TECHNISCHE UNIVERSITÄT WIEN

ACIN
AUTOMATION & CONTROL INSTITUTE
INSTITUT FÜR AUTOMATISIERUNGS- & REGELUNGSTECHNIK

Autonome Navigation eines Feldroboters

DIPLOMARBEIT

Ausgeführt zum Zwecke der Erlangung des akademischen Grades eines

Diplom-Ingenieurs (Dipl.-Ing.)

unter der Leitung von

Ao. Univ.-Prof. Dipl.-Ing. Dr. techn. M. Vincze
Dipl.-Ing. Dr. techn. J. Prankl
Institut für Automatisierungs- und Regelungstechnik

eingereicht an der

Technischen Universität Wien
Fakultät für Maschinenwesen und Betriebswissenschaften

von

Benedikt Widy, BSc.
Mat.Nr.: 1026787
Kapellenweg 2
2113 Karnabrunn

Wien, im August 2018

Vision for Robotics Gruppe
A-1040 Wien, Gusshausstr. 27, Internet: http://www.acin.tuwien.ac.at

Vorwort

Die Grundlage dieser Diplomarbeit ist ein Forschungsprojekt des Instituts, das eigentlich schon im Jahr 2015 beendet wurde. Die Finanzierung des Projekts „FRANC-Field Robot for Advanced Navigation in bio-Crops“ gelang damals über das Forschungsprogramm Sparkling Science des Bundesministeriums für Bildung, Wissenschaft und Forschung. Während des Besuchs einer Lehrveranstaltung des Instituts wurde ich zufällig und erst nach Abschluss des Projekts auf noch ausgeschriebene Diplomarbeiten aufmerksam.

Ich möchte meinen Betreuern danken, dass sie das Einverständnis zu dieser Diplomarbeit gaben, obwohl das gegenständliche Projekt eigentlich bereits beendet war. Trotz meines Maschinenbaustudiums entschied ich mich aus ideologischen Gründen zu einer fakultätsfremden und fachfernen Abschlussarbeit, weil mir die biologische Landwirtschaft und damit auch ihre Weiterentwicklung persönlich sehr am Herzen liegt. Im Bewusstsein, mir viel Grundwissen zur Thematik erst aneignen zu müssen, begann ich die Arbeit im Herbst 2016 mit großem Optimismus. Rückblickend muss ich eingestehen, die Tragweite sicher unterschätzt zu haben. Ich möchte an dieser Stelle meiner Familie, allen voran meiner Freundin Tanja und meinen Eltern, danken, dass sie mich in Zeiten großen Frustes mit ihrer Liebe unterstützt haben.

Wien, im August 2018

Abstract

This work is based on the research project „FRANC-Field Robot for Advanced Navigation in bio-Crops“ which actually had already been finished. The first part of this thesis aims to evaluate, whether the navigation tools of the software-framework, which was designated for the prototype built in the course of the project, are suitable for the robot. A simulation of a function model imitating the kinematic behaviour required by the navigation tools, showed, that they are basically applicable for FRANC. Apart from this, the fusion of selected, cheap sensors was the subject of a closer examination to estimate their applicability for navigation purposes. Therefore an extended kalman filter was used for the fusion of a inertial measurement unit with a magnetometer, GPS and odometry. In addition another kalman-filter, for calculating the pose via the fusion of gyroscope, magnetometer and accelerometer, was developed, which firmly incorporates the occurrence of external accelerations. This was also done with respect to prospective development steps, where the navigation on three-dimensional, uneven terrain is going to play a major role.

Kurzzusammenfassung

Diese Arbeit baut auf dem eigentlich bereits abgeschlossenen Forschungsprojekt „FRANC-Field Robot for Advanced Navigation in bio-Crops“ auf. Für den im Zuge dieses Projektes entwickelten Prototypen eines Feldroboters sollte in einem ersten Schritt die Eignung der Navigationswerkzeuge, des für den Roboter vorgesehenen Software-Frameworks, evaluiert werden. Durch eine Simulation, in der die von diesen Tools geforderte Roboterkinematik durch ein Funktionsmodell imitiert worden war, wurde gezeigt, dass sie als im Grunde für FRANC geeignet anzusehen sind. Weiterführend wurde die Fusion ausgewählter kostengünstiger Sensoren einer näheren Betrachtung unterzogen, um ihre Verwendbarkeit für navigatorische Zwecke abschätzen zu können. Der Fokus lag bei diesem Teil der Arbeit zum einen auf der Fusion von Inertialsensoreinheit mit Magnetometer, GPS und Odometrie mittels erweitertem Kalman-Filter, zum anderen auf der Entwicklung eines Kalman-Filters zur Lageberechnung über die Fusion der Daten von Gyroskop, Magnetometer und Accelerometer unter dezidierter Berücksichtigung externer Beschleunigungen, da für zukünftige Entwicklungsschritte, bei denen die räumliche Navigation auf unebenen Untergründen eine Rolle spielen wird, eine Auseinandersetzung mit diesem Thema unabdingbar ist.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Die technologische Innovation war seit jeher eine starke treibende Kraft in der Menschheitsgeschichte, wobei das Voranschreiten der Technik retrospektiv keineswegs als gleichförmig zu bezeichnen ist. Während zwischen der Eisenzeit und der industriellen Revolution noch viele Jahrhunderte lagen, trennt die Erfindung des Transistors und die digitale Revolution nur noch Jahrzehnte. Die Digitalisierung hält in fast allen Bereichen unseres Lebens Einzug, wobei die Entwicklungen, abhängig von Struktur und Komplexität der Bereiche, unterschiedlich schnell ablaufen. In der Welt der Robotik unterscheidet man häufig, ob eine Struktur in der Roboterumgebung und den Objekten mit denen ein Roboter interagiert, vorliegt oder nicht. Unterliegt eine Umgebung räumlich und zeitlich starken Veränderungen, etwa durch unterschiedliche Untergründe, Lichtverhältnisse oder Vegetation, bezeichnet man sie als unstrukturiert (vgl. [1]). Sind in der Industrie technologische Anwendungen oft von stationärer Natur mit wenig veränderlichen Umgebungsbedingungen, was eine gute Abgrenzung von Interaktionsobjekten zur restlichen Umgebung erlaubt, fällt dies bei beispielsweise landwirtschaftlichen Verwendungszwecken durch die veränderlichen Umgebungsbedingungen ungleich schwerer. Die gegenständliche Arbeit befasst sich mit dem Teilgebiet eines genau solchen, landwirtschaftlichen Verwendungszweckes, mit der autonomen Navigation eines Feldroboters.

Wenngleich man aufgrund des Arbeitstitels vermuten möchte, die Themensetzung sei ausreichend abgegrenzt, um sie im Rahmen dieser Arbeit vollumfänglich zu erarbeiten, wird bereits an dieser Stelle darauf hingewiesen, dass lediglich Teilaspekte (siehe Kapitel 1.3) der Thematik behandelt werden.

1.1 Motivation

Obgleich die menschliche und tierische Arbeitsleistung vor der industriellen Revolution lange Zeit die einzige zur Verfügung stehende Ressource in der Landwirtschaft war, gab es danach durch anfangs rein mechanische Maschinen Alternativen. Historisch betrachtet war zumindest ab Mitte des 20. Jahrhunderts die Wirtschaftlichkeit eines Umstiegs von Handarbeit auf maschinelle Arbeit immer von der Verfügbarkeit menschlicher Arbeitskraft, dem Verhältnis der Kosten für menschliche Arbeitszeit zu den Investitionskosten für Maschinen,

von deren Effizienz und den Energiekosten abhängig. Der verstärkte Einsatz von Maschinen war dabei immer mit einer Reduktion menschlichen Einsatzes in der Landwirtschaft verbunden, zusätzlich korrelierte das Einsparungspotential einer Maschine mit ihrer Größe. Den Maschinenabmessungen waren und sind dabei selbstverständlich Grenzen gesetzt, etwa technologische, rechtliche, auf öffentlichen Straßen, oder auch natürliche, wie die Ausdehnung der bewirtschafteten Flächen. Die Auswirkungen von Bodenverdichtung, verursacht durch den Einsatz landwirtschaftlicher Maschinen, gelangten erst relativ spät, etwa Mitte des 20. Jahrhunderts, ins Bewusstsein der Menschen, sind aber als limitierender Hauptfaktor für die Maschinenmasse zu nennen [2]. Ob und wann sich eine neue Technologie etablierte, hing dabei mitunter von der Art der Innovation ab. Die treibenden Kräfte etwa für energietechnische Innovationen waren vordergründig Energiepreise, für beschäftigungsintensive Anwendungen waren dagegen in erster Linie die Lohnkosten ausschlaggebend [3], [4].

Setzt man die fortwährende Gültigkeit der diesen Zusammenhängen zugrundeliegenden Mechanismen voraus, lässt sich die zukünftige Rolle digitaler Technologie in der Landwirtschaft abschätzen. Das momentane Ausmaß der Digitalisierung soll dabei als Ausgangslage für diese Abschätzung dienen. Digitale Systeme, als Teil größerer Maschinen, hielten rasch nach Erfindung des Transistors in vielen technischen Anwendungen schleichend Einzug. Dabei stieg die Leistungsfähigkeit digitaler Technologie kontinuierlich an, was wiederum das Spektrum möglicher Einsatzzwecke in Richtung rechenintensiverer Prozesse erweiterte [5]. In landwirtschaftlichen Applikationen, beziehungsweise im Ackerbau, sind digitale Assistenzsysteme für herkömmliche, überwiegend mechanische Maschinen, bereits kommerziell erhältlich. Der Trend ist dabei klar erkennbar: Das Ziel sind vollständig autonom agierende Maschinen, um die Arbeitskraft der die Maschine bedienenden Person(en) obsolet zu machen. Die Wirtschaftlichkeit einer Maschine wird durch diesen Sprung vollständig unabhängig von einem wesentlichen Kostenfaktor. Gänzlich autonome Maschinen sind derzeit noch nicht am Markt erhältlich, allerdings existieren sowohl Konzeptstudien als auch funktionierende Prototypen, für welche die Markteinführung teils schon in naher Zukunft angekündigt wurde[1,2,3]. Selbständig fahrende Arbeitsmaschinen mit der Fähigkeit, aufgezeichnete oder einprogrammierte Arbeitsabläufe auszuführen, sind aber bereits erwerbbar[4].

Konzeptionell steht die Steigerung der Effizienz von Landwirtschaftsmaschinen im Ackerbau durch Hochskalierung der Maschinenabmessungen der Idee

[1] Vgl. https://www.caseih.com/emea/de-at/News/Pages/2017-02-26-Autonome-Traktoren-weisen-den-Weg-in-die-Zukunft.aspx
[2] Vgl. https://www.ecorobotix.com/de/autonomen-roboter/
[3] Vgl. https://seedotrun.com/index.php
[4] Vgl. https://www.greenbot.nl/de/greenbot/

modularer Bauweise mit maximaler Flexibilität bei vollständig autonomen Robotern gegenüber. Einmalige, relativ hohe Investitionskosten sind für große, einsatzspezifische Maschinen mit hohen Flächenleistungen je Arbeitsstunde für jeden einzelnen Einsatzzweck aufzuwenden. Sind die Kosten je Maschinenbetriebsstunde aufgrund fehlender Lohnkosten für Bediener aber gering, ist es möglich, dass ein kleineres Gerät in Summe wirtschaftlicher ist. Diese Schlussfolgerung beruht auf der Annahme, dass die Betriebszeit der Maschine nicht durch andere Faktoren, beispielsweise der Witterung, limitiert wird. Erlangen autonome Roboter die Fähigkeiten, Arbeiten, die bis dato nur von Hand mit ausreichender Qualität zu erledigen waren, zu verrichten, als Beispiel kann hier etwa das Jäten von Beikräutern innerhalb der Pflanzenreihen genannt werden, maximiert sich das Einsparungspotential je Maschinenbetriebsstunde. Biologische Landwirtschaft benötigt im Durchschnitt einen höheren Einsatz an menschlicher Arbeit als konventionelle Bewirtschaftung, gleichzeitig ist dieses Mehr an Arbeit aber über das Jahr gleichmäßiger verteilt, was eine Limitierung der möglichen Arbeitszeit durch äußere Umstände nicht zwangsweise wahrscheinlicher macht [6]. Sørensen et al. [7] unterzogen die Einbindung innovativer Technologien in die biologische Landwirtschaft einer genaueren Betrachtung. Bei dieser Untersuchung analysierten die Autoren verschiedene Szenarien für Ackerbau, Milchproduktion und Schweinezucht. Die innovativen Technologien des für diese Arbeit relevanten Ackerbaues sind dabei zum einen ein autonomer Roboter zur Beseitigung von Beikraut und zum anderen eine Maschine zur thermischen Bodenbehandlung vor der Aussaat. Die Maschinen wurden sowohl in Kombination, als auch im separaten Einsatz beurteilt. Ausgehend von den Annahmen (Preis des Roboters, Anteile der verschiedenen Feldfrüchte, ...) dieser Analyse ergab sich für den Fall des autonomen Beikrautbeseitigungsroboters eine Reduktion der Kosten für das Jäten um bis zu 85%.

Alle derzeit erhältlichen Assistenzsysteme für Navigation und autonome Roboter nutzen RTK-GPS (Real Time Kinematic - Global Positioning System) zur Positionsbestimmung. Diese Technologie stellt momentan das technisch Machbare hinsichtlich absoluter Genauigkeit dar, bringt aber hohe einmalige und auch laufende Kosten mit sich. Mehr dazu im Kapitel 2. Ziel dieser Arbeit war es nicht, Technologie der höchsten Preis - und Qualitätsstufe zu implementieren und auf ihre Alltagstauglichkeit zu testen, sondern im Gegenteil, relativ günstige, leicht zugängliche Komponenten auf ihre Eignung für Navigationszwecke zu testen.

Die autonome Navigation als technische Problemstellung beschränkt sich nicht auf die Nutzung in landwirtschaftlichen Anwendungen, sondern ist vielmehr ein wesentliches Grundproblem der gesamten Robotik und nach wie vor Gegenstand der Forschung [8]. Es ist dabei auch anzunehmen, dass gerade in solchen Forschungsgebieten, bei denen es an breiter Front um erstmalig zu etablierende

Technologien geht, einiges an sogenanntem „dark knowledge“(=unzugängliches Wissen) existiert, mit dem sich Unternehmen Wettbewerbsvorteile gegenüber Konkurrenten versprechen[5].

1.2 Ausgangsituation

Das im Vorwort angesprochene Forschungsproject „FRANC - Field Robot for Advanced Navigation in bio-Crops“ fand im Sommer 2015 seinen Abschluss. Im Zuge des Projektes war ein fahrbereiter, quasi-holonomer Roboter mit optischen Sensoren aufgebaut worden, die es ihm ermöglichen, Reihen in Pflanzenkulturen aus unterschiedlichen Anfahrtswinkeln robust zu erkennen. FRANC verfügt über vier Räder, die jeweils über einen eigenen Motor unabhängig voneinander gelenkt und angetrieben werden können (siehe Abbildung 1.1). Die mechanische Konstruktion ist so ausgeführt, dass beliebige Lenkwinkel möglich sind, welche mit Absolutwertgebern erfasst werden, sodass keine Bewegungen für die Initialisierung beim Starten des Roboters notwendig sind. Abgesehen von den optischen Sensoren und jenen obligatorischen im Antriebsstrang waren zu Projektende keine zusätzlichen Sensoren auf FRANC aufgebaut. Die Hardware des Roboters unterteilt sich in drei Module, der mechanische Antriebsstrang, das elektronische on-board System und das Autonomiemodul. Das on-board System verfügt über eine implementierte Steuerung, die automatisch eine fahrbare Lenkposition aus Geschwindigkeitsbefehlen herstellt. Eine Kommunikation mit dem on-board System ist zum einen mit einer Smartphone-App möglich, wodurch FRANC ferngesteuert werden kann, zum anderen über das Autonomiemodul via Software. Das eingesetzte Software-Framework auf diesem Modul, auf dem auch die Reihenerkennung implementiert wurde, ist das Robot Operating System (ROS) (siehe Kapitel 2.1). Dieses Betriebssystem ist darauf ausgelegt, von ihrer Nutzergemeinschaft implementierte Algorithmen über online - Plattformen in Form von Packages und Informationen darüber auszutauschen[6]. Diese entwicklerfreundliche Ausrichtung von ROS ermöglichte es auch dem Verfasser dieser Arbeit, nur einzelne Aspekte der Navigation von FRANC im Umfang einer Diplomarbeit zu erarbeiten.

[5] Vgl. https://derstandard.at/2000063151335/Dark-Knowledge-waechst-Immer-mehr-Wissen-wird-unzugaenglich

[6] Vgl. http://www.ros.org/

Abbildung 1.1: Der Roboter FRANC.

1.3 Zielsetzung

Im Zuge der Diplomarbeit soll ein GPS gestütztes Navigationssystem für FRANC entwickelt werden. Das für FRANC vorgesehene Robot Operating System und die darin enthaltenen Navigationswerkzeuge sollen als Basis dienen, um im Zuge einer Simulation die Auswirkungen des nicht idealen Verhaltens FRANCs auf die Navigation abschätzen zu können. Darüber hinaus soll FRANCs Sensorik für die Navigation erweitert und in Betrieb genommen werden. Der Fokus soll dabei abseits vom technisch Machbaren in erster Linie auf kostengünstige Sensoren gelegt werden. Weiterführend soll den damit einhergehenden qualitativen Defiziten mit entsprechenden Algorithmen entgegengewirkt werden, um feststellen zu können, ob dadurch kostengünstige, autonome outdoor-Navigation ermöglicht werden kann. Die gewählten Sensoren und Algorithmen sind unter realen Einsatzbedingungen zu testen.

1.4 Lösungsweg

Dem sehr allgemein gehaltenen Titel der Diplomarbeit Rechnung tragend, und um die Kernpunkte der Arbeit vor deren Hintergründe besser darzustellen, wurden zusätzlich zur ursprünglichen Zielsetzung, die grundlegenden Fähigkeiten, die das Navigationssystem eines Feldroboters besitzen muss, recherchiert und mit dem Stand der Technik abgeglichen, sodass auch der noch zu leistende Entwicklungsaufwand auf dem jeweiligen Forschungsgebiet deutlich wird. Konkret werden in Kapitel 2.2, Bewegungsplanung und Hindernisvermeidung, sowie in

Kapitel 2.3, Umgebungsmodellierung und Kartierung, Methoden erläutert, die im Weiteren zwar teilweise verwendet werden, aber von den Navigationswerkzeugen des auf FRANC eingesetzten Robot Operating System bereitgestellt werden und deswegen nicht den Kern dieser Arbeit bilden. Kapitel 2.4 beschreibt die Grundlagen für Kapitel 3.1, in dem mit Hilfe einer Simulation eines Funktionsmodells überprüft wird, ob diese Navigationswerkzeuge grundsätzlich für FRANC geeignet sind. Darüber hinaus wurde evaluiert, inwieweit die Fusion kostengünstiger Sensorik zu einer Eignung für Navigationszwecke führt, auch in Hinblick auf zukünftige Entwicklungsschritte. Dafür wurden geeignete low-cost Sensoren ausgewählt und entsprechende Algorithmen implementiert, was in Kapitel 3.3 geschieht. Während der Arbeiten ergab sich dabei unwillkürlich eine Schwerpunktsetzung bei der Datenaufbereitung der Inertialsensoreinheit, die deswegen den Hauptteil dieses Kapitels bildet. Im darauf folgenden Kapitel 4 werden die Ergebnisse für die gewählten Sensoren und Filteralgorithmen zur Roboterlokalisierung mit verschiedenen Filterabstimmungen präsentiert.

2 Grundlagen und Stand der Technik

Im Folgenden werden die unterschiedlichen Aspekte der Navigation von Robotern im Allgemeinen und Feldrobotern im Speziellen umrissen, darunter auch Methoden, die nicht den Kern dieser Arbeit bilden, aber aufgrund der engen Zusammenhänge als unerlässlich befunden wurden, um das Wesentliche der Thematik darzustellen. Wie bereits erwähnt sind die Methoden und Algorithmen für die Navigation von Robotern nach wie vor Hauptthema vieler Forschungsarbeiten, deswegen gibt es für viele Teilaufgaben keine Algorithmen, die in der vorherrschenden Meinung als DER Stand der Technik anzusehen sind, darum wurden Methoden ausgewählt, die als geeignet erachtet wurden, um die Peripherie dieser Arbeit zu erläutern. Die nachfolgende Themenauswahl stellt daher keinesfalls den Anspruch auf Vollständigkeit. Sofern nicht anderweitig ausgeführt, handelt es sich immer um Methoden für die Navigation in zwei Raumdimensionen, also um Bewegungen in einer Ebene.

Die Integration der eigenen Methoden (Kapitel 3) erfolgte über das für den Roboter FRANC vorgesehene Roboterbetriebssystem ROS, das auch jene für die Navigation unerlässliche Algorithmen bereitstellt, die nicht im Fokus dieser Arbeit stehen und daher auch nicht selbst implementiert wurden. Weil sich die nachfolgend beschriebenen Methoden teilweise speziell auf Feldroboter beziehen, wird darauf hingewiesen, dass sie nicht jene sein müssen, die von ROS bereitgestellt werden und für die eigenen Tests verwendet wurden. Nachdem ROS die Basis für die Integration der eigenen Arbeit bildet, werden vor den für die Navigation eigentlich relevanten Methoden die Grundzüge dieses Systems in aller Kürze erläutert.

2.1 Robot Operating System - ROS

ROS ist ein Open-Source Meta- Betriebssystem, das die gesamte Infrastruktur, die zum Betrieb eines Roboters nötig ist, bereitstellt. Die zentralen Elemente des Robot Operating Systems sind:

- *Master:* Kern des Systems, verwaltet alle betriebssystemeigenen Abläufe und ermöglicht so den Betrieb verschiedener Nodes.

- *Nodes:* Eigenständige Prozesse, die Berechnungen ausführen. Hier hat der dezentrale Aufbau von ROS seinen Ursprung. Einzelne Nodes werden meist für die Erledigung relativ überschaubarer Berechnungen programmiert. Sie können allerdings sehr einfach mit anderen Nodes kommunizieren und mit ihnen Daten austauschen. Der Datenaustausch erfolgt hauptsächlich über sogenannte Topics.

- *Topics*: Sie definieren die Art der Daten und den Namen unter dem der Datenaustausch stattfindet. Nodes, die Daten von einem speziellen Topic empfangen möchten, müssen sich für dieses Topic anmelden (to subscribe to a topic) oder können selbst Daten an ein Topic publizieren (to publish to a topic). Die Daten eines Topics werden in Form einzelner Messages ausgetauscht. Für manche Anwendungen ist der Austausch über Topics zu einseitig, deswegen gibt es eine weitere Form der Kommunikation zwischen Nodes, die Services.

- *Services:* Bieten die Möglichkeit der Informationsübermittlung nach dem Frage/Antwort Prinzip. Anfragen (Requests) eines Nodes (Client) werden vom gefragten Node (Server) beantwortet (Response), dabei ist die Struktur von Request und Response von vornherein festgelegt.

- *Parameter Server:* Die dritte Möglichkeit Informationen zu verteilen. Dem Namen entsprechend ist er dazu gedacht, mehr oder minder statische Parameter, beispielsweise Konfigurationsparameter, systemweit verfügbar zu machen.

ROS ähnelt in seinem Aufbau durchaus anderen Roboterbetriebssystemen, wodurch sich ROS allerdings von anderen Systemen abheben möchte, ist die angestrebte Vielfalt an verfügbaren Algorithmen, die der Nutzergemeinschaft zur Verfügung steht bei gleichzeitig einfacher Integration in eigene Projekte. So soll der Open-Source Charakter und dezentrale Aufbau des Systems es den Anwendern ermöglichen, Code von anderen Programmierern wiederzuverwenden und damit die Entwicklungsarbeit auf Anwendungen zu konzentrieren, die eigentlich in ihrem Fokus stehen.

2.2 Bewegungsplanung und Hindernisvermeidung

Um eine Bewegung, die zur Erledigung einer höheren Aufgabe notwendig ist, ausführen zu können, muss diese vorher mit geeigneten Methoden geplant werden. Das Ziel der Bewegungsplanung und Hindernisvermeidung (motion planning and obstacle avoidance) ist es also, zwischen zwei anfahrbaren Punkten,

dem Start und dem Ziel, einen Pfad zu planen, der keine Kollisionen zwischen dem Roboter und seiner Umgebung verursacht. Dieser Aufgabe geht die Frage voraus, ob für ein bestimmtes Szenario überhaupt ein Pfad existiert, kann man diese Frage bejahen, folgt die eigentliche Aufgabe der Pfadberechnung. Ein allgemeine Formulierung dieser beiden Probleme gibt das „piano-mover's problem". Bei dieser Herangehensweise definiert man den Arbeitsraum (workspace), den Raum des Roboters mit seinem Konfigurationsraum (configurationspace) und die Hindernisräume (obstacleregion). Der Arbeitsraum ist dabei jener statische Raum, in dem sich der Roboter und die Hindernisse befinden können. Der Konfigurationsraum besteht aus allen möglichen Konfigurationen, die der Roboter einnehmen kann, der vollständige Konfigurationsvektor besteht aus dem Lagevektor ξ, dem Vektor der Lenkwinkel β sowie dem Raddrehwinkelvektor φ . Für einen Roboter mit Differentialantrieb besteht damit ein Konfigurationsset q aus der Position x, y und der Orientierung θ sowie den Radwinkeln, jedoch keinen Lenkwinkeln, da keine lenkbaren Räder vorhanden sind. Um nun mögliche Konfigurationen zu finden, in denen der Roboter nicht mit den Hindernissen kollidiert, zieht man all jene Konfigurationen heran, bei denen sich der Roboterraum und die Hindernisräume überschneiden und definiert sie als die Kollisionskonfigurationen. Die Menge aller Konfigurationen vermindert um die Kollisionskonfigurationen bezeichnet man als Freiraum (free space), in dem auch Startpunkt und Zielpunkt liegen müssen. Je nach Roboter kann es sein, dass nicht alle Elemente des Konfigurationsvektors für die Kollisionsbestimmung relevant sind. Dem Umstand Rechnung tragend, dass die explizite Berechnung des Freiraumes und eines Pfades darin sehr aufwändig sein kann, wurden Verfahren entwickelt, um mit weniger Recheneinsatz gangbare Lösungen zu finden. Das sogenannte „sampling-based motion planning" basiert auf der Beprobung des Konfigurationsraumes und der Überprüfung der genommenen Probesets auf Kollisionsfreiheit. Durch den Verzicht, nicht die Gesamtheit des Konfigurationsraumes in die Bewegungsplanung miteinzubeziehen, geht die Vollständigkeit des Algorithmus verloren, was bedeutet, dass für zwei beliebige Eingabepunkte nicht zwangsweise richtig erkannt wird, ob eine Lösung vorhanden ist. Es existieren eine Vielzahl an verschiedenen Methoden, um zu entscheiden, wo im Konfigurationsraum die Proben zu entnehmen sind, oft wird eine Diskretisierung mit konstanter Auflösung in den verschiedenen Dimensionen des Konfigurationsraumes verwendet. Es lassen sich zwei Herangehensweisen unterscheiden, wie danach die Proben auf Kollisionsfreiheit untersucht werden, die Algorithmen hinter diesen zwei Methoden unterscheiden sich unter anderem durch ihre Abbruchkriterien:

- *Erstellung einer kompletten Roadmap:* Diese Vorgangsweise wird gewählt, wenn nach Erstellen der Roadmap mehrere Pfadanfragen in kurzer Zeit

möglich sein sollen. Der Algorithmus wird abgebrochen, wenn eine vordefinierte Anzahl an Proben auf Kollisionsfreiheit untersucht wurde.

- *Suche nach einem Pfad über baumartige Strukturen:* Kommt zum Einsatz, wenn für eine Pfadanfrage in möglichst kurzer Zeit eine Lösung gefunden werden soll. Abbruchkriterium ist das Vorliegen einer solchen.

Bevor das „sampling-based motion planning“ erstmals beschrieben wurde, um weniger rechenintensive Algorithmen zu erhalten, gab es andere Ansätze das Pfadplanungsproblem zu lösen. Diese kombinatorischen oder auf Potentialfeldern basierenden Algorithmen können für gewisse Anwendungen sehr gute Lösungen bereitstellen, sind aber nicht einfach auf andere Problemstellungen ausdehnbar [9], [10].

2.2.1 Zwangsbedingungen

Die Bewegungen eines Roboters folgen physikalischen Gesetzen, die mit Differentialgleichungen in der Konfigurationsvariablen q und ihren zeitlichen Ableitungen $\dot{q}$, bei Betrachtung der Kinematik, sowie $\ddot{q}$ hinsichtlich der Kinetik modelliert werden können. Sind diese Ableitungen nicht integrierbar, das heißt sie sind nicht als totales Differential einer Funktion darstellbar, ist es nicht möglich sie als Bedingung in den Konfigurationsraum zu integrieren. In der Praxis äußern sich derartige anholonome Zwangsbedingungen dadurch, dass ein Roboter aufgrund seiner Kinematik und Kinetik möglicherweise jeden Punkt eines Raumes anfahren kann, dabei aber nicht jedem beliebigen Pfad (jeder Trajektorie) dorthin folgen kann [9], [10].

2.2.2 Hindernisvermeidung

Die obig erläuterten Techniken zur Pfadplanung sind ausschließlich globaler Natur, das heißt sie arbeiten mit den gesamten für die Navigation relevanten Informationen ihrer Umwelt. Oft werden Roboter aber in Umgebungen eingesetzt, die entweder völlig unbekannt sind oder auch Veränderungen unterliegen. Für solche Fälle ist es unumgänglich zumindest die unmittelbare Umgebung mit Sensoren zu erfassen und in die Bewegungsplanung miteinzubeziehen. Auch für diese Aufgabe wurden zahlreiche Methoden entwickelt, es wird hier aber lediglich nur der „Dynamic Window Approach (DWA)“ kurz angeschnitten, da dieser Algorithmus unter anderem in ROS schon bei der Grundinstallation inkludiert ist.

Dynamic Window Approach (DWA)

Strukturell unterteilt sich diese Methode in zwei Abschnitte, in einem ersten Schritt erfolgt für eine gewisse Situation die Berechnung der zulässigen Geschwindigkeitsführungsgrößen v (Roboterlängsgeschwindigkeit) und ω (Roboterwinkelgeschwindigkeit), anschließend werden diese mit einer Zielfunktion bewertet. Die Suche nach den zulässigen Geschwindigkeiten findet direkt im Führungsgrößenraum statt. Als zulässig werden dabei all jene Geschwindigkeiten erachtet, die das Fahrzeug generell und auch innerhalb der betrachteten, kurzen Zeitspanne erreichen kann, dabei aber sicher im Sinne der Kollisionsfreiheit sind. Die Berechnung der kollisionsfreien Geschwindigkeiten erfolgt mit der maximalen Verzögerung des Roboters und dem Abstand des Raumpunktes, an dem die Geschwindigkeitsberechnung durchgeführt werden soll, mittels Arbeitssatzes. Durch die Beschränkung auf erreichbare Geschwindigkeiten werden kinematische und dynamische Randbedingungen bei dieser Methode automatisch berücksichtigt. Die Größe des dynamischen Fensters, in dem die Führungsgrößensuche stattfindet, ist von den erreichbaren Beschleunigungen des Roboters und der Zeitspanne abhängig. Die Zielfunktion lautet:

$$G(p,v,a) = \alpha \cdot goal(v,\omega) + \beta \cdot velocity(v,\omega) + \gamma \cdot clearance(v,\omega) \qquad (2.1)$$

Die Zielfunktion ist somit als ein Kompromiss aus Fortschritt ($velocity(v,\omega)$), der hohe Geschwindigkeiten bevorzugt, Zielgerichtetheit ($goal(v,\omega)$), welche die Orientierung in Richtung des Zieles höher bewertet, und Hindernisvermeidung ($clearance(v,\omega)$), bei der Geschwindigkeiten mit großem Abstand von Hindernissen bevorzugt werden zu verstehen. Die Parameter α, β, γ können variiert werden, um Einfluss auf das Verhalten des Algorithmus zu nehmen. Die Geschwindigkeiten u und ω, bei denen die Funktion maximal wird, sind die gesuchte Lösung. Für weitere Details wird auf [10], [11] verwiesen.

Hindernisse in der Feldrobotik

Die im Vergleich zu urbanen und indoor Umfeldern unstrukturierte Umgebung von Feldrobotern erschwert die ohnehin komplexe Aufgabe der Hindernisvermeidung zusätzlich [12]. Algorithmen zur Hindernissvermeidung setzen die klare Kenntnis dessen voraus, welcher Teil der Umgebung als Hindernis anzusehen ist. Bewegt sich ein Roboter überwiegend auf ebenen, annähernd waagrechten Böden, wäre als eine einfache Hinderniserfassung beispielsweise das Herausstehen eines Objektes aus der Bodenebene denkbar. In natürlichem Terrain ist eine solche relativ klare Abgrenzung zwischen Boden und Hindernis selten gegeben, vielmehr müssen auftretende Unebenheiten klassifiziert werden, ob

sich der Roboter darüber hinweg bewegen kann oder ob sie als Hindernisse zu umfahren sind. Dazu ist in Umgebungen mit Vegetation eine Beurteilung allein nach der Geometrie nicht ausreichend, da Pflanzen vollständig oder zumindest teilweise nachgiebig sein können und somit nicht per se als Hindernisse zu klassifizieren sind. Manduchi et al. [12] entwickelten Algorithmen für LIDAR (Light detection and ranging) und Stereo Kameras zur Hinderniserkennung in offroad Umgebungen, die Hindernisse auch zwischen Gräsern robust erkennen. Feldroboter, die sich entlang von Pflanzenreihen bewegen, müssen in der Lage sein auch in den Reihenzwischenräumen eventuelle Hindernisse zu erkennen. Bei der Verwendung von Stereo Kameras ergibt sich dabei der Vorteil, auch die Farbinformation der Umgebung in die Klassifizierung miteinbeziehen zu können. Gerade dieser Vorteil bringt aber die zusätzliche unbestimmte Größe der herrschenden Lichtverhältnisse mit sich, die wesentlichen Einfluss auf die aufgenommenen Bilder der Kamera haben.

2.2.3 Bewegungsplanung für landwirtschaftliche Aufgaben

Die Bewegungsplanung eines Feldroboters unterliegt nicht nur dem Erfordernis, kollisionsfreie Bewegungen auszuführen, sondern Fahrten entsprechend einer übergeordneten Hauptaufgabe durchzuführen. Diese teilen sich in drei Gruppen ein:

- Saatbettbereitung, Anpflanzen und Aussaat
- Pflanzenpflege und Beikrautbeseitigung
- Ernten

Die Bewegungsplanung, respektive die gesamte Navigation, ist eine von mehreren Subaufgaben, welche alle einer dieser drei Hauptaufgaben dienen [1]. Gleichwohl FRANC primär für die Beseitigung von Beikräutern in bestehenden Pflanzenreihen entwickelt wurde, also der zweiten Hauptaufgabe in dieser Liste, werden hier kurz die Unterschiede zu den anderen Aufgaben hinsichtlich der Bewegungsplanung aufgezeigt.

Die Pfadplanung bei bereits bestehenden Pflanzenreihen besteht in erster Linie darin den Roboter entlang dieser Reihen mit dem richtigen lateralen Versatz zu führen, sodass sich die Räder beim Fahren zwischen den Reihen befinden. Die Hinderniserkennung und -vermeidung beschränkt sich dabei auf den Bereich, der nicht von Pflanzen eingenommen wird. Für diese Art der Pfadplanung ist die Reihenerkennung von zentraler Bedeutung. Die Pfadplanung am Ende der Nutzpflanzenreihen beim Wendevorgang im Vorgewende sollte im Idealfall darauf ausgelegt sein, keinen oder möglichst wenig Schaden

am Nutzpflanzenbestand durch den Roboter zu verursachen. Algorithmen zur Planung des Wendevorganges wurden zwar bereits entwickelt, jedoch nicht unter spezieller Rücksichtnahme auf bestehende Nutzpflanzen im Vorgewende. Die Pfadplanung für Erntevorgänge erfolgt unter ähnlichen Prämissen, jene für die Bestellung von Feldern ohne Nutzpflanzenbestand, beispielsweise für die Aussaat, unterliegt jedoch anderen Anforderungen. Solche Aufgaben erfordern meist Algorithmen um die gegenständliche Fläche möglichst effizient und vollständig ohne Lücken und Überschneidungen abzufahren. Um Pfade berechnen zu können, die diesen Anforderungen genügen, sind Informationen über das abzufahrende Terrain nötig. Eine übliche Methode zur dreidimensionalen Darstellung von Geländestrukturen sind sogenannte „Digital Elevation Maps (DEM)“. In DEM werden für die Punkte eines bestimmten Gitternetzes die Höhen hinterlegt, die dazu benutzt werden können, um zwischen diesen Punkten die Höhen zu interpolieren. Jeder Zelle, begrenzt von vier Gitterpunkten in x,y Richtung, ist dabei nur eine Höhenfläche zugewiesen, was nicht immer der Realität entspricht, zum Beispiel bei vertikalen Wänden. Aus diesem Grund wurden flexiblere Formate wie die *Multilevel Surfae Map (MLS)* entworfen, bei dem jeder Zelle mehrere Oberflächen zugeordnet werden können. Hameed et al. [13] veröffentlichten Algorithmen, die unter Verwendung von EDM Pfade planen, welche das betreffende Gelände vollständig mit Seite-zu-Seite Pfaden abdecken und dabei Überschneidungen sowie Lücken minimieren.

2.3 Umgebungsmodellierung und Kartierung

Die Erfassung von dreidimensionalen, räumlichen Strukturen spielt mittlerweile in einer Vielzahl an Anwendungen eine zentrale Rolle. Die Umgebungsmodellierung profitiert aber kaum von diesen ähnlichen Problemstellungen, da hier die fehlende Struktur der Umgebung des Roboters und seiner Interaktionsobjekte schlagend wird. Stattdessen kommen typischerweise adaptierte Versionen aus der 2D Umgebungsmodellierung zum Einsatz. Die Modellierung und Kartierung der Umgebung eines Roboters unterscheidet sich aber in den Anforderungen für 2D und 3D Anwendungen fundamental, da bei dreidimensionaler Betrachtungsweise nicht nur eine dritte Raumkoordinate hinzu kommt, sondern auch zwei zusätzliche Neigungswinkel, die der Roboter in einer solchen Umgebung einnehmen kann. In Summe wird die Lage eines Roboter im Raum also mit sechs Koordinaten festgelegt.

Eine wichtige Methode, die als eine der Schlüsseltechnologien für vollständig autonome Navigation gilt und Umgebungsmodellierung beinhaltet, ist *Simultaneous Localization and Mapping (SLAM)*. Wie die Bezeichnung bereits erahnen lässt, adressiert diese Methode nicht nur das Problem der Umgebungsmodellie-

rung und Kartierung, sondern gleichzeitig auch die Lokalisierung des Roboters in diesem Modell. Da die Umgebung zu Beginn völlig unbekannt ist, findet die Lokalisierung des Roboters in der entstehenden Karte immer relativ zum Startpunkt statt. Grundsätzlich basiert die Methode darauf, die Umgebung des Roboters mittels eines Sensors abzutasten und dazwischen die Bewegung des Roboters zu registrieren. Je nach Sensor zur Umgebungsabtastung erhält man unterschiedliche Informationen über die Umgebung, allen Sensoren ist jedoch gleich dass man niemals ein perfektes Abbild erhält, da es immer Unsicherheiten gibt, die in die Messungen einfließen. Dasselbe gilt auch für die Bewegungen des Roboters. Den Messgrößen wird deswegen ein Messmodell mit Normalverteilung zugrunde gelegt, bei dem die Lage des wahren Wertes mit einer definierten Varianz um einen Erwartungswert beschrieben wird. Die Messgröße der Umgebungserfassung ist dabei nicht das Abbild der Umgebung selbst, sondern die Lage des Roboters, die daraus errechnet wird. Wie diese Berechnung erfolgt, hängt zum einen vom verwendeten Sensortyp ab, zum anderen vom gewählten Klassifikationsschema. Die Verwendung von bekannten, für den Roboter sicher erkennbaren, Orientierungspunkten (*Feature-based*) ist verhältnismäßig einfach im Vergleich etwa zu dem Prinzip der Zugehörigkeit des aktuellen zwei- oder dreidimensionalen Abbildes zu der schon zuvor beobachteten Umgebung (*data association problem*), was als eine der schwierigsten Aufgaben von SLAM gilt. Hinsichtlich der Lösungsfindung unterscheidet man zwei Herangehensweisen, Betrachtung des vollständigen Problems versus online Berechnung. Bei Ersterem wird der ganze Pfad des Roboters unter Einbeziehung aller Daten auf einmal errechnet, während hingegen bei der online Berechnung inkrementell von der letzten Lage ausgegangen wird. Die Algorithmen für letztere Variante sind typischerweise sogenannte Filter, beispielsweise *Extended Kalman Filter (EKF)* (siehe Kapitel 2.4.4) oder *Partikelfilter*. Für statische, strukturierte, nicht zu große Umgebungen gibt es bereits robuste SLAM Implementierungen, für dynamische, unstrukturierte und großflächige Umgebungen besteht aber nach wie vor Forschungs- und Entwicklungsbedarf [14]. Aus diesem Grund ist diese Methode momentan nicht ohne weiteres für landwirtschaftliche Robotern einsetzbar.

2.4 Pfadverfolgung

Liegt nun der günstige Umstand vor, dass ein Roboter mithilfe einer Umgebungskarte und eines Pfadplanungstools einen Pfad erstellen kann, muss der Roboter letztendlich auch in der Lage sein, diesem Pfad folgen zu können. Wie schon im Kapitel 2.2 erwähnt, gibt es differenzielle Zusammenhänge zwischen den Konfigurationsgrößen eines Roboters, die seine Bewegungsfreiheit

einschränken. Je nachdem, welche Teile und Ableitungen des Konfigurationsvektors miteinbezogen werden, lassen sich vier Modelle unterscheiden, um die möglichen Bewegung eines Roboters zu beschreiben:

- kinematisches Lagemodell
- kinematisches Konfigurationsmodell
- dynamisches Lagemodell
- dynamisches Konfigurationsmodell

Wie man aus dem Namen schließen kann, unterscheiden sich die ersten Zwei von den anderen darin, dass nur die Lagegrößen und die Geschwindigkeiten in die Modellbildung einfließen, während bei den letzten Zwei auch die Änderung dieser Geschwindigkeiten Teil des Modells sind. Wenngleich in der Regelungstechnik oft das gesamte zu regelnde System mit Modellen beschrieben wird, um mit einem Regler eine oder mehrere Ausgangsgrößen in ebendiesem zu kontrollieren, ist es auch möglich Regler für Teilsysteme zu entwerfen, um gezielt die Komplexität zu reduzieren. Im Konkreten bedeutet dies, dass es nicht unbedingt notwendig ist, die umfassenderen und dadurch komplexeren dynamischen Modelle für die Beschreibung und Regelung des Robotersystems heranzuziehen, darum wird hier auf weitere Ausführungen zu diesen verzichtet. Die verbleibenden zwei kinematischen Modelle unterscheiden sich in den Elementen des Konfigurationsvektors, die im betrachteten Zustandsraum vorkommenden. Das kinematische Lagemodell beschränkt sich dabei auf die Änderungsraten der Lagekoordinaten und Lenkwinkel als Zustände, beim Konfigurationsmodell dagegen werden auch die Winkelgeschwindigkeiten der Räder in den Zustandsraum miteinbezogen. Wie in Kapitel 3 ausgeführt wird, ist der ROS Navigation Stack nicht für Roboter mit beliebiger Kinematik ausgelegt, sondern beschränkt sich auf holonome Roboter oder welche mit Differentialantrieb. Beide Typen weisen im Gegensatz zu FRANC keine lenkbaren Räder auf, daher ist es durchaus sinnvoll als Modell das kinematischem Lagemodell zu wählen, da dieses bereits ausreicht, um die Unterschiede hinsichtlich der Navigation auszudrücken. Durch den Verzicht auf die Beschreibung der kinematischen Kopplungen zwischen Raddrehwinkel und Lenkwinkel vereinfacht sich der Vergleich mit den vom Navigation Stack unterstützen Modellen.

2.4.1 Kinematisches-Lage-Modell

Campion et al. [15] führten bereits 1996 aus, dass es fünf generische kinematische Lagemodelle gibt, mit denen alle Typen von Robotern beschrieben werden

können. Die Beschreibung in diesem Abschnitt erfolgt jedoch nur für drei dieser Modelle, auch werden exzentrische Lenkräder gänzlich ausgespart.

Wie schon an früherer Stelle bemerkt, kann die Lage eines Roboters in Relation zu einem Bezugskoordinatensystem durch die drei Parameter x,y und θ, wie Abbildung 2.1 illustriert, beschrieben werden. Zusammengefasst werden sie im Lagevektor ξ. Die Anordnung der Achsen im Roboter folgt dabei bestehenden Konventionen[1].

$$\boldsymbol{\xi} = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} \tag{2.2}$$

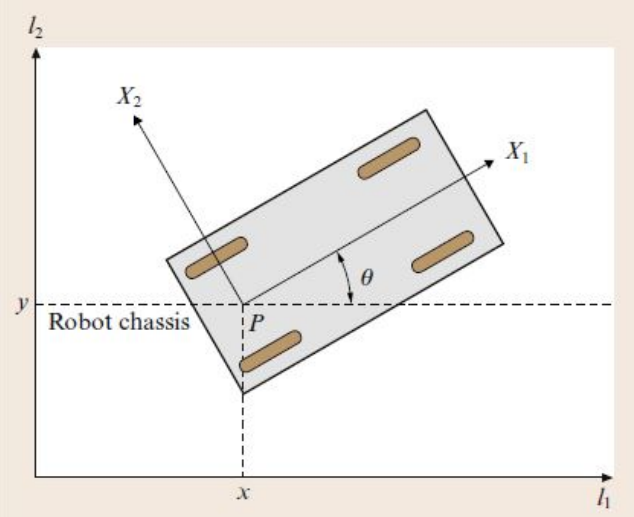


Abbildung 2.1: Lage eines Roboters in der Ebene [16].

Des Weiteren ist es notwendig die Lage und Orientierung der lenkbaren Räder eines Roboters relativ zu dessen eigenem Koordinatensystem zu definieren, die Koordinaten l und α beschreiben dabei die Lage, $\beta(t)$ die Orientierung. Für nicht lenkbare Räder ist β nicht von der Zeit abhängig.
Die kinematische Beschränkung eines konventionellen Rades ist die der Rollbedingung. Diese Bedingung besagt, dass die Geschwindigkeit in Richtung der Radachse ($v_{A_X}^{Rad}$) null ist und die Geschwindigkeit auf Höhe der Radachse normal zu dieser ($v_{A_Y}^{Rad}$) gleich der Relativwinkelgeschwindigkeit des Rades um die Radachse multipliziert mit dem Radradius ist. Führt man ein radeigenes Koordinatensystem ein, mit X_{Rad} entlang der Radachse und Y_{Rad}, einem

[1] Vgl. http://www.ros.org/reps/rep-0105.html

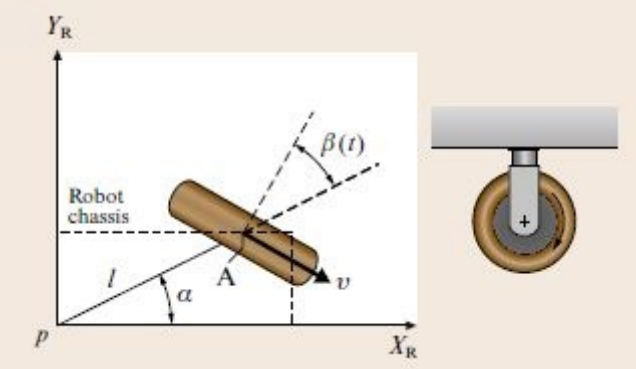


Abbildung 2.2: Lage eines lenkbaren Roboterrades [16].

Rechtssystem entsprechend orthogonal dazu, ergibt sich die Rollbedingung im Radkoordinatensystem zu

$$\begin{pmatrix} v_{A_X} \\ v_{A_Y} \end{pmatrix}^{Rad} + \begin{pmatrix} 0 \\ r \cdot \dot{\varphi} \end{pmatrix} = 0\,. \tag{2.3}$$

Die Geschwindigkeit des Punktes A kann im Roboterkoordinatensystem errechnet werden und danach in das Radkoordinatensystem transformiert werden. Die Transformation beschränkt sich dabei auf eine Drehung des Koordinatensystems. Die Geschwindigkeit des Punktes A im Roboterkoordinatensystem ergibt sich entsprechend Abbildung 2.1 und Abbildung 2.2 zu

$$\mathbf{v_A} = \mathbf{v_P} + \dot{\theta}\mathbf{e_z} \times \mathbf{r_{AP}}\,, \tag{2.4}$$

$$\begin{pmatrix} v_{A_X} \\ v_{A_Y} \end{pmatrix}^{Rob} = \begin{pmatrix} \dot{x}_P^{Rob} + \dot{\theta} l \sin(\alpha) \\ \dot{y}_P^{Rob} + \dot{\theta} l \cos(\alpha) \end{pmatrix}\,. \tag{2.5}$$

Transformiert auf das Radkoordinatensystem erhält man

$$\begin{pmatrix} v_{A_X} \\ v_{A_Y} \end{pmatrix}^{Rad} + \begin{pmatrix} 0 \\ r \cdot \dot{\varphi} \end{pmatrix} = \begin{pmatrix} \cos(\alpha+\beta) & \sin(\alpha+\beta) \\ -\sin(\alpha+\beta) & \cos(\alpha+\beta) \end{pmatrix} \cdot \begin{pmatrix} v_{A_X} \\ v_{A_Y} \end{pmatrix}^{Rob} + \begin{pmatrix} 0 \\ r \cdot \dot{\varphi} \end{pmatrix} = 0\,. \tag{2.6}$$

Für jedes einzelne Rad ergeben sich dadurch die zwei kinematischen Bedingungen, zum einen für die x-Geschwindigkeitskomponente

$$\cos(\alpha+\beta)\dot{x}_P^{Rob} + \sin(\alpha+\beta)\dot{y}_P^{Rob} + \dot{\theta} l \sin(\beta) = 0\,, \tag{2.7}$$

zum anderen

$$-\sin(\alpha+\beta)\dot{x}_P^{Rob}+\cos(\alpha+\beta)\dot{y}_P^{Rob}+\dot{\theta}l\cos(\beta)+r\dot{\varphi}=0 \tag{2.8}$$

für die y-Geschwindigkeitskomponente des Rades. Mit Definition einer erweiterten Rotationsmatrix

$$\mathbf{R}(\theta)=\begin{pmatrix}\cos(\theta) & \sin(\theta) & 0\\ -\sin(\theta) & \cos(\theta) & 0\\ 0 & 0 & 1\end{pmatrix} \tag{2.9}$$

lassen sich die Bedingungen mit der Ableitung des globalen Lagevektors wie von Campion et al. [15] ausdrücken:

$$\begin{pmatrix}\cos(\alpha+\beta) & \sin(\alpha+\beta) & l\sin(\beta)\end{pmatrix}\mathbf{R}(\theta)\dot{\boldsymbol{\xi}}=0\,, \tag{2.10}$$

$$\begin{pmatrix}-\sin(\alpha+\beta) & \cos(\alpha+\beta) & l\cos(\beta)\end{pmatrix}\mathbf{R}(\theta)\dot{\boldsymbol{\xi}}+r\dot{\varphi}=0\,. \tag{2.11}$$

Diese Beziehungen lassen sich auch für alle Räder gleichzeitig geschlossen in folgender Form anschreiben:

$$\mathbf{C}_1(\beta)\mathbf{R}(\theta)\dot{\boldsymbol{\xi}}=0\,, \tag{2.12}$$

$$\mathbf{J}_1\mathbf{R}(\theta)\dot{\boldsymbol{\xi}}+\mathbf{J}_2\dot{\boldsymbol{\varphi}}=0\,. \tag{2.13}$$

Dabei haben die Matrizen $\mathbf{C}_1$,$\mathbf{J}_1$ die Dimension $N\times 3$ und $\mathbf{J}_2$ die Dimension $N\times N$, worin N der Anzahl der Räder entspricht. Die geometrische Interpretation der Matrix $\mathbf{C}_1$ ist in Abbildung 2.3 veranschaulicht.

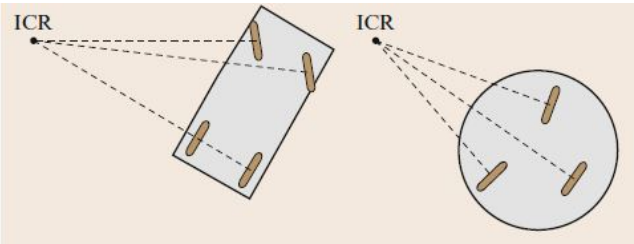


Abbildung 2.3: Lage des momentanen Geschwindigkeitspols (Instantaneous Center of Rotation (ICR)) [16].

Der momentane Geschwindigkeitspol muss auf allen Radachsen liegen, weil entlang dieser der Rollbedingung entsprechend keine Bewegung erfolgen kann. Wie aus Formel (2.14) ersichtlich, gehört die Geschwindigkeit des Roboters im Roboterkoordinatensystem $\mathbf{R}(\theta)\dot{\boldsymbol{\xi}}$ zum Nullraum von $\mathbf{C}_1$.

$$\mathbf{R}(\theta)\dot{\boldsymbol{\xi}}\in\mathcal{N}(\mathbf{C}_1) \tag{2.14}$$

Da nach dem Rangsatz die Dimension des Nullraumes plus dem Rang der Matrix die Anzahl der Spalten ergibt, bedeutet dies im Umkehrschluss, dass für den Fall, dass $\mathbf{C}_1$ vollen Spaltenrang aufweist, der Nullraum ohne Dimension ist und damit keine Bewegung möglich ist. Spaltet man $\mathbf{C}_1$ in $\mathbf{C}_{1f}$ für alle fixen Räder und $\mathbf{C}_{1s}(\beta_s(t))$ für alle lenkbaren auf, muss der Rang von $\mathbf{C}_{1f} \leq 2$ sein, da sonst bereits durch die fixen Räder keine Bewegung möglich ist. Für den Grenzfall $rang(\mathbf{C}_{1f}) = 2$ ist zwar Bewegung möglich, aber nur um einen fixen Geschwindigkeitspol, der Roboter wäre damit nicht steuerbar. Campion et al.[16] definierten daraus folgend den Grad der Mobilität zu

$$\delta_m = 3 - rang(\mathbf{C}_1)\,, \tag{2.15}$$

was der Dimension des Nullraumes von $\mathbf{C}_1$ entspricht. Der Beitrag der lenkbaren Räder $rang(\mathbf{C}_{1s})$ zu $rang(\mathbf{C}_1)$ bezeichnet man als Grad der Steuerbarkeit δ_s. Er gibt an, wieviele unabhängig voneinander lenkbare Räder vorhanden sind. Für $\delta_s = 2$ darf der Roboter kein zusätzliches fixes Rad haben, da ansonsten die zwei lenkbaren Räder nicht unabhängig voneinander orientierbar wären. Existieren mehr als zwei lenkbare Räder, müssen die Lenkbewegungen jedenfalls koordiniert werden, um sicherzustellen, dass Bewegung möglich ist. Je nach Grad der Mobilität und Grad der Steuerbarkeit lassen sich wie bereits erwähnt fünf generische Typen von Robotern unterscheiden, von denen aber lediglich drei Relevanz für diese Arbeit aufweisen.

- *Typ* $\delta_m = 2, \delta_s = 0$: Dieser Robotertyp hat keine lenkbaren Räder ($\delta_s = 0$) aber mindestens ein fixes Rad ($\delta_m = 2$), also ein durch $rang(\mathbf{C}_{1f})$ vermindertes δ_m. Sind mehrere fixe Räder vorhanden, so besitzen sie dieselbe Achse, weil nur so $rang(\mathbf{C}_{1f}) = 1$ sein kann. Der momentane Geschwindigkeitspol liegt immer auf dieser Achse. Roboter mit Differentialantrieb fallen in diese Kategorie.

- *Typ* $\delta_m = 1, \delta_s = 1$: Da $rang(\mathbf{C}_{1s}) = 1$, existiert bei diesen Robotern mindestens ein lenkbares Rad. Dazu muss noch zumindest ein fixes Rad vorhanden sein, da ansonsten $\delta_m = 2$ wäre. Pkw-ähnliche Roboter gehören diesem Typ an.

- *Typ* $\delta_m = 1, \delta_s = 2$: Roboter dieses Typs haben zwei unabhängig voneinander lenkbare Räder aber keine fixen. FRANC ist von diesem Typ, weil FRANC aber mehr als $\delta_s = 2$ lenkbare Räder hat, müssen sie in ihren Lenkbewegungen koordiniert werden, um sicherzustellen, dass Bewegung möglich ist.

Weil die Geschwindigkeit des Roboters in seinem eigenen Koordinatensystem Teil des Nullraums von $\mathbf{C}_1$ sein muss, können die möglichen Geschwindigkeiten

als Linearkombinationen einer Basis dieses Vektorraumes dargestellt werden. Diese Geschwindigkeiten entsprechen somit dem Aufspann des Nullraumes von $\mathbf{C}_1$. Schreibt man die Basis des Nullraumes in die Spalten einer Matrix $\mathbf{\Sigma}$ lässt sich (2.14) abwandeln zu

$$\mathbf{R}(\theta)\dot{\boldsymbol{\xi}} = \mathbf{\Sigma}\boldsymbol{\eta}(t)\,, \tag{2.16}$$

worin $\boldsymbol{\eta}(t)$ der Vekor der zeitvariablen Skalarkoeffizienten ist. Der Grad der Mobilität, die Dimension des Nullraumes und seiner Basis, sowie die des Vektors der Skalarkoeffizienten, sind ident. Um die Geschwindigkeit in globalen Koordinaten zu erhalten, muss die Gleichung lediglich umgeformt werden.

$$\dot{\boldsymbol{\xi}} = \mathbf{R}(\theta)^T\mathbf{\Sigma}\boldsymbol{\eta}(t) \tag{2.17}$$

Besitzt der Roboter lenkbare Räder, ist die Basis $\mathbf{\Sigma}(\boldsymbol{\beta})$ von den Lenkwinkeln abhängig, ansonsten ist sie konstant.

$$\dot{\boldsymbol{\xi}} = \mathbf{R}(\theta)^T\mathbf{\Sigma}(\boldsymbol{\beta}(t))\boldsymbol{\eta}(t) \tag{2.18}$$

$$\dot{\boldsymbol{\beta}} = \boldsymbol{\zeta}(t) \tag{2.19}$$

Die Gleichungen (2.18) und (2.19) entsprechen der Zustandraumdarstellung des kinematischen Lagemodells. Die Vektoren $\boldsymbol{\eta}$ und $\boldsymbol{\zeta}$ besitzen den Charakter von Geschwindigkeitseingangsgrößen. Die kompakte Schreibweise lautet

$$\dot{\mathbf{z}} = \mathbf{B}(\mathbf{z})\mathbf{u}(t)\,. \tag{2.20}$$

worin $\mathbf{z} = \begin{pmatrix}\boldsymbol{\xi}\\ \boldsymbol{\beta}\end{pmatrix}$, $\mathbf{B}(\mathbf{z}) = \begin{pmatrix}\mathbf{R}(\theta)^T\mathbf{\Sigma}(\boldsymbol{\beta}(t)) & 0\\ 0 & \mathbf{I}\end{pmatrix}$ und $\mathbf{u}(t) = \begin{pmatrix}\boldsymbol{\eta}(t)\\ \boldsymbol{\zeta}(t)\end{pmatrix}$ sind. Für einen Roboter mit Differentialantrieb vom Typ (2,0) kann der Punkt P (Abbildung 2.1)im Roboter immer so gewählt werden, dass sich schlussendlich folgendes Modell ergibt [15]:

$$\begin{pmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{pmatrix} = \begin{pmatrix}-\sin(\theta) & 0\\ \cos(\theta) & 0\\ 0 & 1\end{pmatrix}\begin{pmatrix}\eta_1\\ \eta_2\end{pmatrix}\,. \tag{2.21}$$

Selbiges gilt für Roboter vom Typ (1,1), wählt man Punkt P auf der Achse des fixen Rades, respektive in der Mitte der fixen Räder, wenn mehr als ein Rad vorhanden ist, folgt daraus für das kinematische Lagemodell [15]:

$$\begin{pmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\\ \dot{\beta}_1\end{pmatrix} = \begin{pmatrix}-L\sin(\theta)\sin(\beta_1) & 0\\ L\cos(\theta)\sin(\beta_1) & 0\\ \cos(\beta_1) & 0\\ 0 & 1\end{pmatrix}\begin{pmatrix}\eta_1\\ \zeta_1\end{pmatrix}\,. \tag{2.22}$$

Dieses kinematische Lagemodell nennt man häufig PKW-ähnliches Modell oder im Englischen car-like model. Die Eigenheiten des dritten Typs (1,2) werden in den weiteren Betrachtungen nicht mehr relevant sein, darum wird darauf verzichtet, das Modell an dieser Stelle explizit anzuführen.

Manövrierbarkeit

Der Grad der Mobilität des Roboters korrespondiert mit der Dimension von $\boldsymbol{\eta}$, er entspricht der Anzahl an Freiheitsgraden, die unmittelbar über $\boldsymbol{\eta}$ beeinflusst werden können. So lässt sich beispielsweise aus dem kinematischen Lagemodell des Typs (2,0) ablesen, dass die Winkelgeschwindigkeit und translatorische Geschwindigkeit direkt unabhängig voneinander angesteuert werden können. Für das Modell (1,1) sind diese Bewegungen immer gekoppelt, außer bei den Lenkwinkeln $\beta = (2n-1) \cdot \pi/2$ mit $n \in \mathbb{Z}$ und $\beta = 0$, das heißt, bei diesem Typ sind weitere Freiheitsgrade nur indirekt über den integralen Einfluss von ζ zugänglich. Den Grad der Manövrierbarkeit definiert man darum nach Campion [15] zu

$$\delta_M = \delta_m + \delta_s \,. \tag{2.23}$$

Roboter mit gleichem Grad an Manövrierbarkeit aber unterschiedlichen δ_m respektive δ_s sind nicht gleich in ihrem kinematischen Verhalten und können Pfade, die mit dem anderem Typ erstellt wurden, möglicherweise nicht in gleicher Weise reproduzieren.

2.4.2 Pfadfolgeregelung

Bei Verwendung der kinematischen Modelle sind die Stellgrößen beim Reglerentwurf Geschwindigkeiten. Um physikalische Systeme wie Roboter in ihrer Bewegung zu beeinflussen, sind gemäß den Newtonschen Axiomen jedoch Kräfte beziehungsweise Drehmomente notwendig, die von den meist elektrischen Antrieben der Robotern aufgebracht werden. Entwirft man also einen Regler für ein kinematisches Robotermodell, impliziert man damit die Notwendigkeit eines low-level Reglers, der die Drehmomente entsprechend der Geschwindigkeitsstellgröße regelt. Es handelt sich damit um eine kaskadierte Regelung mit einem Geschwindigkeitsregler als innerem Regler. Derartige Servoregler sind weit verbreitet, ebenso existieren Methoden um ihr Regelverhalten an das jeweilige System anzupassen. Nicht nur FRANC akzeptiert Geschwindigkeiten als Stellgrößen, sondern auch ROS stellt Software bereit (siehe Kapitel 3), die darauf ausgelegt ist, Geschwindigkeiten als Stellgrößen auszugeben. Die bis hierher erörterten Zusammenhänge führen zu folgendem, in Abbildung 2.4

dargestellten Regelkreis für die Navigation. Es handelt sich um ein Mehrgrößensystem, die Regelgrößen sind daher vektorielle Größen. Die Größen $\mathbf{d}(t)$ und $\mathbf{r}(t)$ sind die Störgröße und der Messfehler respektive das Messrauschen, alle weiteren Größen ($\boldsymbol{\xi}, \mathbf{u}, \boldsymbol{\varphi},$) stimmen mit der bisher vorgekommenen Notation überein. Der Fokus dieser Arbeit liegt nicht am Entwurf der Regler, sondern auf der Untersuchung und Evaluation eines kostengünstigen Systems für den Block „Messeinrichtung“.

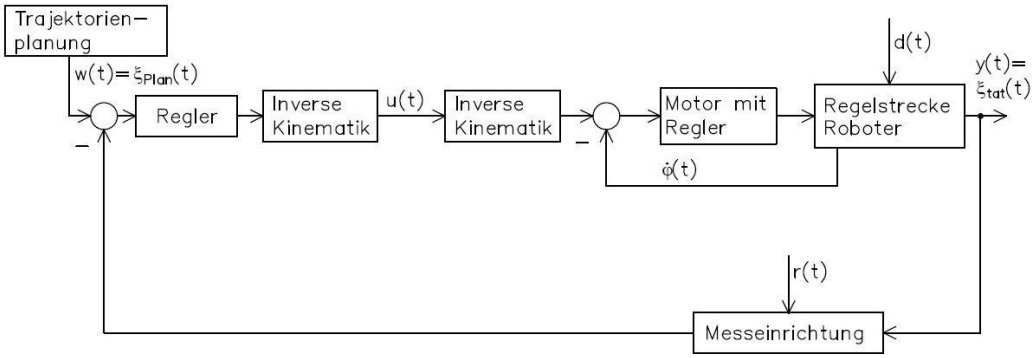


Abbildung 2.4: Blockschaltbild der Pfadfolgeregelung

Relevanz der Eigenlokalisierung

Der in Abbildung 2.4 dargestellte Messfehler $\mathbf{r}(t)$ besitzt die gleiche Übertragungsfunktion wie die Führungsgröße $\mathbf{w}(t)$ selbst, es ist daher im Interesse eines Entwicklers diesen Messfehler so klein wie möglich zu halten, sodass seine Auswirkungen auf das Systemverhalten im Idealfall vernachlässigbar werden. Die im Kapitel 2.3 umrissene Methode SLAM stellt eine, für landwirtschaftliche Umgebungen jedoch noch unausgereifte, Möglichkeit dar, wie die Position eines Roboters gemessen werden kann, während dabei auch noch gleichzeitig eine Umgebungskarte erstellt wird. Nachfolgend finden sich verschiedene Technologien, die für unstrukturierte, großflächige Einsatzgebiete geeigneter sind. Dabei wird, je nach Messmethode, auch speziell auf die Messunsicherheit eingegangen, sowie auf alternative Verfahren zur bloßen Messung, sogenannte Schätzer und Beobachter.

2.4.3 Sensoren zur Lokalisierung

Wie schon die verschiedenen Unteraufgaben zur Navigation, unterscheiden sich auch die Sensoren für indoor Anwendungen von jenen für outdoor Roboter. Dieses Unterkapitel bezieht sich in seinen Ausführungen auf den konkret

vorliegenden Fall eines Feldroboters mit unstrukturierter Umgebung. Unter Bedachtnahme der Einbeziehung von Verfahren abseits alleinstehender Messmethoden, welche direkt die absolute Position liefern, lassen sich die für die Lokalisierung geeigneten Messverfahren nach verschiedenen Gesichtspunkten klassifizieren. So kann die Messung, wie soeben erwähnt, absolut oder relativ erfolgen, wobei Absolutmessungen immer exterozeptiver (EZ) Natur sind, dabei also Eindrücke aus der Umgebung verarbeitet werden müssen. Relative Messungen können auch propriozeptiv (PZ) stattfinden. Nachfolgend ein als geeignet befundener Auszug an Sensoren.

Drehgeber

Sensoren zur Erfassung von Drehbewegungen arbeiten nach unterschiedlichen Prinzipien, in Robotern dienen sie meist dazu die Drehbewegungen oder die Position von Wellen oder Achsen zu erfassen (also propriozeptiv). Die kostengünstigsten Ausführungen sind Potentiometer mit beschränktem Arbeitsweg. Drehgeber mit unbegrenztem Drehwinkel arbeiten überwiegend mit radialen Mustern, oft einfache Strichgitter, auf Sensorscheiben die abgetastet werden. Häufig kommen sogenannte Quadratur Drehgeber zum Einsatz, bei denen die Strichgitter doppelt, jedoch mit Phasenverschiebung, abgetastet werden, was den Vorteil der Drehrichtungserkennung und erhöhter Auflösung mit sich bringt. Bei geeigneter Anordnung mehrerer Muster sind nach diesem Prinzip auch Absolutwinkelgeber realisierbar. Die Abtastung erfolgt je nach Sensorscheibe mittels magnetischer, induktiver, optischer oder kapazitiver Wandler. Bei fahrbaren Robotern sind Drehgeber in aller Regel im Antriebsstrang vorhanden, sodass mit ihrer Hilfe der von den Rädern überstrichene Drehwinkel ermittelt werden kann. Mit einem kinematischen Robotermodell kann mit den Drehwinkeln der Räder auf den zurückgelegten Weg geschlossen werden. Diese Methode bezeichnet man als Odometrie. Sie hat den Nachteil, dass sich, ausgehend von einer Startposition, beim Aufsummieren von Wegstrecken, Fehler in der Berechnung fortpflanzen und anwachsen. Weil der Methode ein kinematisches Modell zugrunde liegt, gibt es eine Vielzahl von Fehlerquellen, die wirksam werden können. Entspricht beispielsweise die angenommene Radgeometrie nur geringfügig nicht dem realen Rad, führt dies sehr schnell zu großen Fehlern bei der ermittelten Endposition. Weitere Fehlerquellen sind etwa auftretender Schlupf, von der idealen horizontalen Ebene abweichender Untergrund, veränderte Fahrzeuggeometrie durch elastische Verformungen oder Überbestimmtheit des kinematischen Modells zufolge nicht idealer Lenkwinkelgeber, um die wichtigsten zu nennen.

Inertiale Messeinheit, Inertial Movement Unit (IMU)

Als Inertial Movement Unit wird die Kombination mehrerer Sensoren bezeichnet, um die Eigenbewegungen eines Systems zu erfassen. IMUs beinhalten zumindest ein Accelerometer (EZ) und ein Gyroskop (PZ), oft aber auch ein Magnetometer (EZ). Inertiale Messeinheiten stehen in enger Verbindung zu Inertialen Navigationssystemen (siehe Kapitel 2.4.5). Da die Messgrößen bei Accelerometer und Gyroskop ausschließlich zeitliche Ableitungen des Weges, respektive der Positionswinkel sind, kann eine Positionsermittlung nur über das (doppelte) Integral -und damit nur relativ- stattfinden. Kommerziell sind IMUs in allen Preis- und Qualitätsklassen erhältlich, speziell die unterste Preisklasse ist für diese Arbeit interessant, näheres dazu im Kapitel 3.

Bakenbasierte Positionsbestimmung

Bakenbasierte Positionsbestimmung beruht auf den geometrischen Zusammenhängen zwischen Raumpunkten und ihren möglichen Bestimmungsgrößen in einem vorgegebenen Bezugssystem. Diese Größen können entweder Entfernungen und Abstände (Trilateration) oder Winkel (Triangulation), theoretisch aber auch eine Kombination daraus sein. Im bekannten Bezugssystem nehmen die Baken bekannte Orte ein und der zu bestimmende Ort wird über die erhaltenen Bestimmungsgrößen, relativ zu den Baken, errechnet. Am Ort der Positionsbestimmung ist mindestens eine Empfangseinheit, je nach Technologie auch eine Sendeeinheit notwendig. Zur Ermittlung der Bestimmungsgrößen ist es zwingend erforderlich, dass uni- oder bilaterale Informationsübermittlung stattfindet. Sind Baken nicht aktiv als Sender am Positionsbestimmungsprozess beteiligt, spricht man von identifikationsbasierter Positionsbestimmung (vgl. Kapitel 2.3). Die Informationsübermittlung kann mittels elektromagnetischer Wellen, optisch oder über Ultraschall erfolgen. Für robotische Anwendungen sind mehrere für indoor und outdoor geeignete Systeme kommerziell erhältlich. Die wohl bekannteste Variante der bakenbasierten Positionsbestimmung ist das *Global Positioning System (GPS).*

Global Positioning System (GPS)

Die vollständige offizielle Bezeichnung NAVSTAR-GPS steht für „Navigational Satellite Timing and Ranging - Global Positioning System“. Es handelt sich also um ein Satellitennavigationssystem (Globales Navigationssatellitensystem - GNSS) und wurde ab den 1970er Jahren in den USA entwickelt. Auch andere Nationen und die EU betreiben Satellitennavigationssysteme, die Entwicklung des russischen Systems GLONASS (Global Navigation Satelite System) begann

zu einer ähnlichen Zeit. China verfügt mit BeiDou ebenso über ein funktionierendes System, das mit geostationären Satelliten aber hauptsächlich den asiatischen Raum abdeckt. Das europäische Galileo befindet sich zwar noch im Aufbau, ist aber bereits seit 2016 für die Allgemeinheit zugänglich. Im Gegensatz zu den drei erstgenannten ist Galileo nicht unter militärischer Kontrolle, sondern von der Europäischen Union in Auftrag gegeben. Die weiteren Ausführungen beschränken sich auf das amerikanische System GPS.

Die Satellitennavigation kann in seiner Funktionsweise, wie bereits erwähnt, als funkbakenbasierte Positionsbestimmung betrachtet werden. Die Satelliten dienen dabei als Baken, die Informationen an die Empfangseinheiten übermitteln. Alle Satelliten senden die Informationen auf denselben Nominalfrequenzen bzw. den gleichen zwei Frequenzbändern, dem L1 und L2-Frequenzband. Auf dem L2-Frequenzband werden Daten für militärische Zwecke in unbekanntem Code übertragen. Wenngleich der übertragene Code nicht bekannt ist, kann die Phasenlage der Trägerfrequenz des L2-Frequenzbandes in sogenannten *Codeless Receivern* zur Verbesserung des Messergebnisses genutzt werden [17]. Auf dem L1 Band wird selbiger Code übertragen, allerdings zusätzlich auch bekannter, zivil nutzbarer Code. Um im Empfänger trotz der selben Nominalfrequenz aller Satelliten die Informationen den jeweiligen Satelliten zuordnen zu können, sendet jeder Satellit die Informationen in einem dem Empfänger bekannten Code (sogenanntes Pseudozufallsrauschen, im Englischen: Pseudo Random Noise Code - PRN Code), der durch seine Eigenschaften (spezielle Werte der Korellationsfunktionen zu sich selbst und den anderen verwendeten Codes) vom Code anderer Satelliten gefiltert werden kann. Der dem Empfänger bekannte Code kann im Empfänger reproduziert werden, wodurch die Phasenverschiebung zum empfangenen Code ermittelt werden kann. So ist es möglich, in einer ersten Näherung auf die Entfernungen zu schließen. Findet Informationsübertragung nämlich nur in eine Richtung statt, müssen abgesehen von den Phasenverschiebungen der einzelnen Satellitensignale, die in die Berechnung einfließen, diese Signale schließlich auch auf eine gemeinsame Zeitbasis bezogen werden, die in diesem Fall ebenso als Unbekannte in die Berechnung einfließt. Aufgrund dieser Unbekannten spricht man bei den erhaltenen Entfernungen von Pseudoentfernungen, weil ein etwaiger Synchronisationsfehler der Uhren die berechneten Entfernungen beeinflusst. Insgesamt muss der Empfänger um nach vier Unbekannten, der Position sowie der Zeitdifferenz zwischen Empfängeruhr und Satellitenuhr, auflösen zu können, mindestens vier Satellitensignale empfangen. Das entstehende Gleichungssystem ist nichtlinear, weshalb die Lösung iterativ ausgehend von Schätzwerten erfolgt.

Zusätzlich zu den zeitlichen Verschiebungen des gesendeten Codes ist es technologisch möglich, auch die Phasenverschiebung der Trägerfrequenz zu

bestimmen. Die Messung erfolgt über den Dopplereffekt der aufgrund der Relativgeschwindigkeit zwischen Empfänger zu einer Änderung der Trägerfrequenz aus Sicht des Empfängers führt. Die Trägerphasenmessung liefert jedoch nur Änderungen der Trägerphase relativ zum Beginn der Messung, die vollständige Anzahl der Wellenlängen zwischen Satellit und Empfänger ist unbekannt und wird als Trägerphasenmehrdeutigkeitswert bezeichnet. Für hoch genaue Anwendungen muss dieser Mehrdeutigkeitswert mit geeigneten Methoden bestimmt werden.

Bei ihrer Ausbreitung bewegen sich die Satellitensignale durch die Atmosphäre, ehe sie beim Empfänger ankommen. Dieser kalkuliert mit der Lichtgeschwindigkeit als Ausbreitungsgeschwindigkeit für die elektromagnetischen Wellen. Die vorherrschenden Bedingungen in den atmosphärischen Schichten, allen voran die Ionosphäre und Troposphäre, haben allerdings Einfluss auf die Wellenausbreitungsgeschwindigkeit. In der Ionosphäre, die sich zwischen 50km und 1000km Höhe erstreckt, ist der Effekt im Gegensatz zur Troposphäre (bis 10km Höhe) zusätzlich frequenzabhängig. Bei der Nutzung beider Frequenzbänder L1 und L2 kann der Ionosphärenfehler deswegen eliminiert werden. Der Laufzeitfehler durch veränderliche Bedingungen in der Troposphäre ist in wesentlichem Maße zusätzlich von der Wegstrecke des Signals, also vom Höhenwinkel des Satelliten abhängig. Neben diesen zwei Fehlerquellen treten auch weitere auf, beipsielsweise Ephemeridenfehler und Satellitenuhrenfehler, der betragsmäßig größte Anteil fällt jedoch auf den Ionosphärenfehler.

Die Angaben zur Genauigkeit von GPS Messungen erfolgen je nach Literatur und Verwendungszweck auf verschiedene Arten. Zur groben Abschätzung des Fehlers findet man oft simple skalare Zahlenwerte, die als zirka-Intervallangaben rund um die wahre Position aufzufassen sind. Solche Angaben, ohne nähere Ausführungen über die Eigenschaften des Intervalls und die Verteilung der Messungen darin, sind nur brauchbar, wenn es um den Vergleich verschiedener Technologien zur Verbesserung der Genauigkeit, wie nachfolgend erläutert, geht. Müssen die Daten eines GPS - Empfängers jedoch weiterverarbeitet werden, reicht eine solche Angabe nicht aus. In solchen Fällen werden andere Formate gewählt, etwa die Angabe mittels *Circular Error Probable (CEP)*. Diese Fehlerangabe besteht aus dem Radius eines Kreises, innerhalb dessen sich ein angegebener Prozentsatz aller Messungen befindet. Es handelt sich also um eine zweidimensionale Fehlerangabe, die sich zumeist auf die horizontalen Abweichungen bezieht. Ist der Prozentsatz nicht separat aufgeführt, kann man davon ausgehen, dass dieser bei 50% liegt. Legt man den Messungen eine zweidimensionale Normalverteilung mit identen Standardabweichungen in beide Richtungen zugrunde, kann mit dem Radius der CEP die Standardabweichung dieser Normalverteilung errechnet werden. Manchmal findet man auch Angaben in Form einer Root Mean Square Deviation (RMSD) oder eines Root Mean

Square Error (RMS Error). Liegt kein systematischer Fehler bei den Messungen vor, sodass man von Erwartungstreue ausgehen kann, entspricht der RMS Error gleich der Standardabweichung. Im National Marine Electronics Association (NMEA) Standard zur Übertragung von Navigationsinformationen, der bei GPS Empfängern meist zum Einsatz kommt, gibt es auch NMEA-Datensätze, die explizit die Standardabweichungen der Positionsbestimmung beinhalten.

Offizielle Stellen geben zur Genauigkeit des GPS Standard Positioning Service, was dem zivil nutzbaren Teil des GPS entspricht, einen Fehler von 7,8m an, innerhalb dessen sich 95% aller Messungen befinden. Das entspricht einer Standardabweichung von etwa vier Metern. Bei diesen Angaben bleiben jedoch Fehler durch Ionosphäre und Troposphäre sowie spezifische Fehler der Empfänger unberücksichtigt[2]. In der Literatur findet man daher oft Genauigkeitsangaben im Bereich von etwa 10m oder auch darüber [17].

GPS Augmentierung

Unterstützende Systeme für GPS wurden aus mehreren Gründen entwickelt, die Steigerung der Positionierungsgenauigkeit ist einer davon.
Das klassische Verfahren der GPS Augmentierung wird als *Differental GPS (DGPS)* bezeichnet. Es beruht auf der Erkenntnis, dass derselbe Fehler eines Empfängers dessen Standort exakt bekannt ist, auch bei Messungen, die zeitgleich im nahen Umfeld stattfinden, mit marginalen Abweichungen auftritt. Darum lassen sich vom Referenzempfänger mit bekanntem Standort Korrekturdaten errechnen, mit denen Empfänger im Umkreis ihre Messungen korrigieren können. Als näheren Umkreis bezeichnet man dabei durchaus einen Radius von mehreren Kilometern. Wie die Empfänger mit den Korrekturdaten versorgt werden, unterscheidet sich je nach Variante, die darüber hinaus aber auch Unterschiede in ihrer Reichweite, der Netzwerkgröße und der erzielbaren Genauigkeit aufweisen. Der klassische Vertreter des Differential GPS ist das *Local Ground Based Augmentation System (Local GBAS)*, bei dem das Korrektursignal nur im näheren Umkreis der Referenzstation zur Verfügung steht. Bei GBAS werden die Korrekturdaten immer terrestrisch an die Empfänger übermittelt. Neben lokalen existieren auch überregionale und landesweite Systeme, bei denen mehrere Referenzstationen zu einem Netzwerk zusammengeschlossen werden. Mit DGPS sind Genauigkeiten im Dezimeterbereich realisierbar. Wird im Zuge der Korrekturdatenermittlung und -auswertung auch die Trägerphasenmehrdeutigkeit bestimmt, liegt die Genauigkeit im Bereich einiger Zentimeter. Erfolgt die Positionsbestimmung zusätzlich noch in Echtzeit, spricht man vom sogenannten *Real Time Kinematic - GPS (RTK-GPS).*

[2]Vgl. https://www.gps.gov/technical/ps/2008-SPS-performance-standard.pdf

Alternativ zu GBAS wurden auch satellitengestützte Augmentierungssysteme entwickelt, man bezeichnet sie analog als *Satellite Based Augmentation System (SBAS)*. Zwar sind für diese Systeme genauso Referenzstationen am Boden nötig, jedoch senden sie die Korrekturdaten nicht terrestrisch, sondern via Satelliten über das L1-Frequenzband. Die dabei abgedeckten Bereiche sind wesentlich weitläufiger, so deckt etwa das *European Geostationary Overlay System (EGNOS)* etwas mehr als Europa ab, verfügt dabei aber momentan nur über rund 30 Referenzstationen. Zum Vergleich: Die deutschlandweite Variante des GBAS, *Satellitenpositionierungsdienst der deutschen Landesvermessung (SAPOS)* verfügt über 270 Referenzstationen auf deutschem Boden. Entsprechend geringer ist die mit EGNOS erreichbare Genauigkeit, sie liegt bei etwa 1-3m. Im Gegensatz zum meist kostenpflichtigen GBAS ist EGNOS jedoch gemeinsam mit einigen anderen SBAS kostenfrei verfügbar. Darüber hinaus haben SBAS, die über die GPS-Frequenzbänder übertragen, den Vorteil keine zusätzliche Schnittstelle für die Korrekturdatenübertragung zu benötigen.

Marktsituation für GNSS

Auf dem Segment der Echtzeit Präzisionspositionierung, die für den Endnutzer in aller Regel nicht kostenlos ist, stehen mehrere kommerzielle Unternehmen im Wettbewerb zueinander, was die Entwicklung der Systeme weiter vorantreibt. Die Firma Trimble etwa verspricht mit ihrem relativ neuen Produkt Trimble RTX Correction Services weltweit über Satellit verfügbare Korrekturdaten, die zu einer horizontalen Genauigkeit von vier Zentimetern führen[3]. Am unteren Ende der Preisskala der RTK respektive DGPS Systeme findet man Kits ohne laufende Kosten mit eigener Referenzstation für unter 1000 Euro, deren Entwicklung in den letzten Jahren teilweise über Crowdfunding ermöglicht wurde[4,5]. Gewöhnliche GPS Module mit SBAS und/oder Galileo/GLONASS Unterstützung, sind hingegen längst bei einer Vielzahl an Anbietern schon für unter 50 Euro erhältlich.

2.4.4 Beobachter und Schätzer

Bei der Regelung von Mehrgrößensystemen ist es erforderlich, dass alle Zustände des Systems gemessen und rückgeführt werden können, was in der Realität aber nicht immer für alle Zustandsgrößen machbar ist. Aus diesem Grund wurden Beobachter entwickelt, um die fehlenden Zustände des Systems aus den zur Verfügung stehenden Eingangsgrößen und den gemessenen Ausgangsgrößen zu

[3]Vgl. http://www.trimble.com/Positioning-Services/Trimble-RTX.aspx
[4]Vgl. https://emlid.com/reachrs/
[5]Vgl. https://www.swiftnav.com/piksi-multi

errechnen. Dazu benötigt der Beobachter das Modell der Regelstrecke mit dessen Hilfe er die Ausgangsgrößen über die Eingangsgrößen berechnet, die dann mit der tatsächlich gemessenen Ausgangsgröße verglichen und rückgeführt wird. Im Zuge dieser Arbeit wird nicht näher auf die Anwendung von Beobachtern eingegangen, es soll lediglich der Unterschied zum nachfolgend Erläuterten herausgestrichen werden, nämlich, dass Beobachter zum Einsatz kommen, wenn die Zustände deterministischer Modelle rekonstruiert werden sollen. Treten stochastische Größen in Systemen mit Beobachtern auf, beispielsweise Messrauschen, wirken die Beobachter spektral wie Filter entsprechend ihrer Übertragungsfunktion. Bei solchen stochastischen Störgrößen muss man davon ausgehen, dass sie über den gesamten Beobachtungszeitraum fortwährend andauern und folglich ihre Auswirkungen nicht asymptotisch abklingen.

Schätzer hingegen dienen dazu in stochastischen Systemen Zustände zu schätzen. Die weiteren Ausführen zu diesem Thema beschränken sich auf einen in Ingenieursanwendungen weit verbreiteten Schätzer, ein Minimum-Varianz Schätzer, das *Kalman-Filter*.

Kalman-Filter

Lineare, stochastische Prozesse, für die das Kalman-Filter entwickelt wurde, sind Prozesse, in deren Modell zumindest eine Zufallsvariable, genauer: eine Familie von Zufallsvariablen, die mit der Zeit geordnet durchlaufen wird, eingeht. Es kann auch sein, dass sich deterministische Prozesse mit stochastischen überlagern, etwa wenn sich der Rauschterm eines Messprozesses mit der Messgröße überlagert. In der Systemtheorie beschäftigt sich die Prozessidentifikation unter anderem mit den mathematischen Modellen die stochastischen Prozessen zugrunde gelegt werden können. Solche mathematische Modelle sind meist Differenzialgleichungen oder im diskreten Fall Differenzengleichungen. Je nach Disziplin, innerhalb derer ein Prozess modelliert werden soll, unterscheiden sich die Methoden mit denen die Modellierung erfolgt. In den Ingenieurswissenschaften folgen Systeme oft physikalischen Gesetzen, die verwendet werden können, um das Systemverhalten abzubilden, wogegen in anderen Disziplinen, beispielsweise den Wirtschaftswissenschaften, die Modellfindung meist für vorhandene Datensätze im Zuge des Identifikationsprozesses erfolgt. Findet die Modellbildung rein theoretisch über a priori bekannte Gesetzmäßigkeiten statt, spricht man von White-Box Modellen, im Gegensatz dazu bezeichnet man die experimentell abgeleiteten Modelle als Black-Box Modelle.

Das Kalman-Filter liefert für lineare stochastische Systeme den Schätzwert $\hat{\mathbf{x}}$, der im Mittel mit dem Systemzustand $\mathbf{x}$ übereinstimmt. Es existiert eine Vielzahl an Versionen, auch abhängig davon, ob ein zeitkontinuierliches oder

zeitdiskretes Systemmodell vorliegt. Für nichtlineare Systeme wurden unter anderem das erweiterte (Extended Kalman Filter, EKF) und das linearisierte Kalman-Filter entwickelt. Auf die Herleitung des Filters wird an dieser Stelle verzichtet, es wird auch lediglich die später verwendete Variante nach Wendel [18] beschrieben. Liegen die Differenzengleichung eines linearen, zeitdiskreten Modells eines stochastischen Systems in der Form

$$\mathbf{x}_{k+1} = \mathbf{A}_k\mathbf{x}_k + \mathbf{B}_k\mathbf{u}_k + \mathbf{G}_k\mathbf{w}_k \tag{2.24}$$

vor, in dem $\mathbf{w}_k$ einem normalverteilten, mittelwertfreien, weißen Rauschenterm entspricht, der das Systemrauschen modelliert, um den Unsicherheiten der Modellbildung Rechnung zu tragen, kann eine Zustandsschätzung mit dem Kalman-Filter stattfinden. Das Rauschen zum Zeitschritt k, das in den Filtervorgang einfließt, wird im Weiteren mit der Kovarianzmatrix $\mathbf{Q}_k$ des Schätzvorganges beschrieben und ist zeitlich unkorreliert mit allen anderen Zeitpunkten.

$$E[\mathbf{w}_i,\mathbf{w}_k^T] = \begin{cases} \mathbf{Q}_k & \text{für } i = k \\ 0 & \text{für } i \neq k \end{cases} \tag{2.25}$$

Etwaiges Rauschen der Eingangsgrößen $\mathbf{u}_k$ kann auch in $\mathbf{Q}_k$ berücksichtigt werden. Weiters ist es notwendig, den Zusammenhang zwischen Zustandsgrößen und Messgrößen, die auch als eine Art Ausgangsgrößen interpretiert werden können, wie folgt ausdrücken zu können:

$$\mathbf{y}_k = \mathbf{H}_k\mathbf{x}_k + \mathbf{V}_k\mathbf{v}_k \tag{2.26}$$

Dabei wird $\mathbf{H}_k$ als Messmatrix, $\mathbf{v}_k$ als das Messrauschen und der gesamte Zusammenhang als Messmodell oder Beobachtungsgleichung bezeichnet. Analog zum Systemrauschen gilt für das Messrauschen:

$$E[\mathbf{v}_i,\mathbf{v}_w^T] = \begin{cases} \mathbf{R}_k & \text{für } i = k \\ 0 & \text{für } i \neq k \end{cases} \tag{2.27}$$

Das Messrauschen und das Systemrauschen werden auch als unkorreliert angenommen.

Auf Systeme, die in diese Form gebracht werden können, ist das Kalman-Filter anwendbar. Wie auch beim Beobachter für deterministische Größen, muss dem Kalman-Filter also sowohl ein Modell des Systems als auch der Zusammenhang mit den Ausgangsgrößen hinterlegt werden. Selbstredend ist man bei der Wahl des Zustandsmodells keineswegs auf Regelstrecken beschränkt, sondern kann alle Zustandsmodelle, welche die obigen Voraussetzungen erfüllen, verwenden.

Aus diesem Grund wird das Kalman-Filter gerne benutzt, um die Informationen mehrerer Sensoren zu fusionieren.

Das Kalman-Filter unterteilt sich strukturell in zwei Schritte, den Estimationsschritt und den Propagationsschritt, die je nach Literatur manchmal auch als Prädiktionsschritt und Korrekturschritt bezeichnet werden. Die Unsicherheit der Schätzung, beschrieben durch die Kovarianzmatrix $\hat{\mathbf{P}}_k$, muss dabei wie die Schätzgrößen rekursiv in der Zeit fortgeführt werden.

Nachfolgend der Prädiktionsschritt/Estimationsschritt der Zustandsgrößen und des Schätzfehlers:

$$\hat{\mathbf{x}}_k^* = \mathbf{A}_k\hat{\mathbf{x}}_k + \mathbf{B}_k\mathbf{u}_k\,, \tag{2.28}$$

$$\hat{\mathbf{P}}_k^* = \mathbf{A}_k\hat{\mathbf{P}}_k\mathbf{A}_k^T + \mathbf{G}_k\mathbf{Q}_k\mathbf{G}_k^T\,, \tag{2.29}$$

und der Korrekturschritt/Propagationsschritt:

$$\hat{\mathbf{x}}_{k+1} = \hat{\mathbf{x}}_k^* + \mathbf{K}_k(\mathbf{y}_k - \mathbf{H}_k\hat{\mathbf{x}}_k^*)\,, \tag{2.30}$$

$$\hat{\mathbf{P}}_{k+1} = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\hat{\mathbf{P}}_k^*\,, \tag{2.31}$$

mit der Kalman-Gain-Matrix $\mathbf{K}_k$ die sich berechnet zu

$$\hat{\mathbf{K}}_k = \hat{\mathbf{P}}_k^*\mathbf{H}_k\mathbf{S}_k^{-1}\,, \tag{2.32}$$

und der Innovationskovarianz

$$\mathbf{S_k} = \mathbf{H}_k\hat{\mathbf{P}}_k^*\mathbf{H}_k^T + \mathbf{V}_k\mathbf{R}_k\mathbf{V}_k^T\,. \tag{2.33}$$

Die Systemmatrizen $\mathbf{A}_k$, $\mathbf{H}_k$, $\mathbf{B}_k$, $\mathbf{V}_k$ und $\mathbf{G}_k$ sind für das gewöhnliche Kalman-Filter bei zeitinvarianten Systemen unveränderlich, eine Indizierung in diesem Fall nicht notwendig. Liegt ein nichtlineares Systemmodell der Form

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k)\,, \tag{2.34}$$

$$\mathbf{y}_k = h(\mathbf{x}_k, \mathbf{v}_k)\,, \tag{2.35}$$

vor, errechnen sich die Matrizen für das Extended-Kalman-Filter wie folgt:

$$\mathbf{A}_k = \frac{\partial f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k)}{\partial \mathbf{x}}\Big|_{\mathbf{x}_k=\hat{\mathbf{x}}_k, \mathbf{u}_k=\mathbf{u}_k, \mathbf{w}_k=\mathbf{0}}\,, \tag{2.36}$$

$$\mathbf{G}_k = \frac{\partial f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k)}{\partial \mathbf{w}}\Big|_{\mathbf{x}_k=\hat{\mathbf{x}}_k, \mathbf{u}_k=\mathbf{u}_k, \mathbf{w}_k=\mathbf{0}}\,, \tag{2.37}$$

$$\mathbf{H}_k = \frac{\partial h(\mathbf{x}_k, \mathbf{v}_k)}{\partial \mathbf{x}}\Big|_{\mathbf{x}_k=\hat{\mathbf{x}}_k^*, \mathbf{v}_k=\mathbf{0}}, \tag{2.38}$$

$$\mathbf{V}_k = \frac{\partial h(\mathbf{x}_k, \mathbf{v}_k)}{\partial \mathbf{v}}\Big|_{\mathbf{x}_k=\hat{\mathbf{x}}_k^*, \mathbf{v}_k=\mathbf{0}}. \tag{2.39}$$

Im Estimations und Propagationsschritt ändern sich die Gleichungen (2.28) und (2.30) entsprechend:

$$\hat{\mathbf{x}}_k^* = f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{0}), \tag{2.40}$$

$$\hat{\mathbf{x}}_{k+1} = \hat{\mathbf{x}}_k^* + \mathbf{K}_k(\mathbf{y}_k - h(\hat{\mathbf{x}}_k^*, \mathbf{0})). \tag{2.41}$$

Auch wenn das Rauschen von Sensoren und damit die Kovarianzmatrix $\mathbf{R}_k$ im Rahmen von Versuchen näherungsweise quantifiziert werden kann, was für das Prozessrauschen $\mathbf{Q}_k$ nicht gilt, erfolgt die Abstimmung von Kalman-Filtern durch Festlegen der zwei Matrizen, das sogenannte Filter-tuning, oft empirisch nach dem Prinzip „trail and error“.

2.4.5 Trägheitsnavigation - inertiale Navigation

Die inertiale Navigation vereint das Konzept der Koppelnavigation (im Englischen: dead reckoning) mit der Technologie der Trägheitssensoren. Als Koppelnavigation bezeichnet man die ungefähre Positionsbestimmung über Kurs und Geschwindigkeit. Will man diese Informationen über Trägheitssensoren ermitteln, sind dazu Accelerometer und Drehratensensoren für jede Raumrichtung erforderlich, also Inertialsensoreinheiten wie schon im Kapitel 2.4.3 erläutert. Sind diese Einheiten fix mit dem gegenständlichen Fahrzeug verbunden, spricht man von Strapdown IMUs, die erst mit der Entwicklung von Inertialsensoren, die nicht auf der Funktionsweise von Kreiselintrumenten mit kardanischer Aufhängung beruhen, ermöglicht wurden [18]. Die Schwachstelle solcher Systemen liegt in ihrer Ungenauigkeit, die durch Aufintegration der Fehler entsteht. Dadurch sind entweder hochgenaue Sensoren notwendig oder man koppelt Inertialnavigationssysteme (INS) mit Absolutpositionierungssystemen, zum Beispiel Satellitennavigationssystemen.

Kopplung INS und GNSS

Inertiale und Satellitennavigationssysteme weisen in vielerlei Hinsicht komplementäre Eigenschaften auf. Bei der relativen Positionsbestimmung inertialer

Systeme kann man bei einer Kopplung mit absolut arbeitenden GNSS das unbegrenzte Anwachsen des Fehlers durch die absoluten Messungen eines GNSS verhindern. Diese oft in niedriger Ausgabefrequenz verfügbaren Positionen des GNSS profitieren im Gegenzug vom meist hochfrequenten Charakter der INS, wodurch sich glatte Bewegungstrajektorien rekonstruieren lassen. Nach Wendel [18] lassen sich die Koppelungsmethoden anhand der zugrundeliegenden Systemarchitekturen nach dem Grad des Systemintegrtion unterteilen, wobei die Unterteilung von den in verschiedener Literatur verwendeten Systembezeichnungen abgeleitet wurde, sodass ein gewisser Spielraum bei der Klassifizierung verbleibt:

- *Loosely Coupled Systems:* Arbeiten das INS und GNSS im Grunde eigenständig nebeneinander und die Informationen des GNSS werden dem INS nur unterstützend zugeführt, liegt diese Integrationsvariante vor. Neben verschiedenen Nachteilen hat diese Methode den Vorteil geringen Entwicklungsaufwandes und ist darum weit verbreitet. Die Fusion der Informationen erfolgt meist über ein Kalman-Filter. Zu Problemen kann es deswegen vor allem dann kommen, wenn die Informationen des GNSS zuvor bereits in einem eigenen Filter verarbeitet wurden und Zeitkorrelationen aufweisen. Im Zusammenhang mit dieser Problematik verweist Wendel [18] auf sogenannte Federated-Filter-Architekturen, auch hier werden sie nicht näher ausgeführt.

- *Tightly Coupled Systems:* Als Tightly Coupled Systems werden gemeinhin welche bezeichnet, bei denen direkt Pseudoentfernungen oder Messungen des Dopplereffekts verarbeitet werden. Findet jedoch keine Stützung des GNSS durch das INS statt, wird manchmal auch von Closely Coupled Systems gesprochen. Die Robustheit steigt gegenüber den Loosely Coupled Systems, nachteilig ist jedoch der wesentlich höhere Entwicklungsaufwand.

- *Ultra-Tight Integration:* Systeme bei denen weder INS noch GNSS den Signalverarbeitungprozess ohne Informationen vom jeweils anderen aufrechterhalten können, gehören zu dieser Kategorie. Die Systeme verschmelzen vollständig zu einem. Der Integrationsaufwand steigt nochmal gegenüber den Tightly Coupled Systems. In der Regel sind solche Systeme nur unter Mitwirkung des GNSS-Empfängerherstellers realisierbar.

3 Methodik

Das bisher Ausgeführte bezog sich teilweise auf weniger spezifische Methoden, die als Teil des Stand der Technik in Robotern eingesetzt werden können, aber wegen ihrer fehlenden Anpassung an landwirtschaftliche Umgebung nicht ohne weiteres für Feldroboter geeignet sind. Einige dieser für die Navigation erforderlichen Grundfunktionen sind bereits in der Grundinstallation des Robot Operating System inkludiert. Diese Algorithmen, die sich unter der Bezeichnung *Navigation Stack* subsumieren, benötigen für ihre Funktion aber zusätzliche, roboterspezifische Algorithmen, die von extern bereitgestellt werden müssen. Die Tools des Navigation Stack wurden dabei für zwei sehr häufige Typen von Robotern entwickelt, für holonome Roboter und solche mit Differentialantrieb, die im vorangegangenen Kapitel näher beschrieben wurden. FRANC gehört keinem dieser Typen an, darum wurde im Zuge dieser Arbeit evaluiert, ob die Funktionen des Navigation Stack dennoch als im Grunde für FRANC geeignet anzusehen sind, wenngleich die Methoden der Funktionen für landwirtschaftliche Zwecke noch anzupassen sind. Diese Tests wurden jedoch nicht am realen Roboter durchgeführt, sondern in einer eigens arrangierten Simulation mit einem Modell des Roboters. Diese Vorgangsweise hat unter anderem die Vorteile besserer Kontrollierbarkeit der Umwelteinflüsse sowie der einfacheren Auswertung der Tests, was insgesamt zu einem wesentlich reduzierten Aufwand für die Tests gegenüber der Durchführung am realen Objekt führt.

3.1 Simulation in Gazebo

Gazebo ist ein Simulator, dessen Entstehungsgeschichte eng mit dem Robot Operating System verknüpft ist. Hinter beiden steht die kalifornische Organisation Open Source Robotics Foundation. Gazebo basiert auf der Open Dynamics Engine, die ebenfalls Open Source zur Verfügung steht. Aufgrund der engen Verbindung zu ROS, existiert eine hohe Kompatibilität zwischen den Systemen, weswegen Gazebo gewählt wurde, um FRANC zu simulieren. Gazebo stellt neben dem eigentlichen Simulator weitere Tools bereit, beispielsweise eine grafische Benutzeroberfläche. Zur einfachen Integration steht für ROS ein Gazebo Package bereit, das alle nötigen Schnittstellen zwischen ROS und Gazebo inkludiert, um einen mit ROS arbeitenden Roboter in Gazebo simulieren zu

können. Abbildung 3.1 gibt einen Überblick über den Umfang des ROS-Gazebo Packages, zusätzlich zum im Bild Aufgeführten bietet es Unterstützung zur Einbindung des ros_control Packages in gazebo - Simulationen.

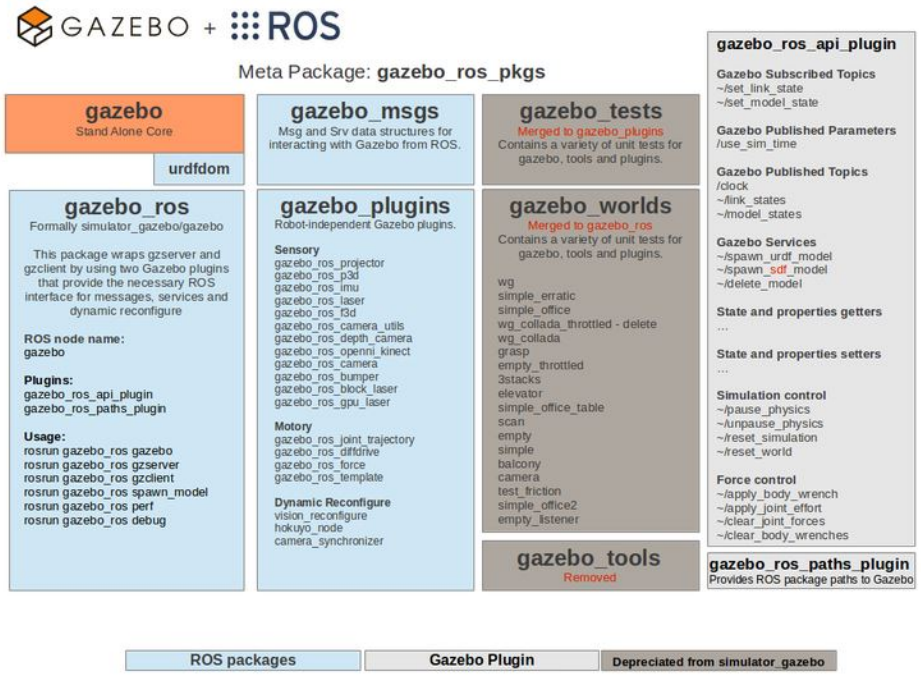


Abbildung 3.1: Übersicht zum ROS-Gazebo Package [19]

3.1.1 Modellierung des Roboters

Die Beschreibung eines Roboters erfolgt in ROS über das XML Format *Unified Robot Description Format (URDF)*. Gazebo benötigt für Simulationen aber vor allem Informationen zu den physikalischen Eigenschaften der Objekte, das dafür vorgesehene Format ist das *Simulation Description Format (SDF)*. Um dennoch Roboter, die in URDF beschrieben sind, mit Gazebo simulieren zu können, ist es möglich in URDF Gazebo-spezifische Informationen mit dem gazebo-tag einzufügen, sodass die Dateien von Gazebo dank eines inkludierten URDF - Parsers ins SDF Format übersetzt werden können. Dadurch wird es möglich sowohl die ROS spezifischen Werkzeuge, wie das grafische Userinterface RVIZ mit dessen Darstellungskapazitäten zu nutzen und gleichzeitig die Simulation in Gazebo mit der modellierten Umgebung zu visualisieren. Die Modellierung dieser Umwelt kann sehr leicht direkt über die grafische Benutzeroberfläche und die online-Objektbibliothek von Gazebo stattfinden. Um die grundlegenden Funktionen des Navigation Stack in Betrieb zu nehmen, ist eine a priori

Kenntnis der Umgebung in Form einer Karte oder dergleichen jedoch nicht zwingend erforderlich, die Modellierung der Umgebungen wird daher nicht näher beleuchtet. Es sei lediglich erwähnt, dass die Umgebungen als *worlds* bezeichnet werden und ebenso als SDF vorliegen müssen. Alle Simulationen in Zusammenhang mit dieser Arbeit fanden mit einfachsten Umweltmodellen statt.

URDF-Modell

Für die Erstellung des Modells wurde die XML-Macro Sprache XACRO verwendet, um durch die Verwendung von Macros eine übersichtlichere Datei zu erhalten. Das erstellte Modell von FRANC wurde auf die für die Simulationen relevanten Funktionen beschränkt, dabei wurden auch die Informationen zu den Massen und Trägheitsmomenten nicht exakt, sondern größenordnungsmäßig hinterlegt. FRANC besitzt wie schon erwähnt vier separat lenkbare Räder. Diese Lenkachsen und die Radlagerungen waren daher im Modell als nicht fixe Verbindungen auszuführen. Die mit einem ROS-Visualisierungstool erzeugte Abbildung 3.2 zeigt die Körper und Gelenke des URDF-Funktionsmodells in Diagrammstruktur.

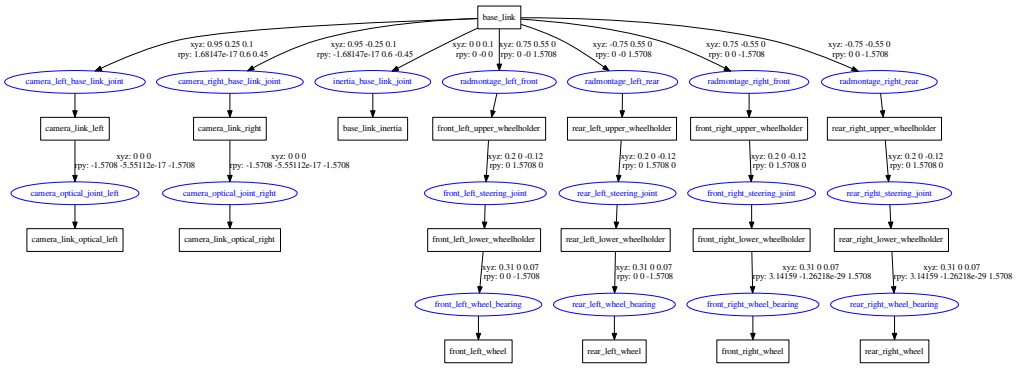
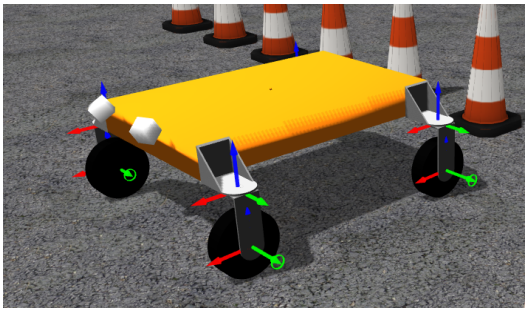
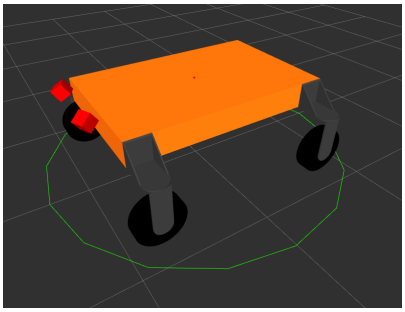


Abbildung 3.2: Diagrammstruktur des URDF - Funktionsmodells

Abbildung 3.3 zeigt die räumliche Darstellung des Modells in Gazebo und RVIZ. In Abbildung 3.3a ist auch ein Ausschnitt der einfachen Umgebung erkennbar, in die ein Asphaltboden und einige Verkehrsleitkegel eingefügt wurden. In dieser Darstellung wurden auch die beweglichen Gelenke des Funktionsmodells sichtbar gemacht, die vertikalen Lenkachsen sowie die horizontalen Radachsen.



(a) Funktionsmodell in Gazebo



(b) Funktionsmodell in RVIZ

Abbildung 3.3: Darstellung des Funktionsmodells

ROS-Plugins

Soll das Funktionsmodell nun über ROS gesteuert, beziehungsweise Sensorinformationen aus der Simulation in ROS verwendet werden, müssen entsprechende Aktuatoren und Sensoren ins Modell integriert werden. Diese Integration erfolgt für Sensoren über gazebo_ros_ plugins, (siehe Abbildung 3.2) die für den jeweiligen Körper, der dem Sensor als Referenz dient, in der URDF Datei angegeben werden müssen. Im Falle eines Sensors werden bei der Einbindung auch gleich die Eigenschaften - sofern veränderbar - des Sensors spezifiziert und der Name des Topics, unter dem die Informationen publiziert werden, festgelegt. In FRANCs Funktionsmodell wurden zwei Kameras integriert, die 3D Informationen in Form von Pointclouds liefern. Der Zweck dieser Kameras wird an späterer Stelle erörtert. Wird das ros_control_plugin für Aktuatoren verwendet, erfolgt die Integration etwas abweichend von dieser Vorgangsweise. In diesem Fall wird das Plugin für den gesamten Roboter ohne Referenz zu einem speziellen Körper integriert. Für die einzelnen Gelenke, die als Aktuatoren agieren sollen, muss dann aber jeweils unter einem transmission-XML-tag der HardwareInterface-Typ festgelegt werden. Dabei ist darauf zu achten, dass nicht jedes HardwareInterface jedem Gelenktyp zugeordnet werden kann. Für das FRANC Funktionsmodell wurden für die Lenkachsen *EffortJointInterfaces* und für die Radachsen *VelocityJointInterfaces* verwendet.

3.1.2 Imitation der Kinematik

In Kapitel 2.4.1 wurde ausgeführt, dass FRANCs kinematisches Lagemodell vom Typ (1,2) ist, also der Grad der Mobilität $\delta_m = 1$ und der Grad der Steuerbarkeit $\delta_s = 2$ ist, was zu einem Grad an Manövrierbarkeit $\delta_M = \delta_m + \delta_s = 3$ führt. Die Werkzeuge des Navigation Stack wurden jedoch für holonome Roboter (Typ (3,0), nicht erläutert) oder solche mit Differentialantrieb (Typ (2,0))

entwickelt. Dieser Umstand macht es notwendig, die Kinematik eines dieser unterstützten Typen mit FRANC so gut es geht zu imitieren, sodass der Navigation Stack trotz verbleibender Abweichungen eventuell verwendet werden kann. Mit FRANCs Grad an Manövrierbarkeit von $\delta_M = 3$ wäre es möglich sowohl einen holonomen Roboter als auch einen mit Differentialantrieb mit gewissen Abschlägen nachzuahmen. Der Aufwand für Letzteren ist aufgrund der einfachen geometrischen Zusammenhänge im kinematischen Lagemodell geringer. Grundsätzlich ist es für Roboter mit gleichem Grad an Manövrierbarkeit möglich, die gleichen Pfade zurückzulegen, jedoch nicht mit demselben zeitlichen Verlauf, weil der integrale Einfluss der lenkbaren Räder im Zeitablauf zu Unterschieden führen kann. Roboter mit dem höheren Grad an Mobilität sind flexibler, da die Lage des momentanen Bewegungspols direkt über $\boldsymbol{\eta}$ festgelegt werden kann. Die Ausgabe des Navigation Stack für Roboter mit Differentialantrieb, wie später noch ausgeführt wird, sind keine Pfade, sondern Geschwindigkeiten in Form einer linearen Geschwindigkeit in Roboterlängsrichtung und einer Winkelgeschwindigkeit um die Hochachse des Roboters. Der diesen Geschwindigkeiten zugrundeliegende Pfad kann daher mit Gewissheit von FRANC auch mit angepasster Kinematik nicht exakt reproduziert werden. Um die Bewegung eines Roboters mit Differentialantrieb zu imitieren, muss die Lage des Bewegungspols (ICR) der Vorgabe so gut es geht nachgeführt werden. Bei Robotern vom Typ (2,0) liegt der momentane Bewegungspol immer auf der gemeinsamen Achse der fixen Räder. Abbildung 3.4 zeigt den geometrischen Zusammenhang zwischen ICR und den Radgeschwindigkeiten. Durch die Lage der Achse und dem Radius R ist die Lage des ICR vollständig bestimmt.

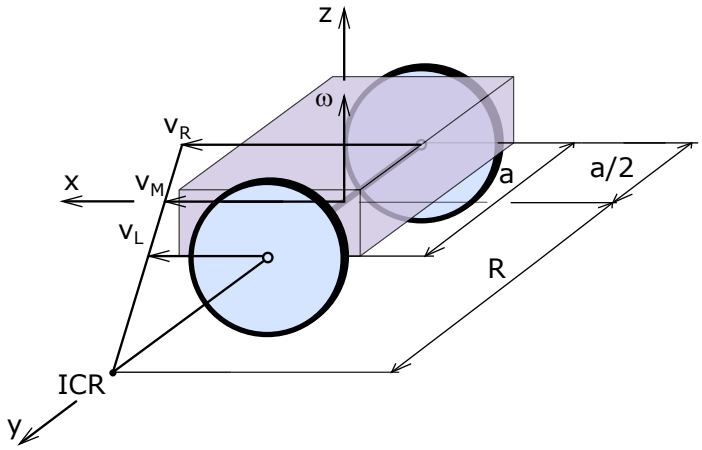


Abbildung 3.4: Bewegungspol für Roboter vom Typ (2,0)

Mit diesem Wissen, der Lineargeschwindigkeit v_M und der Winkelgeschwindig-

keit ω lässt sich die relative Lage des ICR mit der Koordinate R berechnen zu

$$R = \frac{v_M}{\omega} \,. \tag{3.1}$$

Darin entspricht nach Abbildung 3.4 v_M der Roboterlängsgeschwindigkeit. Diese Lage des Bewegungspols relativ zum Roboter muss bei FRANC durch Lenkbewegungen hergestellt werden. R ergibt sich aus der Formel im Roboterkoordinatensystem vorzeichenrichtig. Abbildung 3.5 zeigt die geometrischen Zusammenhänge bei FRANC mit den im Folgenden verwendeten Variablen des Radstandes a und der Lenkwinkel β_i schematisch.

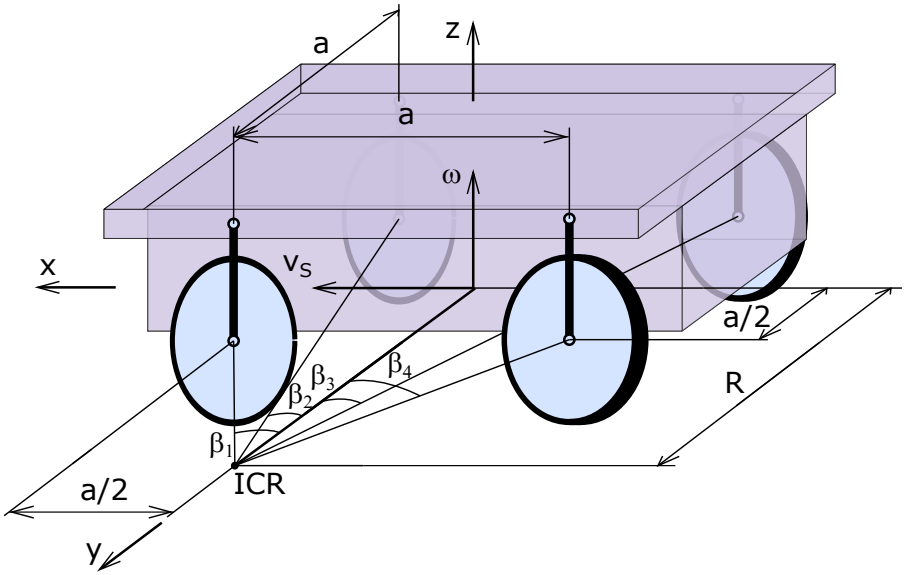


Abbildung 3.5: Bewegungspol bei FRANC

Durch die Festlegung einer fiktiven Achse, die durch den gedachten Mittelpunkt und normal zur Fahrtrichtung verläuft, reduziert sich der Grad der Steuerbarkeit und das kinematische Lagemodell degeneriert zum Typ (1,1), dem Modell für PKW-ähnliche Roboter. Der Grad der Manövrierbarkeit sinkt ebeneso um einen Zähler auf zwei und stimmt dann mit jenem von Robotern mit Differentialantrieben überein. Da bei FRANC alle Lenkachsen im Drehwinkel uneingeschränkt sind, wäre man bei der Lage der fiktiven Achse eigentlich frei, die Wahl durch den gedachten geometrischen Mittelpunkt reduziert jedoch die überstrichene Fläche bei der Kurvenfahrt auf ein Minimum, mit steigender Reduktion bei sinkendem Radius R, was bei der Bewegung durch Pflanzenkulturen ein Vorteil sein kann.

Die Lenkwinkel aus Abbildung 3.5 ergeben sich zu

$$\begin{aligned} \beta_2 = -\beta_3 &= \arctan\left(\frac{\frac{a}{2}}{R+\frac{a}{2}}\right), \\ \beta_1 = -\beta_4 &= \arctan\left(\frac{\frac{a}{2}}{R-\frac{a}{2}}\right). \end{aligned} \tag{3.2}$$

Die Geschwindigkeit jedes Rades errechnet sich aus der gegebenen Winkelgeschwindigkeit des Roboters und dem Abstand des Rades zum ICR. Liegt der Bewegungspol links des Roboters, folgt für die Radgeschwindigkeiten

$$\begin{aligned} v_{v,l} = v_{h,l} &= \omega \cdot \sqrt{\left(\frac{a}{2}\right)^2 + \left(R - \frac{a}{2}\right)^2}, \\ v_{v,r} = v_{h,r} &= \omega \cdot \sqrt{\left(\frac{a}{2}\right)^2 + \left(R + \frac{a}{2}\right)^2}. \end{aligned} \tag{3.3}$$

Dabei stehen die Indizes v und h für vorne/hinten sowie l und r für links/rechts in Roboterlängsrichtung. Liegt der ICR rechts des Roboters, ist auf das korrekte Vorzeichen von R zu achten.
Schreibt man den Radius R für beide kinematischen Modelle in Abhängigkeit der roboterspezifischen Stellgrößen an, lässt sich der Unterschied zwischen den Modellen veranschaulichen. Mittels Strahlensatz lässt sich für den Radius von Typ (2,0) schreiben (siehe Abbildung 3.4):

$$R_{Diff} = \frac{v_L}{\frac{v_R - v_L}{a}} + \frac{a}{2} = \frac{v_L \cdot a}{v_R - v_L} + \frac{a}{2}. \tag{3.4}$$

Für Typ (1,1) mit fiktiver Achse (Abbildung 3.5) ergibt sich mit β_1

$$R_{FRANC} = \frac{\frac{a}{2}}{\tan\beta_1} + \frac{a}{2}. \tag{3.5}$$

Für Geradeausfahrt ($v_R = v_L$) liegt der Bewegungspol für Typ (2,0) im Unendlichen. Bei Stillstand ist die Lage des ICR im Gegensatz zu Typ (1,1) undefiniert. Für FRANC muss die Lage des Bewegungspols in diesem Fall also festgelegt werden, wobei sich zwei spezielle Positionen besonders anbieten. Entweder parallele Räder mit ICR im Unendlichen, was den Vorteil hat, ohne Lenkbewegungen sofort geradeaus fahren zu können, oder aber $R_{FRANC} = 0$, wodurch FRANC ohne Änderung der Lenkwinkel im Stand rotieren kann. Die Wahl fiel auf $\beta_1 = \beta_2 = \beta_3 = \beta_4 = 0$ mit $R_{FRANC} = \infty$. Dadurch ist es für den Fall, dass aus dem Stillstand eine Rotation aus dem Stand ausgeführt werden

soll, notwendig, dass jedes Rad einen gewissen Lenkwinkel zurücklegen muss, ehe die Rotation ausgeführt werden kann. Bei Differentialantrieb hingegen ist es ausreichend, genau entgegengesetzte Beschleunigungen der Räder $\dot{v_L} = -\dot{v_R}$ aufzubringen, woraus $v_L = -v_R$ folgt, sodass sich instantan ein Radius von $R_{Diff} = 0$ einstellt. Werden die Räder aus dem Stillstand jedoch mit $\dot{v_L} = \dot{v_R}$ beschleunigt, folgt daraus unmittelbar $R_{Diff} = \infty$.

3.1.3 Navigation Stack

Der Navigation Stack besteht im Kern aus dem move_base Node. Er stellt teilweise die in Kapitel 2 erläuterten Funktionen bereit, die Pfadplanung, Hindernisvermeidung und die Pfadfolgeregelung inklusive inverser Kinematik. Abbildung 3.6 zeigt den Node übersichtlich mit seinen Schnittstellen und den Informationen, die er zum Betrieb benötigt. Auf die optionalen, zum Betrieb nicht notwendigen Nodes wird im Weiteren nicht eingegangen.

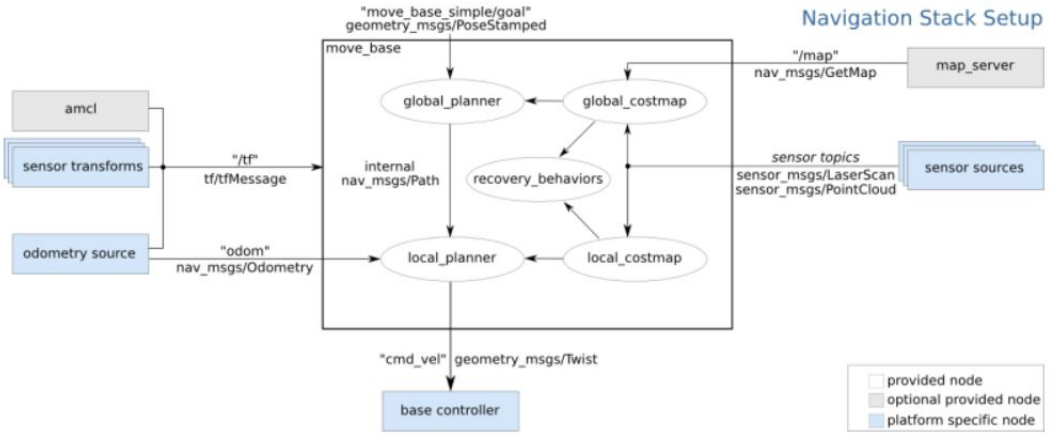


Abbildung 3.6: Übersicht zum Navigation Stack [20]

Odometrie

Im Bild links, die Nodes *odometry source* und *sensor transforms* müssen Informationen über die tatsächlich ausgeführte Bewegung bereitstellen, was nicht zwingend mit der Methode der Odometrie erfolgen muss, bei anderweitiger Methodik muss jedoch ihre Charakteristik und der Einfluss auf das in Abbildung 2.4 dargestellte System bedacht werden. Die Informationen, die dem move_base Node über diese Schnittstellen zugeführt werden, entsprechen im Regelkreis jenen, die aus der *Messeinrichtung* rückgeführt werden.

Umgebungserfassung

Für die Aufgabe der Hindernisvermeidung sind Sensoren zur Erfassung der Umgebung obligatorisch, darum verlangt der move_base Node nach *sensor sources*, die in zwei möglichen Formen, Pointcloud oder Lasescan, bereitgestellt werden können. Der Navigation Stack akzeptiert dabei natürlich auch mehrere Sensoren gleichzeitig.

Zieleingabe

Der Navigation Stack verfügt über eine Action - Programmierschnittstelle, basierend auf dem actionlib Package. Dieses Package ist dazu gedacht, dass beim Ausführen umfangreicher Aufgaben dem Auftraggeber Informationen über den Aufgabenfortschritt nach der Eingabe eines Aufgabenziels zur Verfügung stehen. Über diese Schnittstelle ist es dem Aufgabengeber auch möglich eine Aufgabenerteilung zu stornieren. Nach Beendigung einer Aufgabe ist es dem ausführenden Teil möglich dem Auftraggeber Informationen über das Ergebnis zurückzuliefern. Wie bei den *Services* gibt es eine Client-Anwendung und eine Server-Anwendung. Der beschriebene Informtionsaustausch erfolgt im Falle des Navigation Stack über vier Topics: move_base/goal, move_base/cancel, move_base/feedback, move_base/status. Für den Fall, dass man am Prozessfortschritt nicht interessiert ist, gibt es zur einfacheren Ansteuerung zusätzlich ein Topic „move_base_simple/goal“, das eine simple Zieleingabe ermöglicht aber keine Rückinformationen an den Aufgabengeber vorsieht.

Bewegungsausführung

Wie schon erwähnt gibt der Navigation Stack das Ergebnis seiner Berechnungen in Form von Geschwindigkeiten im Roboterkoordinatensystem über das Topic „cmd_vel“ aus. Es muss daher ein *base controller* vorhanden sein, der diese Geschwindigkeitsbefehle an die Hardware weitergibt. Im vorliegenden Anwendungsfall erfolgt die Weiterleitung via ros_control plugin an in Gazebo simulierte Hardware.

3.1.4 Prozesse der Simulation

Eine Übersicht über alle während der Simulation aktiven Nodes gibt Abbildung 3.7. Die Grafik wurde mit dem ROS Tool *rqt_graph* erstellt. Zur besseren Unterscheidung wurden rund um Nodes (oval) ihre Namespaces (eingefärbte Rechtecke) eingezeichnet, sodass die Topics (weiße Rechtecke) in der Grafik automatisch entsprechend ihrer Zugehörigkeit platziert wurden. Anhand dieser Gruppierungen werden im Folgenden auch kurz die Zusammenhänge zum bisher

Erläuterten hergestellt.

- *move_base, rotes Rechteck rechts:* Der Kern des Navigation Stack, die parallel laufenden Hilfsprozesse, wie soeben in Kapitel 3.1.3 ausgeführt, entsprechen den blau eingefärbten Rechtecken.

- *move_base_simple, blaues Rechteck über rotem Rechteck:* Enthält nur das Topic, mit dem der Navigtion Stack seine einfachen Ziele erhält, entspricht der *Zieleingabe*. RVIZ ermöglicht es dieses Ziel grafisch über die Benutzeroberfläche einzugeben.

- *franc_diff_drive, blaues Rechteck ganz links:* Nimmt die Geschwindigkeitsbefehle des Navigation Stack über „cmd_vel“ entgegen und berechnet daraus mittels inverser Kinematik entsprechend Kapitel 3.1.2 die Bewegungen für die simulierten Aktuatoren. Die Kommunikation mit den Aktuatoren erfolgt über Topics, die an die Controllern im grünen Rechteck rechts daneben gesendet werden. Diese Vorgänge lassen sich unter *Bewegungsausführung* zusammenfassen.

- *franc_description, linkes, grünes Rechteck:* Unter diesem Namespace subsumieren sich alle Topics, die Informationen an die in Gazebo simulierten Aktuatoren liefern oder von ihnen empfangen. Die Controller der Aktuatoren werden mithilfe des Hilfsscripts *controller_spawner* gestartet.

- *gazebo, rechtes, grünes Rechteck:* Hier erfolgt die Simulation mit der Open Dynamics Engine. Die Schnittstellen über die gazebo_ros_plugins sind die soeben erwähnten Topics der Controller, die nicht nur die Eingangsgrößen übermitteln, sondern auch die tatsächlichen Positionen zurückgeben, wie auch die dreidimensionalen Kamerabilder in Form von Pointclouds, die unter dem Namespace *camera* publiziert werden.

- *camera, pointcloudconverterleft, pointcloudconverterright, blaue Rechtecke oben, rechts:* Die aus der Simulation erhaltenen Pointclouds werden in das vom Navigation Stack geforderte Format konvertiert und dem move_base Node zugeführt. Diese Gruppe führt die Aufgaben der *Umgebungserfassung* aus.

- *map_ododm_tf_broadcaster, blaues Rechtecke ganz unten:* Stellt die aktuelle mit *Odometrie* ermittelte Ortsinformation als Transformation via /tf bereit. Das tf-Package dient in ROS dazu, die Transformationen zwischen allen vorkommenden Koordinatensystemen zu verwalten. Die odometrische Berechnung erfolgt eigentlich im franc_diff_drive Node über einfache

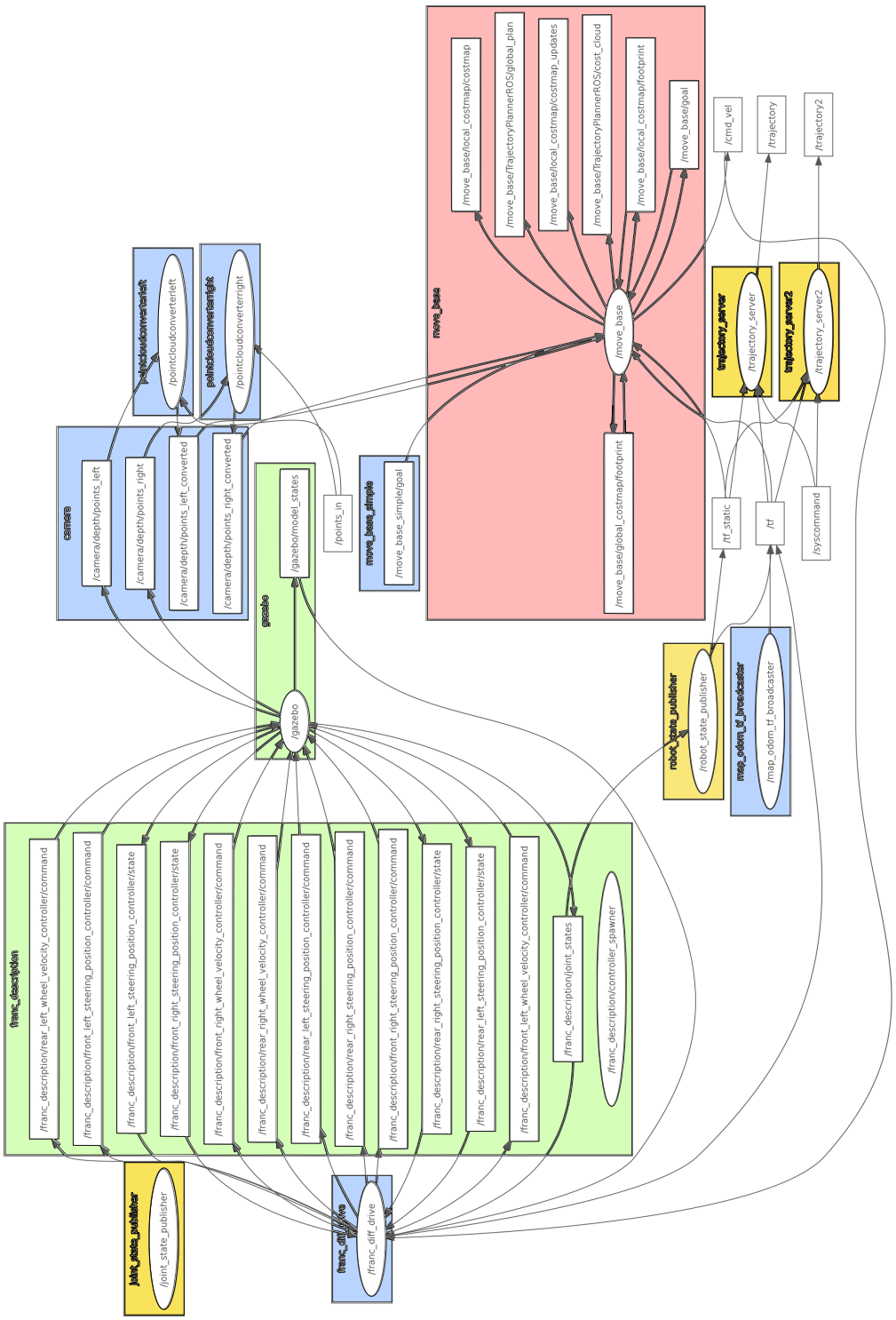


Abbildung 3.7: An der Gazebo-Simulation beteiligte Nodes

numerische Integration. Durch diese Methode pflanzen sich Fehler in der Berechnung, wie schon in Kapitel 2.4.5 erläutert, fort, sodass die Ergebnisse mit wachsender Wegstrecke immer ungenauer werden. Dieser Effekt ist im Zuge der Simulation durchaus gewollt, um die Auswirkungen ungenauer Lokalisierung auf die Navigation abschätzen zu können.

- *joint_state_publisher, robot_state_publisher, trajetory_server, trajectory_server2, gelbe Rechtecke:* Diese Nodes führen ausschließlich Hilfsdienste aus, um etwa die Darstellung des Roboters in RVIZ zu ermöglichen, sind aber ansonsten an der Simulation nicht beteiligt.

3.1.5 Parameter der Simulation

Auf die im Navigation Stack ablaufenden Berechnungen zur Pfadplanung und Hindernisvermeidung kann über implementierte Parameter Einfluss genommen werden. Das ROS Tool zur Paramterverwaltung, der Parameter Server, stellt die für den Betrieb des Navigation Stack erforderlichen Parameter systemweit bereit. In der Praxis erfolgt die Bekanntgabe dieser Parameter über sogenannte YAML Dateien im Launch-File mit dem die Simulation gestartet wird. Launch-Files, werden im XML Format verfasst und für das roslaunch Package benötigt, es ermöglicht unter anderem das gleichzeitige Starten mehrerer Nodes und auch das Setzen von Parametern. Neben der Nutzung des Servers für die Parameter des move_base Nodes greifen auch die gazebo_ros_plugins auf die Kapazitäten des Parameterservers zurück. Um das ros_control Plugin nutzen zu können, muss das gesamte URDF File des Roboters in den Parameterserver geladen werden. Zusätzlich verlangt das Plugin noch weitere Angaben zu den Controllern, die ebenfalls mittels eines YAML Files in den Parameterserver geladen werden.

Parameter des move_base Nodes

Werden notwendige Parameter nicht festgesetzt, so sind für die meisten dieser Parameter Standardwerte hinterlegt, auf die der move_base Node zurückgreifen kann. In Abbildung 3.6 findet man Angaben über die Schnittstellen, über die der Node seine Informationen bezieht, für die Verbindung zur Zieleingabe, der Bewegungsausführung und der Odometrie sind dabei auch die Namen der Topics unter Anführungszeichen angegeben. Bei der Verbindung zu den *sesor sources* kann aufgrund der unbekannten Anzahl an Topics, die Informationen zur Umgebung liefern, kein Name von vornherein angegeben werden. Diese Namen, die Sensortypen und auch die zugehörigen Koordinatensysteme, in denen

die Informationen dargestellt sind, müssen dem move_base Node über den Parameterserver mitgeteilt werden. Um die Aufgabe der Hindernisvermeidung zufriedenstellend ausführen zu können, müssen Form und Maße des *footprint* des Roboters als Parameter festgelegt werden. Wie in Kapitel 2.4.2 näher ausgeführt, ist der Navigationsregler im vorliegenden Regelkreis von der Physik der Stellglieder entkoppelt, um dennoch die Leistungsfähigkeit der Motoren im Betrieb nicht zu überschreiten, ist es möglich und ratsam die maximal zulässigen Beschleunigungen und Geschwindigkeiten im Parameterserver für den move_base Node zu hinterlegen. Der maximalen Beschleunigung der Winkelgeschwindigkeit wird dabei im konkreten Fall zusätzliche Bedeutung zuteil. Angesichts der Abweichungen zwischen dem kinematischen Modell, das dem Navigation Stack für die Pfadplanung und Pfadfolgeregelung zugrunde liegt und der Imitation dieses Modells durch das Funktionsmodell wie in Kapitel 3.1.2 beschrieben, lässt die Kenntnis der Formel für die Lage des Bewegungspols (3.1) nämlich darauf schließen, dass große Beschleunigungen zu erhöhten Abweichungen führen. Eine hohe Winkelbeschleunigung wirkt sich dabei insbesondere auch bei Drehungen des Roboters im Stand aus, da als Ruhestellung der Räder des Funktionsmodells parallele Räder gewählt wurden. Die Totzeit, die das Funktionsmodell benötigt, um die Räder für Rotationen am Stand auf $R_{FRANC} = 0$ zu bewegen, bewirkt eine Differenz zwischen dem Sollwert θ_{soll}, eigentlich vorgegeben für Roboter mit Differentialantrieb, welche den Bewegungspol instantan festlegen können, und dem zurückgelegtem Winkel θ_{tat}. Ist die Winkelgeschwindigkeit zufolge hoher Winkelbeschleunigung während dieser Zeitspanne groß, ergibt sich eine entsprechend große Winkeldifferenz. Mit einer festgelegten, kleinen maximalen Winkelbeschleunigung kann die Differenz und eventuelle Auswirkungen auf den Regelalgorithmus des Navigation Stack gering gehalten werden.

3.2 Feststellung der Eignung des Navigation Stack für FRANC

Im Zuge der Simulation wurden dem Roboter über /move_base_simple/goal Ziele vorgegeben, um so zu testen, ob das Funktionsmodell trotz Abweichungen vom idealen kinematischen Verhalten diese Ziele mit Hilfe des Navigation Stacks erreicht. Bei diesen Tests wurden keine Karten der Umgebung verwendet, sodass für die Pfadplanung keine a priori Kenntnisse über Hindernisse verfügbar sind. Beim Treffen auf solche ist es dem Roboter daher auch nicht möglich, diese zu umfahren, weil keine Informationen für den Raum rund um das Hindernis vorliegen, sondern lediglich das Erkennen und Vermeiden einer

Kollision. Die Pfadplanung erfolgt daher in dieser Konfiguration mittels mehr oder minder geradem Weg vom aktuellen Standort zum Ziel mit Anpassung der Orientierung. In Abbildung 3.8 ist das Verhalten des Funktionsmodells für die Vorgabe mehrerer Navigationsziele durch Trajektorien illustriert. Die gelben Pfeile im Bild entsprechen den sequenziell erteilten Zielen, die Vorgabe eines neuen Zieles erfolgte dabei sobald das jeweils aktuelle Ziel erreicht worden war. Bei der Interpretation des Bewegungsverhaltens muss beachtet werden, dass die Position des Roboters in RVIZ über die Transformation des Roboterkoordinatensystems *base_link* zum Koordinatensystem der Umgebung *odom* festgelegt wird. Diese Transformation entspricht auch jener Information, die dem Navigation Stack als Roboterposition bekannt ist. Der Verlauf der Position von *base_link* in *odom* wurde mit dem grünen Pfad in Abbildung 3.8 dargestellt. Dieser Verlauf ist also jener, der aus Sicht des Navigation Stack zurückgelegt wurde. Wie man erkennt, ist das Funktionsmodell mit dem Navigation Stack durchaus in der Lage vorgegebene Ziele anzufahren. In derselben Abbildung wurde der vom Roboter tatsächlich in Gazebo zurückgelegte Pfad als rote Trajektorie eingezeichnet. Die Differenz zwischen dem tatsächlichen Weg und dem grünen, errechneten Pfad ist allein auf die verwendete Methode der Odometrie zurückzuführen. Einmal eingeflossene Fehler in der Berechnung durch ungenaue Messungen, Messrauschen, Abweichungen des zugrunde gelegten Modells vom realen Verhalten oder Ungenauigkeiten durch numerische Berechnungen pflanzen sich fort und werden ohne weitere Maßnahmen nicht korrigiert. Die Fehlerfortpflanzung ist deutlich beim Vergleich von roter und grüner Trajektorie erkennbar. Da dieser Fehler im Regelkreis dieselbe Übertragungsfunktion wie die Führungsgröße besitzt, schlägt er sich im vollen Maße in der Navigationsgenauigkeit nieder. Das nicht ideale Verhalten durch die Imitation der Differentialantriebskinematik wird hingegen wie eine Störgröße behandelt und durch den Regler weitestgehend eliminiert. Zur Illustration wurden für Abbildung 3.9 die als Stellgrößen ausgegebenen Geschwindigkeiten numerisch integriert, um den fiktiven Weg (hellblaue Trajektorie) eines idealen Roboters darzustellen.

Die Fehlercharakteristik der Odometrie erlaubt es in der Regel nicht, diese Methode in der Robotik alleinstehend für die Eigenlokalisierung einzusetzen.

3.3 Implementierung von Filteralgorithmen zur Roboterlokalisierung

In Kapitel 2.4.3 wurden Systeme zur Roboterlokalisierung aufgeführt, die aufgrund ihrer Eigenschaften alleinstehend für die Roboterlokalisierung in land-

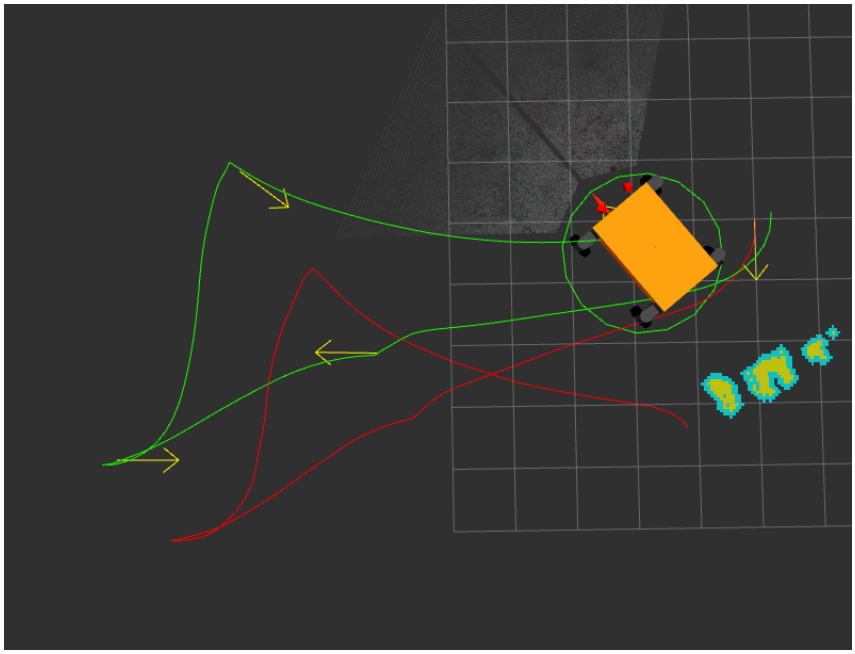


Abbildung 3.8: Navigation des Funktionsmodells zu vorgegebenen Zielen
Grüner Pfad: Der dem Roboter zurückgemeldete Pfad
Roter Pfad: Der tatsächlich in Gazebo zurückgelegte Pfad

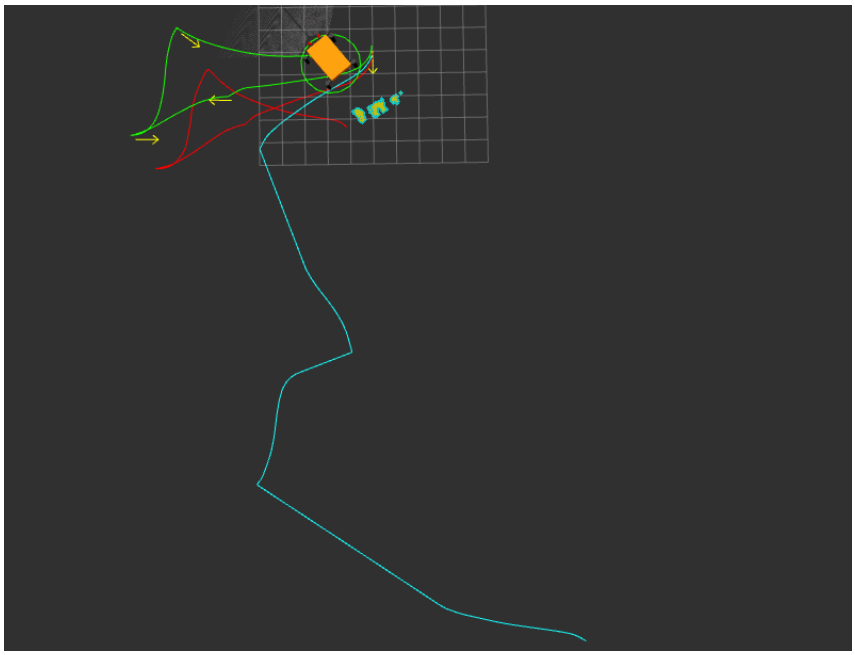


Abbildung 3.9: Fiktiver Weg (blauer Pfad) eines idealen Roboters mit den Stellgrößen desselben Navigationsszenarios wie in Abbildung 3.8

wirtschaftlichen Umgebungen eingesetzt werden können. In der Praxis ist dafür der Einsatz teurer RTK-GPS Receiver alternativlos, was zwangsweise die Frage nach der Leistungsfähigkeit kostengünstigerer Systeme aufwirft, respektive nach Möglichkeiten diese zu steigern. Um mit kostengünstigen Sensoren, in unstrukturierter, landwirtschaftlicher Umgebung, die tatsächliche Bewegung eines Roboters genau genug zu erfassen, um eine Navigation in zufriedenstellender Qualität zu ermöglichen, müssen verschiedene Sensortypen mit geeigneten Methoden kombiniert werden, sodass sich ihre vorteilhaften Eigenschaften im Ergebnis abbilden. Dadurch soll einerseits die Genauigkeit der Positionsbestimmung steigen, aber auch das zeitliche Verhalten der Informationen verbessert werden.
Die in Kapitel 2.4.3 beschriebenen Sensoren, weisen zum Teil in hohem Maße komplementäre Eigenschaften auf, wodurch sie für eine Fusion prädestiniert scheinen. Tabelle 3.1 zeigt die Eignung ausgewählter Sensoren in Hinblick auf für die Positionsbestimmung relevante Bestimmungsgrößen und Charakteristiken in vier Stufen (- - schlecht, - mäßig, + gut, ++ sehr gut). Diese Bewertung gilt nicht für hochpräzise Trägheitssensoren mit entsprechendem Preis.

Sensor	Geschwindigkeit und inkrementelle Positionsänderung	Lagewinkel des Roboters im Raum	Absolutgenauigkeit der Position bzw. Fehlerfortpflanzung
Drehgeber und Odometrie	++	-	-
Inertiale Messeinheit mit Kompass	-	++	- -
GPS mit kostenloser SBAS Unterstützung	-	- -	+

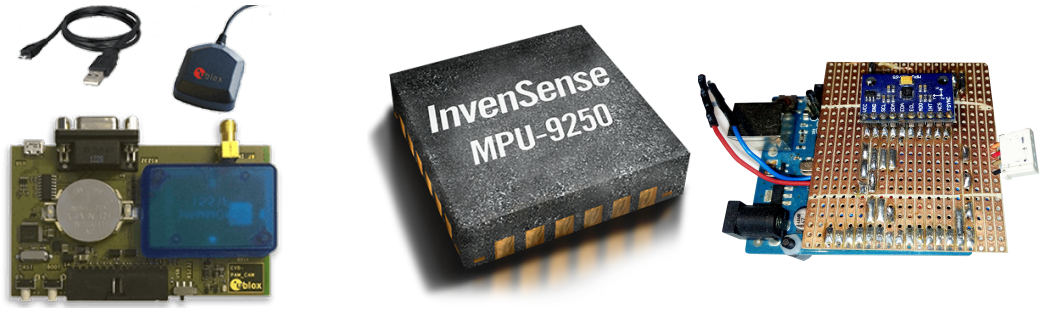
Tabelle 3.1: Stärken und Schwächen ausgewählter Sensoren und Methoden zur Positionsbestimmung

Auf Grundlage dieser unterschiedlichen Merkmale wurden im Zuge dieser Arbeit Filter für kostengünstige Sensoren in ROS implementiert, sodass auf Basis

des Ergebnisses evaluiert werden kann, ob die vorteilhaften Eigenschaften kostengünstiger Sensoren möglicherweise trotz relativ niedriger Qualität der Messungen zu einer Eignung für Navigationszwecke führen.

3.3.1 Auswahl der Sensoren

Da diese Arbeit an ein bereits bestehendes Projekt anknüpft, war die Auswahl der Sensoren teilweise schon getroffen. Die im Antriebsstrang von FRANC ohnehin vorhandenen Drehgeber können natürlich auch für navigatiorische Zwecke odometrisch eingesetzt werden. Zusätzlich war bei Abschluss des Projektes ein noch nicht ins System eingebundener GPS Empfänger vorhanden, der für FRANCs Navigationssystem vorgesehen war. Dabei handelt es sich um ein Gerät der Firma u-blox vom Typ EVK-M8QCAM (siehe Abbildung 3.10a) und ist ab rund 160 Euro erhältlich. Dieser, für GPS Empfänger doch recht hohe Preis, ist eine Folge der zusätzlichen Möglichkeiten, die er dem Nutzer als Evaluation-Kit - Ausführung bietet. Er unterstützt mehrere Augmetierungssysteme, wobei nur das europaweite EGNOS im Zuge dieser Arbeit verwendet wurde, das auch von vielen preisgünstigen Empfängern unterstützt wird. u-blox gibt für den Empfänger eine Genauigkeit von 2,5m CEP, 50% bei einer 24 stündigen, statischen Messung an [21].



(a) Verwendeter GPS-Empfänger [22] (b) Verwendete IMU [23] (c) IMU auf Breakout-Board mit Arduino

Abbildung 3.10: Verwendete Sensoren

Zusätzlich zu diesen vorhandenen Sensoren bietet sich zur Verbesserung der Eigenschaften des Gesamtsystems nach Tabelle 3.1 als sehr gute Ergänzung eine inertiale Messeinheit mit Kompass an, in dieser Kombination oft als *Attitude and Heading Reference System (AHRS)* oder *Magnetic, Angular Rate and Gravity (MARG)* Sensoren bezeichnet, vorausgesetzt die Daten werden mit entsprechenden Methoden verarbeitet. Zu diesem Zweck wurde die in Abbildung 3.10b dargestellte kostengünstige Messeinheit MPU 9250 der Firma

TDK Invensense, bereits für unter zehn Euro erhältlich, ausgewählt. Derartig günstige IMUs sind für die einfache Strapdown-Montageart konzipiert. Sie besteht bei einer Gehäusegröße von 3x3x1mm aus drei *mikroelektromechanischen Systemen (MEMS)*, einem dreiachsigen Accelerometer, einem dreiachsen Gyroskop, sowie einem dreiachsigen, auf dem Halleffekt basierenden, digitalen Magnetometer. Im Folgenden ist mit „IMU“, „Inertialsensor“oder Änlichem der ausgewählte Sensor gemeint, wenngleich er über ein zusätzliches Magnetometer verfügt. Der Bauteil mit dem Chipgehäuse vom Typ „Quad Flat No Leads Package (QFN)“ ist für die Oberflächenmontage auf Leiterplatten vorgesehen. Das Produkt richtet sich eigentlich speziell an Hersteller tragbarer Geräte wie Tablets und Smartphones [23], wo die Montage in aller Regel vollautomatisiert erfolgt. Für die Nutzergemeinschaft von Mikrocontrollern und Ähnlichem ist die IMU aber auch bereits montiert auf Breakout-Boards verschiedener Hersteller verfügbar. Bei diesen Boards handelt es sich um einfache Leiterplatten zur Aufnahme einzelner oder mehrerer Elektronikbauteile, bei denen wichtige Verbindungen, beispielsweise die Spannungsversorgung oder Kommunikationspins, zur leichteren Handhabung auf eine leicht von Hand verlötbare Größe herausgeführt werden. Häufig sind weitere Bauteile verbaut, sodass die Integration für den Endnutzer erleichtert wird. Um die Messwerte der Einheit in ROS verarbeiten zu können, wurde zur Übertragung ein Mikrocontroller der Firma Arduino zwischengeschaltet. Arduino ist ein kommerzielles Unternehmen, das wie die Open Source Robotics Foundation ebenso der Open Source Produktphilosophie folgt. Als klassisches Einstiegsprodukt bietet die Firma den sehr weit verbreiteten Arduino UNO an. Er ist für unter 20 Euro erhältlich und wurde auch als Datenübertragungsgerät für die IMU herangezogen. Die Notwendigkeit eines Microcontrollers zur Datenübertragung ergibt sich aus den von der gewählten IMU unterstützten Datenübertragungssystemen I2C und SPI. Diese Bus-Systeme werden in erster Linie zur Kommunikation zwischen integrierten Schaltkreisen verwendet, weswegen gewöhnliche Computer normalerweise keine solche Schnittstelle aufweisen. Der UNO hingegen unterstützt beide Bus-Systeme und verfügt auch über einen USB Port, wie ihn die meisten Computer auch besitzen. Für Arduino Mikrocontroller steht darüber hinaus in ROS auch ein Package bereit, das in die Arduino - Entwicklungsumgebung eingebunden werden kann, wodurch der Mikrocontroller imstande ist, Topics über die USB Schnittstelle zu publizieren. Computerseitig ist die Verwaltung und das Empfangen von Daten über serielle Ports in ROS mit dem rosserial Package möglich. Abbildung 3.10c zeigt das Breakout-Board mit montierter IMU, aufgelötet auf einer Leiterplatte mit Pins zur Steckmontage auf einem Arduino UNO. Zusätzlich wurde ein Stecker (blaues und rotes Kabel links) für die Stromversorgung mit Batterie, sowie ein Stecker (weißer Stecker rechts) für ein Bluetooth Modul angebracht, sodass die Daten der IMU auch drahtlos

übermittelt werden können. In dieser Konstellation sind Messungen mit einer Rate von zirka 40Hz möglich, was für eine inertiale Messeinheit wenig ist und zur Messung hochdynamischer Bewegungen nicht geeignet wäre. Da bei einem Feldroboter solche allerdings nicht zu erwarten sind, sind 40 Messungen pro Sekunde ausreichend.

3.3.2 Kalibrierung der IMU und des Magnetometers

Eine inertiale Messeinheit misst in Inertialsystemen Größen vektorieller Natur relativ zu den Koordinatensystemen der einzelnen Sensoren. Aufgrund der Erdrotation sind mit der Erdoberfläche verbundene Bezugssysteme zwar keine Inertialsysteme, sie werden für Messungen auf der Erde meist aber dennoch näherungsweise als solche betrachtet. Das verwendete Bezugssystem in dem die zu messenden physikalischen Größen auftreten, ist dabei in der Regel nicht frei fallend, wodurch das Accelerometer der inertialen Messeinheit eine zusammengesetzte Beschleunigung aus Schwerebeschleunigung und der Beschleunigung der Messeinheit relativ zum Bezugssystem misst. Durch die Rotation der Erde sind auch die Messungen des Gyroskops verfälscht, wenn man davon ausgeht, nur an den Drehbewegungen der IMU relativ zum Bezugssystem interessiert zu sein. Magnetometer messen die magnetische Flussdichte an jener Stelle im Bezugssystem, an dem sie sich befinden. Neben dem Magnetfeld der Erde können zusätzliche Magnetfelder vorhanden sein, die sich mit dem der Erde überlagern. Bevor die Messeinheit kalibriert werden kann, muss für die einzelnen Sensoren ein Messmodell erstellt werden, das den Unvollkommenheiten des Messungsvorganges Rechnung trägt. Durch Bestimmung der Fehlerhaftigkeit im Zuge der Kalibrierung können die späteren Messungen im Normalbetrieb korrigiert werden. Metge et al. [24] geben neben der Vorstellung des eigenen Kalibrieralgorithmus' auch einen guten Überblick zum Stand der Technik der Kalibiermethoden. Um die Messwerte der drei Sensoren gemeinsam verarbeiten zu können, müssen die einzelnen Messergebnisse in einem gemeinsamen Referenzkoordinatensystem vorliegen. Für die Transformationen der Messwerte vom Sensorkoordinatensystem in ein gemeinsames existieren mehrere Herangehensweisen. Eine sehr einfache aber in der Praxis durchaus übliche ist die Zugrundelegung der Hypothese, dass die tatsächliche Lage der Sensorkoordinatensysteme den Herstellungsvorgaben sehr nahe kommt, sodass Lageabweichungen der Sensorkoordinatensystemen zueinander vernachlässigt werden können. Für die nachfolgenden Berechnungen wurde von dieser Methode Gebrauch gemacht. Alternativ zu dieser Vorgangsweise müssen Rotationsmatrizen zur Beschreibung der Lage der Koordinatensystemen zueinander im Zuge der Kalibrierung berechnet werden.
Wie auch für Metge et al. [24], [25], [26], [27] diente als Basis für die Kali-

brierung der drei Sensoren ein affines Messmodell, in dem die meisten Fehler berücksichtigt werden können.

$$\mathbf{m}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{b}(t) + \mathbf{n}(t) \tag{3.6}$$

In diesem Modell entspricht $\mathbf{x}(t)$ der zu messenden vektoriellen physikalischen Größe im Koordinatensystem des Sensors. Die Matrix $\mathbf{A}$ und der Vektor $\mathbf{b}(t)$ beinhalten Parameter um die Fehler der Messung abzubilden und müssen für jeden Sensor ermittelt werden. Der Vektor $\mathbf{n}(t)$ modelliert das als mittelwertsfrei angenommene Sensorrauschen. Mit dem gewählten Messmodell und den in $\mathbf{A}$ und $\mathbf{b}(t)$ enthaltenen Parametern ist es möglich, neben den Unvollkommenheiten, die bei allen drei auftreten, auch sensorspezifische zu modellieren. Näheres dazu in den einzelnen Unterabschnitten.

Betrachtet man $\mathbf{A}$ als Ergebnis einer Kombination verschiedener Effekte, lässt sich eine Zerlegung durchführen, sodass man Matrizen erhält, welche die einzelnen Fehler separat abbilden. Dabei unterscheiden sich die Ansätze in der Anzahl und Art der Fehler die modelliert werden. (3.7) zeigt die Zerlegung in zwei Effekte die jedenfalls bei allen drei Sensoren auftreten.

$$\mathbf{A} = \mathbf{A}_s\mathbf{A}_o \tag{3.7}$$

Beide Matrizen haben die Dimension 3x3. $\mathbf{A}_s$ ist eine Diagonalmatrix zur Modellierung der unterschiedlichen Sensorempfindlichkeiten in den drei Achsenrichtungen. Mit der zweiten Matrix $\mathbf{A}_s$ berücksichtigt man Nichtorthogonalität und Fehlausrichtungen der Koordinatenachsen des Sensorkoordinatensystems gegenüber dem der Sensoreinheit. Wenngleich dieser Fehler mit Gewissheit auftritt, kann seine Berechnung bei starkem Sensorrauschen problematisch sein, sodass von seiner Modellierung eventuell abzusehen ist.

Es gibt verschiedene Verfahren, um die Parameter des Messmodells zu ermitteln, Metge et al. [24] unterscheiden dabei in konventionelle Methoden mit Einsatz von teurem Equipment und in weniger restriktive Methoden, speziell für die Kalibrierung von Magnetometer und Accelerometer. Geht es um die Berechnung der Kalibrierparameter sehr präziser und damit auch teurer Sensoren, ist die Verwendung ebenso hochqualitativer Referenzinstrumente dafür naheliegend. Für sehr günstige Messeinheiten steht aufgrund der generell niedrigeren Qualität ein solcher Aufwand jedoch in keiner Relation zur Verbesserung des Ergebnisses gegenüber weniger aufwändigen Methoden.

Die weniger restriktiven Methoden verzichten auf den Einsatz externer Ausrüstung, vielmehr benutzen sie natürlich vorkommende physikalische Größen zur Kalibrierung. Bei Accelerometern in Ruhe wirkt nur die Schwerebeschleunigung auf den Sensor, daher kann sie wegen ihrer bekannten Eigenschaften zur Kalibrierung benutzt werden. Analog kann das natürliche Magnetfeld unseres

Planeten zur Berechnung der Kalibierparameter des Magnetometers herangezogen werden. In der Literatur findet man darüber hinaus oftmals den Ansatz, dass nur der Betrag der natürlichen vektoriellen Größen als konstant und bekannt angenommen werden. Das Messergebnis aller drei Achsen des Sensors muss daher unabhängig von seiner Lage im Betrag mit dem bekannten Wert oder einem Vielfachen davon übereinstimmen.

Kalibrierung des Magnetometers - Ellipsoid-fitting

Magnetometer messen die magnetische Flussdichte, eine gerichtete Größe mit der Einheit Tesla. Im Gegensatz zu Accelerometer und Gyroskop können bei Messungen der magnetischen Flussdichte äußere Einflüsse der unmittelbaren Sensorumgebung nicht per se ausgeschlossen werden, da Magnetismus in physikalischem Zusammenhang mit der molekularen Struktur von Werkstoffen und elektrischem Strom - mit dem der Sensor auch selbst arbeitet - steht, die sich beide der menschlichen Wahrnehmung entziehen. Beschleunigungen und Drehbewegungen als äußere Einflüsse können dagegen - abgesehen von der Erdrotation - auch ohne technische Hilfsmittel gut wahrgenommen und in sehr guter Näherung auch eliminiert werden. Dem Messmodell des Magnetometers ist daher besondere Beachtung zu schenken. Ausgehend von der Annahme, dass sich der Roboter mit der Sensoreinheit im ungestörten Magnetfeld der Erde befindet, verbleibt dennoch die Modellierung magnetischer Störungen verursacht durch die Einheit und den Roboter selbst. Diese magnetischen Einflüsse unterteilt man in solche, die selbst einen magnetischen Fluss erzeugen, der sich mit dem des Erdfelds überlagert (Hard Iron Effekt) und jene, die durch Wechselwirkungen mit dem äußeren Magnetfeld hervorgerufen werden. Dieser Effekt hängt von der magnetischen Permeabilität der Werkstoffe ab und tritt daher in erster Linie bei ferromagnetischen Werkstoffen auf, er wird gemeinhin als Soft Iron Effekt bezeichnet. Abbildung 3.11 verbildlicht die Wechselwirkung zwischen einem äußeren Magnetfeld und ferromagnetischen Materialien.

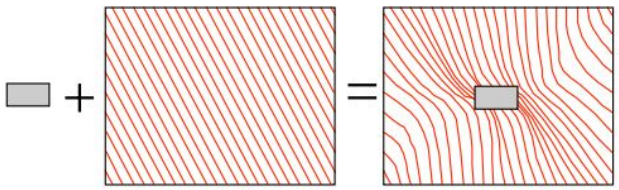


Abbildung 3.11: Soft Iron Effekt durch ferromagnetische Werkstoffe [28]

Führt man diese Effekte mit den allgemeinen Sensorfehlern in einem Messmodell zusammen, lässt es sich schematisch wie in Abbildung 3.12 darstellen.

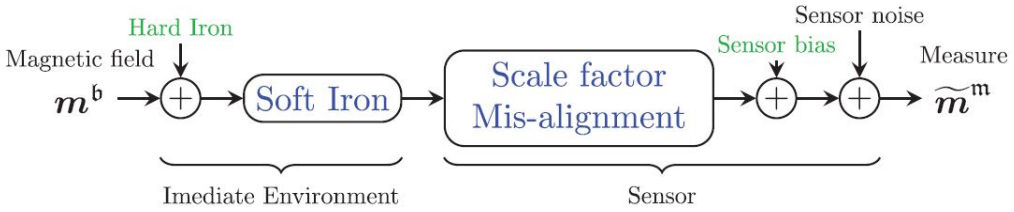


Abbildung 3.12: Schematische Darstellung des Messmodells [24]

Führt man dieses Modell wie in [29] detailliert aus, erhält man:

$$^{m}\tilde{\mathbf{m}} = \mathbf{A}_s\mathbf{A}_o(\mathbf{C}_{si}(^{b}\mathbf{m} + {}^{b}\mathbf{h})) + {}^{b}\mathbf{b} + \mathbf{n}(t)\,. \tag{3.8}$$

Dabei steht die Notation ^{b} und ^{m} für im Referenzkoordinatensystem der Sensoreinheit und des Sensors dargestellte Größen. Ohne Verlust der Allgemeingültigkeit lässt sich dieses Modell mit $\bar{\mathbf{A}} = \mathbf{A}_s\mathbf{A}_o\mathbf{C}_{si}$ und $\mathbf{b} = \mathbf{A}_s\mathbf{A}_o\mathbf{C}_{si}{}^{b}\mathbf{h} + {}^{b}\mathbf{b}$ wieder rückführen auf die Form

$$^{m}\tilde{\mathbf{m}} = \bar{\mathbf{A}}{}^{b}\mathbf{m} + \mathbf{b} + \mathbf{n}(t)\,. \tag{3.9}$$

Bei Messung des unveränderlichen, natürlichen Erdmagnetfeldes ist der Betrag $||^{b}\mathbf{m}||^2 = const.$ Um diese Eigenschaft für den Kalibiervorgang zu nutzen, wird (3.9) ohne $\mathbf{n}(t)$, da mittelwertsfrei, und ohne Kennzeichnung der Koordinatensysteme entsprechend umgeformt

$$\bar{\mathbf{A}}^{-1}(\tilde{\mathbf{m}} - \mathbf{b}) = \mathbf{m}\,, \tag{3.10}$$

$$(\tilde{\mathbf{m}} - \mathbf{b})^T(\bar{\mathbf{A}}^{-1})^T\bar{\mathbf{A}}^{-1}(\tilde{\mathbf{m}} - \mathbf{b}) = \mathbf{m}^T\mathbf{m} = ||\mathbf{m}||^2 = c = 1\,. \tag{3.11}$$

Dabei ist die Konstante c frei wählbar. Weil das Produkt einer Matrix mit seiner Transponierten immer symmetrisch ist, lässt sich die Gleichung mit der symmetrischen Matrix $\mathbf{A} = (\bar{\mathbf{A}}^{-1})^T\bar{\mathbf{A}}^{-1}$ umformen zu

$$\tilde{\mathbf{m}}^T\mathbf{A}\tilde{\mathbf{m}} - 2\mathbf{b}^T\mathbf{A}\tilde{\mathbf{m}} + \mathbf{b}^T\mathbf{A}\mathbf{b} = 1\,. \tag{3.12}$$

Diese Gleichung entspricht, wie vielfach in der Literatur ausgeführt, einer Mittelpunktsquadrik in den Komponenten von $\tilde{\mathbf{m}}$. Die Messgröße, die sich bei beliebiger Sensorlage mit idealem Sensor auf eine zentrische Einheitskugel abbilden würde, beschreibt bei fehlerbehaftetem Sensor somit einen Ellipsoid in allgemeiner Lage im Referenzkoordinatensystem der Sensoreinheit. Die Idee für die Kalibrierung über diese Methode besteht darin, einen Ellipsoid in eine Menge an Messwerten einzupassen und mit seinen Parametern, die jenen

in (3.12) entsprechen, auf den tatsächlichen Wert zurückzurechnen. Mit den Komponenten des Vektors $\tilde{\mathbf{m}}^T = \begin{pmatrix} x & y & z \end{pmatrix}$ und den Koeffizienten der Matrix

$$\mathbf{A} = \begin{pmatrix} A & D & F \\ D & B & E \\ F & E & C \end{pmatrix}$$

und des Vektors

$$-2\mathbf{b}^T\mathbf{A} = \begin{pmatrix} G & H & I \end{pmatrix} \tag{3.13}$$

sowie

$$J = \mathbf{b}^T\mathbf{A}\mathbf{b} - 1$$

lässt sich (3.12) ausformulieren zu

$$Ax^2 + By^2 + Cz^2 + 2Dxy + 2Eyz + 2Fxz + Gx + Hy + Iz + J = 0\,. \tag{3.14}$$

Da man in der Wahl für c und somit auch J frei ist, benötigt man mindestens neun Messwerte, um ein Gleichungssystem für die Unbekannten $A, B, C, D, E, F, G, H, I$ zu erstellen und aufzulösen. In der Praxis werden bei Anwendung dieser Methode eigentlich immer wesentlich mehr als neun Messpunkte in die Berechnung aufgenommen, um den Einfluss des Messrauschens auf das Ergebnis zu reduzieren. Dadurch ergibt sich ein überbestimmtes Gleichungssystem, das beispielsweise mit der Methode der kleinsten Quadrate gelöst werden kann. Mit $J = -1$ ergibt sich das Gleichungssystem für n Messwerte mit der Designmatrix D sowie der Koeffizientenmatrix p in folgender Form

$$\mathbf{D} \cdot \mathbf{p} = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}\,. \tag{3.15}$$

$$\begin{pmatrix} x_1^2 & y_1^2 & z_1^2 & x_1y_1 & y_1z_1 & x_1z_1 & x_1 & y_1 & z_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ x_n^2 & y_n^2 & z_n^2 & x_ny_n & y_nz_n & x_nz_n & x_n & y_n & z_n \end{pmatrix} \cdot \begin{pmatrix} A \\ B \\ C \\ D \\ E \\ F \\ G \\ H \\ I \end{pmatrix} = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}\,. \tag{3.16}$$

Im konkreten Fall kann dies recht einfach mit der Inversen der Produktsummenmatrix erfolgen

$$\mathbf{D}^T\mathbf{D}\mathbf{p} = \mathbf{D}^T \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} , \tag{3.17}$$

$$\mathbf{p} = (\mathbf{D}^T\mathbf{D})^{-1}\mathbf{D}^T \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} . \tag{3.18}$$

Allerdings ist diese Methode sensibel für schlechte Konditionierung, daher sollten die Messpunkte entsprechend gewählt werden. Nach Auflösung des Gleichungssystems kann mit den Koeffizienten G, H, I und (3.13) der Bias $\mathbf{b}$ des Messmodells zu

$$\mathbf{b} = -\frac{1}{2}\mathbf{A}^{-1} \begin{pmatrix} G \\ H \\ I \end{pmatrix} \tag{3.19}$$

errechnet werden. Um aus den restlichen Koeffizienten des Gleichungssystems die noch fehlenden Parameter für die Kalibrierung zu bestimmen, wird der Ellipsoid um den Bias verschoben, sodass sein Zentrum im Ursprung liegt. Für diese Translation erfolgt die Darstellung der Quadrik mit der erweiterten Darstellungsmatrix in homogenen Koordinaten

$$\begin{pmatrix} x & y & z & 1 \end{pmatrix} \begin{pmatrix} \mathbf{A} & -\mathbf{A}\mathbf{b} \\ -\mathbf{b}^T\mathbf{A} & -1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = 0 , \tag{3.20}$$

$$\begin{pmatrix} x & y & z & 1 \end{pmatrix} \begin{pmatrix} \mathbf{I} & 0 \\ \mathbf{b}^T & 1 \end{pmatrix} \begin{pmatrix} \mathbf{A} & -\mathbf{A}\mathbf{b} \\ -\mathbf{b}^T\mathbf{A} & -1 \end{pmatrix} \begin{pmatrix} \mathbf{I} & \mathbf{b} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = 0 , \tag{3.21}$$

$$\begin{pmatrix} x & y & z & 1 \end{pmatrix} \begin{pmatrix} \mathbf{A} & 0 \\ 0 & \mathbf{b}^T\mathbf{A}\mathbf{b} - 2\mathbf{b}^T\mathbf{b} - 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = 0 . \tag{3.22}$$

Mit der Division durch den negativen konstanten Term $\mathbf{b}^T\mathbf{A}\mathbf{b} - 2\mathbf{b}^T\mathbf{b} - 1$ ergibt sich die Normalform der Quadrik

$$\begin{pmatrix} x & y & z \end{pmatrix} \mathbf{A}_e \begin{pmatrix} x \\ y \\ z \end{pmatrix} = 1\,. \tag{3.23}$$

Für die Durchführung einer Hauptachsentransformation müssen die Eigenvektoren und Eigenwerte von $\mathbf{A}_e$ berechnet werden. Da für eine quadratische, symmetrische, positiv definite Matrix die Singulärwerte den Eigenwerten und die Eigenvektoren den Singulärvektoren entsprechen, kann aber statt der Lösung des Eigenwertproblems auch eine Singulärwertzerlegung erfolgen. Mit der Matrix der Eigenvektoren $\mathbf{Q}$ und der Matrix $\mathbf{D}$ mit den zugehörigen Eigenwerten in der Diagonale lässt sich $\mathbf{A}_e$ ausdrücken

$$\mathbf{A}_e = \mathbf{Q}\mathbf{D}\mathbf{Q}^T\,. \tag{3.24}$$

Zieht man die Wurzeln der Eigenwerte für die Diagonaleinträge der Matrix $\mathbf{S}$, lässt sich weiter schreiben

$$\mathbf{A}_e = \mathbf{Q}\mathbf{S}^T\mathbf{S}\mathbf{Q}^T\,. \tag{3.25}$$

Unter Ausnützung der Orthogonalität der Eigenvektormatrix folgt

$$\mathbf{A}_e = \mathbf{Q}\mathbf{S}^T\mathbf{Q}^T\mathbf{Q}\mathbf{S}\mathbf{Q}^T = (\mathbf{Q}\mathbf{S}\mathbf{Q}^T)^T\mathbf{Q}\mathbf{S}\mathbf{Q}^T\,. \tag{3.26}$$

Bei einem Vergleich mit (3.11) erkennt man, dass $\mathbf{Q}\mathbf{S}\mathbf{Q}^T$ der Matrix $\bar{\mathbf{A}}^{-1}$ entspricht. Geometrisch interpretiert ist die linksseitige Multiplikation von diesem $\bar{\mathbf{A}}^{-1}$ zu $(\tilde{\mathbf{m}} - \mathbf{b})$ die Rotation des ins Zentrum verschobenen Ellipsoids mit anschließender Stauchung, sodass er zur Kugel und abschließend zurück in seine ursprüngliche Lage gedreht wird.

Verzichtet man im Messmodell (3.8) auf die Modellierung der von $\mathbf{C}_{si}$ und $\mathbf{A}_o$, ist $\bar{\mathbf{A}}$ nur in der Diagonale besetzt und man passt mit der Methode des Ellispoidfitting einen nicht rotierten, exzentrischen Ellipsoid in die aufgenommenen Messpunkte ein. Die Anzahl der Parameter reduziert sich dadurch um drei (D, E, F) auf sechs, drei für den Bias und drei für die Achsenskalierung.

Zum Zwecke der Vereinfachung des Kalibriervorganges und um die Reproduzierbarkeit der Ergebnisse leichter überprüfen zu können, wurde im Zuge dieser Arbeit eine Vorrichtung konstruiert und gefertigt (siehe Abbildung 3.13), mit der eine automatische Messpunkteaufnahme erfolgen kann. Bei der Fertigung wurde darauf geachtet ferromagnetische Werkstoffe und stromdurchflossene Bauteile nur in möglichst großem Abstand zum Magnetometer einzusetzen. Die Vorrichtung besitzt zwei Schwenkachsen, mit denen die IMU bei der Messpunkteaufnahme einen vorgegebenen Bewegungsablauf durchläuft. Eine der Achsen führt eine azimutale Bewegung über einen Bereich von etwa 270° aus,

wobei die Vorrichtung in der Abbildung mit einer Neigung dieser Achse von 45° zur Vertikalen montiert ist. Ihre Lage wurde im Bild mit einer blauen Linie hervorgehoben.

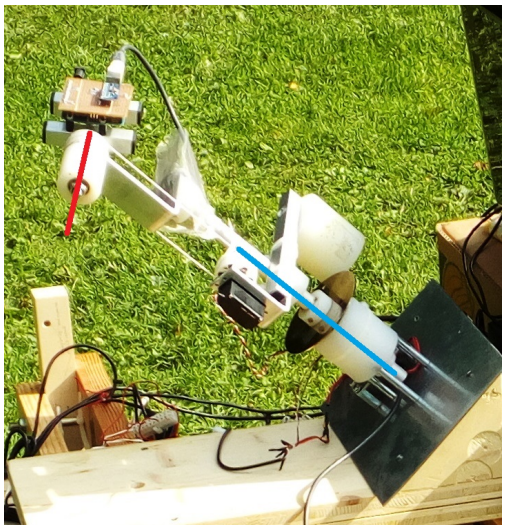
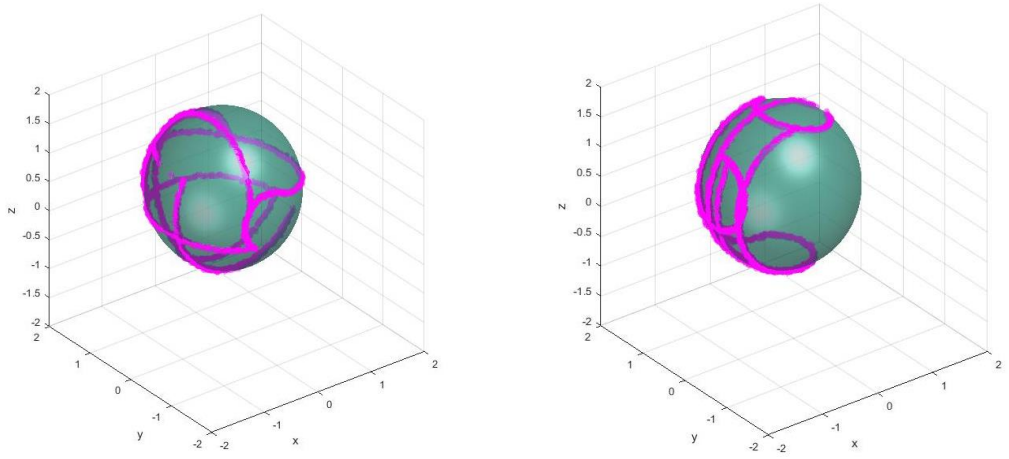


Abbildung 3.13: Vorrichtung zur automatischen Messpunkteaufnahme

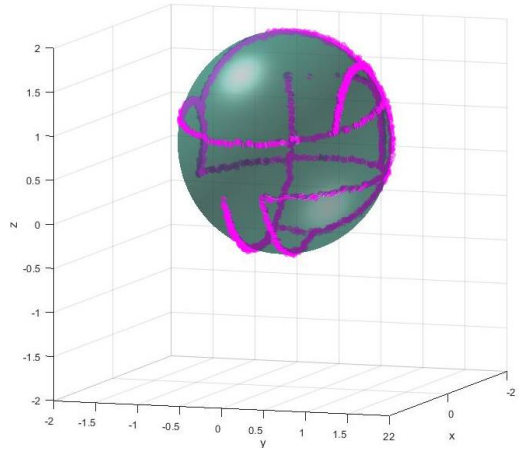
Die andere Achse (im Bild rot eingezeichnet) ermöglicht der IMU bei der Kalibrierung eine Änderung des Elevationswinkels um zirka ±90°. Die Lage der ersten Achse im Raum relativ zum Vektor der magnetischen Flussdichte beeinflusst dabei jene der aufgenommenen Messpunkte erheblich. Sind Vektor und Achse kollinear, bewirkt eine Drehung um die Achse keinerlei Lageänderung des Vektors im Koordinatensystem der Magnetometers. Bei Verwendung einer solchen Vorrichtung unter Ausnutzung natürlich vorhandener Feldgrößen war daher auf einen möglichst großen Winkel zwischen Schwenkachse und Messgröße zu achten. Die Auswirkungen dieses Effekts sind in Abbildung 3.14 ersichtlich. In Abbildung 3.15 wurde in dieselben Messpunkte jeweils ein Ellipsoid in allgemeiner Lage mit vollbesetzter Matrix $\mathbf{A}$ und einer mit $\mathbf{A}$ als Diagonalmatrix eingepasst. Derselbe Vorgang wurde für die Kalibrierung mit kleinem Winkel aus Abbildung 3.14b wiederholt. Die Ergebnisse sind in Tabelle 3.2 aufgelistet, die zweite Zeile der Tabelle entspricht Abbildung 3.14b, die dritte Abbildung 3.15b und die letzte Abbildung 3.15a. Die Zugehörigkeit der angegebenen Achsenskalierungen ist dabei aus der Matrix der Eigenvektoren abzulesen. Erwähnenswert ist die Diskrepanz zwischen optischem Passfehler bei Einpassung eines rotierten Ellipsoids (Abbildung 3.15b), die sich bei Vergleich mit einem nicht-rotierten Ellipsoid (Abbildung 3.15a) nicht in der Residuenquadratsumme widerspiegelt. Im Datenblatt der IMU [23], sind einige Daten zum Verhalten der einzelnen Sensoren bei veränderlicher Sensortemperatur aufgeführt, nicht so allerdings



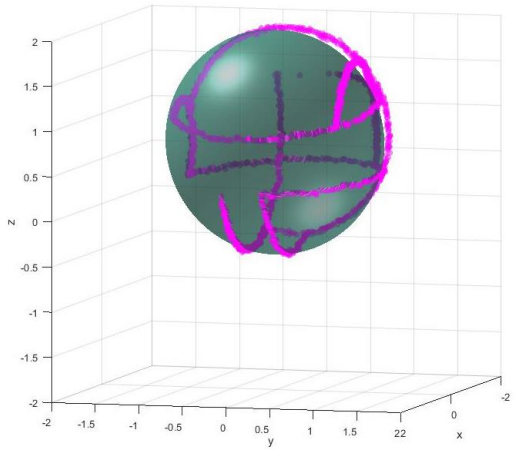
(a) Kalibrierung mit großem Winkel (b) Kalibrierung mit kleinem Winkel

Abbildung 3.14: Auswirkung des Winkels zwischen dem Vektor der magnetischen Flussdichte und der azimutalen Schwenkachse auf die Kalibrierung.

für das Magnetometer. Das integrierte Magnetometer mit der Bezeichnung AK8963 wird nicht vom Hersteller der restlichen Komponenten produziert, jedoch findet man auch direkt im Datenblatt des Herstellers [30] keine Informationen zum Temperaturverhalten. Nowicki et al. [31] untersuchten das Verhalten verschiedener kommerzieller Magnetometer mit Hinblick auf die Sensitivität und den Sensoroffset, dabei wurden Sensoren basierend auf verschiedenen Messprinzipien verwendet. Die Resultate für Magnetometer basierend auf dem Hall-Effekt, der auch dem AK8963 zugrunde liegt, zeigten eine deutliche Temperaturabhängigkeit beim Sensoroffset bei gleichzeitig nahezu gleichbleibender Empfindlichkeit. Während verschiedener Arbeiten mit der IMU, nach bereits erfolgter Kalibrierung, war teilweise noch ein Winkelfehler feststellbar. Da zu diesem Zeitpunkt die Ursache unbekannt war, wurde neben einer möglichen Temperaturabhängigkeit auch ein eventueller Unterschied der Empfindlichkeit für positive und negative Flussdichten in Betracht gezogen. Um eine solche feststellen zu können, wurde ein weiteres einfaches Gerät konstruiert und gefertigt (siehe Abbildung 3.16), das eine Messpunkteaufzeichnung ermöglicht, während die IMU Drehungen um eine fest im Raum stehende Achse vollführt. Bei diesen Drehungen werden über einen Drehgeber (mit einer Auflösung von 1°) die Winkel relativ zu einer beliebigen Startposition parallel zu den Werten des Magnetometers aufgezeichnet. Für die Ermittlung einer eventuellen Temperaturabhängigkeit wurde auf den internen Temperaturgeber der MPU9250 zurückgegriffen, eine Evaluierung, ob der Sensoroffset eine Tem-



(a) Einpassung eines nicht rotierten Ellipsoids



(b) Einpassung eines rotierten Ellipsoids

Abbildung 3.15: Einpassung eines Ellipsoids mit und ohne Modellierung des Soft Iron Effekts und der Nichtorthogonalität der Koordinatenachsen

peraturabhängigkeit aufweist, war somit ohne weitere Messtechnik machbar. Auf eine vorhergehende Kalibrierung des Temperaturgebers wurde mangels Relevanz, durch Zugrundelegung eines linearen Temperaturmessmodells, verzichtet, was jedoch bei einer Kompensationsrechnung eventueller Fehler die Verwendung desselben Temperaturgebers zwingend macht. Um die verschiedenen Temperaturen für die Messungen herzustellen, kam ebenfalls keine spezielle Ausrüstung zum Einsatz, der Raum, in dem die Messungen stattfanden, wurde mit herkömmlichen Heizmethoden aufgeheizt und durch Lüften abgekühlt.

Die Feststellung, ob eventuelle Asymmetrie in den Empfindlichkeiten vorliegt, basiert auf dem Gedanken, dass eine gedachte Linie durch die Messpunkte des Magnetometers, die während einer Drehung der IMU einander laut Encoder gegenüberliegen, durch den Rotationsmittelpunkt gehen muss. Abbildung 3.17 illustriert diese Methode. Aufgrund des starken Rauschverhaltens des Magnetometers ist eine einfache Schnittpunktebildung mit zwei Linien nicht ausreichend, vielmehr wurde der gesamte Bereich zwischen den diagonalen grau-blauen Linien, in dem zwei strichlierte Linien exemplarisch eingezeichnet wurden, zur Ermittlung der y-Koordinate des Rotationsmittelpunktes herangezogen. Selbiges gilt für den Bereich mit den gepunkteten Linien für die x-Koordinate. Für die Aufnahme dieser Messung wurde die IMU um eine Achse, die annähernd er z-Achse entspricht, gedreht. Mit einer solchen Drehung kann daher nur eine Aussage über die x und y-Koordinate des Rotationszentrum getroffen

Konfiguration	Bias	Achsen-skalierung	Eigenvektoren	Residuen-quadrat-summe
kleiner Winkel, rotierter Ellipsoid	$\begin{pmatrix}-0.295\\-0.061\\0.717\end{pmatrix}$	$\begin{pmatrix}1.282\\1.207\\1.200\end{pmatrix}$	$\begin{pmatrix}-0.191 & -0.970 & 0.154\\0.228 & -0.197 & -0.954\\0.955 & -0.147 & 0.258\end{pmatrix}$	28.34
kleiner Winkel, nicht rotierter Ellipsoid	$\begin{pmatrix}-0.305\\-0.058\\0.721\end{pmatrix}$	$\begin{pmatrix}1.278\\1.199\\1.187\end{pmatrix}$	$\begin{pmatrix}0 & 1 & 0\\0 & 0 & 1\\1 & 0 & 0\end{pmatrix}$	36.40
großer Winkel, rotierter Ellipsoid	$\begin{pmatrix}-0.286\\-0.055\\0.726\end{pmatrix}$	$\begin{pmatrix}1.262\\1.203\\1.179\end{pmatrix}$	$\begin{pmatrix}-0.140 & -0.078 & -0.987\\0.384 & 0.915 & -0.0127\\0.913 & -0.396 & -0.098\end{pmatrix}$	29.17
großer Winkel, nicht rotierter Ellipsoid	$\begin{pmatrix}-0.285\\-0.048\\0.731\end{pmatrix}$	$\begin{pmatrix}1.261\\1.209\\1.181\end{pmatrix}$	$\begin{pmatrix}0 & 0 & 1\\0 & 1 & 0\\1 & 0 & 0\end{pmatrix}$	36.74

Tabelle 3.2: Kalibrierergebnisse des Magnetometers für verschiedene Konfigurationen

werden. Für die z-Koordinate wäre eine Rotation um eine Achse geeignet, die einen möglichst großen Winkel mit der z-Achse einschließt. Ausgehend von der bereits festgelegten Voraussetzung ebener Navigation und einer entsprechend am Roboter montierten IMU, sind die Auswirkungen einer z-Exzentrizität auf die Navigation als gering einzuschätzen, deswegen und aus Zeitgründen wurde auf die Ermittlung eventueller z-Exzentrizitäten verzichtet. Grundsätzlich wirkt sich eine vorhandene Exzentrizität auf das Messergebnis ähnlich wie ein Bias aus. Das Ergebnis für die Messung ist in der Abbildung rot eingezeichnet. Weil die Rotationsachse jedoch nur annähernd der z-Achse entpricht, kann das so ermittelte Rotationszentrum nicht einfach mit dem Zentrum des Ellipsoids verglichen werden, um eine eventuelle Exzentrizität festzustellen. Im Weiteren wurde daher diese Methode auf mehrere Messungen zur Ermittlung der wahren Rotationsachse in Kombination mit der Methode des nicht rotierten Ellipsoidfittings bei unterschiedlichen Temperaturen angewandt. Abbildung 3.18 zeigt eine Messreihe mit eingepasstem Ellipsoid und der Rotationsachse (gelb), ermit-

telt mit den kleinsten Fehlerquadraten zu den einzelnen Rotationszentren der „Messringe“. Die Exzentrizität entspricht annähernd der x und y Koordinate des kleinsten Abstands dieser Geraden zum Zentrum des Ellipsoids. In Abbildung 3.19 sind das Zentrum des Ellipsoids als blaues Kreuz und der zugehörige nächste Punkt auf der Geraden als rotes Kreuz in einer Nahaufnahme sichtbar gemacht.

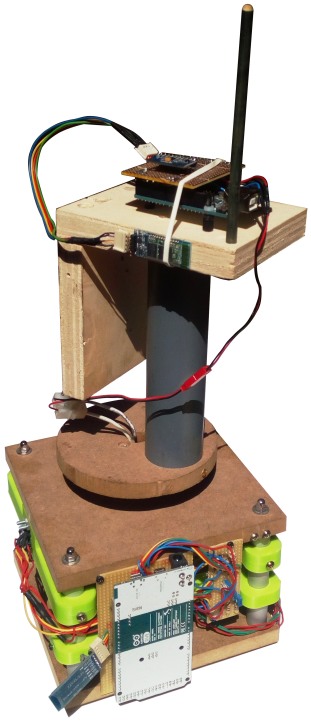


Abbildung 3.16: Vorrichtung zur automatischen Messpunkteaufnahme und Exzentrizitätsbestimmung

Die Lage der Messpunkte in Abbildung 3.18 eignet sich sehr gut, um bei der Ermittlung des Bias **b** in x und y-Richtung den Einfluss des Messrauschens gering zu halten, sodass man ein relativ genaues Ergebnis für diese Parameter erhält. Weil aber im oberen Bereich keine Messpunkte vorhanden sind, ist bei der Berechnung der z-Parameter mit erhöhtem Rauscheinfluss zu rechnen. In den Ergebnissen der temperaturgeführten Messreihen spiegelt sich dieses Verhalten auch wider (siehe Abbildung 3.20 bis Abbildung 3.22), zumal die Messpunkteaufnahme für alle Messungen auf vergleichbare Weise erfolgte.
Wenngleich keine Kalibrierung des Temperatursensors stattfand, soll zur Orientierung der ungefähre Temperaturbereich aufgeführt werden. Der Skalenwert von -3000 entspricht einer nicht näher bekannten Temperatur unter dem Ge-

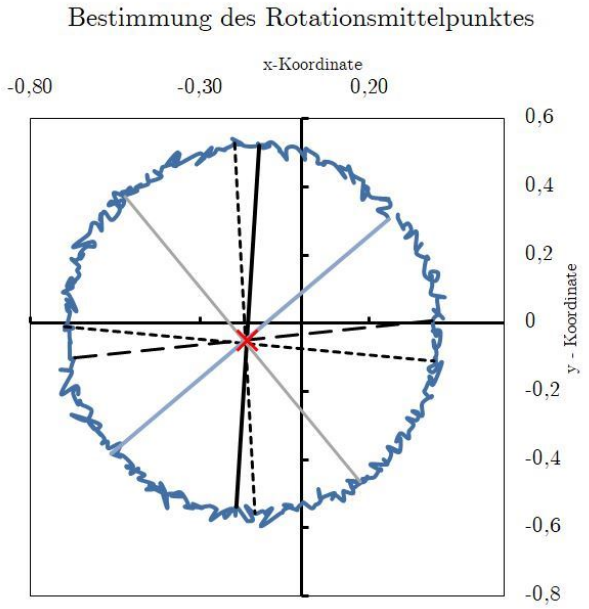


Abbildung 3.17: Bestimmung des Rotationsmittelpunktes

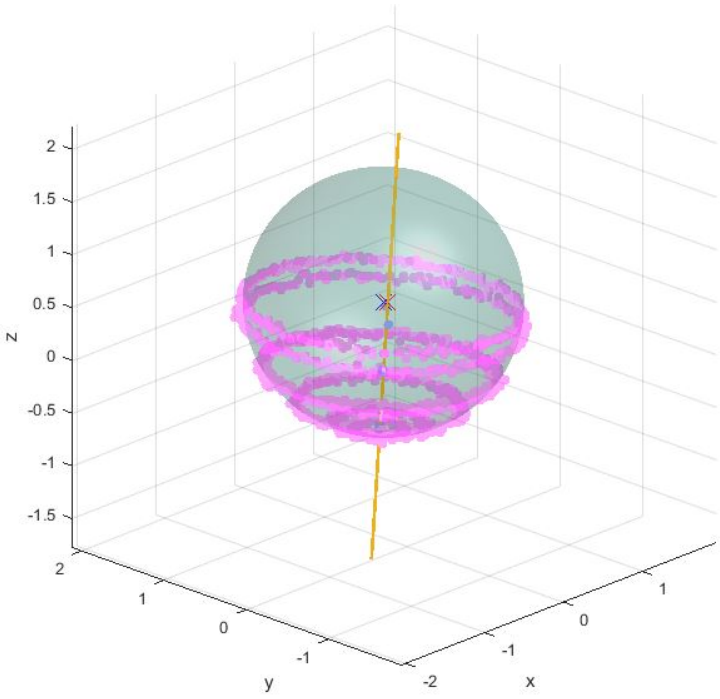


Abbildung 3.18: Eingepasster, nicht rotierter Ellipsoid mit Rotationsachse

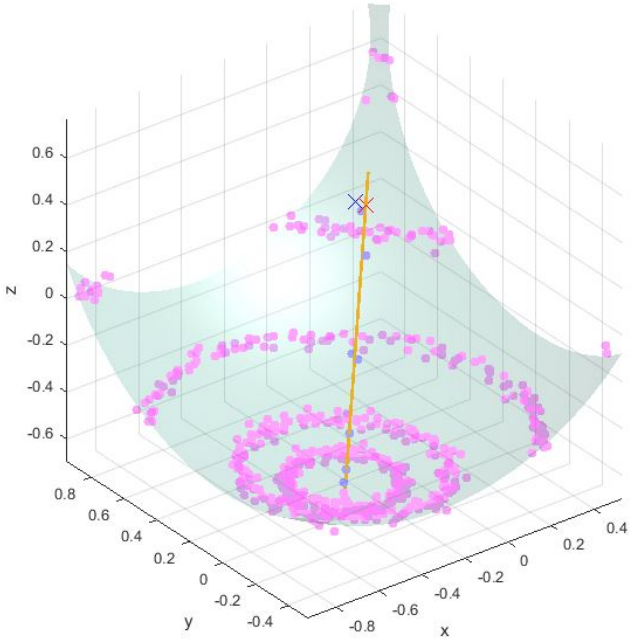
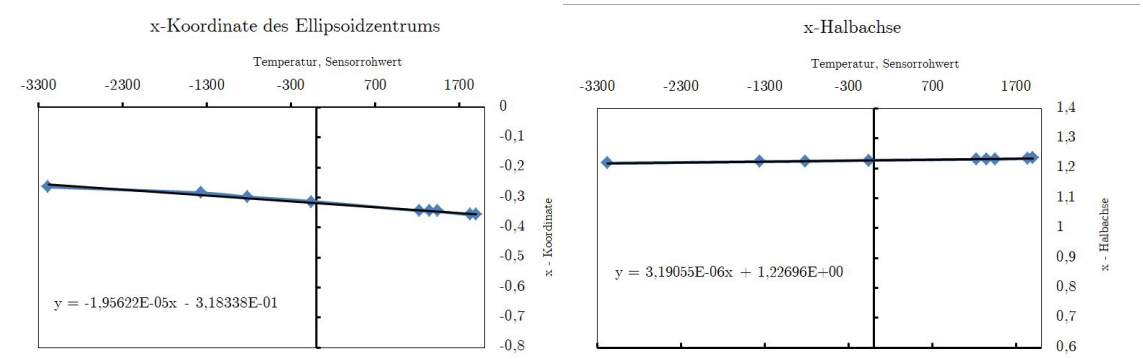


Abbildung 3.19: Nahaufnahme der Rotationsachse und des Ellipsoidzentrums

frierpunkt, das obere Ende der Skala von 2000 wurde vom Temperaturgeber bei erhöhter Raumtemperatur ausgegeben. Bei allen Diagrammen wurden lineare Regressionen durchgeführt, deren Ergebnisse in der jeweiligen Abbildung aufgeführt sind. Die Resultate der Exzentrizitätsermittlung, wie in Abbildung 3.18 und Abbildung 3.19 illustriert, sind in Abbildung 3.23 zu sehen. Wie man aus den Geradengleichungen der Regressionen ablesen kann, sind die Temperaturgradienten der Biaskoordinaten in ihrer Größenordnung um etwa einen Faktor 10 höher als jene der Empfindlichkeiten. Darüber hinaus wirkt sich eine Änderung der Achsenempfindlichkeiten auf die Orientierung des gemessenen Vektors nur dann aus, wenn auch eine signifikante Änderungen relativ zueinander vorhanden ist, sodass sich die Elliptizität über die Temperatur ändert. Beim Vergleich von Abbildung 3.20b und Abbildung 3.21b lässt sich aus dem ähnlichen Temperaturgradienten ableiten, dass die Änderung relativ zueinander vernachlässigbar ist. Durch solche nicht kompensierte Temperaturabhängigkeiten wird also hauptsächlich der Betrag des gemessenen Vektors verfälscht.



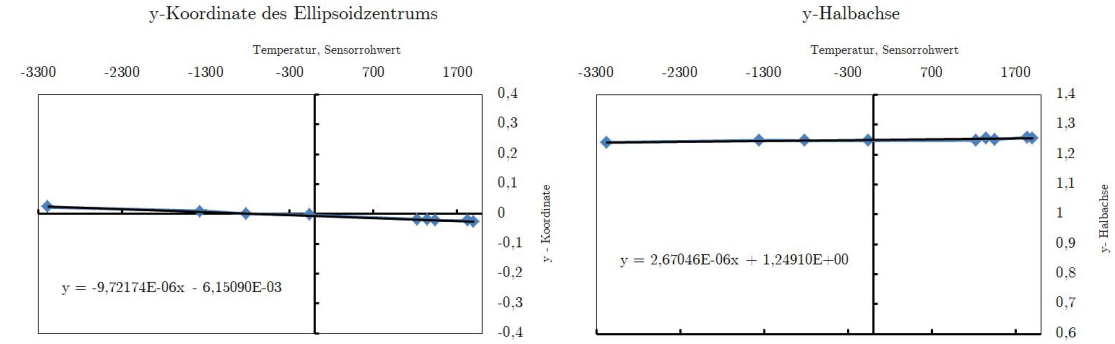
(a) Temperaturabhängigkeit des Magnetometerbias in x-Richtung

(b) Temperaturabhängigkeit der Magnetometerempfindlichkeit in x-Richtung

Abbildung 3.20: Temperaturabhängigkeiten des Kalibrierparameter in x-Richtung

Zusätzliche Anmerkungen zur Kalibrierung des Magnetometers

Um exakte Messergebnisse zu erhalten, ist wie bereits erwähnt eine Kalibrierung erforderlich, die in der späteren Einsatzumgebung des Magnetometers durchgeführt wird. Nur so lassen sich eventuelle Umwelteinflüsse vom Roboter selbst auf das Magnetometer ausgleichen. Dabei müsste die Messpunkteaufnahme eigentlich im fixen Verband mit dem ganzen Roboter erfolgen, was in der Realität bei vielen Anwendungen nur sehr begrenzt möglich ist [28]. Bei der Durchführung der verschiedenen Messreihen wurde beispielsweise festgestellt, dass schon die Verwendung verschiedener USB-Stecker für den Mikrocontroller (USB Buchse links oben in Abbildung 3.10c), sofern diese aus ferromagnetischem Werkstoff hergestellt sind (meist oberflächenbehandeltes Stahlblech), eine Neukalibrierung erforderlich macht. Dieser Umstand erklärt auch die verschiedenen Kalibierergebnisse zwischen Tabelle 3.2 und den restlichen Messreihen. Insgesamt waren diesbezüglich trotz des relativ hohen Aufwandes für die Kalibrierung bis zuletzt Kompromisse notwendig. In der Literatur findet man darüber hinaus auch Arbeiten, die sich mit dynamischen magnetischen Störungen durch elektronische Bauteile auseinandersetzen [24], für diese Arbeit wurden derartige Effekte jedoch nicht näher betrachtet, um in einem angemessenen Rahmen zu bleiben. Zum Thema unterschiedlicher Empfindlichkeiten und damit einhergehender Außermittigkeit des Ellipsoids konnte im Zuge einer Literaturrecherche nichts Ähnliches gefunden werden. Bei Vorhandensein solcher Exzentrizitäten handelt es sich bei der Fläche auf denen die Messpunkte liegen nicht mehr um einen Ellipsoid, sondern um einen aus acht Ellipsoidsegmenten zusammengesetzten Ovoid. Aufgrund der fehlenden weiterführenden Literatur wurde auf eine nähere



(a) Temperaturabhängigkeit des Magnetometerbias in y-Richtung

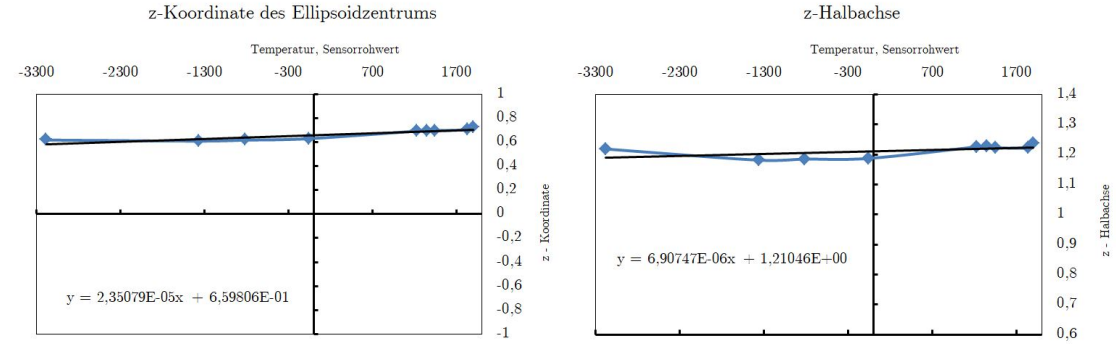
(b) Temperaturabhängigkeit der Magnetometerempfindlichkeit in y-Richtung

Abbildung 3.21: Temperaturabhängigkeiten des Kalibrierparameter in y-Richtung

Untersuchung der Thematik verzichtet. Bei der Methode des Ellipsoid fittings wird es dem Anwender möglich die Ellipsoidfläche durch die Kalibrierung auf eine Einheitskugel zurückzuführen, oder anders ausgedrückt, hat der gemessene Vektor nach der Kalibrierung im Idealfall unabhängig von der Orientierung der IMU einen Betrag von eins, sofern nur das Erdmagnetfeld gemessen wird. Wenn es die weitere Verarbeitung der Messwerte erfordert, ist danach noch eine Skalierung notwendig, sodass die Ausgabe des Magnetometers der tatsächlichen physikalischen Größe entspricht. Im Konkreten wäre dies eine Multiplikation jedes Achsenwertes mit der Totalintensität von zirka 48712nT, die zum Zeitpunkt der Verfassung dieser Arbeit in Wien herrschte [32]. Für Navigationszwecke ist es nicht notwendig, dass der Betrag des Messwertes der tatsächlichen Totalintensität entspricht, da lediglich die Richtungsinformation des Vektors in die weiteren Berechnungen einfließt (siehe Kapitel 3.3.3).

Kalibrierung des Accelerometers

Die Kalibrierung des Beschleunigungssensors erfolgt analog zu jener des Magnetometers mit demselben Messmodell, beschrieben durch Gleichung (3.6). Die durch Matrix **A** modellierten Fehler beschränken sich dabei allerdings auf die Nichtorthogonalität der Koordinatenachsen und die Achsenskalierung. Während der Kalibriersequenz ist daher sicherzustellen, dass keine eventuellen, sich mit der Schwerebeschleunigung überlagernde, Beschleunigungen auftreten. Bei Verwendung der automatischen Vorrichtung in Abbildung 3.13 war es daher notwendig, dass die Messpunkteaufzeichnung nur im Stillstand erfolgt. Zu diesem Zweck wird der schon für die Magnetometerkalibrierung programmierte



(a) Temperaturabhängigkeit des Magnetometerbias in z-Richtung

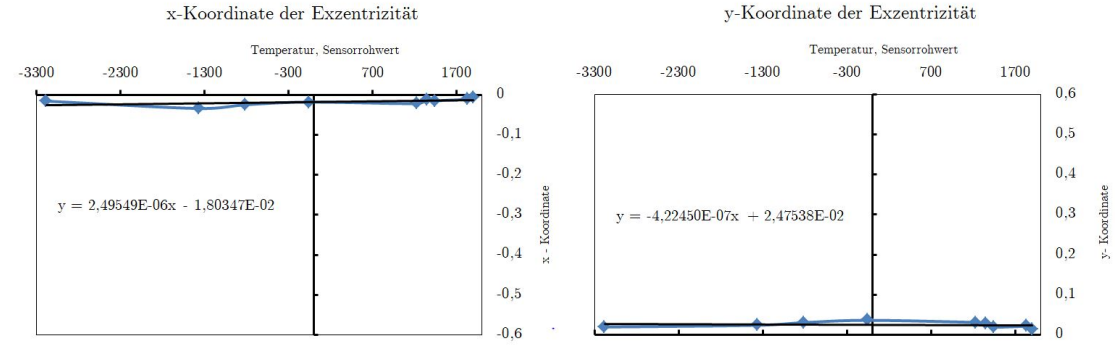
(b) Temperaturabhängigkeit der Magnetometerempfindlichkeit in z-Richtung

Abbildung 3.22: Temperaturabhängigkeiten des Kalibrierparameter in z-Richtung

Bewegungsablauf schrittweise mit zusätzlichen Bewegungspausen durchlaufen, in denen die Messpunkte aufgenommen werden. Die Schrägstellung der blauen Achse in Abbildung 3.13 verhindert eine Koaxialität von Drehachse und Messgröße. Das Ergebnis einer Messreihe mitsamt eingepasstem Ellipsoid zeigt Abbildung 3.24, die zugehörigen Parameter sind in Tabelle 3.3 gelistet. Ein eventueller Kalibrierfehler des Bias wirkt sich beim Beschleunigungssensor anders auf das Endergebnis aus als beim Magnetometer. Bei der in Kapitel 3.3.3 gewählten Methode die Beschleunigungsdaten für navigatorische Zwecke in der Ebene zu nutzen, sind die systematischen Auswirkungen eines Kalibrierfehlers als sehr gering zu bewerten. Auf Untersuchungen zum Temperaturverhalten wurde daher gänzlich verzichtet.

Bias	Achsen-skalierung	Eigenvektoren	Residuen-quadrat-summe
$\begin{pmatrix} -104.5 \\ -113.9 \\ 714.1 \end{pmatrix}$	$\begin{pmatrix} 8295 \\ 8208 \\ 8.192 \end{pmatrix}$	$\begin{pmatrix} -0.0540 & -0.9380 & 0.3421 \\ -0.1764 & -0.3282 & -0.9279 \\ 0.9828 & -0.110 & -0.1477 \end{pmatrix}$	1.353

Tabelle 3.3: Kalibrierergebnisse des Accelerometers



(a) Temperaturabhängigkeit der Exzentrizität in x-Richtung

(b) Temperaturabhängigkeit der Exzentrizität in y-Richtung

Abbildung 3.23: Temperaturabhängigkeiten der ermittelten Exzentrizitäten

Kalibrierung des Gyroskops

Beim gewählten Algorithmus (siehe Kapitel 3.3.3) ist eine Ermittlung des Gyroskopbias nicht unbedingt notwendig. Um ein besseres Ergebnis zu erhalten, ist die Berechnung des Bias im laufenden Messbetrieb ein probates Mittel. Ebenso ist der Filter für eventuelle Kalibrierfehler der Empfindlichkeiten in die verschiedenen Achsenrichtungen des Gyroskops wenig sensibel, sofern keine hochdynamischen Rotationen ausgeführt werden, was bei der Navigation eines Feldroboters auch nicht zu erwarten ist. Die einmalige Kalibrierung der Empfindlichkeiten kann, sofern keine Anforderungen über die genannten hinaus bestehen, über einfache Aufintegration der Sensorrohwerte während einer Rotation um eine der Sensorachsen um einen bekannten Winkel erfolgen. Als Referenzwinkel bieten sich Rotationen um 90° oder 180° an, da diese sehr leicht mit einfachsten Hilfsmitteln überprüfbar sind. Schwieriger ist es sicherzustellen, dass die Rotationsachse mit der Koordinatenachse des Gyroskops zusammenfällt, anderenfalls müssten Ausgleichsrechnungen unter Einbindung der anderen zwei Achsen durchgeführt werden. Bei bereits kalibriertem Accelerometer kann die Schwerebeschleunigung verwendet werden, um die Achsenlage festzustellen, unter der Annahme, dass die Achsen von Gyroskop und Accelerometer koaxial sind. Das Datenblatt der IMU [23] gibt eine ungefähre Auskunft über die Empfindlichkeit in (LSB)/°/s. Für den eingestellten Messbereich von ±1000°/s ist dafür ein Wert von 32.5 (LSB)/°/s angegeben, was umgerechnet ins Bogenmaß einer Empfindlichkeit von rund 1862 (LSB)/rad/s entspricht. Verschiedenen Kalibrierungen mit der eben genannten Methode zeigten, dass die angegebenen Werte den tatsächlichen sehr nahe kommen. Im Datenblatt der IMU sind neben Angaben zur Empfindlichkeit auch welche zum Temperturverhalten aufgeführt,

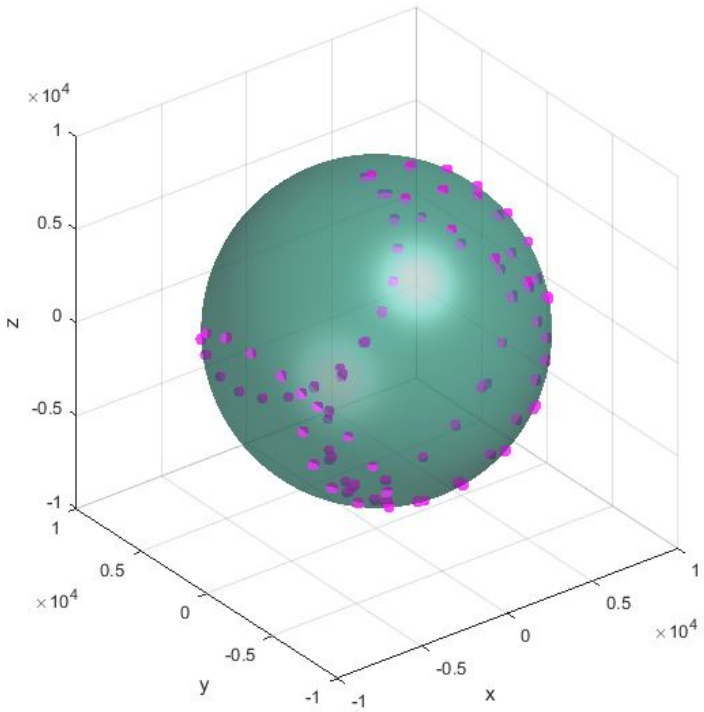


Abbildung 3.24: Messpunkte der Kalibiersequenz des Beschleunigungssensors mit eingepasstem Ellipsoid

was zu dem Schluss führt, dass für eine exakte Kalibierung auch eine diesbezügliche Evaluierung notwendig wäre. Die Summe dieser Umstände und das Wissen über geringe Auswirkungen von Empfindlichkeitsabweichungen veranlasste zur Übernahme der im Datenblatt angegebenen Sensitivität für alle drei Achsen.

3.3.3 Entwickeltes Kalman - Filter zum Fusionieren der IMU Messgrößen

In inertialen Navigationssystemen, angeschnitten in Kapitel 2.4.5, werden Inertialsensoreinheiten verwendet, um über Beschleunigungen und Geschwindigkeiten Strecken zu errechnen, mit denen in letzter Konsequenz schließlich die Position ermittelt wird. Der Problematik von sich akkumulierenden Fehlern wird bei kostengünstigen IMUs durch die niedrige Qualität zusätzliches Gewicht zuteil, sodass diese Methode in diesem Fall für die Navigation unbrauchbar wird. Nach Tabelle 3.1 eignen sich IMUs mit Magnetometer jedoch sehr gut, um die Lage im Raum zu bestimmen, darum wurde der Sensoreinheit für die gegenständliche Diplomarbeit dieser Verwendungszweck zugedacht.

Bei Einschränkung des für die Sensoreinheit zuständigen Datenverarbeitungsalgorithmus auf die Aufgabe der Lageberechnung wird in der Literatur häufig der bereits erwähnte Begriff des Attitude and Heading Reference Systems (AHRS) verwendet. Aufgrund einer Vielzahl an Anwendungsmöglichkeiten in verschiedensten Disziplinen und den dabei verwendeten unterschiedlichen Termini für diese Thematik würde eine vollkommen lückenlose Recherche den Umfang dieser Arbeit übersteigen. Grob ausgedrückt ist der Einsatz solcher kostengünstiger und kleiner Sensoren überall dort interessant, wo mobile Bewegung stattfindet, was neben den Bewegungen jeder Art von autonomen Fahrzeugen natürlich auch für jene von Menschen gilt [33]–[35].

Grundsätzliches zur Lagedarstellung

Die Beschreibung der Lage eines Körpers erfolgt zwangsweise immer relativ zu einer Referenz, die in den allermeisten Fällen mit einem geraden, orthogonalen Rechtssystem definiert wird. Es ist außerdem üblich, auch den beschriebenen Körper mit einem Koordinatensystem gleich dem Referenzsystem zu versehen. Das Koordinatensystem des Körpers wird häufig als *bewegt* oder nur als *Körpersystem* bezeichnet, wogegen das Referenzsystem meist *fix* oder *Weltkoordinatensystem* genannt wird. Bei Verwendung zweier solcher räumlicher Koordinatensysteme verfügt ein Körper grundsätzlich über sechs Freiheitsgrade, drei davon für die Beschreibung der Position des Körpersystemursprungs über Koordinaten in die Achsenrichtungen des fixen Referenzsystems. Für die Navigation relevante Aspekte bei der Beschreibung der Position und den

dabei verwendeten Referenzsystemen werden an späterer Stelle ausgeführt. Die verbleibenden drei Lageparameter sind dann für die Orientierung im Raum zuständig, also der Rotation des Körpersystems gegenüber dem Referenzsystem und damit sind sie auch jene, die es hier näher zu beleuchten gilt. Es existieren mehrere Möglichkeiten die Rotation eines Koordinatensystems gegenüber einem anderen darzustellen, zwei davon werden nachfolgend auch verwendet, Rotationsmatrizen und Quaternionen. Die Notationen sind in den verschiedenen Arbeiten zu der Thematik nicht einheitlich, werden aber meist an entsprechender Stelle einführend erläutert. Darstellungsmethoden, die hier nicht verwendet werden, aber auch weite Verbreitung gefunden haben, sind Rotationen um bewegte Achsen mit Eulerwinkeln, Rotationen um fixe Koordinatenachsen und Rotationen um einen Winkel mit angegebener Rotationsachse.

Rotationsmatrix

Drückt man die Einheitsvektoren der Koordinatenachsen des Körpersystems im fixen Referenzsystem aus ${}^f\mathbf{x}_b, {}^f\mathbf{y}_b, {}^f\mathbf{z}_b$ und schreibt sie als Spalten in eine Matrix, erhält man die Rotationsmatrix ${}^f\mathbf{R}_b = ({}^f\mathbf{x}_b, {}^f\mathbf{y}_b, {}^f\mathbf{z}_b)$. Man spricht bei einer so angegebenen Rotationsmatrix von der Lage des b-Koordinatensystems relativ zum f-Koordinatensystem. Weil die beschriebenen Vektoren Einheitslänge besitzen und orthogonal zueinander stehen, ist auch die Rotationsmatrix orthogonal und hat eine Determinante von eins. Diese Konvention wird auch in ROS vom tf-Packages für die Transformationen zwischen den einzelnen Koordinatensystemen eines Roboters verwendet. Grundsätzlich ist zu beachten, dass Rotationen vorzeichenbehaftet sind, die positive Drehrichtung um einen bestimmten Winkel ergibt sich aus der Achsenrichtung des Referenzsystems und der Rechten-Hand-Regel. Mit der so zusammengestellten Rotationsmatrix ist es möglich Vektoren, beschrieben im Körpersystem, ${}^b\mathbf{v}$, mit

$$ {}^f\mathbf{v} = {}^f\mathbf{R}_b \cdot {}^b\mathbf{v} \tag{3.27} $$

in das Referenzsystem umzurechnen. Um einen Vektor aus dem fixen Koordinatensystem ins „neue“, gedrehte System umzurechnen, ist eine linksseitige Multiplikation dieser Gleichung mit der Inversen der Rotationsmatrix (3.28) notwendig. Wegen ihrer Orthogonalität ist ihre Inverse mit der Transponierten identisch.

$$ {}^b\mathbf{v} = {}^f\mathbf{R}_b^T \cdot {}^f\mathbf{v} \tag{3.28} $$

Aus Sicht des gedrehten Körperkoordinatensystems erscheint ein Vektor des fixen Systems in die Gegenrichtung gedreht, man spricht bei solchen Vektortransformationen, bei denen das Koordinatensystem gedreht wird, von passiven

Drehungen. Rotationsmatrizen lassen sich sehr einfach über Matrixmultiplikationen kombinieren. Für den Fall drei gegebener Koordinatensysteme k,b,f der Rotationsmatrix ${}^{f}\mathbf{R}_{b}$, welche die Lage von b relativ zu f beschreibt und der Rotationsmatrix ${}^{b}\mathbf{R}_{k}$ für die Lage von k relativ zu b kann man die Lage von k relativ zu f mit der Rotationsmatrix

$$ {}^{f}\mathbf{R}_{k} = {}^{f}\mathbf{R}_{b}\,{}^{b}\mathbf{R}_{k} \tag{3.29} $$

beschreiben.

Quaternionen

Quaternionen sind vier-Tupel in einem eigenen Zahlenbereich, benannt nach Hamilton, der sie in [36] erstmals beschrieb, wenngleich sie unabhängig davon bereits wenige Jahre zuvor entdeckt wurden [37]. Die Mathematik der Quaternionen soll hier nicht erschöpfend ausgeführt werden, sondern lediglich die wichtigsten Zusammenhänge bei der Nutzung zur Darstellung von Rotationen. Diebel gibt in [38] einen guten Überblick über die Methoden zur Lagedarstellung, jedoch legt er seinen Ausführungen genau die inverse Form von (3.28) zugrunde. Alle weiteren Gleichungen zur Thematik wurden daher angepasst, um die Konsistenz der Arbeit zu bewahren.
Das Quadrupel einer Quaternion (3.30) besteht immer aus einem Skalar (oft q_0) und einem dreiteiligen Imaginäranteil ($\mathbf{q}_{1:3}$). Zur Rotationsdarstellung werden Einheitsquaternionen mit einem Betrag von eins verwendet. Die Betragsbildung erfolgt analog zur euklidischen Norm.

$$ \mathbf{q} = \begin{bmatrix} q_0 \\ \mathbf{q}_{1:3} \end{bmatrix} \tag{3.30} $$

Die adjungierte Quaternion ist gegeben zu

$$ \bar{\mathbf{q}} = \begin{bmatrix} q_0 \\ -\mathbf{q}_{1:3} \end{bmatrix} \tag{3.31} $$

und ist bei Einheitsquaternionen identisch mit der Inversen. Sie gibt wie die transponierte Rotationsmatrix die Drehung in die Gegenrichtung wieder. Wird nicht nur der Imaginäranteil negiert sondern die gesamte Quaternion, ändert sich die mit der Quaternion dargestellte Lage dagegen nicht. Addition und Subtraktion erfolgen gleich wie in gewöhnlichen Vektorräumen, die Multiplikation zweier Quaternionen hingegen ist per Konvention mit dem Hamilton-Produkt dargestellt als Matrix-Vektor Multiplikation definiert zu

$$\mathbf{pq} = \begin{bmatrix} p_0 & -p_1 & -p_2 & -p_3 \\ p_1 & p_0 & -p_3 & p_2 \\ p_2 & p_3 & p_0 & -p_1 \\ p_3 & -p_2 & p_1 & p_0 \end{bmatrix} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} . \tag{3.32}$$

Es wird manchmal (nicht hier) mit dem Zeichen des tensoriellen Produkts $\otimes$ gekennzeichnet, um eine bessere Unterscheidung zum Skalarprodukt zu erreichen, bei dem die Multiplikation wie beim Skalarprodukt zweier Vektoren des $\mathbb{R}^4$ erfolgt. Bei mehreren Drehungen hintereinander kann die gesamte Drehung wie schon bei den Rotationsmatrizen über eine einfache Multiplikation

$$^f\mathbf{q}_k = {}^f\mathbf{q}_b{}^b\mathbf{q}_k \tag{3.33}$$

errechnet werden. Bei gegebener Quaternion ist es möglich die Rotationsmatrix wie nachfolgend zu errechnen.

$${}^f\mathbf{R}_b({}^f\mathbf{q}_b) = \begin{bmatrix} q_0^2+q_1^2-q_2^2-q_3^2 & 2(q_1q_2-q_0q_3) & 2(q_1q_3+q_0q_2) \\ 2(q_1q_2+q_0q_3) & q_0^2-q_1^2+q_2^2-q_3^2 & 2(q_2q_3-q_0q_1) \\ 2(q_1q_3-q_0q_2) & 2(q_2q_3+q_0q_1) & q_0^2-q_1^2-q_2^2+q_3^2 \end{bmatrix} \tag{3.34}$$

Die Umkehrung ist etwas komplizierter, da wegen der vorkommenden Wurzeln eine Fallunterscheidung durchgeführt werden muss, um keine ungewollten komplexen Ergebnisse zu erhalten. Für diese Unterscheidung wird jedoch auf [38] verwiesen und nur der Fall angeführt, bei dem die Spur der gegebenen Rotationsmatrix vergrößert um eins größer als null sein muss (= positive Diskriminante in (3.35)). Mit den Einträgen der Rotationsmatrix r_{ij} lässt sich die Quaternion ausdrücken zu

$${}^f\mathbf{q}_b({}^f\mathbf{R}_b) = \frac{1}{2} \begin{bmatrix} \sqrt{1+r_{11}+r_{22}+r_{33}} \\ (r_{32}-r_{23})/\sqrt{1+r_{11}+r_{22}+r_{33}} \\ (r_{13}-r_{31})/\sqrt{1+r_{11}+r_{22}+r_{33}} \\ (r_{21}-r_{12})/\sqrt{1+r_{11}+r_{22}+r_{33}} \end{bmatrix} . \tag{3.35}$$

Vorhandene Filteralgorithmen

Unter der großen Anzahl an verschiedenen Algorithmen zur Lageberechnung findet sich ein auffällig großer Anteil, denen eine Version des Kalman-Filters zugrunde liegt[33], [34], [39]–[42] weil das Kalman-Filter wegen seiner Eigenschaften sehr gut für Sensorfusionsalgorithmen geeignet ist und diesbezüglich durchaus als Standardverfahren bezeichnet werden kann [43]. Von den Algorithmen, die nicht das Kalman-Filter verwenden, sei der Entwurf von Madgwick et al. [44] erwähnt, der viel Beachtung gefunden hat.

Bemerkenswert bei einigen der Algorithmen, etwa auch bei Madgwick et al. [44] oder auch Valenti et al. [34], ist die Tatsache, dass für die Berechnungen davon ausgegangen wird, dass das Accelerometer nur die Schwerebeschleunigung misst. Wie auch Aida et al. in [41] feststellt, ist eine solche Annahme für viele Anwendungsfälle nicht zutreffend, sodass die Lageberechnung von äußeren Beschleunigungen verfälscht wird. Aida selbst schlägt daher in [41] einen Algorithmus vor, der dieser Problematik Rechnung trägt und gibt dabei auch einen kleinen Überblick über andere Veröffentlichungen zur Lageberechnung unter Berücksichtigung externer Beschleunigungen. Solche Filter sind oft adaptiver Natur um die Berechnungen in verschiedenen Situationen anzupassen.

Zustandekommen des eigenen Filteralgorithmus

In ROS ist eine Implementierung des Algorithmus von Madgwick verfügbar, es war daher naheliegend in einem ersten Versuch diese zu nutzen, um die Lage der IMU zu errechnen. Da sich aber auch nach längerer Fehlersuche nicht das zu erwartende Ergebnis einstellte, fiel der Entschluss zur Entwicklung eines eigenen Algorithmus. Dabei sollte die Simplizität an oberster Stelle stehen, um die Nachvollziehbarkeit des Filteraufbaues auf ein intuitives Niveau zu ziehen. Zwar sind Quaternionen zur Lagerepräsentation nicht sehr anschaulich, sie haben allerdings gegenüber anderen Methoden Vorteile, beipielsweise tritt bei ihnen keine kardanische Blockade auf (im Englischen: Gimbal Lock). Aus diesem Grund wurde der Filter basierend auf dieser Darstellungsmethode entworfen.

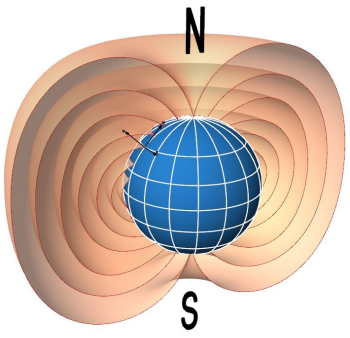
Filteraufbau

Madgwick gibt in seiner Arbeit als Nachteil für Kalman-basierte Filter an, dass diese häufig einen langen Zustandsgrößenvektor besitzen, jedoch wurden schon Filterlayouts entworfen, die genau diesen Umstand vermeiden. Valenti et al. verwenden für den Filter in [34] eine zweigeteilte Filterstruktur, die zu einem sehr einfachen, linearen Kalman-Filter führt. Im ersten Teil wird in einer separaten Berechnung eine Quaternion aus den Messgrößen errechnet, die dann im zweiten Teil dem eigentlichen Filter im Messprozess zugeführt wird. Die Berechnung der Quaternion aus den vektoriellen Beoachtungsgrößen entspricht dabei dem Lösen des bekannten Problem Wahba's [45] aus dem Jahr 1965, für das im Laufe der Jahre verschiedenste Lösungen veröffentlicht wurden. Der zweiteilige Filteraufbau führt in seiner einfachsten Form zu einem sehr übersichtlichen Filter mit einem Zustandsgrößenvektor, der nur die Komponenten der Quaternion enthält. Valenti schlägt für die Berechnung der Quaternion einen algebraischen Algorithmus vor, der im Vergleich mit einem gewählten Factored Quaternion Algorithm (FQA) bei gleicher Leistungsfähigkeit

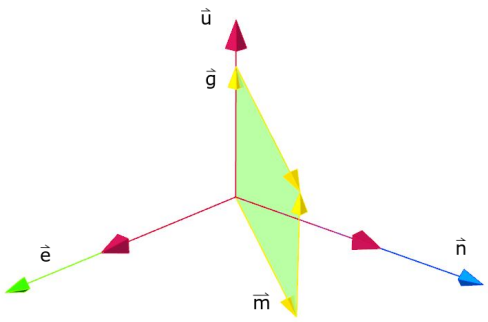
um 65% recheneffizienter ist. Nichtsdestotrotz ist der Algorithmus nicht sehr anschaulich, daher wurde versucht stattdessen eine intuitivere Lösung zu finden.

Algorithmus zur Bestimmung der Orientierung aus den Messgrößen

Ohne an dieser Stelle äußere Beschleunigungen in die Überlegungen miteinzubeziehen, ist es möglich die Lage einer IMU aus den beobachteten Größen Schwerebeschleunigung und magnetische Flussdichte über einfache geometrische Überlegungen herzuleiten. Abbildung 3.25a zeigt schematisch die räumliche Anordnung der Feldlinien der magnetischen Flussdichte des Erdmagnetfeldes. Der magnetische Fluss als gerichtete Größe zeigt dabei vom magnetischen Nordpol (=geografischer Südpol) zum magnetischen Südpol (=geografischer Nordpol). Aus Sicht eines sich auf der nördlichen Hemisphäre befindlichen East-North-Up-Referenzkoordinatensystems (ENU-Koordinatensystem) zeigt der Vektor der magnetischen Flussdichte $\mathbf{m}$ nach Nord-Unten. Nach [32] beträgt der Winkel zur Horizontalen (magnetische Inklination) in Wien zum Zeitpunkt der Verfassung dieser Arbeit 64.6°. Die Fallbeschleunigung $\mathbf{g}$ bildet sich bei Inertialsensoren als lotrecht nach oben zeigender Vektor ab. Die Lage der Vektoren $\mathbf{g}$ und $\mathbf{m}$ (beide gelb in Abbildung 3.25b) kann für lokal begrenzte Anwendungen näherungsweise als unveränderlich angenommen werden.



(a) Idealisierte Visualisierung der magnetischen Feldlinien des Erdmagnetfeldes



(b) Lage der vektoriellen Messgrößen im Raum

Abbildung 3.25: Lage der vektoriellen Messgrößen im Raum

Die Messgrößen $\mathbf{g}$ und $\mathbf{m}$ weisen eine unveränderliche Lage zueinander auf, das heißt, dass sie auch im bewegten Sensorkoordinatensystem ${}^{b}\mathbf{g}$, ${}^{b}\mathbf{m}$ immer dieselbe relative Lage zueinander besitzen. Bildet man mit ihnen das Kreuzprodukt

$$^b\mathbf{e} = {}^b\mathbf{m} \times {}^b\mathbf{g}\,, \tag{3.36}$$

resultiert daraus ein Vektor $^b\mathbf{e}$ orthogonal zu $^b\mathbf{g}$, $^b\mathbf{m}$ mit einer Länge, die in Abbildung 3.25b der grünen Fläche entspricht. Geht man von normierten Ausgangsvektoren aus, ergibt sich mit der Inklination in Wien

$$|{}^b\mathbf{m} \times {}^b\mathbf{g}| = 0.4321\,. \tag{3.37}$$

Um $^b\mathbf{e}$ zu normieren ist also eine Multiplikation mit einem Faktor von 2.3146 notwendig. Führt man diese Multiplikation von Anfang an nur mit $^b\mathbf{m}$ durch, besitzen alle weiteren Vektoren einen Betrag von eins. Bildet man erneut ein Kreuzprodukt (3.38) aus $^b\mathbf{e}$ und $^b\mathbf{g}$ ergibt sich der Vektor $^b\mathbf{n}$.

$$^b\mathbf{n} = {}^b\mathbf{g} \times {}^b\mathbf{e} \tag{3.38}$$

Weil $^b\mathbf{e}$ und $^b\mathbf{g}$ orthogonal zueinander stehen und $^b\mathbf{g}$ als normiert angenommen wird, besitzt $^b\mathbf{n}$ denselben Betrag wie $^b\mathbf{e}$. Hat $^b\mathbf{m}$ eine Länge von 2.3146 sind $^b\mathbf{n}$ und $^b\mathbf{e}$ sogleich normiert und man hat aus den Beobachtungsgrößen über einfache Vektorrechnung ein Rechtssystem abgeleitet, das genau dem ENU-Referenzkoordinatensystem entspricht. Anders ausgedrückt ist das Ergebnis (3.39) nichts anderes als eine Beschreibung der Koordinatenachsen des fixen Koordinatensystems im beweglichen System, also genau die inverse Formulierung der Definition der Rotationsmatrix.

$$^f\mathbf{R}_b^T = ({}^b\mathbf{e}_f, {}^b\mathbf{n}_f, {}^b\mathbf{g}_f) \tag{3.39}$$

Abbildung 3.26 zeigt die Lage der Vektoren schematisch auf der nördlichen Hemisphäre. Aus der Abbildung kann man auch erkennen, dass diese Methode nur funktioniert, solange Schwerebeschleunigung und der Vektor der magnetischen Flussdichte nicht kollinear sind, was nur an den Polen der Fall ist.
Die so gewonnene Rotationsmatrix kann natürlich in eine Quaternion (im Folgenden teilweise als Messquaternion bezeichnet) umgerechnet werden, dieser Schritt wird im nächsten Unterkapitel ausgeführt, in dem auch auf externe Beschleunigungen eingegangen wird.

Wegen der Simplizität des Algorithmus wurde er vom Autor selbst hergeleitet, es war jedoch zu erwarten, dass dieselbe Herangehensweise bereits früher publiziert worden war. Im Zuge der Recherche konnte festgestellt werden, dass Harold Black ihn bereits 1964 in [46] vorstellte und einige andere Wissenschaftler auch weiterführende Analysen zum Algorithmus ([47], [48]) veröffentlichten, wo er als TRIAD bezeichnet wird. Diese Analysen waren dem Autor dieser Arbeit zum

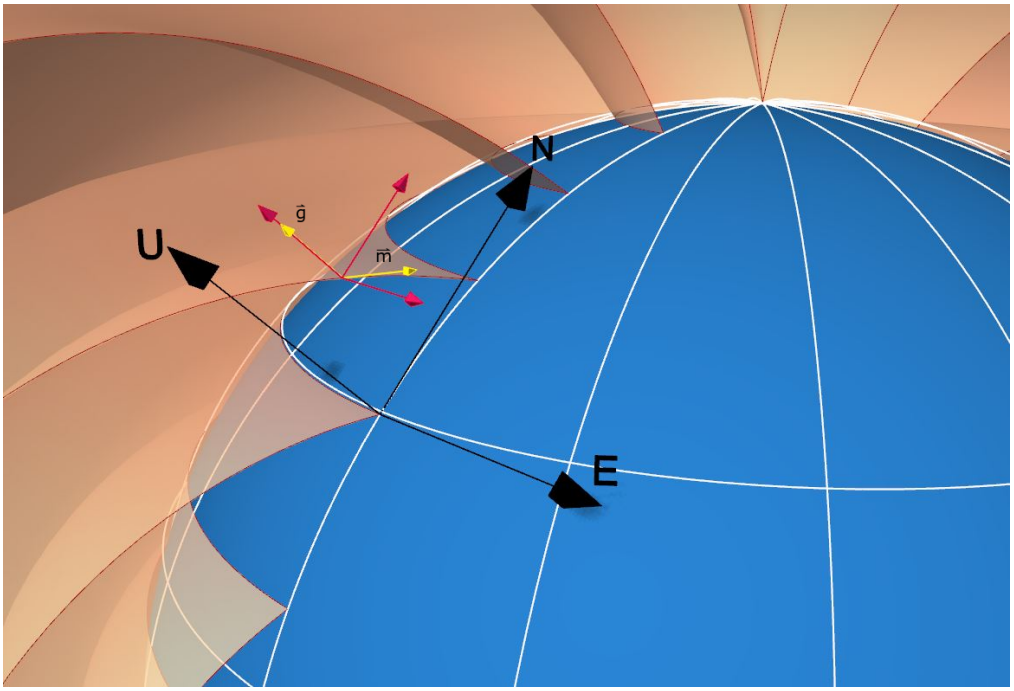


Abbildung 3.26: Lage der Vektoren nahe der Erdoberfläche
Schwarz: East-North-Up Koordinatensystem
Gelb: Vekorielle Messgrößen der IMU
Magenta: Aus den Messgrößen abgeleitetes Koordinatensystem

Zeitpunkt der Filterimplementierung nicht bekannt, Erkenntnisse daraus, speziell Untersuchungen zur Kovarianz, flossen daher nicht in den Filterentwurf ein.

Prozessmodell

Die zeitliche Ableitung einer Quaternion steht mit dem Vektor der Winkelgeschwindigkeiten $\boldsymbol{\omega}_{b,k}$ in erster Näherung in folgendem Zusammenhang:

$$ {}^{f}\dot{\mathbf{q}}_b = \frac{1}{2}\boldsymbol{\Theta}({}^{f}\mathbf{q}_b)\boldsymbol{\omega}_{b,k} = \frac{1}{2}\begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_0 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{bmatrix}\begin{bmatrix} 0 \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{3.40} $$

respektive

$$ {}^{f}\dot{\mathbf{q}}_b = \frac{1}{2}\boldsymbol{\Omega}(\boldsymbol{\omega}_{b,k}){}^{f}\mathbf{q}_b = \frac{1}{2}\begin{bmatrix} 0 & \omega_x & \omega_y & \omega_z \\ -\omega_x & 0 & \omega_z & -\omega_y \\ -\omega_y & -\omega_z & 0 & \omega_x \\ -\omega_z & \omega_y & \omega_x & 0 \end{bmatrix}\begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} . \tag{3.41} $$

Für die Herleitung wird auf [49] verwiesen. (3.40) und (3.41) können jeweils als zeitkontinuierliche Version von (2.24) aufgefasst werden. Nach [50] kann für das zeitvariante Prozessmodell (3.41) die Transitionsmatrix zwischen zwei diskreten Zeitpunkten k und k+1 für abschnittsweise konstantes $\boldsymbol{\omega}_{b,k}$ mit dem Matrixexponential folgendermaßen ausgedrückt werden

$$ {}^{f}\mathbf{q}_{b,k+1} = e^{\frac{1}{2}\boldsymbol{\Omega}(\boldsymbol{\omega}_{b,k})\Delta t} \cdot {}^{f}\mathbf{q}_{b,k} , \tag{3.42} $$

worin Δt die Zeitspanne von t_k bis t_{k+1} bezeichnet was der Lösung der Differentialgleichung (3.41) mit dem Zustand zum Zeitpunkt k als Anfangswert gleichkommt. Bricht man die Potenzreihe des Matrixexponentials in ausreichender Näherung nach dem zweiten Glied ab, erhält man für das zeitdiskrete Zustandsmodell für den Prädiktionsschritt schließlich

$$ {}^{f}\mathbf{q}_{b,k+1} = (\mathbf{I}_{4\times 4} + \frac{1}{2}\boldsymbol{\Omega}(\boldsymbol{\omega}_{b,k})\Delta t) \cdot {}^{f}\mathbf{q}_{b,k} + \mathbf{w}_k , \tag{3.43} $$

$$ {}^{f}\mathbf{q}_{b,k+1} = {}^{f}\mathbf{q}_{b,k} + \frac{1}{2}\boldsymbol{\Omega}(\boldsymbol{\omega}_{b,k})\Delta t \cdot {}^{f}\mathbf{q}_{b,k} + \mathbf{w}_k . \tag{3.44} $$

mit $\mathbf{w}_k$ als mittelwertfreies Prozessrauschen. Näheres zum Rauschverhalten, ausgedrückt durch die Kovarianzmatrizen, im nächsten Unterkapitel. Das Messmodell (vgl. (2.26)), in dem die Messquaternnion der Roationsmatrix (3.39) zur Anwendung kommt lautet

$$\mathbf{y}_{k+1} = {}^{f}\mathbf{q}_{b,k+1} + \mathbf{v}_{k+1} \,. \tag{3.45}$$

Abstimmung des IMU Kalman - Filters

Wie bereits erwähnt, ist die Annahme, es würden keine äußeren Beschleunigungen auftreten in vielen Fällen nicht mit der Realität vereinbar, was sich nach Implementierung eines Filters, basierend auf den bisherigen Ausführungen auch bestätigte. Aus diesem Grund wurde ein Filter entworfen, welches das Auftreten externer Beschleunigungen dezidiert berücksichtigt. Die Problematik dabei ist, dass äußere Beschleunigungen so auftreten können, dass das Ergebnis auch nach Überlagerung mit der Schwerebeschleunigung alle bekannten Eigenschaften der unbeeinflussten Schwerebeschleunigung erfüllt. Um dies zu präzisieren: Es gibt unendlich viele äußere Beschleunigungsvektoren, die mit der Schwerebeschleunigung gemeinsam einen Betrag von eins liefern und dabei mit dem Vektor der magnetischen Flussdichte den richtigen (lokal bekannten) Winkel einschließen. Abbildung 3.27 illustriert diese Mehrdeutigkeit, alle Beschleunigungsvektoren zwischen dem Ende von $\mathbf{g}_{tat}$ zum Rand der roten Ebene führen zu Gesamtbeschleunigungen, die dieselben vektoriellen Eigenschaften wie die Schwerebeschleunigung aufweisen. Im Folgenden werden externe Beschleunigungen als lineare Beschleunigungen bezeichnet, obgleich äußere Beschleunigungen nicht zwangsweise translatorischer Herkunft sein müssen. Die Ursache für äußere Beschleunigungen ist jedoch für die Lageberechnung irrelevant, darum erfolgt hier keine Unterscheidung. Unter der Annahme $\mathbf{a}_{lin} = \mathbf{0}$ würde die Berechnung der Lage für das gezeigte Beispiel mit $\mathbf{m}$ und $\mathbf{g}_2$ erfolgen, anstatt $\mathbf{a}_{lin}$ zuerst vom Messwert $\mathbf{g}_2$ zu subtrahieren und die Lageberechnung mit der tatsächlichen Schwerebeschleunigung durchzuführen.

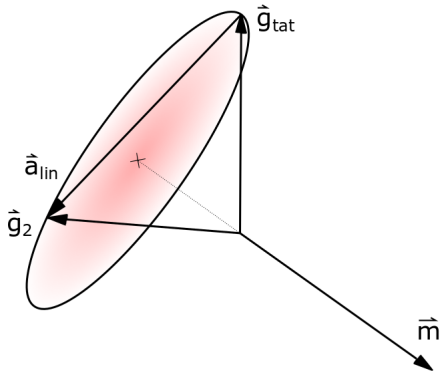


Abbildung 3.27: Mehrdeutigkeit der Berechnung von Lage und linearer Beschleunigung

Berechnet man die lineare Beschleunigung über die letzte bekannte Lage der IMU und zieht sie für die Berechnung der Schwerebeschleunigung vom Messwert ab, um so wieder die Lage der IMU zu errechnen, führt dies auf ein unterbestimmtes System, das durch das Messrauschen nach einiger Zeit beliebige, fiktive Werte für die lineare Beschleunigung ergibt und damit auch eine verfälschte Lage. Die Hypothese, $\mathbf{a}_{lin} = \mathbf{0}$, mit der man ein bestimmtes System erhält, wird daher dahingehend erweitert, dass die lineare Beschleunigung *zu bestimmten Zeitpunkten* null ist. Als Parameter zur Festlegung der Zeitpunkte, an dem die lineare Beschleunigung zu null angenommen wird, wurde die Größe p_{midabl} aus (3.47) gewählt, die sich aus den Komponenten der zeitlichen Änderung der Beschleunigungsmesswerte errechnet. Die Herleitung des oben beschriebenen Kalman-Filters für die Lage erfolgte primär über theoretische Überlegungen, der nachfolgende Teil ergab sich hingegen aus einer Mischung von Theorie und Empirie die während vieler Versuche mit der IMU gewonnen wurde.

$$\mathbf{a}_{\Delta} = \frac{\Delta \mathbf{a}_m}{\Delta t} = \frac{\mathbf{a}_{m,k} - \mathbf{a}_{m,k-1}}{\Delta t} \tag{3.46}$$

$$p_{midabl} = \frac{|a_{\Delta,x}|}{3} + \frac{|a_{\Delta,y}|}{3} + \frac{|a_{\Delta,z}|}{3} \tag{3.47}$$

in (3.46) sind $a_{m,k}$ und $a_{m,k-1}$ die letzten zwei Messwerte des Accelerometers. Nicht nur Änderungen der Beschleunigung führen zu einem höheren Wert dieses Parameters, sondern auch reine Rotationen der IMU, da der Vektor der Schwerebeschleunigung auch dadurch seine Lage im Sensorkoordinatensystem verändert. Weil aber reine Rotationen, bei denen sich die IMU im Rotationsmittelpunkt befindet, wohl eher die Ausnahme darstellen, und bei Rotationen, bei welchen die IMU eine gekrümmte Bahn beschreibt, äußere Beschleunigungen auftreten, wurde dieser Parameter als geeignet befunden, um das Vorhandensein äußerer Beschleunigungen anzuzeigen. Mit der abgeänderten Hypothese, die besagt, dass die lineare Beschleunigung nur zu gewissen Zeitpunkten null ist, ist das System aus Lage und linearer Beschleunigung zu genau diesen Zeitpunkten vollständig bestimmt. Um zwischen diesen Zeitpunkten ein unkontrolliertes Anwachsen der linearen Beschleunigung zu verhindern, beschränkt sich das Konzept zur Berücksichtigung der linearen Beschleunigungen nicht allein auf die Festlegung dieser Zeitpunkte, sondern besitzt insgesamt drei Schichten.

- Festlegung der linearen Beschleunigung auf null ausgelöst durch einen Parameterschwellenwert. Die Berechnung derselben erfolgt ansonsten mit der aktuellen Lage der IMU in einem separaten Kalman-Filter, das dem der Lageberechnung parallel geschaltet ist.

- Für den gegenteiligen Fall, $\mathbf{a}_{lin} \neq \mathbf{0}$, erfolgt dazu eine lose Fesselung der Messquaternion an die Lageänderungsberechnung mit den Messwerten des Gyroskops. Zusätzlich erfolgt eine Erhöhung der Rauschkovarianzmatrix für die Messung.
- Variation des Parameterschwellenwertes in mehreren parallel laufenden Filtersystemen und Auswahl jenes Ergebnisses bei dem der Betrag der linearen Beschleunigung am kleinsten ist.

Zusammenfassend lässt sich das Konzept so formulieren, dass die Nullsetzung der linearen Beschleunigung nur zu bestimmten Zeitpunkten erfolgt und ihr Betrag in den Zeiträumen dazwischen durch entsprechende Maßnahmen möglichst klein gehalten wird.

Rotationsmatrix der Messquaternion

Mit Berücksichtigung der linearen Beschleunigung errechnet sich der Vektor der Schwerebeschleunigung ${}^b\mathbf{g}$ aus dem gemessenen Beschleunigungsvektor ${}^b\mathbf{a}_m$ und der linearen Beschleunigung ${}^b\mathbf{a}_{lin}$

$$ {}^b\mathbf{g} = {}^b\mathbf{a}_m - {}^b\mathbf{a}_{lin} = \begin{pmatrix} a_{m,x} - a_{lin,x} \\ a_{m,y} - a_{lin,y} \\ a_{m,z} - a_{lin,z} \end{pmatrix} . \tag{3.48} $$

Gemeinsam mit dem Vektor der magnetischen Flussdichte ${}^b\mathbf{m} = \begin{pmatrix} m_x & m_y & m_z \end{pmatrix}^T$ erhält man für die Rotationsmatrix (3.39)

$$ {}^f\mathbf{R}_b^T = \begin{pmatrix} \begin{matrix} m_y(a_{m,z}-a_{lin,z})- \\ m_z(a_{m,y}-a_{lin,y}) \end{matrix} & \begin{matrix} -m_z(a_{m,x}-a_{lin,x})(a_{m,z}-a_{lin,z})+m_x(a_{m,z}-a_{lin,z})^2+ \\ m_x(a_{m,y}-a_{lin,y})^2-m_y(a_{m,x}-a_{lin,x})(a_{m,y}-a_{lin,y}) \end{matrix} & a_{m,x}-a_{lin,x} \\ \begin{matrix} m_z(a_{m,x}-a_{lin,x})- \\ m_x(a_{m,z}-a_{lin,z}) \end{matrix} & \begin{matrix} -m_x(a_{m,x}-a_{lin,x})(a_{m,y}-a_{lin,y})+m_y(a_{m,x}-a_{lin,x})^2+ \\ m_y(a_{m,z}-a_{lin,z})^2-m_z(a_{m,y}-a_{lin,y})(a_{m,z}-a_{lin,z}) \end{matrix} & a_{m,y}-a_{lin,y} \\ \begin{matrix} m_x(a_{m,y}-a_{lin,y})- \\ m_y(a_{m,x}-a_{lin,x}) \end{matrix} & \begin{matrix} -m_y(a_{m,y}-a_{lin,y})(a_{m,z}-a_{lin,z})+m_z(a_{m,y}-a_{lin,y})^2+ \\ m_z(a_{m,x}-a_{lin,x})^2-m_x(a_{m,x}-a_{lin,x})(a_{m,z}-a_{lin,z}) \end{matrix} & a_{m,z}-a_{lin,z} \end{pmatrix} \tag{3.49} $$

und daraus die Messquaternion

$$ \mathbf{q}_m = \mathbf{q}_m({}^f\mathbf{R}_b) \tag{3.50} $$

mit (3.35).

Berechnung der linearen Beschleunigung

Die Berechnung der linearen Beschleunigung erfolgt mit der aktuellen Lage der IMU und wird für die eben aufgeführte Rotationsmatrix benötigt, die danach in die Messquaternion umgerechnet wird. Das heißt, die lineare Beschleunigung muss vor dem Korrekturschritt des Lage-Kalman-Filters erfolgen. Zur Vorhersage der linearen Beschleunigung stehen daher die letzte Lageinformation $\mathbf{q}_k$ aus dem letzten Filterdurchlauf und die mit Hilfe des Gyroskops vorhergesagte Lage (Gleichung (3.43)) $\mathbf{q}_k^*$ zur Verfügung. Erfolgt die Berechnung relativ durch Addition zur linearen Beschleunigung des letzten Durchlaufes, muss dies im fixen ENU-Koordinatensystem wie in den nachstehenden Gleichungen (3.51)-(3.55) ausgeführt erfolgen.

$$ {}^{b}\mathbf{a}^*_{linear,k} = {}^{f}\mathbf{R}_b^T(\mathbf{q}_k^*) \cdot {}^{f}\mathbf{a}^*_{linear,k} \tag{3.51} $$

$$ {}^{f}\mathbf{a}^*_{linear,k} = {}^{f}\mathbf{a}_{linear,k} + ({}^{f}\mathbf{a}_{m,k} - {}^{f}\mathbf{a}_{m,k-1}) \tag{3.52} $$

mit

$$ {}^{f}\mathbf{a}_{linear,k} = {}^{f}\mathbf{R}_b(\mathbf{q}_k) \cdot {}^{b}\mathbf{a}_{linear,k} \tag{3.53} $$

$$ {}^{f}\mathbf{a}_{m,k} = {}^{f}\mathbf{R}_b(\mathbf{q}_k^*) \cdot {}^{b}\mathbf{a}_{linear,k} \tag{3.54} $$

$$ {}^{f}\mathbf{a}_{m,k-1} = {}^{f}\mathbf{R}_b(\mathbf{q}_k) \cdot {}^{b}\mathbf{a}_{linear,k-1} \tag{3.55} $$

Der Messwert ${}^{b}\mathbf{a}_{linear,m}$ für das Filter der linearen Beschleunigung errechnet sich analog, jedoch mit ${}^{f}\mathbf{R}_b(\mathbf{q}_{k+1})$ statt ${}^{f}\mathbf{R}_b(\mathbf{q}_k^*)$. Wird über den gesetzten Schwellenwert eine Nullsetzung der linearen Beschleunigung ausgelöst, so geschieht dies indirekt, indem ${}^{f}\mathbf{a}_{linear,k}$ mit null angenommen wird. Bei der absoluten Berechnung (3.56) und (3.57) hat die z-Achse des Koordinatensystems, ausgedrückt durch die Messquaternion, die exakt selbe Richtung wie bei der Quaternion, die zur Berechnung der linearen Beschleunigung verwendet wird.

$$ {}^{b}\mathbf{a}^*_{linear,k} = {}^{b}\mathbf{a}_{m,k} - {}^{b}\mathbf{g}^* \tag{3.56} $$

$$ {}^{b}\mathbf{g}^* = {}^{f}\mathbf{R}_b^T(\mathbf{q}_k^*) \cdot {}^{f}\mathbf{g} = {}^{f}\mathbf{R}_b^T(\mathbf{q}_k^*) \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{3.57} $$

Beide Varianten werden sowohl im Vorhersageschritt als auch im Korrekturschritt berechnet, ausgewählt wird immer die mit dem kleineren Betrag.

Kalman-Filter für den Bias des Gyroskops

Im Abschnitt für die Kalibrierung der IMU wurde auf eine einmalige Ermittlung des Gyroskopbias verzichtet, vielmehr findet die Berechnung im laufenden Betrieb des Filters statt. Dafür wurde ein eigenes Filter implementiert, das wie der Beschleunigungsfilter dem Lagefilter parallel geschaltet ist. Das System funktioniert zwar auch ohne den Gyroskopbias zu kompensieren, weil zu gegebener Zeit immer mit der absoluten Messquaternion korrigiert wird, das Verhalten verbessert sich jedoch zumindest subjektiv. Messungen mit Referenzinstrumenten, um eine Verbesserung objektiv feststellen zu können, wurden jedoch nicht durchgeführt. Das Verhalten des Gyroskopbias in die drei Achsenrichtungen wird dabei als unkorrelierte, Gauß'sche Zufallsbewegung

$$\mathbf{b}_{k+1} = \mathbf{b}_k + \mathbf{w}_k \tag{3.58}$$

modelliert. Die Messung des Bias erfolgt durch

$$\frac{2}{\Delta t}\mathbf{\Theta}({}^f\mathbf{q}_{b,k})^{-1}({}^f\mathbf{q}_{b,k+1} - {}^f\mathbf{q}_{b,k}) = \begin{pmatrix} 0 \\ \boldsymbol{\omega}_{b,q} \end{pmatrix} , \tag{3.59}$$

$$\mathbf{b}_m = \boldsymbol{\omega}_{b,m} - \boldsymbol{\omega}_{b,q} , \tag{3.60}$$

$$\boldsymbol{\omega}_{b,k} = \boldsymbol{\omega}_{b,m} - \mathbf{b}_k , \tag{3.61}$$

auf umgekehrte Weise zu (3.43) jedoch mit (3.40) über einen gesamten Lagefilterdurchlauf hinweg.

Gesamte Filterstruktur

Die Gesamtstruktur der drei parallel laufenden Kalman-Filter für Lage, lineare Beschleunigung und Gyroskop-Bias ist in Abbildung 3.28 übersichtlich dargestellt. Die Kalman-Filter selbst sind durch Doppelpfeile illustriert, die Verwendung der einzelnen Größen mit einfachen Pfeilen dargestellt. Die Abfolge der Rechenschritte wurde mit den eingekreisten Nummern gekennzeichnet. Alle drei Filter sind linear und besitzen auch dasselbe Messmodell (3.45). Da sich alle drei Filter gegenseitig beeinflussen, müssen Maßnahmen getroffen werden, sodass sich unerwünschte Effekte nicht fortpflanzen können. Das dreischichtige Maßnahmenpaket für die lineare Beschleunigung wurde bereits aufgeführt. Für den Bias des Gyroskops wurde Ähnliches implementiert:

- Ausführen des Bias-Kalman-Filters nur bei Unterschreiten des Schwellenwertes.

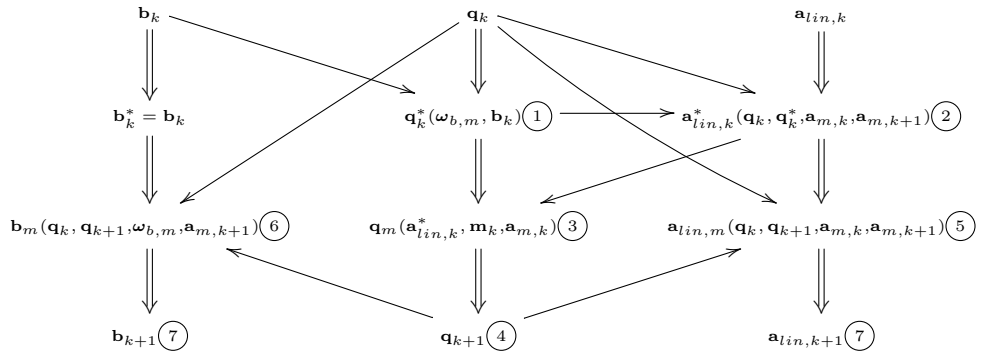


Abbildung 3.28: Struktur des Filtersystems

- Deckelung des Absolutbetrages des Bias in jede Richtung bei etwa dem Doppelten des maximalen Stillstandsbias.
- Permanente Dämpfung bei der Einberechnung des Bias in die Lagevorhersage um eine einstellige Prozentzahl sowie eine starke Dämpfung (40%) bei Überschreiten des Schwellenwertes.

Kovarianzmatrizen

Die einzelnen Varianzen für die Achsenrichtungen der jeweiligen Sensoren werden alle als unkorreliert angenommen. Valenti et al. geben in [34] für die Prozessrauschmatrix des Lagefilters jene Matrix an, die sich durch das Fortpflanzen des Messrauschens des Gyroskops durch das Prozessmodell ergibt. Das Messrauschen in Gleichung (3.43) bildet sich mit derselben linearen Funktion wie die Winkelgeschwindigkeit mit

$$\mathbf{w}_k = \frac{1}{2}\Delta t \boldsymbol{\Theta}({}^f\mathbf{q}_b) \cdot \boldsymbol{\sigma}_{gyro} \tag{3.62}$$

auf die Quaternion ab. Berechnet man das Bias in die Vorhersage der Quaternion ein, lässt sich mit $\boldsymbol{\Sigma}_{gyro} = \sigma_{gyro}^2 \cdot \mathbf{I}_{3\times 3}$ und $\mathbf{P}_{b,k+1}$, der Kovarianzmatrix des letzten Bias-Filter Durchlaufes, die Prozessrauschmatrix des Lagefilters zu

$$\mathbf{Q}_k = \left(\frac{\Delta t}{2}\right)^2 \boldsymbol{\Theta}(\boldsymbol{\Sigma}_g + \mathbf{P}_{b,k+1})\boldsymbol{\Theta}^T \tag{3.63}$$

berechnen. Die Berechnung der Rauschkovarianzmatrix für die Messquaternion erfolgt, wie auch in [34], analog, mit dem Unterschied, dass sowohl die Beschleunigung als auch die magnetische Flussdichte hochgradig nichtlinear

in die Berechnung der Messquaternion einfließen, weswegen die Propagation der Unsicherheiten nur in erster Näherung mit Hilfe der Jacobi-Matrix geschieht. Die Messquaternion als nichtlineare Funktion des Eingangsvektors $\mathbf{u}^T = \begin{pmatrix} g_x & g_y & g_z & m_x & m_y & m_z \end{pmatrix}$ mit der Kovarianzmatrix

$$\mathbf{\Sigma}_u = \begin{pmatrix} \mathbf{\Sigma}_{acc} + \mathbf{P}^*_{a_{linear}} & 0 \\ 0 & \mathbf{\Sigma}_{mag} \end{pmatrix} \tag{3.64}$$

mit den Kovarianzmatrizen

$$\mathbf{\Sigma}_{mag} = \begin{pmatrix} \sigma^2_{mag_x} & 0 & 0 \\ 0 & \sigma^2_{mag_y} & 0 \\ 0 & 0 & \sigma^2_{mag_z} \end{pmatrix},$$

$$\mathbf{\Sigma}_{acc} = \begin{pmatrix} \sigma^2_{acc_x} & 0 & 0 \\ 0 & \sigma^2_{acc_y} & 0 \\ 0 & 0 & \sigma^2_{acc_z} \end{pmatrix}$$

und der Kovarianzmatrix $\mathbf{P}^*_{a_{linear}}$ nach dem Prädiktionsschritt der linearen Beschleunigung. Für die Rauschmatrix der Messquaternion erhält man dann

$$\mathbf{R}_{k+1} = \mathbf{\Sigma}_{q_m} = \mathbf{J}_{q,u} \mathbf{\Sigma}_u \mathbf{J}^T_{q,u} \tag{3.65}$$

mit der Jacobi-Matrix

$$\mathbf{J}_{q,u} = \frac{\partial q_{m,i}}{\partial u_j} = \begin{pmatrix} \frac{\partial q_{m,0}}{\partial g_x} & \frac{\partial q_{m,0}}{\partial g_y} & \frac{\partial q_{m,0}}{\partial g_z} & \frac{\partial q_{m,0}}{\partial m_x} & \frac{\partial q_{m,0}}{\partial m_y} & \frac{\partial q_{m,0}}{\partial m_z} \\ \frac{\partial q_{m,1}}{\partial g_x} & \frac{\partial q_{m,1}}{\partial g_y} & \frac{\partial q_{m,1}}{\partial g_z} & \frac{\partial q_{m,1}}{\partial m_x} & \frac{\partial q_{m,1}}{\partial m_y} & \frac{\partial q_{m,1}}{\partial m_z} \\ \frac{\partial q_{m,2}}{\partial g_x} & \frac{\partial q_{m,2}}{\partial g_y} & \frac{\partial q_{m,2}}{\partial g_z} & \frac{\partial q_{m,2}}{\partial m_x} & \frac{\partial q_{m,2}}{\partial m_y} & \frac{\partial q_{m,2}}{\partial m_z} \\ \frac{\partial q_{m,3}}{\partial g_x} & \frac{\partial q_{m,3}}{\partial g_y} & \frac{\partial q_{m,3}}{\partial g_z} & \frac{\partial q_{m,3}}{\partial m_x} & \frac{\partial q_{m,3}}{\partial m_y} & \frac{\partial q_{m,3}}{\partial m_z} \end{pmatrix}, \tag{3.66}$$

die aufgrund ihrer Länge hier nicht vollständig ausgeführt wird. Sie wurde mit Hilfe der Symbolic Math Toolbox in MATLAB berechnet. Die Kovarianzmatrix für die Messung der linearen Beschleunigung wird die Fortpflanzung der Unsicherheiten durch die absolute Berechnungsmethode

$${}^b\mathbf{a}_{linear,m} = {}^b\mathbf{a}_{m,k} - {}^f\mathbf{R}^T_b(\mathbf{q}_{k+1}) \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{3.67}$$

herangezogen. Die lineare Beschleunigung kann bei dieser Berechnung als Funktion des Eingangsvektors $\mathbf{v}^T = \begin{pmatrix} a_{m,x} & a_{m,y} & a_{m,z} & q_{0,k+1} & q_{1,k+1} & q_{2,k+1} & q_{3,k+1} \end{pmatrix}$ aufgefasst werden. Mit der Jacobimatrix

$$\mathbf{J}_{a_{linear},v} = \frac{\partial a_{linear,i}}{\partial v_j} = \begin{pmatrix} 1 & 0 & 0 & 2q_{2,k+1} & -2q_{3,k+1} & 2q_{0,k+1} & -2q_{1,k+1} \\ 0 & 1 & 0 & -2q_{1,k+1} & -2q_{0,k+1} & -2q_{3,k+1} & -2q_{2,k+1} \\ 0 & 0 & 1 & -2q_{0,k+1} & 2q_{1,k+1} & 2q_{2,k+1} & -2q_{3,k+1} \end{pmatrix} \tag{3.68}$$

und

$$\mathbf{\Sigma}_v = \begin{pmatrix} \mathbf{\Sigma}_{acc} & 0 \\ 0 & \mathbf{P}_{k+1} \end{pmatrix} \tag{3.69}$$

erhält man für die Kovarianzmatrix der Messung der linearen Beschleunigung

$$\mathbf{R}_{a_{linear},k+1} = \mathbf{J}_{a_{linear},v} \mathbf{\Sigma}_v \mathbf{J}^T_{a_{linear},v} \,. \tag{3.70}$$

Aus (3.59) und (3.60) ergibt sich nach dem gleichen Prinzip die Kovarianzmatrix für die Biasmessung

$$\mathbf{R}_{b,k+1} = \frac{4}{\Delta t^2} \mathbf{\Theta}^{-1} (\mathbf{P}_k + \mathbf{P}_{k+1}) (\mathbf{\Theta}^{-1})^T + \mathbf{\Sigma}_{gyro} \,. \tag{3.71}$$

Filterabstimmung

Die Abstimmung des Filters beinhaltet die Quantifizierung der Kovarianzmatrizen für das Prozessrauschen von linearer Beschleunigung und Gyroskopbias, die Festlegung der Schwellenwerte für die drei parallel laufenden Filtersysteme und die Dämpfung bei der Berücksichtigung des Gyroskopbias sowie die Deckelung des Betrages des Gyroskopbias. Ebenso fällt eventuelles Filtertuning, also das Manipulieren von Kovarianzmatrizen per Hand, unter diesen Überbegriff. Insgesamt stellte sich die Abstimmung für das Gyroskopbias als die aufwändigste Aufgabe dar. Hier sind also durchaus Verbesserungen am Filterentwurf möglich. Die Abstimmung der Kovarianzmatrix für die lineare Beschleunigung auf das restliche System und die Festlegung der Schwellenwerte waren dagegen relativ unkompliziert. Häufig findet man für das Prozessrauschen Kovarianzmatrizen mit Vielfachen von Δt und auch in den oben aufgeführten Rauschmatrizen spielt die Zeitspanne eine Rolle, darum wurde Δt als Parameter in die Prozessrauschmatrizen übernommen. Für die sehr volatile Größe der linearen Beschleunigung ohne Vorfaktor,

$$\mathbf{Q}_{a_{linear},k} = \begin{pmatrix} \Delta t & 0 & 0 \\ 0 & \Delta t & 0 \\ 0 & 0 & \Delta t \end{pmatrix} . \tag{3.72}$$

Für die eher statische Größe des Gyroskopbias wurde die Zeitperiode mit entsprechendem Faktor versehen,

$$\mathbf{Q}_{b,k} = 10^{-5} \begin{pmatrix} \Delta t & 0 & 0 \\ 0 & \Delta t & 0 \\ 0 & 0 & \Delta t \end{pmatrix} . \tag{3.73}$$

Die Schwellenwerte des Parameters p_{midabl} für die drei parallel laufenden Filtersysteme wurden mit $th_1 = 0.04, th_2 = 0.05, th_3 = 0.06$ gewählt. Der Betrag des Gyroskopbias wurde bei $b_{max} = 0.04$ je Achse gedeckelt. Bei der Berechnung der Lagevorhersage wird das Gyroskopbias nicht vollständig, sondern abzüglich 3% Dämpfung berücksichtigt. Im Zuge des Filtertunings wurde einzig die Rauschmatrix der Biasmessung zu

$$\mathbf{R}_{b,k+1} = \mathbf{R}_{b,k+1} \frac{2}{\Delta t} \tag{3.74}$$

abgeändert. Diese Abstimmung wurde im Zuge mehrerer Versuche empirisch ermittelt und für geeignet befunden.

Zusätzliche Anmerkungen zum Filter

Wie bereits im Abschnitt zu den Quaternionen erwähnt wurde, ändert sich die dargestellte Rotation nicht, wenn alle Vorzeichen einer Quaternion negiert werden. Kommt es im Zuge des Rechenprozesses zu Operationen zwischen Quaternionen, welche dieselbe oder annähernd dieselbe Roation abbilden, so muss, wie auch in [34] erwähnt, auf Vorzeichengleichheit geachtet werden. Ebenso bewahren nicht alle Operationen die Einheitsnorm, weswegen wiederholte Normierungen durchzuführen sind.

Das Verhalten der Filterung der linearen Beschleunigung ist durchaus als gut bis sehr gut zu bezeichnen, es bestehen aber noch Verbesserungsmöglichkeiten bei der Miteinberechnung des Gyroskopbias. Bei der Abstimmung des Filters für das Bias gilt es mehrere Faktoren gegeneinander abzustimmen, die von mehreren Seiten beeinflusst werden. Ist die Gyroskopempfindlichkeit oder das Magnetometer nicht perfekt kalibriert, führen diese Fehler bei Drehbewegungen im beschriebenen Filteraufbau zu fehlerhaften Messungen des Bias, allerdings ist eine perfekte Kalibrierung der Gyroskopempfindlichkeit aufwändig, zumal hier laut Datenblatt eine Temperaturabhängigkeit nicht auszuschließen ist. Selbige wurde beim Magnetometer nachgewiesen. Einmal eingeflossen, müssen fehlerhafte Biasmessungen daher im entsprechend abgestimmten Filter möglichst rasch wieder korrigiert werden, was der Abstimmung für ein eigentlich wenig veränderliches Gyroskopbias widerspricht. In zukünftigen Entwicklungsschritten wären zusätzliche Maßnahmen, sodass sich diese Fehler nicht oder nur

in äußerst geringer Weise auf die Biasmessung auswirken, wünschenswert. Die Schwellenwerte zur Festlegung der Zeitpunkte für die Nullsetzung der linearen Beschleunigung sind so zu wählen, dass im Betrieb der IMU ausreichend oft eine Nullsetzung erfolgt, um die Stabilität des Filters zu gewährleisten.

Die Ausgabe des Filtersystems ist die Lage des IMU Koordinatensystems gegenüber dem East-North-Up Koordinatensystem, allerdings gegenüber dem *magnetischen* ENU-System. Die als Deklination bezeichnete Abweichung zwischen geografisch und magnetisch Nord gilt es daher im Ergebnis zu berücksichtigen. Während der Versuchsdurchführungen lag sie bei etwa 4.3° [32].

3.3.4 Aufbereitung der GPS Messungen

GPS und IMU liefern absolute Daten in mit der Erde fix verbundenen Koordinatensystemen. Odometrisch errechneten Wegstrecken liegt hingegen eine Fläche zugrunde, auf welcher die Bewegungen des gegenständlichen Fahrzeuges stattfinden, die keinen vorgegebenen Fixbezug zur Erde besitzt. Dabei muss diese Ebene auch nicht gezwungenermaßen eben und horizontal sein, sehr wohl jedoch unter den bereits für diese Arbeit getroffenen Voraussetzungen und Annahmen. Findet die Bewegung lokal begrenzt auf annähernd horizontalen Flächen statt, stellen diese nicht nur eine gute Grundlage für die Odometrie dar, sondern es bietet sich für solche Flächen auch gleichzeitig dasselbe East-North-Up-Koordinatensystem an, demgegenüber auch die Orientierung der IMU dargestellt wird. Die GPS-Informationen zur Position auf der Erde werden jedoch in Längen- und Breitengraden eines Referenzellipsoids (im vorliegenden Fall der sehr häufig verwendete WGS84 Referenzellipsoid) ausgegeben und müssen daher mit geeigneten Projektionsmethoden in ein ähnliches Format gebracht werden [51]. Das Universal Transverse Mercator (UTM) Koordinatensystem erfüllt diese Eigenschaften in sehr guter Näherung. Es besitzt verschiedene Zonen, deren x-Achsen in der Mitte der Zone parallel zum Äquator verlaufen und deren positive Richtung Osten ist. Die y-Achsen (Gitternord) zeigen in den Zonenmittelpunkten zum geografischen Norden. Durch die Projektion der gekrümmten Erdoberfläche auf eine ebene Fläche kommt es zu Verzerrungen, sodass es unter anderem zum Rand der Zonen zu Abweichungen von geografischem Norden und Osten zu Gitternord und Gitterost kommt. Diese Abweichungen sind allerdings gering und werden daher im Zuge dieser Arbeit vernachlässigt, könnten aber ebenso wie die magnetische Deklination einfach kompensiert werden. Für die Durchführung der UTM Projektion steht in ROS mit dem *utm_odometry_node* des *gps_common* Package Code zur Verfügung, der genau diese Projektion ausführt. Die Standardabweichung der GPS-Position wird in Metern ausgegeben, somit ist hier keine Anpassung notwendig und die Kovarianzmatrix kann grundsätzlich ohne Transformationen oder dergleichen

übernommen werden.

3.3.5 Testaufbau

Für einen vereinfachten Versuchsaufbau bei der Fusion von Odometrie, IMU und GPS wurde eine Plattform aufgebaut, die navigatorisch einfacher zu manipulieren ist als FRANC, aber dennoch die erforderlichen odometrischen Daten liefert. Dazu wurde ein handelsüblicher Bollerwagen mit zwei Encodern an der fixen Achse zur Abtastung der Raddrehwinkel ausgestattet und mit einem Potentiometer zur Messung des Lenkwinkels der Drehschemellenkung versehen, siehe Abbildung 3.29.

Abbildung 3.29: Bollerwagen mit Sensoren zur Versuchsdurchführung

Die Struktur der bei der Versuchsdurchführung aktiven ROS Nodes ist in Abbildung 3.30 dargestellt.

3.3.6 Fusion von GPS, Odometrie und IMU mittels EKF

Zur Fusion der Odometriedaten, IMU-Orientierung und GPS-Position stehen in ROS zwei bereits implementierte Filter bereit. Das erste, enthalten im *robot_localization* Package, wurde von Tom Moore et al. in [52] vorgestellt, darüber hinaus ist eine sehr umfangreiche Dokumentation[1] verfügbar. Bei diesem Filter handelt es sich um eine eigentlich sehr leistungsfähige Implementierung eines erweiterten Kalman-Filters ohne Begrenzung der Sensoranzahl,

[1] Vgl. http://docs.ros.org/lunar/api/robot_localization/html/index.html

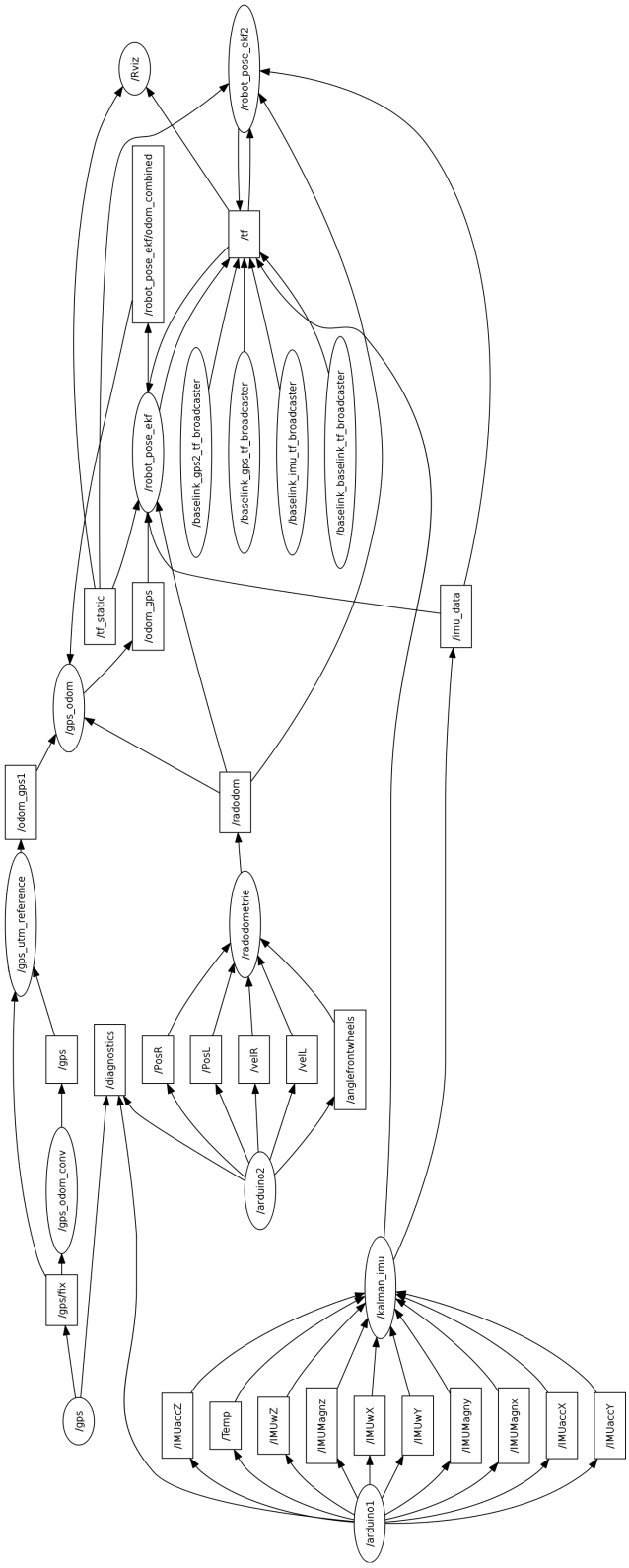


Abbildung 3.30: An der Sensorfusion beteiligte Nodes

die fusioniert werden soll. Leider zeigte das Filter auch nach mehreren Abstimmungsversuchen nicht das gewünschte Verhalten, weswegen als Alternative die Implementierung des *robot_pose_ekf* Package herangezogen wurde. Die Ausgangslage war hier genau diametral: sehr wenig verfügbare Dokumentation, dennoch zeigte das Filter schon bei den ersten Versuchen sofort das erwartete Verhalten. Laut aktueller Dokumentation [53] ist das Filter auf die Fusion von Informationen ausgelegt, die es über die drei Topics *odom*, *imu_data* und *vo* (Vision-Odometry) entgegen nimmt. Außerdem wird eine Anleitung aufgeführt, um dem Filter über das *vo* Topic GPS-Informationen zuzuführen. Über die Parameter *odom_used*, *imu_used* und *vo_used* wird dem Filter mitgeteilt, welche Topics zu fusionierende Informationen liefern. Interessanterweise existiert ein zusätzlicher, nicht dokumentierter Parameter *gps_used*. Nach Recherche wurde der Parameter mit den GPS Informationen mit dem Topic *gps*, wie in [54] beschrieben erfolgreich verwendet. Die Dokumentation gibt auch Auskunft darüber, wie die Informationen über die Topics *odom*, *imu_data* und *vo* in das Filter einfließen, so wird beispielsweise von *odom* nur der Yaw-Winkel berücksichtigt während Roll und Pitch ignoriert werden, zumal räumliche Odometrie an eine (unbekannte) Fläche gebunden ist. Das dem Filter zugrunde liegende Modell ist dennoch in der Lage Bewegung in drei Raumdimensionen abzubilden, die dreidimensionalen Informationen von *imu_data* und *vo* werden daher auch vollständig mit einberechnet. Grundsätzlich werden laut Dokumentation Informationen nicht absolut sondern relativ zueinander in die Berechnung einbezogen, was für die Odometrie erwünscht ist, bei GPS-Positionen den gewünschten Effekt jedoch konterkariert. In [54] wird allerdings aufgeführt, dass die Informationen des GPS bei Verwendung des *gps_used* Parameters absolut verarbeitet werden. Wenngleich das Filter in der Lage ist, eine dreidimensionale Fusion durchzuführen, wurden die Höhendaten des GPS-Receivers für die Fusion mit Hinblick auf die Versuchsdurchführung auf beinahe horizontalen Flächen und der größeren vertikalen Messunsicherheit auf null gesetzt. Der stetig anwachsende Fehler bei odometrischen Berechnungen muss sich auch in der zugehörigen Kovarianzmatrix abbilden. Durch die numerische Integration von Weg und Orientierung und der entsprechenden Fehlerkumulation in beiden Größen, macht es aber keinen Sinn die Berechnung der Varianz der Position mittels Fehlerfortpflanzung durch ein von diesen (völlig falschen) Größen abhängigen Modell durchzuführen, vielmehr ist, wie auch in [53] angegeben, die Änderung der Kovarianz über der Zeit interessant. Da für das Filtertuning ein erheblicher Eingriff in die Kovarianzmatrizen erwartet worden war, wurde für die Odometrie jene Unsicherheit als Ausgangslage für den Abstimmungsprozess herangezogen, die der Unsicherheit der mittleren Geschwindigkeit entspricht. Diese Unsicherheit wurde im verlangten Format der *nav_msgs/Odometry* Messages in jedem Zeitschritt für die Kovarianz der

Geschwindigkeit hinterlegt und zur Kovarianzen von x- und y-Position addiert, dadurch entspricht die Änderung der Positionskovarianz jener der Geschwindigkeit. Die mittlere Geschwindigkeit während eines Zeitschritts errechnet sich zu

$$\bar{v} = \frac{x_{neu} - x_{alt}}{\Delta t}\,, \tag{3.75}$$

mit der von den Encodern gelieferten Wegdifferenz $x_{neu} - x_{alt}$, wobei sowohl $x_{neu} - x_{alt}$ als auch Δt mit einer Unsicherheit durch die Diskretisierung behaftet sind. Bei beiden liegt die Varianz dieses Quantisierungsfehlers unter Zugrundelegung einer stetigen Gleichverteilung bei einem Zwölftel des Quadrats der Diskretisierungsschrittweite. Daraus ergibt sich analog zu (3.70) für eine Wegdifferenz gleich der Diskretisierungsschrittweite die Varianz der mittleren Geschwindigkeit zu

$$\sigma_{\bar{v}}^2 = \frac{\Delta x^2}{6 \cdot \Delta t^2}\,. \tag{3.76}$$

Während der ersten Versuche mit dem Filter wurde der Faktor 6 im Nenner auf 100 erhöht, um der Odometrie stärkeres Gewicht zu verleihen. Im Stillstand, also wenn von den Decodern keine Änderung der Raddrehwinkel gemessen worden war, wurde die Varianz auf den kleinen Wert $\sigma_{\bar{v}}^2 = 10^{-13}$ gesenkt. Weitere Modifikationen an den Varianzen erfolgten nur mehr an der Kovarianzmatrix der GPS Messungen, wie im nächsten Kapitel ausgeführt wird.

4 Ergebnisse und Experimente

Bei der Durchführung der Versuche stellte sich gleich zu Beginn die Frage nach der absoluten Position des Startpunktes, die den GPS-Messungen zugrunde gelegt werden sollte und die damit im Weiteren, weil das GPS als einziger Sensor die absolute Position liefert, zwangsläufig auch die Startposition des Filterergebnisses ist. Damit sich der momentane Fehler der ersten GPS Position nicht vollständig im Filterergebnis niederschlägt, wurde eine GPS-Positionsmittelung implementiert, die solange läuft, bis eine erste Radbewegung von den Decodern detektiert wird. Der so erhaltene Mittelwert wurde als Startwert für den in Abbildung 4.1 dargestellten Testlauf festgelegt. Dadurch wurde die Startposition jedoch von der Verweilzeit vor Bewegungsbeginn abhängig. Um diese unerwünschte Abhängigkeit zu beseitigen wurde ab Abbildung 4.5 für alle weiteren Testläufe dieselbe Startposition hinterlegt, die mit der tatsächlichen Position näherungsweise über die abgebildete Karte ermittelt worden war. Durch diese Maßnahme erhält man ein „pseudo-absolutes“ Ergebnis, das mit der Situation vergleichbar ist, wenn ein autonomes Fahrzeug von einer bekannten Position (beispielsweise eine Garagenausfahrt) startet und nach einer Fahrt zum Ausgangspunkt zurückkehrt. Für alle Abbildungen zur Illustration der Filterergebnisse gilt folgendes Farbschema:

- *gelber Pfad:* Ausgabe des GPS-Receivers,
- *magentafarbener Pfad:* Ergebnis der Odometrie;
- *roter Pfad:* Ausgabe des EKF nach Fusion von GPS, IMU und Odometrie,
- *blauer Pfad:* Ausgabe eines zweiten EKF zur Fusion von IMU und Odometrie.

Der Anfang des roten Pfades ist in den Abbildungen mit S und sein Ende mit E gekennzeichnet . Zur Darstellung wurden die Kapazitäten von *Google Earth™* genutzt.
Durch den gemittelten respektive hinterlegten Startpunkt liegen die ersten GPS-Positionen, die in die Berechnungen des Filters einfließen abseits dieses Punktes, daher wurde eine Anpassung der Kovarianzmatrix vorgenommen, um zu verhindern, dass der Filter diesen ersten Punkten zu starkes Gewicht verleiht.

Abbildung 4.1: Testlauf mit konstanter Varianzzugabe $\sigma^2_{gps,TL1} = \sigma^2_{gps,start} + 3$, © 2017 Google LLC, Verwendung mit Genehmigung. Google und das Google-Logo sind eingetragene Marken von Google LLC

Die Anpassung geschieht durch Addieren der Strecke, die zwischen gemitteltem Startpunkt (S) und der ersten GPS-Position (P) nach Bewegungsbeginn liegt, jedoch multipliziert mit einem abklingenden Faktor,

$$\sigma^2_{gps,start} = \sigma^2_{gps} + |\mathbf{r}_{S,P}| \cdot 0.97^k \,. \tag{4.1}$$

Darin ist k die Anzahl der empfangenen GPS-Positionen seit Bewegungsbeginn. Der GPS-Receiver lieferte während allen Testläufen für die Varianzen in x- und y-Richtung immer denselben Wert, daher wurden bei Modifikationen an der Varianz immer auch beide durch den jeweiligen Wert ersetzt. Im ersten in Abbildung 4.1 und Abbildung 4.2 abgebildeten Testlauf ist deutlich zu erkennen, dass das Filter nach Abklingen der anfänglichen Varianzzugabe die GPS-Position beinahe unverändert übernimmt, was einem gänzlich unerwünschten Verhalten entspricht. Anzumerken ist jedoch, dass der GPS-Receiver während dieses Testlaufes eine sehr kleine Kovarianz in der Größenordnung von eins lieferte. Darum wurde zur Varianz während dieses Testlaufes ein konstanter Betrag von drei addiert, was aber offensichtlich nicht zum gewünschten Ergebnis führte, deswegen wurde die Konstante beim nächsten Testlauf auf zehn erhöht.

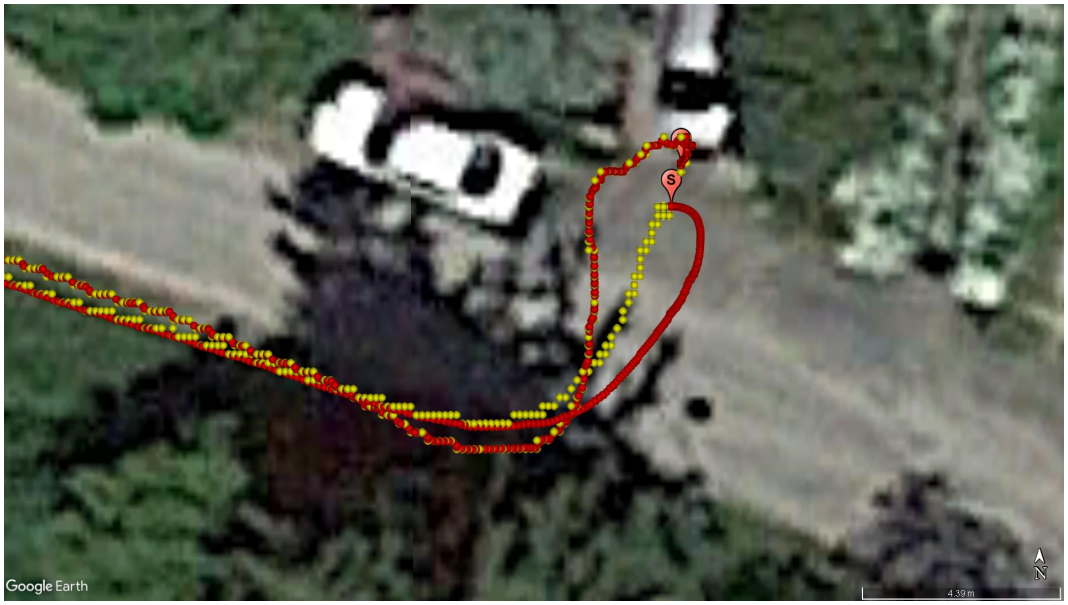


Abbildung 4.2: Testlauf mit konstanter Varianzzugabe $\sigma^2_{gps,TL1} = \sigma^2_{gps,start} + 3$, © 2017 Google LLC, Verwendung mit Genehmigung. Google und das Google-Logo sind eingetragene Marken von Google LLC

Im Testlauf von Abbildung 4.3 und Abbildung 4.4 mit erhöhter additiver Varianzkonstante zeigt sich ein deutlich besseres Filterverhalten, das Filter liefert Ergebnisse, die auch abseits des GPS-Pfades liegen, allerdings lag bei diesem Durchlauf nicht nur die additive Konstante höher, sondern auch die vom GPS-Empfänger selbst übermittelte Varianz lag schon auf einem Niveau von etwa fünf, was insgesamt zu einer Varianz von deutlich über zehn führte. Zusätzlich profitiert der Verlauf des gefilterten Pfades am letzten Stück von der starken Ähnlichkeit zwischen Odometrie und GPS-Pfad in der letzten engen Kurve. Um dies zu verdeutlichen wurde in Abbildung 4.3 auch der Pfad der Odometrie eingeblendet.

Abbildung 4.3: Testlauf mit konstanter Varianzzugabe $\sigma^2_{gps,TL1} = \sigma^2_{gps,start} + 10$, © 2017 Google LLC, Verwendung mit Genehmigung. Google und das Google-Logo sind eingetragene Marken von Google LLC

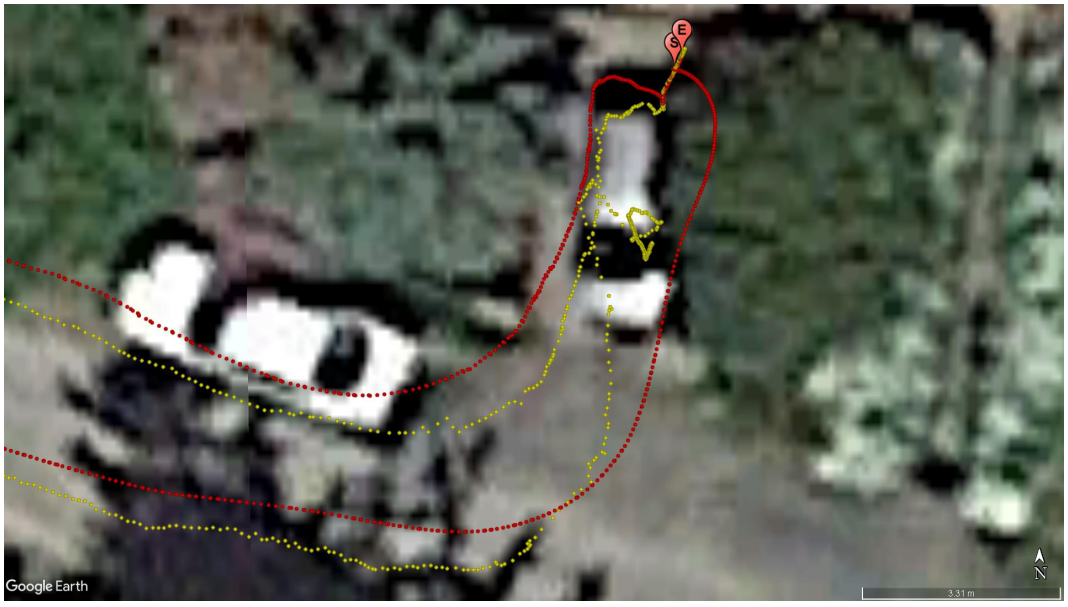


Abbildung 4.4: Testlauf mit konstanter Varianzzugabe $\sigma^2_{gps,TL1} = \sigma^2_{gps,start} + 10$, © 2017 Google LLC, Verwendung mit Genehmigung. Google und das Google-Logo sind eingetragene Marken von Google LLC

Abbildung 4.5: Testlauf mit um einen Faktor erhöhter Varianz $\sigma^2_{gps,TL3} = 2.5 \cdot \sigma^2_{gps} + |\mathbf{r}_{S,P}| \cdot 0.97^k$, © 2017 Google LLC, Verwendung mit Genehmigung. Google und das Google-Logo sind eingetragene Marken von Google LLC

Abbildung 4.6: Testlauf mit um einen Faktor erhöhter Varianz $\sigma^2_{gps,TL3} = 2.5 \cdot \sigma^2_{gps} + |\mathbf{r}_{S,P}| \cdot 0.97^k$, © 2017 Google LLC, Verwendung mit Genehmigung. Google und das Google-Logo sind eingetragene Marken von Google LLC

Als Alternative zur Addition eines konstanten Betrages um die Varianz zu erhöhen wurde im dritten abgebildeten Testlauf (Abbildung 4.5 und Abbildung 4.6) die Multiplikation von σ^2_{gps} mit einem konstanten Faktor von 2.5 getestet. Die Varianz ergibt sich für diesen Testlauf damit zu

$$\sigma^2_{gps,TL3} = 2.5 \cdot \sigma^2_{gps} + |\mathbf{r}_{S,P}| \cdot 0.97^k \,. \tag{4.2}$$

Bei diesem Testlauf war die vom GPS-Empfänger ausgegebene Varianz aber wiederum sehr klein, sodass sich die an das Filter übermittelte Varianz in einem Bereich von etwa drei bis zehn bewegte, was zum selben unerwünschten Verhalten wie in Abbildung 4.2 führte. Als zusätzliches Fehlverhalten ist in Abbildung 4.6, aber auch Abbildung 4.4 deutlich erkennbar, dass das Filter auf dem letzten Teil des Pfades dem GPS folgt, obwohl die Odometrie in Abbildung 4.4 und die Kombination aus Odometrie und IMU Abbildung 4.6 ganz klar Stillstand anzeigen. Um diesem Verhalten entgegenzuwirken wurde für den nächsten Testlauf (Abbildung 4.7 und Abbildung 4.8) für die an das Filter übermittelten GPS-Daten eine Stillstandsmittelung implementiert. Wie bei der ersten Variante der Startpositionsermittlung wird Stillstand über die Encoder der Räder detektiert und der Durchschnitt über alle GPS-Positionen während des Stillstandes gebildet. Allerdings wird der Mittelwert nicht sofort an das Filter weitergeleitet, sondern ein gewichteter Wert aus Durchschnitt und letzter Position des Filters.

$$\begin{pmatrix} x \\ y \end{pmatrix}_{GPS,Stillstand} = \begin{pmatrix} x \\ y \end{pmatrix}_{Mittelwert} \cdot \frac{k}{10000} + \begin{pmatrix} x \\ y \end{pmatrix}_{letztePosition} \cdot \frac{10000 - k}{10000} \tag{4.3}$$

Worin k abermals die Anzahl der GPS-Messungen seit Beginn des Stillstands ist. Für $k > 10000$ gilt $\mathbf{x}_{GPS,Stillstand} = \mathbf{x}_{Mittlewert}$. Bei einer GSP-Messrate von 10Hz geschieht der vollständige Übergang von der letzten Filterposition zum Mittelwert also in einer guten viertel Stunde. Für die Varianz nach Ende des Stillstandes wurde eine Erhöhung wie in (4.1) implementiert.

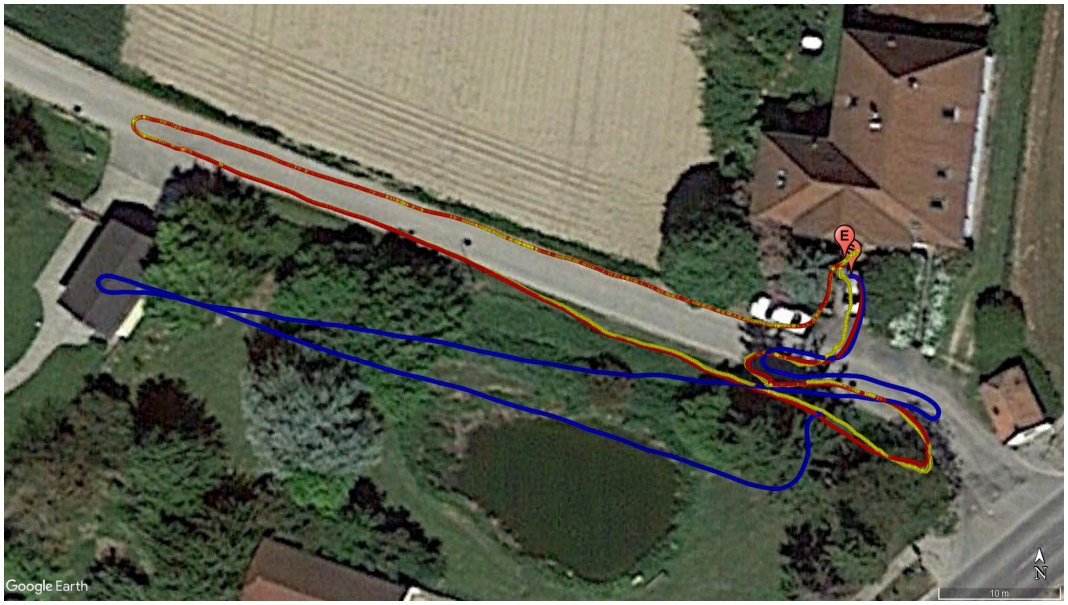


Abbildung 4.7: Testlauf mit um einen Faktor erhöhter Varianz $\sigma^2_{gps,TL4} = 2.5 \cdot \sigma^2_{gps} + |\mathbf{r}_{S,P}| \cdot 0.97^k$, © 2017 Google LLC, Verwendung mit Genehmigung. Google und das Google-Logo sind eingetragene Marken von Google LLC

Abbildung 4.8: Testlauf mit um einen Faktor erhöhter Varianz $\sigma^2_{gps,TL4} = 2.5 \cdot \sigma^2_{gps} + |\mathbf{r}_{S,P}| \cdot 0.97^k$, © 2017 Google LLC, Verwendung mit Genehmigung. Google und das Google-Logo sind eingetragene Marken von Google LLC

Der Faktor von 2.5 vom vorhergehenden Testlauf war nicht verändert worden, jedoch lag das Niveau der Varianz aufgrund der vom GPS-Empfänger gelieferten Varianz leicht über jener des Testlaufes davor, insgesamt aber dennoch unter zehn. Wegen dieser abermals sehr niedrigen Varianz folgt das Filter dem GPS fast exakt, die Verbesserung durch die Stillstandsmittelung ist aber am Ende des roten Pfades klar erkennbar. Für den nächsten Testlauf (Abbildung 4.9 und Abbildung 4.10) wurde ein als minimaler Wert für die Varianz 14.5 festgelegt und der Faktor für die Erhöhung zusätzlich auf 18.5 erhöht.

$$\sigma^2_{gps,TL5} = 18.5 \cdot \sigma^2_{gps} + |\mathbf{r}_{S,P}| \cdot 0.97^k \quad \text{und} \quad \sigma^2_{gps,TL5} \geq 14.5 \tag{4.4}$$

Durch diese Maßnahmen lag das Niveau der Varianz während dieses Testlaufes rund um den sehr hohen Wert von 50. Der vom Filter ausgegebene Pfad wird dadurch wie gewünscht nicht wie zuvor (beinahe) ausschließlich durch den GPS-Pfad bestimmt.

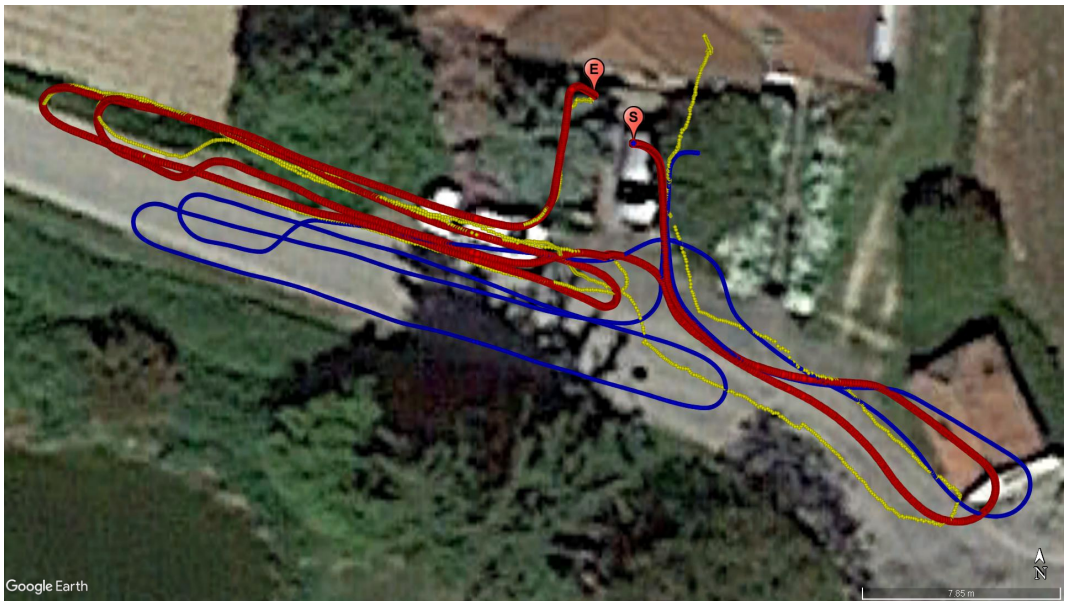


Abbildung 4.9: Testlauf mit um einen Faktor erhöhter Varianz $\sigma^2_{gps,TL5} = 18.5 \cdot \sigma^2_{gps} + |\mathbf{r}_{S,P}| \cdot 0.97^k$ und $\sigma^2_{gps,TL5} \geq 14.5$, © 2017 Google LLC, Verwendung mit Genehmigung. Google und das Google-Logo sind eingetragene Marken von Google LLC

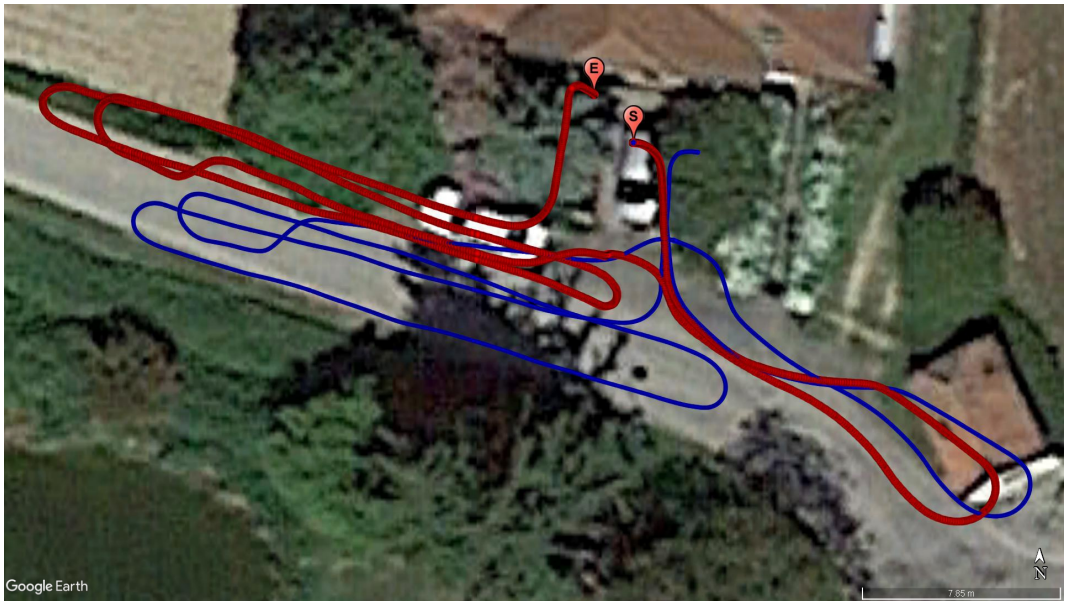


Abbildung 4.10: Testlauf mit um einen Faktor erhöhter Varianz $\sigma^2_{gps,TL5} = 18.5 \cdot \sigma^2_{gps} + |\mathbf{r}_{S,P}| \cdot 0.97^k$ und $\sigma^2_{gps,TL5} \geq 14.5$, © 2017 Google LLC, Verwendung mit Genehmigung. Google und das Google-Logo sind eingetragene Marken von Google LLC

5 Diskussion und Ausblick

Das Ergebnis der Simulation zur Evaluierung, ob sich die Werkzeuge des Navigation Stack für FRANC eignen, wurde bereits an früherer Stelle vorweggenommen, bleibt also noch die Frage, ob FRANC mit den getesteten kostengünstigen Sensoren in zufriedenstellender Weise navigieren könnte. Meist verfügt ein Feldroboter, so auch FRANC, über weitere Algorithmen, beispielsweise eine Reihenerkennung, die ihm zusätzliche Unterstützung für die Navigation bieten. Ist eine solche Reihenstruktur, beispielsweise bei der Aussaat, nicht vorhanden, muss der Roboter aber auch ohne solche zusätzlichen Informationen navigieren können. Wenn man als Abstand zwischen zwei Feldfruchtreihen die Distanz von 0.5m annimmt, so wäre es für einen Feldroboter sinnvoll, dass er mit einer Absolutgenauigkeit navigieren kann, die kleiner als dieser Wert ist. Dadurch wäre es dem Roboter etwa möglich, sofern bereits Pflanzenreihen vorhanden sind, einzelne, beliebige ausgewählte Reihen gezielt anzufahren oder, sofern noch keine Reihen vorhanden sind, zumindest gegebene Grenzen von zu bewirtschaftender Ackerfläche ausreichend genau zu erkennen. Können die Sensoren und Algorithmen zur Positionsbestimmung eine solche Genauigkeit nicht bieten, müssen zusätzliche Methoden implementiert werden, die es dem Roboter ermöglichen, neben eventueller Reihen auch andere Merkmale vor Ort (Straßen, Grenzsteine oder Ähnliches) zur Positionsbestimmung zu nutzen.

Betrachtet man Abbildung 4.10 des letzten Testlaufes, erkennt man, dass das Filterergebnis mit GPS (roter Pfad) kaum besser ist als jenes das sich aus Fusion von Odometrie und IMU ergab (blauer Pfad). Zwar erkennt man stellenweise, beispielsweise ganz rechts und in der Mitte, eine Verbesserung, wo sich der blaue Pfad auf längeren Abschnitten auf nicht befahrbarem Terrain befindet, jedoch ist im linken Teil der Abbildung genau das Gegenteil der Fall, wo der Pfad von Odometrie kombiniert mit der IMU ein deutlich besseres Ergebnis liefert. Hier ist auch unbedingt anzumerken, dass sich alle Testfahrten ausschließlich auf befestigten Untergrund beschränkten und Bereiche mit Vegetation niemals befahren wurden. Der abschließende Befund lässt sich anhand der Distanz zwischen den Anfangs- und Endpunkten der Pfade verdeutlichen. Diese Distanz beträgt für den roten Pfad 2.30m und 2.44m für den blauen. Betrachtet man dieses Ergebnis zusammen mit Abbildung 4.7, wo der Abstand zwischen Anfangs- und Endpunkt des blauen Pfades deutlich größer ist, kann man die Aussage treffen, dass die Genauigkeit des verwendeten

GPS-Systems nicht ausreicht, um auf kurzen Strecken eine Verbesserung des Ergebnisses gegenüber der Fusion von Odometrie und IMU zu erzielen, sondern lediglich dafür geeignet ist, für Bewegungen über längere Distanzen, die Abweichung der Fusion von Odometrie und IMU auf ein Maß zu reduzieren, das in etwa in dem Bereich liegt, der für das GPS-System als statische Genauigkeit angegeben ist. Will man also eine absolute Genauigkeit von unter einem halben Meter erreichen, benötigt man ein GPS-System, das eine solche Genauigkeit bietet. An dieser Stelle muss aber darauf hingewiesen werden, das eine andere Filterarchitektur zur Fusion von GPS, IMU und Odometrie hier möglichweise bessere Ergebnisse liefert. Wie in Kapitel 2.4.4 angeschnitten, sind Kalman-Filter nicht für die Verarbeitung von Daten gedacht, die Zeitkorrelationen aufweisen. Aus diesem Grund war es auch notwendig, die Varianz der GPS-Messungen deutlich zu erhöhen, um zu verhindern, dass das Filter den Pfad des GPS zu optimistisch bewertet. Eine künstliche Unterdrückung der Zeitkorrelationen im GPS-Signal könnte hier möglicherweise bessere Abhilfe schaffen. Nichtsdestotrotz weist die Ausgabe des EKF ein deutlich verbessertes zeitliches Verhalten im Vergleich zum ungefilterten GPS-Signal auf. Die gefilterte Trajektorie ist in ihrer Erscheinung so glatt wie jene, die man durch reine Odometrie erhält, sodass man sie grundsätzlich als Informationsquelle für die Navigationswerkzeuge verwenden kann. Beim Einsatz von günstigen DGPS - oder RTK-GPS Receiver ohne laufende Kosten wäre es wohl auch möglich die gewünschte Genauigkeit zu erreichen. Eine erhöhte Absolutgenauigkeit ist aber auch aus anderen Gründen wünschenswert. Wenngleich alle Tests für die Navigation unter der Annahme einer Bewegung in der Ebene erfolgten, ist dies keinesfalls eine adäquate Annahme für den tatsächlichen Betrieb eines Feldroboters. Viele Äcker weisen in ihrer Topographie mehrere, oft dutzende Meter Höhendifferenz auf, das Navigationssystem eines Feldroboters muss daher auch in der Lage sein auf dreidimensionalen Untergründen zu navigieren und auch ebensolches Kartenmaterial verarbeiten können. Der Hintergedanke, FRANC auf zukünftige Entwicklungsschritte vorzubereiten, bei denen die Navigation und Kartierungsaufgaben auf unebenen Untergründen sicherlich eine Rolle spielen werden, war einer der Gründe, weswegen der vermeintlich übers Ziel hinausschießende Schritt, ein eigenes Filter für die IMU zu entwickeln, in diese Diplomarbeit aufgenommen wurde. Die Genauigkeit der Lagewinkelinformation steht und fällt dabei mit der Kalibrierung der Sensoreinheit, ausführliche Erläuterungen der Methoden dafür waren daher unumgänglich und zeigten die Schwächen günstiger Sensoren. Eine Temperaturabhängigkeit der Kalibrierparameter vergrößert den Aufwand für die Kalibrierung unverhältnismäßig, die Wahl einer gegebenenfalls etwas teureren Sensoreinheit, bei der solche Effekte nicht zu erwarten sind, ist daher zu bevorzugen.

Literatur

[1] A. Bechar und C. Vigneault, „Agricultural robots for field operations: Concepts and components“, *Biosystems Engineering*, Jg. 149, S. 94–111, 2016.

[2] B. Soane und C Van Ouwerkerk, „Soil compaction problems in world agriculture“, in *Developments in agricultural engineering*, Bd. 11, Elsevier, 1994, S. 1–21.

[3] H. Binswanger, „Agricultural mechanization: a comparative historical perspective“, *The World Bank Research Observer*, Jg. 1, Nr. 1, S. 27–56, 1986.

[4] M. Hanna, „Estimating the field capacity of farm machines“, 2016.

[5] G. E. Moore, „Cramming more components onto integrated circuits“, *Proceedings of the IEEE*, Jg. 86, Nr. 1, S. 82–85, 1998.

[6] D. Pimentel, P. Hepperly, J. Hanson, D. Douds und R. Seidel, „Environmental, energetic, and economic comparisons of organic and conventional farming systems“, *BioScience*, Jg. 55, Nr. 7, S. 573–582, 2005.

[7] C. G. Sørensen, N. A. Madsen und B. H. Jacobsen, „Organic farming scenarios: operational analysis and costs of implementing innovative technologies“, *Biosystems engineering*, Jg. 91, Nr. 2, S. 127–137, 2005.

[8] P. Morin und C. Samson, „Motion control of wheeled mobile robots“, in *Springer Handbook of Robotics*, Springer, 2008, S. 799–826.

[9] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.

[10] J. Minguez, F. Lamiraux und J.-P. Laumond, „Motion planning and obstacle avoidance“, in *Springer handbook of robotics*, Springer, 2008, S. 827–852.

[11] O. Brock und O. Khatib, „High-speed navigation using the global dynamic window approach“, in *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, IEEE, Bd. 1, 1999, S. 341–346.

[12] R. Manduchi, A. Castano, A. Talukder und L. Matthies, „Obstacle detection and terrain classification for autonomous off-road navigation“, *Autonomous robots*, Jg. 18, Nr. 1, S. 81–102, 2005.

[13] I. A. Hameed, A. la Cour-Harbo und O. L. Osen, „Side-to-side 3D coverage path planning approach for agricultural robots to minimize skip/overlap areas between swaths“, *Robotics and Autonomous Systems*, Jg. 76, S. 36–45, 2016.

[14] S. Thrun und J. J. Leonard, „Simultaneous localization and mapping“, in *Springer handbook of robotics*, Springer, 2008, S. 871–889.

[15] G. Campion, G. Bastin und B. Dandrea-Novel, „Structural properties and classification of kinematic and dynamic models of wheeled mobile robots“, *IEEE transactions on robotics and automation*, Jg. 12, Nr. 1, S. 47–62, 1996.

[16] G. Campion und W. Chung, „Wheeled robots“, in *Springer handbook of robotics*, Springer, 2008, S. 391–410.

[17] H. Dodel und D. Häupler, „Die Satellitennavigation“, in *Satellitennavigation*, Springer, 2009, S. 1–43.

[18] J. Wendel, *Integrierte Navigationssysteme: Sensordatenfusion, GPS und Inertiale Navigation*. Walter de Gruyter, 2011.

[19] O. S. R. Foundation. (2014). Tutorial: ROS integration overview, Adresse: `http://gazebosim.org/tutorials?tut=ros_overview&cat=connect_ros` (besucht am 13. 03. 2018).

[20] O. S. R. Foundation, *Navigation Stack Setup*, 2016. Adresse: `http://wiki.ros.org/move_base` (besucht am 13. 03. 2018), licensed under CC BY 3.0, https://creativecommons.org/licenses/by/3.0/.

[21] u blox. (2016). CAM-M8 Datasheet, Adresse: `https://www.u-blox.com/sites/default/files/CAM-M8-FW3_DataSheet_%28UBX-15031574%29.pdf` (besucht am 13. 03. 2018).

[22] u blox. (2017). Evaluation-Kit für CAM-M8 GNSS-Antennenmodule, Adresse: `https://www.u-blox.com/de/product/evk-m8xcam` (besucht am 13. 03. 2018).

[23] T. Invensense. (2018). MPU-9250 Nine-Axis (Gyro + Accelerometer + Compass) MEMS MotionTracking? Device, Adresse: `https://www.invensense.com/products/motion-tracking/9-axis/mpu-9250/` (besucht am 13. 03. 2018).

[24] J Metge, R Mégret, A Giremus, Y Berthoumieu und T Décamps, „Calibration of an inertial-magnetic measurement unit without external equipment, in the presence of dynamic magnetic disturbances“, *Measurement Science and Technology*, Jg. 25, Nr. 12, S. 125 106, 2014.

[25] J. Fang, H. Sun, J. Cao, X. Zhang und Y. Tao, „A novel calibration method of magnetic compass based on ellipsoid fitting“, *IEEE Transactions on Instrumentation and Measurement*, Jg. 60, Nr. 6, S. 2053–2061, 2011.

[26] E. M. Hemerly und F. A. Coelho, „Explicit solution for magnetometer calibration“, *IEEE Transactions on Instrumentation and Measurement*, Jg. 63, Nr. 8, S. 2093–2095, 2014.

[27] S Bonnet, C Bassompierre, C Godin, S Lesecq und A Barraud, „Calibration methods for inertial and magnetic sensors“, *Sensors and Actuators A: Physical*, Jg. 156, Nr. 2, S. 302–311, 2009.

[28] M. J. Caruso, „Applications of magnetic sensors for low cost compass systems“, in *Position Location and Navigation Symposium, IEEE 2000*, IEEE, 2000, S. 177–184.

[29] J. F. Vasconcelos, G Elkaim, C. Silvestre, P Oliveira und B. Cardeira, „Geometric approach to strapdown magnetometer calibration in sensor frame“, *IEEE Transactions on Aerospace and Electronic systems*, Jg. 47, Nr. 2, S. 1293–1306, 2011.

[30] A. K. M. Corporation. (2013). AK8963, 3-axis Electronic Compass, Adresse: `https://www.akm.com/akm/en/file/datasheet/AK8963C.pdf` (besucht am 13. 03. 2018).

[31] M. Nowicki, M. Kachniarz und R. Szewczyk, „Temperature error of Hall-effect and magnetoresistive commercial magnetometers“, *Archives of Electrical Engineering*, Jg. 66, Nr. 3, S. 625–630, 2017.

[32] Z. für Meteorologie und Geodynamik (ZAMG). (2018). IGRF Deklinationsrechner, Adresse: `http://www.zamg.ac.at/cms/de/geophysik/produkte-und-services-1/online-deklinationsrechner` (besucht am 13. 03. 2018).

[33] H. G. De Marina, F. J. Pereda, J. M. Giron-Sierra und F. Espinosa, „UAV attitude estimation using unscented Kalman filter and TRIAD“, *IEEE Transactions on Industrial Electronics*, Jg. 59, Nr. 11, S. 4465–4474, 2012.

[34] R. G. Valenti, I. Dryanovski und J. Xiao, „A linear Kalman filter for MARG orientation estimation using the algebraic quaternion algorithm“, *IEEE Transactions on Instrumentation and Measurement*, Jg. 65, Nr. 2, S. 467–481, 2016.

[35] G. Troni und L. L. Whitcomb, „Experimental evaluation of a MEMS inertial measurements unit for Doppler navigation of underwater vehicles“, in *Oceans, 2012*, IEEE, 2012, S. 1–7.

[36] W. R. Hamilton, „LXXVIII. On quaternions; or on a new system of imaginaries in Algebra: To the editors of the Philosophical Magazine and Journal“, *Philosophical Magazine Series 3*, Jg. 25, Nr. 169, S. 489–495, 1844.

[37] S. L. Altmann, „Hamilton, Rodrigues, and the quaternion scandal“, *Mathematics Magazine*, Jg. 62, Nr. 5, S. 291–308, 1989.

[38] J. Diebel, „Representing attitude: Euler angles, unit quaternions, and rotation vectors“, *Matrix*, Jg. 58, Nr. 15-16, S. 1–35, 2006.

[39] D. Choukroun, I. Y. Bar-Itzhack und Y. Oshman, „Novel quaternion Kalman filter“, *IEEE Transactions on Aerospace and Electronic Systems*, Jg. 42, Nr. 1, S. 174–190, 2006.

[40] S. Särkkä, V. Tolvanen, J. Kannala und E. Rahtu, „Adaptive Kalman filtering and smoothing for gravitation tracking in mobile systems“, in *Indoor Positioning and Indoor Navigation (IPIN), 2015 International Conference on*, IEEE, 2015, S. 1–7.

[41] A. Makni, H. Fourati und A. Y. Kibangou, „Adaptive kalman filter for MEMS-IMU based attitude estimation under external acceleration and parsimonious use of gyroscopes“, in *Control Conference (ECC), 2014 European*, IEEE, 2014, S. 1379–1384.

[42] J. L. Marins, X. Yun, E. R. Bachmann, R. B. McGhee und M. J. Zyda, „An extended Kalman filter for quaternion-based orientation estimation using MARG sensors“, in *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, IEEE, Bd. 4, 2001, S. 2003–2011.

[43] H. Durrant-Whyte und T. C. Henderson, „Multisensor data fusion“, in *Springer handbook of robotics*, Springer, 2008, S. 585–610.

[44] S. O. Madgwick, A. J. Harrison und R. Vaidyanathan, „Estimation of IMU and MARG orientation using a gradient descent algorithm“, in *Rehabilitation Robotics (ICORR), 2011 IEEE International Conference on*, IEEE, 2011, S. 1–7.

[45] G. Wahba, „A least squares estimate of satellite attitude“, *SIAM review*, Jg. 7, Nr. 3, S. 409–409, 1965.

[46] H. D. Black, „A passive system for determining the attitude of a satellite“, *AIAA journal*, Jg. 2, Nr. 7, S. 1350–1351, 1964.

[47] M. D. Shuster und S. D. Oh, „Three-axis attitude determination from vector observations“, *Journal of Guidance, Control, and Dynamics*, Jg. 4, Nr. 1, S. 70–77, 1981.

[48] F. L. Markley, „Attitude determination using vector observations: A fast optimal matrix algorithm“, 1993.

[49] N. Trawny und S. I. Roumeliotis, „Indirect Kalman filter for 3D attitude estimation“, *University of Minnesota, Dept. of Comp. Sci. & Eng., Tech. Rep*, Jg. 2, S. 2005, 2005.

[50] P. S. Maybeck, *Stochastic models, estimation, and control*. Academic press, 1982, Bd. 3.

[51] H. Ziezold, *Situative Karten: Orientierung, Exploration und Navigation mit spezialisierbaren Karten*. Springer-Verlag, 2013.

[52] T. Moore und D. Stouch, „A generalized extended kalman filter implementation for the robot operating system“, in *Intelligent Autonomous Systems 13*, Springer, 2016, S. 335–348.

[53] W. Meeussen. (2012). robot_pose_ekf, Adresse: `http://wiki.ros.org/robot_pose_ekf` (besucht am 13. 03. 2018).

[54] M. Hennessey. (2013). robot_pose_ekf, Adresse: `https://www.clearpathrobotics.com/2013/07/blog_gps-using-pose_ekf/` (besucht am 13. 03. 2018).

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass die vorliegende Arbeit nach den anerkannten Grundsätzen für wissenschaftliche Abhandlungen von mir selbstständig erstellt wurde. Alle verwendeten Hilfsmittel, insbesondere die zugrunde gelegte Literatur, sind in dieser Arbeit genannt und aufgelistet. Die aus den Quellen wörtlich entnommenen Stellen, sind als solche kenntlich gemacht.
Das Thema dieser Arbeit wurde von mir bisher weder im In- noch Ausland einer Beurteilerin/einem Beurteiler zur Begutachtung in irgendeiner Form als Prüfungsarbeit vorgelegt. Diese Arbeit stimmt mit der von den Begutachterinnen/Begutachtern beurteilten Arbeit überein.

Wien, im August 2018

Unterschrift