Master Thesis

im Rahmen des
Universitätslehrganges „Geographical Information Science & Systems“
(UNIGIS MSc) am Zentrum für GeoInformatik (Z_GIS)
der Paris Lodron-Universität Salzburg

zum Thema

„GIS und UAV“
Einsatzplanung und Dokumentation

vorgelegt von

Dipl.Ing.(FH) Andreas Feichtner
U1464, UNIGIS MSc Jahrgang 2009

Zur Erlangung des Grades
„Master of Science (Geographical Information Science & Systems) – MSc(GIS)”

Gutachter:
Univ.-Prof. Dr. Thomas Blaschke

Linz, 10. Dezember 2012

Danksagung

Ich bedanke mich herzlichst beim UniGIS Team, das mich in all der Zeit des Studium bestens betreut hat. Besonderen Dank an das Research Studio iSpace und an meine Betreuer Prof. Thomas Blaschke und Manfred Mittlböck, die mich bei der Erstellung der Master Thesis unterstützt und mich fachlich immer bestens beraten haben.

Ein weitere Dank ist an Herrn Helmut Rozmann gerichtet, der mir bei der Konstruktion der Mikro-Drohne fachlich beraten und unterstützt hat.

Einen Dank auch an die Wasserwerke der Stadt Linz, die mir freundlicherweise Geodaten zur Verfügung gestellt haben und an die Stadt Linz, die Ihre Orthofotos unter der „Creative Common Namensnennung – CC BY 3.0“ Lizenz veröffentlicht haben.

Ein besondere Dank gilt meiner ganzen Familie, die Verständnis aufbrachte und mich unterstützte und motivierte.

Erklärung der eigenständigen Abfassung der Arbeit

Ich versichere, diese Masterarbeit ohne fremde Hilfe und ohne Verwendung anderer als der angeführten Quellen angefertigt zu haben, und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat. Alle Ausführungen der Arbeit, die wörtlich oder sinngemäß übernommen wurden, sind entsprechend gekennzeichnet.

Linz, 10. Dezember 2012

Andreas Feichtner

Kurzfassung

Die vorliegende Arbeit befasst sich mit der Erstellung eines GIS Workflow für den Einsatz eines Unbemannten Luftfahrzeuges - UAV, im Speziellen bei dieser Arbeit anhand einer Mikro-Drohne vom Bautyp eines Oktokopters. Mit Methoden und Werkzeugen der Geoinformatik wird in dieser Arbeit ein durchgehender GIS Workflow vorgestellt, der die aufeinanderfolgenden Schritte, ausgehend von der Einsatzplanung mit der Akquirierung von Geodaten des Einsatzgebietes, Durchführung des Einsatzes und nachträgliche Validierung und Dokumentation der gewonnenen Geodaten, exemplarisch vorführt. Der erste Teil beschäftigt sich mit den theoretischen Grundlagen der Intertialen Navigation von unbemannten autonom fliegenden Fluggeräten, und stellt die Konstruktion und den Aufbau der Komponenten, der in dieser Arbeit verwendeten Mikro-Drohne, vor. Für die Lösung wird ein GIS Workflow vorgestellt, der die Flugroutenplanung im Desktop GIS auf Basis der akquirierten Geodaten ermöglicht und mit einer programmierten Erweiterung für das Desktop GIS die geplanten Einsatzdaten aus dem Desktop GIS an die Mikro-Drohne überträgt. Die gewonnenen räumlichen Einsatzdaten werden im Desktop GIS nachträglich in Bezug zu den geplanten Einsatzdaten mit den Mitteln der räumlichen Statistik untersucht und bewertet. Abschließend wird die Wirtschaftlichkeit von Mikro-Drohnen Einsätzen und die Rechtliche Situation dieser beleuchtet.

Abstract

This thesis deals with the creation of GIS workflow for the deployment of a Unmanned Aerial Vehicle (UAV). In particular we investigate an Octocopter micro-drone. In this thesis we develop a complete GIS workflow based upon the methods and tools of Geoinformation Systems. This workflow starts with deployment planning and the acquisition of geodata for the deployment region, then proceeds to the implementation of the deployment and then the validation, documentation and integration of the

acquired geodata. This workflow is presented as a case study. The first part deals with the theoretical basis for inertial navigation of unmanned autonomous flying vehicles and presents the planning and construction of the components of the micro-drone used in this thesis. A workflow is then presented that allows the planning of flight routes in a desktop GIS based upon the previously acquired geodata. A specially programmed extension of the desktop GIS system transfers the deployment data to the micro-drone. After the deployment. the obtained geodata is transferred back to the desktop GIS system, where an analysis and evaluation of the data is undertaken based upon geostatistical methods. Finally the economics and legal implications of micro-drone deployment are investigated.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1. Einführung

1.1. Motivation

Bedingt durch den technischen Fortschritt der Mikroelektronik und der einhergehenden Miniaturisierung von elektronischen Bauteilen in den letzten Jahrzehnten, wurden elektronische Bauteile, wie Sensoren und Mikroprozessoren immer leichter, und im Vergleich zur steigenden Leistung kostengünstiger. Resultierend aus dieser Entwicklung kamen kostengünstige und dabei leistungsstarke Mikrocontroller auf den Markt. Diese elektronischen Bauteile beinhalten neben einem leistungsstarken Prozessorkern auch umfangreiche Peripherie, wie Speicher und unterschiedliche Schnittstellen in einem einzigen Chip. Man spricht hier von „*Single Computer Chip*“ oder von *„System-on-Chip“* [Al-Hashimi 2006: S. 64]. Mikrocontroller sind mittlerweile in vielen alltäglichen Produkten verbaut, so das wir uns ihres Gebrauches nicht mehr bewusst sind. Eingebettete Systeme, wie Produkte, die auf Mikrocontrollern basieren, genannt werden, ermöglichen erst den mannigfaltigen Einsatz von intelligenter Technik, die mittlerweile den Alltag durchdrungen hat. Durch die massenhafte Verfügbarkeit dieser leistungsstarken elektronischen Bauteile, ist es möglich kostengünstige Unbemannte Mikro-Fluggeräte, die autonom Flugeinsätze absolvieren können, zu konstruieren. Essentiell für die Abwicklung autonom ausgeführter Flüge ist neben den leistungsfähigeren Raumsensoren auch die Verfügbarkeit eines externen Positionierungssystem, wie das Navstar Global Position System. Das Navstar GPS ist im Jahr 1993 für die zivile Nutzung freigegeben worden und kostenlos.

Mikro-Fluggeräte und im speziellen VTOL – Vertical Take Off and Landing - können, im Vergleich zu Flächenfluggeräten, durch ihre speziellen Flugeigenschaften auf kleinem Raum manövrieren und schweben. Start- und Landemanöver sind beinahe überall möglich. Sie erweisen sich als nützliches Werkzeug für Datenakquisition im Rahmen der Geoinformatik, da sie flexibel mit unterschiedlichsten Sensorsystemen ausgerüstet werden können. In den letzten Jahren ist die „Echtzeitsensorik“ als Forschungsthema attraktiv geworden. Darunter versteht man, dass beinahe in Echtzeit

räumliche-zeitlich thematische Karten für die Beurteilung von räumlichen Phänomenen erstellt werden. Mikro-Drohnen sind eine ideale Ergänzung hierfür, da sie in Bereiche vordringen können, wo ein Einsatz von humanen Ressourcen zu gefährlich oder zu kostspielig wäre. Auch ist der unkomplizierte und rasche Einsatz von Mikro-Drohnen bei sicherheitsrelevanten Aufgaben, wie zum Beispiel im Katastrophenmanagement, von Vorteil. Durch Integrieren eines im dreidimensionalen Raum mobilen Vehikel für die Sensoren eines solchen Echtzeitsystems, ist es möglich relativ schnell aktuelle Karten für die Beurteilung der Situationen zu erstellten, die Entscheidungsträgern zur Verfügung gestellt werden. Die Voraussetzung solcher Mikro-Drohnen Einsätze ist die Planung des autonom räumlichen Bewegungsablaufes solch eines mobilen Sensorsystem anhand von existierenden georeferenzierten räumlichen Daten wie Orthofoto oder Karten, um sinnvoll verwertbare Daten im Sinne der Aufgabenstellung erfassen zu können. Naheliegend wäre, die Einsatzplanung direkt aus dem Geoinformationssystem zu tätigen, da hier meist die Basisdaten für eine Planung schon vorliegen und somit das Echtzeitsystem ergänzen. Auch anhand schon gewonnener Daten, und deren daraus resultierenden thematischen Karten, kann eine ergänzende Datengewinnung einfacher im Kontext und schneller in der Umsetzung durchgeführt werden.

1.2. Aufgabenstellung und Zielsetzung

Die vorliegende Arbeit möchte exemplarisch aufzeigen, wie Einsätze von unbemannten Mikro-Drohnen unter Zuhilfenahme Geografischer Informationssysteme abgewickelt werden können. Mittels aktuellen Geodaten des Einsatzgebietes sollen die Flugrouten und die Missions-Ereignisse geplant und durchgeführt werden. Nach Durchführung der Flugmission sollen die Abweichungen von den geplanten Routen mit den geflogenen Routen dokumentiert werden, damit die Geodaten für ein allfälliges Postprozessing mit gewonnenen Messergebnissen zur Verfügung gestellt werden können.

Im Allgemeinen soll ein Mikro-Drohneneinsatz folgendermaßen ablaufen:

- Akquirierung von aktuellen Geodaten wie Orthofotos, Höhenschichtenmodell

oder Digitales Oberflächenmodell des Einsatzgebietes

- Zusammenführung der Daten in einem Desktop Geoinformationssystem, unter Berücksichtigung von unterschiedlichem Geodätischen Datum und Projektion
- Digitalisieren der anzufliegenden Wegpunkte in einem Desktop Geoinformationssystem und attributieren dieser Wegpunkte mit den notwendigen Parametern
- Transformation der Flugroute in das für den Drohneneinsatz notwendige WGS84 Datum
- Übermittlung der Wegpunkte als Flugeinsatzplan an die Mikro Drohne
- Durchführung des Drohneneinsatz
- Auswertung des Fluges anhand der aufgezeichneten Flugroute und Dokumention des Einsatzes zur Qualitätssicherung

Die Arbeit soll folgende Fragen beantworten helfen:

- Was für Vorteile bringt die Einsatzplanung für Mikro-Drohnen unter Zuhilfenahme eines Geoinformationssystems mit aktuellen Geodaten?
- Welche geografische Genauigkeit wird beim Einsatz der Mikro-Drohne erreicht und wie groß sind die Abweichungen der geplanten Flugroute von der tatsächlich geflogenen Route ?
- Wie kann man einen Mikro-Drohneneinsatz abwickeln und sinnvoll dokumentieren ?
- Wie kann man gewonnene Messergebnisse eines Drohneneinsatzes mit der aufgezeichneten Flugroute nachträglich verschneiden, um der thematischen Aussage ergänzend zur Verfügung zu stellen ?
- Welche Vor- und Nachteile bringt die Verwendung einer Mikro-Drohne im Vergleich zu professionellen UAV Systemen

1.3. Zeitlicher Ablauf und Struktur der Arbeit

Der zeitliche Ablauf der Arbeit ist, wie in der Grafik (Abb. 1.1) dargestellt, in zwei Hauptkomponenten unterteilt. Zuerst erfolgte die Vorbereitung, wie Literaturrecherchen und allgemeine Orientierung in der Thematik. Anschließend die Evaluierung einer Mikro-Drohne mit den theoretischen Grundlagen der autonomen Steuerung und anschließendem Bau einer Mikro-Drohne mit Erklärung deren Handhabung.

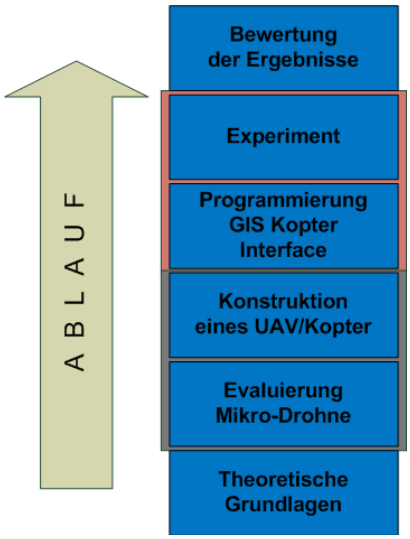


Abbildung 1.1: Zeitlicher Ablauf der Master Thesis Erstellung

Die zweite Hauptkomponente der Arbeit, die nach der Konstruktion der Mikro-Drohne erfolgt, ist der Geoinformationssystem-spezifische Teil der Arbeit. Hier wird das Interface für die Kommunikation zwischen dem Desktop Geoinformationssystem und der Mikro-Drohne programmiert und das anschließende Experiment durchgeführt. Abschließend werden die gewonnene Daten des Experiments bewertet.

Der strukturelle Aufbau der Arbeit unterteilt sich in 8 Kapiteln. Kapitel 1 beschreibt die Einführung und Motivation dieser Arbeit und Kapitel 2 geht auf die dazu benötigte Literatur ein.

In Kapitel 3 erfolgt die Evaluierung einer Mikro-Drohne und die Konstruktion solch

einer Drohne. Die konstruierte Mikro-Drohne dient als Untersuchungswerkzeug für den geoinformationsspezifischen Teil der Arbeit. Dieses Kapitel beschreibt die erforderliche ingenieurmäßigen Fertigkeiten und das theoretische Basiswissen über autonome Steuerung und deren erforderliche Bauteile, die sich der Autor dieser Arbeit im Rahmen der Vorarbeiten und der Konstruktion der Mikro-Drohne angeeignet hat. Des weiteren wird die proprietäre Mikro-Drohnen Software für autonomes Fliegen und Flugtelemetrie evaluiert und deren Funktionen vorgestellt.

Im Kapitel 4 dieser Arbeit wird einerseits auf die Vorbereitung für das Experiment eingegangen. Dazu zählt die Geodatenbeschaffung, die Vorbereitung der Daten für das Desktop Geoinformationssystem und die Erstellung von Featureklassen mit den passenden Attributen für die Mikro-Drohne, sowie die Überführung der von der Mikro-Drohne gelieferten Ergebnisse in ein Format, das im Desktop Geoinformationssystem weiterverarbeitet werden kann. Ergebnis ist ein Konzept eines GIS Workflow für die Flugroutenplanung und Durchführung mit anschließender Sicherstellung der gewonnenen Daten für die Auswertung des Mikro-Drohnen Einsatzes. Des weiteren werden noch die beschafften Geodaten überprüft und verglichen.

In Kapitel 5 werden die notwendigen Werkzeuge, die nicht vom Desktop Geoinformationssystem bereitgestellt werden, konzipiert und mit der Scriptsprache „Python“ implementiert, um den GIS Workflow zu vervollständigen und darzustellen.

In Kapitel 6 wird ein Experiment an Hand der entwickelten GIS Workflow durchgeführt. Es dient dem Test der programmierten Werkzeuge, sowie zur Bereitstellung von Ergebnissen um den erstellten GIS Workflow zu validieren.

In Kapitel 7 werden die Ergebnisse der Master Thesis diskutiert, sowie die rechtliche Situation des Einsatzes von Mikro-Drohnen beleuchtet.

In Kapitel 8 wird die Arbeit zusammengefasst und bewertet wie effizient die Ergebnisse dieser Arbeit sind, sowie Ausblick auf mögliche Fortführung dieser Arbeit gemacht.

2. Literatur

Für das Kapitel 3 – „Technische Evaluierung eines Kopter, Aufbau, Koptersteuerung, Bedienung und Theorie“ wurde das deutschsprachige Standartwerk von Jan Wendel über Integrierte Navigationssysteme herangezogen. Das Werk gibt einen umfassenden Überblick über die zugrundeliegende Theorie und Mathematik der Steuerung autonom fliegender Systeme anhand der verschiedenen Bezugs-Koordinatensysteme und die Umrechnung untereinander. Auf das Prinzip der Sensordatenfusion und diverse Filter wird eingegangenen und die mathematischen Grundlagen derer präsentiert. Abschließend wird das mathematische Model einer Mikro-Drohne ausgearbeitet, welches auf die Mikro-Drohne dieser Arbeit umgelegt wird. Das Werk von Manfred Bauer – Vermessung und Ortung mit Satelliten dient der Erarbeitung der Grundlagen für den Einsatz von Global Positioning System (GPS), das eine Hauptkomponente der Mikro-Drohne ist. Für die Mikro-Drohne und deren Grundlage zur Programmierung wurde das Internet Wiki des „Mikrokopter Projekt“ verwendet. Über das Wiki besteht der Zugriff zum Quellcode der Mikro-Drohnen Firmware und das Kommunikationsprotokoll ist in diesem Wiki ausführlich erklärt. Für die Programmierung der ArcGIS Toolbox mit der Programmiersprache Python wurden einerseits die Hilfe Seiten des „ESRI ArcGIS“ Desktop Geoinformationssystem herangezogen andererseits die offiziellen Webseiten des Python Projekt mit der Dokumention der Programmiersprache. Ergänzend für Informationen über die verwendete Hardware wurden Quellen der Elektronikhersteller herangezogen. Für die benötigten Transformationsparameter und dem in Österreich verwendeten Geodätischen Datum ist die Online Information des Bundesamt für Vermessung und Eichwesen benutzt worden. Der Rechtliche Teil dieser Arbeit wurde beim Internetauftritt des „Institutes für Österreichisches und Internationales Luftfahrtrecht“ recherchiert, welches vom „Österreichischen Wissenschaftsfond“ unterstützt und gefördert wird, sowie das Portal des „Österreichischen Parlaments“ für fachspezifische Antworten des „Bundesministerium für Verkehr, Innovation und Technologie“.

3. Technische Evaluierung einer Mikro-Drohne, Theorie Aufbau und Bedienung

Das folgende Kapitel beschäftigt sich mit dem Aufbau und der Bedienung der Mikro-Drohne, welche für diese Arbeit eingesetzt wird. Einleitend wird der generische Aufbau einer Mikro-Drohne evaluiert. Danach werden die mathematisch theoretischen Grundlagen der Inertialen Navigation einer Mikro-Drohne behandelt, sowie auf die Bedienung der Mikro-Drohne und auf die proprietäre mitgelieferte Software eingegangen. Anschließend der Eigenbau beschrieben, sowie die nach den ersten Erprobungen vorgenommenen Abänderungen und Verbesserungen, die an der Mikro-Drohne vorgenommen wurden.

3.1. Evaluierung Mikro-Drohne und Aufbau der Komponenten

Die evaluierte Mikro-Drohne ist ein unbemanntes Fluggerät und gehört zur Klasse der Senkrechtstarter und Senkrechtlander. Der Fachbegriff hierfür ist VTOL - „Vertical Take Off and Landing“ und wird auch als „Kopter“ bezeichnet. Ein Kopter besteht aus einem Rahmen für den Aufbau, Motoren mit Propellern, die als Rotoren bezeichnet werden und einer Steuerungselektronik mit Verkabelung. Die Energieversorgung für die Elektronik und für die Motoren dieser evaluierten Drohne ist elektrisch und wird von einem Lithiumpolymer Akkumulator bereitgestellt.

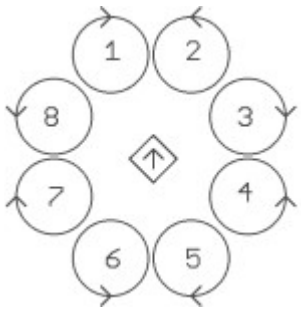


Abbildung 3.1: Rotoranordnung Oktokopter Bildquelle: [mikrokopter wiki mkm]

3.1.1. Rahmen

Der Rahmen der evaluierten Mikro-Drohne besteht aus einer zentralen Mittelscheibe mit einer geradzahligen Anzahl von Auslegern, die vom Mittelpunkt der Scheibe sternförmig symmetrisch nach außen weisend angeordnet sind. Auf der Mittelscheibe sind die Steuerelektronikkomponenten, der Akkumulator für die Energieversorgung, optional eine Vorrichtung zur Befestigung von Fracht sowie die Ausleger montiert. Die kleinst mögliche Anzahl von Auslegern ist Vier um einen Kräfteausgleich zu erreichen. Durch den sternförmigen, symmetrischen Aufbau des Rahmen und der Montage der Komponenten an der zentralen Mittelscheibe ist der Schwerpunkt der Mikro-Drohne im geometrischen Mittelpunkt des Rahmen. Der gemeinsame Schwer- und Mittelpunkt der Mikro-Drohne ermöglicht eine gleichmäßige Verteilung der Schubkräfte an den Antriebseinheiten.

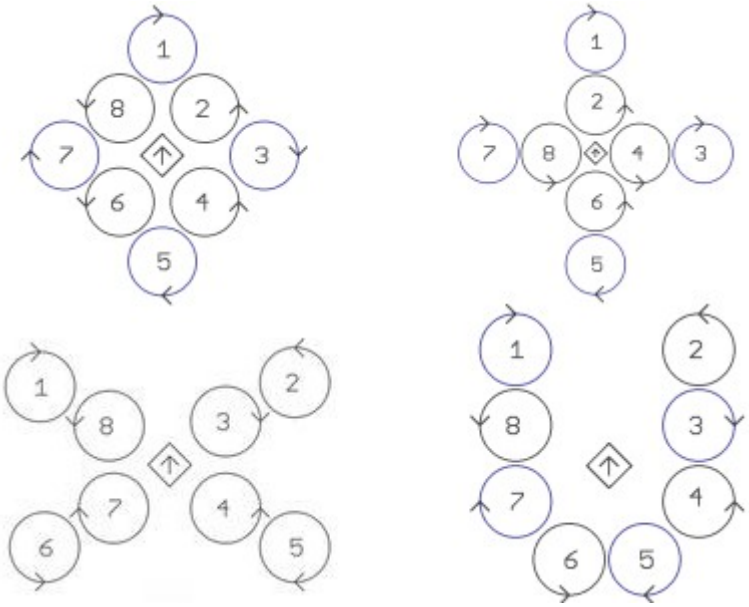


Abbildung 3.2: Weitere Rotorenanordnung Oktokopter
Bildquelle: [mikrokopter wiki mkm]

An den äußeren Enden der Ausleger ist je ein Rotor senkrecht zur Mittelscheibe nach oben zeigend montiert, dessen Schubkraft senkrecht nach unten wirkt. Der Schwerpunkt der Mikro-Drohne liegt horizontal betrachtet unter dem von den Antriebseinheiten erzeugten Schubvektor, was für die Stabilität von Vorteil ist. Um die von den Rotoren erzeugten Drehmomente auszugleichen, ist die Drehrichtung der

Rotoren abwechselnd im Uhrzeigersinn und gegen den Uhrzeigersinn konfiguriert. Neben der beschriebenen sternförmig symmetrischen Anordnung der Rotoren (Abb. 3.1), gibt es noch andere mögliche Anordnungen, wobei die jeweilige Anordnung in der Steuerungselektronik parametrisiert werden muss (Abb 3.2).

3.1.2. Antriebseinheiten

Die Antriebseinheiten einer Mikro-Drohne bestehen aus bürstenlosen Elektromotoren mit montierten Propellern, die senkrecht zur Motorachse stehen und bei Rotation eine Schubkraft erzeugen. Der Vorteil von einem bürstenlosen Motor ist, das der Motor nahezu verschleißfrei ist, da die keine Kohlebürsten verbaut sind. Man kann mit dieser Art von Motor eine hohe Drehzahl erreichen und damit eine große Leistung bei geringem Gewicht übertragen. Um solch einen Motor zu betreiben, benötigt man aber eine spezielle Elektronik, die aus dem Gleichstrom ein magnetisches Drehfeld erzeugt. Die von den Rotoren erzeugte Schubkraft ist technisch betrachtet ein Impulsantrieb und muss eine gleich große Kraft der entgegenwirkenden Gravitationskraft erzeugen, um ein Schweben zu ermöglichen. Beziehungsweise muss eine größere Kraft erzeugt werden um die Mikro-Drohne steigen zu lassen. Der Impuls wird durch eine Beschleunigung der Luftmasse erzeugt. Nach der Newtonschen Mechanik ist ein Impuls das Produkt von Masse und Geschwindigkeit

$$\vec{I} = m\vec{v}$$

und die Kraft die für eine Impulsänderung aufgewendet werden muss ist die Änderung der Geschwindigkeit der Masse pro Zeit

$$F_i = m\frac{\Delta v}{\Delta t}$$

Beim Propeller der Mikro-Drohne wird die Luftmasse nach unten beschleunigt, es kommt zu einer Differenz zwischen der einströmenden Luftgeschwindigkeit v_1 und der ausströmenden Luftgeschwindigkeit v_2 . Die bewegte Luftmasse ergibt sich

vereinfacht aus der Propellerfläche als Kreis betrachtet $A_{proppeller}=\frac{d^2\Pi}{4}$ mit der Dichte des durchströmenden Medium , p das dem Druck entspricht, welche das Medium Luft ausübt, und einer durchschnittlichen Durchströmungs-Geschwindigkeit v_m . Daraus ergibt sich vereinfacht unter Annahme eines idealen Medium ohne Reibung und Turbulenzen die Schubkraft des Propeller

$$F_{schub}=A_{proppeller}\,p\,v_m(v_2-v_1)$$ [Böswirth 2010: S. 71]

Um zur Propellergleichung zu kommen verwenden wir das Gesetz von Bernoulli

$$\frac{1}{2}\,p\,v^2+p=const.$$ [Böswirth 2010: S. 71]

Daraus folgt die Formel welche theoretische Druckzunahme der Propeller erzeugen muss, um die Druckabnahme durch die Geschwindigkeitserhöhung des Luftstrahles von v_1 nach v_2 zu kompensieren, unter der Annahme, dass in der Realität der Druck konstant ist.

$$\frac{1}{2}\,p\,v_1^2+\Delta p=\frac{1}{2}\,p\,v_2^2\rightarrow\Delta p=\frac{1}{2}\,p\,(v_2^2-v_1^2)$$ [Böswirth 2010: S. 71]

$$F_{schub}=A_{proppeller}\,\Delta p$$ [Böswirth 2010: S. 71]

$$A_{proppeller}\frac{1}{2}\,p\,(v_2^2-v_1^2)=A_{proppeller}\,p\,v_m(v_2-v_1)$$ [Böswirth 2010: S. 71]

gemäß der Betz'schen Theorie $v_m=\frac{1}{2}(v_1+v_2)$ und substituieren vom v_m im Term auf der rechten Seite mit und Anwenden des Satzes $(a^2-b^2)=(a+b)(a-b)$

beweist die vereinfachte Propellerschubkraft für eine sich bewegende Mikro-Drohne

$$F_{schub}=A_{proppeller}\,p\,\frac{1}{2}(v_2^2-v_1^2)$$ [Böswirth 2010: S. 71]

Unter Annahme einer schwebenden Mikro-Drohne $v_1\approx 0$ und Eliminieren ergibt sich

$$F_{schub}=A_{proppeller}\frac{1}{2}\,p\,\Delta v^2$$

Die Geschwindigkeitszunahme Δv resultiert aus den aerodynamischen Eigenschaften des Propeller, wie die Anzahl der Propellerblätter, Blattsteigung und Drehzahl des Propellers. Da $A_{proppeller}$ und p konstant sind, muss für die Erhöhung der Schubkraft F_{schub} die Durchströmungs-Geschwindigkeit erhöht werden, die durch die Erhöhung der Drehzahl der Motoren mit den daran befestigten Propellern erreicht wird. Wie schon erwähnt, sind die Motoren bürsten-lose Gleichstrommotoren. Ein bürsten-loser Gleichstrommotor besteht aus einem feststehenden mittleren Strator, an dem elektromagnetische Spulen, deren Anzahl ein vielfaches von drei ist, symmetrisch radial angeordnet sind. Der Rotor ist ein Hohlzylinder der über den Strator an einer Mittelachse übergeschoben ist. An der Innenseite des Rotors sind Permanentmagneten befestigt.

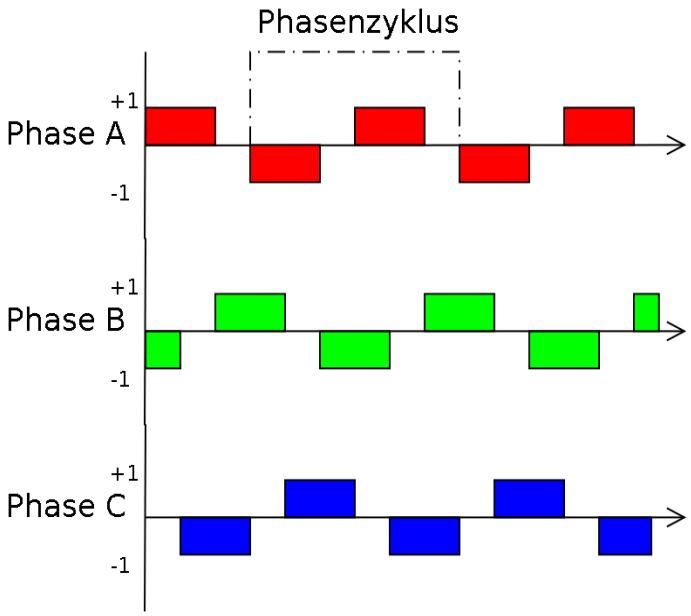


Abbildung 3.3: Stromverlauf des dreiphasigen Drehfeld eines bürstenlosen Gleichstrommotor

Die Bewegung des Rotors wird durch abwechselndes Ein- und Ausschalten von Gleichstrom an den Spulen erreicht, da ein fließender Strom ein Magnetfeld erzeugt, welches die an der Innenseite befestigten Permanentmagneten anzieht oder abstößt, je nach Stromflussrichtung. Jede dritte Spule am Strator ist elektrisch miteinander in Serie

geschaltet und am Ende sind alle drei Leitungen miteinander verbunden (Sternschaltung), sodass der Motor drei Stromzuführungen benötigt. Die drei Stromzuführungen werden abwechselnd mit positiver und negativer Spannung versorgt, die zueinander um $\frac{2}{3}$ auf der Zeitachse verschoben sind. Die Schaltzeit einer positiven oder negativen Spannung eines Phasenanschlusses beträgt ebenfalls $\frac{2}{3}$. Wie aus der Abbildung 3.3 ersichtlich, benötigt man nur drei Phasenanschlüsse, da die entgegengesetzten Spannungen zweier Anschlüsse sich aufheben und der Stromfluss in Summe Null ergibt. Es entsteht ein sich um die Mittelachse des Strator rotierendes Magnetfeld, das die Permanentmagneten des Rotors mitzieht beziehungsweise abstößt und somit die elektrische Energie an den am Rotor befestigten Propeller als mechanische Energie für die Impulserzeugung überträgt.

3.1.3. Elektronische Komponenten

Die elektronischen Komponenten einer Mikro-Drohne bestehen aus einer Reihe von aktiven und passiven Modulen, welche unterschiedliche Funktionen in der Mikro-Drohnen Steuerung übernehmen. Unter aktivem Modul versteht man ein Bauteil, welches einen Mikroprozessor, mit darauf gespeicherter Software, die als „Firmware“ bezeichnet wird, beinhaltet, komplexere Aufgaben übernimmt und mittels eines oder mehrerer Kommunikationsbus mit anderen aktiven Modulen Daten austauscht.

Zu den aktiven Modulen einer Mikro-Drohne zählen:

- Brushless Controller
- Flight Controller
- Navigations Controller (optional)
- GPS Modul (optional)
- Magnetic Compass Modul (optional)

- Remote Controller und Telemetrie Sender und Empfänger

Zu den passiven Bauteilen einer Mikro-Drohne zählen:

- Verkabelung für Kommunikation zwischen Modulen
- Verkabelung für Energieversorgung
- Stromverteilungsplatine
- GPS Antenne

Brushless Controller

Wie in Abbildung 3.3 dargestellt, erfolgt die Erzeugung des magnetischen Drehfeldes eines bürstenlosen Motors durch Verschiebung der Schaltzeitpunkte der drei Phasen zueinander um $\frac{2}{3}$. Die Stromschaltzeitpunkte der einzelnen Phasenanschlüsse des Motors wird von einem „Brushless Controller“ mittels eines Mikroprozessor errechnet und mit Hilfe eines Leistungs-Metall-Oxid-Halbleiter-Feldeffekttransistor (MOSFET), welcher die hohen Stromstärken von bis zu 40 Ampere verarbeiten kann, geschaltet. Am Brushless Controller sind sechs MOSFET's verbaut, für jeden Motorphasenanschluss einer für die positive und einer für die negative Spannungsflanke. Die Leistungsanpassung und somit die Drehzahl des Motors wird durch Pulsweitenmodulation (PWM) erreicht. Dabei wird der Stromfluss zerhackt, sodass in Summe weniger Leistung dem Motor zugeführt wird. Die zugeführte Leistung entspricht der Summe der blauen Flächen (Abb. 3.4). Die theoretische elektrische Leistung P des Motor ohne Verluste ist

$$P = U\,I$$

wobei die Spannung U_{akku} des Akkumulator beinahe konstant ist und sich der Mittelwert der Spannung aus den zerhackten Anteilen errechnen lässt. Daraus ergibt sich, das eine Leistungsregulierung durch die von der Pulsweitenmodulation zugeführten durchschnittlichen Spannung erreicht werden kann, die mit der erreichten

Drehzahl korreliert. Die Schaltzeitpunkte der Pulsweitenmodulation zur Leistungsregelung wird ebenfalls, vom Mikroprozessor des Brushless Controllers errechnet, am jeweiligen MOSFET geschaltet.

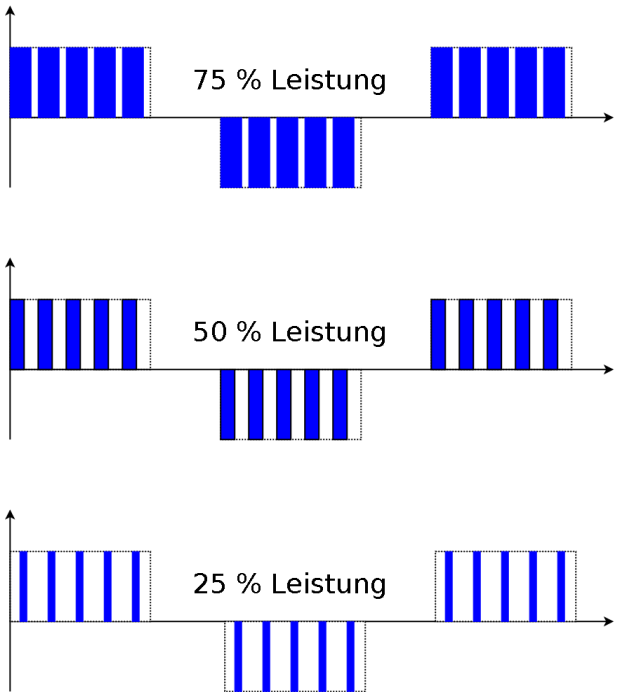


Abbildung 3.4: Strommengenvergleich bei Pulsweitenmodulation einer Phase für die Leistungsregulierung

Die jeweils benötigte Leistung einer Antriebseinheit, um die Mikro-Drohne stabil zu halten, wird von der Flight Controll der Mikro-Drohne anhand von verschiedensten Sensordaten in beinahe Echtzeit errechnet und an den zuständigen Brushless Controller über den I²C Datenbus mitgeteilt. Der Leitungsparameter wird binär in einem Datagramm kodiert auf den I²C Bus übertragen. Der I²C Bus ist ein serielles Datenübertragungssystem für Kommunikation zwischen Mikrocontrollern und elektronischen Bauteilen. Der I²C Bus besteht aus 2 Leitungen. Einer Taktleitung, als *SCL - Serial Clock Line* bezeichnet, die den Übertragungstakt vorgibt, und einer Datenleitung, als *SDA - Serial Data Line* [Semiconductors 2000: S. 9] bezeichnet, auf der die Nachricht in einem Datagramm übertragen wird. An einem I²C Bus sind die beteiligten Kommunikationskomponenten angebunden und es gibt genau ein Master-

Device und ein-bis mehrere Slave-Devices. Im Falle der Mikro-Drohne ist die Flight Controll das Master-Device und die Brushless Controller sind die Slave-Devices (Abb 3.5).

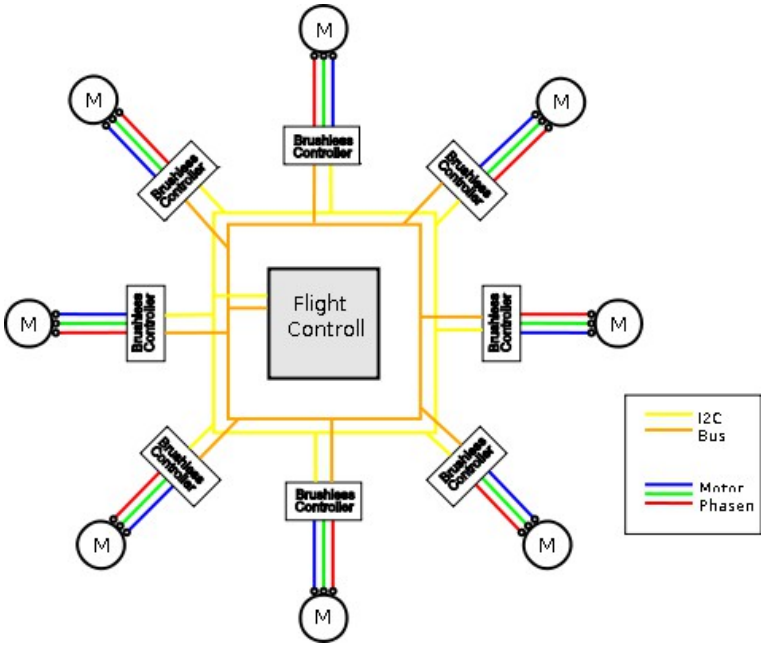


Abbildung 3.5: Verkabelung der Antriebseinheit eines Oktokopter ohne Stromzuführung

Die jeweiligen Devices, für die eine Nachricht bestimmt ist, werden am Beginn des Datagramm binär adressiert. Somit muss jeder Brushless Controller eine eindeutige Adresse besitzen, die vor dem Einbau des Brushless Controller mittels Lötjumper gesetzt werden muss. Die abwechselnden Drehrichtungen der Motoren kann man entweder als Parameter fix im Mikroprozessor des Brushless Controller abspeichern oder am Phasenanschluss der Motoren durch auskreuzen zweier Phasenanschlüsse bestimmen (Abb. 3.5). Noch zu erwähnen ist, das man die maximale Stromzufuhr zu den Motoren ebenfalls als Parameter im Mikrokontroller des Brushless Controller abspeichern kann, sowie die maximale Temperatur, die die MOSFET's erreichen dürfen. Die thermale Obergrenze soll bei 100°C liegen, um eine Zerstörung durch Überhitzung vorzubeugen, da an den MOSFET's keine Kühlkomponenten verbaut sind. Als leistungsregulierende Komponenten müssen die MOSFET's sehr hohe Stromflüsse bedienen und deshalb sind diese Bauteile sehr empfindlich gegenüber Feuchtigkeit. Im

Betrieb sind sie unbedingt vor Feuchtigkeit zu schützen, da sie sonst zerstört werden können und einen Absturz der Mikro-Drohne nach sich ziehen.

Flight Controller

Die Flight Controll ist eines der Herzstücke der Mikro-Drohne. Auf der Flight Controll sind je drei Sensoren für translatorische Beschleunigungen und radiale Bewegungen der Mikro-Drohne verbaut, sowie ein Luftdrucksensor für die relative Höhenmessung. Ebenfalls auf der Flight Controll befinden sich alle wichtigen Anschlüsse für die Peripherie der Mikro-Drohne, welche da wären:

- Remote Controll Sende- Empfangsmodul
- I²C Bus für die Kommunikation mit den Brushless Controllern
- zwei serielle Schnittstellen zur Navigations Controll Komponente
- Anschluss für einen akustischen Signalgeber (Summer)
- vier Servoanschlüsse zur Steuerung von Lastkomponenten, wie zum Beispiel einer Kamera oder anderen Sensoren
- Stromversorgungsanschluss

Zentrales Bauteil auf der Flight Controll bildet ein Mikrocontroller des Herstellers *Atmega* [atmega 1284]. Der Mikroprozessor besitzt im Chip alle benötigten elektronischen Schaltungen um die Sensoren und Schnittstellen der Flight Controll zu bedienen. An den Analog/Digital Wandlern des Mikrocontrollers sind die Sensoren angeschlossen, welche, die von den Sensoren als Stromspannung gelieferten analogen Messwerte, zu digitalen Zahlenwerten für die Weiterverarbeitung im Prozessorkern des Mikrocontroller konvertieren.

Der Luftdrucksensor dient der Flight Controll als Höhenmesser und liefert den aktuell gemessenen Luftdruck gemäß der Barometrischen Höhenformel $P(h)[KPa] = 101.325[kPa] * \exp(-1.29[kg/m^3] * 9.81[m/s^2] * h[m] / 0.101325[kPa])$ [mikrokopter wiki alti] als Gleichspannungswert. Die Spannung am Sensorausgang

nimmt linear mit der Höhe und dem einhergehenden Luftdruck ab (Abb. 3.6).

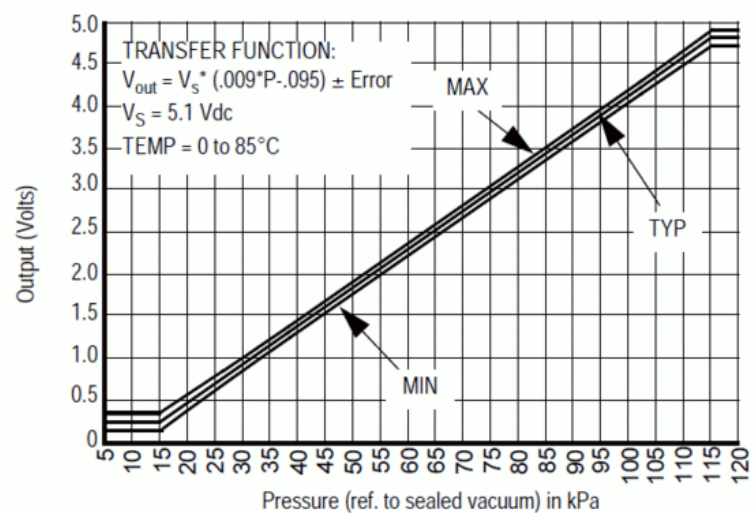


Abbildung 3.6: Funktion der Ausgangsspannung des Luftdrucksensor Bildquelle: [mpx4115]

Durch zusätzliche Operationsverstärker wird eine Auflösung von *20,43 Steps/m* [mikrokopter wiki alti] *erreicht*, was in etwas 5cm entspricht. Die Verstärkerschaltung hat den Nachteil, das die Messwerte nur unterhalb von ~ 1500 Meter plausibel sind, da oberhalb der Spannungsbereich des Höhensensors keine verwertbaren Ergebnisse liefert.

Das Gyroskop, auch Kreiseilinstrument genannt, ist der wichtigste Sensor der Flight Controll. Es dient zur Messung von Winkelgeschwindigkeiten, im Falle der Mikro-Drohne zur Messung der Rotation um eine der drei Raumachsen des Kopters. Bei der Mikro-Drohne ist die X-Achse die Vorwärtsrichtung, die Y-Achse die Seitwärtsrichtung und die Z-Achse die Aufwärtsrichtung. Ein Rotation um die X-Achse bezeichnet man als „Roll“ Bewegung, eine Rotation um die Y-Achse als „Nick“ Bewegung und eine Rotation um die Z-Achse als „Gier“ oder „Yaw“ Bewegung. Auf der Flight Controll ist für jede Raumachse ein Gyroskop verbaut. Ein Gyroskop liefert die aktuelle Winkelgeschwindigkeit einer der Raumachsen der Mikro-Drohne in $°/sec$ als Gleichspannungswert, der an einem Analog/Digital Wandler des Mikrocontroller anliegt. Der Mikrocontroller prozessiert die Information über die Winkelgeschwindigkeiten der drei Raumachsen, um die Mikro-Drohne horizontal und

nach vorwärts zu halten, indem er Steuersignale zu den einzelnen Brushless Controllern der Antriebseinheit sendet. Das auf der Flight Controll verbaute Gyroskop ist vom Typ ein Micro Electro-Mecanical System (MEMS). *Im Gyroskop werden zwei Probemassen durch elektrostatische Anregung zum gegenseitigen Schwingen* $v_a(t)\vec{e}_x$ *entlang der Raumachse X angeregt. Wenn nun eine Drehrate* $\omega\vec{e}_y$ *um Y vorliegt, resultiert aufgrund der Coriolis-Kraft eine Beschleunigung*

$$\vec{a}_c = 2\,\omega\,\vec{e}_y \times v_a(t)\,\vec{e}_x = -2\mathrm{v}_a(t)\,\omega\,\vec{e}_z$$

die eine Schwingung in die Z Achse verursacht. Die Auslenkung der Schwingung wird gemessen und als Drehrate interpretiert [Wendel 2011: S. 62]. Zusätzlich zu den Gyroskopen ist für jede Raumachse je ein Beschleunigungssensor verbaut. Die Beschleunigungssensoren dienen als Stützsensoren für die Gyroskope. Die Beschleunigungssensoren liefern anhand der Gravitationsbeschleunigung die absolute Lage der Mikro-Drohne zum Erdmittelpunkt. Aus den Messdaten der Beschleunigungssensoren wird über einen längeren Zeitraum der Mittelwert berechnet, damit Vibrationsfehler und Fehler, aus Beschleunigungen von Flugmanövern, eliminiert werden können. Die Messdaten der Beschleunigungen in die drei Raumachsen liegen ebenfalls als Gleichspannungswerte an Analog-Digital Wandler Eingänge des Mikrocontrollers an. Ein Beschleunigungssensor ist nach dem Prinzip eines elektrischen Kondensator aufgebaut, wobei der Abstand zwischen den Kondensatorplatten und einer dazwischen gelagerten Probemasse sich durch Beschleunigung verändert (Abb. 3.7). Ein Kondensator speichert eine Ladung Q gemäß der Formel $Q=CU$, wobei C die Kapazität und U die anliegende Spannung ist. *In Ruhelage gilt* $\frac{Q_1}{U_1}=\frac{Q_2}{U_2}$. *Bei Beschleunigung wirkt eine Kraft gemäß* $F=ma$, *die die Probemasse* m *beschleunigt. Im Zusammenhang mit der wirkenden Kraft auf die Probemasse bei unterschiedlichen Ladungen* $F=\frac{Q_1^2}{(2\,\epsilon_0 A)}-\frac{Q_2^2}{(2\,\epsilon_0 A)}$ *mit* A *als Kondenstorfläche und* ϵ_0 *als Dielektrizitätskonstante des Flächenmaterial benötigt*

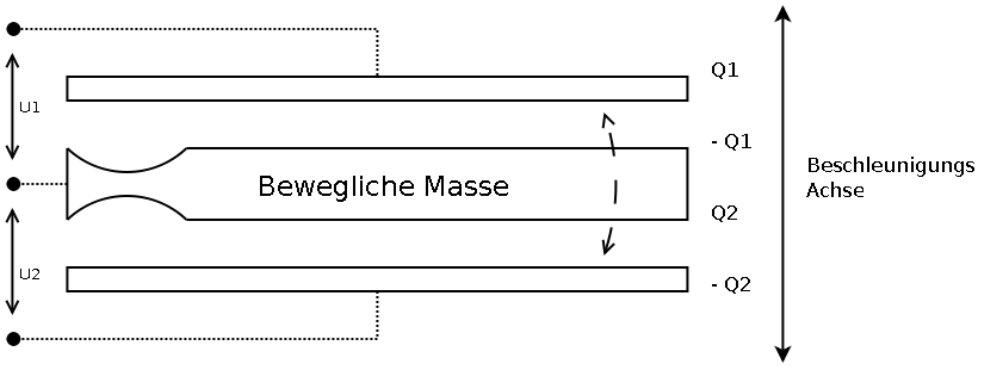


Abbildung 3.7: Prinzip eines Beschleunigungssensor
Bildquelle ähnlich: [Wendel 2011: S. 67]

man eine Ladungsänderung Δq *die eine messbare Spannungsänderung bei Beschleunigung hervorruft.*

Die Beschleunigung ergibt sich aus $a = \frac{(4\,\Delta qQ)}{(2\,m\,\epsilon_0 A)}$ [Wendel 2011: S. 67].

Zur Fernsteuerung der Mikro-Drohne benötigt man ein Sender-Empfängermodul, das mit der Flight Controll verbunden ist, und zu der man von einer Fernsteuerungseinheit Signale schickt. Dies bezeichnet man als RC System (Remote Controll). Die Flight Controll kann zwölf Fernsteuerungskanäle verarbeiten. Vier Kanäle sind für die essentiellen Steuerfunktionen notwendig. Das sind die Rotationsbewegungen um die drei Raumachsen – Nick(Y), Roll(X), Gier(Z), sowie die Steig- und Sinkbewegung, das Gas. Die restlichen acht Kanäle kann man für optionale Funktionen, die später behandelt werden, verwenden. Auf der Fernsteuerungseinheit sind die vier essentiellen Kanäle auf zwei Joysticks gelegt. Die Steuerposition der Joystickachse wird in einem Kanal als zyklisch geradzahliger Byte-wert (Wertebereich [0-254]) mit Puls-Pausen-Modulationstechnik (PPM) übertragen. Für die Funkübertragung der zwölf Kanäle wird das freie 2.4 Gigaherz Frequenzband, welches auch für WLAN genutzt wird, verwendet.

Auf der Flight Controll befindet sich auch der SDA und SCL Lötanschluss des I²C Bus für die Ansteuerung der Brushless Controller. Die Flight Controll ist das Master Device am I²C Bus. Zwei serielle Busanschlüsse sind auf der Flight Controll vorhanden. Einer dient für die Kommunikation mit dem optionalen Navigations Controller (SPI

Interface). Auf dem zweiten seriellen Bus liegen die Telemetrie der Flight Controll an. Hier wird der Telemetrie Sender-Empfänger angeschlossen, oder bei vorhandenem Navigations Controller wird der serielle Bus zum Navigationsmodul durchgeschliffen, an der der Telemetrie Sender-Empfänger angeschlossen wird. Weitere Anschlüsse auf der Flight Controll sind ein Anschluss für einen akustischen Signalgeber, der Information über Betriebszustände liefert wie zum Beispiel Akkumulatoren-Unterspannung oder Fehler am I²C Bus. Außerdem der Anschluss für die Stromversorgung und vier Servoanschlüsse zur Steuerung von Peripherie, wie zum Beispiel eine Kamera oder anderen Sensoren.

Navigations Controller

Der Navigations Controller ist eine optionale elektronische Komponente und erweitert die Funktionalität der Mikro-Drohne um die angeführten Features

- Position Hold
- Coming Home
- Care Free
- autonomer Waypoint Flug
- Follow Me

Um diese erweiterte Funktionalität bedienen zu können, benötigt der Navigations Controller ein Global Position System (GPS) und einen Magnetischen Kompass. Auf dem Navigations Controller wird ein Mikrocontroller mit einem leistungsstarken ARM9 32- Bit Prozessor vom Typ *STR911FAM44X6* [navi arm] für die Dezimal-Koordinatenberechnung mit einer akzeptablen Genauigkeit eingesetzt. Im Falle der Mikro-Drohnen Software wird eine Genauigkeit von sieben Nachkommastellen verwendet. Die erreichte Genauigkeit ist beim Geografischen Gitternetz Koordinatensystem vom Breitengrad abhängig.

Einschub – Abstand zwischen Längengraden und Breitengraden im Geografischen Gitternetz - Rechengenauigkeit

Beim Geografischen Koordinatensystem als Gitternetz verlaufen die Breitengrade als Breitenkreise parallel zum Äquator auf einer gedachten Kugel hin zum Nord- und Südpol. Der Abstand zwischen zwei Breitengraden beträgt $\Delta grad_{lat} = \frac{U_{erde}}{360}$ und aufgrund der Abplattung der Erde an den Polen liegt der Abstand zwischen $110.574\,km/° \; und \; 111.694\,km/° \rightarrow \Theta = 111.134\,km/°$ [wikipedia lat]. Die Längengrade verlaufen zwischen Nordpol und Südpol. Der Abstand zwischen zwei nebeneinander liegenden Längengraden verjüngt sich zu den Polen hin gegen Null. Dies entspricht einer Kosinusfunktion. Daraus folgt (Abstandskonstante als arithmetischer Mittelwert der obigen Werte):

$$\Delta d_{lat}(meter) = (lat_a - lat_b) * 111134$$

$$\Delta d_{lon}(meter) = (lon_a - lon_b) * 111134 * \cos(lat)$$

Unter der Annahme, dass das Experiment bei $48.32\,°N$ und $14.29\,°E$ (Linz) stattfindet liegt die Genauigkeit bei sieben Nachkommastellen in Dezimalgrad bei zirka $0.01\,Meter$ im Zentimeterbereich.

Die für diese Arbeit wichtigste Funktionalität des Navigation Controller ist der autonome Wegpunkteflug der Mikro-Drohne. Dabei werden die Koordinaten von attributierten Wegpunkte n an den Navigations Controller übermittelt, der den Flight Controller ansteuert, um Koordinaten sequenziell anzufliegen. Bei der Funktion Position Hold versucht die Mikro-Drohne die aktuelle Koordinate zu halten. Die Funktion Care Free definiert einen Virtuellen Vorwärtspunkt der Mikro-Drohne, so das die Vorwärtsrichtung der Drohne immer vom Startpunkt weg zeigt, egal welche aktuelle Ausrichtung durch Gier-Bewegungen die Mikro-Drohne hat, weil der Rahmen um die Z-Achse rotationssymmetrisch ist. Coming Home bewirkt eine autonome Rückkehr der Mikro-Drohne zu den Start-Koordinaten. Die Follow Me Funktion ist eine Erweiterung, bei der die Mikro-Drohne einem bewegten GPS Sender folgt. Auf diese Funktion wird

nicht weiter eingegangen. Als Schnittstellen verfügt der Navigations Controller über zwei serielle Interfaces, die mit der Flight Controll verbunden sind. Eines zur Kommunikation mit der Flight Controll (SPI) und eines, an dem die Telemetriedaten von der Flight Controll durchgeschliffen werden. An einem dritten seriellen Interface ist das Sender-Empfänger Modul für die Telemetrie Daten befestigt. Das Sender-Empfänger System für die Telemetrie besteht aus zwei baugleichen Modulen und überträgt ein Standard serielles RS.232 Signal mit 57600 Baud im lizenzfreien 868Mhz-870Mhz Funkband. Das gegenseitige Sender-Empfänger Modul wird an einem PC angeschlossen, somit besteht eine serielle Funkverbindung zwischen Computer und der Mikro-Drohne. Dieser Kommunikationskanal zur Mikro-Drohne wird mit der proprietären Mikrokopter Software verwendet, kann aber auch für eigene Entwicklungen benutzt werden, und wird in dieser Arbeit zur Wegpunkt-Übertragung aus dem Geoinformationssystem an die Mikro-Drohne verwendet. Am Navigations Controller ist auch ein Einschubmodul für eine Mikro SD Memory Card angebaut. Bei eingelegter Speicherkarte werden die GPS Koordinaten und weiterer Telemetriedaten der Mikro-Drohne beim Betrieb im GPX Format (Tabelle 4.11 und Tabelle 4.12) und im KML Format aufgezeichnet, und stehen für ein Postprozessing des Mikro-Drohnen Einsatz zur Verfügung.

Die obig erklärten Funktionen des Navigations Controller benötigen Informationen des Raumbezuges der Mikro-Drohne anhand geografischer Koordinaten und die Information der Ausrichtung der Mikro-Drohne im Raum anhand des Erdmagnetfeldes. Diese Informationen liefern das GPS Modul und das Magnetische Kompass Modul. Die Informationen dienen auch als Stützinformation für den Flight Controller. So liefert das Kompass Modul Stützdaten über die Vorwärts-Ausrichtung der Mikro-Drohne in der horizontalen Ebene und dient der Gier-Stabilisierung.

Magnetic Compass Modul

Das Magnetic Compass Module, weiter hier als Kompass Modul bezeichnet, besitzt einen drei Achsen Magnetfeldsensor (X, Y, Z Raumachse), der von einem am Kompass

Modul verbauten Mikrokontroller bedient wird. Der Anschluss an den Navigations Controller erfolgt über den I²C Bus des Navigation Controller als Master-Device, wobei die magnetischen Werte des Erdmagnetfeldes in drei Raumachsen aufgeteilt im I²C-Datagramm an den Mikrokontroller des Navigations Controller übermittelt werden.

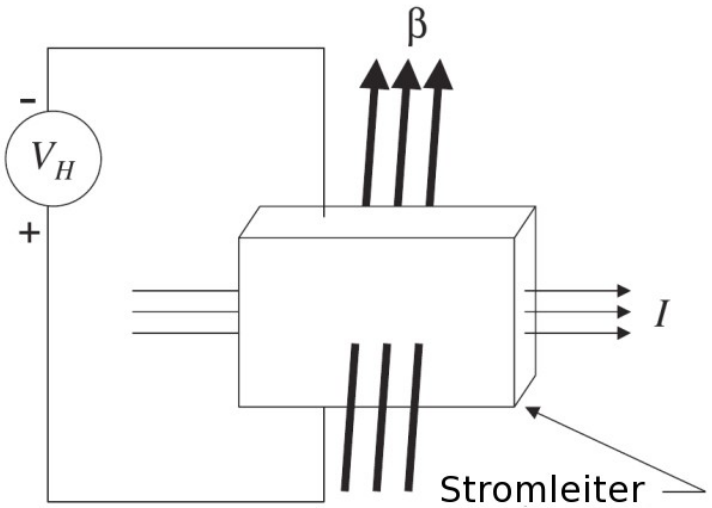


Abbildung 3.8: Halleffekt Bildquelle:[Nyce 2004: S.110]

Das Kompass Modul muss genau parallel zum Navigations Controller liegen, damit nach der Justierung des Kompass Moduls die Werte der drei Raumachsen plausible Ergebnisse liefern. Weiters ist darauf zu achten, das sich keine magnetischen Störquellen im Aufbau der Mikro-Drohne befinden, die den Magnetsensor stören. So sind die drei Phasenanschlüsse zu den Motoren möglichst parallel ohne Schleifen zu verbauen, damit sich die durch Stromfluss entstehenden Magnetfelder gegenseitig aufheben. Der akustische Signalgeber muss möglichst weit außerhalb am Rahmen an einem Ausleger montiert werden. Zur Messung des Erdmagnetfeldes sind am Kompass Modul drei Halleffekt-Sensoren senkrecht zu den drei Raumachsen verbaut. Beim Hall-Effekt wird bei einem mit Strom durchlaufenen Leiter auf dem senkrecht ein Magnetfeld wirkt eine Spannung V_h indiziert (Abb 3.8) die proportional zu der (erd)magnetischen Flussdichte β ist.

$$V_h = C_h \frac{\beta I}{d}$$ [Nyce 2004: S.111]

GPS Modul

Das GPS Modul der MikroDrohne ist ein integriertes GPS Empfangsmodul vom Typ *LEA-4H des Hersteller µBlox* [µBlox LEA-4H]. Es ist über eine serielle Schnittstelle mit dem Mikrokontroller des Navigations Controller verbunden und kommuniziert mit dem Hersteller eigenen proprietären UBX Protokoll. Der Aufbau des UBX Datagramm hat folgende Struktur:

Datagramm struktur	SYNC	CLASS	ID	LENGTH	PAYLOAD	CHECKSUM
Erklärung	Synchronisation besteht immer aus 0xB5, 0x62	Nachrich-tenklasse	Nachrichten-inhaltscode (Kommando)	Länge des Payload als zwei Byte Integer Wert	Nachricht	Prüfsumme des Datagramm ohne SYNC und CHECKSUM
Länge des Feld in Byte	2	1	1	2	Variable	2

Tabelle 3.1: UBX Datagramm Aufbau Quelle: [µBlox ubx]

Es sind acht Nachrichtenklassen im UBX Protokoll spezifiziert um mit dem integriertem GPS Modul zu kommunizieren. Diese Nachrichtenklassen decken alle benötigten Funktionen ab, um das GPS Modul zu konfigurieren, wie zum Beispiel, setzen des Bezugssystem WGS84, die Inbetriebnahme und GPS Koordinaten abzufragen. Für eine detaillierte Erklärung aller Nachrichtenklassen und Kommandos zu diesen Klassen sei auf die UBX Protokoll Spezifikation verwiesen. Die Initialisierung und die Abfrage der Koordinaten wird vom Mikrocontroller des Navigations Controller erledigt.

Das einfache mathematische Modell der GPS Positionsbestimmung basiert auf der vom pythagoräischen Lehrsatz hergeleiteten Kugeloberflächengleichung $(x-x_0)^2+(y-y_0)^2+(z-z_0)^2=r_0^2$, die besagt, dass eine Kugeloberfläche die Menge aller Punkte x, y, z ist mit der Entfernung r_0 zu seinem Mittelpunkt x_0, y_0, z_0 . Ein GPS Satellit befindet sich auf einer Umlaufbahn. Seine bekannten Koordinaten x_0, y_0, z_0 sind der gedachte Kugelmittelpunkt. und die Entfernung zu einem GPS

Empfänger ist r_0 . Die gesuchten Koordinaten eines GPS Empfängers sind x_e, y_e, z_e , die auf der Kugeloberfläche liegen, so gilt $(x_e-x_0)^2+(y_e-y_0)^2+(z_e-z_0)^2=r_0^2$. Die Entfernung r_0 ergibt sich aus der Laufzeit mit Lichtgeschwindigkeit des vom Satelliten ausgestrahlten Funksignals. Die Laufzeit des Funksignals ist immer von externen Einflüssen, wie atmosphärische Störungen, relativistische Effekte und anderen Faktoren, fehlerbehaftet und benötigt einen Korrekturwert K mit dem wir die Gleichung erweitern.

$$(x_e-x_0)^2+(y_e-y_0)^2+(z_e-z_0)^2=(r_0+K)^2$$

Somit haben wir eine Gleichung mit vier unbekannten Variablen x_e, y_e, z_e, r_0 . Wenn wir vier Satelliten mit unterschiedlichen Koordinaten und exakt synchroner Zeit haben, können wir dieses lineare Gleichungssystem lösen, erhalten allerdings zwei Lösungen, da es sich um eine quadratische Gleichung handelt. Eine dieser beiden Lösung liegt immer oberhalb des wirklichen Standortes des Empfängers oder der Zeitpunkt der Ausstrahlung des Funksignals ist nicht synchron zu den drei anderen Austrahlungs-Zeitpunkten und man kann dieses Ergebnis eliminieren. Zur eindeutigen Positionsbestimmung des GPS Empfänger benötigen wir daher mindestens fünf Satelliten um die quadratische Gleichung eindeutig lösen zu können. Für eine vertiefende Theorie der GPS Technik sei auf die Literatur verwiesen.

3.2. Inertiale Navigation – Funktion und Theorie

3.2.1. Grundlagen

Die autonome Steuerung von Unmanned Aerial Vehicles - UAV ist von den im Fluggerät verbauten Raumlage Sensoren sowie von der Qualität der von diesen Sensoren gelieferten Messwerten direkt abhängig. Unter Qualität von Messdaten versteht man die Messgenauigkeit akkurat und über längeren Zeitraum sowie den Messintervall. Das ist die Anzahl aktueller verwertbaren Sensorenwerte pro Zeiteinheit, der je größer umso qualitativ hochwertiger ist. Um einen autonomen Flug über einen längeren Zeitraum abzuwickeln, benötigt man ein auf physikalische mathematische

Gesetze aufbauendes Algorithmengerüst, das die von den Raumsensoren anfallenden Messdaten bewertet, und als Grundlage für sinnvolle Steuerbefehle zum Erreichen eines vorgegebenen Ziels fortlaufend prozessiert. Dies ist die Inertiale Navigation oder auch als Trägheitsnavigation bezeichnet. Die Grundprinzipien der Inertialen Navigation sind die fortlaufenden Messungen der Bewegungen eines sich im Raum frei bewegenden Körpers anhand von Beschleunigungskräften, und die Berechnung der aktuellen räumlichen Lage und Geschwindigkeit des Körpers anhand der gewonnenen Messwerte. Dieser Prozess wird iterativ fortgesetzt und wenn möglich durch vorhandene Stützdaten, wie zum Beispiel GPS Koordinaten, korrigiert.

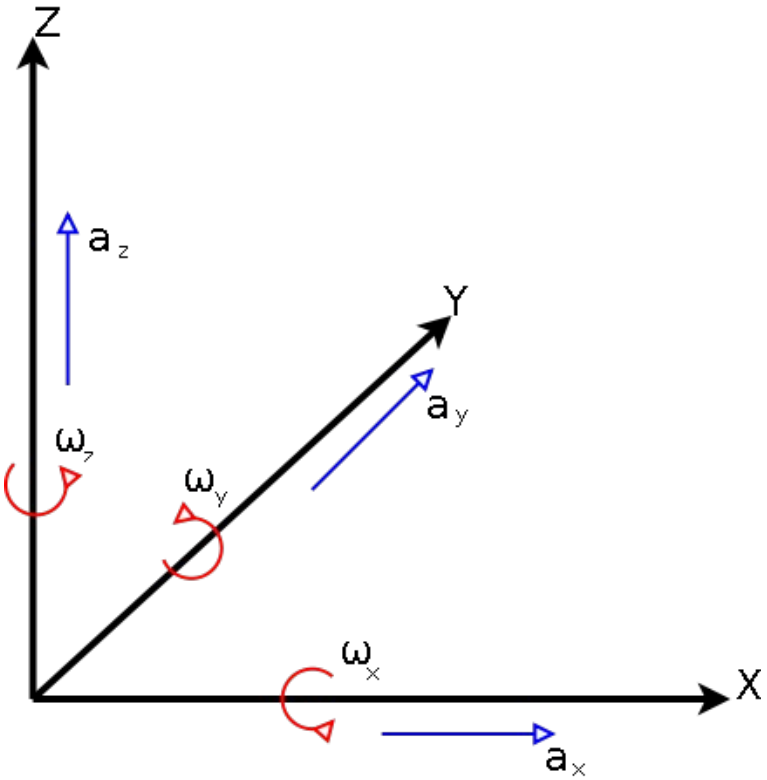


Abbildung 3.9: Kinematische Freiheitsgrade eines frei beweglichen Körper Bildquelle: [6kinematik]

Diesen iterativen Prozess, bei dem für die Berechnung der aktuellen Raumlage, die aktuellen Sensorenwerte und die Raumlage und der Geschwindigkeitsvektor des vorhergehenden Berechnungszykluses einfließen müssen, bezeichnet man auch als Koppelnavigation. Ein im dreidimensionalen Raum frei bewegliche Körper besitzt sechs

kinematische Freiheitsgrade, drei translatorische Freiheitsgrade entlang den Raumachsen und drei radiale Freiheitsgrade um die Raumachsen. Beschleunigungssensoren messen Geschwindigkeitsänderungen der translatorischen Bewegungen und Gyroskope messen Änderungen der Winkelgeschwindigkeit einer Rotation um eine Raumachse (Abb. 3.9). Anhand der aktuellen bekannten Position, Geschwindigkeit und der Beschleunigung auf einen Körper, kann man durch das Weg-Zeit Gesetz die neue Geschwindigkeit und die zurückgelegte Distanz ermitteln. Aus der einfachen Integration der Beschleunigungen über die Zeit resultiert die aktuelle Geschwindigkeit des Körpers, und die zweifache Integration über die Zeit liefert die zurückgelegte Distanz und somit die Position des Körpers. Die gemessene Beschleunigung und der Zeitintervall ist bekannt.

$$s(t)=\iint a(t)\,dt^2 \qquad v(t)=\int a(t)\,dt \quad \text{[6kinematik]}$$

Durch die doppelte Integration des Beschleunigungswertes würde ein ungenauer Zeitwert, oder schlechte Beschleunigungswerte sehr schnell zu falschen Ergebnissen führen. Daher sind ein präziser Zeitgeber und genaue Beschleunigungssensoren notwendig. Die Gyroskope messen die aktuelle Winkelgeschwindigkeit . Eine einfache Integration dieses Wertes über die Zeit liefert den Winkel zu der jeweiligen Raumachse.

$$\phi(t)=\int \omega(t)\,dt \quad \text{[6kinematik]}$$

Anhand dieser Grundlagen kann man einen sogenannten „Strapdown Algorithmus“ für die Inertiale Navigation entwickeln. Der Strapdown Algorithmus besteht im wesentlichen aus drei Schritten, die sich iterativ wiederholen. Bekannte Startwerte des Algorithmus sind die Winkellage ϕ , Geschwindigkeit v und Ort s eines frei beweglichen Körpers im Raum. Der Einfachheit halber werden die Werte mit 0 initialisiert (Abb. 3.10). Der Ort wird anhand mehrerer zueinander in Beziehung stehender Koordinatensysteme beschrieben. Wenn der Algorithmus startet, werden zu diskreten Zeitpunkten die Werte der Gyroskope und der Beschleunigungssensoren ausgelesen. Die durch Integration ermittelten aktuellen Werte der Winkellage, Geschwindigkeit und zurückgelegten Distanz, fließen in die Berechnung der aktuellen

Position in den Koordinaten des Raummodells ein. Die letzten ermittelten Werte dienen als Eingangsparameter für den nächsten Rechenzyklus.

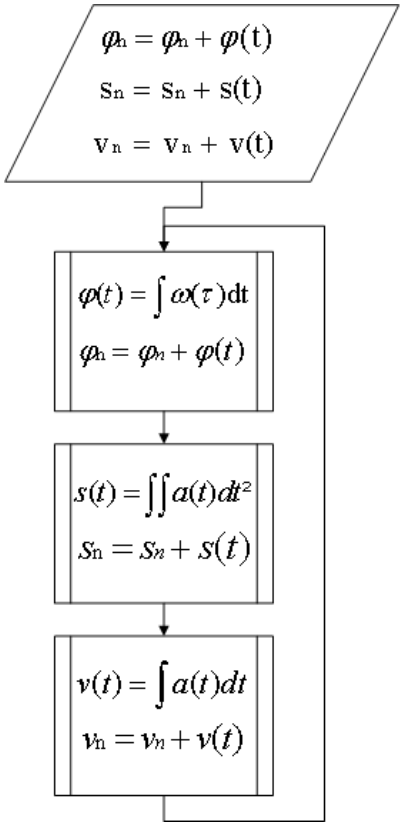


Abbildung 3.10: Prinzip des Strapdown Algorithmus

Das Modell des Strapdown Algorithmus ist in der Realität wesentlich komplexer, da eine Berechnung im dreidimensionalen Raum natürlich jeweils einen Vektor aus der Summe der einzelnen Raumvektorwerten ergibt. Weiters wirken noch externe und aerodynamische physikalische Kräfte auf den Körper, wie Gravitation, Fliehkräfte und Corioliskraft, die in der Berechnung berücksichtigt werden müssen. Auch Korrekturen, aus der, während der Flugzeit stattfindenden Erdrotation fließen in die Berechnung der Koordinaten ein.

3.2.2. Koordinaten und Bezugssystem WGS84

Dem Strapdown Algorithmus liegen wie schon oben erwähnt verschiedene Koordinatensysteme zu Grunde.

- *Das Körperfeste Koordinatensystem (b-frame) ist fest verankert mit dem beweglichen Körper. Die Raumachsen sind zueinander orthogonal und haben ihren Ursprung im beweglichen Körper, meist dem Schwerpunkt. Die x^b Achse weist nach vorne, die y^b Achse nach rechts und die z^b Achse nach unten. Die Sensoren sind fest entlang der Raumachsen der Körperfesten Koordinaten ausgerichtet. In diesem Koordinatensystem erfolgt die Messung der Inertialsensoren.*
- *Das Navigation Koordinatensystem (n-frame), dessen Ursprung ebenfalls im beweglichen Körper liegt und mit dem Ursprung des Körperfesten Koordinatensystems zusammenfällt. Die x^n Achse weißt nach Norden und die y^n Achse weist nach Osten und sie liegen orthogonal zur Schwerbeschleunigung des Erdgravitationsfeldes. Die z^n Achse weist parallel zur Schwerbeschleunigung des Erdgravitationsfeldes.*
- *Das Inertialkoordinatensystem (i-frame), dessen Ursprung fällt mit dem Erdmittelpunkt zusammen. Seine x^i und y^i liegen in der Äquatorebene der Erde und die z^i Achse liegt parallel zur Rotationsachse der Erde. Die Koordinatenachsen sind fix ausgerichtet. Die Messung der Inertialsensoren an dem Körperfesten Koordinatensystem werden bezüglich dem Inertialkoordinatensystem getätigt.*
- *Das Erdfeste Koordinatensystem (e-frame) dessen Ursprung im Erdmittelpunkt liegt. Die x^e und y^e Achse sind an der Äquatorebene ausgerichtet, jedoch fix in Bezug zur Erdrotation. Die z^e Achse fällt mit der Rotationsachse zusammen und deckt sich mit der z^i Achse des Inertialkoordinatensystems. Dieses Koordinatensystem ist äquivalent zum Geografischen Koordinatensystem und wird als ECEF- earh centered, earth fix bezeichnet.* [Wendel 2011: S. 28-29]

Somit ergeben sich zwei Klassen mit je zwei Koordinatensystem Ursprünge. Die Klasse deren Ursprung der Raumkoordinaten im frei beweglichen Körper liegt und zueinander

auf allen drei Freiheitsgraden beweglich ist und die Klasse deren Ursprung im Erdmittelpunkt liegt und mit zwei Freiheitsgraden, der x und y Achsen zueinander beweglich sind. Hierarchisch betrachtet bewegt sich der Ursprung des n-frame und b-frame als Vektor in den Koordinatensystemen des i-frame und e-frame. Die Umrechnung des Vektors zwischen den Koordinatensystemen geschieht mittels Matrizen. Um die erste Klasse, die des frei beweglichen Körpers, im Erdfesten Koordinatensystem richtig referenzieren zu können, ist ein Geodätisches Datum notwendig.

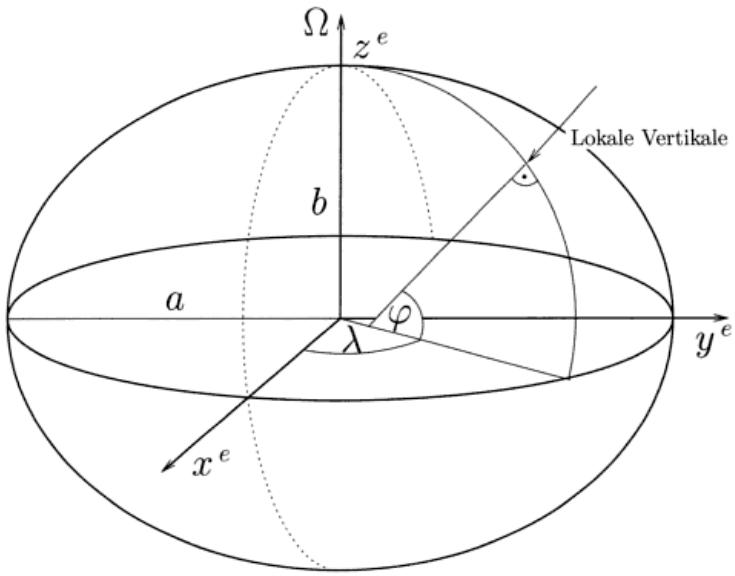


Abbildung 3.11: Erdellipsoid Bildquelle:[Wendel 2011: S. 30]

Das für die Inertiale Navigation zugrundeliegende und verwendete Geodätische Datum ist das World Geodetic System 1984 – WGS84. Es ist ein einheitliches Datum für Georeferezierung auf der Erdoberfläche und auch im erdnahen Weltraum. In der physikalischen Wirklichkeit gleicht die Erde einem Geoid, das durch unsymmetrische Oberflächen und verschiedene Dichten des Erdkörpers und Gravitationsabweichungen hervorgerufen wird. Um eine einheitlich annähernd genaue Georeferenzierung in einem Koordinatensystem zu ermöglichen, wurde versucht ein annähernd genaues

Rotationsellipsoid, das Erdellipsoid, zu definieren. Die Datumsdefinition des WGS84 bestimmt den Masseschwerpunkt der Erde als Ursprung der x_0^e , y_0^e und z_0^e Achsen. Die z^e Achse liegt parallel zur Rotationsachse der Erde. Die x^e Achse schneidet den Nullmeridian von Greenwich an der Äquatorebene und die y^e Achse liegt orthogonal nach Osten weisend dazu (Abb. 3.11). Die dem WGS84 zugrundeliegenden Parameter des Erdmodells als Rotationsellipsoid sind in Tabelle 3.2 definiert. Das WGS84 ist als kartesisches Koordinatensystem ECEF definiert. Um es in die gebräuchliche Longitude, Latitude und Höhe Nomenklatur anzugeben bedarf es einer Koordinatentransformation. Die Koordinatentransformation von der Longitudinale, Latitude und Höhe Nomenklatur in die ECEF Nomenklatur ist einfach möglich nach:

Große Halbachse $a = 6378137\ Meter$
Kleine Halbachse $b = 6356752.3142\ Meter$
Abflachung $f = \frac{a-b}{b} = \frac{1}{298.2572235663}$
Exzentrizität $e = \sqrt{f(2-f)} = 0.0818191908426$
Nord-Süd Krümmungsradius $R_{ns} = a\frac{1-e^2}{(1-e^2\sin^2\phi)^{2/3}}$
Ost-West Krümmungsradius $R_{ow} = \frac{a}{\sqrt{(1-e^2\sin^2\phi)}}$
Durchschnittlicher Krümmungsradius $\sqrt{R_{ns}R_{ow}}$
Erddrehrate $\Omega = 7292115 * 10^{-11}\frac{rad}{sek}$

Tabelle 3.2: Parameter WGS84 [Wendel 2011: S. 31]

$$x_e = \left(\frac{a}{\sqrt{1-\frac{a^2-b^2}{a^2}\sin^2\phi}} + h\right)\cos(\phi)\cos(\lambda)$$

$$y_e = \left(\frac{a}{\sqrt{1-\frac{a^2-b^2}{a^2}\sin^2\phi}} + h\right)\cos(\phi)\sin(\lambda)$$

$$z_e = \left(\frac{a}{\sqrt{1-\frac{a^2-b^2}{a^2}\sin^2\phi}}\left(1-\frac{a^2-b^2}{a^2}\right) + h\right)\sin(\phi)$$

[Bauer 2003: S. 76]

Für die Transformation von ECEF x, y, z kartesische Koordinaten in die ϕ, λ, h Geografische Koordinaten ist nicht trivial, da es keine direkte Lösung dafür gibt. Es gibt eine Reihe von Näherungslösungen, die mehr oder weniger komplex sind, und deren Fehler hinreichend klein ist. Ein Vergleich der Methoden mit deren maximalen Fehler und relativen Berechnungsgeschwindigkeitsindex ist in Tabelle 3.3 angeführt.

Berechnungs methode	Maximaler Fehler					Geschwin-digkeit
	φ (sec)	h (m)	x(m)	y(m)	z(m)	
Lin&Wang	6.87E-011	2.53E-009	9.31E-010	9.31E-010	1.86E-009	42
Bowring	2.88E-008	1.01E-006	9.31E-010	9.31E-010	1.32E-006	50
Paul	3.90E-007	1.14E-006	9.31E-010	9.31E-010	1.21E-005	52
Ozone	6.87E-011	2.42E-009	9.31E-010	9.31E-010	2.33E-009	56
Iteration	1.35E-005	6.12E-005	9.31E-010	9.31E-010	4.19E-004	62
Borkowski	2.88E-008	1.01E-006	9.31E-010	9.31E-010	1.32E-006	63

Tabelle 3.3: Berechnung ECEF nach Geografischen Koordinaten [Gerdan and Deakin 1999: S. 11]

Um mit den Beschleunigungen und Drehraten in den vier Koordinatensystemen arbeiten zu können muss man zwischen den Koordinatensystemen konvertieren. Dies wird mit der mathematischen Disziplin der Matrizenrechnung bewerkstelligt. Die Nomenklatur und die Rechenregeln für das Arbeiten in den Koordinatensystemen wird hier kurz vorgestellt. Die Schreibweise für einen Vektor in den Koordinaten ist

$\overrightarrow{v_{eb}^{n}}$ *Nomenklatur*

*Der Ausdruck bezeichnet einen Geschwindigkeitsvektor, das ist eine eindimensionale Matrix mit den drei Elementen, hier mit den Koordinaten des Navigationskoordinatensystem (**n**-frame) im oberen Indizes. Der untere Indizes gibt an, dass die Geschwindigkeit des Körperfesten Koordinatensystem (**b**-frame) bezüglich des Erdfesten Koordinatensystem (**e**-frame) angegeben wird. Weites gilt die Umkehr und die Verkettung in den Koordinatensystemen. Diese Rechneregeln gelten auch für die Drehraten* ω *der Lage*

$$\vec{v}^{n}_{eb} = -\vec{v}^{n}_{be} \quad ; \quad \vec{\omega}^{n}_{eb} = -\vec{\omega}^{n}_{be}$$

$$\vec{v}^{n}_{eb} = \vec{v}^{n}_{ei} + \vec{v}^{n}_{ib} \quad ; \quad \vec{\omega}^{n}_{eb} = \vec{\omega}^{n}_{ei} + \vec{\omega}^{n}_{ib}$$

Umkehr und Verkettung

Um zwei Vektoren zu verknüpfen bedient man sich der antisymmetrischen oder schiefsymmetrischen Matrix $\boldsymbol{A}$ *um mit der allgemein gültigen Matrizen-Mathematik arbeiten zu können.*

$$\text{Gegeben } \vec{a} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} \text{ und } \vec{b} = \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} \text{ dann gilt}$$

$$\vec{a} \times \vec{b} = \boldsymbol{A}\vec{b} \text{ für } \boldsymbol{A} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_z & 0 \end{bmatrix} \text{ und } \boldsymbol{A} = [\vec{a}\, X]$$

Für die antisymmetrische Matrix einer Drehrate $\boldsymbol{\Omega^{n}_{eb}} = [\omega^{n}_{eb}\, X]$ *gilt auch die Umkehr und Verkettungsregel. Für das Transformieren eines Vektors zwischen zwei Koordinatensystemen braucht man eine Richtungskosinusmatrix* $\boldsymbol{C^{n}_{b}}$ *und transformiert nach der Regel* $\vec{\omega}^{n}_{ei} = \boldsymbol{C^{n}_{b}} \cdot \vec{\omega}^{b}_{ei}$ *hier beispielhaft vom n-frame in den b-frame. Für die Koordinatentransformation von antisysmetrischen Matrizen von Vektoren gilt* $\boldsymbol{\Omega^{n}_{ei}} = \boldsymbol{C^{n}_{b}} \cdot \boldsymbol{\Omega^{b}_{ei}} \cdot \boldsymbol{C^{b}_{n}}$ *. Mit der antisymmetrischen Matrix und der Richtungskosinusmatrix kann man nun Vektoren zwischen beliebigen Koordinatensystemen transformieren. Da die Richtungskosinusmatrix orthogonal ist, ist*

die Inverse ihrer Transponierten $\boldsymbol{C}_{\boldsymbol{b}}^{\boldsymbol{n}}=\boldsymbol{C}_{\boldsymbol{n}}^{\boldsymbol{b},-1}=\boldsymbol{C}_{\boldsymbol{n}}^{\boldsymbol{b},T}$. *Es gilt auch hier die Verkettungsregel. Für die Herleitung der Richtungskosinsusmatrix sei auf die Literatur verwiesen.* [Wendel 2011: S. 34-36]

3.2.3. Berechnungen anhand einer Mikro-Drohne

Bei einer Mikro-Drohne spricht man von einem unteraktuierten System. Dies bedeutet, das weniger in die Freiheitsgrade der Mikro-Drohne wirkende Aktuatoren verbaut sind, als Freiheitsgrade vorhanden. Die Mikro-Drohne besitzt sechs Freiheitsgrade doch die Antriebseinheiten wirken eigentlich nur in einem Freiheitsgrad, nach unten. Diese Art des Antriebsaufbaus hat den Vorteil, das die Mikro-Drohne schwebe flugfähig ist, doch besitzt den Nachteil einer innewohnenden Instabilität. Somit ist die Steuerung der Mikro-Drohne ein Sonderfall unter der Inertialen Navigation. Ergänzend zu den Inertialsensoren besitzt die Mikro-Drohne einen barometrischen Höhenmesser, einen GPS und Erdmagnetfeldsensor, deren Daten in die Navigationslösung mit einfließen. Die hier behandelte Mikro-Drohne besitzt acht Antriebseinheiten von denen Schübe, das entspricht einer Beschleunigung der Luftmassen durch die Propeller, in die Z-Achse des Körperfesten Koordinatensystem (b-frame) gegen das Erdschwerfeld des Inertialen Koordinatensystem eine Geschwindigket im Inertialen Koordinatensystem (i-frame) erzeugt werden.

$$\overrightarrow{v_{ib}^{i}} = (\boldsymbol{C}_{\boldsymbol{b}}^{\boldsymbol{i}}\overrightarrow{a_{ib}^{b}})+\overrightarrow{g_{z}^{i}} = \frac{1}{m}\boldsymbol{C}_{\boldsymbol{b}}^{\boldsymbol{i}}\begin{bmatrix} 0 \\ 0 \\ -F_1 \;\; -F_2 \;\; \dots \;\; -F_8 \end{bmatrix}+\begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}$$ [Wendel 2011: S. 288]

Bei der Rotation ist es komplizierter. Hier wirken Drehmomente der Rotoren und unter Umständen das Drehmoment des Flugkörpers. Analog zur Translation ist Drehmoment das Produkt aus Trägheitsmoment und Winkelgeschwindigkeit. Der Drehimpuls ist das Integral des Drehmoments über die Zeit. $\vec{M}=J\,\vec{\omega}$ *und* $\vec{L}=\int \vec{M}\,dt \;\rightarrow\; \vec{M}=\frac{d\,\vec{L}}{dt}$

Man betrachte der Einfachheit halber das Drehmoment in den Körperfesten Koodinaten (b-frame) im Bezug zum Inertial Koordinatensystem, in dem die

Antriebskräfte wirken.

$$\overrightarrow{M^b}=\boldsymbol{C_i^b}\frac{d\,\overrightarrow{L^i}}{dt} \quad und \quad \overrightarrow{L^i}=\boldsymbol{C_b^i}\overrightarrow{L^b} \;\rightarrow\; \frac{d\,\overrightarrow{L^i}}{dt}=\boldsymbol{C_b^i}\boldsymbol{\Omega_{ib}^b}\overrightarrow{L^b}+\boldsymbol{C_b^i}\overrightarrow{L^b} \;\rightarrow\; \overrightarrow{M^b}=\boldsymbol{\Omega_{ib}^b}\overrightarrow{L^b}+\overrightarrow{L^b}$$

Bei einem Trägkeitsmoment $\boldsymbol{J_F^b}$ *folgt die Eulerische Kreiselgleichung*

$$\overrightarrow{M^b}=\boldsymbol{\Omega_{ib}^b J_F^b}\overrightarrow{\omega_{ib}^b}+\boldsymbol{J_F^b}\overrightarrow{\omega_{ib}^b}$$ [Wendel 2011: S. 289-291]

Für die weitere Herleitung siehe [Wendel 2011: S. 289-291]. Durch den Aufbau der Antriebseinheiten als Oktokopter (Abb. 3.1) ergibt sich der Zusammenhang zwischen Auftrieb und Drehrate um die drei Raumachsen:

X Achse (Roll): $\omega_{ib,x}^b=\omega_{ib,y}^b\omega_{ib,z}^b\frac{J_y-J_z}{J_x}+\frac{(F_8+F_7-F_3-F_4)l}{J_x}$

Y Achse $(Nick)$: $\omega_{ib,y}^b=\omega_{ib,x}^b\omega_{ib,z}^b\frac{J_z-J_x}{J_y}+\frac{(F_1+F_2-F_5-F_6)l}{J_y}$

Z Achse $(Gier)$: $\omega_{ib,z}^b=\omega_{ib,x}^b\omega_{ib,y}^b\frac{J_x-J_y}{J_z}+\frac{(F_2+F_4+F_6+F_8-F_1-F_3-F_5-F_7)lc}{J_z}$

Um nun die Mikro-Drohne frei im Raum zu bewegen sind vier Steuergrößen notwendig. Ein Nick bewegt die Mikro-Drohne entlang der X Achse, ein Roll entlang der Y Achse, das Gier erzeugt eine Rotation um die Z Achse und Gas bewirkt ein Steigen und Sinken der Mikro-Drohne. Dies ergibt vier Steuergrößen.

Steigen: $u_1=F_1+F_2+F_3+F_4+F_5+F_6+F_7+F_8$

Roll: $u_2=F_8+F_7-F_3-F_4$

Nick: $u_3=F_1+F_2-F_5-F_6$

Gier: $u_4=F_2+F_4+F_6+F_8-F_1-F_3-F_5-F_7$

Somit ergibt dies das Bewegungsmodell der Mikro-Drohne mit den vier Stellgrößen $\overrightarrow{v_{ib}^i}$, $\omega_{ib,x}^b$, $\omega_{ib,y}^b$, $\omega_{ib,z}^b$ für die Implementierung in der Steuereinheit.

3.2.4. Kalman Filter und Sensorfusion

Das Kalman Filter ist ein stochastisches Filter für lineare Gleichungssysteme. Es wird verwendet um Messfehler, welche in Messrauschen und Systemrauschen ihren Ursprung haben, zu filtern. Das Messrauschen hat seinen Ursprung in der Ungenauigkeit von Sensoren und ist mehr oder weniger ausgeprägt. Das Systemrauschen hat externe Ursachen. Das Kalman Filter besteht aus mehren mathematischen Gleichungen, die den erwarteten Messwert anhand des vorhergehenden Messzyklus abschätzt. Das Kalman Filter besitzt aufgrund der Einbeziehung des vorhergehenden Messzyklus und der vorhergehenden Schätzung eine iterative Struktur und kann einfach in einem Algorithmus implementiert werden. Man unterscheidet im Kalman Filter zwischen Prädikation und Korrektur. Bei der Prädikation tätigt man eine Schätzung über den Systemzustand in der Zukunft $\widehat{x_{k+1}^-}$ die a priori Schätzung. Danach wird der Schätzwert anhand der tatsächlichen Messung unter Einbeziehen aller bekannten Messparameter, wie zum Beispiel Standartabweichung der Messungen, korrigiert. Diesen Wert bezeichnet man als posteriori Schätzwert $\widehat{x_k^+}$. Es ergibt sich daraus der Iterative Prozess aus drei Schritten

- Prädikation des Systemzustandes
- Messung des Systemzustandes
- Korrektur der Schätzung anhand der Messung

Zur Erstellung eines Kalman Filter geht man von einer linearen zeitdiskreten Differentialgleichung aus , wie sie in $x_{k+1}=A_k x_k+B_k u_k$ *Gleichung 2.71* [Wendel 2011: S. 16] hergeleitet wird. Dieses Systemmodell erweitert man um den Systemrauschanteil w_k , wobei $w_k aus\, N(0,Q_k)$ normal verteilt aus dem Mittelwert 0 mit der Varianz Q_k

$$x_{k+1}=A_k x_k+B_k u_k+w_k \quad \textit{Zustandsgleichung}\ [\text{kalman}]$$

Die Messung des Systems wird mit der Gleichung

$$y_k = H_k x_k + v_k \quad \textit{Messgleichung} \text{ [kalman]}$$

beschrieben, wobei v_k das Messrauschen $v_k \, aus \, N(0, R_k)$ ist. Die Großbuchstaben in den Gleichungen sind Systemmatrizen, die die zugehörigen Systemzustände abbilden.

$$\widehat{x_{k+1}^-} = A_k \widehat{x_k^+} + B_k u_k \quad \textit{a priori Schätzung} \text{ [kalman]}$$

Mit der Korrekturgleichung wird der a priori Schätzwert anhand des Messwertes zur posteriori Schätzung korrigiert.

$$\widehat{x_k^+} = \widehat{x_k^-} + K_k (y_k - H_k \widehat{x_k^-}) \quad \textit{Korrekturgleichung} \text{ [kalman]}$$

Der Term $y_k - H_k \widehat{x_k^-}$ enthält die Qualität der Schätzung. Wenn der Messwert dem a priori Schätzwert sich nähert tendiert dieser Term gegen Null und die a priori Schätzung war sehr genau und es gilt $\widehat{x_k^+} = \widehat{x_k^-}$. Die Matrix K_k ist der Kalman Verstärkungsfaktor und bewirkt, das die posteriori Schätzfehlerkovarianz P_k^+ möglichst minimal ist. Man benötigt daher auch noch die Fehlerkovarianzen der posteriori und a priori Schätzungen P_k^- für die Berechnung des Kalman Verstärkungsfaktor K_k für den Iterativen Prozess. Für tiefer gehende Information über das Kalman Filter sei auf die Literatur verwiesen, z.B. [Welch and Bishop 1995].

Neben dem Einsatz des Kalman Filter zur Glättung der Messwerte der Inertial Sensoren ist ein weiterer Einsatzbereich des Kalman Filter die Sensordatenfusion. Die Grundidee der Datenfusion ist, robustere Information über den Systemzustand zu gewinnen. Dabei kann man verschiedene Sensortypen oder gleiche Sensortypen für die Informationsgewinnung einsetzten und die Ergebnisse fusionieren um:

- Die Zuverlässigkeit eines Systems zu erhöhen (Redundanz)
- Mehrdeutigkeiten zu identifizieren (Gewichtung)
- Unterschiedliche Messraten von Systemen zu kombinieren (Information Aging)
- Messrate zu erhöhen (Information Increaseing)

- Messgenauigkeit bzw. Messauflösung zu erhöhen
- Messsichtbereich abzudecken
- Wirtschaftlichkeit durch kombinierte Sensoren billigerer Bauart

Im Falle der Mikro-Drohne stehen ein GPS Modul und ein Magnetfeldsensor als Stützsensoren für das Inertialsystem zur Verfügung. Die Magentfeldsensoren liefern den Vektor des Erdmagnetfeldes anhand der Gleichung $\vec{h^b}=(h_x^b, h_y^b, h_z^b)$ und können zur Stützung des Gier Winkels der Mikro-Drohne benutzt werden. Falls die Mikro-Drohne horizontal ausgerichtet ist, liefert der Magnetfeldsensor den Gierwinkel anhand $\psi=-\arctan 2(h_y^b h_z^b)$. Um den Messfehler bei nicht horizontaler Lage zu detektieren betrachtet man das Erdmagnetfeld im Navigationskoordinatensystem in Bezug zum gemessenen Magnetfeld im Körperfesten Koordinatensystem $\vec{h^b}=\boldsymbol{C}_{\boldsymbol{b}}^{\boldsymbol{n},T}\vec{h^n}+\vec{v_m}$, wobei $\vec{v_m}$ das Messrauschen ist [Wendel 2011: S. 282]. Anhand dieser Gleichung kann man den Filter entwerfen, um die Magnetfelddaten mit den Gierwinkeln der Inertialsensoren zu fusionieren. Dazu betrachtet man noch einen Fehler des Gierwinkel mit $\vec{\psi}_n^{\hat{n}}=(0,0,\gamma)$ und erhält die *endgültige Messgleichung*

$$\vec{h^b}-\boldsymbol{C}_{\boldsymbol{b}}^{\hat{\boldsymbol{n}},T}\vec{h^n}=-\boldsymbol{C}_{\boldsymbol{b}}^{\hat{\boldsymbol{n}},T}\begin{bmatrix} h_e \\ -h_n \\ 0 \end{bmatrix}\gamma+\vec{v_m}$$ [Wendel 2011: S. 283].

3.3. Konstruktion eines Oktokopter

Dieses Kapitel beschreibt den Bau eines Oktokopter mit acht Antriebseinheiten. Die Mikro-Drohne basiert auf den elektronischen Bauteilen des Mikrokopter Projekts. Es gibt eine Reihe von kommerziellen und nicht-kommerziellen Projekten, die sich mit der Konstruktion, dem Entwurf der elektronischen Schaltungen und Programmierung von autonom fliegenden Mikro-Drohnen, sowie auch von Flächenflugkörpern, beschäftigt. Einen Überblick zu den verschiedenen Projekten mit den dazugehörigen Features gibt es unter „http://multicopter.org/wiki/Multicopter_Table“ nachzulesen. Das Mikrokopter

Projekt als Basis für die Mikro-Drohne wurde gewählt, da die Bauteile im Preis-Leistungsverhältnis gut positioniert sind, und die Software der einzelnen Komponenten zum größten Teil einsehbar ist.

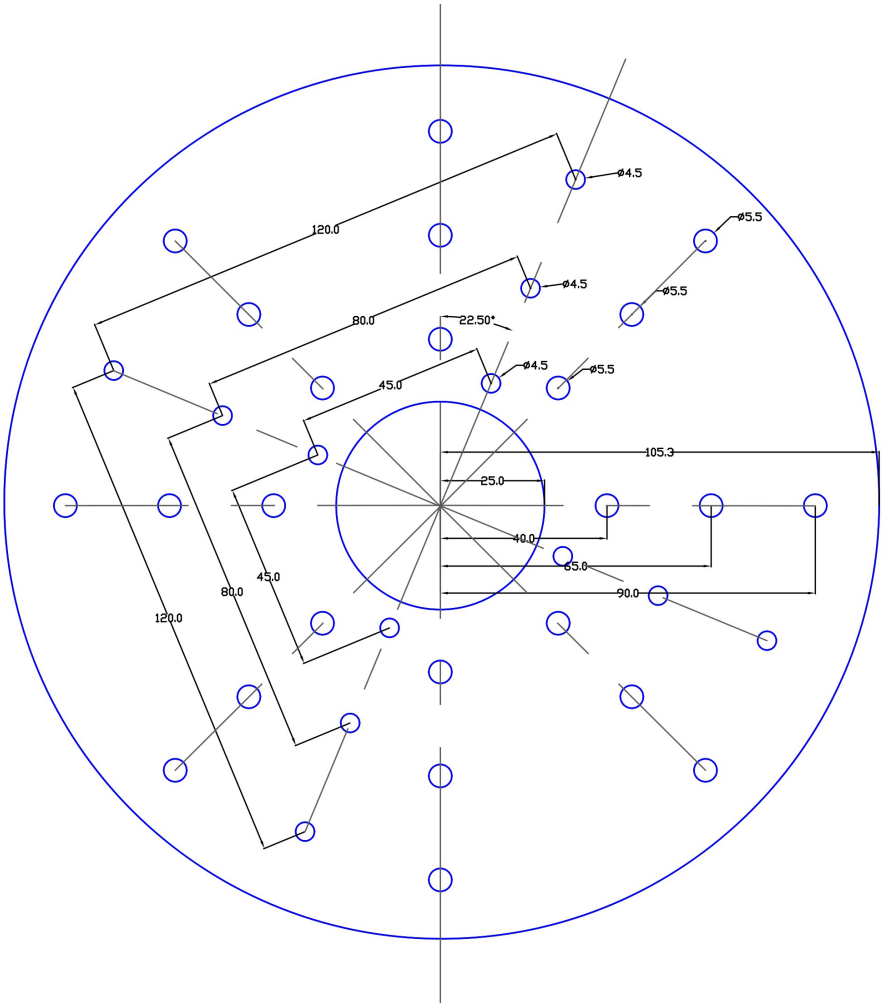


Abbildung 3.12: CAD Entwurf der oberen Mittelscheibe des Oktokopter mit Bemaßung

Das bedeutet der größte Teil des Quellcode ist verfügbar für Erweiterungen und Eigenentwicklungen. Teile der Softwarequellen des Navigations Controller sind aus lizenzrechtlichen Gründen nicht als Quellcode verfügbar. Der Rahmen wurde vom Autor konstruiert, da zum Zeitpunkt des Baus kein derart leistungsstarker Bausatz im Mikrokopter Projekt verfügbar war. Als Material für den Rahmen wurde für den ersten Prototypen eine AlMgSi 0.5 Aluminiumlegierung gewählt. Das Material ist leicht zu

verarbeiten, besitzt ein eine geringe Dichte von 2.7 g/cm³ und eine Zugfestigkeit von zirka 230 N/mm². Das Halbzeug ist einfach und günstig in verschiedenen Stärken als Bleche und Vierkant Halbzeueg in Baumärkten zu beschaffen. Für die zentralen Mittelscheiben der Mikro-Drohne wurde 0.8 mm dickes eloxiertes Aluminiumblech verbaut. Die Bemaßungen und das Layout der Mittelscheiben wurde in einem CAD Programm erstellt (Abb. 3.12) und mittels einer Papierschablone und Reißzeugs auf das Aluminiumblech übertragen. Die Ausarbeitung der Mittelscheiben erfolgte mit einer Blechschere und anschließendem Rundschleifen an einem Bandschleifgerät. Dazu wurde im Zentrum der Scheiben eine Bohrung gemacht, um mittels eines hindurchgeschobenen Rundstab einen möglichst symmetrisch-runden Kreis zu schleifen. Das Mittelloch mit 50 mm Durchmesser wurde nach dem Rundschleifen mittels einer Lochsäge ausgeschnitten.

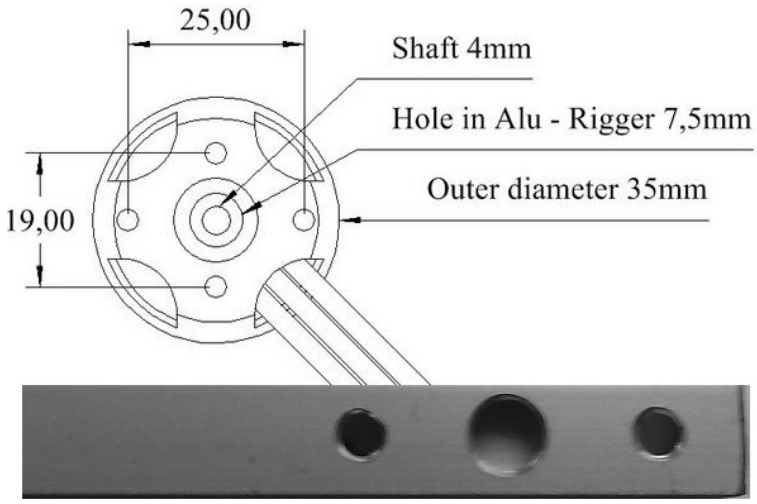


Abbildung 3.13: Motorbefestigung am Auslegerende und Motorbemaßung Bildquelle z.Teil [mikrokopter wiki mk3538]

Zwischen den beiden Scheiben wurden die acht Ausleger befestigt, dafür sind sind die 5.5 mm Bohrungen gedacht. Die Ausleger wurden mit je drei M4 Innensechskant-Zylinderkopfschrauben an M4 Nietmuttern, die auf der oberen Mittelscheibe eingepresst wurden, zwischen den beiden Scheiben fixiert. Diese Konstruktion ermöglicht einen schnellen Austausch von den Auslegern. Für die Befestigung der elektronischen Komponenten sind an der oberen Scheibe 4.5 mm Bohrungen ausgeführt, wo M3

Nietmuttern für die Befestigung eingepresst wurden. Die Ausleger sind aus Aluminium Vierkant Stäben von 10x10x1mm mit einer Länge von je 500 mm konstruiert. An der Außenseite befinden sich die Bohrungen für die Befestigung der Motoren, die mit M3 Innensechskant-Zylinderkopfschrauben fixiert wurden. Zwischen den Bohrungen mit Abstandsmaß von 25 mm wurde eine Auslassbohrung für die Motorwelle (Abb. 3.13) angebracht. An der gegenüberliegenden Seite sind je drei Bohrungen für die Befestigung an der Mittelscheibe. Für die Bohrungen wurde eine Schablone gebaut, die eine einfache und zügige Fertigung der Ausleger mit gleicher Qualität der Bohrbemaßungen ermöglicht. Dies hat sich als Vorteil erwiesen, da die Ausleger bei unkontrollierten Landemanövern die potentielle und kinetische Energie der Mikro-Drohne in Verformungsenergie umwandelt und somit, der im Zentrum der Mikro-Drohne angebrachten Elektronik, einen gewissen Schutz bietet.

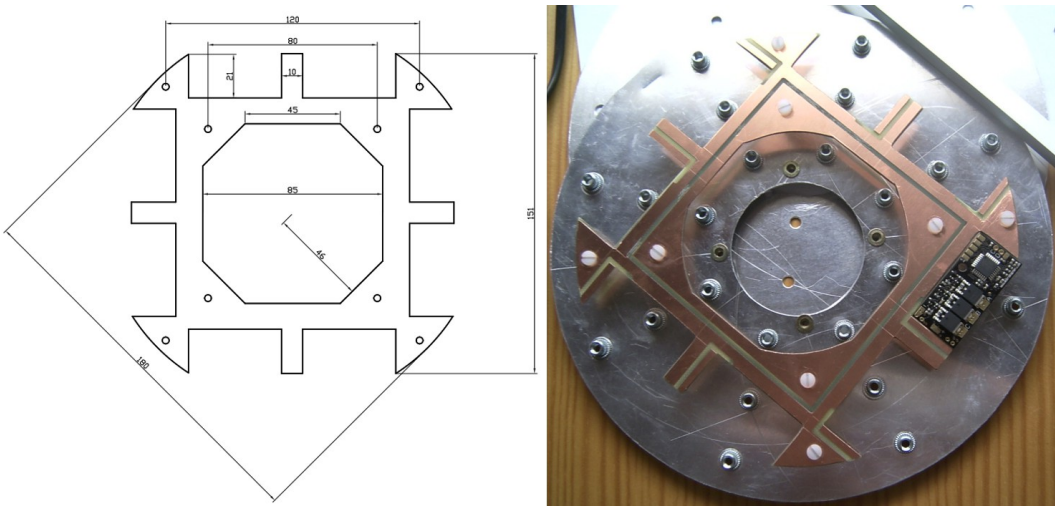


Abbildung 3.14: Stromverteiler Bemaßung und Aufbau an der Mittelscheibe

Der nächste Schritt war die Konstruktion des Stromverteilers. Im Betrieb müssen für die Motoren hohe Stromflüsse mit Spitzenwerten von über einhundert Ampere bedient werden, die von den Brushless Controllern an den angeschlossenen Motorphasen zugeführt werden. Jeder Brushless Controller benötigt vier Anschlüsse, was zu einer Zunahme um Faktor vier der Kabelanzahl je Motor resultiert $n_{kabel} = 4\, m_{motor}$. Um eine Verkabelung zu den einzelne Brushless Controllern zu umgehen, was die Wartung durch Unübersichtlichkeit erschweren würde, wurde das Konzept des Stromverteilers

gewählt. Dabei handelt sich um eine doppelseitig beschichtete Platine, an der auf der Unterseite der Pluspol und an der Oberseite der Minuspol des Akkumulator anliegt. Der Pluspol wurde nach unten gebaut, da die Phasenanschlüsse der Motoren an den Brushless Controllern sich auf der Plusseite befinden und die Motoren von unten angeschlossen wurden. Außerdem wurde auch die Führung des I²C Bus ausgefräst Um einen Stromschluss zu vermeiden, wurde die Stromverteilerplatine mit acht Kunststoffschrauben aus Nylon an der Mittelscheibe befestigt (Abb. 3.14).

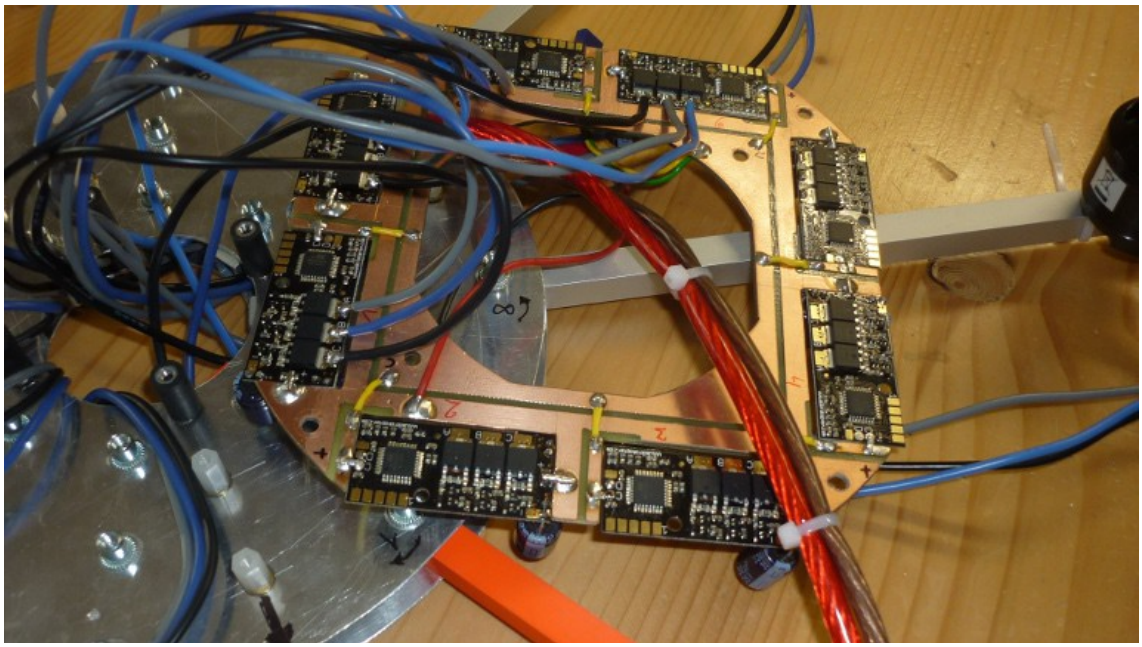


Abbildung 3.15: Anschluss der Motoren

Die Motorenanschlüsse wurden direkt an den Brushless Controllern angelötet. Dies hat den Vorteil einer Gewichtsreduktion, da ein vergoldetest Steckersystem wegen der hohen Stromflüsse massiv und schwer wäre. Dazu wurden die Motoren mittels Kabelbinder an den Auslegern mittig befestigt und die Anschlusskabeln wurden zwischen den Mittelscheiben verlegt und an der fertig bestückten Stromverteilerplatine in der richtigen Reihenfolge assembliert (Abb. 3.15). Nachdem alle Motoren angelötet waren, wurden sie am Ende der Ausleger mittels M3 Schrauben montiert. An der Flight Controll wurden die Anschlüsse zur Stromverteilerplatine und der Anschluss für die Fernsteuerung angelötet. Die mitgelieferten Spannungswandler und

Elektrolytkondensatoren der Flight Controll wurden durch Stärkere mit höherer Eingangsspannungs-Akzeptanz ersetzt. Dies geschah, weil anstatt eines 4 Zellen Akkumulator, der 16 Volt Ausgangsspannung besitzt, ein 5 Zellen Akkumulator mit 20 Volt Ausgangsspannung verwendet wird. Durch die höhere Spannung des Akkumulator ist eine höhere Motordrehzahl möglich und somit besitzt der Oktokopter mehr Leistungsreserve.

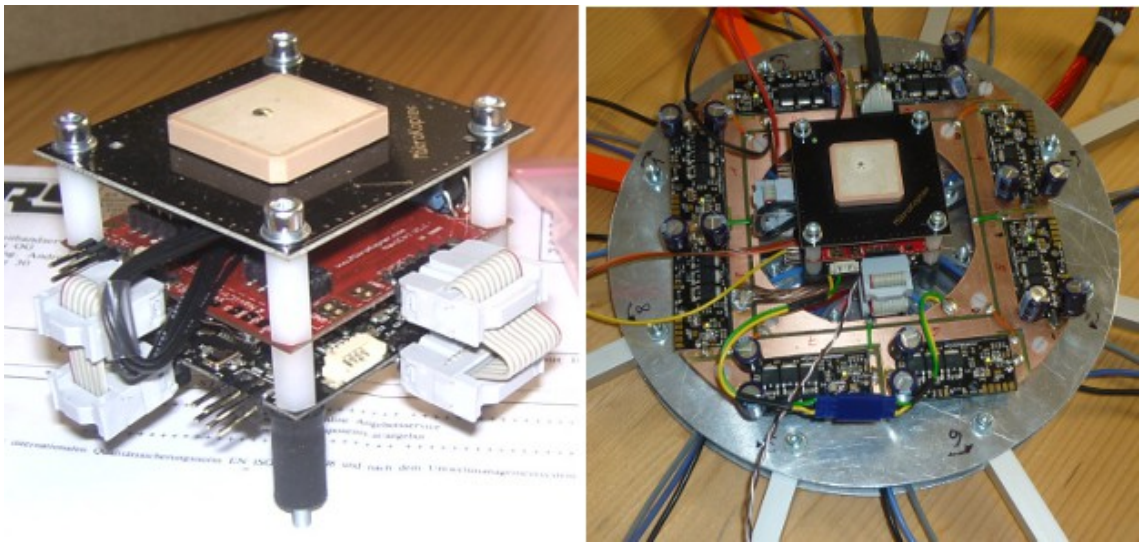


Abbildung 3.16: Steuerplatinen

Die Steuerplatinen wurden dann als Platinensandwich mit Abstandshalter aus Kunststoff zusammengebaut, und im Zentrum der Mittelscheibe mit vibrationsarmen Gummischrauben befestigt (Abb. 3.16). Der Verbindung zwischen der Flight Controll und dem Navigationscontroller besteht aus einem sechspoligen Flachkabel mit Pfostenstecker für den SPI Bus und aus einem zehnpoligen Flachkabel für das serielle Interface. Das GPS Modul wurde mit einem mitgelieferten Kabel am Navigationscontroller angeschlossen. Nach den ersten Funktionstests wurde die Mikro-Drohne fertiggestellt. Dazu wurden die Propeller an den Motoren montiert und ein Landegestell aus Kunststoffbändern an den Auslegern befestigt. Zum Schutz der Elektronik auf der Mittelscheibe wurde eine Haube aus Kunststoff mit Klettverschluss befestigt (Abb.3.17). Der Akkumulator wurde unterhalb der Mittelscheibe in einem Kameraobjektivbehälter untergebracht, der ebenfalls mit Klettverschlüssen an den

Auslegern befestigt wurde. Die Phasenanschlüsse zu den Motoren wurden straff mit Kabelbindern entlang den Auslegern befestigt. Der Fernsteuerungssender, der Telemetriesender und der akustische Signalgeber wurde auch an den Auslegern mit Kabelbinder montiert.

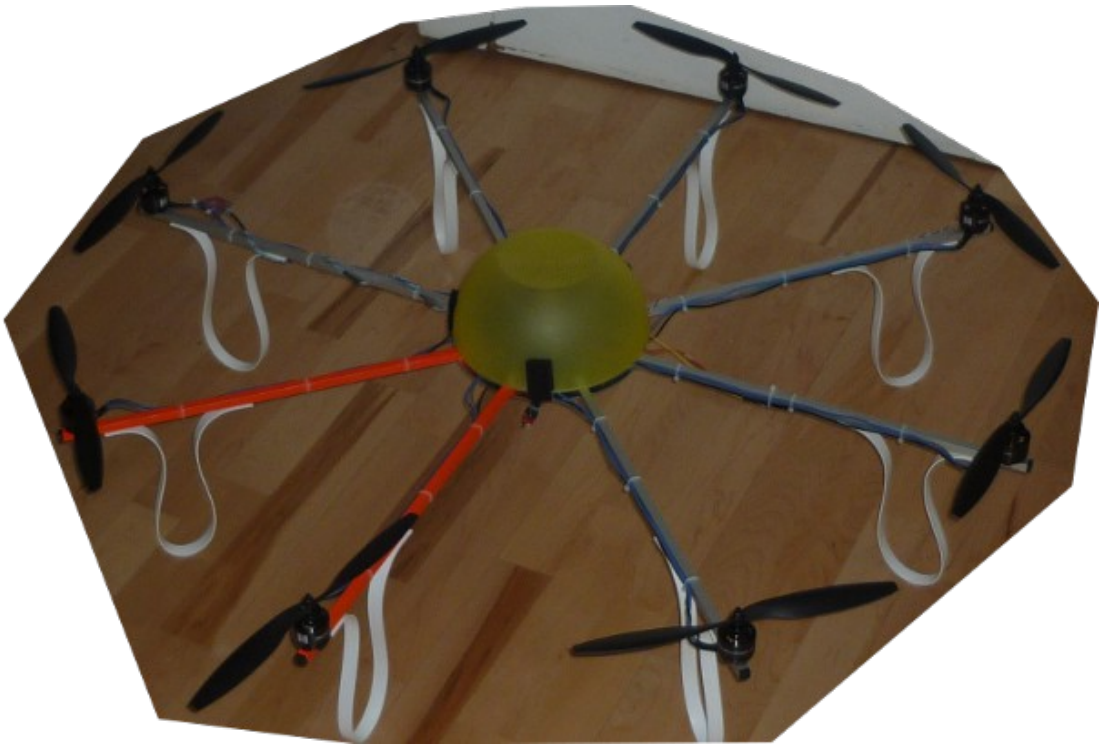


Abbildung 3.17: Prototype Oktokopter Version I

Nach ersten Testflügen zeigten sich gewisse Mängel an der Konstruktion des Oktokopters die nachgebessert wurden. Am Rahmen war die Materialwahl der Mittelscheibe aus Aluminiumblech nachteilig. Wie sich herausstellte kam es durch die wirkenden Kräfte bei den Landemanövern, und durch Abstürze zu persistenter Zugverformung der Mittelscheiben, sodass die Ausleger nicht mehr symmetrisch angeordnet waren. Dies wirkte sich nachteilig auf das Flugverhalten der Mikro-Drohne aus, da die Berechnungen in der Flight Controll bei einer verformten Rahmenstruktur nicht mehr dem zugrundeliegenden mathematischen Modell entsprach. Als neues Material für die Mittelscheiben wurde kohlenstofffaserverstärkter Kunststoff (CFK) mit einer Material-dicke von 2 mm gewählt. Dieses Material weist eine sehr hohe Festigkeit auf, besitzt keine persistente Zugverformung und hat eine geringes spezifisches Gewicht

von ~1.5g/cm³. Da der Entwurf der Mittelscheiben als CAD Modell vorhanden ist, wurde die Konstruktion als Auftragsfräsarbeit an einen spezialisierten Betrieb übertragen. Ein weiters Problem stellte die Festigkeit der zuerst eingesetzten flexiblen Propeller dar. Es kam mehrfach während eines Fluges vor, das ein Propellerblatt nahe der mittleren Bohrung abriss. Die vom Autor vermutete Ursache war, dass durch den Einsatz des 20V Akkumulator die resultierende höhere Drehzahl der Motoren und dadurch einhergehende höhere Fliehkräfte an den Propellerblättern, das Material überbeansprucht war. Die flexiblen Propeller waren möglicherweise dafür nicht ausgelegt. Abhilfe schaffte der Austausch der flexiblen Propeller gegen steife glasfaserverstärkte 13 Zoll Propeller. Als weiterer Vorteil erwies sich die höhere Schubkraft der steifen Propeller. Nachteilig hingegen ist die Bruchempfindlichkeit im Betrieb bei Kollision, wie zum Beispiel bei einer Bruchlandung.

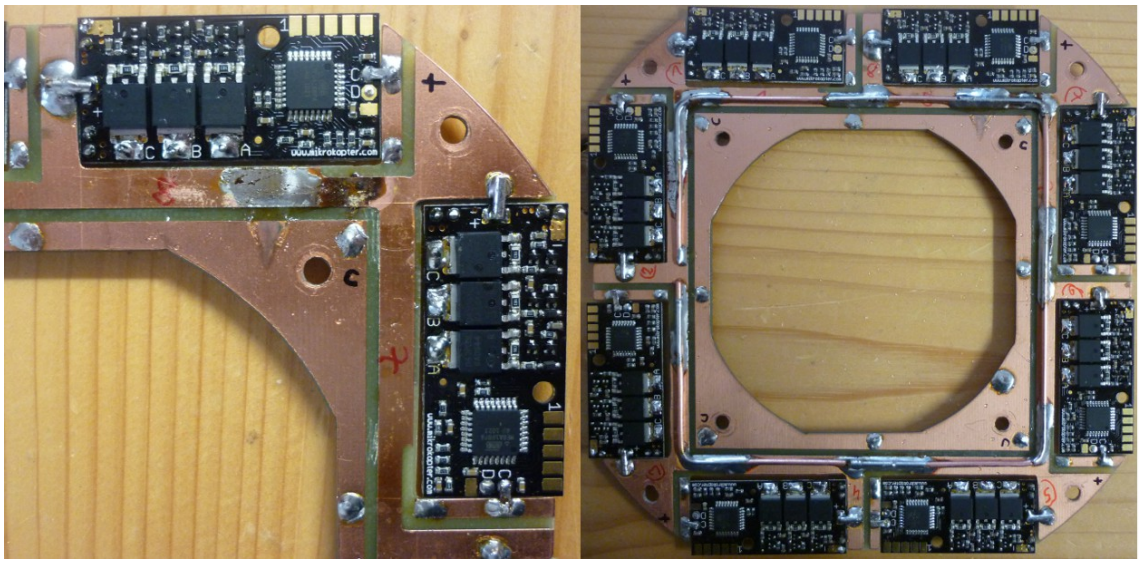


Abbildung 3.18: Eingelötete Stromschiene auf Stromverteilerplatine

An den elektronischen Bauteilen zeigte sich nach den ersten Testflügen Fehler am I²C Bus der Motorsteuerung. Nach Recherche im Forum des Mikrokopter Projekt wurde ein Hinweis gefunden, die Standard-Elektrolytkondensator der Brushless Controller gegen „Low-Impedanz“ Elektrolytkondensatoren auszutauschen. Die Low-Impedanz Elektrolytkondensatoren filtern dynamischer, und daher effektiver Spannungsspitzen, als Standard-Elektrolytkondensatoren und die nach sich ziehenden Störungen an der

Elektronik der Brushless Controller, die durch den Betrieb der bürstenlosen Motoren entstehen. Das Problem der I²C Fehler trat nach dem Austausch nicht mehr auf. Beim Austausch der Elektrolytkondensatoren wurde auch thermische Korrosion auf der Pluspol Seite der Stromverteilerplatine augenscheinlich. Die auf der Platine geführten Strommengen waren zu hoch, sodass es durch Wärmeentwicklung zur Oxidation der Kupferbeschichtung gekommen ist (Abb. 3.18 links). Abhilfe wurde durch eine angelötete Stromschiene geschaffen (Abb. 3.18 rechts). Die Konstruktion des Rahmens der Mikro-Drohne aus größtenteils konventionellen Materialien erwies sich aus ökonomischer Sicht von Vorteil. Die Ausleger und die Landefüße, welche „Verschleißteile“ sind, sind preisgünstig und schnell zu ersetzten. Die durch kohlefaserverstärkten Kunststoff ersetzte Mittelscheibe bewährte sich im Einsatz, da auch nach härteren Landemanövern mit hohen einwirkenden Kräften auf den Rahmen der Mikro-Drohne, keine Verformungen auftraten. Die elektronischen Bauteile des Mikrokopter Projektes haben ein gutes Preis-Leistungsverhältnis und die Software Komponenten unterliegen zum Zeitpunkt der Erstellung dieser Arbeit einer laufenden Weiterentwicklung. Die Mikro-Drohne arbeitet zuverlässig und im Rahmen der gegebenen Umstände sehr genau. Der Begriff Genauigkeit wird später in dieser Arbeit bei einem Experiment definiert und untersucht. Im Vergleich zu einer professionell zu erwerbenden Mikro-Drohne beträgt der Preis der selbstkonstruierten Mikro-Drohne nur einen Bruchteil. Nicht einkalkuliert ist die Arbeitszeit für die Konstruktion und die nachfolgenden Umbauten.

3.4. Bedienung und Software des Mikrokopter

Dieses Kapitel beschreibt die Bedienung der Mikro-Drohne, die dazugehörigen Softwarekomponenten und die Infrastruktur der Mikro-Drohne. Für einen Mikro-Drohnen Einsatz benötigt man mindestens

- Funktionstüchtige Mikro-Drohne
- Remote Controller zur interaktiven Steuerung der Mikro-Drohne
- Laptop mit Windows Betriebssystem für das Mikrokopter Tool

- Sender/Empfänger Modul am Laptop für die Kommunikation zur Mikro-Drohne

Nach dem Zusammenbau der Mikro-Drohne werden als erster Schritt die Firmware der elektronischen Komponenten aktualisiert, oder sichergestellt, dass die aktuellste Firmware installiert ist. Zu diesen Komponenten zählen die Brushless Controller, die Flight Controll, der Navigations Controller und das magnetische Kompass Modul. Damit wird sichergestellt, das die Interaktionen der einzelnen Komponenten miteinander einheitlich und funktionierend ist. Die aktuelle Firmware bezieht man von der Mikrokopter Projekt Webseite. Wenn alle Komponenten miteinander einwandfrei funktionieren werden die elektronischen Komponenten parametrisieren.

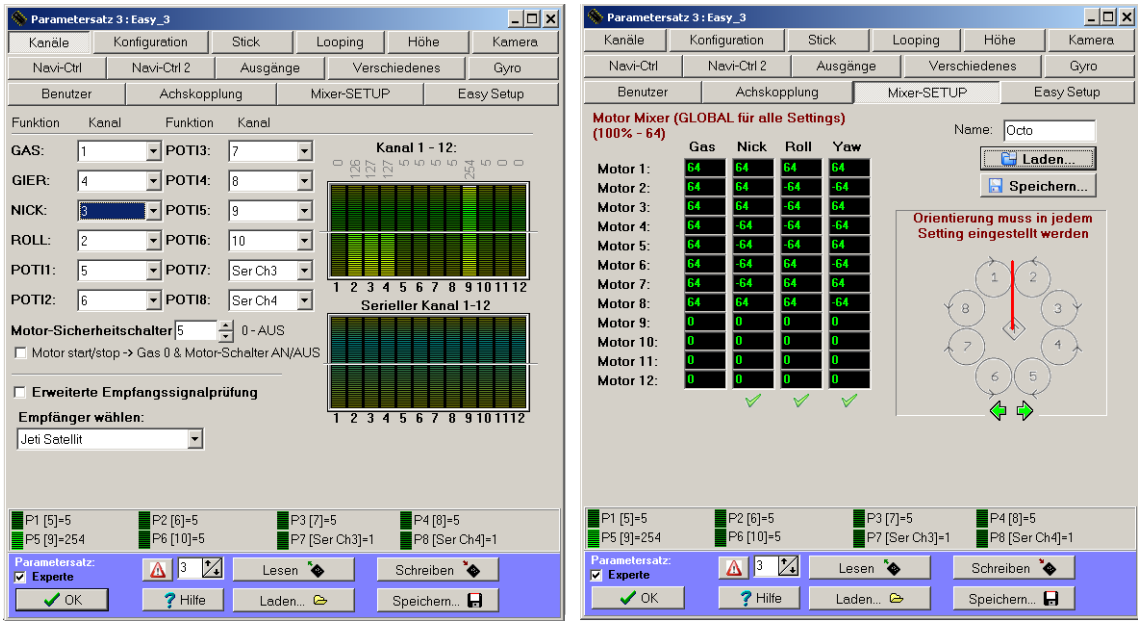


Abbildung 3.19: Paramter für Kanalzuordnung und Mixer

Ein Teil der Firmware ist der statische Programmteil, der aus den Anweisungen und unveränderlichen Variablen besteht. Dieser Anteil ist im nichtflüchtigen Programmspeicher des Mikrokontroller abgelegt, dem sogenannten „Flashram“. Variablen zur Einstellung von ab änderbaren Eigenschaften der Mikro-Drohne wird in einem anderen Speicher abgelegt, dem „Elektrisch Löschbaren Nur-Lese-Speicher“ EEPROM. Es gibt eine Vielzahl von Parametern. Hierfür sei auf das Mikrokopter Projekt Wiki „http://www.mikrokopter.de/ucwiki/MK-Parameter“ verwiesen. Die hier behandelten Parameter betreffen die Kanalzuordnung der interaktiven Fernsteuerung

und das Mixersetup der Motoren. Um die Mikro-Drohne parametrisieren zu können benötigt man das Mikrokopter Tool Programm und eine serielle Verbindung vom Computer zur Mikrodrohne mit einem direkt an der Mikro-Drohne angestecktem Flachkabel oder der Telemetrie Funkverbindung. Die Software der Flight Controll kann zwölf Fernsteuerungskanäle, die frei belegbar sind, verarbeiten. Die ersten vier Kanäle dienen zur interaktiven Steuerung der Mikro-Drohne und entsprechen den vier Steuergleichungen $u_1 - u_4$ aus Kapitel 3.2.2.

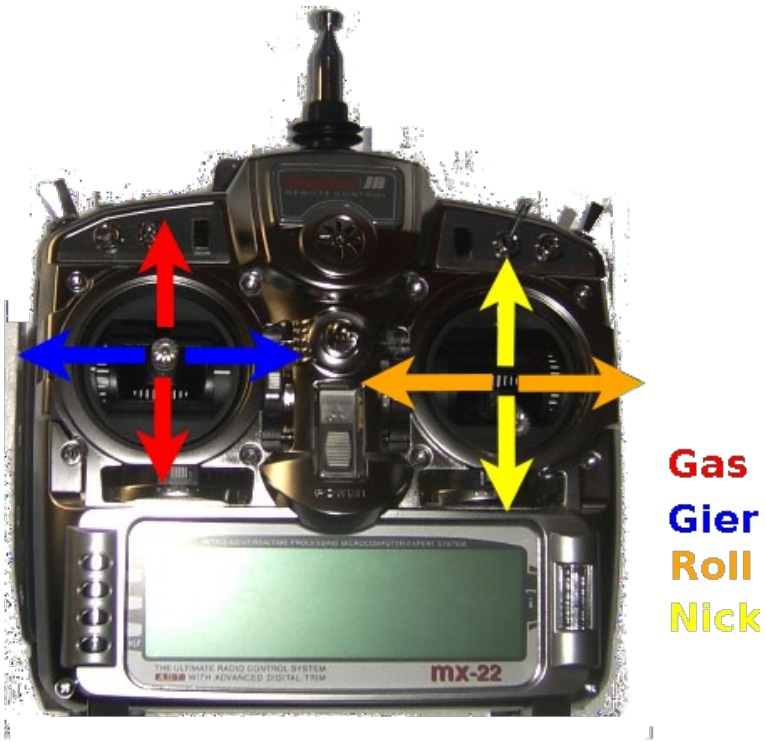


Abbildung 3.20: Belegung der Remote Controll für Koptersteuerung

Die Kanäle für Gas, Gier, Nick und Roll werden mittels Pulldown Item zugeordnet (Abb. 3.19). Sie entsprechen den Joysticks auf der Fernsteuerung, die im Setup der Fernsteuerung eingestellt werden müssen (Abb. 3.20). Die übrigen freien acht Kanäle kann man mit weiteren Eigenschaften und Funktionen der Mikro-Drohne belegen und entsprechen Kippschaltern und Drehpotentiometern auf der Fernsteuerung. Auf der rechten Seite des Parameterfensters der Kanalzuordnung kann man die Funktion anhand der grünen senkrechten Balken überprüfen (Abb. 3.19). Im Mixersetup wird das

Kräftegleichgewicht der Motoren für die Steuerbewegungen parametrisiert. Ein Dezimalwert von 64 entspricht 100% möglicher Motorleistung. Die Anordnung der Motoren wird hierbei in vier Teile zerlegt, indem man ein Koordinatenkreuz durch den Schwerpunkt der Mikro-Drohne zieht. Die positive nach oben zeigende Y Achse entspricht der Vorwärtsrichtung. Die X Achse einspricht dem Nick Kräfteanteil und die Y Achse entspricht dem Roll Kräfteanteil der Steuerbewegungen. In der Tabelle werden die Nick Rückwärts Anteile und die Roll Rechts Anteile als positiver Wert der Motorleistung eingetragen. Die senkrecht dazu stehende Z Achse entspricht der Gier Bewegung. Die Werteanteile der Motorleistung werden für eine Bewegung im Uhrzeigersinn eingetragen (Abb. 3.19). Abschließend zur ersten Inbetriebnahme muss man noch die Beschleunigungssensoren und den magnetischen Kompass einmalig kalibrieren. Der magnetische Kompass wird durch Auslösen der Prozedur und anschließendem Drehen der Mikro-Drohne in den drei Raumachsen kalibriert. Der magnetische Kompass muss bei größerem Ortswechsel neu justiert werden, da sich die Inklination mit dem Breitengrad ändert. Für die Kalibrierung der Beschleunigungssensoren muss die Mikro-Drohne in eine exakt horizontale Position gebracht werden, um den, auf die Konstruktion wirkenden, Gravitationsvektor abzuspeichern, der für die Lageberechnungen benötigt wird.

Vor dem ersten Testflug mit der Mikro-Drohne muss die volle Funktionstüchtigkeit der Drohne sichergestellt werden. Hierfür ist eine Klarliste beziehungsweise Checkliste, wie sie in der Luftfahrt eingesetzt wird, sinnvollerweise anzuwenden.

- Sind alle Kabel befestigt und sind Schäden am Rahmen oder an Kabeln erkennbar ?
- Sind die Propeller in Ordnung und sitzen fest ?
- Ist eine SD Speicherkarte im Navigations Controller eingelegt ?
- Zeigen die farblich markierten Ausleger nach vorne weg (Vorwärtsrichtung) ?
- Sind die Akkumulatoren der Remote Controll und der Mikro-Drohne vollständig

geladen ?

- Steht der Motorsicherheitsschalter auf der Remote Controll auf „On“ (kein starten der Motoren möglich) ? Sind die Schalter für die GPS Funktion „Position Hold“ und die automatische Höhenregelung auf „Off“ ?
- Zuerst Remote Controll einschalten, danach Akkumulator der Mikro-Drohne anschließen und auf die akustische Signalausgabe der Mikro-Drohne achten, die Fehler akustisch signalisieren.
- Überprüfen ob die „LED's“ auf den elektronischen Bauteilen alle Grün leuchten.
- Überprüfen der Funkverbindung mit der Remote Controll und optional überprüfen, ob die Telemetrie Funkverbindung mit dem Laptop aufrecht ist. Auf Fehlerausgabe an der Remote Controll und optional am Laptop im Mikrokopter Tool achten.
- Gyroskope justieren. Dies geschieht indem man den Gas-Gier Stick der Remote Controll nach links-oben schiebt.
- Warten, bis ein GPS Satelliten „fix“ vorhanden ist. Die minimale Anzahl der Satelliten für einen „fix“ ist im Parameter Menü des Mikrokopter Tool vorher einzustellen.
- Starten der Motoren indem man Motorsicherheitsschalter auf „Off“ legt und danach den Gas-Gier Stick nach rechts unten zieht.
- Prüfen, ob alle Motoren gleichmäßig laufen, dazu vorsichtig Gas zugeben und wieder weggeben.
- Prüfen, ob Fehler auf der Remote Controll angezeigt werden

Bevor man startet achtet man darauf, das man selbst einen ausreichenden Sicherheitsabstand zur Mikro-Drohne hat und das sich keine Person oder Ähnliches im geplanten Flugbereich der Mikro-Drohne befindet. Durch vorsichtiges Gasgeben startet man die Mikro-Drohne. Die Steuerbewegungen der Mikro-Drohne sind in Abbildung

3.20 ersichtlich. Durch Umlegen des Schalters für die Höhenregelung versucht die Flight Controll anhand des eingebauten Barometrischen Sensor die aktuelle Höhe zu halten. Mit eingeschalteter Höhenregelung wird durch Verschieben des Gasregel-Sticks nun nicht mehr die Motorleistung reguliert, sondern es wird der Höhenwert in der Flight Controll verschoben. Wenn man mit eingeschaltetem Höhenregler zu landen versucht, muss man darauf achten, dass es in Bodennähe, durch die am Boden reflektierte Luftmasse der Rotoren, zu Schwankungen des Höhensensorwertes kommt, der zu nicht beabsichtigten Steuermanöver führen kann. Der Schalter für die GPS Funktionen hat drei Stellungen. In der „Off“ Stellung ist das GPS deaktiviert. In der mittleren Schaltposition wird die Funktion „Position Hold“ aktiviert. Die Mikro-Drohne versucht anhand der GPS Daten die Position autonom zu halten. Es gibt zwei verschiedene zu parametrierende „Position Hold“ Funktionen. Beim „Statischen Position Hold“ wird nach einer manuellen horizontalen Steuerbewegung (Nick, Roll Kommandos) die erreichte GPS Position als aktueller „Position Hold“ Wert gespeichert. Beim „Dynamischen Position Hold“ bewirkt eine manuelle Steuerbewegung das Verschieben der longitudinale und latitudinalen Koordinaten-Werte im Speicher der Mikro-Drohne. Die dritte Schalterposition besitzt auch zwei Funktionen. Wenn keine Wegpunkte im Navigationscontroller gespeichert sind, löst die dritte Schaltposition die „Coming Home“ Funktion aus. Die Drohne kehrt autonom zum gespeicherten GPS Startpunkt zurück. Wenn Wegpunkte für einen autonomen Flug abgespeichert wurden, wird diese Sequenz zum Abfliegen der Wegpunkte ausgelöst.

Das zum Mikrokopter Projekt gehörende Mikrokopter Tool ist die Software, mit der man über den seriellen Telemetrie Link die Funktionen der Mikro-Drohne ergänzt. Mit dieser Software ist es möglich die Telemetriedaten, welche die Mikro-Drohne ausgibt, in einem Oszilloskop (Scope) zu überwachen, sowie als „Comma Seperated Values“ CSV-Datei für eine spätere Analyse abzuspeichern. Im mittleren Bereich des Hauptfensters existiert ein Informationsfeld, in dem sowohl Betriebsparameter der verschieden Baugruppen ausgegeben werden, als auch Fehlercodes bei Fehlfunktionen (Abb. 3.21). Das Einspielen neuer Firmware sowie die Parametrierung der Mikro-

Drohne wird mit dem Mikrokopter Tool durchgeführt.

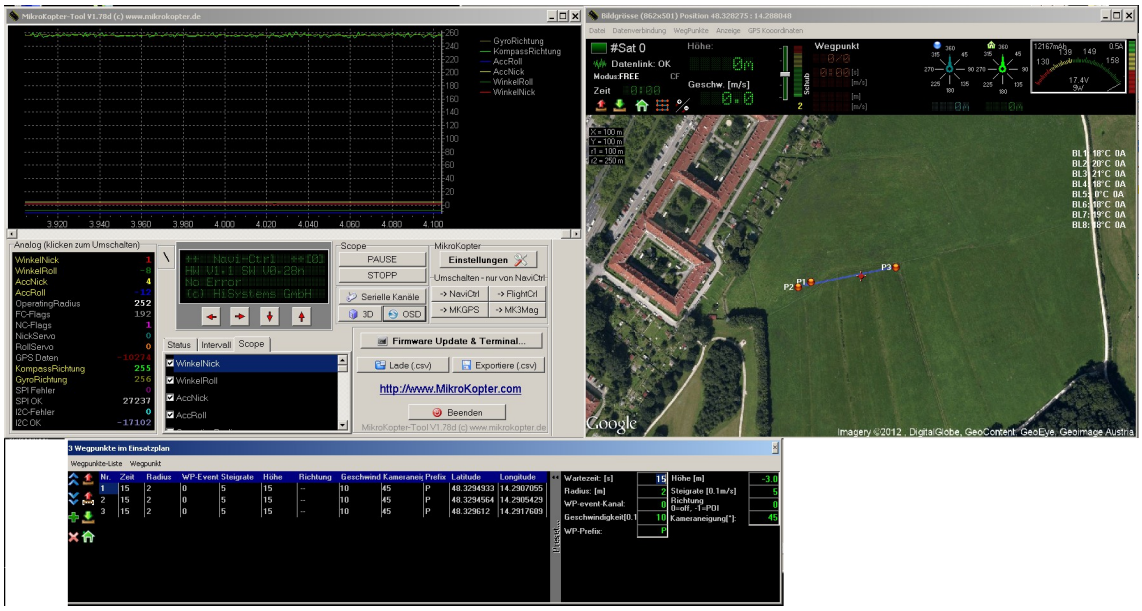


Abbildung 3.21: Mikrokopter Tool Software

Im OSD des Mikrokopter Tools (Abb. 3.21 rechte Seite) ist es möglich Wegpunkte und „Point of Interest“ POI zu editieren und an die Mikro-Drohne zu übermitteln. Dazu benötigt man ein WGS84 georeferenziertes Orthofoto oder eine Karte in UTM Projektion. Die Wegpunkte werden auf der Projektion gesetzt und im Einsatzplan Fenster mit den entsprechenden Attributwerten versehen. Bei einer bestehenden Telemetrieverbindung zur Mikro-Drohne ladet man die Wegpunkte Sequenz in den Speicher des Navigations Controller.

4. Lösungsansatz und Vorbereitung

In diesem Kapitel wird das Testgebiet für das Experiment präsentiert und die dafür akquirierten Daten untersucht und vorgestellt. Des weiteren werden die benötigten Daten, die für die Kommunikation zwischen dem Geoinformationssystem und der Mikro-Drohne verwendet werden untersucht und die Datensätze für die Verwendung mit der Mikro-Drohne im verwendeten Desktop GIS, hier im speziellen ArcGIS 10 vorbereitet.

4.1. Voruntersuchungen des Testgebietes für das Experiment

Für das geplante Experiment mit der Mikro-Drohne wird ein geeignetes Areal benötigt, das den Anforderungen eines idealen Experimentiergebietes entspricht. Diese Anforderungen an das Gebiet sind wie folgt

- Flach und eben ohne Hindernissen, wie Bebauung oder Bewuchs, um die Mikro-Drohne immer in Sicht behalten zu können
- Optimaler GPS Empfang ohne oder nur geringe horizontale Abschattungen der Satellitensignale und freie Sicht nach Süden für den Empfang der EGNOS Korrektursignale
- Wenig frequentierte Gegend um Personen oder Sachschäden durch die Mikro-Drohne auszuschließen
- Vorhandene georeferenzierte Fixpunkte um die Ergebnisse des experimentellen Mikro-Drohnen Einsatzes validieren zu können
- Keine Flugverbotszone - siehe Kapitel 7.2 Rechtliche Aspekte

Als Untersuchungsgebiet (Abb.4.1) für das Experiment wurde das Wasserschutzgebiet Heilham im Norden von Linz, Stadtteil Urfahr-Harbachsiedlung ausgewählt. Die Fläche des gelb umrandeten Areal umfasst zirka 0,1 km² und wird als Wiese, die regelmäßig gemäht wird, genutzt. Eine Begehung im Herbst, dem Zeitpunkt des Experimentes, ist möglich. Das Areal wird selten von Personen frequentiert. Es wird hauptsächlich von

Hundebesitzern als Freilauffläche genutzt. Da sich im Untersuchungsgebiet im Umkreis von $\approx 150\,Meter$ weder Gebäude noch hoher Bewuchs befinden, liegen hier ideale Bedingungen für GPS Datenakquisition vor. Es dürfte zu keinen GPS Signal Abschattungen oder Signalreflexionen, was einen Multipatheffekt erzeugt, kommen.

Abbildung 4.1: Untersuchungsgebiet Gelb umrandet nördlich von Linz/Österreich Bildquelle: Google-Earth

Nach Süden ist freie Sicht, um das EGNOS (European Geostationary Navigation Overlay Service) / SBAS (Satellite Based Augmentation System) nützen zu können. Das EGNOS/SBAS ist eine Differentielle-GPS Erweiterung zum GPS NAVSTAR, GLONASS und Galileo. Das Satellitensegment bestehend aus drei geostationären Satelliten, die Korrektursignale zur Positionsverbesserung aussenden. Der Empfang des EGNOS/SBAS Signals erstreckt sich über beinahe alle Staaten der Europäischen Union. EGNOS wird von der Europäischen Kommission und der ESA betrieben [egnos]. Das

offene EGNOS Service ist seit 1. Oktober 2009 in Betrieb und benötigt keine Zulassung und Zertifizierung seitens des GPS Empfangsgerätes. Es besteht jedoch auch keine Leistungsgarantie und Haftung seitens des Betreibers bei Verwendung des Offenen Dienstes. Zum Zeitpunkt des Experimentes im Herbst 2012 sind laut EGNOS User Support drei Satelliten in Betrieb, die ein SBAS Korrektursignal ausstrahlen (Tab. 4.1).

Satelliten-Name	Inmarsat 3-F2 AOR-E	ARTEMIS	Inmarsat 4-F2 EMEA
PRN Kodierungs-ID	PRN 120	PRN 124	PRN 126
Position	15.5° West	21.5° East	25.0° East
SIS Dienst Signal in Space	SIL aktiv	SIS aktiv	SIS aktiv
SOL Dienst Safty of life	SOL aktiv	SOL Test Mode	SOL aktiv

Tabelle 4.1: EGNOS Satellitensegment [egnos user]

Um die Elevationen der drei Satelliten zu berechnen bediene wir uns der Koordinatenursprungstransformation. Unter der Annahme der Erde als eine Kugel mit Radius $r_e = 6378\,km$ und den Koordinatenursprung im Mittelpunkt zeigen die X und Y Achse orthogonal zueinander in die Äquatorebene und die Z Achse verläuft durch Nord- und Südpol. Der geostationäre EGNOS Satellit steht in einer geostationären Umlaufbahn genau auf der Y Achse mit einem Abstand zum Koordinatenursprung von $r_s = 42164\,km$. Somit sind die Satellitenkoordinaten

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ r_s \\ 0 \end{bmatrix}$$ Ein Beobachter auf der Kugeloberfläche hat die Koordinaten mit einem Winkeloffset λ in der X und Y Achsen Ebene, welches dem Längengrad entspricht, und einen Winkeloffset ϕ orthogonal zur X und Y Achse Ebene, welche dem Breitengrad entspricht und eine Höhe r_e dem Erdradius. Nun transformiert man den Koordinatenursprung vom Mittelpunkt zum Beobachterstandpunkt mittels zwei Drehungen um λ um die Z Achse und ϕ um die X Achse mit

anschließender Verschiebung um r_e . In dem neuen Koordinatensystem mit dem Beobachter als Ursprungspunkt des Koordinatensystems hat der Satellit nun die transformierten Koordinaten $\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} r_s \sin\lambda \\ r_s \cos\lambda \cos\phi - r_e \\ -r_s \cos\lambda \sin\phi \end{bmatrix}$. Die Elevation lässt sich nach der Formel $e = \arctan\left(\frac{y'}{\sqrt{x'^2 + z'^2}}\right)$ ermitteln. Der Azimut berechnet sich nach der Formel $a = \arctan\left(\frac{|z'|}{|x'|}\right)$ und verschieben in den richtigen Quadranten. Vom Experimentiergelände mit den Koordinaten $\lambda = 14.2917$ und $\phi = 48.3296$ haben die EGNOS/SBAS Satelliten die in Tabelle 4.2 errechneten scheinbaren Standorte.

EGNOS Satellit	Inmarsat 3-F2 AOR-E 15.5° West	ARTEMIS 21.5° East	Inmarsat 4-F2 EMEA 25.0° East
Aziumt	217.53°	170.39°	165.79°
Elevation	27.53°	34.07°	33.55°

Tabelle 4.2: Standorte der EGNOS/SBAS vom Experimentiergebiet aus gesehen

Bei einem geschätzten Abstand von $d_{baum} \approx 180\,Meter$ zwischen Experimentierstandort und Baumstandort im Süden (Abb. 4.1) mit einer geschätzten Baumhöhe von $h_{baum} \approx 20\,Meter$ ergibt sich aus der trigonometrischen Formel $\alpha = \arctan\left(\frac{h_{baum}}{d_{baum}}\right)$ ein Winkel von $\alpha < 7°$ zwischen Baumobergrenze und Horizont. Somit sind die EGNOS/SBAS Satelliten von Experimentierstandort einsehbar und können vom GPS Modul der Mikro-Drohne verwendet werden.

4.1.1. Vorvermessung und Bestimmung von Fixpunkten für Experiment

Für das Experiment mit der Mikro-Drohne werden ein oder mehrere Fixpunkte benötigt an welchen man die Abweichungen der Mikro-Drohne beim Anflug eines Wegpunktes überprüfen kann. Anhand dieser Abweichung ist es möglich eine Aussage über die Genauigkeit der Mikro-Drohnen Steuerung tätigen. Im Untersuchungsgebiet befinden sich zwei Grundwassersonden der Linzer Wasserwerke (Abb. 4.2) .

Abbildung 4.2: Grundwassersonden (links: Sonde 1b rechts: Sonde 8)

Auf Anfrage bei den Linzer Wasserwerken wurden dem Autor für diese Arbeit die Vermessungsdaten der Grundwassersonden zur Verfügung gestellt (Tab. 4.3). Die Vermessungsdaten sind im EPSG 31255 / MGI_Austria_GK_Central Format (Tab. 4.6). Das geodätische Datum ist MGI und die verwendetet Projektion Transverse Mercator.

Sondenbezeichnung	X (easting) Wert	Y (northing) Wert
Grundwasser Sonde 1b	71035.00 Meter	355023.48 Meter
Grundwasser Sonde 8	71125.02 Meter	355041.15 Meter

Tabelle 4.3: Vermessungsdaten der Grundwassersonden in EPSG 31255. Quelle: Linz AG Bereich Wasser

Für das Experiment müssen die Vermessungsdaten der Sonden nach WGS84 transformiert werden. Dazu verwendet man den Transformationscode 1618 „AT_MGI

to WGS84“ nach Vorgabe des Österreichischen Bundesamt für Eich- und Vermessungswesen. Die sieben Transformationsparameter sind in Tabelle 4.4 einzusehen.

X Translation	Y Translation	Z Translation	X Rotation	Y Rotation	Z Rotation	Skalierungs-faktor
577.326	90.129	463.919	5.137	1.474	5.297	2.4232

Tabelle 4.4: Transformationsparameter MGI nach WGS84 [Vermessungswesen 2007: S. 2]

Eine Überprüfung der Vermessungsdaten erfolgt im nächsten Kapitel anhand von Orthofoto.

4.2. Datenbeschaffung und Analyse der Datenbestände

Für die Einsatzplanung der Mikro-Drohne benötigt man referenzierte Geodaten, anhand derer man die Wegpunkte, die die Mikro-Drohne anfliegen soll, definieren kann. Die Geodaten können von Onlinediensten stammen, wie von einen Web Feature Service (WFS) oder Web Map Service. Im Prinzip sind alle georeferenzierten Datenbestände möglich, die im verwendeten Desktop Geoinformationssystem zu verarbeiten sind, möglich. In dieser Arbeit werden die freien Geodaten der Stadt Linz/Österreich verwendet.

Die Stadt Linz startete im Sommer 2010 die Initiative „*Open-Commons-Region Linz*“ [Gustav Pomberger 2010: S.11]. Ziel der Initiative ist es, gemeine Daten für Wirtschaft, Bildung und Bevölkerung frei zugänglich zu machen. Für diese Arbeit sind im Besonderen die frei zugänglichen behördlichen Geodatenbestände der Stadt Linz interessant. Diese umfassen die behördlichen Orthofotos des Stadtgebietes von Linz und das Höhenschichtenmodell als GML Datei der Stadt Linz. Die Daten unterliegen der *Creative Commons CC BY 3.0 Lizenz* [ccby3.0], welche die freie Verwendung unter der Bedingung der Namensnennung der Daten-Urheber ermöglicht. Die frei zugänglichen Geodatenbestände der Stadt Linz sind zum Zeitpunkt der Ausarbeitung dieser Master Thesis unter der URL „http://www.data.linz.gv.at/daten/Geodaten/“ abzurufen.

Die freien digitalen Orthofotos der Stadt Linz liegen in einer Auflösung von 20 Zentimeter pro Pixel als georeferenzierte TIFF Kacheln vor. Eine Kachel hat die Auflösung von $3125\,x\,2500$ Pixel und somit eine Ausdehnung von $625\,Meter$ in West-Ost Richtung und $500\,Meter$ in Nord-Süd Richtung. Die Koordinaten der Orthofotos liegen im EPSG 31255 / MGI_Austria_GK_Central Format vor. Das geodätische Datum ist MGI und die verwendete Projektion ist Transverse Mercator, wie die Koordinaten der erhaltenen Grundwassersonden. Pro Kachel ist ein „World File“ vorhanden, um das Orthofoto im Desktop Geoinformationssystem georeferenziert einzubinden. Es ist keine nachträgliche Georeferenzierung notwendig. Das World File ist im ASCII Format und beinhaltet sechs Parameter pro Zeile. Die Parameter des World File sind:

Zeile	Inhalt
1	Größe pro Pixel auf der X Achse
2	Rotation um die Y Achse
3	Rotation um die X Achse
4	Größe pro Pixel auf der Y Achse
5	X Koordinate des Mittelpunktes des oberen linken Pixels
6	Y Koordinate des Mittelpunktes des oberen linken Pixels

Tabelle 4.5: Aufbau World File Quelle: ESRI ArcGIS Help

Anhand der Parameter im World File kann das Desktop Geoinformationssystem die Rasterdaten georeferenziert darstellen. Dabei führt das GIS eine affine Transformation

$$x' = P_{z1} X + P_{z3} Y + P_{z5}$$
$$y' = P_{z2} X + P_{z4} Y + P_{z6}$$

für jedes Pixel durch, um es in realen Koordinaten darzustellen. Der Indizes des Parameters P_{zi} entspricht dem Wert in der Zeile im World File. Die X Variable ist die Spalte des Pixels und die Y Variable die Zeile des Pixel. Die Darstellung ist im Kartesischen Koordinatensystem, daher ist der Y Größenwert negativ, da sich das Orthofoto vom linken oberen Ursprung nach unten ausdehnt. Die gesamte Ausdehnung der verfügbaren Orthofotos des Stadtgebietes Linz

in EPSG 31255 Meter Koordinaten ist links oben: 65000/363000 rechts unten: 82500/339000 . Insgesamt ist die Anzahl der Kacheln 506 Stück mit einem PDF File http://geo.data.linz.gv.at/katalog/geodata/orthofotos/2011/BlattUebersicht2011.pdf als Blattübersicht der Kachelanordnung.

Die verwendeten Raster- und Punktdaten liegen mit MGI Datum vor. Für den Mikro-Drohnen Einsatz werden Koordinaten mit WGS84 Datum vom Navigations Controller benötigt. Für die Transformation des Geografischen Koordinatensystem Ursprungs wird die sieben Parameter Methode nach Helmert verwendet. Die Helmert-Transformationsparameter zur Transformation von MGI nach WGS84 sind in Tabelle 4.4 angeführt. Um das Datum zu transformieren sind drei Translationen, drei Rotationen und eine Skalierung beziehungsweise Maßstabsänderung nötig. Die Transformation erfolgt anhand

- *Der Berechnung der ellipsoiden Breite, Länge und Höhe aus den Gauß-Krüger Koordinaten (Kapitel 3.2.2)*
- *Der Berechnung von X, Y und Z Werten bezüglich des MGI Referenzellipsoids*
- *Der Helmert-Transformation*

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}^N = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} + (1+s) \begin{bmatrix} 1 & R_z & -R_y \\ -R_z & 1 & R_x \\ R_y & -R_x & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

oder nach Multiplikation vereinfacht

$$\begin{aligned} X^N &= T_x + (1+s)(X + R_z Y - R_y Z) \\ Y^N &= T_y + (1+s)(-R_z X + Y + R_x Z) \\ Z^N &= T_z + (1+s)(R_y X - R_x Y + Z) \end{aligned}$$

- *Der Rückberechnung in ellipsoide Breite, Länge und Höhe (Kapitel 3.2.2)*

[Bauer 2003: S. 57]

Für die Rücktransformation multipliziert man die Transformationsparameter mit -1 , somit ist die Transformation reversible mit den gleichen Parametern. Um die Orthofotos mit den Fixpunkten zu analysieren werden sie in ArcGIS 10 geladen. Als Layer Projektion wird EPSG 31255 beziehungsweise MGI Austria GK Central mit den

Parametern aus Tabelle 4.6 gewählt.

MGI_Austria_GK_Central / EPSG 31255	
Projektion	Transverse Mercator
False Easting	0.000000
False Northing	-5000000.000000
Central Meridian	13.333333
Scale Factor	1.000000
Latitude Of Origin	0.000000
Lineare Einheit	Meter
Datum	MGI

Tabelle 4.6: ArcGIS Projektionsparameter Linz

Die Orthofotos des Untersuchungsgebietes sind laut Blattübersicht „5336-33-4-S11.tif“ und „5336-41-2-S11.tif“. Diese Orthofotos des Untersuchungsgebietes werden in ArcGIS 10 mit Layerprojektion EPSG 31255 geladen. Die Sondenstandorte importiert man aus einer Excel Tabelle, lässt die X und Y Werte anzeigen und exportiert sie als Point Feature. Diese Point Features werden mit den sieben Parametern aus Tabelle 4.4 von MGI nach WGS84 transformiert. Mit der Funktion „Calculate Geometry“ werden die WGS84 Längen- und Breitengrade aus den X und Y Werten des Point Features errechnet.

	ArcGIS Transformation	Geoland Transformation[1]	Visuell gemessenen Distanz in ArcGIS
Sonde 8 Longitude	14.29175776	14.29175788	~ 8.7 mm
Sonde 8 Latitude	48.32960740	48.32960738	
Sonde 1b Longitude	14.29054093	14.29054105	~ 9.3 mm
Sonde 1b Latitude	48.32945859	48.32945857	

Tabelle 4.7: Vergleich Transformation ArcGIS mit Geoland Transformation Service

1 Geoland Koordinaten Transformation Service EPSG 31255 → EPSG 4326 http://www.geoland.at/gps_trans.htm

Zum Vergleich werden die Vermessungsdaten der Fixpunkte von EPSG 31255 nach WGS84 mit dem *Geoland Transformationsservice* [geoland trans] transformiert, und die Abweichungen zwischen der ArcGIS 10 Transformation und der Geoland Transformation Service in Tabelle 4.7 verglichen. Die Abweichungen der Transformationen zueinander liegt unter einem Zentimeter.

Für das Mikrokopter OSD sind gebräuchliche prioritäre Geodatenquellen Google Earth oder Google Maps. Im Google Maps Online Karten Dienst sind die beiden Sondenstandorte als 45° aerial hochauflösende Orthofoto eindeutig erkennbar.

	ArcGIS Transformations Koordinaten	Google Maps Koordinaten	Visuell gemessenen Distanz in ArcGIS
Sonde 8 Longitude	14.29175776	14.29175800	~ 309.8 cm
Sonde 8 Latitude	48.32960740	48.32963500	
Sonde 1b Longitude	14.29054093	14.29054200	~ 216.5 cm
Sonde 1b Latitude	48.32945859	48.32947800	

Tabelle 4.8: Vergleich Google Maps erkennbare Fixpunkte in ArcGIS

Um eine Aussage über die Genauigkeit des Google Maps Dienstes zu treffen, sind die Vermessungskoordinaten der Fixpunkte und die visuell georeferenzierten Koordinaten der Fixpunkte aus Google Maps in Tabelle 4.8 gegenübergestellt. Die Abweichungen liegen im Meter-Bereich.

Abschließend zur Voruntersuchung der verwendeten Geodaten hat der Autor die Fixpunkten mit einem GPS Gerät „Garmin GPSMap 60Scx“ vermessen. Beim Garmin GPS Gerät wird die Abweichung der Messdaten als „Circular Error Probable CEP“ oder als Kreisfehlerwahrscheinlichkeit angegeben und ist beim mitteln eines Messpunktes der Wert im Feld „Estimated Accuracy“. Die Kreisfehlerwahrscheinlichkeit (CEP) gibt einen Kreisradius an, in der nach der Gaußschen Normalverteilung 50% der horizontalen Messwerte liegen. In der Realität liegen die Abweichungen zwischen Soll- und Ist-Wert nicht in einem Kreis mit dem Sollwert als Mittelpunkt, sondern in einer Ellipse. *Eine Beschreibung dieser Ellipse ist sehr unpraktisch, daher führte man den*

kreisförmigen CEP ein, der diese Fehlerellipse approximiert [Bauer 2003: S. 148]. Es wurden zwei Messkampagnen an verschieden Tagen unternommen, um einen Vergleich über die GPS Genauigkeit zu erhalten.

Sonde	Messzeitpunkt	Longitude	Latitude	CEP – Garmin Estimated Accuracy	Distanz zum amtlichen Fixpunkt (Sonde)
Sonde 1b	30.07.12 09:51:39	14.290543590	48.329473576	2m	1.71m
Sonde 8	30.07.12 10:11:17	14.291790817	48.329616906	2m	2.68m
Sonde 8	09.08.12 09:42:57	14.291808503	48.329596957	1.6m	3.94m
Sonde 8	09.08.12 10:03:51	14.291805737	48.329598214	1.7m	3.69m

Tabelle 4.9: Ergebnisse der GPS Messkampagne der Sondenvermessung

Das GPS Gerät wurde senkrecht auf den Grundwassersonden platziert und ein gemittelter Messpunkt von zirka eintausend Messwerten, jede Sekunde ein Messwert, über etwa fünfzehn Minuten aufgezeichnet.

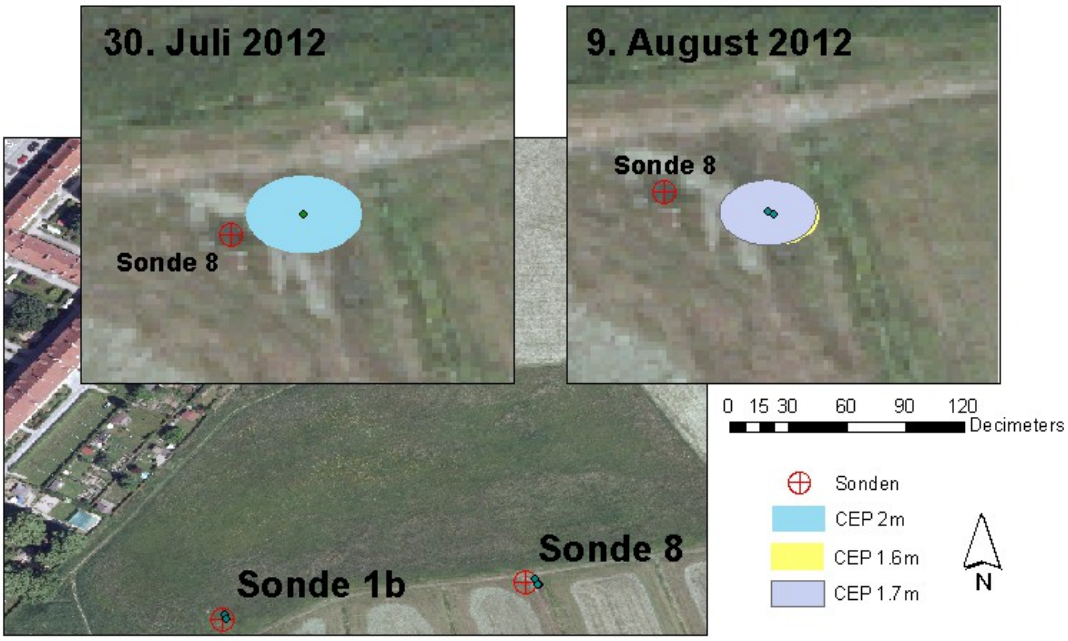


Abbildung 4.3: Vorvermessung der Fixpunkte mit Garmin GPSMap 60SCx

Die Ergebnisse sind in Tabelle 4.9 zusammengefasst. In der Abbildung 4.3 sind die Kreisfehlerwahrscheinlichkeiten bei der Vermessung der Sonde 8 als Buffer grafisch im Maßstab 1:60 dargestellt. Anhand der Voruntersuchungen erkennt man, dass die Abweichungen bei der Transformation der amtlichen Daten zwischen den verschiedenen Geodätischen Datum im Millimeterbereich liegen. Der Lagenunterschied der Fixpunkte zwischen den amtlichen Orthofotos und der Geodaten von Goolge Maps liegt im Meterbereich. Ebenfalls im Meterbereich liegt die Differenz zwischen den amtlichen Daten und der eigenen Vermessung mit einem gängigen GPS Gerät der Marke „Garmin GPSMap 60SCx“, welches bei Feldstudien im wissenschaftlichen Bereich eingesetzt wird.

4.3. Benötigte Datensätze für die Mikro-Drohne

Für den geplanten GIS Workflow ist für die Dokumention des Mikro-Drohnen Einsatzes, nach Durchführung eines Einsatzes, eine Archivierung der Daten des Einsatzes für eine nachfolgende optionale Postprozessierung geplant.

```
########    MikroKopter configuration file    ########
#-------------------------------------------------
# KML logging interval in ms (0 = disabled)
KMLLOGGING      = 500
# GPX logging interval in ms (0 = disabled)
GPXLOGGING      = 1000
# max. altitude in m
MAX_FLYING_ALT   = 0
# max. range in m
MAX_FLYING_RANGE = 0
# Auto-descend range in m (0 = disabled) (only comm. License)
DESCEND_RANGE    = 0
# GPS SBAS mode (0 = off, 1 = on)
GPS_SBAS_DGPS_ON = 1
# minimum time of the Waypoint-Event value (seconds)
MIN_EVENT_TIME   = 2
# GPS configmode (0 = off, 1 = on)
GPSAUTOCONFIG    = 1
```

Tabelle 4.10: SDCard SETTINGS.INI File [mikrokopter wiki gpx]

Die Mikro-Drohne produziert während eines Einsatzes laufend Statusdaten der verschiedenen Mikro-Drohnen Komponenten, sowie räumliche Daten und

Ereignisdaten, die in Beziehung zu den räumlichen Daten stehen. Die Daten, die sich auf die Komponenten beziehen, spiegeln die aktuellen Zustände der Mikro-Drohne, wie Stromverbrauch, Akkuspannung, Motorleistung, Steuerparameter für den Flug, Flugzeit, Temperaturen der Leistungskomponenten, interne Kommunikationsparameter zwischen den Komponenten. Die räumlichen Daten geben die Lage der Mikro-Drohne im dreidimensionalen Raum wieder, sowie Parameter die Einfluss auf die räumliche Lagegenauigkeit haben, wie die Anzahl der sichtbaren und verwendeten Navstar Satelliten und die Stärke des Erdmagnetfeldes. Die Ereignisdaten dokumentieren wenn Wegpunkte beim autonomen Flug erreicht werden und welches Ereignissignal, wie zum Beispiel das Einleiten einer Sensormessung, zu welchem Zeitpunkt wie lange dabei ausgelöst werden soll.

Beschreibung des GPX Trackpoint Tag des Mikrokopter Log File

Trackpoint GPX Tags	Inhalts-Beschreibung
trkpt	Attribute „lat= "„lon=" WGS84 Koordinaten
ele	GPS Höhe in Meter
time	GPS Zeit
sat	Anzahl der Satelliten für 3D Fix

Tabelle 4.11: Trackpointbeschreibung des GPX Logfile [mikrokopter wiki gpx]

Diese Daten stehen einerseits in Echtzeit über die Telemetrie Funkverbindung zur Verfügung, andererseits werden diese Daten auf einer optional mitgeführten Speicherkarte aufgezeichnet. Die Firmware der Mikro-Drohne kann auch dementsprechend parametrisiert werden, sodass ein Start der Mikro-Drohne ohne eingelegte Speicher SD Karte nicht möglich ist. Auf der Speicherkarte befindet sich eine Initialisierungsdatei für die Mikro-Drohne (Tab 4.10), in der man den Logging-Intervall und andere Parameter einstellen kann. Die räumlichen Daten des Mikro-Drohnen Einsatzes stehen nach einem Einsatz als KML und GPX Datenformat zur Verfügung. Die erweiterten Daten werden nur im GPX Format aufgezeichnet. Für diese Arbeit werden in weiterer Folge die Daten der GPX Datei für die Dokumentation des Mikro-Drohnen Einsatzes verwendet. Der Aufbau der GPX Extension ist wie folgt:

Beschreibung der GPX Extension des Mikrokopter GPX Logfile

GPX Extension Tags	Inhalts-Beschreibung
Altimeter	Barometrische Höhe in 5cm Schritten
Variometer	Steig- oder Sinkrate in cm/s
Course	GPS Kurs in Grad
GroundSpeed	Horizontale Geschwindigkeit über Boden in cm/s
VerticalSpeed	Vertikale Geschwindigkeit in cm/s
FlightTime	Flugzeit in Sekunden
Voltage	Spannung des Akkus in Volt
Current	Stromabgabe des Akkus in Ampere
Capacity	Verbrauchte Strommenge in mAh
RCQuality	Signalstärke der Fernsteuerung von 0-254
RCRSSI	Signalstärke ACT 35Mhz (nicht benutzt)
Compass	Gyroskop Stabilisiertes Kompassignal und Rohwert
NickAngle	Nick (Vor) Winkel in Grad
RollAngle	Roll (Seiten) Winkel in Grad
MagnetField	Magnetfeldstärke in Prozent zur Kalibration
MagnetInclination	Magnetische Inklination zum aktuellen Standort
MotorCurrent	Stromverbrauch in 0,1A Schritten pro Motor
BL_Temperature	Temperatur in Celsius der „Brushless Controller"
AvaiableMotorPower	Verfügbare Maximale Motorleistung von 0-254
FC_I2C_ErrorCounter	Fehlerzähler des seriellen I²C/TWI Kommunikationsbus der Motorsteuerung (Brushless Controller)
AnalogInputs	Analoge Eingangsspannung der Navigationskontrolle (digital 0-1024 = analog 0-3.3Volt)
NCFlag	Hexadezimaler BitCode der Navigationskontrolle Fehlermeldungen[2]

2

- •NC_FLAG_FREE 0x01 --> GPS off
- •NC_FLAG_PH 0x02 --> GPS in Position Hold mode
- •NC_FLAG_CH 0x04 --> GPS in Coming Home or Waypoint mode
- •NC_FLAG_RANGE_LIMIT 0x08 --> GPS Target Position is outside the range limit
- •NC_FLAG_NOSERIALLINK 0x10 --> The NC doesn't receive KopterTool-Data
- •NC_FLAG_TARGET_REACHED 0x20 --> GPS Waypoint reached
- •NC_FLAG_MANUAL_CONTROL 0x40 --> Position manual controlled
- •NC_FLAG_GPS_OK 0x80 --> Satfix

GPX Extension Tags	Inhalts-Beschreibung
Servo	Sollwert des Neigewinkels der KameraServo Steuerung in Grad <(Nick, Roll, Point of Interest oder Waypoint Nick)
WP	Name des aktuell angeflogenen Wegpunktes
FCFlags2	Hexadezimaler BitCode der Flugkontrolle Fehlermeldungen[3]
ErrorCode	Aktuelle Fehlermeldung des Kopter
TargetBearing	Zielausrichtung in Grad
TargetDistance	Entfernung zum Ziel
RCSticks	Werte der Remotecontroll: Nick, Roll,Yaw, Gas, Pot1-Pot8 (Wertebereich: 0-254)
GPSSticks	Einfluss der GPS Kontrolle auf das System (Nick, Roll, Gier, GPS-Status[4])

Tabelle 4.12: Extensionsbeschreibung des GPX Logfile [mikrokopter wiki gpx]

3

- •FC_STATUS_MOTOR_RUN 0x01 --> Motors are running
- •FC_STATUS_FLY 0x02 --> MK is Flying
- •FC_STATUS_CALIBRATE 0x04 --> MK is Calibrating
- •FC_STATUS_START 0x08 --> MK is Strating
- •FC_STATUS_EMERGENCY_LANDING 0x10 --> Emergency gas in case of RC-Lost
- •FC_STATUS_LOWBAT 0x20 --> Low Lipo-Voltage
- •FC_STATUS_VARIO_TRIM_UP 0x40 --> Altitude control in Vario-Mode trimming up
- •FC_STATUS_VARIO_TRIM_DOWN 0x80 --> Altitude control in Vario-Mode trimming down
- •FC_STATUS2_CAREFREE 0x01 --> Carefree is active
- •FC_STATUS2_ALTITUDE_CONTROL 0x02 --> Altitude control is active
- •FC_STATUS2_RC_FAILSAVE_ACTIVE 0x04 --> The FC has RC-Lost and the Failsafe-feature is active
- •FC_STATUS2_OUT1_ACTIVE 0x08 --> The Output 1 of the FC is active
- •FC_STATUS2_OUT2_ACTIVE 0x10 --> The Output 2 of the FC is active

4

- •'-' = no GPS fix
- •'/' = off
- •'?' = Coming home, but home Position unknown (goes to PH then)
- •'C' = Coming home
- •'W' = Flying Waypoints
- •'D' = Dynamic Position Hold
- •'P' = Position Hold
- •'m' = Manual controlled

4.4. Einzusetzende Werkzeuge

Für diese Arbeit wurde verschiedene Software eingesetzt und entwickelt. In diesem Kapitel wird die Software mit Ihren Funktionen beschrieben und ihr Einsatzgebiet umrissen.

- ArcGIS 10

  Das Desktop Geoinformationssystem von ESRI ist die zentrale Software für die Erstellung dieser Arbeit. Die ArcGIS Software Suite besteht aus dem ArcMap für die interaktiven visuellen Arbeitsschritte, dem Digitalisieren der Wegpunkte, dem ArcCatalog für die Verwaltung der Daten und den ArcTools mit dem Modelbuilder für die Transformationen, Übermittlung der Wegpunkte an die Mikro-Drohne und der Analyse der gewonnenen räumlichen Daten der Mikro-Drohne zur Dokumention. Mittels der Pythonscript Erweiterung von ArcGIS 10 werden die für diese Arbeit benötigten und programmierten Zusatzfunktionalitäten in das ArcGIS System angebunden.

- Eclipse Integrated Develop Enviroment mit PyDev Modul

  Mit dem Eclipse IDE wird die Erweiterung für die Kommunikation zwischen dem ArcGIS und der Mikro-Drohne mit der Pythonscript Sprache programmiert. Die verwendete Python Erweiterung für Eclipse ist das PyDev Plugin, welches über den Eclipse „Marketplace“ installiert wird. Das ArcGIS 10 Geoinformationssystem besitzt eine Python API Schnittstelle, die anhand des „ArcPy sitepackage“ implementiert ist. Dieses „Python sitepackage“ realisiert den Zugriff auf die ArcGIS Module und den ArcGIS Geoprozessor über ArcPy Module, Klassen und Funktionen. Python ist eine Objektorientierte Scriptsprache. Das bedeutet, dass der Quellcode sofort ausgeführt und getestet werden kann, ohne eine vorhergehende Übersetzung in Maschinencode. Python ist quelloffen, auf den meisten Systemen verfügbar und wird von ESRI „supported“. Für die Kommunikation mit der Mikro-Drohne wird das Python „PySerial sitepackage“ benötigt, um über die serielle Telemetrieverbindung die

Wegpunkte aus dem ArcGIS an die Mikro-Drohne zu senden.

- DNRGPS Applikation

  Die DNRGPS Applikation ist ein Software Werkzeug, um die gewonnenen räumlichen Daten des Mikro-Drohnen Einsatzes aus dem GPX File der Speicherkarte auszulesen, und in ein ArcGIS 10 kompatibles Format für die Dokumention und das Postprozessing zu speichern.. DNRGPS ist quelloffen, und wird vom „Minnesota Department of Natural Resources“ entwickelt und ist Online unter „http://www.mndnr.gov“ (letzter Zugriff 31.Oktober 2012) zu erhalten. Es unterstützt verschiedene räumliche Datenformate zum Lesen und Abspeichern, kann auf GIS Server oder Spatiale Datenbanken zugreifen, sowie auf ArcGIS 10. DNRGPS kann Feature Klassen untereinander konvertieren und zwischen verschiedenen Geodätischen Daten transformieren und projizieren. Des weiteren unterstützt DNRGPS grundlegende räumliche Funktionen wie Flächenberechnung, Umfangberechnung, Längenberechnung und räumlich statistische Operationen für eine Vorabschätzung. Die Attribute des Track von DNRGPS wurde an die Attribute des erweiterten GPX File (Tab. 4.12) der Mikro-Drohne angepasst und ist im Anhang (Tab. 9.3) einzusehen.

5. Einsatzplanung und Dokumentation

5.1. Methodik

Um eine Einsatzplanung aus einem Desktop GIS System abzuwickeln, sind eine Reihe aufeinanderfolgender Schritte sequentiell auszuführen, die den GIS Workflow des Mikro-Drohnen Einsatz definiert.

- Akquisition geeigneter Geodaten für das Einsatzgebiet
- Zusammenstellen der Geodaten im ArcGIS unter einer gemeinsamen geeigneter Projektion und Datum
- Erstellen einer Point Feature Klasse mit den passenden Attributen
- Digitalisieren und Attributieren der Wegpunkte als Point Feature
- Kontrolle der Wegpunkt-Attribute und Wegpunkt Sequenz
- Übertragen der Wegpunkt Sequenz an die Mikro-Drohne und Speichern des optionales Wegpunkt Files im proprietären Mikrokopter OSD Format
- Durchführen des Mikro-Drohnen Einsatz es
- Sichern der Mikro-Drohnen Daten der Speicherkarte
- Mit DNRGPS das gesicherte GPX File in ESRI Shape Format konvertieren
- Validierung, Dokumention und Archivierung des Einsatzes

Akquisition geeigneter Geodaten für das Einsatzgebiet

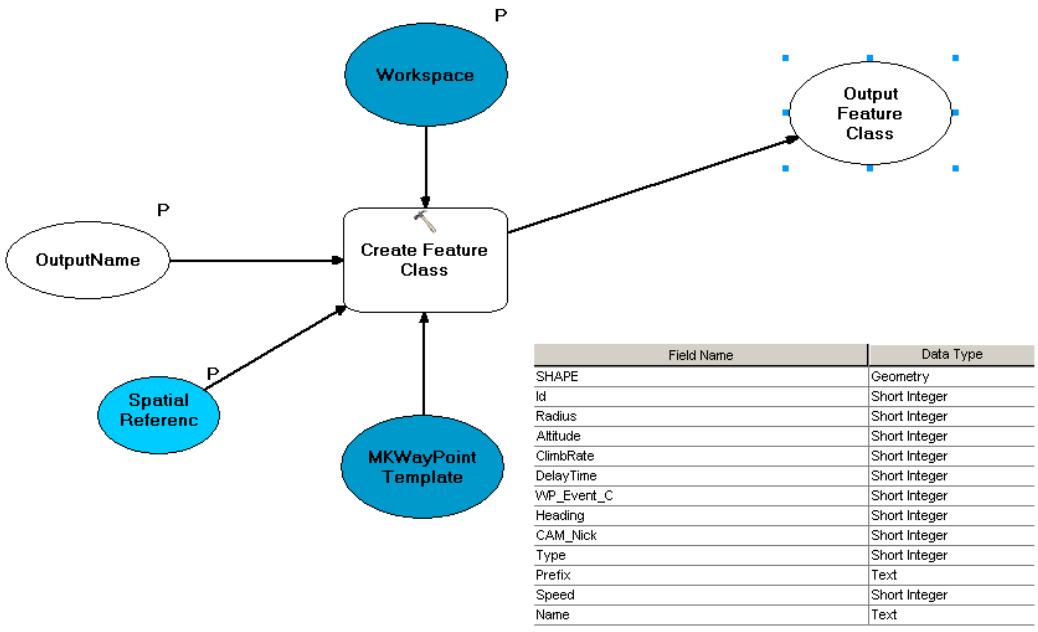
Als Basis für die Planung der Wegpunkte eines Einsatzes werden Orthofotos mit einer geeigneten Auflösung benötigt um das Einsatzgebiet, beziehungsweise die räumlichen Objekte oder Phänomene, die Ziel eines Wegpunktes sind, zu erkennen. Auf die topografische Aktualität der Geodaten ist zu achten damit keine gravierenden Änderungen in der Bebauung oder im Bewuchs die Einsatzplanung behindern oder unmöglich machen. Prinzipiell sind auch Online Karten als Basis für die

Wegpunktplanung möglich insofern sie der Planung der Wegpunkte genügen.

Zusammenstellen der Geodaten im ArcGIS unter einer gemeinsamen geeigneter Projektion und Datum

Da Geodaten aus verschiedenen Quellen stammen können, ist auf eine geeignete und gemeinsame Projektion und auf ein gemeinsames geodätisches Datum zu achten, um die räumliche Übereinstimmung der zusammengestellten Geodaten sicherzustellen. Als geeignete Projektionsbasis des ArcGIS Layer ist die Projektion der Orthofotos zu empfehlen, da diese dann verzerrungsfrei und maßstabsgetreu vorliegen. Zusätzlich in ArcGIS einzubindende Geodaten, wie Fixpunkte, sind vorab in die entsprechende Projektion des ArcGIS Layesr zu transformieren und projizieren.

Erstellen einer Point Feature Klasse mit den passenden Attributen



Field Name	Data Type
SHAPE	Geometry
Id	Short Integer
Radius	Short Integer
Altitude	Short Integer
ClimbRate	Short Integer
DelayTime	Short Integer
WP_Event_C	Short Integer
Heading	Short Integer
CAM_Nick	Short Integer
Type	Short Integer
Prefix	Text
Speed	Short Integer
Name	Text

Abbildung 5.1: Create WayPoint Template mit Attributen des Template

Um die Wegpunkte der Mikro-Drohne auf Grundlage der ArcGIS Basisdaten zu digitalisieren braucht man eine Point Feature Klasse mit den entsprechenden Attributen, welche die Wegpunkte der Mikro-Drohne repräsentiert und in der die Wegpunkte erstellt und attributiert werden. Für diese Arbeit wurde ein Point Feature Template erstellt, anhand dessen man eine Arbeitskopie mit der benötigten Projektion im Modelbuilder

erstellen kann.

Beschreibung des Wegpunkte File		
Name	Beschreibung	Vorkommen
[General]	Start des Wegpunkt Files	Einmalig
FileVersion	Software Version des OSD	Einmalig
NumberOfWaypoints	Anzahl der Wegpunkte im File	Einmalig
[Point*X*]	Wegpunktnummer (*X* ist Zahl 1 bis n)	Wiederholt
Latitude	Koordinate WGS84 Datum	Wiederholt
Longitude	Koordinate WGS84 Datum	Wiederholt
Radius	Kugel Radius in Meter um Wegpunkt, bei dessen erreichen der Wegpunkt als angeflogen gilt.	Wiederholt
Altitude	Wegpunkt Höhe in Meter relativ zur Starthöhe	Wiederholt
ClimbRate	Steig- oder Sinkrate in 0,1m/s Schritten, wenn Wegpunkte in verschiedener Höhe angeflogen werden.	Wiederholt
DelayTime	Verweildauer am Wegpunkt in Sekunden an einem Wegpunkt, bevor ein nächster angeflogen wird.	Wiederholt
WP_Event_Channel_Value	Schaltzeit in 10ms Schritten des Servoausgangs SV2.1	Wiederholt
Heading	Ausrichtung des Kopter von 1-360° Bei 0 richtet sich der Kopter nicht aus Es kann ein anderer Wegpunkt oder Point of Interest angegeben werden mit P*X,* wobei *X* die Wegpunkt Nummer ist.	Wiederholt
Speed	Anfluggeschwindigkeit zum Wegpunkt in 0,1m/s Schritten. Max. 6m/s	Wiederholt
CAM-Nick	Winkel des Nick Servo einer Payload Halterung. Die aktuelle Nick Einstellung beim Start gibt 0° vor.	Wiederholt
Type	1: Wegpunkt 2: Point of Interest	Wiederholt
Prefix	Optionaler Präfix für Wegpunkte	Wiederholt

Tabelle 5.1: Aufbau des Wegpunkte File [mikrokopter wiki wp]

Als Projektion wird die Projektion des verwendetet ArcGIS Layers gewählt. Es soll ein

adäquater Name für die zu erstellende Point Feature Klasse gewählt werden und als Speicherort eine ArcGIS Geodatabase. *Als Geodatabase soll eine File Geodatabase der Personal Geodatabase möglichst vorgezogen werden* [arcgishelp gdb]. Die Attribute der Point Feature Klasse entsprechen den Attributen des Mikrokopter Waypoint Files, dessen Datentypen in Tabelle 5.1 aufgelistet und erklärt sind.

Digitalisieren und Attributieren der Wegpunkte als Point Feature und attributieren

Anhand des Basis Layers im ArcGIS werden die Wegpunkte in der erstellten Point Feature Klasse digitalisiert. Beim Digitalisieren müssen die Attribute der Wegpunkte parametrisiert werden.

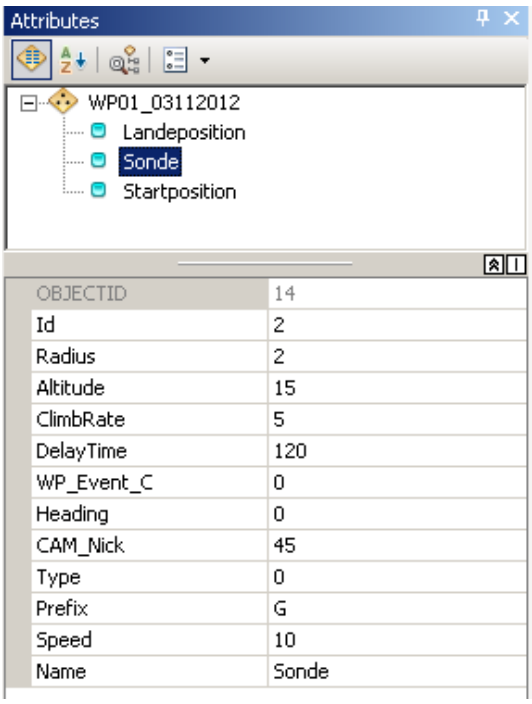


Abbildung 5.2: Attribute des Wegpunkt Feature

Das wichtigste Feld ist die „ID“, welches die fortlaufende Sequenz der Wegpunkte repräsentiert und sequentiell von $1 \rightarrow n$ aufsteigend nummeriert werden muss. In dieser Reihenfolge werden die Wegpunkte vom Navigations Controller der Mikro-Drohne angesteuert. Das „Name“ Feld ist optional und dient zur Orientierung des Bearbeiters. Die restlichen Felder sind mit Standardwerten belegt und können nach den

erforderlichen Gegebenheiten anhand der Parameterbeschreibung in Tabelle 5.1 angepasst werden. Zu beachten ist, dass beim Feld „Type“ ein Wert von 0 einem Wegpunkt entspricht und ein Wert von 1 einem „Point of Interest“. Beim Setzten der Wegpunkte ist darauf zu achten, dass Hindernisse durch Setzten zusätzlicher Wegpunkte umflogen werden, da die Mikro-Drohne zwischen zwei Wegpunkten auf einer Geraden steuert.

Kontrolle der Wegpunkt-Attribute und Wegpunkt Sequenz

Nach dem Fertigstellen der Wegpunkte ist gegebenenfalls eine Kontrolle der Attribute der einzelnen Point Features zu empfehlen, damit die Parameterwerte plausibel sind und die Sequenz des „ID“ Feldes eingehalten wird. Auch eine Kontrolle der Wegpunkt Reihenfolge anhand des „ID“ Feldes soll überprüft werden. Empfehlenswert ist es, einen Buffer von 250 Meter Radius um den Startpunkt zu setzten, da bei der freien Lizenz der Navigations Controller Firmware die Einsatzdistanz der Mikro-Drohne vom Startpunkt mit 250 Meter Radius beschränkt ist. Damit wird sichergestellt, dass die Mikro-Drohne ihr Einsatzgebiet nicht überschreitet. Eine visuelle Kontrolle ob Hindernisse umflogen werden soll, nach Möglichkeit mit den realen Gegebenheiten überprüft werden.

Übertragen der Wegpunkt Sequenz an die Mikro-Drohne und Speichern des optionalen Wegpunkt Files im proprietären Mikrokopter OSD Format

Als Teil dieser Arbeit wurde ein Python Programm geschrieben, das die digitalisierten Wegpunkte aus der Feature Klasse entweder in ein proprietäres Mikrokopter Wegpunkt File schreibt (Abb. 5.4), oder mittels serieller Telemetrie Funkverbindung direkt an die Mikro-Drohne aus ArcGIS heraus sendet und optional das proprietäres Mikrokopter Wegpunkt File schreibt (Abb. 5.3). Dazu wird die Waypoint Feature Klasse, von der im Arbeitslayer verwendeten Projektion, in das WGS84 Geografische Koordinatensystem um-projiziert und transformiert. Die temporäre resultierende Featureklasse wird im Arbeitsspeicher „in_memory\wpfc“ zwischengespeichert. In dieser Arbeit wird beim Digitalisieren das MGI Datum – EPSG Code 4805 verwendet. Daher sind die

Transformationsparameter aus Tabelle 4.4 für die Transformation von MGI nach WGS84 im Modelbuilder vorgegeben. Bei Verwendung eines abweichenden Geografischen Datum muss dies im Modelbuilder angepasst werden.

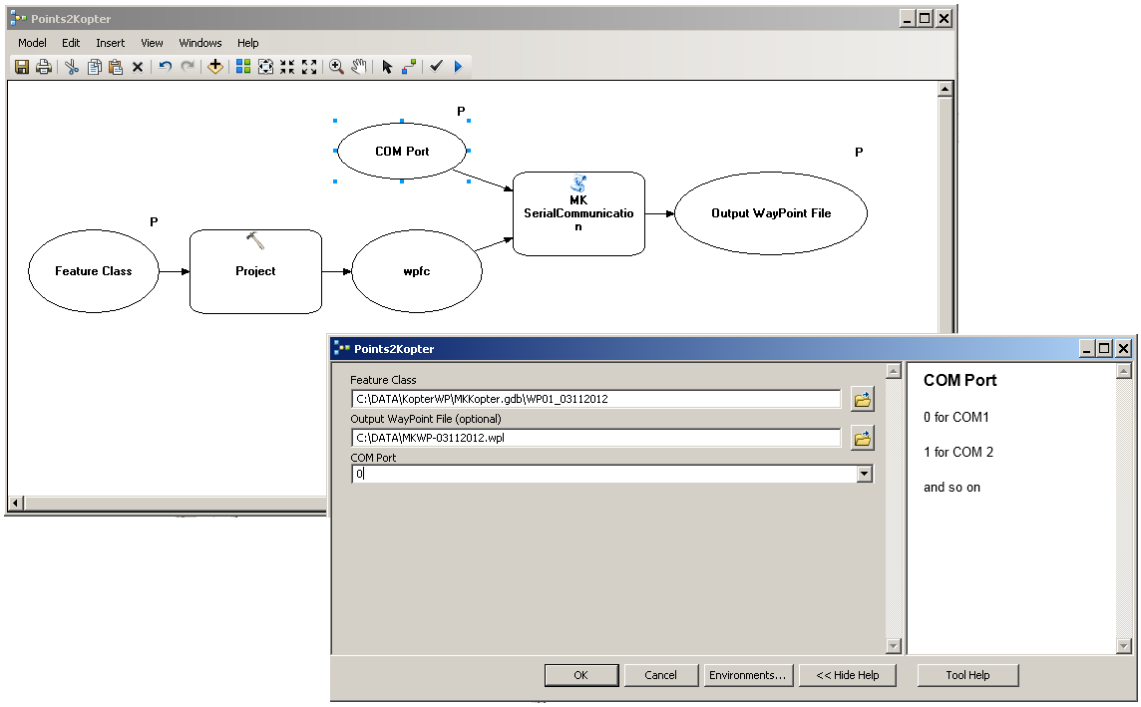


Abbildung 5.3: Waypoints to Kopter Modelbuilder

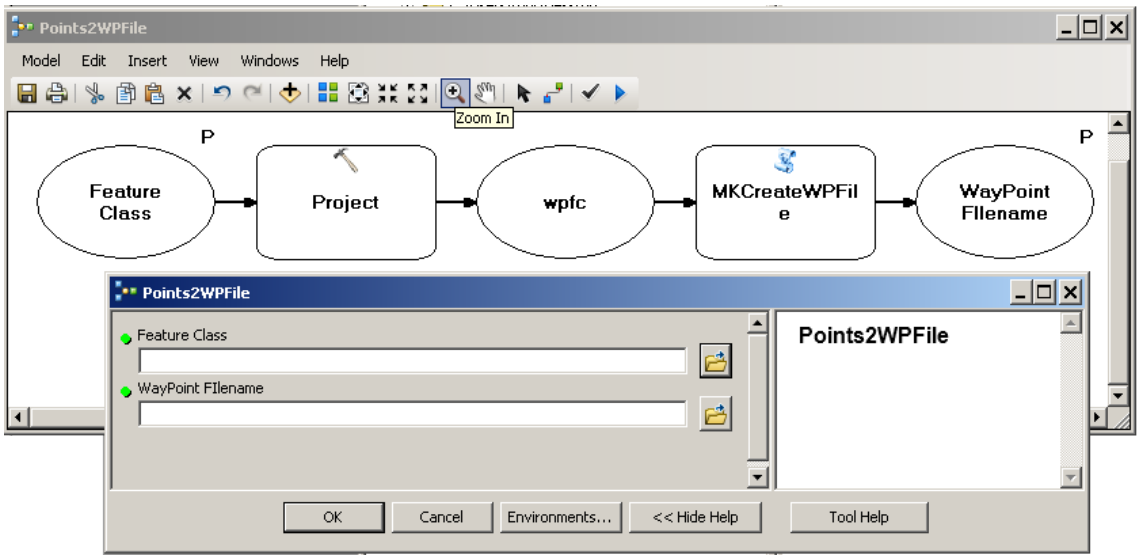


Abbildung 5.4: Waypoints to WPFile

Als Ergebnis dieser Operationen erhält man ein Wegpunkt-File und die Wegpunkt Sequenz im Navigations Controller der Mikro-Drohne. Als Kontrollmöglichkeit kann man im Mikrokopter OSD die Wegpunkt-Sequenz vom Navigations Controller in das Mikrokopter Programm laden. Ein optional vorbereitetes proprietäres Mikrokopter Wegpunkt-File lässt sich zu einem späteren Zeitpunkt bei einem Feldeinsatz mit dem Mikrokopter OSD in den Navigations Controller laden.

Durchführen des Mikro-Drohnen Einsatz

Sobald die Wegpunkt-Sequenz bereit steht kann der Mikro-Drohnen Einsatz absolviert werden. Es ist ratsam vor dem Start der Mikro-Drohne die Klarliste beziehungsweise die Checkliste aus Kapitel 3.4 abzuarbeiten. Es ist unbedingt darauf zu achten das die SD Speicherkarte für die Dokumention des Mikro-Drohnen Einsatzes im Navigations Controller eingelegt ist.

Sichern der Mikro-Drohnen Daten der Speicherkarte

Nach dem Mikro-Drohnen Einsatz sind die gewonnenen Koordinaten des Einsatzes auf der SD Speicherkarte unbedingt zu sichern und wenn möglich eine Kopie davon anzulegen. Die Einsatzdaten liegen auf der Speicherkarte als GPX Format und im KML Format vor und sind in Kapitel 4.3 beschrieben.

Mit DNRGPS das gesicherte GPX File in ESRI Shape Format konvertieren

Das GPX File des Mikro-Drohneneinsatzes wird mit der DNRGPS Applikation geladen und als Shapefile oder in eine ArcGIS File Geodatabase abgespeichert. Für das Abspeichern in der Geodatabase muss eine Verbindung definiert werden und die GPX Daten werden als Point Layer darin abgespeichert. Die erweiterten Attribute des GPX File (Tab 4.12) müssen als DNRGPS Attribute (Tab. 9.3) vorab definiert werden und werden persistent in der DNRGPS Konfiguration abgespeichert.

Validierung, Dokumention und Archivierung des Einsatzes

Die abschließenden Schritte des Mikro-Drohnen Einsatzes werden in Kapitel 6 behandelt und anhand eines Experimentes vorgeführt.

5.2. Programmierung und Erstellen der ArcGIS Toolbox Scripten

Für die Erstellung des proprietäres Mikrokopter Wegpunkt Files und die direkte Übertragung der digitalisierten Wegpunkte der Point Feature Klasse aus dem ArcGIS wurden zwei Python Scripte programmiert, die diese Funktionalitäten implementieren. ArcGIS besitzt eine mächtige Python Schnittstelle, die Zugriff auf den Geoprozessor von ArcGIS bietet und wird von ESRI ArcGIS als Erweiterung für eigene Funktionalitäten unterstützt und propagiert. Das Ziel der Scripten ist die Möglichkeit eine indirekte Kommunikation mittels des Mikrokopter Wegpunkt Files und eine direkte Kommunikation über den seriellen Telemetrie Link aus dem ArcGIS zur Verfügung zu stellen, um die für den Mikro-Drohneneinsatz digitalisierte Wegpunkt Liste an den Navigations Controller der Mikro-Drohne zu übertragen.

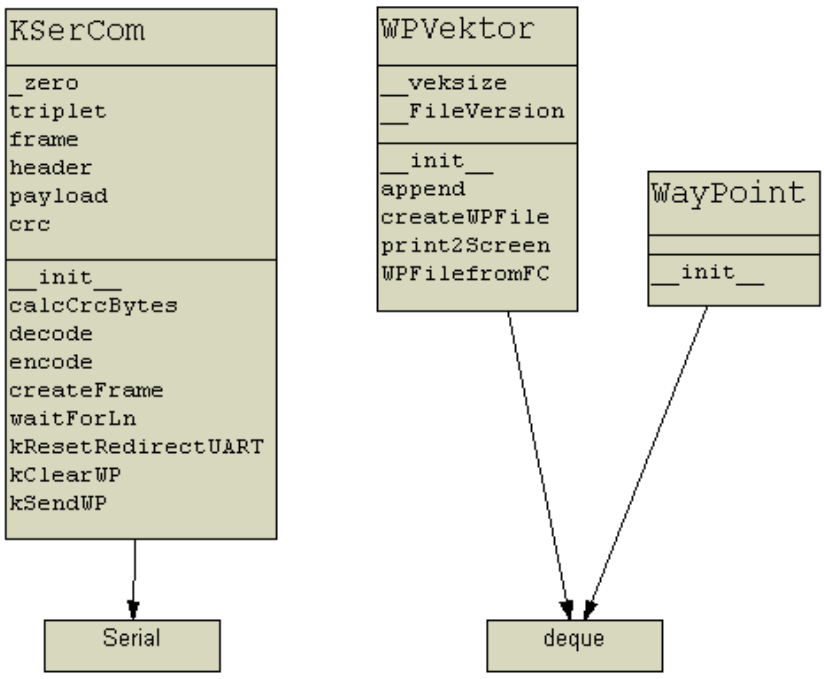


Abbildung 5.5: UML Diagramm

Für die indirekte Kommunikation wird das erstellte Mikrokopter File im Mikrokopter Programm geladen und an die Mikro-Drohne übertragen. Für die direkte Kommunikation aus dem ArcGIS werden die erstellten Wegpunkt Features direkt über

den seriellen Telemetrie Link an den Navigations Controller gesendet. Die erstellten Python Scripts werden als ArcGIS Toolbox im ArcGIS zur Verfügung gestellt. Das Ergebnis der Programmierung der beiden Scripts sind zwei Toolboxen die im Modelbuilder verwendet werden können.

- MKCreateWPFile mit zugrundeliegenden CreateWPFile.py Script
- MKSerialCommunikation mit zugrundeliegenden SerCommunication.py Script

In Abbildung 5.5. ist das UML Diagramm der Klassen beider Scripten abgebildet, wobei die Container Klassen des CreateWPFile.py Script im Script, welches die serielle Kommunikation implementiert, wiederverwendet werden.

5.2.1. MKCreateWPFile - Entwurf und Implementierung

Das MKCreateWPFile Script erstellt aus den Wegpunkt Features ein Wegpunktefile, das direkt in das proprietäre Mikrokopter OSD Programm geladen werden kann. Dazu werden die räumlichen Koordinaten und die Attribute der einzelnen Wegpunkte in ein Textfile geschrieben, das den Aufbau aus Tabelle 5.1 besitzt. Nach einem einmalig vorkommenden Header werden die Punkte in ihrer aufsteigenden Reihenfolge mit den zu jedem Punkt gehörenden zwölf Attributen geschrieben. Um dies zu realisieren müssen in der Toolbox zwei Parameter an das Script übergeben werden, die Point Feature Klasse mit den Wegpunkten, und eine Pfadangabe und ein Dateiname im Filesystem, wo das Wegpunkte Textfile gespeichert werden muss. Die Übergabe aus der Toolbox an das Script erfolgt mit einer Funktion des zentralen „arcpy“ Moduls, der „arcppy.GetParameterAsText(int index)“ das als Rückgabewert ein String Objekt liefert. Bei den Toolbox Eigenschaften wird der Datentyp des Übergabeparameters im Reiter Parameter konfiguriert (Abb 5.6). Anschließend werden die Point Features anhand des ID Feldes aufsteigend sortiert und in eine temporäre Klasse zwischengespeichert. Dafür wird die Methode „arcpy.Sort_management(inputFeatureClass, outputFeatureClass, [["Id", "ASCENDING"]]) „ des Datamanagement Toolset verwendet. Der nächste Schritt ist die Erzeugung eines „SearchCursor“ Objektes der sortierten Feature Klasse mit dem man Zugriff auf die Features der Klasse hat. Die Daten der Point Feature

Klasse liegen gedacht als zweidimensionaler Array vor. Eine Dimension beinhaltet die Features und die zweite Dimension die Attribute der einzelnen Features. Um die Daten einfacher handhaben zu können und eine Wiederverwendung der Funktionalität zu ermöglichen wurden zwei Klassen definiert.

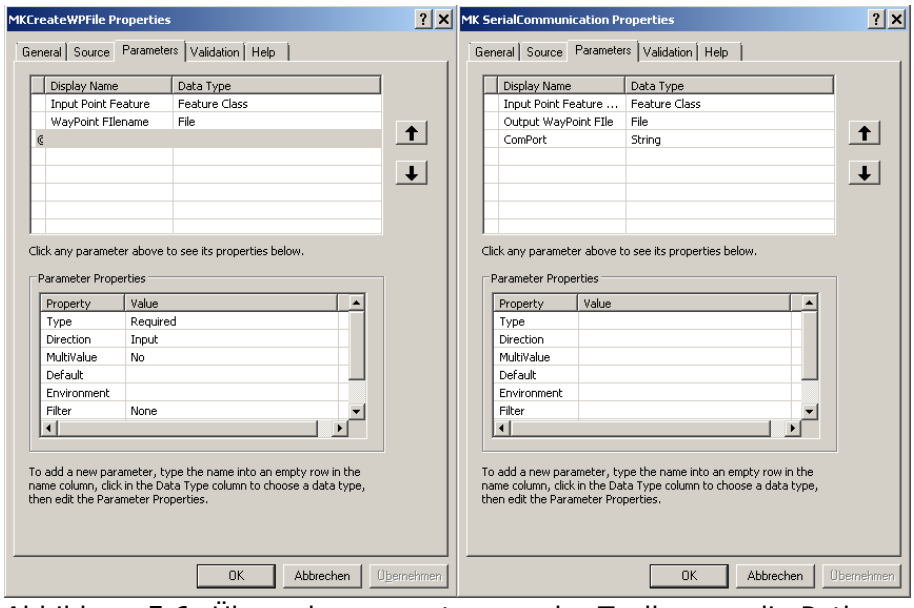


Abbildung 5.6: Übergabeparameter aus der Toolbox an die Python Scripte

Die Klasse „WayPoint“, wurde von der Basisklasse „deque“ abgeleitet und die Klasse „WPVektor“, wurde ebenfalls von der Daten Container Klasse „deque“ abgeleitet. Die Basis-Containerklasse vererbt die wichtigsten Methoden, um die Wegpunkte mit ihren Attributen zu verwalten. Bei der „WayPoint“ Klasse wurde der Constructor um die Übergabeparameter-Liste der Wegpunkt Attributte angepasst. In dieser Klasse werden die einzelnen Point Features als Objekte gespeichert, wobei ein Objekt ein einzelnes Point Feature repräsentiert. Die Klasse „WPVektor“ repräsentiert die Point Feature Klasse. Die Klassenmethode „WPVektor.append“ wurde um eine Zählervariable erweitert, welche die Anzahl der gespeicherten Point Features zählt. Diese Variable wird für die Adressierung der einzelnen Waypoints im Mikrokopter Waypoint File benötigt. Mit dem SearchCursor wird in einer Schleife auf die Point Features zugegriffen und aus ihnen ein WayPoint Objekt erzeugt, das die Attribute eines Point Features als Objekt vom jeweiligen Datentyp beinhaltet. Das erzeugte WayPoint Container Objekt mit den

Objekt-Attributen wird im „WPVektor“ Container eingefügt. Die Schleife wird bis zum letzten Point Feature durchlaufen. Für die Erstellung des Mikrokopter Wegpunkte Textfile wurde die Methode „WPVektor.createWPFile(file)“ erstellt. Die Methode schreibt einmalig den Wegpunkt Header und in einer Schleife die einzelnen Wegpunkte in das Textfile. Mit Hilfe des Mikrokopter OSD kann nun das Wegpunkt File geladen werden, visuell überprüft und die Attribute gegebenenfalls verglichen und an die Mikro-Drohne gesendet werden. Die Implementierung des „MKCreateWPFile.py“ ist im Anhang in Tabelle 9.2 nachzulesen.

5.2.2. MKSerialCommunikation - Entwurf und Implementierung

Das Script für die serielle Kommunikation wurde anhand des „MKCreateWPFile.py“ Scripts erweitert. Es beinhaltet die volle Funktionalität des „MKCreateWPFile.py“ und das Wegpunkte File ist optional zu schreiben möglich (Abb. 5.3). Für den Zugriff auf die serielle Schnittstelle mit Python benötigt man das „pyserial“ Python sitepackage. Aus Kompatibilitätsgründen ist die Version 2.4 zu installieren. Das Script implementiert das Mikrokopter Serielle Protokoll. Mit dem Mikrokopter Seriellen Protokoll ist eine Bidirektionale Kommunikation mit der Mikro-Drohne über den Seriellen Telemetrie Link möglich. Ein Protokoll Frame besitzt den Aufbau der in Tabelle 5.2 angeführt ist.

Start-Byte	Adress-Byte	Kommando	Payload base64 modifizert	CRC Byte1	CRC Byte2	Stop-Byte
'#'	'a'+ AdrByte	Byte	Variable Länge der Daten	Byte	Byte	'\r'
Any Adress-Byte = 0 (a) Flight Controll Adress-Byte =1 (b) Navigations Controller Adress-Byte =2 (c) Magnetischer Kompass Adress-Byte = 3 (d)						

Tabelle 5.2: Generischer Aufbau des Seriellen Protokoll Frame [mkser]

Jedes Protokoll Frame beginnt mit einem Start Byte gefolgt vom Adresse-Byte, das die jeweilige Komponente der Mikro-Drohne, für welche die Nachricht bestimmt ist,

adressiert. Danach kommt ein Kommandobyte. Diese drei Bytes bilden den Header des seriellen Frame. Nach dem Header folgt der Payload mit den kommandospezifischen Daten, die „base64“ kodiert sind. Das Frame wird mit zwei Prüfsummen Bytes und einem '\r' Byte abgeschlossen. Die Prüfsumme wird über die Summe der Bytes des Headers und den kodierten Payload Daten modulo 4096 mittels Divison ohne Restwert (erstes Prüfsummen Byte) und Modulo (zweites Prüfsummen Byte) mit 64 als Divisor berechnet. Die Prüfsumme dient der Sicherstellung der Integrität der übertragenen Daten an die Mikro-Drohne und umgekehrt.

$$CRC_1 = (\sum (header + payload)\, modulo\, 4096)/64$$

$$CRC_2 = (\sum (header + payload)\, modulo\, 4096)\, modulo\, 64$$

Die „base64“ basierende Kodierung dient einer gemeinsamen Repräsentation der Daten auf verschiedenen Systemen, da die zu übertragenden Bytes auf anderen Systemen als Personal Computer, wie Smartphones, PDA's, in einer anderen Codeseite intern repräsentiert werden könnten. Dabei werden jeweils drei Bytes zu einem Triplet zusammengefasst das aus vierundzwanzig Bits besteht und bei jedem sechsten Bit in vier Stück zu je sechs Bit geteilt. Diese vier „Sixbits“ werden in vier Byte übertragen und dazu 64_{10} addiert. Die Dekodierung beim Empfänger erfolgt analog inverse, indem man 64_{10} vom Byte subtrahiert, die zwei führenden Bits, die 0 sind verwirft und die vier resultierenden „Sixbits“ zu drei Bytes zusammenfasst.

Kommando Frame an Navigations Controller – Setze Wegpunkt			Antwort Frame von Navigations Controller – Nummer des Wegpunkt		
Adress-Byte	Kommando	Daten	Adress-Byte	Kommando	Daten
'c' entspricht ('a'+2)	'w'	Waypoint Struct	'c' entspricht ('a'+2)	'W'	Anzahl desr Waypoints

Tabelle 5.3: Send Waypoint Frame [mkser]

Im Speziellen werden bei dieser Arbeit Wegpunkte an die Mikro-Drohne übertragen. Das serielle Kommando dafür ist in Tabelle 5.3 dargestellt. Nach dem Startbyte „#“

wird der Navigations Controller, der die Wegpunkte verarbeitet, mit Adress-Byte „c“ adressiert und das Kommando zum Setzen eines Wegpunktes „w“ gesetzt. Der Header des Frames ist somit „#cw“. Als Payload-Daten wird ein Waypoint C-Struct an den Header angefügt, der base64 kodiert ist. Über den Header und den kodierten Payload werden die Prüfsummen berechnet und hinten angehängt. Abgeschlossen wird das Frame mit einem „\r“.

```
typedef struct
{
  s32 Longitude;  // in 1E-7 deg
  s32 Latitude;   // in 1E-7 deg
  s32 Altitude;   // in mm
  u8 Status;// validity of data
} __attribute__((packed)) GPS_Pos_t;
```

Tabelle 5.4: GPS Position C Struct [mksvn]

Python besitzt im Sprachumfang eine Bibliothek für die Erstellung und Interpretation von C-Structs [python struct].

```
typedef struct
{
  GPS_Pos_t Position;// the gps position of the waypoint, see ubx.h for details
  s16 Heading;       // orientation, 0 no action, 1...360 fix heading, neg. = Index
                     // to POI in WP List
  u8 ToleranceRadius;// in meters, if the MK is within that range around the target,
                     // then the next target is triggered
  u8 HoldTime;       // in seconds, if the was once in the tolerance area around a WP,
                     //this time defines the delay before the next WP is triggered
  u8 Event_Flag;     // future implementation
  u8 Index;          // to indentify different waypoints, workaround for bad
                     // communications PC <-> NC
  u8 Type;           // typeof Waypoint
  u8 WP_EventChannelValue;  // Will be transferred to the FC and can be used as Poti
                            //value there
  u8 AltitudeRate;   // rate to change the setpoint
  u8 Speed;          // rate to change the Position(0 = max)
  u8 CamAngle;       // Camera servo angle in degree (255 -> POI-Automatic)
  u8  Name[4];       // Name of that point (ASCII)
  u8  reserve[2];    // reserve
} __attribute__((packed)) Point_t;
```

Tabelle 5.5: Waypoint C Struct [mksvn]

Ein C-Struct ist eine Datenkapsel für die Erstellung, Speicherung und den Transport von gepackten Binärdaten und stammt ursprünglich von der Programmiersprache C ab. Der Waypoint Struct beinhaltet die Attribute des Wegpunktes mit seinen Datentypen. Der Aufbau des Waypoint Struct mit dem beinhalteten GPS Position Struct ist in Tabelle 5.5 und Tabelle 5.4 dargestellt und wurde dem Quellcode der Navigations Controller

Firmware entnommen. Zu beachten ist, dass die WGS84 Koordinaten des GPS Position Struct als 32Bit Integer Zahl mit Vorzeichen an die Mikro-Drohne übertragen wird. Dazu werden die Koordinaten mit sieben Nachkommastellen vor dem packen in die Struct Datenkapsel mit 10^7 multipliziert. Das, ausgehend vom „MKCreateWPFile.py“ erweiterte Programm besitzt drei Übergabeparameter aus der Toolbox und wurde um die Angabe der seriellen Schnittstelle, an dem die Telemetrieverbindung zur Mikro-Drohne anliegt, erweitert. Die Angabe zur Erzeugung des proprietären Mikrokopter Wegpunkt File ist optional (Abb 5.6).

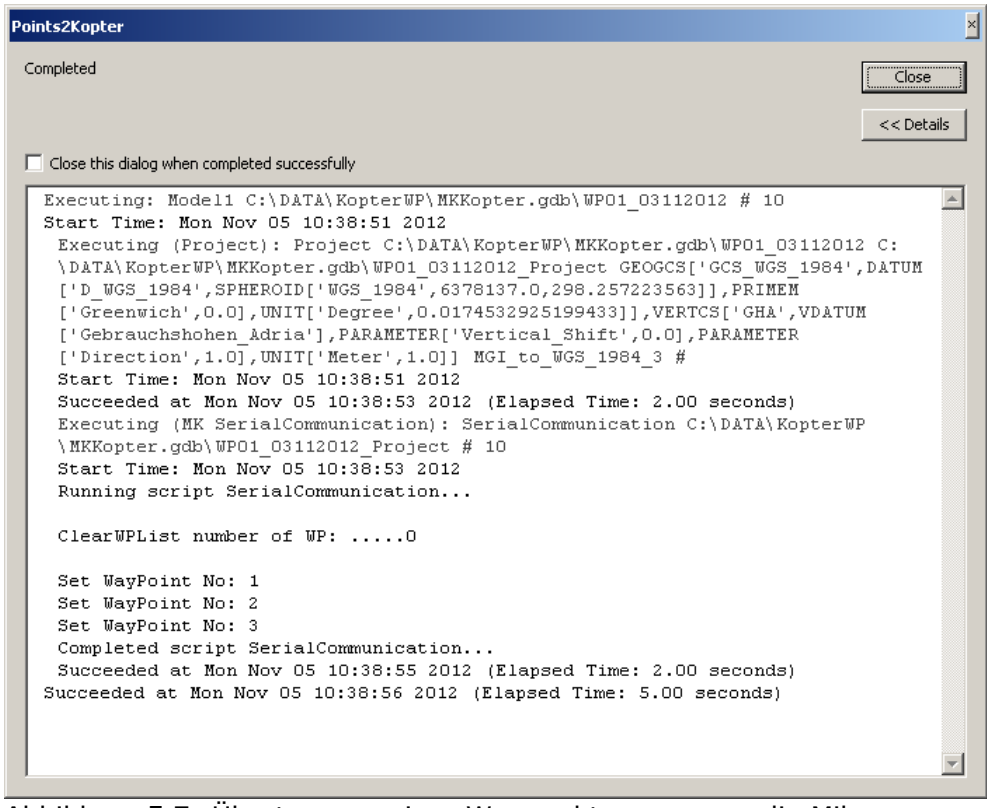


Abbildung 5.7: Übertragung einer Wegpunktsequenz an die Mikro-Drohne aus ArcGIS

Das Programm prüft vorab, ob die Feature Klasse vom Type Point ist und ob die Klasse in der Geografischen Projektion WGS84 vorliegt, da hier im Gegensatz zur Übertragung der Wegpunkte aus dem Mikrokopter OSD keine visuelle Kontrolle der Lage der Wegpunkte möglich ist. Durch Senden eines leeren Waypoint Struct an die Mikro-Drohne werden schon gesetzte Wegpunkte im Navigations Controller gelöscht.

In der Schleife werden mit dem SearchCursor, wie im Kapitel „5.2.1 MKCreateWPFile - Entwurf und Implementierung“ beschrieben, die Point Features gelesen und die Parameter an die Klassenmethode „kSendWP“ übergeben. Diese Methode erzeugt aus den Parametern den Waypoint Struct und mit weiteren Methoden das fertig kodierte Datenframe. Dieses wird über den seriellen Link an die Mikro-Drohne übertragen. Als Bestätigung für eine erfolgreiche Übertragung sendet die Mikro-Drohne ein Antwort frame mit der Sequenznummer des in der Mikro-Drohne gespeicherten Wegpunktes (Abb. 5.7). Bei einer fehlerhaften Verbindung bricht das Script nach zwanzig Sekunden mit einer Fehlermeldung ab. Dies wird durch Setzten der seriellen Klassenvariable „myKser.setTimeout(20)“ erzwungen. Andernfalls würde das Script in eine Endlosschleife geraten, was ist nur durch erzwungenes Beenden des ArcGIS Prozesses zu bereinigen ist.

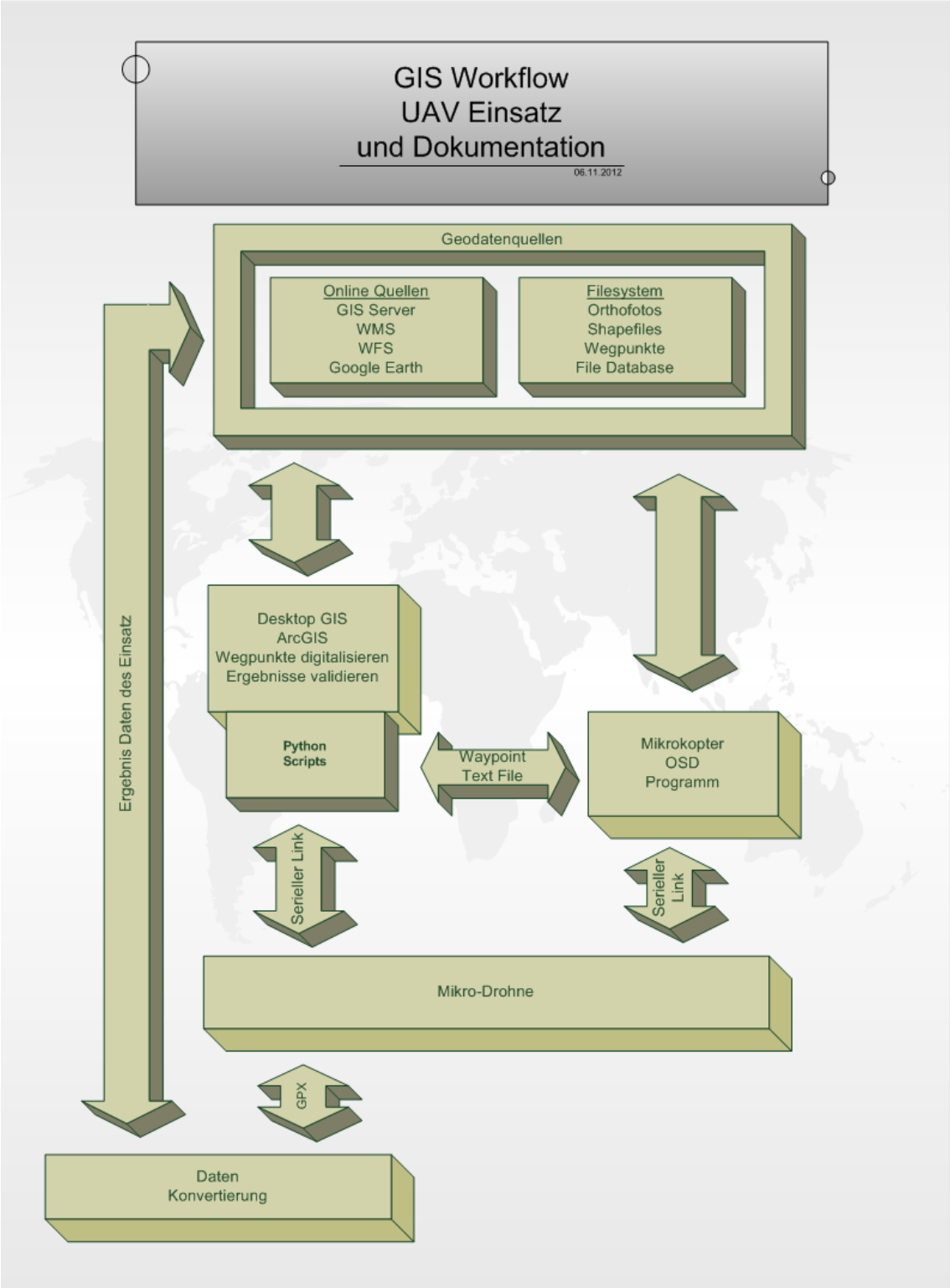


Abbildung 5.8: GIS Workflow UAV Einsatz und Dokumentation

6. Validierung

6.1. Theorie der Fehlerermittlung

Die Genauigkeit der GPS Messungen hängt von etlichen Faktoren ab. Wie in Kapitel 3.1.3 – GPS Modul behandelt, errechnet sich eine gemessene Koordinate anhand der Streckenmessungen zu vier GPS Satelliten durch Lösen der Gleichung mit den vier unbekannten Variablen. Dabei wird die Laufzeit T_0 des vom Satelliten ausgestrahlten Signals bis zum GPS Empfänger gemessen. Unter der Voraussetzung, dass alle Satelliten synchron mit der gleichen Zeit arbeiten, kann man die Formel aus Kapitel 3.1.3 – GPS Modul $(x_e-x_0)^2+(y_e-y_0)^2+(z_e-z_0)^2=(T_0c+tc)^2$ [Bauer 2003: S. 189] nach der Zeit substituieren. Die Variabel tc ist der noch unbekannte Zeitfehler, die Verarbeitungsverzögerung, des GPS Empfängers, wobei der Faktor c die Lichtgeschwindigkeit ist. Daraus folgt die Pseudostrecke r_0 aus $(r_0+K)^2=(T_0c+tc)^2$. Die Satellitenpositionen und Zeiten x_e, y_e, z_e, T_0 werden mit dem ausgestrahlten Signal der Satelliten im Signal kodiert mitgesendet. Wie ersichtlich ist, treten bei der Pseudostreckenmessung und Berechnung Fehler auf, deren Quellen verschieden Ursachen haben.

- Fehler aus falschen Satellitenbahnen (Ephemeridenfehler)
- Zeitfehler der Satelliten
- Geometrische Konstellation der Satelliten
- Störungen in der Ausbreitung des Satellitensignals
- Fehler im Empfänger

Die Summe der Fehler wird als „Dilution of Precision“ (DOP) bezeichnet, die sich aus fünf Faktoren zusammensetzt. Mit der Varianz $\sigma(r)$ der gemessenen Pseudostrecke r_0 lassen sich die Fehler m beschreiben.

$$m_g=\sigma(r)*GDOP \quad Geometrical\ \ Dilution\ of\ Precision$$

$$m_p = \sigma(r) * PDOP \quad \textit{Position Dilution of Precision}$$

$$m_h = \sigma(r) * HDOP \quad \textit{Horizontal Dilution of Precision}$$

$$m_v = \sigma(r) * VDOP \quad \textit{Vertical Dilution of Precision}$$

$$m_t = \sigma(r) * TDOP \quad \textit{Time Dilution of Precision}$$

Die Herleitung der DOP Werte geschieht mit Hilfe der Koeffizenten des linearen Gleichungssystems aus der Positionsbestimmung und Bildung der Varianz-Kovarianzmatrix des Gleichungsystems [Bauer 2003: S. 234ff].

Um die Genauigkeit durch Fehlereliminierung zu erhöhen wurde das Differentielle GPS eingeführt. Das Differentielle GPS ist ein Verfahren, bei dem zusätzliche Korrekturdaten für die Satellitenbahnen und das aktuelle Laufzeitverhalten der GPS Signale von Referenzstationen ausgestrahlt werden, beziehungsweise die Korrekturdaten für ein Postprozessing der gewonnen Messdaten im nachhinein zur Verfügung gestellt werden. Die Referenzstation mit genau bekannter Koordinate misst die lokale GPS Koordinaten und kann anhand seiner genau bekannten Koordinate die Fehlerdifferenz berechnen. Diese Korrekturdaten werden in Echtzeit zur Verfügung gestellt und für eine nachträgliche Korrektur aufgezeichnet. Die Übertragung der Korrekturdaten kann in einem eigenen Funk Frequenzband (Langwelle) oder über Internetdienste mittels des zugrundeliegenden TCP/IP Protokollstack mit NTRIP (Networked Transport of RTCM via Internet Protocol) erfolgen. Als Referenzstation kann ein eigenes DGPS System mit einem Zweiten vermessenen GPS Empfänger verwendet werden oder auf bestehende DGPS Dienste zurückgegriffen werden. Zu beachten ist, dass die Qualität der Korrekturdaten mit der Entfernung zwischen Mess- und Referenzstation abnimmt. In Österreich betreibt das Bundesamt für Eich- und Vermessungswesen das kostenpflichtige „Austrian Positioning Service – APOS“ System mit einem flächendeckenden Gitter von Referenzstationen über das Bundesgebiet [apos].

Ein weiteres Differentielles GPS ist das in Kapitel 4.1 - Voruntersuchungen des Testgebiet für das Experiment – untersuchte SBAS/EGNOS System. Beim

SBAS/EGNOS werden die Korrekturdaten von Satelliten ausgestrahlt (Tab. 4.1). Beim Europäischen EGNOS sind eine Reihe von Referenzstationen (Ranging and Integrity Monitor Stations - RIMS) über den Kontinent verteilt, deren Korrekturdaten zentral prozessiert und fusioniert werden.

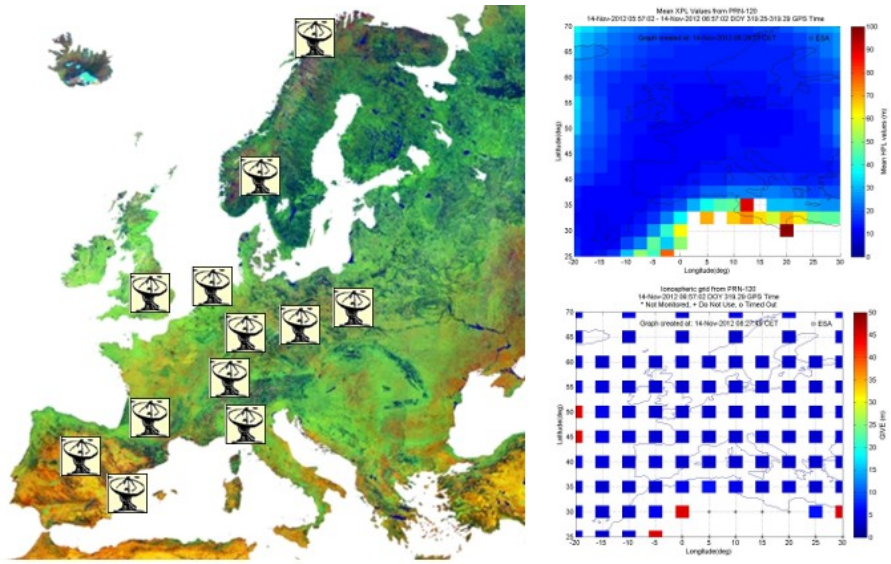


Abbildung 6.1: EGNOS RIMS Stationen mit erzeugten horizontalen Positionierungsfehler und vertikale Ionospheren Fehler Bildquelle: [egons rims]

Die über das EGNOS ausgestrahlten Korrekturdaten beinhalten Informationen über Fehler der Satellitenbahnen und Zeiten, sowie Informationen über die Integrität und Korrekturdaten für Ionosphärische Störungen (Abb 6.1). Das GPS Modul der Mikro-Drohne arbeitet beim Empfang von SBAS/EGNOS Korrekturdaten mit diesen. Trotz DGPS Korrektur sind die gemessenen Daten mit Fehlern behaftet, die man mittels statistischen Methoden abschätzten kann. Die Methoden der Fehlererkennung und Eliminierung werden im Kapitel 6.2 Qualitätssicherung behandelt.

6.2. Qualitätssicherung und Statistik

Aus Satellitenmessung abgeleitete Koordinaten gewinnen ihre vollständige Aussagekraft erst dann, wenn man weiß, mit welchen Abweichungen zwischen den, mit der Messung gewonnenen Koordinaten, und den wahren Koordinaten gerechnet werden muss. [Bauer 2003: S. 145]

Unter der Annahme, dass bei der Gewinnung von GPS Koordinaten kein systematischer Fehler vorliegt, kann man davon ausgehen, dass die gewonnen GPS Koordinaten einer Messkampagne, einer statistischen Normalverteilung unterliegen. Eine Normalverteilung, oder auch Gauß-Verteilung benannt, liegt vor, wenn aus einer Anzahl von wiederholten Messwerten der Einfluss eines einzelnen Messwertes die Summe der Messungen nicht bedeutend verändert. Konkret liegen in einer normalverteilten Messreihe anteilsmäßig wenig Werte mit einem großen zufälligen Fehler, und anteilsmäßig viele Werte mit geringen zufälligen Fehlern vor. Daraus resultiert eine signifikante Konzentration der Messwerte um einen Punkt, der den wahren Wert darstellt. Die Verteilung der Messwerte wird in einer Gaußschen Glockenkurve, der Wahrscheinlichkeitsdichte, beschreiben.

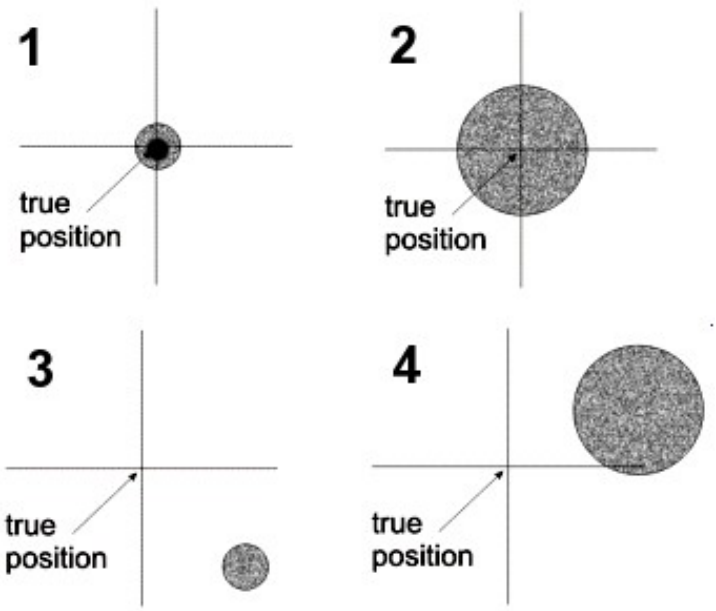


Abbildung 6.2: Zusammenhang von Genauigkeit und Präzision einer GPS Messung Bildquelle: [harre navgen]

Um den Gesamtfehler einer GPS Messung zu charakterisieren unterscheidet man zwischen Genauigkeit, Präzision und Systematischem Fehler. Der Systematische Fehler, dessen Fachbegriff „bias“ ist, ist ein Messfehler, welcher durch falsche Kalibrierung oder falsch arbeitender Messeinrichtung entsteht. Dieser kann mit statistischen Mitteln

nicht aufgehoben werden. Die Präzision (precision) beschreibt den zufälligen Messfehler und die Genauigkeit (accuracy) den Gesamtfehler. In Abbildung 6.2 ist der Zusammenhang von Genauigkeit und Präzision dargestellt.

- **1**: hohe Genauigkeit und hohe Präzision
- **2**: hohe Genauigkeit und wenig Präzision
- **3**: wenig Genauigkeit und hohe Präzision
- **4**: wenig Genauigkeit und wenig Präzision

Bei der statistischen Wahrscheinlichkeitsverteilung existieren mehrere Größen um Fehler zu beschreiben. Der „Mögliche Fehler“ (Probable Error – PE) definiert den Intervall, indem $50\,\%$ der Messwerte um den Median liegen. Die Standardabweichung (Standard Deviation – SD) beschreibt die Streuung der Messwerte um ihren arithmetischen Mittelwert und ist die Quadratwurzel der Varianz.

Die Standardabweichung berechnet sich nach $SD = \sqrt{\sigma^2}$ *und* $\sigma = \frac{\sum_{i=1}^{n} (x_i - \overline{X})^2}{n}$

wobei der arithmetische Mittelwert einer Messreihe $\overline{X} = \frac{\sum_{i=1}^{n} x_i}{n}$ ist.

Genauigkeitsmaße	Statistische Wahrscheinlichkeit
Mögliche Fehler (PE)	50%
Standardabweichung (SD)	68%
Doppelte Standardabweichung (2SD)	95%

Tabelle 6.1: Zusammenhang Eindimensionale Statistik und Wahrscheinlichkeit [Bauer 2003: S. 147]

Weitere statistische Genauigkeitsmaße sind die doppelte Standardabweichung 2σ

und die dreifache Standardabweichung 3σ, sowie PE_{95}, der analog zu den 50% „Mögliche Fehlern“ 95% der Messwerte im Intervall um den Median beschreibt. Der Zusammenhang zwischen den „Genauigkeitsmaßen“ und der Wahrscheinlichkeiten ist in Tabelle 6.1 gegeben.

Um mit räumlichen Daten statistisch arbeiten zu können, müssen die beschriebenen statistischen Modelle auf zwei Dimensionen, beziehungsweise drei Dimensionen erweitert werden. Analog zur eindimensionalen Statistik kann man mit den mehrdimensionalen „Genauigkeitsmaßen“ räumliche Messfehler beschreiben. Bei der zweidimensionalen Genauigkeit ergibt sich das Problem, dass es zwei Standardabweichungen gibt, in die X Achse und in die Y Achse. Daraus resultiert eine Fehlerellipse, da die Standardabweichungen der beiden Achsen voneinander immer unterschiedlich sind. Die Standardabweichungsellipse wird daher von einem Satz von Daten beschrieben, den Halbachsen der Ellipse, die Varianzen der X und Y Richtung, sowie die Ausrichtung der Ellipse.

$$SDE_x = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \overline{X})^2}{n}}$$

$$SDE_y = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \overline{Y})^2}{n}}$$

$$\tan(\theta) = \frac{A+B}{C}$$

$$A = \left(\sum_{i=1}^{n}\tilde{x}_i^2 - \sum_{i=1}^{n}\tilde{y}_i^2\right)$$

$$B = \sqrt{\left(\sum_{i=1}^{n}\tilde{x}_i^2 - \sum_{i=1}^{n}\tilde{y}_i^2\right)^2 + 4\left(\sum_{i=1}^{n}\tilde{x}_i^2\tilde{y}_i^2\right)^2}$$

$$C = 2\sum_{i=1}^{n}\tilde{x}_i^2\tilde{y}_i^2$$

$$\sigma_x = \sqrt{\frac{\left(\sum_{i=1}^{n}(\tilde{x}_i\cos\theta - \tilde{y}_i\sin\theta)\right)^2}{n}}$$

$$\sigma_y = \sqrt{\frac{\left(\sum_{i=1}^{n}(\tilde{x}_i\sin\theta - \tilde{y}_i\cos\theta)\right)^2}{n}}$$

Standardabweichungsellipse und Standardabweichungen der X Achse und Y Achse [arcgis spatstat]

Da die Beschreibung der Ellipse nicht praktikabel kann man die zweidimensionalen

Genauigkeitsmaße als approximierten Kreise beschreiben.

$$SD = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \overline{X})^2}{n} + \frac{\sum_{i=1}^{n}(y_i - \overline{Y})^2}{n}} \qquad \overline{X} = \frac{\sum_{i=1}^{n} x_i}{n}$$

$$dRMS = SD = \sqrt{A^2 + B^2} \qquad \overline{Y} = \frac{\sum_{i=1}^{n} y_i}{n}$$

$$\sigma_{2d} = \sqrt{\sigma_x^2 + \sigma_y^2}$$

Standardabweichungskreis und arithmetischer Mittelpunkt [arcgis spatstat]

Analog zu den eindimensionalen Genauigkeitsmaßen stehen die zweidimensionalen Genauigkeitsmaße im Zusammenhang, der in Tabelle 6.2 beschrieben wird.

Zweidimensionale Genauigkeitsmaße	Statistische Wahrscheinlichkeit
Kreisfehlerwahrscheinlichkeit (CEP)	50%
Standardabweichung (SD - 1dRMS)	63.2% – 68.3%
Doppelte Standardabweichung (2dRMS)	95.4% - 98.2%

Tabelle 6.2: Zusammenhang zweidimensionale Statistik und Wahrscheinlichkeit [Bauer 2003: S. 149]

Mit Hilfe dieser statistischen Werkzeuge kann man die Genauigkeit und Präzision der gemessenen GPS Punkte als Prozentanteil der Punkte, die sich in einem Kreis mit bestimmten Radius befinden, ausdrücken.

6.3. Experiment

Das im Abschluss dieser Arbeit durchgeführte Experiment diente zur Überprüfung des entwickelten GIS Workflow und zur empirischen Untersuchung der Genauigkeit und Präzision des Wegpunkt Fluges der Mikro-Drohne. Dabei steuert die Mikro-Drohne einen genau vermessener Fixpunkt an und schwebt nach Erreichen dieses Fixpunktes sechzig Sekunden über diesem Fixpunkt. Im Sekundenintervall wird die aktuell gemessene GPS Koordinate für die nachträgliche Auswertung des Experimentes

aufgezeichnet. Anhand der zusätzlich aufgezeichneten Attribute (Tab. 4.12) kann bestimmt werden, welche aufgezeichneten GPS Koordinaten zum Schwebeflug über dem Fixpunkt gehören und für die Auswertung herangezogen werden.

6.3.1. Vorbereitung und Durchführung des Experimentes

Als vorbereitende Maßnahme wurde mit dem „ArcGIS Catalog“ Werkzeug ein Verzeichnis mit einer File-Geodatenbank erstellt und die vorbereiteten Geodaten, die Orthofotos des Experimentiergebietes und die Grundwassersonden in dieser Geodatenbank zusammengeführt. Die Rasterdaten wurden mit dem Conversion Tool „Raster to Geodatabase“ importiert. Das in Kapitel 4 erstellte Shapefile mit der Featurclass der Sondenstandorte wurde in die Geodatenbank exportiert. Weitere Verzeichnisse für das optionale Wegpunkt File und für die zu erzeugenden Shapefiles der Ergebnisse wurden auch darin erstellt. Der Umweg über Shapefiles wurde gewählt, da ein direkter Export des GPX Files des Mikro-Drohnen Einsatzes mit den DNRGPS Programm nicht funktionierte. Die Shapefiles wurden später mit der richtigen Projektion in die Geodatenbank importiert. Abschließend zu den Vorbereitungen wurde in der Geodatenbank ein Wegpunkt Template als Point Featureclass mit dem „MKCreateWayPointTemplateFeature“ Modelbuilder Script aus Kapitel 5.1 Abbildung 5.1 erzeugt. Im Falle dieses Experimentes mit der Projektion „EPSG 31255“. Die Digitalisierung der Wegpunkte erfolgte in der gewählten Projektion. Im ArcGIS wurden die für das Experiment zusammengestellte Geodaten mit der einheitlichen Projektion „EPSG 31255“ geladen. Es wurden drei Wegpunkte digitalisiert, eine Startposition, der Standort der „Sonde 8“ und eine Landeposition.

Name	Latitude	Longitude	Fangradius	Anflug-geschwindigkeit	Aufenthaltsdauer
Start	48.3294174	14.2916972	2 Meter	10 dm/sek	10 Sekunden
Sonde 8	48.3296074	14.2917578	2 Meter	10 dm/sek	60 Sekunden
Landung	48.3294046	14.2915991	2 Meter	10 dm/sek	10 Sekunden

Tabelle 6.3: Attribute der Wegpunkte des Experimentes

Der Sondenstandort wurde mit „Point Snapping“ Funktion digitalisiert, damit der Punkt deckungsgleich mit dem Sondenstandort ist. Nach dem Digitalisieren wurden die drei Punkte attributiert. Die wichtigen Parameter der drei Wegpunkte sind in Tabelle 6.3 angeführt. Die Featureklasse mit den Wegpunkten wurde testweise an die Mikro-Drohne mit der seriellen Funkverbindung übertragen (Abb. 6.3) und als optionales Wegpunktfile gespeichert (Tab.9.3). Das gespeicherte Wegpunktfile wurde anschließend im Feldversuch für das Experiment verwendet.

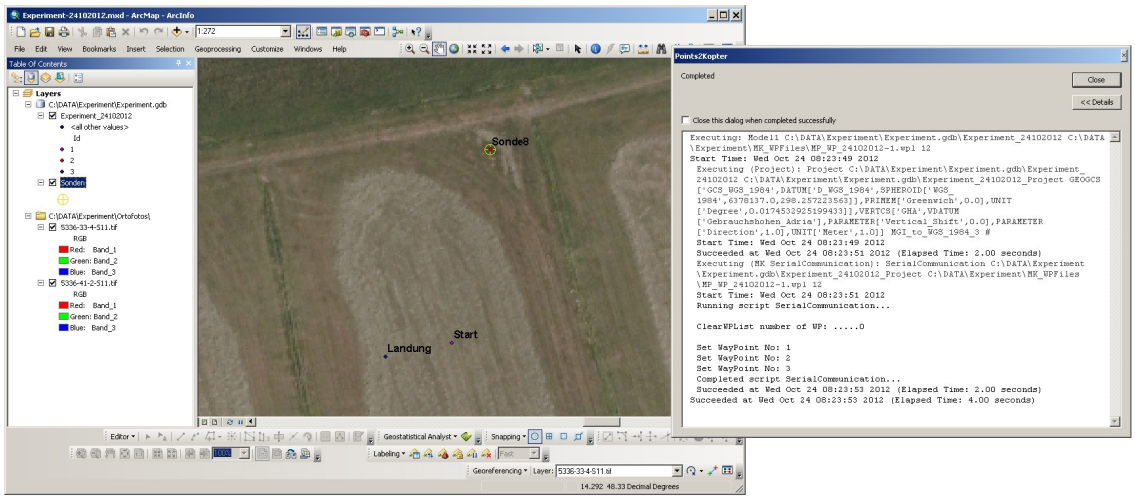


Abbildung 6.3: Serielle Übertragung der Wegpunkte

Vor der Durchführung des Experimentes wurde sichergestellt, das die Speicherkarte in die Mikro-Drohne eingelegt wurde. Es wurden drei Flüge mit den digitalisierten Wegpunkten absolviert, wobei beim letzten Flug die Aufenthaltsdauer über dem Sondenstandort auf 120 Sekunden gesetzt wurde. Nach Durchführung des Experimentes wurden die aufgezeichneten Telemetrie Daten als „CSV“ File gespeichert, sowie die erzeugten Daten auf der Speicherkarte der Mikro-Drohne gesichert.

6.3.2. Auswertung der GPS Daten des Experimentes

Die Auswertung der gewonnen GPS Daten auf der Speicherkarte der Mikro-Drohne erfolgt in ArcGIS. Um die Daten auswerten zu können, wurden sie zuerst mit dem DNRGPS Programm in ein ShapeFile als Point Feature Klasse konvertiert. Das DNRGPS Programm bietet die Funktion zur Berechnung der CEP (Circular Error of

Probabilities) an. Dabei werden die Radien von Kreisen vom arithmetischen Mittelpunktes aus berechnet, in dem ein definierter Prozentsatz von Messpunkten liegt. Um die Funktion zu nutzen, wurden nur die Messpunkte des Schwebefluges über dem Fixpunkt, für die Berechnung der CEP Werte herangezogen. Dazu betrachtet man die Attribute der Felder „NCFlags“ und „WP“ des im DNRGPS geladenen GPX File (Tab 4.12). Der Attributwert des „NCFlag“ kodiert den aktuellen Status des Navigations Controller in den Bit Werten des enthaltenen Bytes. Der dritte Bitwert als „1“ oder „TRUE“, als Hexadezimalwert 0x04, kodiert, dass sich die Mikro-Drohne im „Waypoint“ Modus befindet. Der sechste Bitwert, als Hexadezimalwert 0x20, kodiert das der Wegpunkt im Fangradius erreicht ist. Der achte Bitwert, als Hexadezimalwert 0x80, kodiert einen GPS Satelliten-fix des Navigations Controllers. Die logische Bitaddition dieser drei Werte ergibt einen Hexadezimalwert von 0xA4 (binär: 10100100). Der Wert im „WP“ Feld enthält den aktuell anzufliegenden Wegpunktnamen, der bei der Digitalisierung gesetzt wurde. Wenn somit der Wegpunktname im „WP“ Feld stimmt und das sechste Bit im „NCFlags“ Feld vom Navigations Controller gesetzt wurde, ist der Wegpunkt erreicht. Die Auswahl der GPS Messpunkte des Schwebfluges über dem Fixpunkt, der Sonde 8, kann manuell im DNRGPS, oder mit der „Select by Attribut“ Funktion in ArcGIS erfolgen. Das entsprechende SQL Statement lautet:

```
SELECT * FROM  <PointFeatureClass> WHERE "WP" = 'S2 ,2,3,0' AND "NCFlag" = '0xA4'
```

Flugnummer	CEP 50%	CEP 90%	CEP 95%	CEP 98%
Flug 1	1.22	1.97	2.11	2.27
Flug 2	0.72	1.84	2.46	2.77
Flug 3	1.85	6.43	8.15	10.19

Tabelle 6.4: Circular Error of Probabilities ermittelt mit DNRGPS

Mit den selektierten Messpunkten der Schwebeflüge über dem Fixpunkt wurden im DNRGPS vorab die CEP Werte berechnet (Tab. 6.4). Die selektierten Messpunkte des Schwebfluges über der Sonde lagen im WGS84 – EPSG 4326 Geografischen

Koordinatensystem vor. Die Auswertung der gewonnen Daten wurde in der selben Projektion wie die Digitalisierung der Wegpunkte getätigt. Daher wurden die Shapefiles der selektierten Messpunkte nach EPSG 31255 transformiert. Für den Datumswechsel von WGS84 nach MGI wurden die Transformationsparameter aus Tabelle 4.4 verwendet. Als vertikales Koordinatensystem wurde EPSG 5778 - GHA verwendet. Mit dem ArcGIS Tool Datamanagement → Projections and Transformations → Feature → Project wurden die Featureklassen transformiert und projiziert und die resultierenden Featureklassen in der vor dem Experiment erstellten Geodatenbank gespeichert. Somit lagen die Daten für die Auswertung einheitlich vor. Folgende Geoprozessierungen wurden auf die selektierten Point Features angewendet:

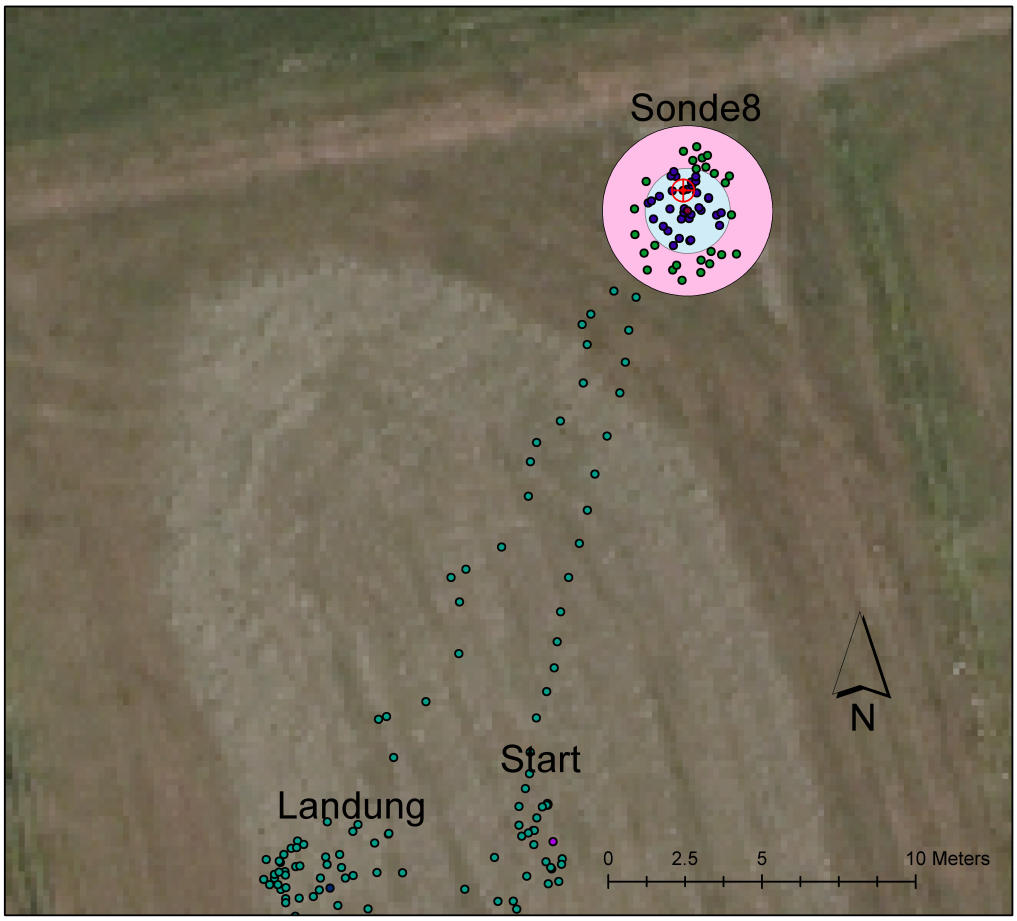
- Standardabweichung mit dem „Standard Distance Tool“ mit einfacher und doppelter Standardabweichung. Resultat sind zwei Kreispolygone, die die Präzision der Messreihe beschreibt
- Clip der selektierten Point Features mit dem einfachen und doppelten Standardabweichungs Polygon. Resultate sind zwei Featureklassen mit den Punkten, die in der einfachen und doppelten Standardabweichung liegen. Das Verhältnis der Punkte innerhalb der Standardabweichung und der selektierten Punkte beschreibt ebenfalls die Präzision der Messung.
- Arithmetischer Mittelwert der selektierten Point Features. Resultat ist der arithmetische Mittelpunkt, der für die nächste Prozessierung benötigt wird.
- Point Distanz aus der Analysis Toolbox zwischen:
    - Point der Sonde 8 (Fixpunkt) und dem arithmetischen Mittelpunkt. Resultat ist eine Tabelle mit dem Abstand zwischen Fixpunkt und Mittelwert, der die Genauigkeit der Messreihe beschreibt.
    - Point der Sonde 8 zu den selektierten Punkten einer Messreihe. Resultat ist eine Tabelle mit den Abständen der Punkte der Messreihe zur Sonde 8 (Fixpunkt) und das Feld „NEAR_FID“ aufsteigend sortiert, beschreibt die Genauigkeit über die Zeit.

- Arithmetischer Mittelpunkt zu den selektierten Punkten einer Messreihe. Resultat ist eine Tabelle mit den Abständen der Messreihe zum arithmetischen Mittelpunkt und das Feld „NEAR_FID“ aufsteigend sortiert, beschreibt die Präzision über die Zeit.

	Flug 1	Flug 2	Flug 3
Anzahl der Messpunkten	59	59	119
Arithmetischer Mittelpunkt Datum: WGS84 Longitude: Latidude:	 14.291760 48.329601	 14.291757 48.329606	 14.291756 48.329603
Arithmetischer Mittelpunkt Datum: EPSG 31255 X Achse Y Achse	 71125.16 355040.50	 71124.99 355040.97	 71124.87 355040.66
Standardabweichung	1.38 m	1.15 m	3.72 m
Anteil Messpunkte Standardabweichung Absolut: Prozentual:	 33 55.93%	 44 74.58%	 91 76.47%
Standardellipse X Achse: Y Achse: Rotation:	 1.16 m 1.61 m 10.1°	 0.78 m 1.43 m 29.1°	 4.06 m 3.35 m 108.5°
Anteil Messpunkte Standardellipse Absolut: Prozentual:	 32 54.24%	 40 67.80%	 90 75.63%
Distanz Fixpunkt zu arithmetischen Mittelpunkt	0.68 m	0.17 m	0.52 m
Interpretation Genauigkeit (accuracy)	hoch	sehr hoch	hoch
Interpretation Präzision (precision)	hoch	hoch	nieder

Tabelle 6.5: Ergebnisse des Experiment

Die Ergebnisse der drei Flüge des Experimentes sind in Tabelle 6.4 und Tabelle 6.5 numerisch zusammengefasst und interpretiert. Eine grafische Aufbereitung der drei Flüge mit den Zeitreihen der Punktdistanzen sind in Abbildung 6.4 bis Abbildung 6.6 ersichtlich.



	Aufgezeichnete Punkte	Prozentualer Anteil	Wert in Meter
Flug 1	59	100%	
CEP 50	29	50%	1.22 m
CEP 95	56	95%	2.11 m
Standardabweichung	33	56%	1.38 m
Doppelte Standardabweichung	59	100%	2.76 m
Genauigkeit - Abstand Sonde zu Mittelwert			0.68 m

Sonde 8
Kopter SD
Kopter 2SD
Standardabweichung
Doppelte Standardabweichung

Datum: 24. Okt. 2012
GPS Startzeit: 10:13:09 GMT+2
GPS Endzeit: 10:14:08 GMT+2
Autor: Andreas Feichtner U1464

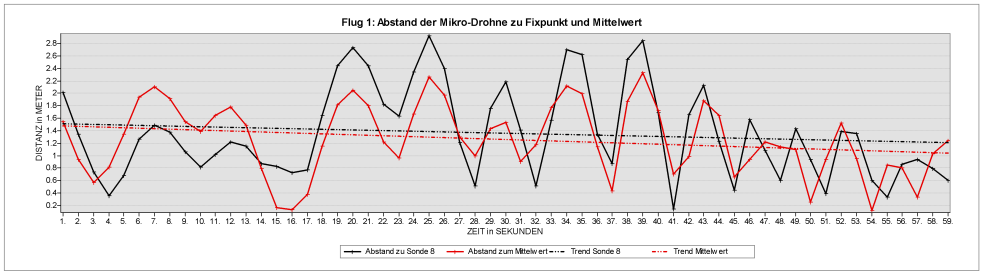
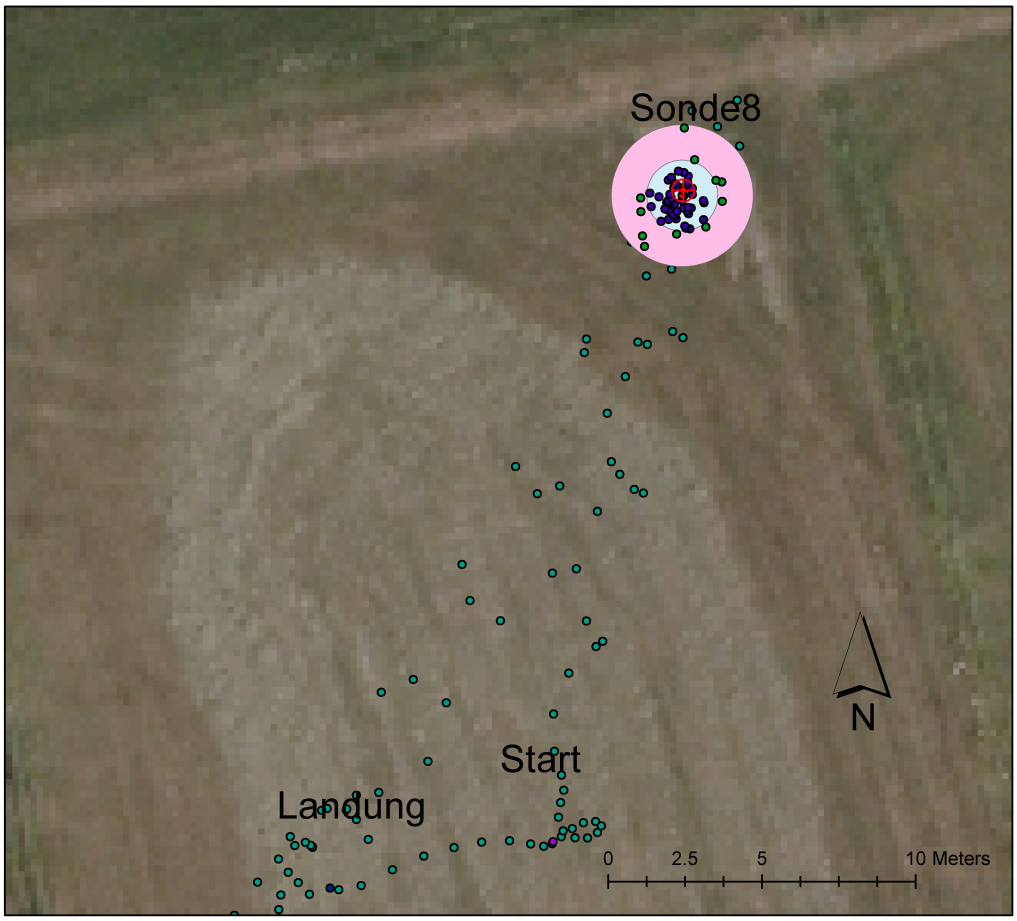


Abbildung 6.4: Auswertung Flug 1



	Aufgezeichnete Punkte	Prozentualer Anteil	Wert in Meter
Flug 2	59	100%	
CEP 50	29	50%	0.72 m
CEP 95	56	95%	2.46 m
Standardabweichung	44	75%	1.15 m
Doppelte Standardabweichung	59	93%	2.30 m
Genauigkeit - Abstand Sonde zu Mittelwert			0.17 m

Sonde 8
Kopter SD
Kopter 2SD
Standardabweichung
Doppelte Standardabweichung

Datum: 24. Okt. 2012
GPS Startzeit: 10:21:35 GMT+2
GPS Endzeit: 10:22:34 GMT+2
Autor: Andreas Feichtner U1464

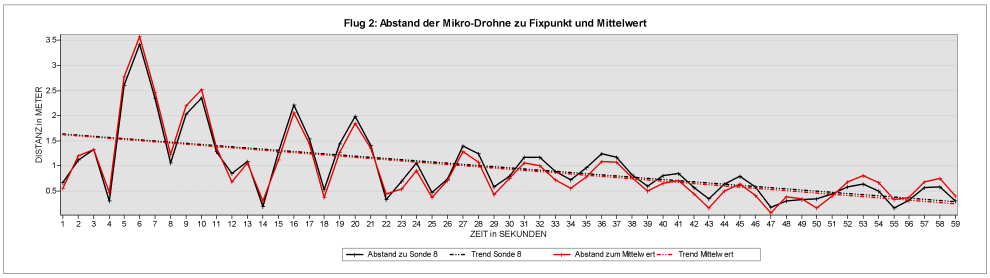
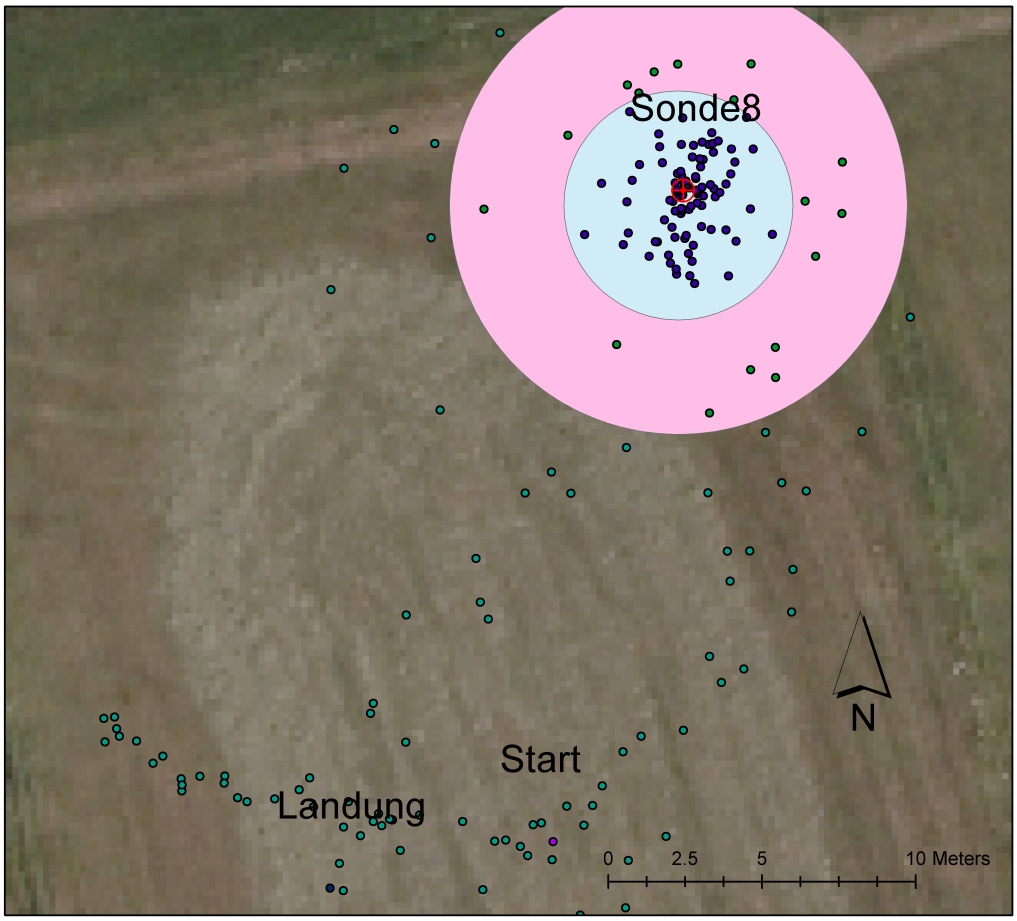


Abbildung 6.5: Auswertung Flug 2



	Aufgezeichnete Punkte	Prozentualer Anteil	Wert in Meter
Flug 3	119	100%	
CEP 50	59	50%	1.85 m
CEP 95	113	95%	8.15 m
Standardabweichung	33	76%	3.72 m
Doppelte Standardabweichung	59	91%	7.44 m
Genauigkeit - Abstand Sonde zu Mittelwert			0.52 m

Sonde 8
Kopter SD
Kopter 2SD
Standardabweichung
Doppelte Standardabweichung

Datum: 24. Okt. 2012
GPS Startzeit: 10:27:03 GMT+2
GPS Endzeit: 10:29:04 GMT+2
Autor: Andreas Feichtner U1464

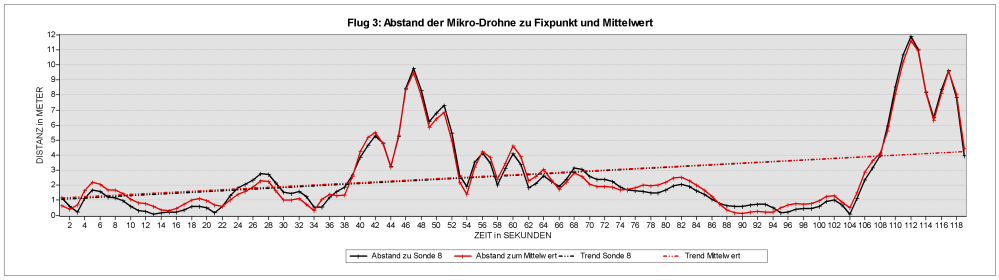


Abbildung 6.6: Auswertung Flug 3

7. Diskussion

7.1. Wirtschaftlichkeit

Wenn man die Anschaffungskosten und die laufenden Betriebskosten einer Mikro-Drohne mit bemannten Fluggeräten vergleicht, sind die anfallenden Einsatzkosten einer Mikro-Drohne im Vergleich zueinander minimal. Die räumliche Genauigkeit, die mit Mikro-Drohnen erreicht wird, öffnet viele Einsatzgebiete. Für Luftbildaufnahmen sind Mikro-Drohnen mittlerweile ein wachsendes Geschäftsfeld, dessen Kunden von Privatpersonen, Unternehmen bis zu Behörden und wissenschaftlichen Institutionen reicht. Hierbei werden sich besonders bei den Luftbildaufnahmen neue Geschäftsfelder auftun, die aus dem flexiblen und kostengünstigen Einsatz von Mikro-Drohnen erwachsen. Möglich sind neben dem klassischen Feld der Luftbildaufnahmen, der Einsatz von Mikro-Drohnen für Überwachungsaufgaben von Industrieanlagen, Energiegewinnungsanlagen, wie Windräder und Photovoltaik, oder Sendeanlagen. Auch Überwachungsaufgaben im öffentlichen Interesse, wie Großveranstaltungen oder Verfügbarkeit von Verkehrsverbindungen sind kostengünstig und einfach zu tätigen. Ein möglicher Einsatz von Mikro-Drohnen bei Katastrophen für rasche Informationsgewinnung kann monetär nicht bewertet werden, da die Risiken für Einsatzkräfte oft vorab nicht eingeschätzt werden können. Weitere noch nicht absehbare Einsatzfelder, wie zum Beispiel experimentelle Überprüfung von Modellen der akustischen Raumplanung würden unkalkulierbare Planungsfehler aufdecken helfen. Zusammengefasst kann man die Wirtschaftlichkeit von Mikro-Drohnen positiv bewerten, da der Einsatz von Mikro-Drohnen die Wartungsaufgaben diverser Großanlagen rationalisieren kann, wo heute noch meist Spezialkräfte angefordert werden müssen. Auch die Möglichkeiten neuer Geschäftsfelder durch Mikro-Drohnen Einsätze könnte neue wirtschaftliche Impulse von noch nicht absehbaren Chancen bieten.

7.2. Rechtliche Aspekte

Die Beleuchtung der rechtlichen Lage für einen Mikro-Drohnen Einsatz bezieht sich hier nur auf das Österreichische Luftrecht und die Österreichische Lufthoheitszone. In anderen Ländern gilt das jeweilige Staatliche Recht. Alle hier zitierten Paragraphen beziehen sich ebenfalls auf das Österreichische Luftrecht [luftrecht at].

Das Österreichische Luftfahrtrecht besteht aus dem Luftfahrtgesetz und diversen Verordnungen und Erlassen von zuständigen Behörden. Da der Autor keine fundierte rechtliche Ausbildung besitzt, werden die rechtlichen Aspekte eines Mikro-Drohnen Einsatzes hier nur diskutiert.

Im Österreichischen Luftfahrtrecht sind Mikro-Drohnen im Sinne eines Flugmodells beziehungsweise als „Unmanned Aerial Vehicle“ - UAV nicht definiert. Nach Meinung des Autors fällt eine Mikro-Drohne im Österreichischen Luftfahrtrecht in die Klasse der Fluggeräte, die im Österreichischen Fluggesetz im §22 Absatz 1 Punkt 2 klassifiziert sind beziehungsweise explizit als Flugmodell im §5 Punkt 5 der Zivilluftfahrzeug- und Luftfahrtgerät-Verordnung 2010 - ZLLV 2010. Ein Flugmodell ist *ein Gerät, das selbstständig im Flug verwendet werden kann* (selbstständig fliegen kann) *ohne Luftfahrzeug lt. §11 Luftfahrtgesetz zu sein.* Die Annahme, dass eine Mikro-Drohne ein Flugmodell ist, geht aus dem Startgewicht des Fluggerätes hervor, das für Flugmodelle die *Freiheit des Luftraumes* nach §2 Absatz 1 Luftfahrtgesetz gewährt. Die Benutzung von Fluggeräten unter 25 Kilogramm Startgewicht außerhalb von Sicherheitszonen ist laut §129 Absatz 1 Luftfahrtgesetz frei. Die Sicherheitszonen und die Zuständigkeit für eine Bewilligung zum Flug innerhalb von Sicherheitszonen ist auch in §129 Absatz 2 geregelt. Aktuell zum Zeitpunkt der Erstellung dieser Arbeit ist aber der Betrieb von Mikro-Drohnen mit Befrachtung, wie zum Beispiel einer Kamera für Luftbildaufnahmen oder einem Sensorensystem, ohne Lizenz verboten. Dies geht aus einer Parlamentarischen Anfrage an die Frau Bundesministerin für Verkehr, Innovation und Technologie vom 21. April 2011 hervor.

„Nach derzeitiger Rechtslage sind unbemannte (Modelle) Luftfahrzeuge mit Kamera gemäß § 11 Abs. 1 Luftfahrtgesetz als Luftfahrzeuge zu qualifizieren - mit allen daraus

folgenden Konsequenzen (erforderliche Lufttüchtigkeitszertifizierungen, Registrierungspflicht, Pilotenschein für den Steuerer, Einhaltung der Luftverkehrsregeln etc.). Da diese Voraussetzungen jedoch von keinem dieser Flugobjekte erfüllt werden (können), dürfen diese nach geltender Rechtslage nicht betrieben werden." [Bures 2011: S. 1] Der §11 Absatz 1 beschreibt Luftfahrzeuge als Fahrzeuge, *die sich zur Fortbewegung von Personen oder Sachen in der Luft ohne mechanische Verbindung mit der Erde eignen, gleichgültig, ob sie schwerer als Luft (zum Beispiel Flugzeuge, Segelflugzeuge, Hänge- und Paragleiter, Schwingenflugzeuge, Hubschrauber, Tragschrauber und Fallschirme) oder leichter als Luft (zum Beispiel Luftschiffe und Freiballone).* Somit ist der lizenzfreie Betrieb von Flugmodellen beziehungsweise Mikro-Drohnen in Österreich nur ohne Fracht, nach Meinung des Bundesministeriums erlaubt, da es sich bei der Befrachtung von einem Fluggerät in ein Luftfahrzeug im Sinne des §11 Absatz 1 Luftfahrtgesetz verwandelt, und somit alle Pflichten und Rechte eines Luftfahrzeuges besitzt. Eine Regelung für den Betrieb von Mikro-Drohnen sowie von Unmanned Aerial Vehicle" - UAV soll in der nächsten Novellierung des Luftfahrtrechtes kommen.

Fazit: Der Betrieb einer Mikro-Drohne, ohne Fracht, ist im Sinne eines Flugmodells frei. Der Betrieb einer Mikro-Drohne mit Fracht, wie einer Kamera oder einem Sensorensystem, ist nur als Luftfahrzeug im Sinne des §11 Luftfahrtgesetz möglich.

8. Zusammenfassung und Ausblicke

In diesem abschließenden Kapitel werden die gestellten Fragen dieser Arbeit zusammengefasst und beantwortet, sowie Ausblicke auf mögliche Weiterentwicklungen der gewonnenen Ergebnisse aufgezeigt.

8.1. Zusammenfassung der Ergebnisse

Wie kann man einen Mikro-Drohneneinsatz abwickeln und sinnvoll dokumentieren ?

Mit der vorliegenden Arbeit wurde gezeigt, wie man einen Mikro-Drohneneinsatz unter Zuhilfenahme eines Geoinformationssystems planen, durchführen und dokumentieren kann. Es wurde beispielhaft in einem GIS Workflow gezeigt, wie man mit dessen Hilfe einen Mikro-Drohnen Einsatz, unter Abarbeitung von sequentiell aufeinanderfolgenden Schritten, abwickeln kann. Die gewonnenen räumlichen Koordinaten der Mikro-Drohnen Positionen während des Einsatzes wurden verwendet um den Einsatz zu validieren und zu dokumentieren, um damit eine Aussage über die räumliche Genauigkeit und Präzision des Mikro-Drohneneinsatzes zu gewinnen.

Was für Vorteile bringt die Einsatzplanung für Mikro-Drohnen unter Zuhilfenahme eines Geoinformationssystems mit aktuellen Geodaten?

Durch das Verwenden eines Geoinformationssystems für die Abwicklung eines Mikro-Drohneneinsatzes wird ein konsistenter Datenfluss während der Durchführung gewährleistet. Dieser beginnt bei der Auswahl der Geodaten als Grundlage für die Einsatzplanung. Mit einem Geoinformationssystem kann man auf validierte Geodaten unterschiedlicher Quellen zurückgreifen. Aufgrund dieser Geodaten als Basis kann man die Wegpunktplanung mit einer einheitlichen Projektion und Datum digitalisieren und attributieren und damit die geografische Richtigkeit der Wegpunkte gewährleisten. Da die gewonnenen geografischen Koordinaten des Mikro-Drohneneinsatzes, auf Grundlage der Planung im Geoinformationssystem in Beziehung zur Planung analysiert und validiert werden, ist eine gesamtheitliche Aussage des Mikro-Drohneneinsatzes möglich. Die Archivierung und Dokumention der gewonnenen Daten geschieht

gemeinsam mit den Basisdaten der Einsatzplanung und gewährleistet eine hohe Qualität bei einem späteren Rückgriff auf die Einsatzdaten.

Wie kann man gewonnene Messergebnisse eines Drohneneinsatzes mit der aufgezeichneten Flugroute nachträglich verschneiden, um der thematischen Aussage ergänzend zur Verfügung zu stellen ?

Im Geoinformationssystem liegen die gemeinsamen Daten der Einsatzplanung, der Ergebnisse und die GPS Koordinaten des Drohneneinsatzes, vor. Mit den Mitteln und Funktionen der statistischen Werkzeuge des Geoinformationssystems ist es möglich die gewonnen Daten in Beziehung zu den geplanten Daten zu stellen, um eine Aussage über die Genauigkeit und die Präzision des Einsatzes zu machen. Weiters kann man mit den Werkzeugen des Geoinformationssystems den Einsatz thematisch aufbereiten um die Ergebnisse einem breiteren Publikum verständlich zu präsentieren.

Welche geografische Genauigkeit wird beim Einsatz der Mikro-Drohne erreicht und wie groß sind die Abweichungen der geplanten Flugroute von der tatsächlich geflogenen Route ?

Anhand des Experimentes wurde gemessen, dass die Abweichungen zwischen dem geplanten anzufliegenden Punkt und der tatsächlichen räumlichen Lage der Mikro-Drohne im einstelligen Meterbereich liegt. Die Genauigkeit und Präzision des Mikro-Drohneneinsatzes hängt von vielen Parametern ab, auf die schwer Einfluss genommen werden kann. Das betrifft einerseits die Qualität der GPS Daten der Mikro-Drohne, wie die Satellitenkonstellation des Navstar Systemes, die Möglichkeit der Nutzung der SBAS/DGPS Korrekturdaten, mögliche Abschattungen und Reflektionen von GPS Satellitensignalen anhand der topografischen Situation des Einsatzgebietes. Andererseits ist die Genauigkeit und Präzision abhängig von den Steuerparametern der Mikro-Drohne und wie agil diese auf externe Einflüsse, wie Windstöße reagiert. Hier besteht noch Potential für Erreichung von höherer Genauigkeit und Präzision der räumlichen Lage der Mikro-Drohne, durch empirische Annäherung an optimale Steuerungsparameter Kombinationen.

Welche Vor- und Nachteile bringt die Verwendung einer Mikro-Drohne im Vergleich zu professionellen UAV Systemen ?

Der primäre Vorteil einer Mikro-Drohne im Vergleich zu einem professionellen UAV System besteht in den Anschaffungskosten mit einem geschätzten Kostenvorteil von 1:10 , wobei hier die Entwicklungszeit der Eigenbau Mikro-Drohne monetär nicht bewertet ist. Über einen Vergleich der Genauigkeit und Präzision zwischen den beiden Systemen kann keine Aussage vom Autor getätigt werden, wobei die erreichte Genauigkeit und Präzision der in dieser Arbeit verwendete Mikro-Drohne vom Autor als zufriedenstellend, und für geografisch-spezifische Einsätze als ausreichend bewertet wird. Ein weiterer Vorteil ist die Tatsache, dass durch die Eigenentwicklung ein Um- und Ausbau für spezifische Fragestellungen eines Mikro-Drohneneinsatzes gegeben ist. Beispielhaft sei hier der Einsatz für drohnenbasierte Umweltbeobachtung und Echtzeitkartierung erwähnt, der einen Aufbau spezifischer Sensorsystemen bedarf. Nachteilig im Vergleich zu einem professionellen schlüsselfertigen UAV System ist der Aufwand, um sich die Wissensbasis für den Bau und Betrieb einer Eigenbau Mikro-Drohne anzueignen.

8.2. Ausblicke und weitere Fortführung der Arbeit

Ein fortführender Aspekt dieser Arbeit wäre eine direkte Übernahme der GPS Koordinaten in Echtzeit in das Geoinformationssystem mittels der seriellen Funkverbindung.

Kommando Frame an Navigations Controller – Request NaviCtrl Struct			Antwort Frame von Navigations Controller – NaviCtrl Struct		
Adress-Byte	Kommando	Daten	Adress-Byte	Kommando	Daten
'c'	'o'	Byte d.Sende Intervall	'c'	'O'	NaviCtrl Strcut

Tabelle 8.1: Request NaviCtrl Struct Frame [mkser]

Das serielle Protokoll der Mikro-Drohne implementiert auch die Abfrage der „NaviCtrl

Struct“ Datenkapsel (Tab. 8.1). Der „NaviCtrl Struct“ beinhaltet die Daten, die im GPX File der Speicherkarte aufgezeichnet werden (Tab. 8.2). Dies würde eine automatisierte und direkte Aufzeichnung der GPS Koordinaten der Mikro-Drohne in Echtzeit ermöglichen.

```
typedef struct
{
 u8 Version;                          // version of the data structure
 GPS_Pos_t CurrentPosition;           // see ubx.h for details
 GPS_Pos_t TargetPosition;
 GPS_PosDev_t TargetPositionDeviation;
 GPS_Pos_t HomePosition;
 GPS_PosDev_t HomePositionDeviation;
 u8  WaypointIndex;                   // index of current waypoints running
                                      //from 0 to WaypointNumber-1
 u8  WaypointNumber;                  // number of stored waypoints
 u8  SatsInUse;                       // number of satellites used for position
                                      // solution
 s16 Altimeter;                       // hight according to air pressure
 s16 Variometer;                      // climb(+) and sink(-) rate
 u16 FlyingTime;                      // in seconds
 u8  UBat;                            // Battery Voltage in 0.1 Volts
 u16 GroundSpeed;                     // speed over ground in cm/s (2D)
 s16 Heading;                         // current flight direction in °
                                      // as angle to north
 s16 CompassHeading;                  // current compass value in °
 s8  AngleNick;                       // current Nick angle in 1°
 s8  AngleRoll;                       // current Rick angle in 1°
 u8  RC_Quality;                      // RC_Quality
 u8  FCStatusFlags;                   // Flags from FC
 u8  NCFlags;                         // Flags from NC
 u8  Errorcode;                       // 0 --> okay
 u8  OperatingRadius;                 // current operation radius around
                                      // the Home Position in m
 s16 TopSpeed;                        // velocity in vertical direction in cm/s
 u8  TargetHoldTime;                  // time in s to stay at the given target,
                                      // counts down to 0 if target has been reached
 u8  FCStatusFlags2;                  // StatusFlags2 (since version 5 added)
 s16 SetpointAltitude;                // setpoint for altitude
 u8  Gas;                             // for future use
 u16 Current;                         // actual current in 0.1A steps
 u16 UsedCapacity;                    // used capacity in mAh
} __attribute__((packed)) NaviData_t;
```

Tabelle 8.2: NaviCtrl C Struct [mksvn]

Anhand der Verfügbarkeit der aktuellen Mikro-Drohnen Koordinaten im Geoinformationssystem wären viele Möglichkeiten der Fortführung dieser Arbeit gegeben. Die Punkt-Koordinaten könnten direkt in eine entsprechende Feature Klasse gespeichert werden und stehen für eine Prozessierung im Geoinformationssystem zur Verfügung. Anhand der aktuellen Koordinaten und gegebenenfalls Sensormesswerte an dieser Koordinate kann die Mikro-Drohne automatisiert oder interaktiv in Echtzeit zu neuen Koordinaten für weitere Messreihen nachgeführt werden, um Hotspots zu

detektieren oder thematische Echtzeitkartierung zu ermöglichen und dabei stetig zu verfeinern. Als Basis könnte der ArcGIS Tracking Server dienen, der für das Sammeln und Senden von Echtzeitdaten konzipiert wurde.

8.3. Von 2D nach 3D

Bei dem in dieser Arbeit programmierten Python Scripts wird mit projizierten zweidimensionalen Punktkoordinaten gearbeitet. Diese sind somit kein echtes 3D. Die Flughöhe der Mikro-Drohne ist immer relativ, in Meter zur Abflughöhe und wird, von dem an der Mikro-Drohne verbauten Luftdrucksensor ermittelt. Wenn man echte dreidimensionalen Geodaten bei der Digitalisierung der Wegpunkte als Grundlage nimmt, und dreidimensional Wegpunkte digitalisiert, könnte die aktuelle Höhe der Wegpunkte, relativ zum Bodenlayer, ermittelt werden und das entsprechende Feld der Flughöhe attributiert werden. Auch eine automatisierte Validierung der digitalisierten Wegpunkte Sequenz ist anzudenken. Die Mikro-Drohne fliegt zwischen zwei Wegpunkten immer auf der kürzesten Strecke, der Geraden. Wenn man eine Linie zwischen zwei Wegpunkten mit einem Sicherheitsbuffer nimmt und dieses Linefeature mit einem aktuellen Digitalen Oberflächen Modell des Gebietes verschneidet, könnten vorab Kollisionen detektiert werden. Ein Algorithmus könnte auch soweit entwickelt werden, das diese Kollisionen automatisiert, durch Setzten zusätzliche dreidimensionaler Wegpunkte zwischen diesen zwei Wegpunkten eliminiert werden. Die Auswertung der gewonnenen Koordinaten der Mikro-Drohne könnte weiterhin mit projizierten zweidimensionalen Koordinaten erfolgen oder mit Mitteln der dreidimensionalen Statistik.

9. Anhang

```
import sys, arcpy, struct, time
from serial import Serial
from collections import deque

#--------------------------------------------------------------------
class WayPoint(deque):
    def __init__(self, lat, lon, rad, alt, climb, delay, wpevent, head, speed, cam,
                 typ, prefix):
        self.append(["Latitude", lat])
        self.append(["Longitude",lon])
        self.append(["Radius", rad])
        self.append(["Altitude", alt])
        self.append(["ClimbRate", climb])
        self.append(["DelayTime", delay])
        self.append(["WP_Event_Channel_Value", wpevent])
        self.append(["Heading", head])
        self.append(["Speed", speed])
        self.append(["CAM-Nick", cam])
        self.append(["Type", typ])
        self.append(["Prefix", prefix])
#--------------------------------------------------------------------
class WPVektor(deque):
    def __init__(self):
        self.__veksize=0
        self.__FileVersion=3

    def append(self, other):
        self.__veksize=self.__veksize+1
        super(WPVektor,self).append(other)

    def createWPFile(self,filename):
        f=open(filename,'w')
        f.write("[General]\n")
        f.write("FileVersion="+str(self.__FileVersion)+"\n")
        f.write("NumberOfWaypoints=" + str(self.__veksize) + "\n")
        j=1
        f.write("[Point"+str(j)+"]\n")
        while True:
            try:
                x=self.popleft()
                while True:
                    try:
                        y = x.popleft()
                        if (y[0]=='Type' and y[1]==0):
                            y[1] = 1
                        elif (y[0]=='Type' and y[1]==1):
                            y[1] = 2
                        f.write(y[0] + "=" + str(y[1])+"\n")

                    except IndexError:
                        break
                j=j+1
                if j <= self.__veksize:
                    f.write("[Point"+str(j)+"]\n")
            except IndexError:
                break
    # for debug purpose
    def print2Screen(self):
        print self.__veksize
        while True:
            try:
                x=self.popleft()
```

```
                while True:
                    try:
                        print x.popleft()
                    except IndexError:
                        break
                print
            except IndexError:
                break
#--------------------------------------------------------------------
class KSerCom(Serial):

    def __init__(self, port, baud):
        self._zero = ord('=')
        self.triplet = struct.Struct('BBB')
        Serial.__init__(self, port, baud)

    def calcCrcBytes(self,st):
        crc = 0
        for c in st:
            crc += ord(c)
        crc %= 4096
        return chr(crc / 64 + ord('='))+ chr(crc % 64 + ord('='))

    def decode(self, code):
        data = ""
        while len(code) >= 4:
            a = ord(code[0]) - self._zero
            b = ord(code[1]) - self._zero
            c = ord(code[2]) - self._zero
            d = ord(code[3]) - self._zero
            x = ((a & 0x3f) << 2) | ((b & 0xf0) >> 4)
            y = ((b & 0x0f) << 4) | ((c & 0xfc) >> 2)
            z = ((c & 0x03) << 6) |  (d & 0x3f)

            data += self.triplet.pack(x, y, z)
            code = code[4:]
        return data

    def encode(self, data):
        code = ""
        while len(data):
            while len(data)<3:
            data += chr(0)
            (x, y, z) = self.triplet.unpack(data[:3])

            a = (x & 0xfc) >> 2
            b = ((x & 0x03) << 4) | ((y & 0xf0) >> 4)
            c = ((y & 0x0f) << 2) | ((z & 0xc0) >> 6)
            d = z & 0x3f

            code += chr( self._zero + a )
            code += chr( self._zero + b )
            code += chr( self._zero + c )
            code += chr( self._zero + d )
            data = data[3:]
        return code

    def createFrame(self, cmd, board, data):
        self.frame = []
        self.header = '#' + board + cmd
        self.payload = self.encode(data)
        self.frame =  self.header + self.payload
        self.crc = self.calcCrcBytes(self.frame)
        self.frame = self.frame + self.crc + '\r'
        return self.frame
```

```
    def waitForLn(self):
        answer = self.readline(None, eol='\r')
        time.sleep(0.1)
        return answer

    def kResetRedirectUART(self):
        reset = "#" + chr(0x1b) + chr(0x1b)+ chr(0x55) + chr(0xaa) + chr(0) + '\r'
        self.write(reset)
        time.sleep(0.1)
        self.write('\r')
        return

    def kClearWP(self):
        self.write(self.createFrame('w', 'c', struct.Struct('30x').pack()))
        while True:
            ans_frame = self.readline(None, eol='\r')
            #Check set WayPoint answer
            if (ans_frame[0] == '#') and (ans_frame[2] == 'W'):
                #if Version answer check checksum of answer frame
                # 1st part received CRC, 2nd part calculated CRC of ans_frame
                if (ans_frame[-3:-1] != self.calcCrcBytes(ans_frame[:-3])):
                    return "CRC ERROR"
                # cut payload from frame and decode base64
                backcut = ans_frame[:-3]
                payload = backcut[3:]
                receive_bdata = self.decode(payload)
                receive_bdata
                gotWPNo =  struct.unpack('BBB', receive_bdata)
                return gotWPNo[0]

    def kSendWP(self, index, lat, lon, rad, alt, climb, delay, wpevent, head, speed,
                cam, typ, prefix):
        wp_gps_status = chr(0x01) # NEWDATA
        wp_lon = int(lon*10**7)
        wp_lat = int(lat*10**7)
        wp_alt = int(alt*10)
        gps_pos_t = struct.Struct('iiic').pack(wp_lat, wp_lon,wp_alt, wp_gps_status)

        wp_heading = int(head) # u16
        wp_tolrad = chr(rad) # u8
        wp_holdtime = chr(delay) #u8
        wp_eventflag = chr(0x00) #u8
        wp_index = chr(index) #u8
        wp_type = chr(typ) #u8
        wp_eventchvalue = chr(wpevent) #u8
        wp_altrate = chr(climb) #u8
        wp_speed = chr(speed) #u8
        wp_camangel = chr(cam) #u8
        wp_name = str(prefix)
        point_t = struct.Struct('Hccccccccccxxxxx').pack(wp_heading, wp_tolrad, \
                                wp_holdtime, wp_eventflag,\
                            wp_index, wp_type, wp_eventchvalue, wp_altrate, wp_speed,
                            wp_camangel, wp_name)
        wpStruct = gps_pos_t + point_t
        self.write(self.createFrame('w', 'c', wpStruct))
        while True:
            ans_frame = self.readline(None, eol='\r')
            #Check set WayPoint answer
            if (ans_frame[0] == '#') and (ans_frame[2] == 'W'):
                #if Version answer check checksum of answer frame
                # 1st part received CRC, 2nd part calculated CRC of ans_frame
                if (ans_frame[-3:-1] != self.calcCrcBytes(ans_frame[:-3])):
                    return "CRC ERROR"
                # cut payload from frame and decode base64
                backcut = ans_frame[:-3]
                payload = backcut[3:]
```

```
                receive_bdata = self.decode(payload)
                gotWPNo =  struct.unpack('BBB', receive_bdata)
                return gotWPNo[0]
#--------------------------------------------------------------------
# Main
myFC = arcpy.GetParameterAsText(0)
myWPFile = arcpy.GetParameterAsText(1)
comIF = arcpy.GetParameterAsText(2) #COM1 = 0
BAUD = 57600 # Kopter Baudrate
mySortedFC ="in_memory\\tmp_fc"

myKser = KSerCom(int(comIF), BAUD)
myKser.kResetRedirectUART()
myKser.setTimeout(20)

vektor=WPVektor()

if (arcpy.Describe(myFC).ShapeType) != "Point":
    arcpy.AddError("Geometrie of Input FC is not Point")
    sys.exit(1)

if (arcpy.Describe(myFC).spatialReference.name) != "GCS_WGS_1984":
    arcpy.AddError("Spatial Reference of Input FC is not WGS84")
    arcpy.AddError("Spatial Reference is: " +
arcpy.Describe(myFC).spatialReference.name)
    sys.exit(1)

try:
    arcpy.AddMessage("\nClearWPList number of WP: ....." + str(myKser.kClearWP())+"\n")
    arcpy.Sort_management(myFC, mySortedFC, [["Id", "ASCENDING"]])
    sCur = arcpy.SearchCursor(mySortedFC)

    index = 1
    for row in sCur:
            lat=round(row.getValue("Shape").getPart().Y,7)
            lon=round(row.getValue("Shape").getPart().X,7)
            rad=row.getValue("Radius")
            alt=row.getValue("Altitude")
            climb=row.getValue("ClimbRate")
            delay=row.getValue("DelayTime")
            wpevent=row.getValue("WP_Event_C")
            head=row.getValue("Heading")
            speed=row.getValue("Speed")
            cam=row.getValue("CAM_Nick")
            typ=row.getValue("Type")
            prefix=row.getValue("Prefix")
            vektor.append(WayPoint(lat, lon, rad, alt, climb, delay, wpevent, head,
                                   speed, cam, typ, prefix))
            setwpno = myKser.kSendWP(index, lon, lat, rad, alt, climb, delay, wpevent,
                                     head, speed, cam, typ, prefix)
            arcpy.AddMessage("Set WayPoint No: " + str(setwpno))
            index = index +1
            time.sleep(0.2)
    if myWPFile != '':
        vektor.createWPFile(myWPFile)
except Exception as e:
    print e.message
finally:
    myKser.close()
    del row, sCur, mySortedFC
```

Tabelle 9.1: Sourcecode Pythonscript MKSerialCommunikation

```
import arcpy, sys
from collections import deque

#-------------------------------------------------------------------------------------------
class WayPoint(deque):
    def __init__(self, lat, lon, rad, alt, climb, delay, wpevent, head, speed, cam,
typ, prefix):
        self.append(["Latitude", lat])
        self.append(["Longitude",lon])
        self.append(["Radius", rad])
        self.append(["Altitude", alt])
        self.append(["ClimbRate", climb])
        self.append(["DelayTime", delay])
        self.append(["WP_Event_Channel_Value", wpevent])
        self.append(["Heading", head])
        self.append(["Speed", speed])
        self.append(["CAM-Nick", cam])
        self.append(["Type", typ])
        self.append(["Prefix", prefix])
#-------------------------------------------------------------------------------------------
class WPVektor(deque):
    def __init__(self):
        self.__veksize=0
        self.__FileVersion=3

    def append(self, other):
        self.__veksize=self.__veksize+1
        super(WPVektor,self).append(other)

    def createWPFile(self,filename):
        f=open(filename,'w')
        f.write("[General]\n")
        f.write("FileVersion="+str(self.__FileVersion)+"\n")
        f.write("NumberOfWaypoints=" + str(self.__veksize) + "\n")
        j=1
        f.write("[Point"+str(j)+"]\n")
        while True:
            try:
                x=self.popleft()
                while True:
                    try:
                        y = x.popleft()
                        if (y[0]=='Type' and y[1]==0):
                            y[1] = 1
                        elif (y[0]=='Type' and y[1]==1):
                            y[1] = 2
                        f.write(y[0] + "=" + str(y[1])+"\n")

                    except IndexError:
                        break
                j=j+1
                if j <= self.__veksize:
                    f.write("[Point"+str(j)+"]\n")
            except IndexError:
                break
    # for debug purpose
    def print2Screen(self):
        print self.__veksize
        while True:
            try:
                x=self.popleft()
                while True:
                    try:
                        print x.popleft()
                    except IndexError:
                        break
```

```
                print
            except IndexError:
                break

    def WPFilefromFC(self, fc):
        pass

#-------------------------------------------------------------------------------
# Main

myFC = arcpy.GetParameterAsText(0)
myWPFile = arcpy.GetParameterAsText(1)
mySortedFC ="in_memory\\tmp_fc"

if (arcpy.Describe(myFC).ShapeType) != "Point":
    arcpy.AddError("Geometrie of Input FC is not Point")
    sys.exit(1)
try:
    ## Sort by ID in memory Tool DataManagement/General/Sort
    arcpy.Sort_management(myFC, mySortedFC, [["Id", "ASCENDING"]])
    sCur = arcpy.SearchCursor(mySortedFC)
    vektor=WPVektor()
    for row in sCur:
            lat=round(row.getValue("Shape").getPart().Y,7)
            lon=round(row.getValue("Shape").getPart().X,7)
            rad=row.getValue("Radius")
            alt=row.getValue("Altitude")
            climb=row.getValue("ClimbRate")
            delay=row.getValue("DelayTime")
            wpevent=row.getValue("WP_Event_C")
            head=row.getValue("Heading")
            speed=row.getValue("Speed")
            cam=row.getValue("CAM_Nick")
            typ=row.getValue("Type")
            prefix=row.getValue("Prefix")
            vektor.append(WayPoint(lat, lon, rad, alt, climb, delay, wpevent, head,
                           speed, cam, typ, prefix))
    vektor.createWPFile(myWPFile)
except Exception as e:
    print e.message
finally:
    del row, sCur, mySortedFC
```

Tabelle 9.2: Sourcecode Pythonscript CreateWPFile

```
[General]
FileVersion=3
NumberOfWaypoints=3
[Point1]
Latitude=48.3294174
Longitude=14.2916972
Radius=2
Altitude=5
ClimbRate=5
DelayTime=10
WP_Event_Channel_Value=0
Heading=359
Speed=10
CAM-Nick=45
Type=1
Prefix=P
[Point2]
Latitude=48.3296074
Longitude=14.2917578
Radius=2
Altitude=5
ClimbRate=5
DelayTime=60
WP_Event_Channel_Value=0
Heading=359
Speed=10
CAM-Nick=45
Type=1
Prefix=S
[Point3]
Latitude=48.3294046
Longitude=14.2915991
Radius=2
Altitude=5
ClimbRate=5
DelayTime=10
WP_Event_Channel_Value=0
Heading=359
Speed=10
CAM-Nick=45
Type=1
Prefix=L
```

Tabelle 9.3: Erzeugtes Mikrokopter Waypointfile des Experimentes

```
<trackFields>
. . . . . .
. . . . . . . . Default fields
. . . . . .

<field visible="true" precision="2" length="8" type="number" name="sat" />
<field visible="true" precision="0" length="25" type="string" name="Altimeter" />
<field visible="true" precision="0" length="25" type="string" name="Variometer" />
<field visible="true" precision="0" length="5" type="number" name="Course" />
<field visible="true" precision="0" length="5" type="number" name="GroundSpeed" />
<field visible="true" precision="0" length="5" type="number" name="VerticalSpeed" />
<field visible="true" precision="0" length="5" type="number" name="FlightTime" />
<field visible="true" precision="2" length="3" type="number" name="Voltage" />
<field visible="true" precision="2" length="3" type="number" name="Current" />
<field visible="true" precision="0" length="9" type="number" name="Capacity" />
<field visible="true" precision="0" length="3" type="number" name="RCQuality" />
<field visible="true" precision="0" length="3" type="number" name="RCRSSI" />
<field visible="true" precision="0" length="12" type="string" name="Compass" />
<field visible="true" precision="0" length="3" type="number" name="NickAngle" />
<field visible="true" precision="0" length="3" type="number" name="RollAngle" />
<field visible="true" precision="0" length="3" type="number" name="MagnetField" />
<field visible="true" precision="0" length="12" type="string"
name="MagnetInclination" />
<field visible="true" precision="0" length="48" type="string" name="MotorCurrent" />
<field visible="true" precision="0" length="48" type="string" name="BL_Temperature" />
<field visible="true" precision="0" length="3" type="number"
name="AvaiableMotorPower" />
<field visible="true" precision="0" length="3" type="number" name="FC_I2C_ErrorCounter"
/>
<field visible="true" precision="0" length="24" type="string" name="AnalogInputs" />
<field visible="true" precision="0" length="12" type="string" name="NCFlag" />
<field visible="true" precision="0" length="24" type="string" name="Servo" />
<field visible="true" precision="0" length="254" type="string" name="WP" />
<field visible="true" precision="0" length="24" type="string" name="FCFlags2" />
<field visible="true" precision="0" length="3" type="number" name="ErrorCode" />
<field visible="true" precision="0" length="3" type="number" name="TargetBearing" />
<field visible="true" precision="0" length="3" type="number" name="TargetDistance" />
<field visible="true" precision="0" length="124" type="string" name="RCSticks" />
<field visible="true" precision="2" length="24" type="string" name="GPSSticks" />
</trackFields>
```

Tabelle 9.4: Attributerweiterung des DNRGPS Konfiguration-File (Ausschnitt)

10. Internetquellen

[6kinematik] <http://www.electronic-engineering.ch/study/ins/Low-Cost_INS_Report_DE.pdf>
Letzter Zugriff: 4.10.2012

[µBlox LEA-4H] <http://www.ublox.com/images/downloads/Product_Docs/LEA-4x_Data_Sheet(GPS.G4-MS4-06143).pdf>

Letzter Zugriff: 30.7.2012

[µBlox ubx] <http://www.u-blox.com/images/downloads/Product_Docs/u-blox5_Protocol_Specifications(GPS.G5-X-07036).pdf>

Letzter Zugriff: 3.10.2012

[apos]
<http://www.bev.gv.at/portal/page_pageid=713,2152237&_dad=portal&_schema=PORTAL>

Letzter Zugriff: 12.11.2012

[arcgishelp gdb]
<http://help.arcgis.com/de/arcgisdesktop/10.0/help/index.html#//003n00000007000000>

Letzter Zugriff: 3.11.2012

[arcgis spatstat]
<http://help.arcgis.com/de/arcgisdesktop/10.0/help/index.html#//005p00000014000000>

Letzter Zugriff: 8.11.2012

[atmega 1284] <http://www.atmel.com/devices/atmega1284.aspx>

Letzter Zugriff: 28.7.2012

[bev wgs84]
<http://www.bev.gv.at/pls/portal/docs/PAGE/BEV_PORTAL_CONTENT_ALLGEMEIN/0200_PRODUKTE/PDF/KOORD-SYS.PDF>

Letzter Zugriff: 6.7.2012

[ccby3.0] <http://creativecommons.org/licenses/by/3.0/at/>

Letzter Zugriff: 14.7 2012

[egnos] <http://egnos-portal.gsa.europa.eu/>

Letzter Zugriff: 14.7.2012

[egnos user] <http://egnos-user-support.essp-sas.eu/egnos_ops/index.php>

Letzter Zugriff: 21.10.2012

[egons rims] <http://www.egnos-pro.esa.int/IMAGEtech/imagetech_realtime.html>

Letzter Zugriff: 14.11.2012

[geoland trans] <http://www.geoland.at/gps_trans.htm>

Letzter Zugriff: 26.10.2012

[harre navgen] <http://www.mar-it.de/NavGen/navgen.htm>

Letzter Zugriff: 15.11.2012

[kalman] <http://www.mi.hs-rm.de/~schwan/Projects/CG/CarreraCV/doku/kalman/kalman.htm>

Letzter Zugriff: 3.9.2012

[luftrecht at] <http://www.luftfahrtrecht.at/home/home.php>

Letzter Zugriff: 17.9. 2012

[mikrokopter wiki mkm] <http://www.mikrokopter.de/ucwiki/mkm>

Letzter Zugriff: 20.9.2012

[mikrokopter wiki gpx] <http://www.mikrokopter.de/ucwiki/GPX>

Letzter Zugriff: 20.7.2012

[mikrokopter wiki wp] <http://www.mikrokopter.de/ucwiki/MikroKopterTool-OSD#Wegpunkte_-_Vorgaben>

Letzter Zugriff: 20.7.2012

[mikrokopter wiki alti] <http://www.mikrokopter.de/ucwiki/Höhensensor?highlight=Luftdrucksensor>

Letzter Zugriff: 23.9.2012

[mikrokopter wiki mk3538] <http://www.mikrokopter.de/ucwiki/MK3538>

Letzter Zugriff: 25.7.2012

[mksvn] <http://svn.mikrokopter.de/>

Letzter Zugriff: 4.Nov. 2012

[mkser] <http://www.mikrokopter.de/ucwiki/en/SerialProtocol>

Letzter Zugriff: 4.Nov. 2012

[mpx4115] <http://www.freescale.com/files/sensors/doc/data_sheet/MPX4115.pdf>

Letzter Zugriff: 28.7.2012

[navi arm] <http://www.st.com/internet/mcu/product/165893.jsp>

Letzter Zugriff: 30.8.2012

[python struct] <http://docs.python.org/2/library/struct.html>

Letzter Zugriff: 6.1.2012

[wiki CPE] <http://de.wikipedia.org/wiki/Circular_Error_Probable>

Letzter Zugriff: 26.10.2012

[wikipedia i2c] <http://de.wikipedia.org/wiki/I2C>

Letzter Zugriff: 25.9.2012

[wikipedia lat] <http://de.wikipedia.org/wiki/Geographische_Breite>

Letzter Zugriff: 1.10.2012

11. Literaturverzeichnis

Al-Hashimi, B. 2006. *System-on-chip: next generation electronics*. Iet.

Bauer, M. 2003. *Vermessung und Ortung mit Satelliten*. Herbert Wichmann Verlag.

Böswirth, L. 2010. *Technische Strömungslehre: Lehr-und Übungsbuch*. Vieweg+ Teubner Verlag.

Bures, B. D. (2011) BM für Verkehr, Innovation und Technolgie Anfragebeantwortung. 7668/AB XXIV. GP.

Gerdan, G. P. & R. E. Deakin (1999) Transforming cartesian coordinates X, Y, Z to geographical coordinates ϕ, λ, h. *Australian surveyor,* 44**,** 55.

Gustav Pomberger, R. P., René Riedl, Stefan Schiffer (2010) Studie Open-Commons-Region Linz.

Nyce, D. S. 2004. *Linear Position Sensors: Theory and Application*. Wiley Online Library.

Semiconductors, P. (2000) The I2C-bus specification. *Philips Semiconductors,* 9397**,** 00954.

Vermessungswesen, B. f. E.-u. (2007) Tranformation von MGI nach wgs84.

Welch, G. & G. Bishop. 1995. An introduction to the Kalman filter.

Wendel, J. 2011. *Integrierte Navigationssysteme: Sensordatenfusion, GPS und Inertiale Navigation*. Oldenbourg Wissenschaftsverlag.