

**Friedrich-Alexander-Universität Erlangen-Nürnberg**



**Lehrstuhl für Informationstechnik  
mit dem Schwerpunkt Kommunikationselektronik**



Professor Dr.-Ing. Jörn Thielecke

**Diplomarbeit**

**Thema:**

Horizontale Geschwindigkeitsregelung eines Quadrocopter mit Hilfe von  
Laserdaten

Bearbeiter: B.Eng. Matthias Welter

Betreuer: Dipl.-Inf. Manuel Stahl  
Dipl.-Ing. Christian Strobel

Beginn: 01. August 2014

Ende: 31. Januar 2015

---

## Bestätigung

---

Erklärung:

Ich versichere, dass ich die Arbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe und, dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen wurde. Alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, sind als solche gekennzeichnet.

Erlangen, den 31.01.2015

---

Matthias Welter

---

## Thema und Aufgabenstellung

---

**Thema:**

Horizontale Geschwindigkeitsregelung eines Quadrocopter mit Hilfe von Laserdaten

**Aufgabenstellung:**

Um das manuelle sowie automatisierte Navigieren eines Quadrocopters in der horizontalen Ebene zu vereinfachen ist es von Vorteil, die Bewegung ausschließlich in Form von Geschwindigkeiten in x- und y-Richtung vorzugeben. Manuell soll die Vorgabe über die Fernsteuerung erfolgen. Für das automatisierte Navigieren ist eine Schnittstelle zum Übergeben der Sollwerte vorzusehen. Die Geschwindigkeit ist anhand der vom Laserscanner erfassten Daten zu ermitteln.

***Ziel ist es eine Regelung zu entwerfen, welche die horizontale Geschwindigkeit des Quadrocopters auf den Sollwert einregelt.***

Optional kann eine automatisierte relative Positionsverschiebung des Quadrocopters implementiert werden.

Die Arbeitsschritte sind:

- Literaturrecherche
- Auswahl und Integration einer geeigneten Methode zur Bestimmung der relativen Position aus den Laserdaten
- Bestimmung der Geschwindigkeit in der x-y-Ebene
- Entwurf und Implementierung einer Geschwindigkeitsregelung
- Optional: Integration einer automatisierten relativen Positionsverschiebung

**Klassifikation:**

Robotik, Regelungstechnik, Informatik, Elektrotechnik, Sensorik

---

## Kurzzusammenfassung

---

Ziel aktueller Forschungen ist die teilweise bzw. vollständig autonome Fortbewegung von Quadrocoptern. Diese Arbeit befasst sich mit der Positions- und Geschwindigkeitsregelung eines solchen Fluggerätes, basierend auf Entfernungsmessungen, die mittels eines auf dem Quadrocopter befindlichen Laserscanners aufgenommen werden. Zur Lokalisierung und Regelung werden bereits entwickelte Funktionsbausteine zu einem funktionierenden Gesamtsystem zusammengefügt. So ist der Schwerpunkt darauf ausgelegt, die Theorie und die Konzepte dieser Bausteine offenzulegen und nachzuweisen.

Dabei wird mit der Lokalisierung. Dort ist gezeigt wie sich trotz eines ständig wechselnden Neigungswinkel des Quadrocopter, anhand der Messungdaten des Lasers die Position in der Ebene einer unbekannten Umgebung bestimmen lässt. Anschließend fokussierte sich die Arbeit auf die Verifizierung der implementierten Regelung. Zunächst wird das kinematische Modell hergeleitet. Darauf aufbauend ist die Korrektheit der von den Entwicklern der Positions- und Geschwindigkeitsregelung veröffentlichten Formel und Annahmen für die Bestandteile der Regelung durch die mathematische Herleitung, Teilsimulationen sowie Literaturverweise dargelegt.

Zu guter Letzt wird die Funktionsfähigkeit der Regelung in Verbindung mit der über den Laser bestimmten Positionsdaten mittels Flugversuchen, sowohl für die Positionsregelung als auch für die Geschwindigkeitsregelung, unter Beweis gestellt.

---

## Abstract

---

*Die englische Version der Kurzzusammenfassung. Für die Länge gelten die Gleichen Vorgaben wie für die deutsche Version.*

---

## Vorwort

---

*Hier können allgemeine Hinweise zur Arbeit gegeben werden, bspw. wie man mit englischen Begriffen, Abkürzungen und Codeabschnitten umgeht. Der nachfolgende Text kann als Beispiel gesehen werden, ist aber keinesfalls verpflichtend und sollte der eigenen Konvention angepasst werden!*

---

Diese Arbeit behandelt ein aktuelles technisches Thema, die Verwendung von englischen Begriffen ist unumgänglich. Soweit sinnvoll findet eine deutsche Übersetzung Verwendung. Nicht übersetzbare Begriffe, die eine wichtige Bedeutung für diese Arbeit haben, werden in einer Fußnote erklärt. Ausdrücke und Bezeichnungen aus Standards werden allgemein nicht übersetzt. Englische Begriffe sind im Text kursiv geschrieben. Wörter, die im deutschen Sprachgebrauch alltägliche Anwendung finden, wie beispielsweise Computer, Software, Internet etc., sind nicht kursiv geschrieben.

Bei erstmaliger Verwendung von Abkürzungen wird die volle Bezeichnung ausgeschrieben und das Kürzel dahinter in Klammern gesetzt. In der Folge wird nur die Abkürzung benutzt.

Quelltexte von Programmen sowie programmiertechnische Bezeichnungen und Schlüsselwörter werden durch die Verwendung von Schreibmaschinenschrift hervorgehoben.

Am Anfang der Arbeit befindet sich ein Abkürzungsverzeichnis, in dem alle in dieser Arbeit genannten Abkürzungen und deren ausgeschriebenen Formen enthalten sind. Im Anschluss an den Ausblick werden die wichtigsten Begriffe im Glossar zusätzlich kurz erläutert.

---

## Abkürzungsverzeichnis

---

AscTec	Ascending Technologies
ENU	East-North-Up
FCU	Flight Control Unit
FOAW	First-Order Adaptive Windowing
HLP	High Level Processor
I <sup>2</sup> C	Inter Integrated Circuit
ICP	Iterative Closest Point
IIR	Infinite Impulse Response
IMU	Inertial Measurement Unit
IP	Internet Protocol
LLP	Low Level Processor
NED	North-East-Down
PC	Personal Computer
ROS	Robot Operation System
SPI	Serial Peripheral Interface

UART Universal Asynchronous Receiver/Transmitter

VSLAM Visual Simultaneous Localization and Mapping

---

## Glossar

---

### *Asynchronous Receiver/Transmitter*

Eine gängige serielle Schnittstelle zum Senden und Empfangen von Daten über eine Datenleitung. Bildet den Standard der seriellen Schnittstellen an *Personal Computer (PC)s* und Mikrocontrollern.

### *Inter Integrated Circuit*

Von Philips Semiconductor entwickelter serielles Bussystem. Hauptsächlich zur gerätiinternen Kommunikation zwischen verschiedenen Schaltungsteilen eingesetzt.

### *Internet Protocol*

Ist das am weitverbreitete Netzwerkprotokoll in Computernetzen und stellt die Grundlage des Internets dar

### *Serial Peripheral Interface*

Ein von Motorola entwickelter Standard eines synchronen seriellen Datenbus zur Vernetzung zweier digitaler Schaltungen nach dem Master-Slave-Prinzip

### *Visual Simultaneous Localization and Mapping*

SLAM ist eine Methode der mobilen Robotik zur Schätzung der eigenen Position in einer unbekannten Umgebung. Dafür wird eine Karte des Raumes aus den Messdaten erstellt. Der Zusatz V sagt dabei aus, dass diese Messdaten von einem visuellen Sensor, sprich einer Kamera zur Verfügung gestellt werden. Im deutschen steht SLAM für Simultane Lokalisierung und Kartenerstellung.

---

# Inhaltsverzeichnis

---

Abkürzungsverzeichnis	i
Glossar	iii
1 Einleitung	1
1.1 Ausgangssituation . . . . .	2
1.2 Aufbau der Arbeit . . . . .	3
2 Systemarchitektur des Quadrocopters	5
2.1 Grundlegende Funktionsweise eines Quadrocopters . . . . .	5
2.2 Hardwareaufbau . . . . .	7
2.3 Softwarestruktur . . . . .	9
3 Grundlagen	12
3.1 Das Robot Operation System ROS . . . . .	12
3.2 Einführung in die Koordinatensysteme und Koordinatentransformationen . .	14
3.2.1 Koordinatensysteme . . . . .	14
3.2.2 Koordinatentransformationen . . . . .	17
4 2D Positionsbestimmung	21
4.1 Orthogonale Laserprojektion . . . . .	22
4.2 Positionsbestimmung über Scanmatching . . . . .	25
5 Positionsregelung	32
5.1 Struktur der Positionsregelung . . . . .	33
5.2 Modellbildung . . . . .	36
5.3 Exakte Zustandslinearisierung . . . . .	40

5.4	Vorsteuerung/Referenzmodell . . . . .	43
5.5	Folgeregler . . . . .	49
5.5.1	Einstellung der Dynamik mittels Polvorgabe . . . . .	52
5.5.2	Automatische Optimierung der Reglerparameter . . . . .	55
5.6	Zustandsschätzung . . . . .	56
5.6.1	Geschwindigkeitsbestimmung über die Inertialsenorik . . . . .	57
5.6.2	Geschwindigkeit als Ableitung der Position . . . . .	58
5.6.3	Geschwindigkeitsbestimmung über die Methode First-Order Adaptive Windowing . . . . .	60
5.6.4	Bestimmung der Zustände mittels eines Fusionsfilter . . . . .	65
5.7	Simulation der Positionsreglung . . . . .	71
6	Flugversuche	75
6.1	Anflug einer Koordinate im Raum . . . . .	75
6.2	Geschwindigkeitsregelung . . . . .	79
7	Zusammenfassung und Ausblick	83
7.1	Zusammenfassung . . . . .	83
7.2	Ausblick . . . . .	84
8	2D Positionsbestimmung	85
8.1	Orthogonale Laserprojektion . . . . .	86
8.2	Positonsbestimmung über Scanmatching . . . . .	89
9	Grundlagen	97
9.1	Das Robot Operation System <i>Robot Operation System (ROS)</i> . . . . .	97
9.2	Einführung in die Koordinatensysteme und Koordinatentransformationen . . . . .	99
9.2.1	Koordinatensysteme . . . . .	99
9.2.2	Koordinatentransformationen . . . . .	102
	Literaturverzeichnis	106
	Abbildungsverzeichnis	108
	Tabellenverzeichnis	110

A Datenblatt Hokuyo Laserscanner	111
B Definition flacher Mehrgrößensysteme	118
C Simulinkmodell	121
C.1 Gesamtübersicht Positionsregelung . . . . .	122
C.2 Translationsmodell . . . . .	123
C.3 Modell der Inversion . . . . .	124
C.4 Modell der Vorsteuerung . . . . .	125
C.5 Modell der Folgeregelung . . . . .	125
C.6 Modell des Fusionsfilters . . . . .	126

# KAPITEL 1

---

## Einleitung

---

Quadrocopter gehören zur Kategorie der Hubschrauber, da ihre Rotoren sowohl zum Erzeugen des Auftrieb als auch für den Vortrieb genutzt werden. Im Gegensatz zu einrotorigen Maschinen benötigen sie keine aufwendigen mechanischen Komponenten, wie eine Taulmelscheibe, um die Schubkraft sowie die Neigung des Fluggerätes zu variieren. Beim Quadrocopter werden diese Funktionen durch die vier im quadratischen Eckpunkt befindlichen Rotoren mittels individueller Drehzahlsteuerung realisiert. Der mechanische Aufbau ist somit deutlich primitiver. Um dieses Flugsystem stabil zu fliegen, ist der Regelungstechnische Aufwand auf der elektrisch motorischen Steuerungsseite der Rotoren erheblich und sehr herausfordernd. Dank der technischen Fortschritte im Bereich der Mikrosystemtechnik als auch der Prozessoren stehen heutzutage die benötigten Technologien bereit. Kompakte sowie kostengünstige *Microelectromechanical systems (MEMS)*-Accelerometer und -Gyroskope ermöglichen es in Verbindung mit leistungsfähigen Recheneinheiten eine Steuereinheit für Quadrocopter zu entwickeln. Ein Grund warum sich der Quadrocopter in der Kategorie der unbemannten Fluggeräte weiter wachsender Beliebtheit erfreut. Anwendung finden sie unter anderem beim Filmen, ausgerüstet mit einem Kamerasystem liefern sie spektakuläre Luftaufnahmen. Im weiteren humanitären Einsatz dienen sie zur Aufklärung in der Katastrophenhilfe, so kann zum Beispiel ohne Gefährdung von Rettern ein einsturzgefährdetes Gebäude untersucht werden.

Ziel vieler Forschungen ist es, dass der Quadrocopter solche Flüge zunehmend autonom durchführt. So fokussiert sich diese Masterthesis auf das automatisierte Anfliegen von Koordinaten in einem unbekannten Raum. Die Bestimmung der Position erfolgt dabei mittels einen Laserscanners, der auf dem Fluggerät montiert ist. Auch kann eine solche Positions-

reglung der Geschwindigkeitsregelung dienen, was die Steuerung über die Fernbedienung vereinfacht.

Eingesetzt werden soll der automatisierte Quadrocopter später am Fraunhofer-Institut für Integrierte Schaltungen als bewegliches Referenzsystem für die dort entwickelten Lokalisierungssysteme mittels *Radio-Frequency IDentification (RFID)* oder *Wireless Local Area Network (WLAN)*. Durch die Automatisierung lassen sich so dynamische Bewegungsabläufe zuverlässig reproduzieren, womit jederzeit vergleichbare Messungen in einem Raum durchgeführt werden können.

### 1.1 Ausgangssituation

Zur Lokalisierung ist auf dem Quadrocopter ein Laserscanner angebracht. Die von ihm aufgenommen Entfernungsmessungen beziehen sich auf die Sendequelle des Lasers. So enthalten die Rohdaten keine Informationen über die absolute Position. Es existieren jedoch Methoden, die es ermöglichen den Quadrocopter mittels solcher Umgebungscans in einer unbekannten Umgebung zu lokalisieren. Zu beachten sind hierbei die verschiedenen Neigungswinkel des Quadrocopters zur Fortbewegung. Das bedeutet, die Ausrichtung des Lasers entspricht nicht zwangsläufig der horizontalen Ebene des Raums. Diesen Einfluss gilt es bei Integration einer 2D-Positionsbestimmung in unbekannten Räumen zu berücksichtigen.

Bei den Literaturrecherchen zur Geschwindigkeits- und Positionsregelung stellte sich heraus, dass für den verwendeten Quadrocopter Pelican der Firma *Ascending Technologies (AscTec)* ein frei verfügbares Framework existiert, welches die wesentlichen Punkte der Aufgabenstellung (siehe „Thema und Aufgabenstellung“) beinhaltet. So enthält es eine Positionsregelung, welche ebenfalls als Geschwindigkeitsregelung genutzt werden kann. Des Weiteren verfügt es über einen Fusionsfilter zur Interpolation niederfrequenter Positionsdaten, in dem diese mit den Beschleunigungswerten vereinigt werden. Der Aufbau dieses Frameworks ist dabei im Paper [3] beschrieben. Hier ist auch die Funktionsfähigkeit der Algorithmen in Verbindung mit einem visuellen Positionierungssystem dargelegt.

In Absprache mit den Betreuern dieser Masterthesis ist deshalb der Schwerpunkt weg von der Entwicklung einer Geschwindigkeitsregelung, hin zur Verifizierung der existierenden Positionsregelung verschoben worden. Das bedeutet, dass das Ziel darin liegt, die im Paper [3] beschriebenen Annahmen und Formeln durch ihre Herleitungen zu bestätigen. Außerdem

gilt es die Frage zu klären, ob die Regelung mit den über den Laserscanner ermittelten Positionsdaten zurecht kommt und wie sie dafür zu parametrieren ist.

## 1.2 Aufbau der Arbeit

Mit den wissenschaftlichen Schwerpunkten, die Theorie hinter der Positionsbestimmung mittels eines Laserscanners aufzuzeigen sowie das Konzept als auch die Funktionsfähigkeit der bereits entwickelten Positionsregelung offenzulegen, ist die Masterthesis wie folgt aufgebaut.

Zu Beginn wird im Rahmen von Kapitel 2 die grundlegende Funktionsweise eines Quadrocopters dargelegt. Mit dem Zweck ein Verständnis zu generieren, wie durch gezieltes Ansteuern der vier Rotoren eine Bewegung des Quadrocopters hervorgerufen werden kann. Anschließend wird die im Rahmen dieser Arbeit zum Einsatz kommende Hardware genauer spezifiziert. Dabei wird erläutert, welche Sensoren und Recheneinheiten zur Anwendung kommen und wo diese verbaut sind. Abgeschlossen wird das Kapitel 2 mit der Darlegung der Kommunikationsarchitektur, das aufzeigt wie die Komponenten untereinander vernetzt sind und über welche Schnittstellen sie verfügen und kommunizieren.

Kapitel 9 ist das Grundlagenkapitel. Hier werden Themen behandelt, die in mehreren Abschnitten der Arbeit relevant sind. Begonnen wird mit dem *ROS*. Dabei werden der Aufbau und die Vorteile einer Verwendung dieses Betriebssystems aufgezeigt. Gefolgt wird dieser Abschnitt von einem für das Verständnis der Arbeit essenziellen Teilbereich, den Koordinatensystemen. Hier werden alle benötigten Koordinatensysteme beschrieben. Wo sie sich befinden, welchen zwecks sie erfüllen und wie sie miteinander verknüpft sind. Auf diese Weise wird zum Ende des Kapitels 9 erläutert, wie Werte aus einem Koordinatensystem in ein anderes transformiert werden können.

Aufbauend auf diesen Kapiteln folgt in Kapitel 8 die Behandlung des ersten Kernpunktes dieser Arbeit. Die relative Positionsbestimmung mittels eines Laserscanners. Dabei wird aufgezeigt, welche Bausteine zur Ermittlung der Position, die unter *ROS* frei zur Verfügung stehen, Anwendung finden. Um ein Verständnis zu erlangen wie diese funktionieren, wird die Mathematik der Algorithmen die bei der Positionsbestimmung zum Einsatz kommen hergeleitet.

Nach dem veranschaulicht ist, wie die Lokalisierung mittels einen Lasersanners abläuft, wird in Kapitel 5 die Validierung des Frameworks zur Positionsregelung in Angriff genommen.

Hierfür wird zunächst das der Regelung zugrunde liegende kinematische Modell aufgestellt. Da es sich um ein nichtlineares Modell handelt, wird daraufhin die Richtigkeit der angewandten Inversion zur Linearisierung des Systems nachgewiesen. Da es sich bei der zu validierenden Positionsregelung um eine Zwei-Freiheitsgrade-Regelung handelt, folgt anschließend die Untersuchung der Vorsteuerung. Bevor die Korrektheit der Folgeregelung beweisbar ist, wird dargelegt über welche Methode sich diese einstellen lässt. Da es sich bei dem Folgeregler um einen Zustandsregler handelt, wird zu der Position noch die Geschwindigkeit als Regelgröße benötigt. Im letzten Abschnitt wird das Fusionsfilter untersucht, als auch evaluiert, ob weitere Möglichkeiten zur Bestimmung der Geschwindigkeit existieren.

Zum Schluss erfolgt in Kapitel 6 die Auswertung eines praktischen Flugversuches, der die Funktionsfähigkeit des Reglers auch für eine Positionsbestimmung mittels Laser unter Beweis stellt.

Kapitel 7 dient schlussendlich dazu die Ergebnisse der Masterthesis zusammenzufassen und einen Ausblick zu geben, wie diese Arbeit weitergeführt werden kann.

# KAPITEL 2

---

## Systemarchitektur des Quadrocopters

---

Zu Beginn wird in diesem Kapitel die grundlegende Funktionsweise des Quadrocopters erläutert. Anschließend wird die bei dieser Arbeit zum Einsatz kommende Hardware und die Verknüpfung der einzelnen Komponenten miteinander dargelegt. Dabei geht es darum, aufzuzeigen an welchen Stellen Software bereits fest implementiert ist und wo eigene Algorithmen integriert werden können.

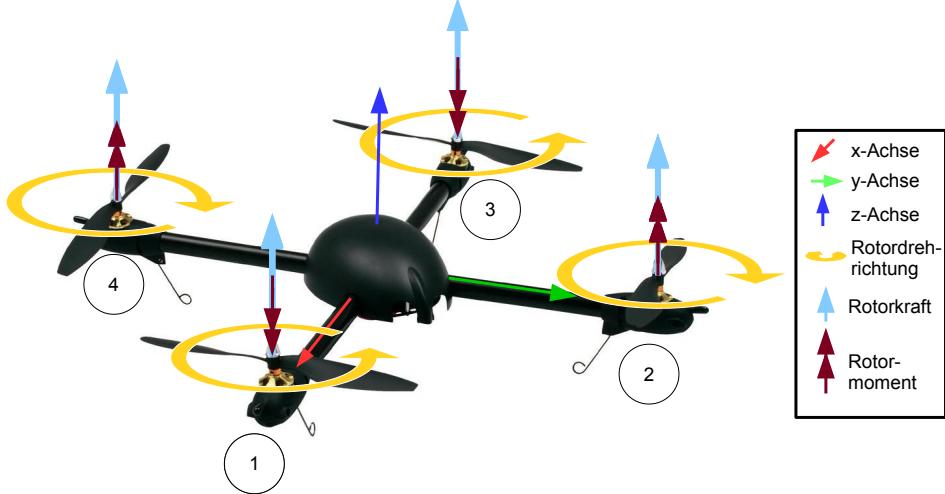
### 2.1 Grundlegende Funktionsweise eines Quadrocopters

Ziel dieses Kapitel ist es, ein Verständnis dafür zu erhalten, wie über die gezielte Ansteuerungen der vier Rotoren eine Bewegung des Quadrocopters hervorgerufen werden kann. Dabei wird auf die wirkenden Kräfte und Momente eingegangen, ohne tief in die Physik einzusteigen.

Der Schub der Rotoren ist Anhand der Drehzahl  $n_i$  der Rotorblätter individuell einstellbar. Somit lassen sich die Kräfte  $F_i$ , die an den Endpunkten der Quadrocopterarme der Länge  $l$  wirken, vorgeben. Die Gesamtkraft aller Rotoren ergibt den Schubvektor  $S^b$ .

$$S^b = \begin{bmatrix} S_x^b \\ S_y^b \\ S_z^b \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ F_1 + F_2 + F_3 + F_4 \end{bmatrix} \quad (2.1)$$

Damit der Qudrocopter eine Bewegung im Raum vollziehen kann, muss dieser Vektor aus der Vertikalen ausgelenkt werden. Dies wird durch eine Änderung der Lage realisiert. Reduziert man zum Beispiel die Drehzahl  $n_1$  und erhöht gleichzeitig die Drehzahl  $n_3$  hat das



**Abbildung 2.1:** Momente und Kräfte an einem Quadrocopter

resultierende Kräfteungleichgewicht ein positives Moment um die  $y^b$ -Achse zur Folge. Der Quadrocopter dreht sich um die  $y^b$ -Achse, der Pitch-Winkel ändert sich. Der Quadrocopter erfährt in der horizontalen Ebene des Raums eine Beschleunigung. Das gleiche Prinzip gilt auch für den Roll-Winkel, sprich Rotation um die  $x^b$ -Achse. Hier ist allerdings der Drehzahldifferenz zwischen  $n_2$  und  $n_4$  verantwortlich für die Rotation.

Eine Änderung der Orientierung um die Hochachse  $z$ , sprich Änderung des Yaw-Winkels, lässt sich ebenfalls über Variation der Rotordrehzahlen hervorrufen. Dabei kommt der Effekt zum tragen, dass der Widerstand der umgebende Luft entgegen der Drehrichtung der Motoren eine Kraft auf die Rotorblätter einprägt. Je nach Drehrichtung der Rotoren wirkt damit ein Moment auf den Quadrocopter. Diese Momente, die an den Armen des Quadrocopters angreifen, lassen sich zur Vereinfachung in den Schwerpunkt verschieben. Damit bei gleicher Drehzahl aller Rotorblätter ein Momentengleichgewicht herrscht, drehen sich die Motoren eins und drei gegen, die Motoren zwei und vier mit dem Uhrzeigersinn. Um die gewünschte Rotation zu erzielen, wird die Drehzahl  $n_1$  und  $n_3$  erhöht und gleichzeitig  $n_2$  und  $n_4$  reduziert. Das Ergebnis ist eine Rotation in positive Richtung.

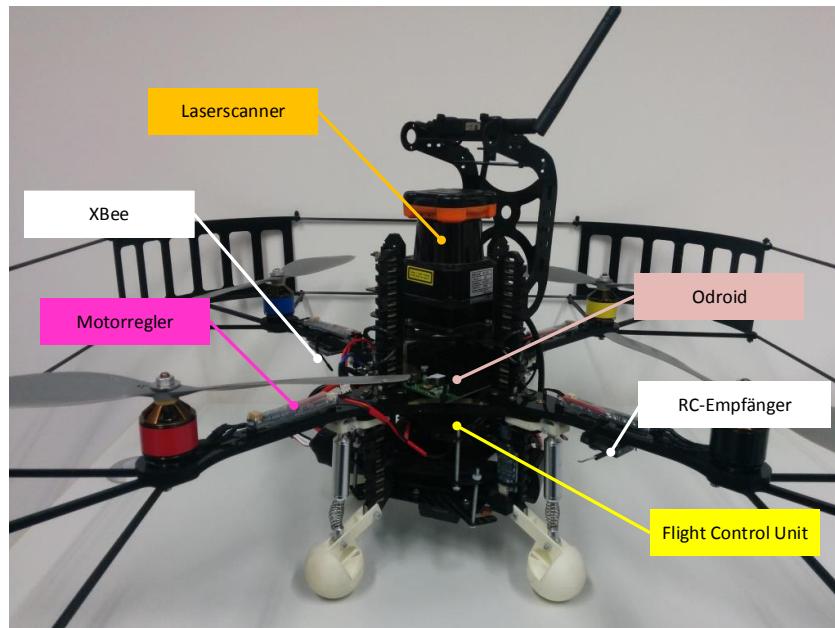
Zusammenfassen lassen sich die für die Rotation um die Quadrocopter-Achsen verantwortlichen Momente  $M_{x,y,z}^b$  in einem Vektor  $M^b$ .

$$M^b = \begin{bmatrix} M_x^b \\ M_y^b \\ M_z^b \end{bmatrix} = \begin{bmatrix} l(F_3 - F_1) \\ l(F_2 - F_4) \\ M_1 - M_2 + M_3 - M_4 \end{bmatrix} \quad (2.2)$$

Aufzuklären ist, warum mit einer Erhöhung der Drehzahl auch immer eine Reduzierung des Gegenparts verknüpft ist. Die Begründung lautet, dass der Schubvektor  $S^b$  durch eine Rotation möglichst wenig beeinflusst werden soll.

## 2.2 Hardwareaufbau

Zum Einsatz kommt der AscTec Pelican der Firma *AscTec* [2]. Dieser Quadrocopter ist eine spezielle Entwicklung für die Forschung. Seine Turmstruktur ermöglicht eine einfache Integration zusätzlicher Sensoren und Nutzlasten. Durch die Flexibilität im Aufbau ist das Ziel dieses Teilkapitels, einen Überblick zur Position der einzelnen Komponenten zu geben. Begleitend zum Text ist der Aufbau in Abbildung 2.2 dargestellt.



**Abbildung 2.2:** Hardwareaufbau des Quadrocopter

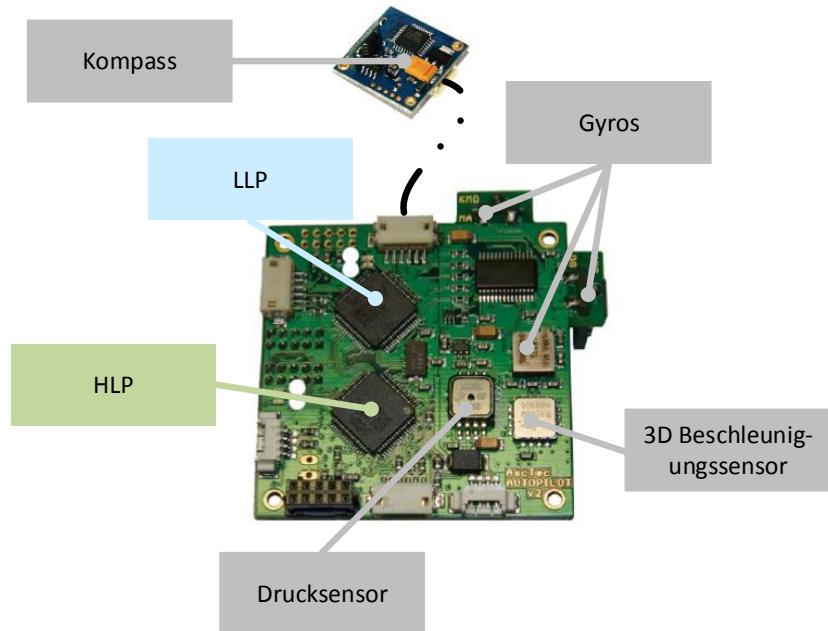
Für jeden der vier mit einem Propeller verbundenen Elektromotoren sind separate Motorcontroller zuständig. Diese sorgen dafür, dass sich die von der *Flight Control Unit (FCU)* angeforderten Drehzahlen einstellen.

Die *FCU* ist die zentrale Steuer- und Regeleinheit des Quadrocopters. Sie besitzt zwei ARM7 Prozessoren, einen *Low Level Processor (LLP)* und einen *High Level Processor (HLP)*, zudem verschiedene Kommunikationsschnittstellen (vgl. Kapitel 2.4). Zusätzlich besitzt *FCU* eine inertiale Messeinheit (engl. *Inertial Measurement Unit (IMU)*). Diese Einheit wird zur Bewegungsdetektion sowie zur Bestimmung der Lage und Ausrichtung benötigt. Sie ist nicht zur Positionsbestimmung in einem ortsfesten Koordinatensystem geeignet. Bestandteile der *IMU* sind ein 3D-Beschleunigungssensor, drei Drehratensensoren(Gyros), ein Kompass sowie ein Drucksensor zur Ermittlung der Flughöhe anhand des Luftdrucks. Verbaut sind die Sensoren mit Ausnahme des Kompass direkt auf der Platine (siehe Abbildung 2.3).

Da der Einsatzbereich im Indoorbereich liegt, ist der Drucksensor zur Höhenbestimmung in geschlossenen Räumen nicht geeignet. Er liefert erst ab einer Höhe von 5m zuverlässige Werte. Daher wurde in einer vorangegangen Arbeit von Jan Kallwies [11] die Hardware um ein Modul zur Messung der Höhe im Indoorbereich erweitert. Auf diesem Modul befinden sich zwei Infrarotsensoren für den Nahbereich. Beide zusammen decken einen Bereich von 4 cm bis 142 cm ab. Erweitert wird der Messbereich durch einen Ultraschallsensor für Entfernungsmessungen von bis zu 5 m. Aus diesen drei Sensordaten wird über einen Extended-Kalman-Filter die Flughöhe bestimmt. Eine genaue Beschreibung dieses Fusionsfilters kann in der Arbeit von Jan Kallwies [11] nachgelesen werden. Da in dieser Arbeit die Navigation in der horizontale Ebene den Schwerpunkt darstellt, wird dieses Modul in nicht weiter behandelt.

Um in der Horizontalen die Navigation zu gewährleisten, muss die Position des Flugkörpers in der xy-Ebene bekannt sein. Da dies, wie schon beschrieben, nicht mit der Inertialsenorik möglich ist, wurde in die Turmstruktur der Laserscanner UTM-30LX der Firma Hokuyo integriert. Dieser Scanner hat eine maximale Reichweite von 30 m und ein Abtastbereich von 270°. Die Umlauffrequenz beträgt dabei 40 Hz, d.h. alle 25 ms steht ein neuer Umgebungsscan zur Verfügung.

Damit zur Berechnung der Position sowie der Implementierung weiterer Algorithmen und Funktionen ausreichend Rechenleistung zur Verfügung steht, befindet sich auf dem Quadrocopter ein zusätzlicher Odroid-X Mikrocomputer mit einem Quad Core Prozessor mit 1.4 Ghz und 1024MB LP-DDR2 Arbeitsspeicher. Außerdem besitzt diese Entwicklungs-



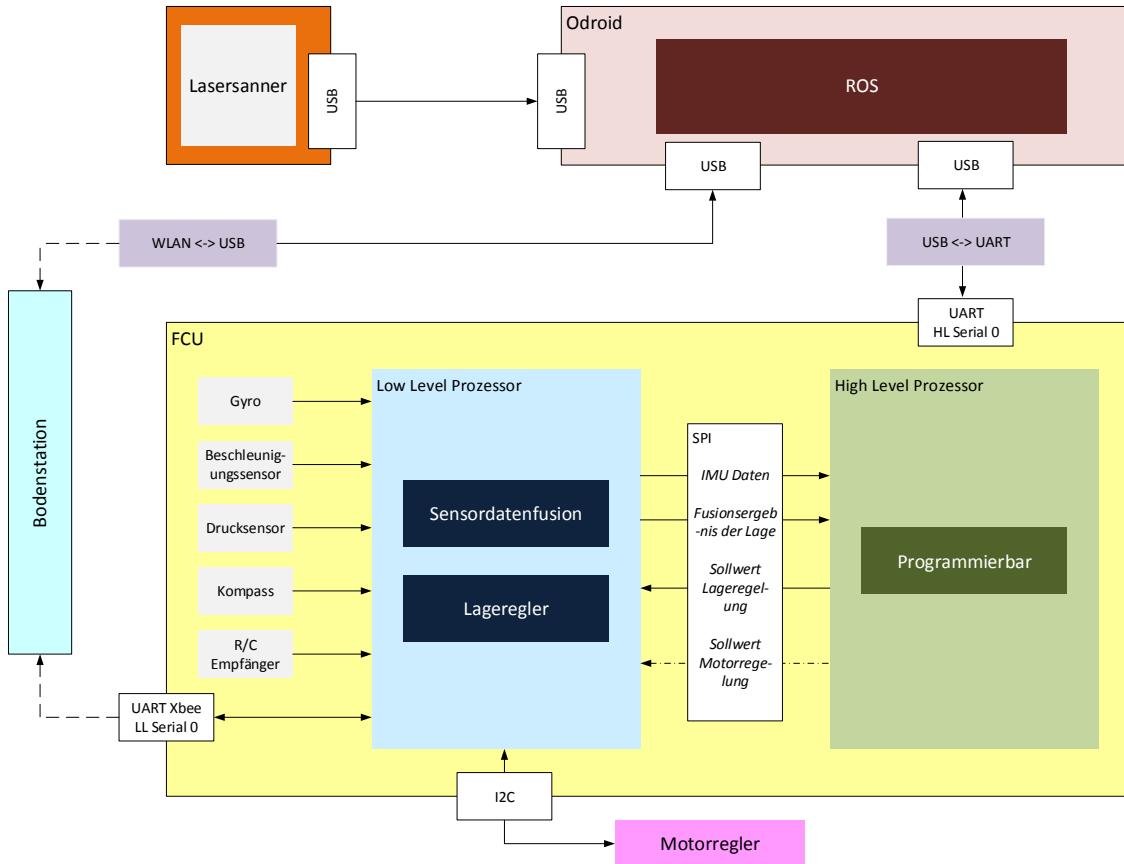
**Abbildung 2.3:** Platine der *FCU*

plattform sechs USB-Schnittstellen sowie einen 10/100Mbps Ethernet-Anschluss.

## 2.3 Softwarestruktur

Nachdem im vorhergegangenen Kapitel 2.2 die verbaute Hardware vorgestellt wurde, geht es in diesem Abschnitt um die Softwarestruktur (Abbildung 2.4). Es wird aufgezeigt, welche Software bereits fest implementiert ist und wo adaptive Applikationen integriert werden können. Des Weiteren wird die Kommunikationsstruktur dargelegt, wie und über welche Protokolle die einzelnen Komponenten miteinander kommunizieren.

Beginnend mit den beiden Prozessoren der *FCU*, deren Hauptschleifen der Software mit einer Frequenz von 1kHz durchlaufen werden und die über einen *Serial Peripheral Interface (SPI)* Bussystem verknüpft sind, wird zunächst der *LLP* betrachtet. Auf dem Low Level Prozessor befindet sich die Sensordatenfusion der *IMU*-Sensorik zur Lagebestimmung des Quadrocopters. Darauf basiert die Lageregelung, die das Flugverhalten stabilisiert. Dabei werden die geforderten Sollwinkel bzw. die Solllage, die dem *LLP* über die Fernbedienung oder den *HLP* übergeben werden, eingestellt. Kombiniert mit der Schubvorgabe werden



**Abbildung 2.4:** Kommunikationsstruktur des Quadrocopters

den Motorreglern die jeweiligen Solldrehzahlen der Rotoren über einen *Inter Integrated Circuit (I<sup>2</sup>C)*-Bus übergeben. Diese Algorithmen sind fest eingepflegt und gewährleisten bei Experimentalflügen eine sichere Rückfallebene. Mit dem *LLP* stellt *AscTec* dem Benutzer eine Art White-Box zur Verfügung, d.h. die Integration ist bekannt, jedoch nicht deren Umsetzung. Überwachen lässt sich der *LLP* über einen externen *PC*, in Abbildung 2.3 als Bodenstation bezeichnet. Zur Kommunikation werden zwei XBee Funkmodule benötigt. Eines ist am *Universal Asynchronous Receiver/Transmitter (UART)* *LL-Serial0* Port der *FCU* angeschlossen, das andere am *USB* Port der Bodenstation. Mit der AutoPilot Software lassen sich unter anderem der Akkustand, die *IMU*-Daten sowie die Stellgrößen der Fernsteuerung betrachten. Außerdem ist es möglich, Parameter der Sensorfusion und der Lageregelung auszulesen und zu verändern.

Mit dem *HLP* stellt *AscTec* eine Entwicklungsumgebung zur Implementierung eigener Algorithmen auf der *FCU* zur Verfügung. Hier können erweiternde Programmteile integriert werden, die den Lageregler des *LLP* ansprechen oder die direkt den Motorcontroller über den *LLP* mit Solldrehzahlen speisen.

Die experimentelle Software auf dem *HLP* kann über die Fernbedienung aktiviert und deaktiviert werden. Eine fehlerhafte Programmierung des *HLP* kann kritische Flugmanöver hervorrufen. Damit diese nicht zum Absturz führen, kann über die Fernbedienung die Experimentalsoftware deaktiviert und das Flugsystem über die ausgereifte Lageregelung auf dem *LLP* stabilisiert werden (Rückfallebene).

Wie schon in Kapitel 2.2 beschrieben, befindet sich auf dem Quadrocopter zur Erhöhung der Rechenleistung der Odroid-X. Anders als bei den auf der *FCU* befindlichen Prozessoren, besitzt das Odroid Bord ein Betriebssystem. Es handelt sich dabei um das Opensource Betriebssystem Ubuntu 13.04. Dieses wurde ausgewählt, da es die Installation eines weiteren opensource Betriebssystems ermöglicht, dem *ROS*, einem Software Framework für Roboteranwendungen (siehe Kapitel 9.1). Zum Einsatz kommt der Odroid-X bei der Implementierung der Positionsbestimmung (Kapitel 8). Verbunden ist es zum einen über einen USB-Port mit dem Laserscanner. Zum anderen mittels eines weiteren USB-Port über einen *Future Technology Devices International (FTDI)*-Konverter am HL-Serial0 Port des *HLP* angeschlossen. Von der Bodenstation kann über *WLAN* eine *Secure Shell (SSH)* Verbindung aufgebaut werden, die in Folge die Entwicklungsplattform bedient.

Jetzt ist bekannt, wie die einzelnen Komponenten untereinander vernetzt sind. Im weiteren Verlauf der Arbeit lässt sich nachvollziehen, an welchen Stellen die Anwendungen implementiert werden und über welche Verbindungen sie miteinander kommunizieren.

# KAPITEL 3

---

## Grundlagen

---

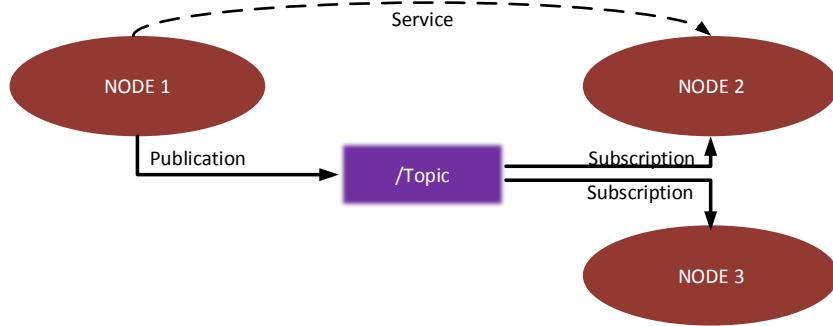
Das Kapitel Grundlagen behandelt die Themen, die in mehreren Abschnitten der Arbeit relevant sind. Dabei handelt es sich um das Robot Operation System, die verwendeten Koordinatensysteme und die Transformation zwischen ihnen.

### 3.1 Das Robot Operation System ROS

Ziel dieses Unterkapitel ist es das Opensource Betriebssystem *ROS* mit seinem Aufbau und seinen Vorteilen vorzustellen.

*ROS* stellt dem Softwareentwickler Bibliotheken und Werkzeuge zur Verfügung, um Roboteranwendungen zu erstellen. Das auf einem *Internet Protocol (IP)*-basierende modulare Kommunikationsframework ermöglicht die Verknüpfung von Anwendungssoftware, Sensoren und Aktoren, sogar unter mehreren Robotern. Die Grundlage dafür ist die sogenannte Hardwareabstraktion. Durch hardwarespezifische Module wird erreicht, dass Komponenten unterschiedlicher Hersteller miteinander verbunden werden können. Im Rahmen dieser Arbeit gilt dies für Hokuyo Lasersanner und die *FCU* von *AscTec*. Des Weiteren ermöglicht es eine hardwareunabhängige Programmierung, die in den Programmiersprachen C/C++ oder in Python erfolgen kann. Jede Hardwareabstraktion oder Anwendung wird als Node, bzw. Konten bezeichnet und läuft als eigener Prozess.

Der Austausch von Daten zwischen den Nodes erfolgt über so genannte Topics (Abbildung 9.1). Dabei werden von den Knoten Nachrichten (engl. Messages) in Topics gepostet und veröffentlicht (publication). Benötigt ein weiterer Knoten den Inhalt eines Topics, kann

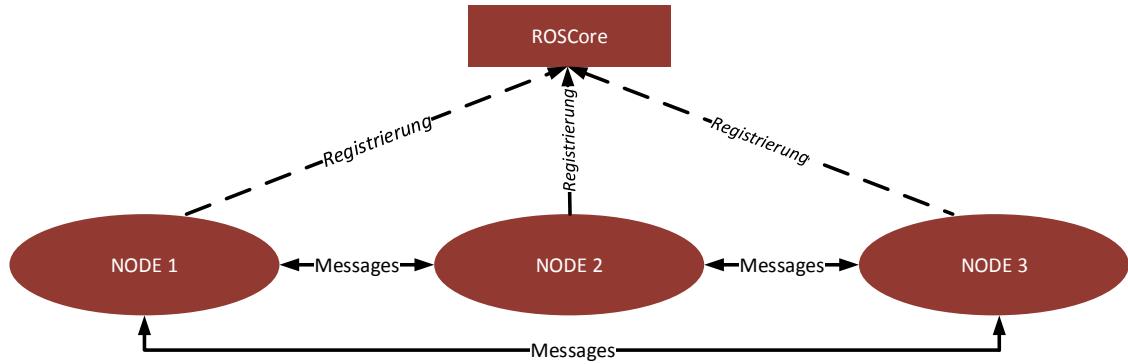


**Abbildung 3.1:** Kommunikation von Nodes über Topics und Services

er es abonnieren (subscription). Sobald die Nachricht im Knoten aktualisiert ist, wird sie in den abonnierenden Knoten übertragen. Dabei sind Knoten nicht auf ein Topic beschränkt. Es können beliebig viele Topics beschrieben oder empfangen werden. Alternativ zu dieser Art der asynchronen Datenübertragung bietet *ROS* die Möglichkeit einer synchronen Kommunikation zwischen zwei Nodes über Services. Auf diese Weise wird auf einem Knoten ein Service gestartet. Dieser dient als Server und agiert nach dem Anfrage-Antwort-Prinzip. Schickt ein anderer Knoten eine Anfrage, wird ihm die geforderte Nachricht zu gesendet.

Anzumerken ist, dass durch das verwendete *IP*-Protokoll keine deterministische Versendung der Nachrichten gewährleistet ist. Es ist möglich, dass Nachrichten gleichen Typs in Paketen zusammengefasst werden. Damit ist Echtzeit unter *ROS* nicht garantiert. Dies betrifft vor allem Knoten deren abonnierte Topics mit einer hohen Update rate gepublished werden. Unkritisch dagegen sind Aktualisierungen in einer Frequenz kleiner 100 Hz, hierbei sind Echtzeitanwendungen möglich. Bei der Programmierung empfiehlt es sich dennoch auf Topics mit einem Zeitstempel (engl. timestamp) zurückzugreifen.

Der größte Vorzug von *ROS* ist die ständig wachsende Community. Hier stellen Forscher aus der ganzen Welt ihre Algorithmen und Hardwareabstraktionen zur Verfügung. Folglich ist es möglich, bei der Erstellung einer Roboteranwendung auf Bausteine zurückzugreifen, die ohne diese Plattform selbst zu implementieren wären. Abgesehen davon, stellt *ROS* eine Vielzahl von Hilfsmitteln bereit. Darunter fallen unter anderem Debug-Tools über die sich Topics sowohl auslesen als auch plotten lassen. Des Weiteren steht mit RViz ein Werkzeug bereit, mit dem sich Messdaten in einer 3D-Umgebung visualisieren lassen. Aber auch für die Entwicklung von Anwendungen existieren Hilfsmittel. So gibt es zum



**Abbildung 3.2:** Registrierung der Knoten

Beispiel die Transferfunktion (`/tf`), über die Daten automatisch in jedes in *ROS* definierte Koordinatensystem übertragen werden.

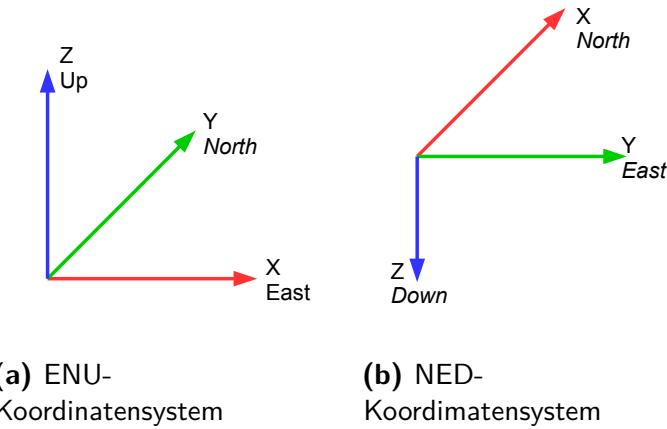
## 3.2 Einführung in die Koordinatensysteme und Koordinatentransformationen

Anhand von Koordinatensystemen und Transformationen lässt sich die Lage von Punkten und Objekten in einem Raum mathematisch beschreiben, die Grundvoraussetzung zur Bestimmung der Position des Quadrocopters im 2D-Raum (Kapitel 8). Ferner ermöglicht die Einführung von Koordinatensystemen die mathematisch/physikalische Beschreibung des Quadrocopters, und stellt die Grundlage zur Modellbildung sowie des Reglerentwurfs dar (Kapitel 5).

### 3.2.1 Koordinatensysteme

Über ein Koordinatensystem lässt sich ein Vektor oder die Position eines Punktes bezogen auf den Koordinatenursprung in einer zweidimensionalen Ebene, bzw. in einem dreidimensionalen Raum beschreiben. Ziel dieses Teilausschnittes ist die Beschreibung der in dieser Arbeit eingeführten Koordinatensysteme.

Zu Beginn werden zwei Konventionen hinsichtlich der Bezugssysteme vorgestellt. Nummer eins, die in Abbildung 9.3a dargestellte *East-North-Up (ENU)* Konvention. Diese wird vor



**Abbildung 3.3:** Konvention von Koordinatensystemen

allem in der Landnavigation eingesetzt. In diesem Fall zeigt die z-Achse nach oben. Bei der zweiten Konvention, hauptsächlich eingesetzt in der Wasser-, Luft- und Raumfahrt, handelt es sich um das *North-East-Down (NED)* Bezugssystem (Abbildung 9.3b). Die z-Achse zeigt nach unten. Anzumerken ist, dass in dieser Arbeit die Ausrichtung der x- und y -Achse nicht wie in Abbildung 9.3 den Himmelsrichtungen entsprechen. Die Begriffe *ENU* und *NED* dienen dabei zur Beschreibung der Ausrichtung der Koordinatenachsen in Abhängigkeit der positiven z-Achse.

Bei der folgenden Einführung der Koordinatensysteme (Abbildung 9.4) handelt es sich ausschließlich um kartesische, dass heißt, orthogonale Koordinatensysteme, die nach der *ENU* Konvention ausgerichtet sind. Das steht erstmals im Widerspruch mit dem Abschnitt zuvor, dort ist das *NED* als Koordinatensystem für Flugkörper eingeführt. Allerdings basieren die *ROS* Koordinatensysteme auf der *ENU* Konvention. Daraus erfolgt die Wahl von Bezugssystemen mit positiver z-Achse nach oben.

Wie aus Abbildung 9.4 zu entnehmen, sind vier xyz-Koordinatensysteme definiert.

- **n-frame(Lokaler Navigationsframe):** Ist ein ortsfestes Koordinatensystem zur Beschreibung der Position im Raum. Da es in dieser Arbeit um die horizontale Positionsregelung geht, ist ausschließlich die xy-Ebene von Interesse. Der Ursprung des Koordinatensystems wird bei jedem Systemstart neu initialisiert. Zu beachten ist, dass dies beim vollautonomen Flug in Räumen erfolgt. Dabei beziehen sich die Sollpositionen nicht auf ein Raumkoordinatensystem mit festem Ursprung, sondern auf den beim Systemstart initialisierten Bezugspunkt. Keinen Einfluss hat diese Tatsache

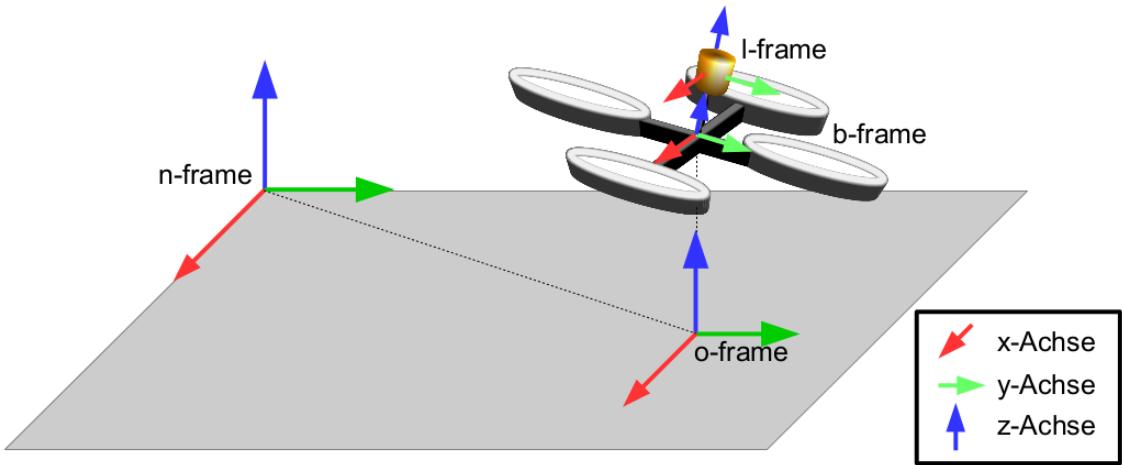


Abbildung 3.4: In der Arbeit angewandte Koordinatensysteme

auf die Geschwindigkeitsregelung per Fernsteuerung, da an dieser Stelle die relative Bewegung von Interesse ist.

Anbei der Hinweis, dass die Verwendung eines xyz Navigationsframes die Krümmung der Erdoberfläche vernachlässigt. Das ist legitim, da die Drohne in Gebäuden zum Einsatz kommt. Erfolgt eine weltweit Navigation wird ein rotationselliptische Koordinatensystem [19] benötigt.

- **b-frame(Bodyframe):** Das Koordinatensystem ist fest mit dem Rahmen des Quadrocopters verbunden. Es ist somit ein körperfestes Koordinatensystem. In diesem Fall befindet sich der Ursprung des Systems im Schwerpunkt. Die x-Achse zeigt in die als Vorne definierte Richtung. Die y- und z-Achse sind davon abhängig nach der *ENU* Konvention angeordnet. Informationen, die sich auf dieses Referenzsystem beziehen, sind unter anderen die *IMU*-Daten. Außerdem lässt sich mit diesem System die Lage des Quadrocopters im n-frame über die Position des Nullpunkts und Drehwinkel beschreiben.
- **l-frame(Laserframe):** Ist ebenfalls ein körperfestes Koordinatensystem, in dem die Entfernungsmessungen des Lasers aufgetragen werden. Der Bezugspunkt liegt folglich in der Sendequelle des Lasers. Die Ausrichtung der Achsen entspricht der des b-frames, mit Ausnahme eines Offsets in z-Richtung.

- **o-frame(Orthogonalframe):** Es handelt sich um ein objektbezogenes Bezugssystem, welches mit dem b-frame verknüpft ist. Das o-frame beschreibt die Orientierung um die z-Achse sowie die Position des Quadrocopters in der horizontalen Ebene des n-frames. Hiermit wird der Quadrocopter in der zu navigierenden xy-Ebene abgebildet.

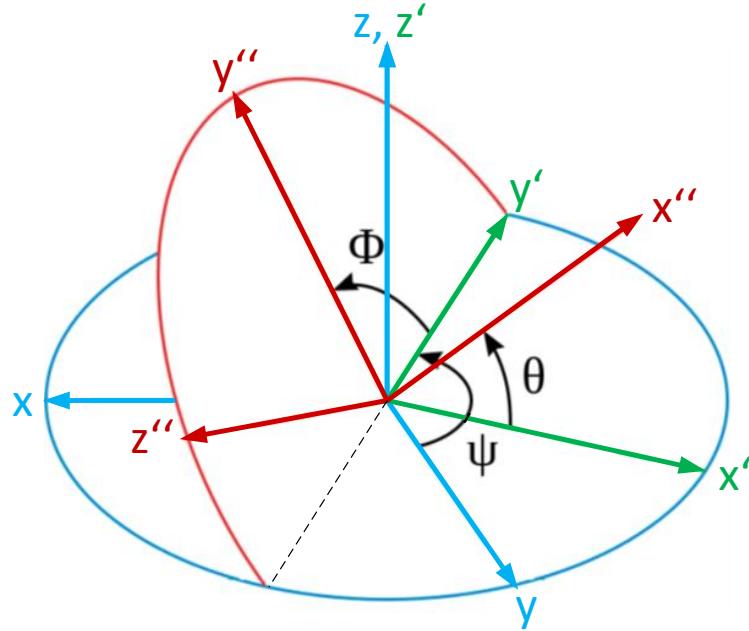
Nachdem sich die Messwerte, wie zum Beispiel die *IMU*-Daten oder die Laserdaten, auf unterschiedliche Koordinatensysteme beziehen, wird eine Koordinatentransformation (Kapitel 9.2.2) erforderlich. Anhand derer lassen sich Vektoren und Koordinaten in die verschiedenen Bezugssysteme übertragen.

### 3.2.2 Koordinatentransformationen

Damit Daten eines Referenzsystems in einen anderen transformiert werden können, muss deren Orientierung zueinander beschreibbar sein. Nach [Buchholz Flugregelung] ist dies über die Rotationswinkel  $\phi$  (Rollwinkel/engl. roll), Rotation um die x-Achse sowie der Winkel  $\theta$  (Nickwinkel/engl. pitch) und  $\psi$  (Gierwinkel/engl. yaw) für die y- und z-Achse erreichbar. Die Reihenfolge, um die Achsen gedreht werden, ist dabei nicht beliebig. Sie ist in verschiedenen Konventionen festgelegt. In dieser Arbeit wird die in der Luftfahrt- und Fahrzeugtechnik gebräuchliche z,y',x"- Konvention (Abbildung 9.5) angewendet.

Das Koordinatensystem wird zu Beginn um den Winkel  $\psi$ , d.h. um die z-Achse gedreht. Daraus ergibt sich das in Abbildung 9.5 grün eingezeichnet Koordinatensystem. Anschließend wird dieses um die Achse y' rotiert, sprich den Winkel  $\theta$ . Zuletzt erfolgt eine Drehung mit dem Winkel  $\phi$  um die x"-Achse. Das Ergebnis stellt das rote x",y",z"-Koordinatensystem dar. Hierbei ist zu beachten, dass die Reihenfolge der Winkel einzuhalten ist, da sonst die beschriebene von der tatsächlichen Lage abweicht. Ein veranschaulichendes Beispiel, wohin unterschiedliche Abfolgen bei der Rotation führen, ist in der Literatur [Literaturverzeichnis] von Herr Thielecke zu finden.

Nach Luftfahrtkonvention lässt sich eine Transformationsmatrix  $M$  aufstellen, mit der sich Vektoren und Koordinaten vom xyz-Koordinatensystem (Bsp.: n-frame) in das x"y"z"-Koordinatensystem (Bsp.: b-frame) überführen lassen. Hierfür sind zunächst die drei Transformationsmatrizen notwendig, die jeweils eine Rotation um eine Koordinatenachse beschreiben [4]. Diese sind wie folgt definiert:



**Abbildung 3.5:**  $z, y', x''$ -Konvention

- Drehung um die  $z$ -Achse mit dem Winkel  $\psi$

$$M_z = \begin{bmatrix} \cos \psi & \sin \psi & 0 \\ -\sin \psi & \cos \psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3.1)$$

- Drehung um die  $y$ -Achse mit dem Winkel  $\theta$

$$M_y = \begin{bmatrix} \cos \theta & 0 & -\sin \theta \\ 0 & 1 & 0 \\ \sin \theta & 0 & \cos \theta \end{bmatrix} \quad (3.2)$$

- Drehung um die  $x$ -Achse mit dem Winkel  $\phi$

$$M_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \phi & \sin \phi \\ 0 & -\sin \phi & \cos \phi \end{bmatrix} \quad (3.3)$$

Aus diesen Rotationsmatrizen ergibt sich über Matrizenmultiplikation eine Gesamttransformationsmatrix. Die Reihenfolge der Multiplikation entspricht der in der Konvention festgelegten Drehfolge, die von rechts nach links gelesen wird. Daraus folgt:

$$M_{bn} = M_x \cdot M_y \cdot M_z$$

$$= \begin{bmatrix} \cos \theta \cos \psi & \cos \theta \sin \psi & -\sin \theta \\ \sin \phi \sin \theta \cos \psi - \cos \phi \sin \psi & \sin \phi \sin \theta \sin \psi + \cos \phi \cos \psi & \sin \phi \cos \theta \\ \cos \phi \sin \theta \cos \psi + \sin \phi \sin \psi & \cos \phi \sin \theta \sin \psi - \sin \phi \cos \psi & \cos \phi \cos \theta \end{bmatrix} \quad (3.4)$$

Mit Hilfe dieser Transformationmatrix lassen sich nach [4] Vektorgrößen zum Beispiel aus dem n-frame ins b-frame übertragen.

$$\begin{bmatrix} x^b \\ y^b \\ z^b \end{bmatrix} = M_{bn} \cdot \begin{bmatrix} x^n \\ y^n \\ z^n \end{bmatrix} \quad (3.5)$$

Anzumerken ist, dass wenn es sich bei dem Vektor um eine Koordinate handelt, zusätzlich der Abstand der Koordinatenursprünge zu addieren ist. Für die Rücktransformation wird die Gesamttransformationsmatrix transponiert. Demzufolge ergibt sich:

$$\begin{bmatrix} x^n \\ y^n \\ z^n \end{bmatrix} = M_{bn}^T \cdot \begin{bmatrix} x^b \\ y^b \\ z^b \end{bmatrix} = M_{nb} \cdot \begin{bmatrix} x^b \\ y^b \\ z^b \end{bmatrix} \quad (3.6)$$

Nunmehr lassen sich Vektoren in beide Richtungen in die verschiedenen Bezugssysteme überführen. Nachteil der Methode mit Eulerwinkel ist, dass diese auf Grund der trigonometrischen Funktionen nur für Winkel  $\phi, \theta, \psi = \{x \in \mathbb{R} | -\pi \leq x \leq \pi\}$  eindeutig durchführbar sind. Wird dieser Bereich überschritten, lässt sich die Lage über Quaternionen beschreiben. Durch die Beschreibung der dreidimensionalen Orientierung in einem vierdimensionalen Raum ist die Lage auch für Rotationen um ein Vielfaches von  $2\pi$  eindeutig charakterisiert. Die genaue Definition findet sich in der Literatur [19] und [4]. Da der Definitionsbereich der Eulerwinkel für diese Arbeit im betrachteten Bereich ausreicht, ist ausschließlich die Umrechnung der in Quaternion  $(w_q, x_q, y_q, z_q)$  angegebenen Orientierungsdaten der IMU

in Eulerwinkel ( $\phi, \theta, \psi$ ) erforderlich. Zu beachten ist, dass die folgende Umwandlung nur für die  $z,y',x''$ -Konvention Gültigkeit besitzt.

$$\phi = \arctan\left(\frac{2(y_q z_q + w_q x_q)}{w_q^2 - x_q^2 - y_q^2 + z_q^2}\right) \quad (3.7)$$

$$\theta = \arcsin(2(w_q y_q - x_q z_q)) \quad (3.8)$$

$$\psi = \arctan\left(\frac{2(x_q y_q + w_q z_q)}{w_q^2 + x_q^2 - y_q^2 - z_q^2}\right) \quad (3.9)$$

Mit dieser letzten Umrechnung sind alle Grundlagen für die Arbeit gelegt. So bilden die Koordinatensystem und Transformationen die Basis für die nachkommende Positionsbestimmung.

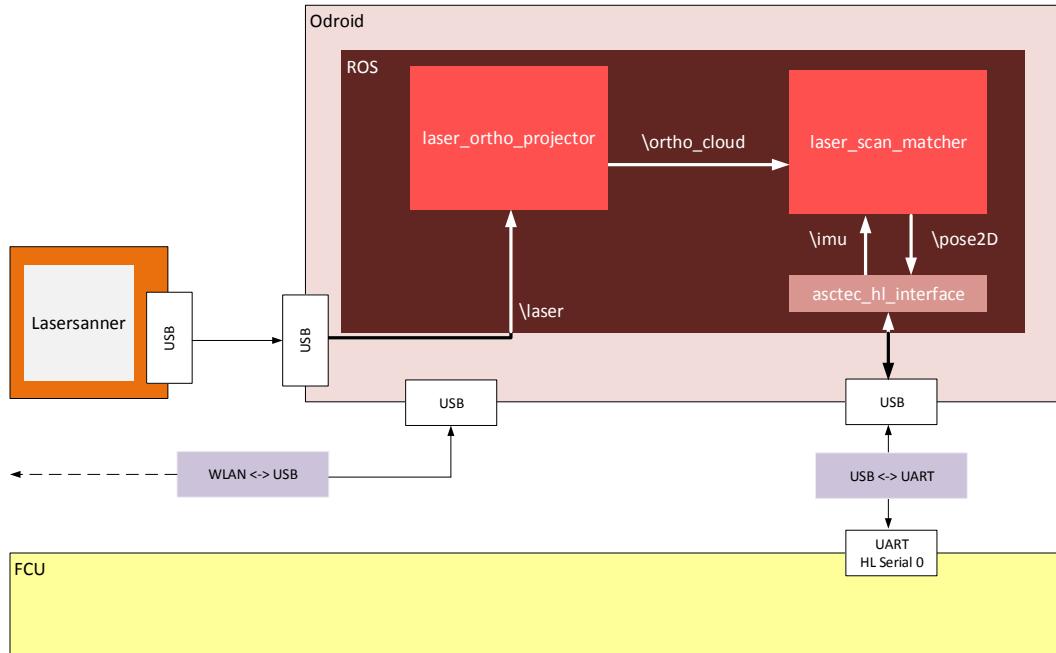
# KAPITEL 4

---

## Zweidimensionale Positionsbestimmung des Quadrocoters in xy-Ebene des Navigationsframes

---

Wie in der Aufgabenstellung beschrieben, erfolgt die Positionsbestimmung über den auf dem Quadrocopter montierten Lasersanner. Es wird als ein Onboard-Lokalisierungssystem bezeichnet. Dabei erzeugt der Laser einen zweidimensionalen Scan der Umgebung. Die aufgenommen Entfernungen sind im l-frame definiert, d.h. sie beziehen sich immer auf den Ursprung des Lasers. Damit enthalten diese Rohdaten keine Information über die Position und Orientierung des Quadrocopters in der xy-Ebene des Navigationskoordinatensystem (n-frame). Allerdings lassen sich aus den Umgebungscans mittels der Methode des „scan-matching“ über relative Positionsverschiebungen die Position in einer zweidimensionalen Ebene bestimmen (Kapitel 8.2). Damit diese für die Bestimmung der Quadrocopterposition in der horizontalen Ebene eingesetzt werden können, ist es von Nöten, die Laserdaten in das o-frame zu projizieren (Kapitel 8.1). Als Folge lassen sich für das o-frame und damit für den Quadrocopter, Position und Orientierung in der xy-Ebene des n-frames bestimmen. Beide Vorgänge sind bereits für *ROS* implementiert und sind Bestandteil der *scan\_tools*. Genauer gesagt, handelt es sich folglich um den „*laser\_ortho\_projector*“- und den „*laser\_scan\_matcher*“-Knoten (Abbildung 8.1). Ziel dieses Kapitel 8 ist es, die Mathematik dahinter zu erläutern, sodass ein Verständnis über die Funktionsweise der Knoten entsteht.

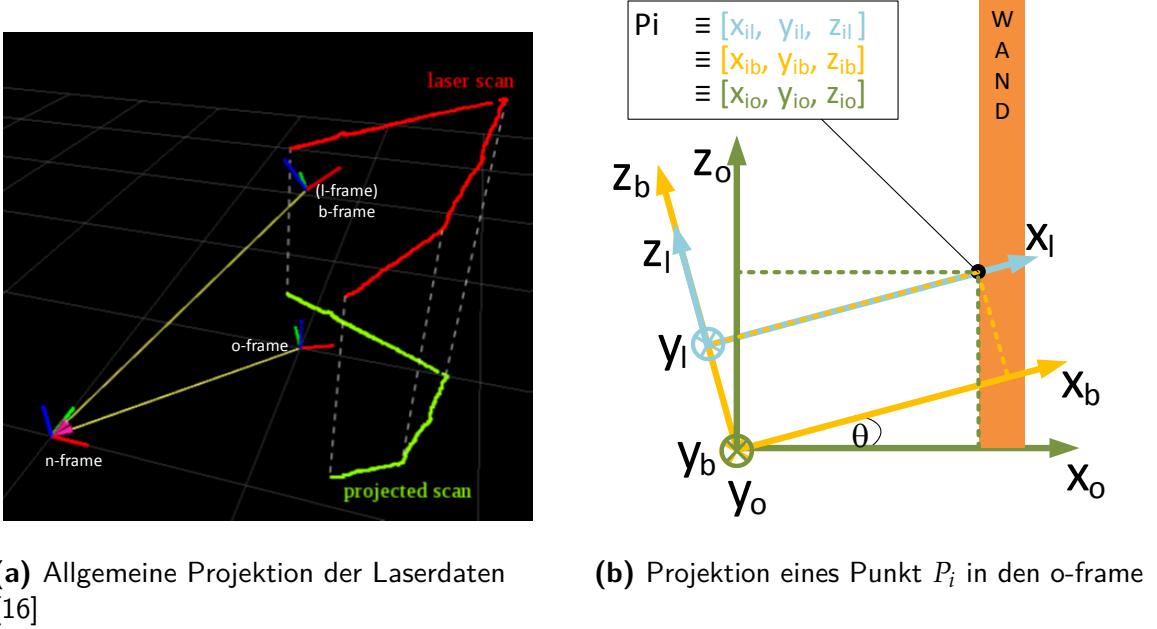


**Abbildung 4.1:** Verknüpfung des „laser\_ortho\_projector“- und des „laser\_scan\_matcher“-Knoten

## 4.1 Projektion der Laserdaten in das o-frame auf der xy-Ebene des n-frames („laser\_ortho\_projector“)

Die Projektion der Laserdaten ins o-frame erfolgt orthogonal zur xy-Ebene des n-frames.

In allgemeiner Form ist dies in Abbildung 8.2a dargestellt. Grundlage für die orthogonale Projektion ist die Annahme, dass es sich bei den erfassten Objekten hauptsächlich um senkrechte Gegenstände handelt. Das heißt, ihre Form variiert nicht mit der Höhe in der sie erfasst werden. In geschlossenen Räumen ist diese Annahme zutreffend, da es sich bei den Objekten hauptsächlich um senkrechte Wände handelt. Durch Erfüllung dieser Voraussetzungen sowie der Annahme kleiner Neigungswinkel des Quadrocopters kann die Flughöhe vernachlässigt werden. Das ist aus der Abbildung 8.2b entnehmbar. Eine Verschiebung des Koordinaten Ursprungs des b-frames auf der z-Achse des o-frames hat demzufolge keinen Einfluss auf die Projektion. Folglich kann für beide Koordinatensystem der identische Ursprung angenommen werden. Unter Beachtung dieser Annahmen kann der Einfluss des Roll-( $\phi$ ) und Nickwinkels ( $\theta$ ) auf die Entfernungsmessung eliminiert werden. Die Winkel-



**Abbildung 4.2:** Projektion der Laserdaten

größen liefert die *IMU*. Der mathematische Ablauf der orthogonalen Transformation wird basierend auf Literatur [16] im Folgenden dargelegt.

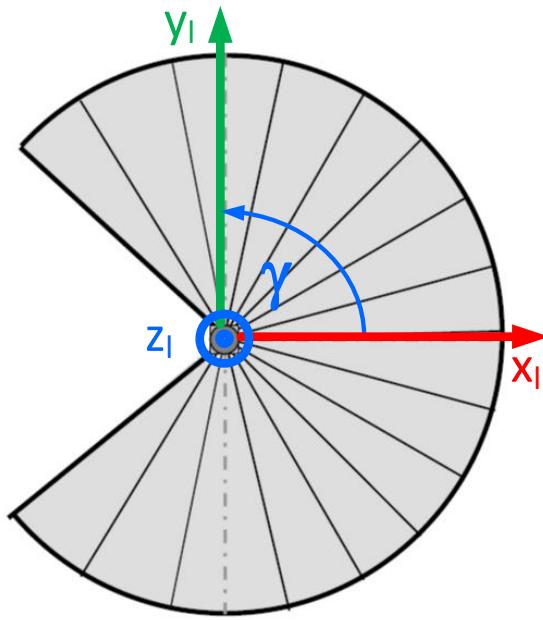
Entfernungsdaten eines Laserumlaufs bestehen aus mehreren diskreten Abtastungen (Abbildung 8.3). Übergeben werden sie in Form von Entfernungen  $r_i$  in einem Array ((ROS)Topic: \laser Abbildung 8.1). Mittels der Schrittweite von  $0.25^\circ$  lassen sich anhand des Indizes  $i$  für jeden Messpunkt ein Winkel  $\gamma_i$  zuweisen.

$$\gamma_i = 135^\circ - 0.25^\circ \cdot i \quad (4.1)$$

Die Position eines Punktes  $p_i$  ist somit über  $\{r_i, \gamma_i\}$  definiert. Zur weiteren Verwendung ist es notwendig, die Messungen in das kartesische Koordinatensystem des I-frame zu übertragen.

$$p_i^l = [\cos(\gamma_i) \cdot r_i, \sin(\gamma_i) \cdot r_i, 0]^T \quad (4.2)$$

Da der Bezugspunkt des o-frames im Schwerpunkt des Quadrocoptes liegen soll, in dem auch der b-frame seinen Ursprung hat, ist es erforderlich die Laserdaten vom I-frame ins



**Abbildung 4.3:** Draufsicht I-frame (Datenblatt Anhang A)

b-frame zu transformieren. Wie schon in Kapitel 9.2 erwähnt, handelt es sich um eine konstante Transformation. Genauer gesagt, um einen Offset von  $10\text{cm}$  auf der  $z^b$ -Achse, denn der Laser ist oberhalb des Quadrocopterschwerpunktes montiert.

$$p_i^b = \begin{bmatrix} \cos(\gamma_i) \cdot r_i, & \sin(\gamma_i) \cdot r_i, & 0.1 \end{bmatrix}^T \quad (4.3)$$

Nachdem die Laserpunkte im b-frame definiert sind, kann die Transformation der Umgebungsdaten in die xy-Ebene des o-frames erfolgen. Angesichts der Tatsache, dass die Winkel  $\phi$  und  $\theta$  als Verdrehung um die Achsen des o-frames definiert sind, erfordert zur Umrechnung der Laserdaten die in Kapitel 9.2 eingeführte Gleichung 9.6 zur inversen Koordinatentransformation. Dabei wird der Yaw-Winkel zu Null gesetzt. Grund hierfür ist, dass die Ausrichtung der Flugrichtung in der zweidimensionalen Ebene mit der des b-frames übereinstimmen sollen. Daraus ergibt sich für die Transformationsmatrix,

$$M_{ob} = \begin{bmatrix} \cos \theta & \sin \phi \sin \theta & \cos \phi \sin \theta \\ 0 & \cos \phi & -\sin \phi \\ -\sin \theta & \sin \phi \cos \theta & \cos \phi \cos \theta \end{bmatrix} \quad (4.4)$$

, über die sich die Positionen der Laserpunkte  $P_i^b$  im o-frame bestimmten lassen.

$$p_i^o = M_{ob} \cdot P_i^b \quad (4.5)$$

Wiederum kann unter der Annahme von rechtwinkligen Objekten die Höhe des Punktes in  $z^o$ -Achse zu Null gesetzt werden. Demzufolge sind die Punkte  $p_i^l$  auf der xy-Ebene des o-frame folgendermaßen abgebildet.

$$p_i^o = \begin{bmatrix} \cos \theta \cos(\gamma_i) \cdot r_i + \sin \phi \sin \theta \sin(\gamma_i) \cdot r_i + \cos \phi \sin \theta \cdot 0.1 \\ \cos \phi \sin(\gamma_i) \cdot r_i - \sin \phi \cdot 0.1 \\ 0 \end{bmatrix} \quad (4.6)$$

Alle Punkte  $p_i^o$  eines Umlaufs eines Scans  $S$ .

$$S^o = [p_i^o | i = 1..1080] \quad (4.7)$$

Die Beschreibung  $S$  der Laserscans im o-frame ist die Basis für die im anschließende Kapitel 8.2 behandelte Positionsbestimmung in der zweidimensionalen Ebene des n-frames.

## 4.2 Positionsbestimmung anhand der ins o-frame überführten Laserdaten über scanmatching

Aufbauend auf den im Kapitel 8.1 vorgestellten Laser\_ortho\_projektor, ist es Aufgabe des folgenden Teilkapitels zu erläutern, wie anhand eines Referenzscans  $S_{ref}$  und einem weiteren Scan  $S_{neu}$  die Position in der euklidischen xy-Ebene des n-frames bestimmt werden kann. Zur Anwendung kommt hier die Methode des Scanmatching. Dabei gilt die Annahme, dass für jeden Scan  $S_{neu}$  und dessen dazugehörigen Position  $P_{neu}$  eine Rotation  $M_z^o$  um  $\psi^o$ , inklusive Translation  $T$  existiert, sodass die beiden Punktwolken  $S_{neu}$  und  $S_{ref}$  übereinander liegen (Abbildung 8.4). Definiert ist  $S_{ref}$  dabei an der Stelle  $P_{ref}$ .

Ausgangspunkt sind zwei im o-frame definierte Punktwolken.

$$S_{ref} = [p_{ref_i} | i = 1..n_{ref}] \quad (4.8)$$

$$S_{neu} = [p_{neu_i} | i = 1..n_{neu}] \quad (4.9)$$

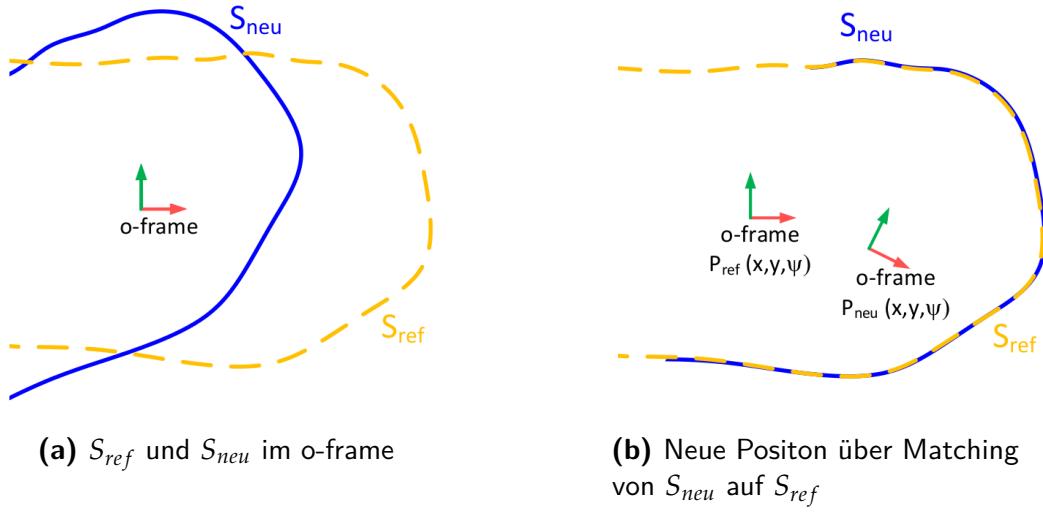


Abbildung 4.4: Prinzip Scanmatching

Dargestellt in Abbildung 8.4a.

Zur Bestimmung des Matchings bzw. der Rotation  $M_z^o$  und der Translation  $T$  wurde im Jahr 1992 unter anderem von Paul Besl der *Iterative Closest Point (ICP)*-Algorithmus entwickelt. Dabei handelt es sich um einen iterativen Algorithmus, dessen Ablauf im Folgenden schrittweise erläutert ist.

- **Schritt 1,** Punktekorrespondenz.

Hierbei bekommt jeder Punkt des  $S_{neu}$  einen korrespondierenden Punkt aus der Punktfolge  $S_{ref}$  zugewiesen. Man spricht hierbei von der Point-to-Point Arithmetik. Als Korrespondenzpunkt  $p'_{ref_i}$  des Scanpunktes  $p_{neu_i}$  wird der nächstliegende Punkt des Referenzscans  $S_{ref}$  zugewiesen. Dabei kann ein Punkt  $p_{ref}$  die Bedingung des Closest-Point für mehrere Punkte des neuen Scans  $S_{neu}$  erfüllen.

$$S'_{ref} = [p'_{ref_i} | i = 1..(n' = n_{neu})] \quad (4.10)$$

Die Suche der entsprechenden Werte erfolgt über die Methode der erschöpfenden Suche (Brute-Force-Methode), d.h. für jeden Punkt werden alle Punktabstände ermittelt und der Punkt mit der geringsten Entfernung zugewiesen. Für die 1080 Messpunkte eines Umlaufes müssen so pro Scan 1166400 Kombinationen untersucht werden. Das Ergebnis ist eine Datenmenge  $S'_{ref}$ , deren Anzahl den Wert von  $S_{neu}$  entspricht.

Dieser Schritt benötigt auf Grund des nicht optimierten Suchalgorithmus die höchste Rechenleistung. Laut [15] stellt dies für die Verarbeitung von 2D-Laserscans kein Problem dar.

- **Schritt 2,** Bestimmung des Rotationswinkels  $\Delta\psi^o$  und der Translation  $T$

Wie zu beginn angemerkt, soll eine rotatorische und translatorische Transformation zur Überlappung von  $S_{neu}$  mit  $S_{ref}$  führen. Idealität und eine sehr kleine Änderung vorausgesetzt, kann jeder Punkt  $p_{neu_i}$  von  $S_{neu}$  in den entsprechenden Punkt  $p_{ref_i}$  von  $S_{ref}$  über (8.11) umgerechnet werden.

$$p_{ref_i}(M_z^o(\psi^o), T) = M_z^o(\psi^o) \cdot p_{neu_i} + P_{ref} + T \quad (4.11)$$

Die zweidimensionale Roationsmatrix  $M_z^o$  entspricht dabei der in Kapitel 9.2.2 eingeführten Koordinatentransformationsmatrix (9.1) reduziert auf die x und y Anteile.

$$M_z^o(\psi^o) = \begin{bmatrix} \cos \psi^o & -\sin \psi^o \\ \sin \psi^o & \cos \psi^o \end{bmatrix} \quad (4.12)$$

In der Praxis ist die Umgebung nicht ideal und aufgrund der hohen Dynamik können die Änderungen zwischen zwei Messwerten unbestimmt größer ausfallen. Dadurch sind nicht alle Werte von  $S_{ref}$  und  $S_{neu}$  Index über Index vergleichbar. Der Grund weshalb in Schritt 1 zum neuen Scan  $S_{neu}$  ein korrespondier Scan  $S'_{ref}$  eingeführt wird. Dieser findet im folgenden Anwendung. Ein Einsetzen der Punkte  $p'_{ref_i}$  von  $S'_{ref}$  in die Gleichung (8.11) ist nicht die Lösung. Da Aufgrund von möglichen Mehrfachkorrespondenzen die Gleichung nicht eindeutig gelöst werden kann. Laut [13] kann jedoch diese Formel (8.11) als Fehlgleichung herangezogen werden. Mit Hilfe der least-square Methode kann darüber die Transformation bestimmt werden, für sich die kleinste Summe der quadratischen Abweichungen ergibt.

$$E(M_z^o(\Delta\psi^o), T) = \frac{1}{n'} \sum_{i=1}^{n'} \|p'_{ref_i} - (M_z^o(\Delta\psi^o) \cdot p_{neu_i} + T)\|^2 \quad (4.13)$$

Um das Minimum abhängig von  $E(M_z^o(\Delta\psi^o), T)$  bestimmen zu können, wird nach [15] der Schwerpunkt  $(c_{ref}, c_{neu})$  der korrespondierenden Punkte ermittelt.

$$c_{ref} = \frac{1}{n'} \sum_{i=1}^{n'} p'_{ref_i} \quad (4.14)$$

$$c_{neu} = \frac{1}{n'} \sum_{i=1}^{n'} p_{neu_i} \quad (4.15)$$

Damit beschreiben sich die Punktwolken folgendermaßen:

$$\tilde{S}'_{ref} = [\tilde{p}'_{pref_i} = p'_{pref_i} - c_{ref} | i = 1..n'] \quad (4.16)$$

$$\tilde{S}_{neu} = [\tilde{p}_{neu_i} = p_{neu_i} - c_{neu} | i = 1..n'] \quad (4.17)$$

Setzt man 8.16 und 8.17 in die Fehlergleichung 8.13 ein, resultiert:

$$\begin{aligned} E(M_z^o(\Delta\psi^o), T) &= \frac{1}{n'} \sum_{i=1}^{n'} \|\tilde{p}_{ref_i} - M_z^o(\Delta\psi^o) \cdot \tilde{p}_{neu_i} - (T - c_{ref} + M_z^o(\Delta\psi^o) \cdot c_{neu})\|^2 \\ &= \frac{1}{n'} \sum_{i=1}^{n'} \|\tilde{p}_{ref_i} - M_z^o(\Delta\psi^o) \cdot \tilde{p}_{neu_i} - \tilde{T}\|^2 \end{aligned} \quad (4.18)$$

Über  $\tilde{T}$  ist die Abweichung der Schwerpunkte translatorisch als auch rotatorisch beschrieben. Damit beide Schwerpunkte genau über einander liegen ist  $\tilde{T} = 0$  zusetzen. Daraus ergibt sich:

$$0 = T - c_{ref} + M_z^o(\Delta\psi^o) \cdot c_{neu} \quad (4.19)$$

Sowie eine Fehlergleichung, die nun ausschließlich von der Rotation abhängig ist und dessen Betrag sich wie folgt darstellt:

$$\begin{aligned} E(M_z^o(\Delta\psi^o)) &= \frac{1}{n'} \sum_{i=1}^{n'} \|\tilde{p}_{ref_i} - M_z^o(\Delta\psi^o) \cdot \tilde{p}_{neu_i}\|^2 \\ &= \frac{1}{n'} \sum_{i=1}^{n'} (\tilde{p}_{ref_i}^T \tilde{p}_{ref_i} + \tilde{p}_{neu_i}^T \tilde{p}_{neu_i} - 2\tilde{p}_{ref_i}^T \cdot M_z^o(\Delta\psi^o) \cdot \tilde{p}_{neu_i}) \end{aligned} \quad (4.20)$$

Weiterhin auf die Literatur [15] beziehend, ist es zur Minimierung von  $E(M_z^o(\Delta\psi^o))$  ausreichend den gemischten Term zu  $\tilde{E}(M_z^o(\Delta\psi^o))$  maximieren.

$$\tilde{E}(M_z^o(\Delta\psi^o))_{max} = \underset{M_z^o(\Delta\psi^o)}{\operatorname{argmax}} \sum_{i=1}^{n'} (2\tilde{p}_{ref_i}^T \cdot M_z^o(\Delta\psi^o) \cdot \tilde{p}_{neu_i}) \quad (4.21)$$

Beziehungsweise abhängig von  $\Delta\psi^o$ .

$$\begin{aligned} \tilde{E}(\Delta\psi^o)_{max} = \operatorname{argmax}_{\Delta\psi^o} & \sum_{i=1}^{n'} (2(\cos(\Delta\psi^o)(\tilde{x}_{ref} \cdot \tilde{x}_{neu} + \tilde{y}_{ref} \cdot \tilde{y}_{neu}) \\ & + \sin(\Delta\psi^o)(\tilde{y}_{ref} \cdot \tilde{x}_{neu} + \tilde{x}_{ref} \cdot \tilde{y}_{neu})) \end{aligned} \quad (4.22)$$

Aufgrund der trigonometrischen Addition besitzt der Summand ein Maximum für  $\Delta\psi^o$ , wenn

$$\frac{\delta \tilde{E}(\Delta\psi^o)_{max}}{\delta(\Delta\psi^o)} = 0 \quad (4.23)$$

$$\begin{aligned} 0 = & \sum_{i=1}^{n'} (-\sin(\Delta\psi^o)(\tilde{x}_{ref} \cdot \tilde{x}_{neu} + \tilde{y}_{ref} \cdot \tilde{y}_{neu}) \\ & + \cos(\Delta\psi^o)(\tilde{y}_{ref} \cdot \tilde{x}_{neu} + \tilde{x}_{ref} \cdot \tilde{y}_{neu})) \end{aligned} \quad (4.24)$$

Draus folgt für  $\Delta\psi^o$

$$\Delta\psi^o = \arctan\left(\frac{\sum_{i=1}^{n'} (\tilde{y}_{ref} \cdot \tilde{x}_{neu} + \tilde{x}_{ref} \cdot \tilde{y}_{neu})}{\sum_{i=1}^{n'} (\tilde{x}_{ref} \cdot \tilde{x}_{neu} + \tilde{y}_{ref} \cdot \tilde{y}_{neu})}\right) \quad (4.25)$$

Mit dem Ergebnis für  $\Delta\psi^o$  kann unter Verwendung von Gleichung 8.19 die Translation T bestimmt werden.

$$T = c_{ref} - M_z^o(\Delta\psi^o) \cdot c_{neu} \quad (4.26)$$

Somit sind Translation und Rotation für die Summe der kleinsten Quadrate bestimmt.

- **Schritt 3,** Vergleich von  $E(M_z^o(\Delta\psi^o), T)$  mit Schwellwert  $E_{max}$ . Die kleinste Summe der quadratischen Fehler für die ermittelte Transformation wird mit dem Schwellwert des Maximal zulässigen Fehlers  $E_{max}$  verglichen. Bei Unterschreitung ist das Abbruchkriterium erfüllt. Es handelt sich bei der Rotation  $\Delta\psi^o$  und der Translation  $T$  um die Transformation, welche die neue Position  $P_{neu}$  mit einer ausreichenden Genauigkeit beschreibt. Ist das Kriterium nicht erfüllt, werden die Transformationswerte auf den Scan angewendet und der *ICP*-Algorithmus beginnt iterative bei Schritt 1 von Neuem.

Für die Positionsbestimmung im n-frame wird mit der ersten Position  $P_{ref}$  der Bezugspunkt des Navigationskoordinatensystems gesetzt. Darauf aufbauend, werden die weiteren Translationen bzw. Rotationen addiert.

$$P_{neu} = P_{ref} + T \quad (4.27)$$

$$\psi_{neu}^o = \psi_{ref}^o + \Delta\psi^o \quad (4.28)$$

Anzumerken ist hier, dass  $P_{ref}$  nicht im Ursprung verweilt, sondern sich auf den Referenzscan  $S_{ref}$  bezieht. Somit ist die absolute Position im Raum über relative Positionsverschiebungen bestimmt. Dies hat aufgrund des maximal zulässigen Schätzfehler  $E_{max}$  zur Folge, dass über einen längeren Zeitraum die geschätzte absolute Position von der Realen abweicht. Um diesen Drift auf eine Minimum zu reduzieren, ist es sinnvoll nicht immer auf den vorherigen Umlauf des aktuellen Scans  $S_{neu}$  als Referenz  $S_{ref}$  zurückzugreifen, sondern auf einen in der nahen Vergangenheit liegenden Scan, bei dem die Summe der quadratischen Fehler sehr gering war.

Der vorgestellte *ICP*-Algorithmus kann weiterhin verbessert werden. Anstelle der Point-to-Point Arithmetik in Schritt 1 kann eine Point-to-Line Arithmetik zur Ermittlung der Konvergenzpunkte angewendet werden. Mit diesem Thema beschäftigt sich das Paper [5]. Außerdem ist möglich, über die Inertialsenorik ausgehend von  $P_{ref}$  eine neue Position  $P_{imu}$  zu bestimmen. In  $P_{imu}$  wird der neue  $S_{neu}$  gelegt. So muss lediglich der Fehler der *IMU*-Positionsschätzung über den *ICP*-Algorithmus eliminiert werden. Dies ist von Vorteil, wenn zwei Scans weit auseinander liegen, wie zum Beispiel in Abbildung 8.4b. Es vereinfacht Schritt 1. Behandelt wird dies unter anderem in [16].

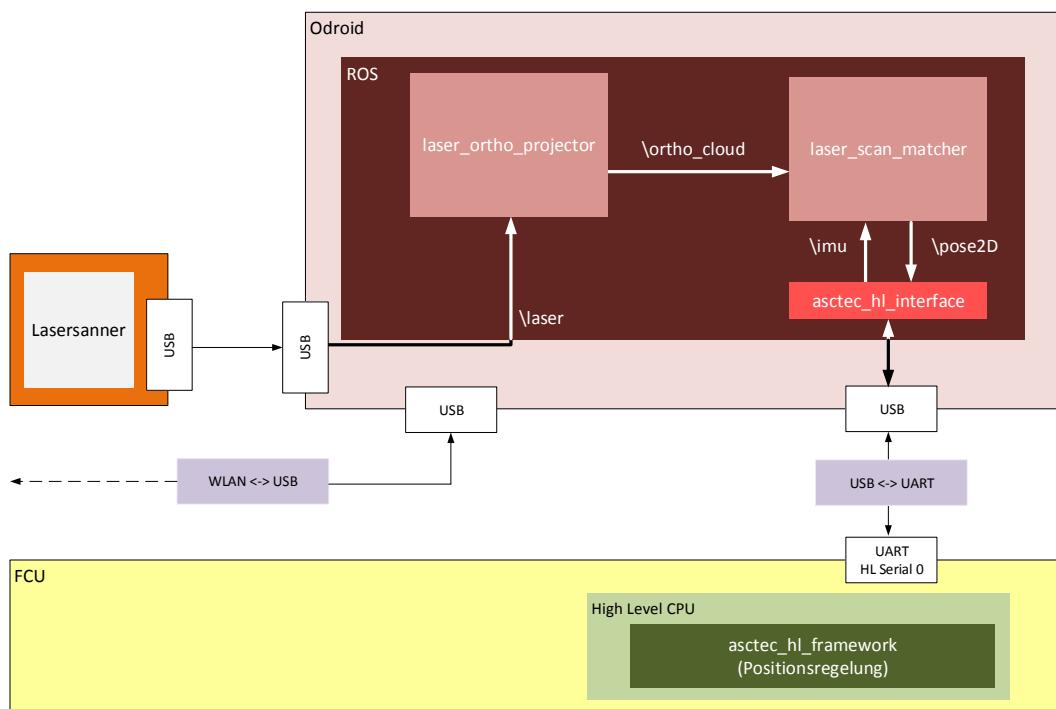
Mit der Beschreibung des Scanmatchingverfahren ist der letzte Baustein zur Lokalisierung des Quadrocopters in der horizontalen Ebene eines geschlossenen Raumes mittels eines

2D-Laserscanners geliefert. Darauf aufbauend, ist es möglich eine Positionsregelung zu implementieren.

# KAPITEL 5

## Verifizierung der Positionsregelung der ETH-Zürich in Verbindung mit einem Laserscanner

In einer Zusammenarbeit von *AscTec* und der ETH-Zürich ist ein Regler zur Positionierung des Pelican Quadrocopters in geschlossenen Räumen entwickelt und veröffentlicht worden. Der Entwurf basierte auf einer monokularen Kamera, die mittels eines *Visual Simultaneous*



**Abbildung 5.1:** AscTec\_hl\_framework in der Kommunikationstruktur

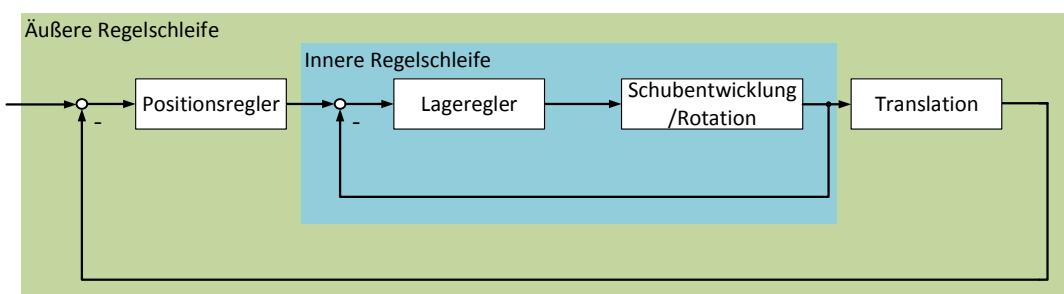
*Localization and Mapping (VSLAM)*-Algorithmus und der Fusion der *IMU* die Position des Flugobjekts in einer unbekannten Umgebung ermitteln kann. Die dazugehörigen Regelung- und Fusionsalgorithmen sind im Paper [3] beschrieben. Außerdem stehen diese als *asc-tec\_hl\_framework* zum Download [1] zur Verfügung und wurden im Rahmen dieser Arbeit auf den *HLP* geladen (Abbildung 5.1).

Aufgabe dieses Kapitel ist es, die veröffentlichten [3] Annahmen und Formeln der Fusion und Regelung zu verifizieren. Dies erfolgt über Literaturrecherchen und Herleitung der publizierten Gleichungen. Unter Berücksichtigung, dass die Position nun mehr vom Laser über die in Kapitel 8 vorgestellten Algorithmen bestimmt wird, ist Abschnitt 5.1 darauf ausgelegt eine Übersicht über die Bestandteile der Regelung zu geben. Herleitung der Komponenten erfolgt in den anschließenden Unterkapiteln.

## 5.1 Struktur der Positionsregelung

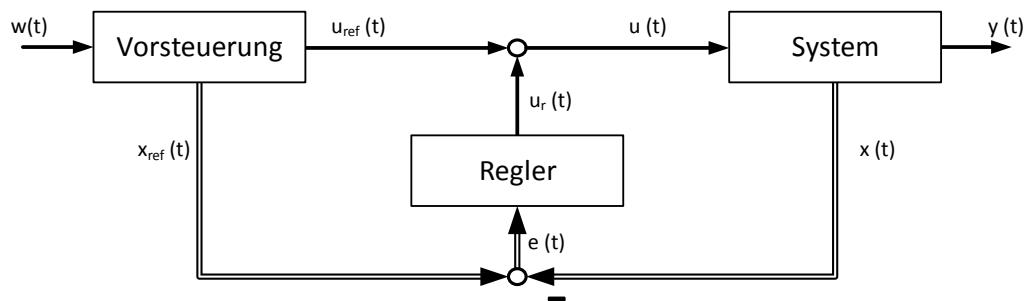
Das Hauptaugenmerk dieses Unterkapitel liegt darauf, wie sich die Positionsregelung in die in Kapitel 2.3 vorgestellte Systemstruktur einfügt. Welche Softwarekomponenten dafür auf dem *HLP* integriert sind. Wie die Kommunikation zwischen ihnen und der Umgebung aussieht. Mit dem Ziel ein grundlegendes Verständnis für die Funktionsweise der Regelung zu generieren.

Die Positionsregelung wird auf die Lageregelung (engl. attitude control) aufgesetzt. Daraus resultiert eine Kaskadenstruktur (Abbildung 5.2). Diese Vorgehensweise ist nachvollziehbar, da die Lageregelung bereits fest auf dem *LLP* implementiert ist. Diese besteht aus einem Regelalgorithmus, der anhand der Regeldifferenz der Orientierung  $e_{ori}$ , welche durch



**Abbildung 5.2:** Kaskadenstruktur der vereinfachten Positionsregelung

die Abweichung der Soll-Orientierung  $O_{des}$  zu Ist-Orientierung  $O$  resultiert, in Verbindung mit der Sollschnellvorgabe  $T_s$  die Drehzahlen  $n_{1..4}$  der Rotoren berechnet und einstellt. Die Ist-Orientierung  $O = [\phi \ \theta \ \psi]^T$  wird mittels eines Fusionsfilter bestimmt. Dieser fusioniert die Messwerte der auf der *IMU* befindlichen Gyroscope sowie der Beschleunigungssensoren mit den Daten des 3D-Kompass. Wie dieser unterlagerte Regler und der dazugehörige Zustandsschätzer genau umgesetzt sind, ist nicht bekannt. Eine mögliche Ausführung ist in Paper [9] beschrieben. Die Ungewissheit über den Regleraufbau der Lageregelung stellt für den Entwurf der überlagerten Positionsregelung keine Problematik dar. Von Interesse ist lediglich, dass die Annahme einer sehr hohen Dynamik dieses Reglers zur Vereinfachung des für die Positionsregelung benötigen Modells (Kapitel 5.2) führt. Hohe Dynamik bedeutet, dass der Sollwinkel in einer sehr kurzen Zeit  $t \rightarrow 0 \text{ s}$  erreicht wird. Basierend auf einer exakten Zustandslinearisierung (Kapitel 5.3) der inneren Schleife in Verbindung mit dem Translationsmodell, ist die äußere Reglerschleife zur Positionsregelung auf dem *HLP* realisiert. Dank der Inversion, entspricht dem Stellgesetz der exakten Zustandslinearisierung, kann für die Positionierung des Quadrocopters eine lineare zwei Freiheitsgrade Regelung 5.3 angewandt werden. Diese besteht aus einer Vorsteuerung und einem Folgeregler. Die Vorsteuerung auf dem *HLP* ist in Form eines Referenzmodells (Kapitel 5.4), das dem linearisierten Modell nachempfunden ist, ausgeführt. Anhand der vorgegebenen Soll-Position  $P_{des}^n = [x \ y \ z]^T_{des}$  wird eine Referenz-Trajektorie<sup>1</sup> zur Überführung des Quadrocopters aus der aktuellen Ist-Position in die Soll-Position berechnet. Ergebnis ist ein Stellwert für die Inversion, der unter theoretischer Betrachtung die gewünschte Bahnbewegung des



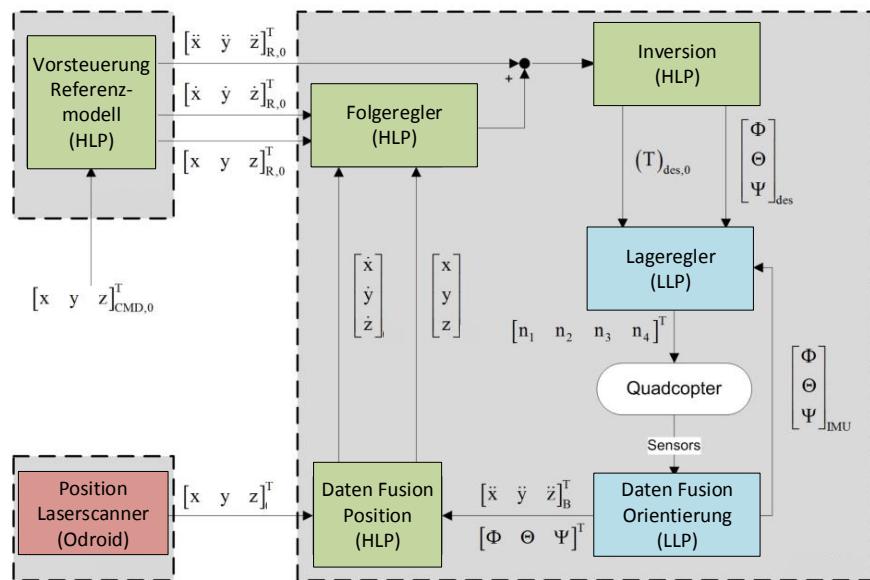
**Abbildung 5.3:** Struktur zwei Freiheitsgraderegelung

<sup>1</sup> Trajektorien beschreiben einen zeitabhängigen Verlauf eines Wertes in einem Bezugssystem

Quadrocopters zur Folge hat. In einem realen System ist dies durch Einflüsse aus der Umgebung, bspw. Wind nicht gewährleistet. Deshalb ist zusätzlich der Folgeregler (Kapitel 5.5) implementiert, dessen Aufgabe darin besteht Abweichungen der realen Zustände des Quadrocopters von den Referenzwerten auszuregeln. Die dafür benötigten Zustandsgrößen des Flugsystems (Kapitel 5.6) werden durch die Fusion, der über Laser bestimmten Position (Kapitel 8) sowie den Beschleunigungswerten der *IMU* bestimmt.

Bevor jede einzelne Komponente in den folgenden Kapiteln hergeleitet wird, ist in Abbildung 5.4 die Verknüpfung aller Komponenten noch einmal grafisch dargestellt.

Anzumerken ist, dass die in Verbindung mit *AscTec* entworfene Positionsregelung zur Ausrichtung des Quadrocopters in einem dreidimensionalen Raum entwickelt ist. Für die vertikale Positionierung ist bereits in der vorausgegangenen Arbeit[11] von Jan Kallwies eine Regelung beschrieben. Da diese Regelung Effekte, wie den Groundeffekt<sup>1</sup> berücksichtigt, ist es die Aufgabe eines auf diese Arbeit folgenden studentischen Projektes, selbige in



**Abbildung 5.4:** Struktur der Regelung (AscTec\_Framework)

<sup>1</sup> In Bodennähe verhindert der Untergrund das schnelle Abströmen des durch die Rotoren erzeugten Luftstroms. Die daraus resultierende Krafterhöhung bei gleichbleibender Drehzahl der Rotoren, wird als Groundeffekt bezeichnet.

das System der ETH-Zürich einzupflegen. Dadurch reduziert sich die Validierung der Reglerstruktur auf die horizontale Ebene.

## 5.2 Modellbildung

Die Modellbildung ist die Grundlage für den systematischen Entwurf einer Zustandsregelung. Dabei wird das Systemverhalten in Form von Differentialgleichungen abgebildet. Diese beschreiben die zeitliche Veränderung einer Ausgangsgröße in Abhängigkeit ihrer zeitlichen Ableitungen sowie veränderlichen Eingangsgrößen. So lassen sich unter Beachtung der physikalischen Gesetze Bewegungsgleichungen aufstellen, welche das räumliche und zeitliche Verhalten eines mechatronischen Systems abbilden.

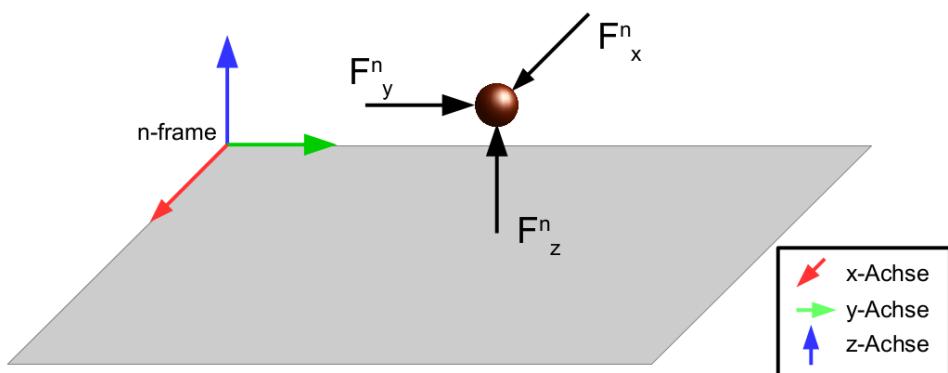
Bevor die Bewegungsgleichungen des Quadrocopter aufgestellt werden können, ist es notwendig, die Freiheitsgrade des dynamischen Systems gegenüber dem Bezugssystem zu bestimmen. Da zwischen dem n-frame(Bezugssystem) und dem b-frame(Quadrocopter) keine mechanische Verbindung oder konstante Koordinatentransformation existiert, besitzt das Flugsystem sechs Freiheitsgrade, bestehend aus drei rotativen  $[\phi \ \theta \ \psi]^T$  und drei translatorischen  $[x \ y \ z]^T$ . Sechs Freiheitsgrade sind gleichzusetzen mit sechs Differentialgleichungen zur Beschreibung der Bewegung im Raum. Damit sind nicht alle Dynamiken des Systems abgebildet. So sind zur Darstellung des Gesamtsystems weitere Differentialgleichungen aufzustellen. Zum einen ist hier die Dynamik der Elektromotoren einzukalkulieren, welche die Rotoren antreiben. Des Weiteren auch die durch die Rotation der Rotorblätter ausgelöste Schubentwicklung nach den Gesetzen der Strömungslehre. Unter Beachtung aller Aspekte entsteht so eine mathematisch meist nichtlineare sowie sehr aufwendige und komplexe Systembeschreibung. Da mit zunehmender Größe des Modells auch die Fehleranfälligkeit steigt, wird in der Modellbildung folgender Leitsatz wiederholt aufgeführt. „Ein Modell sollte das zu regelnde Verhalten so einfach wie möglich und so detailliert wie nötig darstellen.“ Wird die Struktur der implementierten Flugregelung (Abbildung 5.4) betrachtet, reduziert sich das Modell aus Sicht der Positionsregelung erheblich.

Grund hierfür ist die in Bild 5.2 dargestellte Kaskadenstruktur der Regelung. In Verbindung mit Abbildung 5.4 ist zu erkennen, dass der Lageregelung als Eingangsgrößen eine Soll-Orientierung  $O_{des} = [\phi \ \theta \ \psi]^T$  und eine Schubvorgabe  $T_{des}$  übergeben wird. Aus Sicht der Positionsregelung sind das auch die Eingänge des zu regelnden Modells. Fest steht, dass sowohl die rotative Dynamik als auch die Schubentwicklung über den auf dem

LLP befindliche Regler eingestellt wird. Dieser ist ab Werk so gut parametriert, dass die Zeitkonstante zwischen Vorgabe und Einstellen des Sollwerts sehr gering ist. Dies wurde durch einen Versuch bestätigt. Da der Aufbau der Lageregelung nicht bekannt ist, wurde dazu das experimentelle Systemidentifikationstool von Manfred Ottens herangezogen. Um die Zeitkonstante schätzen zu können, wird das Übergangsverhalten von Ist- zu Soll-Winkel als PT1-Glied abstrahiert. Die Zeitkonstante  $T$  wird aus den Messdaten des Sollwertes und des Istwertes ermittelt. Dabei ergibt das Mittel über mehrere Messungen eine Zeitkonstante von  $T \approx 0.1$  s. Mit der Gewissheit, dass die Dynamik der Positionsregelung um ein Vielfaches geringer ausfällt, ist beim Entwurf die Zeitkonstante  $T$  dieser Regelung zu vernachlässigen. Es gilt somit Soll- entspricht Ist-Winkel.

$$O_{des} = O = [\phi \ \theta \ \psi]^T \quad (5.1)$$

Somit reduziert sich die für die Positionsregelung zu modellierende Dynamik auf die translatorischen Differenzialgleichungen. Um zur Bestimmung dieser die Newtonsche Gesetze anwenden zu können, werden diese im n-frame definiert. Dabei kann der Quadrocopter als widerstandsfreie Kugel im dreidimensionalen Raum des Navigationskoordinatensystems abgebildet werden. Auf diese wirkt parallel zur z-Achse des n-frames die Gravitationskraft  $F_g^n$  und die in Richtung der  $z^b$ -Achse angreifende Gesamtschub  $T$  (Gleichung 2.1) der Rotoren. Letztgenannte Kraft wird zur Anwendung der Newtonischen Gesetze [18] in Kraftkomponenten des n-frames zerlegt (Abbildung 5.5). Da die Dynamik der Lageregelung vernachlässigt wird, ist dies über eine einfache Koordinatentransformation vom b-frame ins



**Abbildung 5.5:** Auf den Quadrocopter wirkende Kraft. Definiert im n-frame

n-frame realisierbar. Die entsprechende Transformation ist in Kapitel 9.2.2 in Formel 9.6 eingeführt. Daraus ergibt sich:

$$F^n = \begin{bmatrix} F_x^n \\ F_y^n \\ F_z^n \end{bmatrix} = M_{nb} \cdot F^b - F_g = M_{nb} \cdot \begin{bmatrix} 0 \\ 0 \\ T \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ F_g \end{bmatrix}. \quad (5.2)$$

Nach dem zweiten Newtonschen Gesetz lässt sich die Beschleunigung  $a$  des Körpers bestimmen. Dieses besagt, eine Änderung der Bewegung resultiert proportional und gradlinig in Richtung der wirkenden Kraft. Dabei gilt die Beziehung.

$$F^n = \begin{bmatrix} F_x^n \\ F_y^n \\ F_z^n \end{bmatrix} = m \cdot a = m \cdot \begin{bmatrix} a_x^n \\ a_y^n \\ a_z^n \end{bmatrix} \quad (5.3)$$

Für die konstante Masse  $m (= 1.863 \text{ kg})$  des Quadrocoters, lassen sich die Beschleunigungen des Körpers im dreidimensionalen Raum durch Einsetzen von (5.2) in (5.3) berechnen.

$$\begin{bmatrix} a_x^n \\ a_y^n \\ a_z^n \end{bmatrix} = \frac{1}{m} \cdot (M_{nb} \cdot \begin{bmatrix} 0 \\ 0 \\ T \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ F_g \end{bmatrix}) \quad (5.4)$$

Basierend auf dem Weg-Zeit-Gesetz [18] bestimmt sich für eine Anfangsgeschwindigkeit  $v_0^n$  und eine Startposition  $P_0^n$  die Position des Quadrocopters über die doppelte Integration der Beschleunigung aus Gleichung (5.4).

$$\begin{aligned} a^n &= \dot{v}^n = \ddot{P}^n \\ v^n &= \dot{a}^n = \int a^n dt + v_0^n \\ P^n &= \int v^n dt + P_0^n \end{aligned} \quad (5.5)$$

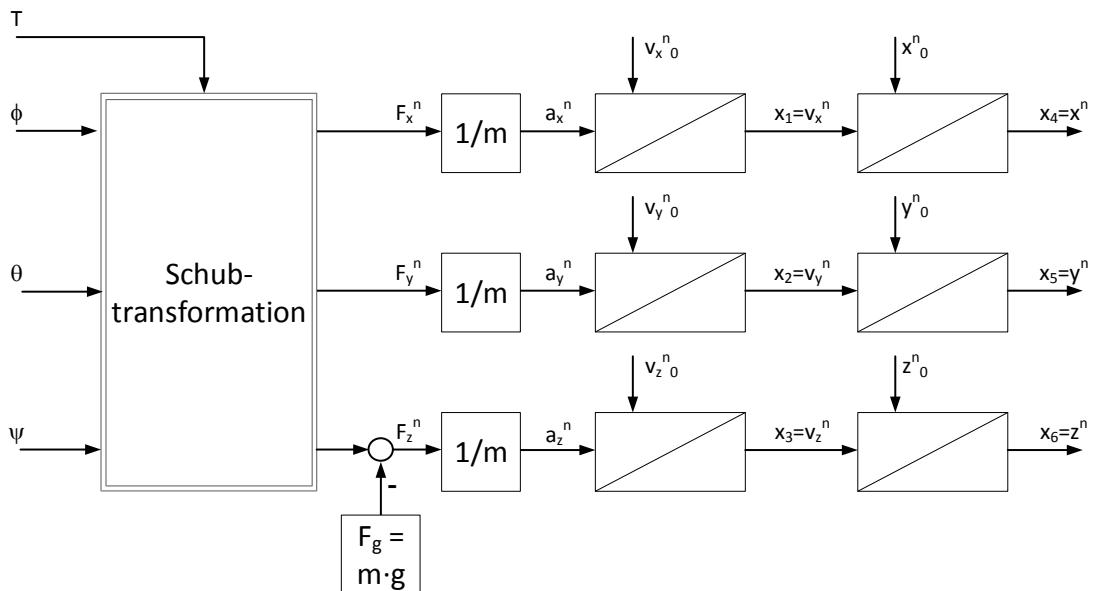
Mit diesen Gleichungen (5.5) und (5.4) ist das translatorische Systemverhalten des Quadrocopters im n-frame mathematisch beschreibbar. Abhängig der Einganggrößen  $O_{des}$

und  $T_{des}$ . Die resultierende Beschreibung der Zustände  $x = [v_x \ v_y \ v_z \ x \ y \ z]^T$  des Modells ergibt.

$$\begin{aligned}\dot{x}_1 &= \frac{1}{m} \cdot ((\cos \phi \sin \theta \cos \psi + \sin \phi \sin \psi) \cdot T) \\ \dot{x}_2 &= \frac{1}{m} \cdot ((\cos \phi \sin \theta \sin \psi - \sin \phi \cos \psi) \cdot T) \\ \dot{x}_3 &= \frac{1}{m} \cdot ((\cos \phi \cos \theta) \cdot T) - g \\ \dot{x}_4 &= x_1 \\ \dot{x}_5 &= x_2 \\ \dot{x}_6 &= x_3\end{aligned}\quad (5.6)$$

Zur Veranschaulichung ist das Modell zusätzlich in Abbildung 5.6 visualisiert.

Durch die Koordinatentransformation bzw. die trigonometrischen Funktionen ist die Systembeschreibung nichtlinear. Das bedeutet, dass von der Änderung der Eingangsgrößen keine direkte proportionale Änderung der Ausgangsgröße ableitbar ist. Somit ist eine direkte Ansteuerung der Lageregelung über einen linearen Positionsregler nicht möglich.



**Abbildung 5.6:** Modellstruktur des Quadrocopters

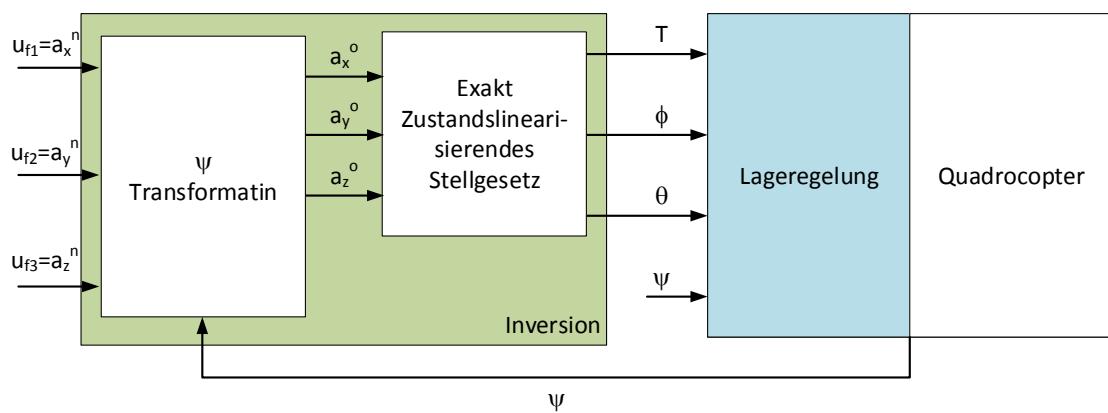
Da allerdings ein solcher linearer Regler vorgesehen ist, wird ein Baustein benötigt, der für fiktive Eingänge das Systemverhalten linearisiert.

## 5.3 Exakte Zustandslinearisierung

Zur Linearisierung eines nichtlinearen Modells wird in den Grundlagen der Regelungstechnik [14] die Arbeitspunktlinearisierung gelehrt. Nachteil einer solchen Linearisierung ist, dass sie oft nur Gültigkeit für eine kleine Umgebung um den Arbeitspunkt besitzt. Um mit dem Quadrocopter schnelle Positionswechsel vollziehen zu können, werden jedoch hohe Stellwinkel benötigt. Somit ist eine annähernde Linearisierung für einen Arbeitspunkt nicht ausreichend. Es wird daher eine Methode benötigt, die das Modell über den gesamten Arbeitsbereich linearisiert.

In [7] wird dafür die exakte Zustandslinearisierung eingeführt. Dabei wird dem nichtlinearen System eine Inversion vorgeschaltet. Diese beinhaltet ein linearisierendes Stellgesetz, sodass sich das zu regelnde System für jeden Ausgang als entkoppelt und lineare Integrierekette darstellt. Die Herleitung eines solchen Stellgesetz mathematisch sehr komplex und anspruchsvoll. Der Grund, warum heutzutage noch viele Regelungen basierend auf der Arbeitspunktlinearisierung entworfen werden.

Exakte Zustandslinearisierung bedeutet für das Quadrocoptermodell (Kapitel 5.2), es ist ein Stellgesetz für  $u = [T \ \phi \ \theta \ \psi]$  gesucht, welches das System für die Ausgänge



**Abbildung 5.7:** Gesamtinversion

$y = \begin{bmatrix} x & y & z \end{bmatrix}$  entkoppelt. Für die Positionsregelung stehen so die fiktiven Eingänge  $u_f = \begin{bmatrix} a_x^n & a_y^n & y_z^n \end{bmatrix}^T$  zur Verfügung. Das bedeutet, es können ohne Berücksichtigung der Orientierung des Quadrocopters Beschleunigungswerte im n-frame vorgegeben werden.

Damit eine solche Inversion für die Position als Ausgang  $y = \begin{bmatrix} x & y & z \end{bmatrix}$  möglich ist, muss es sich bei diesem Ausgang um einen so genannten flachen Ausgang  $y_f = \begin{bmatrix} y_{f1} & y_{f2} & y_{f3} \end{bmatrix}$  handeln.

Per Definition (Anhang B) muss dafür das Kriterium, Anzahl der flachen Ausgänge  $p_y$  entsprechen der Anzahl der Eingänge  $p_u$  des nichtlinearen Systems, erfüllt sein. Dieses Kriterium ist für  $y_f = \begin{bmatrix} x & y & z \end{bmatrix}$  und die Eingangsgrößen  $u = \begin{bmatrix} T & \phi & \theta & \psi \end{bmatrix}$  verletzt. Nach [7] heißt das, es muss ein neuer fiktiver Ausgang gesucht werden, sodass  $p_y = p_u$  gilt. Dieser neue Ausgang entspricht nicht mehr der Position. Das bedeutet, Positionsangaben müssen erst in die neuen Ausgänge transformiert werden. Auch der gewünschte fiktive Eingang der  $u_f = \begin{bmatrix} a_x^n & a_y^n & y_z^n \end{bmatrix}^T$  ist nicht mehr gegeben.

Aus diesem Grund bedient sich die in Kooperation mit AscTec entwickelte Inversion [3] einem Trick. Um die Position als flachen Ausgang zu erhalten, bedient man sich der Tatsache, dass eine Veränderung des  $\psi$ -Winkels zu keiner Auslenkung des Schubvektors  $T$  führt. So können die Beschleunigungsvorgaben  $u_f$  über eine einfache Transformation (Gleichung 9.1) um den Winkel  $\psi$  ins o-frame übertragen werden  $\tilde{u}_f = \begin{bmatrix} a_x^o & a_y^o & y_z^o \end{bmatrix}^T$ . Gleichermaßen gilt auch für den zugehörigen Ausgang  $y_f$ . Demzufolge ergibt sich für diesen  $\tilde{y}_f = \begin{bmatrix} y_{f1}^o & y_{f2}^o & y_{f3}^o \end{bmatrix}$ . Durch den Trick reduziert sich für die Inversion die Anzahl der Eingänge um einen. So entspricht die Anzahl der verbliebenen Eingängen  $\tilde{u} = \begin{bmatrix} \phi & \theta & T \end{bmatrix}$  denen des gewünschten flachen Ausgangs. Damit ist die Flachheit des Ausgangs  $\tilde{y}_f$  nicht erwiesen. Es fehlt per Definition (vgl. Anhang B) noch der Beweis, dass sich alle Zustände  $\tilde{x}$  (vgl. Abbildung 5.6) und die reduzierten Eingänge  $\tilde{u}$  durch die flachen Ausgänge  $\tilde{y}_f$  und dessen Ableitungen darstellen lassen. Für die Zustände ist dieser Beweis, ohne großen Rechenaufwand möglich.

$$\begin{aligned}
 \tilde{y}_{f1}^o &= x^o = x_4 \\
 \tilde{y}_{f2}^o &= y^o = x_5 \\
 \tilde{y}_{f3}^o &= z^o = x_6 \\
 \dot{\tilde{y}}_{f1}^o &= \dot{x}^o = x_1 \\
 \dot{\tilde{y}}_{f2}^o &= \dot{x}^o = x_2 \\
 \dot{\tilde{y}}_{f3}^o &= \dot{z}^o = x_3
 \end{aligned} \tag{5.7}$$

Für die Eingangsgrößen  $\tilde{u}$  werden die Zustandsgleichungen für  $\dot{\tilde{x}}_1$ ,  $\dot{\tilde{x}}_2$  und  $\dot{\tilde{x}}_3$  (vgl. Gleichung 5.6) nach  $\phi$ ,  $\theta$  und  $T$  aufgelöst. Dabei entspricht  $\dot{\tilde{x}}_1 = a_x^o = \ddot{\tilde{y}}_{f1} = \tilde{u}_{f1}$ ,  $\dot{\tilde{x}}_2 = a_y^o = \ddot{\tilde{y}}_{f2} = \tilde{u}_{f2}$  sowie  $\dot{\tilde{x}}_3 = a_z^o = \ddot{\tilde{y}}_{f3} = \tilde{u}_{f3}$ . Damit wird der Schub  $T$  als Betrag der im o-frame wirkenden Kräfte, bzw. geforderten Beschleunigungen multipliziert mit der Masse dargestellt.

$$\begin{aligned}
 T &= m \cdot \sqrt{\ddot{y}_{f1}^2 + \ddot{y}_f^2 + (\ddot{y}_{f3} + g)^2} \\
 &= m \cdot \sqrt{a_x^{o2} + a_y^{o2} + (a_z^o + g)^2}
 \end{aligned} \tag{5.8}$$

Da  $T$  bestimmt ist, ist unter Beachtung das  $\psi = 0$ , durch die vorgelagerte Transformation,  $\dot{\tilde{x}}_2$  aus Gleichung 5.6 nach  $\phi$  aufzulösen. Damit ergibt sich:

$$\begin{aligned}
 \phi &= -\arcsin\left(\frac{\ddot{\tilde{y}}_{f2} \cdot m}{T}\right) \\
 &= -\arcsin\left(\frac{a_y^o \cdot m}{T}\right).
 \end{aligned} \tag{5.9}$$

$\theta$  lässt sich aufgrund der trigonometrischen Beziehung  $\tan\alpha = \frac{\sin\alpha}{\cos\alpha}$  aus den Zustandsgleichungen für  $\dot{\tilde{x}}_1$  und  $\dot{\tilde{x}}_3$  berechnen.

$$\begin{aligned}\theta &= \arctan\left(\frac{\ddot{y}_{f1}}{\ddot{y}_{f3} + g}\right) \\ &= \arctan\left(\frac{a_x^o}{a_z^o + g}\right)\end{aligned}\tag{5.10}$$

Die Stellgesetze der exakten Zustandslinearisierung sind damit über Gleichung (5.8), (5.9) und (5.10) mathematisch beschrieben. In Verbindung mit der Koordinatentransformation von  $\psi$  ergibt sich die in Abbildung 5.7 dargestellte Inversion.

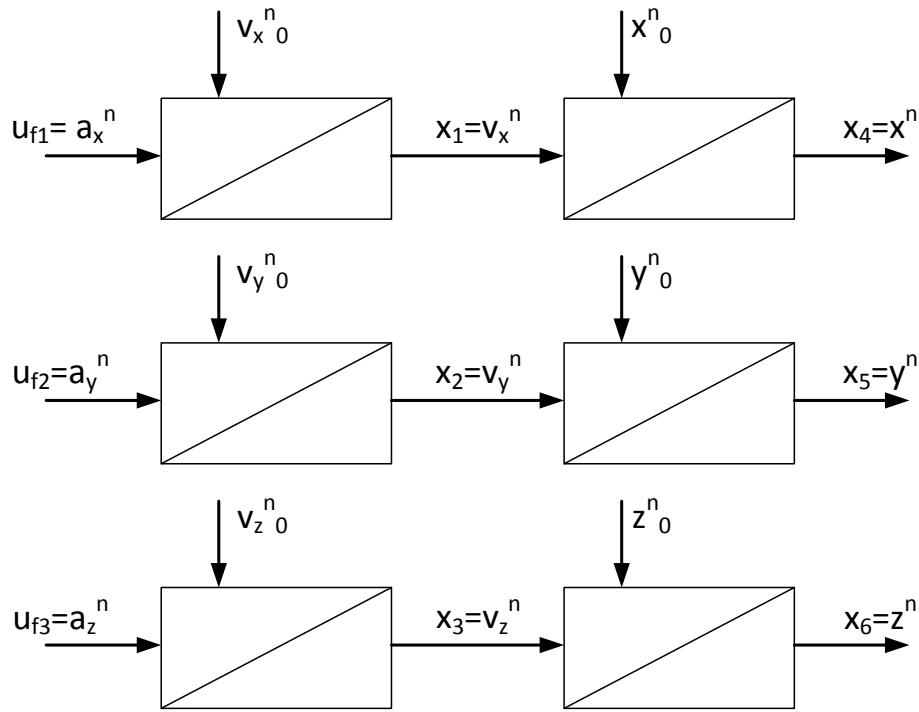
Die Richtigkeit lässt sich über eine Simulation beweisen. Das Modell besteht dabei aus Inversion (Abbildung 5.7) und Translationsmodell (Abbildung 5.6). Über den neuen Eingang  $u_f$  werden dabei Beschleunigungswerte im n-frame vorgegeben (Abbildung 5.9a). Unter Einfluss ständig wechselnder Orientierung um die Hochachse (Abbildung), ergeben sich die in der Abbildung dargestellten Vorgaben für das exaktzustandslinearisierende Stellgesetz. Die daraus resultierende Schubvorgabe und Winkel werden dem Bewegungsmodell übergeben. Mit dem Ergebnis, dass die resultierenden Beschleunigungen des Modells den Beschleunigungsvorgaben entsprechen.

Dadurch ist das zu regelnde System als lineares Modell zu sehen, bestehend aus drei entkoppelten Integrierketten (Abbildung 5.8). Dieses System ist jedoch instabil. Das bedeutet für die Vorgabe einer konstanten Beschleunigung schwingt der Quadrocopter auf keiner konstanten Position ein, sondern Integriert seine Position immer weiter auf. Begründet ist dieses Verhalten auch durch das Weg-Zeit-Gesetz (Gleichung 5.5). Jedoch besteht die Möglichkeit einen Beschleunigungsverlauf vorzugeben, für den der Quadrocopter in eine neue Position überführt wird. Dafür wird eine Vorsteuerung benötigt.

## 5.4 Vorsteuerung/Referenzmodell

Eine Vorsteuerung erzeugt anhand einer Sollwertvorgabe einen Stellgrößenverlauf, der erforderlich ist, um das System im störungsfreien Fall in diesen Zustand zu überführen. In [3] wird deshalb auch von Referenzmodell gesprochen.

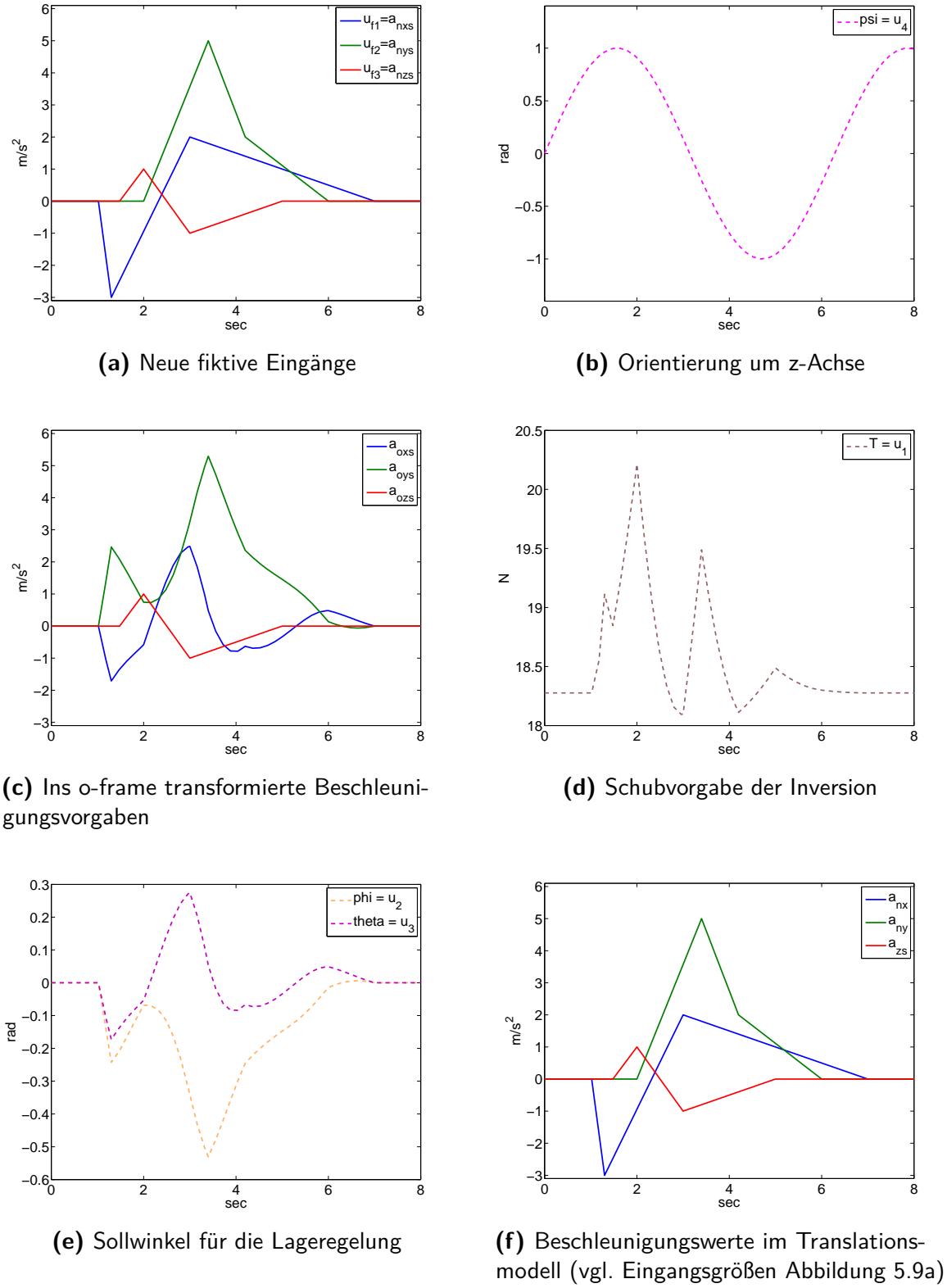
In Sachen Positionsregelung bedeutet das die Übergabe einer anzufliegende Position  $P_{cmd}^n$  anhand der die Vorsteuerung eine Trajektorie generiert. Damit der Quadrocopter dem vorge-



**Abbildung 5.8:** Modell aus Sicht der Positionsregelung dank Inversion

geben Pfad  $P_{ref}^n$  folgen kann, muss dessen Verlauf mindestens einmal stetig differenzierbar sein. Das bedeutet, die zweite Ableitung der Position  $\ddot{P}_{ref}^n$  und somit die Eingangsgröße  $U_f$  der Inversion muss mindestens Stückweise existieren, sodass der Referenzpfad  $P_{ref}^n$  als auch die Referenzgeschwindigkeit  $\dot{P}_{ref}^n$  einen kontinuierlichen Verlauf aufweisen (Abbildung 5.11). Damit dies gewährleistet ist, ist nach [7] folgende Differenzialgleichung für das Referenzmodell aufzustellen.

$$\ddot{P}_{ref}^n + c_1 \cdot \dot{P}_{ref}^n + c_0 \cdot P_{ref}^n = c_0 \cdot P_{cmd}^n \quad (5.11)$$



**Abbildung 5.9:** Simulation der Inversion

Hierbei kann die Dynamik mit der das Fluggerät überführt wird, über die Koeffizienten  $c_0$  und  $c_1$  eingestellt werden. Stellt man für (5.11) die Übertragungsfunktion  $G(s)$  auf.

$$\begin{aligned} G(s) &= \frac{P_{ref}^n}{P_{cmd}^n} = \frac{c_0}{s^2 + c_1 \cdot s + c_0} \\ &= \frac{1}{\frac{1}{c_0}s^2 + \frac{c_1}{c_0} \cdot s + 1} \end{aligned} \quad (5.12)$$

Ist erkennbar, dass diese in ihrer Form einem PT2-Glied entspricht.

$$G_{PT2}(s) = \frac{1}{(\frac{1}{\omega_0})^2 s^2 + \frac{2D}{\omega_0} \cdot s + 1} \quad (5.13)$$

Setzt man (5.13) mit (5.12) gleich, kann man die Koeffizienten  $c_0$  und  $c_1$  abhängig der gewünschten Eigenfrequenz  $\omega_0$  und Dämpfung  $D$  wählen.

$$c_0 = \omega_0^2 \quad (5.14)$$

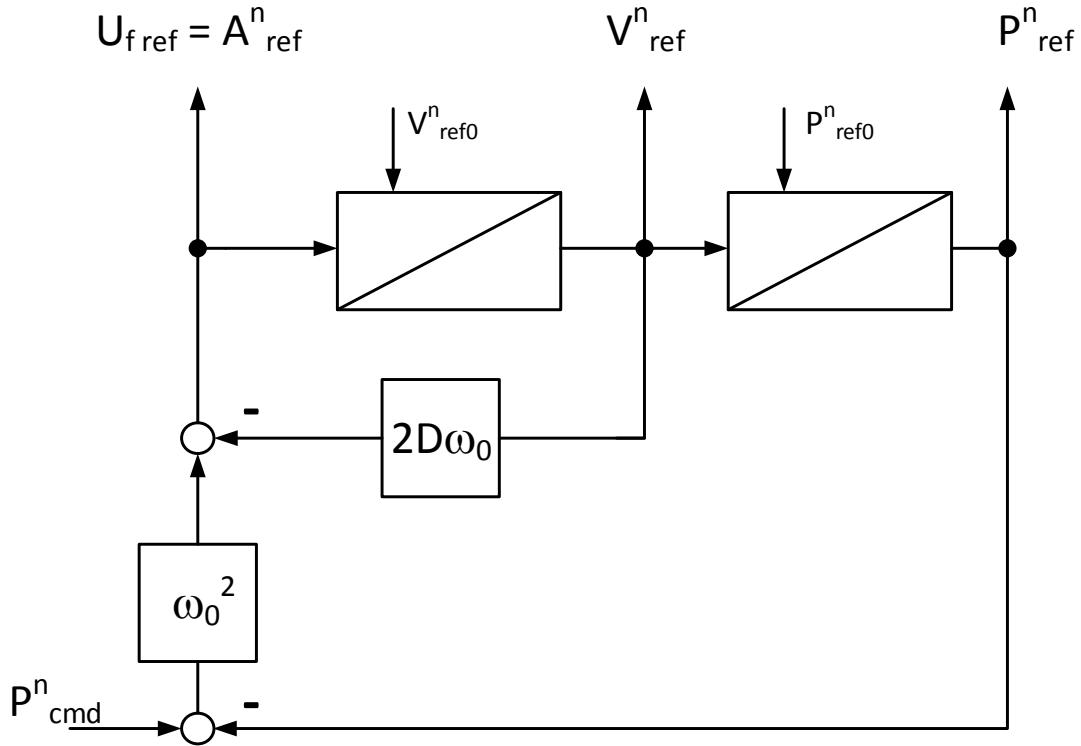
$$c_1 = 2D\omega_0 \quad (5.15)$$

Der Vorteil, die Dynamik der Vorsteuerung ist über die in den Grundlagen der Regelungs-technik vermittelten Entwurfskriterien für PT2-Glieder einstellbar. So lässt sich über  $\omega$  einstellen, wie schnell die Zielkoordinate erreicht werden soll. Das wie, lässt sich über die Dämpfung  $D$  bestimmen. So wird für alle  $D \leq 1$  eine Pfad ohne Überschwinger generiert. Die Dynamikvorgabe ist durch die maximal zulässige Stellgröße beschränkt. So kann zumindest  $\omega$  nicht beliebig groß gewählt werden.

Der Stellgrößenverlauf  $u_{ref} = \ddot{P}_{ref}^n$  ist damit über (5.11) mit (5.14) und (5.15) folgendermaßen festgelegt.

$$u_{ref} = \omega_0^2 \cdot (P_{cmd}^n - P_{ref}^n) - 2D\omega_0 \cdot \dot{P}_{ref}^n \quad (5.16)$$

Das resultierende Strukturbild der Vorsteuerung ist in Abbildung 5.10 visualisiert. Für eine Positionsverschiebung von 1 m erzeugt das Referenzmodell die in Abbildung 5.11 Trajektorie bzw. den in Grafik 5.11c dargestellten Stellwertverlauf. Hier ist anzumerken, dass aufgrund der entkoppelten Integriererketten (Kapitel 5.3) eine separate Dynamikvorgabe für jeden Zweig des Zustandslinearisierten Modells möglich ist. Weiterhin besteht die Möglichkeit



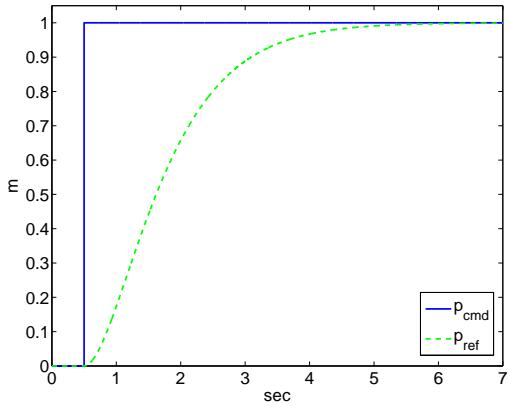
**Abbildung 5.10:** Strukturbild Vorsteuerung

die Vorsteuerung auszubauen, sodass Geschwindigkeitsvorgaben gemacht werden können. Dies erleichtert unter anderem das manuelle Navigieren mittels einer Fernsteuerung. Dabei erfolgt die Vorgabe der Soll-Geschwindigkeiten  $v_{cmd}^o$  im o-frame. Diese Vorgabe wird ins n-frame transformiert und dort über die Zeit integriert. Somit ergibt sich eine stetige Positions vorgabe für  $P_{cmd}^n$ .

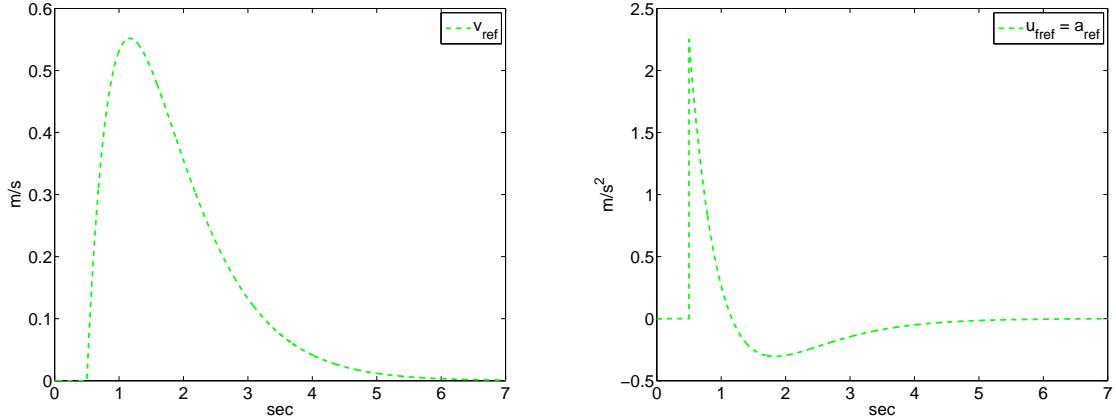
$$P_{cmd}^n = \int (M_z^T \cdot v_{cmd}^o) dt + P_{cmd0}^n \quad (5.17)$$

Mit dieser wird das Referenzmodell (5.16) gespeist. Es wird eine Stellgröße erzeugt, für welches die Position des Quadrocopter der stetigen Positions vorgabe (5.17) folgt. Die maximale Geschwindigkeit ist deshalb durch die Grenzfrequenz  $\omega_0$  des Referenzmodells begrenzt. Je größer  $\omega_0$  ist, desto höher die maximal mögliche Geschwindigkeit.

Die Vorsteuerung ist hiermit für alle Anwendungszwecke entworfen. Speist man das exaktzustandslinearisierte Stellgesetz mit dem in Abbildung 5.11 generierten Sollwertverlauf,



**(a)** Positionsvorgabe über  $p_{cmd}$  und generierte Positionsvorgabe  $p_{ref}$



**(b)** Geschwindigkeitsvorgabe  $v_{ref} = \dot{p}_{ref}$  der Trajektorie

**(c)** Beschleunigungsvorgabe  $u_{fref} = a_{ref} = \ddot{p}_{ref}$

**Abbildung 5.11:** Simulationsergebnis des Referenzmodells für eine Achse mit  $\omega_0 = 1.5 \text{ Hz}$   $D = 1$

so folgt das Modell des Quadrocopter der Trajektorie. Dies gilt allerdings nur für den Fall konsistenter Anfangszustände (Abbildung 5.12a). Ist dies nicht gegeben, ist wie in Abbildung 5.12b zu erkennen, dass der Positionsverlauf des Translationsmodells für den gleichen Stellgrößenverlauf nicht der Referenz entspricht. Das gleiche gilt für Unsicherheiten der Modellparameter (Abbildung 5.12c). Hierbei berechnet die Inversion einen unzureichenden oder überdimensionierten Schubvektor.

Unter anderem können äußere Krafteinwirkungen, wie zum Beispiel Winde wirken, die den Quadrocopter von der Trajektorie auslenken. Um all diesen Effekten entgegenzuwirken benötigt man einen Folgeregler. Dessen Aufgabe besteht darin, diese Einflüsse zu eliminieren, sodass der Modellverlauf auch im gestörten Fall der Trajektorie entspricht.

## 5.5 Folgeregler

Inkonsistente Anfangsbedingungen, externe Störungen sowie Modellunsicherheiten bewirken einen Ausgangsfolgefehler  $e = P_{ref}^n - P^n$ . Dieser pflanzt sich weiter fort, da es sich bei dem Stellsignal  $u_{f_{ref}}$  (5.16) nicht um das Stellsignal  $u_f$  handelt, welches den Quadrocopter zurück auf die Trajektorie führt. Das gilt auch im Falle eines unzureichenden Stellsignalverlaufs in Folge von Modellunsicherheiten. Modellieren lässt sich die Fortpflanzung des Folgefehlers aufgrund des linearisierten Modells in Form einer doppelten Integriererkette. Zweifache Integriererkette ergo instabiles System. Damit später die Flugbahn des Quadrocopter mit der Solltrajektorie konvergiert, muss die Fehlerdynamik stabilisiert werden. Zu diesem Zwecke werden die Zustandsgrößen des Ausgangsfolgefehlermodells zurückgeführt (Abbildung 5.13). Daraus ergibt sich für die Vorgabe, dass der Folgefehler  $e$  in einer endlichen Zeit gegen 0 strebt, folgende Differentialgleichung.

$$\ddot{e} + \tilde{c}_1 \cdot \dot{e} + \tilde{c}_0 \cdot e = 0 \quad (5.18)$$

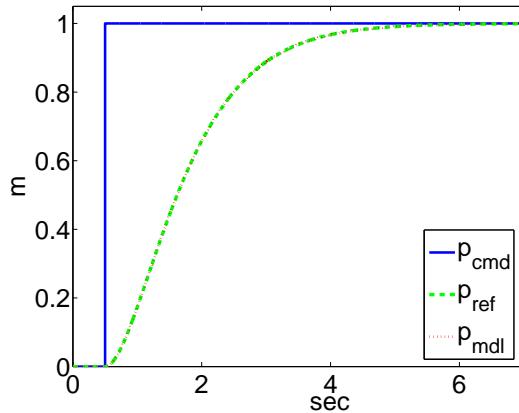
Mit

$$\begin{aligned} e &= P_{ref}^n - P^n \\ \dot{e} &= \dot{P}_{ref}^n - \dot{P}^n \\ \ddot{e} &= u_{f_{ref}} - u_f \end{aligned} \quad (5.19)$$

Diese kann nach  $u_f$  umgestellt werden. Dadurch ergibt sich das nachfolgende Stellgesetz für die Eingangsgröße  $u_f$  des zustandslinearsierenden Systems, welches den Folgefehler ausregelt.

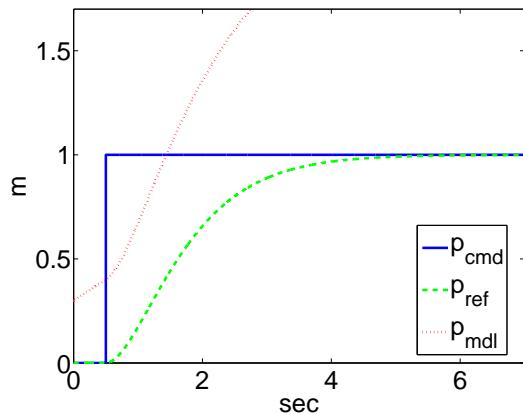
$$u_f = \underbrace{u_{f_{ref}}}_{\text{Vorsteuerung}} + \tilde{c}_1 \cdot (\dot{P}_{ref}^n - \dot{P}^n) + \tilde{c}_0 \cdot (P_{ref}^n - P^n) \quad (5.20)$$

Das Annäherungsverhalten ist abhängig von den Koeffizienten  $\tilde{c}_0$  und  $\tilde{c}_1$ . Anhand von (5.18) lassen sich für diese mittels der Polvorgabe (Kapitel 5.5.1) die Dynamik vorgegeben.



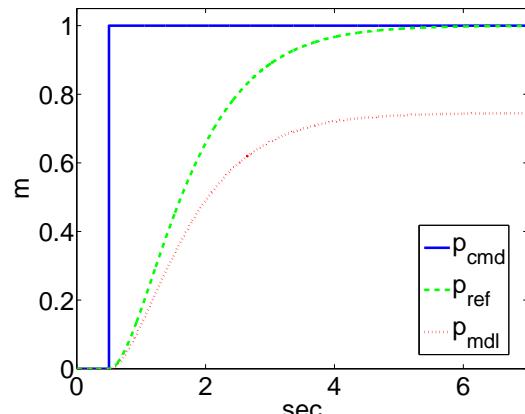
(a) Konsistente Anfangszustände

$p_{ref0} = p_{mdl0} = 0$  und  
 $v_{ref0} = v_{mdl0} = 0$ , ideales Modell



(b) Inkonsistente Anfangszustände

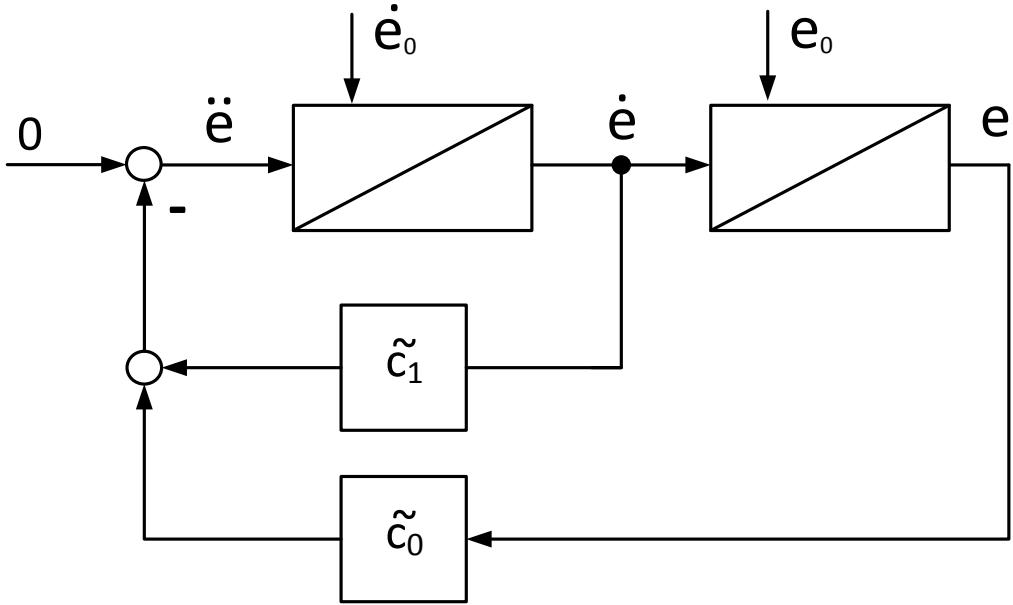
$p_{ref0} = 0 \neq p_{mdl0} = 0.3$  und  
 $v_{ref0} = 0 \neq v_{mdl0} = 0.2$ , ideales  
Modell



(c) Unsicherheit der Modellparameter

$m_{inv} \neq m_{mdl}$

Abbildung 5.12: Steuerung der Regelstrecke über das Referenzmodell



**Abbildung 5.13:** Ausgangsfolgefehlermodell mit Zustandsrückführung

Nicht bekämpft werden durch dieses Stellgesetz konstante Dauerstörungen. Hervorgerufen zum Beispiel durch Winde, die über einen längeren Zeitraum als konstant anzusehen sind. Mit einer konstanten Kraft wirken sie auf das Flugsystem. Beachtet man die von dieser Kraft hervorgerufenen Beschleunigungsbeitrag  $a_{st}$  in der Fehlerdifferenzialgleichung.

$$\ddot{e} + \tilde{c}_1 \cdot \dot{e} + \tilde{c}_0 \cdot e + a_{st} = 0 \quad (5.21)$$

Folgt im stationären Zustand, dass heißt alle Zeitableitungen gleich Null, ein dauerhafter Positionsfehler.

$$e = -\frac{a_{st}}{\tilde{c}_0} \quad (5.22)$$

Lösung dieses Problems ist nach [7] die Erweiterung des Folgereglers mit einer integrierenden Komponente.

$$\ddot{e} + \tilde{c}_1 \cdot \dot{e} + \tilde{c}_0 \cdot e + \tilde{c}_{-1} \int_{I-Anteil} e(\tau) d\tau + a_{st} = 0 \quad (5.23)$$

Der I-Anteil liefert ein Signalanteil zur Kompensation der Störung im stationären Zustand. Den Beweis dafür erhält man, wenn man die Integro-Differentialgleichung (5.23) nach der Zeit ableitet. Danach ergibt sich folgende Differentialgleichung.

$$\ddot{e} + \tilde{c}_1 \cdot \ddot{e} + \tilde{c}_0 \cdot \dot{e} + \tilde{c}_{-1} \cdot e = 0 \quad (5.24)$$

Im eingeschwungenen Zustand ergibt sich so für den Positionsfehler

$$e = 0 \quad (5.25)$$

Das Stellgesetz für einen Regler mit I-Anteil lässt sich unter Vernachlässigung der Dauerstörung  $a_{st}$  aus Gleichung (5.23) entwickeln (vgl. Abbildung 5.14).

$$u_f = \underbrace{u_{ref}}_{\text{Vorsteuerung}} + \tilde{c}_1 \cdot (\dot{P}_{ref}^n - \dot{P}^n) + \tilde{c}_0 \cdot (P_{ref}^n - P^n) + \tilde{c}_{-1} \int e(\tau) d\tau \quad (5.26)$$

*Folgeregler inklusive I-Anteil*

Dieses Stellgesetz führt den Quadrocopter entlang der Referenztrajektorie. Zur erkennen ist dies auch in der Simulation (5.15) mit Regler. So wird für inkonsistente Anfangsbedingungen (Abbildung 5.15a) das Fluggerät auf die Trajektorie geführt. Für Modellunsicherheiten (Abbildung 5.15b) passt der Folgeregler das Stellsignal so an, dass der Quadrocopter entlang des vorgegebenen Pfades fliegt. Vergleich hierzu reine Vorsteuerung (Abbildung 5.12).

Wie zuvor kann das Einschwingverhalten für den Folgeregler mit I-Anteil über die Polvorgabe festgelegt werden. Für diesen Fall ist die Differentialgleichung (5.24) zu verwenden.

### 5.5.1 Einstellung der Dynamik mittels Polvorgabe

Anhand der Lage der Polstellen einer Übertragungsfunktion, werden Rückschlüsse über die Stabilität und das Einschwingverhalten getroffen. Ziel einer Zustandsregelung, wie der Folgeregelung ist, die Pollagen der zu regelnden Strecke so zu verschieben, dass das Ein-/Ausgangsverhalten die gewünschte Dynamik aufweist.

Zunächst ist zu klären, was die Polstellenlage in der komplexen Ebene über das Verhalten aussagt. Dabei entspricht die Anzahl der Pole  $\lambda_i$  der Ordnung  $n$  des Systems.

- Befinden sich alle Pole  $\lambda_i$  links der Imaginärachse ( $Re(\lambda_i) < 0$ ) ist das System asymptotisch stabil, ansonsten instabil (Abbildung 5.16).

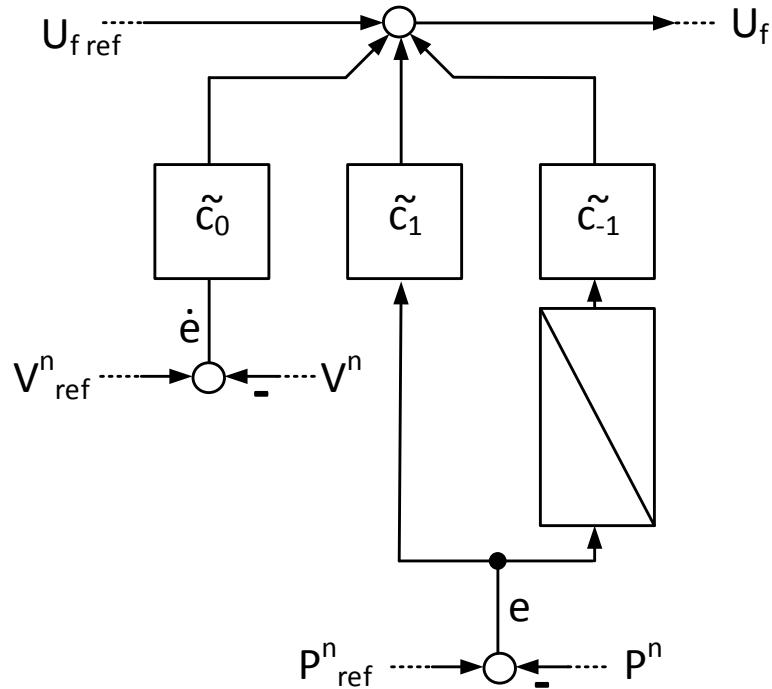
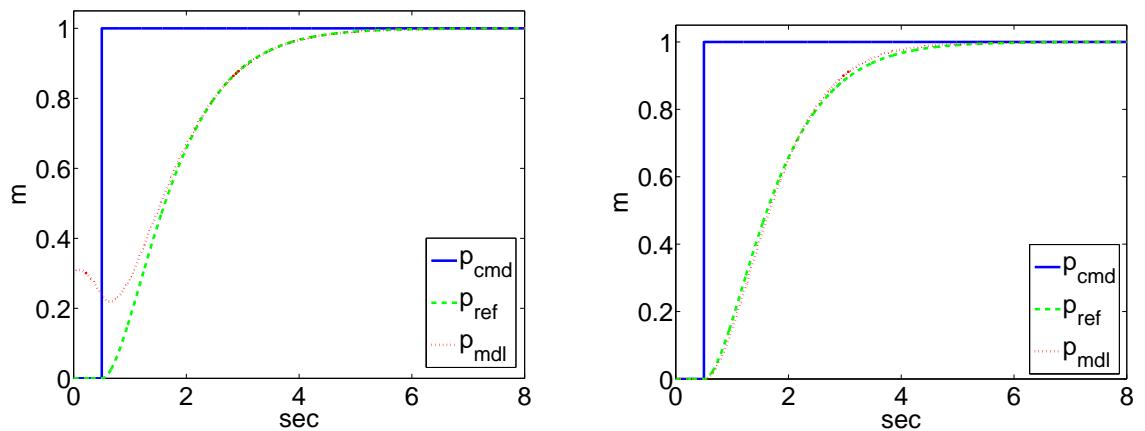


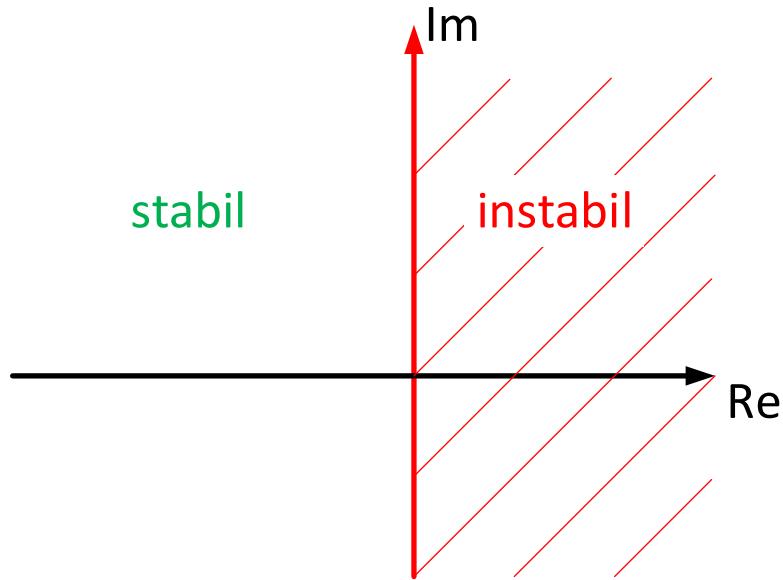
Abbildung 5.14: Folgeregler



(a) Inkonsistente Anfangszustände  
 $p_{ref0} = 0 \neq p_{mdl0} = 0.3$  und  
 $v_{ref0} = 0 \neq v_{mdl0} = 0.2$ , ideales  
Modell

(b) Unsicherheit der Modellparameter  $m_{inv} = 1.863 \neq m_{mdl} = 2.5$

Abbildung 5.15: Auswirkung von Folgeregler ( $\tilde{c}_1 = 4$ ,  $\tilde{c}_0 = 4$ ,  $\tilde{c}_{-1} = 0.01$ ) auf inkonsistente Anfangszustände und Modellunsicherheiten



**Abbildung 5.16:** Stabilitätsgebiet

- Komplexe Pole tauchen nur in Formen von Polpaaren auf. Ist ein komplexes Polpaar vorhanden, führt das System nach Anregung eine Schwingung aus. Befinden sich die Polpaare in der linken Halbebene des Imaginärteils, nimmt die Amplitude der Schwingung exponentiell ab.
- Je weiter links sich die Pole auf der reellen Achse befinden, desto schneller ist das System.
- Bei mehreren Polstellen wird das Verhalten hauptsächlich über den Pol bzw. das Polpaar mit dem größten Realteil bestimmt. Sie werden deshalb als dominante Pole bezeichnet.

Betrachtet man eine beliebige Übertragungsfunktion im Bildbereich.

$$G_e(s) = \frac{Z(s)}{N(s)} \quad (5.27)$$

Entsprechen die Polstellen  $\lambda_i$  des Systems den Nullstellen  $s_i$  des Nennerpolynoms.

$$N(s) = s^n + c_{n-1} \cdot s^{n-1} + \cdots + c_1 \cdot s + c_0 = \prod_n^{i=1} (s - \lambda_i) = 0 \quad (5.28)$$

Das dynamische Verhalten lässt sich anhand dieser Pollage analysieren. Im Umkehrschluss kann die Dynamik für frei wählbare Koeffizienten des Nennerpolynoms über die Vorgabe von Polstellen  $\lambda_{si}$ , über einen Koeffizientenvergleich, erfolgen.

$$N(s) = s^n + c_{n-1} \cdot s^{n-1} + \cdots + c_1 \cdot s + c_0 = \prod_n^{i=1} (s - \lambda_{si}) \quad (5.29)$$

Somit ist es möglich, über die Koeffizienten von (5.18) oder (5.24) das Einschwingverhalten des Folgereglers mittels der Platzierung von Polen vorzugeben. Wie jedoch die Polstellen optimal bestimmt werden, dafür gibt es keine generelle Vorgehensweise. In der Regel werden sie empirisch über Simulationen festgelegt, anschließend am realen Modell getestet und gegebenenfalls optimiert. Diese Optimierung geschieht meist händisch. Hier gibt es Bemühungen, diese letzte Optimierung am realen System zu automatisieren.

### 5.5.2 Automatische Optimierung der Reglerparameter

In der Simulation ermittelte Parameter müssen häufig für das reale System leicht angepasst werden, mit dem Ziel die Abweichung von der Referenztrajektorie möglichst gering zu halten. Von Vorteil ist es deshalb, einen Algorithmus zu implementieren, der online und kontinuierlich die Einstellung optimiert. Die Bezeichnung dieses Vorgangs heißt selftuning.

Der implementierte Folgeregler inklusive I-Anteil setzt sich aus einem Proportionalverstärker  $P$ , einem Differentiellen Anteil  $D$  sowie einem Integrationsanteil  $I$  zusammen.

$$u_f = \underbrace{u_{ref}}_{\text{Vorsteuerung}} + \underbrace{\tilde{c}_1 \cdot (\dot{P}_{ref}^n - \dot{P}^n)}_{D-\text{Anteil}} + \underbrace{\tilde{c}_0 \cdot (P_{ref}^n - P^n)}_{P-\text{Anteil}} + \underbrace{\tilde{c}_{-1} \int e(\tau) d\tau}_{I-\text{Anteil}} \quad (5.30)$$

Dadurch können Tuning-Algorithmen verwendet werden, die für einen PID-Regler entwickelt worden sind. Somit kann der unter [12] vorgestellte Algorithmus angewendet werden. Dieser

optimiert die Regelparameter über eine adaptive Interaktion. Die Interaktion besteht dabei aus drei Differentialgleichungen zur Anpassung der Koeffizienten des Reglers.

$$\begin{aligned}\dot{k}_p &= \dot{\tilde{c}}_0 = \gamma \cdot e^2 \\ \dot{k}_i &= \dot{\tilde{c}}_{-1} = \gamma \cdot e \cdot \int e(\tau) d\tau \\ \dot{k}_d &= \dot{\tilde{c}}_1 = \gamma \cdot e \cdot \dot{e}\end{aligned}\tag{5.31}$$

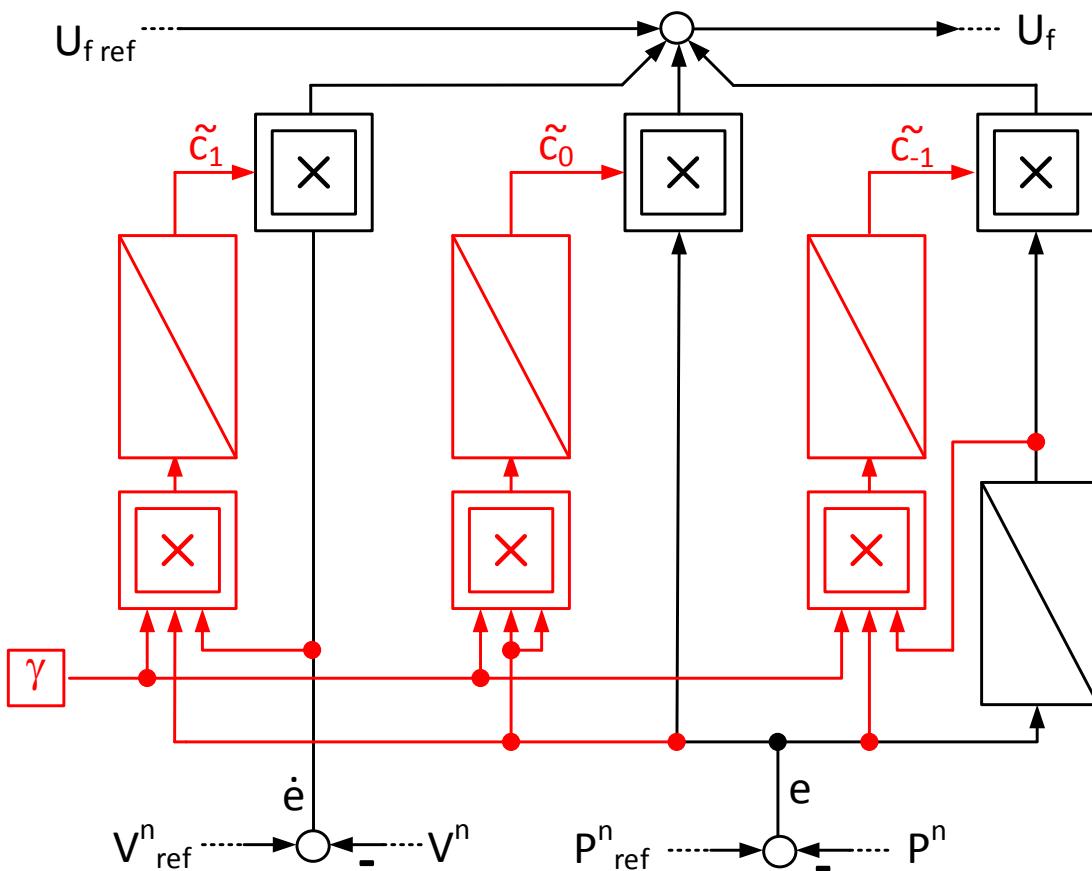
Dabei entspricht  $\gamma$  dem Anpassungskoeffizient. In [12] wird  $\gamma = 10$  empfohlen. Die Struktur, des sich daraus ergebenden Folgeregler mit Selbtoptimierung, ist in Abbildung 5.17 dargestellt. Anzumerken ist, dass die Qualität der Optimierung abhängig von den zu Beginn übergebenen Parametern ist. Die zuvor zum Beispiel über die Polvorgabe bestimmten werden.

Im beschränkten Zeitraum der Arbeit konnte dieser Algorithmus nicht vollständig untersucht und umgesetzt werden. Die im beschrieben Paper [12] veröffentlichten Ergebnis des Optimierungsalgorithmus legen jedoch eine weiterführende Untersuchung dieses Teilbereichs im Anschluss dieses Ausarbeitung nahe.

## 5.6 Zustandsschätzung

Bei der Folgeregelung handelt es sich um einen Zustandsregler. Betrachtet man das Stellgesetz (5.26) bedeutet dies, dass die Position  $P^n$  und die Geschwindigkeit  $v^n$  mit der sich der Quadrocopter im n-frame bewegt zur Verfügung stehen müssen. Bereits bekannt ist, dass die Position  $P^n$  mittels des Lasers (Kapitel 8.2) bestimmt wird. Die Updaterate beträgt dabei nur 40Hz. Daher ist es sinnvoll Zwischenwerte zu schätzen, sodass die Positionsdaten dem Folgeregler in einer höheren Taktrate zur Verfügung stehen.

Der Zustand der Geschwindigkeit  $v^n$  ist messtechnisch sehr schwer zu ermitteln. Eine Möglichkeit zur messtechnischen Erfassung ist die Integration eines Dopplerradars [20] auf dem Quadrocopter, der auf dem Doppler-Effekt basiert. Dabei wird ein periodisches Signal ausgesendet und je nach Geschwindigkeit verkleinert sich die Periodendauer, bzw. vergrößert sich für negative Geschwindigkeiten. So kann anhand der Frequenzänderung des reflektierten Signals auf die Geschwindigkeit geschlossen werden. Diese Umsetzung bedeutet die Integration eines weiteren Sensor. Aus Gewichts- und Kostengründen sind deshalb



**Abbildung 5.17:** Folgeregler inklusive Selftuning

Methoden gefragt, die über die vorhandene Sensorik (IMU und Laser) die Geschwindigkeit des Quadrocopters feststellen.

### 5.6.1 Geschwindigkeitsbestimmung über die Inertialsenorik

Die *IMU* beinhaltet einen 3D-Bewegungssensor. Dieser nimmt die Beschleunigung in x, y und z- Achse des Quadrocopters auf und stellt die Messwerte mit einer Frequenz von 1 kHz ( $T_{imu} = 1 \text{ ms}$ ) zur Verfügung. Aufgrund des Weg-Zeit-Gesetzes (5.5) lässt sich die Geschwindigkeit über die Integration der Beschleunigung bestimmen. Für den diskreten

Fall lässt sich diese mittels des Euler-Verfahrens approximieren. Zuvor werden dafür die Beschleunigungsmesswerte ins n-frame transformiert (Gleichung (9.6)).

$$v_{k+1}^n = v_k^n + T_{imu} \cdot (M_{nb} \cdot a_k^b) = v_k^n + T \cdot a_k^n \quad (5.32)$$

Auf diese Weise ist die Geschwindigkeit bestimmt. In der Praxis führt diese Methode allerdings zu keinem guten Ergebnis. Grund dafür ist das Sensorrauschen  $r_{a_k}$  sowie der Bias b, der trotz Initialisierung nicht vollständig zu eliminieren ist. Damit entspricht der Messwert  $a_k^b$  nicht der tatsächlichen Beschleunigung  $a_{k_{tat}}^b$ .

$$a_k^b = a_{k_{tat}}^b + r_{a_k} + b \quad (5.33)$$

Das Problem ist, der Fehler wird mit integriert. Damit driften geschätzte und reale Geschwindigkeit auseinander. Dadurch ist eine Geschwindigkeitsbestimmung ausschließlich über die Inertialsensorik nicht möglich.

### 5.6.2 Geschwindigkeit als Ableitung der Position

Ebenfalls auf Basis des Weg-Zeit-Gesetzes (5.5) bestimmt sich die Geschwindigkeit über die Ableitung der Quadrocopterpositionen. Da es sich bei den Positionsdaten um diskrete Werte handelt, lässt sich die Ableitung anhand des Euler-Verfahrens (5.34) approximieren.

$$P_k^n = P_{k-1}^n + T_l \cdot v_k^n \quad (5.34)$$

Für die Geschwindigkeit ergibt sich

$$v_k^n = \frac{P_k^n - P_{k-1}^n}{T_l} \quad (5.35)$$

Die Abtastzeit  $T_l = 25 \text{ ms}$  ( $f_l = 40 \text{ Hz}$ ). Sie entspricht der Updaterate der Umgebungs-scans des Lasers mit denen die Algorithmen zur Positionsbestimmung gespeist werden. Bei der Approximation der Geschwindigkeit ist zu beachten, dass das Positionssignal aufgrund von Sensorrauschen von Laser und Gyroskop sowie der Varianz des ICP-Algorithmus eine

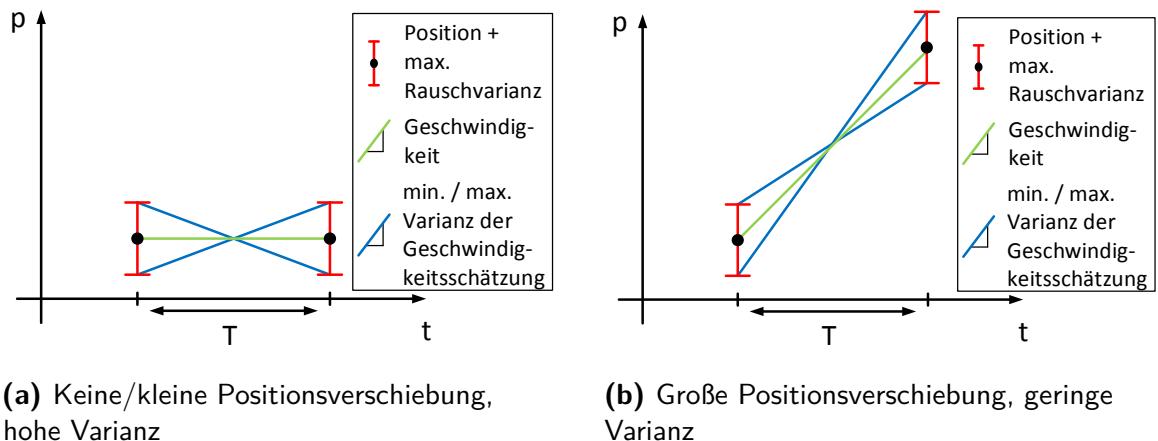
Abweichung vom tatsächlichen Positions Wert  $P_{k_{tat}}^n$  aufweist. Diese Abweichung ist nicht konstant und ist als Positionsräuschen  $r_{P_k}$  darzustellen.

$$P_k^n = P_{k_{tat}}^n + r_{P_k} \quad (5.36)$$

Dieses Positionsräuschen  $r_P$  überträgt sich auf die Geschwindigkeit.

$$v_k^n = \frac{P_k^n - P_{k-1}^n}{T_l} + \frac{r_{P_k} - r_{P_{k-1}}}{T_l} \quad (5.37)$$

Diese führt nicht dazu, dass wie bei der Geschwindigkeitsbestimmung mittels der Beschleunigungssensoren, der Schätzwert und Realwert auseinander divergieren. Nichtsdestotrotz besitzt das Rauschen einen Einfluss auf die Qualität der Geschwindigkeitsberechnung. So fällt bei niedrigen Geschwindigkeiten der Abstand zwischen zwei Positions Werten geringer aus. Der Betrag des Rauschanteils bleibt konstant. Dies führt vor allem bei Schwebeflügen sowie Flügen mit geringer Dynamik zu einer erhöhten Unsicherheit der Schätzung (Abbildung 5.18a). Zwar zeigt die Grafik 5.18b, dass der Einfluss des Rauschen auf die Varianz der errechneten Geschwindigkeit (5.37) mit der Größe der Positionsverschiebung abnimmt. Ungeachtet dessen ist es gerade notwendig für Bewegungen mit geringer Geschwindigkeit den Einfluss des Positionsräuschens zu minimieren. Geht man davon aus, dass die Frequenz des Rauschanteils oberhalb des Frequenzbandes der Positionsänderungen liegt, kann das



**Abbildung 5.18:** Auswirkung der Größe der Positionsverschiebung innerhalb eines Schätzwertes auf die Varianz der Geschwindigkeitsschätzung

Nutzsignal mittels eines Tiefpassfilters vom Rauschanteil getrennt werden. Das Tiefpassfilter lässt sich dabei mittels eines rekursiven *Infinite Impulse Response (IIR)*-Filters realisieren [8]. Die gefilterte Geschwindigkeit  $\hat{v}_k^n$  hängt dabei nicht nur von den vorherigen Messwerten  $v^n$ , sondern auch von den vorherigen Filterergebnissen  $\hat{v}^n$  ab.

$$\hat{v}_k^n = \sum_{j=0}^n a_j \cdot v_{k-j}^n + \sum_{i=1}^n b_i \cdot \hat{v}_{k-i}^n \quad (5.38)$$

Mittels der Koeffizienten  $a_j$  und  $b_i$  wird das gewünschte Filterverhalten, zum Beispiel das eines Butterworth-Filters, mit der benötigten Grenzfrequenz realisiert. Die Ordnung  $n$  des Filter beschreibt wie viele der vorangegangenen Messwerten in die Filterung mit einzubeziehen sind. Beim Design des Filters werden jedoch Kompromisse zwischen Zeitverzögerung, Phasenverzerrung, Dämpfung und Stoppbandeigenschaften getroffen. Für die Folgeregelung ist es dabei wichtig, dass die beiden erstgenannten Kriterien möglichst gering ausfallen. Ist dies nicht der Fall, wirkt es destabilisierend auf den Regler.

Des Weiteren lassen sich Aufgrund der geringen Updaterate von 40Hz der Positionsdaten, die Dynamik des Quadrocopters und das Positionsrauschen spektral nicht eindeutig voneinander trennen. Unterdrückt man das Positionsrauschen, werden auch schnelle Positionsänderungen des Quadrocopters weg gefiltert. Für die hohe Dynamik des Quadrocopters ist die Anwendung eines Tiefpassfilter mit Grenzfrequenz nicht gangbar. Gesucht ist eine Methode, die für Flüge mit geringer Dynamik die Varianz der Geschwindigkeitsschätzung, verursacht durch Positionsrauschen, möglichst stark verringert. Gleichzeitig jedoch der hohen Dynamik des Quadrocopters Folge leisten kann.

### 5.6.3 Geschwindigkeitsbestimmung über die Methode First-Order Adaptive Windowing

Wie in Kapitel 5.6.2 erläutert, nimmt die Varianz der Geschwindigkeitsschätzung ab, je weiter die Positionsveränderungen auseinander liegen. Hierfür ist auf Grafik 5.18b verwiesen. Der gleiche Effekt tritt auf, je mehr Abtastschritte  $n$  der für die Euler-Approximation verwendete Bezugspunkte  $P_{k-n}^n$  in der Vergangenheit liegen.

$$\hat{v}_k^n = \frac{P_k^n - P_{k-n}^n}{nT_l} = \frac{1}{n} \sum_{j=0}^{n-1} \hat{v}_{k-j}^n \quad (5.39)$$

Dabei ist die Verwendung der Bezugsposition  $P_{k-n}^n$  gleichzusetzen mit der Mittlung der letzten  $n$  Geschwindigkeitsschätzungen  $(v_k^n, v_{k-1}^n, \dots, v_{k-n}^n)$  mittels (5.35). Deshalb spricht man auch von Fensterung beziehungsweise Windowing. Siehe zur Veranschaulichung Abbildung 5.19. Vergrößert man das Fenster, besitzt das eine äquivalente Wirkung wie die Reduzierung der Abtastrate. Das Windowing verhält sich wie ein Filter. So führt ein großes Fenster bei geringen Dynamiken zu einer sehr präzisen Geschwindigkeitsschätzung durch Rauschunterdrückung. Schätzungen für hochfrequente Bewegungen des Quadrocopter werden jedoch stark gedämpft und zeitverzögert ausgegeben, womit die Verlässlichkeit der Schätzung abnimmt. Es wirkt ähnlich wie das Filter (5.38). Allerdings mit dem Vorteil, dass das Filterverhalten nur über einen Parameter einstellbar ist, der Fenstergröße  $n$ . So sollte das Window klein sein, wenn der Quadrocopter hochfrequente Änderungen der Bewegung vornimmt, damit die abrupten Geschwindigkeitsänderungen möglichst ungedämpft erfasst werden. Bei geringen Dynamiken soll die Fenstergröße jedoch zunehmen, wodurch der Einfluss des Positionsrauschens minimiert werden soll. Dabei erfolgt die Anpassung der Fenstergröße online und für jeden Punkt abhängig der Achse des n-frame. Im Paper [10] wird dafür die Methode des *FOAW* vorgestellt. Diese besagt, existiert eine Grade zwischen den Positions値en  $p_k^n$  und  $p_{k-n}^n$  einer Achse, die alle  $n$  Punkte innerhalb einer Toleranz  $\pm d$  schneidet, stellt die Steigung dieser und somit die Geschwindigkeit einen optimalen Kompromiss zwischen Rauschunterdrückung (Präzision) und Verlässlichkeit dar. Die möglichen Schätzwerte für die Geschwindigkeit, abhängig von der Fenstergröße  $n$  und der Toleranz  $d$ , sind im folgenden Datensatz dargestellt.

$$\hat{v}_k \in \left[ \frac{p_k^n - p_{k-n}^n}{nT_l} - \frac{2d}{nT_l}, \frac{p_k^n - p_{k-n}^n}{nT_l} + \frac{2d}{nT_l} \right] \quad (5.40)$$

Die Toleranz  $d$  ist dabei durch den Betrag der maximalen Rauschabweichung festgelegt und als Konstante zu sehen.

$$d = ||r_{max}|| \quad (5.41)$$

Zu erkennen ist, dass die Varianz der Schätzung für eine steigende Fenstergröße  $n$  abnimmt. Somit ist ausgehend von der aktuellen Position  $p_k^n$  die maximale mögliche Fenstergröße  $n = \max\{1, 2, 3, \dots\}$  zu ermitteln, für die gilt,

$$|p_{k-i}^n - \hat{p}_{k-i}^n| \leq d \quad \forall i \in \{1, 2, \dots, n\} \quad (5.42)$$

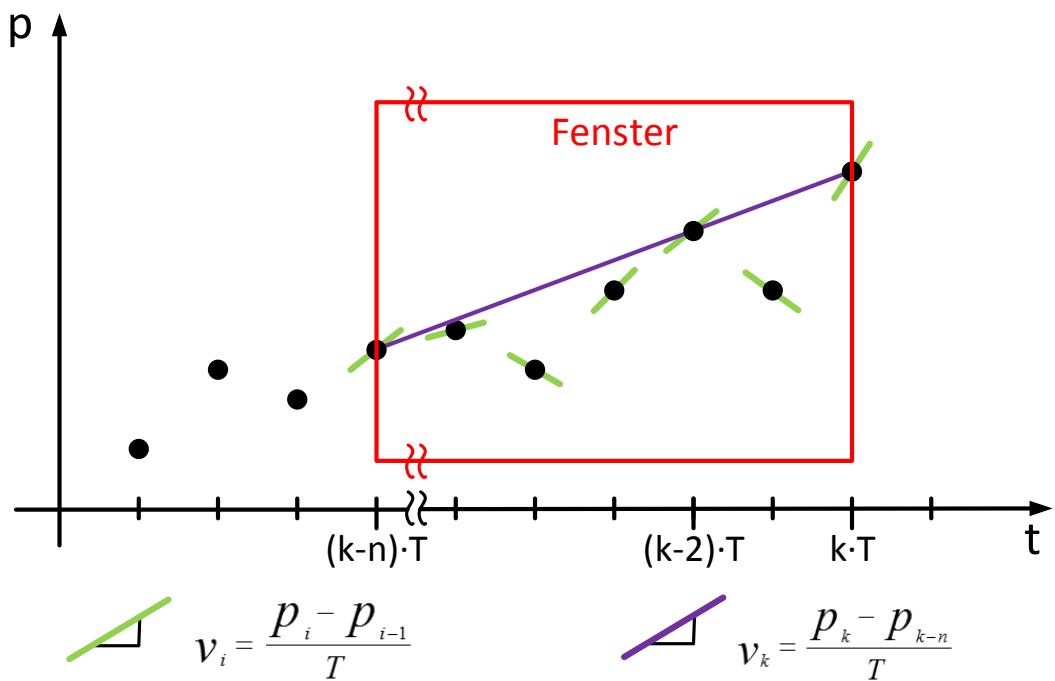


Abbildung 5.19: Fensterung

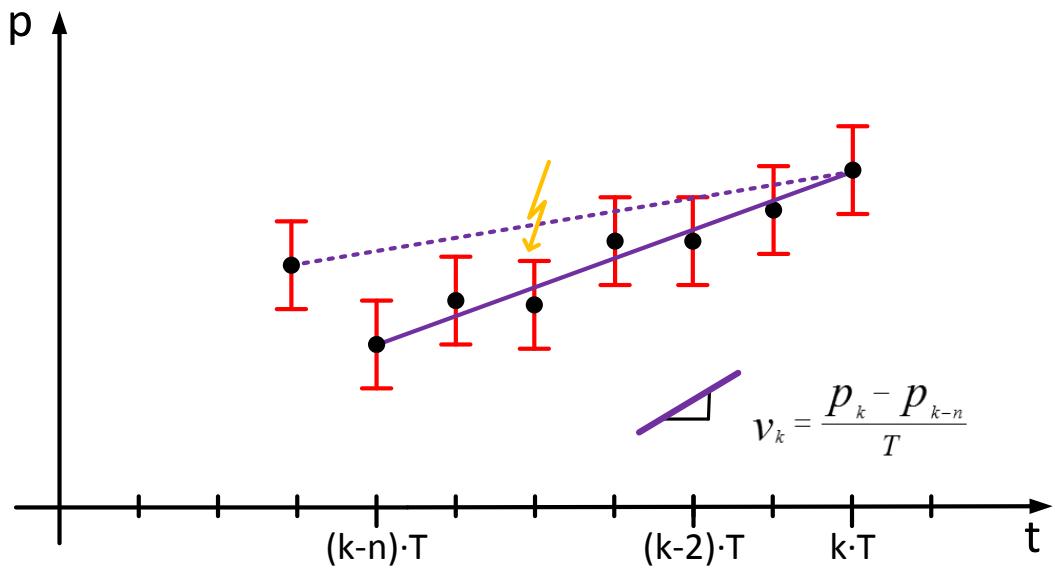


Abbildung 5.20: End-fit First-Order Adaptive Windowing (FOAW)

Dabei entspricht  $\hat{p}_{k-i}^n$  den Punkten auf der approximierenden Geradengleichung.

$$\hat{p}_{k-i}^n = a_n + b_n \cdot (k - i) T_l \quad (5.43)$$

In [10] ist dazu ein iterativer Algorithmus zur Lösung dieses Problems beschrieben. Dabei sind die Parameter der Geradengleichung (5.43) wie folgt parametriert.

$$a_n = \frac{k \cdot p_{k-i}^n + (n - k) p_k^n}{n} \quad (5.44)$$

$$b_n = \frac{p_k^n - p_{k-n}^n}{n T_l} \quad (5.45)$$

Da die Steigung  $b_n$  aus den zwei Rändern des Fensters gebildet wird, spricht man von der End-fit-FOAW.

Der Ablauf des Iterations-Algorithmus sieht wie folgt aus.

- **Schritt 1:** Man verschiebt die Zeitachse so, dass für den Zeitpunkt des neusten Wertes  $p_k^n$  gilt  $t_k = k \cdot T_l = 0$ . Die Folge ist eine vom Faktor  $k$  unabhängige Geradengleichung.

$$\hat{p}_{k-i}^n = a_n + b_n \cdot (-i) \cdot T_l \quad (5.46)$$

Für die gilt,

$$a_n = p_k^n \quad (5.47)$$

und  $b_n$  weiterhin über (5.44) berechnet wird. Vorteil, es muss jeweils nur die Steigung neu berechnet werden.

- **Schritt 2:** Fenstergröße  $j = 1$ .
- **Schritt 3:** Berechnung des Parameters  $b_j$  (5.45) der Geradengleichung (5.46) in Abhängigkeit der Fenstergröße  $j$ .
- **Schritt 4:** Überprüfen ob die berechnete Gerade alle im Fester befindlichen Punkte  $p_{k-i}^n, i \in \{1, 2, \dots, j\}$  innerhalb der Toleranz passiert. Bedingung (5.42)

- **Schritt 5:** Ist Schritt 4 erfüllt, inkrementiere die Fenstergröße  $j = j + 1$  und gehe zu Schritt 2. Wenn Bedingung nicht erfüllt ist, ist die Steigung der vorhergegangen Geradengleichung als Schätzwert der Geschwindigkeit auszugeben.

$$\hat{v}_k = b_{j-1} = b_n \quad (5.48)$$

Die maximale Fenstergröße  $n$  entspricht für Position  $p_k^n$  somit  $n = j - 1$

Dieser Vorgang wird für jede Aktualisierung  $k = k + 1$  des Positions Wert  $p_k^n$  neu gestartet.

Die End-fit-FOAW verwendet für den Schätzvorgang die zwei Endpunkte des Fensters. In [10] wird deshalb zur Verbesserung die Best-fit FOAW vorgestellt. Dabei wird die Steigung über die Methode der Summe kleinsten quadratischen Fehler, bestimmt. Das bedeutet für die zeitnormierte Geradengleichung ist eine Steigung  $b_n$  gesucht, für die die Summe der Fehlerquadrat  $e^2$  ein Minimum aufweist.

$$e_{min}^2 = \min \sum_{i=0}^n (p_{k-i}^n - \hat{p}_{k-i}^n)^2 \quad (5.49)$$

mit

$$\hat{p}_{k-i}^n = p_k^n - b_n \cdot i \cdot T_l \quad (5.50)$$

Um das Minimum der Fehlergleichung (5.49) in Abhängigkeit der Steigung bestimmen zu können, ist diese nach  $b_n$  abzuleiten. Da es sich um eine quadratische Gleichung handelt, weist der Fehler einen Tiefpunkt auf, wenn diese Ableitung Null ergibt.

$$\frac{de^2}{db_n} = 0 \quad (5.51)$$

Löst man unter dieser Bedingung die Ableitung nach  $b_n$  auf. Erhält man die Steigung die die Summe der kleinsten quadratischen Fehler aufweist.

$$b_n = \frac{(n \sum_{i=0}^n y_{k-i} - 2 \sum_{i=0}^n i \cdot y_{k-i}) \cdot 6}{T_l n(n+1)(n+2)} \quad (5.52)$$

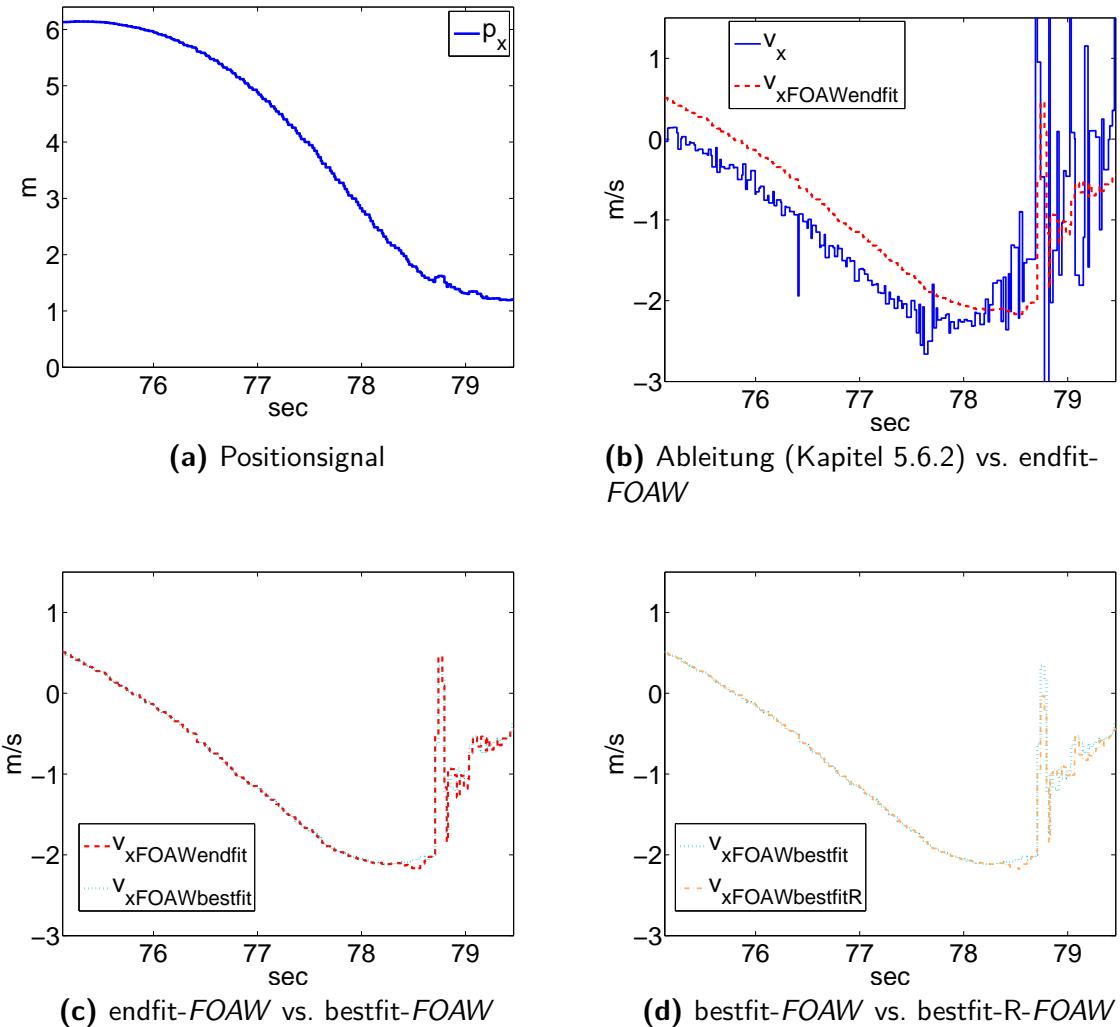
Nun ist die Steigung allgemeingültig für variable Fenstergrößen bestimmt. Die Umsetzung der best-fit-FOAW in der Software erfolgt nach dem gleichen Schema wie bei der end-fit-FOAW. Einziger Unterschied, zur Berechnung der Steigung  $b_n$  wir die Gleichung (5.52) verwendet.

Das Ergebnis ist eine präzisere Schätzung bei gleichbleibender Zuverlässigkeit der Messung. Beide Methoden sind anfällig gegen sogenannte Outlier. Einzelne Signalwerte, die einen höheren Fehler als die maximale durch das Rauschen verursachte Abweichung  $d$  aufweisen. Outliner lassen sich sehr einfach durch ein robustes Abbruchkriterium in Schritt 4 bekämpfen. So gilt die Bedingung (5.42) erst dann für nicht erfüllt, wenn zwei aufeinanderfolgende Werte dem Kriterium nicht gerecht sind. Erst dadurch wird die Vergrößerung des Fensters gestoppt. Diese Variante nennt sich best-fit-FOAW-R.

Die Methoden des FOAW bieten eine sehr gute Möglichkeit die Geschwindigkeit auf Basis eines verrauschten Positionssignals zu ermitteln. Problematisch ist allerdings die geringe Abtastrate von  $40Hz$  des Lasers. Schon bei Flügen mit mittlerer Dynamik führt das dazu, dass die Fenstergröße stark reduziert wird. Somit auch der Einfluss des Positionsrauschens stark zunimmt. Zu sehen ist dies auch in Abbildung 5.21 wo die Methoden des FOAW anhand einer realen Messung miteinander verglichen werden. Als Folge kann selbst dieses Konzept, limitiert durch die Tastrate des Lasers, der Dynamik des Quadrocopters nicht folgen. Es ist klar, dass eine Methode benötigt wird, die die Position und die Geschwindigkeit in einer höheren Taktrate zur Verfügung stellt. Gleichzeitig müssen beide Signale ausreichend rauschunterdrückt sein.

#### 5.6.4 Bestimmung der Zustände mittels eines Fusionsfilter

Wie aus den vorherigen Unterkapitel zur Zustandsschätzung erkennbar, wird keine der Methoden denen zuvor gestellten Ansprüche gerecht. So erfüllt die Integration der Beschleunigung (Kapitel 5.6.1) zwar das Kriterium zur Erhöhung der Taktrate, da die Beschleunigungsmesswerte in einem Takt von  $1kHz$  zur Verfügung stehen. Durch den Bias des Beschleunigungssensors divergieren jedoch innerhalb kürzester Zeit die geschätzten Zustände von den realen Zuständen. Dieses Phänomen tritt bei den Anwendungen Kapitel 5.6.2 und Kapitel 5.6.3 nicht auf. Der Grund, sie beziehen sich auf die vom Laser berechneten Positionswerte. Die Abweichungen dieser Zustandswerte befinden sich in einem konstanten Rahmen um die tatsächliche Position. Dieses Positionsrauschen sorgt bei der Geschwindigkeitsschätzung über die Ableitung der Positionswerte dafür, dass das errechnete Geschwindigkeitssignal ebenfalls einen Rauschanteil aufweist. Diesen Rauschanteil zu minimieren ist Ziel der FOAW Methode. Aufgrund der geringen Abtastrate von  $40Hz$  kann auch dieser Ansatz der hohen Dynamik nicht Folge leisten. Schon für geringe Dynamiken ist die Fenstergröße stark dezimiert und damit auch die Rauschunterdrückung.



**Abbildung 5.21:** Vergleich der Filterergebnisse für FOAW

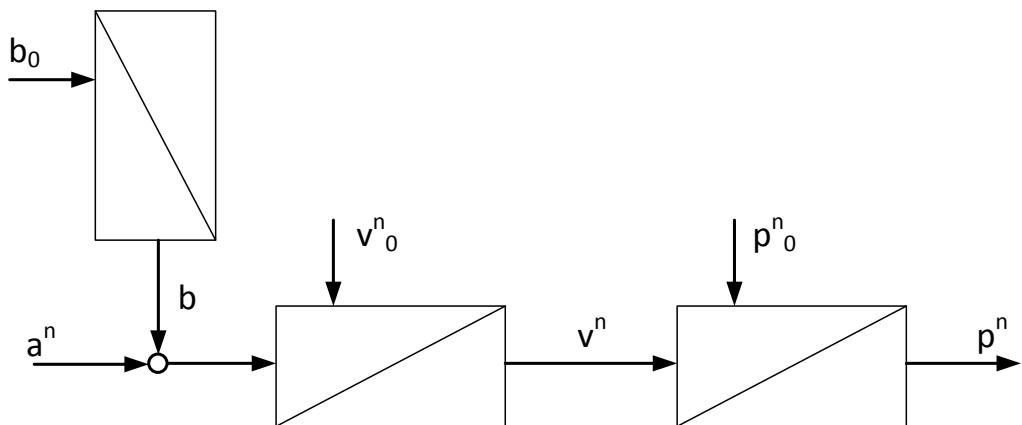
Das Ergebnis, Beschleunigungssensor der *IMU* und Laser bieten jeweils für sich keine zuverlässigen Zustandswerte. Abhilfe ist die Fusion beider Sensordaten mittels eines Fusionsfilters. Dabei sorgen die Beschleunigungswerte mit ihrer hohen Update rate von  $1\text{Khz}$  für eine schnelle Reaktionsfähigkeit und tragen somit der Dynamik des Quadrocopters sorge. Die Positionsdaten des Laser indes verhindern ein Abdriften der Schätzwerte. Als Folge stellen sie die Zuverlässigkeit der Fusionsergebnisse sicher.

Das auf dem *HLP* implementierte Fusionsfilter basiert auf einem Luenberger Beobachter[3]. Dieser besteht aus einem linearen Streckenmodell, dessen Zustände  $\hat{x}$  mittels eines Korrekturterms ( $L \cdot (y - \hat{y})$ ) auf die Zustände  $x$  der Regelstrecke einschwingen. Die Zustands-

differentialgleichung eines Luenberger Beobachter stellt sich in allgemeiner Form [17] wie folgt dar.

$$\begin{aligned}\dot{\hat{x}} &= A \cdot \hat{x} + L \cdot (y - \hat{y}) + B \cdot u \\ \hat{y} &= C \cdot \hat{x}\end{aligned}\quad (5.53)$$

Dabei entspricht  $A$  der Dynamikmatrix,  $B$  der Eingangsmatrix und  $C$  der Ausgangsmatrix des linearen Streckenmodells. Aufgrund der Inversion ist das Bewegungsmodell für jede Achse im n-frame als Integrationsglied 2. Ordnung gegeben (Abbildung 5.8). Dabei entsprechen die Eingangsgrößen  $u$  den ins n-frame transformierten Beschleunigungswerten der *IMU*. Da die Ketten entkoppelt sind, muss der Beobachter nicht für alle drei Ketten gleichzeitig entworfen werden. Es ist ausreichend ein Integrationsglied 2. Ordnung als Streckenmodell heranzuziehen. Weiterhin ist bekannt, dass die Beschleunigungswerte der *IMU* einen Bias  $b$  aufweisen. Um diese Störgröße zu berücksichtigen kann das Streckenmodell nach [6] wie in Abbildung 5.22 erweitert werden. Dieses erweiterte Modell ist über die Zustandsdifferentialgleichung (5.54)



**Abbildung 5.22:** Erweitertes Streckenmodell

beschrieben

$$\begin{aligned}\dot{x} &= \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \cdot x + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \cdot u \\ &\quad A \qquad \qquad \qquad B \\ y &= \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \cdot x \\ &\quad C\end{aligned}\tag{5.54}$$

mit,

$$\hat{x} = [p^n \ v^n \ b]^T\tag{5.55}$$

Anhand von (5.53) folgt daraus für den Beobachter.

$$\begin{aligned}\dot{\hat{x}} &= \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \cdot \hat{x} + \begin{bmatrix} l_1 \\ l_2 \\ l_3 \end{bmatrix} \cdot (y - \hat{y}) + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \cdot u \\ &\quad A \qquad \qquad \qquad L \qquad \qquad \qquad B \\ \hat{y} &= \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \cdot \hat{x} \\ &\quad C\end{aligned}\tag{5.56}$$

mit,

$$\hat{x} = [\hat{p}^n \ \hat{v}^n \ \hat{b}]^T\tag{5.57}$$

Vorteil des erweiterten Beobachters (Abbildung 5.23) ist, dass der Bias des Beschleunigungssensors ebenfalls geschätzt wird. Das sorgt dafür, dass die Zuverlässigkeit der für die Folgeregelung benötigten Zustände  $\hat{v}$  und  $\hat{p}$  steigt. Das Einschwingverhalten mit dem die Beobachterwerte  $\hat{x}$  auf die realen Zustände  $x$  konvergieren, lässt sich über die Beobachtermatrix  $L$  bestimmen. Da der Zustandsfehler  $\tilde{x} = x - \hat{x}$  gilt, ergibt sich für die Dynamik des Zustandsfehlers

$$\dot{\tilde{x}} = \dot{x} - \dot{\hat{x}} = (A - LC) \cdot \tilde{x}\tag{5.58}$$

Um diese mittels Polvorgabe einstellen zu können, muss daraus das charakteristische Polynom gebildet werden [17].

$$N(s) = \det(A - LC)\tag{5.59}$$

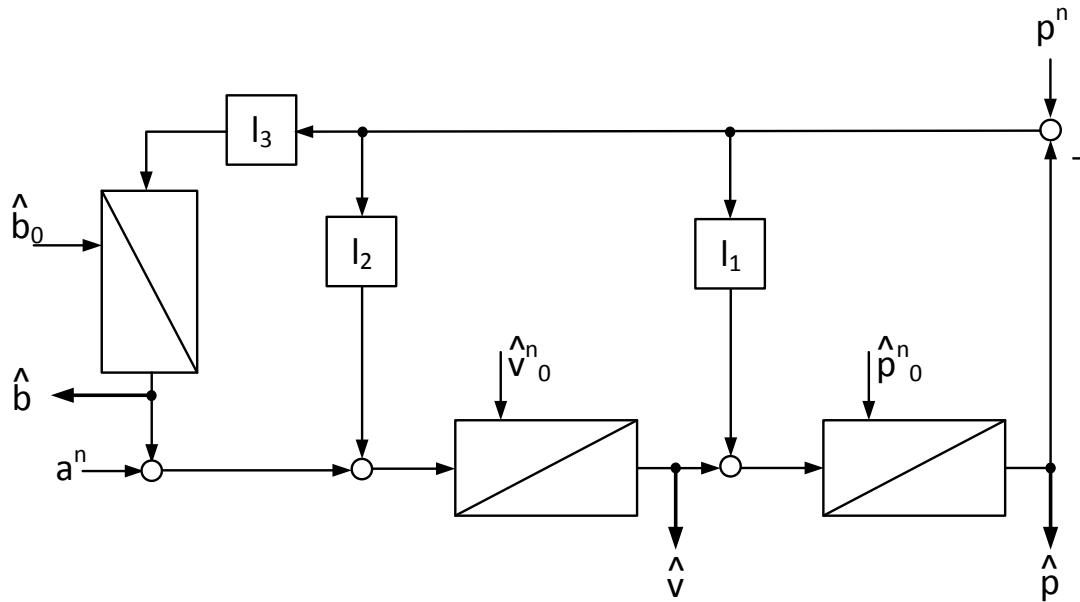


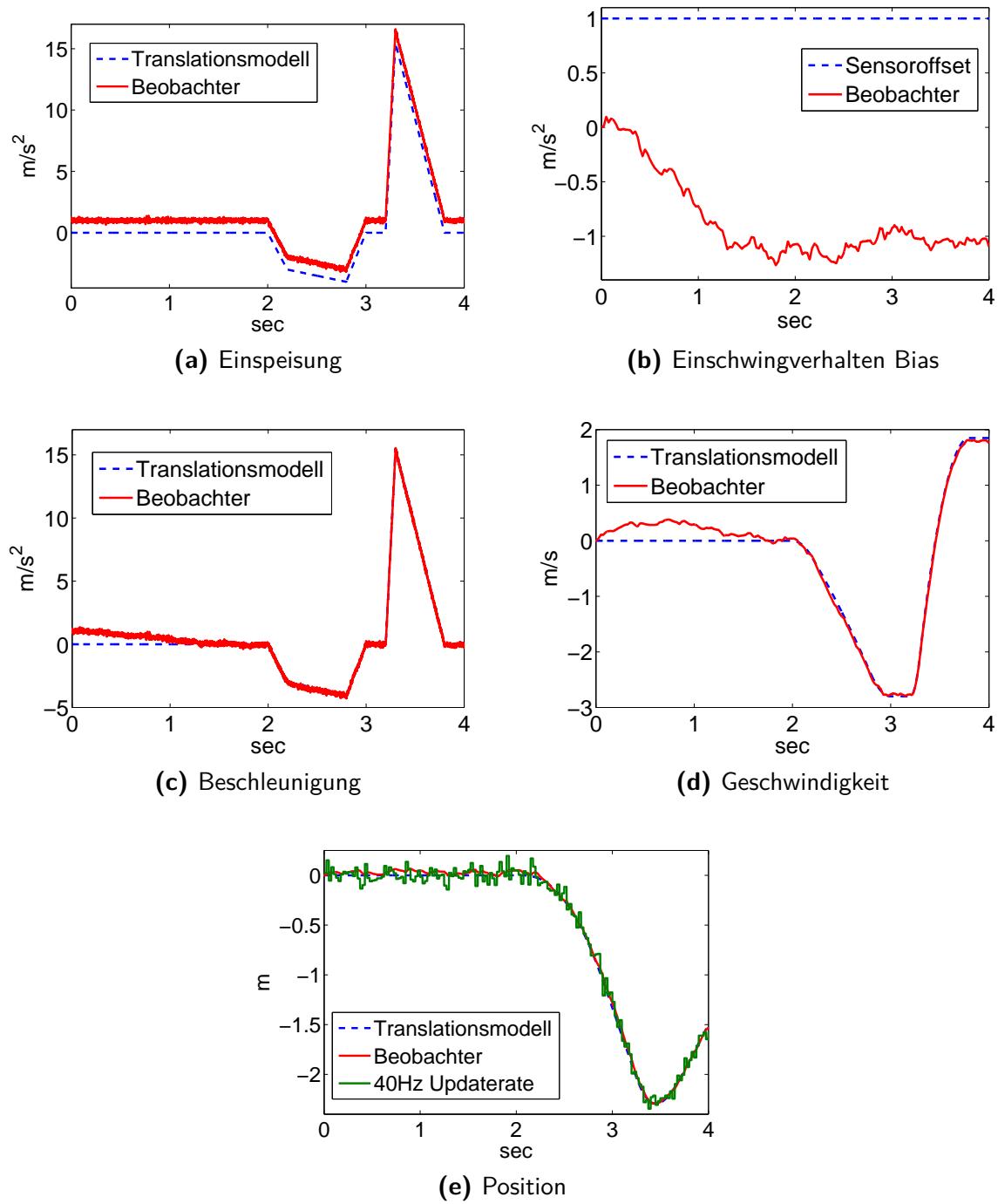
Abbildung 5.23: Strukturbild Beobachter

ergibt für das erweiterte Zustandsmodell (5.56),

$$N(s) = s^3 + l_1 \cdot s^2 + l_2 \cdot s + l_3 \quad (5.60)$$

Anhand dieses Polynom sind Koeffizienten der Beobachtermatrix  $L$  nach Kapitel 5.5.1 über Polvorgabe vorgebar.

Um Beispielhaft das Einschwingverhalten des Fusionsfilters für eine Beobachtermatrix mit  $L = [18 \ 46 \ 3.75]^T$  aufzeigen zu können, wird das zustandslinearisierte Streckenmodell mit einem variierenden Beschleunigungssignal gespeist. Um dem Beobachter ein realistischen Beschleunigungswert zur Verfügung zu stellen, wird diesem ein Signal mit überlagerten Rauschen und einem Bias zugeführt (Abbildung 5.24a). Auch das Positionsergebnis des Streckenmodells wird für das Fusionsfilter mit einem Rauschen beaufschlagt. Zusätzlich wird dieses verrauschte Positionssignal mit einer Frequenz von 40 Hz abgetastet, die der Abtastrate des Lasers entspricht. Mit diesen störbehafteten Eingangsgrößen wird der Beobachter gespeist. Abbildung 5.24b zeigt, wie der geschätzte Bias  $\hat{b}$  auf den invertierten Wert des Beschleunigungsoffset einschwingt. Auf diese Weise wird der Bias  $b$  kompensiert (Abbildung 5.24c). In Abbildung 5.24d ist deutlich zu erkennen, welchen Einfluss



**Abbildung 5.24:** Simulationsergebnisse für das Fusionsfilter

dies auf die Qualität der Geschwindigkeitsschätzung nimmt. Im Gesamtergebnis stellt die Datenfusion dem Folgeregler ein rauschreduziertes Geschwindigkeits- (Abbildung 5.24d) als auch Positionssignal (Abbildung 5.24e) zur Verfügung, das obendrauf noch die Daten mit einer Taktrate von  $1 \text{ kHz}$  bereit stellt. In Sachen Zuverlässigkeit und Genauigkeit ist diese Methode allen vorher genannten Methodiken überlegen. Womit deren Sinnhaftigkeit zur Zustandsschätzung im Rahmen des asctec\_hl\_framework dargelegt ist. Im Ergebnis ist damit der letzte Bestandteil der Positionsregelung hergeleitet und verifiziert.

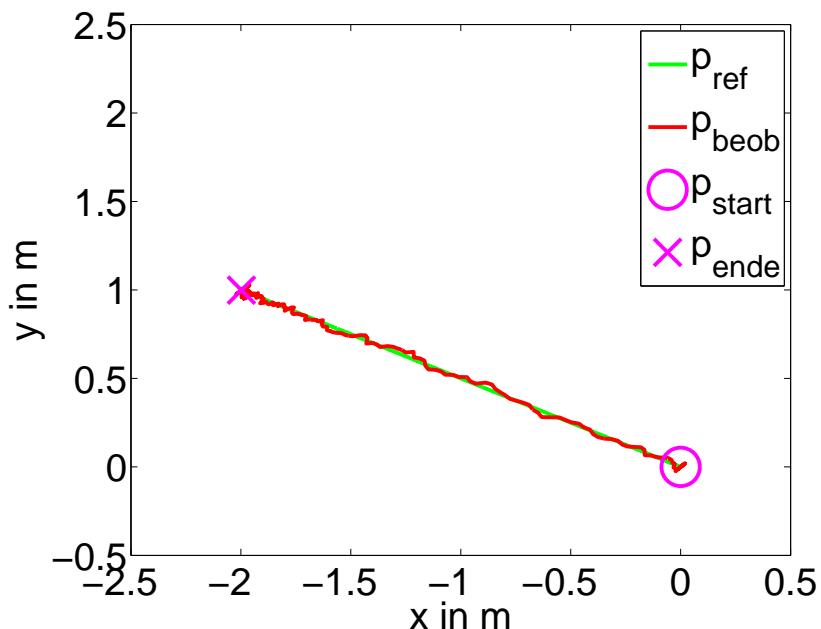
## 5.7 Simulation der Positionsreglung

In den vorangegangen Unterkapiteln 5.3 bis 5.6 wurden die einzelnen Komponenten der Positionsregelung durchleuchtet, deren Herleitung aufgezeigt und ihr Richtigkeit mittels eines Simulationsbeispiels dargelegt. Aufgabe dieses Kapitels ist es, die Funktionsfähigkeit des Gesamtsystems mittels einer Simulation zu beweisen.

Der Simulationsaufbau basiert auf dem in Kapitel 5.2 eingeführten Bewegungsmodell des Quadrocopters. Um die Simulation möglichst realistisch zu gestalten sind die Ausgangsgrößen denen der Realität angenähert. So sind die Beschleunigungswerte als auch die Positionsdaten mit einem weißen Rauschen überlagert. Darüber hinaus, wird die Postion nur in einem diskreten Zeitintervall von  $T_l = 25 \text{ ms}$  aktualisiert. Aus diesen Ausgangswerten interpoliert das Fusionsfilter die Positionsdaten und bestimmt den Zustand der Geschwindigkeit. Parametriert ist der Beobachter (Gleichung (5.56)) mit  $L = \begin{bmatrix} 18 & 46 & 3.75 \end{bmatrix}^T$ . Die Eingangsgrößen des Bewegungsmodells bestehen aus der Schubvorgabe sowie den Winkeln zur Beschreibung der Orientierung ( $u = \begin{bmatrix} T & \phi & \theta & \psi \end{bmatrix}^T$ ). Daran angeschlossen befindet sich die Inversion (Kapitel 5.3), welche das System zustandslinearisiert, so dass als neue Eingangsgrößen die Beschleunigungsgrößen in Richtung der n-frame Koordinatensysteme vorgegeben werden. Die Vorgabe eines Beschleunigungsverlaufs zur Überführung des Modells in eine neue Position ist Aufgabe der Kapitel 5.4 beschriebene Vorsteuerung. Diese ist in Form eines Referenzmodell realisiert und liefert zusätzlich Referenzwerte zum Pfad und des Geschwindigkeitsverlaufs. Die Dynamik (Gleichung (5.16)) mit der die Überführung statt findet, ist für die Simulation mit  $\omega = 1.5$  und  $D = 1$  festgelegt. Weichen Zustände des Bewegungsmodells, ermittelt durch den Beobachter, von der vorgegeben Trajektorie des Referenzmodells ab, greift der lineare Folgeregler ein. Dieser sorgt dafür, das der simulierte Bewegungspfad des Quadrocopter mit dem des Referenzmodells konvergiert. Somit elimi-

niert er Modellunsicherheiten beim Entwurf der Positionsregelung sowie externe Störungen. Die Koeffizienten der mit einem I-Anteil ausgeführten Regelung (Gleichung (5.26)) sind mit  $\tilde{c}_1 = 4$ ,  $\tilde{c}_0 = 5$  und  $\tilde{c}_1 = 0.01$  festgelegt.

Simuliert wird eine Positionsverschiebung des Quadrocopters in der horizontalen Ebene des n-frames. Dabei wird das Modell vom Ursprung des Koordinatensystems in den Punkt  $x = -2 \text{ m}$  und  $y = 1 \text{ m}$  überführt. In Abbildung 5.25 ist das dazugehörige Simulationsergebnis dargestellt. Dort ist zu sehen, wie das Bewegungsmodell entlang des vorgegebenen Pfades in die neue Koordinate übergeführt wird. Zu erkennen ist, dass trotz der filternden Wirkung des Beobachters der Einfluss des Rauschen als auch der niederfrequenten Abtastung nicht vollständig eliminiert wird. Der zeitliche Verlauf der Zustände dieses simulierten Anflugs der neuen Koordinate ist in Abbildung 5.26 separat für die x- und y-Achse dargelegt. Da die Simulation auf eine Variation des  $\psi$  Winkels verzichtet und damit die Orientierung des o-frames mit der des n-frames über den Simulationszeitraum übereinstimmt, können die Neigungswinkel des Quadrocopters einer Achsebewegung des n-frames zugeordnet werden. Somit ist der eingestellte Nickwinkel  $\theta$  (Abbildung 5.26e) für die Bewegung auf der x-Achse, bzw. der Rollwinkel  $\phi$  (Abbildung 5.26f) für die Bewegung auf der y-Achse ausschlaggebende Größe. Dabei ist in beiden Verläufen (Abbildung 5.26f und 5.26e) zu erkennen, dass sich im



**Abbildung 5.25:** Simulierte Positionsverschiebung des Quadrocopters

Zeitpunkt der Übergabe des neuen Positionswert die Winkel sprunghaft ändern. Das liegt daran, dass die Winkel mittels der Inversion aus den Beschleunigungsvorgaben bestimmt werden. Aufgrund der Stetigkeitsbedingung mit der die Vorsteuerung entworfen ist(Kapitel 5.4), müssen die Beschleunigungsvorgaben nur stückweise existieren, das bedeutet diese können sich sprungartig ändern. Da zudem in der Simulation die Zeitkonstanten der Lageregelung vernachlässigt sind, folgt das Bewegungsmodell der Vorgabe eins zu eins.

Alles in allem zeigen die Simulationsergebnisse, dass die Regelung in der Theorie die an sie gestellten Ansprüche erfüllt. Um die Positionsregelung abschließend verifizieren zu können, ist deren Funktionsfähigkeit auf dem realen Quadrocopter zu untersuchen.

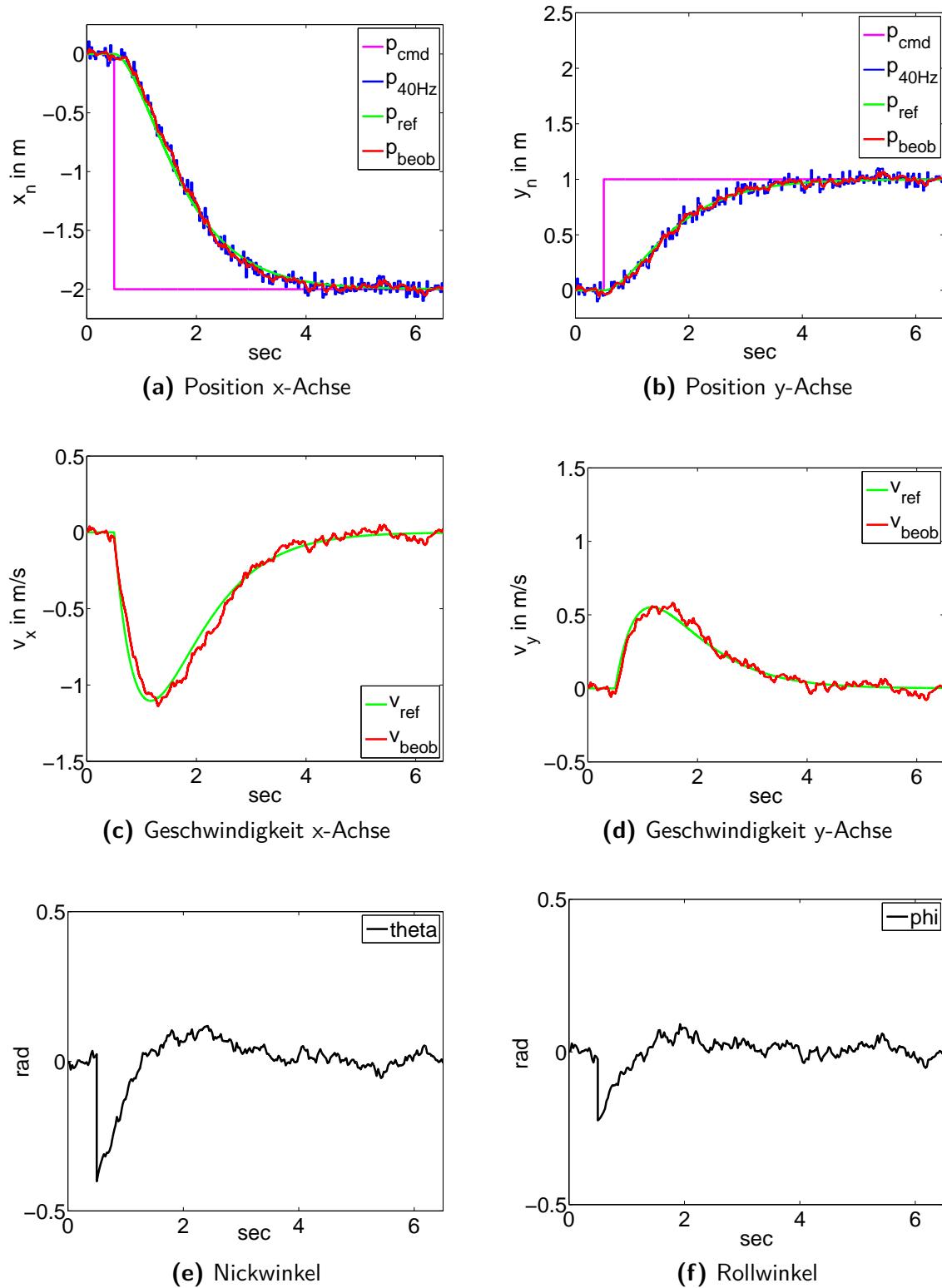


Abbildung 5.26: Simulationsergebnisse des Gesamtmodells

# KAPITEL 6

---

## Verifizierung der Positionsregelung am realen System

---

Nach dem die Funktionsfähigkeit der von *AscTec* und ETH-Zürich entworfenen Positionsregelung mittels Herleitung der einzelnen Komponenten sowie einer abschließenden Simulation des Gesamtsystems (Kapitel 5.7) in der Theorie bewiesen ist, fokussiert sich dieses Kapitel 6 auf die Verifizierung der Regelung am realen System. So gilt es zu darauf zu achten, ob die Zeitkonstante der Lagerregelung, die in der Modellbildung vernachlässigt worden ist, eine Auswirkung auf die Positionsregelung hat. Ausweisen wird sich, ob die Positionsbestimmung mittels eines Lasers anhand der in Kapitel 8 beschrieben orthogonalen Projektion sowie das scanmatching-Verfahren anwendbar ist.

Dafür wird in Kapitel 6.1 eine Messung untersucht, bei der der Positionsregelung eine neue Koordinate übergeben wird, in die der Quadrocopter überführt werden soll. Im anschließenden Kapitel 6.2 wird bezugnehmend auf das ursprüngliche Thema der Arbeit, der Geschwindigkeitsregelung des Quadrocopter, die Funktionsfähigkeit der Positionsregelung als eben diese Geschwindigkeitsregelung dargelegt.

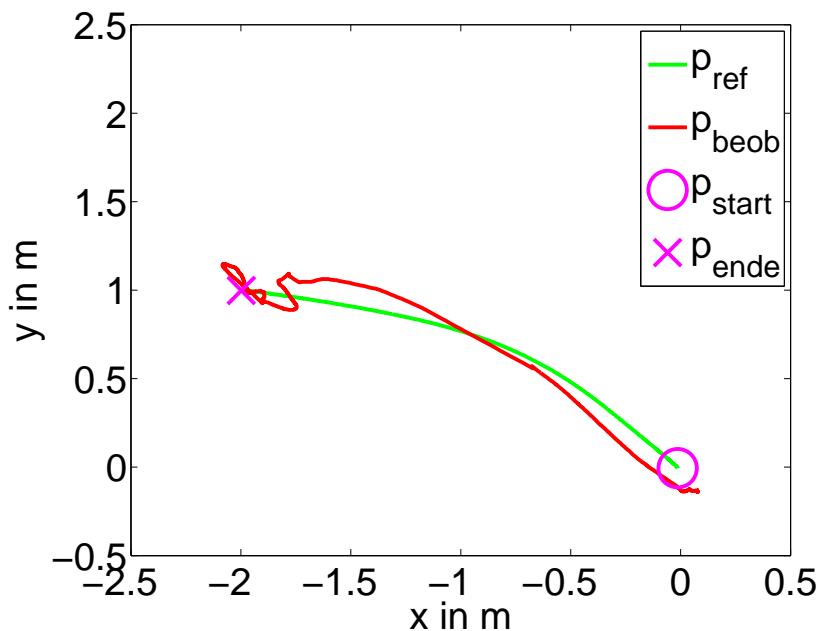
### 6.1 Anflug einer Koordinate im Raum

Um die Messergebnisse mit denen der Simulation in Kapitel 5.7 vergleichbar zu gestalten, entspricht die Ausgangskoordinate ( $x = 0, y = 0$ ) als auch die Zielkoordinate ( $x = -2, y = 1$ ) denen des Simulationsbeispiels. Des Weiteren stimmen die Parameter des Beobachters ( $L = [18 \ 46 \ 3.75]^T$ ) sowie die der Positionsregelung ( $\tilde{c}_1 = 4, \tilde{c}_0 = 5, \tilde{c}_1 = 0.01$ ) mit der in der Simulation verwendeten Parametrierung überein. Zusätzlich erfolgte auch hier die Ausrichtung des Quadrocopters so, dass die Orientierung des o-frames

über den Zeitraum der Messung mit dem n-frame übereinstimmt ( $\psi = 0$ ). Damit ist hier ebenfalls die Achsenbewegung einem bestimmten Neigungswinkel zuzuordnen.

Betrachtet man zunächst den Verlauf des Referenzpfades in Abbildung 6.1 und vergleicht diesen mit dem in der Abbildung 5.25 dargestellten der Simulation, fällt auf, dass diese nicht den gleichen Verlauf aufzeigen. Die Erklärung ist, dass es sich bei dem auf dem *HLP* realisierte Referenzmodell nicht eins zu eins um die in Paper [3] als auch in Kapitel 5.4 vorgestellte Vorsteuerung handelt. Präziser ausgedrückt, deutet das Messergebnis darauf hin, dass die Vorsteuerung eine Trajektorie mit einer höheren Stetigkeitsbedingung generiert [6]. Das bedeutet, der vorgegebene Pfad ist zweimal stetig differenzierbar und nicht nur einmal, wie in Kapitel 5.4 beschrieben. Zur Folge hat dies, dass der gewünschte Beschleunigungsverlauf und die damit verknüpfte Vorgabe der Neigungswinkel stetig ist, d.h. keine sprunghaften Änderungen gefordert werden. Bestätigt wird diese Annahme, wenn man den Verlauf der vorgegebenen Neigungswinkel in Abbildung 6.2 mit den Winkel der Simulation (Abbildung 5.26) vergleicht.

Untersucht wird jetzt die Flugbahn des Quadrocopter anhand der vom Beobachter geschätzten Positionsdaten in Abbildung 6.1. Erkennbar ist, dass die Position des Quadrocopter ausgehend von einer inkonsistenten Anfangsbedingung dank des Folgereglers zu Beginn



**Abbildung 6.1:** Positionsverschiebung des Quadrocopters

mit der Referenztrajektorie konvergiert. Ab der Koordinate  $x = -1$  und  $y = 0.75$  entfernt sich allerdings der Quadrocopter von dem vorgegebenen Pfad, bevor er sich schließlich über der Zielposition einschwingt. Um zu ergründen warum die Position des Flugsystems plötzlich vom Referenzpfad divergiert, ist der zeitliche Verlauf der Zustände für jede Achse zu untersuchen (6.2). Dabei lässt sich anhand Abbildung 6.2a und Abbildung 6.2b erkennen, dass kurz bevor der Quadrocopter den Referenzpfad schneidet, für circa eine Sekunde keine Positionsupdates vom Laser zur Verfügung stehen. Dies hat zur Folge, dass der Schätzwert des Beobachters auf der letzten bekannten Position verweilt, obwohl sich der Quadrocopter weiter bewegt. Die Folge, die Differenz zwischen Ist- und Soll-Position steigt, obgleich das nicht der Realität entspricht. Gleichzeitig sinkt auch der Betrag der geschätzten Geschwindigkeit, zu sehen in Abbildung 5.26c, bzw. für die y-Achse in Abbildung 5.26d. Das Ergebnis, der Zustandsfolgeregel sorgt dafür, dass der Betrag der Neigungswinkel erhöht wird und somit der Quadrocopter fälschlicher Weise weiter beschleunigt und sich dessen Geschwindigkeit erhöht. Der Grund warum der Quadrocopter plötzlich vom Referenzpfad divergiert (Abbildung 5.25). Nach dieser circa einen Sekunde ohne Positionsupdate wird die Position erneut zuverlässig über den Laser bestimmt. Der Beobachter schätzt die zu hohe Geschwindigkeit mit der sich der Quadrocopter tatsächlich bewegt (Abbildung 5.26c und Abbildung 5.26d). Da jetzt Position und Geschwindigkeit deutlich von den Vorgaben des Referenzmodells abweichen, fordert der Folgeregel in Verbindung mit der Inversion entgegengesetzte Neigungswinkel, um den Quadrocopter abzubremsen und zurück auf die gewünschte Flugbahn zu führen. In Abbildung 6.2e und Abbildung 6.2f ist zu erkennen, dass der Betrag dieser Winkel deutlich über den zuvor geforderten liegt. Da die Lageregelung die vorgegebenen Winkel nahezu ohne Zeitverzögerung einstellt, erfolgt die Bremsung bzw. die Richtungsänderung hin zur Referenztrajektorie sehr abrupt. Klar zu erkennen ist die zackige Bewegung in Abbildung 6.1 kurz vor Erreichen der Zielkoordinate. Der Quadrocopter überfliegt dabei nochmals den Referenzpfad, schwingt sich allerdings aufgrund der stabilisierenden Auslegung des Folgereglers und den kontinuierlich vorhanden Positionsdaten in einer Umgebung  $\pm 10 \text{ cm}$  um die Endposition ein. Womit bewiesen ist, dass die Positionsregelung mittels eines Lasersanners ausführbar ist.

Rein optisch wirkte das Regelverhalten dieser Messung von außen betrachtet sehr nervös, da ohne Kenntnis der Messdaten kein ersichtlicher Grund für die großen Stellwinkel zu erkennen war. Des Weiteren treten solche Definitionslücken bei der Lokalisierung nur sporadisch auf, so dass der Quadrocopter in den meisten Fällen in einer flüssigen Bewegung in die neue Position überführt wird. Nichtsdestotrotz gilt es den Einfluss der Lücken in der Positionsbe-

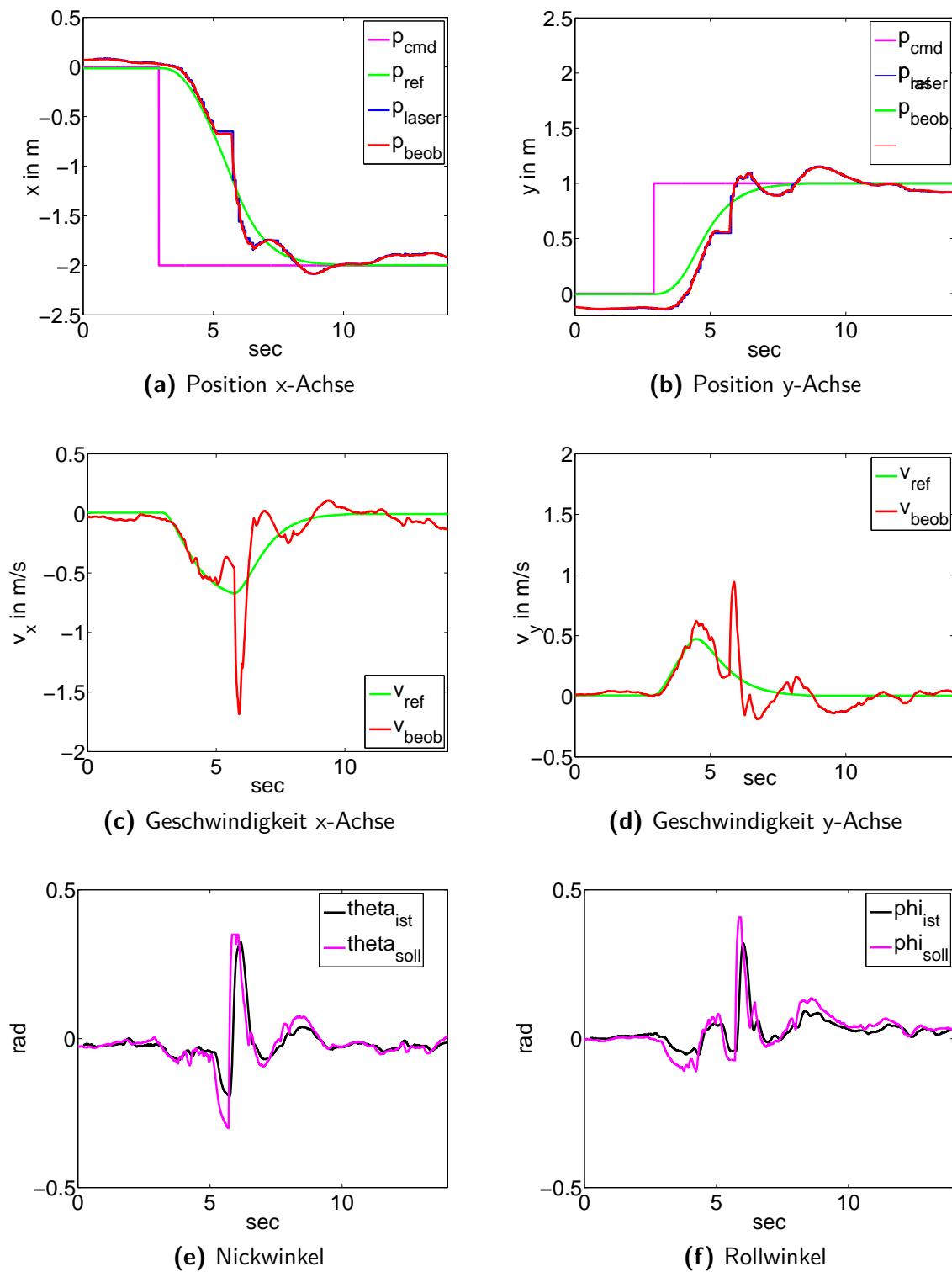


Abbildung 6.2: Messergebnisse Flug

stimmung zu minimieren, da vor allem bei längeren Flügen die Wahrscheinlichkeit, dass ein solcher Fehler auftritt, steigt. So gilt es im Anschluss dieser Masterthesis zu untersuchen, welche Umstände zu einer lückenhaften Lokalisierung mittels Laser führen. Vorstellbar ist, dass das von *ROS* verwendete IP-Protokoll zeitweise zu viele Umgebungscans in einem Datenpaket zusammenfasst, womit die physische Abtastrate reduziert wird. Alternativ könnte eine weitere Ursache darin liegen, dass der Laser für zu große Neigungswinkel den Boden bzw. die Decke erfasst, somit der ICP-Algorithmus (Kapitel 8.2) die aufeinanderfolgenden Scans nicht übereinander legen kann und so zu keinem Positionsergebnis kommt. Wenn dem so ist, existiert die Möglichkeit die maximalen Stellwinkel zu begrenzen. Dies ist als erste Massnahme zu empfehlen, führt jedoch zu einer Beschränkung der Dynamik. Deshalb sollte in diesem Fall untersucht werden, ob ein *3D-Simultaneous Localization and Mapping (SLAM)*-Algorithmus zur Positionsbestimmung nicht die besser Alternative darstellt. Voraussetzung dafür ist jedoch eine genaue Höhenbestimmung des Quadrocopters. Diese wurde zwar von Jan Kallwies [11] entwickelt, steht momentan allerdings nicht auf dem Quadrocopter zur Verfügung. Eine kurzfristige Lösung zur Bekämpfung des Einfluss von Lokalisierungslücken ist die Implementierung eines Algorithmus, der ausbleibende Positionsdaten zuverlässig erkennt. Schlägt dieser Algorithmus an, könnten die zulässigen Neigungswinkel auf ein Minimum reduziert werden, sodass der Quadrocopter nicht weiter beschleunigt. Nach Erhalten neuer Positionsdaten ist diese Limitierung aufzuheben.

Schlussendlich ist, auch ohne diese weiteren Ideen/Maßnahmen, die Funktionsfähigkeit der von AscTec in Verbindung mit der ETH-Zürich entwickelten Positionsregelung bewiesen. Zum Abschluss gilt es nach wie vor die Anwendbarkeit der Positionsregelung als Geschwindigkeitsregler, mittels Fernsteuerung aufzuzeigen.

## 6.2 Geschwindigkeitsregelung

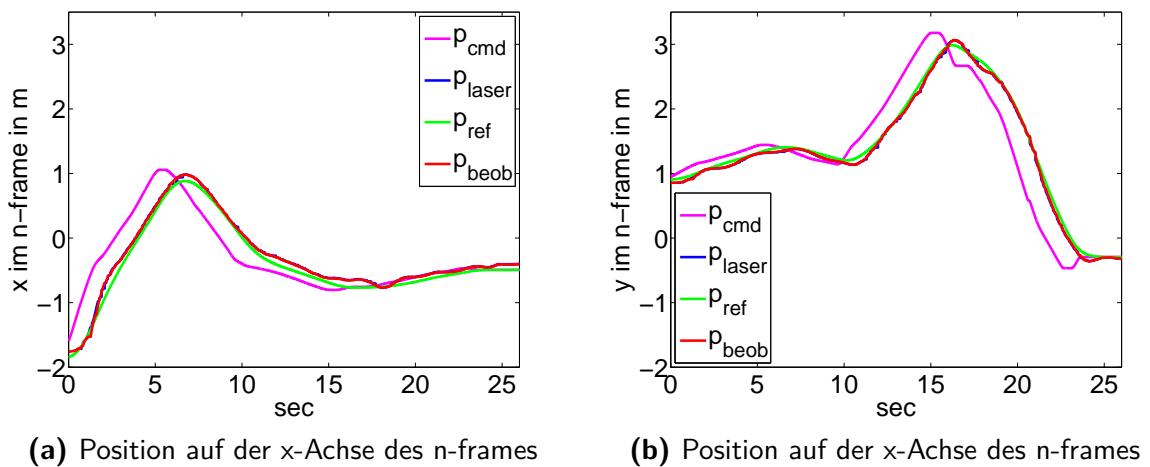
In der ursprünglichen Ausschreibung zu dieser Masterthesis ist die Implementierung einer Geschwindigkeitsregelung als Schwerpunkt der Arbeit ausgegeben. In Folge dessen wird in diesem Kapitel abschließend die Anwendbarkeit der Positionsregelung als Geschwindigkeitsregelung mittels eines Testergebnisses offengelegt.

Die Vorgabe der Geschwindigkeiten erfolgt im o-frame, dass heißt es wird bestimmt mit welcher Geschwindigkeit sich der Quadrocopter vorwärts, rückwärts oder seitlich fortzubewegen hat. Da allerdings die Positionsregelung auf den Koordinaten des Navigationskoordina-

tensystems fußt, ist die gewünschte Geschwindigkeit zunächst ins n-frame zu transformieren. Dort sind diese, wie im Kapitel 5.4 beschrieben, zu integrieren. Auf diese Weise erfolgt für Geschwindigkeiten ungleich Null eine stetige Verschiebung der anzufliegenden Sollposition. Die Größe der Positionsverschiebung ist dabei proportional zu den geforderten Geschwindigkeiten. Da der Quadrocopter mittels der Positionsregelung der Sollposition folgt (Abbildung 6.3), spricht man auch von einer Pseudo-Geschwindigkeitsregelung.

Abbildung 6.4 beleuchtet den zur Positionsverschiebung führenden Geschwindigkeitsverlauf im o-frame. Die Vorgabe der Geschwindigkeit erfolgte dabei über den Bedienhebel der Fernsteuerung, der bei manueller Steuerung zur Vorgabe der Neigungswinkel dient. Bei aktiver Geschwindigkeitsregelung werden die Rohdaten dieses Bedienhebel jedoch Geschwindigkeitsvorgaben im o-frame zugeordnet. Der Umrechnungsfaktor kann beliebig festgelegt werden, solange der Quadrocopter der maximal einforderbaren Geschwindigkeit folgen kann. In diesem Fall betrug der Faktor bei der Durchführung der Messung 0.0004, das bedeutet für den Rohwertbereich der Fernbedienung von  $-2047$  bis  $2048$  lassen sich Geschwindigkeiten in einem Bereich von  $-0.8 \frac{m}{s}$  bis  $0.8 \frac{m}{s}$  vorgeben.

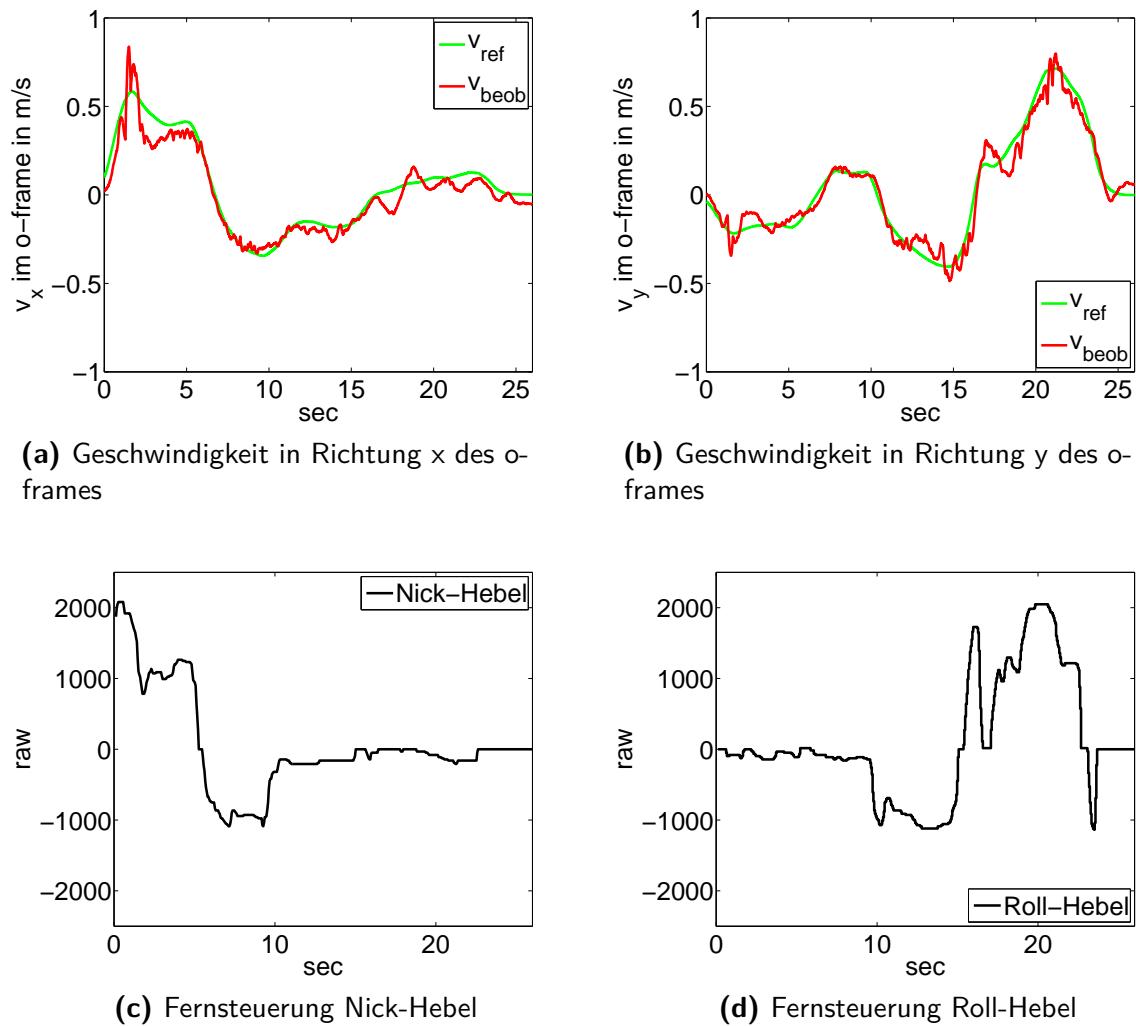
Um zu prüfen, ob sich für den Stellwertverlauf über die Fernbedienung (Abbildung 6.4c und Abbildung 6.4c) die geforderten Geschwindigkeiten aus der Sicht des Quadrocopters einstellen, sind die vom Referenzmodell generierte Sollgeschwindigkeit, als auch die vom Beobachter geschätzte Geschwindigkeit ins o-frame transformiert worden (Abbildung 6.4a und Abbildung 6.4b). Auf den ersten Blick fällt auf, dass der qualitative Verlauf der



**Abbildung 6.3:** Positionsverlauf Versuch Geschwindigkeitsregler

Vorgaben mit der sich tatsächlich einstellenden Geschwindigkeit übereinstimmt. Weiterhin ist zu beobachten, dass sich Geschwindigkeitsvorgaben des Bedienhebels gedämpft und phasenverschoben auf den Sollwertverlauf des Referenzmodells auswirken, noch deutlicher erkennbar ist dies in Abbildung 6.3. Zu erklären ist das durch die Zeitkonstanten des Referenzmodells, die für diesen Flugversuch über die Eigenfrequenz  $\omega_0 = 1.5$  und die Dämpfung  $D = 1$  festgelegt sind. Ist gewünscht, dass sich Vorgaben über die Fernsteuerung schneller einstellen, ist daher die Eigenfrequenz  $\omega_0$  des Referenzmodells zu erhöhen. Dies wird auch empfohlen, um ein besseres Feedback über geforderte Geschwindigkeiten zu erlangen.

Abschließend bewertet, zeigen die Messdaten dieses Flugversuches (Abbildung 6.4), dass sich die auf dem *HLP* befindliche Positionsregelung auch dem formulierten Wunsch der Ausschreibung nach einer Geschwindigkeitsregelung erfüllt. Womit die Arbeit ihren Abschluss findet.



**Abbildung 6.4:** Messergebnisse Geschwindigkeitsregelung

# KAPITEL 7

---

## Zusammenfassung und Ausblick

---

### 7.1 Zusammenfassung

Im Rahmen dieser Masterthesis wurde eine Positions- und Geschwindigkeitsreglung, basierend auf einem Onboardlokalisierungssystem, mittels eines Laserscanners auf dem Pelican Quadrocopter der Firma *AscTec* implementiert. Zur Lokalisierung und Regelung wurden dabei bereits entwickelte Funktionsbausteine zu einem funktionierenden Gesamtsystem zusammengefügt. So ist der Schwerpunkt darauf ausgelegt, die Theorie und die Konzepte dieser Bausteine offenzulegen.

Begonnen wurde mit der Lokalisierung des Quadrocopters. Dort wird aufgezeigt, wie der Einfluss der Neigungswinkel auf die Entfernungsmessung mittels einer orthogonalen Projektion der Messwerte auf die horizontale Ebene des Raumes eliminiert wird. Die Grundvoraussetzung für den bei Positionsbestimmung zum Einsatz kommenden Scanmatching-Algorithmus, dessen Funktionsweise in dieser Arbeit ebenfalls mathematisch dargelegt ist.

Im nächsten Schritt erfolgte die Validierung der von *AscTec* in Verbindung mit der ETH-Zürich entwickelten Positionsregelung. Angelehnt an die Funktionsbeschreibung des Reglers in [3] bestand die Herausforderung darin, den dort beschriebenen Aufbau der Regelung mittels Literaturverweise, der Herleitung der verwendeten Regelalgorithmen sowie anhand von Simulationsbeispielen zu legitimieren. Zusätzlich wurde aufgezeigt, über welche Stellschrauben die Komponenten der Regelung zu parametrieren sind.

Nachdem die Legitimierung der Positionsregelung mit einer abschließenden Simulation des Gesamtsystems in der Theorie bewiesen war, erfolgten Flugversuche mit dem Quadrocopter. Diese belegten die Funktionsfähigkeit der Positionsregelung als auch der Geschwindigkeits-

regelung in der Praxis, was auch aus den in dieser Arbeit veröffentlichten Messdaten zweier Flugversuche hervorgeht. Allerdings stellte sich dabei heraus, dass es kleine Defizite bei der Positionsbestimmung mittels des Lasers zu beheben gibt. Vereinzelt zeigte sich, dass über kurze Zeiträume von circa einer Sekunde die Position des Quadrocopters nicht aktualisiert wird. Dies führte zu einer Fehlinterpretation der Positionsregelung über den Zustand des Quadrocopters, was im ungünstigsten Fall zum Absturz des Flugobjektes führen kann. Hier gilt es im Anschluss dieser Arbeit Nachforschungen zur Ergründung der Ursache dieser Positionsfehlstellen anzustellen.

## 7.2 Ausblick

Ungeachtet der aktuell bekannten Defizite bei der Positionsbestimmung ist die Empfehlung zunächst die Höhenschätzung von Jan Kallwies [11] auf dem Quadrocopter zu integrieren. Dies würde nicht nur dazu führen, dass mittels der Positionsregelung Koordinaten in einem dreidimensionalen Raum vollständig autonom angeflogen werden können, sondern eröffnet gleichzeitig neue Möglichkeiten bei Positionsbestimmung mittels des Lasers. So kann unter Berücksichtigung der Flughöhe zur Bestimmung der Position ein 3D-SLAM-Algorithmus eingesetzt werden. Diese Art der Lokalisierung hat gegenüber des simplen Scanmatching-Verfahren den Vorteil, dass es unabhängig von senkrechten Eigenschaften der Umgebung ist. Ob dies jedoch dazu führt das keine Lücken in der Positionsbestimmung auftreten, ist nicht garantiert. Deshalb gilt es, sich zusätzlich mit der Entwicklung eines Notlaufprogramms zu beschäftigen. Dieses könnte so aufgebaut sein, dass es ausbleibende Aktualisierungen der Positionsdaten erkennt, alle von der Regelung geforderten Stellwinkel zurück nimmt und erst nach Erhalten neuer Positionsdaten diese wieder freigibt. Gleichzeitig wäre eine akustische Warnung denkbar, so dass der Bediener im Ernstfall die Regelung komplett deaktivieren und das Flugsystem rechtzeitig manuell stabilisieren kann.

Alles in allem ist das Ziel mit dem Quadrocopter einer autonome Referenzplattform für das am Fraunhofer-Institut für Integrierte Schaltungen entwickelte Lokalisierungssysteme durch diese Arbeit ein Stück näher gerückt.

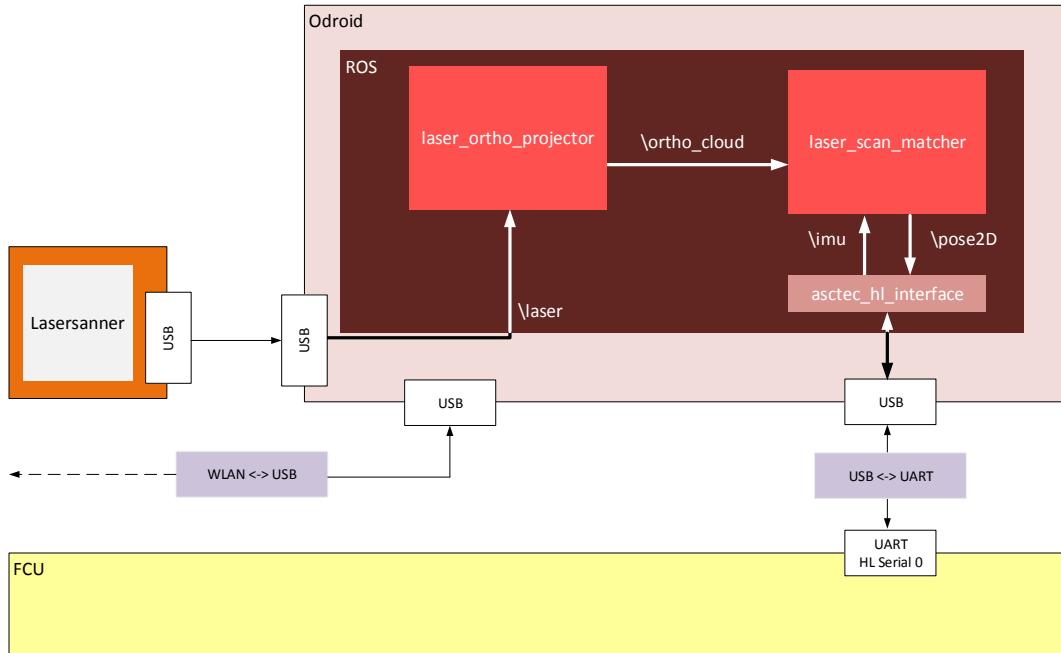
# KAPITEL 8

---

## Zweidimensionale Positionsbestimmung des Quadrocoters in xy-Ebene des Navigationsframes

---

Sowie Wie in der Aufgabenstellung beschrieben, erfolgt die Positionsbestimmung über den auf dem Quadrocopter montierten Lasersanner. Man spricht hierbei von einem Es wird als ein Onboard-Lokalisierungssystem bezeichnet. Dabei erzeugt der Laser einen zweidimensionalen Scan der Umgebung. Die aufgenommen Entfernung sind im l-frame definiert, d.h. sie beziehen sich auf immer auf den Ursprung des Lasers. Damit enthalten diese Rohdaten keine Information über die Position und Orientierung des Quadrocopters in der xy-Ebene des Navigationskoordinatensystem (n-frame). Jedoch lässt sich aus Allerdings lassen sich aus den Umgebungscans mittels der Methode des „scanmatching“ die Position in einer zweidimensionalen Ebenen bestimmen (Kapitel 8.2). Damit diese jedoch für die Bestimmung der Quadrocopterposition in der horizontalen Ebene eingesetzt werden kann können, ist es von Nöten, die Laserdaten in das o-frame zu projizieren (Kapitel 8.1). Erst dann lässt Als Folge lassen sich für das o-frame und somit dem damit für den Quadrocopter, Position und Orientierung in der xy-Ebene des n-frames bestimmen. Beide Vorgänge sind bereits für ROS implementiert und sind Bestandteil der scan\_tools. Genauer gesagt, handelt es sich dabei folglich um dem „laser\_ortho\_projector“- und dem „laser\_scan\_matcher“-Knoten (Abbildung 8.1). Ziel der dieses Kapitel 8 ist es, die Mathematik dahinter zu erläutern, sodass man eine Verständnis der über die Funktionsweise der Knoten erhält entsteht.



**Abbildung 8.1:** Verknüpfung des „laser\_ortho\_projector“- und des „laser\_scan\_matcher“-Knoten GRAFIK EVENTUELLE ÜBERARBEITEN DA ASC-TEC\_HL\_INTERFACE KNOTEN DER FÜR DIE KOMMUNIKATION VERANTWORTLICH IST FEHLT

## 8.1 Projektion der Laserdaten in das o-frame auf der xy-Ebene des n-frames („laser\_ortho\_projector“)

Die Projektion der Laserdaten ins o-frame erfolgt orthogonal zur xy-Ebene des n-frames.

In allgemeiner Form ist dies in Abbildung 8.2a dargestellt. Grundlage für die orthogonalen Projektion ist die Annahme, das dass es sich bei den erfassten Objekten um senkrechte Gegenstände handelt. Das heißt, ihre Form variiert nicht mit der Höhe in der sie erfasst werden. Für geschlossen-In geschlossenen Räumen ist diese Annahme zutreffend, da es sich bei den Objekten hauptsächlich um senkrechte Wände handelt. Durch Erfüllung dieser Voraussetzungen kann die Flughöhe des Quadrocopters vernachlässigt werden. Dies kann man aus Abbildung 8.2b entnehmen Das ist aus der Abbildung 8.2b entnehmbar. Eine Verschiebung des Koordinaten Ursprungs des b-frames auf der z-Achse des o-frames hat demzufolge keinen Einfluss auf die Projektion. Folglich kann für beide Koordinatensystem der identischen Ursprung angenommen werden. Unter Beachtung dieser Annahmen kann der

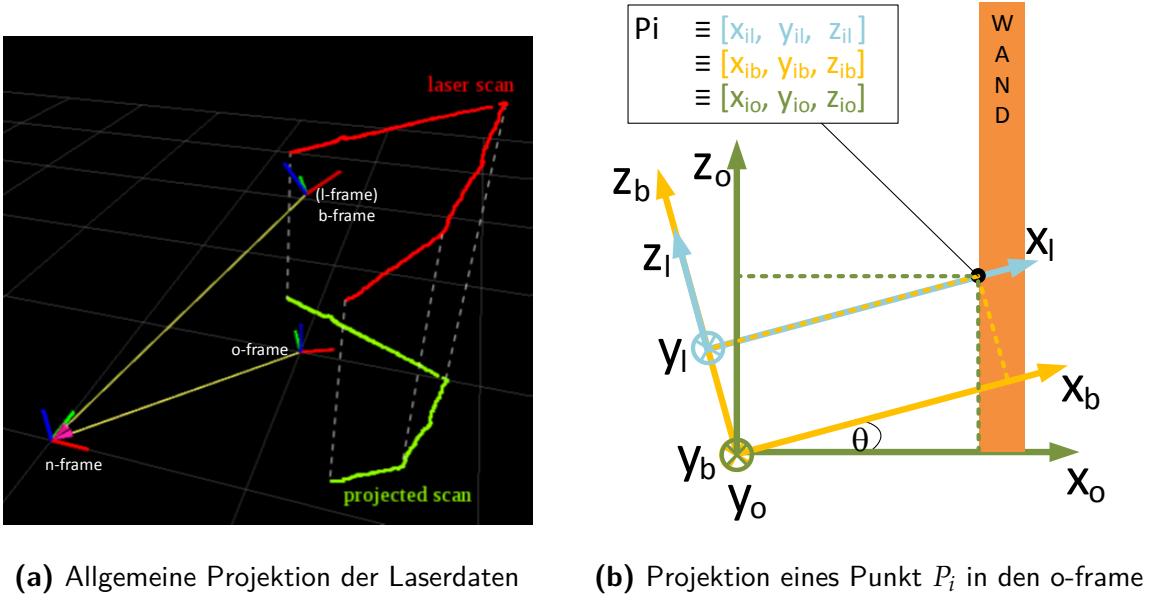


Abbildung 8.2: Projektion der Laserdaten

Einfluss des Roll-( $\phi$ ) und Nickwinkels ( $\theta$ ) auf die Entfernungsmessung eliminiert werden. Die Winkelgrößen liefern die *IMU*. Der mathematische Ablauf der orthogonalen Transformation wird basierend auf Literatur [16] im Folgenden dargelegt.

Entfernungsdaten eines Laserumlaufs ~~besteht bestehen~~ aus mehreren diskreten Abtastungen (Abbildung 8.3). Übergeben werden sie in Form von ~~Entfernung Entfernungen~~  $r_i$  in einem Array (~~Topic (ROS) Topic:~~ \laser Abbildung 8.1). Mittels der Schrittweite von  $0.25^\circ$  ~~lässt lassen~~ sich anhand des Indizes  $i$  für jeden Messpunkt einen Winkel  $\gamma_i$  zuweisen.

$$\gamma_i = 135^\circ - 0.25^\circ \cdot i \quad (8.1)$$

Die Entfernung eines Punktes  $p_i$  ist somit über  $\{r_i, \gamma_i\}$  definiert. Zur weiteren Verwendung ist es notwendig ~~die Messungen im kartesischen~~, ~~die Messungen in das kartesische~~ Koordinatensystem des l-frame zu übertragen.

$$p_i^l = [\cos(\gamma_i) \cdot r_i, \sin(\gamma_i) \cdot r_i, 0]^T \quad (8.2)$$

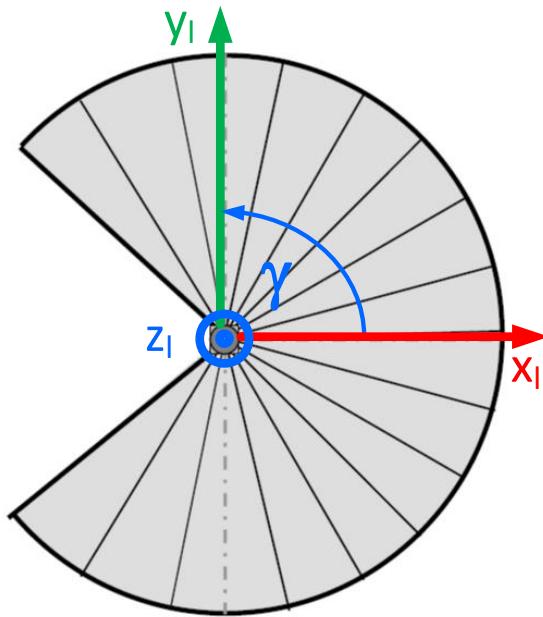


Abbildung 8.3: Draufsicht I-frame

Da der Bezugspunkt des o-frames im Schwerpunkt des Quadrocopters liegen soll, in dem auch der b-frame seinen Ursprung hat, ist es von nötigen erforderlich, die Laserdaten vom I-frame ins b-frame zu transformieren. Wie schon in Kapitel 9.2 erwähnt, handelt es sich dabei um eine konstante Transformation. Genauer gesagt, um einen Offset von 10cm auf der  $z^b$ -Achse, da der Laser denn der Laser ist, oberhalb des Quadrocopterschwerpunktes montiert ist.

$$p_i^b = [\cos(\gamma_i) \cdot r_i, \sin(\gamma_i) \cdot r_i, 0.1]^T \quad (8.3)$$

Nun da Nachdem die Laserpunkte im b-frame definiert sind, kann die Transformation der Umgebungsdaten in die xy-Ebene des o-frames erfolgen. Angesichts der Tatsache, das dass die Winkel  $\phi$  und  $\theta$  als Verdrehung um die Achsen des o-frames definiert sind, benötigt man erfordert zur Umrechnung der Laserdaten die in Kapitel 9.2 eingeführte Gleichung 9.6 zur inversen Koordinatentransformation. Dabei wird der Yaw-Winkel zu Null gesetzt. Grund hierfür ist, das dass die Ausrichtung der Flugrichtung in der zweidimensionalen Ebene mit der des b-frames übereinstimmen sollen. Daraus ergibt sich für die Transformationsmatrix,

$$M_{ob} = \begin{bmatrix} \cos \theta & \sin \phi \sin \theta & \cos \phi \sin \theta \\ 0 & \cos \phi & -\sin \phi \\ -\sin \theta & \sin \phi \cos \theta & \cos \phi \cos \theta \end{bmatrix} \quad (8.4)$$

über die sich die Positionen der Laserpunkte  $P_i^b$  im o-frame bestimmten lassen.

$$p_i^o = M_{ob} \cdot P_i^b \quad (8.5)$$

**Erneut Wiederum** kann unter der Annahme von rechtwinkligen Objekten die Höhe des Punktes in  $z^o$ -Achse zu Null gesetzt werden. **Somit Demzufolge** sind die Punkte  $p_i^l$  auf der xy-Ebene des o-frame folgendermaßen abgebildet.

$$p_i^o = \begin{bmatrix} \cos \theta \cos(\gamma_i) \cdot r_i + \sin \phi \sin \theta \sin(\gamma_i) \cdot r_i + \cos \phi \sin \theta \cdot 0.1 \\ \cos \phi \sin(\gamma_i) \cdot r_i - \sin \phi \cdot 0.1 \\ 0 \end{bmatrix} \quad (8.6)$$

Alle Punkte  $p_i^o$  eines Umlaufs **einen Scan eines Scans**  $S$ .

$$S^o = [p_i^o | i = 1..1080] \quad (8.7)$$

**Diese Die** Beschreibung  $S$  der Laserscans im o-frame ist die Basis für die im anschließende Kapitel 8.2 behandelte Positionsbestimmung in der zweidimensionalen Ebene des n-frames.

## 8.2 Positionsbestimmung anhand der ins o-frame überführten Laserdaten über scanmatching

Aufbauend auf den im Kapitel 8.1 vorgestellten `Laser_ortho_projektor`, ist es Aufgabe des **Folgenden-folgenden** Teilkapitels zu erläutern, wie anhand eines Referenzscans  $S_{ref}$  und einem weiteren Scan  $S_{neu}$  die Position in der euklidischen xy-Ebene des n-frames bestimmt werden kann. Zur Anwendung kommt hier die Methode des Scanmatching. Dabei gilt die Annahme, **das-dass** für jeden Scan  $S_{neu}$  und dessen dazugehörigen Position  $P_{neu}$  eine Rotation  $M_z^o$  um  $\psi^o$ , inklusive Translation  $T$  existiert, **so-dass-sodass** die beiden Datenwolken  $S_{neu}$  und  $S_{ref}$  übereinander liegen (Abbildung 8.4). Definiert ist  $S_{ref}$  dabei

an der Stelle  $P_{ref}$ .

Ausgangspunkt sind zwei im o-frame definierte Datenwolken.

$$S_{ref} = [p_{ref_i} | i = 1..n_{ref}] \quad (8.8)$$

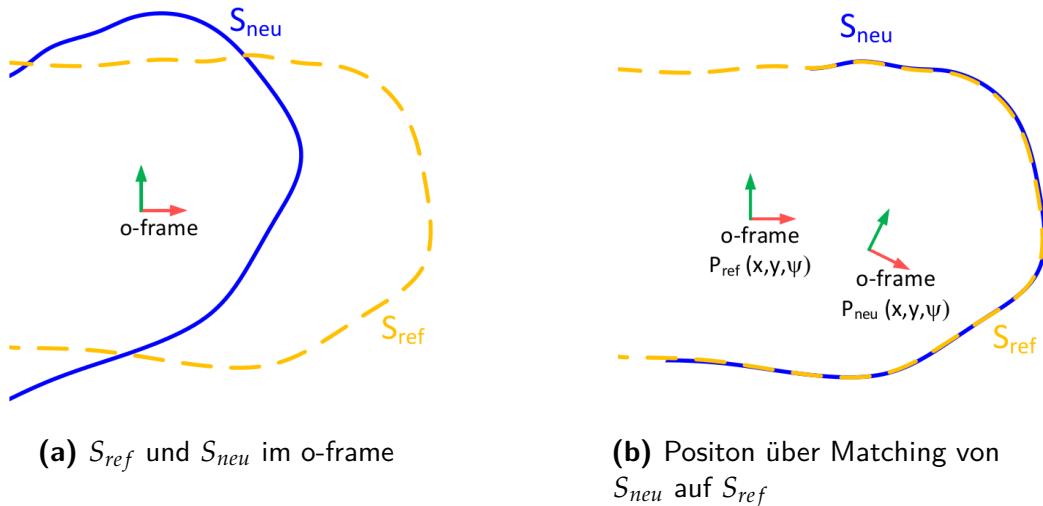
$$S_{neu} = [p_{neu_i} | i = 1..n_{neu}] \quad (8.9)$$

Dargestellt in Abbildung 8.4a.

Zur Bestimmung des Matchings bzw. der Rotation  $M_z^o$  und der Translation  $T$  wurde im Jahr 1992 unter anderem von Paul Besl der ICP-Algorithmus entwickelt. Dabei handelt es sich um einen iterativen Algorithmus. Abgebildet sind die einzelnen Integrationsschritte in einem Flussdiagramm in Abbildung 8.5. Daran orientierend wird im folgenden jeder einzelne Vorgang erläutert.

- **Schritt 1,** Punktekorrespondenz.

Hierbei bekommt jeder Punkt des  $S_{neu}$  einen korrespondierenden Punkt aus der Datenwolke  $S_{ref}$  zugewiesen. Man spricht hierbei von der Point-to-Point Arithmetik. Als Korrespondenzpunkt  $p'_{ref_i}$  des Scanpunktes  $p_{neu_i}$  wird der nächstliegende Punkt



**Abbildung 8.4:** Prinzip Scanmatching

des Referenzscans  $S_{ref}$  zugewiesen. Dabei kann ein Punkt  $p_{ref}$  die Bedinung des Closest-Point für mehrere Punkte des neuen Scans  $S_{neu}$  erfüllen.

$$S'_{ref} = [p'_{ref_i} | i = 1..(n' = n_{neu})] \quad (8.10)$$

Die Suche der entsprechenden Werte erfolgt über die Methode der erschöpfenden Suche (Brute-Force-Methode), d.h. für jeden Punkt werden alle Punktabstände ermittelt und der Punkt mit der geringsten Entfernung zugewiesen. Das Ergebnis ist eine Datenmenge  $S'_{ref}$ , deren Anzahl an Werten denen von  $S_{neu}$  entspricht.

Dieser Schritt benötigt auf Grund des nicht optimierten Suchalgorithmus die höchste Rechenleistung. Laut [LITERATUR AUT4] stellt dies für die Verarbeitung von 2D-Laserscans kein Problem dar.

- **Schritt 2,** Bestimmung des ~~Rotationswinkel~~ ~~Rotationswinkels~~  $\Delta\psi^o$  und der Translation  $T$

Wie ~~schon zu beginn angekündigen~~ ~~zu beginn angemerkt~~, soll eine rotatorische und translatorische Transformation zur Überlappung von  $S_{neu}$  mit  $S_{ref}$  führen. Idealität und eine sehr kleine Änderung vorausgesetzt ~~könnte jeder Punkt~~, ~~kann jeder Punkt~~  $p_{neu_i}$  von  $S_{neu}$  in den entsprechenden Punkt  $p_{ref_i}$  von  $S_{ref}$  über (8.11) umgerechnet werden.

$$p_{ref_i}(M_z^o(\psi^o), T) = M_z^o(\psi^o) \cdot p_{neu_i} + P_{ref} + T \quad (8.11)$$

Die ~~Zweidimensionale~~ ~~zweidimensionale~~ Roationsmatrix  $M_z^o$  entspricht dabei der in Kapitel 9.2.2 eingeführten Koordinatentransformationsmatrix (9.1) reduziert auf die x und y Anteile.

$$M_z^o(\psi^o) = \begin{bmatrix} \cos \psi^o & -\sin \psi^o \\ \sin \psi^o & \cos \psi^o \end{bmatrix} \quad (8.12)$$

In der Praxis ist die Umgebung nicht ideal und aufgrund der hohen Dynamik können die Änderungen zwischen zwei Messwerten ~~unbestimmt~~ größer ~~Ausfallen~~ ~~ausfallen~~. Dadurch sind nicht alle Werte von  $S_{ref}$  und  $S_{neu}$  Indize ~~für~~ ~~über~~ Indize vergleichbar. Der Grund ~~warum in Vorgang~~ ~~weshalb in Schritt~~ 1 ~~zu zum zum~~ ~~neuen~~ Scan  $S_{neu}$

ein korrespondier Scan  $S'_{ref}$  eingeführt wurde. Der im Folgenden Anwendung findet Ein einsetzen wird. Dieser findet im folgenden Anwendung. Ein Einsetzen der Punkte  $p'_{ref_i}$  von  $S'_{ref}$  in die Gleichung (8.11) ist jedoch nicht die Lösung des Problems, da es zu keiner eindeutigen Ergebnis führt. Da Aufgrund von möglichen Mehrfachkorrespondenzen die Gleichung nicht eindeutig gelöst werden kann. Laut [13] kann aus dieser jedoch diese Formel (8.11) als Fehlgleichung herangezogen werden. Mit dieser lässt sich über die least-squar-Methode die Hilfe der least-square Methode kann darüber die Transformation bestimmt werden, für sich die kleinste Summe der quadratischen Abweichungen ermitteln ergibt.

$$E(M_z^o(\Delta\psi^o), T) = \frac{1}{n'} \sum_{i=1}^{n'} \|p'_{ref_i} - (M_z^o(\Delta\psi^o) \cdot p_{neu_i} + T)\|^2 \quad (8.13)$$

Um das Minimum von abhängig von  $E(M_z^o(\Delta\psi^o), T)$  bestimmen zu können, wird nach [15] der Schwerpunkt ( $c_{ref}, c_{neu}$ ) der korrespondierenden Punkte ermittelt.

$$c_{ref} = \frac{1}{n'} \sum_{i=1}^{n'} p'_{ref_i} \quad (8.14)$$

$$c_{neu} = \frac{1}{n'} \sum_{i=1}^{n'} p_{neu_i} \quad (8.15)$$

Damit ergibt beschreiben sich die Datenwolken folgendermaßen beschreiben.:

$$\tilde{S}'_{ref} = [\tilde{p}'_{pref_i} = p'_{pref_i} - c_{ref} | i = 1..n'] \quad (8.16)$$

$$\tilde{S}_{neu} = [\tilde{p}_{neu_i} = p_{neu_i} - c_{neu} | i = 1..n'] \quad (8.17)$$

Setzt man 8.16 und 8.17 in die Fehlgleichung ?? resultiert 8.13 ein, resultiert:

$$\begin{aligned} E(M_z^o(\Delta\psi^o), T) &= \frac{1}{n'} \sum_{i=1}^{n'} \|\tilde{p}_{ref_i} - M_z^o(\Delta\psi^o) \cdot \tilde{p}_{neu_i} - (T - c_{ref} + M_z^o(\Delta\psi^o) \cdot c_{neu})\|^2 \\ &= \frac{1}{n'} \sum_{i=1}^{n'} \|\tilde{p}_{ref_i} - M_z^o(\Delta\psi^o) \cdot \tilde{p}_{neu_i} - \tilde{T}\|^2 \end{aligned} \quad (8.18)$$

Über  $\tilde{T}$  ist die Abweichung der Schwerpunkte translatorisch als auch rotatorisch beschrieben. Damit beide Schwerpunkte genau über einander liegen ist  $\tilde{T} = 0$  zusetzen. Daraus ergibt sich  $\text{---}$ :

$$0 = T - c_{ref} + M_z^o(\Delta\psi^o) \cdot c_{neu} \quad (8.19)$$

~~und eine Fehlergleichung die nun mehr nur noch~~ Sowie eine Fehlergleichung, die nun ausschließlich von der Rotation abhängig ist und dessen Betrag sich wie folgt darstellen lässt. darstellt:

$$\begin{aligned} E(M_z^o(\Delta\psi^o)) &= \frac{1}{n'} \sum_{i=1}^{n'} ||\tilde{p}_{ref_i} - M_z^o(\Delta\psi^o) \cdot \tilde{p}_{neu_i}||^2 \\ &= \frac{1}{n'} \sum_{i=1}^{n'} (\tilde{p}_{ref_i}^T \tilde{p}_{ref_i} + \tilde{p}_{neu_i}^T \tilde{p}_{neu_i} - 2\tilde{p}_{ref_i}^T \cdot M_z^o(\Delta\psi^o) \cdot \tilde{p}_{neu_i}) \end{aligned} \quad (8.20)$$

Weiterhin auf die Literatur [15] beziehend ist es zur Minimierung von  $E(M_z^o(\Delta\psi^o))$  ausreichend den gemischten Term zu  $\tilde{E}(M_z^o(\Delta\psi^o))$  maximieren.

$$\tilde{E}(M_z^o(\Delta\psi^o))_{max} = \underset{M_z^o(\Delta\psi^o)}{\operatorname{argmax}} \sum_{i=1}^{n'} (2\tilde{p}_{ref_i}^T \cdot M_z^o(\Delta\psi^o) \cdot \tilde{p}_{neu_i}) \quad (8.21)$$

Beziehungsweise abhängig von  $\Delta\psi^o$ .

$$\begin{aligned} \tilde{E}(\Delta\psi^o)_{max} &= \underset{\Delta\psi^o}{\operatorname{argmax}} \sum_{i=1}^{n'} (2(\cos(\Delta\psi^o)(\tilde{x}_{ref} \cdot \tilde{x}_{neu} + \tilde{y}_{ref} \cdot \tilde{y}_{neu}) \\ &\quad + \sin(\Delta\psi^o)(\tilde{y}_{ref} \cdot \tilde{x}_{neu} + \tilde{x}_{ref} \cdot \tilde{y}_{neu})) \end{aligned} \quad (8.22)$$

Aufgrund der trigonometrischen Addition besitzt der Summand eine Maximum für  $\Delta\psi^o$  wenn

$$\frac{\delta \tilde{E}(\Delta\psi^o)_{max}}{\delta(\Delta\psi^o)} = 0 \quad (8.23)$$

$$0 = \sum_{i=1}^{n'} (-\sin(\Delta\psi^o)(\tilde{x}_{ref} \cdot \tilde{x}_{neu} + \tilde{y}_{ref} \cdot \tilde{y}_{neu}) + \cos(\Delta\psi^o)(\tilde{y}_{ref} \cdot \tilde{x}_{neu} + \tilde{x}_{ref} \cdot \tilde{y}_{neu})) \quad (8.24)$$

~~daraus~~ Draus folgt für  $\Delta\psi^o$

$$\Delta\psi^o = \arctan\left(\frac{\sum_{i=1}^{n'} (\tilde{y}_{ref} \cdot \tilde{x}_{neu} + \tilde{x}_{ref} \cdot \tilde{y}_{neu})}{\sum_{i=1}^{n'} (\tilde{x}_{ref} \cdot \tilde{x}_{neu} + \tilde{y}_{ref} \cdot \tilde{y}_{neu})}\right) \quad (8.25)$$

Mit dem Ergebnis für  $\Delta\psi^o$  kann unter Verwendung von Gleichung 8.19 die Translation T bestimmt werden.

$$T = c_{ref} - M_z^o(\Delta\psi^o) \cdot c_{neu} \quad (8.26)$$

Somit sind Translation und Rotation bestimmt.

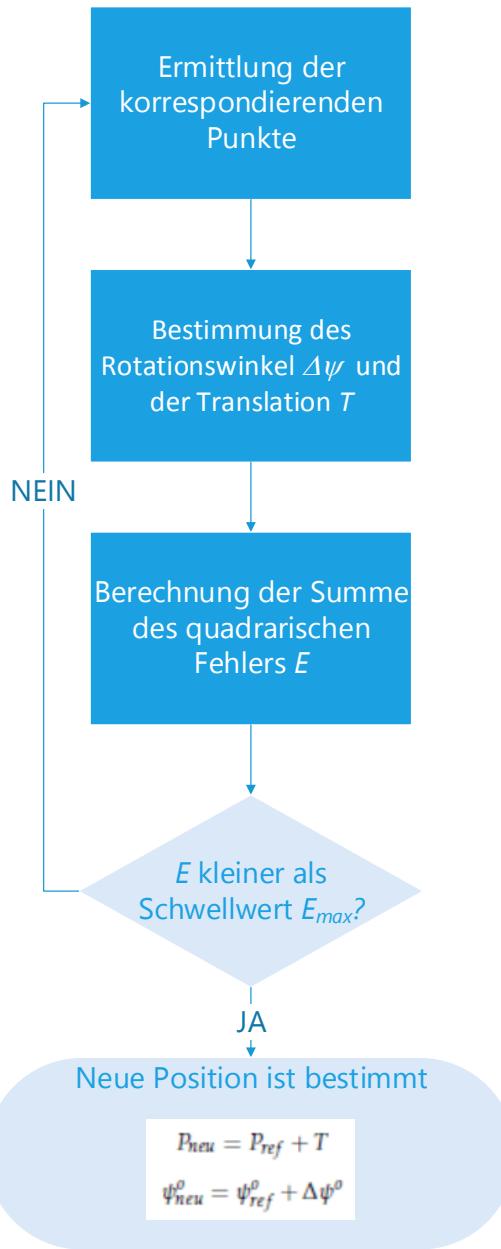
~~Schritt 3, Ermittlung der Summe der quadratischen Fehler  $E(M_z^o(\Delta\psi^o), T)$~~

~~Die aus der Formel 8.25 stammende Rotation und die mit Hilfe der Gleichung 8.26 berechneten Translation werden in die Formel des quadratischen Fehlers eingesetzt für die Summe der kleinsten Quadrate bestimmt.~~

- ~~Schritt 43~~, Vergleich von  $E(M_z^o(\Delta\psi^o), T)$  mit Schwellwert  $E_{max}$

~~Der in Vorgang 4 berechnete quadratische Fehler wird mit Die kleinste Summe der quadratischen Fehler für die ermittelte Transformation wird mit dem Schwellwert des Maximal zulässigen Fehlers  $E_{max}$  verglichen. Wird unterschritten Bei Unterschreitung ist das Abbruchkriterium erfüllt, handelt es bei Es handelt sich bei der Rotation  $\Delta\psi^o$  und der Translation T Werte die Transformation bzw. um die Transformation, welche die neue Position  $P_{neu}$  mit einer ausreichenden Genauigkeit beschreiben beschreibt. Ist das Kriterium beginnt nicht erfüllt, werden die Transformationswerte auf den Scan angewendet und der ICP-Algorithmus bei Vorgang beginnt iterative bei Schritt 1 und bestimmt neue Korrespondenzen von Neuem.~~

Für die Positionsbestimmung im n-frame wird mit der ersten Position  $P_{ref}$  der Bezugspunkt des Navigationskoordinatensystems gesetzt. Darauf aufbauend wird, werden die weiteren Translationen bzw. Rotationen addiert.



**Abbildung 8.5:** Flussdiagramm ICP-Algorithmus

$$P_{neu} = P_{ref} + T \quad (8.27)$$

$$\psi_{neu}^o = \psi_{ref}^o + \Delta\psi^o \quad (8.28)$$

Anzumerken ~~sei hier das ist hier, dass~~  $P_{ref}$  nicht im Ursprung verweilt, sondern sich auf den Referenzscan  $S_{ref}$  bezieht. Dieser kann ~~der vorige den vorigen~~ Umlauf des aktuellen Scans  $S_{neu}$  darstellen. Oder einen in der nahen Vergangenheit liegenden Scan, bei dem die Summe der quadratischen Fehler sehr gering war.

Der ~~vorgestellt~~ vorgestellte ICP-Algorithmus kann weiterhin verbessert werden. ~~So kann anstelle~~ Anstelle der Point-to-Point Arithmetik in Vorgang Schritt 1 kann eine Point-to-Line Arithmetik zur Ermittlung der Konvergenzpunkte angewendet werden. Mit diesem Thema beschäftigt sich das Paper [5]. Außerdem ist möglich, über die Inertialsenorik ausgehend von  $P_{ref}$  eine neue Position  $P_{imu}$  zu bestimmen. In  $P_{imu}$  wird der neue  $S_{neu}$  gelegt. So muss lediglich der Fehler der IMU-Positionsschätzung über den ICP-Algorithmus eliminiert werden. Dies ist ~~besonders nützlich von Vorteil~~, wenn zwei Scans weit auseinander liegen, wie zum Beispiel in Abbildung 8.4b. Es vereinfacht Vorgang Schritt 1. Behandelt wird dies unter anderem in [16].

Mit der Beschreibung Scanmatchingverfahren ist der letzte Baustein zur Lokalisierung des Quadrocopters in der horizontalen Ebene eines geschlossenen Raums mittels eines 2D Laser geliefertworden. Darauf aufbauend, ist es möglich eine Positionsregelung zu implementieren.

# KAPITEL 9

---

## Grundlagen

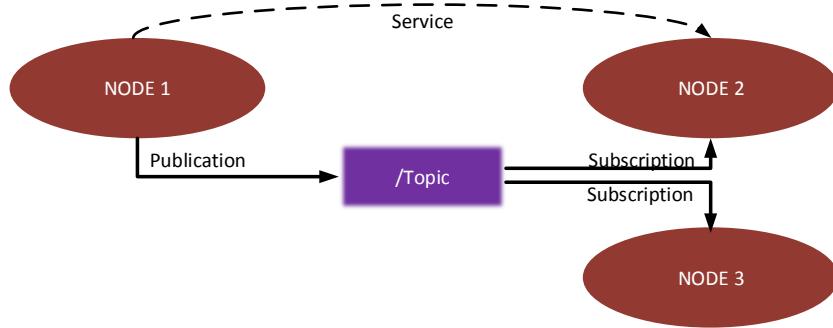
---

Das Kapitel Grundlagen behandelt die Themen, die in mehreren Abschnitten ~~dieser für diese~~ Arbeit relevant sind. Dabei handelt es sich um das Robot Operation System, die verwendeten Koordinatensysteme und die Transformation zwischen ihnen.

### 9.1 Das Robot Operation System *ROS*

Ziel dieses Unterkapitel ist es das Opensource Betriebssystem *ROS* ~~vorzustellen. Wie es aufgebaut ist und welche Vorzüge es besitzt mit seinem Aufbau und seinen Vorzügen vorzustellen.~~

*ROS* stellt dem Softwareentwickler Bibliotheken und Werkzeuge zur Verfügung, ~~die Helfen um~~ Roboteranwendungen zu erstellen. Das auf einem *IP*-basierende modulare Kommunikationsframework ermöglicht die Verknüpfung von Anwendungssoftware, Sensoren und Aktoren, sogar unter mehreren Robotern. Die Grundlage dafür ist die sogenannte Hardwareabstraktion. ~~Dabei wird durch hardwarespezifische Module erreicht, das Durch hardwarespezifische Module wird erreicht, dass~~ Komponenten unterschiedlicher Hersteller miteinander verbunden werden können. In ~~unserem Fall der Anwendung den~~ Hokuyo Lasersanner und ~~den~~ AscTec FCU. ~~Außerdem Des Weiteren~~ ermöglicht es eine hardwareunabhängige Programmierung, die in den Programmiersprachen C/C++ oder in Python erfolgen kann. Jede Hardwareabstraktion oder Anwendung wird als Node, bzw. Konten bezeichnet und läuft als eigener Prozess.

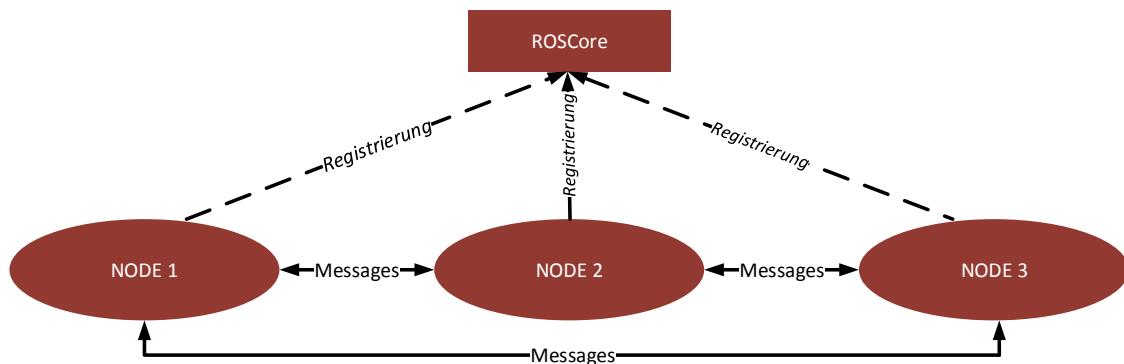


**Abbildung 9.1:** Kommunikation von Nodes über Topics und Services

Der Austausch von Daten zwischen den Nodes erfolgt über so genannte Topics (Abbildung 9.1]. Dabei werden von den Knoten Nachrichten (engl. Messages) in Topics gepostet und somit veröffentlicht (publication). Benötigt ein weiterer Knoten den Inhalt dieses **Topic**, kann er es abonnieren (subscription). Sobald die Nachricht im Knoten aktualisiert wurde, wird sie in den abonnierenden Knoten übertragen. Dabei sind Knoten nicht auf ein Topic beschränkt, es können beliebig viele Topics beschrieben oder empfangen werden. Alternative zu dieser Art der asynchronen Datenübertragung, bietet ROS die Möglichkeit einer **Synchrone synchronen** Kommunikation zwischen zwei Nodes über Services. Dabei Auf diese Weise wird auf einem Knoten ein Service gestartet. Dieser dient als Server und agiert nach dem Anfrage-Antwort-Prinzip. Schickt ein anderer Knoten eine Anfrage, wird ihm die geforderte Nachricht zu gesendet.

Anzumerken ist, dass durch das verwendete IP-Protokoll keine deterministische Versendung der Nachrichten nicht gewährleistet ist, da es sein kann, dass Nachrichten gleichen Types. Es ist möglich, dass Nachrichten gleichen Typs in Paketen zusammengefasst werden. Bei der Programmierung empfiehlt es sich daher auf Topics mit einem Zeitstempel (engl. timestamp) zurückzugreifen. Die Echtzeitfähigkeit des ROS ist durch allerdings dadurch nicht gefährdet.

Der wohl größte Vorteil von ROS ist die ständig wachsende Community. So stellen Forscher aus der ganzen Welt ihre Algorithmen und Hardwareabstraktionen zur Verfügung. Dadurch ist es möglich Folglich ist es realisierbar, bei der Erstellung einer Roboteranwendung auf Bausteine zurück zugreifen, die ohne diese Plattform selbst zu implementieren wären. Abgesehen davon, stellt ROS eine Vielzahl von Hilfsmitteln, wie zum Beispiel die Transferfunktion (/tf) bereit. Hier, bereit. An dieser Stelle lassen sich



**Abbildung 9.2:** Registrierung der Knoten

Koordinatensysteme definieren. Die Transformation der Daten wird **dann** automatisch von *ROS* durchgeführt.

## 9.2 Einführung in die Koordinatensysteme und Koordinatentransformationen

Anhand von Koordinatensystemen und Transformationen **lässt lassen** sich die Lage von Punkten und Objekten in **einen-einem** Raum mathematisch beschreiben. **Die**, **die** Grundvoraussetzung zur Bestimmung der Position des Quadrocopters im 2D-Raum (siehe Kapitel HIER MUSS NOCH EINE REFERENZ HIN). **Außerdem**, **Ferner** ermöglicht die Einführung von Koordinatensystemen die mathematisch/physikalische Beschreibung des Quadrocopters **und stellt somit**, **und stellt** die Grundlage zur Modellbildung **und Reglerentwurf** sowie **des Reglerentwurfs dar** (siehe Kapitel so und so).

### 9.2.1 Koordinatensysteme

Über ein Koordinatensystem lässt sich ein Vektor oder die Position eines Punktes bezogen auf den Koordinatenursprung in einer zweidimensionalen Ebene, bzw. in einem dreidimensionalen Raum beschreiben. Ziel dieser Teilaufgabe ist die **Erläuterung**, **Beschreibung** der in dieser Arbeit eingeführten Koordinatensysteme.

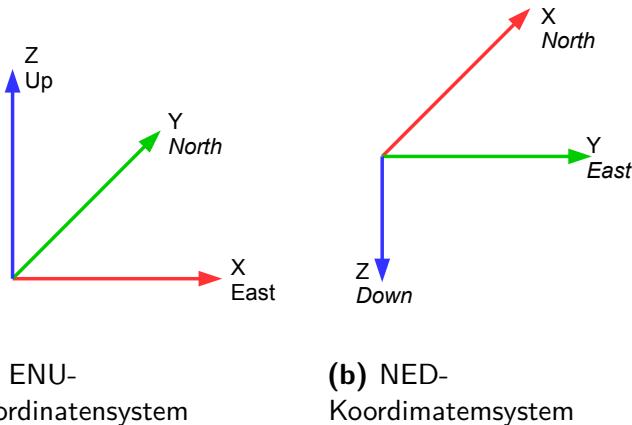


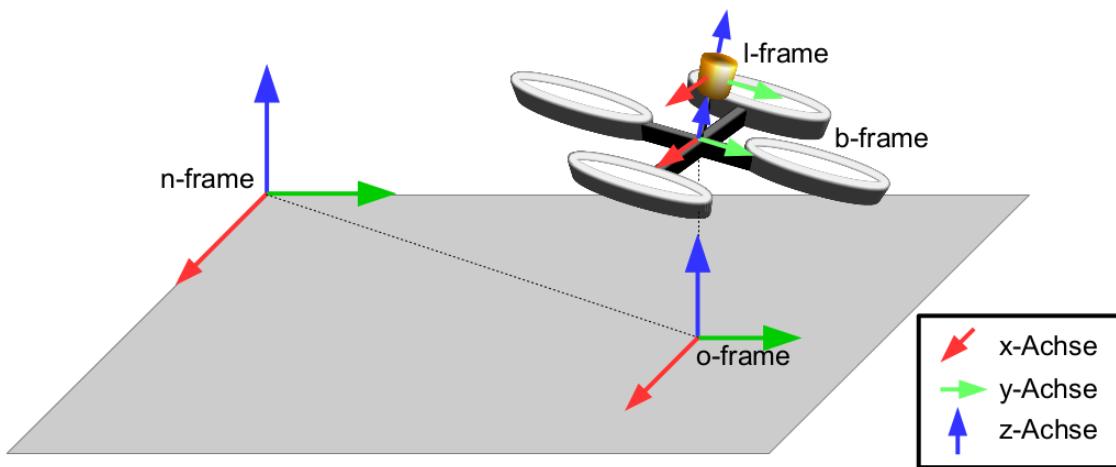
Abbildung 9.3: test

Zu Beginn werden nun zwei Konventionen bezüglich zwei Konventionen hinsichtlich der Bezugssysteme vorgestellt. Nummer eins, die in Abbildung 9.3a dargestellte *ENU* Konvention. Diese wird in vor allem bei vor allem in der Landnavigation eingesetzt. Hier In diesem Fall zeigt die z-Achse nach oben. Bei der zweiten Konvention, hauptsächlich eingesetzt in der Wasser-, Luft- und Raumfahrt eingesetzt, handelt es sich um das *NED* Bezugssystem (Abbildung 9.3b). Die z-Achse zeigt nach unten. Anzumerken ist, dass dass in dieser Arbeit die Ausrichtung der x- und y -Achse nicht wie in Abbildung 9.3 und auch der Namensgebung entsprechend den Himmelsrichtungen entspricht sondern der Namensgebung analog den Himmelsrichtungen entsprechen. Die Begriffe *ENU* und *NED* dienen hier dabei zur Beschreibung der Ausrichtung der Koordinatenachsen in Abhängigkeit der positiven z-Achse.

Bei der nun folgenden Einführung der Koordinatensysteme (Abbildung 9.4) handelt es sich ausschließlich um kartesische, das heißt d.h. orthogonale Koordinatensysteme, die nach der *ENU* Konvention ausgerichtet sind. Dies steht erstmal Das steht erstmals im Widerspruch mit dem Abschnitt zuvor, dort ist das *NED* als Koordinatensystem für Flugkörper eingeführt worden. Es ist allerdings so, dass Grundsätzlich basieren die *ROS* Koordinatensysteme auf *ENU* basieren. Deshalb Konvention. Daraus erfolgt die Wahl von Bezugssystemen mit positiver z-Achse nach oben.

Wie aus Abbildung 9.4 zu entnehmen sind vier xyz-Koordinatensysteme definiert.

- **n-frame(Lokaler Navigationsframe):** Ortsfestes Ist ein ortsfestes Koordinatensystem zur Beschreibung der Position im Raum. Da es in dieser Arbeit um die horizontale



**Abbildung 9.4:** In der Arbeit angewandte Koordinatensysteme

Positionsregelung geht, ist hier ausschließlich die xy-Ebene von Interesse. Der Ursprung des Koordinatensystems wird bei jedem Systemstart neu initialisiert. Zu beachten ist, dass dies beim vollautonomen Flug in Räumen, dabei erfolgt. Dabei beziehen sich die Sollpositionen nicht auf ein Raumkoordinatensystem mit festem Ursprung, sondern auf den beim Systemstart initialisierten Bezugspunkt. Keinen Einfluss hat diese Tatsache auf die Geschwindigkeitsregelung per Fernsteuerung, da hier an dieser Stelle die relative Bewegung von Interesse ist.

~~Es sei darauf hingewiesen~~ Anbei der Hinweis, dass die Verwendung eines xyz Navigationsframes die Krümmung der Erdoberfläche vernachlässigt. Diese ist Das erscheint legitim, da die Drohne in Gebäuden zum Einsatz kommt. Möchte man jedoch Weltweit navigieren, benötigt man ein rotationsellipsoide Koordinatensystem Erfolgt eine weltweite Navigation wird ein rotationsellipsoide Koordinatensystem [THIELECKE LITVERWEIS] benötigt.

- **b-frame(Bodyframe):** Dieses Das Koordinatensystem ist fest mit dem Rahmen des Quadrocopters verbunden. Man spricht dabei von einem Es entspricht einem körperfesten Koordinatensystem. Dabei In diesem Fall befindet sich der Ursprung des Systems im Schwerpunkt, die x-Achse zeigt in die als Vorne definierte Richtung. Die y- und z-Achse sind abhängig davon nach von der ENU Konvention angeordnet. Informationen die sich auf dieses Referenzsystem beziehen, sind unter anderen die

*IMU*-Daten. Außerdem lässt sich mit diesem System die Lage des Quadrocoters im n-frame über die Position des Nullpunkts und Drehwinkel beschreiben.

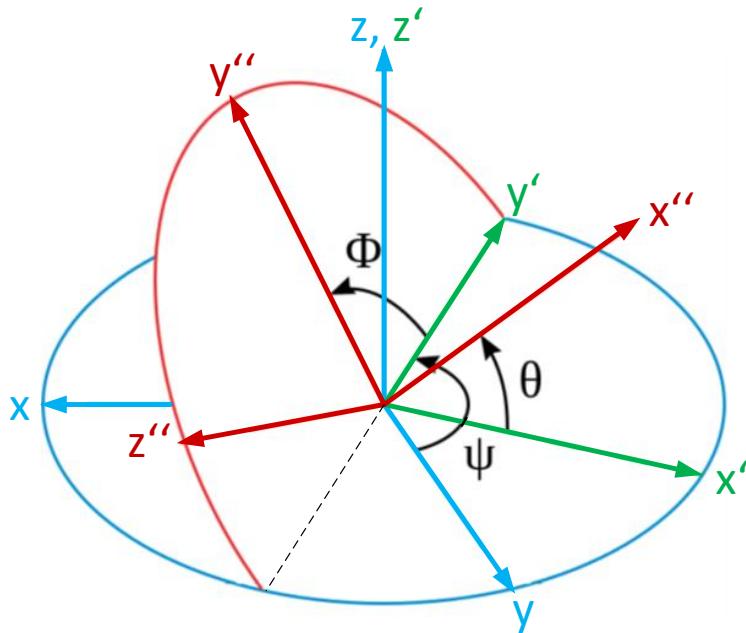
- **I-frame(laserframe):** *Ebenfalls ist ebenfalls* ein körperfestes Koordinatensystem. *In die dem, in dem die* Entfernungsmessungen des Lasers aufgetragen werden. Der Bezugspunkt liegt *dabei folglich* in der Sendequelle des Lasers. Die Ausrichtung der Achsen entspricht der des b-frames, mit Ausnahme eines Offsets in z-Richtung.
- **o-frame(Orthogonalframe):** *Hierbei handelt es Es handelt* sich um ein objektbezogenes Bezugssystem, *dessen Orientierung um seine welches mit dem b-frame verknüpft ist. Das o-frame beschreibt die Orientierung um die z-Achse und sowie* die Position des Ursprungs im n-frame abhängig von dem Orthogonal über der Ebene befindlichen b-frame ist. Dadurch *Quadrocopters in der horizontalen Ebene des n-frames. Hiermit* wird der Quadrocopter in der zu *Navigierenden navigierenden* xy-Ebene abgebildet.

*Da sich Messwerte Nachdem sich die Messwerte*, wie zum Beispiel die *IMU*-Daten oder die Laserdaten *auf unterschiedlichen, auf unterschiedliche* Koordinatensysteme beziehen, *benötigt man wird eine* Koordinatentransformation (Kapitel 9.2.2) *Mit Hilfe der erforderlich. Mit dieser Unterstützung* lassen sich die Vektoren und Koordinaten in die verschiedenen Bezugssysteme übertragen.

### 9.2.2 Koordinatentransformationen

Damit Daten eines Referenzsystems in einen anderen transformiert transformiert werden können, muss deren Orientierung zueinander beschreibbar sein. Nach [Buchholz Flugregelung] ist dies über die Rotationswinkel  $\phi$  (Rollwinkel/engl. roll), Rotation um die x-Achse sowie der Winkel  $\theta$  (Nickwinkel/engl. pitch) und  $\psi$  (Gierwinkel/engl. yaw) für die y- und z-Achse möglich erreichbar. Die Reihenfolge, um die die Achsen gedreht werden, ist dabei nicht beliebig. Sie ist in verschiedenen Konventionen festgelegt. In dieser Arbeit wird die in der Luftfahrt- und Fahrzeugtechnik gebräuchliche z,y',x"-Konvention (Abbildung 9.5) angewendet.

Das Koordinatensystem wird zu Beginn um den Winkel  $\psi$ , d.h. um die z-Achse gedreht. Daraus ergibt sich das in Abbildung 9.5 grün eingezeichnet Koordinatensystem. Dieses rotiert *man anschließend* um die Achse y', sprich den Winkel  $\theta$ . Zuletzt erfolgt eine Drehung mit dem Winkel  $\phi$  um die x"-Achse. Das Ergebnis *ist stellt* das rote x",y",z"-Koordinatensystem.


 Abbildung 9.5:  $z, y', x''$ -Konvention

~~Es sei nochmal darauf hingewiesen, dass dar. Hierbei ist zu beachten, dass~~ die Reihenfolge der Winkel einzuhalten ist, da sonst die beschriebene von der tatsächlichen Lage abweicht. Ein veranschaulichendes Beispiel ~~worin~~, wohin unterschiedliche Abfolgen bei der Rotation führen, ist in der Literatur [Literaturverzeichnis] von Herr Thielecke zu finden.

Nach Luftfahrtkonvention lässt sich eine Transformationsmatrix  $M$  aufstellen, mit der sich Vektoren und Koordinaten vom xyz-Koordinatensystem (Bsp.: n-frame) in das  $x''y''z''$ -Koordinatensystem (Bsp.: b-frame) überführen lassen. ~~Dafür benötigt man~~ Hierfür sind zunächst die drei Transformationsmatrizen notwendig, die jeweils eine Rotation um eine Koordinatenachse beschreiben [Literaturverzeichnis]. Diese sind wie folgt definiert:

- Drehung um die z-Achse mit dem Winkel  $\psi$

$$M_z = \begin{bmatrix} \cos \psi & \sin \psi & 0 \\ -\sin \psi & \cos \psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (9.1)$$

- Drehung um die y-Achse mit dem Winkel  $\theta$

$$M_y = \begin{bmatrix} \cos \theta & 0 & -\sin \theta \\ 0 & 1 & 0 \\ \sin \theta & 0 & \cos \theta \end{bmatrix} \quad (9.2)$$

- Drehung um die x-Achse mit dem Winkel  $\phi$

$$M_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \phi & \sin \phi \\ 0 & -\sin \phi & \cos \phi \end{bmatrix} \quad (9.3)$$

Aus diesen Rotationsmatrizen ~~lässt erstellt~~ sich über Matrizenmultiplikation eine Gesamttransformationsmatrix ~~aufstellen~~. Die Reihenfolge der Multiplikation entspricht der in der Konvention festgelegten Drehfolge, die von rechts nach links gelesen . Somit wird. Demzufolge er gibt sich:

$$\begin{aligned} M_{bn} &= M_x \cdot M_y \cdot M_z \\ &= \begin{bmatrix} \cos \theta \cos \psi & \cos \theta \sin \psi & -\sin \theta \\ \sin \phi \sin \theta \cos \psi - \cos \phi \sin \psi & \sin \phi \sin \theta \sin \psi + \cos \phi \cos \psi & \sin \phi \cos \theta \\ \cos \phi \sin \theta \cos \psi + \sin \phi \sin \psi & \cos \phi \sin \theta \sin \psi - \sin \phi \cos \psi & \cos \phi \cos \theta \end{bmatrix} \end{aligned} \quad (9.4)$$

Mit Hilfe dieser Transformationsmatrix ~~lässt überträgt~~ sich jetzt ein Vektor zum Beispiel aus dem n-frame ins b-frame ~~übertragen~~.

$$\begin{bmatrix} x^b \\ y^b \\ z^b \end{bmatrix} = M_{bn} \cdot \begin{bmatrix} x^n \\ y^n \\ z^n \end{bmatrix} \quad (9.5)$$

Handelt es sich ~~bei~~ um eine Koordinate, ist zusätzlich ~~noch~~ der Abstand der Koordinatenursprünge zu addieren. Für die Rücktransformation ~~muss wird~~ die Gesamttransformationsmatrix transponiert ~~werden~~. Daraus folgt:

$$\begin{bmatrix} x^n \\ y^n \\ z^n \end{bmatrix} = M_{bn}^T \cdot \begin{bmatrix} x^b \\ y^b \\ z^b \end{bmatrix} = M_{nb} \cdot \begin{bmatrix} x^b \\ y^b \\ z^b \end{bmatrix} \quad (9.6)$$

Nun Dabei lassen sich Vektoren in beide Richtungen in die verschiedenen Bezugssysteme überführen. Nachteil der Methode mit Eulerwinkel ist, das dass diese auf Grund der trigonometrischen Funktionen nur für Winkel  $\phi, \theta, \psi = \{x \in \mathbb{R} | -\pi \leq x \leq \pi\}$  eindeutig durchführbar ist sind. Wird dieser Bereich überschritten, lässt sich die Lage über Quaternionen beschreiben. Durch die Beschreibung der dreidimensionalen Orientierung in einem vierdimensionalen Raum lassen, ist die Lage auch für Rotationen um eine vielfaches von  $2\pi$  eindeutig charakterisiert. Die genaue Definition findet sich in der Literatur [19] und [4]. Da die der Definitionsbereich der Eulerwinkel für den in der Arbeit diese Arbeit im betrachteten Bereich ausreicht, ist ausschließlich die Umrechnung der in Quaternion ( $w_q, x_q, y_q, z_q$ ) angegebenen Orientierungsdaten der *IMU* in Eulerwinkel( $\phi, \theta, \psi$ ) erforderlich. Zu beachten ist, das die nun dass die folgende Umwandlung nur für die  $z, y', x''$ -Konvention Gültigkeit besitzt.

$$\phi = \arctan\left(\frac{2(y_q z_q + w_q x_q)}{w_q^2 - x_q^2 - y_q^2 + z_q^2}\right) \quad (9.7)$$

$$\theta = \arcsin(2(w_q y_q - x_q z_q)) \quad (9.8)$$

$$\psi = \arctan\left(\frac{2(x_q y_q + w_q z_q)}{w_q^2 + x_q^2 - y_q^2 - z_q^2}\right) \quad (9.9)$$

Mit dieser letzten Umrechnung sind alle Grundlagen für die Arbeit gelegt. So Auf diese Weise bilden die Koordinatensystem und Transformationen die Basis für die nachkommende Positionsbestimmung.

---

## Literaturverzeichnis

---

- [1] *Git-Repository asctec\_hl\_framework*. [https://github.com/ethz-asl/asctec\\_mav\\_framework](https://github.com/ethz-asl/asctec_mav_framework). – Eingesehen am 23.10.2014 (Zitiert auf Seite 33)
- [2] *Homepage von Ascending Technologies*. <http://www.asctec.de/>. – Eingesehen am 08.01.2015 (Zitiert auf Seite 7)
- [3] ACHTELIK, Markus ; ACHTELIK, Michael ; WEISS, Stephan ; SIEGWART, Roland: Onboard IMU and monocular vision based control for MAVs in unknown in- and outdoor environments. In: *ICRA*, 2011 (Zitiert auf Seiten 2, 33, 41, 43, 66, 76 und 83)
- [4] BUCHHOLZ, Jörg J.: Regelungstechnik und Flugregler / Hochschule Bremen. 2014. – Forschungsbericht (Zitiert auf Seiten 17, 19 und 105)
- [5] CENSI, Andrea: An ICP variant using a point-to-line metric. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. Pasadena, CA, May 2008 (Zitiert auf Seiten 30 und 96)
- [6] DEUTSCHER, Joachim: *Mehrgrößen Zustandsregelung MZR*. 2013  
(Zitiert auf Seiten 67 und 76)
- [7] DEUTSCHER, Joachim: *Regelung nichtlinearer Systeme*. 2013  
(Zitiert auf Seiten 40, 41, 44 und 51)
- [8] GARDILL, Markus ; WEIGEL, Robert: *DES Digitale Elektrische Systeme*. 2014  
(Zitiert auf Seite 60)
- [9] HOFFMANN, Frank ; GODDEMEIER, Niklas ; BERTRAM, Torsten: Attitude estimation and control of a quadrocopter. In: *IROS'10*, 2010, S. 1072–1077 (Zitiert auf Seite 34)

- [10] JANABI-SHARIFI, Farrokh ; HAYWARD, Vincent ; J. CHEN, Chung shin: *Discrete-Time Adaptive Windowing for Velocity Estimation.* 2000 (Zitiert auf Seiten 61, 63 und 64)
- [11] KALLWIES, Jan: *Höhenbestimmung und Höhenregelung bei einem Quadrocopter*, Diplomarbeit, 2013 (Zitiert auf Seiten 8, 35, 79 und 84)
- [12] LIN, Feng ; BRANDT, Robert ; SAIKALIS, George: Self-Tuning of PID Controllers by Interaction. (2000) (Zitiert auf Seiten 55 und 56)
- [13] LU, F. ; MILIOS, E.E: Robot Pose Estimation in Unknown Environments by Matching 2D Range Scans. In: *IEEE Conference on Computer Vision and Pattern Recognition*. 1994, S. 935–938 (Zitiert auf Seiten 27 und 92)
- [14] LUNZE, Jan: *Regelungstechnik 1*. Springer London, Limited, 2008 (Springer-Lehrbuch Bd. 1). – ISBN 9783540689096 (Zitiert auf Seite 40)
- [15] MAY, Stefan: *Skriptum AUT4 Mobile Robotik.* 2013 (Zitiert auf Seiten 27, 29, 92 und 93)
- [16] MORRIS, William ; DRYANOVSKI, Ivan ; XIAO, Jizhong ; MEMBER, Senior: 3D indoor mapping for micro-uavs using hybrid range finders and multi-volume occupancy grids. In: *In RSS 2010 workshop on RGB-D: Advanced Reasoning with Depth Cameras*, 2010 (Zitiert auf Seiten 23, 30, 87 und 96)
- [17] ROPPENECKER, Günter: *Regelungstechnik B.* 2012 (Zitiert auf Seiten 67 und 68)
- [18] SCHULZ, W. ; HERING, Ekbert ; KURZ, G. ; MARTIN, Rolf ; STOHRER, Martin: *Physik für Ingenieure*. Springer (Engineering Online library). <http://books.google.de/books?id=NP9mLJ11FjAC>. – ISBN 9783540351481 (Zitiert auf Seiten 37 und 38)
- [19] THIELECKE, Jörn: *Eingebettete Navigationssysteme.* 2014 (Zitiert auf Seiten 16, 19 und 105)
- [20] VOSSIEK, Martin: *Radarssysteme RAS.* 2012 (Zitiert auf Seite 56)

---

## Abbildungsverzeichnis

---

2.1	Hardwareaufbau	6
2.2	Hardwareaufbau	7
2.3	fcuplatine	9
2.4	Kommunikationsstruktur	10
3.1	Topic und Service	13
3.2	Registrierung der Knoten	14
3.3	Kovention von Koordinatensystemen	15
3.4	Koordinatensysteme	16
3.5	$z,y',x''$ -Konvention	18
4.1	Verknüpfung der Scantools	22
4.2	Projektion der Laserdaten	23
4.3	I-frame	24
4.4	Laserprojektion	26
5.1	AscTec_hl_framework in der Kommunikationstruktur	32
5.2	Kaskadenstruktur	33
5.3	Gesamtstruktur Regelung	34
5.4	Gesamtstruktur Regelung	35
5.9	Simulation Inversion	45
5.11	Simulationsergebnisse Referenzmodell	48
5.12	Referenzmodell als Vorsteuerung	50
5.14	Folgeregler	53
5.15	Grafiken Folgeregler	53
5.16	Stabilitätsgebiet	54

5.17 Selftuning Folgeregler . . . . .	57
5.18 Auswirkung der Positionsverschiebung auf Varianz der Geschwindigkeit . . . . .	59
5.19 Fensterung . . . . .	62
5.20 End-fit <i>FOAW</i> . . . . .	62
5.21 Messergebnis für <i>FOAW</i> . . . . .	66
5.22 Erweitertes Streckenmodell . . . . .	67
5.23 Beobachter . . . . .	69
5.24 Simulation Beobachter . . . . .	70
5.25 Simulierte Positionsverschiebung des Quadrocopters . . . . .	72
5.26 Simulation Gesamtmodell . . . . .	74
6.1 Positionsverschiebung des Quadrocopters . . . . .	76
6.2 Messergebnisse Flug . . . . .	78
6.3 Positionsverlauf Versuch Geschwindigkeitsregler . . . . .	80
6.4 Messergebnisse Geschwindigkeitsregelung . . . . .	82
8.1 Verknüpfung der Scantools . . . . .	86
8.2 Laserprojektion . . . . .	87
8.3 I-frame . . . . .	88
8.4 Laserprojektion . . . . .	90
8.5 I-frame . . . . .	95
9.1 Topic und Service . . . . .	98
9.2 Registrierung der Knoten . . . . .	99
9.3 Konvention . . . . .	100
9.4 Koordinatensysteme . . . . .	101
9.5 z,y',x"-Konvention . . . . .	103

---

## Tabellenverzeichnis

---

## **ANHANG A**

---

Datenblatt Hokuyo Laserscanner

---

Date: 2012.11.27

# Scanning Laser Range Finder UTM-30LX/LN Specification

△ × 2	Correction of Repeated Accuracy Representation			3	2012.11.27	Kamon	RS-0155
△ × 1	LED Display in Specificaions added			3	2012.10.23	Kamon	RS-0143
△ × 2	Important Notes on IF is added. <u>External dimension error correction.</u>			3,4	2011.7.6	Kamon	PR-6178
△ × 3	Changes in output signal			3,4,6	2010.7.26	Kamon	PR-5893
△ × 2	Correction on synchronization output			2,4	2009.5.18	Takai	PR-5647
△ × 2	Changes in laser( $\lambda$ :870nm → 905nm)			2,3	2009.4.14	Kamon	PR-5635
△ × 1	Correction			4	2008.8.18	Kamitani	PR-5503
△ × 1	Cautions were added			6	2008.5.1	Kamitani	PR-5466
Symbol	Amendment Details			Amendment	Date	Amended by	Number
Approved by	Checked by	Drawn by	Designed by	Title	<b>UTM-30LX/LN</b> Specification		
MORI	KAMITANI	KAMON	HINO		Drawing No.	<b>C-42-3615</b>	

## 1. Introduction

### 1.1 Operation principles 905nm $\triangle$

UTM-30LX/LN use laser source ( $\lambda = 870\text{nm}$ ) to scan  $270^\circ$  semicircular field (Figure 1). It measures distance to objects in the range and co-ordinates of those point calculated using the step angle. Sensor's measurement data along with the angle are transmitted via communication channel. Laser safety class 1.

Sensor is divided into two types depending upon the type of output.

### 1.2 Type

#### 1.2.1 U TM-30LX

Synchronous output signal is available. The timing chart of this signal is shown in section 6 (Figure 3).  $\triangle$  This synchronous signal can be obtain at each scan. These are mainly intended for robotic applications.

#### 1.2.2 UTM-30LN

It outputs warning signal whenever there is any object in the preset area. These are mainly intended for area protection.

## 2. Structure (Laser range figure)

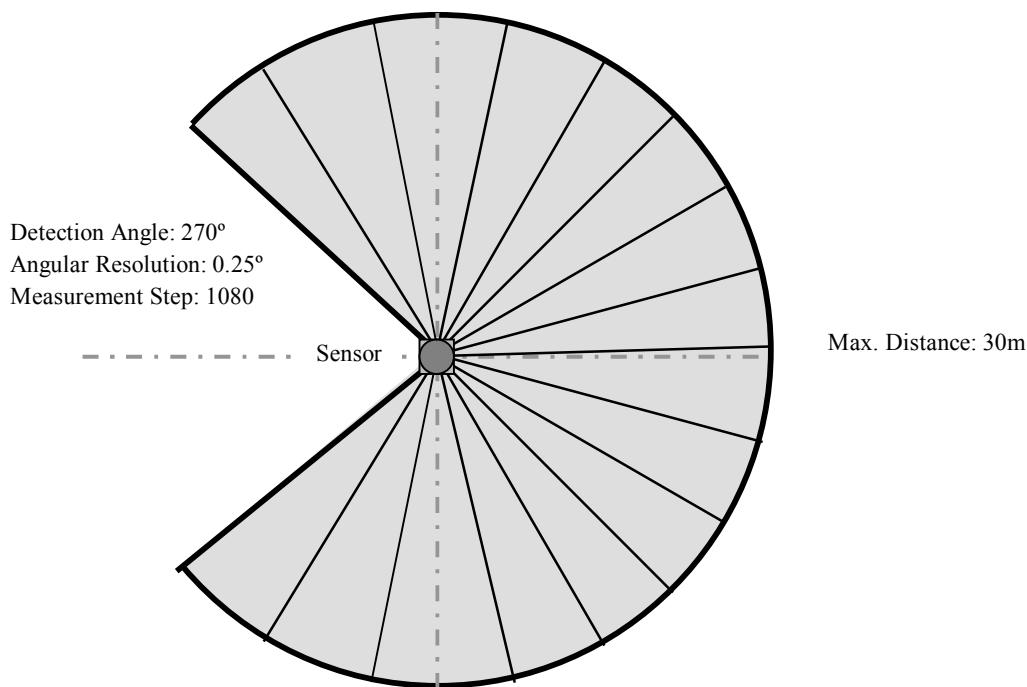


Figure 1

## 3. Important note

- This sensor is not a safety device/tool
- This sensor is not for use in military applications
- Read specifications carefully before use.

Title	UTM-30LX/LN Specification	Drawing No	C-42-3615	2/6
-------	---------------------------	------------	-----------	-----

## 4. Specifications

Product Name	Scanning Laser Range Finder	
Model	<b>UTM-30LX</b>	<b>UTM-30LN</b>
Light Source	Laser Semiconductor $\lambda = 870\text{nm}$ , 905nm	△ Laser Class 1
Supply Voltage	12VDC $\pm 10\%$	
Supply Current	Max: 1A, Normal : 0.7A	
Power Consumption	Less than 8W	
Detection Range and Detection Object	Guaranteed Range: 0.1 ~ 30m (White Kent Sheet) Maximum Range : 0.1 ~ 60m Minimum detectable width at 10m : 130mm (Vary with distance)	
Accuracy	Under 3000lx : White Kent Sheet: $\pm 30\text{mm}^{\ast 1}$ (0.1m to 10m) Under 100000lx : White Kent Sheet: $\pm 50\text{mm}^{\ast 1}$ (0.1m to 10m)	
Measurement Resolution and Repeated Accuracy	1mm 0.1 – 10m : $\sigma < 10\text{mm}$ , 10 – 30m : $\sigma < 30\text{mm}$ (White Kent Sheet) Under 3000lx : $\sigma < 10\text{mm}^{\ast 1}$ (White Kent Sheet up to 10m) ▲ Under 100000lx : $\sigma < 30\text{mm}^{\ast 1}$ (White Kent Sheet up to 10m) ▲	
Scan Angle	270°	
Angular Resolution	0.25° (360°/1440)	
Scan Speed	25ms (Motor speed : 2400rpm)	
Interface	USB Ver2.0 Full Speed (12Mbps)	
Output	Synchronous Output 1- Point	Detection Output 1- Point ▲
LED Display △	Green : Power supply Red : Normal Operation (Continuous), Malfunction (Blink)	Power supply Object detection inside area (Continuous) Malfunction (Blink)
Ambient Condition (Temperature, Humidity)	-10°C ~ +50°C Less than 85%RH (Without Dew, Frost)	
Storage Temperature	-25~75°C	
Environmental Effect	Measured distance will be shorter than the actual distance under rain, snow and direct sunlight* <sup>2</sup> .	
Vibration Resistance	10 ~ 55Hz Double amplitude 1.5mm in each X, Y, Z axis for 2hrs. 55 ~ 200Hz 98m/s <sup>2</sup> sweep of 2min in each X, Y, Z axis for 1hrs.	
Impact Resistance	196m/s <sup>2</sup> In each X, Y, Z axis 10 times.	
Protective Structure	Optics: IP64	
Insulation Resistance	10MΩ DC500V Megger	
Weight	210g (Without cable)	
Case	Polycarbonate	
External Dimension (W×D×H)	60mm×60mm×87mm ▲ MC-40-3127	

\*<sup>1</sup> Under Standard Test Condition (Accuracy can not be guaranteed under direct sunlight.)

\*<sup>2</sup> For sensor functions, please verify the in an indoor environment of 1000 lx or less. In avoiding unnecessary disturbance cause by the raindrops, perform necessary signal processing for LX type and switch OFF the delay function for LN type.

## 5. Quality Reference Value

Vibration resistance during operation	10~150Hz 19.6m/s <sup>2</sup> Sweep of 2min in each X,Y,Z axis for 30min
Impact resistance during operation	49m/s <sup>2</sup> X, Y,Z axis 10 times
Angular Speed	$2\pi/\text{s}$ (1Hz)
Angular Acceleration	$\pi/2\text{rad}/\text{s}^2$
Life-span	5 Years (Varies with operating conditions)
Noise Level	Less than 25dB at 300 mm
Certification	FDA Approval (21 CFR part 1040.10 and 1040.11)

Title	UTM-30LX/LN Specification	Drawing No	<b>C-42-3615</b>	3/6
-------	---------------------------	------------	------------------	-----

## 6. Interface

### 6.1 Robot Cable 4 Pin

Color	Function
Brown	+12 V
Blue	0 V
Green	Synchronous Output/ Detection Output <span style="color: blue;">△</span>
White	COM Output (0V: Common to Power)

Note: 0 V of the power supply and Output is not internally connected.

Short circuit the 0V (Blue) and COM Output (White) during wiring. △ △

### 6.2 USB Connector

TYPE-A

**Note:**

SG for communication and GND are connected internally (Isolated with Input -VIN).

Isolate the device from any connection that generate electric noise.

This sensor is compatible with SCIP2.0 communication protocol standard.

### 6.3 Output circuit diagram

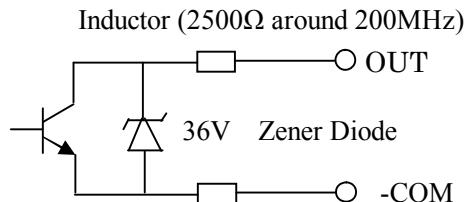


Figure 2

## 7. Control Signal

### 7.1 Synchronous Output (UTM-30LX)

1 pulse is approximately 1 ms. Output signal Synchronization timing chart is shown below. (Figure 3).

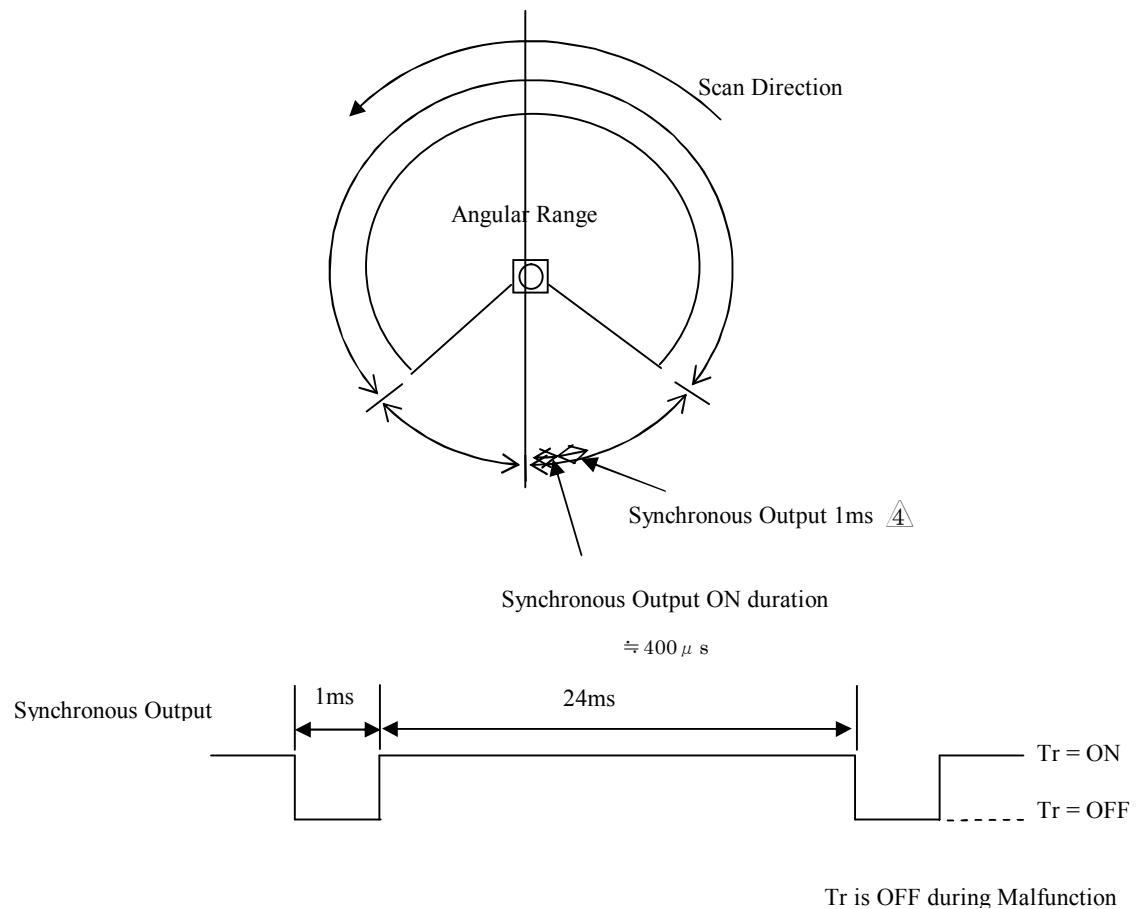


Figure 3

## 7.2 Detection Output (UTM-30LN)

When the signal is set for detection output .The signal switches OFF when obstacle exist inside the area.  
(Output signal is ON when obstacle does not exist.)

Area can be set using 3~7 co-ordinate points.

Maximum of the output delay is 128 times (3.2 sec)

Example

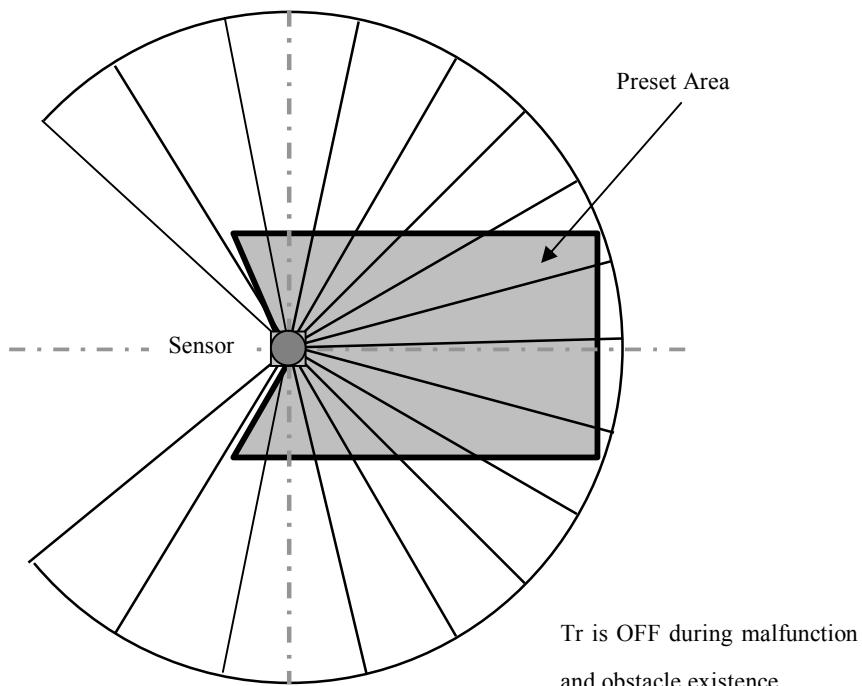


Figure 4

## 8. Malfunction Output:

1. Laser malfunction : When laser does not radiate or exceeds safety class 1.
2. Motor malfunction : When rotation speed is differ from the default value (> 25 m/s).

Synchronous/Warning signal will be turned OFF when these malfunctions are detected. Error details can be obtain via communication.

## 9. Cautions

Heat is generated as the sensor runs at a very high speed. The heat generated is concentrated at the bottom of the sensor. Please mount heatsinks or any appropriate component to release the generated heat. An aluminum plate (200 x 200 x 2) is recommended as the heatsinks.

Error could happen when 2 or more identical sensor is mounted at the same detection plane. This is because the sensor could not identify the origin of the received laser pulses. When this error occur, it will cause 1 -2 step difference, performing data filtering could overcome this problem.

Title	UTM-30LX/LN Specification	Drawing No	C-42-3615	6/6
-------	---------------------------	------------	-----------	-----

## **ANHANG B**

---

Definition flacher Mehrgrößensysteme

---

## Vorlesung: „Regelung nichtlinearer Systeme“

### Flachheit nichtlinearer Mehrgrößensysteme

#### 1. Definition

Ein nichtlineares Mehrgrößensystem  $n$ -ter Ordnung

$$\dot{x} = f(x, u) \quad \text{mit} \quad \text{rang } \frac{\partial f(x, u)}{\partial u} = p, \quad \forall x \in U(x_0), \quad \forall u \in U(u_0) \quad (1)$$

mit  $p$  Eingangsgrößen  $u$  heißt *flach*, wenn es einen fiktiven Ausgang  $y_f = [y_{f1} \dots y_{fp}]^T$  gibt, der die folgenden Bedingungen erfüllt:

1. Die Komponenten  $y_{fi}$ ,  $i = 1, 2, \dots, p$ , des fiktiven Ausgangs lassen sich als Funktion der Zustandsgrößen  $x_\nu$ ,  $\nu = 1, 2, \dots, n$ , und  $u_i$ ,  $i = 1, 2, \dots, p$ , sowie einer endlichen Anzahl von Zeitableitungen  $u_i^{(k_i)}$ ,  $k_i = 1, 2, \dots, \alpha_i$ , ausdrücken, d. h.

$$y_f = \Phi(x, u_1, \dot{u}_1, \dots, u_1^{(\alpha_1)}, \dots, u_p, \dot{u}_p, \dots, u_p^{(\alpha_p)}) = \Phi(x, u, \dot{u}, \dots, u^{(\alpha)}) \quad (2)$$

mit dem Ableitungstupel  $u^{(k)} = (u_1^{(k_1)}, \dots, u_p^{(k_p)})$  und  $k = (k_1, \dots, k_p)$ .

2. Die Zustandsgrößen  $x_\nu$ ,  $\nu = 1, 2, \dots, n$ , und die Eingangsgrößen  $u_i$ ,  $i = 1, 2, \dots, p$ , lassen sich als Funktion der  $y_{fi}$ ,  $i = 1, 2, \dots, p$ , und einer endlichen Anzahl von deren Zeitableitung  $y_{fi}^{(k_i)}$ ,  $k_i = 1, 2, \dots, \kappa_i$ , darstellen, d. h.

$$x = \psi_x(y_{f1}, \dots, y_{f1}^{(\kappa_1-1)}, \dots, y_{fp}, \dots, y_{fp}^{(\kappa_p-1)}) = \psi_x(y_f, \dots, y_f^{(\kappa-1)}) \quad (3)$$

$$u = \psi_u(y_{f1}, \dots, y_{f1}^{(\kappa_1)}, \dots, y_{fp}, \dots, y_{fp}^{(\kappa_p)}) = \psi_u(y_f, \dots, y_f^{(\kappa)}). \quad (4)$$

3. Die Komponenten von  $y_f$  sind *differentiell unabhängig*, d. h. sie erfüllen keine Differentialgleichung der Form

$$\varphi(y_f, \dots, y_f^{(\gamma)}) = 0. \quad (5)$$

Wenn der fiktive Ausgang  $y_f$  die Bedingungen 1–3 erfüllt, so ist er ein *flacher Ausgang* des Systems (1). Ist Bedingung 2 erfüllt, dann ist Bedingung 3 äquivalent zu  $\dim y_f = \dim u = p$ .

## 2. Anmerkungen zur Flachheitsdefinition

- Die Beziehungen (3) und (4) besagen, dass für jede beliebige aber hinreichend oft differenzierbare Trajektorie  $y_{f\star}(t)$  des flachen Ausgangs  $y_f$  die Trajektorien

$$\begin{aligned}x_\star(t) &= \psi_x(y_{f\star}(t), \dots, y_{f\star}^{(\kappa-1)}(t)) \\ u_\star(t) &= \psi_u(y_{f\star}(t), \dots, y_{f\star}^{(\kappa)}(t))\end{aligned}$$

Lösung des Anfangswertproblems

$$\text{DGL: } \dot{x}(t) = f(x(t), u(t)), \quad t > 0 \quad (7)$$

$$\text{AB: } x(0) = x_0 \quad (8)$$

für  $x_0 = \psi_x(y_{f\star}(0), \dots, y_{f\star}^{(\kappa-1)}(0))$  sind. Dabei muss die Systemdifferentialgleichung (7) nicht gelöst werden. Da sich somit die Systemdynamik (d.h. alle Trajektorien des Systems) durch  $y_f$  und endlich viele von dessen Zeitableitungen darstellen lässt, spricht man auch von einer (*endlichen*) *differentiellen Parametrierung* des Systems (7) durch den flachen Ausgang  $y_f$ .

- Die differentielle Unabhängigkeit der Komponenten  $y_{fi}$  des flachen Ausgangs (siehe Bedingung 3 in Abschnitt 1) bedeutet, dass die Trajektorien der Komponenten beliebig und unabhängig voneinander vorgegeben werden dürfen, da sie keine Lösung einer Differentialgleichung sind. Die Trajektorien müssen jedoch hinreichend oft differenzierbar sein, damit die Trajektorien von  $x$  und  $u$  gemäß (3) und (4) existieren.
- Die Bedingung  $\text{rang } \frac{\partial f(x,u)}{\partial u} = p$  an das System (1) stellt sicher, dass die Eingangsgrößen  $u$  des System zumindest lokal voneinander (differentiell) unabhängig sind.
- Der flache Ausgang ist nicht eindeutig, d. h. für ein System kann es mehrere flache Ausgänge geben. Verschiedene flache Ausgänge für dasselbe System können jedoch stets gemäß

$$y_f = \theta(\bar{y}_f, \dot{\bar{y}}_f, \dots, \bar{y}_f^{(s)}) \Leftrightarrow \bar{y}_f = \bar{\theta}(y_f, \dot{y}_f, \dots, y_f^{(s)}) \quad (9)$$

ineinander umgerechnet werden.

- Die Funktion  $\psi_u$  in (4) hängt von den Zeitableitungen des flachen Ausgangs bis zur Ordnung  $\kappa$  ab, da aufgrund von (3) und (1) die Beziehung  $\dot{\psi}_x = f(\psi_x, \psi_u)$  gelten muss.
- Regelgrößen  $y = h(x)$ , die nicht mit dem flachen Ausgang übereinstimmen, lassen sich stets unter Verwendung von (3) gemäß

$$y = h(\psi_x) = \psi_y(y_f, \dots, y_f^{(\beta)}) \quad (10)$$

differentiell parametrieren. Dabei gilt  $\beta = (\beta_1, \beta_2, \dots, \beta_p)$  mit  $\beta_i \leq \kappa_i - 1$ .

- Wenn Bedingung 2 erfüllt ist, dann lässt sich (4) als  $p$  Differentialgleichungen für den flachen Ausgang  $y_f$  in Abhängigkeit von  $u$  interpretieren. Folglich kann der flache Ausgang  $y_f$  nicht die Differentialgleichung (5) erfüllen, da er Lösung der Differentialgleichung (4) ist.

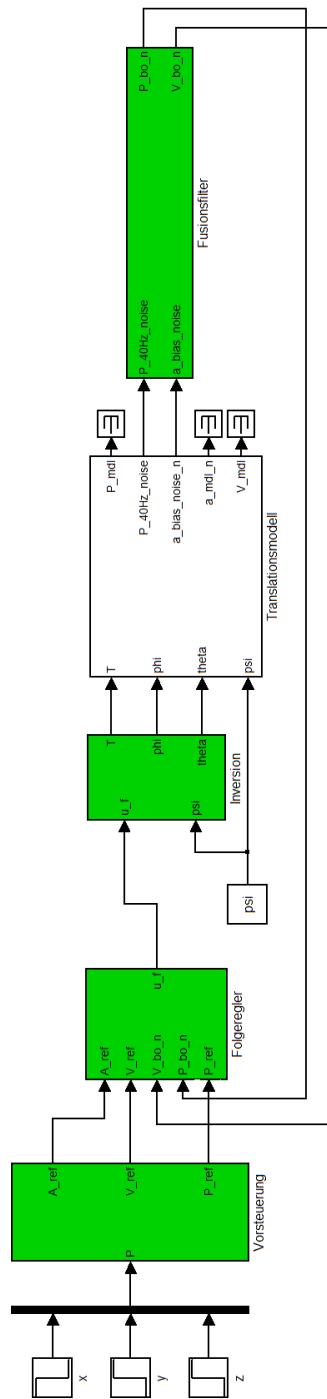
# **ANHANG C**

---

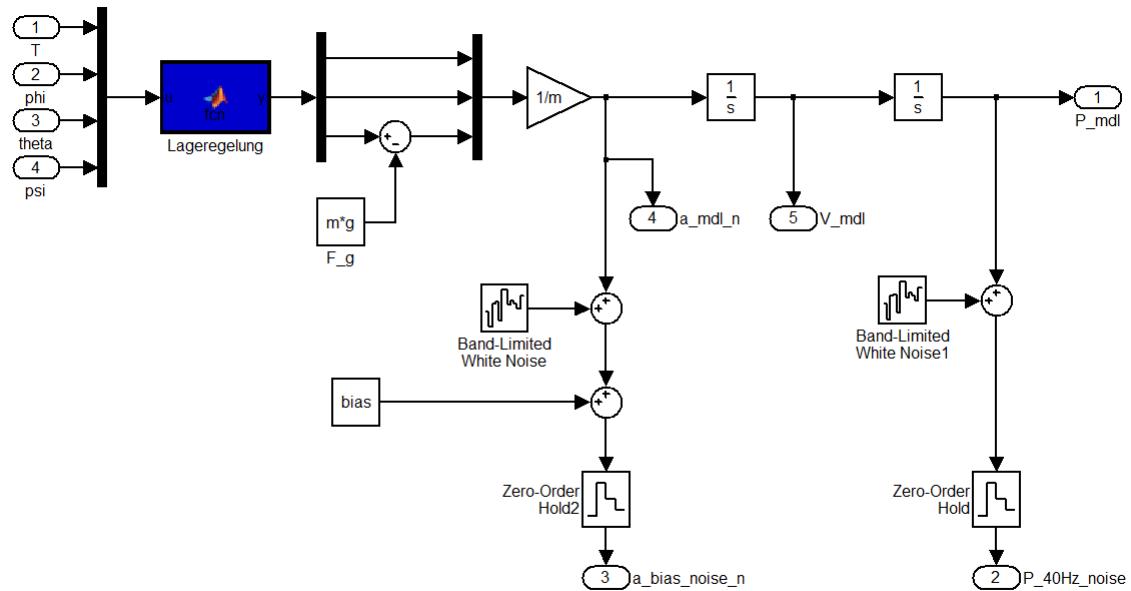
## **Simulinkmodell**

---

## C.1 Gesamtübersicht Positionsregelung



## C.2 Translationsmodell



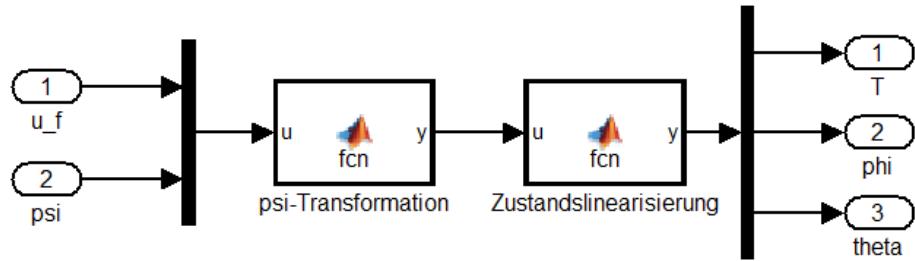
### ■ Lageregelung

```

1   function y = fcn(u)
2     T      = u(1);
3     phi    = u(2);
4     theta  = u(3);
5     psi    = u(4);
6
7     Mt = [cos(theta) * cos(psi) sin(phi) * sin(theta) * cos(psi) - cos(
6       phi) * sin(psi) cos(phi) * sin(theta) * cos(psi) + sin(phi) *
6       sin(psi); cos(theta) * sin(psi) sin(phi) * sin(theta) * sin(psi)
6       + cos(phi) * cos(psi) cos(phi) * sin(theta) * sin(psi) - sin(
6       phi) * cos(psi); -sin(theta) sin(phi) * cos(theta) cos(phi) *
6       cos(theta);];
8     y = Mt * [0; 0; T];

```

## C.3 Modell der Inversion



- $\psi$ -Transformation

```

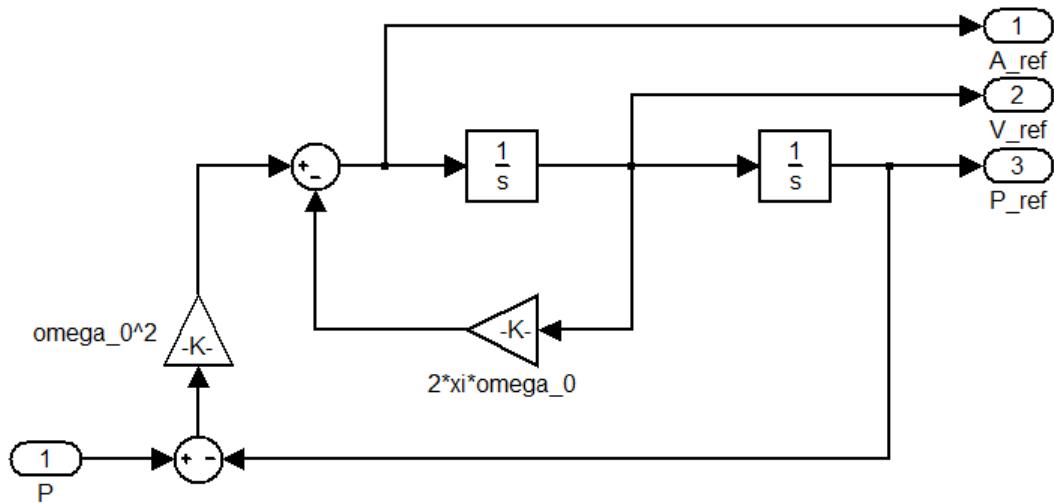
1 function y = fcn(u)
2   a_nx = u(1);
3   a_ny = u(2);
4   a_nz = u(3);
5   psi = u(4);
6
7   Mz = [cos(psi) sin(psi) 0; -sin(psi) cos(psi) 0; 0 0 1];
8   y = Mz * [a_nx;a_ny;a_nz];
  
```

- Zustandslinearisierung

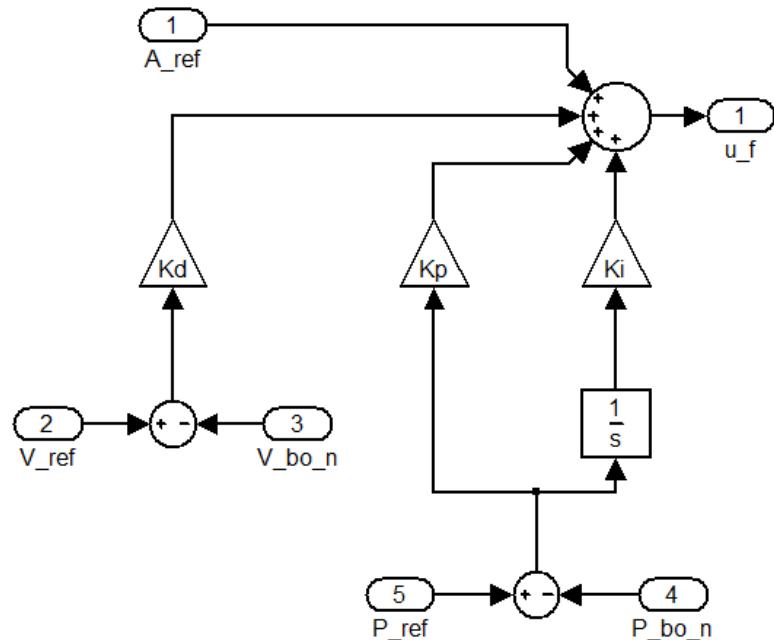
```

1 function y = fcn(u)
2   a_ox = u(1);
3   a_oy = u(2);
4   a_oz = u(3);
5   m = 1.863;
6   g = 9.81;
7
8   T      = m*(sqrt(a_ox^2+a_oy^2+(a_oz+g)^2));
9   phi    = asin(-(a_oy*m)/T);
10  theta  = atan(a_ox/(a_oz+g));
11
12  y = [T;phi;theta];
  
```

## C.4 Modell der Vorsteuerung



## C.5 Modell der Folgeregelung



## C.6 Modell des Fusionsfilters

