiner Linken verdreht war. Es wurden 1000 Messungen durchgeführt, bei denen jeweils der Winkel $\Delta\Psi$ bestimmt worden ist. Bild 3.6 zeigt die Verteilung der ermittelten Winkel.

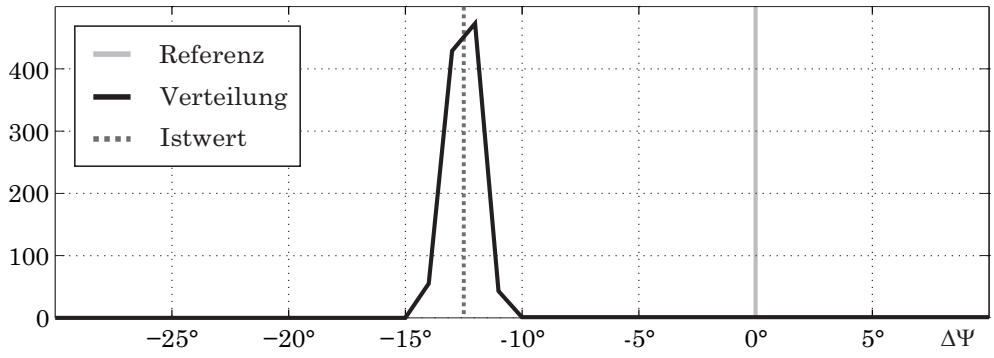



**Bild 3.6:** Verteilung der ermittelten Winkel $\Delta\Psi$

Die in schwarz angegebene Verteilungskurve zeigt ein deutliches Maximum, welches in der Region des durch die senkrechte gestrichelte Linie gekennzeichneten gesuchten Winkels von 12,5° ist. Die senkrechte graue Linie auf Höhe des Nullpunktes markiert analog zu den obigen Abbildungen die Referenz, zu der jeweils der Winkel bestimmt wurde. Aus Bild 3.6 wird deutlich, dass die Genauigkeit des vorgestellten Verfahrens unter Berücksichtigung des Quantisierungsrauschens bei ±1,5° liegt. Die späteren Ausführungen zum implementierten SLAM-Algorithmus werden zeigen, dass diese völlig ausreichend ist.

Dieser Abschnitt zeigt, dass in einer vorwiegend rechtwinkligen Umgebung die Ausrichtung des Roboters ohne die zeitaufwendige Berechnung der Kreuzkorrelation auf Basis von Winkelhistogrammen funktioniert. Im Hinblick auf die später verwendete Hardware ist die damit verbundene geringere Durchlaufzeit des gesamten Algorithmus sehr wichtig.

## 3.2 Von den Rohdaten zur orthogonalen Linie

In diesem Abschnitt werden die ersten drei Schritte des auf Seite 26 in Bild 3.1 dargestellten Schemas des SLAM-Algorithmus beschrieben. Wie eingangs dieses Kapitels erläutert, wird ein Laserscanner als zentraler optischer Sensor verwendet. Bei einer Messung wird der gesamte Sichtbereich mit einer bestimmten Winkelauflösung abgetastet. Damit setzt sich ein Datensatz $P$ aus vielen Einzelmessungen wie folgt zusammen:

$$P = \{p_1,\ p_2,\ \ldots\ p_N\}. \tag{3.6}$$

Die resultierenden Abtastwerte $p_1$ bis $p_N$ repräsentieren die Entfernung zu einem eventuellen Hindernis. Dabei markiert der Index $N$ die maximale Anzahl von Messpunkten. Wird innerhalb der Reichweite des Laserscanners bei einem Abtastpunkt $p_i$ kein Hindernis detektiert, ergibt sich $p_i = 0$. Aufgrund der Abmessungen der später verwendeten Flugplattform und der angegebenen Reichweite des Laserscanners ergibt sich ein Wertebereich, in dem die Messwerte $p_1$ bis $p_N$ theoretisch liegen müssen. Die Roboterabmessungen ergeben dabei die untere Grenze des Intervalls, die Reichweite dementsprechend die Obergrenze. Liegen Messwerte außerhalb des so definierten Gültigkeitsbereichs, werden diese im ersten Schritt des SLAM-Algorithmus eliminiert. Die Anzahl der gültigen Messdaten ergibt sich damit zu

$$P_{\mathrm{v}} = \{p_1,\ p_2,\ \ldots\ p_{N-E}\}. \tag{3.7}$$

Der Index $E$ repräsentiert dabei die Anzahl der verworfenen Messungen. Die gültigen Messwerte werden im Anschluss unter Berücksichtigung der Ausrichtung $\Psi$ des Roboters aus der Polardarstellung in kartesische Koordinaten des roboterfesten Koordinatensystems transformiert. Jeder gültige Abtastwert wird damit durch ein Wertepaar beschrieben. Aus Gleichung 3.7 ergibt sich damit

$$P_{\mathrm{v},xy} = \{(p_{1,x_{\mathrm{R}}},p_{1,y_{\mathrm{R}}}),\ (p_{2,x_{\mathrm{R}}},p_{2,y_{\mathrm{R}}}),\ \ldots\ (p_{N-E,x_{\mathrm{R}}},p_{N-E,y_{\mathrm{R}}})\}. \tag{3.8}$$

Mit dieser Transformation ist der Schritt der „Vorverarbeitung der Messwerte“ abgeschlossen. Im Folgenden wird beschrieben, wie die Ermittlung von Linien innerhalb des in Gleichung 3.8 dargestellten Datensatzes erfolgt.

Zunächst wird der Datensatz in verschiedene Segmente eingeteilt. Ist der Euklidische Abstand zweier aufeinander folgender Messwerte $p_i$ und $p_{i+1}$ größer als der Parameter $q$, wird die Punktemenge unterteilt und die Punkte $p_1$ bis $p_i$ dem Segment $S_1$ zugeordnet. Die Daten $p_{i+1}$ bis $p_{N-E}$

werden auf gleiche Weise überprüft und bei erneuter Überschreitung des Parameters $q$ wird ein weiteres Segment abgespalten. Auf diese Weise lässt sich der Datensatz wie folgt darstellen:

$$P_{\mathrm{v},S} = \{S_1,\ S_2,\ \ldots\ S_M\}. \tag{3.9}$$

Der Index $M$ markiert dabei die maximale Anzahl von gefundenen Segmenten. Die weiteren Erläuterungen zur Liniensuche sollen anhand eines kleinen Beispiels erfolgen. Bild 3.7 zeigt einen im Labor aufgenommenen Datensatz mit zwei Segmenten.

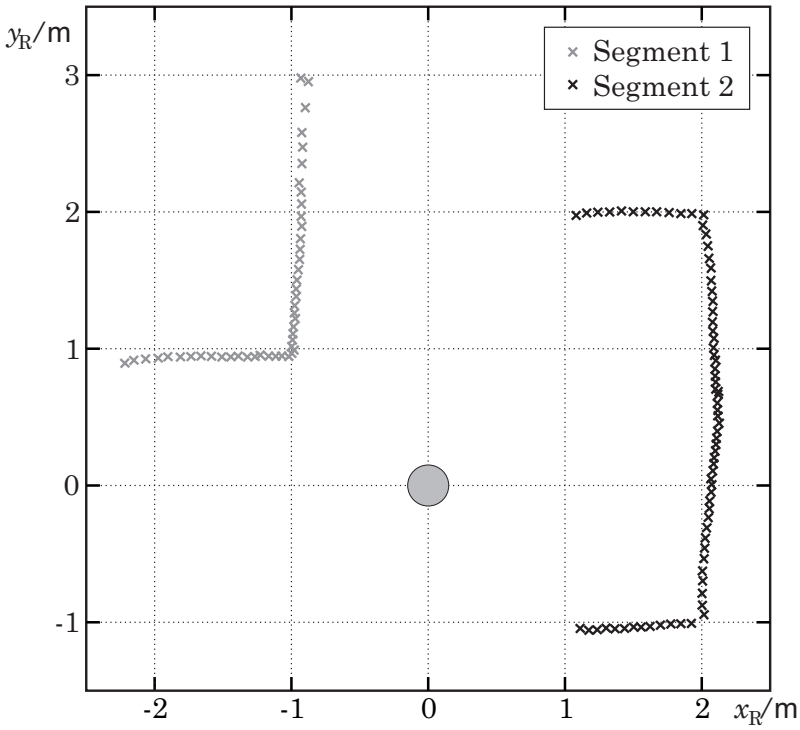



**Bild 3.7:** Beispielhafter Datensatz mit zwei Segmenten $S_1$ und $S_2$

Bei der Segmentierung ist die Wahl der Trennvariable $q$ von besonderer Bedeutung. Wählt man den Abstand zur Bildung eines neuen Segments zu niedrig, enstehen viele kleine Punktcluster. Diese führen dann im weiteren Verlauf des Algorithmus zu vielen einzelnen Linien. Die weiteren Ausführungen werden zeigen, dass somit die Laufzeit steigt und die Gefahr besteht, dass die resultierende Karte unübersichtlich wird. Wählt

man den Parameter $q$ hingegen zu groß, beinhalten die resultierenden Segmente sehr viele Punkte und die anschließende Liniensuche liefert unter Umständen Hindernisse, die in der Realität nicht vorhanden sind. Bild 3.7 zeigt das Beispiel für einen Flur, bei dem rechts und links die Struktur der Wand erfasst wird. Bei hinreichend großem $q$ würden beide Segmente zu einem verschmelzen und die spätere Liniensuche würde eine waagerechte Linie quer über den Flur zur Folge haben. Dieser Umstand würde die Realität nicht richtig wiedergeben und beim Betrachter zu falschen Schlüssen führen. Man muss hier daher einen geeigneten Kompromiss für den Parameter $q$ finden. Die Wahl dieser und aller weiteren Entscheidungsvariablen wird in Zusammenhang mit den Experimenten in Kapitel 4 angegeben.

Die Ermittlung von Linien innerhalb eines Segments erfolgt mit Hilfe des *Split and Merge* Verfahrens nach [18]. Dabei wird ein Segment zunächst so lange geteilt, bis die Punktemengen der resultierenden Untersegmente nur noch einer Linie entsprechen. Im Anschluss werden dann gegebenenfalls offensichtlich zusammengehörende Linien gleicher Richtung zusammengefasst. Dieses Verfahren soll im Folgenden anhand des Segments $S_2$ aus Bild 3.7 eingehend erläutert werden.

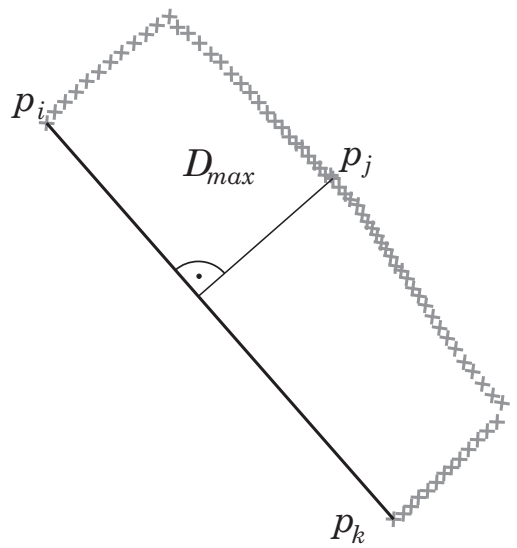
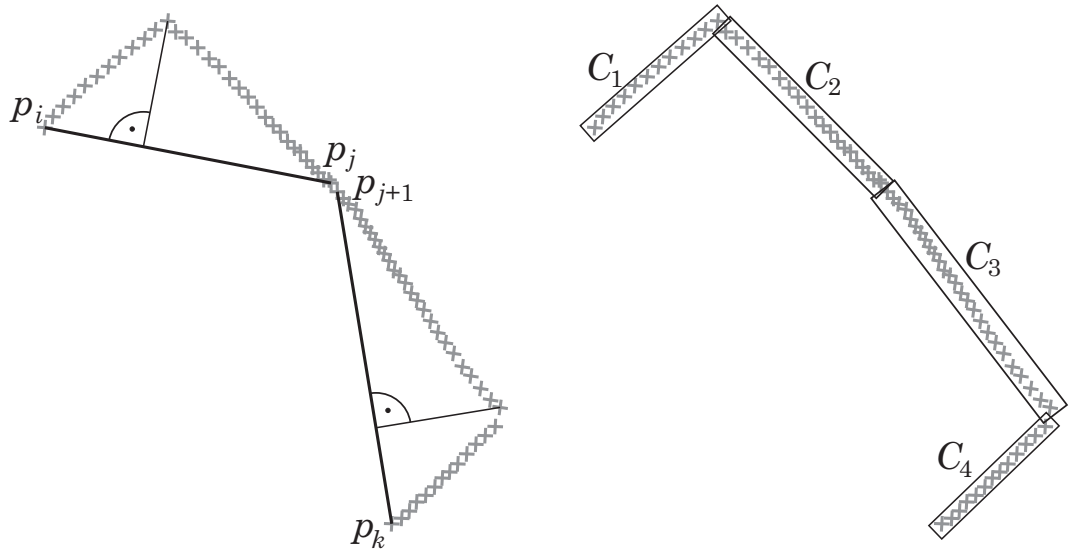



**Bild 3.8:** 1. Aufspaltung des untersuchten Segments

Im ersten Schritt wird eine virtuelle Linie zwischen dem ersten und dem letzten Punkt des untersuchten Segments gebildet. Im Anschluss wird von jedem Punkt $p_i$ des Segments das Lot zu dieser Verbindungslinie gefällt und der dazugehörige Abstand $D$ bestimmt. Bild 3.8 zeigt das prinzipielle Vorgehen am verwendeten Beispiel.
In diesem Fall hat der Punkt $p_j$ den maximalen senkrechten Abstand $D_{\max}$ zur virtuellen Gerade zwischen dem Punkt $p_i$ und dem Punkt $p_k$, der im vorliegenden Fall dem Punkt $p_{N-E}$ entspricht. Gilt $D_{\max} > D_{\mathrm{sm}}$, wird das untersuchte Segment so geteilt, dass der eine Bereich von $p_i$ bis $p_j$ und der andere von $p_{j+1}$ bis $p_k$ läuft. Die Variable $D_{\mathrm{sm}}$ gibt dabei den maximal zulässigen Abstand an. Nach dem gleichen Prinzip werden die beiden Untersegmente wieder untersucht. Der Algorithmus bricht ab, wenn für den betrachteten Bereich $D_{\max} \leq D_{\mathrm{sm}}$ gilt. Im hier vorliegenden Fall ergeben sich die in Bild 3.9 gezeigten vier Untersegmente (Punktcluster) $C_1$ bis $C_4$.




(a) 2. Aufspaltung des Segments (b) Resultierende Punktcluster

**Bild 3.9:** Weitere Aufspaltung mit resultierenden Untersegmenten $C_1$ bis $C_4$

Aus den Erläuterungen zu diesem Verfahren wird deutlich, dass die Wahl der Abstandsvariable $D_{\mathrm{sm}}$ von entscheidender Bedeutung ist. Auch beim folgenden Schritt der Zusammenfassung von Linien mit annähernd gleicher Richtung ist sie relevant. Daher wird erst zu einem späteren

Zeitpunkt auf den Einfluss dieser Variable auf den SLAM-Algorithmus eingegangen. Beim Zusammenführen von Linien, dem *merging*, wird eine virtuelle Linie zwischen dem ersten Punkt des ersten Untersegments und dem letzten Punkt des folgenden Untersegments gezogen. Im Anschluss wird wieder der senkrechte Abstand der einzelnen Punkte zu dieser Linie ermittelt. Gilt $D_{\max} \leq D_{\mathrm{sm}}$, werden die betrachteten Bereiche zusammengeführt. Aus Bild 3.10 wird deutlich, dass dies im vorliegenden Beispiel auf die Untersegmente $C_2$ und $C_3$ zutrifft.

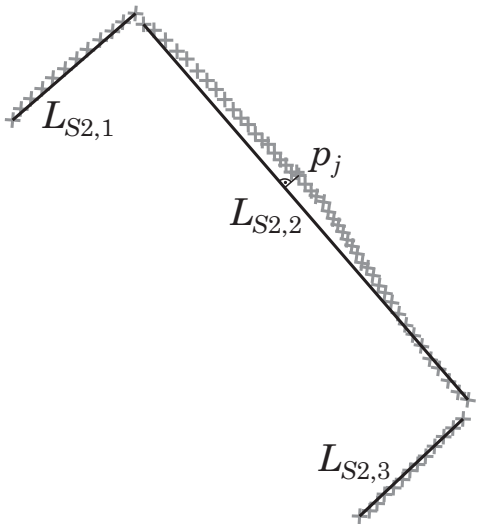



**Bild 3.10:** Resultierende Linien

Damit erhält man in diesem Fall drei Linien, die sich aus der Verbindung des ersten und letzten Punktes des jeweiligen Untersegments ergeben. Diese sind mit dem Parameter $L$ bezeichnet. Die zugehörigen Indizes bezeichnen das Ausgangssegment $S_2$ und die laufende Nummer der Linie.

Aus den Betrachtungen wird deutlich, dass die Anzahl der ermittelten Linien aus einem Segment mit der Wahl der Entscheidungsvariable $D_{\mathrm{sm}}$ zusammenhängt. Wählt man den Wert zu klein, ergeben sich viele vergleichsweise kurze Abschnitte. Eine große Anzahl von Linien wirkt sich negativ auf die Laufzeit des Algorithmus aus und führt wie oben bereits erwähnt unter Umständen zu einer sehr unübersichtlichen Karte. Wählt man den Wert zu groß, dann wird die Umgebung nur sehr grob

approximiert und der geringe Informationsgehalt der resultierenden Karte erschwert die Navigation des Roboters und nicht zuletzt die Interpretation durch den Betrachter. Bei der Wahl von $D_{\mathrm{sm}}$ geht es daher darum, einen geeigneten Kompromiss in Bezug auf Laufzeit und der gewünschten Genauigkeit der Karte zu finden.

Im dritten Schritt des auf Seite 26 in Bild 3.1 dargestellten Schemas werden die in der beschriebenen Weise aus den Rohdaten ermittelten Linien auf Orthogonalität geprüft und gegebenenfalls in ein entsprechendes achsenparalleles Raster übertragen. Bild 3.11 zeigt für das hier verwendete Beispiel die aus den Messwerten ermittelten Linien.

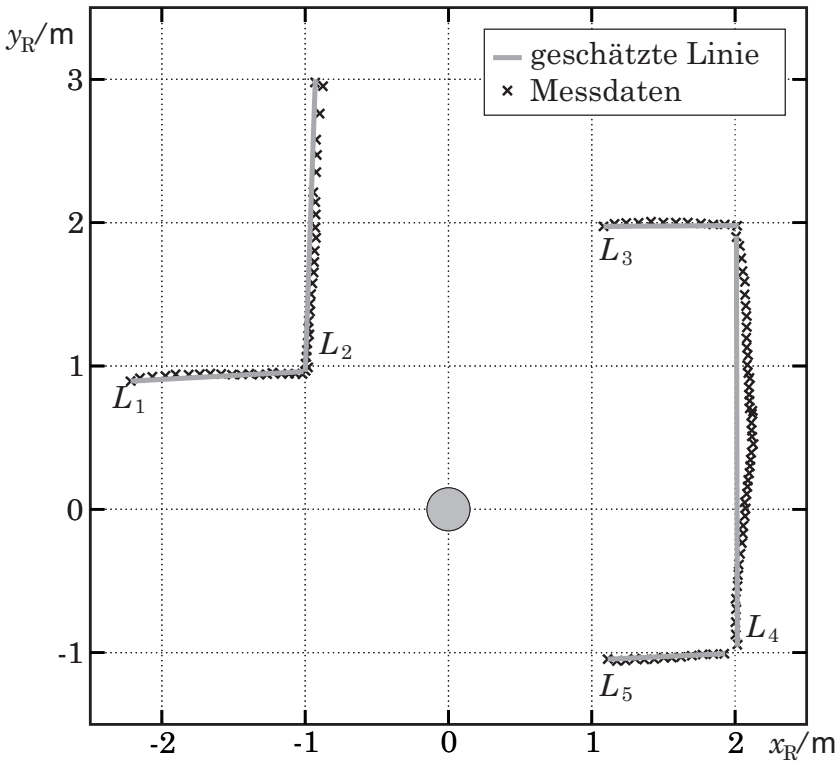



**Bild 3.11:** Vergleich von Messdaten mit geschätzten Linien

Die Betrachtung des Bildes zeigt, dass die in grau dargestellten Linien leicht von einem rechtwinkligen Raster abweichen. Würde man also das bisher erzielte Ergebnis nur auf Orthogonalität überprüfen, hätte dies unter Umständen zur Folge, dass alle Linien verworfen werden würden, da sie nicht in das entsprechende Raster passen. Ein solcher Informati-

onsverlust führt zum Scheitern des Verfahrens. Daher wird ein Toleranzbereich $\phi_{\mathrm{T}}$ eingeführt, in dem sich die Abweichungen zum rechtwinkligen Raster bewegen dürfen. Die Winkel der aus den Messwerten extrahierten Linien ergeben sich aus der Hesseschen Normalform, bei der eine Gerade durch ihren Abstand und den eingeschlossenen Winkel $\Delta\phi$ zwischen ihrem Lot durch den Ursprung und der $x$ - Achse beschrieben wird [8]. Im Verlauf der Arbeit hat sich gezeigt, dass es nicht sinnvoll ist, die Winkeltoleranz $\phi_{\mathrm{T}}$ mit einem konstanten Wert zu belegen. Wählt man sie zu groß, hat das eine ungenaue Karte zur Folge, da die Wahrscheinlichkeit steigt, dass Konturen, die tatsächlich keine rechtwinklige Struktur haben, in ein solches Raster gepresst werden. Wählt man den Wert für $\phi_{\mathrm{T}}$ hingegen zu klein, tritt ein Problem auf, welches analog zum oben genannten bei der Überprüfung auf Orthogonalität ist. Durch die beschriebene Bestimmung der Linien, bei der unabhängig von ihrer jeweiligen Länge die Zugehörigkeit der Punkte mit Hilfe der Abstandsvariablen $D_{\mathrm{sm}}$ überprüft wurde, können bei vergleichsweise kurzen Linien größere Winkelabweichungen zum tatsächlichen Verlauf auftreten.

**Bild 3.12:** Linie mit zulässigem Toleranzband der zugehörigen Messwerte

Bild 3.12 zeigt ein Beispiel mit einer Linie und den zugehörigen Messpunkten, die durch Kreuze gekennzeichnet sind. Alle liegen in dem in hellgrau dargestellten Toleranzband. Die mögliche Winkelabweichung $\Delta\phi$ ergibt sich aus dem Winkel zwischen einer Diagonalen durch das Toleranzband und der eingezeichneten waagerechten Linie. Bei konstantem $D_{\mathrm{sm}}$ hängt die maximal mögliche Winkelabweichung daher von der Länge der Linie ab. Je kürzer die Linie ist, desto größere Winkelabweichungen sind möglich. Da die Länge einer Linie bei näherungsweise gleichem Euklidischen Abstand der Messwerte von der Anzahl der zugehörigen Punkte abhängt, ist es naheliegend, die zulässige Toleranz an die Anzahl der zur jeweiligen Linie gehörenden Messpunkte zu knüpfen.

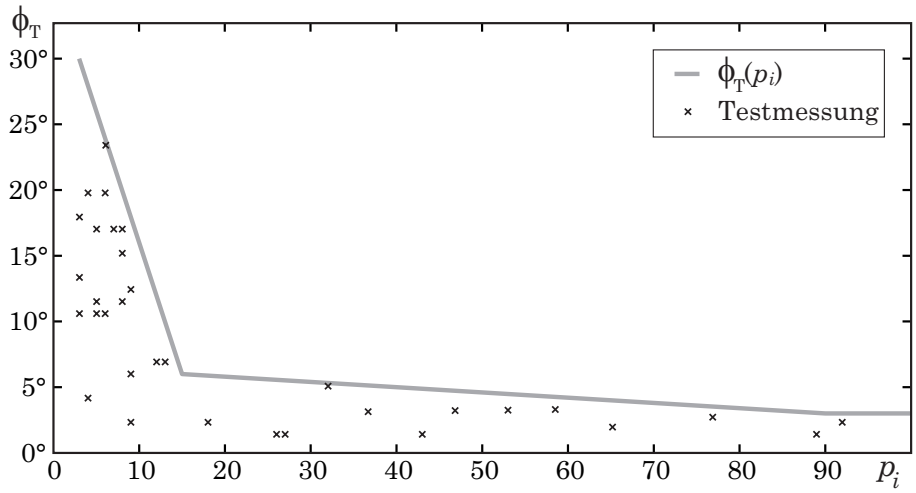



**Bild 3.13:** Winkeltoleranz $\phi_T$ in Abhängigkeit der Punktanzahl

Bild 3.13 zeigt einen empirisch ermittelten Funktionsverlauf der im weiteren Verlauf der Arbeit zur Anwendung kommt. Die mit schwarzen Kreuzen eingezeichneten Messergebnisse basieren auf Testmessungen in einer strikt rechtwinkligen Umgebung. Es ist deutlich zu erkennen, dass bei wenigen Stützstellen die Ungenauigkeit der ermittelten Winkel $\phi$ signifikant zunimmt. Der in grau eingezeichnete Kurvenverlauf trägt dieser Eigenschaft Rechnung und beschreibt den Zusammenhang zwischen der Anzahl der Stützstellen einer ermittelten Linie und der zulässigen Winkeltoleranz $\phi_T$.
Liegt der Winkel der betrachteten Linie innerhalb der Toleranz, gilt also $\Delta\phi \leq \phi_T(p_i)$, wird sie um ihren Mittelpunkt so gedreht, dass sie in einem rechtwinkligen Raster abgebildet werden kann. Die jeweiligen Start- und Endpunkte werden entsprechend angepasst. Gilt $\Delta\phi > \phi_T(p_i)$, wird die entsprechende Linie verworfen und dementsprechend nicht in der späteren Karte auftauchen. Wendet man dieses Verfahren auf das hier betrachtete Beispiel an, werden alle Linien nach einer entsprechenden Anpassung zur weiteren Verwendung übernommen. Bild 3.14 zeigt das zugehörige Resultat.

Damit ist gezeigt, wie aus den Rohdaten des Laserscanners orthogonale Linien ermittelt werden können. Das hier erreichte Zwischenziel wird in

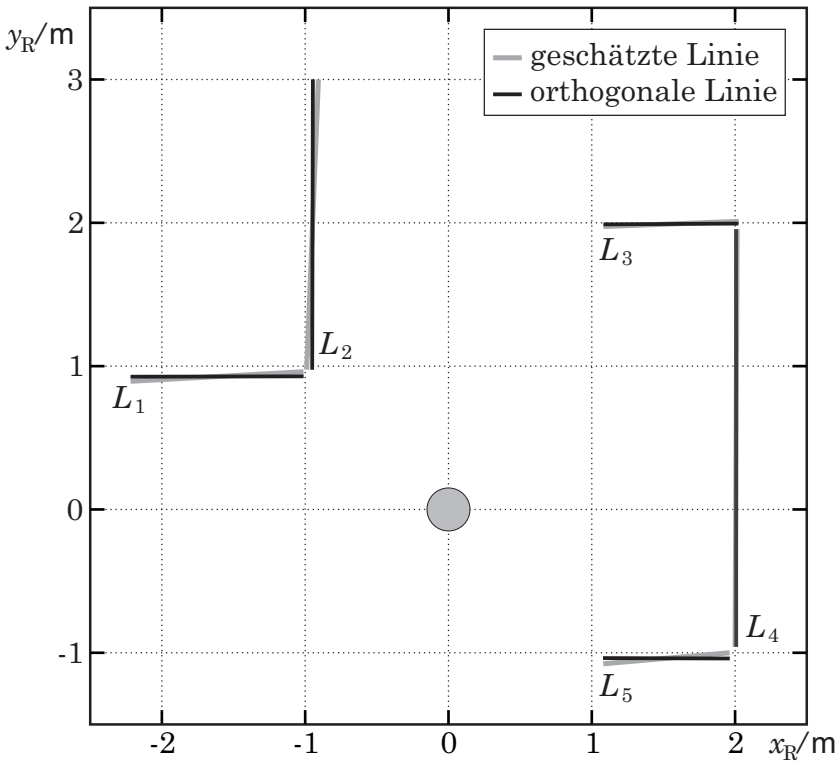



**Bild 3.14:** Linien nach der Orthogonalisierung

der Literatur rund um die SLAM-Algorithmen oft als lokale Karte bezeichnet, da sie die unmittelbare Umgebung des Roboters abbildet. Im folgenden Abschnitt wird gezeigt, wie die einzelnen lokalen Karten zu einer globalen Karte zusammengeführt werden.

## 3.3 Aktualisierung und Optimierung der globalen Karte

Zunächst liegt der Fokus in diesem Abschnitt auf der Eingliederung der aus den aktuellen Scandaten extrahierten lokalen in die bis dahin bekannte globale Karte. Da sich der Roboter im Allgemeinen durch den Raum bewegt, muss die Verschiebung zwischen den bekannten und den neuen Linien ermittelt werden, um die aktuelle Position bestimmen und die neuen Informationen richtig übernehmen und speichern zu können. Aufgrund der Voraussetzung der orthogonalen Linien und der Kenntnis der Ausrichtung $\Psi$ des Roboters sind alle zu betrachtenden Linien parallel zu den Achsen des Weltkoordinatensystems. Die Ermittlung der Verschiebung entlang der $x_E$- und der $y_E$-Achse kann daher getrennt

voneinander erfolgen. Grundsätzlich werden bei diesem Verfahren die lokale und die globale Karte miteinander verglichen, damit möglichst viele der bisher gesammelten Informationen in die Abschätzung der jeweiligen Verschiebung mit eingehen. Vergleicht man lediglich zwei aufeinanderfolgende lokale Karten miteinander, ist die Informationsbasis deutlich kleiner und der gesamte Algorithmus wird deutlich anfälliger gegenüber Messfehlern. Der Rechenaufwand wird durch den Vergleich mit der globalen Karte zwar leicht erhöht, die Qualität des Ergebnisses jedoch deutlich gesteigert. Die Erläuterung des Verfahrens erfolgt anhand eines Beispiels für eine Verschiebung in $x_{\mathrm{E}}$ - Richtung. Für die zweite Achse ist das Verfahren analog anzuwenden.
Bild 3.15 zeigt aus Gründen der Übersichtlichkeit nur Linien, die parallel zur $y_{\mathrm{E}}$ - Achse verlaufen. Der relevante Bereich der globalen Karte ist in grau, die zu integrierenden Daten in schwarz angegeben. Die hier verwendeten Daten sind ein Auszug einer Messung eines Flures. Die Einbuchtungen auf der linken und rechten Seite sind geschlossene Türen zu vom Flur abzweigenden Zimmern.

Zu Beginn müssen zunächst alle sinnvollen Verschiebungen $l$ zwischen den Linien beider Karten ermittelt werden. Um den gesamten SLAM-Algorithmus zu beschleunigen, empfiehlt es sich, eine maximal zulässige Verschiebung $l_{\max}$ zu definieren. Diese Obergrenze kann auf Basis der kommandierten Geschwindigkeit des Roboters und der für den Laserscanner gewählten Abtastzeit abgeschätzt werden. Im Verlauf dieser Arbeit hat sich gezeigt, dass hier eine großzügige Wahl sinnvoll ist, da im Flug später Störungen wie Luftströme dazu führen können, dass die kommandierte Geschwindigkeit überschritten wird. Wählt man den Wert jedoch zu groß, dann ist der durch die Einführung dieses Parameters gewünschte Vorteil hinfällig, wie die Erläuterungen zu dem obigen Beispiel in Bild 3.15 zeigen werden.
Eine mögliche Verschiebung wird genau dann als sinnvoll betrachtet, wenn der Abstand $l$ zweier sich überlappendener Linien aus lokaler und globaler Karte unterhalb der eingeführten Schwelle $l_{\max}$ ist. Der Begriff Überlappung wird in diesem Kontext verwendet, wenn zwei Linien einen gemeinsamen $y$ - Achsenabschnitt haben. Betrachtet man die Linie $L_{\mathrm{l},1}$

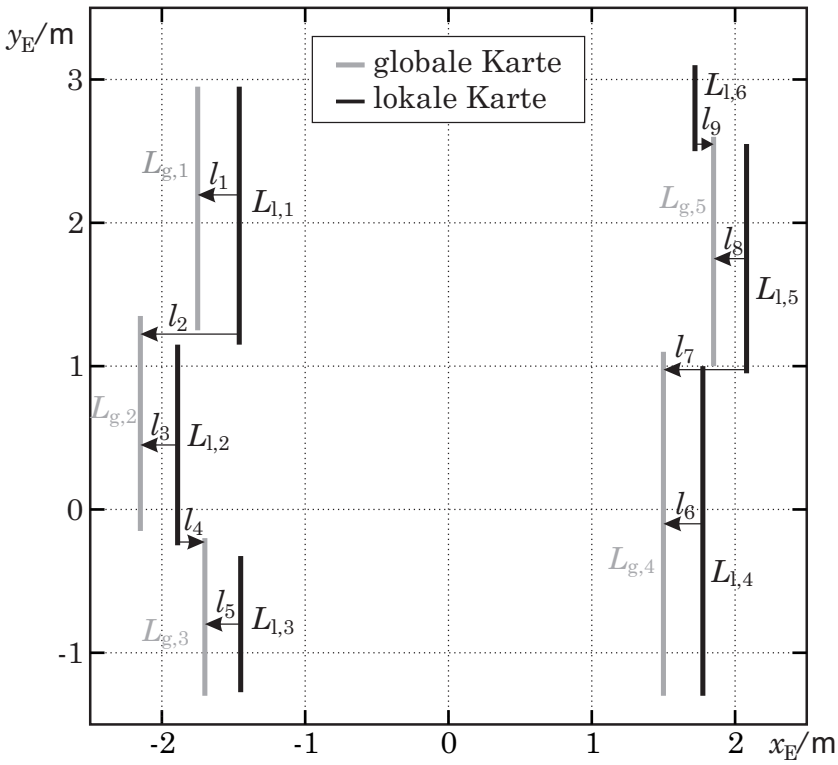



**Bild 3.15:** Ermittlung aller zulässigen Verschiebungen

im oberen linken Bereich von Bild 3.15, so gibt es grundsätzlich drei Überlappungen dieser zu den in grau eingezeichneten bereits bekannten Linien ($L_{g,1}, L_{g,2}, L_{g,5}$). Der Abstand $l$ zu den beiden erst genannten ist unterhalb von $l_{max}$ und die Werte $l_1$ und $l_2$ werden für die weitere Berechnung verwendet. Der Abstand zwischen den Linien $L_{l,1}$ und $L_{g,5}$ leistet hier offensichtlich keinen sinnvollen Beitrag zur Ermittlung der Verschiebung des Roboters. Dieses konkrete Beispiel verdeutlicht, dass die Wahl von $l_{max}$ nicht nur auf Basis der Geschwindigkeit und der Abtastrate gewählt werden sollte, sondern dabei auch die Abmessungen der zu erwartenden Umgebung eine Rolle spielen. Da in engeren Umgebungen aber üblicherweise die Geschwindigkeit zurückgenommen wird, geht die Struktur der Umgebung indirekt in die Wahl von $l_{max}$ ein.
In Bild 3.15 sind alle sinnvollen Verschiebungen durch kleine Pfeile eingetragen. Die zugehörigen Parameter $l_1$ bis $l_9$ bilden die Basis für die Bestimmung der aktuellen Position und die Kartenaktualisierung. In diesem Schritt wird auch der Tatsache Rechnung getragen, dass die lokale Karte eine bisher unbekannte Linie $L_{l,6}$ beinhaltet.

In Bild 3.16 tauchen die ermittelten Verschiebungen wieder auf. Sie sind unter Berücksichtigung des jeweiligen Vorzeichens und des Betrags im oberen Bereich der Abbildung aufgetragen und mit der zugehörigen laufenden Nummer gekennzeichnet. Bei genauerer Betrachtung ist zu erkennen, dass es einen relativ engen Bereich gibt, in dem vergleichsweise viele dieser möglichen Verschiebungen liegen. Im Folgenden wird erläutert, wie man diese Häufung bestimmen und daraus die tatsächliche Verschiebung abschätzen kann.

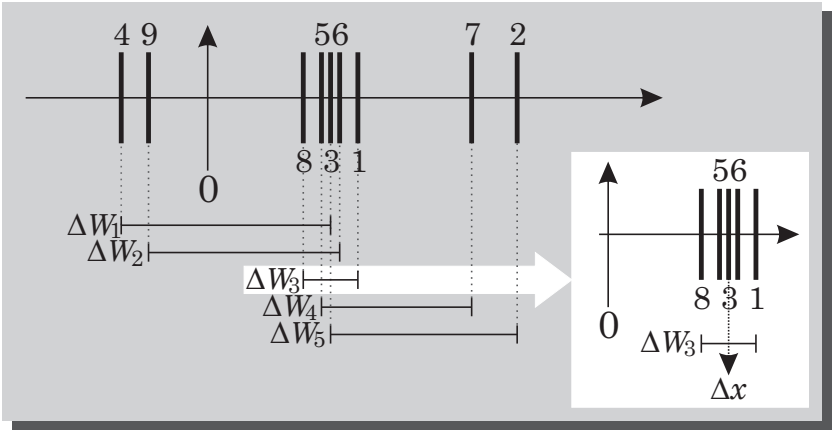



**Bild 3.16:** Synthese der wahrscheinlichen Verschiebung $\Delta x$

Für die Bestimmung der gesuchten Häufung ist die Anzahl der lokalen Linien von Bedeutung, zu denen mindestens eine sinnvolle Verschiebung ermittelt wurde. Dieser Parameter wird mit der Variable $L_{\mathrm{l,N}}$ bezeichnet. In dem hier gezeigten Beispiel ist dies für alle sechs Linien der in Bild 3.15 in schwarz dargestellten lokalen Karte der Fall. Daher gilt hier $L_{\mathrm{l,N}} = 6$. Aufgrund dieses Parameters ergibt die Größe eines Suchfensters $W$ (engl. *window*)

$$W = L_{\mathrm{l,N}} - 1. \tag{3.10}$$

Durch die Subtraktion der Konstante 1 trägt man dem Umstand Rechnung, dass in der lokalen Karte Linien auftreten, die bisher noch nicht bekannt waren. Im Verlauf dieser Arbeit hat sich diese Wahl der Konstante als gut herausgestellt. Abhängig ist die Wahl vor allem von

der gewünschten Geschwindigkeit des Roboters. Je schneller sich dieser bewegt, desto größer ist die Wahrscheinlichkeit, dass bisher unbekannte Strukturen in der lokalen Karte auftauchen. Daher ist der Offset mit steigender Geschwindigkeit dementsprechend größer zu wählen.
Das gewählte Fenster wird von links nach rechts so über die aufgetragenen Abstände $l_1$ bis $l_9$ gelegt, dass immer die Differenz $\Delta W_i$ zwischen dem letzten und dem ersten Wert des jeweils betrachteten Intervalls gebildet wird. Im hier vorliegenden Beispiel gilt gemäß Gleichung (3.10) $W = 5$. Bei den neun ermittelten Abständen lassen sich demnach fünf Intervalle bilden. Die erste Differenzbildung erfolgt gemäß Bild 3.16 zwischen $l_3$ und $l_4$. Der ermittelte Wert $\Delta W_1$ wird gespeichert und das Fenster im Anschluss um eins nach rechts verschoben. $\Delta W_2$ ergibt sich damit aus der Differenz von $l_6$ und $l_9$. Nach diesem Schema werden alle möglichen Intervalle ausgewertet und die jeweiligen Ergebnisse $\Delta W_i$ gespeichert. Es ist offensichtlich, dass das Minimum aus dem Intervall $[\Delta W_1, \Delta W_2, \ldots, \Delta W_W]$ die gesuchte Häufung markiert.
Im weiß hinterlegten Bereich von Bild 3.16 sind die zum Fenster $W_3$ gehörigen Abstände noch einmal aufgetragen. Betrachtet man diese in Bild 3.15, so zeigt sich, dass diese fünf Werte die Abstände der bereits bekannten Linien aus lokaler und globaler Karte sind. Im Idealfall wären alle diese Abstände gleich groß. Aufgrund von Messfehlern und des hier beschriebene Schätzverfahrens zur Ermittlung der Linien ist dies jedoch im Allgemeinen nicht der Fall. Um die gesuchte Verschiebung $\Delta x$ endgültig abzuschätzen, wird daher der arithmetische Mittelwert gebildet.

Stellt man lokale und globale Karte unter Berücksichtigung der geschätzten Verschiebung $\Delta x$ erneut in einer Abbildung dar, ergibt sich Bild 3.17. Hier ist klar ersichtlich, dass die jeweiligen Linien in Bezug auf die $x_{\mathrm{E}}$ - Koordinate eine hohe Übereinstimmung aufweisen.
Damit ist gezeigt, dass das vorgestellte Verfahren zur Lokalisierung und anschließender Aktualisierung der globalen Karte funktionsfähig ist. Der Fall, dass bisher unbekannte Linien auftauchen, wurde in dem betrachteten Beispiel ebenfalls berücksichtigt.

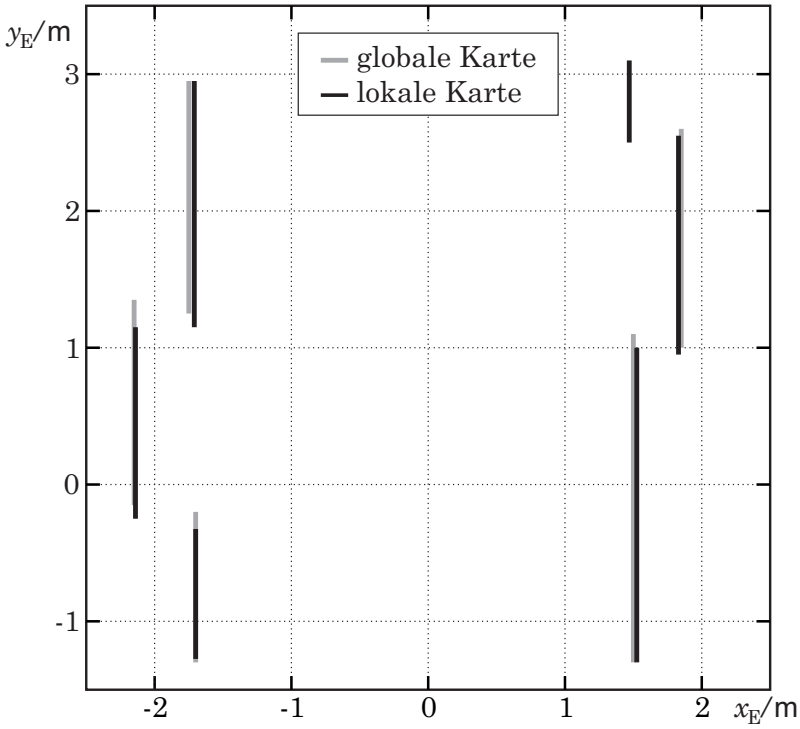



**Bild 3.17:** Validierung der ermittelten Verschiebung $\Delta x$

Um die Verschiebung $\Delta y$ zu bestimmen, kann das Verfahren analog angewendet werden. Die Reihenfolge, in der die Verschiebung entlang der beiden Achsen ermittelt werden, ist grundsätzlich beliebig. Im Falle von gewissen Vorkenntnissen über die zu erkundene Umgebung wäre es sinnvoll, die Reihenfolge so anzupassen, dass zunächst die Achse betrachtet wird, für die die Informationsbasis größer ist. Im Rahmen dieser Arbeit wird jedoch von unbekannten Innenräumen ausgegangen und zunächst $\Delta x$ und dann $\Delta y$ bestimmt. Diese Reihenfolge ist willkürlich gewählt. Sie hat sich in den später vorgestellten Szenarien jedoch nie als nachteilig erwiesen.

Nachdem durch das gezeigte Verfahren die aktuelle Position des Roboters bestimmt wurde und die aktuellen Daten erfolgreich in die globale Karte übernommen wurden, erfolgt ein abschließender Optimierungsschritt, um die Gesamtanzahl der Linien zu minimieren. Dieser Schritt ist aus zwei Gründen wichtig. Zum einen wird dadurch der Anstieg der Zykluszeit des Algorithmus mit fortlaufender Dauer verringert und zum anderen bleibt die resultierende Karte für den Betrachter übersichtlich.

Im Rahmen dieser Optimierung werden die Linien zusammengefasst, welche im Auge des Betrachters ohnehin zusammen gehören. Des Weiteren werden Linien gefluchtet. Dies führt dazu, dass die Abschnitte vor und hinter einem Türsturz oder einem abzweigenden Gang auf gleicher Höhe liegen. Der zuletzt beschriebene Schritt wirkt sich nicht auf die Laufzeit, sondern zu großen Teilen nur auf die Übersichtlichkeit der Karte aus. Zunächst wird daher die mit Blick auf die Laufzeit wichtigere Zusammenfassung von Linien beschrieben.
Zwei Linien sollen immer dann zusammengefasst werden, wenn sie parallel verlaufen, eine Überlappung aufweisen und einen geringen Abstand zueinander haben. Aus dieser Forderung leitet sich ab, dass der maximal zulässige Abstand $l_{\text{merge}}$ definiert werden muss. Anhand von Bild 3.17 kann man erkennen, worauf es bei der Wahl dieser Variablen ankommt. Wählt man sie zu groß, dann würden die Konturen verschwinden, die im vorliegenden Beispiel durch Türstürze entstehen, und man hätte links und rechts eine durchgehende Linie. Eine derart drastische Vereinfachung der Karte ist dem Verfahren nicht zuträglich, da so zur Bestimmung folgender Verschiebung deutlich weniger Marken vorhanden wären. Wählt man den Wert hingegen zu klein, bleiben sehr viele doppelte Linien in der globalen Karte erhalten. Dies führt, wie oben bereits erwähnt, zu einer Laufzeitverlängerung. Zudem wird dadurch unnötig Speicherplatz belegt und die Karte wird zunehmend unübersichtlich. Eine geeignete Wahl des Parameters $l_{\text{merge}}$ wird ebenfalls in Kapitel 4.3 angegeben.

Die Zusammenfassung beider Linien basiert auf einem gewichteten Mittelwert. Ausschlaggebend ist hier die Anzahl der jeweiligen Stützstellen der betrachteten Linien. Je mehr Messpunkte die Basis einer Linie bilden, desto stärker geht der entsprechende Achsenabschnitt in die Mittelwertbildung ein. Dies führt unter anderem dazu, dass lange, durch viele Stützstellen ermittelte Linien nur gering durch einen kurzen eventuell fehlerbehafteten Abschnitt beeinflusst werden. Aufgrund dieser Tatsache führt eine einzelne fehlerhafte Messung nicht zum Scheitern des Verfahrens.

Die Bedingungen für das Fluchten sind ähnlich. Zwei Linien sollen genau dann in Flucht gebracht werden, wenn sie die selbe Richtung haben, keine Überlappung aufweisen und nur einen geringen Unterschied im Achsenabschnitt senkrecht zu ihrer Laufrichtung haben. Die hierfür erforderliche Abstandsvariable $l_{line}$ muss aus den gleichen Grund wie $l_{\text{merge}}$ mit Bedacht gewählt werden. Sind die Kriterien für eine Fluchtung erfüllt, geschieht dies ebenfalls mit Hilfe des gewichteten Mittelwerts. Das Motiv hierfür wurde ebenfalls oben erläutert.

Die eben beschriebene Optimierung der Karte markiert gemäß Bild 3.1 auf Seite 26 den letzten der fünf Bestandteile des SLAM-Algorithmus. Damit sind alle erforderlichen Schritte dargestellt. Die Validierung des gesamten Verfahrens mit Ausnahme der eingangs dieses Kapitels erläuterten Winkelabschätzung erfolgt im nächsten Abschnitt.

## 3.4 Validierung durch Simulation

In diesem Abschnitt erfolgt die Validierung des in diesem Kapitel vorgestellten SLAM-Algorithmus durch eine Simulation. Dabei wurde mit Hilfe der Software Matlab/Simulink® der in Bild 3.18 in grau dargestellte fiktive Flurabschnitt mit einer konstanten Geschwindigkeit von 1 m/s durchquert.

Die gewählte Umgebung ist strikt rechtwinklig strukturiert und die Ausrichtung des Roboters ist während der Simulation konstant parallel zur $y_{\text{E}}$ - Achse. Die verwendete Abtastrate liegt bei 1 Hz und die Winkelauflösung des Laserscanners wurde bei einer Reichweite von 15 m auf 1° gesetzt. Wie das später verwendete Gerät deckt auch der simulierte Sensor einen Sichtbereich von 270° ab, wobei der Mittelpunkt des zugehörigen Messintervalls mit der Bewegungsrichtung des Roboters zusammenfällt. Aufgrund der idealen Voraussetzungen in der Simulationsumgebung wurden die in den vorigen Abschnitten benannten Entscheidungsvariablen für die Simulation großzügig gewählt. Die Übersicht der Wahl dieser Konstanten bei den realen Experimenten ist im Abschnitt 4.3 angegeben.

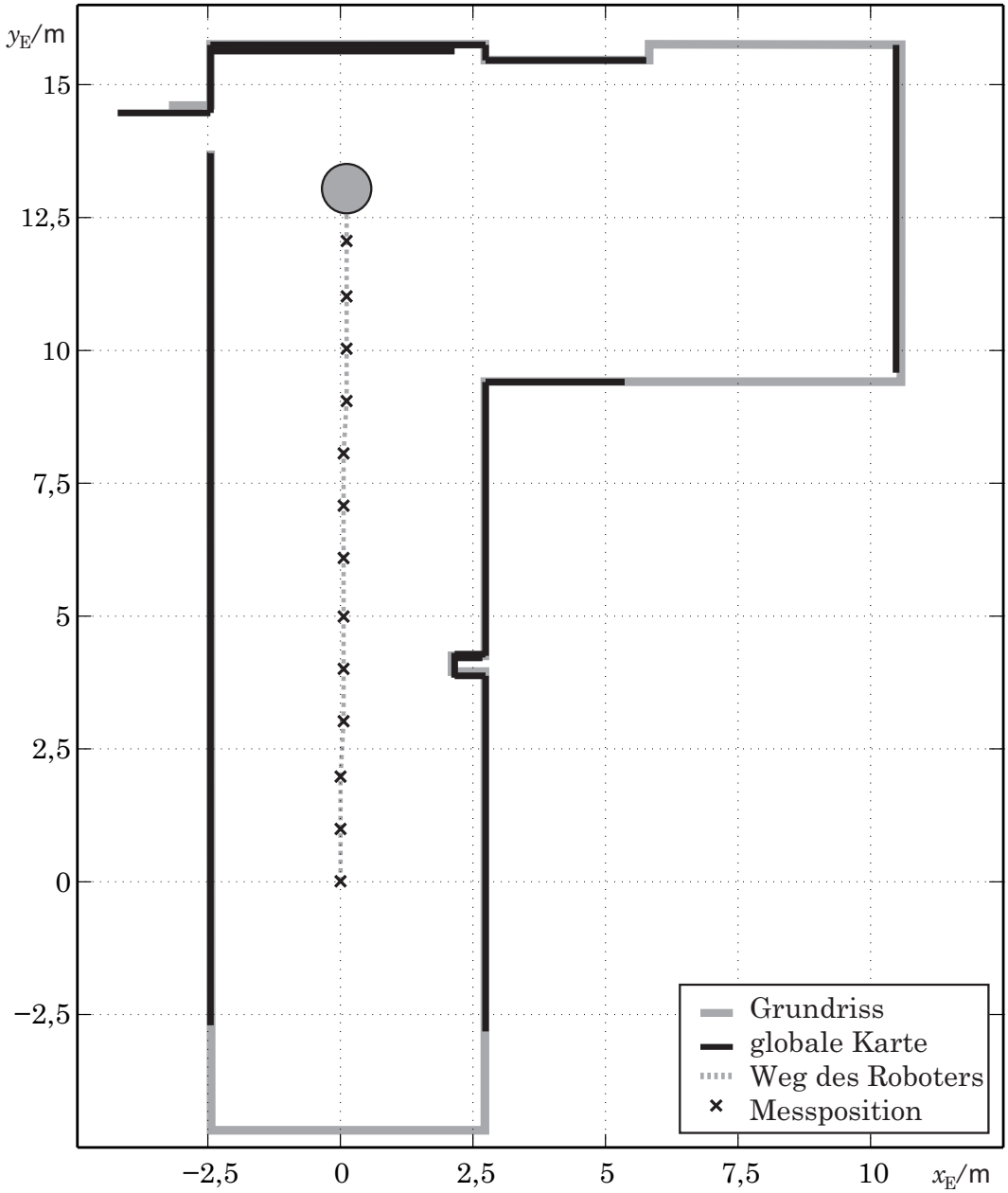
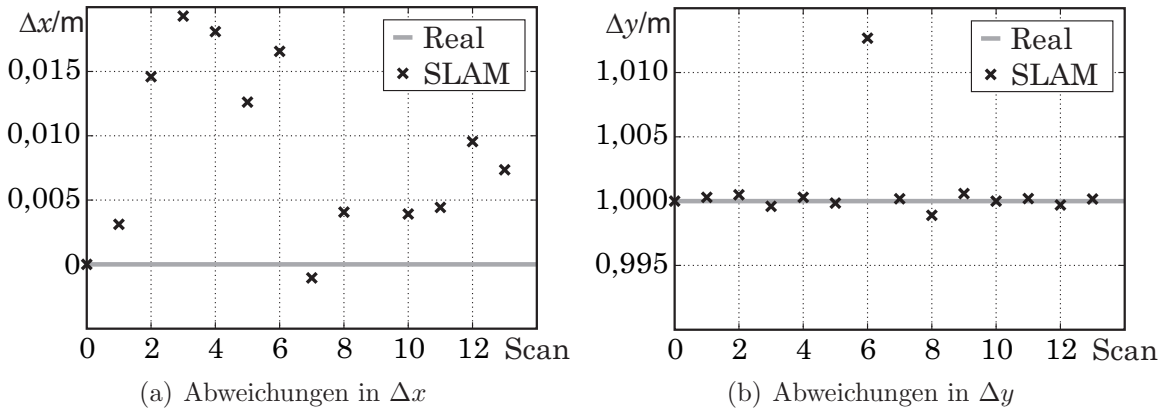



**Bild 3.18:** Simulationsergebnis des SLAM-Algorithmus

Bild 3.18 zeigt das Ergebnis der Simulation. Der Roboter startet im Ursprung des Koordinatensystems. Die durch die Messpunkte ermittelte Position des Roboters ist mit schwarzen Kreuzen markiert. Die in der Simulation bestimmte globale Karte ist durch die schwarzen Linien dargestellt. Es wird deutlich, dass diese die Umgebung sehr gut wiedergeben. Der Grund für die leichten Abweichungen liegt in den

vielen durchgeführten Rundungen während der Berechnung innerhalb des SLAM-Algorithmus und der Simulation des Laserscanners. So wird in Anlehnung an die späteren Experimente die Auflösung der Karte auf 0,01 m gesetzt. Analog dazu werden alle auftretenden Werte gerundet und weitere Nachkommastellen vernachlässigt, die beispielsweise bei der Berechnung der trigonometrischen Funktionen auftreten können.

In Bild 3.19 sind die während der Simulation berechneten Verschiebungen $\Delta x$ und $\Delta y$ gezeigt. Sie sind jeweils über der zugehörigen Scannummer aufgetragen. Es wird deutlich, dass die Abweichungen zum tatsächlichen Wert im Bereich von wenigen Millimetern liegen. Bei Betrachtung der Verschiebung $\Delta y$ wird ein Ausreißer deutlich. Dieser ist auf die erstmalige Detektion der oberen horizontal verlaufenden Wand in Bild 3.18 zurückzuführen. Insgesamt bestätigt Bild 3.19 das gute Ergebnis.




(a) Abweichungen in $\Delta x$ (b) Abweichungen in $\Delta y$

**Bild 3.19:** Abweichung zwischen idealer und berechneter Verschiebung

Dieser Abschnitt zeigt, dass der vorgestellte SLAM-Algorithmus unter idealen Bedingungen funktionsfähig ist. Im Rahmen eines ersten Experiments werden in Kapitel 4 die Entscheidungsvariablen so angepasst, dass der Algorithmus eine möglichst hohe Robustheit gegenüber den in der Realität auftretenden Störungen aufweist.

# 4 Experimentelle Ergebnisse

Die Experimente zur Validierung des im vorangegangenen Kapitel vorgestellten SLAM-Algorithmus werden mit einem Quadrotor durchgeführt, der im ersten Teil dieses Kapitels vorgestellt wird. Im Anschluss werden die weiteren, zur Durchführung der Experimente verwendeten Hardwarekomponenten beschrieben. Anhand von Ergebnissen verschiedener Flugexperimente wird schließlich die Funktionalität des gesamten Systems validiert. Teilweise wurden hier gezeigte Resultate bereits im Rahmen eigener Konferenzbeiträge [2, 3] veröffentlicht. Es ist ein vorbereitender Versuch dargestellt, in dem der Fokus zum einen auf dem SLAM-Algorithmus und der geeigneten Wahl der Entscheidungsvariablen liegt. Zum anderen werden daraus Randbedingungen für die zum autonomen Flug erforderlichen Regler abgeleitet. Im letzten Abschnitt wird anhand mehrerer Szenarien gezeigt, dass mit der gewählten Kombination aus Navigationsstrategie, Reglern und SLAM-Algorithmus das im Rahmen dieser Arbeit gesteckte Ziele einer autonomen Erkundung und Kartographierung durch einen Flugroboter erreicht wurde.

## 4.1 Beschreibung und Modellierung des Quadrotors

Zunächst wird allgemein das Funktionsprinzip eines Quadrotors beschrieben und die besonderen Flugeigenschaften im Hinblick auf die Anwendung in Innenräumen herausgearbeitet. Im weiteren Verlauf wird der im Rahmen dieser Arbeit verwendete Quadrotor AR100B® der Firma AirRobot® vorgestellt. Die Gründe, die zur Wahl dieses Systems führten, werden ebenfalls diskutiert. Die Beschreibung schließt mit der Einführung eines dynamischen Mehrgrößenmodells. Anhand dieses Modells werden später die für den autonomen Flug erforderlichen Regler entworfen.

### 4.1.1 Funktionsprinzip eines Quadrotors

Quadrotoren zeichnen sich, wie es bereits aus der Bezeichnung hervor geht, durch vier Rotoren aus, die in einer Ebene angeordnet sind. Im Zuge der Hubschrauberentwicklung wurden bereits zu Beginn des 20. Jahrhunderts erste Geräte dieser Bauart präsentiert, so wie beispielsweise das Fluggerät von Bothezat. Dieser Quadrotor hat in den Jahren 1922 und 1923 über 100 Testflüge absolviert, ist dabei aber nie über eine Flughöhe von 5m hinaus gekommen [17].

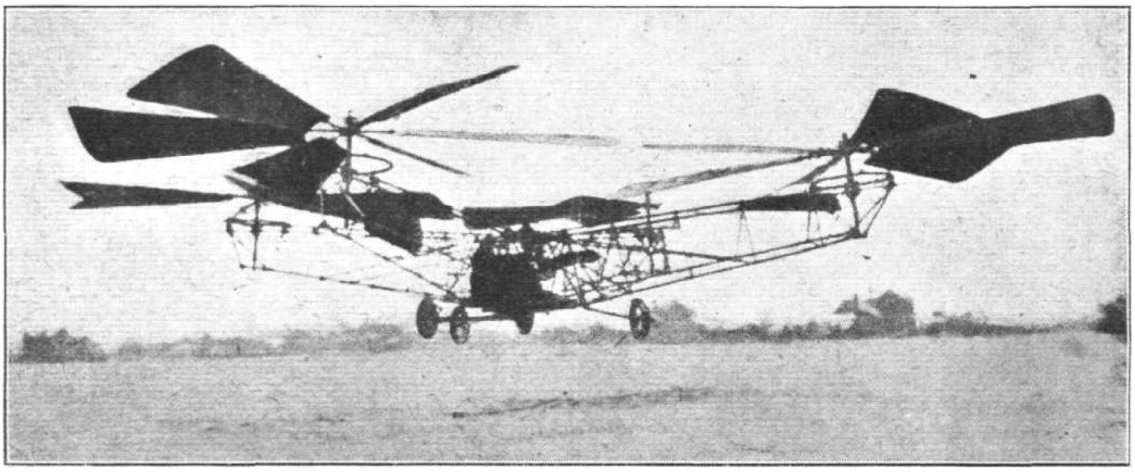


**Bild 4.1:** Quadrotor von Bothezat im Jahre 1922 [13]

Bei den bemannten Drehflüglern hat sich das Konzept des Quadrotors jedoch nicht durchsetzen können. Die heute verbreiteten konventionellen Hubschrauber zeichnen sich in der Regel durch eine Bauart aus, die von einem Haupt- und einem Heckrotor geprägt ist. Der Hauptrotor erzeugt Auftrieb und durch Verstellung der einzelnen Rotorblätter Schub in die gewünschte Flugrichtung. Der Heckrotor gleicht das Drehmoment aus, welches durch die Rotation des Hauptrotors auf den Hubschrauber wirkt. Zusätzlich wird er zur Steuerung der Drehung um die Rotorachse des Hauptrotors, die in diesem Zusammenhang auch als Gierachse bezeichnet wird, verwendet. Diese Kombination aus beiden Rotoren versetzt den Hubschrauber in die Lage, in einer Position zu schweben und sich dabei um die Gierachse drehen zu können [11].

Im Bereich der unbemannten Drehflügler erfreut sich der Quadrotor hingegen wachsender Beliebtheit [56]. Ein wesentlicher Grund dafür ist, dass sein mechanischer Aufbau im Gegensatz zu dem eines konventionellen Hubschraubers deutlich einfacher ist. So wird beispielsweise keine mechanisch sehr aufwändige und wartungsintensive Taumelscheibe für die Verstellung der Rotorblätter benötigt. Bild 4.2 zeigt den schematischen Aufbau eines Quadrotors.

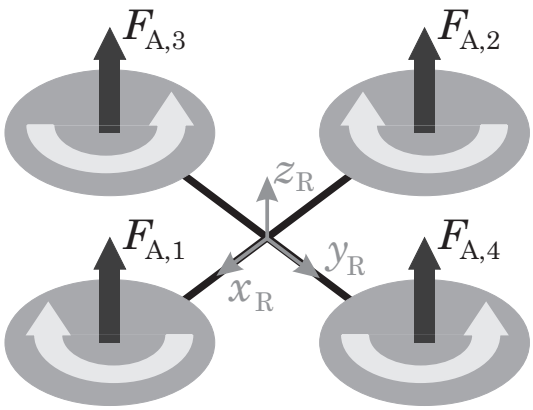



**Bild 4.2:** Schematische Darstellung eines Quadrotors nach [44]

Es wird deutlich, dass die Rotoren, die im Bild 4.2 durch die grauen Kreise dargestellt sind, in einem Kreuz angeordnet sind. Die jeweils gegenüberliegenden Rotoren haben die gleiche Drehrichtung, die durch die hellen Pfeile angedeutet wird. Die Rotoren 1 und 2, die entlang der $x_R$ - Achse des roboterfesten Koordinatensystems angeordnet sind, drehen im Uhrzeigersinn und erzeugen jeweils eine Auftriebskraft $F_{A,1}$ bzw. $F_{A,2}$. Die beiden übrigen Rotoren drehen entgegen dem Uhrzeigersinn und erzeugen ebenfalls eine Auftriebskraft, die hier mit $F_{A,3}$ bzw. $F_{A,4}$ bezeichnet ist [17].

Befindet sich der Quadrotor im Schwebeflug, sind, wie in Bild 4.2 durch die Länge der schwarzen Pfeile dargestellt, alle Auftriebskräfte gleich groß ($F_{A,1} = F_{A,2} = F_{A,3} = F_{A,4}$) und die Summe der Auftriebskräfte gleich der Gewichtskraft des Quadrotors ($F_{A,1} + F_{A,2} + F_{A,3} + F_{A,4} = F_{Quad}$). Bedingt durch die unterschiedlichen Drehrichtungen der Rotoren hebt sich das Drehmoment um

die Gierachse, die $z_\mathrm{R}$ - Achse des roboterfesten Koordinatensystems, auf. Die Auftriebskraft ist bei starren Rotorblättern unter anderem von der Drehzahl der Rotoren abhängig [11]. Daher kann durch eine gleichmäßige Änderung der Drehzahl der vier Rotoren die Bewegung entlang der Gierachse beeinflusst werden. Die Geschwindigkeit, mit der sich der Quadrotor entlang dieser Achse bewegt, wird in diesem Zusammenhang als Steiggeschwindigkeit bezeichnet.

Der Quadrotor ist ebenso wie der konventionelle Hubschrauber in der Lage, eine Drehung um die Gierachse durchzuführen. Hierzu muss die Auftriebskraft der einzelnen Rotoren, wie in Bild 4.3 dargestellt, verändert werden.
Aus der Darstellung wird deutlich, dass die Auftriebskräfte der gegenüberliegenden Rotoren $F_\mathrm{A,3}$ und $F_\mathrm{A,4}$ im Vergleich zu Bild 4.2 reduziert wurden. Dadurch reduziert sich auch das zugehörige Drehmoment. Damit ist das Drehmoment, das durch die Rotoren 1 und 2 erzeugt wird und gegen den Uhrzeigersinn wirkt, nun größer ($F_\mathrm{A,1} + F_\mathrm{A,2} > F_\mathrm{A,3} + F_\mathrm{A,4}$). Dies führt zu einer Drehung des Quadrotors um die Gierachse gegen den Uhrzeigersinn. Diese Art der Bewegung wird auch als *Gieren* bezeichnet. Für die Lage des Quadrotors in Bezug auf die Ebene, die durch die $x_\mathrm{R}$ - Achse und die $y_\mathrm{R}$ - Achse aufgespannt wird, hat diese Art der Auftriebsverteilung keinen Einfluss.

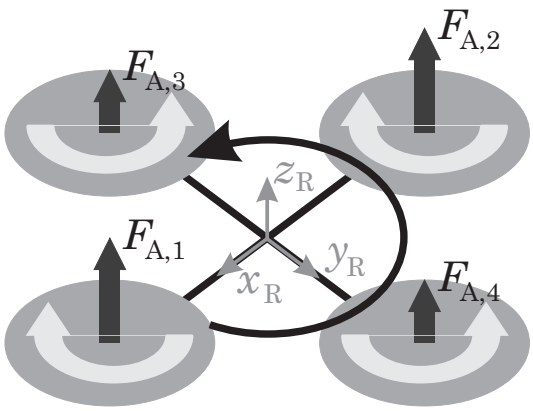
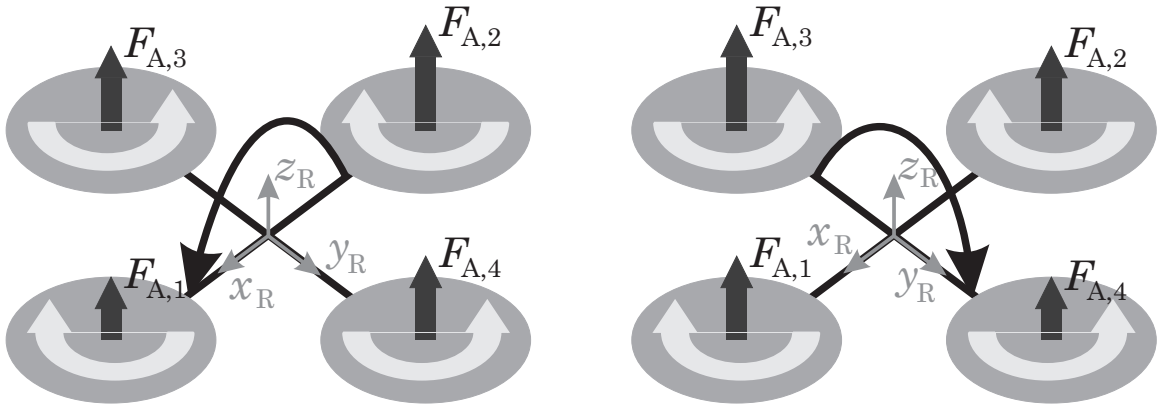



**Bild 4.3:** Schematische Darstellung der Drehung um die $z_\mathrm{R}$ - Achse (gieren)

Um Bewegungen des Quadrotors in der Ebene, die durch die $x_{\mathrm{E}}$ - Achse und die $y_{\mathrm{E}}$ - Achse des Weltkoordinatensystems aufgespannt wird, zu erzeugen, muss die Drehzahl einzelner Rotoren und damit die Auftriebsverteilung verändert werden. Bild 4.4 zeigt die prinzipielle Funktionsweise dieser Bewegungen.




(a) Bewegung entlang der $x_{\mathrm{R}}$ - Achse (nicken) (b) Bewegung entlang der $y_{\mathrm{R}}$ - Achse (rollen)

**Bild 4.4:** Schematische Darstellung der Nick- und Rollbewegung

Mit der in Bild 4.4(a) dargestellten Auftriebsverteilung, bei der die Auftriebskraft $F_{\mathrm{A},1}$ deutlich geringer als die Auftriebskraft $F_{\mathrm{A},2}$ ist, neigt sich der Quadrotor um die $y_{\mathrm{R}}$ - Achse, die als Nickachse bezeichnet wird. Daher spricht man in diesem Zusammenhang bei dieser Art der Bewegung auch vom *Nicken* und der zugehörige Winkel wird durch die Variable $\Theta_{\mathrm{R}}$ repräsentiert. Dieses Nicken führt für $\Psi = 0$ zu einer Bewegung des Quadrotors entlang der $x_{\mathrm{E}}$ - Achse. Um einer zusätzlichen Rotation um die Gierachse entgegenzuwirken, müssen die Auftriebskräfte der übrigen Rotoren dahingehend verändert werden, dass kein entsprechendes Drehmoment auf den Quadrotor wirkt. Daher muss die Summe der Auftriebskräfte der jeweils gegenüberliegenden Rotoren gleich groß sein ($F_{\mathrm{A},1} + F_{\mathrm{A},2} = F_{\mathrm{A},3} + F_{\mathrm{A},4}$).
In Bild 4.4(b) ist die Auftriebsverteilung gezeigt, die für $\Psi = 0$ zu einer Bewegung entlang der $y_{\mathrm{E}}$ - Achse führt. Hier dreht sich der Quadrotor um die $x_{\mathrm{R}}$ - Achse des roboterfesten Koordinatensystems, die man auch Rollachse nennt. Dementsprechend wird die dargestellte Bewegung auch als *Rollen* bezeichnet. Den zugehörigen Winkel gibt die Variable $\Phi_{\mathrm{R}}$ an.

Die Ausführungen dieses Abschnitts verdeutlichen, dass ein Quadrotor die gleichen Flugeigenschaften wie ein konventioneller Hubschrauber besitzt. Aufgrund des einfachen und robusten mechanischen Aufbaus in Kombination mit dem intuitiven Steuerungsprinzip ist der Quadrotor für die im Rahmen dieser Arbeit behandelte autonome Erkundung und Kartographierung unbekannter Innenräume sehr gut geeignet. Der hier verwendete Flugroboter wird im weiteren Verlauf im Detail vorgestellt.

### 4.1.2 Die Flugplattform AR100B®

Der AR100B® der Firma AirRobot® ist ein kommerziell vertriebenes System mit zahlreichen Ausstattungsmerkmalen, die in diesem Abschnitt vorgestellt werden. Bild 4.5 zeigt den Quadrotor, der sich grundsätzlich von einem Bediener durch eine Funkfernsteuerung steuern lässt.
Alle technischen Daten der in Bild 4.5 dargestellten Flugplattform können aus der vom Hersteller mitgelieferten Dokumentation [60] entnommen werden. Die wichtigsten sind in Tabelle 4.1 zusammengefasst. Der in Bild 4.5 gezeigte Quadrotor verfügt über zahlreiche Sensoren. Auf die drei wichtigsten wird im Folgenden kurz eingegangen.

Die zentrale Messeinheit bildet hier, wie auch bei vielen anderen Fluggeräten, die IMU. Durch dieses Sensorsystem können alle translatorischen und rotatorischen Beschleunigungsmessungen durchgeführt werden, die sich für das roboterfeste Koordinatensystem ergeben. Daher ist diese Messeinheit sehr wichtig für die Stabilisierung der Flugplattform.
Um Informationen über die Flughöhe zu bekommen, gibt es im Falle des AR100B® zwei Möglichkeiten. Zum einen verfügt das Gerät über ein Barometer, welches den Luftdruck misst. Bei konstanten Witterungsbedingungen kann so anhand eines in der Startphase gebildeten Referenzwertes die Flughöhe durch den aktuell gemessenen Luftdruck bestimmt werden. Zum anderen kann das ebenfalls an Bord befindliche GPS genutzt werden. Abhängig von der Anzahl der zur Verfügung stehenden empfangbaren Satellitensignale kann die Position und die

Höhe über Normalhöhennull mittels dieses Sensors ermittelt werden [59]. Um die Position auf der Erdoberfläche zu bestimmen, sind drei Satelliten erforderlich. Zusätzliche Informationen über die Höhe erhält man, wenn zeitgleich das Signal von vier Satelliten empfangen werden kann.

**Bild 4.5:** Der AR100B® mit einer Kamera

**Tabelle 4.1:** Auszug der technischen Daten des AR100B® [60]

| | |
|---|---|
| L x B x H | 1 m x 1 m x 0,23 m |
| Reichweite | bis 1000 m |
| Traglast | 0,4 kg |
| Flugzeit | bis zu 25 min |
| max. Geschwindigkeit | 13 m/s |

Auf Basis der IMU - Daten hat der Hersteller des Systems eine leistungsfähige Lageregelung zur Stabilisierung der Plattform realisiert. Die zur Regelung erforderlichen Berechnungen werden auf einem an Bord befindlichen Microcontroller durchgeführt. Das daraus resultierende Stellsignal wirkt auf die Drehzahl der vier Motoren, die die jeweiligen Rotoren so antreiben, dass sich die gewünschte Lage ($\Theta_R$,$\Phi_R$,$\Psi_R$) des Quadrotors einstellt. Diese kann entweder über den mitgelieferten Handsender oder durch eine unterhalb des Korpus befindliche Schnittstelle kommandiert werden. Das Steuerkommando des Handsenders

hat jedoch Priorität. Diese Tatsache wird bei den im weiteren Verlauf dieses Kapitels beschriebenen Experimenten ausgenutzt, um im Notfall die durch das im Abschnitt 4.2 vorgestellte Modul generierten Steuerkommandos zu überstimmen und einen Absturz zu vermeiden.

Eine weitere Möglichkeit, um von einem Computer generierte Steuerkommandos an das Fluggerät zu übermitteln, ist, das Pulsmuster des Handsenders am Boden durch einen Computer zu emulieren. Computer und Handsender können dann in einem aus dem Modellflug bekannten „Lehrer - Schüler - Modus“ betrieben werden. Der Computer stellt in dieser Betriebsart den Schüler da und die von ihm generierten Signale werden über den HF-Teil des Handsenders, der als Lehrer fungiert, an den Quadrotor gesendet. Der Bediener des Handsenders kann auch in dieser Betriebsart jeder Zeit eingreifen und die Kontrolle übernehmen. Ein weiterer Vorteil dieser Variante ist, dass der „Lehrer - Schüler - Modus“ die Freigabe einzelner Kanäle zulässt. Man hat somit beispielsweise die Möglichkeit, zum Test bestimmter Regler nur den dafür erforderlichen Kanal für den Computer zu öffnen. Alle übrigen Kanäle können weiter durch den Bediener kontrolliert werden. So wird das Risiko von Abstürzen und die damit verbundene Beschädigung des Fluggeräts minimiert.
Die geschilderte Betriebsart wird im folgenden Abschnitt genutzt, um bei der Modellierung und anschließenden Validierung ein hohes Maß an Sicherheit zu gewährleisten und Regelkreise unabhängig voneinander überprüfen zu können.

### 4.1.3 Dynamisches Modell der Flugplattform

In Bezug auf die Modellierung und Regelung von Quadrotoren sind bereits einige Arbeiten durchgeführt worden [14, 15, 35]. Im Bereich der Modellierung unbemannter Fluggeräte gibt die Zusammenstellung von Valavanis aus dem Jahr 2007 [56] einen guten Überblick. Die dort beschriebene Modellierung ist in ihren Grundzügen im Abschnitt A.1 vorgestellt. Die Anzahl der für die hier dargestellte Form der Modellierung erforderlichen Parameter ist sehr groß und setzt sehr genaue

Kenntnisse über die Flugplattform und ihre Bestandteile voraus. Da der Hersteller des in dieser Arbeit verwendeten industriellen Quadrotors viele der erforderlichen Daten nicht zur Verfügung stellt, wird die Verwendung dieses Modellansatzes erheblich erschwert. Hinzu kommt, dass von Seiten des Herstellers bereits Regelungen, bezogen auf alle Freiheitsgerade $(x_\mathrm{R}, y_\mathrm{R}, z_\mathrm{R}, \Theta_\mathrm{R}, \Phi_\mathrm{R}, \Psi_\mathrm{R})$, auf dem Quadrotor implementiert wurden und die Struktur sowie die Parameter dieser Regelung ebenfalls unbekannt sind. Aufgrund dieser Tatsachen wurde im Rahmen dieser Arbeit ein dynamisches Modell eines Quadrotors inklusive einer bereits vorhandenen Lageregelung erstellt. Die Komplexität dieses Modells, welches hier vorgestellt wird, ist aufgrund der nicht verfügbaren detaillierten Parameter deutlich geringer.

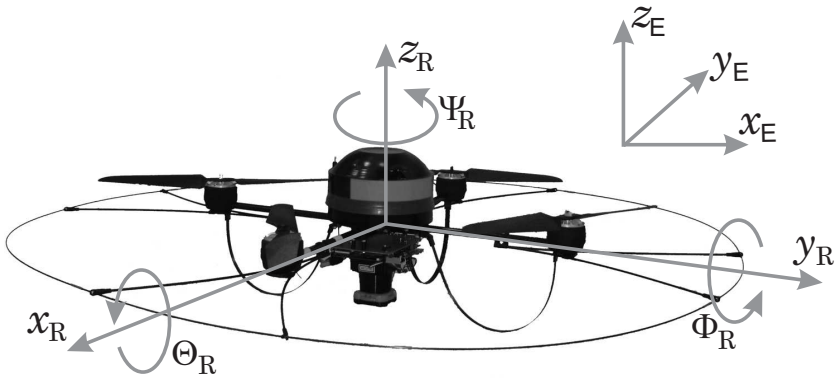



**Bild 4.6:** Verwendeten Koordinatensysteme am Beispiel des AR100B®

Bild 4.6 gibt einen Überblick über die für die Modellierung wichtigen translatorischen ($x_\mathrm{R}$, $y_\mathrm{R}$, $z_\mathrm{R}$) und rotatorischen ($\Theta_\mathrm{R}$, $\Phi_\mathrm{R}$, $\Psi_\mathrm{R}$) Koordinaten. Die Unterscheidung zwischen Erd- und Roboterkoordinatensystem ist erforderlich, da sich der Roboter um die Gierachse drehen kann und später die durch den Roboter gewonnenen Informationen über die Umgebung unabhängig von seiner Ausrichtung gespeichert werden sollen.

Für die folgende Modellierung wurde der vom Hersteller mitgelieferte Handsender durch eine Signalleitung mit einer Bodenstation verbunden und so in den Versuchsaufbau integriert. Die gesamte Anordnung zeigt Bild 4.7.

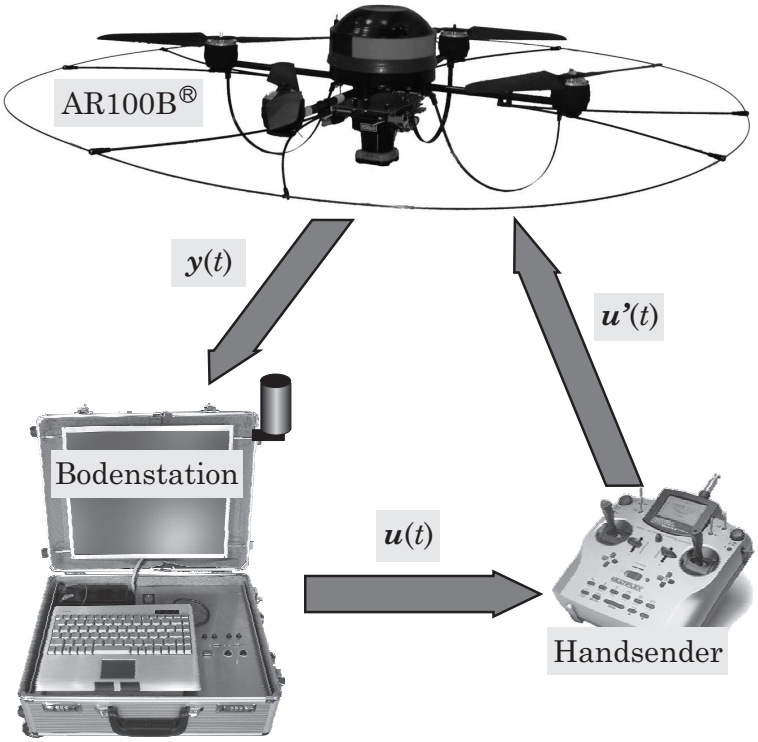



**Bild 4.7:** Versuchsanordnung bei der Modellierung

Die in Bild 4.7 dargestellte Bodenstation ist ein Eigenbau, der im Wesentlichen aus einem handelsüblichen PC, einem Microcontroller und einer Empfangseinheit für die gesendeten Daten $\boldsymbol{y}(t)$ des Fluggeräts besteht. Der PC dient dabei als Programmierplattform für den Microcontroller und zum Daten-Monitoring. Auf dem Microcontroller werden die vom Fluggerät empfangenen Daten verarbeitet und die Steuergrößen $u_z$, $u_x$, $u_y$ und $u_\Psi$ erzeugt. Damit diese vom Handsender interpretiert und an die Flugplattform übermittelt werden können, werden sie durch eine Pulslängenmodulation repräsentiert. Das zur Übertragung der Steuergrößen verwendete Pulsmuster ist schematisch in Bild 4.8 dargestellt. Es entspricht jenem, welches der Handsender seinerseits in Abhängigkeit der jeweiligen Knüppelstellungen erzeugt. Durch entsprechende Einstellungen am Handsender kann man einen oder mehrere Kanäle auswählen, über die Daten direkt vom Microcontroller an das Fluggerät übermittelt werden. Die übrigen Kanäle

können über den Handsender manuell kontrolliert werden. Somit kann das Fluggerät vollständig oder nur teilweise mit Hilfe des Rechners gesteuert werden.

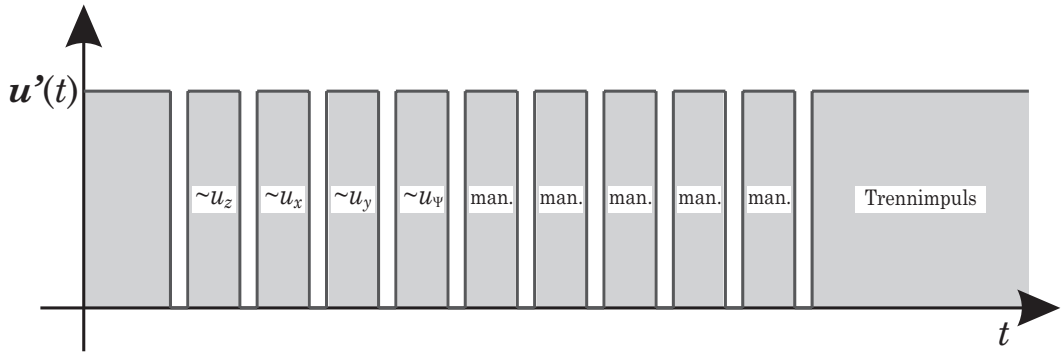



**Bild 4.8:** Pulsmuster zur Übertragung der Steuergrößen

Das dargestellte Pulsmuster verfügt über neun Pulse, die abhängig von den zugehörigen Knüppelstellungen in ihrer Länge zwischen 0,6 ms und 1,8 ms variieren. Die minimale Länge entspricht dabei dem Zahlenwert 0, die maximale Länge einem Wert von 255. Der Zahlenwert 128 beschreibt die Mittelstellung des zugehörigen Knüppels. Für die nachfolgend beschriebene Modellbildung werden die Eingangsgrößen immer bezogen auf die Abweichung zu der Mittelstellung angegeben. Da für die Modellierung des Fluggeräts nur die ersten vier Kanäle relevant sind, werden die übrigen immer manuell kontrolliert und sind entsprechend gekennzeichnet. Der Trennimpuls ist mit 10 ms deutlich länger als die übrigen Pulse und markiert sowohl das Ende der aktuellen Pulsfolge sowie den Beginn einer neuen.

Durch die vom Hersteller bereits implementierten Regler besteht für die anstehende Modellierung folgender Zusammenhang zwischen den Steuergrößen und den zu messenden Ausgangsgrößen. Die Steiggeschwindigkeit ${}^{\mathrm{R}}\dot{z}_{\mathrm{E}}$ ergibt sich aus der Steuergröße $u_z$ und die Rotationsgeschwindigkeit $\dot{\Psi}$ aus der Steuergröße $u_\Psi$. Die Eingangsgrößen $u_x$ und $u_y$ bewirken eine Änderung der Anstellwinkel $\Theta_{\mathrm{R}}$ und $\Phi_{\mathrm{R}}$ und führen so zu den resultierenden Geschwindigkeiten ${}^{\mathrm{R}}\dot{x}_{\mathrm{E}}$ und ${}^{\mathrm{R}}\dot{y}_{\mathrm{E}}$. Durch die Verwendung von zwei Indizes wird deutlich, dass es sich jeweils um die Geschwindigkeit des Ro-

boters im Weltkoordinatensystem handelt. Durch verschiedene Flugexperimente wurde der jeweilige dynamische Zusammenhang zwischen den Eingangsgrößen ($u_z$, $u_x$, $u_y$, $u_\Psi$) und den zugehörigen Ausgangsgrößen (${}^{\mathrm{R}}\dot{z}_{\mathrm{E}}$, ${}^{\mathrm{R}}\dot{x}_{\mathrm{E}}$, ${}^{\mathrm{R}}\dot{y}_{\mathrm{E}}$, $\dot{\Psi}_{\mathrm{R}}$) ermittelt. Die einzelnen Schritte sind in Anlehnung an einen eigenen Konferenzbeitrag [1] in den Abschnitten A.2 und A.3 detailliert beschrieben. Ergebnis dieser Betrachtungen ist das in Bild 4.9 dargestellte nichtlineare Mehrgrößenmodell.

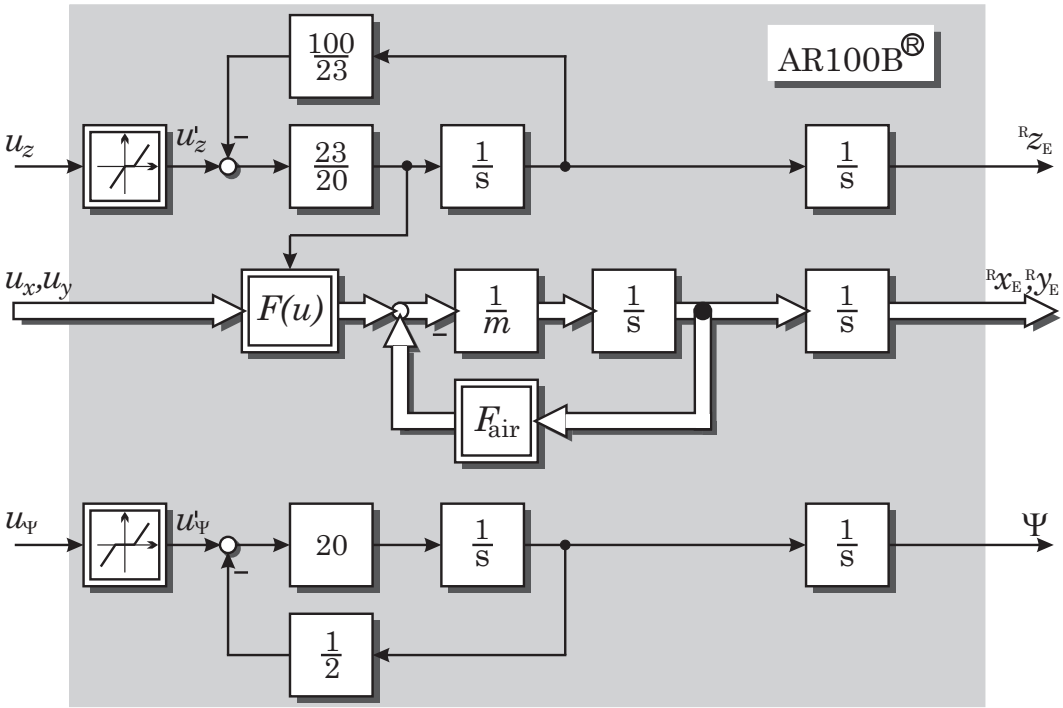



**Bild 4.9:** Nichtlineares Mehrgrößenmodel des AR100B®

Bei der Modellierung wurde der mögliche Einfluss von Wind vernachlässigt, da im Rahmen dieser Arbeit der Betrieb des Quadrotors in Innenräumen betrachtet wird. Für die Kartographierung ist dessen Position in Weltkoordinaten von Interesse. Für die Darstellung der Ausgangsgrößen des Modells werden dementsprechend ebenfalls zwei Indizes verwendet. Die Ermittlung der einzelnen Parameter und die Verwendung der beiden in Bild 4.9 eingezeichneten Totzonen ist den Abschnitten A.2 und A.3 zu entnehmen. Aufgrund der im Allgemeinen beliebigen Ausrichtung $\Psi$ des Roboters ist diese bei der Umrechnung der Stellgrößen $u_x$ und $u_y$ auf die Ansteuersignale für die Motoren zu berücksichtigen.

Auch die Flughöhe des Quadrotors ist in der Regel nicht konstant. Daher muss eine Kopplung zwischen der im oberen Bereich von Bild 4.9 dargestellten Modellierung der translatorischen Bewegung entlang der $z_\mathrm{E}$ - Achse und dem die $x_\mathrm{E}$ / $y_\mathrm{E}$ - Ebene betreffenden Modellzweig berücksichtigt werden. Bild 4.9 zeigt, dass diese Kopplung auf die Umrechnung der Eingangsgrößen auf die Anstellwinkel wirkt. Bei dieser im Abschnitt A.3 ausgeführten Berechnung wird die Erdbeschleunigung $g$ ausgenutzt. Bei auftretenden Höhenänderungen muss hier zusätzlich die Beschleunigung ${}^{\mathrm{R}}\ddot{z}_\mathrm{E}$ berücksichtigt werden.
Damit ist das dynamische Modell des lagegeregelten Quadrotors mit hinreichender Genauigkeit beschrieben. Das gezeigte Modell bildet die Grundlage für verschiedene Simulationen im Hinblick auf das Bewegungsverhalten bei der Erkundung und Kartographierung unbekannter Innenräume und die Synthese der Regler für den autonomen Flugbetrieb.

## 4.2 Die verwendete Nutzlast ICAM

Um mit dem industriellen Quadrotor eine autonome Erkundung und Kartographierung von Innenräumen realisieren zu können, war der Aufbau einer eigenen Nutzlast erforderlich. Bild 4.10 zeigt die entstandene ICAM (engl. ***I**ntelligent **C**ontrol **A**nd **M**easurement Unit*), deren Hauptbestandteile der Laserscanner und zwei Microcontroller sind. Diese Komponenten werden im Verlauf dieses Abschnitts noch genauer beschrieben.

Zusätzlich zu den zentralen Komponenten sind eine Reihe an Peripheriebausteinen erforderlich. So ist neben verschiedenen Spannungsreglern ein Baustein zur Pegelanpassung verbaut, um eine Kompatibilität zwischen den auf unterschiedlichen Spannungsebenen liegenden seriellen Schnittstellen sicher zu stellen.
Mit einem Gewicht von ca. 0,45 kg liegt die ICAM leicht oberhalb der vom Hersteller des Quadrotors angegebenen maximalen Nutzlast (siehe Tabelle 4.1 auf Seite 57). Die Spannungsversorgung erfolgt ebenso wie die Kommunikation mit der Flugplattform über die im vorigen Abschnitt angesprochene Schnittstelle unterhalb des Korpus. Die auf einem

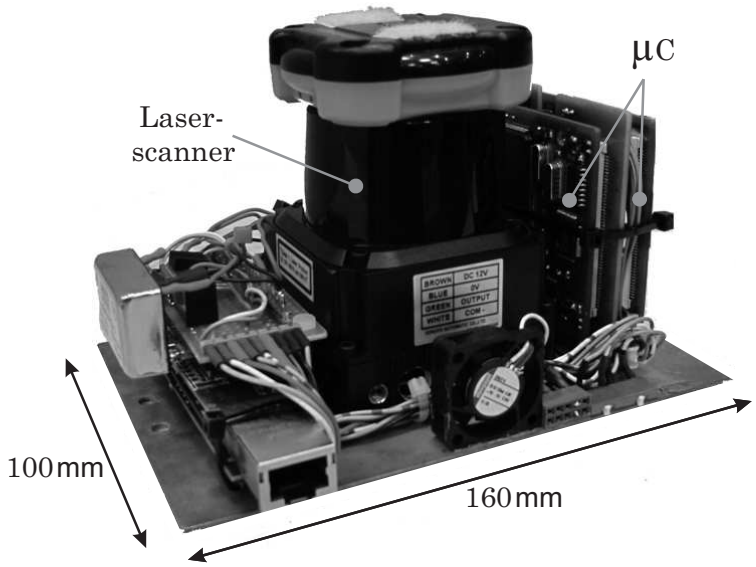



**Bild 4.10:** Nutzlast ICAM mit Laserscanner und Microcontrollern

der Microcontroller realisierten und für den autonomen Flug benötigten Regler greifen auf Sensordaten des AR100B® und des Laserscanners zurück. Die aus den Berechnungen resultierenden Steuerkommandos werden dann über die Schnittstelle direkt an den Quadrotor übermittelt. Ein Kontakt zu einer Bodenstation ist dafür nicht erforderlich. Einzig zur Übertragung der Sollwerte und zum Daten-Monitoring wird die Bodenstation benötigt, die ein handelsübliches Notebook darstellt. Die Datenverbindung zwischen Bodenstation und ICAM wird über eine kabellose Netzwerkverbindung (engl. ***W****ireless* ***L****ocal* ***A****rea* ***N****etwork*, kurz WLAN) realisiert.

*Der Laserscanner UTM-30LX*

Der von der Firma Hokuyo® hergestellte Laserscanner UTM-30LX, den Bild 4.11 zeigt, ist der wichtigste Sensor in Bezug auf den autonomen Flug und die Kartographierung. Die Gründe, die zur Wahl dieses Sensors führten, wurden bereits zu Beginn von Kapitel 3 genannt. Die wesentlichen Merkmale sind in Tabelle 4.2 zusammengefasst.
Ein Laserscanner ist eine optische Messeinrichtung. Der hier verwendete arbeitet im infraroten Bereich mit einer Wellenlänge von $\lambda = 905\,\mathsf{nm}$. Die Entfernung zu einem Objekt wird durch die Oberflächenreflektion des gesendeten Laserstrahls bestimmt. Das Messprinzip bei Produkten

der Firma Hokuyo® ist das Phasendifferenzverfahren, bei dem aufgrund der Laufzeit eine Phasenverschiebung zwischen dem gesendeten und dem reflektierten Signal ausgenutzt wird [36].

**Bild 4.11:** Laserscanner UTM-30LX von Hokuyo®

**Tabelle 4.2:** Technischen Daten des UBG-04LX-F01(Rapid URG)® [36]

| | |
|---|---|
| L x B x H | 60 mm x 60 mm x 87 mm |
| Detektionsentfernung | 30 m |
| Genauigkeit | 0,1 m $\mapsto$ 10 m: $\pm$ 30 mm 10 m $\mapsto$ 30 m: $\pm$ 50 mm |
| Scanbereich | 270° |
| Abtastintervall | 25 msec |
| Gewicht | 0,36 kg |

Da der Laserstrahl mechanisch über einen rotierenden Spiegel in einer Ebene abgelenkt wird, beinhaltet ein vollständiger Scan mehrere Messpunkte in dieser Ebene. Mit dem hier verwendeten Gerät lässt sich ein Bereich von 270° mit einer maximalen Auflösung von 0,25° erfassen. Im Verlauf dieser Arbeit wird der Laserscanner lediglich mit einer Auflösung von 1° betrieben, um die Anzahl der Messwerte von 1080 auf 270 zu reduzieren.

Ein Nachteil bei der Abstandsmessung mittels Laserscanner ist, dass der gesendete Laserstrahl nicht notwendigerweise senkrecht zur Oberfläche des Hindernisses auftrifft. Trifft er auf eine schräge Oberfläche, kommt es durch Streuung zu einer geringeren messbaren Reflektion. Dadurch kommt es zu Ungenauigkeiten bei der Entfernungsmessung, die mit wachsendem Abstand größer werden. Dies spiegelt sich auch bei den in Tabelle 4.2 angegebenen Toleranzen wider.

*Das Microcontrollermodul M167*

Aufgrund der Echtzeitanwendung im Flugbetrieb ist ein Microcontroller besonders gut geeignet, da hier im Hinblick auf die später angestrebten Abtastzeiten von 0,2s für die Regelung und 1s für den SLAM-Algorithmus nur vergleichsweise geringe Latenzzeiten bei der Ausführung der jeweiligen Programme auftreten. Es handelt sich um eine Rechenplattform, die sehr oft in hardwarenahen Echtzeitanwendungen verwendet wird. Bild 4.12 zeigt das hier verwendete Modul.

Zusätzlich zu dem 16 bit Controller C167CR der Firma Infineon® beinhaltet das Board zahlreiche Peripherieelemente wie beispielsweise Treiberbausteine und Speicher. Durch die große Anzahl serieller Schnittstellen ist dieses Modul besonders gut für die geplante Anwendung geeignet. Die wichtigsten Daten sind in Tabelle 4.3 zusammengefasst.
Ein weiterer Vorteil des Microcontrollermoduls ist, dass über die an der Unterseite befindlichen Steckerleisten alle wichtigen Schnittstellen mechanisch einfach zugänglich sind. Der Aufbau der in Bild 4.10 gezeigten Lasteinheit und die Handhabung im Testbetrieb wurden so deutlich erleichtert.

Auf den beiden in der Lasteinheit verbauten Modulen wurden alle für den Verlauf der Arbeit relevanten Algorithmen mit Hilfe der Programmiersprache C implementiert. Ein Controller übernimmt die regelungstechnischen Aufgaben für den autonomen Flug, der andere die Kartographierung.

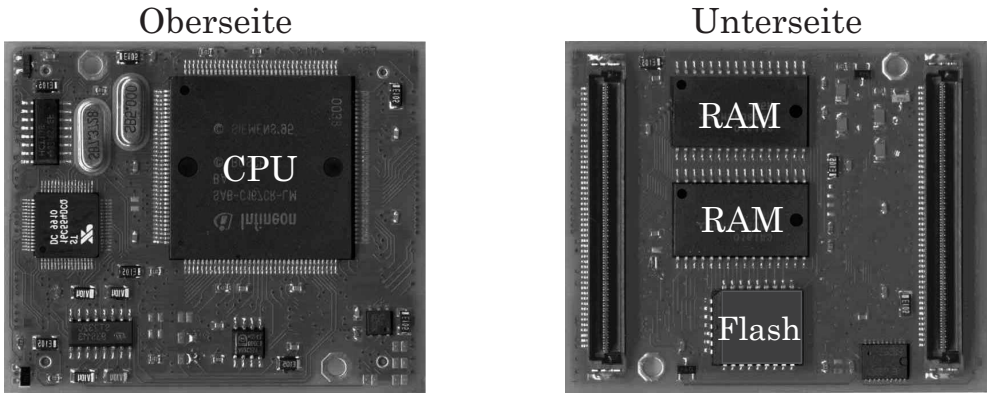



**Bild 4.12:** Microcontrollermodul M167

**Tabelle 4.3:** Auszug der technischen Daten des M167® [26]

| | |
|---|---|
| L x B x H | 64 mm x 54 mm x 10 mm |
| CPU | C167CS (16 bit 24 MHz) |
| Arbeitsspeicher | 1 Mb |
| Flashspeicher | 512 Kb |

## 4.3 Experiment zur Parameteranpassung

Nach der Einführung der verwendeten Hardware wird zunächst die Durchführung eines vorbereitenden Experiments beschrieben, welches zur Ermittlung der in Kapitel 3 eingeführten Entscheidungsvariablen dient. Dabei stehen neben der Genauigkeit der globalen Karte auch die Robustheit gegenüber im Flug auftretenden Störungen und die Zykluszeit des SLAM-Algorithmus im Vordergrund. Hierzu wurde das bereits aus der Simulation in Abschnitt 3.4 bekannte Szenario in einer Sporthalle mit Hilfe von Stellwänden und Planen aufgebaut. Die Flugplattform wurde mit montierter ICAM manuell durch diese Umgebung navigiert. Dabei hat der Bediener versucht, eine konstante Geschwindigkeit von ca. 1 m/s zu erreichen und analog zu der durchgeführten Simulation die Ausrichtung des Quadrotors während des Fluges achsenparallel zum Weltkoordinatensystem zu halten. Während des Fluges mit annähernd konstanter Höhe hat der Laserscanner mit einer Abtastrate von 1 Hz Messungen durchgeführt und die zugehörigen Daten auf dem Micro-

controller gespeichert. Diese wurden im Anschluss zur Ermittlung der gesuchten Variablen verwendet. Hierzu wurde auf Basis der Messdaten unter Verwendung der in der ICAM verbauten Hardware die globale Karte mit verschiedenen Parametersätzen berechnet. Die Reichweite des Laserscanners wurde dabei auf 15 m reduziert. Zum einen wird es dadurch bereits in kleineren Räumen möglich, die Leistungsfähigkeit des SLAM-Algorithmus zu überprüfen und zum anderen wird der Einfluss von Messfehlern reduziert, die gemäß der Herstellerangaben mit zunehmender Entfernung steigen.

**Tabelle 4.4:** Wahl der Entscheidungsvariablen

| Variable | $q$ | $D_{\mathrm{sm}}$ | $l_{\mathrm{merge}}$ | $l_{\mathrm{line}}$ |
|---|---|---|---|---|
| Wert | 0,3 m | 0,05 m | 0,1 m | 0,1 m |

Nach mehreren Durchläufen des SLAM-Algorithmus stellten sich die in Tabelle 4.4 zusammengefassten Variablen als sinnvolle Wahl heraus. Die Laufzeit bei diesen Werten beträgt auf der hier verwendeten Hardware maximal 0,9 s und im Mittel 0,7 s. Da die Zeit mit der Anzahl der in der globalen Karte verzeichneten Linien leicht variiert, kann auf Basis des angegebenen Parametersatzes eine echtzeitfähige Implementierung des Algorithmus mit einer Abtastzeit von 1,0 s erfolgen. Neben den tabellarisch aufgelisteten Variablen wurden bei der Berechnung der lokalen Karte lediglich Linien berücksichtigt, die mindestens auf drei Stützstellen basieren und ein Länge von mindestens 0,2 m haben. Dies erhöht zusätzlich die Übersichtlichkeit der resultierenden Karte, die Bild 4.13 zeigt.

Das mit dem in Tabelle 4.4 angegebenen Parametersatz erzielte Ergebnis zeigt eine gute Übereinstimmung mit der Realität. Die Struktur des untersuchten Raums ist klar erkennbar und auch die tatsächlichen Abmessungen werden durch die vom SLAM-Algorithmus generierte Karte gut wiedergegeben. Es fällt jedoch auf, dass in Bild 4.13 einige Linien doppelt vorhanden sind. Diese Ungenauigkeit hat neben dem

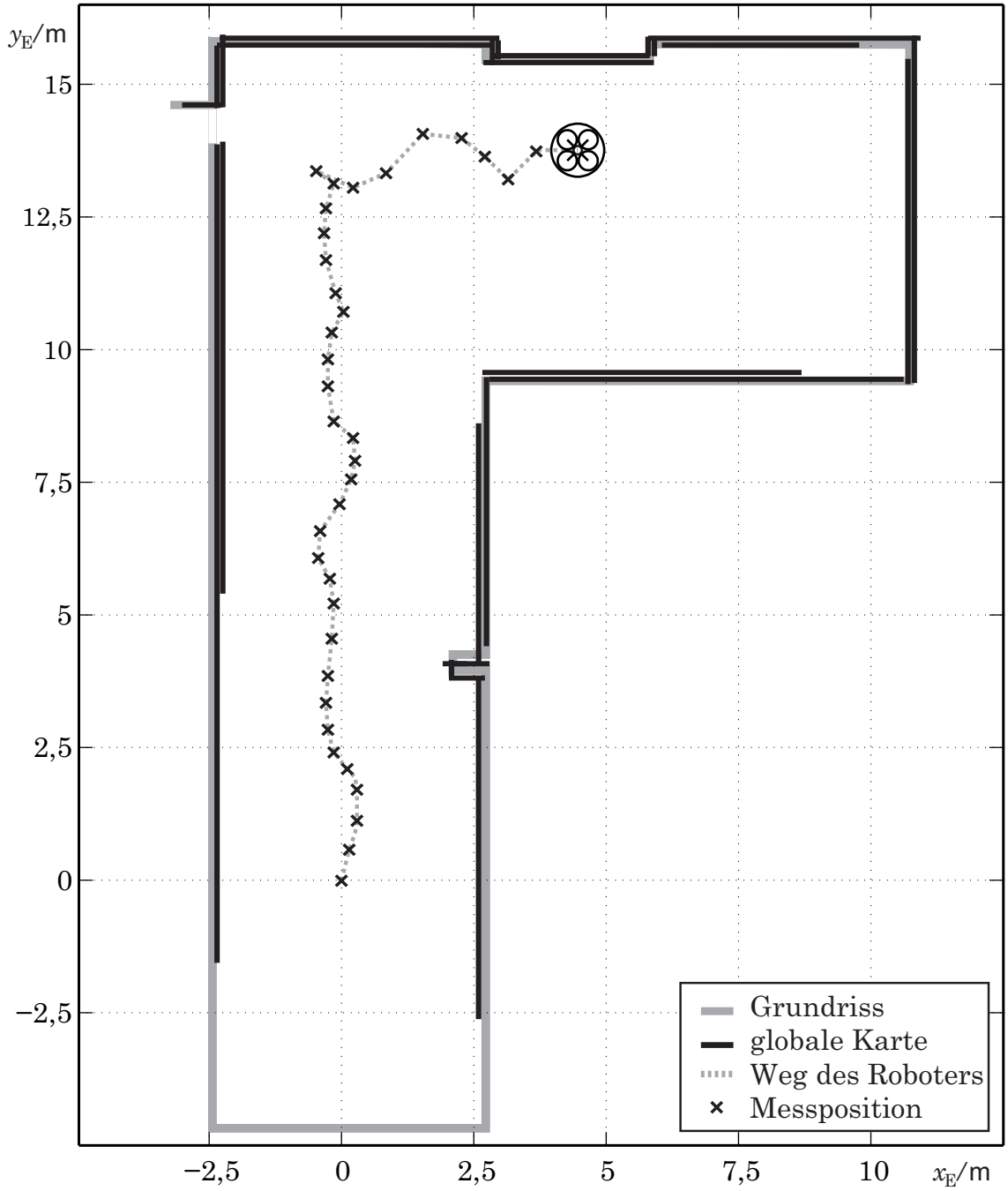



**Bild 4.13:** Ergebnis der Offline-Rechnung mit Daten aus manuellem Flug

Messrauschen des Laserscanners noch eine weitere Ursache. Aufgrund seiner Flugeigenschaften neigt sich der Quadrotor in die Richtung, in die er fliegt. Damit neigt sich auch der Laserscanner und die Entfernungsmessung wird so beeinflusst. Um diesem Fehler entgegen zu wirken, könnte eine entsprechende Vorrichtung dafür sorgen, dass die Neigungen des Flugroboters kompensiert werden und der Laserscanner

immer in einer Ebene parallel zum Boden misst. Eine solche Vorrichtung bedeutet zusätzliches Gewicht. Aus Abschnitt 4.2 wird deutlich, dass die maximale Nutzlast des Quadrotors bereits ohne Neigemechanismus leicht überschritten ist. Daher kann bei weiterer Verwendung der bisher vorgestellten Hardware eine solche Komponente nicht nachgerüstet werden.
Eine weitere Möglichkeit wäre, bei bekannter Neigung, diese bei der Umrechnung der Messdaten in kartesische Koordinaten zu berücksichtigen. Die entsprechenden Lagewinkel stehen als Messgröße zwar zur Verfügung, unterliegen jedoch einem starken Rauschen. Zudem ist die Abtastrate für die Implementierung eines Kalman-Filters, welches oft zur Aufbereitung verrauschter Messsignale verwendet wird, zu niedrig. Im Rahmen dieser Arbeit wird daher der Einfluss des aus der Neigung resultierenden Messfehlers auf 1 % reduziert, indem die Stellgrößen $u_x$ und $u_y$ begrenzt werden. Aufgrund der trigonometrischen Funktionen ergibt sich, dass die Neigung der Plattform einen Winkel von 8° nicht übersteigen darf, um die eben aufgestellte Forderung eines Fehlers von maximal 1 % zu erfüllen.

Bild 4.13 zeigt darüber hinaus, dass die vorgestellte Methode zur Abschätzung der Ausrichtung zuverlässig funktioniert. Zwar wurde während des Fluges versucht, die Ausrichtung konstant zu halten, doch wie bei einer manuellen Bedienung zu erwarten, unterliegen alle Steuerkommandos und somit auch die zugehörigen Zustände gewissen Schwankungen. Diese werden durch den Algorithmus jedoch hinreichend gut kompensiert.
Aus dem eingezeichneten Weg des Roboters wird deutlich, dass versucht wurde, parallel zu der auf der linken Seite befindlichen Wand zu fliegen. In der oberen linken Ecke in Bild 4.13 wurde eine 90° - Drehung kommandiert. Bei der Berechnung der Karte durch den SLAM-Algorithmus wurde bei den zugehörigen Datensätzen des Laserscanners dieser Offset bei der Bestimmung der Ausrichtung berücksichtigt. Die Ausführungen im folgenden Abschnitt werden zeigen, dass diese Strategie auch beim autonomen Flug verfolgt wird.

## 4.4 Autonome Erkundung und Kartographierung

In diesem Abschnitt wird anhand von drei verschiedenen Szenarien gezeigt, dass das gesamte System im autonomen Betrieb funktionstüchtig ist. Zu Beginn dieses Abschnitts liegt der Fokus auf der Auswahl der Regler und der Navigationsstrategie für den autonomen Flug in Innenräumen. Hierbei ist die Hardware, auf der die zugehörigen Algorithmen später realisiert werden sollen, von besonderer Bedeutung. Analog zu dem SLAM-Algorithmus sollen auch diese auf einem 16bit Microcontroller implementiert werden und mit einer Taktrate von 5 Hz lauffähig sein. Eine höhere Taktung ist nicht erforderlich, da der verwendete Quadrotor üblicherweise manuell bedient wird und dabei oftmals deutlich höhere Reaktionszeiten auftreten.

Aus Kapitel 4.1.3 geht hervor, dass für den autonomen Betrieb vier Ausgangsgrößen $({}^{\mathrm{R}}x_{\mathrm{E}}, {}^{\mathrm{R}}y_{\mathrm{E}}, {}^{\mathrm{R}}z_{\mathrm{E}}, \Psi)$ durch entsprechende Regler geeignet beeinflusst werden müssen, um das Fluggerät eindeutig im Raum zu positionieren. Da im Innenraum im Allgemeinen nur vergleichsweise kleine Abweichungen zu den Sollwerten auftreten und sich das System mit niedrigen Geschwindigkeiten bewegt, wurde das auf Seite 62 in Bild 4.9 vorgestellte nichtlineare Modell für die entsprechenden Arbeitspunkte linearisiert. Da der Fokus dieser Arbeit auf der Kartographierung der Innenräume liegt und die Laufzeit des gesamten Algorithmus möglichst gering bleiben soll, wurden auf Basis dieses vereinfachten Modells lineare Standardregler ermittelt, die eine hinreichende Regelgüte erzielen. Für die Regelung der Flughöhe und der Positionierung in $x$- und $y$-Richtung fiel die Wahl auf PD-Regler, für die Gierachse wurde lediglich ein P-Regler implementiert. Dabei wurden für die jeweiligen Regelkreise die im vorherigen Abschnitt angesprochenen Stellgrößenbeschränkungen beachtet, um eine zu starke Neigung des Quadrotors zu verhindern.

Die zur Regelung erforderlichen Istwerte werden im Falle der Ausrichtung nach dem in Abschnitt 3.1 beschriebenen Verfahren ermittelt. Die aktuelle Position in $x$- und $y$-Richtung wird ebenfalls über den Laserscanner erfasst. Als Sensor für die Flughöhe dient das im Qua-

drotor verbaute Barometer. Die Verwendung dieses Sensors ist in Innenräumen nur bedingt geeignet, da es zu erheblichen Schwankungen im Luftdruck kommen kann. Der durch die Rotoren erzeugte Abwind verwirbelt die Luft unterhalb des Flugroboters. In im Vergleich zu seiner Größe engen Räumen wirken sich diese Abwinde durch Reflektion am Boden und an den Wänden negativ aus das Flugverhalten und die Luftdruckmessung aus. Da die Umgebung, die bei den später gezeigten Experimenten durchflogen wird, zum großen Teil durch Stellwände in einem großen Raum realisiert ist, liefert die Verwendung des Barometers zufriedenstellende Ergebnisse. Sind die Räume so klein, dass sich die Verwendung eines solchen Sensors nicht eignet, kann durch Montage eines Winkelspiegels ebenfalls der Laserscanner zur Ermittlung der Höhe verwendet werden [31]. Bild 4.14 zeigt eine entsprechende schematische Darstellung.

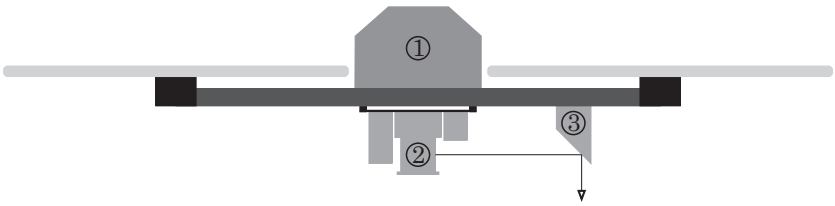



**Bild 4.14:** Quatrotor (1) mit Winkelspiegel (3) zur Ablenkung des Messstrahls vom Laserscanner (2)

Durch die Montage eines Winkelspiegels wäre der für die Kartographierung zur Verfügung stehende Messbereich des Laserscanners leicht reduziert. Diese Einschränkung hätte bei dem hier verwendeten Sensor nur geringe Auswirkungen auf die Qualität der aus dem SLAM-Algorithmus resultierenden Karte. Aufgrund des erhöhten Gewichts und der oben beschriebenen Beschaffenheit der Umgebung wurde bei der Durchführung der folgenden Versuche jedoch auf einen Winkelspiegel verzichtet.

Aus den gleichen Gründen, die zu einer Wahl der einfachen Regler führen, wird auch eine vergleichsweise einfache Navigationsstrategie gewählt. Der Quadrotor soll sich mit konstantem Abstand und konstan-

ter Geschwindigkeit parallel zu einer Wand auf seiner, in Flugrichtung gesehen, linken Seite bewegen. Da eine vorwiegend rechtwinklige Struktur der Umgebung vorausgesetzt wird, ändert der Roboter seine Flugrichtung nur bei Detektion einer 90° - Ecke. Trifft er auf eine Ecke, bei der die sogenannte Führungswand nach rechts fortgesetzt wird, führt er eine Drehung um 90° auf der Stelle aus und setzt seinen Flug fort. Knickt die Wand nach links ab, so bewegt sich der Quadrotor mit einem konstanten Radius um die zugehörige Ecke. Danach schwenkt er wieder auf den Parallelflug ein. Die jeweiligen Drehungen werden als Offset bei der Abschätzung der Ausrichtung berücksichtigt. Um grundsätzlich die Funktionsfähigkeit der autonomen Erkundung und Kartographierung von Innenräumen zu zeigen, ist dieses Verfahren völlig ausreichend. Die Verwendung einer komplexeren Strategie hat auf den hier verwendeten SLAM-Algorithmus keine gravierenden Auswirkungen.

Im ersten vorgestellten Szenario steht die Eigenschaft des *Loop closing* im Vordergrund. Hier geht es darum, eine geschlossene Karte zu erhalten, wenn der Roboter sich in einer Schleife bewegt und dabei den Sichtkontakt zu seinem Ausgangspunkt zwischenzeitlich verliert. Kehrt er aus anderer Richtung wieder zu diesem Punkt zurück, sollte durch den SLAM-Algorithmus sichergestellt sein, dass dies erkannt wird. Diese Eigenschaft ist ein zentrales Qualitätsmerkmal für die Genauigkeit der Kartographierung. Arbeitet das Verfahren unsauber, kommt es bei der Durchquerung solcher Schleifen oft dazu, dass der Anschluss nicht wieder gefunden wird. Die Hauptursache dafür ist die fehlerhafte Bestimmung der Ausrichtung $\Psi$ während des Fluges oder der Fahrt.
Im ersten Experiment wurde der zu durchquerende Raum sehr großzügig angelegt, um den Einfluss der vom Quadrotor selbst verursachten Verwirbelungen zu reduzieren. Alle hier gezeigten Experimente wurden in einer auf dem Campus befindlichen Sporthalle durchgeführt. Die jeweiligen Raumstrukturen wurden mit Stellwänden aufgebaut. Bild 4.15 zeigt den Quadrotor in einer solchen Testumgebung.

Während der Flugversuche wird ein handelsübliches Notebook als Bodenstation zum Monitoring der Daten verwendet. Dieses ist über WLAN

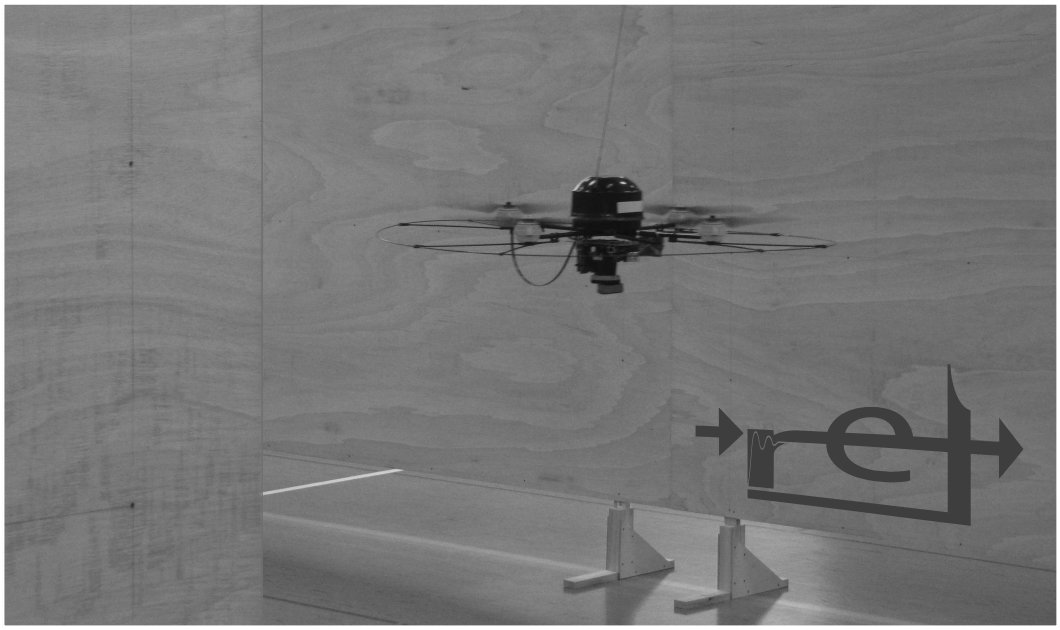


**Bild 4.15:** Quadrotor in der Testumgebung

mit der ICAM verbunden. Die für die Navigationsstrategie erforderlichen Sollwerte werden ebenfalls mittels dieses Computers übertragen. Wie im vorherigen Kapitel 4.1 beschrieben wird eine Schnittstelle unterhalb des Quadrotorkorpus genutzt, um die Steuersignale vom Microcontroller an das Fluggerät zu übermitteln. Zur Sicherheit ist bei allen Experimenten ein Bediener mit Handsender anwesend, der im Notfall eingreifen kann, um so ein gewisses Maß an Sicherheit für das Umfeld und den Flugroboter selbst zu gewährleisten. Das erste so absolvierte Testszenario zeigt Bild 4.16.

Die grauen Umrisslinien markieren den Grundriss. Aufgrund der genannten Navigationsstrategie orientiert sich der im Ursprung gestartete Roboter an den Außenwänden des gezeigten Szenarios. Bei der Betrachtung von Bild 4.16 wird deutlich, dass zwischen Grundriss und der in schwarz eingezeichneten ermittelten Karte sowohl bei der Struktur als auch bei den Abmessungen eine hohe Übereinstimmung herrscht. Es sind nur wenige Linien doppelt vorhanden und das oben angesprochene *Loop closing* ist erfolgreich. Dies ist ein Beleg dafür, dass das zur Abschätzung der Ausrichtung $\Psi$ vorgestellte Verfahren zuverlässig arbeitet. Damit ist gezeigt, dass der SLAM-Algorithmus

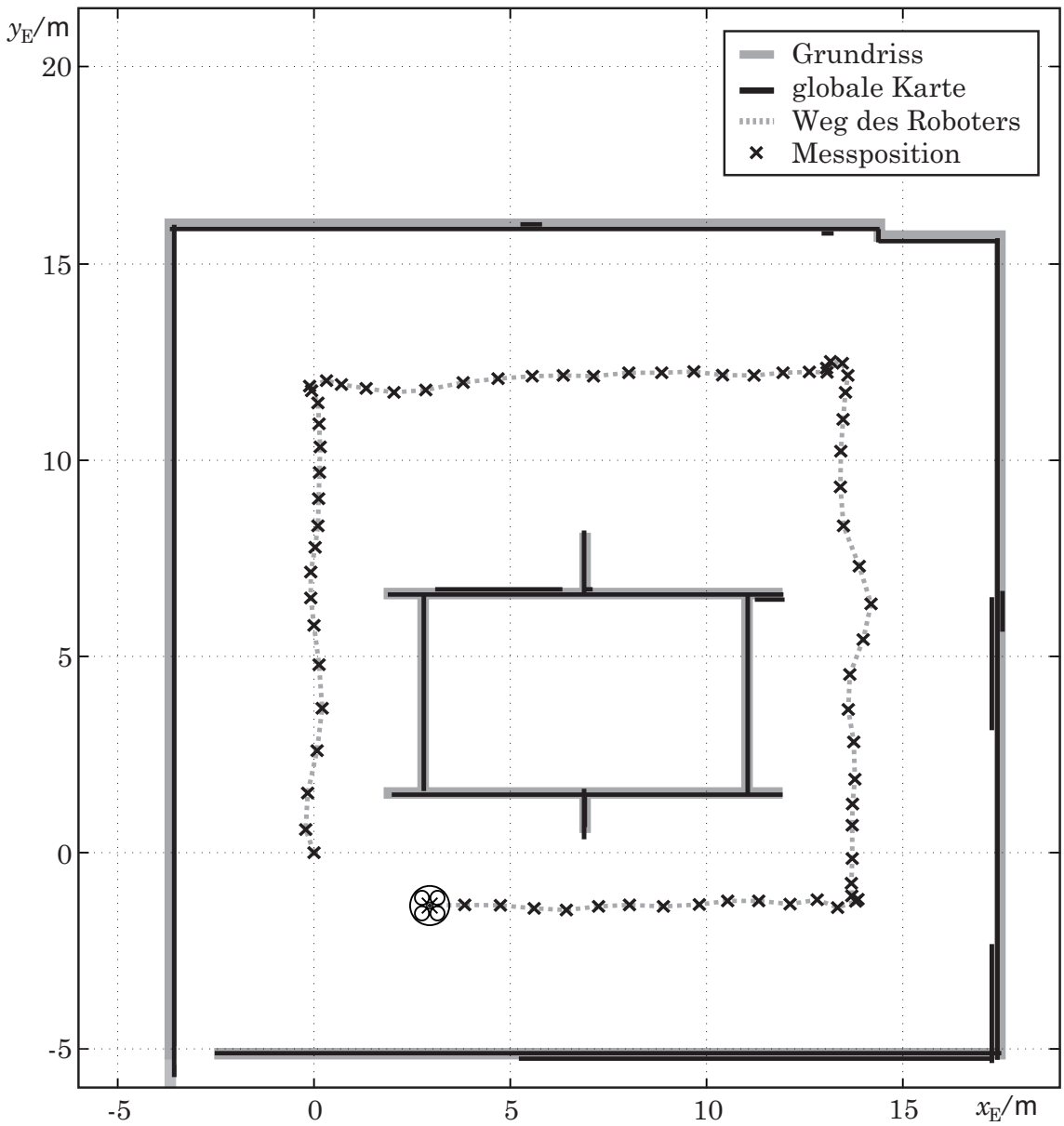



**Bild 4.16:** Grundriss und errechnete Karte mit Positionsdaten des Roboters

in einer Echtzeitanwendung funktioniert und alle dafür erforderlichen Berechnungen mit der vergleichsweise einfachen in der ICAM verbauten Hardware an Bord des Flugroboters realisiert werden können.
Auch die gewählten Regler für den autonomen Flug zeigen die gewünschte Funktion. Dies kann man bei Betrachtung des in Bild 4.16 eingezeichneten Weges des Roboters erkennen. Mit Ausnahme der Ecken, in denen er die kommandierte 90° - Drehung ausführt, hat sich der Quadrotor mit annähernd konstantem Abstand zur in Flugrichtung gesehen linken Wand bewegt. Anhand der eingezeichneten durch den

SLAM-Algorithmus ermittelten Positionen des Roboters, die gleichzeitig die Messpositionen für die Kartographie darstellen, wird zudem deutlich, dass die Geschwindigkeit keinen großen Schwankungen unterliegt. Damit ist gezeigt, dass auf Basis des in Kapitel 3 vorgestellten Verfahrens und der in Abschnitt 4.2 eingeführten ICAM die autonome Erkundung und Kartographierung unbekannter Innenräume durch einen geeigneten Flugroboter möglich ist.

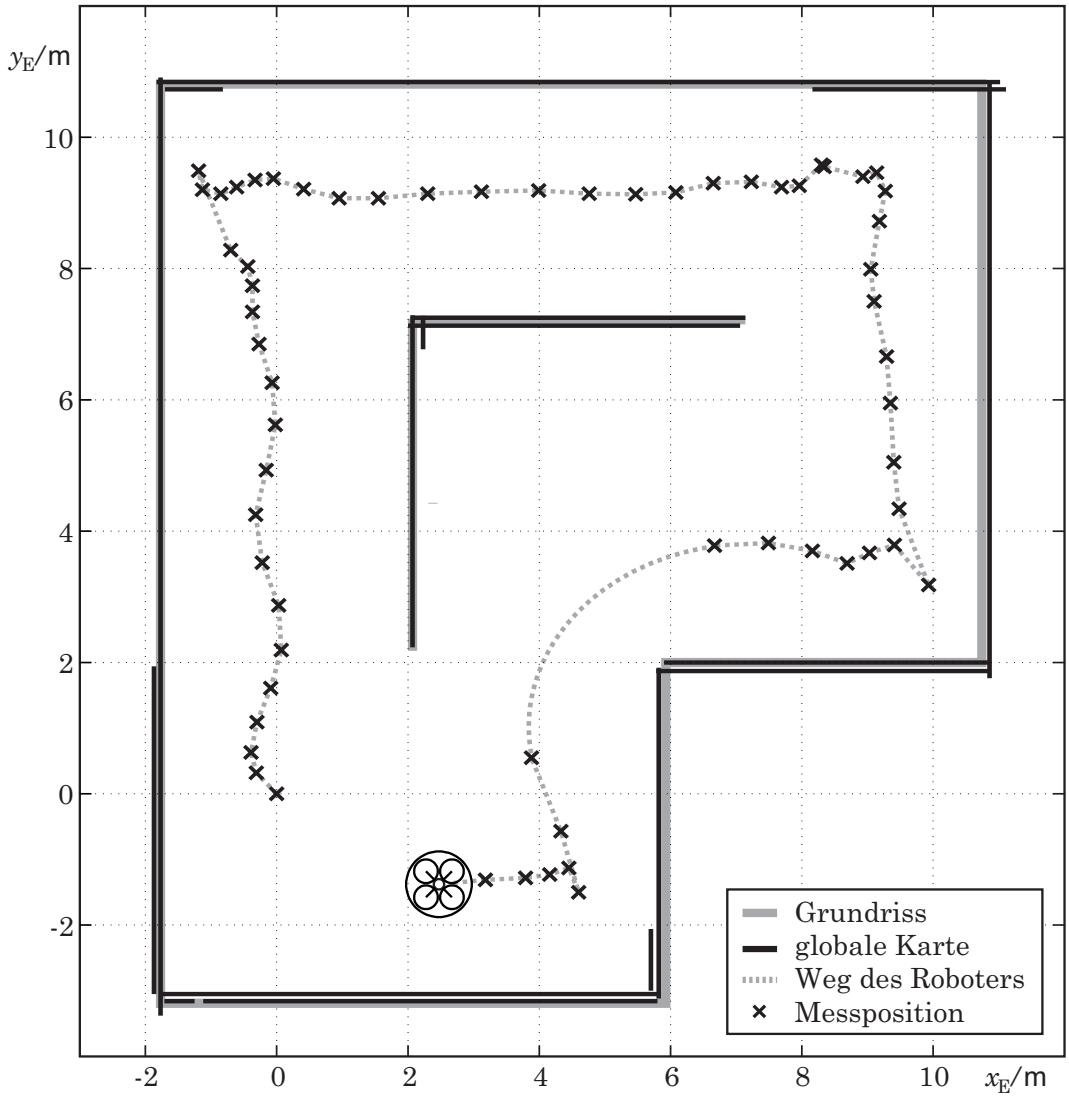



**Bild 4.17:** Grundriss und errechnete Karte mit Positionsdaten des Roboters

Im zweiten Szenario durchfliegt der Quadrotor eine leicht verwinkelte Struktur mit deutlich schmäleren Gängen als im vorherigen Beispiel.

Durch diese Verengung treten deutlich mehr Luftverwirbelungen auf, die das Flugverhalten nachhaltig negativ beeinflussen. Aufgrund dieser Störungen werden durch die Regelalgorithmen stark variierende Stelleingriffe berechnet, die zu einem unruhigen Flugverhalten führen. Bild 4.17 zeigt das Ergebnis für dieses Beispiel.
Anhand der grau gestrichelten Linie, die den Weg des Roboters repräsentiert, werden die oben diskutierten Störeinflüsse deutlich. Besonders zu Beginn der im Ursprung startenden Linie zeigt sich, dass der Abstand zur Wand deutlich stärker schwankt als beim Szenario zuvor. Auch das Flugverhalten in den Ecken ist vergleichsweise unruhiger. Trotz des schlechteren Flugverhaltens wird im Bereich der Kartographierung ein gutes Ergebnis erzielt. Die Zahl der doppelten Linien hat leicht zugenommen, aber die Struktur der zu erkundenden Umgebung wird nach wie vor eindeutig wiedergegeben. Der Vergleich zwischen dem in grau gezeichneten Grundriss und der in schwarz angegebenen globalen Karte zeigt zudem, dass deren Abmessungen ebenfalls stimmig sind. In Bezug auf das Hauptziel dieser Arbeit kann man daher auch bei diesem Ergebnis von einem Erfolg sprechen.

Im dritten und abschließenden Beispiel wird ein komplexeres Szenario betrachtet. Dabei soll der Roboter zunächst einen Gang entlang fliegen. Zweigt ein Raum ab, soll dieser erst untersucht werden, bevor der Flug entlang des Flurs fortgesetzt wird. Dieser Versuch soll zeigen, dass der SLAM-Algorithmus auch in größeren und komplexeren Strukturen funktionsfähig ist. Des Weiteren soll der Quadrotor noch einmal dichter an der Wand entlang fliegen, um den Einfluss der störenden Luftverwirbelungen weiter zu erhöhen und auf ein Maß zu bringen, das auch bei einem Flug in einem realen Bürogebäude zu erwarten ist.

Bild 4.18 zeigt, dass auch in diesem Fall die Struktur der erkundeten Umgebung in der errechneten Karte klar wiedergegeben wird. Auch die Abmessungen stimmen wie in den beiden Szenarien zuvor gut überein. Die Anzahl der doppelten Linien ist erneut leicht erhöht. Für einen Betrachter sind jedoch alle charakteristischen Eigenschaften der untersuchten Umgebung einfach zu erkennen. Anhand der eingezeichneten

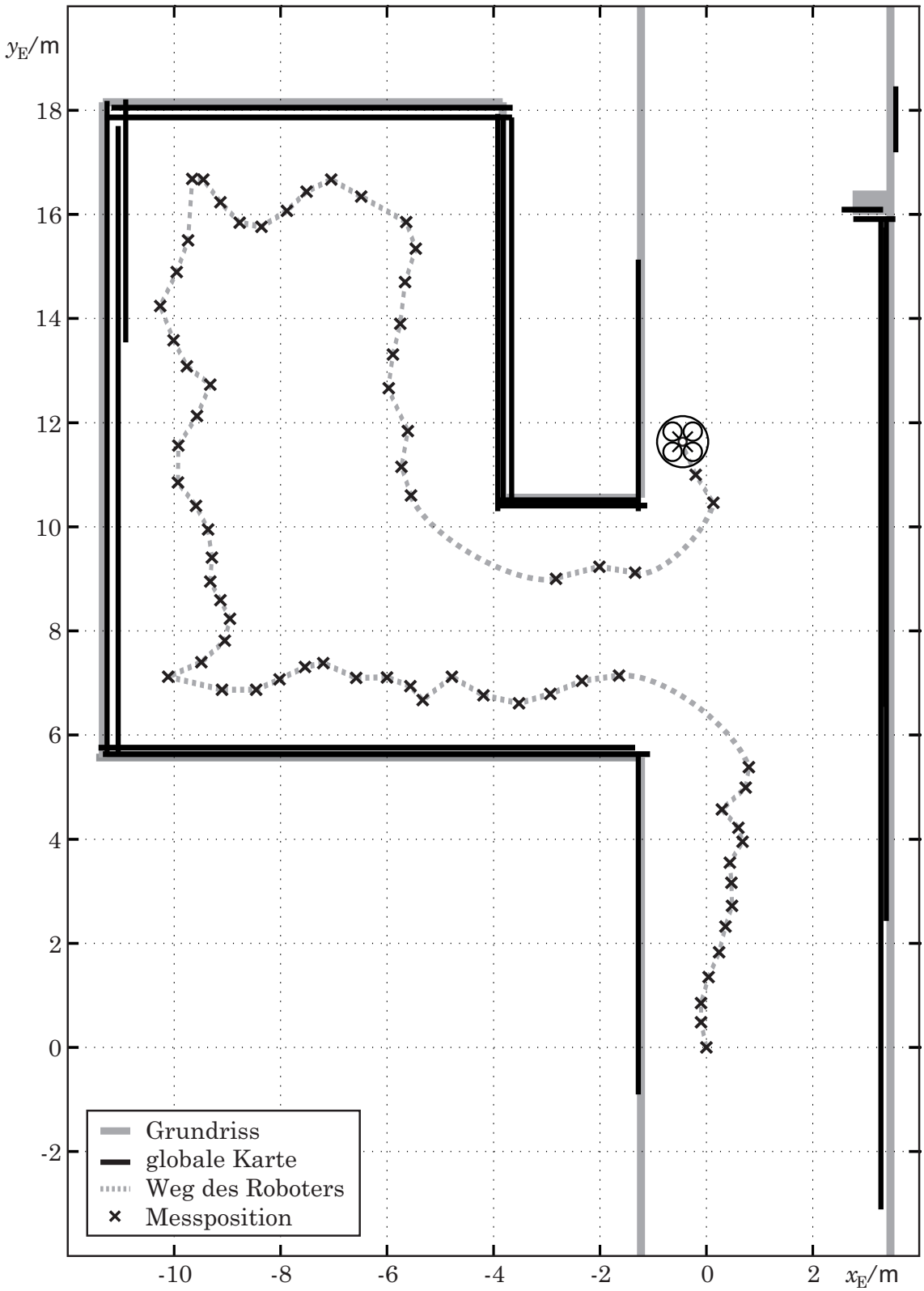



**Bild 4.18**: Grundriss und errechnete Karte mit Positionsdaten des Roboters

Bewegung ist zu erkennen, dass sich der Roboter gemäß der Vorgabe durch das Szenario bewegt hat. Die grau gestrichelte Linie lässt ebenfalls den Schluss zu, dass das Flugverhalten aufgrund der Nähe zur Wand erwartungsgemäß unruhig war. Insgesamt kann aber auch hier von einem erfolgreichen Experiment gesprochen werden.

Ein Versuch in einem realen Büroflur ist aufgrund der Größe und Art der verwendeten Flugplattform nicht sinnvoll. Zum einen ist die Güte der für den autonomen Flug benötigten Regler aufgrund der Unkenntnis über die Struktur und Parameter der herstellerseitigen Lageregelung deutlich eingeschränkt. Zum anderen ist bei einem Durchmesser des Fluggeräts von 1 m die Anzahl der passierbaren Türen in einem solchen Gebäude sehr gering. Daher könnten nur sehr weitläufige Flure abgeflogen werden. Dies würde gegenüber den hier durchgeführten Experimenten kaum einen Informationsgewinn bedeuten.

Die drei hier gezeigten Experimente und die zugehörigen Ergebnisse aus den Bildern 4.16, 4.17 und 4.18 zeigen, dass mit dem vorgestellten Verfahren die autonome Erkundung und Kartographierung von Innenräumen durch einen Flugroboter möglich ist. Damit ist das Ziel der hier vorliegenden Arbeit erreicht.

# 5 Zusammenfassung und Ausblick

Im Rahmen der vorliegenden Arbeit wurde ein Ansatz zur autonomen Erkundung und Kartographierung unbekannter Innenräume durch einen Flugroboter vorgestellt. Die Berechnungen für den zu diesem Zweck entwickelten echtzeitfähigen, auf orthogonalen Linien basierenden SLAM (engl. ***S****imultaneous* ***L****ocalization* ***A****nd* ***M****apping*)-Algorithmus erfolgen dabei ebenso wie die der für den autonomen Flug erforderlichen Regelschleifen an Bord des hier verwendeten und durch eine ICAM (engl. ***I****ntelligent* ***C****ontrol* ***A****nd* ***M****easurement Unit*) erweiterten industriellen Quadrotors. Damit ist während der Mission keine Verbindung zu einer Bodenstation erforderlich. Gerade bei der Aufklärung von Innenräumen befindet sich die zum Roboter gehörende Bodenstation oft außerhalb des untersuchten Gebäudes. Aufgrund der möglichen Störungen der Funkverbindung zu dem eingesetzten Roboter ist es für seine vollständige Autonomie wichtig, dass alle erforderlichen Berechnungen an Bord durchgeführt werden. Diese Neuerung wurde hier erreicht und damit setzt sich diese Arbeit von den bisher veröffentlichten ab.

Kapitel 2 gab einen Überblick über die Thematik der SLAM-Algorithmen und beschrieb den Stand der Technik auf diesem Gebiet zu Beginn der dieser Abhandlung zu Grunde liegenden Forschungsarbeit. Neben den verschiedenen Methoden zur Kartographierung wurden auch die etablierten Scan-Matching-Verfahren dargestellt und im Hinblick auf das im Rahmen dieser Arbeit verfolgte Ziel bewertet. Der dritte Abschnitt befasste sich mit der Navigation von Robotern. Sollen diese Aufgaben selbstständig erfüllen, benötigen sie eine Navigationsstrategie, aus der sich das gewünschte Verhalten ergibt. Die bestehenden Verfahren wurden anhand ihrer Komplexität kategorisiert und bewertet. Aus dem dargestellten Stand der Technik wurde die Notwendigkeit einiger Neuerungen entwickelt, die Hauptbestandteil der vorliegenden Arbeit sind.

Der verwendete echtzeitfähige SLAM-Algorithmus wurde in Kapitel 3 im Detail vorgestellt. Zunächst lag der Fokus auf der Abschätzung der Ausrichtung des Roboters. In Anlehnung an das aus der Literatur bekannte Korrelationsverfahren ist hier eine Methode vorgestellt worden, welche die für diese Arbeit geltende Voraussetzung der überwiegend rechtwinklig strukturierten Umgebung dahingehend ausnutzt, dass nur auf Basis der Winkelhistogramme eine zuverlässige Schätzung der Ausrichtung möglich ist.
Der fünf Schritte beinhaltende SLAM-Algorithmus ist im Anschluss beschrieben worden. Besonderes Merkmal des vorgestellten Verfahrens ist die iterationsfreie Eingliederung einer Messung des im Rahmen dieser Arbeit verwendeten Laserscanners in eine globale Karte, in der die bis dato gesammelten Daten gespeichert sind. In vielen bisher veröffentlichten Abhandlungen werden dazu iterative Verfahren verwendet. Eine abschließende Simulation zeigt, dass der vorgestellte Algorithmus unter idealen Bedingungen funktionstüchtig ist.

Zu Beginn von Kapitel 4 erfolgte die Einführung der verwendeten Flugplattform. Zunächst wurde allgemein das Flugprinzip von Quadrotoren erläutert und dabei auf die Vorteile im Hinblick auf diese Arbeit hingewiesen. Anschließend ist der industrielle Quadrotor AR100B® der Firma AirRobot® vorgestellt und detailliert erläutert worden. Daran schloss sich die Modellierung dieses bereits lagegeregelten Systems an. Im Detail ist das dazu gewählte Vorgehen im Anhang A beschrieben. Das synthetisierte nichtlineare Mehrgrößenmodell bildete die Grundlage für die Auslegung der für den autonomen Flug erforderlichen Regler.

Nach Einführung und Beschreibung der selbst konstruierten Nutzlast ICAM im weiteren Verlauf von Kapitel 4 wurde ein vorbereitendes Experiment beschrieben, welches zur Ermittlung verschiedener Parameter erforderlich war. Bei der Wahl der Variablen wurde darauf geachtet, dass die resultierende Karte möglichst genau und für den Betrachter übersichtlich ist. Des Weiteren war bei der Wahl wichtig, dass der SLAM-Algorithmus eine ausreichende Robustheit gegenüber Messfehlern und Störungen aus dem Flugbetrieb aufweist.

Die Funktionsfähigkeit des gesamten Systems wurde anhand von drei Szenarien untersucht. Die jeweiligen globalen Karten gaben die untersuchte Umgebung struktur- und maßstabsgetreu wieder. Die erzielten Ergebnisse zeigen, dass die autonome Kartographierung von Innenräumen durch einen geeigneten Flugroboter möglich ist und im Hinblick auf die Kartographierung zentrale Anforderungen wie das *Loop Closing* erfüllt werden. Die definierten Ziele und Aufgaben konnten somit erfolgreich umgesetzt und implementiert werden.

Zukünftig sollten auch Linien in die Karte mit aufgenommen werden, die nicht Bestandteil des orthogonalen Rasters sind. Dies erhöht den Speicherbedarf nur minimal, führt aber zu einer genaueren Abbildung der Wirklichkeit. Wichtig ist, ausgehend von dem hier vorgestellten Ansatz zur Eingliederung der aktuellen Messung, dass ausreichend orthogonale Strukturen vorhanden sind, die für diesen wichtigen Schritt innerhalb des SLAM-Algorithmus herangezogen werden können. Aus dem gleichen Grund empfiehlt sich auch die Einteilung in ein bestimmtes Winkelraster. Ließe man alle Winkel zu, würde der Algorithmus nach jetzigem Stand scheitern, da es durch Messfehler und andere Störungen bei den detektierten Linien nahezu immer zu Abweichungen vom orthogonalen Raster kommt.

Soll die Güte der für den autonomen Flug erforderlichen Regler verbessert werden, sollte man anstelle des industriellen Quadrotors eine Variante wählen, bei der alle Details bekannt sind und alle Sensordaten in der maximal möglichen Abtastrate zur Verfügung stehen. In dem Fall kann man auf die in der Literatur vielfältig vorhandenen deutlich leistungsfähigeren Algorithmen zurückgreifen. Damit sollte sich das Flugverhalten gerade in beengten Umgebungen deutlich verbessern.

Durch die Montage einer zusätzlichen Kamera könnten bei der Erkundung wesentlich mehr Informationen gewonnen werden. Mit Hilfe des aus dem SLAM-Algorithmus hervorgehenden Grundrisses, der bei einer bestehenden Funkverbindung ebenso wie die Kamerabilder während der Mission an eine Bodenstation übertragen werden können, wäre der Be-

trachter in der Lage, die Kamerabilder unmittelbar eindeutig einer Position in dem untersuchten Raum oder Gebäude zuzuordnen. Darüber hinaus wäre mit Hilfe bekannter Algorithmen aus der Bildverarbeitung eine Erweiterung auf eine dreidimensionale Kartographierung möglich. Um einen solchen Algorithmus ebenfalls ohne Verwendung einer Bodenstation zu realisieren, müssten allerdings auch die hier verwendeten Microcontroller durch leistungsfähigere Rechner ersetzt werden.

# A Modellierung des Quadrotors

In diesem Kapitel wird die Entwicklung eines nichtlinearen dynamischen Modells des verwendeten industriellen Quadrotors beschrieben. Im ersten Abschnitt wird hierzu kurz das aus der Literatur bekannte Modell eines ungeregelten Systems vorgestellt.
Die Struktur des Modells für den hier verwendeten von Seiten des Herstellers bereits lagegeregelten Quadrotor ergibt sich aus dem Flugprinzip und den daraus resultierenden physikalischen Zusammenhängen. Die Erstellung des Modells erfolgt in zwei Schritten. Der erste beinhaltet die translatorischen und rotatorischen Bewegungen in Bezug auf die Gierachse. Im zweiten Schritt werden dann die translatorischen Bewegungen entlang der $x_{\mathrm{E}}$ - Achse und der $y_{\mathrm{E}}$ - Achse betrachtet.

## A.1 Modellansatz aus der Literatur

Die Dynamik eines Quadrotors lässt sich zum einen auf Basis der Kräfte und Momente beschreiben, die aufgrund des Flugprinzips entstehen. Zum anderen müssen die aerodynamischen Zusammenhänge betrachtet werden. Dabei lassen sich die aerodynamischen Kräfte und Momente aus einer Kombination der Theorie eines idealen scheibenförmigen Aktuators (engl. *momentum theory*) und der Blatt-Elemente-Theorie (engl. *blade element theory*) beschreiben [40].
Ausgehend von der aerodynamischen Betrachtung müssen neben der Schubkraft eines Rotors vier weitere Effekte berücksichtigt werden [16]. Zum einen handelt es sich dabei um die Querkraft (engl. *hub force*), die der Drehung des Rotors entgegen wirkt und um das Schleppmoment (engl. *drag moment*), welches das Moment beschreibt, das zur Überwindung der internen Reibkräfte erforderlich ist. Zum anderen muss das Verdrehungsmoment des Rotors (engl. *rolling moment*), das die unterschiedlichen Auftriebskräfte eines Rotors beschreibt und der gestörte Luftfluss in Bodennähe (engl. *ground effect*) in Betracht

gezogen werden. Basierend auf diesen Effekten und den am Quadrotor auftretenden Massenträgheiten lassen sich sechs Bewegungsgleichungen formulieren, aus denen das Zustandsraummodell entwickelt werden kann [16].
Die zugehörige Zustandsdarstellung der Form $\dot{\boldsymbol{X}} = f(\boldsymbol{X},\boldsymbol{U})$ beinhaltet den Zustandsvektor

$$\boldsymbol{X} = \begin{bmatrix} \Theta_\mathrm{R} & \dot{\Theta}_\mathrm{R} & \Phi_\mathrm{R} & \dot{\Phi}_\mathrm{R} & \Psi_\mathrm{R} & \dot{\Psi}_\mathrm{R} & z_\mathrm{R} & \dot{z}_\mathrm{R} & y_\mathrm{R} & \dot{y}_\mathrm{R} & x_\mathrm{R} & \dot{x}_\mathrm{R} \end{bmatrix}^\mathrm{T} \tag{A.1}$$

und den Vektor der Eingangsgrößen

$$\boldsymbol{U} = [U_1 \;\; U_2 \;\; U_3 \;\; U_4]^\mathrm{T}. \tag{A.2}$$

Dabei lassen sich die Eingangsgrößen wie folgt durch die jeweilige Rotationsgeschwindigkeit $\Omega$ der einzelnen Rotoren, den Schubkoeffizienten $b$ (engl. *thrust coefficient*) und den Strömungswiderstandskoeffizienten $d$ (engl. *drag coefficient*) ausdrücken:

$$U_1 = b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2), \tag{A.3}$$

$$U_2 = b(-\Omega_2^2 + \Omega_4^2), \tag{A.4}$$

$$U_3 = b(\Omega_1^2 - \Omega_3^2), \tag{A.5}$$

$$U_4 = d(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2). \tag{A.6}$$

Basierend auf den angegebenen Zuständen und Eingangsgrößen sowie der Vereinfachung, dass nur kleine Abweichungen vom Schwebeflug betrachtet werden, beschreibt der folgende funktionale Zusammenhang in sehr guter Näherung das dynamische Verhalten eines Quadrotors [16]:

$$\dot{\boldsymbol{X}} = f(\boldsymbol{X},\boldsymbol{U}) = \begin{pmatrix} \dot{\Theta}_{\mathrm{R}} \\ \dot{\Phi}_{\mathrm{R}}\dot{\Psi}_{\mathrm{R}}a_1 + \dot{\Phi}_{\mathrm{R}}a_2\Omega_{\mathrm{r}} + b_1U_2 \\ \dot{\Phi}_{\mathrm{R}} \\ \dot{\Theta}_{\mathrm{R}}\dot{\Psi}_{\mathrm{R}}a_3 + \dot{\Theta}_{\mathrm{R}}a_4\Omega_{\mathrm{r}} + b_2U_3 \\ \dot{\Psi}_{\mathrm{R}} \\ \dot{\Theta}_{\mathrm{R}}\dot{\Phi}_{\mathrm{R}}a_5 + b_3U_4 \\ \dot{z} \\ g - (\cos\Theta\cos\Phi)\frac{1}{m}U_1 \\ \dot{y} \\ c_{\mathrm{y}}\frac{1}{m}U_1 \\ \dot{x} \\ c_{\mathrm{x}}\frac{1}{m}U_1 \end{pmatrix}. \tag{A.7}$$

Die im oberen Bereich von Gleichung (A.7) verwendeten Koeffizienten $a_i$ ergeben sich wie folgt aus der Massenträgheit $J$ des Quadrotorrumpfs und der einzelnen Rotoren:

$$a_1 = \frac{(J_{y_{\mathrm{R}}} - J_{z_{\mathrm{R}}})}{J_{x_{\mathrm{R}}}}, \tag{A.8}$$

$$a_2 = \frac{J_{\mathrm{r}}}{J_{x_{\mathrm{R}}}}, \tag{A.9}$$

$$a_3 = \frac{(J_{z_{\mathrm{R}}} - J_{x_{\mathrm{R}}})}{J_{y_{\mathrm{R}}}}, \tag{A.10}$$

$$a_4 = \frac{J_{\mathrm{r}}}{J_{y_{\mathrm{R}}}}, \tag{A.11}$$

$$a_5 = \frac{(J_{x_{\mathrm{R}}} - J_{y_{\mathrm{R}}})}{J_{z_{\mathrm{R}}}}. \tag{A.12}$$

Bei der Berechnung der Koeffizienten $b_i$ muss zusätzlich noch der Abstand der Antriebsmotoren vom Mittelpunkt des Quadrotors berücksichtigt werden. Dieser Abstand ist für alle vier Antriebsmotoren gleich und wird mit dem Parameter $l$ bezeichnet.

$$b_1 = \frac{l}{J_{x_\mathrm{R}}} \tag{A.13}$$

$$b_2 = \frac{l}{J_{y_\mathrm{R}}} \tag{A.14}$$

$$b_3 = \frac{1}{J_{z_\mathrm{R}}} \tag{A.15}$$

Im unteren Bereich von Gleichung (A.7) wird deutlich, dass neben der Gravitationskonstante $g = 9{,}81\,\mathsf{m/s^2}$ auch die Masse $m$ des Quadrotors von Bedeutung ist. Die beiden verbleibenden Koeffizienten $c_\mathrm{x}$ und $c_\mathrm{y}$ setzen sich wie folgt zusammen:

$$c_\mathrm{x} = \cos\Theta \sin\Phi \cos\Psi + \sin\Theta \sin\Psi, \tag{A.16}$$

$$c_\mathrm{y} = \cos\Theta \sin\Phi \cos\Psi - \sin\Theta \cos\Psi. \tag{A.17}$$

Mit den Gleichungen (A.1) bis (A.17) ist nach [16] das dynamische Modell eines Quadrotors beschrieben. Auf dessen Basis können die für den Flugbetrieb erforderlichen Regler entworfen werden.

## A.2 Translation und Rotation bezogen auf die Gierachse

In diesem Abschnitt wird die Dynamik des geregelten Quadrotors in Bezug auf die $z_\mathrm{R}$ - Achse des roboterfesten Koordinatensystems betrachtet. Zunächst wird der Fokus dabei auf die translatorische Bewegung gelegt, die zu einer Höhenänderung des Roboters führt. Im zweiten Teil liegt das Augenmerk auf der rotatorischen Bewegung, die auch als *Gieren* bezeichnet wird.

*Translation*

Die Analyse der Bewegungen entlang der $z_\mathrm{R}$ - Achse basiert auf der Annahme, dass das Fluggerät waagerecht in der Luft liegt. Damit gilt $\Theta_\mathrm{R} = \Phi_\mathrm{R} = 0$ und die $z_\mathrm{R}$ - Achse fällt mit der $z_\mathrm{E}$ - Achse zusammen.

Aufgrund dieser Voraussetzung können die von dem auf dem Quadrotor befindlichen Höhensensor ermittelten Daten direkt zur Ermittlung des dynamischen Verhaltens verwendet werden.
Auf Basis der Gleichung für eine beschleunigte Bewegung [39] ergibt sich, dass sich das dynamische Verhalten zwischen der Eingangsgröße $u_z$ und der Ausgangsgröße ${}^{\mathrm{R}}z_{\mathrm{E}}$, welche die Position des Roboters auf der $z_{\mathrm{E}}$ - Achse angibt, allgemein durch ein Modell 2. Ordnung beschreiben lässt. Da bei den zur Modellbildung durchgeführten Flugversuchen nur Abweichungen der Flughöhe ausgehend von einem Schwebeflug betrachtet wurden, lassen sich die Anfangsgeschwindigkeit $\dot{z}_0$ und die Ausgangsposition $z_0$ vernachlässigen ($\dot{z}_0 = z_0 = 0$). Die Position auf der $z_{\mathrm{E}}$ - Achse hängt damit vordergründig nur noch von der zugehörigen Beschleunigung ${}^{\mathrm{R}}\ddot{z}_{\mathrm{E}}$ ab. Für die Modellbildung müssen jedoch grundsätzlich auch die Massenträgheit und die Reibung berücksichtigt werden.
Zur Ermittlung des gesuchten dynamischen Modells wurden Flugversuche durchgeführt. Bei diesen wurde aus dem Schwebeflug heraus die für diesen Versuch relevante Eingangsgröße $u_z$ sprungförmig verändert. Die Messdaten des an Bord befindlichen barometrischen Höhensensors wurden durch einen Computer am Boden mit einer Abtastrate von 10 Messungen pro Sekunde aufgezeichnet. Auf diese Weise konnte die Sprungantwort des lagegeregelten Quadrotors aufgenommen werden. Bild A.1 zeigt die Ergebnisse dieser Messungen.

Die Kurvenverläufe in Bild A.1 zeigen die Höhenänderung $\Delta^{\mathrm{R}}z_{\mathrm{E}}$ in Abhängigkeit von der Zeit für verschiedene Steuergrößen. Die Zahlen an den Kurven geben jeweils an, mit wie viel Prozent der maximal möglichen Stellgröße aus dem Intervall [128 255] das System angeregt wurde. Die jeweilige Steuergröße wurde bei allen sechs Versuchen nach einer Sekunde ($\Delta t = 1\,\mathsf{s}$) auf das System gegeben.
Gemäß des Verfahrens zur Identifikation von Integralelementen mit Verzögerung nach [43] wurden die jeweiligen Kurven graphisch ausgewertet, um die Modellparameter zu ermitteln. In erster Näherung lässt sich das dynamische Verhalten durch ein Verzögerungsglied 1. Ordnung, einem sogenannten $\mathrm{PT}_1$ - Glied, beschreiben. Für diese Struktur muss

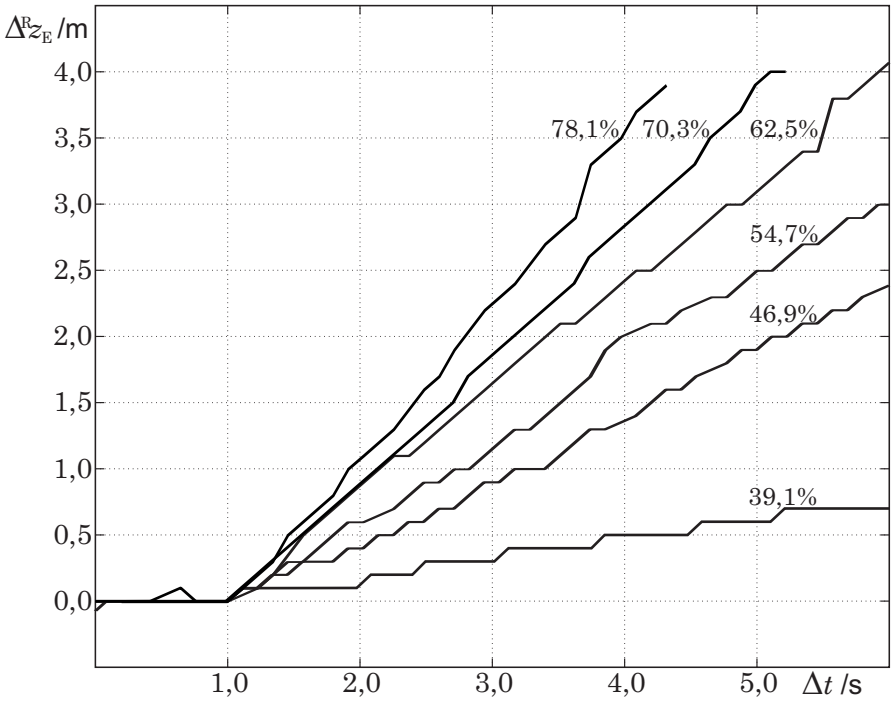



**Bild A.1:** Aufgenommene Sprungantworten für verschiedene Steuergrößen

zum einen die Zeitkonstante $T_{z,\mathrm{R}}$ und zum anderen der Verstärkungsfaktor $K_{z,\mathrm{R}}$ bestimmt werden. Die Vorgehensweise ist am Beispiel der Kurve für 54,7% der maximalen Steuergröße in Bild A.2 gezeigt.
In dem Bereich der Kurve, in dem die Steigung konstant ist, wird eine Gerade angelegt. Diese Gerade ist in Bild A.2 in schwarz dargestellt. Aus der Steigung $\frac{\Delta^{\mathrm{R}} z_{\mathrm{E}}}{\Delta t}$ dieser Geraden ergibt sich die zugehörige Steiggeschwindigkeit ${}^{\mathrm{R}}\dot{z}_{\mathrm{E}}$. Aus dieser und der zugehörigen Eingangsgröße $u_z$ kann der Verstärkungsfaktor $K_{z,\mathrm{R}}$ berechnet werden. Die Differenz $\Delta t$ des Schnittpunktes der eingezeichneten Geraden mit der Nulllinie ($\Delta^{\mathrm{R}} z_{\mathrm{E}} = 0$) und dem Moment des Aufschaltens der Steuergröße $u_z$ zum Zeitpunkt $t = 1\mathsf{s}$ ergibt die Zeitkonstante $T_{z,\mathrm{R}}$. Dies ist im unteren linken Bereich von Bild A.2 zu sehen.

Auf diese Weise wurden alle sechs Kurvenverläufe untersucht und insgesamt sechs Zeitkonstanten und Verstärkungsfaktoren bestimmt. Durch Mittelwertbildung erhält man die beiden gesuchten Größen für die Modellierung. Die Zeitkonstante ergibt sich zu $T_{z,\mathrm{R}} = 0{,}2$ und der Verstärkungsfaktor zu $K_{z,\mathrm{R}} = 0{,}23$. Trägt man die resultierenden

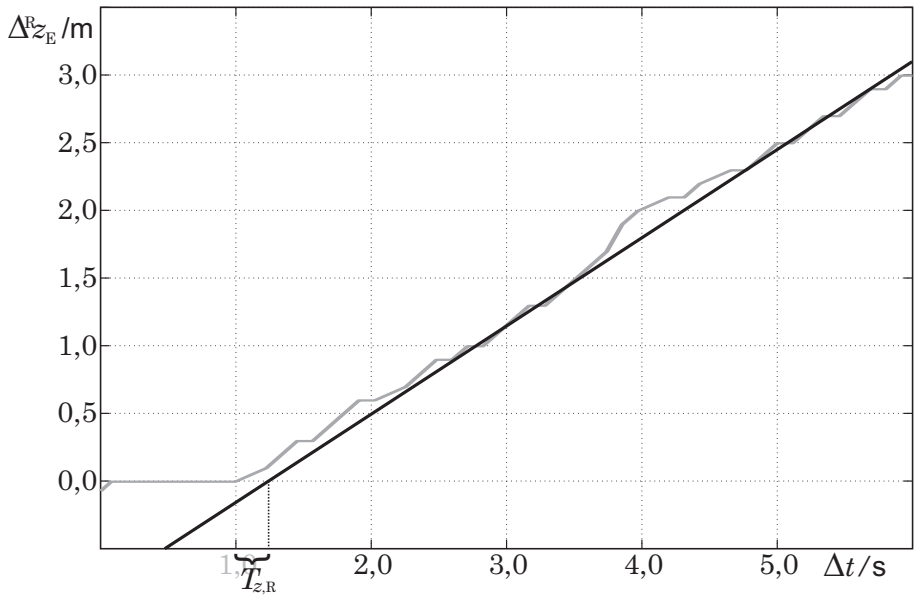



**Bild A.2:** Ermittlung von Zeitkonstante $T_{z,\mathrm{R}}$ und Verstärkungsfaktor $K_{z,\mathrm{R}}$

Steiggeschwindigkeiten ${}^{\mathrm{R}}\dot{z}_{\mathrm{E}}$ über der angelegten Steuergröße $u_z$ auf, so ergibt sich ein annähernd linearer Zusammenhang. Die resultierende ideale Gerade verläuft jedoch nicht durch den Ursprung, sondern schneidet die $y$ - Achse, auf der die Steiggeschwindigkeit aufgetragen ist, erst bei einem Wert von ${}^{\mathrm{R}}\dot{z}_{\mathrm{E}} = -1\,\mathsf{m/s}$. Dieser Effekt ist auf das hohe Gesamtgewicht des Fluggerätes zurück zu führen. Das Gesamtgewicht von Flugplattform und ICAM liegt bei ca. 1,65 kg und ist damit an der oberen Grenze des zulässigen Bereichs. Durch das hohe Gewicht wird der vom Hersteller zur Auslegung des Reglers verwendete Arbeitspunkt nicht eingehalten und somit kommt es zu dem ermittelten nichtlinearen Effekt. Dieser muss bei der späteren Implementierung des Modells in einer Simulationsumgebung berücksichtigt werden. Mit den Nutzlasten des Herstellers wird dieses hohe Gewicht in der Regel nicht erreicht.
Für die spätere Anwendung ist die Flughöhe des Quadrotors von Interesse. Um ein Modell für den Entwurf einer Höhenregelung zu erhalten, muss zusätzlich zu dem ermittelten $\mathrm{PT}_1$ - Glied ein Integrator verwendet werden. Im hier vorliegenden Fall ergibt sich damit die lineare Übertragungsfunktion

$$G_{z,\mathrm{ol}} = \frac{0{,}23}{0{,}2s+1} \cdot \frac{1}{s}, \tag{A.18}$$

die durch eine Totzone zu erweitern ist, damit dem oben genannten nichtlinearen Effekt Rechnung getragen werden kann. Um eine Validierung des messtechnisch ermittelten Modells durchführen zu können, wurde die Übertragungsfunktion des offenen Kreises (engl. ***o**pen **l**oop*) $G_{z,\mathrm{ol}}$ aus Gleichung (A.18) in Matlab-Simulink® umgesetzt. Die Realisierung durch entsprechende Verstärkungs- und Integrationsglieder zeigt Bild A.3. Alle für diese Modellierung relevanten Größen gehen aus diesem Bild hervor.

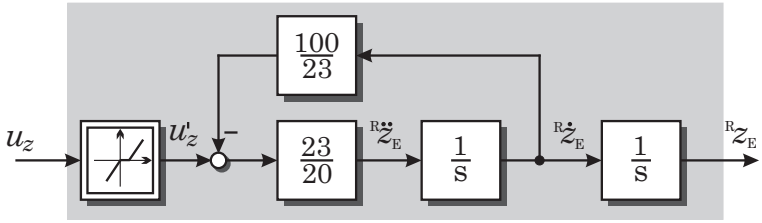



**Bild A.3:** Blockschaltbild des Modells zur Translation entlang der $z_{\mathrm{E}}$ - Achse

Die verwendete Totzone ist nicht symmetrisch zum Ursprung, da der genannte nichtlineare Effekt lediglich das Steigen des Quadrotors betrifft. Aus Messungen ergibt sich die folgende Gleichung:

$$u'_z = \begin{cases} u_z & \text{für} \quad u_z < 128 \\ 0 & \text{für} \quad 128 \geq u_z \leq 171 \\ u_z - 43 & \text{für} \quad u_z < 171. \end{cases} \tag{A.19}$$

Zur Validierung des in Bild A.3 dargestellten Modells wurde dieses für die hier relevanten Stellgrößen analytisch ausgewertet. Bild A.4 stellt die entsprechenden Ergebnisse beginnend mit einer Stellgröße $u_z = 168$ (31,1% des Maximalwertes bezogen auf das Intervall [128 255]) bis zu $u_z = 228$ (78,1% des Maximalwerts) und der oben beschriebenen Messungen gegenüber.

Aus Bild A.4 geht der durch die Totzone verursachte nichlineare Verlauf hervor. Bis auf den Messwert für $u_z = 188$ (46,9% des Maximalwerts) liegen alle Messungen in Bild A.4 im Bereich der bestimmten Gerade.

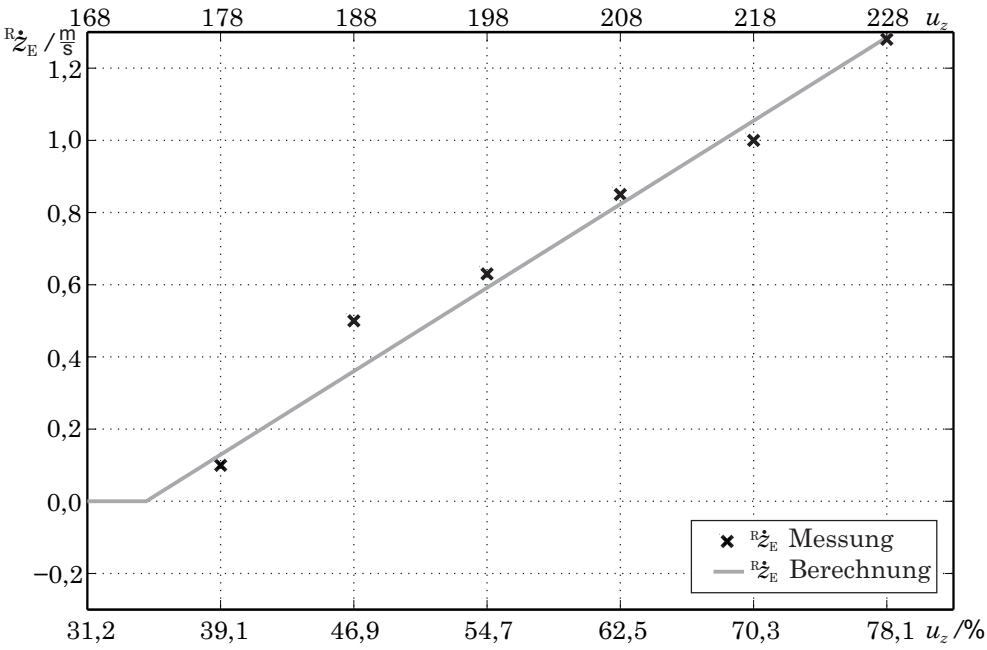



**Bild A.4:** Gegenüberstellung von Messung und Modell

Um eine mathematisch belegte Qualitätsaussage über den Verstärkungsfaktor $K_{z,\mathrm{R}}$ des entwickelten Modells machen zu können, wurde der Korrelationskoeffizient $r_{a,b}$ zwischen Messung und Simulation bestimmt. Hierzu wurde die Gleichung für den Korrelationskoeffizienten

$$r_{a,b} = \frac{\sum_{i=1}^{n}(a_i - \bar{a})(b_i - \bar{b})}{\sqrt{\sum_{i=1}^{n}(a_i - \bar{a})^2 \ \sum_{i=1}^{n}(b_i - \bar{b})^2}} \tag{A.20}$$

für die sechs Stützstellen ausgewertet. Der Parameter $\bar{a}$ bezeichnet dabei den Mittelwert der Simulationsgrößen, der Parameter $\bar{b}$ den Mittelwert der Messwerte. Für $a_i$ und $b_i$ sind jeweils die Werte aus Simulation und Messung an der entsprechenden Stelle einzusetzen. In diesem Fall gilt $n = 6$ und der Korrelationskoeffizient ergibt sich zu $r_{a,b} = 0{,}987$. Gemäß [8] liegt der Wert für eine 100 prozentige lineare Abhängigkeit beider Verläufe bei $r_{a,b} = 1$, bei keiner linearen Abhängigkeit bei $r_{a,b} = 0$. Das hier erzielte Ergebnis ist daher sehr zufriedenstellend und der ermittelte Verstärkungsfaktor $K_{z,\mathrm{R}}$ kann für das Modell verwendet werden.

Um das Modell in Bezug auf die Zeitkonstante $T_{z,\mathrm{R}}$ zu validieren, ist aufgrund der vergleichsweise kleinen Zeitkonstante eine Betrachtung des

geschlossenen Kreises sinnvoll. Stellt man das dynamische Verhalten der Simulation und einer entsprechenden Messung gegenüber und zeigt sich eine gute Übereinstimmung, muss das Modell folglich ein gutes Abbild des realen Systems sein. Mit dem Ziel einer geringen Überschwingweite, einer schnellen Einstellzeit und guter Sollwertfolge wurde auf Basis eines linearisierten Modells ein Regler mit proportionalem und differentiellem Anteil, ein sogenannter PD-Regler, entworfen. Im Anschluss wurde der geschlossene Kreis simuliert und ein entsprechendes Flugexperiment nach dem eingangs dieses Kapitels erwähnten Verfahren durchgeführt. Das Ergebnis zeigt Bild A.5.

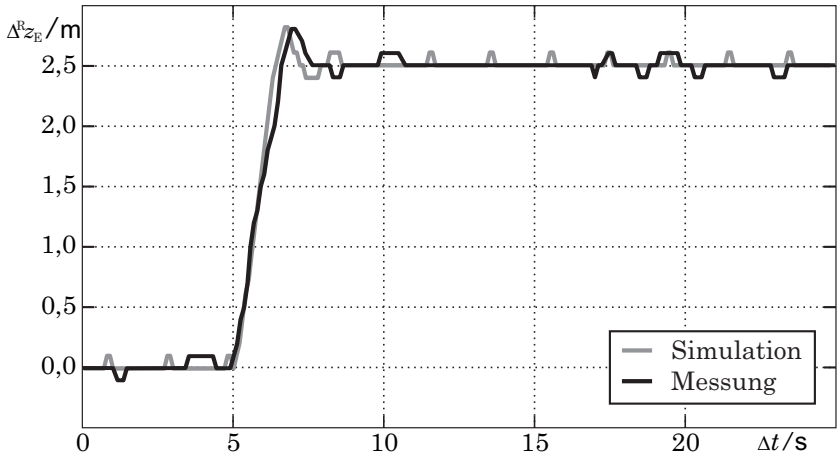



**Bild A.5:** Gegenüberstellung der Sprungantworten im geschlossenen Kreis

Auf das reale System und das Simulationsmodell wurde nach jeweils fünf Sekunden eine Änderung des Höhensollwerts von 1,5 m auf 4,0 m gegeben und die Änderung der Flughöhe über der Zeit aufgenommen. Der Verlauf beider Sprungantworten des geschlossenen Kreises in Bild A.5 zeigt eine gute Übereinstimmung. Das Überschwingen ist in beiden Fällen gleich groß. Der geringe zeitliche Verzug des realen Systems gegenüber der Simulation ist für die spätere Anwendung der Innenraumerkundung vernachlässigbar. Der Vergleich beider Kurven ist ein Beleg dafür, dass das dynamische Verhalten des Modells das reale System abbildet und die Bestimmung der Zeitkonstante $T_{z,\mathrm{R}} = 0{,}2$ eine gute Approximation darstellt. Damit sind die Modellierung und die Validierung der Translation entlang der $z_{\mathrm{E}}$ - Achse erfolgreich abgeschlossen.

*Rotation*

Das Vorgehen zur Modellierung der rotatorischen Bewegungen um die $z_{\mathrm{R}}$ - Achse erfolgt analog zu den translatorischen Bewegungen. So wird auch hier angenommen, dass das Fluggerät waagerecht in der Luft liegt und wiederum die $z_{\mathrm{R}}$ - Achse mit der $z_{\mathrm{E}}$ - Achse zusammen fällt. Die Messwerte des an Bord befindlichen Magnetometers können damit im weiteren Verlauf direkt zur Identifikation der zu bestimmenden Modellparameter verwendet werden. Da es sich auch hier um eine beschleunigte Bewegung handelt, ist auch in diesem Fall ein Modell 2. Ordnung durch geeignete Flugversuche zu ermitteln. Zunächst liegt der Fokus dabei auf der durch die Regelung des Herstellers beeinflussten Drehrate $\dot{\Psi}$. Analog zu den vorherigen Ausführungen wird ausgehend vom Schwebeflug die hier relevante Eingangsgröße $u_{\Psi}$ sprungförmig verändert und die Messdaten des an Bord befindlichen Magnetometers durch den am Boden befindlichen Computer mit einer Abtastrate von 10 Hz aufgezeichnet. Die resultierenden Sprungantworten sind in Bild A.6 dargestellt.

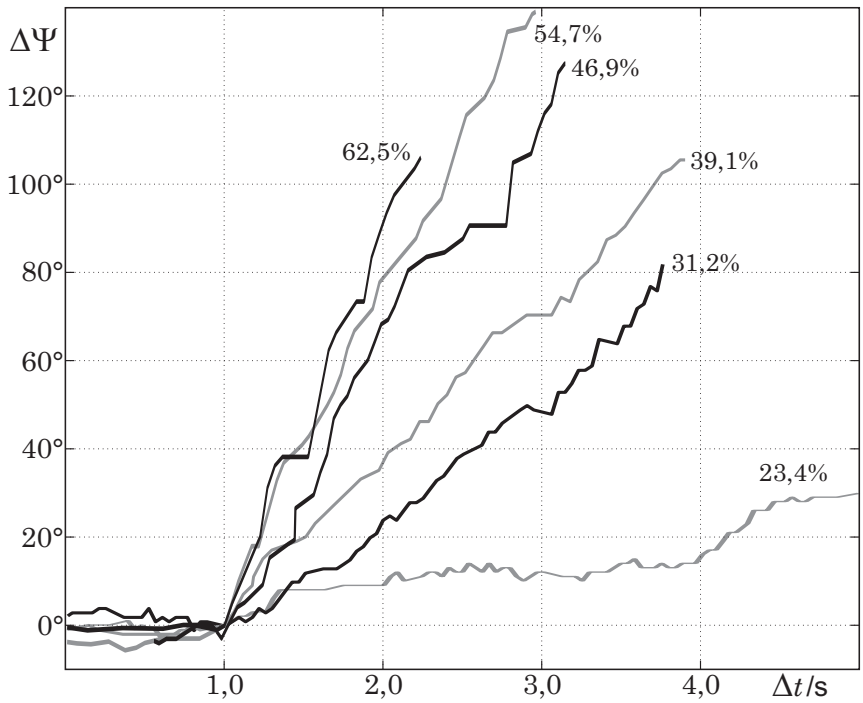



**Bild A.6:** Aufgenommene Sprungantworten für verschiedene Steuergrößen

Die Winkeländerung $\Delta\Psi$ ist in Abhängigkeit von der Zeit für verschiedene Steuergrößen aufgetragen. Die Zahlen an den Kurven in Bild A.6 geben analog zu Bild A.1 an, mit wie viel Prozent der maximal möglichen Stellgröße das System angeregt wurde. Die jeweilige Steuergröße wurde bei allen sechs Versuchen nach einer Sekunde auf das System gegeben.
Die jeweiligen Kurven wurden wiederum graphisch gemäß des Verfahrens zur Identifikation von Integralelementen mit Verzögerung nach [43] ausgewertet. Durch Mittelung der jeweils sechs resultierenden Zeitkonstanten und Verstärkungsfaktoren wurde die Zeitkonstante zu $T_\Psi = 0{,}1$ und der Verstärkungsfaktor zu $K_\Psi = 2$ bestimmt. Trägt man die resultierenden Drehraten $\dot{\Psi}$ über der angelegten Steuergröße $u_\Psi$ auf, so ergibt sich wie bei der Translation eine Gerade, die jedoch erwartungsgemäß auch nicht durch den Ursprung verläuft. Sie schneidet die $x$ - Achse, auf der die Steuergrößen aufgetragen sind, bei einem Wert von 20% der maximalen Steuergröße. Die Ursache dieses nichtlinearen Effekts liegt vermutlich darin, dass der Einfluss kleiner Abweichungen des Steuerknüppels bei der manuellen Bedienung des Quadrotors reduziert werden soll. Links- und Rechtsdrehungen werden dadurch gleichermaßen beeinflusst. Für kleine Stellgrößen wird jeweils keine messbare Drehrate erzielt und daher wird dieser Effekt später durch eine entsprechende Totzone bei der Modellbildung berücksichtigt. Diese wird mit der linearen Übertragungsfunktion

$$G_{\Psi,\mathrm{ol}} = \left(\frac{2}{0{,}1s+1}\right) \cdot \frac{1}{s} \tag{A.21}$$

entsprechend verknüpft. Der nachgeschaltete Integrator trägt der Tatsache Rechnung, dass für die spätere Regelung wiederum der Winkel $\Psi$ selbst und nicht dessen Ableitung relevant ist. Bild A.7 zeigt das Modell, welches zur Validierung des messtechnisch ermittelten Größen in Matlab-Simulink® erstellt wurde. Die Totzone ist im linken Bereich des Bildes zu erkennen. Gleichung (A.22) gibt die zugehörige Definition an. Wiederum gehen alle für diese Modellierung relevanten Größen aus dieser Darstellung hervor.

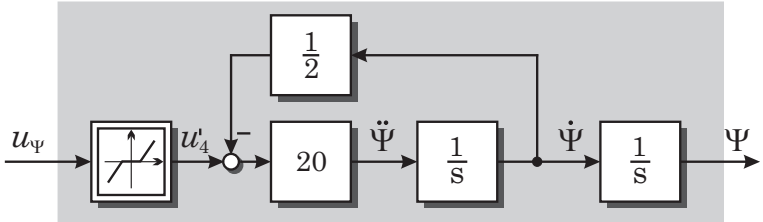



**Bild A.7:** Blockschaltbild des Modells zur Rotation um die $z_R$ - Achse

$$u'_4 = \begin{cases} u_\Psi + 27 & \text{für} \quad u_\Psi < 101 \\ 0 & \text{für} \quad 101 \geq u_\Psi \leq 155 \\ u_\Psi - 27 & \text{für} \quad u_\Psi < 155 \end{cases} \tag{A.22}$$

Zur Validierung des Modells wurde dieses erneut für den gesamten relevanten Bereich der Stellgrößen analytisch ausgewertet. Bild A.8 stellt die Ergebnisse der Berechnungen von einer Stellgröße $u_\Psi = 148$ (15,6% des Maximalwertes bezogen auf das Intervall [128 255]) bis $u_\Psi = 208$ (62,5% des Maximalwerts) und der durchgeführten Messungen gegenüber.

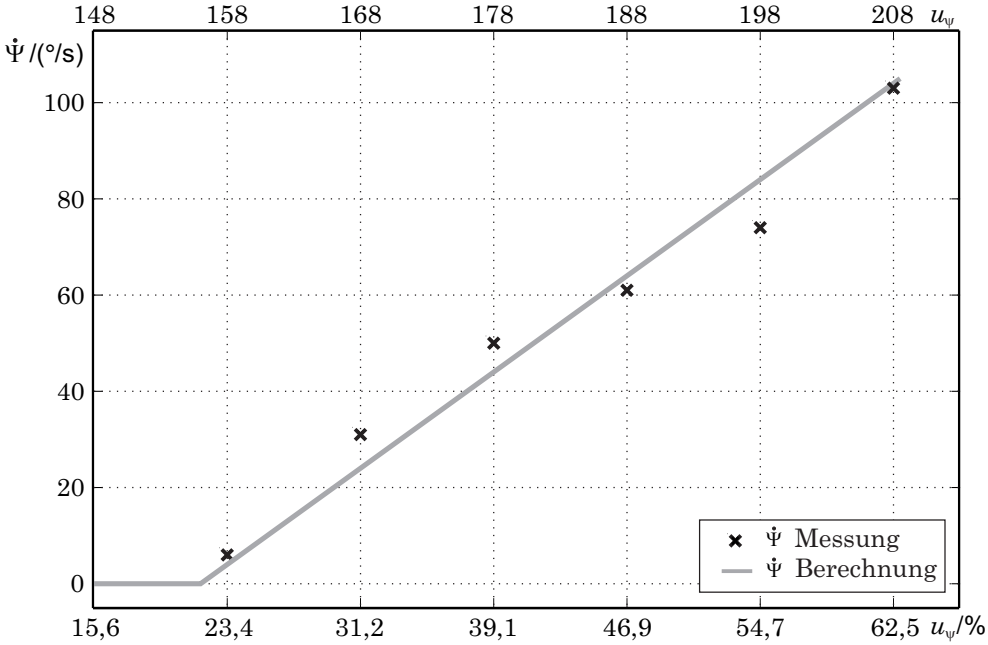



**Bild A.8:** Gegenüberstellung von Messung und Simulation

Aus Bild A.8 geht der durch die Totzone verursachte nichlineare Verlauf hervor, der eine gute Übereinstimmung mit den durchgeführten Messungen zeigt. Um auch hier eine mathematisch belegte Qualitätsaussage über den Verstärkungsfaktor $K_\Psi$ des entwickelten Modells machen zu können, wurde auch in diesem Fall der Korrelationskoeffizient $r_{ab}$ zwischen Messung und dem Ergebnis der analytischen Betrachtung bestimmt. Hierzu wurde die Gleichung (A.20) für die vorliegenden Stützstellen ausgewertet. Der Korrelationskoeffizient ergibt sich zu $r_{ab} = 0{,}989$ und belegt die Vermutung der guten Übereinstimmung. Der ermittelte Verstärkungsfaktor $K_\Psi$ kann für das Modell verwendet werden.
Zur Validierung der Zeitkonstante $T_\Psi$ ist aus den bereits bei der Modellierung der Translation entlang der $z_\mathrm{E}$ - Achse genannten Gründen die Betrachtung des geschlossenen Kreises erforderlich. Zu diesem Zweck ist auch hier ein Regler im Hinblick auf eine geringe Überschwingweite und Einstellzeit sowie einer guten Sollwertfolge entworfen worden. Analog zu den Betrachtungen bei der Translation wurde ein PD-Regler für die Simulationen und die Flugversuche verwendet. Die entsprechenden Ergebnisse sind in Bild A.9 dargestellt.

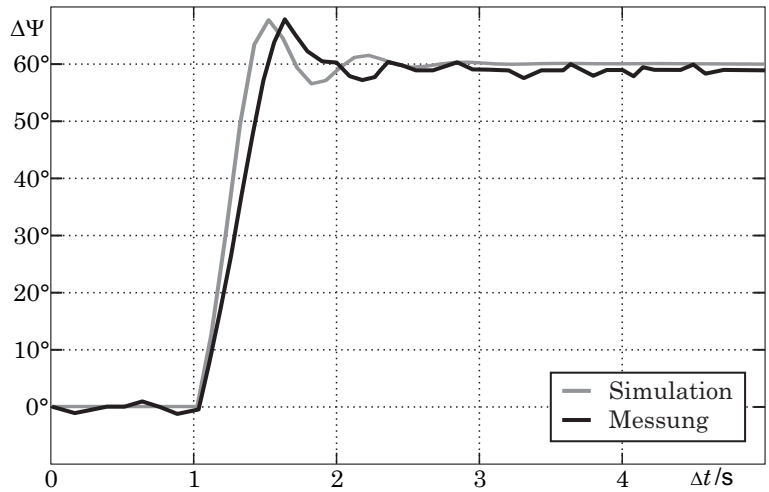



**Bild A.9:** Gegenüberstellung der Sprungantworten im geschlossenen Kreis

In Simulation und Flugversuch wurde jeweils nach einer Sekunde die Führungsgröße $w_\Psi$ sprungförmig um 60 ° geändert. Die resultierenden

Sprungantworten sind in Bild A.9 gegenüber gestellt und weisen eine hohe Übereinstimmung auf. Es ist ein leichter zeitlicher Verzug erkennbar. Weiter ist auffällig, dass der Wert im eingeschwungenen Zustand leicht variiert. Diese beiden Unterschiede sind mit Blick auf die spätere Anwendung jedoch vernachlässigbar. Die hohe Ähnlichkeit beider Kurvenverläufe belegt, dass das dynamische Verhalten des Modells das reale System hinreichend genau abbildet und die bestimmte Zeitkonstante $T_\Psi = 0{,}1$ plausibel ist.
Die Modellbildung der Translation und Rotation in Bezug auf die $z_\mathrm{E}$ - Achse des Fluggeräts ist damit vollständig beschrieben. Im Folgenden werden die translatorischen Bewegungen in der $x_\mathrm{E}$ / $y_\mathrm{E}$ - Ebene betrachtet.

## A.3 Translation in der $x_\mathrm{E}$ / $y_\mathrm{E}$ - Ebene

In diesem Abschnitt wird ein dynamisches Modell des lagegeregelten Quadrotors für die translatorischen Bewegungen in der $x_\mathrm{E}$ / $y_\mathrm{E}$ - Ebene entwickelt. Die erforderlichen Parameter werden zum einen durch Flugversuche und zum anderen durch physikalische Zusammenhänge identifiziert. Ausgangspunkt der Betrachtungen ist wiederum der Schwebeflug des Fluggeräts. Weiterhin wird für die Modellierung angenommen, dass während der translatorischen Bewegungen in der $x_\mathrm{E}$ / $y_\mathrm{E}$ - Ebene keine translatorischen oder rotatorischen Bewegungen bezogen auf die $z_\mathrm{R}$ - Achse erfolgen. Der Quadrotor agiert daher in gleichbleibender Flughöhe und mit konstanter Ausrichtung. Die Ausrichtung ist dabei so gewählt, dass im Schwebeflug $x_\mathrm{E} = x_\mathrm{R}$ und $y_\mathrm{E} = y_\mathrm{R}$ gilt. Ziel ist es, den dynamischen Zusammenhang zwischen den Eingangsgrößen ($u_x$, $u_y$) und der Position der Flugplattform in der betrachteten Ebene zu beschreiben.
Die Eingangsgrößen $u_x$ und $u_y$ werden durch die bereits vorhandene Lageregelung in die Anstellwinkel $\Phi_\mathrm{R}$ und $\Theta_\mathrm{R}$ umgesetzt. Im ersten Schritt wird daher dieser Zusammenhang betrachtet. Da durch die an Bord des Quadrotors befindliche IMU die Anstellwinkel $\Phi_\mathrm{R}$ und $\Theta_\mathrm{R}$ gemessen werden können, ließen sich die benötigten Daten durch Flugversuche ermitteln. Hierzu wurde die im vorherigen Abschnitt bereits beschriebene Ver-

suchsanordnung verwendet. Ausgehend vom Schwebeflug wurde die Eingangsgröße $u_y$ durch den am Boden befindlichen Rechner sprungförmig verändert und der zugehörige Anstellwinkel $\Theta_\mathrm{R}$ aufgezeichnet. Mit Blick auf die spätere Anwendung in Innenräumen wurden vergleichsweise kleine Steuergrößen betrachtet. Für 23,4%, 27,3%, 31,2% und 35,2% des maximalen Werts von $u_y$ aus dem Intervall [128 255] erfolgten jeweils fünf Versuche. Durch die niedrigen Stelleingriffe erfolgte die resultierende Winkeländerung sehr schnell. Ein Zeitverzug zwischen der Änderung der Eingangsgröße und der Einstellung des Anstellwinkels war mit der verwendeten Versuchsanordnung nicht messbar. Mit Blick auf die aus der Änderung des Anstellwinkels $\Theta_\mathrm{R}$ hervorgehenden translatorischen Bewegung entlang der $y_\mathrm{E}$ - Achse kann der Zusammenhang zwischen der Eingangsgröße $u_y$ und dem Anstellwinkel $\Theta_\mathrm{R}$ daher statisch modelliert werden. Nach Mittelung der verschiedenen Messergebnisse und mit Kenntnis des maximalen Anstellwinkels von $\Theta_\mathrm{R} = 25°$ ergibt sich der folgende funktionale Zusammenhang.

$$\Theta_\mathrm{R} = \mathrm{sign}\,(u_y - 128) \cdot (e^{\left(\frac{|u_y - 128|}{100}\right)} - 1) \cdot 9{,}6 \tag{A.23}$$

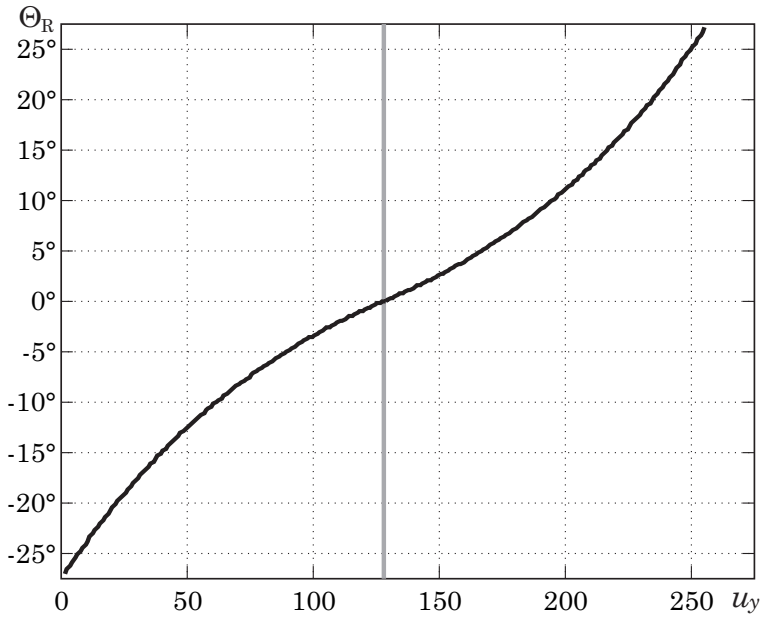



**Bild A.10:** Zusammenhang von Eingangsgröße $u_y$ und Anstellwinkel $\Theta_\mathrm{R}$

Der in Bild A.10 in schwarz dargestellte exponentielle Verlauf des betrachteten Zusammenhangs ist im Bereich des Modellflugs durchaus üblich, um im Bereich der Mittelstellung des Knüppels eine niedrige Empfindlichkeit zu erzielen. Die graue Linie in Bild A.10 markiert die Mitte des Intervalls der Steuergröße $u_y$. Die Subtraktion in Gleichung (A.23) ist erforderlich, da man andernfalls für $e^0$ den Funktionswert 1 erhält. Das Vorzeichen der Eingangsgröße $u_y$ muss hier gesondert betrachtet werden, damit der Kurvenverlauf punktsymmetrisch zum Ursprung ist. Dieser funktionale Zusammenhang lässt sich aufgrund der Symmetrie des Fluggeräts direkt auf die Eingangsgröße $u_x$ und den zugehörigen Anstellwinkel $\Phi_\mathrm{R}$ übertragen.
Die durch die Eingangsgrößen erzeugte Neigung des Quadrotors führt zu einer Vortriebskraft in die Neigungsrichtung und damit zu einer translatorischen Bewegung. Geht man von einer gleichbleibenden Flughöhe aus, so kann die resultierende Vortriebskraft aus der Erdbeschleunigung $g = 9{,}81\mathsf{m/s^2}$ und der Masse des Quadrotors $m$ bestimmt werden. Bild A.11 zeigt beispielhaft die entsprechenden Zusammenhänge für die Bewegung entlang einer Achse.

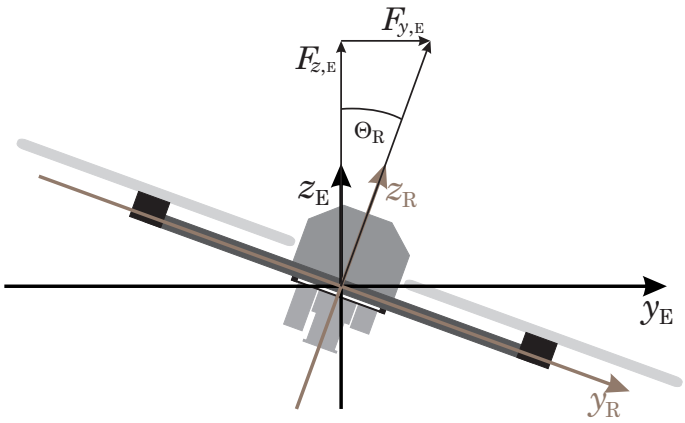



**Bild A.11:** Beispielhafte Verteilung der Kräfte am Quadrotor

Für die in Bild A.11 angegebene Kraft $F_{z,\mathrm{E}}$ gilt im Schwebeflug $F_{z,\mathrm{E}} = m \cdot g$. Durch Gleichung (A.23) lässt sich aus einer Steuergröße $u_y$ der eingezeichnete Winkel $\Theta_\mathrm{R}$ bestimmen. Damit ergibt sich die in Bild A.11 eingezeichnete Vortriebskraft $F_{y,\mathrm{E}}$ zu

$$F_{y,\mathrm{E}} = \tan(\Theta_\mathrm{R}) \cdot F_{z,\mathrm{E}} = \tan(\Theta_\mathrm{R}) \cdot m \cdot g. \tag{A.24}$$

Aufgrund der Symmetrie des Fluggeräts lässt sich der in Gleichung (A.24) dargestellte Zusammenhang wiederum auf die Eingangsgröße $u_x$ und die zugehörige Vortriebskraft $F_{x,\mathrm{R}}$ übertragen. Damit ist der erste Teil der Modellierung abgeschlossen. Aufgrund der Tatsache, dass Kräfte und Beschleunigungen bezogen auf die gleiche Richtung proportional zueinander sind, bildet ein Modellansatz 2. Ordnung die Basis für die weitere Modellbildung, die die dynamischen Zusammenhänge zwischen den Vortriebskräften und der Position des Quadrotors beschreiben sollen.
Der Quadrotor fliegt aufgrund seines Funktionsprinzips, welches in Abschnitt 4.1.1 beschrieben wurde, immer in die Richtung, in die er geneigt ist. Da im Allgemeinen zeitgleich eine Neigung um die $x_\mathrm{R}$ - Achse und die $y_\mathrm{R}$ - Achse auftreten kann und die Dynamiken für die resultierenden Bewegungen maßgeblich über die Luftreibung $F_\mathrm{air}$ verkoppelt sind, ist es für die weitere Modellbildung notwendig, beide Achsen nicht länger unabhängig voneinander zu betrachten. Damit ergibt sich für das zu erstellende Modell eine Mehrgrößenstruktur mit zwei Eingängen $u_x$ und $u_y$ und zwei Ausgängen ${}^\mathrm{R}x_\mathrm{E}$ und ${}^\mathrm{R}y_\mathrm{E}$, die in Bild A.12 dargestellt ist.

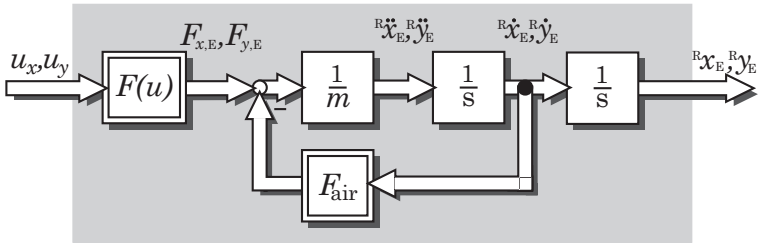



**Bild A.12:** Modellstruktur der Translation in der $x_\mathrm{E} / y_\mathrm{E}$ - Ebene

Die dargestellte Struktur beinhaltet gemäß der gewählten Modellordnung zwei Integratoren pro Koordinate. Der Proportionalitätsfaktor $\frac{1}{m}$ ist erforderlich, um aus den vorliegenden Vortriebskräften $F_{x,\mathrm{E}}$ und $F_{y,\mathrm{E}}$ die entsprechenden translatorischen Beschleunigungen ${}^\mathrm{R}\ddot{x}_\mathrm{E}$ und ${}^\mathrm{R}\ddot{y}_\mathrm{E}$ zu erhalten. Der Funktionsblock $F(u)$ repräsentiert Gleichung

(A.24) und ist durch die doppelte Einrahmung als nichtlinearer Block gekennzeichnet. Gleiches gilt für den zur Luftreibung $F_{\text{air}}$ gehörigen Block. Ausgehend von dem aus der Literatur bekannten Ansatz [29]

$$F_{\text{air}} = \text{sign}\,(v) \cdot c_{\text{W}} \cdot A \cdot \frac{\rho}{2} \cdot v^2 \tag{A.25}$$

wird deutlich, dass die Geschwindigkeit $v$ quadratisch in die Berechnung eingeht und der Zusammenhang zwischen Luftreibung und Geschwindigkeit nichtlinear ist. Im Allgemeinen muss für Gleichung (A.25) die Geschwindigkeit in der Luft verwendet werden. Diese kann sich, beispielsweise durch Einfluss von Wind, von der Geschwindigkeit über Grund unterscheiden. Für die Modellbildung werden ideale Bedingungen voraus gesetzt und daher können hier die Geschwindigkeiten über Grund ${}^{\text{R}}\dot{x}_{\text{E}}$ und ${}^{\text{R}}\dot{y}_{\text{E}}$ zur Bestimmung der Luftreibung verwendet werden. Die Bewegung des Quadrotors kann grundsätzlich in eine beliebige Richtung innerhalb der betrachteten Ebene erfolgen. Die resultierende Geschwindigkeit $v_{\text{air}}$ ergibt sich dabei zu

$$v_{\text{air}} = \sqrt{{}^{\text{R}}\dot{x}_{\text{E}}^2 + {}^{\text{R}}\dot{y}_{\text{E}}^2}. \tag{A.26}$$

Der Parameter $\rho$ in Gleichung (A.25) repräsentiert die Dichte der Luft, die bei einer Temperatur von 15°C gemäß [29] einen Wert von $\rho = 1{,}225\mathsf{kg/m^3}$ beträgt. Die Konstante $c_{\text{W}}$ beschreibt den Widerstandsbeiwert. Dieser dimensionslose Wert ist bei senkrecht zur Strömungsrichtung stehenden Flächen in der Literatur [21] mit $c_{\text{W}} = 1{,}3$ angegeben. Die Variable $A$ repräsentiert diese angeströmten Flächen, die im Folgenden genauer betrachtet werden.
Die angeströmte Fläche $A$ ergibt sich zum einen aus den Abmessungen des Rumpfes und der darunter befindlichen Nutzlast. Dieser Teil wird vereinfacht als Quader mit einer angeströmten Fläche von

$$A_1 = 0{,}21\mathsf{m} \cdot 0{,}14\mathsf{m} = 0.0294\mathsf{m}^2 \tag{A.27}$$

angenommen. Dies ist stark vereinfacht der Anteil, der auf die Flächen parallel zur $z_\mathrm{R}$ - Achse zurückzuführen ist. Da die Flächen der Motorgehäuse und Stangen vergleichsweise klein sind, werden diese hier vernachlässigt. Zum anderen müssen auch die Flächen senkrecht zur $z_\mathrm{R}$ - Achse betrachtet werden. Diese Fläche setzt sich maßgeblich aus den Flächen der vier Rotorblätter sowie der Fläche der Abdeckhaube zusammen und ergibt sich zu

$$A_2 = 0.0754\mathsf{m}^2. \tag{A.28}$$

Durch die Neigung der gesamten Plattform werden beide Flächenanteile jeweils in Abhängigkeit des Anstellwinkels angeströmt. Die Fläche $A_1$ geht mit dem Cosinus, die Fläche $A_2$ mit dem Sinus des resultierenden Neigungswinkels in die Berechnungen ein. Wendet man die bisherigen Angaben auf Gleichung (A.25) an, ergibt sich die Luftreibung für den hier vorliegenden Fall zu

$$|F_\mathrm{air}| = c_\mathrm{W} \cdot (\cos(\alpha) \cdot A_1 + \sin(\alpha) \cdot A_2) \cdot \frac{\rho}{2} \cdot \left({}^\mathrm{R}\dot{x}_\mathrm{E}^2 + {}^\mathrm{R}\dot{y}_\mathrm{E}^2\right). \tag{A.29}$$

In dieser Gleichung wird nur der Betrag der Reibungskraft betrachtet. Das Vorzeichen ist später abhängig von der Flugrichtung und wird bei der Zerlegung in die $x$ - und $y$ - Anteile berücksichtigt. Um Gleichung (A.29) vollständig zu beschreiben, muss noch der resultierende Neigungswinkel $\alpha$ bestimmt werden, der sich einer Überlagerung der beiden Anstellwinkel $\Theta_\mathrm{R}$ und $\Phi_\mathrm{R}$ ergibt. Da eine Transformation von der Reihenfolge der jeweiligen Winkeländerung abhängig ist, wird hier wiederum die Voraussetzung der gleichbleibenden Flughöhe ausgenutzt. In diesem Fall kann der resultierende Neigungswinkel $\alpha$ auf Basis der anliegenden Beschleunigungen ${}^\mathrm{R}\ddot{x}_\mathrm{E}$ und ${}^\mathrm{R}\ddot{y}_\mathrm{E}$ berechnet werden. Analog zu den Betrachtungen zu Bestimmung der Vortriebskräfte $F_{x,\mathrm{E}}$ und $F_{y,\mathrm{E}}$ ergibt sich der resultierende Neigungswinkel zu

$$\alpha = \operatorname{atan}\left(\frac{\sqrt{F_{x,\mathrm{E}}^2 + F_{y,\mathrm{E}}^2}}{F_{z,\mathrm{E}}}\right) = \operatorname{atan}\left(\frac{\sqrt{{}^\mathrm{R}\ddot{x}_\mathrm{E}^2 + {}^\mathrm{R}\ddot{y}_\mathrm{E}^2}}{g}\right). \tag{A.30}$$

Die Wirkrichtung der Luftreibung $F_{\mathrm{air}}$ lässt sich aus der Betrachtung der Geschwindigkeiten ${}^{\mathrm{R}}\dot{x}_{\mathrm{E}}$ und ${}^{\mathrm{R}}\dot{y}_{\mathrm{E}}$ bestimmen. Werden beide Komponenten als Bestandteile einer komplexen Zahl aufgefasst, kann man sie in Betrag und Phase zerlegen. Die Phase entspricht dabei der Flugrichtung, der die Luftreibung $F_{\mathrm{air}}$ entgegen wirkt. Wird diese unter Berücksichtigung der Flugrichtung zerlegt, so erhält man bei den jeweiligen Anteilen $F_{\mathrm{air},x}$ und $F_{\mathrm{air},y}$ das entsprechende Vorzeichen. Damit ist der dynamische Zusammenhang zwischen den aus dem ersten Teil der Modellbildung hervorgegangenen Kräften $F_{x,\mathrm{E}}$ und $F_{y,\mathrm{E}}$ und den resultierenden Geschwindigkeiten ${}^{\mathrm{R}}\dot{x}_{\mathrm{E}}$ und ${}^{\mathrm{R}}\dot{y}_{\mathrm{E}}$ unter den genannten Voraussetzungen vollständig beschrieben. Um die später wichtige Position in der $x_{\mathrm{E}}$ / $y_{\mathrm{E}}$ - Ebene zu erhalten, wird jeweils ein Integrator in das Modell eingeführt. Die Validierung des vorgestellten Modells erfolgt im geschlossenen Kreis. Es werden Ergebnisse aus einem Flugversuch und einer entsprechenden Simulation gegenübergestellt. Für den Flugversuch wurde wiederum auf die in Bild 4.7 auf Seite 60 dargestellte Versuchsanordnung zurückgegriffen. Dabei sind in diesem Fall die Werte des an Bord befindlichen GPS-Empfängers ausgewertet worden. Gemäß den zur Modellierung genannten Voraussetzungen ist der Quadrotor während des Fluges so ausgerichtet, dass er sich parallel zu den Achsen des Weltkoordinatensystems bewegt. Die $x_{\mathrm{R}}$ - Achse fällt dabei mit der Ost-West-Achse zusammen und die $y_{\mathrm{R}}$ - Achse verläuft entsprechend in Nord-Süd-Richtung. Zudem ist die Flughöhe während des Experiments konstant. Aufgrund dieser Voraussetzung können die Messwerte des GPS-Empfängers direkt zur Validierung des Modells verwendet werden.

Um die Validierung des geschlossenen Kreises durchführen zu können, wurde für jede der beiden betrachteten Achsen ein PD-Regler entworfen, der ein geeignetes Führungsübertragungsverhalten sicherstellen soll. Aufgrund der bereits mehrfach angesprochenen Symmetrie der Flugplattform ist die Parametrisierung der Regler für beide Achsen gleich. In dem anschließenden Experiment wurde sowohl auf das Modell als auch auf das reale System eine sprungförmige Änderung der Position von 36 m aufgeschaltet. In Bild A.13 sind die resultierenden Sprungantworten dargestellt.

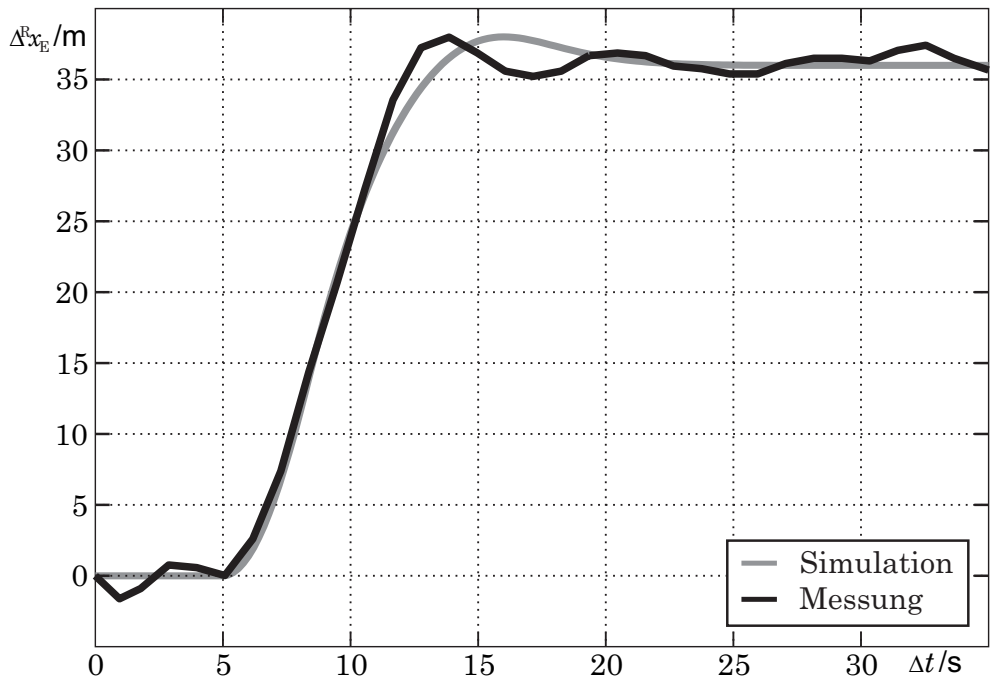



**Bild A.13:** Gegenüberstellung der Sprungantworten im geschlossenen Kreis

Es wird deutlich, dass Modell und reales System in der ersten Phase der Sprungantwort besonders gut übereinstimmen. Im Bereich der Sollwertannäherung weist die in schwarz dargestellte Sprungantwort des realen Systems das Überschwingen etwas früher auf als die der Simulation. Ebenso ist im Bereich der Sollwertfolge bei dem realen System eine deutlich höhere Welligkeit erkennbar. Diese beiden Effekte sind in erster Linie auf die Genauigkeit des GPS-Empfängers zurückzuführen. Diese liegt bei dem verwendeten Gerät bei $\pm 5\,\mathrm{m}$ und wird zusätzlich durch diverse Störungen in der Übertragung überlagert. Ein genauere Beschreibung der möglichen Fehler bei GPS-Empfängern ist in [9] und [59] zu finden. Im Rahmen der zur Verfügung stehenden Genauigkeit der Positionsbestimmung weisen die in Bild A.13 gezeigten Kurvenverläufe eine gute Übereinstimmung auf. Somit bildet das hier hergeleitete Modell das reale System hinreichend genau ab.

# B Intelligente Navigationsstrategie

In diesem Kapitel wird ein Algorithmus zur Erkundung unbekannter Innenräume vorgestellt. Die zentrale Eigenschaft dieser Navigationsstrategie ist, dass der Roboter neugierig bleibt. Auf Basis der durch den parallel laufenden SLAM-Algorithmus generierten Karte werden automatisiert Zielpunkte ermittelt, die einen hohen Informationsgewinn erwarten lassen. Das Erreichen dieser Ziele wird durch eine Überlagerung einer lokalen und einer globalen Navigationsebene erreicht. Das jeweils verwendete Verfahren wird im Folgenden vorgestellt. Im Anschluss wird die Ermittlung der Zielpunkte aus der in der Mission erstellten Karte beschrieben. Abschließend erfolgt eine Validierung der vorgestellten Navigationsstrategie anhand eines Beispiels.

## B.1 Lokale und globale Navigation

Bei der Navigation unterscheidet man in der Literatur grundsätzlich zwischen lokalen und globalen Techniken [51]. Dabei ergibt eine Kombination aus beiden einen Navigationsalgorithmus, der flexibel und zuverlässig in vielen unterschiedlichen Umgebungen arbeitet. In Abschnitt 2.3 wurde dieses bereits in Zusammenhang mit der Planungs- und Auftragsebene diskutiert. Hier wird die Kombination einer vereinfachten Potentialfeldmethode nach [38] und dem A*-Algorithmus [33] beschrieben. Beide Verfahren verwenden dabei die durch den SLAM-Algorithmus generierte metrisch kontinuierliche Karte zur Navigation.

Die Potentialfeldmethode kommt üblicherweise bei gitterbasierten Karten zur Anwendung [38]. Dabei wird jeder Gitterzelle der Karte ein Potential zugeordnet. Diese Verteilung muss differenzierbar sein, da sich durch Bildung eines Gradienten die virtuelle Kraft ergibt, die auf den als geladenes Teilchen modellierten Roboter wirkt. In Bild B.1 ist das Prinzip dieser Methode beispielhaft dargestellt.

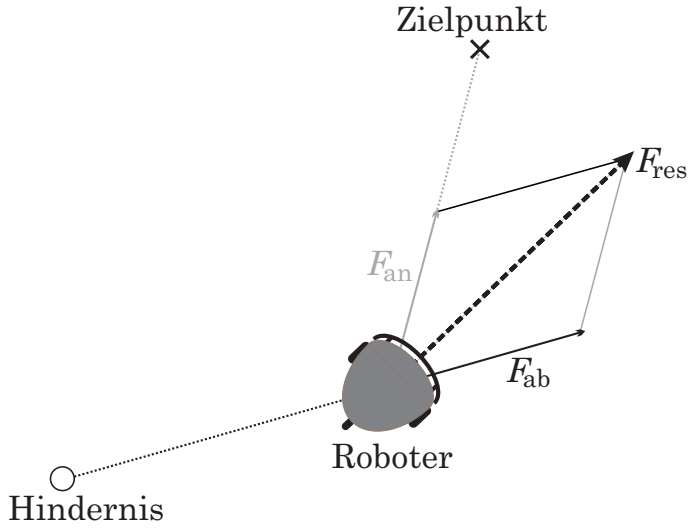



**Bild B.1:** Prinzip der lokalen Navigation [4]

Anhand von Bild B.1 wird deutlich, dass die durch das Hindernis hervorgerufene abstoßende Kraft $\boldsymbol{F}_{\text{ab}}$ vektoriell mit der anziehenden Kraft $\boldsymbol{F}_{\text{an}}$ des Zielpunkts addiert wird. Das Ergebnis ist hier mit $F_{\text{res}}$ bezeichnet und bildet die Basis für die Orientierung des Roboters, der sich bis zum nächsten Abtastzeitpunkt in Richtung des Kraftvektors $\boldsymbol{F}_{\text{res}}$ bewegt. Aus Gründen der Übersichtlichkeit wurde hier auf die Darstellung der einzelnen Gitterzellen mit dem jeweiligen Potential verzichtet.

Da im Rahmen dieser Arbeit eine metrisch kontinuierliche Karte verwendet wird, kann das Verfahren nicht in seiner eigentlichen Form zur Anwendung kommen. Die Bestimmung der virtuellen Kräfte erfolgt hier nicht auf Basis der Berechnung von Gradienten, sondern sie ergeben sich direkt aus den in der Karte enthaltenen Objekten. Dazu werden die bisher kartographierten Wände, die innerhalb des SLAM-Algorithmus als Linie repräsentiert sind, durch äquidistante Punkte $H_i$ dargestellt. Jedem Punkt wird dabei eine repulsive Kraft $\boldsymbol{F}_{H_i}$ auf Basis von Gleichung (B.1) in Abhängigkeit des Abstands zum Roboter zugeordnet.

$$\begin{aligned} F_{H_i,x} &= \frac{1}{k_1}\cos(\beta)\frac{1}{D(H_i)^3} \\ F_{H_i,y} &= \frac{1}{k_1}\sin(\beta)\frac{1}{D(H_i)^3} \end{aligned} \tag{B.1}$$

Die Funktion $D(H_i)$ beschreibt dabei den Euklidischen Abstand $D$ zwischen Roboter und dem Punkt $H_i$ des Hindernisses. Dieser geht in dritter Potenz in die Berechnung ein, um in unmittelbarer Umgebung der Wände ein ausreichend große abstoßende Kraft sicherzustellen. Gleichzeitig nimmt ihr Betrag mit zunehmendem Abstand stark genug ab, um auch eine Navigation in, im Verhältnis zur Größe des Roboters, engen Umgebungen zu ermöglichen. Die Konstante $k_1$ ist ein zusätzlicher Parameter, der in Abhängigkeit der zu erwartenden Umgebung zu wählen ist. Bei weitläufigen Arealen sollte man ihn grundsätzlich höher wählen, damit global gesehen der Einfluss der durch den Zielpunkt hervorgerufenen anziehenden Kraft $\boldsymbol{F}_{\text{an}}$ (vgl. Gleichung (B.3)) größer ist. Die für Gleichung (B.1) relevanten Parameter verdeutlicht Bild B.2.

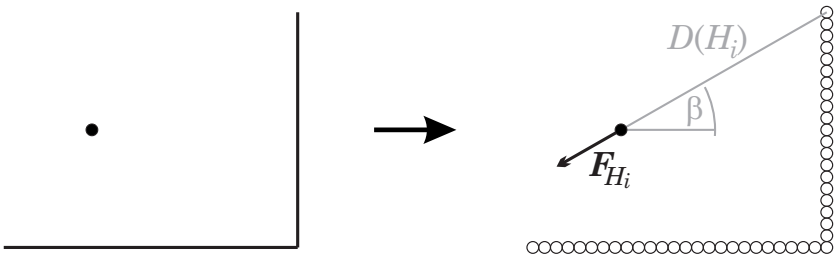



**Bild B.2:** Repräsentation der Hindernisse [4]

Im linken Bereich des Bildes ist die Darstellung der Wände in der Karte gezeigt. Rechts sind die beiden Wände durch die äquidistanten Kreise dargestellt, der schwarze Punkt markiert in beiden Fällen die Position des Roboters. Aus Gründen der Übersicht ist in Bild B.2 nur die resultierende Kraft eines Hindernispunkts $H_i$ eingezeichnet. Die Komponenten der abstoßenden Kraft $\boldsymbol{F}_{\text{ab}}$ ergeben sich schließlich zu

$$\begin{aligned} F_{\text{ab},x} &= \sum F_{H_i,x} \\ F_{\text{ab},y} &= \sum F_{H_i,y}. \end{aligned} \tag{B.2}$$

Mit Hilfe der lokalen Navigation soll der Roboter einen Zielpunkt erreichen. Aus Bild B.1 wird deutlich, dass von diesem eine anziehende Kraft

$\boldsymbol{F}_{\text{an}}$ auf den Roboter ausgeht. Diese ist im gesamten durch die Karte abgedeckten Bereich konstant und ihre Komponenten ergeben sich somit zu

$$\begin{aligned} F_{\text{an},x} &= \cos(\gamma)k_2 \\ F_{\text{an},y} &= \sin(\gamma)k_2. \end{aligned} \tag{B.3}$$

Über die Konstante $k_2$ kann dabei die Intensität der virtuellen Kraft gewählt werden. Wichtig ist, dass über die Parameter $k_1$ und $k_2$ ein geeignetes Verhältnis zwischen anziehender und abstoßender Kraft eingestellt wird. Wählt man $k_2$ zu groß, hat das unter Umständen die Kollision des Roboters mit Hindernissen zur Folge. Wählt man das Verhältnis hingegen zu stark zu Gunsten von $k_1$ so wird der Zielpunkt gegebenenfalls nie erreicht. Hat man keine Informationen über das zu erkundende Areal, sollte $k_1 = k_2 = 1$ gelten.

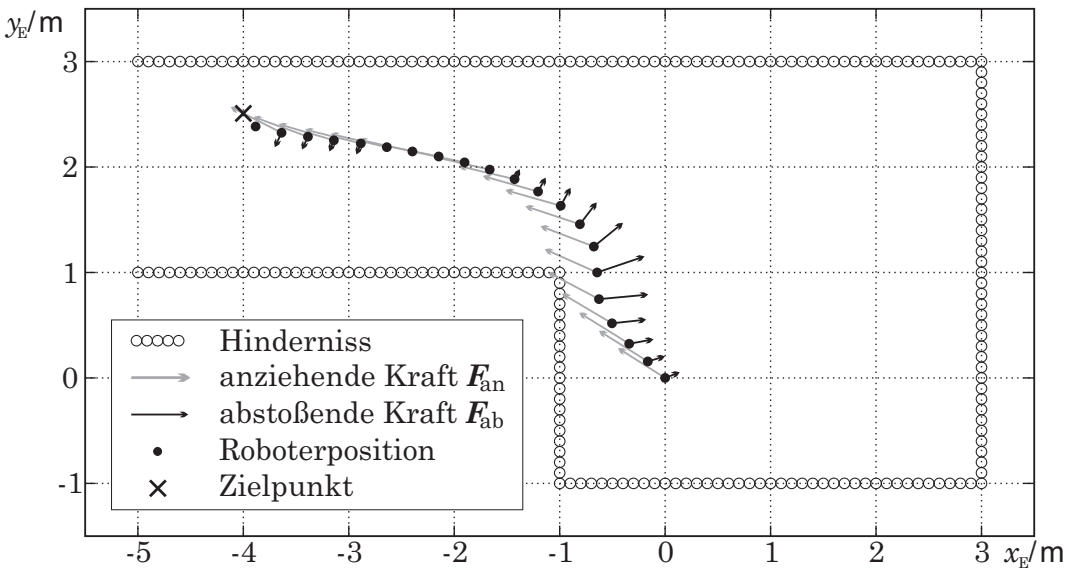



**Bild B.3:** Simulationsergebnis bei lokaler Navigation [53]

Bild B.3 zeigt das Ergebnis einer Simulation, die zur Validierung des vorgestellten Verfahrens dient. Der Roboter startet im Ursprung und soll mittels lokaler Navigation den im linken oberen Bereich eingezeichneten Zielpunkt erreichen. Die bei der jeweiligen Position des Roboters auftretende repulsive Kraft $\boldsymbol{F}_{\text{ab}}$ ist durch schwarze, die attraktive Kraft

$\boldsymbol{F}_{\text{an}}$ durch graue Pfeile repräsentiert. Es zeigt sich, dass sich diese Kräfte gerade im Bereich der Ecke zu einer Fahrt in $y$ - Richtung führen. Nachdem diese passiert ist, schwenkt der Roboter auf einen Kurs ein, der direkt auf den Zielpunkt führt. Betrachtet man die Region des Zielpunkts, so ist zu erkennen, dass dieser nicht exakt erreicht wird. Dies liegt daran, dass er im Wirkungsbereich der abstoßenden Kräfte liegt, die durch die obere Wand hervorgerufen werden. Im Allgemeinen muss daher ein Fangbereich definiert werden, um den Erfolg dieses Verfahrens sicher zu stellen. Des Weiteren ist bei der Wahl des Ziels zu beachten, dass dieses nicht zu dicht an einem Hindernis liegt. Wie die einzelnen Zielpunkte ermittelt werden, wird im Abschnitt B.2 ausführlich beschrieben.
Erweitert man das dargestellte Verfahren um die gesonderte Berücksichtigung des aktuellen Scans, wird zudem erreicht, dass der Roboter auf Hindernisse reagiert, die nicht in der globalen Karte verzeichnet sind. So kann beispielsweise der Scanbereich in mehrere Sektoren aufgeteilt werden. Tritt dann ein Hindernis in einem solchen auf, ist dieser gesperrt und der Roboter muss seinen Kurs entsprechend korrigieren. Diese Art der Navigation ist in Bezug auf die in Abschnitt 2.3 dargestellten Ebenen im Bereich der Reflexe oder Reaktionen einzuordnen.

Die globale Navigation basiert auf einem Sichtbarkeitsgraphen. Ein solcher Graph repräsentiert alle möglichen Verbindungen zwischen Wegpunkten, welche im vorliegenden Fall aus den Ecken der globalen Karte hervorgehen. Die aktuelle Position des Roboters sowie das zu erreichende Ziel stellen ebenfalls einen Wegpunkt dar. In Bild B.4 ist zur Veranschaulichung ein kleines Beispiel mit 5 Wegpunkten angegeben.
Es zeigt sich, dass die in Bild B.4 mit $WP_2$, $WP_3$ und $WP_4$ bezeichneten Wegpunkte aus den grau markierten Ecken der Karte bestimmt werden. Sie werden jeweils in einem konstanten Abstand auf der Seite der Wand gesetzt, die sicher durch den Roboter zu erreichen ist. Die gewählte Distanz zwischen Wegpunkt und Hindernis ist dabei proportional zur Größe des Roboters. Die aktuelle Position wird durch $WP_1$ und das Ziel durch $WP_5$ repräsentiert.

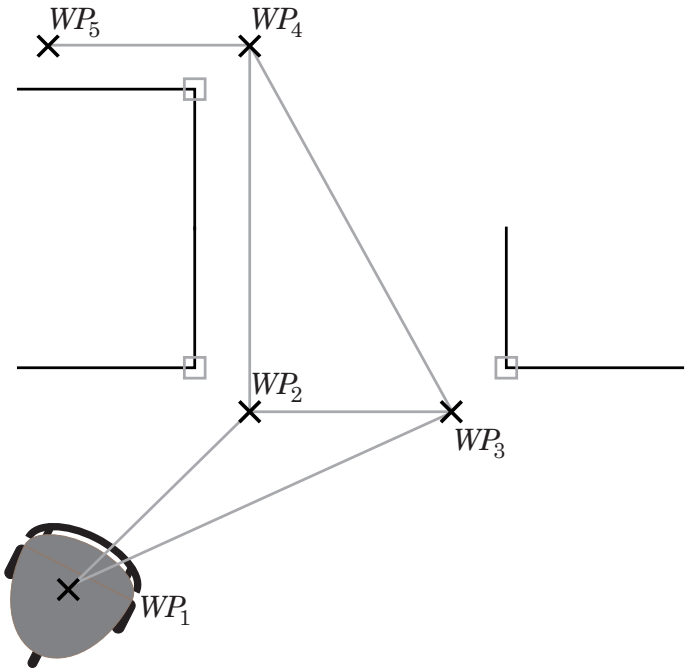



**Bild B.4:** Beispiel für globale Navigation mit 5 Wegpunkten [4]

Die grauen Verbindungslinien stellen den Sichtbarkeitsgraphen dar, in dem mit Hilfe des A*-Algorithmus nach dem kürzesten Weg gesucht wird. Durch dieses Verfahren findet man stets effizient den optimalen Pfad, in der Hinsicht, das die kleinst mögliche Anzahl an Wegpunkten untersucht werden muss, um die kürzeste Strecke zwischen der aktuellen Position und dem Ziel zu finden [33]. Eine ausführliche Beschreibung der Funktionsweise findet sich in der genannten Quelle. In dem in Bild B.4 gezeigten einfachen Beispiel ergibt sich entsprechend der Weg von $WP_1$ über $WP_2$ und $WP_4$ zum Ziel. Die jeweiligen Wegpunkte werden dabei im späteren Versuch durch die oben beschriebene lokale Navigation angefahren, um auf die unmittelbare Umgebung besser reagieren zu können. Liegt das gewählte Ziel so, dass es durch den Roboter nicht erreicht werden kann, weil er durch das in Abschnitt 2.3 beschriebene Kräftegleichgewicht daran gehindert wird, bricht die lokale Navigation ab und es wird gegebenenfalls ein neues Ziel gesucht. Das Zusammenspiel von lokaler und globaler Navigation wird anhand des in Abschnitt B.3 dargestellten Versuchs deutlich.

## B.2 Ermittlung von Zielpunkten

Soll ein unbekanntes Gebiet durch einen Roboter erkundet werden, ist eine Navigationsstrategie erforderlich. Mit dem hier dargestellten Verfahren soll erreicht werden, dass der Roboter neugierig bleibt. In diesem Zusammenhang bedeutet das, dass er sich Ziele sucht, bei denen ein möglichst hoher Informationsgewinn zu erwarten ist. Das entwickelte Verfahren basiert auf der Karte, die durch den orthogonalen SLAM-Algorithmus erstellt wird und auf den folgenden Verhaltensregeln, deren Priorität mit der Reihenfolge der Auflistung einhergeht [4].

1. Erkundung von Gebieten mit einer geringen Dichte von Merkmalen.
2. Vermeidung von Gebieten, die bereits bekannt sind.
3. Untersuchung von Gebieten, die sich nah an der aktuellen Position befinden.
4. Untersuchung von abgelegenen Gebieten erst, wenn die unmittelbare Umgebung erkundet ist.

Die zentrale und zugleich wichtigste Regel ist dabei die erst genannte. Durch sie wird die Neugierde erreicht. Bei den Merkmalen handelt es sich im Rahmen dieser Arbeit um Wände, die bereits in der Karte verzeichnet sind. Mit den übrigen drei Regeln soll in erster Linie vermieden werden, dass der Roboter während der Erkundung zu oft Bereiche durchfährt, die bereits bekannt sind. Gebiete mit geringer Merkmaldichte sind in der Regel an den offenen Enden der vorhandenen Karte zu finden. Im Falle eines langen Flures könnte es ohne diese zusätzlichen Regeln dazu kommen, dass der Roboter ständig von einer Seite des Flurs zur anderen fährt. Dabei würde er den ihm bekannten mittleren Bereich des Flurs sehr häufig passieren und ein signifikanter Anteil der zur Verfügung stehenden Zeit und Energie ohne Informationsgewinn verbraucht werden. Dies gilt es zu vermeiden.
Anhand von Bild B.5 wird gezeigt, nach welchem Verfahren potentielle Ziele auf ihren möglichen Informationsgewinn hin bewertet werden. Es zeigt einen Kartenausschnitt, bei dem die beiden oberen Enden

untersucht werden sollen. Dazu werden zunächst die beiden potentiellen Zielpunkte $Z_1$ und $Z_2$ generiert. Ähnlich wie bei den Wegpunkten bei der globalen Navigation werden diese mit konstantem Abstand zu den offenen Enden der Karte in diese hineingelegt. Die zu wählende Distanz hängt dabei wiederum von der Größe des Roboters ab.

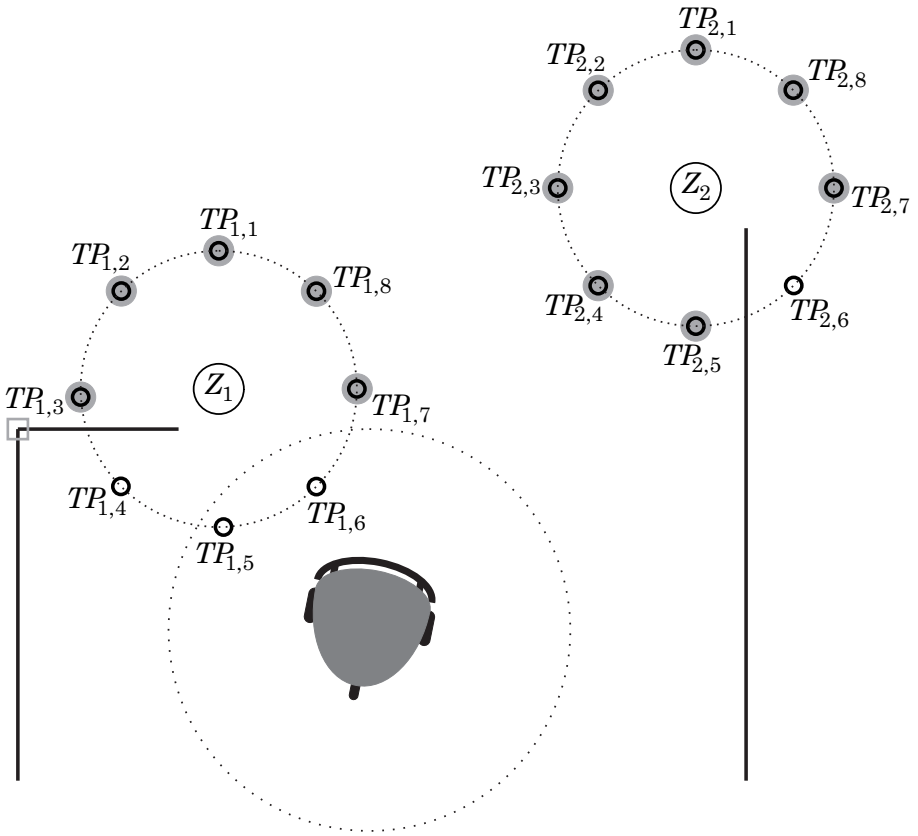



**Bild B.5:** Ermittlung von Zielpunkten [4]

Aus Bild B.5 wird deutlich, dass um die gewählten Zielpunkte $Z_1$ und $Z_2$ mehrere Testpunkte $TP_i$ gelegt werden, die sich auf einer Kreisbahn befinden. Diese ist durch eine gestrichelte Linie dargestellt. Versuche mit dem hier verwendeten Bodenroboter haben gezeigt, dass der für den Kreis gewählte Radius dem zweifachen Durchmesser des Vehikels entprechen sollte. Die Anzahl der verwendeten Testpunkte $TP_i$ hängt mit der Struktur der zu erwartenden Umgebung zusammen. Je stärker ein Raum verwinkelt ist, desto mehr Punkte sind erforderlich, um eine eindeutige Bewertung zu erreichen. In dem hier gezeigten Beispiel werden acht Testpunkte verwendet.

Bei der Bestimmung des möglichen Informationsgewinns eines Zielpunkts $Z_i$ unterscheidet man bei den zugehörigen Testpunkten $TP_{i,j}$ zwischen gültigen und ungültigen. Besteht keine Sichtverbindung zwischen $Z_i$ und $TP_{i,j}$ oder befindet sich der Punkt $TP_{i,j}$ in unmittelbarer Umgebung des Roboters oder des bisher zurückgelegten Weges, ist dieser ungültig. In diesem Fall gilt $e(TP_{i,j}) = 0$. Alle anderen Testpunkte bezeichnet man als gültig und es gilt $e(TP_{i,j}) = 1$. Die zugehörige Berechnungsvorschrift für den betrachteten Zielpunkt lautet:

$$e(Z_i) = \frac{\sum_{j=1}^{8} e(s_{i,j})}{8}. \tag{B.4}$$

Der Nenner in Gleichung (B.4) ergibt sich aus der Anzahl der verwendeten Testpunkte. Bei maximal möglichem Informationsgewinn ergibt sich daher $e(Z_i) = 1$. Wendet man die Gleichung auf das in Bild B.5 dargestellte Beispiel an, ergibt sich die Attraktivität vom ersten Zielpunkt zu $e(Z_1) = 0{,}625$ und die des zweiten zu $e(Z_2) = 0{,}875$. Damit würde der Roboter in diesem Fall den Punkt $Z_2$ ansteuern.
Durch die vorgestellte Form der Bewertung von möglichen Zielen wird den oben aufgeführten Regeln Rechnung getragen. Existieren zwei oder mehr Zielpunkte mit gleicher Attraktivität, wird auf Basis der dritten Vorgabe das nächst gelegene Ziel gewählt. Ob dieses mit lokaler oder globaler Navigation anzufahren ist, hängt von der zu überbrückenden Distanz ab. Im folgenden Abschnitt wird das vorgestellte Bewertungsverfahren anhand eines Testszenarios im Zusammenspiel mit globaler und lokaler Navigation getestet.

## B.3 Validierung durch Experiment

Die Überprüfung der vorgestellten Navigationsstrategie erfolgt durch das hier beschriebene Experiment bei dem ein kleiner wendiger Bodenroboter verwendet wird. Dieser verfügt über zwei in einer Achse angeordnete Antriebsräder, die separat angesteuert werden können. Damit ist er in der Lage, Drehungen auf der Stelle auszuführen. Im Heckbereich ist zudem ein frei laufendes Stützrad montiert.

Die zu untersuchenden Umgebung ist mit Stellwänden in einem Labor aufgebaut worden. Das so realisierte Testareal hat lediglich eine Ausdehnung von wenigen Metern. Um die Strategie in vollem Umfang testen zu können, wurde die Reichweite des verwendeten Laserscanners hier auf 2,5 m begrenzt.

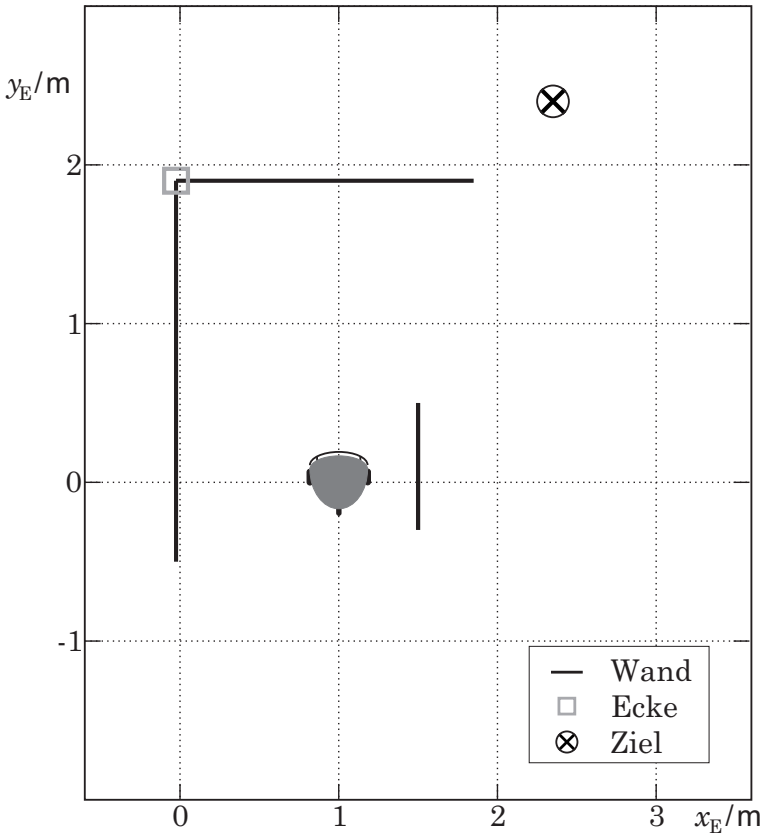



**Bild B.6:** Wahl des ersten Zielpunkts [53]

Zu Beginn des Versuchs scannt der Roboter die Umgebung und verarbeitet die erhaltenen Daten mit Hilfe des im Rahmen dieser Arbeit vorgestellten SLAM-Algorithmus zu einer Karte. Auf deren Basis wird das erste Ziel gewählt. Dabei werden offene Enden der Karte vernachlässigt, die sich in unmittelbarer Nähe des Roboters befinden. In Bild B.6 ist die Situation nach Verarbeitung des ersten Scans dargestellt.
Der in grau maßstabsgetreu dargestellte Roboter befindet sich in der auf einen Scan basierenden Karte im Koordinatenursprung, da die detektierten Strukturen in Relation zu seiner Position eingezeichnet wurden. Die

ermittelten Wände sind durch schwarze Linien, die zugehörigen Ecken mit grauen Kästchen gekennzeichnet. Der ermittelte Zielpunkt ist durch einen Kreis mit einem Kreuz markiert. Als Entfernungsschwelle für den Einsatz der globalen Navigation wurden in diesem Beispiel 3 m verwendet. Das gewählte Ziel liegt unterhalb dieser Entfernung und wird daher mit Hilfe der lokalen Navigation angesteuert. Während der Fahrt werden die neu gewonnen Informationen in der globalen Karte verzeichnet.

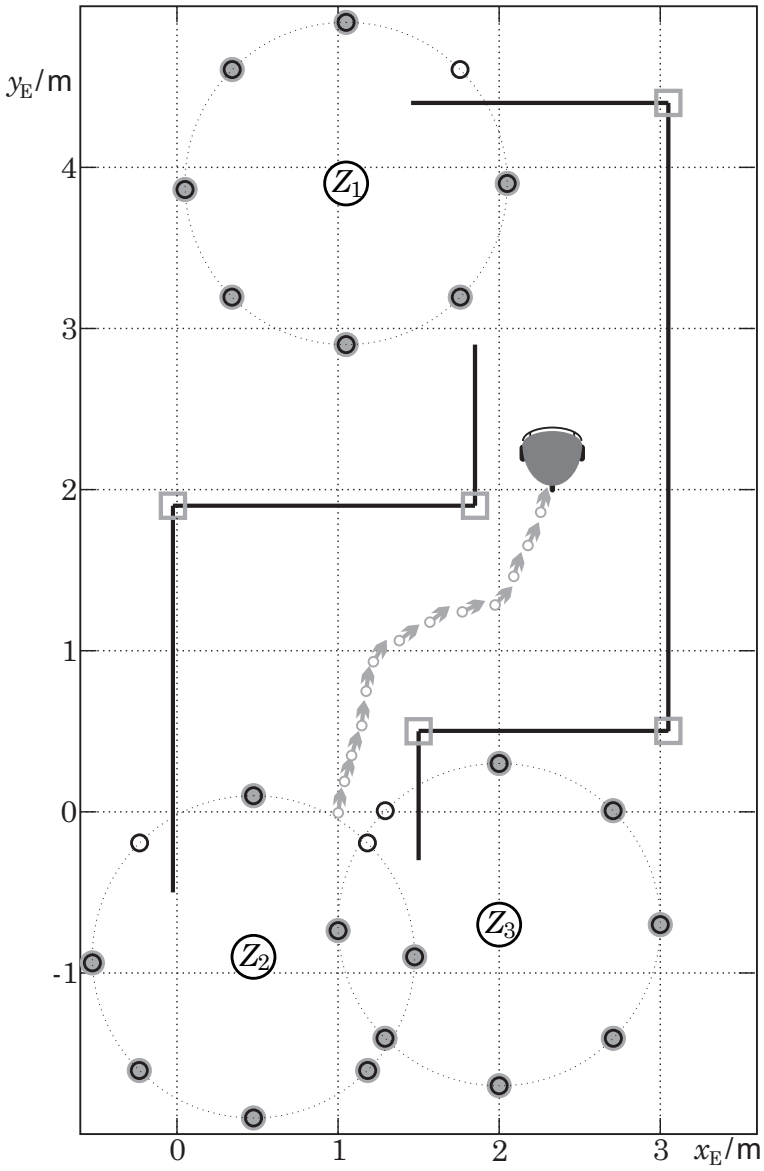



**Bild B.7:** Bewertung potentieller Ziele [53]

Bild B.7 zeigt die Situation bei Erreichen des Ziels. Der bisher zurückgelegte Weg ist in grau markiert. Die Kreise geben die Positionen und die zugehörigen Pfeile die jeweils bestimmte Ausrichtung des Roboters an. Auf Basis der aktuellen globalen Karte, die wiederum durch die schwarzen Linien dargestellt ist, führt der Roboter nun eine Bewertung neuer möglicher Ziele durch. Dazu werden, wie im vorangegangenen Abschnitt B.2 geschildert, an den offenen Enden der bisher bekannten Umgebung potentielle Zielpunkte $Z_1$ bis $Z_3$ gesetzt. An offenen Enden, die sich in unmittelbarer Nähe des Roboters befinden, entfällt dies, um zu häufige Zielpunktberechnungen zu vermeiden. Die zugehörigen Testpunkte $TP_{i,j}$ sind ebenfalls Bestandteil von Bild B.7 und analog zu Abschnitt B.2 sind die gültigen jeweils grau hinterlegt. Aus der Bewertung ergibt sich nach Gleichung (B.4) die Attraktivität vom zweiten Punkt zu $e(Z_2) = 0{,}750$. Für die anderen beiden gilt $e(Z_1) = e(Z_3) = 0{,}875$, da jeweils nur einer der acht Testpunkte ungültig ist. Aufgrund der dritten oben genannten Regel fällt die Wahl hier auf den Punkt $Z_1$, da dieser näher an der aktuellen Position liegt. Erneut setzt sich der Roboter auf Basis der lokalen Navigation in Bewegung und erweitert währenddessen durch die neuen Informationen die globale Karte.

Ist der gewählte Zielpunkt $Z_1$ erreicht, ist die Karte im oberen Bereich geschlossen. Dies wird aus Bild B.8 deutlich. Da sich im unteren Bereich der Karte nichts verändert hat, verbleiben für eine erneute Zielsuche die beiden bereits bekannten Punkte aus der vorhergehenden Abbildung B.7. Gemäß der bereits gezeigten Bewertung fällt die Wahl zunächst auf das Ziel $Z_3$. Dieses ist so weit entfernt, dass die globale Navigation zum Einsatz kommt. Mit Hilfe des in Abschnitt B.1 beschriebenen Verfahrens werden nun Wegpunkte gesetzt und der zugehörige Sichtbarkeitsgraph gebildet. Dabei fällt auf, dass aufgrund der Breite des Roboters der gewählte Zielpunkt, der in Bild B.8 durch $WP_{11}$ gekennzeichnet ist, nicht kollisionsfrei zu erreichen wäre. Die grau gestrichelte Linie zwischen $WP_{10}$ und $WP_{11}$ verläuft zu nah an einer Wand. Somit wird dieses Ziel verworfen und das mit der nächst niedrigeren Attraktivität gewählt. Dies ist hier $Z_2$. Wegpunkte und Sichtbarkeitsgraph werden entsprechend angepasst. Das Resultat bildet der grau gezeichnete Graph mit dem ak-

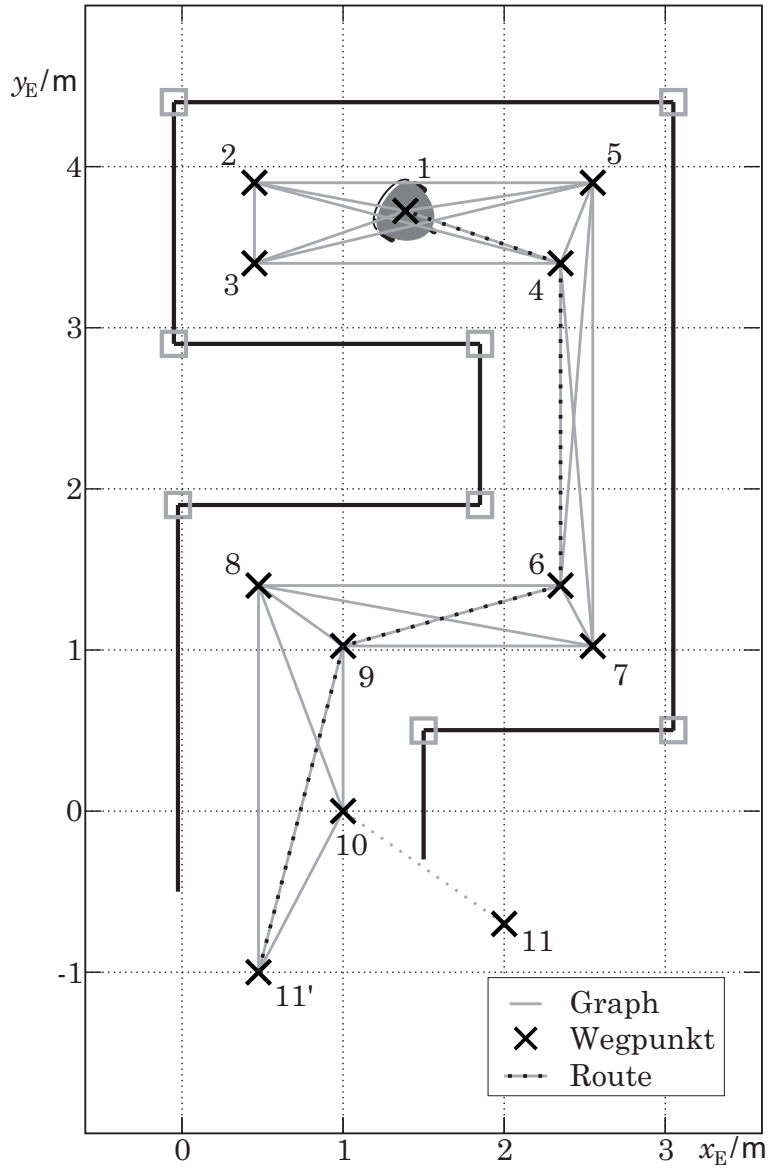



**Bild B.8:** Globale Navigation mit A*-Algorithmus [53]

tualisierten Ziel, welches durch den Wegpunkt $WP_{11'}$ dargestellt ist. Die grauen Linien, die in Bild B.8 schwarz gestrichelt überlagert sind, markieren den mit Hilfe des A*-Algorithmus berechneten kürzesten Weg. Er führt von der aktuellen Position des Roboters über $WP_4$, $WP_6$ und $WP_9$ zum gewählten Ziel $WP_{11'}$. Der Roboter wird auf Basis der lokalen Navigation nun die jeweiligen Wegpunkte nacheinander anfahren, um dieses zu erreichen. Dort angekommen wird die globale Karte geschlossen sein und damit ist die Umgebung vollständig erkundet.

Dieses Experiment zeigt, dass das Zusammenspiel zwischen globaler und lokaler Navigation funktioniert. Des Weiteren wird anhand des gewählten Beispiels deutlich, dass die oben formulierten Regeln zum Verhalten des Roboters in der Praxis eingehalten werden und die vorgestellte Strategie, bei der der Roboter eigenständig Ziele mit hohem möglichen Informationsgewinn generiert, das gewünschte Resultat liefert.

# C Symbolverzeichnis

## C.1 Verwendete Abkürzungen

| | |
|---|---|
| B | Breite |
| CPU | Central Processing Unit |
| EKF | Extended Kalman Filter |
| engl | englisch |
| GPS | Global Positioning System |
| H | Höhe |
| HF | Hochfrequenz |
| ICAM | Intelligent Control And Measurement Unit |
| ICP | Iterative Closest Points |
| IDC | Iterative Dual Correspondence |
| IMR | Iterative Matching Range |
| IMU | Inertial Measurement Unit |
| L | Länge |
| PSM | Polar Scan Matching |
| Quad | Quadrotor |
| SLAM | Simultaneous Localization And Mapping |
| UAV | Unmanned Aerial Vehicle |
| VTOL | Vertical Take-Off and Landing |
| WLAN | Wireless Local Area Network |

## C.2 Lateinische Buchstaben

| | |
|---|---|
| $A$ | Fläche |
| $b$ | Schubkoeffizient |
| $C$ | Punktcluster |
| $d$ | Strömungswiderstandskoeffizient |
| $D$ | Abstand |
| $F$ | Kraft |

| | |
|---|---|
| $g$ | Gravitationskonstante |
| $H$ | Hindernispunkt |
| $J$ | Trägheitsmoment |
| $k$ | allgemeine Konstante |
| $l$ | Länge |
| $L$ | Linie |
| $m$ | Masse |
| $N$ | Anzahl |
| $p$ | Messwert der Entfernungsmessung |
| $P$ | Datensatz der Entfernungsmessung |
| $q$ | Trennvariable bei Segmentierung |
| $S$ | Segment |
| $TP$ | Testpunkt |
| $u$ | Eingangsgröße im Zeitbereich |
| $\boldsymbol{u}$ | Vektor der Eingangsgrößen im Zeitbereich |
| $U$ | Eingangsgröße im Frequenzbereich |
| $\boldsymbol{U}$ | Vektor mit Eingangsgrößen im Frequenzbereich |
| $W$ | Anzahl der Messungen eines Fensters (engl. **W**indow) |
| $WP$ | Wegpunkt |
| $x$ | Achsenbezeichnung |
| $\boldsymbol{X}$ | Zustandsvektor im Frequenzbereich |
| $y$ | Achsenbezeichnung |
| $\boldsymbol{y}$ | Vektor der Ausgangsgrößen im Zeitbereich |
| $Z$ | Zielpunkt |
| $z$ | Achsenbezeichnung |

## C.3 Griechische Buchstaben

| | |
|---|---|
| $\alpha$ | Lagewinkel des Quadrotors |
| $\beta$ | Winkel zur Abstandsberechnung |
| $\gamma$ | Hilfswinkel |
| $\Theta$ | Rotatorische Koordinate |
| $\lambda$ | Wellenlänge |
| $\phi$ | Winkel der Hesseschen Normalform |

| | |
|---|---|
| $\Phi$ | Rotatorische Koordinate |
| $\Psi$ | Rotatorische Koordinate |
| $\Omega$ | Rotationsgeschwindigkeit |
| $\rho$ | Dichte |

## C.4 Tiefgestellte Indizes

| | |
|---|---|
| $\square_{\mathrm{A}}$ | Auftrieb |
| $\square_{\mathrm{ab}}$ | abstoßend |
| $\square_{\mathrm{air}}$ | bezogen auf die Luft (engl. air) |
| $\square_{\mathrm{an}}$ | anziehend |
| $\square_{\mathrm{cl}}$ | geschlossener Regelkreis (engl. **c**losed **l**oop) |
| $\square_{\mathrm{dis}}$ | Störung |
| $\square_{\mathrm{diff}}$ | Differenz |
| $\square_{\mathrm{E}}$ | Erdkoordinate |
| $\square_{E}$ | Anzahl der Messfehler |
| $\square_{i}$ | allgemeine Indexvariable |
| $\square_{j}$ | allgemeine Indexvariable |
| $\square_{L}$ | Linie |
| $\square_{\mathrm{line}}$ | Fluchten von Linien |
| $\square_{M}$ | maximale Anzahl gültiger Messwerte |
| $\square_{\mathrm{max}}$ | Maximalwert |
| $\square_{\mathrm{merge}}$ | Zusammenführung von Linien |
| $\square_{\mathrm{min}}$ | Minimalwert |
| $\square_{N}$ | maximale Anzahl aller Messwerte |
| $\square_{\mathrm{ol}}$ | offener Regelkreis (engl. **o**pen **l**oop) |
| $\square_{\mathrm{R}}$ | Roboterkoordinate |
| $\square_{\mathrm{r}}$ | Rotorblatt |
| $\square_{\mathrm{res}}$ | resultierender Wert |
| $\square_{\mathrm{soll}}$ | Sollwert |
| $\square_{S}$ | Segment |
| $\square_{sm}$ | **s**plit and **m**erge |
| $\square_{\mathrm{T}}$ | Toleranz |
| $\square_{\mathrm{v}}$ | gültig (engl. **v**alid) |

| | |
|---|---|
| $\square_x$ | bezogen auf die zugehörige Koordinatenachse |
| $\square_y$ | bezogen auf die zugehörige Koordinatenachse |
| $\square_z$ | bezogen auf die zugehörige Koordinatenachse |
| $\square_\Psi$ | bezogen auf die Gierachse |

## C.5 Hochgestellte Indizes

| | |
|---|---|
| $\square^{'}$ | Referenzwert |
| ${}^{\mathrm{R}}\square$ | bezogen auf den Roboter |
| $\square^{\mathrm{T}}$ | Transponiert |
| $\dot{\square}$ | erste Ableitung |
| $\ddot{\square}$ | zweite Ableitung |
| $\bar{\square}$ | Mittelwert |

# Literatur


[1] M. Alpen, K. Frick, und J. Horn. *Nonlinear Modeling and Position Control of an Industrial Quadrotor with On-Board Attitude Control.* Proceedings of the IEEE International Conference on Control and Automation (ICCA), I : 2329 - 2334, Christchurch, Neuseeland, 2009.

[2] M. Alpen, K. Frick, und J. Horn. *On the Way to a Real-Time On-Board Orthogonal SLAM for an Indoor UAV.* Proceedings of the 4th International Conference on Intelligent Robotics and Applications (ICIRA), I : 1 - 11, Aachen, Deutschland, 2011.

[3] M. Alpen, K. Frick, und J. Horn. *A Real-Time On-Board Orthogonal SLAM for an Indoor UAV.* Proceedings of the 5th International Conference on Intelligent Robotics and Applications (ICIRA), I : 542 - 551, Montreal, Kanada, 2012.

[4] M. Alpen, S. Stüdemann, und J. Horn. *Time Efficient Strategy to Explore Unknown Indoor Environments by Mobile Ground Robots.* Proceedings of the 3th International Conference on Intelligent Robotics and Applications (ICIRA), II : 215 - 226, Shanghai, China, 2010.

[5] M. Alpen, C. Willrodt, K. Frick, und J. Horn. *On-board SLAM for Indoor UAV Using a Laser Range Finder.* SPIE Defence, Security and Sensing, Orlando, USA, 2010.

[6] S. B. Anderson. *Historical Overview of V/STOL Aircraft Technology.* Moffett Field, Kalifornien, USA, 1981.

[7] T. Bailey und H. Durrant-Whyte. *Simultaneous Localization and Mapping (SLAM): Part II State of the Art.* IEEE Robotics & Automation Magazine, 13(3) : 108 - 117, 2006.

[8] H.-J. Bartsch. *Taschenbuch mathematischer Formeln.* Fachbuchverl. Leipzig im Carl Hanser Verlag, München, 19. Auflage, 2001.

[9] M. Bauer. *Vermessung und Ortung mit Satelliten.* Weichmann-Verlag, Heidelberg, 5. Auflage, 2003.

[10] P. J. Besl und N. D. McKay. *A Method for Registration of 3-D Shapes.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 14(2) : 239 - 255, 1992.

[11] W. Bittner. *Flugmechanik der Hubschrauber: Technologie, das flugdynamische System Hubschrauber, Flugstabilitäten, Steuerbarkeit.* Springer-Verlag, Berlin, 3. Auflage, 2009.

[12] J. Borenstein und Y. Koren. *The Vector Field Histogram - Fast Obstacle Avoidance for Mobile Robots.* IEEE Transactions on Robotics and Automation, 7(3) : 278 - 288, 1991.

[13] G. Bothezat. *The De Bothezat Helicopter.* Flight - Aircraft Engineer & Airships, No. 9, Vol XIV, London, England, 1923.

[14] S. Bouabdallah, P. Murrieri, und R. Siegwart. *Design and Control of an Indoor Micro Quadrotor.* Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 4393 - 4398, New Orleans, USA, 2004.

[15] S. Bouabdallah, A. Noth, und R. Siegwart. *PID vs. LQ Control Techniques Applied to an Indoor Micro Quadrotor.* Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2451 - 2456, Sendai, Japan, 2004.

[16] S. Bouabdallah und R. Siegwart. *Full Control of a Quadrotor.* Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 153 - 158, San Diego, USA, 2007.

[17] P. Castillo, R. Lozano, und A. E. Dzul. *Modelling and Control of Mini-Flying Machines.* Springer-Verlag, London, 2005.

[18] Y.-H Choi, T.-K Lee, und S.-Y Oh. *A Line Feature Based SLAM with Low Grade Range Sensors Using Geometric Constraints and Active Exploration for Mobile Robot.* Autonomous Robots, 24(1) : 13 - 27, 2008.

[19] Y.-H Choi und S.-Y Oh. *Grid-based Visual SLAM in Complex Environment.* Proceedings of the IEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2563 - 2569, Peking, China, 2006.

[20] A. Diosi und L. Kleeman. *Laser Scan Matching in Polar Coordinates with Application to SLAM.* Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 3317 - 3322, Edmonton, Kanada, 2005.

[21] P. Dobrinski, G. Krakau, und A. Vogel. *Physik für Ingenieure.* Teubner-Verlag, Stuttgart, 1993.

[22] H. Durrant-Whyte und T. Bailey. *Simultaneous Localization and Mapping: Part I The Essential Algorithms.* IEEE Robotics & Automation Magazine, 13(2) : 99 - 110, 2006.

[23] A. Eliazar und R. Parr. *DP-SLAM: Fast, Robust Simultaneous Localization and Mapping without Predetermined Landmarks.* Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI), 1135 - 1142, Acapulco, Mexico, 2003.

[24] A. Eliazar und R. Parr. *DP-SLAM 2.0.* Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 1314 - 1320, New Orleans, USA, 2004.

[25] D. Fox, W. Burgard, und S. Thrun. *The Dynamic Window Approach to Collision Avoidance.* IEEE Robotics & Automation Magazine, 4(1 : 23 - 33), 1997.

[26] E. Frenzel und S. Berg. *M167 Microcontroller Board - Datenblatt.* frenzel + berg electronic GmbH & Co. KG, Ulm, 2006.

[27] K. Fujimura. *Motion Planning in Dynamic Environments.* Springer-Verlag, Tokyo, 1992.

[28] T. Geese. *Das fliegende Auge.* Feuerwehr - Retten, Löschen, Bergen, 9: 73 - 75, 2011.

[29] E. Goetsch. *Einführung in die Luftfahrzeugtechnik.* Leuchtturm-Verlag, Alsbach, 1980.

[30] G. Grisetti, C. Stachniss, und W. Burgard. *Improved Techniques for Grid Mapping with Rao-Blackwellized Particle Filters.* IEEE Transactions on Robotics, 23(1) : 34 - 46, 2007.

[31] S. Grzonka, G. Grisetti, und W. Burgard. *Towards a Navigation System for Autonomous Indoor Flying.* Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2878 - 2883, Kobe, Japan, 2009.

[32] D. Hähnel, W. Burgard, D. Fox, und S. Thrun. *An Efficient FastSLAM Algorithm for Generating Maps of Large Scale Cyclic Environments from Raw Laser Measurements.* Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 206 - 211, Las Vegas, USA, 2003.

[33] P. Hart, N. Nilsson, und B. Raphael. *A Formal Basis for the Heuristic Determination of Minimum Cost Paths.* IEEE Transactions on Systems Science and Cybernetics, 4(2): 100 - 107, 1968.

[34] J. Hertzberg, K. Lingemann, und A. Nüchter. *Mobile Roboter: Eine Einführung aus Sicht der Informatik.* eXamen.press. Springer-Verlag, Berlin, Heidelberg, 2012.

[35] G. M. Hoffmann, H. Huang, S. L. Waslander, und C. J. Tomlin. *Quadrotor Helicopter Flight Dynamics and Control - Theory and Experiment.* Proceedings of the AIAA Guidance, Navigation and Control Conference and Exhibit, Hilton Head, USA, 2009.

[36] Hokuyo. *Scanning Laser Range Finder UTM-30LX/LN: Specification.* Hokuyo Automatic CO., L.T.D., Osaka, Japan, 2009.

[37] M. Y. Ibrahim und A. Fernandes. *Study on mobile robot navigation techniques.* Proceedings of the IEEE International Conference on Industrial Technology, 230 - 236, Hammamet, Tunesien, 2004.

[38] O. Khatib. *Real-time Obstacle Avoidance for Manipulators and Mobile Robots.* Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 500 - 505, Stanford, USA, 1985.

[39] H. Kuchling. *Taschenbuch der Physik mit Tabellen.* Fachbuchverl. Leipzig im Carl Hanser Verlag, München, 20. Auflage, 2011.

[40] J. G. Leishman. *Principles of Helicopter Aerodynamics.* Cambridge Univ. Press, New York, USA, 2. Auflage, 2006.

[41] K. Lingemann, H. Surmann, A. Nüchter, und J. Hertzberg. *High-Speed Laser Localization for Mobile Robots.* Robotics and Autonomous Systems, 51(4): 275 - 296, 2005.

[42] F. Lu und E. E. Milios. *Robot Pose Estimation in Unknown Environments by Matching 2D Range Scans.* Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 935 - 938, Seattle, USA, 1994.

[43] H. Lutz und W. Wendt. *Taschenbuch der Regelungstechnik.* Harri Deutsch Verlag, Frankfurt a. M., 7. Auflage, 2007.

[44] O. Meister. *Entwurf und Realisierung einer Aufklärungsplattform auf Basis eines unbemannten Minihelikopters mit autonomen Flugfähigkeiten.* Logos Verlag Berlin GmbH, Berlin, 2010.

[45] M. Montemerlo, S. Thrun, D. Koller, und B. Wegbreit. *FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem.* Proceedings of the AAAI National Conference on Artificial Intelligence, 593 - 598, Edmonton, Kanada, 2002.

[46] G. Neuhaus. *Ein Stück vom Himmel für das Heer: Sachstand und Perspektiven zu Unmanned Aerial Systems des Heeres.* http://www.europaeische-sicherheit.de/Ausgaben/2011/10_2011/03_Neuhaus/2011,10,03.html, 30.04.2012, 2011.

[47] P. Newman, M. Bosse, und J. Leonard. *Autonomous Feature-Based Exploration.* Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 1234 - 1240, Taipei, Taiwan, 2003.

[48] V. Nguyen, A. Harati, A. Martinelli, R. Siegwart, und N. Tomatis. *Orthogonal SLAM: a Step toward Lightweight Indoor Autonomous Navigation.* Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 5007 - 5012, Peking, China, 2006.

[49] N. J. Nilsson. *A Mobile Automation: an Application of Artificial Intelligence Techniques.* Stanford Research Institute Menlo Park, Kalifornien, USA, 1969.

[50] A. Nüchter. *3D robotic mapping.* Springer-Verlag, Berlin, 2009.

[51] S. Rao, S. Iyengar, C. Jorgensen, und C. Weisbin. *Concurrent Algorithms for Autonomous Robot Navigation in an Unexplored Terrain.* Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 1137 - 1144, San Francisco, USA, 1986.

[52] B. Siciliano und O. Khatib. *Springer Handbook of Robotics.* Springer-Verlag, Berlin, 2008.

[53] S. Stüdemann. *Untersuchung zur Bewegungsstrategie autonomer mobiler Roboter in unbekannten Umgebungen am Beispiel des Landroboters Scorpion - Diplomarbeit.* Helmut-Schmidt-Universität / Universität der Bundeswehr Hamburg, Hamburg, 2010.

[54] S. Thrun, W. Burgard, und D. Fox. *Probabilistic Robotics.* MIT Press, Cambridge, Massachusetts, USA, 2006.

[55] S. Thrun und M. Montemerlo. *The GraphSLAM Algorithm with Applications to Large-Scale Mapping of Urban Structures.* International Journal on Robotics Research, 25(5) : 403 - 430, 2006.

[56] K. P. Valavanis. *Advances in Unmanned Aerial Vehicles: State of the Art and the Road to Autonomy*, volume 33 of *Intelligent Systems, Control and Automation.* Springer-Verlag, Dordrecht, Niederlande, 2007.

[57] Z. Wang, S. Huang, und G. Dissanayake. *Simultaneous Localization and Mapping: Exactly Sparse Information Filters*, volume 3 of *New frontiers in robotics.* World Scientific, Singapur, 2011.

[58] G. Weiß, C. Wetzler, und E. v. Puttkamer. *Keeping Track of Position and orientation of Moving Indoor Systems by Correlation of Range-Finder Scans.* München, 1994.

[59] J. Wendel. *Integrierte Navigationssysteme: Sensordatenfusion, GPS und Inertiale Navigation.* Oldenbourg-Verlag, München, 2. Auflage, 2011.

[60] B. Wiggerich. *AR 100B - Operation Instructions.* Airrobot GmbH&Co. KG, Arnsberg, 2011.

[61] C. Willrodt. *Untersuchung von Einsatzmöglichkeiten eines Laserscanners bei der Umgebungserkennung für einen mobilen Roboter - Diplomarbeit.* Helmut-Schmidt-Universität / Universität der Bundeswehr Hamburg, Hamburg, 2009.

**Lebenslauf**

*Persönliche Daten*

| | |
|---|---|
| Name: | Mirco Alpen |
| Geburtsdatum: | 10.01.1978 |
| Geburtsort: | Rendsburg |
| Familienstand: | verheiratet |

*Beruflicher Werdegang*

| | |
|---|---|
| 05/2007 – 09/2013 | Wissenschaftlicher Mitarbeiter der Helmut-Schmidt-Universität / Universität der Bundes-wehr Hamburg |

*Hochschulausbildung*

| | |
|---|---|
| 10/2001 – 02/2007 | Studium der Elektrotechnik und Informationstechnik mit Schwerpunkt Mess- Steuer- und Regelungs-technik an der Technischen Universität Hamburg-Harburg, Abschluss Diplom-Ingenieur |
| 07/2006 – 02/2007 | Diplomarbeit im Bereich Flugzeugsystemtechnik mit dem Thema „Synthese und Modellierung des Regelungs-konzepts eines elektrischen Mehrphasen-motors zur Blattsteuerung von Hubschraubern“ |

*Berufsausbildung*

| | |
|---|---|
| 07/2001 – 09/2001 | Beschäftigung als Industrieelektroniker bei der Hauni-Maschinenbau AG in Hamburg |
| 08/1998 – 06/2001 | Ausbildung zum Industrieelektroniker bei der Hauni-Maschinenbau AG in Hamburg |

*Wehrdienst*

| | |
|---|---|
| 07/1997 – 04/1998 | Grundwehrdienst in Bad Segeberg |

*Schulausbildung*

| | |
|---|---|
| 06/1997 | Abitur am Sachsenwald-Gymnasium Reinbek |