Development & Construction of an Autonomous Path-Following Drone

Eduard Scherer

December 20, 2024

Contents

1 Introduction **3**

2 Personal Motivation **3**

3 Literature Review **3**
3.1 General Software Considerations . . . 3
3.1.1 Open Source Software . . . 3
3.1.2 Flight Softwares . . . 3

4 Methodology **4**
4.1 Drone Overview . . . 4
4.2 parts . . . 5
4.2.1 Fc . . . 5
4.2.2 ESC . . . 5
4.2.3 Motor . . . 5
4.2.4 Battery . . . 6
4.2.5 GNSS(Global Navigation Satellite System)/Compass . . . 7
4.2.6 Radio/Transmitter . . . 7
4.2.7 Smoke Stopper . . . 8
4.2.8 Problems with the Parts . . . 8
4.2.9 Propellers and Battery Charger . . . 9
4.2.10 Data Transfer Protocols . . . 9
4.3 mechanical work . . . 9
4.4 ardupilot . . . 9
4.4.1 Ground Station . . . 9
4.4.2 Firmware Installation . . . 10
4.4.3 GPS Connection . . . 10
4.4.4 Receiver/Transmitter . . . 12
4.4.5 Motor Test . . . 14
4.4.6 Compass Calibration . . . 14

4.4.7 Road to First Flight 14
4.4.8 RaspberryPi Setup 16
4.4.9 MAVProxy Installation 17
4.4.10 Dronekit 17

5 Results **21**

6 Discussion and Outlook **21**

7 Conclusion **21**

8 References **21**

Bibliography **21**

9 Table of Figures **24**

1 Introduction

In the world of today where technology is advancing faster than ever before this "Maturitätsarbeit" tries to shine a light into the corner of self-built and autonomous drones.

However it first needs to be defined what a drone even is. According to the Cambridge Dictionary a drone is: *an aircraft that does not have a pilot but is controlled by someone on the ground* [14]. In this work drone will refer to a quadcopter which is a drone with four motors and propellers and not a conventional unmanned aerial vehicle (UAV), which is mostly used for military grade drones.

In current conflicts the meanings of UAV and quadcopters have become more entangled than ever. The typical hobbyist drone is now used widely and efficiently to target enemy forces. In the Ukraine war drones are one of the most effective weapons to use against the Russian aggressor for example they use drones with thermite canisters attached to them to target the enemies under tree covers [19]. Since the war began the drones also became more and more autonomous, meanwhile the drone pilots are able to only lock onto the target and let the drone follow and explode near it [17]. This increased use in war has also had and impact in the hobby side of the project. The most used electronic speed controller (ESC) firmware BLHeli_32 ceased its operation due to laws being passed in Norway that penalised firms that are not able to verify that their products are not used in wars [16]. Which is really difficult to do regarding that both sides of the Ukraine war are just buying the controllers online.

2 Personal Motivation

3 Literature Review

3.1 General Software Considerations

3.1.1 Open Source Software

An important part of this "Maturitätsarbeit" is open source software. Open source software is software which everybody can download and develop in their own time. The main benefit to closed software is that it is free for everyone to have, but it comes with the downside that if the developing team loses interest in it and it gets outdate when it needs to work with other things

3.1.2 Flight Softwares

There are three main softwares to consider when it comes to drones Betaflight, INAV and Ardupilot. All of them are open source. Multiwii is the origin of Betaflight and INAV. Multwii was Arduino based and then upgraded to

Baseflight to be able to use the STM32 chips. Then it was forked to Cleanflight, which was later forked again into Betaflight and INAV[5].

The following three paragraphs are mostly based on [9] and [10].

Betaflight is in general the go-to option for first person view drones, commonly known as FPV drones, for either filming or racing. It is the most beginner friendly out of the three, because it has a large community, which results in a wide range of tutorials. When a new board comes out it is normally made to be used with Betaflight. However Betaflight lacks the option for different types of vehicles and generally the automated features are less developed compared to the other two.

INAV offers basic autonomous flight using waypoints and automated landing. It does not only support quadcopters, but also boats, rovers, planes and wings. It has a similar interface to Betaflight so switching from one to the other is easier than switching to Ardupilot.

Ardupilot basically offers everything the other two have to offer and more for example Submarines and VTOLs. It is not commonly used with FPV, yet you can if you want to. It is more complicated to get into, but as a trade off you will be able to customize everything to your will. It is also the only option for a companion computer like the RaspberryPi. A few years ago it was really expensive to start, because it only supported the Pixhawk family of flight controllers which cost several hundred Franks a piece. However in recent years it has began to support more and cheaper flight controllers.

4 Methodology

4.1 Drone Overview

If you want to fly a drone you have two options available either you buy one or you built one. The second option is for people who are interested in tinkering with their electronics until they work, not only the better option but also the cheaper one.

The main parts needed for a drone are the flight controller (Fc), the ESC, the receiver, the battery, servos and of course the frame, in addition you can also add a global positioning system (GPS), LEDs and more. The flight controller and electronic speed controller are referred to as Fc and ESC. The Fc runs the chosen software and controls every other part in one way or another. It is directly connected to the ESC, which controls the motors and is connected to the battery. The Fc also communicates with the receiver and GPS if there is one. In my case there will also a RaspberryPi which also connects to the Fc.

4.2 parts

4.2.1 Flight Controller

There are two different kind of flight controller the All-in-One (AIO) and just a Fc. The AIO is not only a Fc but also the ESC in one board. This has the advantage of only needing one board instead of two or five. However if only parts of the ESC or the Fc are damaged you need to replace the whole board which is more expensive than replacing only the Fc or ESC.

Betaflight and INAV both support a wide variety of Fcs compared to Ardupilot which is only supporting a very specific sample of boards[30]. They have the option of open and closed hardware. Because the open hardware Fcs are quite expensive, so I decided to go with a closed one. The chip used in Fcs is usually a STM32. There are multiple generations of it the mainly used ones are F4, F7 and H7. The F7 and H7 are much faster than the F4 chips so the decision was not that difficult to make. I then decided to go with the Kakute H7 v1.3 (MPU6000)[32] from Holybro, because it was ok in the price and available as a stack. It comes shipped with Betaflight so I will need to flash Ardupilot.

4.2.2 Electronic Speed Controler

There are two different kind of ESCs, 4in1 and single ESCs. If you use single ESCs you will need one for each motor instead of a single board for all of them. The advantage of 4in1 ESCs is that you do not need a power distribution board, because it is already incorporated in the ESC, and that it can come in a stack. A stack is a Fc and ESC mounted on top of each other. It normally comes with stack screws. The disadvantage is that if a part of the ESC is damaged you need to replace the whole board. What needs to be looked upon buying a ESC is that the peak current of the motor is not higher than the burst current of the ESC, because it could damage a two high current could damage the ESC.

My decision was to go with a 4in1 on a stack, because it is slightly cheaper and normally easier to wire. The only option with the Kakute H7 was the stack with the Tekko32 4in1 with a continuous current of either 50, 60 or 65 Ampere. I chose the 50A[38] one, because you do not need a high continuous current rating when flying rather slowly and the motors I chose have a peak current of around 42A.

At the time I bought the ESC it was still shipped with BLHeli32 which as mentioned has seized their operations. I could however flash AM32 onto it.

4.2.3 Motor

There are two types of motor brushed and brushless ones. The difference between the two types is that the brushed motors are mechanically driven and the brushless motors are electrically driven. Therefor brushless motors also need an ESC to function compared to brushed ones. Brushed motors are used in really

small drones with 1S batteries. However even in the small drones brushless motors are the more popular choice[3].

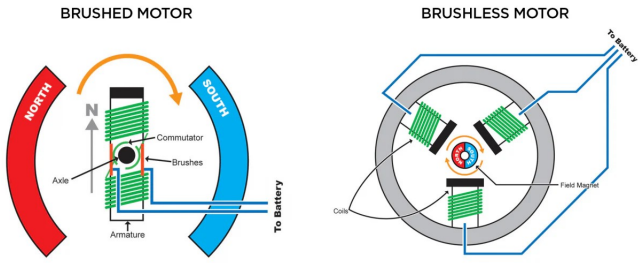

Figure 1: [2]

The numbers that are seen on the motors like 2207 are describing the stator of the motor itself. In the case of a 2207 that would be 22mm diameter and a height of 7mm. The usual stator sizes of 5 inch drones are either 2207 or 2306. There is also the KV value which has to be considered. The KV value is the number of revolutions per minute(rpm) a motor turns when one volt is applied. The lower the KV value the more efficient is the motor and the higher the KV value the more responsive it is. The KV value for a 5 inch drone with a 4S battery is between 2300 and 2800. I chose the Iflight Xing-E Pro 2207[41] with a KV value of 2450, because the IFlight-Xing motors are known to be of good quality.

4.2.4 Battery

There are two main types of batteries: lithium polymere (LiPo) and lithium-ion (Li-ion) batteries. LiPo batteries have a tendency to go up in flames. They have compared to the Li-ion batteries a much higher discharge rate, also denoted as C value, so they are well suited for racing drones. However Li-ion batteries have a higher energy density, which means that they can store more mAh with the same weight as LiPo batteries, so they are more common in long range flying. The batteries normally have multiple cells, for example a 4S LiPo batterie contains 4 cells. This is important, because the more cells you have the higher is the voltage and so you will need a motor with a lower KV value.

I first wanted to buy two Li-ion battery packs, however they are hard to get and much more expensive. The other option would have been to solder them together myself, but it requires some soldering skill, which I did not have then. So I decided to go with 4S LiPo batteries from the Tattu R-line with 120C and 2000mAh[37], because I want something that can fly longer than just three or four minutes. I chose the brand Tattu, because it was the only known brand that was on AliExpress from where I got all the other components, so it seemed

to be easier to also chose a battery from there.

4.2.5 GNSS(Global Navigation Satellite System)/Compass

There are two different kinds of GNSS[1] modules the normal GPS/Compass boards and real time kinematic(RTK) GPS. Usually the chips used in both the RTK GPS and the normal GPS are from Ublox. The RTK GPS can achieve a accuracy of 1cm by incorporating information correction data from a RTCM, short for radio technical commision for maritime[2]. However the correction data is either subscription based, is not guaranteed to cover all of the area or you need to build one yourself, which is quite complicated[35]. For a small drone it is also not worth it to have a RTK GPS that costs several hundred Franks instead of a GPS with a 2m circular error probable (CEP) which costs less than 30 Franks. Some of the GPS units also have a compass built-in so you will not need to buy an extra compass when using ArduPilot.

Explanation 4.1: circular error probable (CEP)

The CEP refers to how close to the real value the GPS normally is. So if a GPS has a CEP of 2m it is normally in the range of 2 meters of the correct value. [13]

I chose the Holybro Micro M10 GPS[33], because it comes from the same brand as the Fc and will be easier to connect. In addition to that the M10 chip is the newest version of Ublox chips and it would be senseless to buy an older version. It also is at least as good as other better-known GPS as the Matek M10Q[18] and comes with a built-in compass that is needed for ArduPilot.

4.2.6 Radio/Transmitter

The following information about transmitter protocols is based on two videos[7][20].

FrSky

There are three FrSky protocols that are not compatible with each other ACCST, ACCESS and FrSky R9 ACCESS. ACCST has a range of around 1-2 km in open air. The firmware is built into almost all radios. It has a good latency but not as good as the other options. ACCESS is the successor to ACCST and has about the same range, but a lower latency, however not a class leading one. FrSky R9 ACCCESS gives you a range of over 50km in ideal condition. Unfortunately, the FrSky ACCESS porotcols are only supported by FrSky radios. There is one problem with FrSky. It is quite a difficult to manage, because of the three protocols that are not compatible with each other.

[1] The GNSS is usually referred to as GPS even though GPS is only the American GNSS system

[2] It was first used for the positions of boats and other vessels

TBS Crossfire/Tracer and Immersion RC Ghost

TBS Crossfire has a range of over 100km and is easy to use. It is also well tested and stable and has a good latency. TBS Tracer can have also over 10km, but drains the battery when flying further away. It is more focused on racing due to the low-latency. Immersion RC Ghost as a range of over 10km. It is possible to switch between a low-latency for racing and a high-latency for long range flying.

ExpressLRS

ExpressLRS is the best in long range and in the combination of long range and latency. It has been shown by Wezley Varty[12][3] that it really can fly up to 100km[4]. It is together with mLRS the only two options who are open source.

mLrs

The main difference between ExpressLRS and mLRS is that mLRS has a higher latency to send larger data packages to your telemetry device. Which is something that is needed if you want to adjust something or get more data from your drone over the Mavlink protocol during the flight.

Transmitter/Radio

I've decided to go with the Radiomaster RP4TD ExpressLRS 2.4GHz True Diversity Receiver[21] on a recommendation of a friend. It also is compatible with mLRS if I want to switch later on.

I went with the Radiomaster Boxer[22] radio, because it is somewhat in the middle range from radios and seems to be quite reliable and has many knobs to customize.

4.2.7 Smoke Stopper

A part that stops the ESC from short circuiting due to wrongly soldered parts and can save you quite a lot of money. There are two groups of smoke stoppers one that you buy and gets destroyed when the ESC short circuits instead of the ESC. There is another category that does not destroy itself and there are also some that you can solder together on your own[1].

4.2.8 Problems with the Parts

There were two problems that arose one less severe than the other. One problem was that my ordered Kakute-Tekko stack did not include stack screws, which

[3] Due to him being fined by the Australian government he took down his own video, so only a copy from an other YouTube channel exists.

[4] There are two version a 900MHz and a 2.4GHz and the main difference is that the 2.4GHz only at least over 30km but unlikely over 100km, but it has the lowest latency compared to any other protocol.

they normally do. Stack screws are just screws that you can use to mount the stack onto the frame. So I needed to purchase them separately.

The more severe problem I ran into was that the batteries from AliExpress were first held by the swiss border control and then I received two insect traps instead of batteries. Luckily I ordered one of the same batteries from Conrad, because the other two took too long to deliver.

Two months later when my father decided open the insect traps. To our shock the LiPo batteries were inside of the insect traps at the bottom. And in hindsight it was also obvious that they were in there, because it had the numbering 3 4s 2000 on top which stands for version 3 of the 4s batteries with 2000 mAh. However, we dismissed the numbering on top as just some random numbers put there.

4.2.9 Propellers and Battery Charger

For the propellers(props) and battery chargers I went of the recommendations from AOS-RC and FPVknowitall. For the props I chose the Foxeer Donut 5145[23] and the HQ 5x4.3x3 V1S[24]. And for the battery charger I went with the cheapest option the 608 AC Lipo Battery Charger[25].

4.2.10 Data Transfer Protocols

SPI

Uart

I2C

4.3 mechanical work

4.4 ardupilot

The following is based on the ArduPilot copter documentation[28].

4.4.1 Ground Station

To configure ArduPilot there are multiple softwares so called ground control stations(GCS) required. They are normally ground-based and can transmit data via with a wireless telemetry device or USB cable. With the telemetry device they can also control the drone from the ground and alter the route it is autonomously flying.

The most widely used GCS is Mission Planner(MP)[34] it is widely used, but runs only on Windows and Mac OS. It also has a wiki which helps you install it. I will use it for the first-time configuration of my drone.

Another GCS is MavProxy it is based on Python and is only for the Linux operation system(OS). Which is the OS I will use on the RaspberryPi companion computer.

4.4.2 Firmware Installation

For the first time installation I need to flash the Kakute H7 with ArduPilot, because it is shipped with Betaflight as mentioned earlier on. For that I downloaded the ArduPilot firmware[29] for the Kakute H7. Afterwards the STM32CubeProgrammer[36], to flash the firmware onto the Fc. I connected the Fc in device firmware updgrad (DFU) mode directly with the computer using a USB cable. Then selected the USB port with which the Fc is connected and flashed the Fc. A reboot to get out of the DFU mode was required before connecting the Fc to MP and saw the first Yaw measurements. The progress of flashing the firmware was straight forward which will not be the case for the rest of the configuration.

Explanation 4.2: device firmware upgrade (DFU)

The DFU mode is the mode, which allows the user to upload new firmware to the Fc. It is normally accessible through a button which needs to be pressed, while connecting the battery.

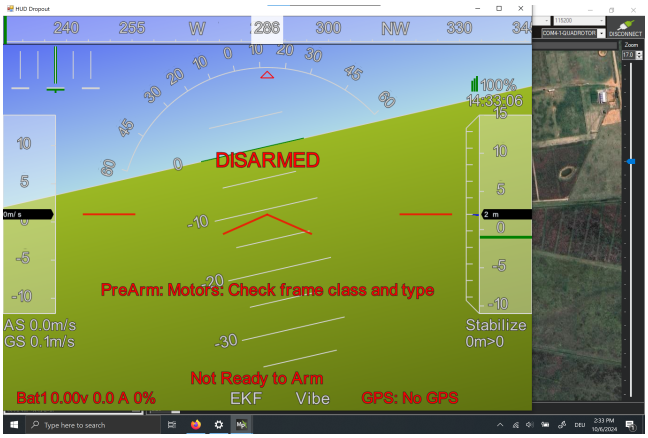

Figure 2:

4.4.3 GPS Connection

When I tried to get a GPS connection a No GPS message appeared. Changing the `GPS_Type = 2` for the Ublox GPS did not change the No GPS error message.

Even though I was certain that the GPS worked well, because the compass of the GPS was already detected in the compass calibration tab.

I then went outside, away from metal tables and as far away from my computer that was possible with the USB cable it still did not show up in MP. I also

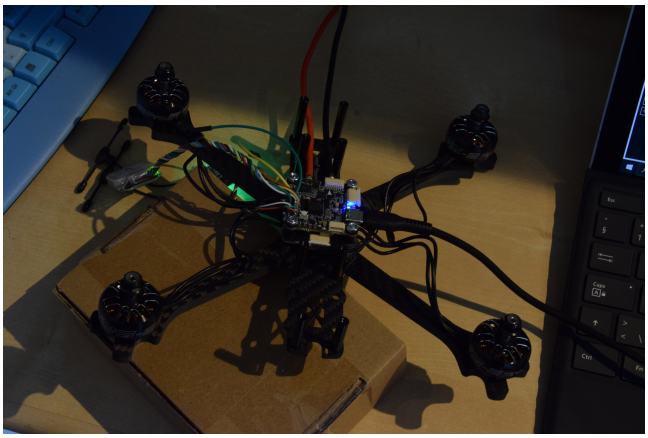

Figure 3:

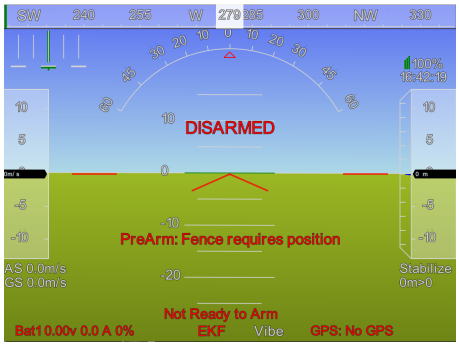

Figure 4:

checked the soldering and if the cables from the GPS to the Fc are connected correctly and both seemed fine. The GPS itself was not damaged, because the blue led was blinking constantly which means it is connecting to a GNSS.

The problem was that the `Serial3_Protocol`(Which is for the Uart3, which will be used for the connection to the RaspberryPi) was set to 5 which means GPS, however by being on 5 it blocked the Uart4, to which my GPS was really connected. After disabling the Uart3 it finally worked.

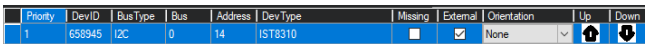

Figure 5:

The GPS is quite precise outside Figure 6.

Figure 6:

It also works sometimes inside and is precisely on my room, but sometimes it shows that I'm in Poland the middle of the Atlantic Ocean or Iceland.

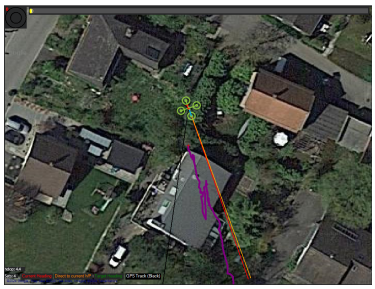

Figure 7: At home

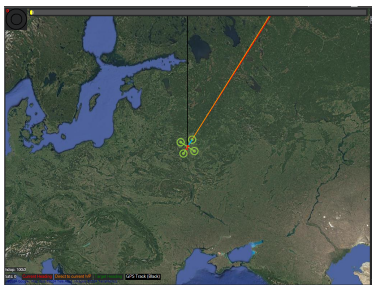

Figure 8:

4.4.4 Receiver/Transmitter

To get the connection between the receiver and radio I first followed the ExpressLRS[26] page. I changed the `Serial6_Protocol` to 23 and the `RSSI_Type` to 3 for the receiver protocol. I also changed the `RC_Options` to the correct bitmask.

The connection between the radio and receiver then worked, because the receiver had a constant blue light and the receiver said telemetry recovered after it was lost for a moment. However as it comes it did not have a connection to the Fc. I thought about changing the receiver from Uart6 to Uart 1, because Uart1 is the usual Uart for a receiver on the Kakute H7, but it would require a JST and

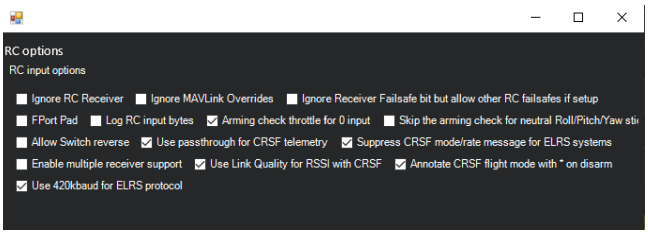

Figure 9:

more soldering. So I searched for a different solution. I looked at the Kakute H7 tab in the documentation and saw that for the Kakute H7 I needed to set `Brd_Alt_Config` to 1 for the CRSF interface of the receiver. `Brd_Alt_Config` is a Fc specific parameter so it makes sense that it did not come up in the search for a solution.

Battery

When I plugged in the battery the smoke stopper in between did not light up. The battery was not immediately recognized in MP and I needed to change the battery settings for the Tekko32 F4 4in1 I only needed to change the `Batt_Monitor` to 4 the other parameters were already correct. I was now able to see the voltage and amperage of the battery.

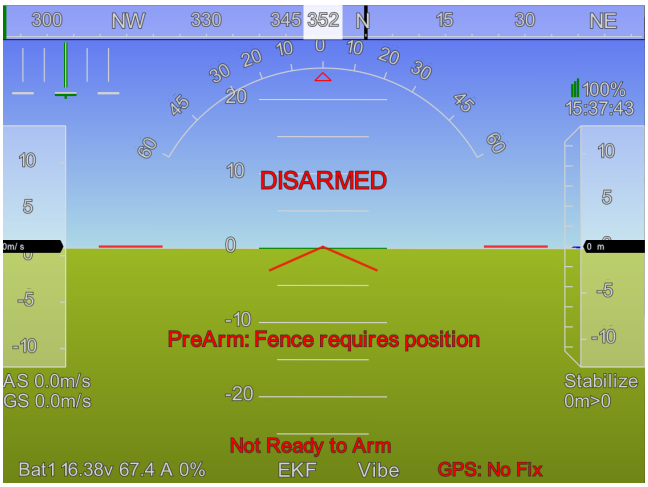

Figure 10:

The only prearm check missing now is the compass calibration.(Figure 9[5])

[5] I'm currently inside and the GPS has no connection, but it works as soon as I'm under free Sky.

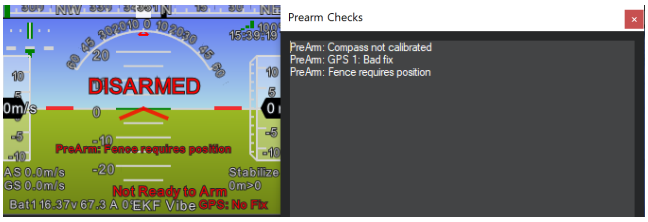

Figure 11:

4.4.5 Motor Test

The motor test works correct after assigning the right position to the motors, because I'm using the M5-M8 ports instead of the M1-M4. I also needed to set `Mot_PWM_Type` to dshot 600 for the
Motor testing works after connecting assigning the right position to the motors and setting the mot_pwm_type to dshot 600. Dshot is just the digital protocol for communication between the Fc and ESC. After some time the motors began to beep, but after testing them again they stopped so it was most likely a inactivity warning. A new error also appeared called battery failsafe. This was caused by a unstable connection between the ESC and Fc and so the Fc took the connection from the computer as power source and had to little power to use the motors.

4.4.6 Compass Calibration

I first tried to callibrate the compass inside, but it turns out that you need a good GPS lock. I went outside and tried calibrating the compass and it failed multiple times. I set the fitness to relaxed and it still failed. The Holybro docs[27] recommended to set the parameter `Compass_Orient` to 6 for proper orientation. It failed again. I was definitely far enough away from metal or my computer which could influence the calibration.
I then fixed it temporarily by doing the large vehical MagCal, however this only took away the prearm message and did not really work. What I did not know at the time is that the ArduPilot documentation warns from using this calibration, because it can look as if it is correctly configured, but in reality the orientation is incorrect. I later on needed to disassamble the copter anyway, because a wire between the motor and ESC came off and needed to be soldered on again, and put the GPS module directly onto the Fc and it worked using the normal compass calibration.

4.4.7 Road to First Flight

All prearm checks are gone except the occasional magfield variance error(it disapeared completely after being able to calibrate the compass normally), which went away after positioning the drone further away from the computer. To arm

Explanation 4.3: to arm

To arm a drone means that the motors begin to spin without producing enough thrust to lift the drone from the ground. It is done when you want to fly before the actual flight.

the drone I still needed to assign a switch on the radio. I did that by setting `RC5_Option` to 153 which means to arm the drone when I flick the switch number 5. Even though there was no prearm check that failed I was not able to arm the drone, also when I disabled the parameter `Arming_Check` it did not work. The Fc still rejected the force arm command from MP. The command however does not disable the geofence which I enabled some when earlier. I disabled it and was able to test the motors spinning on my bench. When I tried it with blades one of them hit one of the antennas from the receiver and tear it off. I was able to just attach it, but I'm not one hundred percent sure that this antenna is working. After this little hiccup a new prearm error appeared, crashdump bin detected. It comes from crashes with your drone to analyze what went wrong. However, because I've only tested it on the bench, I concluded that it is insignificant and that I can delete it. But it is not just easily deleted and so I ignored it and did all the other prearm checks and then disabled `Arming_Check` by setting it to 0. I then armed the quadcopter and throttled up. The only thing it did was spinning on the ground, because some of the motors are turning in the wrong direction. Through a blog post[15] I was also able to delete the crashdump.bin file by flashing new firmware onto the Fc. I tried to reverse the motors by using the reverse button in MP. It worked then by setting `Servo_BLh_Auto` to 1 and open it in BLHeliSuite32[11]. By using the reversed mode in BLHeliSuite32 I was able to get the right motors spinning clockwise. It is also advisable to turn off beacon delay, when working on the bench, because it lets the motors beep after 10 minutes of being unused.

After arming the quad and pushing the throttle I was able to get the drone to fly, but it was shaking violently also known as wobbling and then somewhat crashed it into the ground causing the props to get minimal damage. I tried to tighten everything on the drone, because until that point everything was just somewhat loosely taped onto it, the wobbling still continued. Trying new and also the other props, as there could be a prop imbalance that causes the drone to wobble, had no effect. Through a video[6] about drone wobbling I came to the conclusion that my proportional, integral, derivative (PID) values are for a 9 inch drone not a 5 inch drone.

Explanation 4.4: proportional, integral, derivative (PID)

The PID refers to three variables who control the error adjusting system in a drone. With error is meant the difference between how much the motors rotate and how much they should rotate.

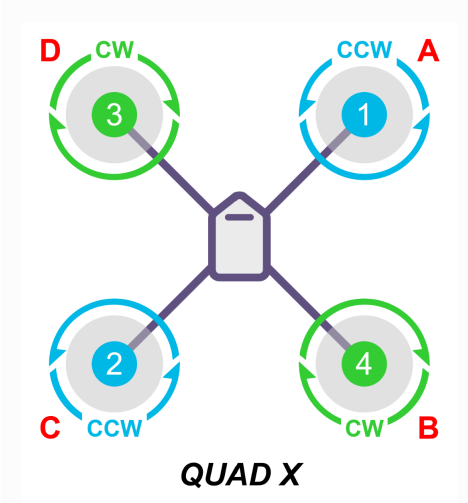

Figure 12: Correct turning directions for my drone

After adjusting them via the intial tuning parameter quick adjustment settings the drone flew. But when I put the stick left the drone flew right and vice versa. I was able to put the `RC2_Reversed` to 1, which reversed the controls.

4.4.8 RaspberryPi Setup

To let the drone fly on its own a companion computer is needed, because the Fc does not have enough power to process more complex operations as autonomous flight.

Choosing the RaspberryPi instead of an other companion computer is advisable, due to it being the most used in drone projects.

The RaspberryPi is not shipped with an microSD card and through that also not with the operating system (OS). To download the OS, the RaspberryPi Imager is used. Through the imager it is possible to create an account for the RaspberryPi, configure the wifi and the possibility for the secure shell (SSH) connection.

Explanation 4.5: operating system(OS)

A OS is a software, which controls all the hardware of a computer.

To see a desktop instead of only a command line based interface with an SSH a virtual network computing (VNC) is used. In this case the recommended option was RealVNC viewer a widely spread VNC. For this RealVNC server needs to

be installed over the terminal via `sudo apt-get realvnc-vnc-server` [8][6].

4.4.9 MAVProxy Installation

The following was based on the MAVProxy Documentation [39] from the ArduPilot website.

First all the needed packages need to be installed through the `sudo apt-get install` command.

```
sudo apt-get install python3-dev python3-opencv python3-wxgtk4
    .0 python3-pip python3-matplotlib python3-lxml python3-
    pygame
pip3 install PyYAML
```

What is not said in the MAVProxy Documentation, is that you need a virtual environment (venv) and can not download it without one or the this error; `error: externally-managed-environment` pops up. To get around this issue you need to create and activate a venv.

```
python3 -m venv mavproxy-env
source mavproxy-env/bin/activate
pip install MAVProxy
```

To connect the drone over MAVProxy it is needed to know how the Fc is connected to the Raspberry Pi. The command `dmesg | tail` will provide the information.

The command `mavproxy.py --master=/dev/ttyACM0 --baudrate 115200 --aircraft MyCopter`, a error will appear if `\ttyUSB0` is used. If the Raspberry Pi is connected over the serial ports, `\serial0` is used instead of `\ttyACM0` and if the baudrate is changed also take the other one, in this case it would be 921600. If the Raspberry Pi will not have another power source except over the Fc, the 5+ V pin on the Raspberry Pi needs to be connected to a 5 volt pin on the Fc and additionally a ground pin. There will be a low voltage warning that pops up, but the RaspberryPi works fine. In addition the `Serial3_Protocol` needs to be changed to 2 for the Mavlink protocol.

When connected to the Fc you can use simple simple commands to change the parameters, as `param set arming_check 0`, or to arm the copter with `arm throttle`.

4.4.10 Dronekit

It needs to be mentioned that the software dronekit is not maintained very well, as it says in the Github repository [31].The following is mostly based on the Dronekit Documentation [40].

[6] realvnc-vnc-viewer only required if you want to see a VNC from the RaspberryPi

First the Dronekit library needs to be installed in the venv with `pip install dronekit`. To create a file in the venv over the terminal `nano dronekittest.py` is used. Noteworthy is that it is not useful to name the file the same as the library itself, because python will confuse it. In the created document you then can connect the Fc to the Raspberry Pi as can be seen in Listing 1.

```
from dronekit import connect

vehicle = connect('/dev/serial0', baud=912600, wait_ready=True)
```

Listing 1: Python DroneKit Example

Afterwards an error occurred.

```
dronekit/__init__.py" , line 2689, in <module>
class Parameters(collections.MutableMapping, HasObservers):
^^^^^^^^^^^^^^^^^^^^^^^^^^
AttributeError: module 'collections' has no attribute '
    MutableMapping'
```

The source of the error is that in python 3.10 the abstract base class, MutableMapping, was moved from collections to collections.abc. This needs to be changed(Listing 4) in the dronekit source code. Which can be accessed with the the command Listing 2

```
nano /home/EduPi/mavproxy-env/lib/python3.11/site-packages/
    dronekit/__init__.py
```

Listing 2: accessing source code

```
class Parameters(collections.MutableMapping, HasObservers):
```

Listing 3: original source code

to:

```
class Parameters(collections.abc.MutableMapping, HasObservers):
```

Listing 4: changed line

After this the program worked flawlessly and it was able to give information from the Fc over the terminal. As shown with the example Listing 5, which

```
print( "Autopilot version: %s" %vehicle.version)
```

Listing 5: information retrieval

```
Autopilot version: APM:Copter-4.5.7
```

Listing 6: output from Listing 5

Uart ports are seen as /dev/ttyS0 or as can be looked at via:

```
ls /dev/ttys0
```

Error:

```
Failed to connect to /dev/ttys0 : [Errno 2] could not open
    port /dev/ttys0: [Errno 2] No such file or directory: '/
    dev/ttys0'
```

needed to add

```
sudo usermod -a -G dialout $USER
```

which adds dialout so the pins to sudo

The main problem of the outdated Dronkitpython library is, that the function `vehicle.channels`, which should read the channel values from the Radiocontroler, is only returning none. It is a long-known issue and to circumvent it a decorator is used [4].

Explanation 4.6: Decorator

```python
@vehicle.on_message("RC_CHANNELS")
def rc_channel_listener(vehicle, name, message):
global latest_rc_channels
latest_rc_channels = message

def get_rc_channel_value(channel_number):
global latest_rc_channels
if latest_rc_channels is None:
return None
channel_value = getattr(latest_rc_channels, f"chan{channel_number
    }_raw", None)
return channel_value
```

Listing 7: decorator for channel values

trying to get a channel value over dronekitpython... does not work

seeing it over mavproxy via the status command, but not the rc status

is probably a bug in dronekit, might have found the issue in question and a "solution" for it

after a lot of trial and error I got this which works to retrieve a value from the radio

```python
from dronekit import connect, VehicleMode
import time
```

```
3
4  vehicle = connect('/dev/serial0', baud=912600, wait_ready=True)
5
6  latest_rc_channels = None
7
8  @vehicle.on_message("RC_CHANNELS")
9  def rc_channel_listener(vehicle, name, message):
10   global latest_rc_channels
11   latest_rc_channels = message
12
13 def get_rc_channel_value(channel_number):
14   global latest_rc_channels
15   if latest_rc_channels is None:
16     return None
17   channel_value = getattr(latest_rc_channels, f"chan{
       channel_number}_raw", None)
18   return channel_value
19
20 while True:
21   time.sleep(1)
22   print(get_rc_channel_value(6))
```

home coordinates are set every time the vehicle is armed

there is no special dronekit command to set a new home location, however it would be possible to access it over mavlink

`vehicle.home_location.lat` gives you the latitude of the home location set by ardupilot while arming the quad.

you can access it in the local frame via `vehicle.location.local_frame`

reading a file in python and exporting coordinates. done by `p = Path(__file__).with_name('coordinates.txt')` to get the correct file, which is in the same folder

opening the file via `p.open('r')` exporting the data... using the exported data to create new list for alt/lon/alt coordinate to have all the alt in one list and so on.

converting the coordinates from the coordinates.txt to

trying to let the drone fly with the `vehicle.simple_takeoff(Zcoordinate)` command does not work

5 Results

6 Discussion and Outlook

7 Conclusion

8 References

Bibliography

[1] Joshua Bardwell. *Why you need a smoke stopper — HOW TO MAKE A SMOKE STOPPER*. May 23, 2019. URL: https://www.youtube.com/watch?v=I5aOTAmEwLE (visited on 10/20/2024).

[2] Tom Begley. *What are brushless RC Cars?* Aug. 20, 2019. URL: https://www.rcgeeks.co.uk/blogs/news/what-are-brushless-rc-cars (visited on 10/19/2024).

[3] Oscar Liang. *Brushed Motors vs Brushless Motors for Quadcopter*. Apr. 2, 2019. URL: https://oscarliang.com/brushed-vs-brushless-motor/ (visited on 10/19/2024).

[4] phil-toppe, Andre van Calster, and mrthomasbarnard. *Getting 'None' for all channel values #969*. Aug. 26, 2019. URL: https://github.com/dronekit/dronekit-python/issues/969 (visited on 12/18/2024).

[5] Paweł Spychalski. *A brief history of a flight controller - From MultiWii to Betaflight and beyond*. Mar. 18, 2020. URL: https://quadmeup.com/a-brief-history-of-a-flight-controller-from-multiwii-to-betaflight-and-beyond/ (visited on 10/14/2024).

[6] Andrew Stapleton. Nov. 25, 2020. URL: https://www.youtube.com/watch?v=5c4BP7BlEAw (visited on 10/20/2024).

[7] Joshua Bardwell. Apr. 7, 2021. URL: https://www.youtube.com/watch?v=a8cy5BK5SbU (visited on 10/20/2024).

[8] IONOS editorial team. *How to set up a VNC on your Raspberry Pi*. Nov. 22, 2022. URL: https://www.ionos.com/digitalguide/server/configuration/setting-up-virtual-network-computing-on-raspberry-pi/ (visited on 10/31/2024).

[9] Oscar Liang. *Flight Controller Firmware for FPV Drone: Choosing Between Betaflight, iNav, Ardupilot*. Feb. 23, 2023. URL: https://oscarliang.com/fc-firmware/ (visited on 10/18/2024).

[10] Lee Schofield. *Choosing between Betaflight. INAV and Ardupilot: A guide for new builders*. Mar. 11, 2023. URL: https://www.youtube.com/watch?v=Y12tAXnGptY&list=PLf77vRmH7VrGk0IP6TiCyleA9ws_ooGxY&index=26 (visited on 10/18/2024).

[11] sskaug. *BLHeliSuite32 Rev32.10.0.0*. Nov. 12, 2023. URL: https://github.com/bitdump/BLHeli/releases (visited on 10/15/2024).

[12] Wezley Varty and Dax Neal. Aug. 29, 2023. URL: https://www.youtube.com/watch?app=desktop&v=CYJ2UOrlXgM (visited on 10/20/2024).

[13] Jack Wu. *The most detailed explanation of P-10 Pro accuracy*. Jan. 29, 2023. URL: https://gpswebshop.com/blogs/tech-support-by-vendors-columbus/what-does-the-p-10-pro-0-5m-cep50-and-1-5m-cep95-horizontal-accuracy-mean (visited on 11/22/2024).

[14] *drone*. Dec. 19, 2024. URL: https://dictionary.cambridge.org/dictionary/english/drone.

[15] Chris Edelen and Andras Schaffer. *Can't delete Crashdump data*. June 8, 2024. URL: https://discuss.ardupilot.org/t/cant-delete-crashdump-data/119045 (visited on 10/20/2024).

[16] Oscar Liang. June 3, 2024. URL: https://oscarliang.com/end-of-blheli_32/ (visited on 12/20/2024).

[17] Paul Mozur and Adam Satariano. July 2, 2024. URL: https://www.nytimes.com/2024/07/02/technology/ukraine-war-ai-weapons.html (visited on 12/19/2024).

[18] Andy Piper. Aug. 20, 2024. URL: https://discuss.ardupilot.org/t/the-great-gps-showdown-speed-precision-and-a-few-surprises/122661 (visited on 10/20/2024).

[19] Marc Santora. *Rise of the Dragons: Fire-Breathing Drones Duel in Ukraine*. Oct. 12, 2024. URL: https://www.nytimes.com/2024/10/12/world/europe/ukraine-russia-dragon-drones.html (visited on 12/19/2024).

[20] Lee Schofield. Oct. 13, 2024. URL: https://www.youtube.com/watch?v=E9JYwwWr0LU (visited on 10/20/2024).

[21] URL: https://www.radiomasterrc.com/products/rp4td-expresslrs-2-4ghz-diversity-receiver?variant=48177685070055 (visited on 10/20/2024).

[22] URL: https://www.radiomasterrc.com/products/boxer-radio-controller-m2?variant=45666886910183 (visited on 10/20/2024).

[23] URL: https://www.foxeer.com/foxeer-donut-5145-props-g-520 (visited on 10/20/2024).

[24] URL: https://www.hqprop.com/hq-durable-prop-5x43x3v1s-2cw2ccw-poly-carbonate-p0048.html (visited on 10/20/2024).

[25] URL: https://isdtshop.com/en-ch/products/isdt-608ac (visited on 10/20/2024).

[26] URL: https://www.expresslrs.org/quick-start/ardupilot-setup/ (visited on 10/20/2024).

[27] URL: https://docs.holybro.com/gps-and-rtk-system/f9p-h-rtk-series/ardupilot-ist8310-compass-orientation (visited on 10/20/2024).

[28] *ArduPilot Copter*. URL: https://ardupilot.org/copter/ (visited on 10/19/2024).

[29] *ArduPilot Firmware builds*. URL: https://www.st.com/en/development-tools/stm32cubeprog.html (visited on 09/29/2024).

[30] *Choosing an Autopilot*. URL: https://ardupilot.org/copter/docs/common-autopilots.html (visited on 10/18/2024).

[31] *DroneKit Python*. URL: https://github.com/dronekit/dronekit-python (visited on 11/03/2024).

[32] *Kakute H7 v1.3 (MPU6000)*. URL: https://holybro.com/products/kakute-h7 (visited on 10/18/2024).

[33] *Micro M10 GPS*. URL: https://holybro.com/products/micro-m10-m9n-gps?variant=42981482954941 (visited on 10/20/2024).

[34] *Mission Planner Home*. URL: https://ardupilot.org/planner/index.html#home (visited on 09/29/2024).

[35] Nathan Seidle. *What is GPS RTK?* URL: https://learn.sparkfun.com/tutorials/what-is-gps-rtk (visited on 10/20/2024).

[36] *STM32CubeProgrammer software for all STM32*. URL: https://www.st.com/en/development-tools/stm32cubeprog.html (visited on 09/29/2024).

[37] *Tattu 2000mAh 4S 120C 14.8V R-Line Version 3.0 Lipo Battery Pack with XT60 Plug*. URL: https://genstattu.com/ta-rl3-120c-2000-4s1p.html (visited on 10/19/2024).

[38] *Tekko32 F4 4in1 50A ESC(AM32)*. URL: https://holybro.com/products/tekko32-f4-4in1-50a-esc (visited on 10/18/2024).

[39] Andrew Tridgell, Peter Barker, and Stephen Dade. *MAVProxy. A UAV ground station software package for MAVLink based systems*. URL: https://ardupilot.org/mavproxy/ (visited on 10/31/2024).

[40] *Welcome to DroneKit-Python's documentation!* URL: https://dronekit.netlify.app/ (visited on 11/03/2024).

[41] *XING-E Pro 2207 2-6S FPV Motor*. URL: https://shop.iflight.com/xing-e-pro-2207-2-6s-fpv-nextgen-motor-pro874 (visited on 10/19/2024).

List of Tables

List of Figures

1 [2] 6
2 10
3 11
4 11
5 12
6 12
7 At home 12
8 12
9 13
10 13
11 14
12 Correct turning directions for my drone 16

9 Table of Figures

Acronyms

AIO All-in-One. 4

CEP circular error probable. 6

DFU device firmware updgrad. 9

ESC electronic speed controller. 4, 5

Fc flight controller. 4, 9, 17, 18

GPS global positioning system. 4, 6

Li-ion lithium-ion. 5, 6

LiPo lithium polymere. 5, 6

OS operating system. 16

PID proportional, integral, derivative. 15, 16

SSH secure shell. 16

venv virtual environment. 17

VNC virtual network computing. 16