Julius-Maximilians-Universität Würzburg
Fakultät für Mathematik und Informatik

Informationstechnik für Luft- und Raumfahrt
Lehrstuhl für Informatik 8 Prof. Dr. Sergio Montenegro





Bachelorarbeit

Laserscanner basierte Hinderniserkennung für einen autonomen Quadrokopter

Vorgelegt von

Robin Leblebici

Matr.-Nr.: 1880418

Prüfer: Prof. Dr. Sergio Montenegro

Betreuende wissenschaftliche Mitarbeiter: Dipl.-Ing. Nils Gageik

Würzburg, 18. 09. 2015

Erklärung

Ich versichere, dass ich die vorliegende Arbeit einschließlich aller beigefügter Materialien selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe.
Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder unveröffentlichten Werken entnommen sind, sind in jedem Einzelfall unter Angabe der Quelle deutlich als solche kenntlich gemacht. Die Arbeit ist in gleicher oder ähnlicher Form noch nicht als Prüfungsarbeit eingereicht worden.
Mir ist bekannt, dass Zuwiderhandlungen gegen diese Erklärung und bewusste Täuschungen die Benotung der Arbeit mit der Note 5.0 zur Folge haben kann.

Würzburg, 18. 09. 2015

_____

Vorname Nachname

Aufgabenstellung

Laserscanner basierte Hinderniserkennung für einen autonomen Quadrokopter

Die Fortschritte im Bereich Sensorik, Aktuatorik und Mikrotechnik ermöglichen heutzutage den kostengünstigen Bau kleiner, unbemannter Luftfahrzeuge (UAV, unmanned aerial vehicle, Drohne) wie Quadrokopter. Die Forschung und Entwicklung dieser Systeme wurde in den letzten Jahren aufgrund der vielfältigen Anwendungsmöglichkeiten stark vorangetrieben. Wenngleich im Bereich UAV viel geforscht wurde, ist das Thema Autonomes Flugobjekt längst noch nicht vollständig behandelt. Insbesondere der Indoor-Betrieb ist aufgrund von Hindernissen und fehlender absoluter Positionsstützung durch GPS eine Herausforderung.

Der Aufbau eines eigenen autonomen Systems wird daher am Lehrstuhl Aerospace Information Technology der Uni Würzburg erforscht und erprobt. Im Rahmen dieses Forschungsvorhabens ist eine geeignete Signalverarbeitung zur Hinderniserkennung für den bereits integrierten Laserscanner zu erarbeiten und zu integrieren. Die Daten sollen der Kollisionsvermeidung und / oder dem Mapping dienen. Die bisherige Lösung zur Hinderniserkennung mittels Laserscanner ist noch nicht zur Kollisionsvermeidung geeignet.

Es ist ein Konzept zur Signalverarbeitung zu erarbeiten und in das bestehende System zu integrieren bzw. dieses darum zu erweitern. Die Lösung sollte nach Möglichkeit kompatibel zu bisherigen Ansätze zur Signalverarbeitung und Kollisionsvermeidung sein bzw. bisherige Methoden sind entsprechend des erstellten Konzeptes anzupassen. Die Lösung mag allein der Kollisionsvermeidung dienen oder dem bereits vorhandenen oder einem neu entwickelten Mapping. Zur Aufgabe gehörte eine ausführliche Dokumentation und aussagekräftige Evaluierung.

Aufgabenstellung (Stichpunktartig):

- Konzept Signalverarbeitung zur Hinderniserkennung
- Implementierung Signalverarbeitung
- Integration in Kollisionsvermeidung oder Mapping (Optional eins oder beides)
- Evaluierung
- Dokumentation

Zusammenfassung

Die vorliegende Arbeit befasst sich mit der Integration eines Systems zur Laserscanner basierten Hinderniserkennung auf einem autonomen Quadrokopter. Explizit wird der Histogramm Filter als Konzept zur Signalverarbeitung von Laserscannerdaten vorgestellt und seine Implementierung beschrieben. Ferner wird eine Methode zur dynamischen Objekterkennung vorgestellt und, unter Verwendung des rekursiven Bayes Filters und Anwendung eines Occupancy Grids, ein Konzept zur Hinderniserkennung beschrieben. Zusätzlich wird die Anbindung der Hinderniserkennung an ein System zur Kollisionsvermeidung dokumentiert. Des Weiteren wurde ein System zur Kartierung auf Basis des Robot Operating System (ROS) implementiert und evaluiert.
Als Quadrokopter wurde der am Lehrstuhl VIII für Informatik der Universität Würzburg entwickelte AQopterI8 eingesetzt. Als Laserscanner wurde der von der Firma RoboPeak hergestellte RPLidar verwendet und die Integration in das bestehende System beschrieben.
Das vorgestellte System ermöglicht eine autonome Hinderniserkennung mit einem Laserscanner. Darüber hinaus wird ein am Lehrstuhl VIII für Informatik der Universität Würzburg entwickelter Regler zur Kollisionsvermeidung verwendet und evaluiert, welcher das Halten eines fixen Abstands zu einem Hindernis ermöglicht.

Inhaltsverzeichnis

1 Einleitung **1**

1.1 Aufbau der Arbeit 2

2 Stand der Technik **3**

2.1 Überblick 3

2.2 Sensoren zur Hinderniserkennung 3

2.2.1 Messverfahren 3

2.2.2 Messeigenschaften 5

2.2.3 Übersicht aktueller Sensortechnik 6

2.3 Bestehende Konzepte und Projekte 9

2.3.1 Simultane Lokalisierung und Kartenerstellung 9

2.3.2 Kartenunabhängige Hinderniserkennung und Kollisionsvermeidung . . . 13

3 Konzept **15**

3.1 Überblick 15

3.2 Problematik und Zielsetzung 16

3.3 Vorverarbeitung 17

3.4 Histogramm Filter 19

3.4.1 Überlagernde Felder 20

3.4.2 Überlagernde Sektoren 21

3.4.3 Zeitliche Betrachtung 22

3.5 Nachbearbeitung 22

3.6 Dynamische Erkennung von Objekten 23

3.7 Rekursiver Bayes Filter 25

4 Implementierung **28**

4.1 Überblick 28

4.2 Softwaredesign 28

4.2.1 Implementierung des Histogram Filters 29

4.2.2 Zwischenwertberechnung 30

4.2.3 GUI-Design 31

4.3 Robot Operating System 32

4.4 Verwendete Hardware 33

4.4.1 Robopeak Lidar 33

4.4.2 AQopterI8 34

4.4.3 Hardwareintegration 35

5 Evaluierung **38**

5.1 Überblick 38

5.2 Histogramm Filter basierte Hinderniserkennung 38

5.2.1 Testaufbau 38

5.2.2 Vergleich der Histogramm Filter Klassen 39

5.2.3 Diskrepanz zu minimalem Sektorwert 42

5.2.4 Zwischenwertberechnung 43

5.2.5 Auswirkung der Parameter 44

5.3 Kartierung 46

5.3.1 Testaufbau 46

5.3.2 Versuchsdurchführung 46

5.4 Kollisionsvermeidung 48

5.4.1 Anfängliche Problematik 48

5.4.2 Versuchsdurchführung 50

5.4.3 Auswertung des Regelverhaltens 50

6 Diskussion und Ausblick **51**

7 Literaturverzeichnis **53**

1 Einleitung

Die Anwendungsmöglichkeiten moderner UAVs, Quadrokopter und anderer Multikopter-Drohnen-Systeme sind sehr vielfältig und die Komplexität der Einsatzgebiete nimmt stetig zu. So werden UAVs heutzutage zur Erkundung und Kartierung a priori unbekannter Umgebungen eingesetzt, zu Luftbildaufnahmen genutzt und können in für den Menschen nicht zugängliche oder gefährliche Umgebungen vordringen. Auch im Bereich der Vermessungstechnik finden UAVs immer häufiger Anwendung, sei es zur Inspektion von Gebäudefassaden, Windkraftanlagen oder der Vermessung archäologischer Grabungsstätten sowie historischer Monumente [34].

Für diese und viele weitere Anwendungsfälle bietet sich die Nutzung von assistiven kollisionsvermeidenden Systemen auf heteronom gesteuerten UAVs an, um den Piloten bei der Steuerung in komplexen Umgebungen und schwierigen Witterungsverhältnissen zu unterstützen. Auch der zunehmende Gebrauch privater UAVs von zum Teil unerfahrenen Piloten birgt Sicherheitsrisiken, die über ein integriertes Kollisionsvermeidungssystem minimiert werden können. Gerade im autonomen Flug besteht die Notwendigkeit einer zuverlässigen Hinderniserkennung und Kollisionsvermeidung für UAVs und ist aus ersichtlichen Gründen unabdingbar.

Die Fortschritte im Bereich der Mikroelektronik und Sensorik ermöglichen es heutzutage eine Vielzahl an Sensoren auf UAVs zur Hinderniserkennung zu nutzen, weil sie mittlerweile besonders kostengünstig und leicht sind. Allgemein basieren viele Systeme, die Laserscanner zur Hinderniserkennung nutzen, auf der Erstellung einer Karte der gesamten Umgebung und werden selten zur direkten Hinderniserkennung und Kollisionsvermeidung eingesetzt. Aus diesem Grund wird sich diese Arbeit mit der Thematik beschäftigen, indem ein System vorgestellt und entwickelt wird, bei dem auf die Verwendung einer Karte der Umgebung verzichtet wird.

1.1 Aufbau der Arbeit

In Kapitel *Stand der Technik* soll zunächst ein Überblick über aktuelle Sensortechnik zur Hinderniserkennung gegeben werden, sowie ihrer Funktionsweise und Vor- und Nachteile erläutert werden. Zusätzlich werden bestehende Konzepte und Projekte vorgestellt, welche bereits erfolgreich Systeme zur Kollisionsvermeidung auf UAVs implementierten.

Im darauf folgenden Kapitel *Konzept* sollen Methoden zur Signalverarbeitung von Laserscannerdaten vorgestellt werden. Explizit wird in diesem Kapitel auf die Verwendung des Histogramm Filters und seinen Erweiterungen eingegangen, eine Methode zur dynamischen Erkennung von Objekten beschrieben und die Verwendung des rekursiven Bayes-Filter in Kombination mit einem Occupancy Grids vorgestellt.

In Kapitel *Implementierung* wird der für diese Arbeit verwendete Laserscanner RPLidar vorgestellt und auf die Integration des Laserscanners in das bereits bestehende System der AQopterI8 Quadrokopter eingegangen. Des Weiteren wird die softwareseitige Implementierung des Histogramm-Filters erklärt, sowie eine Lösung zur Kartierung basierend auf dem Robot Operating System (ROS) vorgestellt.

Das Kapitel *Evaluierung* befasst sich mit der Auswertung der implementierten Konzepte. Es werden die durchgeführten Testreihen und Experimente zur Findung der optimalen Parameter dokumentiert und die Ergebnisse des Mappings präsentiert.

Im letzten Kapitel *Diskussion und Ausblick* werden die Ergebnisse der Arbeit zusammengefasst und ein Ausblick über mögliche Optimierungen und Erweiterungen des entwickelten Systems gegeben.

2 Stand der Technik

2.1 Überblick

Im folgenden Abschnitt 2.2 sollen die Vor- und Nachteile, als auch die Funktionsweise der gängigsten Sensoren veranschaulicht werden. Außerdem werden in Abschnitt 2.3.1 bestehende Projekte und Algorithmen zur Kartierung und Lokalisierung erläutert, sowie in Abschnitt 2.3.2 bestehende Konzepte und Techniken zur Kollisionsvermeidung und Hinderniserkennung auf Basis dieser Sensoren vorgestellt.

2.2 Sensoren zur Hinderniserkennung

Allgemein existieren drei Messverfahren anhand derer Abstandssensoren die Distanz zu einem Objekt bestimmen können: Die Signallaufzeitmessung, das Triangulationsverfahren und die Phasendifferenzmessung, welche im folgenden Abschnitt 2.2.1 erklärt werden sollen. Des Weiteren sollen in Abschnitt 2.2.2 die Messeigenschaften aktiver Abstandssensoren spezifiziert werden und ferner in Abschnitt 2.2.3 aktuelle Sensoren vorgestellt werden.

2.2.1 Messverfahren

Grundsätzlich bestehen Abstandssensoren aus zwei Elementen: Einem aktiven emittierenden Sender und einem detektierenden Empfänger, welcher das vom Sender emittierte, an einem Objekt reflektierte Signal empfängt. Aus diesem Grund werden Abstandssensoren auch als aktive Sensoren bezeichnet.

Beim **Triangulationsverfahren** wird meist ein kontinuierliches Lichtsignal ausgesandt und dabei mittels eines PSD- (Position Sensitive Detector) oder CCD- (Charde Coupled Device) Elements

der Einfallswinkel des reflektierten Lichtstrahls gemessen. Sender und Empfänger besitzen konstruktiv bedingt einen fixen, bekannten Abstand b zueinander und bilden so zusammen mit dem reflektierten Lichtstrahl ein rechtwinkliges Dreieck (vgl. Abb.: 2.1). Über den gemessenen Einfallswinkel θ, bzw. den Abstand g zum Mittelpunkt des Sensors wird das Dreieck voll bestimmt und es kann die Entfernung d zum Objekt über folgende Formel

$$d = b \cdot \tan(\theta) = b \cdot \frac{f}{g} \tag{2.1}$$

bestimmt werden.

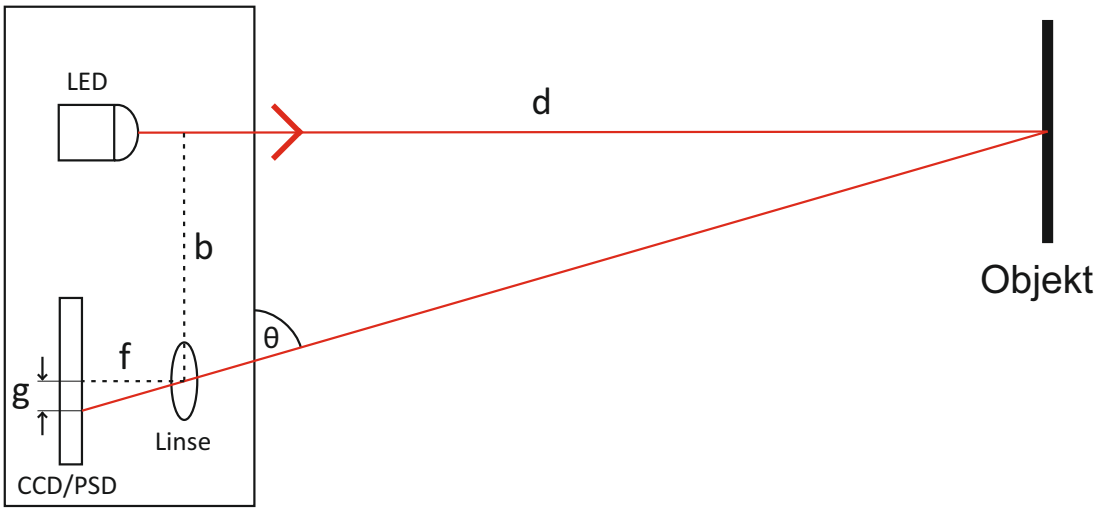


Abbildung 2.1: Prinzip des Triangulationsverfahrens, Quelle: Nach Berkovic und Shafir [4] Abb.: 8 S.449

Bei der **Signallaufzeitmessung** werden elektromagnetische Wellen (Radar), Schallwellen (Sonar) oder Lichtimpulse (Lidar) pulsartig ausgesandt und dabei die Laufzeit der Welle vom Zeitpunkt der Aussendung t_0 bis zum Empfang t_1 gemessen. Die Distanz zwischen Sender und Objekt kann über die Formel

$$d = \frac{c \cdot \Delta t}{2 \cdot n} \tag{2.2}$$

ermittelt werden. Hierbei entspricht c für elektromagnetische Wellen der Lichtgeschwindigkeit bzw. für Schallwellen der Schallgeschwindigkeit, $\Delta t = t_1 - t_0$ der Laufzeit des Signals und n dem Brechungsindex des Mediums, welcher für elektromagnetische Wellen in Luft etwa 1,00011 entspricht.

Bei der **Phasendifferenzmessung** wird eine kontinuierliche elektromagnetische Welle mit Wellenlänge λ über einen Strahlenteiler aufgesplittet, wobei der aufgesplittete Referenzstrahl direkt von einem Phasenmessgerät in konstanter Entfernung l gemessen wird und der Hauptstrahl zunächst ausgesandt, reflektiert und darauf vom Phasenmessgerät erfasst wird. Die vom reflektierten Strahl zurückgelegte Strecke lässt sich über die Formel

$$d = l + \frac{\theta}{2\pi} \cdot \frac{\lambda}{2} \tag{2.3}$$

berechnen, wobei d der gemessene Abstand und θ der Phasenunterschied zwischen Haupt- und Referenzstrahl ist. Da die Wellenlänge des sichtbaren Lichts ($\approx$ 380 -780 nm) klein im Verhätlnis zu den gemessenen ABständen ist und bereits ab einem Phasenunterschied von $\theta > 360°$ der Abstand nicht mehr eindeutig bestimmbar ist, müssen beide Strahlen auf eine niedrigere Frequenz mit größerer Wellenlänge moduliert werden, um größere Distanzen messen zu können.

Allgemein gewährleisten alle drei Methoden eine Abstandsmessung unabhängig von der Intensität des reflektierten Signals. Es ist jedoch ist zu beachten, dass Material, Farbe und Oberflächenbeschaffenheit des Objekts zu ungewollten Reflexionen, Absorption oder zu Streuung führen kann, so dass eine Detektion aufgrund zu niedriger Intensität des reflektierten Signals nicht mehr möglich ist.

2.2.2 Messeigenschaften

Allgemein lassen sich aktive Abstandssensoren durch folgenden Eigenschaften beschreiben:

Reichweite: Spezifiziert die minimale und maximale Reichweite des Sensors.

Genauigkeit: Beschreibt die Messabweichung einer einzelnen Abstandsmessung in Abhängigkeit der Entfernung zum gemessenen Objekt.

Abtastzeit: Benötigte Zeit von zwei aufeinanderfolgende Messungen.

Signalquelle: Definiert Wellenlänge, Leistung sowie den Typ des verwendeten Emitters.

Field of View/Öffnungswinkel: Spezifiziert den horizontal und vertikal sichtbaren Erfassungsbereich (bis zu $360°$).

Winkelauflösung: Entspricht dem Winkelabstand zwischen zwei aufeinander folgenden Abstandsmessungen und ist somit abhängig von der maximalen Anzahl an Abstandsmessungen pro Umdrehung (gilt nur für Laserscanner).

2.2.3 Übersicht aktueller Sensortechnik

Der folgende Abschnitt befasst sich mit den gängigsten aktiven Abstandssensoren, welche heutzutage genutzt werden. Dabei soll allgemein auf die Vor- und Nachteile der jeweiligen Sensoren sowie auf die Einsatzfähigkeit im Bezug zur Hinderniserkennung eingegangen werden. Zusätzlich sollen Einsatzgebiete und Einsatzmöglichkeiten dieser Sensoren für kleinere UAVs dargelegt werden.

Infrarot-Abstandssensoren

Bei Infrarot-Abstandssensoren besteht der Sender aus einer IRED (Infrared Emitting Diode), welche kontinuierlich Licht im Infrarotbereich mit einer Wellenlänge von 880-950 nm emittiert. Die Abstandsmessung beruht hierbei auf dem Triangulationsverfahren.
Da die Bestandteile eines Infrarot-Sensors meist sehr kostengünstig, kompakt sind sowie ein geringes Gewicht (< 10 g) aufweisen, können diese gut in kleineren UAVs eingesetzt werden. Je nach Ausführung haben Infrarot-Sensoren eine eher geringe Reichweite von 10-150 cm, jedoch eine hohe Abtastfrequenz von 30-50 Hz, wie zum Beispiel die Infrarot-Sensoren von Sharp [41]. Außerdem haben Infrarot-Abstandssensoren aufgrund ihres Öffnungswinkels keine besonders hohe Richtgenauigkeit, womit dennoch die ungefähre Position eines Objekts bestimmt werden kann. Somit eignen sie sich zur Hinderniserkennung, jedoch schlecht zur Umgebungserfassung und Kartierung der Umgebung.

Laser-Abstandssensoren

Bei Laser-Abstandssensoren besteht der Sender aus einer Laserdiode, welche Laserlicht mit einer fixen Wellenlänge emittiert. Grundlegend kann jedes der Messverfahren aus Abschnitt 2.2.1 zur Abstandsmessung genutzt werden. Die Genauigkeit, sowie die Reichweite, variiert je nach verwendetem Verfahren. Allgemein haben Laser aufgrund ihrer Intensität eine deutlich höhere Reichweite als z.B. Infrarot-Abstandssensoren. Selbst auf kleineren UAVs können Laserabstandssensoren mit einer Reichweite von ca. 40 m [31], z.B. zur Höhenmessung genutzt werden.

Aufgrund ihres sehr kleinen Öffnungswinkels eignen sich einzelne oder selbst mehrere ringförmig angeordnete Sensoren eher schlecht zur Hinderniserkennung und Umgebungserfassung, da der abdeckbare Bereich sehr klein ist.

Laserscanner

Die Abstandsmessung erfolgt bei Laserscannern oft über die Signallaufzeitmessung. Alternativ werden das Triangulationsverfahren oder auch Phasendifferenzmessungen eingesetzt, welche noch höhere Genauigkeiten ermöglichen.
Allgemein besteht bei einem Laserscanner der Sender aus einer Laserdiode, welcher je nach Anwendungsfall ein pulsartiges oder kontinuierliches Lasersignal mit einer Wellenlänge von 600-1000 nm emittiert. Grundprinzip hierbei ist eine zweidimensionale Umgebungserfassung mit einer Abdeckung von bis zu $360°$ durch Rotation des Sender-Empfänger-Systems. Bei der Signallaufzeitmessung wird der Laserstrahl, meistens durch einen Spiegel, so umgelenkt, dass dieser in einer Ebene rotiert (vgl. Abb.: 2.2). Das reflektierte Lasersignal wird mit Hilfe des Spiegels zurück auf den Empfänger geworfen. Da die Ausbreitungsgeschwindigkeit des Lichts im Verhältnis zur Rotationsgeschwindigkeit des Spiegels deutlich höher ist, kann eine Messung erfolgen, bevor der Spiegel zu weit rotiert ist [13].

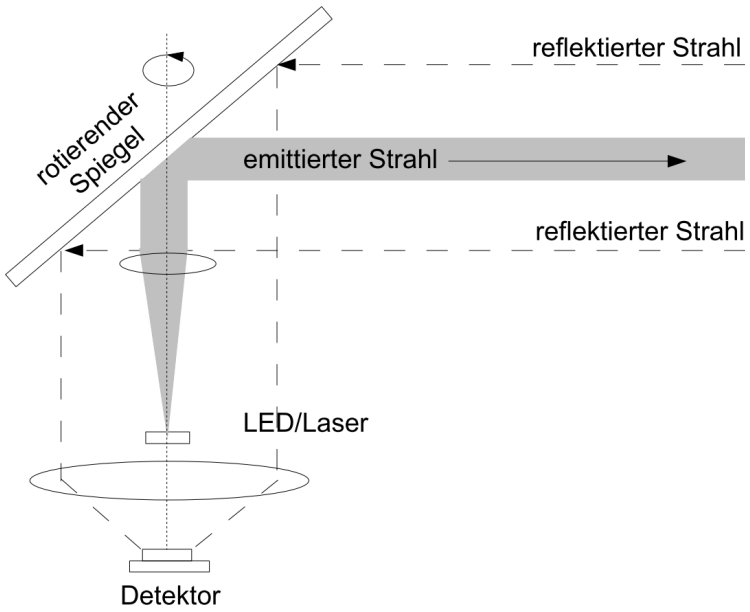


Abbildung 2.2: Funktionsweise eines Laserscanners, Quelle: Doert [13] Abb.: 2.4 S. 9

Bei Laserscannern, die anhand des Triangulationsverfahren funktionieren, werden Laserdiode und Empfänger zusammen rotiert, da ein fixer Abstand zwischen beiden erhalten bleiben muss (vgl. Abb.: 2.1). Auch 3D-Laserscanner-Systeme werden bereits eingesetzt, bei denen z.B. einezusätzliche Kippbewegung des Aufbaus für eine dreidimensionale Abbildung der Umgebung sorgt [11].

Der Anwendungsbereich heutiger Laserscanner ist sehr vielfältig. Im Bereich autonomer Fahrzeuge, in der terrestrischen Messtechnik, oder in Airborne-Laserscanning-Systemen werden sehr leistungsstarke Laserscanner-Systeme mit hohen Reichweiten eingesetzt. In der Robotik und im Bereich kleinerer UAVs, finden sich, aufgrund von benötigter Mobilität und geringerer Tragfähigkeit, oft schwächere, aber dafür auch günstigere und leichtere Systeme. Gerade Abmessungen und Gewicht sind besonders für UAVs von Bedeutung.
Hochwertige Laserscanner-Systeme für kleinere UAVs haben ein Gewicht von ca. 250-400 g, eine Reichweite von ca. 30 m, eine Genauigkeit von $\pm$30-50 mm, eine Winkelauflösung von $\approx 0{,}25\,^{\circ}$ und eine Abtastzeit von bis zu 25 ms, wie zum Beispiel der Hokuyo UTM-30LN [22].

Die Vorteile von Laserscannern gegenüber anderen Abstandssensoren (wie z.B. Infrarot) liegen prinzipiell bei der Dichte und Anzahl der Messwerte, sowie der zweidimensionalen Abtastung durch ein Sichtfeld von bis zu $360°$. Daher eignen sich Laserscanner besonders gut zur Kartierung und, aufgrund ihrer Genauigkeit, bevorzugt mit SLAM-Algorithmen verwendet.
Ein bedeutender Nachteil (besonders für kleinere UAVs) ist das relativ hohe Gewicht. In den letzten Jahren wurden diesbezüglich Fortschritte erzielt, da explizit Laserscanner für kleinere und leichte Roboterplattformen entwickelt worden sind.
Allgemein sind Laserscanner verhältnismäßig teurer als einfache Abstandssensoren, wie Infrarot, Laser oder Ultraschall. Des Weiteren erfordert die große Menge an Messdaten oft deutlich leistungsstärkere Systeme zur Signalverarbeitung und rechenintensive Algorithmen zur Datenauswertung.

Akustische Abstandssensoren

Akustische Abstandssensoren arbeiten im Ultraschallbereich. Grundprinzip ist hierbei die Signallaufzeitmessung von pulsartig ausgesandten akustischen Signalen. Grundsätzlich breiten sich

Schallwellen kugelförmig aus, so dass akustische Abstandssensoren selbst mit gerichteten Schallwellen einen großen Messbereich abdecken. Aus diesem Grund eignen sich diese Sensoren gut zur Kollisionsvermeidung, wobei kleinere Objekte im Messbereich nicht zuverlässig detektiert werden können. Ein weiterer Nachteil von akustischen Sensoren ist die relativ geringe Reichweite von ca. 6 m [35], da die Ausbreitung von Schallwellen durch Luftabsorption und Streuung gedämpft wird und zusätzlich noch temperaturabhängig ist. Zudem kann es bei der Verwendung von mehreren akustischen Sensoren zu Messungenauigkeiten kommen, wenn das reflektierte Signal fälschlicherweise von einem anderen akustischen Sensor gemessen wird.
Allgemein eignen sich akustische Abstandssensoren sehr gut zum Einsatz auf kleineren UAVs, da sie aufgrund ihrer Bauweise relativ kompakt und leicht sind [35].

3D-Kamerasysteme

3D-Kamerasysteme, auch Time-Of-Flight Kameras genannt, besitzen einen so genannten Photonic Mixing Device (PMD) Sensor mit einer Auflösung von bis zu 204x204 Pixel, welcher nach dem Prinzip der Signallaufzeitmessung funktioniert. Als Emitter wird eine LED verwendet, welche im GHz-Bereich moduliertes Licht aussendet. Der PMD-Sensor ermöglicht es, für jedes Pixel einen Abstand zu messen, so dass echte 3D-Tiefenbilder der Umgebung mit Bildraten von bis zu 160 Bildern pro Sekunde [30] erstellt werden können. Aus diesem Grund eignen sich Time-Of-Flight Kameras sehr gut zur Umgebungserfassung sowie zur Hinderniserkennung. Sie können auch in 3D-Mapping-Algorithmen integriert werden.
Eine Erweiterung von PMD-Kameras sind sogenannte RGB-D Kameras, welche zusätzlich zu den Entfernungsmessungen ein farbechtes Bild liefern.

2.3 Bestehende Konzepte und Projekte

2.3.1 Simultane Lokalisierung und Kartenerstellung

Ein weit verbreiteter Ansatz zur Weiterverarbeitung von prinzipiell allen umgebungserfassenden Sensoren (insbesondere Laserscannern) ist das sogenannte Simultaneous Localization and Mapping (SLAM). Gerade für UAVs erlaubt dies die Wahrnehmung einer a priori unbekannten Umgebung, ihrer Bewegung und Position. Alternativ bestehen für UAVs nur wenige Möglichkeiten ihre Position in geschlossenen Räumen zu bestimmen, wie das Integrieren der Beschleunigungsmes-

sungen eines Inertialsensors (stark fehlerbehaftet), die Verwendung eines optischen Fluss-Sensors [18] oder eines externen optischen Tracking Systems [28]. In den letzten Jahren wurde eine Vielzahl von Konzepten entwickelt, welche auf unterschiedliche Weise das SLAM-Problem lösen: In Echtzeit eine Karte einer unbekannten Umgebung zu erstellen und sich in dieser zu lokalisieren. Ausgehend von dieser Karte können Trajektorien geplant werden um Kollisionen zu vermeiden oder um Hindenrisse zu umfliegen.

Das Grundprinzip ist bei allen SLAM-Konzepten ähnlich: Es wird eine Karte der Umgebung erstellt und diese durch Abtasten der Umgebung mit Hilfe von Abstandssensoren aktualisiert und erweitert. Da ausschließlich neue und gültige Umgebungsinformationen in die Karte eingefügt werden können, falls Position und Ausrichtung des UAVs oder Roboters innerhalb dieser Karte bekannt ist, muss in jedem Schritt eine Lokalisierung innerhalb dieser Karte erfolgen. Hierzu existieren bereits verschiedene erprobte Ansätze, von denen exemplarisch das Iterative Closest Point Verfahren und der Monte-Carlo Partikel Filter im Folgenden erklärt werden sollen.

Iterative Closet Point

Der Iterative Closest Point (ICP) Algorithmus, der auf McKay und Besl [25] zurück geht, wurde ursprünglich dazu konzipiert, die Translation t und Rotation R zweier korrespondierender Punktwolken $P = p_1, ..., p_n$ und $P^* = p_1^*, ..., p_n^*$ zueinander zu bestimmen, wobei $P, P^* \in \mathbb{R}^3$. Hierzu werden t und R iterativ so angepasst, dass die Summe des quadratischen Fehlers der Funktion

$$E(R,t) = \frac{1}{n} \sum_{i=1}^{n} ||p_i - Rp_i^* - t||^2 \tag{2.4}$$

minimiert wird. Dazu wird im ersten Schritt zu jedem Punkt in P der nächste Nachbar aus P^* bestimmt. Jedes dieser Paare wird nun gewichtet und ggf. sogar verworfen. Zu den übrig gebliebenen Paaren wird die benötige Translation und Rotation errechnet. Für alle gefunden Translations- und Rotationswerte wird nun die quadratische Fehlerfunktion (vgl. 2.4) ausgerechnet und die Translations- und Rotationswerte mit dem kleinsten quadratischen Fehler ausgewählt. Die Punktwolke P^* wird dann mit den gefundenen Translations- und Rotationswerten transformiert. Diese Schritte werden so lange ausgeführt, bis E einen bestimmten Grenzwert unterschritten hat, oder eine vorher festgelegte maximale Anzahl an Iterationen erreicht wurde.

Der Algorithmus eignet sich gut für die Verarbeitung von Laserscannerdaten, da die Abstands-

messungen als Punkte eines 2- oder 3-dimensionalen Koordinatensystem verstanden werden können. Somit kann aus zwei aufeinanderfolgenden Laserscanmessungen die relative Bewegung (Translation) und Ausrichtungsänderung (Rotation) bestimmt werden. Je nach Anzahl der verwendeten Messpunkte steigt die Laufzeit des Algorithmus jedoch quadratisch mit der Anzahl der verwendeten Messpunkte an, da besonders bei der Suche des nächsten Nachbarn alle Punktepaare miteinander verglichen werden. Allerdings existieren Erweiterung des Algorithmus, die diese Problematik zum Teil umgehen und den Algorithmus optimieren und effizienter machen. Wie z.B. der Trimmed Iterative Closest Point Algorithmus der über einen Schätzwert nur die wahrscheinlich nächsten Punkte betrachtet [10].
Ein weiteres Problem dieses Algorithmus ist jedoch das so genannte Closed Loop Problem, das auftritt, falls sich das UAV schleifenförmig bewegt und nicht wieder zurück zur ursprünglichen Anfangsposition findet. Da die gemessenen Translation- und Rotationswerte nach jeder Iteration nicht exakt mit der Realität übereinstimmen, wird immer ein kleiner Fehler integriert, so dass nach einer gewissen Zeit eine deutliche Abweichung zu bemerken ist. Diese muss anderweitig, zum Beispiel durch einen zusätzlichen Sensor und Algorithmus, korrigiert werden muss.
Um Translation und Rotation kontinuierlich bestimmen zu können, benötigt der Algorithmus genügend Messpunkte in ausreichend kurzen Zeitabständen.

Monte-Carlo Partikel Filter

Um mit Störungen aus zufälligen Messfehlern und unvermeidlichem physikalischem Messrauschen umgehen zu können, eignet sich die sequenzielle Monte-Carlo Methode, welche als Verfahren der stochastischen Zuständsschätzer klassifiziert werden kann.
Als Zustand wird die Position des UAVs betrachtet. Da die Position nur indirekt aus den Abstandsmessungen bestimmt werden kann, liefert ein Partikel-Filter letztendlich eine Verteilung von geschätzten wahrscheinlichen Aufenthaltsorten. Hierzu wird eine Vielzahl von so genannten Partikeln erzeugt, von denen jedes einen möglichen Aufenthaltsort innerhalb der Karte mit einer zusätzlich gespeicherten Aufenthaltswahrscheinlichkeit repräsentiert. Für alle Partikel wird überprüft wie gut die aktuellen Abstandsmessungen mit dem geschätzten Aufenthaltsort des Partikels korrelieren und anhand dieser Information die Aufenthaltswahrscheinlichkeit angepasst. Unterschreitet die Aufenthaltswahrscheinlichkeit einen bestimmten Grenzwert, so wird dieses Partikel verworfen, so dass nur wahrscheinliche Aufenthaltsorte weiter betrachtet werden. Häufig

wird eine Mindestanzahl an Partikeln verwendet, so dass verworfene Partikel ersetzt werden. Die ersetzten Partikeln werden je nach Ansatz in der Nähe des wahrscheinlichen Aufenthaltsortes gestreut, oder zufällig in der Karte verteilt.
Im besten Fall sollten alle Partikel zu einer Position konvergieren, so dass aus dieser Punktwolke die tatsächliche Position abgeleitet werden kann.
Der Monte-Carlo Partikel Filter ist multi-modal, was bedeutet, dass die Verteilung der Aufenthaltsorte mehrere Maxima, also mehrere Punktwolken an verschiedenen Orten innerhalb der Karte, haben kann. Dies kann im schlechtesten Fall zu einem sprunghaften Verhalten der Position führen, besonders für symmetrische Umgebungen. Zusätzlich benötigen die Berechnungen der Aufenthaltswahrscheinlichkeiten der einzelnen Partikel viel Rechenaufwand, sind jedoch parallelisierbar.
Partikel-Filter ermöglichen eine driftfreie Lokalisierung innerhalb einer bereits bestehenden oder simultan erstellten Karte, was ein deutlicher Vorteil gegenüber anderen Verfahren wie dem ICP-Algorithmus ist.

Verwandte Arbeiten und Projekte

Konzeptuell und auf theoretischer Ebene existieren nach dem heutigen Stand diverse SLAM-Algorithmen, die sich je nach Einsatzgebiet in der wissenschaftlichen Anwendung bereits bewährt haben. In der Realität erschweren besonders verrauschte Sensorwerte und limitierte Prozessorleistung, gerade auf voll autonomen UAVs, die Umsetzung und Implementierung dieser Algorithmen, so dass eine Anpassung der SLAM-Algorithmen an die realen Bedingungen und des verwendeten UAVs notwendig ist. Dahingehend hat sich besonders im letzten Jahrzehnt eine Vielzahl an wissenschaftlichen Arbeiten und Projekten mit eben dieser Umsetzung befasst.
Die Arbeit von Achtelik et al. [1] ermöglicht unter der Verwendung des GMapping Algorithmus [19], eines Hokuyo Laserscanners und eines stereooptischen Kamerasystems die Erstellung von 2D-Karten und darin simultaner Lokalisierung.
Grzonka et al. [20] erweiterte das 2D-SLAM zu einem Multi-Level SLAM, welches aus mehreren vertikal übereinander gestapelten 2D-Karten besteht. Zur Lokalisierung innerhalb einer 2D-Karten wurde der Monte-Carlo Localization Algorithmus Partikel Filter [16] verwendet. Dies erlaubt eine 3D-Erfassung und Kartierung der Umgebung, sowie Navigation und Kollisionsvermeidung.

Blösch et al. [7] entwickelte einen Algorithmus, welcher optischen 3D-SLAM ermöglicht. Hierzu mussten die Bilddaten der auf dem Quadrokopter montierten Kamera jedoch auf einer externen Bodenstation ausgewertet werden.
Shen et al. [42], als auch Winkvist [43] benutzten einen Laserscanner sowie den ICP-Algorithmus zur Lokalisierung und ermöglichten dadurch ein 3D-SLAM. Der ICP-Algorithmus wurde von Shen zusätzlich mit einer Kamera erweitert, um das Closed Loop Problem zu lösen.
Eine heutzutage weitverbreite Möglichkeit um SLAM Algorithmen zu implementieren, bietet das sogenannte Robot Operating System [33]. ROS, welches auf verschiedenen Betriebssystemen und Hardware als Middleware agiert, liefert als Plattform verschiedene so genannte Packages, welche voll funktionsfähige SLAM-Algorithmen bereitstellen, die für unterschiedliche umgebungserfassende Sensoren ausgelegt sind.
Ein Beispiel hierfür ist der von Kohlbrecher et al. [24] entwickelte 3D-SLAM Algorithmus, unter dem Namen hector_slam [26], welcher auf Laserscannerdaten basiert, oder das rgdbslam-package [15], welches die Daten eines RGB-D Sensors auswertet.
Allgemein ermöglichen SLAM-Algorithmen ausreichend genaue Karten der Umgebung zu erstellen und sich in diesen ohne Positionsdrift zu lokalisieren. Diese Algorithmen sind jedoch sehr komplex und aufwendig, so dass zum Teil zusätzliche Ressourcen, wie zum Beispiel eine externe Bodenstation, benötigt werden, um die benötigte Rechenleistung zur Verfügung zu stellen. Alternativ kann auf Kosten der Genauigkeit die benötigte Rechenleistung verringert werden.
Die Fortschritte in den letzten Jahren haben jedoch gezeigt, dass funktionsfähige SLAM-Algorithmen auf UAVs implementiert werden können und sich diese zur Hinderniserkennung und Kollisionsvermeidung nutzen lassen.

2.3.2 Kartenunabhängige Hinderniserkennung und Kollisionsvermeidung

Es nicht zwangsläufig notwendig, SLAM-Algorithmen zur Umgebungserfassung zu nutzen. Es können selbst anhand von Abstandssensoren wie Infrarot, Ultraschall oder Laser, zuverlässig Hindernisse erkannt und Kollisionen vermieden werden. Diese Vorgehensweise ist deutlich ressourcenschonender als SLAM-Algorithmen und eignet sich darüber hinaus zum Einsatz auf Mikrocontrollern mit begrenzter Rechenleistung [17]. Zudem bringt die Verwendung einer Karte wei-

tere Probleme mit sich, wie zum Beispiel die Aktualisierung bei sich bewegenden Objekten, was für eine effektive Kollisionsvermeidung berücksichtigt werden muss.

Allgemein ist, mit der steigenden Anzahl von UAVs in privatem Gebrauch, eine Kollisionsvermeidung für UAVs aus Sicherheitsgründen zunehmend relevant, um Abstürze zu vermeiden. Dahingehend entwickelten selbst namhafte UAV-Hersteller wie DJI [12] unlängst Systeme zur Kollisionsvermeidung, welche in diesem Fall auf Ultraschallsensoren und einem stereooptischen Kamerasystem basieren.

In der Forschung wurden 2007 erste Systeme von Roberts et al. [36] entwickelt, welche Infrarotsensoren einsetzten. Becker et al. [3], sowie Bouabdallah [8] [9] benutzten vier Ultraschallsensoren zur Umgebungserfassung und entwickelten ein System zur Kollisionsvermeidung. Park und Kim [29] verwendeten Stereokameras zur Hinderniserkennung und Alvarez et al. [2] entwickelten ein System, welches lediglich mit einer Kamera eine Kollisionsvermeidung ermöglicht. Hierzu wurde die Bewegungsparallaxe berücksichtigt, die Datenverarbeitung musste jedoch auf externer Hardware verarbeitet werden.

Viele Systeme, die LiDAR basierte Laserscanner zur Hinderniserkennung nutzen wurden jedoch entweder nicht für kleinere UAVs konzipiert (s. Sabatini et al. [40]) oder bedürfen zusätzlicher externer Hardware [23] und können somit nicht auf einem autonomen UAV eingesetzt werden.

3 Konzept

3.1 Überblick

Das folgende Kapitel befasst sich mit der Signal- und Datenverarbeitung von 2D-Laserscannern zur zweidimensionalen Hinderniserkennung, welche in einem weiteren Schritt zur Kollisionsvermeidung genutzt werden sollen. Zur Kollisionsvermeidung wird auf den von Gageik et al. [17] am Lehrstuhl für Informatik VIII der Universität Würzburg entwickelten Algorithmus zurückgegriffen. Das hier vorgestellten Konzept bietet somit Lösungen zu dem Problem Laserscanner basierter Hinderniserkennung.
In Abschnitt 3.2 soll zunächst die Problematik der Hinderniserkennung erläutert und die Anforderungen an das Konzept veranschaulicht werden. Im Folgenden wird in Abschnitt 3.4 der Histogram Filter als Konzept zur Signalverarbeitung von Laserscannerdaten vorgestellt. Der Filter beruht dabei auf einer bereits vorhandenen Lösung [Gageik] und erweitert das Konzept um verschiedene Variationen sowie Vor- und Nachbearbeitungsschritte. In Abschnitt 3.6 wird eine Methode zur dynamischen Objekterkennung erklärt, sowie die Verwendung des rekursiven Bayes Filters in Abschnitt 3.7 erläutert.

Das Diagramm in Abb. 3.1 gibt eine Übersicht über die Integration der Signalverarbeitung in das bestehende System. Die Signalverarbeitung besteht im allgemeinen aus drei Teilen: Vorverarbeitung, Hinderniserkennung und Nachbearbeitung. Bei der Vorverarbeitung werden die Sensordaten des Laserscanners und des Inertialsensors benötigt, um die Messdaten des Laserscanners orthogonal projizieren zu können (vgl. Abschnitt 3.3). Zusätzlich werden die Sensordaten für den Histogramm Filter in Sektoren aufgeteilt (vgl. Abschnitt 3.4). Der Histogramm Filter ist das wesentliche Grundkonzept der hier verwendeten Hinderniserkennung und folgt, je nach Variation, im darauf folgenden Prozess. Die gefilterten Werte werden in einem letzten Schritt nochmals

nachbearbeitet, bevor sie fusioniert werden und in die Regelung zur Kollisionsvermeidung eingehen.

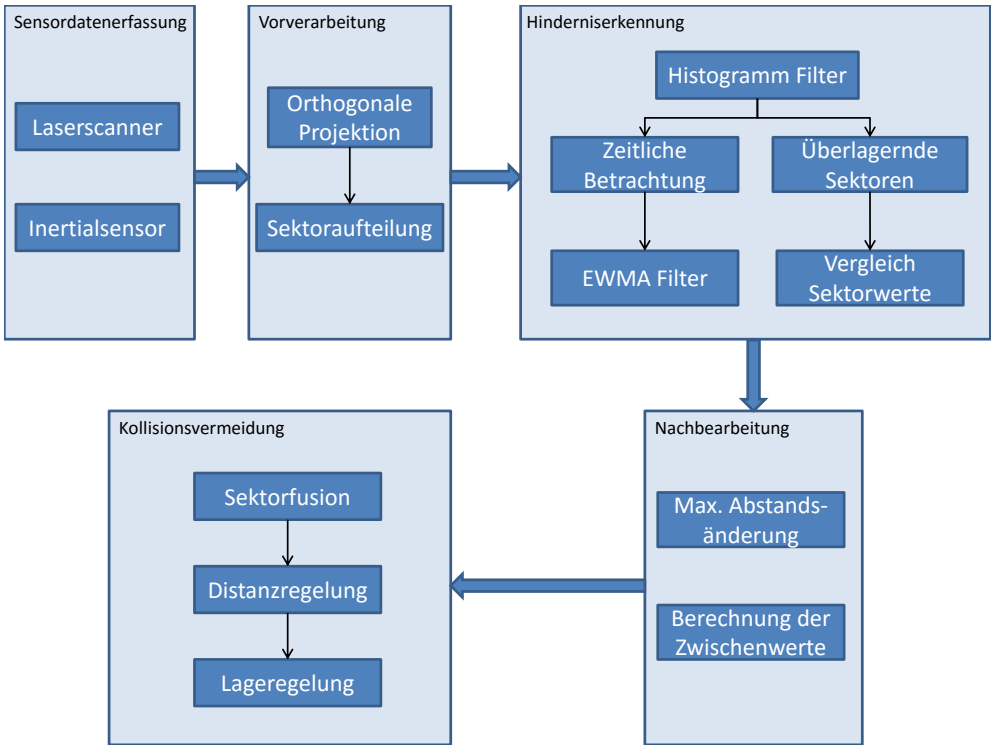


Abbildung 3.1: Integration der Signalverarbeitung des Laserscanners in das bestehende System des AQopterI8

3.2 Problematik und Zielsetzung

Allgemein sind vier Abstandswerte (Vorne, Hinten, Links, Rechts) für eine zweidimensionale Kollisionsvermeidung ausreichend, da alle Hindernisse über diese Abstände angegeben werden können [17]. Allgemein sind pro Richtung nur die Hindernisse mit geringstem Abstand für eine Regelung interessant, da nur diese als Gefahrenquelle für mögliche Kollisionen in Frage kommen. Es hat sich gezeigt, dass es nicht möglich ist nur die Abstandsmessung mit geringstem Abstand zu verwenden [17]. Der Grund dafür ist das Rauschen der Messdaten, sowie zufällige Messfehler. Außerdem treten bei einem Laserscanner messtechnisch bedingt die Messpunkte von zwei aufeinander folgenden Scans nicht exakt an den gleichen Stellen auf und bilden somit nicht die gleiche Stelle aus dem vorherigen Scan ab. Diese Winkelabweichung verursacht ein zusätzlich Rauschen, welches berücksichtigt werden muss.

Algorithmen zur Datenverarbeitung, welche möglichst alle Laserscannerdaten in linearer Lauf-

zeit zur weiteren Verarbeitung filtern, robust gegenüber Störeinflüssen sind und in Echtzeit ohne Verzögerung agieren, sind notwendig für eine effektive Kollisionsvermeidung und gelten somit als Anforderung an das hier vorgestellte Konzept.

3.3 Vorverarbeitung

Um die Abstandsmessungen des Laserscanners korrekt auswerten zu können, muss die die Lage des UAVs berücksichtigt werden. Die Messebene des Laserscanners ist zwangsläufig abhängig von der Lage des UAVs, welches durch ein Quaternion der Form

$$q = q_0 + iq_1 + jq_2 + kq_3,\ mit\ i^2 = j^2 = k^2 = ijk = -1 \tag{3.1}$$

dargestellt werden kann.
Jede Abstandsmessung des Laserscanners muss nun anhand des Lagequaternions q rotiert und orthogonal auf die Ursprungsebene projiziert werden. Hierzu interpretiert man zunächst alle Abstandsmessungen als Punkte P in Polarkoordinaten mit Abstand d und Winkel φ (vgl. Abb.: 3.2).

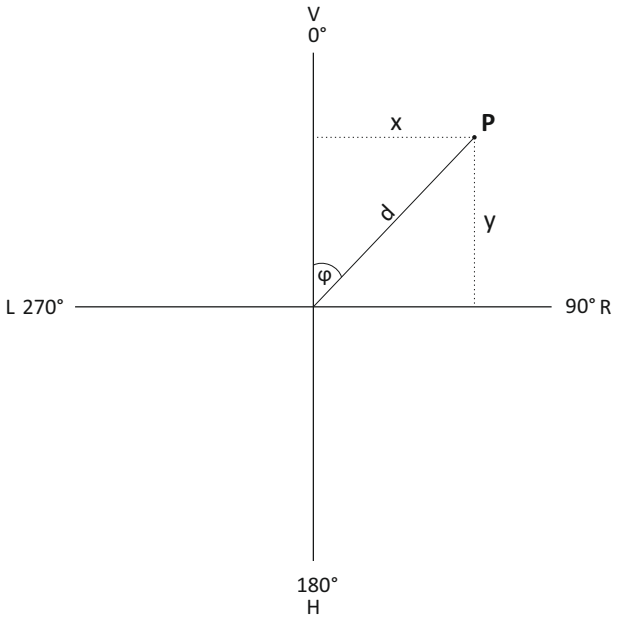


Abbildung 3.2: Orientierung und Darstellung der Messebene des Laserscanners als Polarkoordinatensystem

Im nächsten Schritt werden diese Punkte in kartesische Koordinaten umgewandelt und es ergibt sich ein Vektor v_{xyz} der Form:

$$v_{xyz} = \begin{bmatrix} d \cdot \sin\varphi \\ d \cdot \cos\varphi \\ 0 \end{bmatrix} \tag{3.2}$$

Der Vektor v_{xyz} wird nun anhand des Lagequaternions q rotiert. Die Rotation eines Vektor um ein Quaternion ergibt sich durch Gleichung 3.3, falls q einem Einheitsquaternion entspricht:

$$v_q = q \cdot \begin{bmatrix} 0 \\ v_x \\ v_y \\ v_z \end{bmatrix} \cdot q^* \tag{3.3}$$

q^* stellt hierbei die Konjugation des Quaternion q da und v_q als Quaternion den rotierten Vektor. Der rotierte Vektor entspricht nun:

$$v_{rot} = \begin{bmatrix} v_{q_1} \\ v_{q_2} \\ v_{q_3} \end{bmatrix} \tag{3.4}$$

Die Projektion des Vektors v_{rot} auf die Ursprungsebene ist nun v_{rot} mit $v_{rot_z} = 0$.
Der rotierte und projizierte Vektor wird anschließend zurück in Polarkoordinaten transformiert, wobei gilt:

$$d = \sqrt{v_{rot_x}^2 + v_{rot_y}^2} \tag{3.5}$$

$$\varphi = atan2(v_{rot_x}, v_{rot_y}) \tag{3.6}$$

Je nach Anforderung an die Genauigkeit ist die orthogonale Projektion der Abstandswerte nicht notwendig, da für kleine Auslenkungen aus dem Equilibrium die Abstandsmessung ungefähr der orthogonalen Projektion entspricht. Bei 2D-SLAM-Algorithmen wird diese Genauigkeit allerdings benötigt. In extremen Situation mit starker Lageänderung des UAVs muss die Projektion ebenfalls berücksichtigt werden.

3.4 Histogramm Filter

Der Histogramm Filter ist eine effektive Methode, um eine Vielzahl an Messdaten weiterzuverarbeiten. Der Messbereich des Laserscanners wird hierzu zunächst in Sektoren aufgeteilt (beispielhaft dargestellt in Abb.: 3.3), welche wiederum in Felder gleichen Abstands mit Feldbreite b aufgeteilt werden. Für jeden Sektor wird separat ein eigener Histogramm Filter genutzt. Jede Abstandsmessung kann somit je nach Winkel einem der Sektoren eindeutig zugeordnet werden.

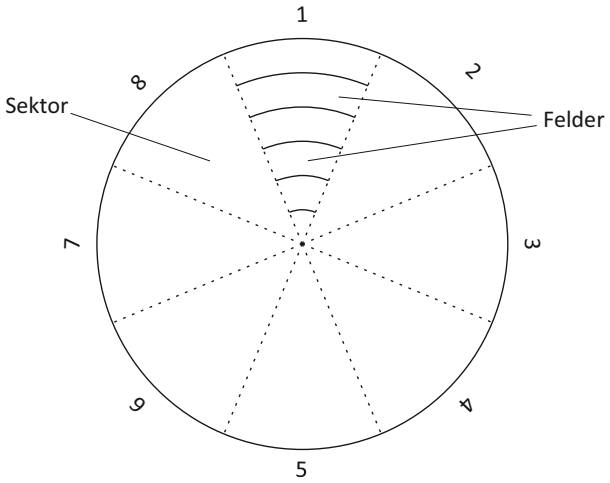


Abbildung 3.3: Aufteilung der Umgebung in Sektoren

Die Funktionsweise des Histogramm Filters ist beispielhaft in Abb.: 3.4 für einen einzelnen Sektor dargestellt. Jede Abstandsmessung des Laserscanners fällt in eines der Felder des Histogramm Filters. Vor dem Einfügen wird der Abstand der Messung auf die Mitte des jeweiligen Sektors projiziert:

$$d_{proj} = d_i \cdot cos(\frac{n_{sektor} \cdot 2\pi}{N} - \varphi_i) \tag{3.7}$$

Wobei d_i der gemessene Abstandswert, n_{sektor} der Index des Sektors, N die Anzahl an Sektoren und φ_i der Winkel der Abstandsmessung in Polarkoordinaten ist (vgl. Abb.: 3.2). Beim Einfügen der Messwerte wird über alle bereits in das jeweilige Feld einsortierten Messpunkte der Mittelwert gebildet und die Anzahl der vorkommenden Messpunkte gespeichert, bzw. erhöht. Wurden alle Messwerte eingefügt, wird das Feld des Histogramm Filters mit dem geringsten Abstand ausgewählt, welches eine Mindestanzahl von Messpunkten über einem festgelegten Schwellwert aufweist (Schwellwert $= 4$ in Abb. 3.4) und aus diesem Feld der bereits errechnete Mittelwert als Abstandswert für diesen Sektor übernommen. Alle anderen Messwerte werden verworfen. Falls in keinem Feld ausreichend Messpunkte vorhanden sind, wird angenommen, dass sich kein

Objekt oder Hindernis in diesem Sektor befindet und es wird die maximale Messreichweite des Laserscanners D für den Sektor angenommen.

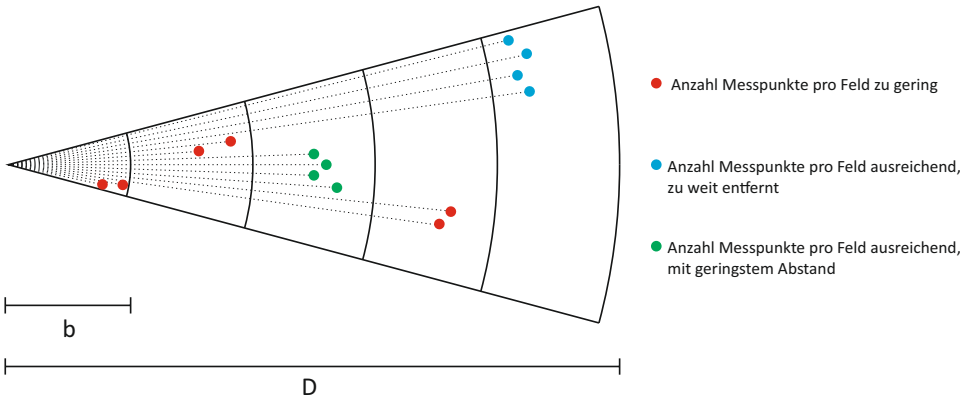


Abbildung 3.4: Konzept des Histogramm Filters

Mit diesem Verfahren ist sichergestellt, dass keine Ausreißer die Messung stören können. Des Weiteren wird durch die Mittelwertbildung das Rauschen reduziert.

3.4.1 Überlagernde Felder

Eines der Hauptprobleme des Histogramm Filters ist das sprunghaftes Verhalten der gefilterten Abstandswerte, für den Fall, dass die Messpunkte ungünstig, z.B. um eine Feldgrenze, verteilt liegen (vgl. Abb.: 3.5, oben). Verstärkt durch das Rauschen der Abstandsmessungen, sowie die Eigenbewegung des UAVs, kann es vorkommen, dass zunächst ausreichend Messpunkte pro Feld vorkommen, in der darauf folgenden Messung jedoch nicht. So könnten kurzzeitig Hindernisse nicht erkannt werden, was zu einem sprunghaften Verhalten führen und eine Regelung auf Basis dieser Werte negativ beeinflussen würde.

Eine mögliche Lösung ist das Hinzufügen eines zweiten Histogramm Filters für jeden Sektor, bei dem die Felder um die Hälfte der Feldbreite b versetzt sind (vgl. Abb.: 3.5). Somit würde zu jeder Zeit von einem der beiden Filter das korrekte Hindernis erkannt werden. Die Fusion der beiden Filter erfolgt einfach über den Vergleich der beiden gefilterten Werte, von denen der mit dem geringsten Abstand ausgewählt wird. Zusätzlich wird dadurch bei der Bildung des Mittelwertes eine höhere Genauigkeit erreicht. Diese Methode ist jedoch aufwendiger, da die doppelte Anzahl an Histogramm Filtern benötigt wird und verarbeitet werden muss.

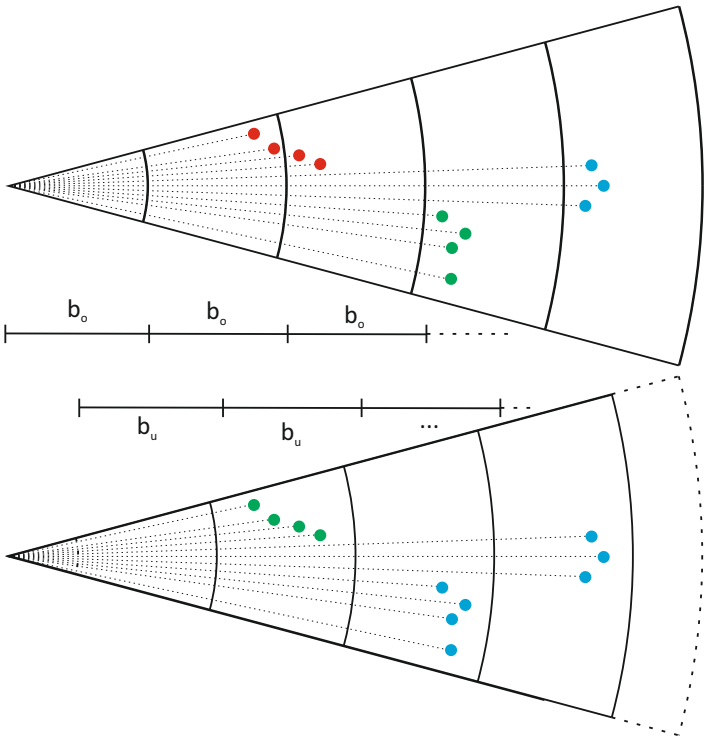


Abbildung 3.5: Überlagernde Felder des Histogramm Filters,

3.4.2 Überlagernde Sektoren

Ein ähnliches Problem tritt auf, falls ein Objekt auf der Sektorgrenze gemessen wird (vgl. Abb.: 3.6). Ein möglicher Lösungsansatz wäre hierfür ein sich Überschneiden der Sektoren, welche im Anschluss fusioniert werden müssen.

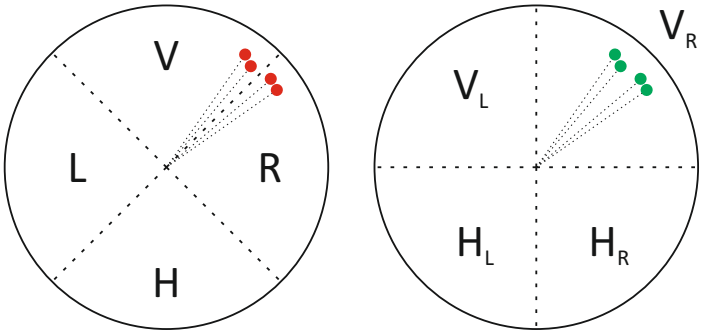


Abbildung 3.6: Überlagernde Sektoren der Histogramm Filter

3.4.3 Zeitliche Betrachtung

Ein weiterer Ansatz, das sprunghafte Verhalten der Messwerte zu verhindern, ist die vorherigen Messwerte der Histogramm Felder zu betrachten. An dieser Stelle wurde sich für einen exponentiell geglätteten Mittelwertfilter (EWMA, siehe Gleichung 3.8) entschieden, so dass nur der vorherige Messwert betrachtet werden muss. Dies hat den Vorteil einer deutlich kürzeren Verarbeitungszeit im Vergleich z.B. zu einem gewichteten Mittelwertfilter.
Falls ein Objekt in einem Feld eines Sektors gemessen wurde, wird angenommen, dass dieses sich bei der nächsten Messung wahrscheinlich immer noch in diesem Feld befindet. Dazu werden die Eigenschaften des Histogramm Filters der vorherigen Messung (Anzahl und Abstand pro Feld) gespeichert und mit dem aktuellen Histogramm Filter verglichen. Die Anzahl der Messpunkte des vorherigen und des aktuellen Felds werden addiert und mit einem festgelegten Schwellwert verglichen.
Der gefilterte Abstand wird anhand des EWMA-Filters über

$$d = \alpha \cdot d_{aktuell} + (1 - \alpha) \cdot d_{vorher} \tag{3.8}$$

mit Parameter $\alpha \leq 1$ berechnet.

3.5 Nachbearbeitung

Bei großen Abständen kann es vorkommen, dass nur wenige, oder nur sporadisch genügend Abstandsmessungen gemessen werden können, da aufgrund der Entfernung nur wenige Strahlen mit genügend Intensität reflektiert werden. Ist dies der Fall, kommt es ebenfalls zu einem sprunghaften Verhalten, in dem der gemessene Wert des Sektors zwischen der maximalen Messreichweite des Laserscanners (kein Objekt detektiert) und einem zufälligen Messwert hin und her springt. Um dem entgegen zu wirken, wird eine maximale Geschwindigkeit v_{max} des UAVs definiert, welche ausreichend groß gewählt werden muss. Aus dieser lässt sich die maximale Entfernungsändernung Δd pro Scan bestimmen. Unter der Annahme, dass sich das Hindernis nicht bewegt, kann sich der aus dem Histogramm Filter bestimmte Abstand des Sektors von zwei aufeinander folgenden Messungen nicht um mehr als Δd verändern. Ist dies dennoch der Fall, so kann die

Messung für diesen Scan ignoriert werden. Für die nächsten Scans wird $2\,\Delta d, 3\,\Delta d, \ldots, n\,\Delta d$ für diesen Sektor angenommen, so lange bis gilt:

$$d_{aktuell} < d_{vorher} + i\,\Delta d \tag{3.9}$$

wobei d_{vorher} der Messwert aus den i-vorherigen Schritten ist und konstant bleibt, bis 3.9 erfüllt ist. Dann wird $d_{aktuell}$ als Abstandswert des Sektors übernommen und i zurückgesetzt. Alternativ kann der Messwert auch Schrittweise um Δd erhöht werden, so dass der gefilterte Wert nicht fälschlicherweise konstant bleibt.
Für einen Laserscanner mit einer Abtastzeit von 200 ms und einer maximalen Geschwindigkeit $v_{max} = 2{,}5\,\frac{m}{s}$ ergibt sich eine maximale Entfernungsänderung von $\Delta d = 0{,}5$ m. Anzumerken ist, dass nur Abstandsmessungen gefiltert werden deren Wert größer ist als die vorherige Messung. Somit werden kleinere Abstandsmessungen immer berücksichtigt.
Bei einer negativen Abstandsänderung (von nah auf weit), kommt es zu dem Effekt, dass ein Objekt noch für kurze Zeit vom Filter detektiert wird (bis 3.9 erfüllt), obwohl es sich nicht mehr im Sichtbereich des Sektors befindet. Dies muss im Kollisionsvermeidungsregler berücksichtigt werden.

3.6 Dynamische Erkennung von Objekten

Um die vier Abstandswerte (Vorne, Hinten, Links, Rechts) zur Kollisionsvermeidung zu bestimmen, ist es nicht zwingend notwendig, den Messbereich des Laserscanners in Sektoren aufzuteilen. Ein anderer Ansatz ist das dynamische Erkennen von Objekten, beispielhaft dargestellt in Abb.: 3.7.
Bei dieser Methode werden die Messpunkte gruppiert, falls der Abstand zweier Messpunkte kleiner als ein bestimmter Grenzwert ist. Die Messpunkte werden hierzu als Punkte in Polarkoordinaten mit Eigenschaft $P(r, \varphi)$ betrachtet. Für den Abstand zweier Punkte $P_1(r_1, \varphi_1))$, $P_2(r_2, \varphi_2)$ in Polarkoordinaten ergibt sich

$$d = \sqrt{r_1^2 + r_2^2 - 2r_1 r_2 \cos(\varphi_2 - \varphi_1)}. \tag{3.10}$$

Es werden iterativ immer zwei aufeinander folgende Messwerte miteinander verglichen und je nach Abstand gruppiert, mit dem nächsten Messpunkt verglichen oder verworfen. Um Rauschen zu vermeiden müssen Objekte mit einer geringen Anzahl an Messpunkten ebenfalls verworfen werden. Zusätzlich muss eine maximale Anzahl an Punkten pro Objekt definiert werden, so dass längliche Objekte mit vielen Messpunkten, wie zum Beispiel Wände, in mehrere Objekte aufgeteilt werden. Der Vorteil dieses Konzepts gegenüber der Einteilung in Sektoren ist, dass die Position der Objekte genauer bekannt ist.

Im nächsten Schritt muss der kürzeste Abstand und sowie Winkel zu jedem Objekt bestimmt werden. Hierzu können die Messpunkte jedes Objekts zunächst vorverarbeitet werden. Eine Möglichkeit wäre die Anwendung eines gewichteten Mittelwertfilters. Eine andere Methode wäre die Anwendung linearer Regression der Messpunkte, unter der Annahme, dass sich alle Objekte aus Teilstücken von Geraden beschreiben lassen können. Je nachdem in welchem Quadranten (Vorne, Hinten, Links, Rechts) sich das Objekte befindet, können so leicht die Abstände zur Kollisionsvermeidung bestimmt werden. Es kann auch direkt Anhand der Vektoren $O_i(r, \varphi)$ eine Regelung realisiert werden.

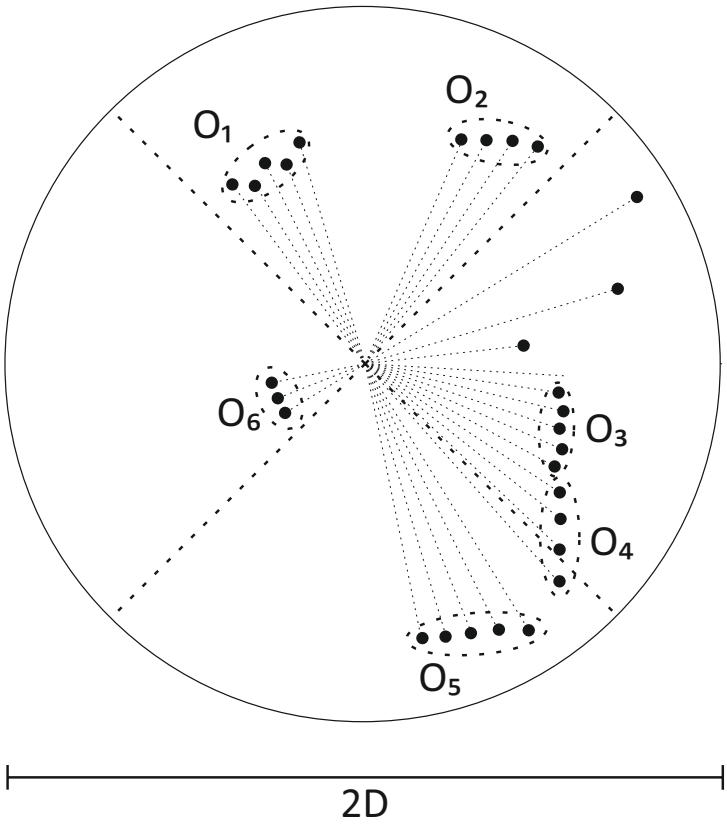


Abbildung 3.7: Prinzip des dynamischen Erkennens von Objekten

3.7 Rekursiver Bayes Filter

Eine weitere Möglichkeit ist die Verwendung eines so genannten Occupancy Grids [14] (beispielhaft dargestellt in Abb. 3.8) unter der Verwendung des Rekursiven Bayes Filters [5, pp. 77–81]. Jede Zelle des Occupancy Grids besitzt eine feste Größe und repräsentiert hier eine Zufallsvariable $P(H)$. Diese gibt angibt, mit welcher Wahrscheinlichkeit diese Zelle blockiert ($P(H) = 0$), oder frei ($P(H) = 1$) ist. Zur Vereinfachung wird angenommen, dass die Wahrscheinlichkeit, dass eine Zelle belegt ist, unabhängig von den Wahrscheinlichkeitswerten ihrer Nachbarzellen ist. Das Ziel dieses Algorithmus ist es, die Messdaten des Laserscanners in die Map (Occupancy Grid) zu integrieren. Es ist anzumerken, dass hierbei keine sich erweiternde Karte der gesamten Umgebung erstellt werden soll (wie bei SLAM), sondern lediglich die aktuelle Umgebung, definiert durch die maximale Messreichweite des Laserscanners. Es findet keine Lokalisierung statt, das UAV befindet sich also stets im Mittelpunkt des Occupancy Grid.

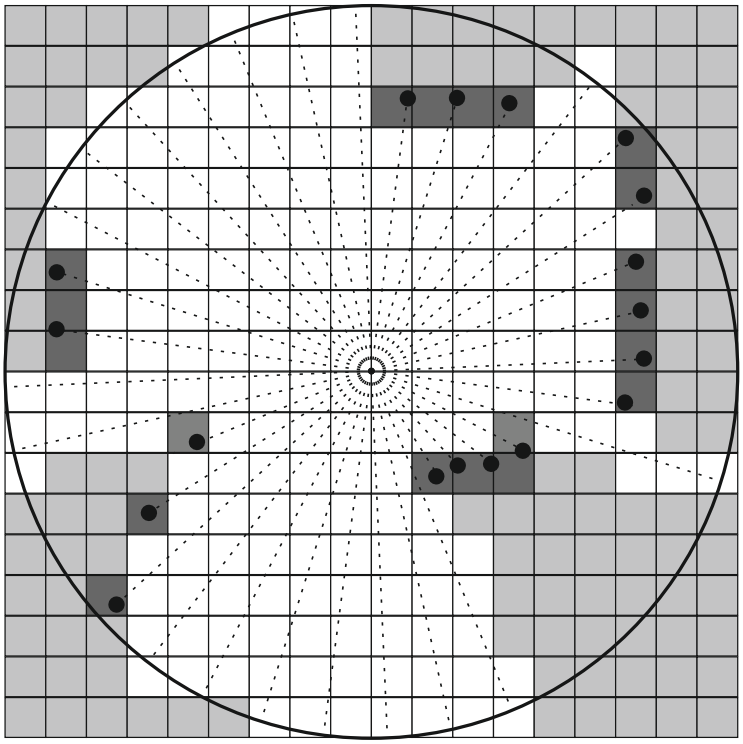


Abbildung 3.8: Prinzip und Darstellung eines Occupancy Grids

Zu Beginn wird eine uniforme Verteilung der Wahrscheinlichkeit der Map angenommen, vorausgesetzt, dass zu diesem Zeitpunkt keine Informationen über die Umgebung existieren. Es wird also für alle Zellen $P(H) = 0,5$ angenommen. Unter Verwendung des Bayes-Theorems

$$P(H|D) = \frac{P(H) \times P(D|H)}{P(D)} \tag{3.11}$$

können nun Messdaten in die Map integriert werden. Wobei $P(H|D)$ als „posterior“ bezeichnet wird, also in diesem Fall der Wahrscheinlichkeit der Zelle, dass diese frei oder blockiert ist, nachdem die Messung D integriert wurde. $P(H)$ wird auch als „prior“, also als der vorherige Zustand der Zelle, aufgefasst. $P(D)$ dient der Normalisierung der Zufallsvariable, $P(D|H)$ beschreibt die Vertrauenswürdigkeit der Messung und gibt die Wahrscheinlichkeit an, ob diese Zelle tatsächlich frei ist, gegeben, dass diese vom Laserscanner als frei gemessen wurde. Für $P(D|H)$ werden konstante Werte angenommen, beispielsweise

$$P(D|\text{„}frei\text{“}) = 0,8 \tag{3.12}$$

$$P(D| \sim \text{„}frei\text{“}) = 0,2. \tag{3.13}$$

Der Filter wird als rekursiv bezeichnet, da die Zufallsvariable $P(H)$, welche eine Zelle des Occupancy Grids repräsentiert, mit $P(H|D)$ aktualisiert wird. Nach Integration des Messdaten gilt

$$P(H) = P(H|D). \tag{3.14}$$

Somit folgt für jede Aktualisierung

$$P(H|D) = \frac{P(H) \times P(D|H)}{P(H) \times P(D|H) + (1 - P(H)) \times P(D| \sim H)}. \tag{3.15}$$

Um die Map zu aktualisieren wird jede Abstandsmessung als Strahl aufgefasst und bestimmt, welche Zellen der Map dieser Strahl durchläuft (vgl. Abb.: 3.8). Alle durchlaufenen Zellen werden als „frei“ aufgefasst und anhand der Gleichungen 3.15 und 3.12 aktualisiert. Endet der Strahl in einer Zelle, so wird diese als blockiert aufgefasst und anhand der Gleichungen 3.15 und 3.13 aktualisiert. Eine Zelle wird erst als tatsächlich „frei“ angenommen, falls beispielsweise $P(H) >$ 0,95 ist, was bedeutet, dass die Zelle mit einer Wahrscheinlichkeit von mehr als 95 % „frei“ ist.

Umgekehrt äquivalent gilt dies für „blockierte“ Zellen.
Anhand dieser Karte der Umgebung können Objekte und Hindernisse erkannt werden. Aufgrund der Wahrscheinlichkeitsverteilung eignet sich der Bayes Filter gut um verrauschte Messdaten und Messfehler zu korrigieren. Andererseits benötigt das Auffassen und Einfügen der Abstandsmessungen als Strahlen in die Map hohe Rechenleistung, um in Echtzeit zu funktionieren.

4 Implementierung

4.1 Überblick

Das folgende Kapitel befasst sich mit der Implementierung und Integration eines Laserscanners in das vorhandene System. In Abschnitt 4.2 soll zunächst die Implementierung der Software zur Signalverarbeitung eines Laserscanners beschrieben werden und dabei auf die Integration in das vorhandene Systemzur Kollisionsvermeidung eingegangen werden. Darauf wird in Abschnitt 4.3 der verwendete SLAM-Algorithmus auf Basis des Robot Operating Systems vorgestellt. In Abschnitt 4.4.1 soll zunächst der verwendete Laserscanner RPLidar vorgestellt werden und in Abschnitt 4.4.2 das verwendete und bereits vorhandene UAV-System des AQopterI8 Projekts. Abschnitt 4.4.3 befasst sich mit der Hardwareintegration des RPLidar und erläutert die Problematik bei der Verwendung des Laserscanners auf dem AQopterI8.

4.2 Softwaredesign

Die bereits bestehende Kontrollsoftware des AQopterI8-Projekts wurde in QT [32] und somit in der Programmiersprache C++ geschrieben. Die Signalverarbeitung des RPLidars wurde in die bestehende Kontrollsoftware integriert. Es wurde das Konzept des Histogram Filters aus Abschnitt 3.3 und Abschnitt 3.4 implementiert, worauf im folgenden Abschnitt detaillierter eingegangen wird. Die errechneten Sektoren werden über die UART-Schnittstelle an den Mikrocontroller gesendet und dort weiter zur Kollisionsvermeidung genutzt. Zusätzlich werden die einstellbaren Parameter des Filters und die GUI-Elemente erklärt.

4.2.1 Implementierung des Histogram Filters

Um die Signalverarbeitung möglichst modular und erweiterbar zu gestalten, wurden die unterschiedlichen Eigenschaften der Filter in Klassen ausgelagert, um durch Vererbung möglichst redundanzfreien und leicht wartbaren Code zu erzeugen. Außerdem wurde das Konzept des Polymorphismus verwendet, so dass zur Laufzeit des Programms zwischen den einzelnen Filtern gewechselt werden kann. Die Darstellung der Filterklassen soll in Abb.: 4.1 veranschaulicht werden.

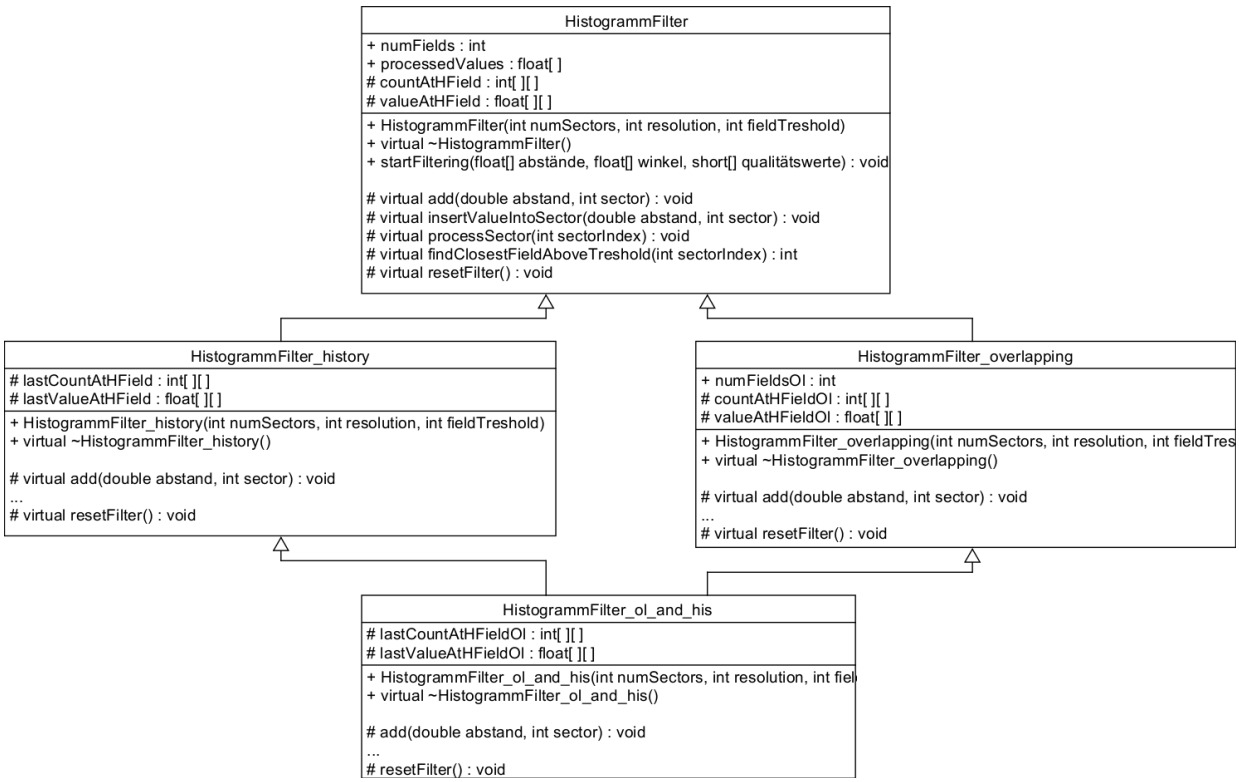


Abbildung 4.1: UML Diagramm der Filterklassen

Der Zusammenhang zwischen den implementierten Klassen und den Konzepten aus Kapitel 3 ergibt sich aus Tabelle 4.1.

Funktionsweise

Die Filter können dynamisch in Abhängigkeit der Parameter numSectors (Anzahl der Sektoren), resolution (Feldbreite b) und fieldTreshold (Schwellwert) initialisiert werden (vgl. Abschnitt 3.4). Die Methode startFiltering(..) der Oberklasse HistogrammFilter bekommt als Übergabeparameter die Rohdaten des Laserscanners (vgl.: Abschnitt 4.4.1) übergeben. Sie ist die Initialmethode,

Nummer	Klassenname	Kapitel und Beschreibung
1	HistogrammFilter	Abschnitt 3.4: Standard Histogramm Filter ohne Erweiterung.
2	HistorgrammFilter_overlapping	Abschnitt 3.4.2: Histogramm Filter mit überlagernden Feldern.
3	HistogrammFilter_history	Abschnitt 3.4.3: Histogramm Filter mit zeitlicher Betrachtung und EWMA-Filter.
4	HistogrammFilter_ol_and_his	Kombination aus 2 und 3.

Tabelle 4.1: Zusammenhang der Konzepte und implementierten Klassen.

welche die Rohdaten mit Hilfe der fünf virtuellen Methoden auf die entsprechenden Sektoren und Felder des Histogramm Filters aufteilt. Die gefilterten Sektoren sind nach Ablauf der Methode in proccessedValues gespeichert, werden an den Mikrocontroller weitergeleitet und können ebenfalls in der GUI dargestellt werden (siehe Abschnitt 4.2.3).

4.2.2 Zwischenwertberechnung

Der bereits vorhandene Regler zur Kollisionsvermeidung wurde für Infrarot- und Ultraschallsensoren ausgelegt [17], welche mit einer höheren Abtastfrequenz als der RPLidar Messdaten bereitstellen können. Um ein ähnliches Regelverhalten zu erreichen, müssten entweder die Regelparameter für den RPLidar neu eingestellt werden, oder die Abtastfrequenz erhöht werden. Da letzteres physikalisch nur bis zu einem Grenzwert möglich ist, wurden softwareseitig Zwischenwerte aus den letzten gültigen Messungen für jeden Sektor berechnet, die zur Regelung benutzt werden können. Die Errechnung der Zwischenwerte ergibt sich aus folgender Gleichung:

$$d_z = d_0 + \sum_{n=1}^{N} \frac{d_0 - d_{0-n}}{2 \cdot n \cdot N} \tag{4.1}$$

Wobei d_0 dem letzten Sektorwert entspricht, d_{0-n} dem Wert aus der n-vorherigen Messung und N die Anzahl der betrachteten, in der Vergangenheit liegenden Werte ist.
Die anhand Gleichung 4.1 vorhergesagten Zwischenwerte sind jedoch fehlerbehaftet, ermöglichen aber eine Regelung mit der doppelten Abtastfrequenz des RPLidars. Das Intervall des Kollisionsvermeidungsreglers konnte somit von 180 ms auf 90 ms gesenkt werden, für das es ursprünglich ausgelegt wurde.

4.2.3 GUI-Design

Die bereits bestehende Anzeige zur Umgebungserfassung der AQopterI8-Software wurde genutzt und um die einstellbaren Parameter der Signalverarbeitung erweitert. Die einzelnen GUI Parameter sollen im Folgenden erklärt werden und der Bezug zu den Konzepten aus den entsprechenden Kapiteln hergestellt werden. Die GUI ist in Abb.: 4.2 dargestellt und die Parameter wurden nummeriert.

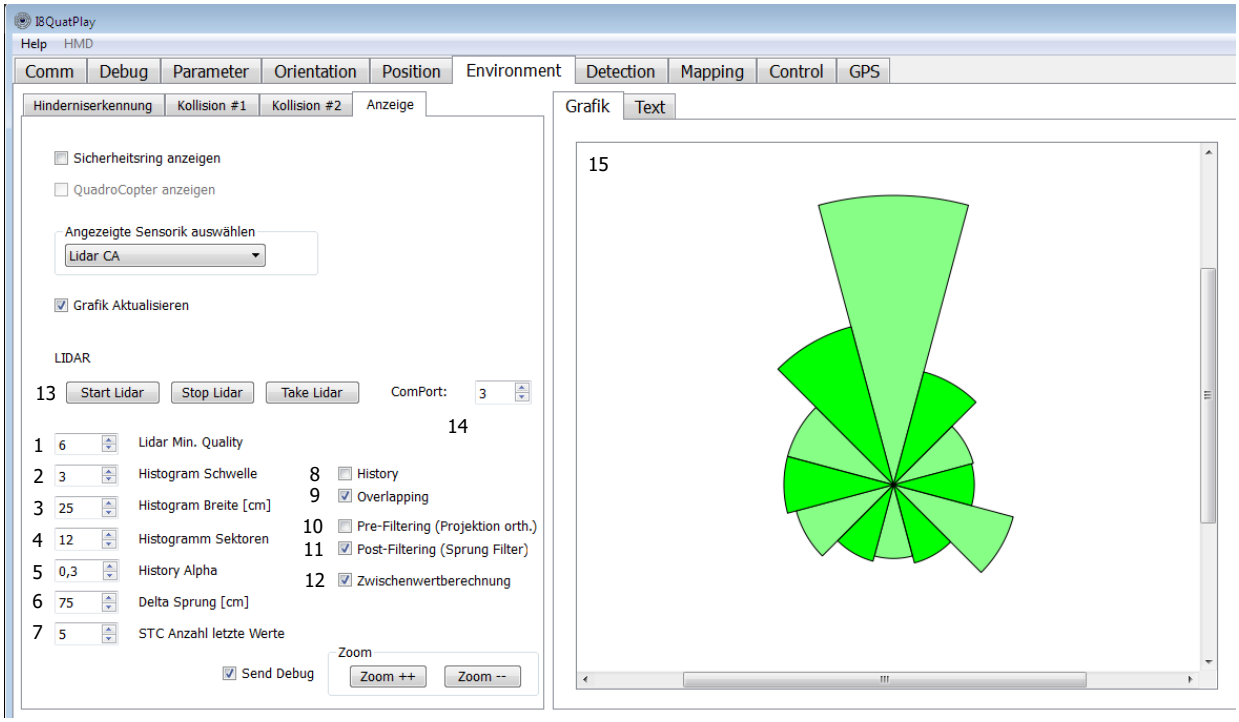


Abbildung 4.2: Grafische Benutzeroberfläche der Kontrollsoftware

Einstellbare Parameter

1. Minimaler Qualitätswert des reflektierten Lichtsignals des RPLidar, um als gültige Messung in die Filterung einzugehen (vgl. Abschnitt 4.4.1).
2. Minimale Anzahl an benötigten gültigen Messung pro Histogrammfeld (siehe Schwellwert in Abschnitt 3.4).
3. Definiert die Feldbreite b eines Histogrammfelds (vgl. Abb.: 3.4).
4. Anzahl an Sektoren in die das Sichtfeld des Laserscanners aufgeteilt wird (vgl. Abb.: 3.3).
5. Parameter α des EWMA Filters (vgl. Abschnitt 3.4.3).
6. Maximale Änderung des gefilterten Messwerts pro Scan (vgl. Abschnitt 3.5).

7. Anzahl der betrachteten, in der Vergangenheit liegenden Werte N (vgl. Formel 4.1).
8. Aktiviert den Histogramm Filter mit zeitlicher Betrachtung (vgl. Abschnitt 3.4.3).
9. Aktiviert den Histogramm Filter mit überlagernden Histogramm Feldern (vgl. Abschnitt 3.4.1).
10. Aktiviert die Vorverarbeitung der Rohdaten des Laserscanners (vgl. Abschnitt 3.3).
11. Aktiviert die Nachbearbeitung der gefilterten Sektorwerte (vgl. Abschnitt 3.5).
12. Aktiviert die Berechnung der Zwischenwerte (vgl. Abschnitt 4.2.2).
13. Startet bzw. stoppt die Signalverarbeitung des Laserscanners.
14. Spezifiziert den Port der seriellen Schnittstelle an den der RPLidar angeschlossen ist.
15. Dient der Visualisierung der gefilterten Sektorwerte.

4.3 Robot Operating System

Um eine Kartierung der Umgebung zu ermöglichen, wurde auf das bereits in Abschnitt 2.3.1 erwähnte Robot Operating System zurückgegriffen. ROS wurde ursprünglich für Ubuntu-Derivate entwickelt und wurde daher auf Linux Mint MATE auf dem Pico-ITX Board implementiert. Als ROS-Version wurde das letzte stabile Release Indigo Igloo verwendet.
ROS basiert auf dem Publisher-Subscriber Modell, welches eine einfache Kommunikation zwischen mehreren Packages erlaubt und vordefinierte Nachrichten-Typen zur Verfügung stellt.
Für den RPLidar existiert für ROS ein offizielles Treiber-Package [39], welches auf einem dafür vorhergesehenen Topic „scan“die Laserscannerdaten veröffentlicht.
Zur Kartierung wurde das variable hector_slam Package verwendet [26], da mit diesem, je nach zusätzlich verwendeter Sensorik, 2D- oder 3D-Karten der Umgebung auf Basis von Laserscannerdaten erstellt werden können. Zusätzlich bietet es eine Option zur Lokalisierung ohne jegliche Bewegungsinformationen, welche hier verwendet wurde. Das hector_slam Package greift dazu lediglich auf die Rohdaten des RPLidars über das Topic „scan“des RPLidar Package zu. Die Übereinstimmung und Angleichung der Laserscannerdaten erfolgt hierbei über eine Normalverteilungs-Transformation [6]. Die Lokalisierung basiert demnach ausschließlich auf den Daten des Laserscanners und ermöglicht eine simultane Lokalisierung und Kartenerstellung der Umgebung.

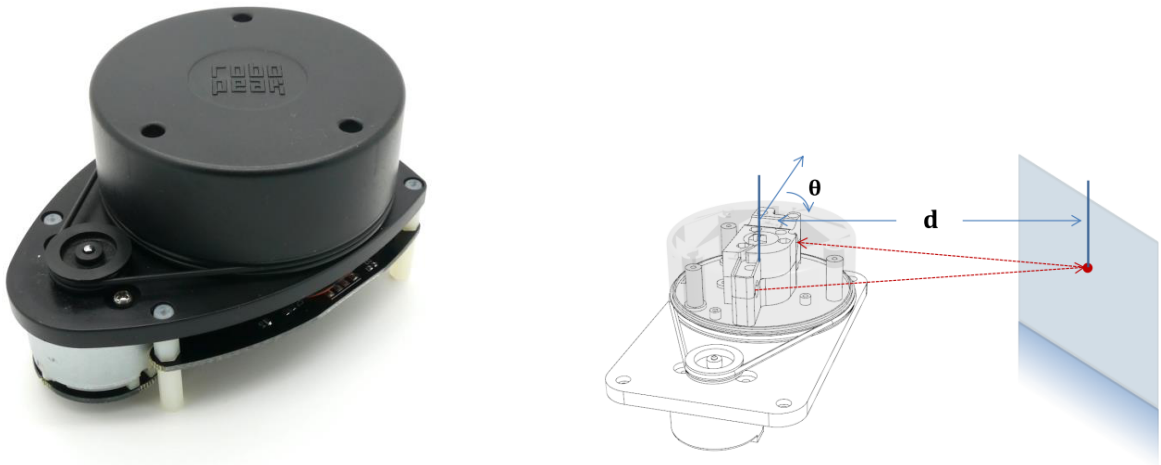
4.4 Verwendete Hardware

4.4.1 Robopeak Lidar

Als Laserscanner wurde das Development Kit RPLidar der Firma RoboPeak eingesetzt [38]. Aufgrund seines geringen Gewichts (200 g) und Abmessungen (70x98.5x60 mm) eignet sich dieser Sensor sehr gut zum Einsatz auf kleineren UAVs.

Der 2D-Laserscanner nutzt einen 3 mW-Laser mit einer Wellenlänge von 785 nm und ermöglicht es, unter Verwendung eines Triangulationsensors, Objekte in einer Reichweite von bis zu 6 m zu erfassen. Der Aufbau des Laserscanners ermöglicht eine 360 °-Umgebungserfassung, so dass der RPLidar als einzig notwendiger Sensor zur Hinderniserkennung eingesetzt werden kann. Bei einer Abtastfrequenz von 5 Hz liefert der Sensor eine Winkelauflösung von ca. einem Grad, dies entspricht bis zu 360 gültigen Messungen pro Scan. Die maximale Abtastfrequenz beträgt 10 Hz. Über einen 7-adrigen Molexstecker kann der RPLidar direkt angesteuert werden. Neben Strom- und Spannungsversorgung benötigt der Laserscanner ein PWM-Signal zur Steuerung des Motors. Die Kommunikation erfolgt über eine UART-Schnittstelle. Alternativ ist die Verwendung der mitgelieferten USB-Bridge möglich, welche alle oben genannten Anforderungen bereits integriert.

Als Rohdaten liefert der RPLidar neben Abstandsmessung d und Winkel θ noch zusätzlich einen Qualitätswert für jede Messung, der angibt, mit welcher Intensität das Lasersignal reflektiert wurde.



(a) Darstellung Quelle: Robopeak [37] **(b)** Funktionsweise Quelle: Robopeak [37, p. 2]

Abbildung 4.3: Der verwendete Laserscanner RPLidar.

4.4.2 AQopterI8

Als Basis wird ein Quadrokopter des AQopterI8-Projekts des Lehrstuhls für Informatik VIII der Universität Würzburg verwendet. Der modulare Aufbau (dargestellt in Abb.: 4.4) erlaubt die Nutzung einer Vielzahl von bereits implementierten Sensoren, welche je nach Einsatzzweck und Umgebung angepasst werden können. Zusätzlich zu dem AVR32 Mikrocontroller AT32UC3A0512 wird das LP-180 Pico-ITX Board mit Windows 7 als Betriebssystem verwendet. Der Mikrocontroller verarbeitet Höhen- und Lageregelungsdaten, wird zur Steuerung des Quadrokopters benötigt und implementiert den Algorithmus zur Kollisionsvermeidung. Das Pico-ITX Board wird zur Signalverarbeitung der Laserscannerdaten sowie zur Implementation der Filterkonzepte aus Abschnitt 4.2 verwendet. Die Kommunikation des Mikrocontrollers und des Pico-ITX Boards erfolgt über eine Universal Synchronous/Asynchronous Receiver Transmitter-Schnittstelle (USART). Der RPLidar wird über die USB-Bridge direkt an das Pico-ITX Board angeschlossen. Des Weiteren wird ein USB-WLAN Modul genutzt um Remote-Zugriff auf das Pico-ITX Board zu ermöglichen.

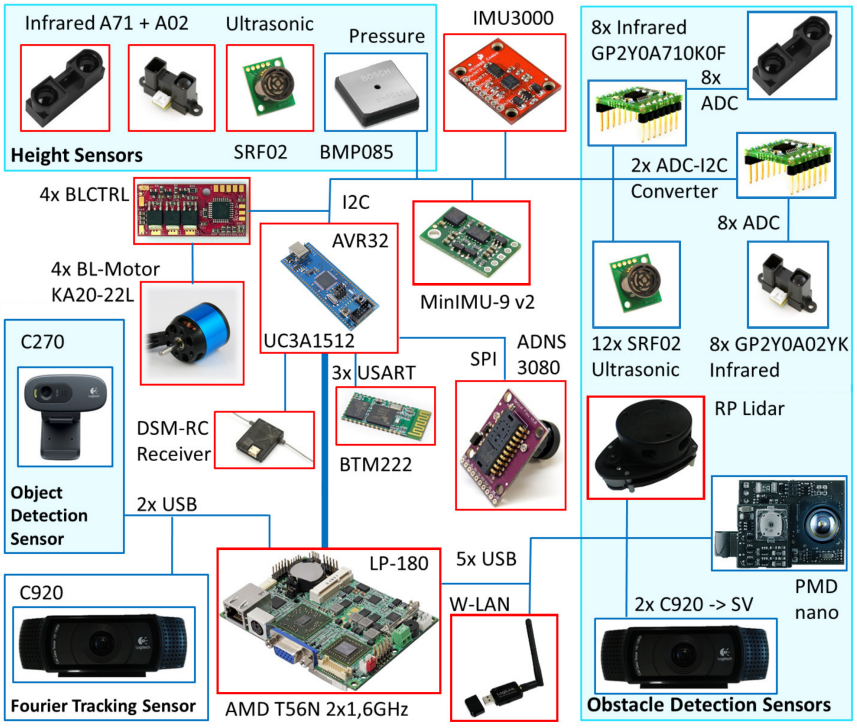


Abbildung 4.4: Modulare Hardware und Sensoren des AQopterI8, verwendete Hardware in rot markiert, Quelle: Gageik et al. [17, p. 2]

4.4.3 Hardwareintegration

Halterung Version 1

Um den RPLidar in das bestehende System zu integrieren, wurde eine Halterung entworfen (vgl. Abb. 4.5), welche möglich stabil und gleichzeitig leicht sein soll. Zusätzlich darf das Sichtfeld des Laserscanners nicht beeinträchtigt werden. Die Halterung in Abb. 4.5 wurde aus Acrylnitril-Butadien-Styrol-Kunststoff (ABS) mit dem Replicator 2X 3D-Drucker der Firma Makerbot mit 20% Füllungsgrad (infill) gedruckt, um ein möglichst leichtes und ausreichend stabiles Bauteil zu erhalten. Des Weiteren muss der Laserscanners auf der Oberseite des Quadrokopters befestigt werden, um der oben genannten Anforderung an das Sichtfeld des Laserscanners zu genügen. Allgemein wäre es aufgrund des relativ hohen Gewichts, im Vergleich zu den anderen Komponenten des Quadrokopters, sinnvoll den Laserscanner in der Nähe des Schwerpunkts des Quadrokopters zu platzieren, um die Flugeigenschaften nicht negativ zu beeinflussen. Dies hätte jedoch die Folge, dass die dort verbaute Hardware einzelne Bereiche des Sichtfelds des Laserscanners verdecken würde.
Zudem wurde eine weitere Halterung für den WLAN-Adapter an der Halterung für den Laserscanner angebracht, welche ebenfalls so integriert ist, dass der Adapter das Sichtfeld nicht verdeckt.

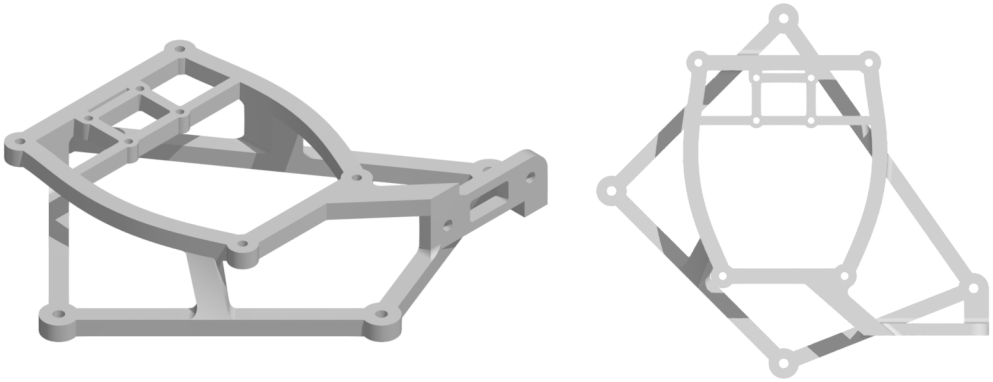


Abbildung 4.5: Design der Halterung: Version 1

Problematik Die Auswirkung der Eigenrotation des Laserscanners auf den gesamten Aufbau des Quadrokopters wurde zunächst unterschätzt. Selbst im stationären Zustand war ein Aufschwingen der Konstruktion beobachtbar. Während des Flugs zeigte sich ein sehr starker Drift der Lageerfassungssensoren, welche nach wenigen Minuten bereits mehr als 10 ° erreichte und manu-

ell ausgeglichen werden musste. Eine autonome Kollisionsvermeidung war somit nicht möglich. Um andere Fehlerquellen auszuschließen wurde der Inertialsensor getauscht, sowie ein zusätzliches Accelerometer eingesetzt, die Aufhängung der Sensoren getauscht und eine zusätzliche Dämpfplatte eingebaut. Diese Maßnahmen erbrachten zwar eine Verbesserung, konnten das Problem jedoch nicht vollständig ausbessern. Es ist außerdem anzumerken, dass kein Drift auftrat, wenn der Laserscanner montiert, jedoch nicht angeschlossen war.

Halterung Version 2

Um den oben angesprochenen Problemen entgegen zu wirken, wurde eine zweite Halterung entwickelt (vgl. Abb.: 4.6). Zusätzliche vertikale Halterungen verleihen der Konstruktion deutlich mehr Stabilität. Durch die erhöhte Steifigkeit werden die Vibrationen durch den Laserscanner zwar nicht gedämpft, jedoch das Aufschwingen verhindert.

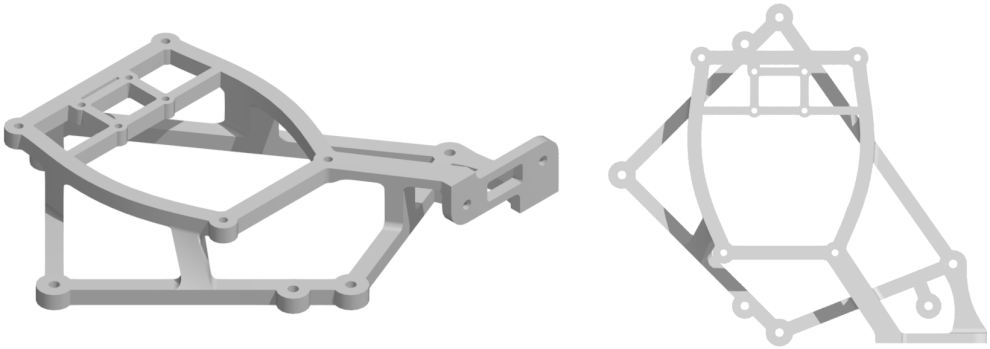


Abbildung 4.6: Design der Halterung: Version 2

Darüber hinaus wurde die Halterung für das Pico-ITX Board (vgl. Abb.: 4.7) so angepasst, dass es auf horizontaler Ebene alle vertikalen Abstandshalter verbindet, so dass eine noch höhere Biegesteifigkeit erreicht wird.

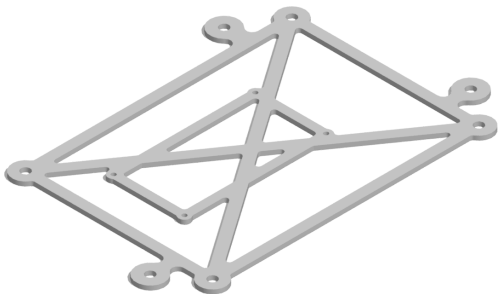


Abbildung 4.7: Anpassung der PC Halterung

Zusätzlich wurden die Propeller ausgewuchtet, einer der Motoren getauscht, sowie der I-Teil des Lagereglers angepasst, um eine Schieflage des Quadrokopters bei Start zu vermeiden. Außerdem wurde die Kalibrierung der Lagesensoren im ausgeschalteten Zustand des Laserscanners durchgeführt, so dass durch die Rotation des Laserscanners und somit einhergehenden Vibrationen die Kalibrierung nicht gestört werden kann.
Mit diesem Design und den weiteren Verbesserungen konnte der Sensordrift minimiert und der Laserscanner in das System integriert werden.

5 Evaluierung

5.1 Überblick

Das folgende Kapitel befasst sich mit der Evaluierung der in Kapitel 4 implementierten Konzepte. In Abschnitt 5.2.1 wird zunächst der zur Evaluation genutzte Testaufbau beschrieben und daraufhin in Abschnitt 5.2.2 die verschiedenen Histogramm Filter getestet und verglichen. Ferner wird in Abschnitt 5.2.4 auf die Berechnung der Zwischenwerte eingegangen und in Abschnitt 5.2.5 auf die Auswirkung der einstellbaren Parameter der Histogramm Filter.
In Abschnitt 5.3 werden anschließend die Ergebnisse des implementierten Mapping-Konzepts vorgestellt und zuletzt die Resultate der Kollisionsvermeidung unter Verwendung der Histogramm Filter zur Laserscanner basierten Hinderniserkennung präsentiert (Abschnitt 5.4).

5.2 Histogramm Filter basierte Hinderniserkennung

5.2.1 Testaufbau

Um die Hinderniserkennung evaluieren zu können, wurde das optische Tracking System Flex 3 von OptiTrack verwendet [27]. Mit einer Abtastzeit von lediglich $10ms$ und einer Genauigkeit im Millimeterbereich bietet das System eine deutlich höhere Genauigkeit als der verwendete Laserscanner und eignet sich somit zur Evaluierung. Die Messdaten des OptiTrack Systems wurden zur Auswertung über ein lokales Netzwerk an die Bodenstation gesendet. Die gefilterten Sektorwerte der Histogramm Filter wurden vom Pico-ITX Board per UART an den Mikrocontroller gesendet und weiter über Bluetooth verschickt, um eine möglich kurze Latenzzeit zu erhalten.
Der beschriebene Testaufbau wurde für alle folgenden Messreihen in diesem Abschnitt 5.2 verwendet.

Versuchsdurchführung

Der Quadrokopter wurde auf einem beweglichen Untersatz montiert und geradlinig auf ein senkrecht zur Bewegungsachse platziertes Hindernis hin und wieder zurück bewegt, um das zeitliche Verhalten der Abstandswerte vergleichen zu können. Um realitätsnahe Bedingungen zu simulieren wurde stets eine Bewegung mit geringer und eine Bewegung mit hoher Geschwindigkeit durchgeführt. Es wurde lediglich ein Sektor des Histogramm Filters ausgewertet, da sich das Verhalten für alle anderen Sektoren reproduzieren lässt.

Fehlerquellen

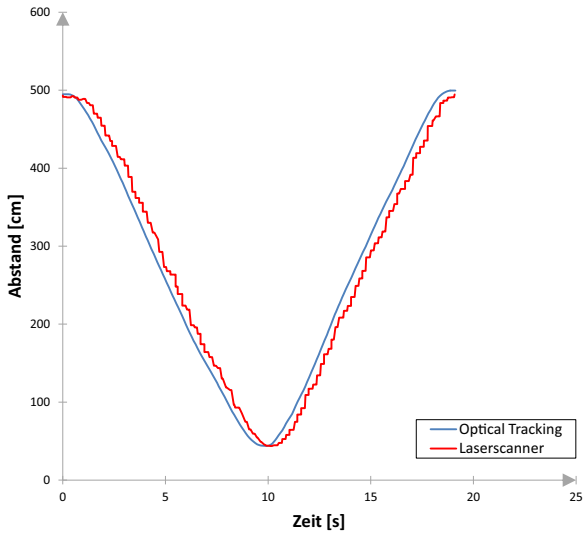
Der beschriebene Testaufbau weist einige Ungenauigkeiten auf. Zum einen ist die Diskrepanz zwischen den Abstandsmessungen des Laserscanners und den Messungen des optischen Tracking Systems zu nennen, welche selbst durch genaue Kalibrierung nur minimiert und nicht verhindert werden kann. Das System wurde so kalibriert, dass bei einem mittleren Abstand des Laserscanners von 2,5 m die Messwerte beider Systeme übereinstimmen. Es handelt sich hierbei um einen systematischen Fehler im unteren Zentimeterbereich. Allerdings ist dieser Fehler für alle Messungen gleich, für den Fall, dass sich Aufbau und Kalibrierung des Systems nicht ändert, so dass das zeitliche Verhalten dennoch ausgewertet und verglichen werden kann.
Andererseits sind zufällige Fehler, welche die Vergleichbarkeit der Messungen erschweren, nicht auszuschließen. Da die Bewegungen manuell ausgeführt wurden entstehen Messungenauigkeiten durch das Abweichung der geradlinigen Bewegung der einzelnen Messungen zueinander sowie durch die unterschiedlichen Geschwindigkeiten.

5.2.2 Vergleich der Histogramm Filter Klassen

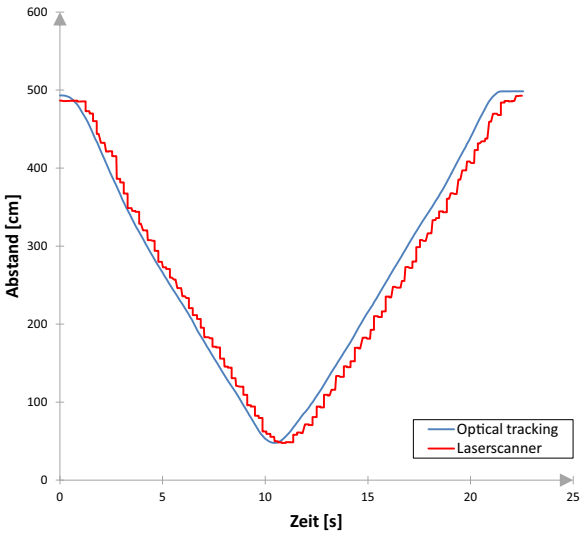
Um die Histogramm Filter besser vergleichen zu können, wurde auf eine Berechnung der Zwischenwerte verzichtet. Die Auswirkungen der Zwischenwertberechnungen werden separat in Abschnitt 5.2.4 behandelt.
Die Histogramm Filter aus Abb. 5.1a bis 5.1d und Abb. 5.2a bis 5.2d entsprechen hierbei den Konzepten aus Abschnitt 3.4, deren Implementierung in Abschnitt 4.2.1 beschrieben wurde.

Test 1: Niedrige Geschwindigkeit

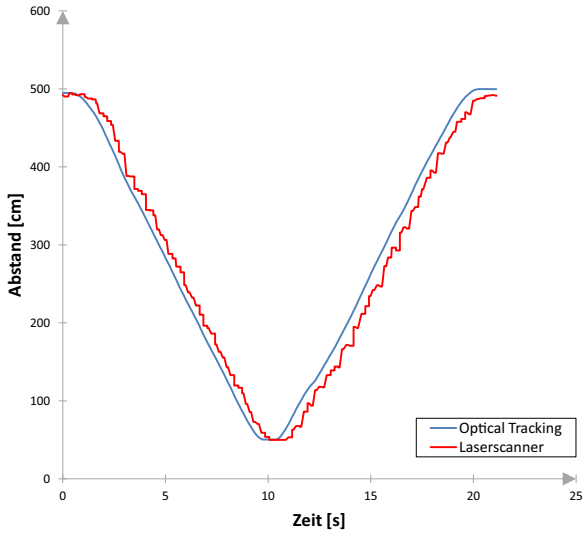
Betrachtet man das zeitliche Verhalten der Filter für niedrige Geschwindigkeiten in Abb.: 5.1, so fällt die Abweichung der gefilterten Abstandsmessungen des Laserscanners, im Vergleich zum optischen Tracking, für alle Filter gering aus. In Abb. 5.1b und 5.1c ist jedoch eine Latenz zu erkennen, welche vermutlich durch die zeitliche EWMA-Filterung der Messwerte induziert wird. Zusätzlich sind diese Signal stärker verrauscht.



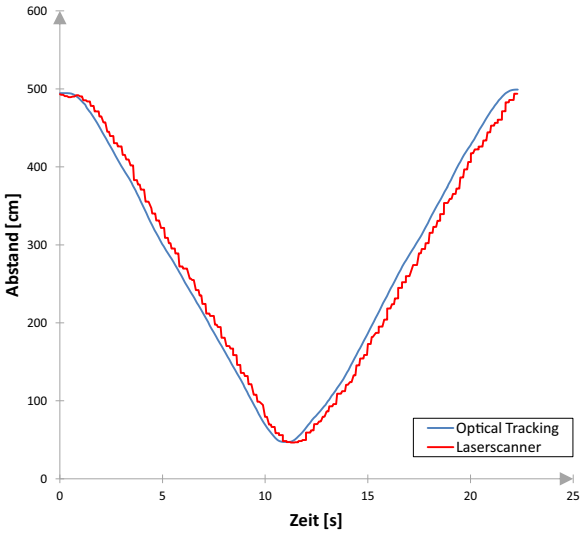
(a) Standard Filter



(b) Überlagernde Felder + zeitliche Betrachtung



(c) Zeitliche Betrachtung

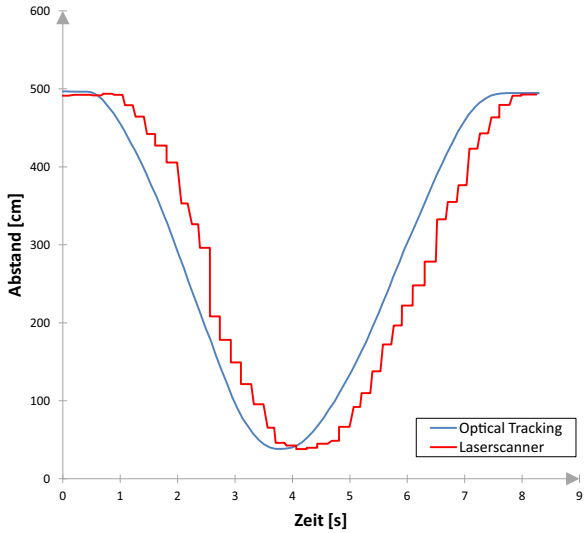


(d) Überlagernde Felder

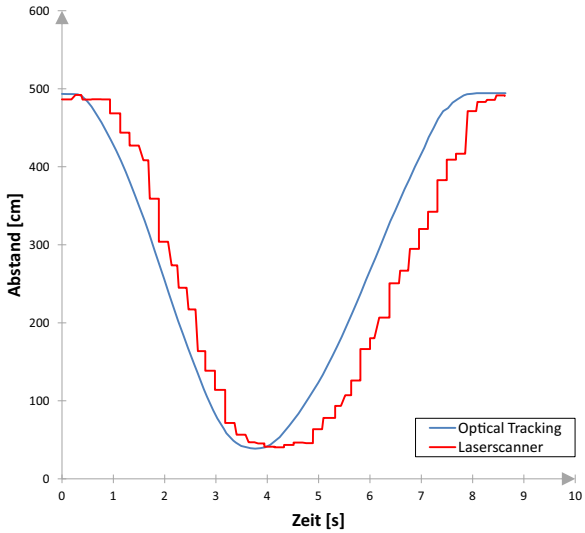
Abbildung 5.1: Vergleich des Histogramm Filter Verhaltens bei niedriger Geschwindigkeit

Test 2: Hohe Geschwindigkeit

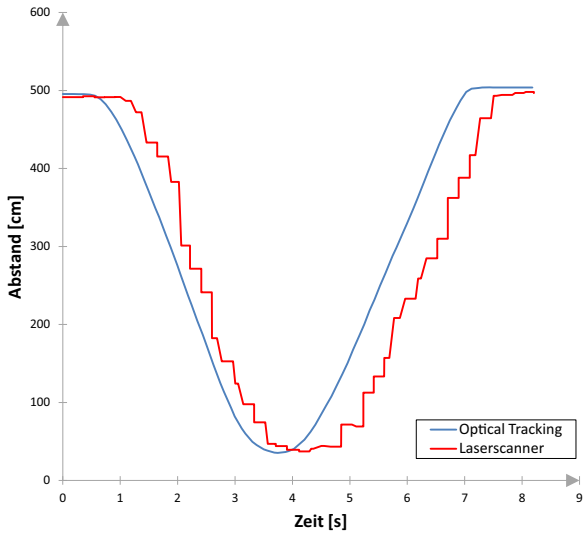
Bei schnellen Bewegungen sind die Unterschiede deutlicher zu erkennen. Die Abweichung der Messwerte der Filter mit zeitlicher Betrachtung ist sichtbar größer. Es kommt zum Teil zu einer Verzögerung des Signals von bis zu $\approx 500\,\text{ms}$, wohingegen die Abweichung in Abb.: 5.2a und 5.2d geringer ausfällt. Allgemein ist zwischen dem standardmäßigen Histogramm Filter in Abb.: 5.2a ohne Erweiterung und dem Histogramm Filter mit überlagernden Feldern in Abb.: 5.2d nur ein geringer Unterschied feststellbar. Der Filter mit überlagernden Feldern hat hier leichte Vorzügen im Hinblick auf die Verzögerungszeit.



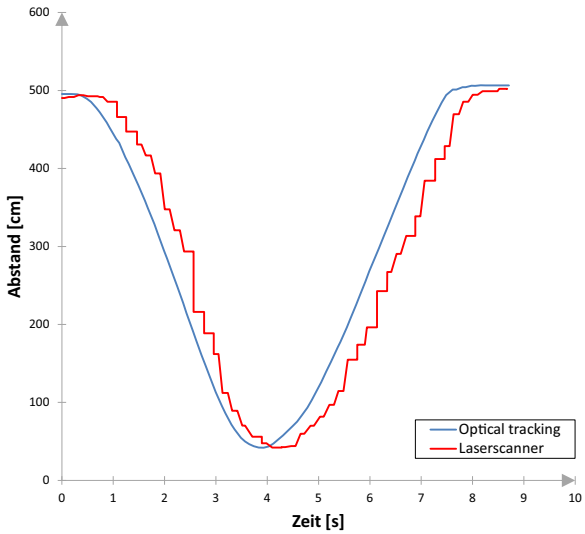
(a) Standard Filter



(b) Überlagernde Felder + zeitliche Betrachtung



(c) Zeitliche Betrachtung



(d) Überlagernde Felder

Abbildung 5.2: Vergleich des Histogramm Filter Verhaltens bei hoher Geschwindigkeit

5.2.3 Diskrepanz zu minimalem Sektorwert

Um zu testen, ob allgemein die Verzögerung der Histogramm Filterung geschuldet ist, oder ob die Verzögerung grundlegend auf die Abtastfrequenz des RPLidars zurückzuführen ist, wurde der gefilterten Wert mit dem der geringsten Abstandsmessung des Sektors verglichen (vgl. Abb.: 5.3). Verwendet wurde hierfür der Histogramm Filter mit überlagernden Feldern, da dieser in den Versuchen zuvor die besten Ergebnisse geliefert hat.

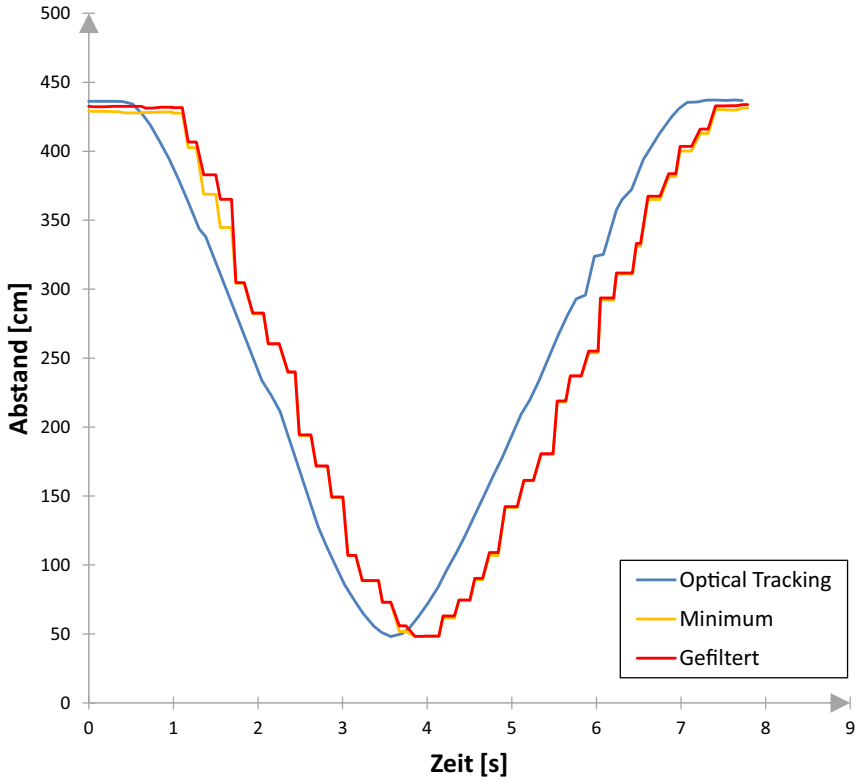


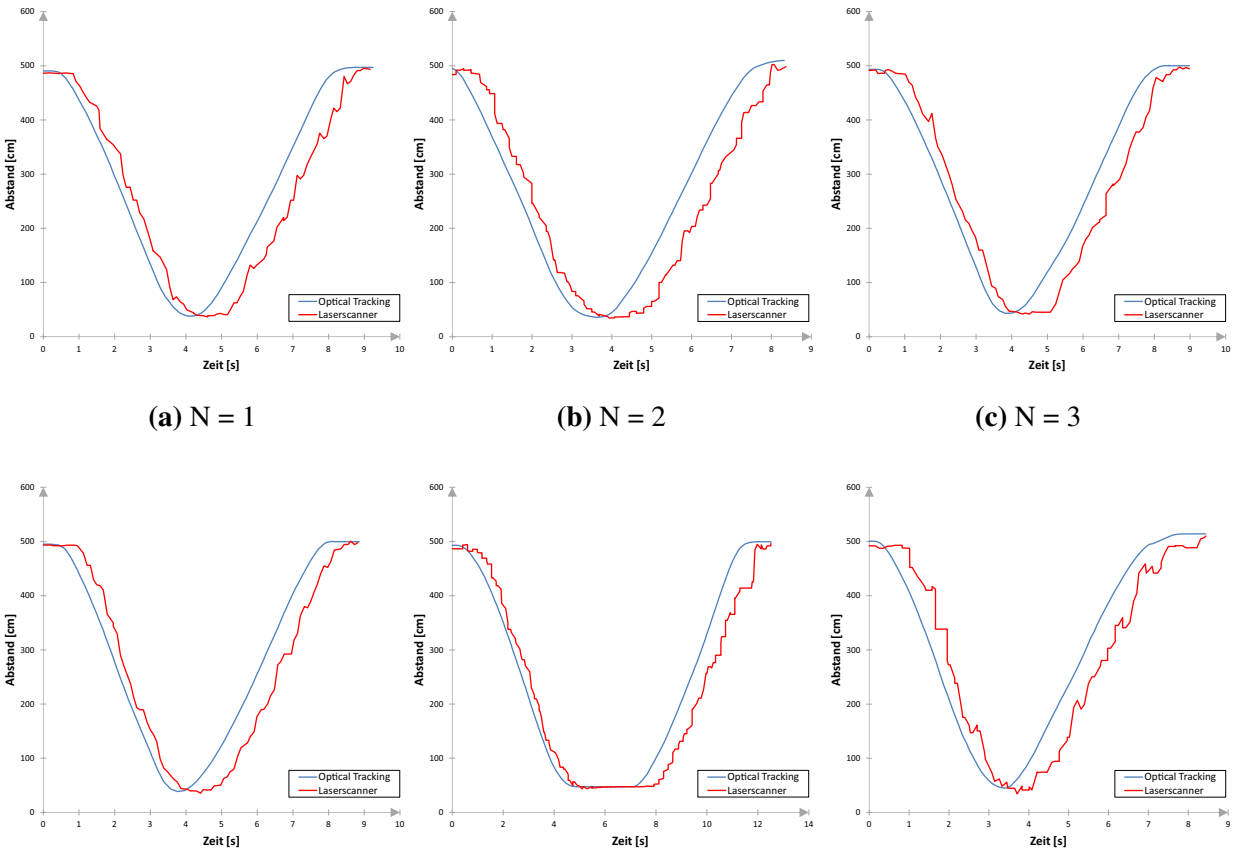
Abbildung 5.3: Vergleich des gefilterten Werts mit minimaler Abstandsmessung des Sektors

Es zeigt sich stellenweise ein etwas besseres Verhalten des minimalen Abstands. Zumindest bei Bewegung in Richtung des Hindernis ist die Verzögerung zum Teil geringer. Bei der Rückbewegung ist der minimale Abstandswert jedoch minimal schlechter.

Allgemein wirkt sich die Latenz zwischen dem wahren und dem gefilterten Abstand negativ auf die Kollisionsvermeidung aus, da Hindernisse verzögert erkannt werden. Die hier gemessene Diskrepanz zwischen dem gefilterten und minimalen Wert ist jedoch so gering, dass eine Hinderniserkennung mit deutlich geringerer Verzögerung nicht möglich. Die Verzögerung ist somit nicht der Filterung geschuldet.

5.2.4 Zwischenwertberechnung

Wie in Abb.: 5.4 zu erkennen, ist die Vorausberechnung der Zwischenwerte aus den vergangenen Werten problematisch. Je nach Anzahl N der in der Vergangenheit betrachteten Messungen zeigt sich ein sehr unterschiedliches Verhalten. Wird N zu klein gewählt (vgl. Abb.: 5.4a), so überholt die Voraussage zum Teil den darauf folgenden Wert. Werden andererseits zu viele in der Vergangenheit liegende Werte betrachtet, so läuft die Voraussage dem gemessenen Wert deutlich hinterher (vgl. Abb.: 5.4f). Dies erklärt sich dadurch, dass die in der Vergangenheit betrachteten Werte bei Bewegungen mit hoher Geschwindigkeit keine Relevanz mehr für die Voraussage haben. Der optimale Wert für N hängt somit von der aktuellen Geschwindigkeit ab und sollte nicht konstant gewählt werden. Es zeigt sich jedoch für $3 \leq N \leq 6$ ein akzeptables Verhalten.



(a) N = 1 **(b)** N = 2 **(c)** N = 3

(d) N = 6 **(e)** N = 9 **(f)** N = 12

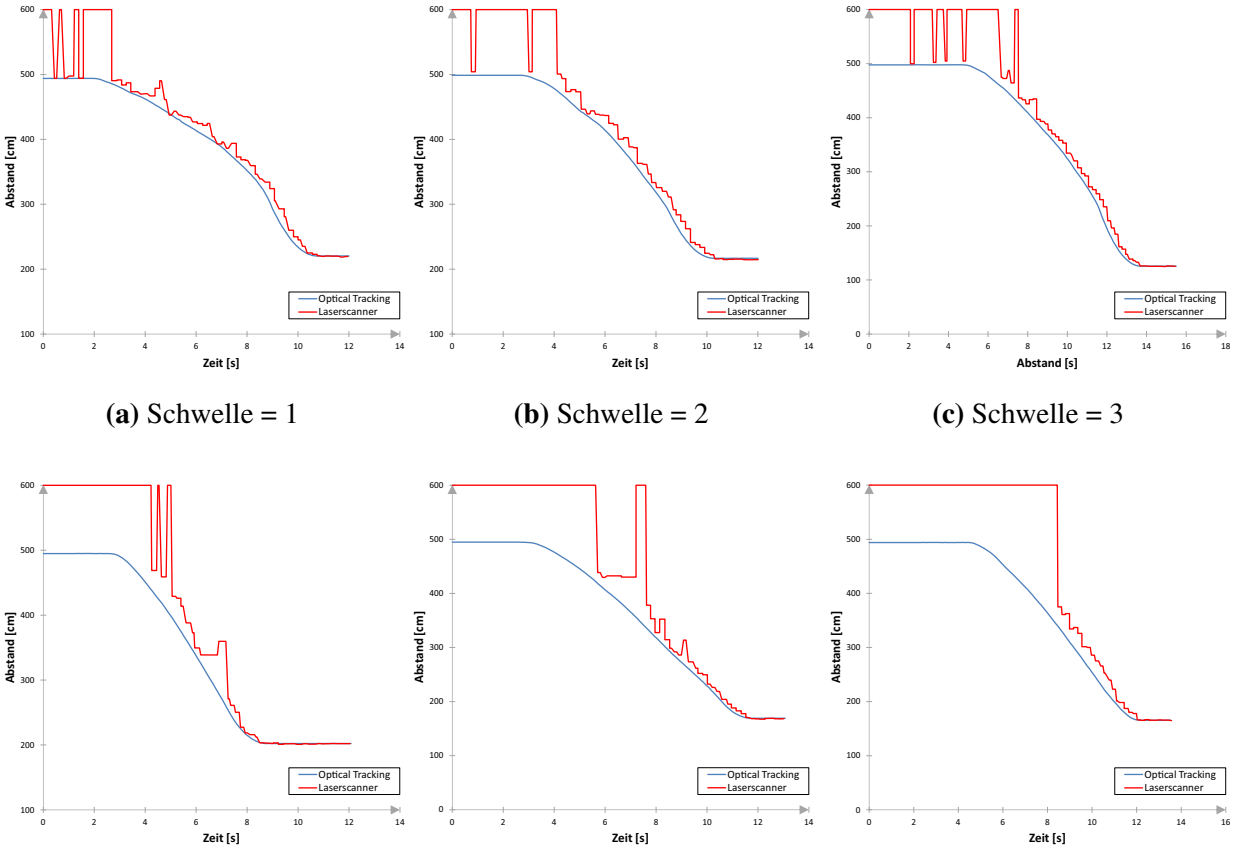
Abbildung 5.4: Auswirkung der Zwischenwertberechnung, ausgehend von der Anzahl der in der Vergangenheit betrachteten Werte N

5.2.5 Auswirkung der Parameter

Um die Auswirkung der Parameter des Histogramm Filters zu testen, wurde prinzipiell der gleiche Aufbau wie in Abschnitt 5.2.1 verwendet, mit dem Unterschied, dass das Hindernis nicht orthogonal zur Bewegungsachse, sondern schräg in einem Winkel von 45 ° platziert wurde, was die Anforderung an die Histogramm Filterung erhöht. Zudem wurde mit eine relativ nierdrigen Geschwindigkeit und nur in Richtung des Hindernis gemessen.

Schwellwert

Der Schwellwert Parameter beeinflusst maßgeblich das zeitliche Verhalten der gefilterten Abstandswerte. Wird der Parameter klein gewählt (Abb. 5.5a bis 5.5c) sind die Werte stark verrauscht und führen zu einem sichtbar sprunghaften Verhalten gerade bei großer Entfernung. Die Verzögerung des Signals ist jedoch deutlich geringer.



(a) Schwelle = 1 **(b)** Schwelle = 2 **(c)** Schwelle = 3

(d) Schwelle = 4 **(e)** Schwelle = 5 **(f)** Schwelle = 6

Abbildung 5.5: Auswirkung des Schwellwerts des Histogramm Filters auf die gefilterten Abstandswerte

Wählt man den Schwellwert größer (Abb. 5.5d bis 5.5f), wird das Rauschen minimiert und die Anzahl der Sprünge verringert sich. Das Signal ist nunmehr deutlich verzögert. Zusätzlich kann das sprunghafte Verhalten auch für kürzere Entfernungen auftreten (Abb. 5.5d und 5.5e). Allgemein lässt sich dieses Verhalten jedoch durch den Filter aus Abschnitt 3.5 vermeiden.
Je nach gewünschtem Verhalten muss der Schwellwert angepasst werden. Da im Hinblick auf Kollisionsvermeidung ein verzögerungsfreies Verhalten erforderlich ist, wurde ein Schwellwert von 3 aus Abb. 5.5c übernommen und zusammen mit dem bereits oben genannten Filter aus Abschnitt 3.5 kombiniert.

Feldbreite

Die Feldbreite beeinflusst ebenfalls stark das sprunghafte Verhalten der Abstandswerte. Allgemein steigt die Genauigkeit der Messung je kleiner die Feldbreite b gewählt wird. Jedoch ist dann nicht mehr gewährleistet, dass ausreichend Messwerte in den einzelnen Feldern vorkommen. Es zeigt sich in Abb. 5.6, dass das Hindernis für die Feldbreite von 5-15 cm nicht mehr erkannt wird. Gewählt wurde für diese Messung ein konstanter Schwellwert von 5.
Die Feldbreite ist somit an den Schwellwert gekoppelt. Es gilt: Je größer die Feldbreite, desto größer kann der Schwellwert gewählt werden, und umgekehrt. Allgemein hängen die gewählten Parameter vom verwendeten Laserscanner ab. Je höher die Auflösung des Laserscanners, desto höher kann der Schwellwert, und desto niedriger die Feldbreite gewählt werden, was die Genauigkeit des Verfahrens verbessert.

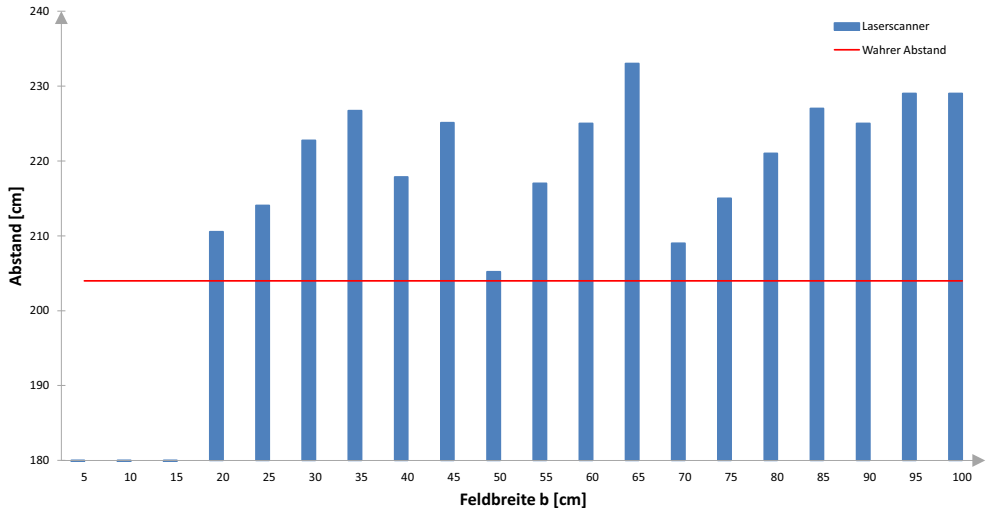


Abbildung 5.6: Auswirkung der Feldbreite b auf den gefilterten Messwert im Vergleich zum wahren Abstand

Zusätzlich zeigt sich in Abb. 5.6 die Problematik bei der Erkennung von nicht parallel zur Sektoröffnung ausgerichteten Hindernissen. Je nach Feldbreite variiert der Abstand deutlich. Es ist jedoch anzumerken, dass diese Abbildung keine qualitative Aussage über die Verwendung der Größe der Feldbreite zulässt. Je nach Orientierung und Entfernung des Hindernisses und der Wahl des Schwellwerts sind die Werte andersartig verteilt. Der nicht optimal gewählte Schwellwert von 5 (vgl. Abschnitt 5.2.5) verschlechtert zudem das Ergebnis der Messung.
Aufgrund des relativ gering gewählten Schwellwert Parameters von 3, wurde eine Feldbreite von $b = 20\,\text{cm}$ gewählt.

5.3 Kartierung

5.3.1 Testaufbau

Um das in ROS implementierte hector_slam zu evaluieren, wurde eine für den Laserscanner messbare Umgebungen aus Weichmatten aufgebaut (vgl. Abb. 5.7). Der Quadrokopter wurde dazu erneut auf einem beweglichen Untersatz montiert und so während der Bewegung auf konstanter Höhe gehalten. Es wurden ausschließlich 2D-Karten der Umgebung erstellt. Die Höheninformationen wurde nicht in die Kartenerstellung mit einbezogen, da hierzu aus Zeitgründen softwareseitig die Anbindung an die IMU und den Höheninformationssensor fehlte.
Ausgewertet und dargestellt wurden die Karten über das RVIZ Package [21], welches zeitgleich die Erstellung der Karte sowie die Anzeige der aktuellen Pose und Trajektorie ermöglicht.

5.3.2 Versuchsdurchführung

Es wurden Versuche mit zwei verschiedenen Testumgebungen durchgeführt, welche deutlich unterschiedliche Ergebnisse hervorbrachten.

Testumgebung 1

Testumgebung 1 (dargestellt in Abb. 5.8a) stellte sich für das vorhandene System als problematisch heraus. In Abb. 5.8b ist deutlich ein Orientierungs- sowie Lokalisierungsfehler zu erkennen, der im weiteren Verlauf der Kartenerstellung nicht mehr korrigiert werden kann, so dass die Abbildung der Karte unbrauchbar und deutlich verschoben ist. Es wird vermutet, dass das System

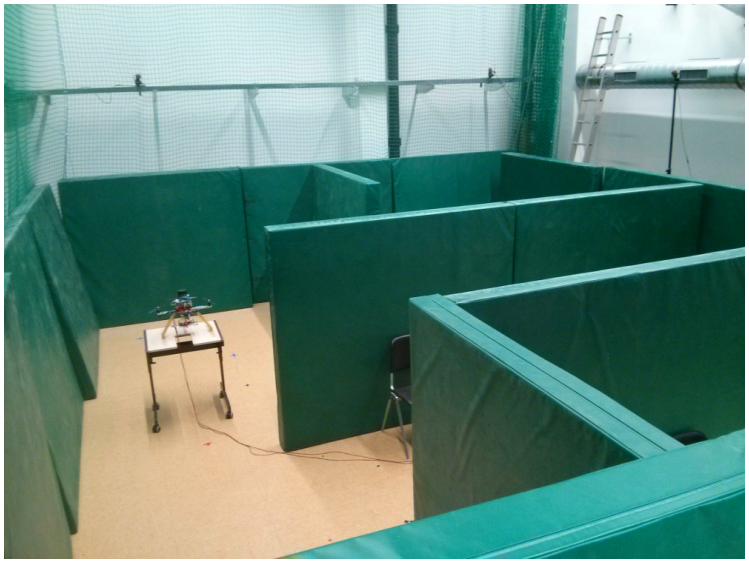
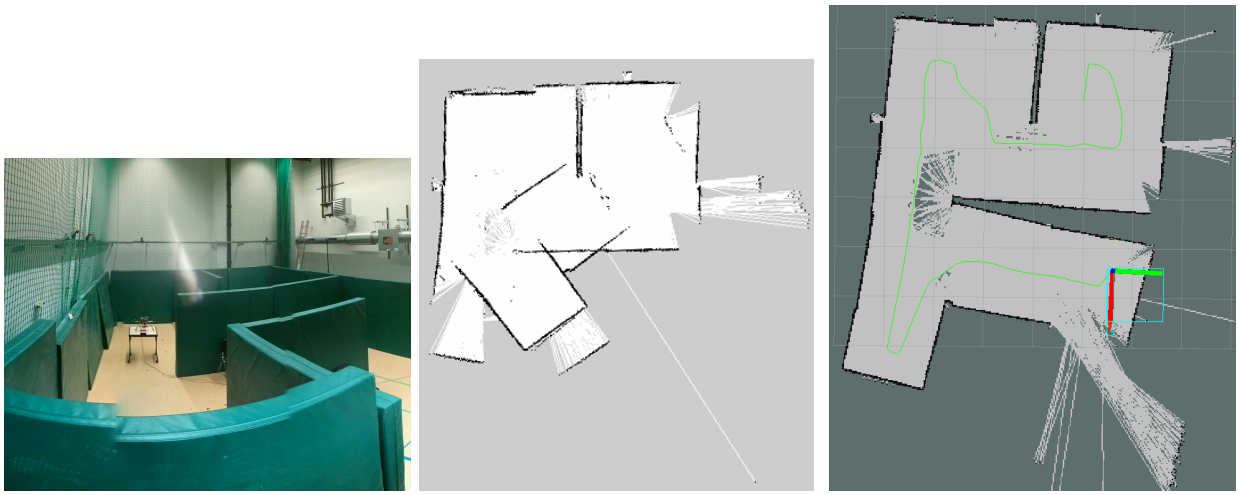


Abbildung 5.7: Testumgebung zur Kartenerstellung

bei zu schnellen Translations- wie Rotationsbewegungen versagt, was ggf. der niedrigen Abtastfrequenz des RPLidars geschuldet ist.

Abb. 5.8c zeigt ein besseres, jedoch nicht fehlerfreies Ergebnis. Auch hier weicht die Orientierung von der tatsächlichen Pose ab, so dass der untere Teil der Karte leicht schräg angefügt wurde. Vermutlich lassen die relativ großen Entfernungen der Matten zueinander eine Lokalisierung aufgrund einer unzureichenden Anzahl an Messpunkten nicht zu. Der Testaufbau wurde daher in Testumgebung 2 angepasst, indem auf durchgehende Wände verzichtet wurde.



(a) Testumgebung 1

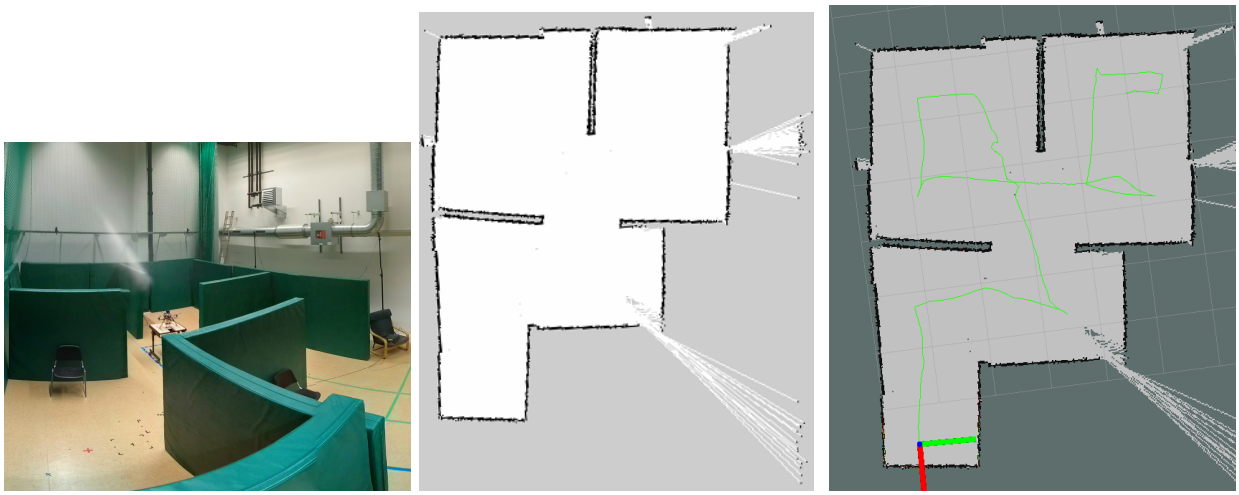
(b) Karte aus Versuch 1

(c) Karte aus Versuch 2, dargestellt in RVIZ mit Trajektorie in grün

Abbildung 5.8: Auswertung der Kartenerstellung für Testumgebung 1

Testumgebung 2

Die in Abb. 5.9a dargestellte Testumgebung 2 lieferte sichtbar bessere Ergebnisse. Es konnte eine relativ genaue Karte der Umgebung erstellt (vgl. Abb. 5.9b) und der Quadrokopter in dieser lokalisiert werden (vgl. Abb. 5.9c). Es ist jedoch anzumerken, dass auf Rotationsbewegung nach Möglichkeit verzichtet wurde, wodurch sich das Ergebnis ebenfalls verbesserte.



(a) Testumgebung 2 **(b)** Karte aus Versuch 3 **(c)** Karte aus Versuch 3, dargestellt in RVIZ mit Trajektorie in grün

Abbildung 5.9: Auswertung der Kartenerstellung für Testumgebung 2

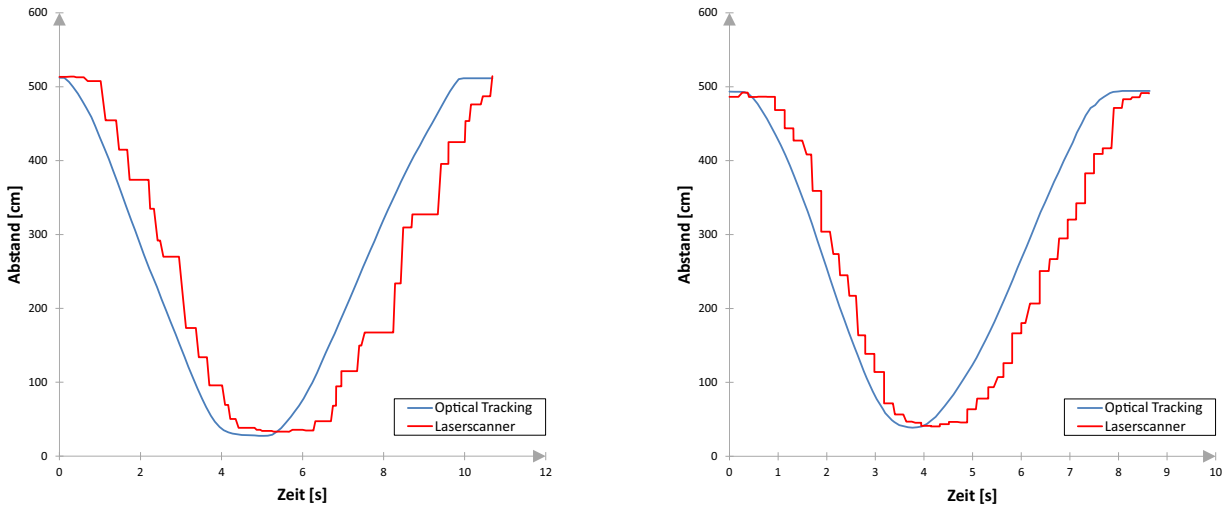
5.4 Kollisionsvermeidung

5.4.1 Anfängliche Problematik

Erste Experimente mit dem bereits vorhandenen Kollisionsvermeidungsreglers, auf Basis der gefilterten Laserscanner Abstandsmessungen waren nicht erfolgreich. Es zeigte sich ein enorm verzögertes Regelverhalten, wodurch eine verlässliche Kollisionsvermeidung nicht möglich war. Es stellte sich heraus, dass Synchronisationsproblem der UART-Schnittstelle zwischen Pico-ITX Board und Mikrocontroller dafür verantwortlich waren. Da das verwendete Betriebssystem Windows 7 auf dem Pico-ITX Board keine Echtzeitanforderungen erfüllt, konnten die Laserscannerdaten nicht in konstanten Intervallen, sondern nur mit einem Jitter von ca. $\pm 50\,\text{ms}$, verschickt werden. Je nach Scheduling auf Seiten des Betriebssystems wurden die Daten vereinzelnd deutlich verzögerter verschickt. Somit konnten die Daten nicht rechtzeitig vom Kollisionsvermeidungsregler auf Seite des Mikrocontrollers genutzt werden.

Verstärkt wurde das Synchronisationsproblem dadurch, dass das vorhandene Kommunikationsprotokoll nicht für diese Art der Anwendung ausgelegt war. Der Buffer der UART-Schnittstelle wurde anfänglich nur alle 100 ms ausgelesen, so dass sich mehrere Datenpakete im Buffer sammeln konnten. Beim Auslesen des Buffers wurden stets alle Pakete ausgelesen und die Daten aus den vorherigen Paketen somit mit den aktuellsten Daten überschrieben, so dass teilweise Abstandsdaten nicht vom Kollisionsvermeidungsregler genutzt werden konnten und übersprungen wurden.

Die verschiedenen Verhaltensweiseb sind in Abb. 5.10 gegenübergestellt. In Abb. 5.10a zeigt sich zum Teil eine Verzögerung zum tatsächlichen Wert von bis zu 1,5 Sekunden. Zudem ist das Intervall deutlich unregelmäßiger. In Abb. 5.10b kommen die Abstandswerte in relativ konstanten Intervallen an.



(a) Zufällige Verzögerung der Abstandswerte **(b)** Konstantes Intervall der Abstandswerte

Abbildung 5.10: Auswirkungen der Kommunikationsverzögerung

Der Buffer der UART-Schnittstelle wurde hierzu alle 10 ms ausgelesen, was das Jitter deutlich verringert, jedoch nicht gänzlich verhindern konnte.

Allgemein erwartet der Kollisionsvermeidungsregler neue Abstandswerte in einem Intervall von 90 ms. Für den Fall, dass dennoch eine Verzögerung eintritt, wurde der Regler zusätzlich so angepasst, dass die Regelung nur ausgeführt wird, wenn tatsächlich neue Abstandswerte vorliegen.

5.4.2 Versuchsdurchführung

Die Auswertung des Kollisionsvermeidungsreglers erfolgte während des Flugs. Hierzu wurde der Quadrokopter in Richtung einer Ecke geflogen, so dass das Regelverhalten für den Roll- und Nickwinkel evaluiert werden konnte. Die Filterung der Abstände (Vorne, Links) erfolgt auf Seite des Kollisionsvermeidungsregler und wird aus den gefilterten Sektoren des Histogramm Filters fusioniert. Als Parameter des Histogramm Filters sind die bestimmten Werte aus Abschnitt 5.2.5 übernommen worden.

5.4.3 Auswertung des Regelverhaltens

In Diagramm Abb. 5.11 wird das Verhalten des Kollisionsvermeidungsreglers veranschaulicht, indem die Abstände der Hinderniserkennung mit dem Soll-Abstand des Reglers verglichen werden. Allgemein ist eine Regelung um den Soll-Abstand zu erkennen. Das Regelverhalten ist jedoch nicht konstant und wird durch Oszillation negativ beeinflusst. Die Abstände schwanken hierbei in einer Größenordnung von $\pm 40\,\text{cm}$ um den Sollwert. Zusätzlich zeigt sich ein Verhalten der Abstandswerte, dass während vorheriger Tests am Boden nicht beobachtet werden konnte. Es sind mehrfach Ausreißer aus den Messwerten zu erkennen, die allerdings nicht als Ursache für das oszillierende Regelverhalten ausgemacht werden können. Durch Optimierung der Regelparameter kann ein besseres Regelverhalten erreicht werden (aus Zeitgründen nicht durchgeführt).

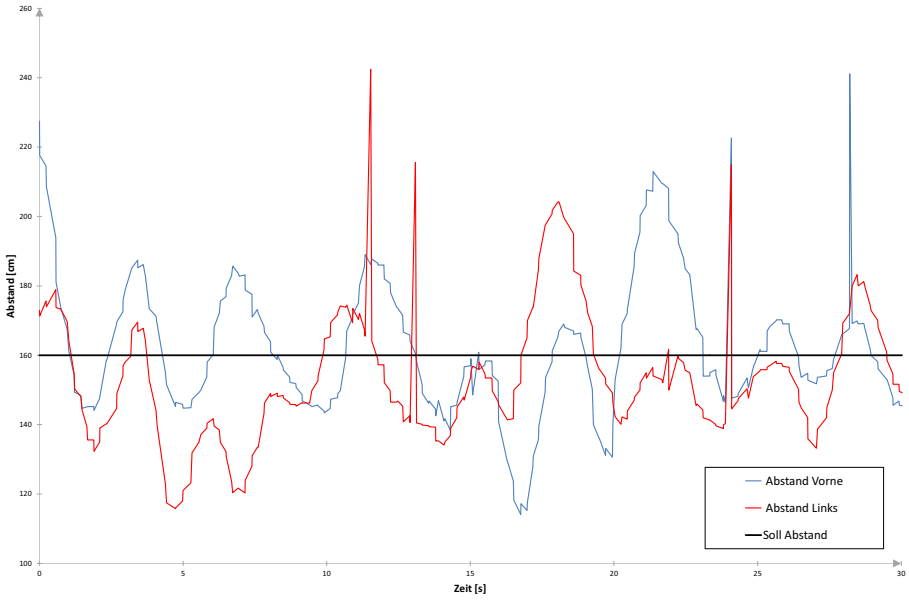


Abbildung 5.11: Diagramm des Regelverhatens der Kollisionsvermeidung

6 Diskussion und Ausblick

Die Evaluierung hat gezeigt, dass der RPLidar prinzipiell als Sensor zur Kollisionsvermeidung und das Konzept des Histogramm Filters zur Hinderniserkennung genutzt werden kann. Das System in seiner jetzigen Form bietet allerdings noch Raum für Verbesserungen.

Zunächst könnte die Abtastfrequenz des Laserscanners auf seine maximale Abtastfrequenz von 10 Hz erhöht werden, um dadurch die Verzögerung des Regelverhaltens weiter zu minimieren. Hierzu muss lediglich der Motor des RPLidars über ein zusätzliches PWM-Signal angesteuert werden und auf die PWM Anbindung der UART-Bridge verzichtet werden. Die Parameter des Histogramm Filters müssen daraufhin erneut angepasst werden.
Allgemein bietet es sich an, die Signalverarbeitung der Laserscannerdaten auf einen Mikrocontroller mit integriertem PWM-Ausgang auszulagern, welcher als Echtzeitsystem die Problematik des Jitters bei der Kommunikation deutlich verringern sollte. Zusätzlich könnte somit auf das Pico-ITX Board verzichtet werden und ein nicht unerheblicher Teil des Gewichts des UAVs eingespart werden. Die Synchronisationsprobleme könnten auch über eine Erweiterung des Kommunikationsprotokolls behoben werden, indem bei der Messung der Laserscannerdaten ein Zeitstempel aufgenommen wird und dem verschickten Datenpaket angehängt wird, welches auf der Seite des Kollisionsvermeidungsreglers in die Bestimmung aller zeitabhängigen Größen mit einfließen könnte.

Zusätzlich sollte durch Optimierung und Anpassung der Parameter des Kollisionsvermeidungsreglers (an die Abtastfrequenz des RPLidar) ein besseres Regelverhalten erreicht werden können. Des Weiteren wäre ein zusätzliches redundantes System zur Hinderniserkennung wünschenswert, da der RPLidar als 2D-Laserscanner nur die obere Ebene des UAVs abdecken kann. So können Objekte, die unterhalb dieser Ebene liegen, nicht erkannt werden, was zu Kollisionen führen kann.

Außerdem wurde gezeigt, dass der RPLidar als Laserscanner für SLAM Konzepte genutzt werden kann. Das System eignet sich in der aktuellen Form jedoch nicht zum Einsatz während des Flugs auf einem UAV. Hierzu müssten die Höhen- sowie die Lagesensoren des UAVs mit in den SLAM Algorithmus einfließen. Hierbei wäre mit den verwendeten ROS Packages sogar eine 3D-SLAM Lösung denkbar. Zusätzlich könnte der bereits vorhandene und integrierte Optical Flow Sensor des AQopterI8 Projekts verwendet werden, um eine verbesserte Positionsbestimmung und dadurch robustere SLAM-Lösung zu ermöglichen.

7 Literaturverzeichnis

[1] ACHTELIK, Markus ; BACHRACH, Abraham ; HE, Ruijie ; PRENTICE, Samuel ; ROY, Nicholas: Autonomous navigation and exploration of a quadrotor helicopter in GPS-denied indoor environments. In: *First Symposium on Indoor Flight*, 2009

[2] ALVAREZ, H ; PAZ, LM ; STURM, J ; CREMERS, D: Collision avoidance for quadrotors with a monocular camera. In: *Proc. of the Int. Symp. on Experimental Robotics (ISER)*, 2014

[3] BECKER, Marcelo ; SAMPAIO, Rafael Coronel B. ; BOUABDALLAH, Samir ; PERROT, Vincent d. ; SIEGWART, Roland: In-flight collision avoidance controller based only on OS4 embedded sensors. In: *Journal of the Brazilian Society of Mechanical Sciences and Engineering* 34 (2012), 09, S. 294 – 307. – ISSN 1678-5878

[4] BERKOVIC, Garry ; SHAFIR, Ehud: Optical methods for distance and displacement measurements. In: *Adv. Opt. Photon.* 4 (2012), Dec, Nr. 4, S. 441–471. – URL `http://aop.osa.org/abstract.cfm?URI=aop-4-4-441`

[5] BESSIERE, Pierre ; LAUGIER, Christian ; SIEGWART, Roland: *Probabilistic reasoning and decision making in sensory-motor systems*. Bd. 46. Springer Science & Business Media, 2008

[6] BIBER, Peter ; STRASSER, Wolfgang: The normal distributions transform: A new approach to laser scan matching. In: *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on* Bd. 3 IEEE (Veranst.), 2003, S. 2743–2748

[7] BLÖSCH, Michael ; WEISS, Stephan ; SCARAMUZZA, Davide ; SIEGWART, Roland: Vision based MAV navigation in unknown and unstructured environments. In: *Robotics and automation (ICRA), 2010 IEEE international conference on* IEEE (Veranst.), 2010, S. 21–28

[8] BOUABDALLAH, Samir: *Design and control of quadrotors with application to autonomous flying*, École Polytechnique federale de Lausanne, Dissertation, 2007

[9] BOUABDALLAH, Samir ; BECKER, Marcelo ; PERROT, Vincent de ; SIEGWART, Roland Y. ; SIEGWART, Roland Y. ; SIEGWART, Roland Y.: *Toward obstacle avoidance on quadrotors*. Citeseer, 2007

[10] CHETVERIKOV, D. ; BESL, P.J. ; SVIRKO, D. ; STEPANOV, D. ; KRSEK, P.: The Trimmed Iterative Closest Point algorithm 16th International Conference on Pattern Recognition (Veranst.), IEEE, 2002, S. 545–548

[11] CLICKMOX: *Minefly R-SCAN3D*. `http://clickmox.com/products/minefly/`. – zuletzt abgerufen am 16.09.2015

[12] DJI: *DJI Guidance System*. `https://dev.dji.com/en/products/devices/guidance`. – zuletzt abgerufen am 17.08.2015

[13] DOERT, Colin: *Simultane Lokalisierung und Kartenerstellung eines autonomen Roboters in einer dynamischen Umgebung bei Anwesenheit von Personen*, Technische Universität Dortmund, Diplomarbeit, Januar 2009

[14] ELFES, Alberto: *Occupancy Grids: A Probabilistic Framework for Robot Perception and Navigation*. Pittsburgh, PA, USA, Dissertation, 1989

[15] ENDRES, Felix ; HESS, Juergen ; ENGELHARD, Nikolas: *rgbdslam ROS-Package*. `http://wiki.ros.org/rgbdslam`. – zuletzt abgerufen am 11.08.2015

[16] FOX, D. ; BURGARD, W. ; DELLAERT, F. ; THRUN, S.: Monte Carlo Localization: Efficient Position Estimation for Mobile Robots. In: *Proc. of the National Conference on Artificial Intelligence*, 1999

[17] GAGEIK, Nils ; BENZ, Paul ; MONTENEGRO, Sergio: Obstacle Detection and Collision Avoidance for a UAV with Complementary Low-Cost Sensors. (2015)

[18] GAGEIK, Nils ; STROHMEIER, Michael ; MONTENEGRO, Sergio: An autonomous UAV with an optical flow sensor for positioning and navigation. In: *International Journal of Advanced Robotic Systems* 10 (2013), S. 341

[19] GRISETTI, Giorgio ; STACHNISS, Cyrill ; BURGARD, Wolfram: Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling. In: *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on* IEEE (Veranst.), 2005, S. 2432–2437

[20] GRZONKA, Slawomir ; GRISETTI, Giorgio ; BURGARD, Wolfram: A Fully Autonomous Indoor Quadrotor. In: *Robotics, IEEE Transactions on* 28 (2012), Nr. 1, S. 90–100

[21] HERSHBERGER, Dave ; GOSSOW, David ; FAUST, Josh: *rviz ROS-Package*. `http://wiki.ros.org/rviz`. – zuletzt abgerufen am 13.09.2015

[22] HOKUYO: *UTM-30LN*. `https://www.hokuyo-aut.jp/02sensor/07scanner/utm_30ln.html`. – zuletzt abgerufen am 05.08.2015

[23] JACKSON, Kimberly F.: *Development and evaluation of a collision avoidance system for supervisory control of a micro aerial vehicle*, Massachusetts Institute of Technology, Dissertation, 2012

[24] KOHLBRECHER, S. ; MEYER, J. ; STRYK, O. von ; KLINGAUF, U.: A Flexible and Scalable SLAM System with Full 3D Motion Estimation. In: *Proc. IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)* IEEE (Veranst.), November 2011

[25] MCKAY, Neil D. ; BESL, P.J.: A method for registration of 3-D shapes. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14 (1992), 2, Nr. 2, S. 239–256

[26] MEYER, Johannes: *hector_slam ROS-Package*. `http://wiki.ros.org/hector_slam`. – zuletzt abgerufen am 11.08.2015

[27] OPTITRACK: *Flex 3*. `http://www.optitrack.com/products/flex-3/`. – zuletzt abgerufen am 12.09.2015

[28] OPTITRACK: *Motive:Tracker*. `http://www.optitrack.com/products/motive/tracker/indepth.html`. – zuletzt abgerufen am 16.09.2015

[29] PARK, Jongho ; KIM, Youdan: Stereo vision based collision avoidance of quadrotor UAV. In: *Control, Automation and Systems (ICCAS), 2012 12th International Conference on*, Oct 2012, S. 173–178

[30] PTEXT: *Bluetechnix bringt ToF 3D-Kamera mit 160 fps auf den Markt*. http://www.ptext.at/nachrichten/bluetechnix-bringt-tof-3d-kamera-160-fps-markt-528977. – zuletzt abgerufen am 31.08.2015

[31] PULSEDLIGHT: *LIDAR-Lite v1*. http://lidarlite.com/docs/v2/pdf/LIDAR-Lite-v1-docs.pdf. – zuletzt abgerufen am 16.09.2015

[32] QT: *C++-Klassenbibliothek für plattformübergreifende Programmierung grafischer Benutzeroberflächen*. http://www.qt.io/. – zuletzt abgerufen am 29.08.2015

[33] QUIGLEY, Morgan ; CONLEY, Ken ; GERKEY, Brian P. ; FAUST, Josh ; FOOTE, Tully ; LEIBS, Jeremy ; WHEELER, Rob ; NG, Andrew Y.: ROS: an open-source Robot Operating System. In: *ICRA Workshop on Open Source Software*, 2009

[34] REMONDINO, Fabio ; BARAZZETTI, L ; NEX, Francesco ; SCAIONI, Marco ; SARAZZI, Daniele: UAV photogrammetry for mapping and 3d modeling–current status and future perspectives. In: *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 38 (2011), Nr. 1, S. C22

[35] RN-WISSEN: *Vergleichstabelle akustischer Sensoren*. http://rn-wissen.de/wiki/index.php/Sensorarten#Vergleichstabelle. – zuletzt abgerufen am 31.08.2015

[36] ROBERTS, James F. ; STIRLING, Timothy ; ZUFFEREY, Jean-Christophe ; FLOREANO, Dario: Quadrotor using minimal sensing for autonomous indoor flight. In: *European Micro Air Vehicle Conference and Flight Competition (EMAV2007)*, 2007

[37] ROBOPEAK: *2D lidar brief datasheet*. http://www.dfrobot.com/image/data/DFR0315/robopeak_2d_lidar_brief_en.pdf. – zuletzt abgerufen am 29.08.2015

[38] ROBOPEAK: *RPLidar Laserscanner*. http://www.robopeak.com/blog/?cat=5. – zuletzt abgerufen am 11.08.2015

[39] ROBOPEAK: *rplidar ROS-Package*. http://wiki.ros.org/rplidar. – zuletzt abgerufen am 12.09.2015

[40] SABATINI, Roberto ; GARDI, Alessandro ; RICHARDSON, Mark A.: LIDAR obstacle warning and avoidance system for unmanned aircraft. In: *International Journal of Mechanical, Aerospace, Industrial and Mechatronics Engineering* 8 (2014), Nr. 4, S. 702–713

[41] SHARP: *GP2Y0A02YK0F*. `https://www.sparkfun.com/datasheets/Sensors/Infrared/gp2y0a02yk_e.pdf`. – zuletzt abgerufen am 03.08.2015

[42] SHEN, Shaojie ; MICHAEL, Nathan ; KUMAR, Vijay: Autonomous multi-floor indoor navigation with a computationally constrained MAV. In: *Robotics and automation (ICRA), 2011 IEEE international conference on* IEEE (Veranst.), 2011, S. 20–25

[43] WINKVIST, Stefan: *Low computational SLAM for an autonomous indoor aerial inspection vehicle*, University of Warwick, Dissertation, 2013