3D Navigation anhand eines externen Umweltmodels

Studienarbeit

für die Prüfung zum

Bachelor of Science

des Studienganges Informationstechnik

an der

Dualen Hochschule Baden-Württemberg Karlsruhe

von

Maximilian Burr, Johannes Vater, Fabian Droll

Abgabedatum 22. Mai 2023

Matrikelnummern	Burr: 6844761 Vater: 2713858 Droll: 7368743
Kurs	TINF20B3
Gutachter der Studienarbeit	Prof. Dr. Marcus Strand

Erklärung

Wir versicheren hiermit, dass wir unsere Studienarbeit mit dem Thema: „3D Navigation anhand eines externen Umweltmodels“ selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt haben. Wir versichere zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.

Ort Datum Unterschrift

Sperrvermerk

Der Inhalt dieser Arbeit darf weder als Ganzes noch in Auszügen Personen auerhalb des Prüfungsprozesses und des Evaluationsverfahrens zugänglich gemacht werden, sofern keine anders lautende Genehmigung der Ausbildungsstätte vorliegt.

Zusammenfassung

Kurzfassung

3D Navigation anhand eines externen Umweltmodels

Zusammenfassung

Abstract

3D navigation using an external environmental model

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Liste der Algorithmen

Formelverzeichnis

Abkürzungsverzeichnis

GPS Global Positioning System 8
ROS Robot Operating System 11
EKF Extended Kalman Filter 18

Kapitel 1

Einleitung

GPS Signale sucht man in Räumen vergeblich. Das GPS System wurde nicht dafür konzipiert in Räumen zu funktionieren. In dieser Studienarbeit geht es darum zu ermitteln auf welche Art und Weise eine Drohne in einem 3D Modell ohne GPS navigieren kann. 3D Navigation ist ein wichtiger Aspekt bei der Entwicklung von Systemen, die in einer virtuellen oder realen Umgebung agieren. Die Nutzung eines externen Umweltmodells bietet hierbei viele Vorteile, da es die Navigation vereinfacht und gleichzeitig die Genauigkeit verbessert. Ein externes Umweltmodell kann dabei in Form einer digitalen Karte oder eines virtuellen Raums im Computer vorliegen. In diesem Zusammenhang wird die 3D Navigation anhand eines externen Umweltmodells untersucht und die Vorteile, die sich daraus ergeben, werden dargestellt.

1.1 Ausgangssituation

Prinzipiell hat eine Drohne verschiedene Möglichkeiten eine Positionsbestimmung durchzuführen. Eine Drohne kann mithilfe von Kameras oder inertialen Sensoren wie Gyroskopen und Beschleunigungssensoren die Position relativ zu einer Startposition bestimmen, oder Global Positioning System (GPS) verwenden um die Position mithilfe von Satelliten möglichst genau zu bestimmen. Schnell hat man festgestellt, dass die genaue Positionsbestimmung über GPS in Gebäuden nicht funktioniert, da die Signale der Satelliten zu schwach sind. Außerdem bietet GPS ohne großen Aufwand keine Möglichkeiten eine genaue Bestimmung von unter zwei Metern zu gewährleisten. Um eine Navigation ohne GPS zu tätigen muss zudem ein 3D Modell der Umgebung der Drohne übermittelt werden. Zu Beginn muss das 3D Modell in dem die Drohne sich bewegt, noch erstellt werden.

1.2 Motivation

In der heutigen Zeit wird das Navigieren in komplexen Umgebungen immer wichtiger und herausfordernder. Egal, ob in der Luftfahrt, bei der Navigation von Drohnen oder autonomen Fahrzeugen - die Fähigkeit, präzise und schnell in dreidimensionalen Räumen zu navigieren, ist unverzichtbar geworden.

Um diese Herausforderungen zu meistern kann ein externes Umweltmodell verwendet werden. Es ermöglicht eine präzisere und realitätsnähere Darstellung der Umgebung und kann somit eine genauere Navigation ermöglichen. Durch die Verwendung von 3D-Modellen kann man beispielsweise Hindernisse in der Umgebung besser identifizieren und umfahren, ohne dass es zu Kollisionen kommt. Auch die Planung von Routen und die Optimierung von Fahrzeugbewegungen können durch ein externes Umweltmodel verbessert werden. Zudem wird durch das Verwenden eines 3D-Modells die Navigation innerhalb schwieriger Umgebungen erleichtern. Der Vorteil hiervon ist, dass man nun gefährliche Gebiete deutlich einfacher erkunden kann. Aber auch in Bereichen wie der Architektur, der Planung von Fabrikanlagen oder der Simulation von Gefahrensituationen kann die Verwendung von 3D-Modellen für eine effiziente Navigation von großer Bedeutung sein.

Insgesamt bietet die 3D-Navigation mit Hilfe von externen Umweltmodellen also zahlreiche Vorteile und kann in vielen verschiedenen Anwendungsbereichen von Nutzen sein. Sei es in der Logistik, der Aufklährung oder der Suche nach z.B. Personen in Gefahrenumgebungen.

Diese Arbeit dient dazu Grundlagen zur Navigation mithilfe eines 3D Modells der Umgebung in einem dreidimensionalen Raum zu erarbeiten. Diese Arbeit kann danach in weiteren Arbeiten verwendet werden, um die Navigation in erweiterten Teilbereichen einzusetzen und sie um weitere Funktionen zu ergänzen.

Kapitel 2

Problemstellung

Die Navigation anhand eines 3D Modell bringt gewisse theoretische Probleme mit sich.

2.1 Erwartete Probleme

In der Vorbereitung der Arbeit wurden bereits Probleme erkannt, die auf jeden Fall gelöst oder Alternativen dazu gefunden werden müssen, um die erfolgreiche Umsetzung des Projekts zu ermöglichen. Im Folgendem werden diese Probleme erläutert.

- Drift
-

2.1.1 Lokalisierung der Drohne

Von vorneherein war klar, dass die Lokalisierung der Drohne nicht über GPS getätigt werden kann. Für die Aufgaben, welche die Drohne übernehmen soll, ist eine bestmögliche Positionserkennung vonnöten. Die Genauigkeit mittels GPS unter den technischen Bedingungen der Drohne beträgt nur zwei Meter. Das weitaus größer Problem mit GPS ist allerdings, dass man innerhalb von Gebäude keine Verbindung zu den Satelliten herstellen kann. Aufgrund dieser Probleme ist eine Verwendung von GPS zur Lokalisierung der Drohne nicht möglich. Dazu mussten alternative Wege gefunden werden, um eine genaue Lokalisierung ohne GPS in Gebäuden zu ermöglichen. Folgende Theoretischen Lösungsansätze konnten dabei verfolgt werden.

Die verwendete COEX Clover Drohne (siehe Kapitel 3.3) verfügt standardmäßig über die in 2.1.1 aufgeführten Sensoren.

- Beschleunigungssensoren
- Gyroskope
- Magnetometer

- Videokamera
- GPS
- Rangefinder

2.1.2 Erfassung des 3D Modells

Das 3D Modell kann entweder von Hand erstellt werden oder mit einem Scanner erfasst werden. Es musste eine Lösung gefunden werden die 3D Modelle zu erstellen und die Drohne innerhalb dieser Modelle abzubilden, zu lokalisieren/tracken und zu navigieren.

2.1.3 Nutzung der Software

Die Clover Drohne wird standardmäßig mit der PX4 Software betrieben, diese Software kann über eine Schnittstelle mit Robot Operating System (ROS) verbunden werden. ROS wird für die Entwicklung der Navigationssytems, während PX4 die Steuerung der Drohnenhardware übernimmt. Zum Zeitpunkt der Studienarbeit ist ROS Noetic Ninjemys die empfohlene Version der Entwickler mit den ausgereiftesten Paketen. Für die Entwicklung mit ROS muss eine entsprechende Entwicklungsumgebung installiert werden, die auch eine Simulation der Drohne ermöglicht. Auf der Clover Drohne ist ein Raspberry Pi 4 installiert, der die Ausführung der ROS Programme ermöglicht, die Software muss jedoch ressourcenschonend sein, um mit den restriktiven Ressourcen des Raspberry Pis umgehen zu können.

2.2 Gewünschtes Ergebnis

Kapitel 3

Theoretische Grundlagen

3.1 Sensoren

3.1.1 Magnetometer

Bei Magnetometern handelt es sich um Sensoren, die das Magnetfeld im Umfeld des Sensors messen können. Diese können sowohl in der Luft, als auch auf dem Boden eingesetzt werden. Meistens werden für diese Messungen Hall Sensoren oder Induktivitäts-Sensoren verwendet. Die Sensoren sind in der Lage, das Magnetfeld in drei Dimensionen zu messen und darzustellen.

Die Messung des Magnetfeldes erfolgt dabei in der Regel über einen Halbleiter, der durch das Magnetfeld beeinflusst wird. und einer Elektronik, die das Signal aufbereitet und ausgibt. Magnetometer können jedoch nur die Richtung des Magnetfeldes messen, jedoch nicht die Stärke des Magnetfeldes. Magnetometer werden verwendet um die Ausrichtung von Geräten in bestimmten Koordinatensystemen zu bestimmen. Sie können für diese Ausrichtungsbestimmung auch in der Luft verwendet werden.

3.1.2 Kamera

Bild Kamera

Infrarot Kamera

3.1.3 Abstandssensoren

3.2 ROS - Robot Operating System

ROS ist eine Open-Source-Plattform, die speziell für die Entwicklung von Robotersoftware entwickelt wurde. Es bietet eine Reihe von Bibliotheken, Tools und Frameworks, die es Entwicklern ermöglichen, komplexe Robotikanwendungen zu erstellen und zu betreiben. ROS wurde von Willow Garage entwickelt und ist heute ein weit verbreitetes Framework in der Robotik-Community.

ROS besteht aus verschiedenen Modulen, die es ermöglichen, Roboterhard- und software zu abstrahieren und zu standardisieren. Die Plattform bietet eine Vielzahl von Werkzeugen für die Entwicklung von Robotik-Software, einschließlich Visualisierungstools, Datenverarbeitungs- und Analysetools sowie eine umfassende Dokumentation.

ROS ist so konzipiert, dass es auf einer Vielzahl von Betriebssystemen und Hardwarearchitekturen laufen kann und bietet Unterstützung für eine breite Palette von Robotern und Sensoren. Es ist auch bekannt für seine Fähigkeit zur Zusammenarbeit zwischen verschiedenen Robotern, die miteinander kommunizieren und Aufgaben gemeinsam erledigen können.

Dank seiner leistungsstarken Funktionen und Flexibilität ist ROS zu einem der wichtigsten Frameworks für die Robotik-Entwicklung geworden und wird in vielen Anwendungen eingesetzt, von industriellen Robotern bis hin zu autonomen Fahrzeugen.

3.2.1 Nodes

In ROS werden Funktionen und Prozesse durch sogenannte Nodes realisiert. Eine Node ist eine ausführbare Einheit, die in einem ROS-System arbeitet und über eine eindeutige Identifikation verfügt. Jede Node hat eine spezifische Aufgabe, wie beispielsweise das Sammeln von Sensordaten, die Ausführung einer spezifischen Berechnung oder das Steuern eines Aktors.

Nodes können miteinander kommunizieren, indem sie Nachrichten senden und empfangen. Nachrichten sind definierte Datenstrukturen, die Informationen zwischen Nodes transportieren. Nodes können auch Services anbieten oder anfordern, um eine bestimmte Aktion auszuführen.

Eine wichtige Funktion von Nodes ist ihre Fähigkeit zur Verteilung. In ROS können Nodes auf verschiedenen Hosts oder in verschiedenen Prozessen ausgeführt werden. Dadurch können komplexe ROS-Systeme erstellt werden, die aus vielen miteinander verbundenen Nodes bestehen.

Nodes können auch in einer ROS-Graphenstruktur organisiert werden. Diese Struktur zeigt die Abhängigkeiten zwischen Nodes und die Art der Kommunikation zwischen ihnen an. Die ROS-Graphenstruktur kann mit Werkzeugen wie rqt_graph visualisiert werden, um eine bessere Übersicht über das System zu erhalten.

Die Verwendung von Nodes in ROS ermöglicht eine hohe Flexibilität und Modularität bei der Entwicklung von Robotik-Anwendungen. Entwickler können einzelne Nodes erstellen, testen und optimieren, bevor sie sie in einem größeren System einsetzen. Darüber hinaus können Nodes wiederverwendet werden, um ähnliche Funktionen in verschiedenen Anwendungen auszuführen.

Insgesamt sind Nodes eine zentrale Komponente von ROS und ermöglichen es Entwicklern, komplexe Roboteranwendungen mit einer hohen Flexibilität und Modularität zu erstellen.

3.2.2 Topics

In ROS werden Daten zwischen Nodes durch sogenannte Topics ausgetauscht. Ein Topic ist eine benannte Kommunikationsleitung, über die Nodes Nachrichten senden und empfangen können. Topics ermöglichen die einfache und flexible Kommunikation zwischen Nodes, ohne dass die Nodes über die genaue Identität des Empfängers Bescheid wissen müssen.

Ein Topic hat einen bestimmten Datentyp, der definiert, welche Art von Daten zwischen Nodes ausgetauscht werden können. Es können beispielsweise Sensordaten wie Bilder oder Entfernungsmessungen, oder Steuerbefehle für Aktoren wie Motoren oder Greifer übertragen werden.

Nodes können sich auf ein Topic abonnieren, um die Nachrichten, die auf diesem Topic veröffentlicht werden, zu empfangen. Jedes Mal, wenn eine Nachricht auf einem Topic veröffentlicht wird, wird sie an alle Nodes weitergeleitet, die auf dieses Topic abonniert sind.

Topics können auch von Nodes veröffentlicht werden, um Nachrichten an andere Nodes zu senden. Eine Node, die ein Topic veröffentlicht, wird als Publisher bezeichnet. Der Publisher kann regelmäßig Nachrichten auf einem Topic veröffentlichen, um andere Nodes über Änderungen in der Umgebung oder im System zu informieren.

Die Verwendung von Topics in ROS ermöglicht eine einfache und flexible Kommunikation zwischen Nodes, was besonders in komplexen Systemen von Vorteil ist. Nodes können sich auf mehrere Topics abonnieren und Nachrichten an mehrere Topics veröffentlichen, was eine effektive und modulare Datenverarbeitung ermöglicht. Zudem können Topics auf mehreren Hosts oder in verschiedenen Prozessen ausgeführt werden, was eine Skalierung des ROS-Systems ermöglicht.

Insgesamt sind Topics eine wichtige Komponente von ROS und ermöglichen es Entwicklern, eine einfache und effektive Kommunikation zwischen Nodes in Robotik-Anwendungen zu realisieren.

3.2.3 Publish and Subscripe Pattern

Das Publish-Subscribe-Pattern ist ein grundlegendes Muster der ROS-Kommunikation und ermöglicht eine effektive und modulare Datenverarbeitung in verteilten Systemen.

Beim Publish-Subscribe-Pattern senden Nodes, die Informationen über eine bestimmte Ressource verarbeiten, die Informationen an ein Topic, das als Vermittler dient. Nodes, die an den Informationen interessiert sind, abonnieren das Topic und erhalten alle zukünftigen Nachrichten, die von Nodes veröffentlicht werden, die mit dem Topic verbunden sind.

Dieses Muster hat mehrere Vorteile. Zum einen ermöglicht es eine flexible Architektur, in der Nodes unabhängig voneinander arbeiten und sich auf das Abonnieren und Veröffentlichen von Topics konzentrieren können, ohne die genaue Identität des Empfängers oder Senders zu kennen. Zum anderen ermöglicht es eine effektive Datenverarbeitung, da mehrere Nodes dieselben Informationen von einem Publisher erhalten können.

Ein weiterer Vorteil des Publish-Subscribe-Patterns ist, dass es eine einfache Möglichkeit bietet, den Zustand von Ressourcen zu überwachen oder auf Änderungen in Echtzeit zu reagieren. So kann beispielsweise eine Node, die eine Kamera überwacht, die Bilder auf

einem Topic veröffentlichen. Andere Nodes, die an der Verarbeitung dieser Bilder beteiligt sind, können sich auf das Topic abonnieren und die Informationen in Echtzeit verarbeiten.

Das Publish-Subscribe-Pattern ist ein grundlegendes Konzept in ROS und wird in der Regel für die Kommunikation zwischen Nodes verwendet. Es ermöglicht eine effektive und modulare Datenverarbeitung in verteilten Systemen und ist ein wesentlicher Bestandteil von ROS, um komplexe Robotik-Anwendungen zu realisieren.

3.2.4 Objekterkennung

In ROS gibt es verschiedene Methoden zur Objekterkennung, die in Robotik-Anwendungen eingesetzt werden können. Die Objekterkennung ist ein wichtiger Schritt in der automatisierten Wahrnehmung von Robotern, da sie es ihnen ermöglicht, ihre Umgebung zu verstehen und darauf zu reagieren.

Eine häufig verwendete Methode zur Objekterkennung in ROS ist die Verwendung von 3D-Sensoren wie Lidar oder Kinect. Diese Sensoren erfassen Daten über die Umgebung des Roboters und können dabei helfen, Objekte zu identifizieren und ihre Position und Orientierung im Raum zu bestimmen.

Eine weitere Methode zur Objekterkennung in ROS ist die Verwendung von Bildverarbeitungs-Algorithmen. Dabei können beispielsweise Farb- oder Formmerkmale verwendet werden, um Objekte in Bildern zu erkennen und ihre Position und Ausrichtung zu bestimmen.

Eine weiterentwickelte Methode zur Objekterkennung in ROS ist die Verwendung von Deep-Learning-Methoden wie Convolutional Neural Networks (CNNs). Dabei werden CNNs trainiert, um Objekte in Bildern oder Punktwolken zu erkennen und zu klassifizieren. Diese Methode erfordert jedoch ein umfangreiches Training und eine hohe Rechenleistung, um in Echtzeit ausgeführt zu werden.

Die Objekterkennung ist ein wichtiger Schritt in der automatisierten Wahrnehmung von Robotern, da sie es ihnen ermöglicht, ihre Umgebung zu verstehen und darauf zu reagieren. ROS bietet verschiedene Methoden zur Objekterkennung, die in Robotik-Anwendungen eingesetzt werden können. Die Wahl der richtigen Methode hängt von den Anforderungen der Anwendung ab und erfordert oft eine sorgfältige Abwägung zwischen Genauigkeit, Geschwindigkeit und Komplexität.

3.2.5 QR-Codes

ROS bietet verschiedene Möglichkeiten zur Erkennung von QR-Codes in Robotik-Anwendungen. QR-Codes sind zweidimensionale Barcodes, die Informationen wie URLs, Texte oder andere Daten enthalten können. Die Erkennung von QR-Codes kann in ROS-basierten Anwendungen genutzt werden, um Informationen zu lesen, Roboter zu navigieren oder um eine Interaktion mit der Umgebung zu ermöglichen.

Es gibt verschiedene ROS-Pakete, die die QR-Code-Erkennung erleichtern. Ein Beispiel ist das ßbar_ros"Paket, das ein Wrapper für die Open-Source-ZBar-Bibliothek ist, die QR-Codes und andere Barcodes erkennt. Das zbar_ros-Paket ermöglicht es, den Inhalt von QR-Codes aus dem Kamerabild zu extrahieren und als ROS-Topic zu veröffentlichen, der von anderen Nodes abonniert werden kann. Das Paket bietet auch Optionen zur

Konfiguration der QR-Code-Erkennung, wie beispielsweise die Festlegung der Mindestgröße des Codes oder die Einstellung der Scan-Frequenz.

Ein weiteres ROS-Paket, das die QR-Code-Erkennung erleichtert, ist das "ros_qr_detector"Paket. Dieses Paket basiert auf der OpenCV-Bibliothek und bietet eine einfache Möglichkeit, QR-Codes in ROS-basierten Anwendungen zu erkennen. Das Paket bietet auch die Möglichkeit, QR-Codes aus der Kamerabildanzeige auszuschneiden und als separate Bilder zu speichern.

Die Erkennung von QR-Codes in ROS-Anwendungen kann für verschiedene Anwendungsfälle nützlich sein. Beispielsweise können QR-Codes als Marker verwendet werden

3.3 Drohne/Multicopter

Bei der für das Projekt verwendete Drohne handelt es sich um eine Coex Clover Drohne. Dies ist eine programmierbare Drohne, die besonders für Bildungszwecke eingesetzt wird. Sie ist sowohl für den Einsatz draußen sowie auch in Gebäuden geeignet.

Abbildung 3.1: Coex Clover Drohne [Robots o. D.]

Zu Beginn erhält man hierbei einen Bausatz, welcher dann zu einem Quadrokopter zusammengebaut werden kann. Der Vorteil hierbei ist zudem, dass die gesamte Drohne ohne Löten zusammengesetzt werden kann. Zu den einzelnen Bestandteilen der Drohne kommen, noch eine Dokumentation sowie verschiedene Bibliotheken, die es ermöglichen, die Drohne zusammen bauen und fliegen lassen zu können.
Die Durch die Verwendung verschiedener Open-Source Komponenten lässt sich die Drohne programmieren, wodurch ein vielseitiger Einsatzbereich entsteht.

Die Coex Clover Drohne soll laut Herstellerinformationen bis zu 15 Minuten am Stück fliegen können und in dieser Zeit eine Maximalhöhe von 500 Metern bei einer Höchstge-

Abbildung 3.2: Bausatz Coex Clover Drohne [COEX o. D.]

schwindigkeit von bis zu 72 km/h erreichen können [vgl. Coex o. D.]

Es gibt verschiedene Versionen der Drohne, bei der hier eingesetzten, handelt es sich um die Coex Clover 4.2. Wichtige Bestandteile dieser sind hierbei:

- Flight-Controller Coex Pix
- Raspberry Pi 4
-
-
-
-

Zu den Hauptbestandteilen der Drohne zählen zum einen ein Raspberry Pi 4 sowie der Flightcontroller Coex Pix. Diese bilden die Grundlage zur Programmierung und Steuerung der Coes Clover Drohne und ermöglichen es zudem die Drohne über drahtlos per WLAN zu verbinden.
Die Drohne ist ein Quadrokopter und besitzt somit vier Motoren, welche einzelnd angesteuert werden können. Sie besitzt zudem eine Vielzahl verschiedener Sensoren, auf welche in Kapitel **??** genauer eingegangen wird. Zu diesen zählen unter anderem ein Gyroskop, Magnetometer sowie ein Laseranstandssensor und eine Kamera, die unten an der Drohne angebracht sind. Zum Schutz befindet sich zudem außen einen Rahmen.

3.4 3D-Modelle

3.4.1 3D-Scanning

3.5 Positionsbestimmung

3.5.1 Spatial Mapping

3.5.2 Inertielle Positionsbestimmung

3.6 Regelsysteme

3.6.1 PID-Regler

3.6.2 Extended Kalman Filter 2

Der Extended Kalman Filter (EKF)2-Algorithmus, der in der Drohnensteuerung verwendet wird, ist eine Erweiterung des klassischen erweiterten Kalman-Filters, die speziell auf die Bedürfnisse der Drohnensteuerung zugeschnitten ist. Der EKF2 ist ein geschätzter Zustandsregler, der die aktuellen Zustände (z.B. Position, Geschwindigkeit, Orientierung) einer Drohne schätzt, basierend auf Messungen von Sensoren (z.B. GPS, IMU, Magnetometer).

Im Gegensatz zum klassischen EKF verwendet der EKF2 eine modifizierte Version der Kalman-Filter-Formeln, um den Einfluss von Sensorrauschen und Messfehlern besser zu berücksichtigen. Insbesondere verwendet der EKF2 eine sogenannte İnnovation Covariance Matrix", die die Varianz der Messfehler repräsentiert und in die Filtergleichungen eingebaut wird. Diese Innovation Covariance Matrix wird iterativ während des Betriebs des Filters aktualisiert, um den Sensorrauschen und Messfehlern besser gerecht zu werden.

Darüber hinaus verwendet der EKF2 eine modifizierte Version der State Transition Matrix, die die Nichtlinearitäten des Systems besser modellieren kann. Diese modifizierte Matrix wird ebenfalls iterativ während des Filterbetriebs aktualisiert, um den Änderungen im Systemverhalten besser gerecht zu werden.

In der Drohnensteuerung wird der EKF2-Algorithmus verwendet, um die aktuellen Zustände der Drohne (z.B. Position, Geschwindigkeit, Orientierung) in Echtzeit zu schätzen. Diese Schätzungen werden dann verwendet, um die Steuerbefehle der Drohne zu generieren, um sie auf Kurs zu halten und sicher zu navigieren.

3.7 Simulationstechnik

3.8 Problembehebung

Kapitel 4

Probleme

4.1 Aufgetretene Probleme

4.1.1 Lösungen

Kapitel 5

Hardware Implementierung

5.1 Komponenten

- Raspberry Pi 4 1 GB
- COEX Clover 4.2

5.2 COEX Clover 4.2

5.2.1 Abstandssensor

5.2.2 PX4 FlightController

- Barometer
- Gyroskop
- Accelerometer
- Magnetometer

5.2.3 Kamera

5.3 3D Scanner

5.3.1 Microsoft Hololens

5.3.2 Microsoft Kinect

Kapitel 6

Software Implementierung

6.1 Systemarchitektur

6.2 Softwarearchitektur

6.3 Kalibrierung der COEX Drohne

6.4 Drohnen Autopilot

6.5 3D Modell

6.5.1 3D Scan

Microsoft HoloLens 2

Microsoft Kinect 2

6.5.2 3D Modell Vorbereitung

6.6 Navigation

6.7 Simulation

Kapitel 7

Fazit

7.1 Ausblick

7.2 Erwartungen

7.3 Probleme

Literatur

COEX [o. D.] *A comprehensive aerial robotics solution*. URL: https://coex.tech/clover [besucht am 08. 02. 2023] [siehe S. 17].

Coex [o. D.] *COEX Clover*. URL: https://clover.coex.tech/en/ [besucht am 06. 02. 2023] [siehe S. 17].

Robots, ROS [o. D.] *COEX Clover*. URL: https://robots.ros.org/clover/ [besucht am 06. 02. 2023] [siehe S. 16].

Liste der ToDo's