UNIVERSITÄT SIEGEN
Naturwissenschaftlich Technische Fakultät
Departement Elektrotechnik und Informatik
Lehrstuhl für Regelungs- Und Steuerungstechnik

Aufbau und Steuerung des DJI F450 Quadcopters über den Pixhawk-Flugcontroller

Studienarbeit

im Studiengang Elektrotechnik
Automatisierungs- und Energietechnik

vorgelegt von

Amina Dridi
Matrikelnummer: 1026597

an der
Universität Siegen

August 2019

Erstprüfer: Prof. Dr. Hubert Roth
Zweitprüfer: M. sc. Omar Gamal

Eidesstattliche Erklärung

Ich, Amina Dridi, Matrikel-Nr. 1026597, versichere hiermit, dass ich meine Studienarbeit mit dem Thema

" Aufbau und Steuerung des DJI F450 Quadcopters über den Pixhawk-Flugcontroller "

selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, wobei ich alle wörtlichen und sinngemäßen Zitate als solche gekennzeichnet habe. Die Arbeit wurde bisher keinem anderen Prüfungsamt vorgelegt und auch nicht veröffentlicht.

Siegen, den 23.08.2019

_____

Amina Dridi

Danksagung

An dieser Stelle möchte ich mich bei all denjenigen bedanken, die mich bei der Anfertigung dieser Studienarbeit unterstützt und motiviert haben.

Zuerst gebührt mein herzlicher Dank Herrn Professor Hubert Roth, der mir die Möglichkeit gegeben hat, meine Studienarbeit im Bereich der Regelungs- und Steuerungstechnik zu schreiben, wonach ich mich ebenfalls bei meiner Masterarbeit richten werde.

Ein besonderer Dank gilt Herrn M. Sc. Omar Gamal, der meine Studienarbeit betreut und in zahlreichen Stunden Korrektur gelesen hat. Für die Mühe, die hilfreiche Erklärungen und die konstruktive Kritik bei der Erstellung dieser Arbeit, möchte ich mich herzlich bedanken.

Ebenfalls geht ein großer Dank an Herrn Dipl.-Ing. Peter Sahm, der mir wertvolle Hinweise und hilfreiche Tipps zur Verfügung gestellt hat.

Abschließend möchte ich mich bei meinen Eltern bedanken, die mir mein Studium durch ihre Unterstützung ermöglicht haben und stets ein offenes Ohr für meine Sorgen hatten.

Inhaltsverzeichnis

Inhaltsverzeichnis I

Abbildungsverzeichnis III

Tabellenverzeichnis V

Abkürzungsverzeichnis VI

1 Einleitung . . . 1
1.1 Motivation . . . 1
1.2 Aufgabenstellung . . . 1
2 Theoretische Grundlagen . . . 3
2.1 Quadcopter . . . 3
2.1.1 Flugdynamik des Quadcopters . . . 3
2.1.2 Komponenten des Quadrocopters . . . 5
2.1.2.1 Sensorsysteme . . . 5
2.1.2.2 Steuerungssysteme . . . 8
2.1.2.3 Funksteuerung . . . 10
2.1.2.4 Akkumulator . . . 10
2.1.2.5 Elektronischer Drehzahlregler . . . 11
2.1.2.6 Brushless Gleichstrommotoren . . . 12
2.1.2.7 Propeller . . . 12
2.1.2.8 Rahmen . . . 12
2.1.3 Regelung des Quadcopters . . . 13
2.2 Robot Operating System . . . 15
2.2.1 ROS-Pakete . . . 17
3 Aufbau der Hardware . . . 19
3.1 3D-Modellierung der PixHawk- und RC-Empfänger-Box . . . 19
3.2 Verkabelung des DJI F450 Quadrocopters . . . 20
3.2.1 Positionierung des Flugcontrollers . . . 21
3.2.2 Anschluss der Versorgungsleitung . . . 21
3.2.3 Anschließen des Funkempfängers . . . 21
3.2.4 Anschluss der DJI E300 15A OPTO-ESCs und 2212/920 KV-Motoren . . . 22
3.2.5 Anschluss des RPI3 Model B+ . . . 22

3.2.6 Anschließen der Kamera 23
4 Betrieb des PixHawk-Flugcontrollers mit der PX4-Firmware 24
4.1 Einrichten des FC über den MP 24
4.1.1 Auswahl des Fluggerätetyps und Sensorkalibrierung . . 24
4.1.2 Kalibrierung des RC-Senders 25
4.1.3 Motor- und ESC-Einstellung 26
4.1.4 Motortest über MP 27
4.1.5 PID-Tuning über MP 28
4.2 Einrichten des FC über den QGC 33
4.2.1 Einstellen des Airframe und Sensorkalibrierung 33
4.2.2 Einstellen des RC-Systems und der Flugmodi 35
4.2.3 Motoreinstellungen 38
4.2.4 Testen der BLDC-Motoren und der E300-ESCs 38
4.2.5 PID-Tuning über QGC 39
4.3 Vergleich zwischen QGC- und MP-Software 40
5 Autonome Steuerung des Quadcopters mittels ROS 41
5.1 Raspberry PI 3 41
5.2 Ubuntu-Mate- und ROS-Installation auf dem RPI3 42
5.3 Ansteuerung des Quadcopters mittels ROS und Apriltags 42
5.3.1 Bildverarbeitung 42
5.3.2 Detektierung des Apriltags2 43
5.4 Herstellung der Kommunikation zwischen PixHawk und ROS über den Raspberry PI 3 44
5.5 Takeoff- und Land-Modus mittels der Apriltags-Erkennung . . . 45
6 Ergebnisse 56
6.1 Flugtest 56
7 Diskussion und Interpretation der Ergebnisse 60

Literaturverzeichnis **61**

Anhang **64**

Abbildungsverzeichnis

Abbildung 2.1 Quadcopter-Konfigurationen 3
Abbildung 2.2 Physikalisches Modell des Quadcopters [2] 4
Abbildung 2.3 Beschleunigungssensor [4] 6
Abbildung 2.4 Gyroskop [5] 7
Abbildung 2.5 Pixhawk-Flugcontroller [9] 9
Abbildung 2.6 Aufbau eines Drehzahlreglers 11
Abbildung 2.7 PID-Regelkreis 13
Abbildung 2.8 Überwachung der Akku-Spannung 16
Abbildung 2.9 Einschalten eines Motors 16
Abbildung 2.10 AprilTag, ID:12 [15] 17
Abbildung 3.1 Autopilot-Box 19
Abbildung 3.2 RC-Empfänger-Box 20
Abbildung 3.3 Aufbau des Quadcopter-Systems 20
Abbildung 3.4 Blockschaltbild des Funkempfängersystems 21
Abbildung 3.5 FC-ESC-Motor-Verbindung 22
Abbildung 3.6 RPI3-PixHawk-Verbindung [17] 23
Abbildung 4.1 Modell-Konfiguration [28] 24
Abbildung 4.2 ACC-Kalibrierung [29] 25
Abbildung 4.3 Kompasskalibrierung [30] 25
Abbildung 4.4 Sender-Kalibrierung [31] 26
Abbildung 4.5 Hauptausgänge des PixHawk-Autopilots [32] 27
Abbildung 4.6 Reihenfolge der Motoren [33] 27
Abbildung 4.7 Motortest in MP [34] 27
Abbildung 4.8 Reihenfolge der Motoren [35] 28
Abbildung 4.9 Attitude Control 28
Abbildung 4.10 Versuchsaufbau für das PID-Tuning 29
Abbildung 4.11 Anfangswert P = 0,13 30
Abbildung 4.12 Optimaler Wert P = 0,18 30
Abbildung 4.13 Schwingungserzeugender Wert P = 0,19 30
Abbildung 4.14 Anfangswert D = 0,001 31
Abbildung 4.15 Optimaler Wert D = 0,006 31
Abbildung 4.16 Schwingungserzeugender Wert D = 0,007 31
Abbildung 4.17 Anfangswert I = 0,03 32

Abbildung 4.18 Optimaler Wert I = 0,09 32
Abbildung 4.19 Schwingungserzeugender Wert I = 0,1 32
Abbildung 4.20 Orientierung des PixHawk-Autpilots [20] 33
Abbildung 4.21 Kompasskalibrierung [21] 34
Abbildung 4.22 Beschleunigungssensorkalibrierung [22] 34
Abbildung 4.23 Kalibrierung des Gyroskops [23] 35
Abbildung 4.24 Kalibrierung des level Horizon [24] 35
Abbildung 4.25 Dx6i-SPEKTRUM-Funksender [25] 36
Abbildung 4.26 Kalibrierung der Funke [26] 36
Abbildung 4.27 Motor-ESC-Test 39
Abbildung 5.1 Raspberry PI 3 Model B+ [36] 41
Abbildung 5.2 ROS-Graph der usb_cam-Node vor der Kamerakalibrierung 42
Abbildung 5.3 ROS-Graph der usb_cam-Node nach der Kamerakalibrierung 43
Abbildung 5.4 ROS-Graph der usb_cam- und apriltags2_ros-Node 44
Abbildung 5.5 Setzen des Offbord-Modus und Einschalten der Motoren . . 47
Abbildung 5.6 Takeoff durch Detektierung des Apriltags-ID12 49
Abbildung 5.7 Land durch Detektierung des Apriltags-ID13 51
Abbildung 5.8 Initialisierung der Variablen 53
Abbildung 5.9 Land durch Detektierung des Apriltags-ID12 54
Abbildung 5.10 Land durch Detektierung des Apriltags-ID13 55
Abbildung 6.1 Flugtest 56
Abbildung 6.2 Soll-/Istwert-Vergleich der Roll-Bewegung 57
Abbildung 6.3 Soll-/Istwert-Vergleich der Pitch-Bewegung 57
Abbildung 6.4 Soll-/Istwert-Vergleich der Yaw-Bewegung 57
Abbildung 6.5 Soll-/Istwert-Vergleich der Roll-Bewegung 58
Abbildung 6.6 Soll-/Istwert-Vergleich der Pitch-Bewegung 58
Abbildung 6.7 Soll-/Istwert-Vergleich der Yaw-Bewegung 59

Tabellenverzeichnis

Tabelle 4.1 Steuerungskanäle gemäß MP 26
Tabelle 4.2 Flugmodi in MP 26
Tabelle 4.3 Arbeitspunkt des Attitude-Reglers 33
Tabelle 4.4 Steuerungskanäle gemäß QGC 37
Tabelle 4.5 Flugmodi in QGC 38

Abkürzungsverzeichnis

FC Flugcontroller

RPI 3 Raspberry PI 3

GPS Global Postioning System

ACC Accelerometer

Gyro Gyroskop

SPI Serial Peripheral Interface

I^2C Inter Integrated Circuit

USB Universal Serial Bus

UART Universal Asynchronous Receiver Transmitter

PWM Pulsweitenmodulation

PPM Puls Position Modulation

RC Radio Control

Lipo-Akku Lithium Polymer-Akkumulator

PM Power Modul

BLDC-Motor Bürstenloser Gleichstrommotor

Textausrichtung

THRO Throttle

AILE Aileron

ELEV Elevator

RUDD Rudder

QGC QGroundControl

MP Mission Planner

TELEM Telemetry

1 Einleitung

Ein Quadcopter ist eine Drohne mit einer Steuereinheit im Schwerpunkt und vier Antrieben (Rotoren und Propellern), die in einer Ebene angeordnet sind. Quadcopter sind in verschiedenen Ausführungen zu finden und lassen sich rein über die Drehzahländerung der einzelnen Motoren steuern. Sie können mit allerlei Sensoren erweitert werden, wodurch eine präzise Verwendung sowie eine maximale Sicherheit gewährleistet wird. Durch die Kombination von Sensorsystemen wird die Information über die Fluglage sowie die Position bereitgestellt. So kann die Steuerung eines Quadcopters manuell über einen Funksender oder autonom über eine GPS-Unterstützung erfolgen. Aufgrund dessen sind ihre Einsatzgebiete vielfältig. Denn neben den Modellen für Hobby-Piloten lassen sich größere und professionelle Modelle für dienstliche Anwendungen benötigen. Hierzu zählen zahlreiche Einsätze, wie beispielsweise in Foto- und Videographie, Unfälle verschiedenster Art, Militärdienste, Personensuche, Transport von Hilfsgütern in Entwicklungsländern, Warenlieferungen und Versenden von Paketen. Ausgestattete Quadcopter mit kleiner Last wie Kamera können dank ihrer vier Motoren einen Auftrieb erzeugen, ohne dabei ihre Beschleunigung und Wendigkeit zu verlieren.

1.1 Motivation

Regelungs- und Steuerungstechnik ist ein Teilgebiet der Elektrotechnik, das erforderlich ist, um Systeme zu beeinflussen und deren Verhalten zu stabilisieren. Denn ohne Messungen durchzuführen und Regelalgorithmen zu implementieren, kann ein System nicht erfolgreich geregelt werden.
Der Quadcopter kann als fliegender Roboter Bewegungen in sechs Freiheitsgraden ausführen. So stellt sich oft die Problematik dabei, wie eine optimale Regelung zu erreichen ist, um eine richtige Steuerung zu verwirklichen, bei der der Quadcopter ruhig und stabil in der Luft fliegt. Ferner ist es zu erzielen, dass der Quadcopter auch bei autonomen Steuerungen seine Stabilität beibehält. So kann er in verschiedenen Modi bzw. auf verschiedener Arten gesteuert werden. In diesem Fall kann eine autonome Betriebsnahme des Quacopters mit anderen Bereichen wie Echtzeitlernsysteme und Computer vision kombiniert werden und macht sich als intelligentes System in vielen Bereichen vorteilhaft verwendbar.

1.2 Aufgabenstellung

Prinzipiell besteht die Studienarbeit aus drei Teillaufgaben: Die Verkabelung des Quacopters, das Einrichten des Flugcontrollers über eine Bodenstation-Software und die Steuerung der Drohne im Manual-, Land- und Offboard-Modus. Zuerst wird der Quadcopter auf die notwendigen Bauteile erweitert. Diese sollen so platziert werden, dass die Leitungen vorteilhaft geführt werden können und der Drohne ein Gleichgewicht beim

Fliegen gewährleistet wird. Bei der zweiten Teilaufgabe werden die in dem Autopilot integrierten Sensoren für die PID-Regelung und die Funksteuerung kalibriert. Daraufhin werden Parameter eingestellt, wie beispielsweise der Bereich des PWM-Output-Signals und des RC-Input-Signals, die Flugmodi, die Quad-Konfiguration und die Sicherheit. Beim letzten Teil handelt es sich um die Steuerung der Drohne im Manual-Modus, bei dem die Sollwerte aus der Funke vorgegeben werden. Anschließend erfolgt das Land-Modus durch das Umschalten des Autopilots, sodass die Drohne automatisch landet. Abschließend wird die Drohne im Offboard-Modus durch ein auf dem Raspberry PI geschriebenen C++-Programm angetrieben. Hierbei erhält der Autopilot in Abhängigkeit von eingelesenen AprilTags-Daten die Positionssollwerte, die zum Takeoff und Land notwendig sind. Dies erfolgt unter Verwendung des Robot Operating System.

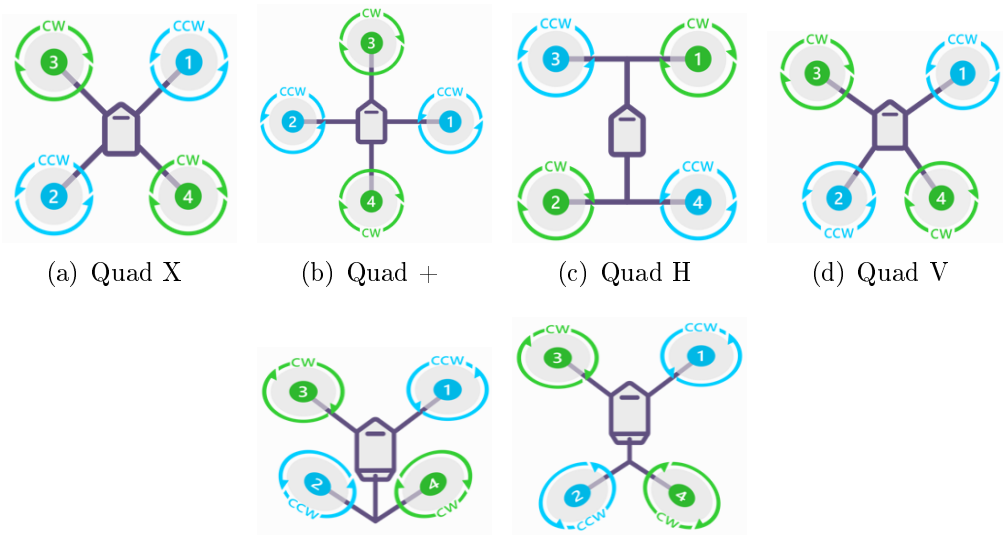
2 Theoretische Grundlagen

2.1 Quadcopter

2.1.1 Flugdynamik des Quadcopters

Bei einem Multicopter kann die Schubkraft jedes Propellers proportional über die Drehzahl des dazugehörigen Motors gesteuert werden, wodurch auch ein Drehmoment entgegen der Propeller-Drehrichtung generiert wird. Ein Quadcopter besitzt vier Antriebseinheiten. Dabei rotieren zwei Propeller im Uhrzeigersinn und zwei Propeller gegen den Uhrzeigersinn jeweils mit gleicher Geschwindigkeit. Dies führt dazu, dass ein Drehmomentausgleich durch die gegenläufige Propeller-Drehrichtungen entsteht. Das heißt, die von den Propellern auf die Flugplattform übertragenen Drehmomente heben sich gegenseitig auf, wodurch der Quadcopter seine Orientierung beibehält und eine Gier-Bewegung über die horizontalen Ebene deaktiviert wird.

Wie in der Abbildung 2.1 dargestellt ist, können Quadcopter in verschiedenen, symmetrischen und unsymmetrischen, Konfigurationen aufgebaut werden.



(a) Quad X (b) Quad + (c) Quad H (d) Quad V

(e) Quad-V Tail (f) Quad-A Tail

Abbildung 2.1: Quadcopter-Konfigurationen

Gewöhnlicherweise wird ein Quadcopter über eine +-, X- oder H-Konfiguration gesteuert, da diese symmetrisch sind. Hierbei beeinflusst der Unterschied zwischen denen das Flugverhalten des Quadcopters. Eine einfache Steuerung lässt durch die am meisten verbreiteten Konfiguration, die +-Konfiguration (Abbildung 2.1 (b)), ermöglichen. Denn eine Rotation um die Quer- und Längsachse wird durch die Variation der Drehzahl eines einzigen Motors erreicht. Bei der X- und H-Konfiguration (Abbildung 2.1 (a) und Abbildung 2.1 (c)) wird eine Flugänderung in der Quer- und Längsachse von einem

Motorpaar gesteuert. In dem Autopilot wird die verwendete Konfiguration eingestellt, damit die von der Funke gesendeten Steuersignale an die speziell für die jeweilige Konfiguration programmierte Steuerelektronik übertragen wird, die gegebenenfalls nach einer Drehzahlregelung die PWM-Signale für die einzelnen Motoren zur Verfügung stellt.

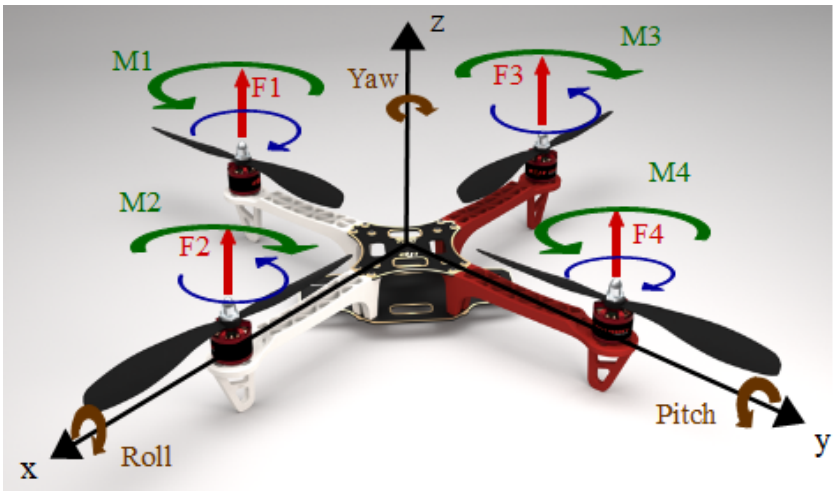


Abbildung 2.2: Physikalisches Modell des Quadcopters [2]

Wie auf der Abbildung 2.2 zu sehen ist, verfügt der Quadcopter über ein körpereigenes Koordinatensystem, das aus Nick-, Gier- und Rollachse besteht. Um den Quadcopter steigen oder sinken zu lassen (lineare Bewegung in positive/negative Richtung der z-Achse), werden die Drehgeschwindigkeiten aller Propeller gleichmäßig erhöht bzw. verringert. Eine Erhöhung der Drehgeschwindigkeit der Propeller bewirkt eine Erhöhung der Kraft, welche an den einzelnen Propellern orthogonal zur xy-Ebene wirkt. Für einen Abflug muss also die Summe aller an den einzelnen Propellern wirkenden Kräfte größer als die Gewichtskraft sein, für einen Landflug muss entsprechend die Gesamtpropellerkraft kleiner als die Gewichtskraft sein. Um die Drohne in der Luft bei einer bestimmten Höhe schweben zu lassen, muss die Gesamtpropellerkraft genau gleich der Gewichtskraft sein ($F_1 + F_2 + F_3 + F_4 = F_G$). Um eine lineare Bewegung entlang der x- und y-Achse zu erreichen, muss die Drehzahl einzelner Motoren verändert werden, wodurch eine Neigung entlang dieser Achsen zustande kommt. Eine Drehung der Drohne um die Gierachse erfolgt durch die Variation der Geschwindigkeiten aller Propeller. Somit findet keine Drehmoment-Neutralisierung statt und die Drohne kann sich um die eigene z-Achse rotieren. Eine Rotationsbewegung um die Roll- bzw. Nickachse wird dadurch ermöglicht, dass die auf der jeweils anderen Achse liegenden Motoren mit unterschiedlicher Drehzahl angesteuert werden. Die Veränderung der Drehzahl der Motoren erfolgt umgekehrt proportional, sodass die Summe aller Drehmomente konstant bleibt und die Drohne nicht in eine Gier-Bewegung gelangt. Hierbei ist erwähnenswert, dass die Konfiguration des Quadcopters eine Rolle bei der Drehzahlsteuerung der jeweiligen Motoren und somit bei der Bestimmung der Bewegungen spielt. Wie in Abbildung 2.2 erkennbar, besitzt der Quadcopter die X-Konfiguration, was bedeutet,

dass die Motoren geometrisch um 45° voneinander versetzt sind.

Steuerung der Motoren für Erzeugung einer bestimmten Rotation:

a) Gierachse (Yaw):
   - Im Uhrzeigersinn: M_2 und M_3 erhöhen, M_1 und M_4 proportional verringern.
   - Gegen den Uhrzeigersinn: M_1 und M_4 erhöhen, M_2 und M_3 proportional verringern.

b) Rollachse (Roll):
   - Rechts rollen: M_1 und M_2 erhöhen, M_3 und M_4 verringern.
   - Links rollen: M_3 und M_4 erhöhen, M_1 und M_2 verringern.

c) Nickachse (Pitch):
   - Vorne nicken: M_2 und M_4 erhöhen, M_1 und M_3 verringern.
   - Hinten nicken: M_1 und M_3 erhöhen, M_2 und M_4 verringern.

2.1.2 Komponenten des Quadrocopters

2.1.2.1 Sensorsysteme

Um ein System erfolgreich zu steuern und zu regeln, ist es erforderlich Messsysteme einzubauen. In diesem Unterkapitel wird ein Überblick über Sensoren gegeben, die zur Erfassung physikalischer Größen in den Quadcopter eingesetzt werden.

Global Positioning System

Bei einem autonomen Flug einer Drohne in Outdoor-Bereichen, bietet sich die Möglichkeit, die Drohne mit einem GPS-Modul zu erweitern, um es zu sichern, sie jederzeit zu verfolgen. Ein weiterer Grund für den Einsatz eines GPS ist die Positionsbestimmung. Über die GPS-Satelliten wird dafür gesorgt, dass der Quadcopter seine Position trotz des Luftwiderstandes oder starkem Seitenwind problemlos halten kann, indem der mit dem GPS-Modul verbundenen Autopilot selbstständig gegensteuert. Damit der GPS-Empfänger seine eigene Position und Geschwindigkeit genauer ermitteln kann, muss dieser mit mindestens vier GPS-Satelliten Kontakt haben. Theoretisch würden gesendete Signale aus drei Satelliten für die Bestimmung der x-,y- und z-Komponenten reichen. Da aber die Laufzeit für die Berechnung der Geschwindigkeit korrekt gemessen werden muss, wird ein viertes Satellit benötigt. Ebenfalls ist ein Kompass in das Navigationsgerät integriert. Jederzeit wird durch ein Informationssignal ermittelt, in welche Himmelrichtung sich der Quadcopter bewegt. Dabei ist der Vorteil, dass der Kompass immer funktioniert, falls das GPS-Gerät unter Umständen einen schwachen Empfang hat, da die Kompass-Funktionsweise auf dem Erdmagnetfeld basiert.

Beschleunigungssensor und Gyroskop

Damit der Quadcopter beim Fliegen ein stabiles Verhalten aufweisen kann, müssen die plötzlich auftretenden Kippmomente automatisch ausgeregelt werden. Als Maßnahme kommen für eine ordentliche Lageregelung Beschleunigungssensoren und Gyroskope zum Einsatz. Während die Beschleunigungssensoren zur Erfassung der translatorischen Bewegungen (in x-, y- und z-Richtung) dient, werden die Gyroskope für die Messung der rotatorischen Bewegungen (Gier-, Nick- und Roll-Winkel) eingesetzt. Diese sind als MEMS-Sensoren (microelectormechanical systems) zum Beispiel in den PixHawk-Flugcontroller integriert.

Messprinzip des ACC: Wie Anhand der Abbildung 2.2 zu sehen ist, bildet der Sensor ein Balken, welcher an zwei Federn angehängt ist und als träge Masse fungiert. An diesem Balken sind bewegliche Messstege angebracht, welche in ein festes Elektrodenpaar auf der Gegenseite hineinragen. Die zu messende Beschleunigung wird nach dem zweiten Newton'schen Axiom mit der Masse in Kraft umgesetzt (Aktionsprinzip) $F_a = m \cdot a$. Befindet sich das System im Ruhezustand, ist die Kapazität C zwischen den Messstegen und der jeweiligen starren Elektrode gleich. Wird nun der Sensor beschleunigt, so ändern sich die Kapazitäten ΔC durch die Abstandsänderung l, was wiederum messbar ist.

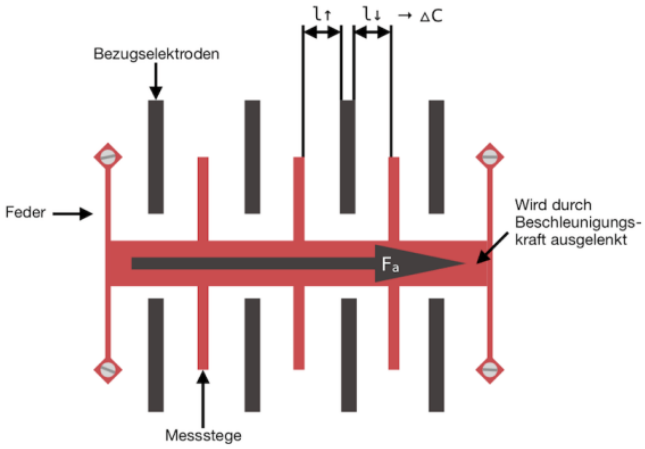


Abbildung 2.3: Beschleunigungssensor [4]

Messprinzip des Gyros: Nach dem ähnlichen Prinzip funktioniert das Gyrokop (Abbildung 2.3). Allerdings wird hier nicht die Beschleunigung gemessen, sondern die senkrecht zu der Bewegungsrichtung und Rotationsachse wirkende Corioliskraft F_c. Diese bewirkt ebenfalls eine zu messenden Kapazitätsänderung ΔC. Mit der Kombination beider Sensortypen kann die Fluglage des Quadcopters gemessen und somit geregelt werden.

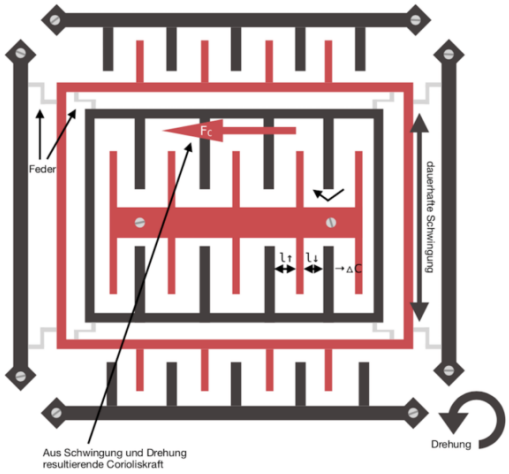


Abbildung 2.4: Gyroskop [5]

Ultraschallsensor

Um zu verhindern, dass der Quadcopter mit Gegenständen zusammenstößt, werden Ultraschallsensoren für die Distanzmessung eingesetzt. Diese gehören zu den akustischen Sensoren und sind in der Lage, Objekte berührungslos zu erkennen. Hierbei wird ein akustisches Signal ausgesendet, dessen Sendefrequenz außerhalb des menschlichen Hörvermögens liegt. Das Messprinzip beruht auf der Laufzeitmessung von hochfrequenten Schallimpulsen: Der Sensor sendet ein akustisches Signal aus. Trifft der Ultraschallimpuls auf einen Gegenstand, wird er reflektiert. Somit entsteht das sogenannte Echo, das vom Sensor wieder empfangen wird. Aus der Zeitspanne zwischen dem Senden und Empfangen des Signals und der festgelegten Schallgeschwindigkeit in der Luft lässt sich der Abstand zum Objekt berechnen.

$$r = \frac{c \cdot \Delta t}{2} \tag{2.1}$$

Mit r: Abstand, c: Schallgeschwindigkeit in Luft und Δt: Laufzeit zwischen Senden und Empfangen des Signals.

Barometer

Damit dem Quadcopter ein gelungenes Abfliegen bzw. Landen gewährleistet wird, sowie während der Navigation den vorgegebenen Höhensollwert erreicht wird, muss eine Höhenregelung erfolgen. Dabei muss die aktuelle Höhe für die Ermittlung der Regeldifferenz gemessen werden. Hierfür bietet sich neben vielen möglichen Sensortypen der Drucksensor. Im Vergleich zu den optischen und akustischen Sensoren arbeitet der Drucksensor nicht mittels der Distanzmessung, sondern berechnet den absoluten Druckwert. Damit kann jederzeit die absolute Höhe gegenüber dem Startpunkt be-

stimmt werden. So lautet die barometrische Höhenformel [6] wie folgend:

$$p(h) = p(h_0) \cdot e^{-\frac{mgh}{K_B T}} \tag{2.2}$$

Mit m: Masse, g: Erdbeschleunigung, T: Temperatur und K_B: Bolzmann Konstante.

Kamera

Eine Live view Bildübertragung ermöglicht sich durch das Ausstatten des Quadcopters mit einem Kamerasystem. Dies ist optimal, um jederzeit zu sehen, was die Drohne sieht und um die Orientierung bei großer Distanz beizubehalten. Darüber hinaus ermöglicht der Einbau einer Kamera eine autonome Navigation der Drohne und kann damit einen GPS-Einsatz ersetzen. Sie kann außerdem statische und dynamische Hindernisse erkennen und für die Regelung der Höhenlage verwendet. Eine weitere Kamera-Option ist die optische Flusspositionierung. Dabei nimmt die Kamera Live-Bilder auf, und stabilisiert die Drohne anhand der eingelesenen Daten aus den Bildern automatisch an einer bestimmten Position. Zusätzlich bietet die Kamera eine vorteilhafte Anwendung an wie das Ansteuern auf Basis optischer Erkennungen, was bei der vorliegenden Studienarbeit eine Teilaufgabe ist.

2.1.2.2 Steuerungssysteme

Bei Multicoptern ist ein Flugcontroller erforderlich, um die Motoren mit der gewünschten Drehzahl zu steuern, die Messsysteme zu kalibrieren und mögliche Flugmodi wie Manual-, Mission-, Follow Me-Modus usw. einzustellen. Hierfür stehen viele verschiedene Flugcontroller-Typen auf dem Markt zur Verfügung, unter denen sind zum Beispiel PixHawk, ArduCopter, NAZA und MultiWii zu erwähnen [7], [8]:

- Der PixHawk-FC ist im Rahmen eines studentischen Projekts an der ETH-Zürich entwickelt worden. Dabei war das Ziel, eine Autopilot-Plattform sowohl für die wissenschaftliche und industrielle als auch für den Hobby-Bereich zur Verfügung zu stellen. Als Firmware kann der PixHawk-FC mit dem PX4- oder ArdudPilot Flight Stack betrieben werden. Die Konfigurationsprozesse und die Sensorkalibrierung können mit Hilfe der QGroundControl- oder der Mission Planner-Software durchgeführt werden.

- Der Arducopter-FC ist insbesondere für autonome Navigation von Multicoptern sehr gut geeignet. Er wird über die Bodenstation-Software Mission Planner eingerichtet, in der genaue Routen geplant werden können und ebenfalls die Ist-und Sollwerte der Winkelgeschwindigkeiten geplottet werden können.

- Der NAZA-FC von DJI verfügt hervorragende Flugeigenschaften, die maßgeblich zur Erhöhung der Flugsicherheit beitragen. Mittels DJI Assistant 2-Software können die Flugparameter und -zustände nach Bedürfnissen fein eingestellt werden.
- Der MultiWii-FC basiert auf einer Arduino-Plattform, welche aus einem Input-/ Output-Board mit einem Mikrocontroller besteht. Als Steuerungssoftware bietet sich ein Java-Programm an, welches eine Einstellung der Flug- sowie der PID-Regelparameter und eine grafische Darstellung der Ist- und Sollwerte ermöglicht.

PixHawk-Flugcontroller

Die Auswahl eines geeigneten FC hängt von den technischen Eigenschaften des jeweiligen FC bzw. von den Vorteilen, die jeder FC-Typ bei der Bedienung des Quadcopters bringen kann.
Im Rahmen dieser Studienarbeit wird der PixHawk-FC für die Flugsteuerung verwendet. Er wird als Herzstück des Quadrocopters angesehen, da er durch das Einlesen der von Sensoren erfassten Werte die aktuelle Fluglage mit der gewünschten Fluglage vergleicht und daraus die Abweichung berechnet. Dies bedeutet, dass der FC versucht, durch die Korrektur der Abweichungen per gezielter Regelung der Motorendrehzahlen, die aktuelle Fluglage an die Steuerbefehle der Fernbedienung auszugleichen.

Weiterhin werden einige Eigenschaften und Peripherien des PixHawk-FC erwähnt.

Abbildung 2.5: Pixhawk-Flugcontroller [9]

Der PixHawk-Flugcontroller besitzt einen Cortex-M4F-Prozessor mit der Erweiterung FPU (Floating Point Unit), der die Eigenschaft hat, effizient digitale Signale zu steuern. Als Sensoren sind in ihm Gyroskope, Accelerometer, Barometer und Magnetometer integriert, die die Istwerte für die bereits vorhandene PID-Regelung liefern. Für die Kommunikation mit anderen Geräten verfügt der PixHawk-FC Schnittstellen, wie beispielsweise SPI, I^2C, USB und UART. Bezüglich der Sicherung sind die Peripherie-Ausgänge vor Überströmen und die Eingänge vor elektrostatischen Entladungen geschützt (ESD-Schutz). Außerdem ist eine Leuchtdiode eingesetzt worden, die Statusinformationen für den Benutzer in Bezug auf Flugbereitschaft zeigt. Ebenfalls liefert die Safety-Switch-LED Informationen über das System [10]:

- Konstantes Blinken: System ist initialisiert.
- Wechselndes Blinken: System ist bereit, Main-Outputs sind für die Übertragung der PWM-Sollwerte aktiviert.
- Durchleuchten: Motoren sind geschaltet und können die Drohne zum Bewegen bringen.

Außerdem kann an den Pixhawk-FC ein Buzzer angeschlossen werden, welcher verschiedene Töne spielt, die eine Information über den momentanen Zustand des Autopilots liefern. Dazu gehören beispielsweise, das Starten des Systems, das Aktivieren der PWM-Outputs, das Ein- und Ausschalten der Motoren und der Auftritt von Fehlern.

2.1.2.3 Funksteuerung

Ein Funkfernsteuerungssystem kommt zum Einsatz, um die Drohne per Hand über einen Sender mittels Funksignalen aus der Ferne zu steuern. Damit kann man sowohl die Bewegung des Quadrocopters, sowie die Geschwindigkeit, Translationen und Rotationen festlegen, als auch den erforderlichen Flugmodus aktivieren. Außerdem kann ein RC-System Informationen empfangen und am Display anzeigen, wie z. B. den Akkustand. Nun zu der Kommunikation zwischen dem Flugcontroller und der Funkfernsteuerung: Der Funkempfänger wird über ein PPM-Sum-Empfänger an den Flugcontroller angeschlossen. Dieser kommuniziert mit dem in der Fernsteuereinheit vorhandenen Funksender und kann Signale mit der Frequenz 2,4 Ghz aufnehmen. Somit werden die empfangenen Signale an den Flugcontroller weitergeleitet und er bestimmt, wie die Befehle basierend auf dem aktuellen Flugmodus und der Fluglage zu interpretieren sind, und entsprechend die Motoren ansteuert. Hierbei hängt die Anzahl der Kanäle, die von dem RC-System unterstützt werden, von der Anzahl der verwendeten physikalischen Steuerungen mit denen die Quadrocopter-Bewegung festgelegt wird.

2.1.2.4 Akkumulator

Batterien kommen zum Einsatz, um ein System mit Strom zu versorgen. Hierfür sind Batterien vieler Arten zu finden, unter denen sind zum Beispiel Zink-Kohle-, Alkali-Mangan-, Zink-Luft-, Silberoxid-Zink und Lithium-Polymer-Batterie zu erwähnen. Die Auswahl eines geeigneten Batterie-Typen hängt davon ab, inwiefern er belastbar ist und sich entlädt sowie von der Anwendung bzw. von den Kriterien wie Energiebedarf und Betriebsdauer eines Gerätes. Für einen niedrigen Energieverbrauch wird zum Beispiel Zink-Kohle-Batterie eingesetzt. Dagegen werden Lipo-Polymer-Akkus bei hoher Belastbarkeit verwendet [12]. Wie aus dem Namen herzuleiten ist, besteht ein Lipo-Akku größtenteils aus Polymerfolien, welche eine elektrische Leitfähigkeit besitzen und zum Ladungstransport der Ionen dienen. Die Anode besteht aus Graphit und die Kathode aus Lithium-Metalloxid und beide befinden sich an den äußerlichen Seiten des Polymerfolienpakets. Lipo-Batterien werden oft eingebaut, da diese viele Vorteile aufweisen:

Sie sind leicht, da sie eine hohe Energiedichte bei niedrigem Gewicht besitzen. Ebenfalls sind sie einfach in der Handhabung, werden nicht von sich selbst entladen, können höhere Leistungen erreichen und liefern in einem großen Zeitrahmen einen konstanten hohen Strom. Zu den Nachteilen gehört, dass Lipo-Akkus teuer erhältlich, empfindlich bei Kurzschluss und Überladung sind und bei zu hoher Leistungsabgabe überhitzt werden. Prinzipiell ist das Ladeverfahren von Lipo-Akkus ein Konstantspannungsverfahren, dies bedeutet, dass das Ladegerät den Lipo so lädt, dass die Spannung konstant bleibt. Beim Erreichen der Ladeschlußspannung wird der Ladestrom reduziert. [11]

2.1.2.5 Elektronischer Drehzahlregler

Für eine stabile Fluglage der Drohne ist es erforderlich einen elektronischen Drehzahlregler vor jedem Motor einzubauen. Der Hauptzweck eines ESC besteht darin, den Motor zu regeln. Zudem erlaubt er dem FC die Motorgeschwindigkeit und -drehrichtung anzusteuern. Er vergleicht den Drehzahlgeschwindigkeit-Istwert des jeweiligen angeschlossenen Motors mit dem vom Flugcontroller vorgegebenen Geschwindigkeit-Sollwert und passt die Drehzahl bei Bedarf an. Dieser Prozess passiert so rapid, sodass das Flugsystem währenddessen ruhig in der Luft bleiben kann.

Wie die Abbildung 2.7 zeigt, besteht ein ESC hauptsächlich aus drei Einheiten: die Steuer-, Treiber- und Leistungseinheit. In der Steuerelektronik, Motor Control Unit (MCU), wird der Soll-/Istwert-Vergleich durchgeführt und daraus eine Stellgröße in Form eines PWM-Signals den Gate-Treibern zur Verfügung gestellt. Gate-Treiber sind elektronischer Schaltungen, die zum Einsatz kommen, um die Leistungselektronik anzusteuern. Die Leistungseinheit ist eine Halbbrücke aus Feld-Effekt-Transistoren, die durch von den Gate-Treibern bereitgestellten Ausgangsspannung in bestimmten Zeitspannen ein- und ausschalten. Das schnelle Schalten der FETs bewirkt eine präzisere Drehzahlregelung auch bei niedrigen Drehzahl.

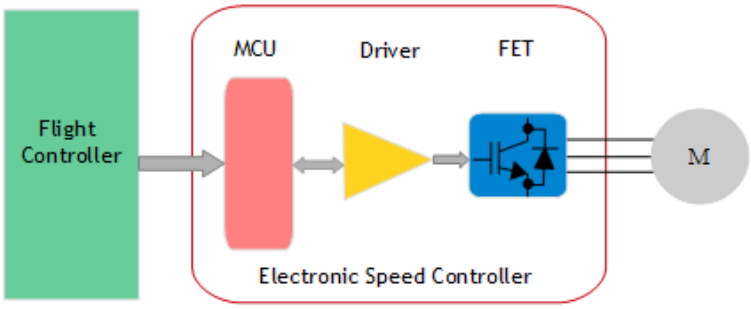


Abbildung 2.6: Aufbau eines Drehzahlreglers

2.1.2.6 Brushless Gleichstrommotoren

Der Quadrocopter wird mithilfe von bürstenlosen dreiphasigen Gleichstrommotoren zum Fliegen gebracht. Ein BLDC-Motor besteht aus einem Rotor, welcher mit Permanentmagneten realisiert wird, und einem Ständer, der drei um 120° versetzte Wicklungen umfasst. Er funktioniert nach dem Prinzip der durch die Elektronik erfolgte Kommutierung d.h. der Stromfluss übergeht von einem Zweig zum anderen, sodass immer abwechselnd eine Phase auf positive Spannung, eine Phase auf negative Spannung und eine Phase auf null V liegt. Hierbei ist der Vorteil, dass der Motor auch bei sehr niedrigen Drehzahlen betrieben werden kann und sich in passender Zeit regeln lässt, da die Kommutierung zu feldorientierter Regelung der Drehzahl und des Drehmoments dient.

2.1.2.7 Propeller

Eine sehr wichtige Komponente, welcher das Flugverhalten und die Flugzeit der Drohne stark beeinflusst, ist der verwendete Propeller. Hierbei gibt es zwei Kenngrößen bei den Propellern, die berücksichtigt werden müssen: der Durchmesser und die Steigung. Schon bei kleinerem Drehmoment benötigt ein kleiner Propeller für denselben Schub mehr Drehzahl als ein großer. Je größer ein Propeller ist, desto langsamer muss sich dieser drehen, um den gleichen Standschub zu erzeugen. Daraus lässt sich feststellen, dass ein größerer, langsam drehender Propeller effizienter ist, als der kleinere, schneller drehende. Des Weiteren spielt die Steigung eine wichtige Rolle und gibt an, wie oft sich ein Propeller pro Umdrehung nach oben bewegen kann. Grundsätzlich ist ein Propeller mit größerer Steigung eher für schnellen und dynamischen Flug passend. Jedoch steigt der Strombedarf an aufgrund der möglich erreichten höheren Fluggeschwindigkeiten. Ein weiterer Faktor, der nicht zu vernachlässigen ist, ist das Gewicht der Propeller. Dies wirkt auf das gesamte Massenträgheitsmoment des Quadcopters aus. Deshalb sind schwere Propeller für bestimmte Anwendungen schlecht geeignet. [13]

Bei der Montage der Propeller ist auf die Drehrichtung zu achten. Diese ist sowohl an den Motoren, als auch an den Propellern schon gekennzeichnet. Insgesamt sind zwei Propeller mit Rechtsgewinde und zwei mit Linksgewinde. Jeder Propeller wird auf die Achse des dazu gehörigen Motors aufgesteckt und gedreht bis er ganz befestigt ist. Ein falsches Anbauen der Propeller bzw. ein Anbauen von beschädigten Propellern kann zu einem instabilen Flugverhalten sowie zum Absturz der Drohne führen und möglicherweise durch Überschlagen Schaden verursachen.

2.1.2.8 Rahmen

Der Rahmen ist ein Hauptbestandteil eines Quadcopters, welcher genug Raum für die Platzierung der erforderlichen Bauteile verfügen soll. Für optimale Kabelbefestigungen

eignen sich Trägerplatten mit mehreren Bohrungen. Ebenfalls ist es vorteilhaft, wenn der Rahmen mit unterschiedlich farbigen Armen gestattet ist, wodurch die Flugrichtung des Quadcopters besser einschätzt werden kann und die Motoren mit der gleichen Drehrichtung einfacher erkannt werden können. Im Rahmen des Studienprojekts kommt der DJI F450 Flame Wheel-Rahmen zum Einsatz. Dieser ist aus robusten Materialien gefertigt worden und weist genügend Stabilität bei einem Absturz auf. Auf der Trägerplatte im Herzen des Rahmens bietet sich an, den Autopilot sowie das GPS-Modul zu befestigen. Außerdem entfällt eine komplizierte Verkablung, da der Rahmen genug Raum verfügt, um eine Lipo-Batterie auf die Grundplatte zu platzieren, an der die Stromversorgung und die Anschlüsse der Geschwindigkeitsregler verlötet werden.

2.1.3 Regelung des Quadcopters

Der PID-Regler wird universell in Systemen am meisten eingesetzt. Er versucht die Geschwindigkeit sowie die Position des Quadcopters auf den vorgegebenen Sollwert auszuregeln. Um dies zu ermöglichen, braucht er die aktuellen Istwerte in Form von Messwerten, die zurückgeführt werden müssen, um einen Vergleich durchzuführen, wie in Abbildung 2.7 dargestellt ist. Sind Sollwerte und Istwerte identisch, so ist eine Korrektur nicht nötig. Kommt eine positive oder negative Regelabweichung zustande, so ändert sich die Ausgangsfrequenz am Umrichter, bis der Sollwert und der Rückführungswert übereinstimmen.

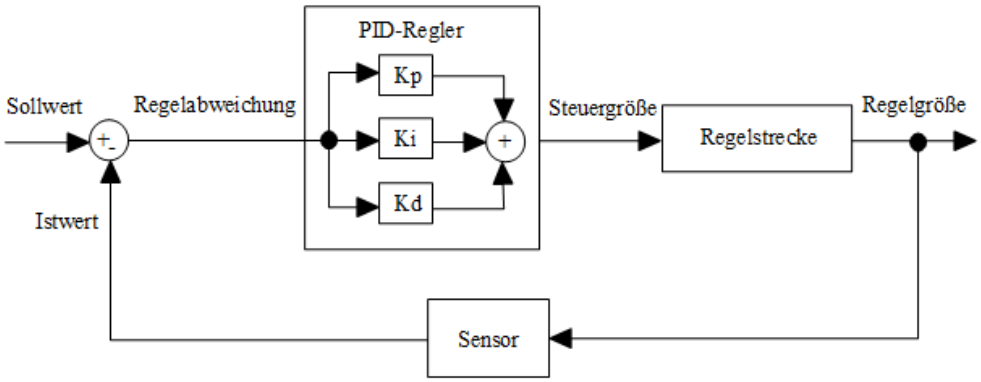


Abbildung 2.7: PID-Regelkreis

Die Übertragungsfunktion des PID-Reglers im Laplace-Bereich [3] lautet:

$$G_R(s) = K_P + K_I \cdot \frac{1}{s} + K_D \cdot s \tag{2.3}$$

mit $K_I = \frac{K_P}{T_N}$ und $K_D = K_P \cdot T_V$, wobei K_P der Proportionalwirkende Regler, K_I der Integralwirkende Regler und K_D der Differenzialwirkende Regler, T_N die Nachstellzeit und T_V die Vorhaltezeit ist.

Wirkung der einzelnen Anteile der PID-Regelung auf das Flugverhalten:

P-Anteil:

Der proportionale Anteil hat einen Einfluss nur auf die momentane Differenz zwischen dem Ist- und dem Sollwert. Er ist somit für eine rapide Reaktion verantwortlich und sollte daher so hoch wie möglich eingestellt werden, ohne dabei Schwingungen entstehen zu lassen, denn bei hohem P-Anteil sind die Schwingungen hochfrequent, während bei zu niedriger P-Verstärkung die Reaktion auf die Eingabeänderungen sich verlangsamt.

I-Anteil:

Durch den Integrator wird der Sollwert erreicht. Er integriert die Regelabweichung und wird größer, wenn die gewünschte Rate nach längerer Zeit nicht erreicht wird. Dieser sollte jedoch nicht zu hoch eingestellt werden, da sonst die Schwingungen niederfrequent werden.

D-Anteil:

Der Differenzierer bewertet die Veränderung der Regelabweichung und reagiert auf deren Änderungsgeschwindigkeit. Das bedeutet, dass er die Regelabweichung kurzzeitig verstärken und damit die Reaktion des Reglers beschleunigen kann. Hierbei wird bei zu hohem D-Anteil das Rauschen verstärkt, was dazu führen kann, dass die Motoren heiß werden können.

2.2 Robot Operating System

Bei der autonomen Steuerung eines Roboters sind vielerlei erforderliche Herausforderungen zu bewältigen. So sollte ein Roboter die Fähigkeit besitzen, durch die Erfassung seiner Umgebung und die ununterbrochene Verarbeitung der Messdaten ohne den menschlichen Eingriff im Echtzeit präzis zu reagieren, seine Bewegungen selber zu steuern, sich zu positionieren, seinen Weg zu planen, sich selber zu kontrollieren und bei auftretenden Störungen sich selber zu regeln. Um Dies all zu realisieren, kommt das Robot Operating System in komplexen Systemen zum Einsatz, welches als ein komfortables Framework angesehen wird, da dieses bereits Basisprogramme und zahlreichen nützliche Funktionalitäten verfügt. Somit werden der Aufwand und die Zeit gespart und es lassen sich Programme einfach darauf aufbauen. Darüber hinaus besitzt ROS weitere Vorteile und Dienste, die im folgenden kleinen Exkurs in der ROS-Welt präsentiert und verdeutlicht werden: ROS ist kein Betriebssystem, sondern ein Set von Bibliotheken und Tools und läuft unter Ubuntu mit der MATE-Oberfläche, das für RPI 3 angepasst ist und auf dem eigentlichen Betriebssystem Linux basiert ist. Es bietet ganz viele Möglichkeiten in Form von fertigen Lösungen (wie beispielsweise Pfadplanung, Ermittlung der Position durch Umrechnungen mit Achsen, usw.), Simulationen mit 3D-Visualisierungen und Software, um was auf einem Roboter zu realisieren. ROS stellt Anwendungen und Dienste zur Verfügung, die man von einem Standard-Betriebssystem erwarten würde:

- Hardware-Abstraktion
- Gerätetreiber
- Implementierung von viel genutzten Funktionalitäten
- Inter-Prozess-Kommunikation
- Paket-Management

Fernerhin besteht ROS aus ganz vielen fertigen Kommandos in der Konsole jedoch auch aus selbst geschriebenen Programmen. Hierbei ist roscore eine Sammlung von Knoten und Programmen. Um den ROS-Master auszuführen, wird der wichtigste Befehl roscore gestartet. Der ROS-Master ermöglicht es die Knoten untereinander (peer-to-peer) zu erkennen und miteinander zu kommunizieren. Ein Programm wird in ROS als Knoten bezeichnet. Ein Knoten ist ein Skript, das in Python oder C++ programmiert wird. Diese Knoten kommunizieren zusammen über den TCP/IP-Protokoll. Dabei werden mehrere Knoten miteinander verbunden, was eine Kommunikation einer Ebene mit der anderen synchron und asynchron und eine Überwachung eines Roboters aus jeder Ebene ermöglicht. Außerdem enthält ROS Pakete wie zum Beispiel für Laserscanner, Kamera, Fernsteuerung, usw., die bereits fertig programmierte Knoten beinhalten. Über ein Topic, das als ein Informationsbus arbeitet, können Nachrichten identifiziert

werden. Somit werden Informationen aus einem oder mehreren Knoten auf ein Topic veröffentlicht (Publication) und dann von einem oder mehreren Knoten abonniert (Subscription). [14]

Beispiel zum Publish-Subscribe-Verfahren: Während ein Roboter fährt, wird sicherheitshalber erwünscht eine Information über den Batteriezustand ständig zu erfahren. In dem Roboter sind bereits ein Einplatinencomputer und ein A/D-Wandler eingebaut. So läuft über den Einplatinencomputer beispielsweise Raspberry PI ein Knoten namens Batterie_Publisher, welcher mittels des A/D-Wandlers die Batteriespannung ausliest. Der Batterie_Pubisher veröffentlicht ohne Beauftragung und ohne Unterbrechung den Spannungswert als Nachricht mit dem Thema Spannung. Auf dem Raspberry PI läuft ebenso ein weiterer Knote Batterie_Subscriber, der das Thema Spannung abonniert und den Spannungswert andauernd anzeigt.

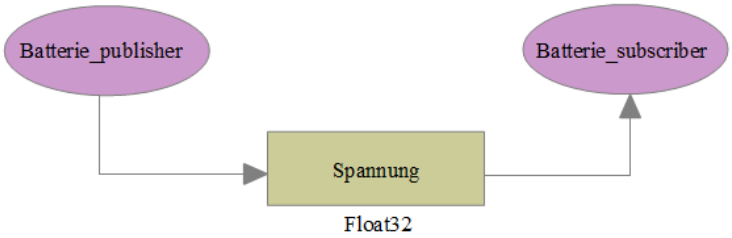


Abbildung 2.8: Überwachung der Akku-Spannung

Neben dem Publish-Subscribe-Verfahren, bei dem der Informationsaustausch unaufgefordert und andauernd stattfindet, gibt es ein zweites Verfahren, das über Services läuft. Dabei wird ein Service von einem Knoten nach etwas gefragt oder mit etwas beauftragt und er reagiert darauf und antwortet.

Beispiel zum Service-Verfahren: Auf dem gleichen obenerwähnten Roboter läuft über den Raspberry PI ein Knoten Motor_Client, welcher Anforderung zum Anschalten des Motors an den Motor_Server schickt. Der Motor_Server erhält die Einschalten-Anfrage mit dem booleschen Wert true. Nach dem Einschalten des Motors antwortet er dem Motor_Client mit dem booleschen Wert success.

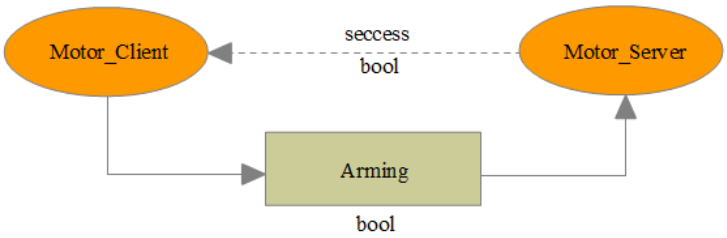


Abbildung 2.9: Einschalten eines Motors

Um mit ROS zu arbeiten, muss zunächst der sogenannte catkin-Workspace erzeugt werden. Zusätzlich ist es möglich im catkin-Arbeitsbereich Pakete zu erstellen. Ein ROS-Paket enthält C++/Python-Programme als Knoten, Bibliotheken, Konfigurations-, Message- und Server-Dateien, wobei in ihm auch Knoten erzeugt werden können. Somit stellt es sich als ein erforderlicher Baustein dar. Der Zweck eines Pakets ist, eine nützliche Funktionalität und eine einfache Verwendung zu erzielen, so dass die Software wiederverwendbar ist.

2.2.1 ROS-Pakete

Bei der vorliegenden Studienarbeit werden insgesamt sechs ROS-Pakete benötigt.

Bildverarbeitung

Zunächst muss das *usb_cam*-Paket für das Treiben der für die Detektierung verwendeten USB-Kamera installiert werden. Dies ermöglicht den Datenaustausch zwischen dem RPI 3 und der Kamera über die USB-Schnittstelle. Weiterhin wird das ROS-Paket *camera_calibration*, welches eine einfache Kalibrierung mit einem Schachbrett-Kalibrierungsziel ermöglicht, und das ROS-Paket *image_proc* für die Filterung der Abbildungsfehler benötigt.

Apriltag-Detektierung

AprilTags können beispielsweise bei Robotern, Objekten, Menschen, Tieren usw. eingesetzt werden, um diese unter Kontrolle zu halten. Dies geschieht durch die Erkennung der Marker sowie durch die Berechnung der relativen Posen zwischen dem Kamerarahmen und dem äußeren schwarzen Rand des erkannten Tags. Die Pose wird aus den vier Eckpunkten, Kameramatrix und bekannter Markergröße ermittelt.

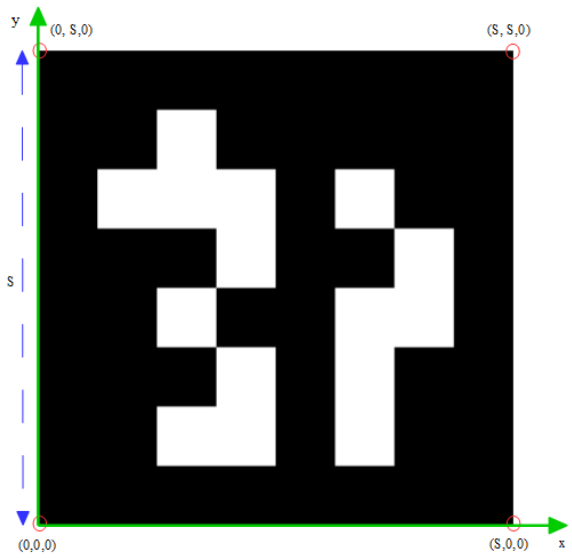


Abbildung 2.10: AprilTag, ID:12 [15]

Außerdem dienen Apriltags zu Steuerung von Robotern zum Beispiel, wenn anhand der erkannten Daten bestimmte translatorische oder rotatorische Bewegungen realisiert werden sollen oder gewisse Distanz gehalten werden soll. Ebenfalls können Apriltags auf einen großen Abstand erkannt werden, weil sie eine niedrige Auflösung besitzen. Zusätzlich ist die durchgängige weiß-zu-schwarz-Kante um jeden AprilTag bei exakter Posenschätzung von Vorteil. Im Folgenden wird das ROS-Paket *apriltags2_ros* verwendet. Dieses Paket kann viele Herausforderungen im Bezug auf die Wahrnehmung von Robotersystemen vereinfachen durch die Verwendung von Referenzmarken als Bezugspunkte. Es bietet also vollständigen Zugriff auf die Anpassungen von Apriltags2-Kernalgorithmus und stellt das Erkennungsbild und Position der erkannten Tags für ROS-Topics verfügbar.

Kommunikation zwischen RPI 3 und PixHawk-Autopilot

Damit die empfangene und gesendete Informationen von beiden Seiten, von dem RPI3 und von dem Flugcontroller, "verstanden" werden, werden das MAVROS- und das MAVLink-Paket mit Hilfe der Anleitung auf der Dronecode-Internetseite [39] auf den RPI 3 heruntergeladen. Das Mavros-Paket ermöglicht eine Konvertierung zwischen den ROS-Topics und den MAVLink-Nachrichten, sodass der MAVLink-fähige Autopilot mit ROS kommunizieren kann. Zusätzlich bietet das Mavros-Paket eine UDP-MAVLink-Bridge für Bodenkontrollstationen.

3 Aufbau der Hardware

In diesem Kapitel wird einen Überblick über den gesamten Aufbau des Quadcopter-systems verschaffen. Zunächst wird erklärt, wie zwei Boxen angefertigt werden, in denen einige Komponenten platziert werden sollen. Anschließend folgt eine ausführliche Beschreibung über den Anschluss der Bauteilen an den PixHawk-FC. Zum Anschließen sind folgende Bauteile notwendig: der PixHawk-FC, die Versorgungsleitungen, das SPEKTRUM-Funkempfängersystem, die DJI E300 15 A OPTO-ECSs, die DJI E300 2212/920 KVMotoren, die DJI 9450-Propeller, der RPI 3 Model B+ und die Logiteck-USB-Kamera.

3.1 3D-Modellierung der PixHawk- und RC-Empfänger-Box

Um die Komponenten vorteilhaft zu befestigen und miteinander zu verbinden, werden mithilfe des 3D-CAD-Programms Solidworks zwei Boxen modelliert. Die erste Box (Abbildung 3.1) hat die gleiche Form wie die der oberen Basis-Platte der Flame Wheel F450 DJI-Trägerplatte. Hierbei müssen die Maße, wie die Dicke und die Höhe der Box vorteilhaft gewählt werden, sodass die benötigten Komponenten reinpassen. Zudem darf das Gewicht nicht zu schwer werden, weil es sonst einen zu hohen Bedarf an Strom für das Fliegen des Quadrocopters benötigen würde. In der Box befinden sich der Autopilot, der I^2C-Splitter, der Buzzer und der Sicherheitsknopf. Ebenfalls werden die gesamten Leitungen zu den ESCs, dem PPM-Empfänger, dem Leistungsmodul, dem Raspberry pi 3 und dem GPS-Modul durch die an der Box gefertigten Bohrungen ausgeführt.

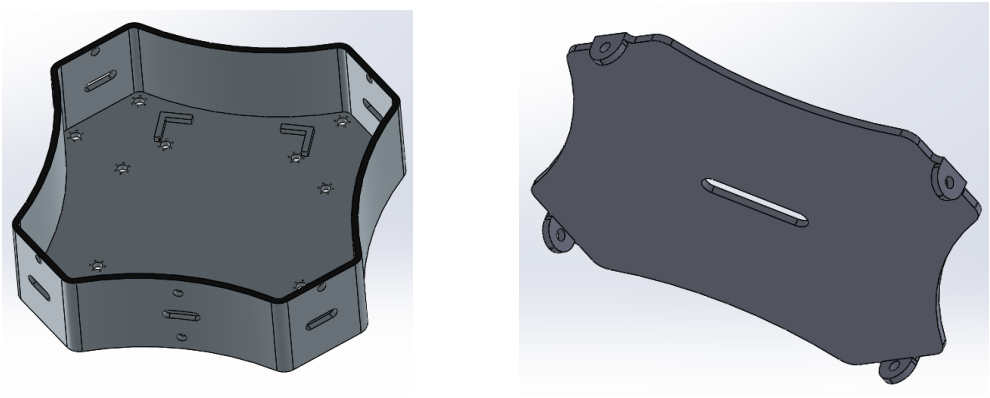


Abbildung 3.1: Autopilot-Box

Die zweite Box (Abbildung 3.2) wird seitlich auf der unteren Basis-Platte des DJI Wheel-Rahmens befestigt und umfasst den Empfängerteil des RC-Systems. In der Box befindet sich der PPM-Empfänger. An der rechteckigen Öffnung sollen der AR6200-Empfänger und der Satellitenempfänger befestigt werden, wobei die Antennen nach draußen zeigen sollen.

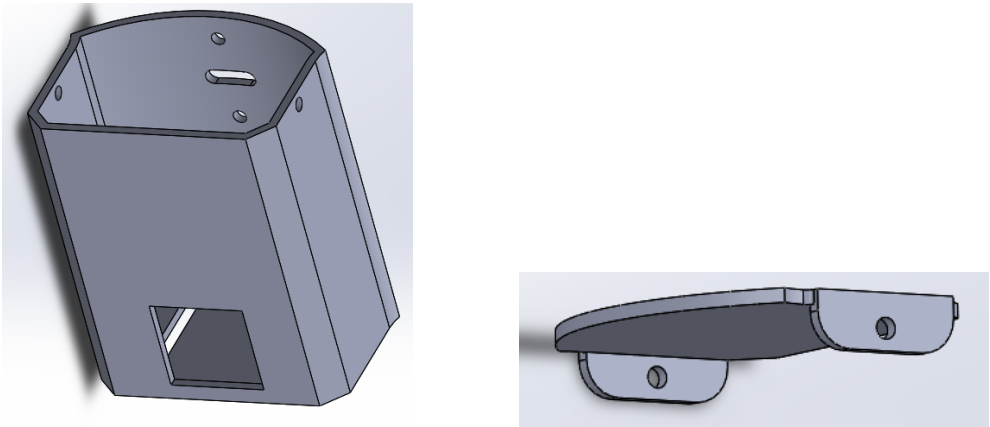


Abbildung 3.2: RC-Empfänger-Box

3.2 Verkabelung des DJI F450 Quadrocopters

Als Unterstützung bietet die Dronecode-PX4-Internetseite [16] eine ausführliche Erklärung zur Verkabelung der Komponenten mit dem PixHawk-FC. Somit kann der Aufbau anhand des folgenden Verkabelungsschema verdeutlicht werden:

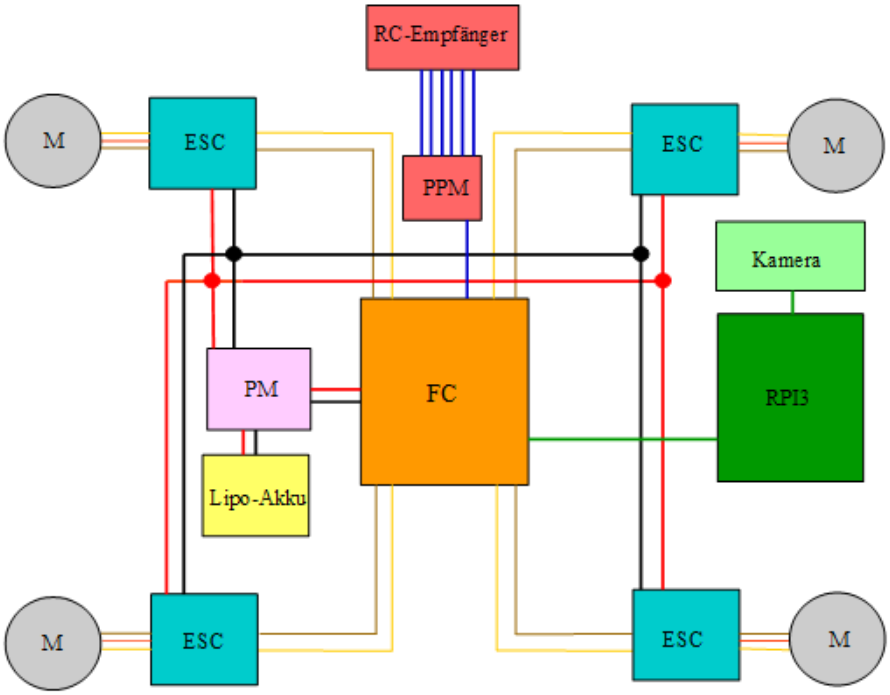


Abbildung 3.3: Aufbau des Quadcopter-Systems

Damit die Äquilibrierung des Quadrocopters bewahrt wird, sowie eine hohe Wendigkeit im Flugverhalten gesorgt wird, befinden sich die meisten Massen wie FC, Akku, Sensoren, usw. in seinem Zentrum. Daraus ergeben sich kleine Trägheitsmomente hinsichtlich der Rotationsachsen, die zu einer hohen Winkelbeschleunigung um diese Achsen führen und somit eine schnelle Lageänderung ermöglichen.

3.2.1 Positionierung des Flugcontrollers

Für eine geschickte Verkabelung, ein Gleichgewicht beim Fliegen sowie eine einfache Erkennung der Flugrichtung wird der Flugcontroller im Zentrum der Box auf der Trägerplatte des DJI F450 Flame-Wheel-Rahmens positioniert.

3.2.2 Anschluss der Versorgungsleitung

Über das Leistungsmodul wird dem ganzen System eine Spannungsversorgung bereitgestellt. Zunächst wird der PM-Ausgang mit einem sechsadrigen Kabel mit dem POWER-Port des PixHawk-FC verbunden. Weiterhin wird der PM-Hauptausgang über XT60-Stecken an der Verteilerplatine verlötet, um die ESCs mit Spannung zu versorgen. Anschließend wird der PM-Eingang über XT60-Stecker an die LiPo-Batterie angeschlossen.

3.2.3 Anschließen des Funkempfängers

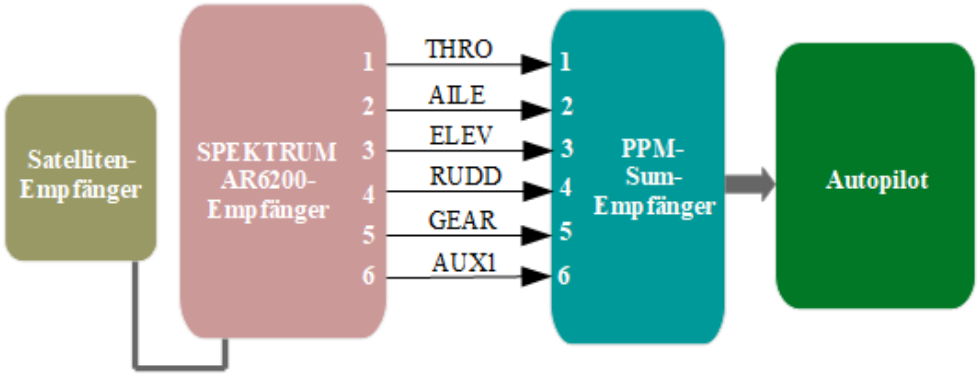


Abbildung 3.4: Blockschaltbild des Funkempfängersystems

Als Erstes wird der PPM-Sum-Empfänger Stecker an den RC Pin-Input des PixHawk-FC mittels eines Servosteckers angeschlossen. Der PPM-Sum-Empfänger hat die Aufgabe, die PWM-Signale der einzelnen Kanäle in ein PPM-Signal zu wandeln. Hierbei werden die PWM-Signale nicht gleichzeitig, sondern nacheinander übertragen. Dadurch wird ein PPM Signal als eine Reihe von PWM-Signalen erzeugt und an den PixHawk-FC weitergeleitet. Das bedeutet, dass für mehrere Kanäle (sechs Kanäle: AUX1, GEAR, RUDD, ELEV, AILE und THRO) nur ein Kabel benötigt wird. Man braucht also nur einen Servostecker mit drei Leitungen für die Spannungsversorgung, die Masse und die Signalübertragung. Als Nächstes wird der SPEKTRUM-AR6200-Empfänger mit dem PPM-Sum-Empfänger verbunden. Die aus dem PPM- Sum-Empfänger ausgeführten Leitungen werden an den SPEKTRUM-AR6200-Empfänger eingesteckt, da dessen Anschlüsse mit den Kanalnummern versehen sind. Um den Empfang der Signale zu verstärken, wird zusätzlich ein Satellitenempfänger an den AR6200-Empfänger ange-

schlossen. Nun können die von der DX6i-SPEKTRUM-Funke gesendeten Signale an dem Autopilot ankommen. Einstellungen im Dx6i-Sender sind im Anhang zu finden.

3.2.4 Anschluss der DJI E300 15A OPTO-ESCs und 2212/920 KV-Motoren

Ein ECS ist ein relativ kleiner Bauteil, der als Verbindungsstück zwischen dem Motor, der Batterie und dem PixHawk-FC dient. Um die Motorphasen mit Spannung zu versorgen, werden die aus dem E300-ECS ausgeführten Phasenleitungen über Rundstecker mit den Motorphasen verbunden. Das dreiadrige Kabel des Geschwindigkeitsreglers wird mittels eines Servosteckers an einen für den Motor gestellten FC-Hauptausgang (Hauptausgänge, die mit 1, 2, 3 und 4 gekennzeichnet sind)angeschlossen. Da dem E300-ESC intern 5 V als Spannungsversorgung für die Gate-Treiber bereits zur Verfügung gestellt wird, werden nur die Leitung für die Signalübertragung und die GND-Leitung an den PixHawk-FC angeschlossen. Die Strom- und Spannungsversorgung für die ESC-Leistungselektronik wird über das Leistungsmodul aus dem Lipo-Akku gewonnen.

Über Rundverbinder werden jeweils drei Leitungen aus einem Geschwindigkeitsregler mit den drei Phasen des jeweiligen BLDC-Motors verbunden, um die drei Wicklungen mit Spannung zu versorgen und die Drehzahl zu regeln.

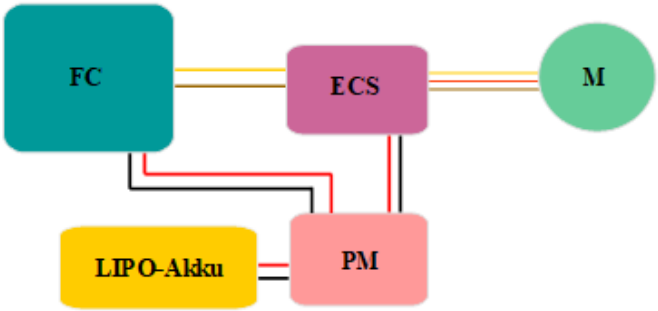


Abbildung 3.5: FC-ESC-Motor-Verbindung

3.2.5 Anschluss des RPI 3 Model B+

Da der Datenaustausch über die GPIO-Pins 14 und 15 des RPI 3 erfolgt, werden drei Jumper-Leitungen für das empfangene Signal, das gesendete Signal und GND benötigt. Eine Jumper-Leitung für die Spannungsversorgung wird in der Regel nicht gebraucht, da der PixHawk-FC bereits über den POWER-Port mit Spannung von dem Leistungsmodul betrieben wird. Die drei Jumper-Leitungen werden an den drei TELEM2-Stecker-Leitungen angelötet und jeweils mit Schrumpfschlauch isoliert. Im Fall der Kommunikationsüberprüfung durch eine Software-Übertragung bei nicht geschalteten Motoren wird, wie in der Abbildung 3.6 zu sehen ist, die 5 V-Leitung an den RPI 3-Pin angeschlossen, um den Autopilot mit Spannung zu versorgen.

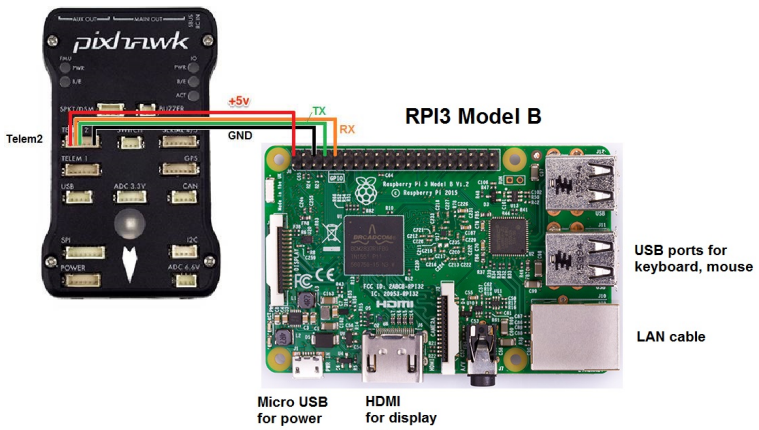


Abbildung 3.6: RPI3-PixHawk-Verbindung [17]

3.2.6 Anschließen der Kamera

Die Kamera kommuniziert mit dem Autopilot über den RPI 3, an den sie über eine USB-Schnittstelle angeschlossen ist. Der Kopf der Kamera wird am DJI F450 Flame Wheel-Rahmen befestigt.

4 Betrieb des PixHawk-Flugcontrollers mit der PX4-Firmware

Als Ground Station bieten sich verschiedene Softwareanwendungen an, die dazu dienen einen Autopilot einzustellen. Im Rahmen dieser Studienarbeit werden zwei Bodenstation-Softwares verwendet. QGroundControl ist auf der Dronecode-Seite [18] als Open Source zu finden und kann unter merhreren Betriebssystemen laufen. Um einen Vergleich zu ziehen, wird daraufhin der FC über den Mission Planner als weitere Software eingerichtet. Dieser steht auf der Mission Planner Home-Seite [19] zum Installieren zur Verfügung. Die Konfigurationsprozesse am Pixhawk-FC können entweder über die PX4- oder die Ardupilot-Firmware durchgeführt werden. In diesem Fall wird die PX4-Firmware verwendet. Zum Schluss wird ein Vergleich zwischen der QGC- und der MP-Software angestellt.

4.1 Einrichten des FC über den MP

4.1.1 Auswahl des Fluggerätetyps und Sensorkalibrierung

Da es sich bei dem verwendeten DJI F450 Flame Wheel-Typ von vier Rotoren handelt, wird Quad als Rahmenklasse eingestellt. Weiterhin wird die X-Rahmenform festgelegt.

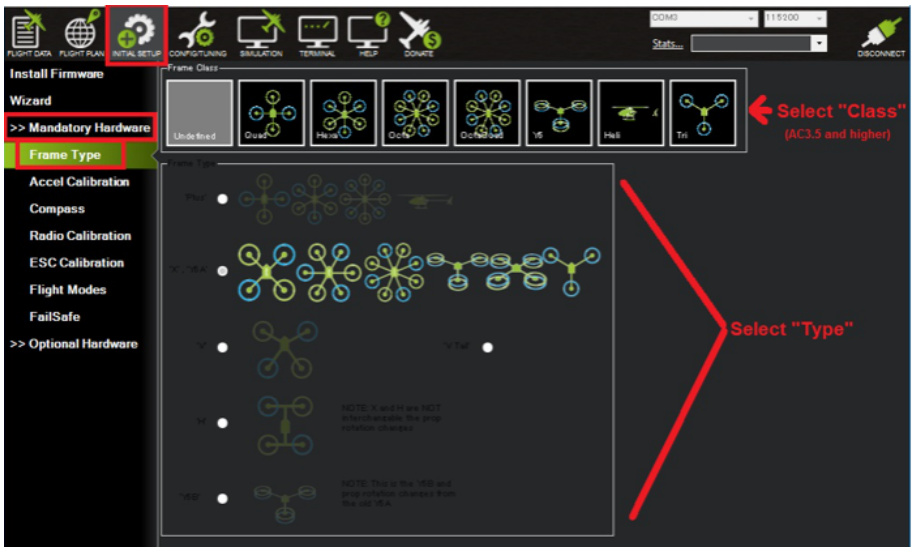


Abbildung 4.1: Modell-Konfiguration [28]

ACC:

Als Nächstes wird der ACC kalibriert. Der Autopilot wird in den vorgegebenen Ausrichtungen positioniert und dabei ruhig gehalten. Sehr wichtig ist den Level Horizon festzulegen. Denn diese Position entspricht der horizontalen Haltung, die der FC als Level beim Fliegen wahrnimmt.

Abbildung 4.2: ACC-Kalibrierung [29]

Kompass:

Für die Kompasskalibrierung wird bei MP der Parameter AHRS-ORIENTATION auf custom eingestellt. Die Setzung dieses Parameters erlaubt die Verwendung der Parameter, die zum Setzen des Roll-, Pitch- und Yaw-Versatzes der Autopilot-Einbaulage dienen. Hierbei ist der Rotationsversatz auf Null gesetzt. Mit einem Click auf Start werden nun die Offsets berechnet, indem der Autopilot gehalten und so gedreht wird, dass jede seiner Seiten, vorne, hinten, links, rechts, oben und unten eine Weile nach unten Richtung Erde zeigt. Im weiteren Verlauf muss der Autopilot neu gestartet werden, bevor mit den restlichen Einstellungen fortgefahren wird.

Abbildung 4.3: Kompasskalibrierung [30]

4.1.2 Kalibrierung des RC-Senders

Der Kalibrierungsprozess der Funke-Steuerungen ist unkompliziert. Es geht um das Festlegen der maximalen und minimalen PWM-Wert der einzelnen Kanäle, indem die Sticks und den GEAR-Schalter über den gesamten Bereich, von der minimalen bis zur maximalen Position bewegt werden.

Abbildung 4.4: Sender-Kalibrierung [31]

Die Kanalzuordnung kann entweder in MP oder am PPM-Sum-Empfänger geändert werden. In diesem Fall wurde die Kanalzuordnung durch das Umstecken der Roll-, Pitch- und Throttle-Kanalleitung am PPM-Sum-Empfänger wie folgend definiert:

Tabelle 4.1: Steuerungskanäle gemäß MP

Kanal	Signal
1	Roll
2	Pitch
3	Throttle
4	Yaw
5	Gear

Im Folgenden werden die benötigten Flugmodi eingestellt. Analog wie in QGC wird über den fünften Kanal, den GEAR-Schalter am RC-Sender, zwischen den Flug-Modi gewechselt. Der einzige Unterschied liegt daran, dass der Offboard-Modus in QGC dem Guided_nogps -Modus in MP und der Manual-Modus dem Stabilize-Modus entspricht.

Tabelle 4.2: Flugmodi in MP

Gear	ELEV D/R	Flugmodus
0	0	Land
0	1	Guided_nogps
1	0	Stabilize
1	1	Stabilize

4.1.3 Motor- und ESC-Einstellung

Anschließend werden die Main-Outputs aktiviert, die an die ESCs angeschlossen werden. Wie auf den Abbildungen 4.14 und 4.15 zu sehen ist, wird bei der Verwendung der

X-Konfiguration jedem Output ein Motor mit der gleichen Nummerierung zugeordnet, wobei die Steuerung für den jeweiligen Motor in Abhängigkeit von der Fluglage bereits vordefiniert ist.

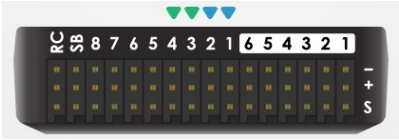


Abbildung 4.5: Hauptausgänge des PixHawk-Autopilots [32]

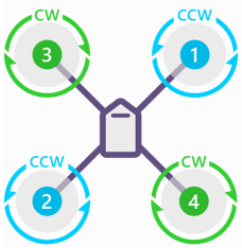


Abbildung 4.6: Reihenfolge der Motoren [33]

Zudem werden der minimale und der maximale Standard-PWM-Wert, 1000 μs und 2000 μs, festgelegt. Damit die Motoren nach dem Arming nicht direkt anlaufen, wenn der Gasknüppel noch in der Nullposition ist, werden ebenso der Standard-PWM_MIN- und -PWM_MAX-Wert bei den ESCs eingestellt.

4.1.4 Motortest über MP

Bevor der Autopilot in Betrieb genommen wird, bietet sich in dem MP die Reihenfolge und die Drehrichtung der einzelnen Motoren zu überprüfen.

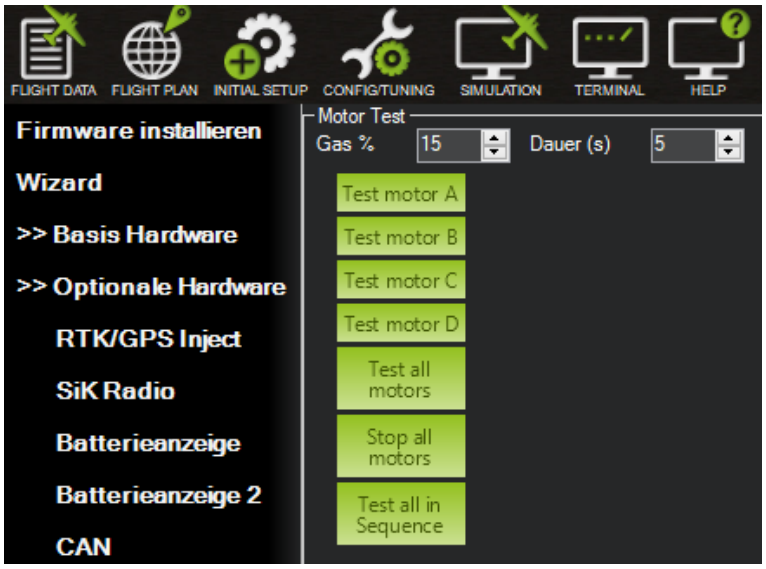


Abbildung 4.7: Motortest in MP [34]

Der PixHawk-FC wird mit dem MP via MAVLink verbunden. Daraufhin wird eine Anlaufzeit von 5 s sowie ein prozentuale Gaswert beginnend ab 5 Prozent eingestellt. Falls die Motoren bei diesem Gaswert nicht anlaufen, wird mit 10 Prozent Gas versucht. Wenn die Motoren immer noch nicht anlaufen, wird der Gas weiter um 5 Prozent

erhöht. Die Zuordnung der Motoren der X-Konfiguration wird wie folgend definiert:

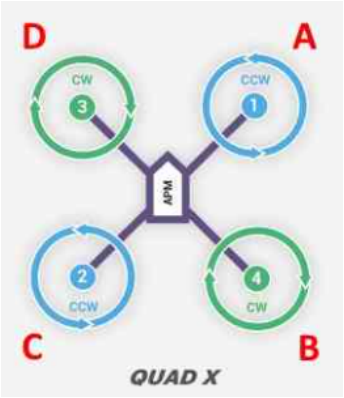


Abbildung 4.8: Reihenfolge der Motoren [35]

Dabei können die Motoren einzelnen, nacheinander oder gleichzeitig getestet werden. Als Ergebnis hat sich festgestellt, dass die Motoren erst ab 15 Prozent anlaufen, und dass sie schon in der richtigen Reihenfolge an den Autopilot angeschlossen sind.

4.1.5 PID-Tuning über MP

Für jeden Winkel steht in dem FC eine Regelung zur Verfügung. Jede Regelung besteht aus zwei Subreglern: Der übergeordnete Regler ist der Lageregler, der unter Verwendung einer P-Regelung den Winkelfehler in eine gewünschte Winkelrate umwandelt und an den untergeordneten Ratenregler als Sollwert weitergibt. Weiterhin wird die Regelabweichung zwischen dem Soll- und dem Istwert der Drehrate von der PID-Regelung in einem Motorbefehl mit hohem Pegel umgewandelt.

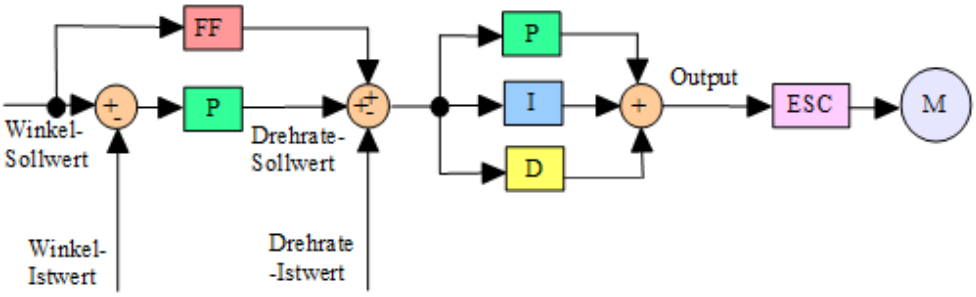


Abbildung 4.9: Attitude Control

Durch diese Regelungen wird der Drohne ein stabiles Flugverhalten gewährleistet, sodass sie auch beim Auftauchen von Störgrößen, wie beispielsweise bei starkem Wind, einen guten Schwebeflug aufweist. Dafür müssen geeignete Regelparameter eingestellt werden. Damit das PID-Tuning gelingt, sind einige Schritte durchzuführen. Als Erstes ist erwähnenswert, dass der Quadcopter so getunt werden muss, wie man ihn später

fliegen lassen will. Das heißt, beim Tunen soll der mit den benötigten Bauteilen ausgestattete Quadcopter so gehängt werden, dass er so genau wie möglich ausbalanciert ist und der Massenschwerpunkt im Zentrum der drei Achsen liegt. Ebenfalls sollen nach dem Tunen keine Änderungen am Quadcopter vorgenommen werden, wie zum Beispiel das Ersetzen der bisherigen Komponenten durch Komponente mit anderen Eigenschaften. Somit können die Reglerreaktionen effektiv umgesetzt werden. Darüber hinaus ist der Autopilot für das Tuning-Verfahren auf den Stabilize-Modus zu setzten. Da der Quadcopter symmetrisch ist, wird mit dem Rollregler und dem Pitchregler angefangen. Zunächst wird der P-Anteil langsam hochgefahren, dann der D-Anteil und eventuell der I-Anteil, bis jeder Regler den Arbeitspunkt erreicht, an dem er optimal arbeitet.
Hierzu wurde ein Versuch aufgebaut, bei dem der Quadcopter mit einem Spielraum befestigt wird, sodass er sich noch frei bewegen kann. Dies ermöglicht eine bessere Analyse des Flugverhaltens.

Abbildung 4.10: Versuchsaufbau für das PID-Tuning

Für präzisere Bestimmung des Regler-Arbeitspunkts werden zusätzlich bei jedem Einstellungsschritt die Ist- und Sollwerte der Rotationsbewegungen im MP grafisch dargestellt. Mit Hilfe der Plots wird ermittelt, ob und ab welchem Parameterwert Oszillationen zustande kommen.

Einstellung der Regelparameter für Roll/Pitch:

Bei der Winkelregelung hat sich der Wert P = 4,5 als optimaler Wert ergeben, welcher bei dem Tuning des Roll-/Pitch-Drehratenreglers verwendet wird. Dabei werden außerdem die Standardwerte sowohl des P- als auch des PID-Reglers für die Yaw-Bewegung verwendet. Die Regler-Werte bei der Pitch- und Roll-Bewegung sind identisch. Wie anhand der nachfolgenden Plots zu sehen ist, wird zunächst der P-Anteil hochgefahren bis Oszillationen erzeugt werden, dann wird er wieder reduziert bis zu dem optimalen Wert (Abbildungen 4.19, 4.20 und 4.21), wobei der I- und D-Anteil ausgeschaltet sind. Auf der gleichen Art und Weise wird als Nächstes der D-Anteil (Abbildungen 4.22, 4.23 und 4.24) und abschließend der I-Anteil (Abbildungen 4.25, 4.26 und 4.27) eingestellt.

P-Anteil:

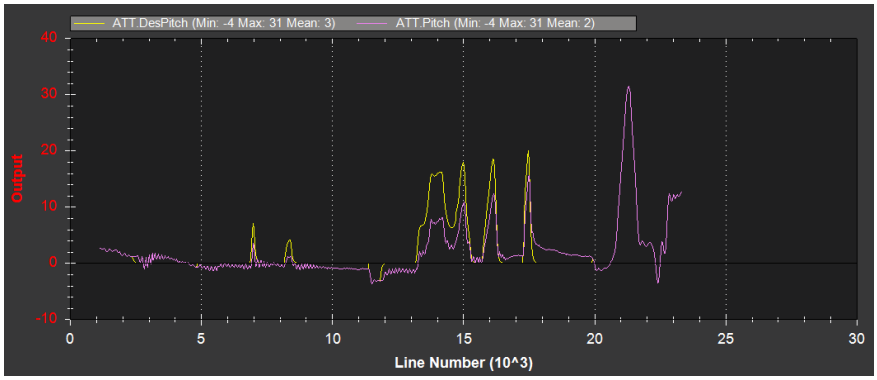


Abbildung 4.11: Anfangswert P = 0,13

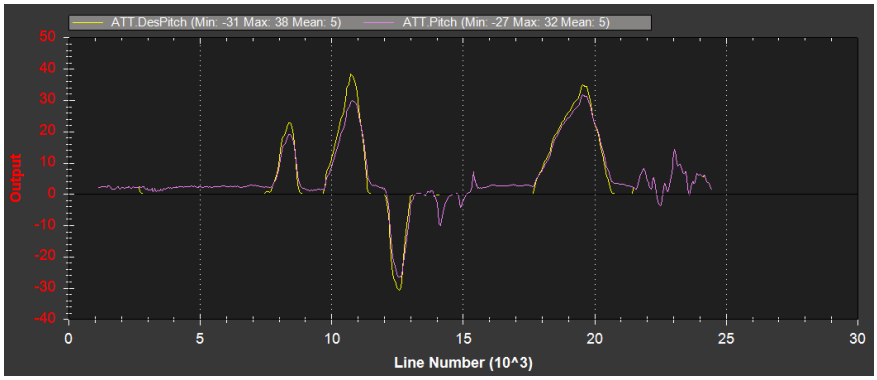


Abbildung 4.12: Optimaler Wert P = 0,18

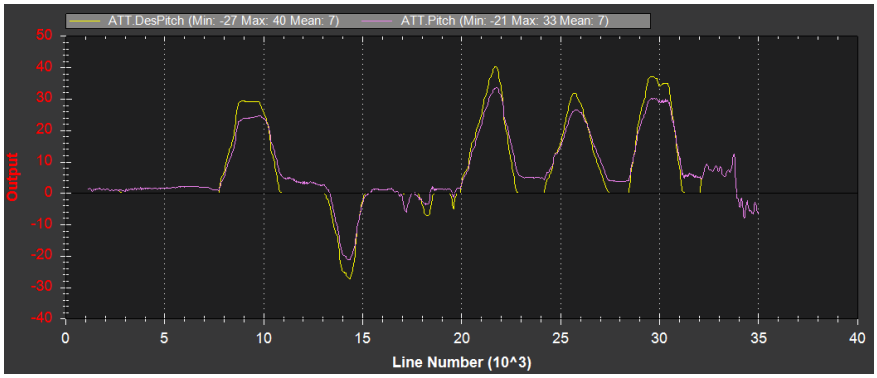


Abbildung 4.13: Schwingungserzeugender Wert P = 0,19

D-Anteil:

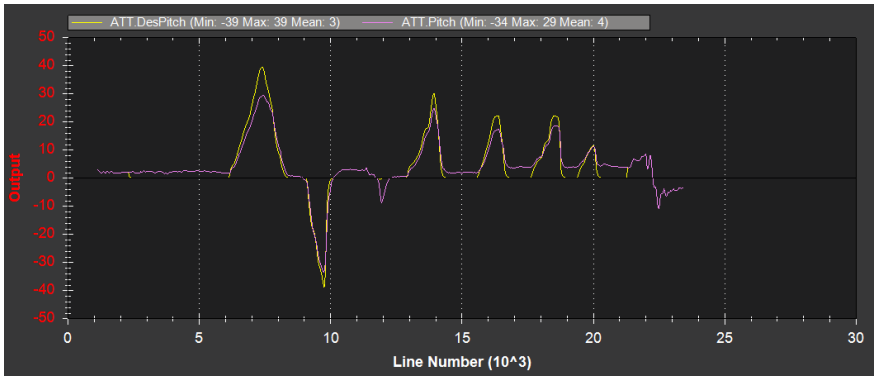


Abbildung 4.14: Anfangswert D = 0,001

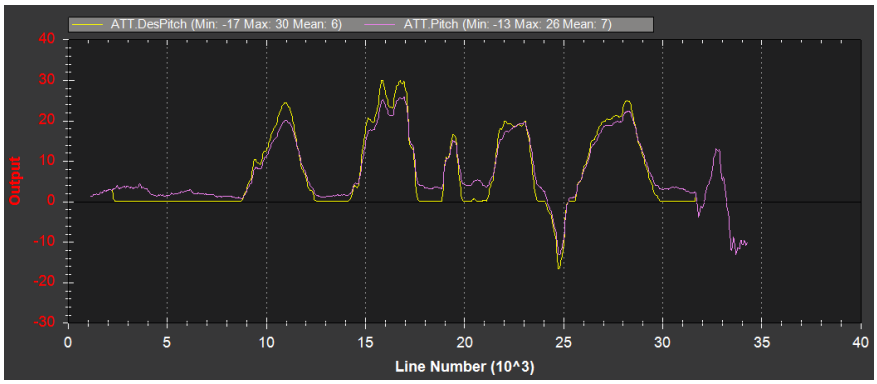


Abbildung 4.15: Optimaler Wert D = 0,006

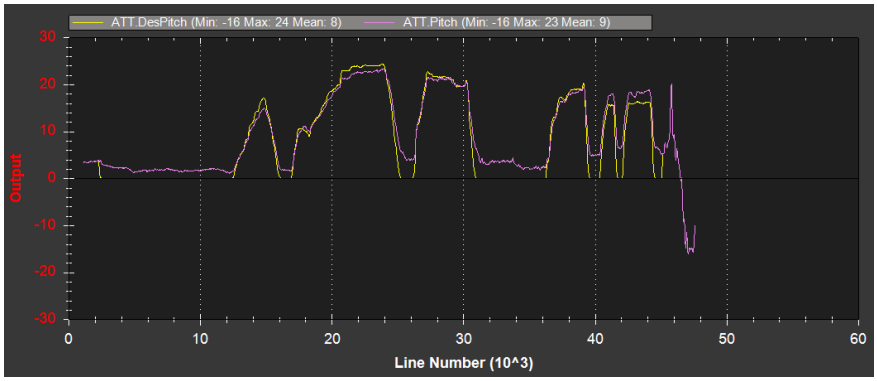


Abbildung 4.16: Schwingungserzeugender Wert D = 0,007

I-Anteil:

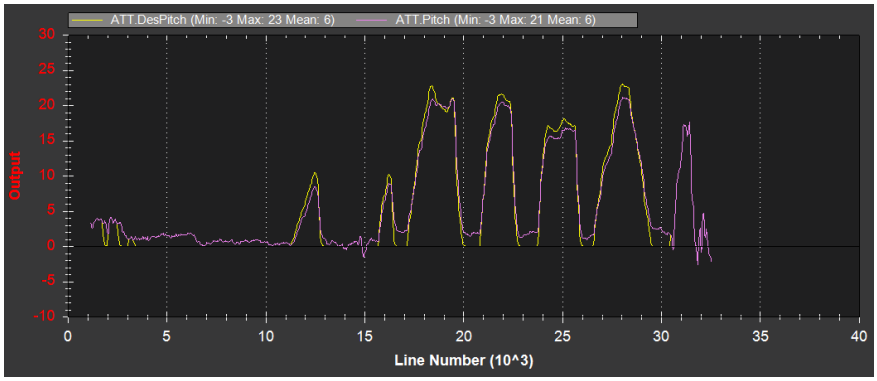


Abbildung 4.17: Anfangswert I = 0,03

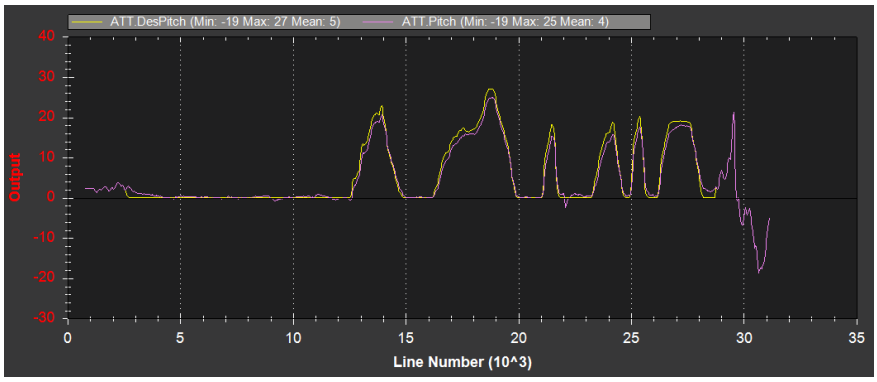


Abbildung 4.18: Optimaler Wert I = 0,09

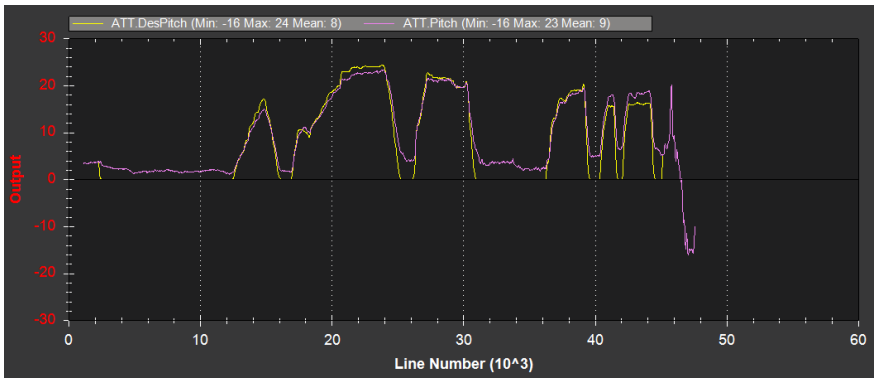


Abbildung 4.19: Schwingungserzeugender Wert I = 0,1

Somit hat sich folgender Arbeitspunkt ergeben:

Tabelle 4.3: Arbeitspunkt des Attitude-Reglers

Winkel Roll/Pitch	Winkel Yaw
P = 4,5	P = 4,5
Drehrate Roll/Pitch	Drehrate Yaw
P = 0,18	P = 0,2
I = 0,09	I = 0,018
D = 0,006	D = 0

Sämtliche eingestellte Parameter sind in dem Anhang zu finden.

4.2 Einrichten des FC über den QGC

4.2.1 Einstellen des Airframe und Sensorkalibrierung

Zunächst wird als Modell des Quadrocopters die X-Konfiguration für DJI F450 Flame Wheel eingestellt. Weiterhin wird der PixHawk-FC nach der von der PX4-Firmware angezeigten Anleitung in verschiedenen Richtungen positioniert und um verschiedene Rotationsachsen rotiert, um den Kompass, den ACC, den Gyro und den level Horizon zu kalibrieren. Da der Kompass und der Flugregler aufrecht am Flugzeug montiert und nach vorne gerichtet sind, wird vor dem Kalibrierungsvorgang zunächst die Standardorientierung ROTATION_NONE bei Set Orientations gesetzt, was bedeutet, dass der Pfeil nach vorne zeigt, wo der Yaw-Winkel 0° beträgt.

Abbildung 4.20: Orientierung des PixHawk-Autpilots [20]

Kompass:

Bei der Kompasskalibrierung wird der Autopilot wie nach den aufgelistet festgelegten Ausrichtungen positioniert und dabei um die angegebenen Achsen rotiert.

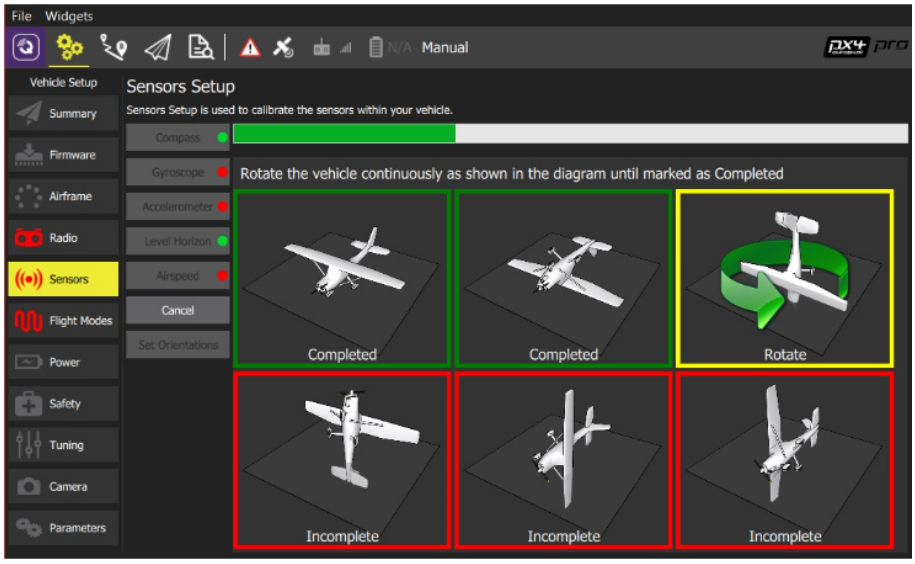


Abbildung 4.21: Kompasskalibrierung [21]

ACC:

Bei der Beschleunigungssensorkalibrierung wird der Autopilot in unterschiedliche Positionen gerichtet, in denen er für kurze Zeit verbleibt, um daraus den Nullpunkt zu ermitteln.

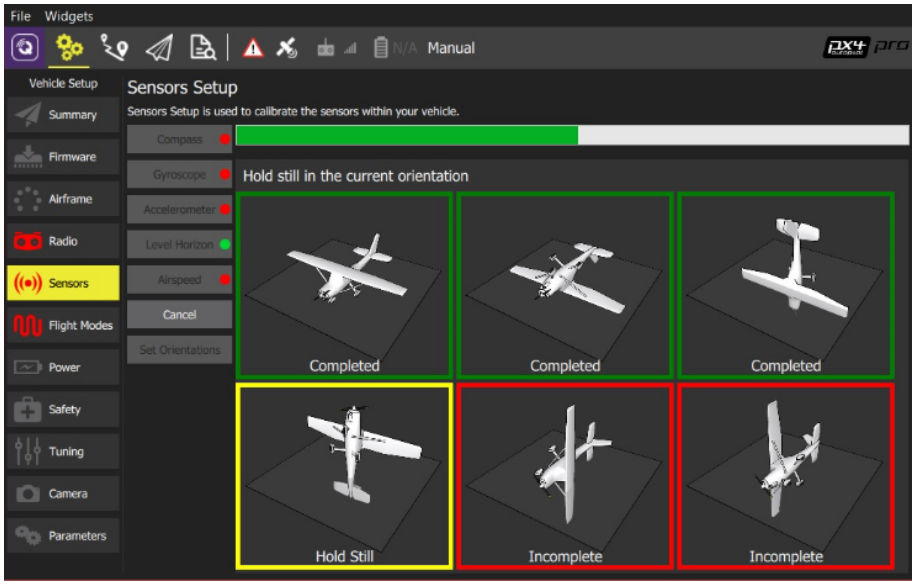


Abbildung 4.22: Beschleunigungssensorkalibrierung [22]

Gyroskop:

Während der Kalibrierung des Gyroskops wird Autopilot auf eine ebene Fläche abgestellt und für eine Weile ruhig gehalten.

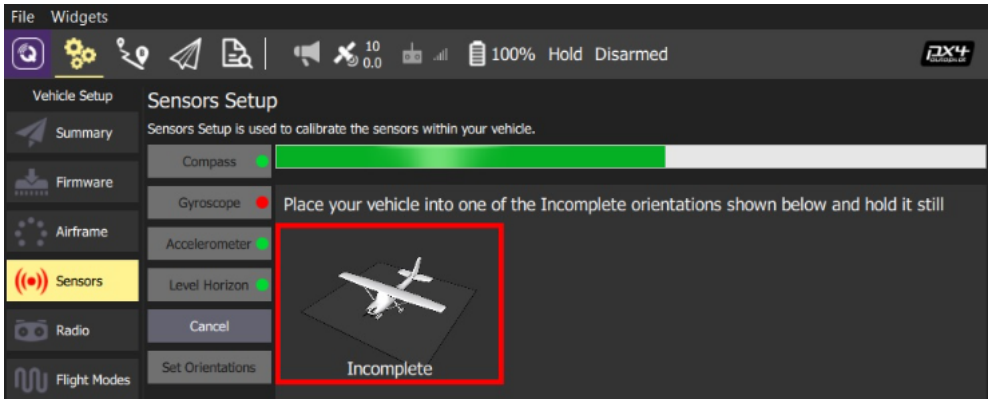


Abbildung 4.23: Kalibrierung des Gyroskops [23]

Level Horizon:

Zum Schluss wird der Horizont für den Autopilot kalibriert. Dabei ist der Autopilot waagerecht zu positionieren, während die Informationen erfasst werden. Die Position während des Horizontalfluges entspricht der Schwebeposition des Quadcopters.

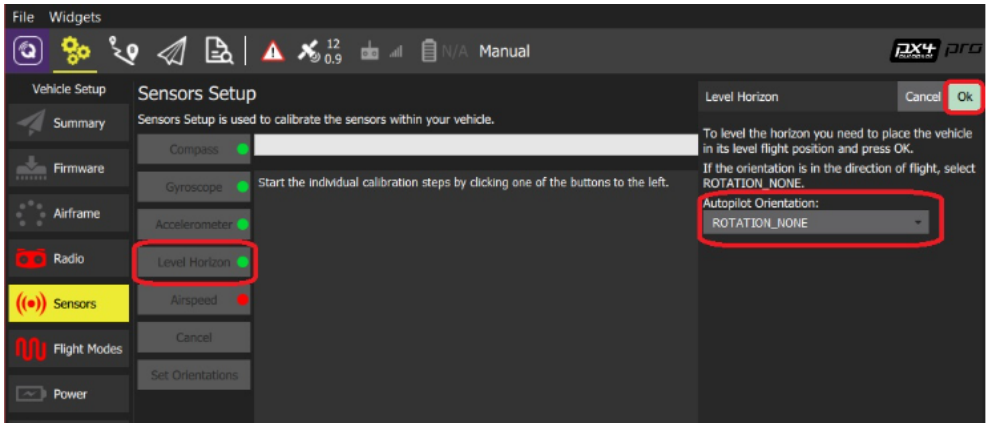


Abbildung 4.24: Kalibrierung des level Horizon [24]

4.2.2 Einstellen des RC-Systems und der Flugmodi

Der Dx6i-Spektrum-Funksender arbeitet in dem 2,4 Ghz-Frequenzbereich und eignet sich gut damit für die Fernsteuerung von fliegenden Fahrzeugen wie ACRO-Typen und Hubschrauben. Er lässt sich leicht programmieren und gewährleistet eine sichere Steuerung. Wie in der Abbildung 4.6 erkennbar ist, verfügt der Dx6i-Sender sechs Steuerkanäle: für die Roll- und Pitch-Bewegung mittels des Quer- und Höhenruder-Knüppels, für die Yaw-Bewegung mittels des Seitenruder-Knüppels, die Ab- und Auf-Bewegung mittels des Throttle-Knüppels, den Flugmodus-Wechsel mit dem GEAR-Schalter gemixt

mit dem ELEV D/R-Schalter und mit dem FLAP-Schalter für die Wölbklappenfunktion bei ACRO-Typen, die eine Roll- und Pitch-Bewegung für Abfliegen und Landen in bestimmte vordefinierten Positionen zu fahren ermöglicht.

Abbildung 4.25: Dx6i-SPEKTRUM-Funksender [25]

Für die Einstellung des RC-Systems muss zuerst der AR6200-Empfänger zum Betrieb mit dem Spektrum-DX6i-Sender gebunden werden. Beim Binding geht es darum, den Empfänger so zu programmieren, dass er durch einen spezifischen Code mit diesem Sender verknüpft wird und nur auf ihn reagiert. Nun wird die Sender-Kalibrierung mit Leichtigkeit durchgeführt. Als Erstes muss sichergestellt werden, dass QGC die richtigen Stick-Positionen anzeigt, die während der Kalibrierung eingehalten werden müssen. Dann kann der Seiten-/Gas-Knüppel sowie der Quer-/Höhenruder-Knüppel, wie nach Anleitung angezeigt wird, bewegt werden.

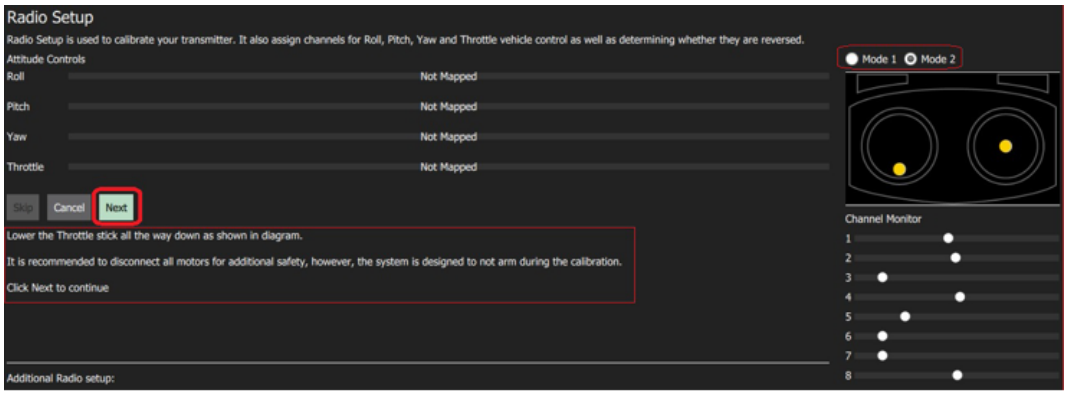


Abbildung 4.26: Kalibrierung der Funke [26]

Die Kanäle sind im Autopilot mit den gleichen Nummern definiert genauso wie in der Funke.

Tabelle 4.4: Steuerungskanäle gemäß QGC

Kanal	Signal
1	Throttle
2	Roll
3	Pitch
4	Yaw
5	Gear
6	AUX

Zuletzt wird im QGC eingegeben, um welche Flugmodi es sich während des Betriebs handeln soll. Hierfür werden das Manual-, Land- und Offboard-Modus ausgewählt.

Manual-Modus:

Der Flugcontroller bekommt die Sollwerte aus dem Funksender für die Regelung der Fluglage. Die Steuerung der Roll-/Pitch-Winkel über den Roll/-Pitch-Stick führt zur Links-/Rechtsbewegung sowie Vor-/Rückwärtsbewegung. Über den Throttle-Stick steuert man die Ab-/Auf-Geschwindigkeit und über den Yaw-Stick die Winkelgeschwindigkeit über der horizontalen Ebene. Für die Regelung werden die Istwerte von den internen Sensoren Gyro und ACC geliefert.

Land-Modus:

Bei diesem Betriebszustand landet die Drohne automatisch an der Startposition. Hierbei erfordert der Flugcontroller keine Benutzereingaben und benötigt für die Regelung externe Sensoren wie GPS und Drucksensoren.

Offboard-Modus:

Als Sollwertgeber wird der externe Computer RPI 3 verwendet. Dieser liefert Positions-, Geschwindigkeits-, Lagereferenz- und Ziel-Sollwerte, die von Mavros bereitgestellt werden, über seine serielle UART-Schnittstelle und Mavlink zum TELEM2-Port an dem Flugcontroller. Zur Fluglageregelung werden ebenfalls GPS und Barometer benötigt.

Für einen Wechsel zwischen den drei möglichen obengenannten Flugmodi steht der dem Gear-Schalter zugeordneten Fernsteuerkanal 5 zur Verfügung. Da mit einem Schalter nur zwei Betriebszustände aktiviert werden können, wird am RC-System die Mixer-Funktion aktiviert, um den Gear-Schalter mit dem ELEV D/R-Schalter zu mixen und daraus die in der Tabelle 4.2 dargestellten Betriebszustände zu erhalten. [27]
Bei der Flight Modes-Konfiguration wird weiterhin der dem Yaw-Stick zugeordneten RC-Steuerkanal 4 zusätzlich für das Einschalten der Motoren verwendet, indem der Yaw-Stick für drei Sekunden ganz unten links gehalten wird. Für das Ausschalten der

Tabelle 4.5: Flugmodi in QGC

Gear	ELEV D/R	Flugmodus
0	0	Land
0	1	Offboard
1	0	Manual
1	1	Manual

Motoren wird sicherheitshalber in dem QGround Control der Parameter COM_DISARM_LAND unter der Sicherheitseinstellung aktiviert, sodass die Motoren nach dem Landen bzw. wenn der Throttle-Kanal in der Nullposition ist, automatisch nach 3 Sekunden ausschalten.

4.2.3 Motoreinstellungen

Über den QGround Control müssen einige Motorparameter in dem Autopilot eingestellt werden. Hierzu zählen die PWM-Main-Outputs und die PWM-RC-Inputs. Da die eingesetzte BLDC-Motoren unterschiedlichen Bedarf an Spannung aufweisen, muss einen geeigneten Wert gewählt werden, bei dem alle Motoren richtig laufen, um eine nützliche Schubkraft zu erreichen. Standardmäßig sind die Werte PWM_MIN = $1000\,\mu s$ und PWM_MAX = $2000\,\mu s$ bei den Main-Outputs sowie bei den RC-Inputs zu wählen. Nun wird nach diesen PWM-Einstellungen das Verhalten der Motoren beobachtet. Dabei werden die Motoren über die ECSs und den Autopilot von der Funke angesteuert. Allerdings hat sich durch den Versuch festgestellt, dass zwei Motoren sich nicht richtig drehen lassen. Der Grund dafür ist, dass der minimale Wert $1000\,\mu s$ nicht ausreichend ist, um eine gewünschte Drehzahl kontinuierlich zu erreichen. Auf Grund dessen wird allmählich der PWM_MIN der Main-Outputs erhöht, bis sich ein passender Wert für das Spinning alle Motoren ergeben hat, welcher $1140\,\mu s$ beträgt. Bei dem Parameter Radio Calibration werden die Subparameter RCx_MIN, RCx_MAX und RCx_TRIM eingestellt. Damit nach dem Armen die Motoren nicht direkt anlaufen, wird für die Nullposition des RC-Gas-Knüppels, also bei dem RC1_TRIM, ein etwas kleiner PWM_MIN-Wert von $1100\,\mu s$eingegeben während für die Mittelpostion des Yaw-, Roll- und Pitch-Sticks bei dem RCx_TRIM der Wert $1500\,\mu s$ eingestellt wird, der genau zwischen $1000\,\mu s$ und $2000\,\mu s$ liegt.

4.2.4 Testen der BLDC-Motoren und der E300-ESCs

Vor den Verkabelungs- und Programmierungsprozessen muss als Erstes bei jedem Motor und jedem ESC die Funktionsweise überprüft werden: der E300-ESC-Servostecker wird direkt an den Throttle-Port des AR6200-Empfängers angeschlossen. Der AR6200-Empfänger bekommt eine Spannungsversorgung von 5 V von dem Pixhawk-FC, der durch die USB-Schnittstelle an dem PC angeschlossen ist. Die drei Phasen werden über Rundverbinder an den dazugehörigen Motor und die Versorgungs- sowie GND-

Leitungen werden an eine Spannungsquelle, die auf eine Gleichspannung von 12 V bei einem Strom von 3 A eingestellt ist, angeschlossen. Dabei muss der Throttle-/Yaw-Knüppel des DX6i-Senders unten auf der Nullposition positioniert werden. Der Roll-/Pitch-Knüppel (Quer-/Seitenruder-Knüppel) muss in der Mittelposition bleiben, da sonst der ESC nicht starten kann. Weiterhin werden das RC-System und der E300-ESC angeschaltet. Nun kann man durch die Bewegung des Gasknüppels den Motor ansteuern bzw. die Drehzahl minimieren und maximieren. Als Ergebnis stellt sich fest, dass alle ESCs bereits kalibriert sind (ESCs sind angelernt), d. h. der PWM-Bereich des Senders mit dem der ECSs übereinstimmt und alle Rotoren wie gewünscht sich drehen.

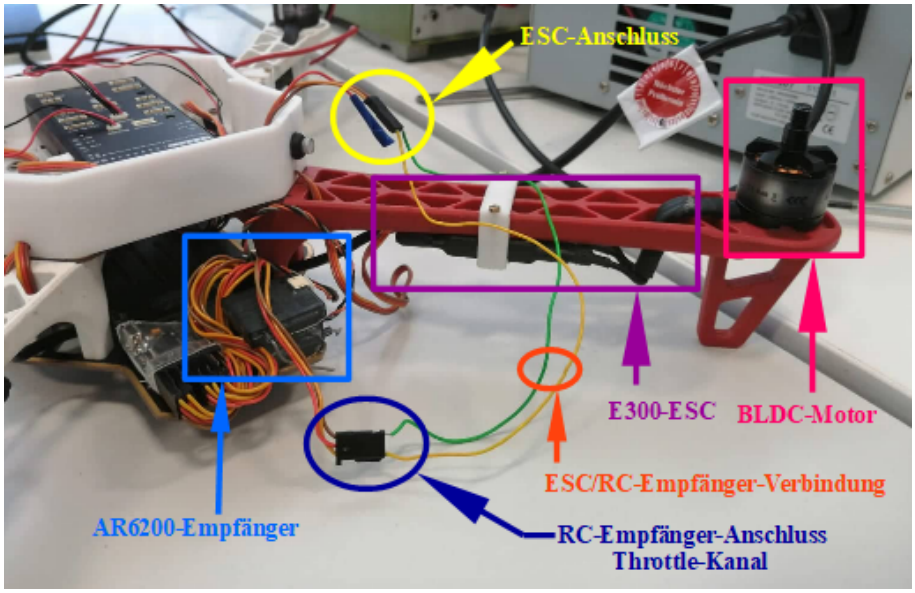


Abbildung 4.27: Motor-ESC-Test

Ebenfalls können die ESC-PWM-Minimum und -Maximum über die QGC-Software kalibriert werden. Zur Sicherheit dürfen die Propeller zunächst nicht montiert werden. Bei der Kalibrierung werden die BLDC-Motoren über die ESCs mit dem Flugcontroller verbunden, eine Spannungsquelle, die auf eine Spannung von 12 V eingestellt ist und einen Strom bis zu maximal 25 A zur Verfügung stellt, wird an das Leistungsmodul angeschlossen und der Flugcontroller wird für die Kommunikation mit dem QGround Control über eine USB-Schnittstelle an den Computer angeschlossen. Mit Hilfe der Anleitung führt man die Kalibrierung durch, um sicher zu stellen, dass der PWM-Bereich des Flugcontrollers identisch mit dem der ECSs ist.

4.2.5 PID-Tuning über QGC

Die Schritte der Parametrisierung des PID-Reglers über den QGC werden anlog, wie unter dem Subkapitel *PID-Tuning über MP* beschrieben ist, allerdings im Acro-Modus durchgeführt. In dem QGroundControl bietet sich an, unter der erweiterten Einstellung

des PID-Tuning die Ist-und Sollwertsignale des Winkels sowie der Winkelgeschwindigkeit von Roll, Pitch und Yaw im Stillstand grafisch darzustellen sowie die PID-Parameter anzupassen. Anhand der Graphen ist erkennbar, dass nach Durchführung der Sensorkalibrierung bei Roll und Pitch einen Ist-Sollwert-Offset zustande kommt. Dieser ist allerdings nicht beeinflussbar, jedoch vernachlässigbar, da dieser minimal ist und im Toleranzbereich zwischen 0,5 deg und 1,5 deg liegt. Bei dem Yaw sind Winkel-Istwert und Winkel-Sollwert identisch. Bei der Winkelgeschwindigkeit weisen alle einen Offset auf, welcher durch Änderung von MC_Roll_P, MC_Pitch_P und MC_Yaw_P-Parametern einstellbar ist. Dieser Parameter entspricht der Proportionalverstärkung, das heißt die gewünschte Winkelgeschwindigkeit in $\frac{rad}{s}$ für den Fehler von 1 rad. Sämtliche eingestellte Parameter sind in dem Anhang zu finden.

4.3 Vergleich zwischen QGC- und MP-Software

Im Prinzip sind beide Bodenstation-Softwares unkompliziert in der Bedienung. Allerdings verfügt der MP mehr Vorteile gegenüber den QGC. Als Erstes ist zu erwähnen, dass bei dem MP die Main-Outputs beliebig zu aktivieren sind, während bei dem QGC automatisch nur die ersten vier Main-Outputs aktiviert werden, wenn als Rahmenklasse Quad eingestellt ist. Als weiterer Vorteil bei dem MP ist es möglich, die Motoren einzuschalten, während die Software bereits gestartet ist. Dies wird bei dem QGC nicht erlaubt. Außerdem bietet der MP das Autotuning des PID-Reglers, welches eine einfache und genaue Methode ist, die Regelparameter zu ermitteln. Bei der Motoreinstellung über den MP laufen alle Motoren bei dem Standard-PWM_MIN-Wert 1000 μs mit gleicher Drehzahlgeschwindigkeit an. Bei der Einstellung über den QGC muss der PWM_MIN-Wert höher als der Standardwert liegen, damit alle Motoren das gleiche Verhalten aufweisen. Ebenfalls können die Logs der Soll- und Istwerte anhand einer Option direkt in der MP-Software geplottet werden. Bei dem QGC müssen die Logs zunächst heruntergeladen werden und dann in einem Online-Tool grafisch dargestellt werden. Darüber hinaus bietet sich die Möglichkeit im MP an, die Motoren vor der Betriebsnahme mit wenig Gas auf die Reihenfolge und die Anlaufrichtung zu testen. Dies stellt sicher, ob sie an die richtigen Output-Pins des Autopilots angeschlossen sind. Zudem können den Main-Outputs verschiedene Funktionen zugewiesen werden und somit auch für andere Zwecke verwendet werden. Bei der Verwendung von QGC können die Motoren sowohl in Automodi als auch im Manual-Modus ein- und ausgeschaltet werden. Im Gegensatz dazu, kann der Autopilot, wenn er über MP konfiguriert ist, die Motoren in Automodi nicht ein- und ausschalten. Bei Bereitstellung der Sollwerte aus einem Companion Computer für den PixHawk-FC, bietet sich die Möglichkeit bei MP Guided oder Guided_nogps, das heißt, den Automodus mit oder ohne Anschluss eines GPS-Moduls, zu setzen. Bei QGC ist ein GPS-Modul erforderlich, um die Drohne in Offboard-Modus zu betreiben.

5 Autonome Steuerung des Quadcopters mittels ROS

In diesem Kapitel werden die Schritte hinsichtlich des autonomen Antriebs des Quadcopters mittels ROS anhand der Apriltag-Detektierung beschrieben. Als Erstes wird die Hardware, der RPI 3, vorgestellt, der dazu dient, die Sollwerte für den Autopilot im Offboard/Guided_nogps-Modus bereitzustellen. Dann wird darauf eingegangen, welches Betriebssystem verwendet wird, unter ihm ROS laufen kann, bzw. wie dieses auf den RPI 3 installiert wird. Im Weiteren wird ein Überblick über die Erkennung des Apriltags gegeben. Daraufhin wird eine ausführliche Erklärung zu der Herstellung der Verbindung zwischen dem RPI 3 und dem PixHawk-FC über die UART-Schnittstelle verschaffen. Abschließend wird der C++-Quellcode, welcher die Schritte der autonome Steuerung enthält, als Flussdiagramm dargestellt und allmählich erklärt.

5.1 Raspberry PI 3

Der RPI 3 ist ein Einplatinencomputer, der einen Mikroprozessor mit ARM-Architektur verfügt und als Robotersteuerung oder leistungsfähiges Entwicklungsboard eingesetzt wird. Als Betriebssystem kommen vorallem Linux-Distributionen zum Einsatz. Das Laden des Betriebssystems erfolgt durch eine wechselbare SD-Speicherkarte.

Abbildung 5.1: Raspberry PI 3 Model B+ [36]

Wie auf der Abbildung 5.1 zu sehen ist, stellt im Rahmen der Studienarbeit verwendete RPI3 Model B+ vier USB-Anschlüsse, ein Netzwerkkabel- und HDMI-Anschluss zur Verfügung. Ebenfalls sind WLAN und Bluetooth bereits integriert. Über die GPIO-Schnittstelle kann der RPI 3 mit Sensoren und elektronischen Bauteilen direkt kommunizieren. Über eine mini-USB-Schnittstelle wird er mit Strom versorgt. Allerdings hat er die Besonderheit im Gegensatz zu den alten Modellen mehr Leistung aufzunehmen (2 A bis 2,5 A bei 5 V).

5.2 Ubuntu-Mate- und ROS-Installation auf dem RPI3

Damit ROS verwendet werden kann, muss zunächst das Betriebssystem Ubuntu Mate auf den RPI 3 installiert werden. Hierbei wird die neuste Version von Ubuntu Mate und das Programm Win32 Disk Imager auf einen Computer installiert. Weiterhin wird in dem Win32 Disk Imager das heruntergeladene Image von Ubuntu Mate geöffnet dann auf eine SD-Karte geschrieben. Nachdem das Image auf die SD-Karte des RPI überspielt wird, muss dies als Nächstes auf den RPI3 installiert werden. Nun wird die SD-Karte in den RPI 3 eingesteckt und alle benötigte Geräte wie Maus, Tastatur, Bildschirm, Netzwerkkabel und Netzteil werden angeschlossen. Nach der Durchführung einiger Einstellungen kann anschließend ein Benutzer angelegt werden.
Für die Installation von ROS wird die Anleitung dazu auf der Internetseite [37] zur Verfügung gestellt. Hierbei werden die Befehle Schritt für Schritt in das Terminal-Mate eingegeben, um die benötigte ROS-Pakete zu installieren.

5.3 Ansteuerung des Quadcopters mittels ROS und Apriltags

In diesem Unterkapitel wird auf ein Teilgebiet des Computer Vision, die optische Zeichenerkennung, eingegangen. Allgemein bedeutet Computer Vision die Fähigkeit eines Computers zu sehen, das heißt die Fähigkeit aus realen Bildern Daten zu extrahieren, um damit z. B. Modelle zu generieren, und Sehvorgänge in dreidimensionalen Welt nachzubilden. Im Rahmen dieser Studienarbeit ist das Ziel, die Drohne durch die Apriltags-Erkennung und einen dazu passenden C++-Quellcode abfliegen und landen zu lassen.

5.3.1 Bildverarbeitung

Für die Tags-Detektierung wird eine logitech-Kamera, die an eine USB-Schnittstelle des RPI 3 angeschlossen wird, verwendet. Als Erstes müssen die heruntergeladene Pakete kompiliert werden. Dann wird ein Terminal durch den Befehl roscore als Master gestartet, damit die Knoten bzw. die Launch-Dateien als Slaves gestartet werden können. Um sicher einzugehen, dass die USB-Kamera angeschlossen ist bzw. von ROS erkannt wird, wird in das Command der Befehl ls usb eingegeben. Mit der Eingabe von ls/dev| grep video erfährt man, ob die Video-Geräteknoten angelegt sind.

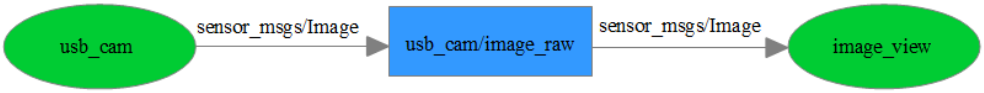


Abbildung 5.2: ROS-Graph der usb_cam-Node vor der Kamerakalibrierung

Wie in der Abbildung 5.2 dargestellt ist, wird weiterhin der usb_cam-Treiber akti-

viert, indem der usb_cam_test.launch-File gestartet wird. Da die usb-Kamera noch nicht kalibriert ist, wird zunächst nur das Topic usb_cam/raw_image des Nachrichtentyps sensor_msgs/Image veröffentlicht.

Als Nächstes wird die Kamera unter Verwendung eines Schachbretts kalibriert. Hierbei wird die Kalibrierungsknote cameracalibrator.py ausgeführt, wodurch das Kalibrierungsfenster geöffnet wird, in dem das Schachbrett zu sehen sein soll, welches in verschiedene Richtungen und um verschiedenen Achsen bewegt wird. Anschließend werden die Kalibrierungsergebnisse in einem yaml-File gespeichert, das in dem /usb_cam/camera_info-Topic des Nachrichtentyps sensor_msgs/CameraInfo aufgerufen wird. Um die Abbildung zu rektifizieren, d. h. Verzerrungen in Bilddaten zu eliminieren, liegt der image_proc-Knote nach dem Kameratreiber-Knote. Hierfür ist das camera_rect.launch-File zu starten, wodurch das Topic /usb_cam/camera_info, das die Kalibrierungsparameter als Metadaten beinhaltet, und das Topic usb_cam/raw_image, das eine ununterbrochene Bildaufnahme erlaubt, von der image_proc-Knote abonniert werden. Veröffentlicht wird das Topic /usb_cam/image_rect des Nachrichtentyps sensor_msgs/Image, das unverzerrtes Bild enthält.

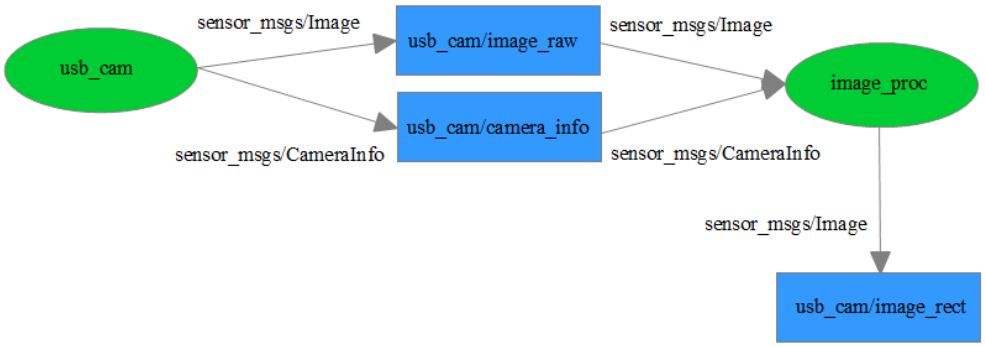


Abbildung 5.3: ROS-Graph der usb_cam-Node nach der Kamerakalibrierung

5.3.2 Detektierung des Apriltags2

Vorerst muss das verwendete Apriltags2-Bild vordefiniert werden. Deshalb werden sowohl die ID-Nummer und die Größe in das tags.yaml-File als auch das tag_family in das settings.yaml-File eingefügt. Nach dem Starten des camera_rect.launch-Files wird das continuous_detection.launch-File ausgeführt. Dies führt dazu, dass die zwei Topics /usb_cam/camera_info und /usb_cam/image_rect von dem apriltags2_ros_continuous-Knoten abonniert werden und als Outputs werden die Topics /tf, /tag_detections und /tag_detections_image veröffentlicht. Das /tf-Topic mit dem Nachrichtentyp tf2_msgs/TFMessage beinhaltet Daten über die translatorische und rotatorische Bewegungen des erkannten Apriltags2-Bildes. Das /tag_detections-Topic des

Nachrichtentyps apriltags2_ros/AprilTagDetectionArray enthält die ID-Nummer, die Größe und die Pose-Informationen als geometry_msgs/PoseWithCovarianceStamped-Nachricht, die die Position, die Orientation und die Kovarianz-Matrix relativ zu der Kamera umfasst. Ebenfalls enthält das /tag_detections_image-Topic mit dem Nachrichtentyp sensor_msgs/Image das rektifizierte Bild allerdings mit erkannten hervorgehobenen tags, welches durch das Abonnieren von der image_view-Knote in dem Videofenster ausgeben lässt.

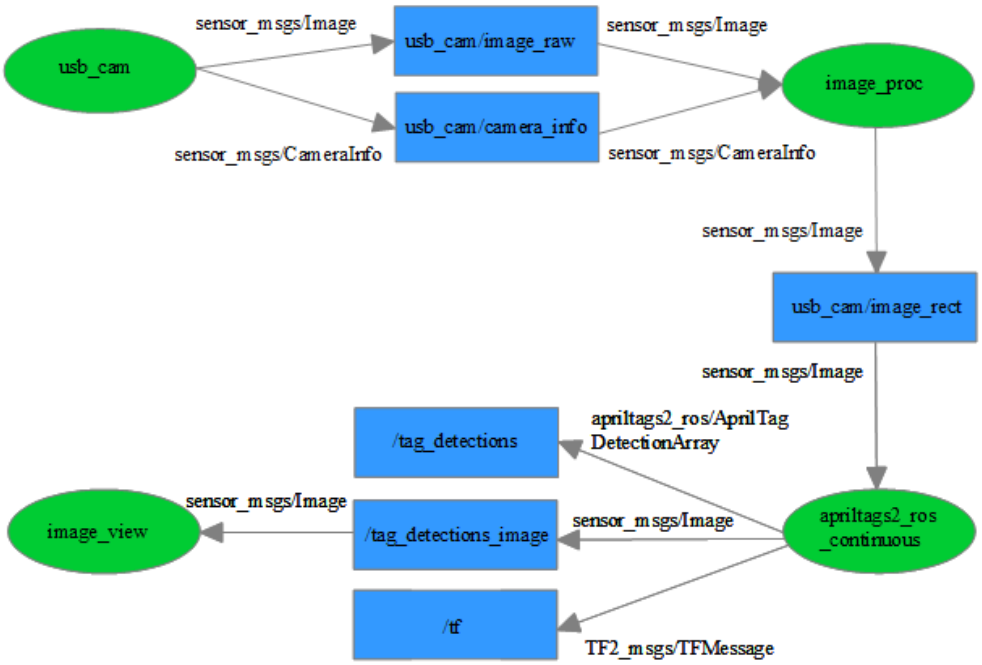


Abbildung 5.4: ROS-Graph der usb_cam- und apriltags2_ros-Node

5.4 Herstellung der Kommunikation zwischen PixHawk und ROS über den Raspberry PI 3

Die Kommunikation zwischen dem Flugcontroller und dem RPI 3 erfolgt über die GPIO-serielle Schnittstelle, den UART an den Pins 14 (Tx) und 15 (Rx). Um dies zu ermöglichen, muss vorher das GPIO konfiguriert werden bzw. die Pins aktiviert werden. In diesem Abschnitt werden die Konfigurationsschritte verdeutlicht. [38]

Durch die Eingabe von ls -l/dev in die Konsole erscheint das Standard RPI 3 serielle Port-Alias ($serial\,1 \longrightarrow ttyAMA0$), an dem eine Bluetooth-Kommunikation liegt. Da bei dem RPI3 der serial 0 standardmäßig auf die GPIO-Pins 14 und 15 verweist, muss dies aktiviert werden. Dies geschieht über die config.txt-Datei: Durch die Eingabe von sudo nano /boot/config.txt in die Konsole öffnet sich die config.txt-Datei. Ganz unten wird der Eintrag enable_uart = 1 hinzugefügt und die Core Frequenz auf 250 MHz durch die Eingabe von core_freq = 250 fixiert. Anschließend wird die Datei gespeichert

und das System von neu gestartet. Nun ist der serial 0, der ttyS0 benutzt, aktiviert und in der Liste der seriellen Port-Aliases zu sehen ($serial\,0 \longrightarrow ttyS0$). Wenn eine Software übertragen wird, greift sie standardmäßig auf den /dev/ttyAMA0, welcher eine hohe Performance besitzt und eine volle Leistung beim Abschalten der Bluetooth-Funktion erbringt. Da aber wie bereits erwähnt der serial 0 den Pins 14 und 15 zugewiesen ist, werden die Ports /dev/ttyAMA0 und /dev/ttyS0 vertauscht, sodass der serial 0 den /dev/ttyAMA0 benutzt. Hierbei wird der folgende Befehl dtoverlay = pi3-miniuart-bt am Ende der config.text-Datei hinzugefügt und das System von neu gestartet. Um zu überprüfen, ob die Ports vertauscht wurden, wird eine Übersicht über die Belegung erneut angezeigt ($serial\,0 \longrightarrow ttyAMA0, serial\,1 \longrightarrow ttyS0$). Für die serielle Kommunikation reicht nicht aus den UART zu aktivieren. Es muss der Permission-Befehl sudo chmod a+rw in die Konsole eingegeben werden, um eine Software-Übertragung zu erlauben.

Zu beachten ist, dass bei der QGC-Software der Parameter SYS_COMPANION auf Companion Link mit der Baudrate 921600, was für den TELEM2-Port steht, eingestellt wird. Falls die MP-Software verwendet wird, so muss bei dem Parameter SERIAL2_BAUD der Wert 921 gesetzt werden, welcher für die Baudrate 921600 des TELEM2-Ports steht. Des Weiteren müssen in dem px4.launch-File der Parameter FCU-Connection-URL für den Hauptkommunikationsknoten mavros_node und der Parameter GCS-Connection-URL für die Kommunikationsschnittstelle mit der Bodenstation-Software gsc_bridge angepasst werden.

Zum Testen der Kommunikation zwischen dem FC und dem RPI 3 wird nun mit dem Befehl roslaunch mavros px4.launch die px4.launch-Datei gestartet. Die Ausgabe in dem Command zeigt, dass Informationen aus dem FC detektiert sind, womit bestätigt wird, dass der FC für einen Datenaustausch mit dem RPI 3 bereit ist.

5.5 Takeoff- und Land-Modus mittels der Apriltags-Erkennung

In diesem Unterkapitel wird das cpp-Programm namens Takeoff_Land_node verdeutlicht, welches die Schritte zum Ansteuern der Drohne mit Hilfe der Apriltag-Detektierung enthält.

Als Erstes muss die in Mavros/scr-Ordner gespeicherte Takeoff_Land_node.cpp-Datei als ausführbare Datei in der CMakeList.txt-Datei bei add_dependencies() sowie bei target_link_libraries() hinzugefügt werden, damit sie kompiliert wird. Das add_dependencies() stellt sicher, dass die eingebundene Nachrichtendateien als Header-Dateien generiert werden, bevor sie verwendet werden. Das target_link_libraries() ermöglicht den Takeoff-Land-node.cpp-Knote durch die Eingabe von rosrun in das Command aufzurufen. Um die Daten des tag_detections-Topic aus dem apriltas2-

ros-Knoten verwenden zu können, muss ebenso die Nachrichtenquelle des Topic apriltags2_ros sowohl in der CMakeList.txt-Datei bei find_package() und catkin_package(CATKIN_DEPENDS) als auch in der package.xml-Datei bei build_export_depend und run_depend eingefügt werden.
Als Folgendes wird der C++-Quellcode bei Einrichtung des CF mit QGC anhand der Flowchart-Darstellung schrittweise erklärt.

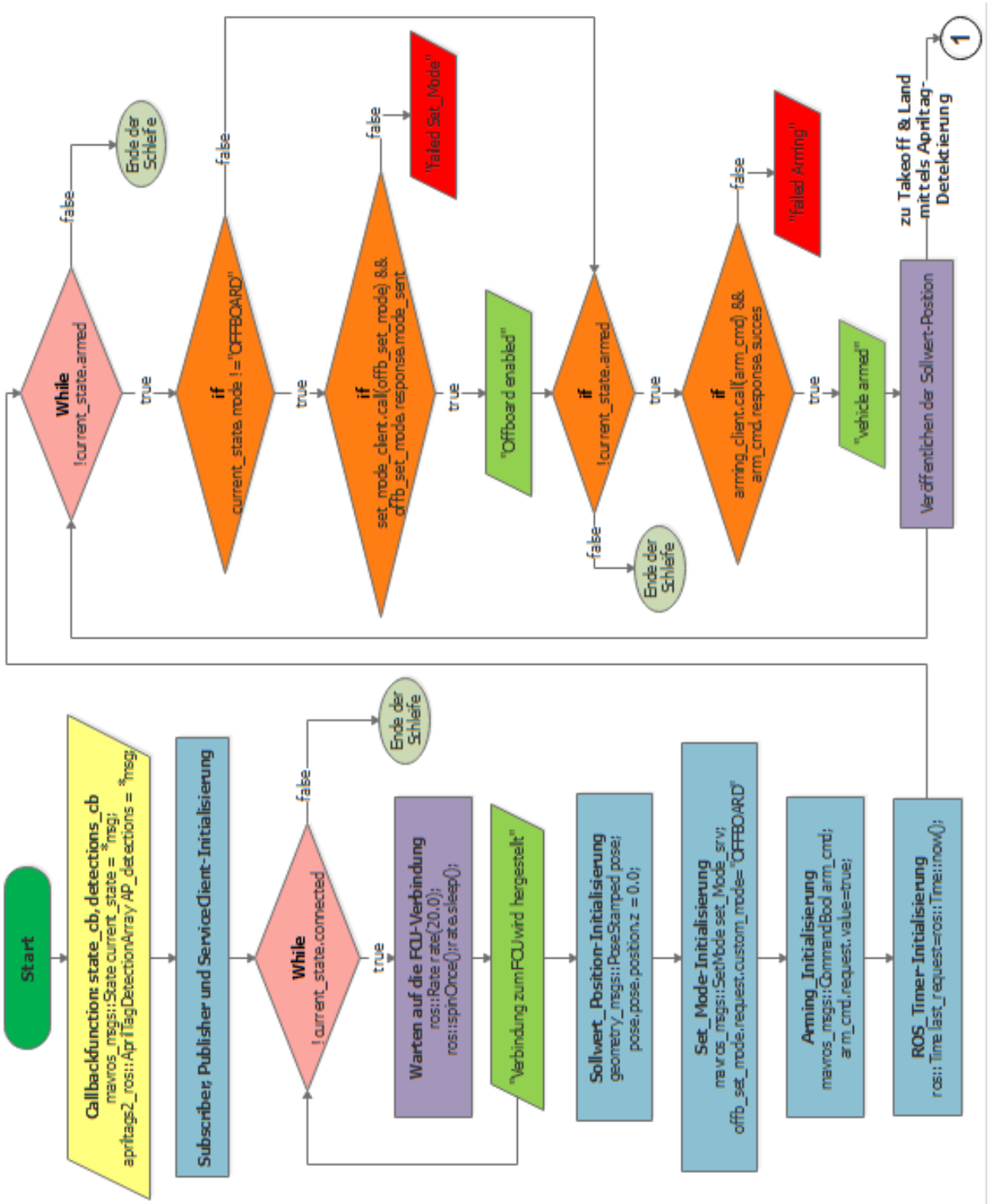


Abbildung 5.5: Setzen des Offbord-Modus und Einschalten der Motoren

Zunächst werden die in Mavros-Paket vorhandene Messages und Services zum Betrieb von Topics inkludiert. Nun um den aktuellen Stromzustand des Autopilots zu überprüfen und zu speichern, wird die Callback-Funktion state_cb erstellt. Hierbei wird der Stromzustand der State-Nachricht als Pointer zugewiesen, um die Nachricht zu abbonieren und auf die in ihr gespeicherten Variablen zuzugreifen. Somit wird die Verbindung zum FC, das bereits gesetzte Modus sowie den Motorbetriebszustand abfragen. Ebenfalls wird eine weitere Callback-Funktion benötigt. Damit wird über die Variable AP_detections eine Information geschaffen, ob eine Apriltag-Detektierung stattfindet und Daten wie id und size des Tags aus dem tag_detections-Topic bekommen.
Zudem werden Variablen initialisiert, die für das Abbonieren des FC-Status und der Apriltag-Detektierungen, das Veröffentlichen der local-position-Sollwerte und das Anfordern nach der Aktivierung des Offboard-Modes sowie nach dem Schalten der Motoren zuständig sind.

Im Weiteren wird die Veröffentlichen-Rate der Sollwerte festgelegt. Diese darf minimal 2 Hz sein, da es sonst auf das vorherige Modus automatisch gewechselt wird, wenn die Zeit zwischen zwei Veröffentlichen-Befehle länger als 500 ms ist. Die darauf folgende while-Schleife wird zum überprüfen der Verbindung zum FC, erstellt.
Als Voraussetzung für das Aktivieren des Offboard-Modus müssen vor dem Starten bereits Position-Sollwerte durch die for-schleife veröffentlicht werden. Die Position (0,0,0) wird als Startposition der Drohne vordefiniert.
Außerdem werden die Aufforderungen bezüglich des Offboard-Modus bzw. des Einschaltens der Motoren vordefiniert. Zudem wird die Zeit der letzten Anfrage als aktuelle ROS-Uhrzeit festgelegt.
Anhand der while-Schleife lässt sich der FC sein Modus in das Offboard wechseln, woraufhin die Drohne durch das Schalten der Motoren für den Flugvorgang bereit wird. Um den Autopilot nicht mit den Anforderungen zu überfluten, werden die Serviceanrufe um 5 Sekunden verteilt. Fernerhin werden die Position-Sollwerte mit der definierten Rate an dem FC weitergesendet.
Im Folgenden Abschnitt wird der zweiter Teil des Programms vorgestellt, bei ihm es sich darum geht, Position-Sollwerte sowie Ausschalten-Befehl unter Bedingung der Erkennung der Apriltags an dem FC zu senden.

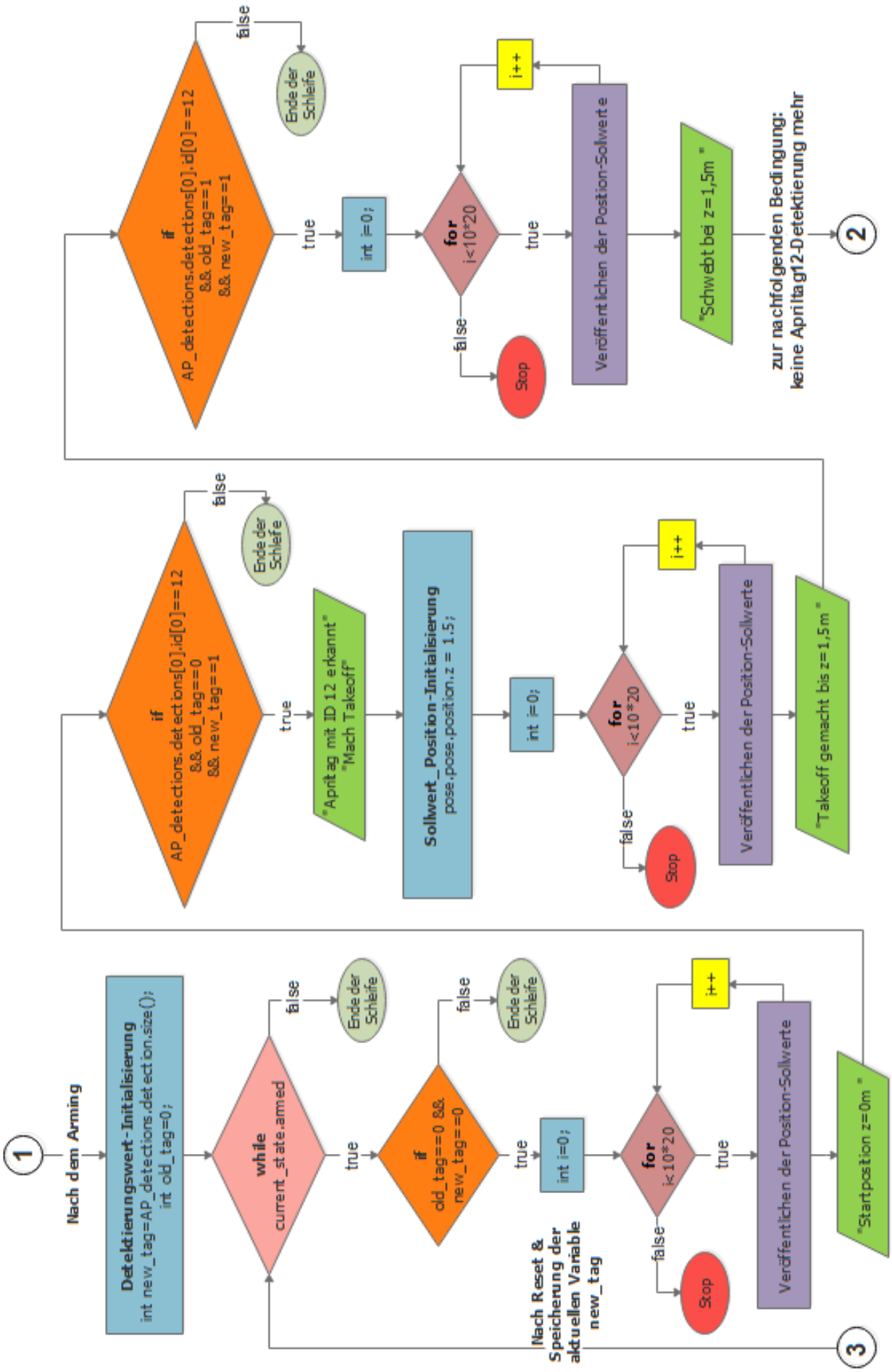


Abbildung 5.6: Takeoff durch Detektierung des Apriltags-ID12

Der restliche Quellcode ist ziemlich selbsterklärend. In der Variable new_tag wird über AP_detections.detections.size(), die der Spaltenanzahl des detections-Array entspricht, die aktuelle Information gespeichert, ob ein Tag in dem Moment erkannt wird. Der Variable old_tag, welche den alten Wert für die nächste Anweisung enthält, wird die Null als Anfangswert zugewiesen, welche leeres Array bzw. keine Tag-Erkennung bedeutet. Am Ende der while-Schleife wird für die nächste Bedingung der neue Wert aus dem vorherigen Bedingung als alter Wert gesetzt und der neue Wert als aktueller Wert gespeichert. Für den Fall, dass ein Tag erkannt wird, ist zusätzlich über AP_detections.detections[0].id[0] die id-Nummer erforderlich, damit der Autopilot zwischen dem Takeoff-Vorgang und dem Land-Vorgang sich unterscheiden kann. Hierbei soll der Tag mit der id-Nummer 12 den Takeoff-Vorgang antreiben, während der Tag mit der id-Nummer 13 das Land-Modus bewirkt.

Nach dem Einschalten der Motoren bleibt der Quadcopter als Erstes an der Startposition, solange kein Tag detektiert wird. Wird nun der Detektierungszustand geändert mit der Bedingung, dass die id-Nummer 12 lautet, so fliegt die Drohne bis 1,5 m hoch. Während die Kamera ohne Unterbrechung die gleiche id-Nummer einliest, schwebt die Drohne immer noch bei der gleichen Höhe. Wird der Apriltag mit id 12 vor der Kamera entfernt, so schwebt die Drohne weiter bei der Höhe 1,5 m. Erkennt die Kamera den Apriltag mit id 12 von neu, so bleibt die Drohne auf der Höhe 1,5 m. Solange der Apriltag mit id 12 abwechselnd wieder entfernt und erkannt wird, schwebt die Drohne andauernd bei dem vorgegebenen Position-Sollwert.

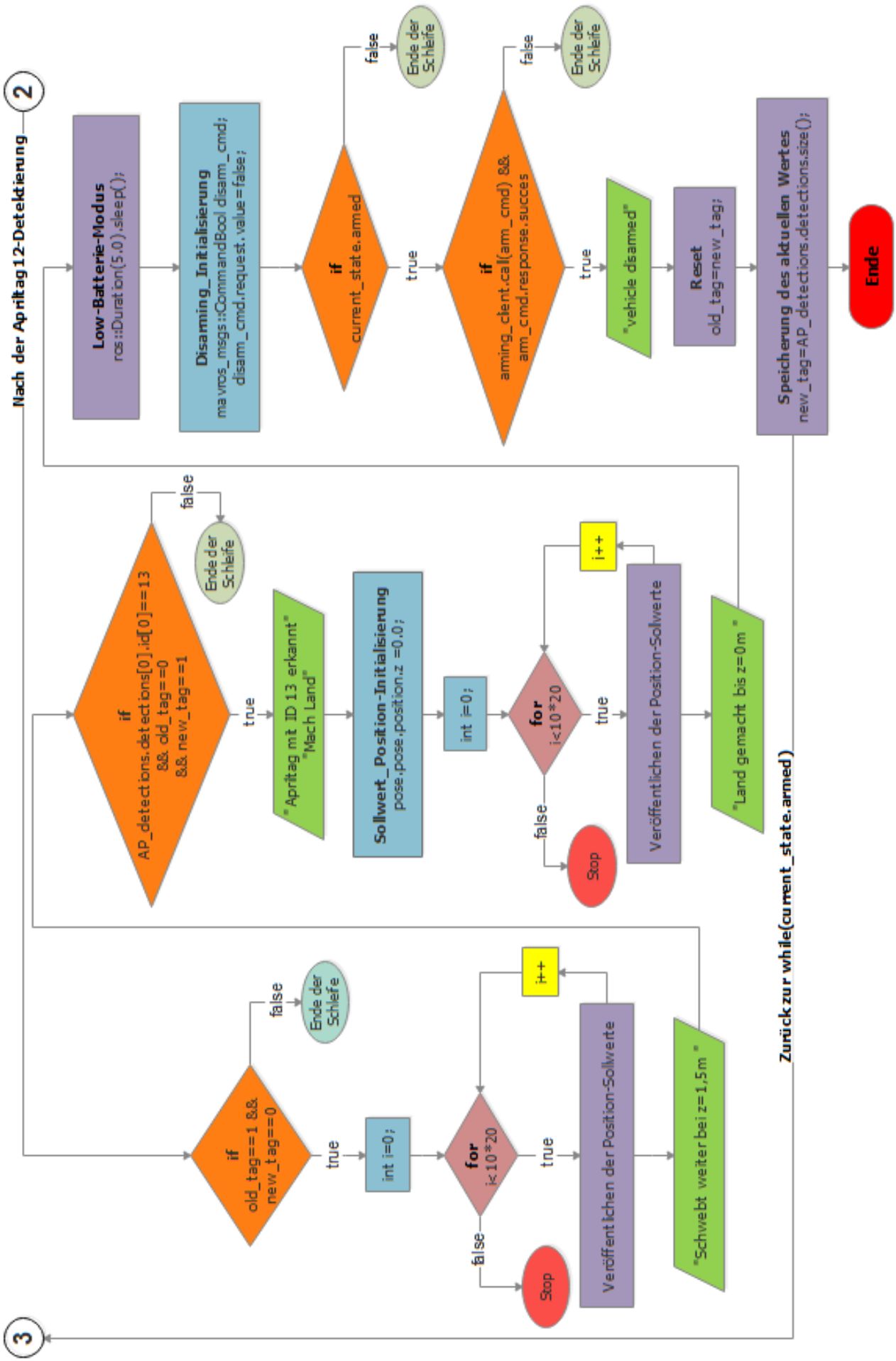


Abbildung 5.7: Land durch Detektierung des Apriltags-ID13

Durch das Detektieren des Apriltags mit id 13 wird dem FC der Landungsbefehl gesendet. Unabhängig davon, in welcher z-Position die Drohne sich befindet, landet sie an der Startposition wieder. Nach dem Landen tritt der Autopilot in das Low-Battery-Modus für 3 s. Anschließend werden die Motoren ausgeschaltet.
Für die If-Bedingungen muss der neue Detektierungswert als alter Wert gesetzt und der aktuelle Wert in der neuen Variable gespeichert.

Nun um die Funktionalität des Programms zu prüfen, wird der catkin_workspace kompiliert. Weiterhin wird der Permission-Befehl für die Verwendung der UART-Schnittstelle eingegeben und die px4.launch-Datei gestartet. Anschließend wird die Takeoff_Land_-node-Datei aufgerufen.
Als Ergebnis ist festzustellen, dass die Daten zwischen dem RPI 3 und FC erfolgreich ausgetauscht werden: Der Autopilot stellt sich in das Offboard-Modus ein und die Motoren schalten sich an. Fernerhin werden die Befehle in Abhängigkeit von der Apriltags-Erkennung von dem FC angewendet.

Nun wird das C++-Steuerprogramm bei Einrichtung des FC mit MP ebenso als Flussdiagramm dargestellt. Dies beinhaltet die gleichen Initialisierungsschritte sowie die selbe Steuerung mittels Apriltags wie bei dem oben erläuterten Flussdiagramm, wobei hier der Offboard-Modus Guided_nogps-Modus heißt und das Modus-Setzen sowie das Motor-Ein- und Ausschalten nicht automatisiert werden und über die Funke erfolgen.

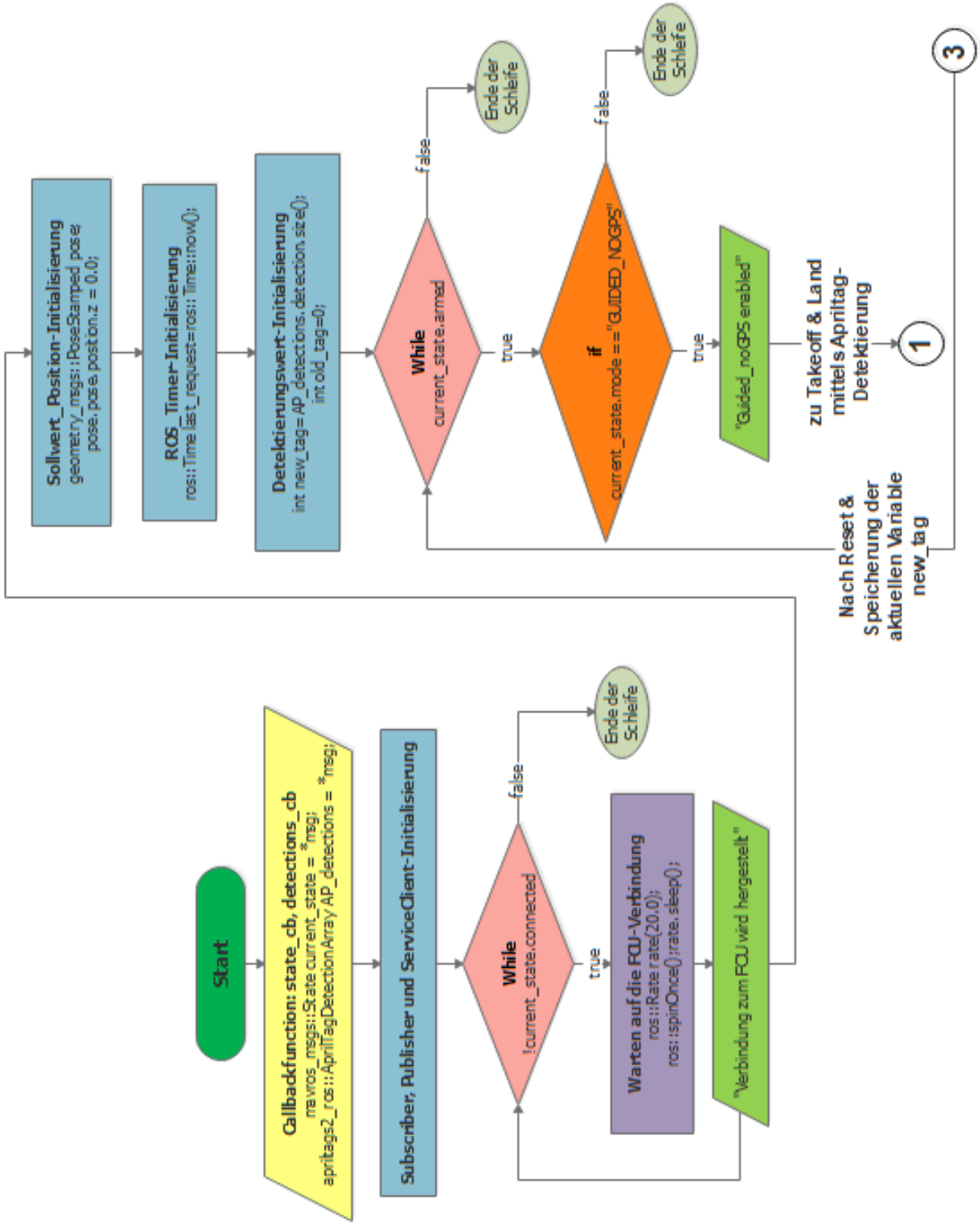


Abbildung 5.8: Initialisierung der Variablen

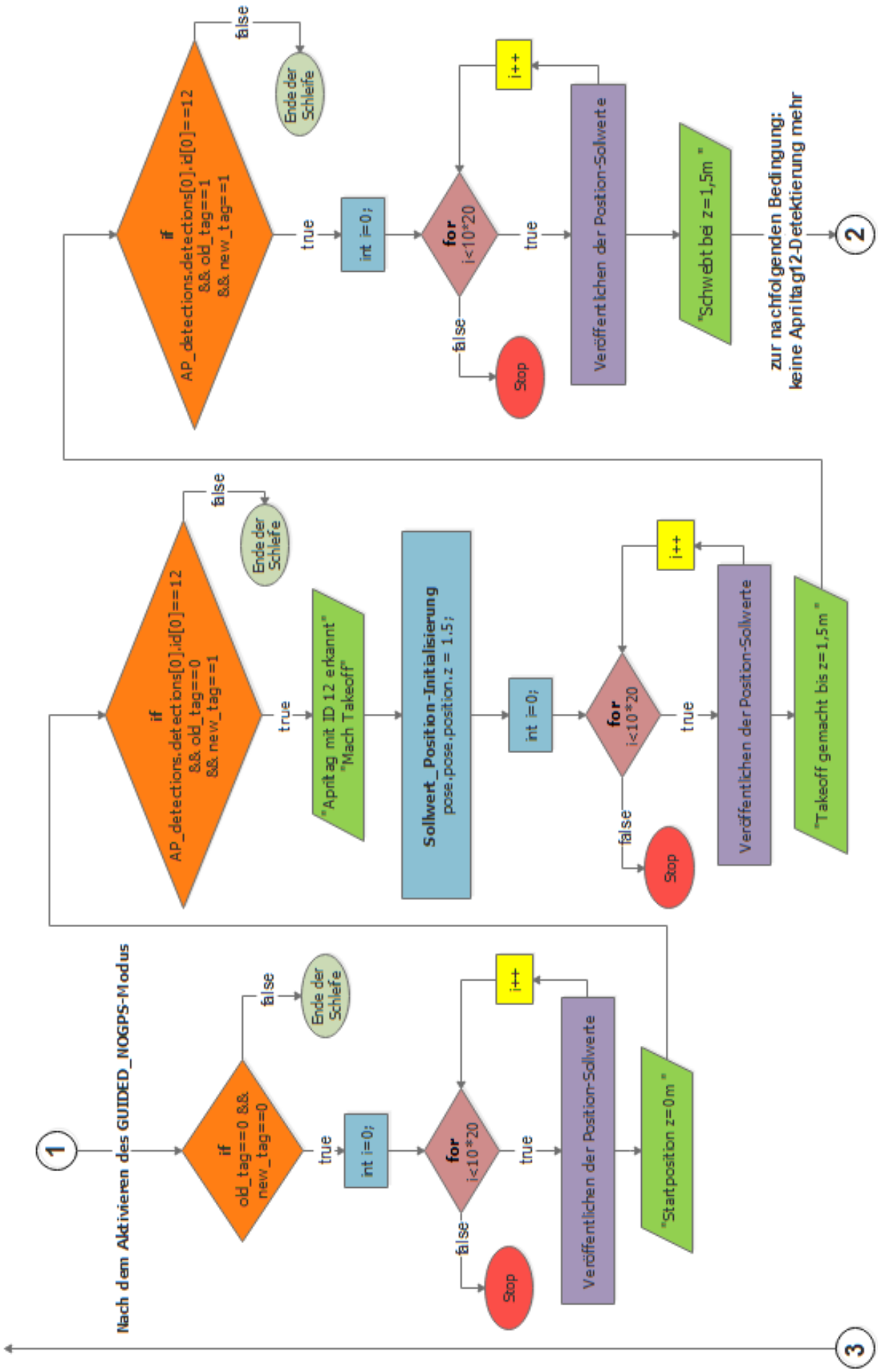


Abbildung 5.9: Land durch Detektierung des Apriltags-ID12

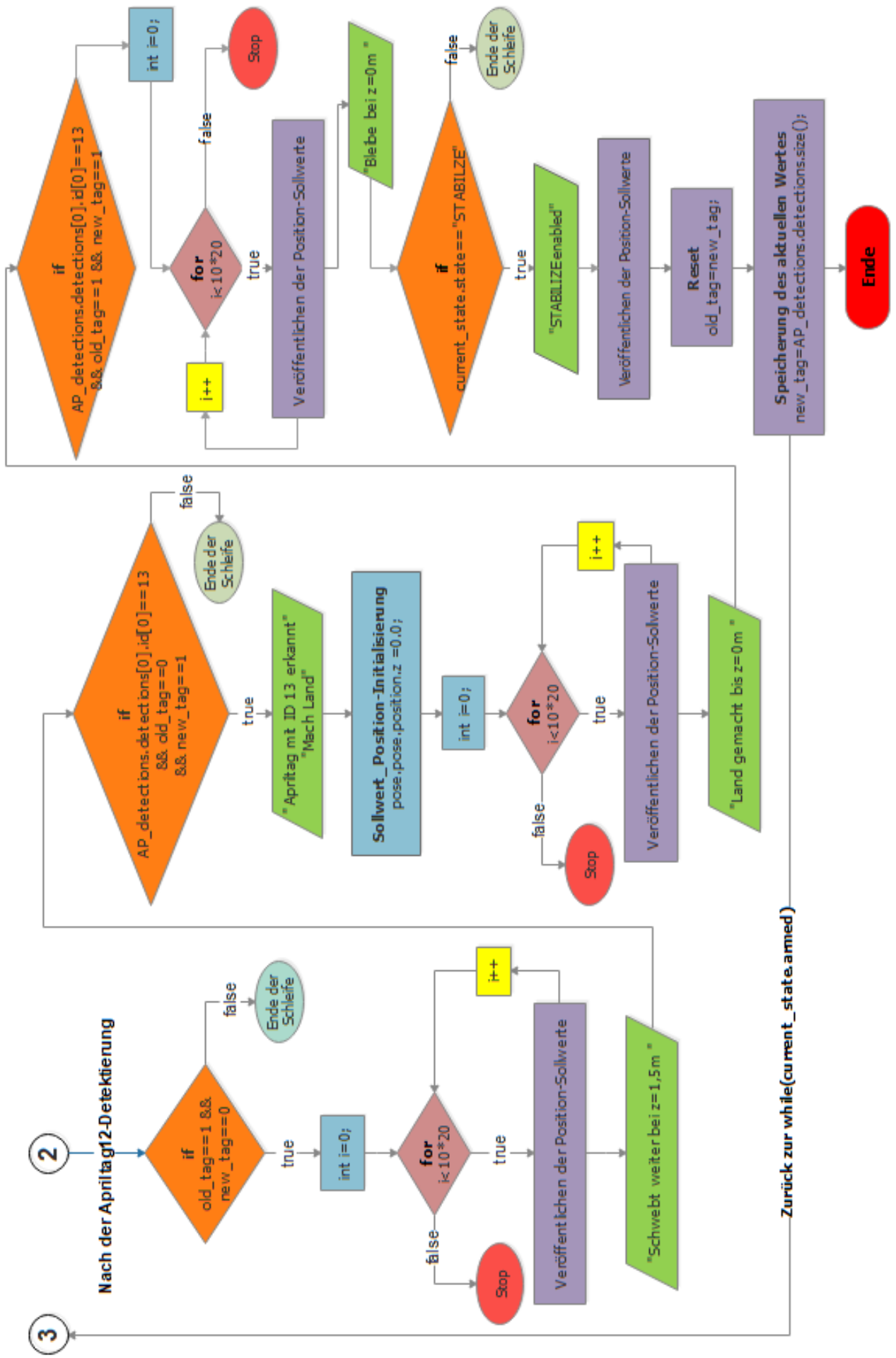


Abbildung 5.10: Land durch Detektierung des Apriltags-ID13

6 Ergebnisse

In dem letzten Abschnitt wird der Flugtestaufbau beschrieben und die Ergebnisse des Flugverhaltens grafisch dargestellt.

6.1 Flugtest

Unter einem Netz in dem Labor des Regelungs- und Steuerungstechnik Lehrstuhls der Universität Siegen wird die Drohne auf das Flugverhalten sowie auf die Regelreaktion überprüft. So wird sie in Stabilize-Mode geflogen, was bedeutet, dass die Sollwerte für Gas, Pitch-, Roll- und Yaw-Bewegungen aus der Funke vorgegeben werden.

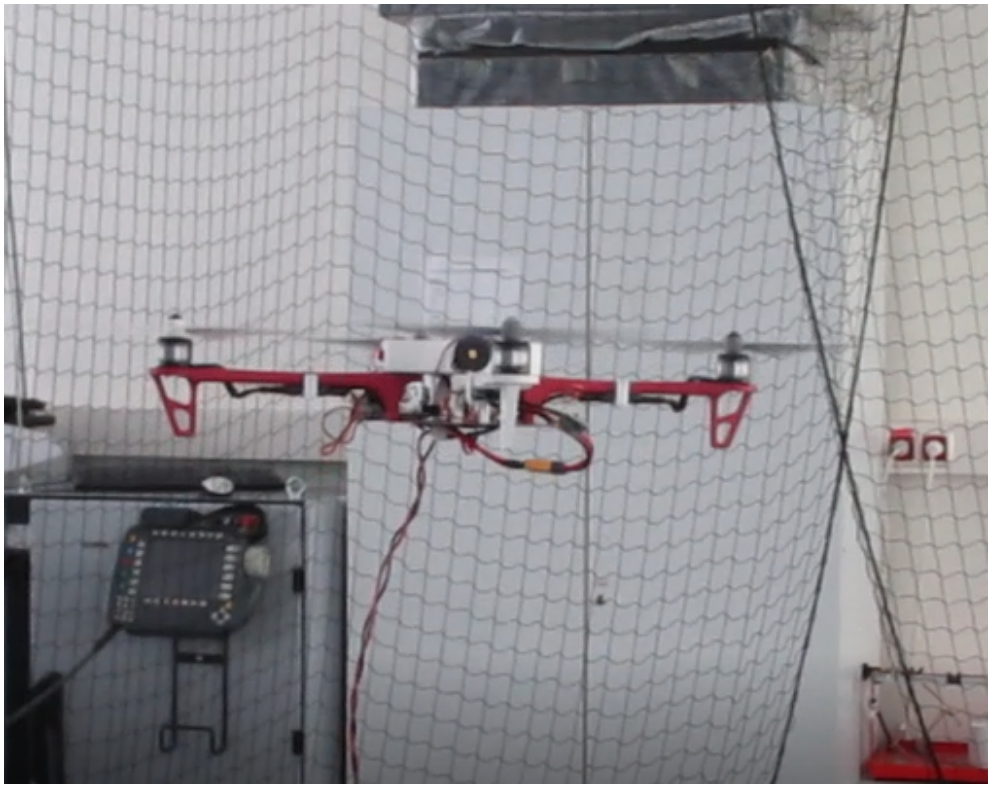


Abbildung 6.1: Flugtest

Zunächst werden die Motoren eingeschaltet und es wird wenig Gas gegeben, bis der Sättigungspunkt der Beschleunigungskurve, der bei 0,65 liegt, erreicht wird. Daraufhin wird der Gas schrittweise erhöht, bis der Quadcopter sich abhebt und seine Lage regelt, bei der er gerade in der Luft liegt. Für ein Schweben auf einer höheren X/Y-Ebene wird der Gaswert erhöht. Nun wird die Drohne um verschiedene Achsen rotiert und in verschiedene Richtungen in lineare Bewegung gebracht. Um die Drohne landen zu lassen, wird der Gas allmählich verringert, bis sie am Boden gut ankommt.
Für eine genauere Analyse des Regelverhaltens, werden die Ist- und Sollwerte der Roll-, Pitch- und Yaw-Rotationen in MP geplottet und in Vergleich gesetzt.

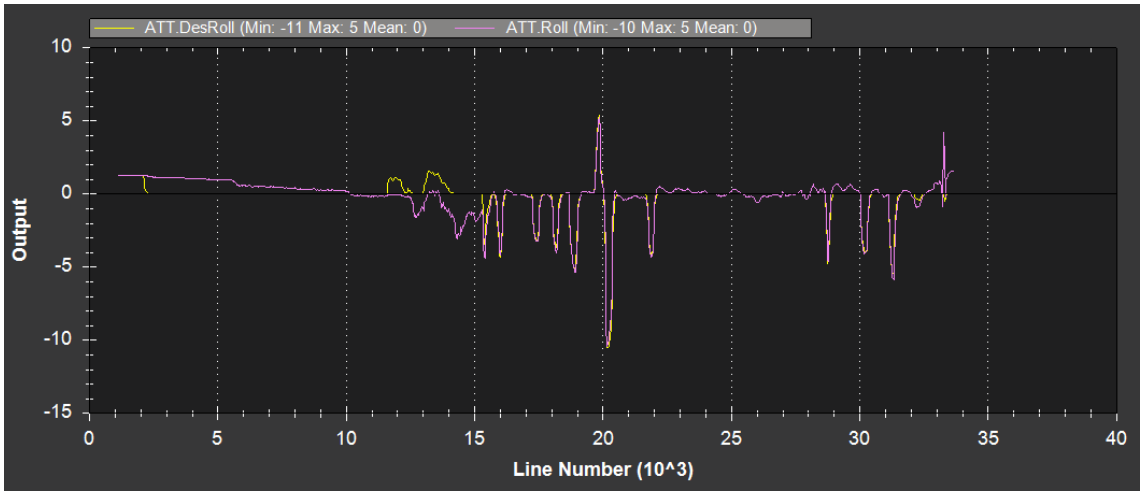


Abbildung 6.2: Soll-/Istwert-Vergleich der Roll-Bewegung

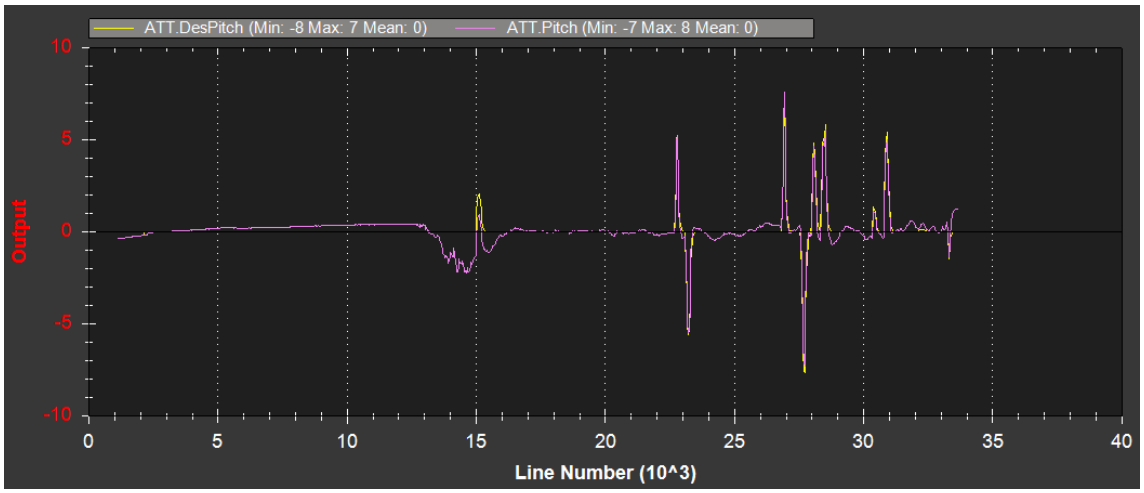


Abbildung 6.3: Soll-/Istwert-Vergleich der Pitch-Bewegung

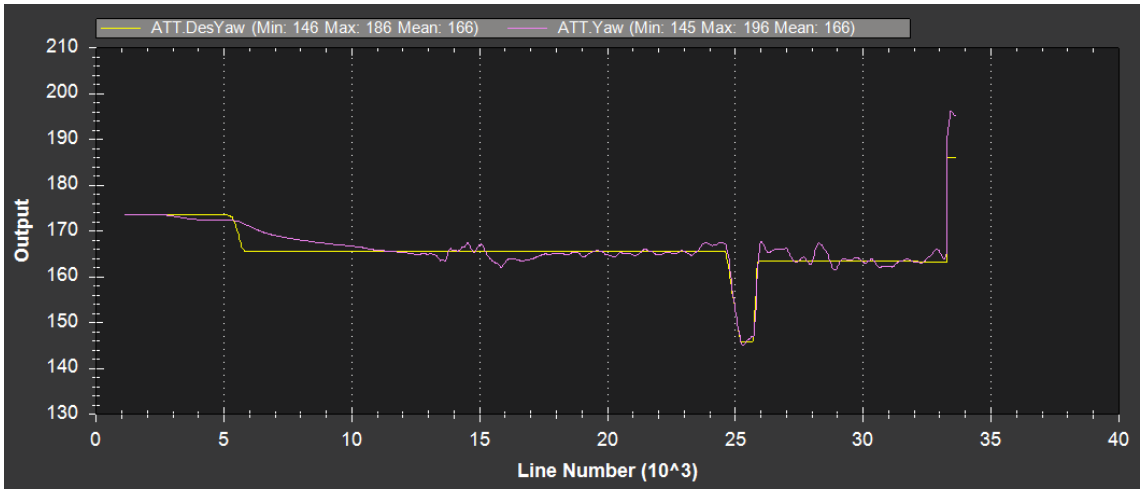


Abbildung 6.4: Soll-/Istwert-Vergleich der Yaw-Bewegung

Einführen von ungewollten Roll-, Pitch- und Yaw-Bewegungen als Störgröße:

Um sicher einzugehen, dass der Quadcopter beim Auftritt von Störungen wie z. B. ungewollte Bewegungen sich regelt und seine Lage beibehält, werden zwei weiteren Flugtests durchgeführt.

Ein Seil wird an einem Bein des Quadcopter-Landegestells verbunden. Während die Drohne bei konstanten Rotation-Sollwerten schwebt, wird der Seil nach unten gezogen, um eine Roll- und Pitch-Bewegung zu erzeugen. Weiterhin wird er seitlich für eine ungewollte Gier-Bewegung gezogen. Abschließend werden ebenfalls Roll-, Pitch- und Yaw-Ist- und -Sollwerte geplottet (Abbildung 6.5, 6.6 und 6.7), um das Regelverhalten bzw. die Regelreaktion auf eine plötzlich auftretende Störgröße zu interpretieren.

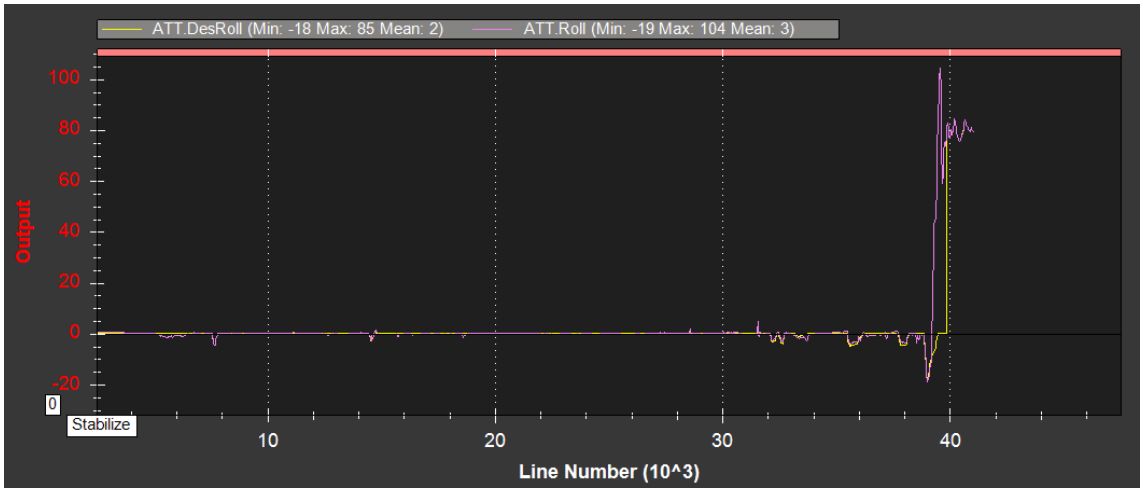


Abbildung 6.5: Soll-/Istwert-Vergleich der Roll-Bewegung

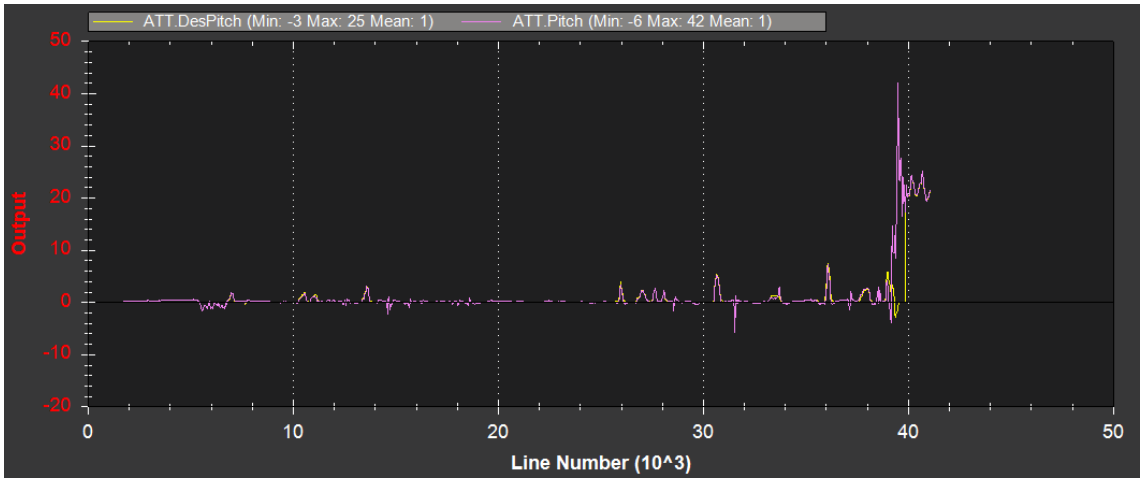


Abbildung 6.6: Soll-/Istwert-Vergleich der Pitch-Bewegung

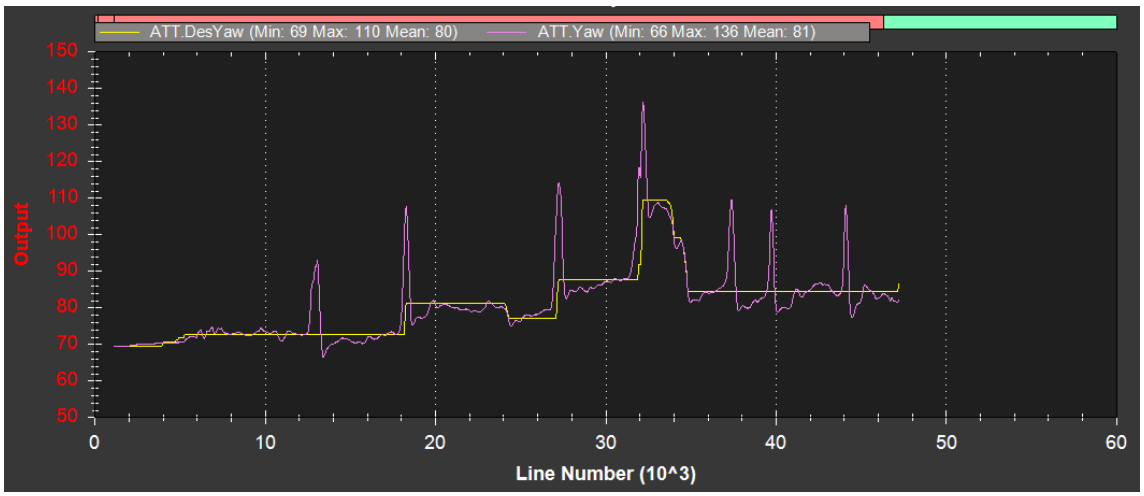


Abbildung 6.7: Soll-/Istwert-Vergleich der Yaw-Bewegung

7 Diskussion und Interpretation der Ergebnisse

Als Fazit zum Flugtest und zu den Soll-/Istwert-Plots wurden folgende Punkte angemerkt:

- Keine Schwingungen beim Abfliegen, Fliegen und Landen.
- Quadcopter fliegt ruhig in der Luft.
- Sollwerte aus der Fernsteuerung werden in optimaler Zeit gesetzt.
- Attitude-Regler arbeitet hervorragend auch bei eingeführten Störungen.
- Quadcopter fliegt stabil und gerade, wobei er nicht an einer gewissen Höhe bleibt.

An dieser Stelle ist zu erwähnen, dass eine Höhenregelung erforderlich ist, die das Flugverhalten noch optimieren würde. Dafür werden Abstandssensoren wie Barometer benötigt, um den Höhen-Istwert für den Altitude-P-Regler zu ermitteln. Die Höhenregelung dient zu einem geraden Schweben an der horizontalen Ebene sowie einem sicheren Abfliegen und Landen. Weiterhin ist diese für Steuerungen in Automodi bzw. bei Outdoor-Navigation unerlässlich und wird vorausgesetzt. So kann die Drohne beispielsweise im Offboard-/Guided_nogps-mode bei der Steuerung mittels der Apriltags-Detektierung mit einem passenden Gaswert einen Takeoff bis zu der vorgegeben Z-Position schaffen, bei der sie gerade schwebt und zwar ohne hin- und her-Bewegung entlang der Z-Achse. Für den Landen-Vorgang wird sie ebenso automatisch passend angetrieben.

Nach dem Testen der Drohne durch die Steuerung mittels der Funke sowie nach der Ausführung des auf dem RPI 3 geschriebenen Steuerprogramms wurde die Funktionalität des PixHawk-Autopilots bestätigt, dass er fähig ist, die Lage des Quadcopters im Manual/Stabilize-Modus zu regeln und im Offboard/Guided_nogps-Modus anhand der Daten, die sich aus dem optischen Erkennungsvorgang herleiten, Befehle zum Ansteuern der Motoren anzuwenden.

Literaturverzeichnis

[1] Quadcopter-Konfigurationen
http://ardupilot.org/copter/docs/connect-escs-and-motors.html

[2] Physikalisches Modell des Quadcopters
https://www.dji.com/newsroom/news/dji-releases-flamewheel-f330-arf-kit

[3] Übertragungsfunktion des PID-Reglers
Digitale Regelungstechnik-Skript, Prof. Roth

[4] Beschleunigungssensor
https://electreeks.de/assembly-centre/anleitungen/gyro2/

[5] Gyroskop
https://electreeks.de/assembly-centre/anleitungen/gyro2/

[6] Druckformel in Abhängigkeit von der Höhe
Prozessmesstechnik-Skript, Prof. Gronwald

[7] Flugcontroller-Typen
https://www.losmuchachos.at/multicopter/multicopter-flight-control-die-qual-der-wahl/

[8] PixHawk-Flugcontroller
http://pixhawk.org/

[9] Pixhawk-Flugcontroller
https://docs.px4.io/en/flight_controller/HKPilot32.html

[10] Safety-Switch-LED
https://docs.px4.io/en/flight_controller/pixhawk.html

[11] Lipo-Batterie
http://lipo-welt.de/lipo-akku/

[12] Batterie-Typen
https://www.reichelt.de/reicheltpedia/index.php/Batterie

[13] Propeller
http://www.kopterforum.de/tutorials/article/1-der-propeller-bei-multi-coptern/

[14] Robot Operating System
https://www.generationrobots.com/blog/de/2016/03/ros-robot-operating-system/

[15] Apriltag mit der ID-Nummer 12
https://www.dotproduct3d.com/uploads/8/5/1/1/85115558/apriltags1-20.pdf

[16] Verkabelung des Quadcopters
https://docs.px4.io/en/assembly/quick_start_pixhawk.html

[17] Raspberry Pi 3-Pixhawk-Verbindung
http://ardupilot.org/dev/docs/raspberry-pi-via-mavlink.html

[18] Download der QGroundControl-Bodenstation
http://qgroundcontrol.com/downloads/

[19] Download der Mission Planner-Bodenstation
http://ardupilot.org/planner/docs/mission-planner-installation.html

[20] Orientation des Autopilots über QGC
https://docs.px4.io/en/config/flight_controller_orientation.html

[21] Kompasskalibrierung über QGC
https://docs.px4.io/en/config/compass.html

[22] ACC-Kalibrierung über QGC
https://docs.px4.io/en/config/accelerometer.html

[23] Gyroskop-Kalibrierung über QGC
http://docs.px4.io/en/config/gyroscope.html

[24] Level Horizon-Kalibrierung über QGC
https://docs.px4.io/en/config/level_horizon_calibration.html

[25] Dx6i-SPEKTRUM-Funksender
https://www.amazon.de/Spektrum-DX6i-2-4GHz-DSM-X-Sender/dp/B0042YR1O8

[26] RC-Kalibrierung über QGC
https://docs.px4.io/en/config/radio.html

[27] Flugmodi
https://docs.px4.io/en/getting_started/flight_modes.html

[28] Einstellen der Rahmenklasse und des Rahmentyps über MP
http://ardupilot.org/copter/docs/frame-type-configuration.html

[29] Beschleunigungssensorkalibrierung über MP
http://ardupilot.org/copter/docs/common-accelerometer-calibration.html

[30] Kompasskalibrierung über MP
http://ardupilot.org/copter/docs/common-compass-calibration-in-mission-planner.html

[31] Sender-Kalibrierung über MP
http://ardupilot.org/copter/docs/common-radio-control-calibration.html

[32] Output-Pins des PixHawk-Autopilots
http://ardupilot.org/copter/docs/connect-escs-and-motors.html

[33] X-Konfiguration des Quadcopters
http://ardupilot.org/copter/docs/connect-escs-and-motors.html

[34] Motortest über MP
http://ardupilot.org/copter/docs/connect-escs-and-motors.html

[35] Reihenflge der Motoren
http://ardupilot.org/copter/docs/connect-escs-and-motors.html

[36] Raspberry PI 3 Model B+
https://ch.rs-online.com/web/p/entwicklungskits-prozessor-mikro-controller/1365473/

[37] ROS-Installation auf dem RPI3
http://wiki.ros.org/kinetic/Installation/Ubuntu/

[38] RPI3-Pins-Konfiguration
https://raspberry.tips/raspber-rypi-einsteiger/uart-auf-dem-raspberrypi-nutzen
https://www.raspberrypi.org/documentation/configuration/uart.md

[39] Mavros-Paket-Installation
https://dev.px4.io/en/ros/mavros_installation.html

Anhang

In diesem Abschnitt werden sowohl einige QGC- als auch MP-Parameter aufgelistet. Restliche und nicht aufgeführte Parameter sind auf Standardwerte gesetzt.

1) Parameter des PixHawk-Autopilots im QGroundcontrol

Fluggerätetyp:

Fluggerätetyp = Quadrotor X
Fahrzeug = DJI Flame Wheel F450

Sicherheit:

Low Battery Failsafe = Warning
RC Loss Failsafe = Land Mode
RC Loss Timeout = 0,5
Data Link Loss Failsafe = Land Mode
RTL Climb To = 30,0
Loiter Alt = 5,0
Land Delay = 5

Sensoren:

Kalibrierung nach der Anleitung im QGC, hierbei ergeben sich einige Werte aus der Kalibrierung

Sender-Kalibrierung:

RC_Chan_CNT = 6
RC_Map_Throttle = Channel 1
RC_Map_Roll = Channel 2

RC_Map_Pitch = Channel 3

RC_Map_Yaw = Channel 4

RC_Map_AUX1 = Channel 6

RC_Map_ARM_SW = Channel 4

RC_Map_FLT_Mode = Channel 5

RCx_DZ = $0\,\mu s$

RCx_max = $2000\,\mu s$

Throttle:

RC1_min = $1100\,\mu s$

RC1_Trim = $1100\,\mu s$

Roll, Pitch, Yaw:

RCx_min = $1000\,\mu s$

RCx_Trim = $1500\,\mu s$

Flugmodi:

Flug Modus 3 = Manual

Flug Modus 4 = Offboard

Flug Modus 6 = Land

PWM_Outputs:

PWM_MAX = $2000\,\mu s$

PWM_MIN = $1140\,\mu s$

Multicopter Attitude Control:

MC_Roll_P = 4,50

MC_Rollrate_P = 0,20

MC_Rollrate_I = 0,150

MC_Rollrate_D = 0,0070

MC_Pitch_P = 4,50

MC_Pitchrate_P = 0,20

MC_Pitchrate_I = 0,150

MC_Pitchrate_D = 0,0070

MC_Yaw_P = 2,80
MC_Yawrate_P = 0,20
MC_Yawrate_I = 0,10
MC_Yawrate_D = 0,00

Multicopter Position Control:
MPC_MANTHR_MIN = 0%

Return-Mode:

RTL_Descend_ALT = 5,0
RTL_Land_Delay = 5,0
RTL_Return_ALT = 30,0

System:

sys_Companion = Companion Link 921600 (TELEM2)
sys_FMU_Task = Enable

Mission:

Nav_ACC_RAD = 2,0
Nav_DLL_ACT = Land Mode
Nav_RLL_ACT = Land Mode

Commander:

COM_ARM_EKF_AB = 0,0024m/s
COM_DISARM_LAND = 3
COM_FLTMODE3 = Manual
COM_FLTMODE4 = Offboard
COM_FLTMODE3 = Land

2) Parameter des PixHawk-Autopilots im Mission Planner

Rahmentyp:

Frame_Class = 1: Quad
Frame_Type = 1: X
BRD_Type = 2: PixHawk Board Type

Orientation:

AHRS_ORIENTATION = 1: Custom
AHRS_CUSTOM_ROLL = 0
AHRS_CUSTOM_PITCH = 0
AHRS_CUSTOM_YAW = 0

FaileSafe:

RC-Steuerung: Enabled always Land
FS PWM = 1073
FS GCS nicht aktiviert

Sensoren:

Kalibrierung nach der Anleitung im MP, hierbei ergeben sich einige Werte aus der Kalibrierung.

Sender-Kalibrierung:

RCMAP_Pitch = Channel 2
RCMAP_Roll = Channel 1
RCMAP_Throttle = Channel 3
RCMAP_Yaw = Channel 4
FLTMODE_CH = Channel 5
RC1_DZ = $20\,\mu s$
RC1_MAX = $1921\,\mu s$
RC1_MIN = $1085\,\mu s$
RC1_REVERSED = 0
RC1_TRIM = $1504\mu s$
RC2DZ = $20\,\mu s$

RC2_MAX = 1919 μs
RC2_MIN = 1085 μs
RC2_REVERSED = 0
RC2_TRIM = 1504μs
RC3_DZ = 30 μs
RC3_MAX = 1922 μs
RC3_MIN = 1073 μs
RC3_REVERSED = 0
RC3_TRIM = 1073μs
RC4_DZ = 20 μs
RC4_MAX = 1910 μs
RC4_MIN = 1085 μs
RC4_REVERSED = 0
RC4_TRIM = 1499 μs

Flugmodi:

FLTMODE3 = 0: Stabilize
FLTMODE4 = 20: Guided_nogps
FLTMODE6 = 9: Land

Motoren:

MOT_PWM_MAX = 2000 μs
MOT_PWM_MIN = 1000 μs
MOT_PWM_TYPE = 0: Normal
MOT_Safe_Disarm = 0: PWM enabled while disarmed
MOT_THST_HOVER = 0,35
MOT_SPIN_ARM = 0,1
MOT_SPIN_MAX = 0,95
MOT_SPIN_MIN = 0,15
MOT_Spool_Time = 0,5 s
MOT_THST-EXPO = 0,65
Arming_check = 3540: Barometer, Compass, RC-Channels, Board Volta-

ge, Battery Level, Logging available, Hardware Safety Switch
FS_crash_check = 0

Servo Output:

Servo1_Function = 33: Motor 1
Servo2_Function = 34: Motor 2
Servo3_Function = 35: Motor 3
Servo4_Function = 36: Motor 4
Servo_MAX = 2000 μs
Servo_MIN = 1000 μs
Servo_TRIM = 1500 μs

ESC Calibration:

ESC_Calibration = 0: Normal
ESC_Type = 0: Normal
Output PWM Min = 1000 μs
Output PWM Max = 2000 μs

PID-Regelung:

Stabilize Roll P = 4,5
Rate Roll P = 0,135
Rate Roll I = 0,090
Rate Roll D = 0,0036
Stabilize Pitch P = 4,5
Rate Pitch P = 0,135
Rate Pitch I = 0,090
Rate Pitch D = 0,0036
Stabilize Yaw P = 4,5
Rate Yaw P = 0,180
Rate Yaw I = 0,018
Rate Yaw D = 0,000

System:

SERIAL2_BAUD = 921: 921600 (TELEM2)

3) Einstellungen im DX6i-Sender

Für die DJI F450-Drohne sind folgende Parameter zu setzen:

D/R & Expo:

AILE 0 100% + INH
ELEV 0 100% + INH
RUDD 0 100% + INH
AILE 1 42% + INH
ELEV 1 42% + INH
RUDD 1 35% + INH

TRAVEL ADJ:

GEAR (UP) 86%
GEAR (DOWN) 36%
FLAP (UP) 0%
FLAP (DOWN) 100%

MIX 1:

GEAR - GEAR ACT
RATE D - 93% U 0%
SW ELE D/R TRIM ACT

REVERSE:

THRO-N RUDD-R
AILE-R GEAR-R
ELEV-N FLAP-N

THRO CUT:

POSITION-INH

WINGTAILMIX:

DUALAILE INH
V-TAIL INH
ELEVON INH

D/R COMBI:

D/R SW: RUDD

TIMER:

10:00