Autonomous Drone

Engineering, Construction, and Programming of a Fully Autonomous Drone

Matteo Giger 6bG

Kantonsschule Rychenberg

Supervisor: Michael Oettli

2nd Supervisor: Dr. Cyril Caspar

Submitted: 6.12.2022

1. Introduction

1.1 Preface

Drones have been envisioned as the future of commercial delivery for years, but it has yet to become reality. Amazon has shown this to be more than mere science fiction in 2013[1]. Nine years, two billion dollars, and several unfulfilled promises later, Amazon's drone delivery is still undergoing a testing phase[2].

A drone capable of fully autonomous commercial delivery must be engineered with numerous concerns in mind, such as safety, privacy, and profitability. This has led to a consolidation of the usage of cameras as primary systems of detection, as they provide a cheap and safe way to detect obstacles. The privacy concern we encounter relates to flying drones equipped with cameras over residential areas, which is often neglected. Although other sensors are used in a complementary manner, the omission of cameras is never considered.

1.2 Regulations

Confusion and misinformation about the laws regarding drones are widespread, as they differ vastly internationally. Switzerland is considered to have very lenient regulations, which is expected to change once Switzerland adopts the EU regulations for the operation of drones[3]. As for now the operation of unmanned aerial vehicles in the range of 0.5 kilograms to 30 kilograms requires liability insurance that pays for damages up to at least one million Swiss francs[4]. Furthermore, the operator of the vehicle must have direct eye contact with the drone and the ability to ensure control of the drone at all times. Flying drones that weigh over 0.5 kilograms in proximity to airports is prohibited. In the larger radii of airports, they

[1] David Pierce, "Delivery drones are coming: Jeff Bezos promises half-hour shipping with Amazon Prime Air". https://www.theverge.com/2013/12/1/5164340/delivery-drones-are-coming-jeff-bezos-previews-half-hour-shipping (29.10.2022)
[2] Spencer Soper and Matt Day, "Amazon drone crashes hit Jeff Bezos' delivery dreams". https://www.bloomberg.com/news/features/2022-04-10/amazon-drone-crashes-delays-put-bezos-s-delivery-dream-at-risk?sref=VV6WcQFL (29.10.2022)
[3] Bundesrecht, "Update: EU-Drohnenregulierung in der Schweiz". https://www.bazl.admin.ch/bazl/de/home/drohnen/drohnen-und-flugmodelle/Europaeische_Drohnenregulierung_uebernommen.html (29.10.2022)
[4] Bundesrecht, "Verordnung des UVEK über Luftfahrzeuge besonderer Kategorien". https://www.fedlex.admin.ch/eli/cc/1994/3076_3076_3076/de - fn-d39694e131 (29.10.2022)

are allowed to fly up to an altitude of 150 metres[5]. Prohibitive zones also include spaces closer than 100 metres to groups of more than twelve people[6].

1.3 Personal Motivation

I started taking an interest in the magical inner workings of computers early on in my life. After building my first computer, my fascination only grew and I decided to build my own keyboard, which among other things made me learn soldering. Meanwhile, computer programming caught my interest, which I taught myself in C++. After learning about electrodynamics in our physics class and encountering *Arduino*[7] in the physics practical, I was determined to merge my interests into a single project.

Therefore, drones were an optimal topic; they not only suit my aforementioned interests, but also deal with a concern I share in regard to digitalisation, the loss of privacy. A network of delivery drones using cameras implies recording dissenting third parties on their property, which is morally questionable.

As a technology enthusiast, I have high hopes for autonomous drones to take over tasks that previously required human labour. This should however not come at the cost of privacy. Given this, the project aims to be a proof of concept for an autonomous drone capable of conserving the core principle of privacy. I hope to inspire other projects to take similar approaches.

2. Concept

2.1 Objective

The objective of my research project is to engineer, construct, and program a fully autonomous drone capable of avoiding obstacles in an unknown environment without the aid of cameras. The success criteria are the following:

(i) The drone is capable of flying reliably in any direction.

---

[5] BAZL, “Flugeinschränkungen in der Schweiz”. https://www.bazl.admin.ch/bazl/de/home/drohnen/drohnen-und-flugmodelle/drohnenkarte.html (29.10.2022)
[6] Bundesrecht, “Verordnung des UVEK über Luftfahrzeuge besonderer Kategorien”. https://www.fedlex.admin.ch/eli/cc/1994/3076_3076_3076/de - fn-d39694e131 (29.10.2022)
[7] Arduino: https://www.arduino.cc/ (30.11.2022)

(ii) The drone can compete in terms of flight time with commercially available drones.

(iii) The drone flies from A to B without manual input, including take-off and landing.

(iv) The drone avoids obstacles along the way by:

(a) choosing an avoidance path at its altitude.

(b) attempting to fly over the obstacle if there is no way to avoid it at the current altitude or if the avoidance path leads too far away from the original destination path.

(v) The drone is able to fly an avoidance path traced during a previous flight to the same destination and adjusts the path on the flight if new obstacles have appeared since.

2.2 Limitations

The following assumptions are made about the obstacles:

(i) Obstacles do not have any parts that are overhanging by more than 1 metre.

(ii) Obstacles are stationary during the drone's flight.

(iii) Obstacles are not levitating.

2.3 Approach

A 360-degree **li**ght **d**etection **a**nd **r**anging (abbr: lidar) sensor will allow the drone to detect obstacles. Unlike cameras, the lidar scans a two-dimensional plane, therefore the drone must fly in a way such that all obstacles that lead to a collision intersect the scanned plane. The lidar is placed at the top of the drone but as it must fly at an angle to move, the scanned plane is not parallel to the ground, which allows the sensor to detect obstacles for the entire height of the drone.

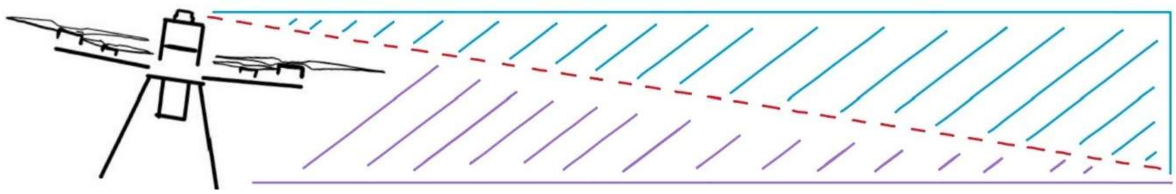


Figure 1: Scanned plane (red)

As visible in Picture 1 the scanned plane covers the upper part of the drone close to it and the lower part further away. If obstacles are only detected close to it, evasion might not be possible anymore. Therefore, the aforementioned limitations, are necessary, as the cross sections of the remaining obstacles are fully connected to the ground and thus the lower part of the obstacles is detected further away from the drone.

There will be two modes of autonomous behaviour:

EM - Exploration Mode:

The drone attempts to fly to the target coordinates in a straight line. If obstacles block its path, the drone will attempt to avoid them while maintaining altitude. Should this not be possible, the drone will attempt to fly over the obstacle. Waypoints are saved such that when connected by straight lines they are congruent with the flight trajectory.

WM - Waypoint Mode:

The drone flies to target coordinates via the waypoints saved in the exploration phase. Obstacles that were already present in exploration mode (EM) will now be avoided according to these waypoints. The lidar is still active in case some obstacles have moved or new obstacles appear. If these obstruct the flight path, the drone will follow the exploration mode procedure until the drone has reached the next waypoint.

3. Physical Components

3.1 Selection

3.1.1 Introduction

The market for individual drone parts is largely upheld by the sport of drone racing[8]. Racing drones are pushed to peak performance and crashes occur regularly, which is why their parts need to offer high performance at a cheap price.

However, my drone had different requirements. As many parts come from Asia, delivery takes months. Due to the limited time frame of this project, no parts could be replaced in case of a crash. Additionally, there was no need for racing-level performance. The aim was to build a reliable drone, which can compete in terms of flight time with commercially

[8] Red Bull UAE, “Drone Racing: The Sport of the Future”. https://www.redbull.com/mea-en/drone-racing-the-sport-of-the-future (20.8.2022).

available drones, while having extra load capacity. Furthermore, it had to provide an interface that a traditional computer would be able to utilise. To achieve said goal, I had to carefully study numerous articles on drone parts and determine the most suitable arrangement.

3.1.2 Individual Drone Parts

There are two computers onboard: the flight controller, which focuses on the low-level task of keeping the drone level, and the companion computer, which manages the high-level tasks such as recognizing obstacles and deciding how the drone should move. For my purposes, I chose the *Pixhawk 4*[9] and the *Raspberry Pi 3B+*[10].

The *Pixhawk 4* is a highly sophisticated flight controller, which provides various advanced features, such as maintaining the position of the drone against wind and safe landing in case of communication loss with the companion computer. This provides the reliability, which the chosen companion computer – the *Raspberry Pi 3B+* – lacks. As the *Raspberry Pi* has no integrated sensors, it relies on the sensors of the flight controller and a single lidar sensor to recognize obstacles.

This is rather uncommon for drones since most drones use a set of cameras along with computer vision to avoid obstacles. The lidar works similarly to the well-known radar, but instead of radio waves, it uses laser beams, which allows lidars to create higher-resolution images. The lidar sends out a laser beam, which is then reflected by the object it hits, back to the lidar. Multiplying the speed of light with half of the time it took for the laser beam to return results in the distance between the lidar and the object. Using a lidar sensor instead of a camera means less computation is required to process the recorded data and the drone has no issue with low-light environments. Additionally, it does not infringe on privacy (see sections 1.3 and 2.1).

The *LD06 lidar*[11] scans a two-dimensional plane and does not detect anything above or below the scanned plane. To scan a different plane, the drone can either change its altitude or tilt by accelerating, which is a task of the flight controller, as the flight controller tells the electronic speed controllers the speed at which each motor must rotate. To reach the motors, the current runs through the next important component needed to build a drone, the

[9] Pixhawk 4: https://docs.px4.io/main/en/flight_controller/pixhawk4.html (2.8.2022).
[10] Raspberry Pi 3B+: https://www.raspberrypi.com/products/raspberry-pi-3-model-b-plus/ (2.8.2022).
[11] LDRobot LD06: https://www.ldrobot.com/product/en/98 (5.8.2022).

electronic speed controllers, so they must be able to handle a high enough current. The *Hobbywing brushless electronic speed controllers* are rated at 40 ampères, which is more than enough for the *SunnySky v3508 580kv*[12] motor paired with their *EOLO 11-inch*[13] *carbon fiber propeller*. But due to their efficiency, they still provide up to 11.28 newtons of thrust, which is the equivalent to the force required to lift 1150 grams. Using a bigger propeller would be even more efficient, but if the propellers are too close together, their airflow could interfere with each other. For this reason, the frame also plays a key role as it determines the spacing of the propellers. The frame should additionally be lightweight, durable, and provide enough space for all the electronics.

The *ZD550* is a building kit for a frame with a 550-millimetre diameter. It is made entirely of carbon fibre, allowing it to be both lightweight and durable.

Picture 1: ZD550 Frame building kit

The transmitter and the receiver are used to control the drone manually. The transmitter sends a signal to the receiver and the receiver then passes this signal onto the previously mentioned flight controller. Although the aim of this project is an autonomous drone it is necessary to be able to intervene manually at any moment as programming errors can always occur. The *Taranis X9D Plus SE 2019 transmitter*[14] is compatible with the *X8R receiver*[15] and together they provide all the basic functions. Note that the electronics need to be powered at different voltages, which means a voltage converter is required. The *PM07*

[12] SunnySky v3508 580kv: https://sunnyskyusa.com/collections/v-motors/products/sunnysky-v3508-motor?variant=45680383759 (5.8.2022).
[13] SunnySky EOLO 11-inch propeller: https://sunnyskyusa.com/collections/polymer-propellers/products/eolo-cn11-4-5-prop (5.8.2022)
[14] FrSky Taranis X9D+ SE 2019: https://www.frsky-rc.com/product/taranis-x9d-plus-se-2019/ (8.8.2022).
[15] FrSky X8R: https://www.frsky-rc.com/product/x8r/ (8.8.2022).

is a power management board with built-in voltage converters to supply all the electronics with the correct voltage.

Lastly, a battery to power everything is necessary. Lithium-Polymer batteries[16] are the gold standard for drones, because of their high energy-to-weight ratio and the ability to discharge at a high rate. The voltage of the batteries depends on their cell configuration. Because some parts are only compatible with the voltage of a LiPo battery with four cells connected in series, they are the only batteries being considered. The most important criterion is how much energy the battery stores. However, as the capacity increases so does the weight, which in turn means the motors require more power to keep the drone airborne. Therefore, the calculations in the following subchapter are required to find the optimal battery.

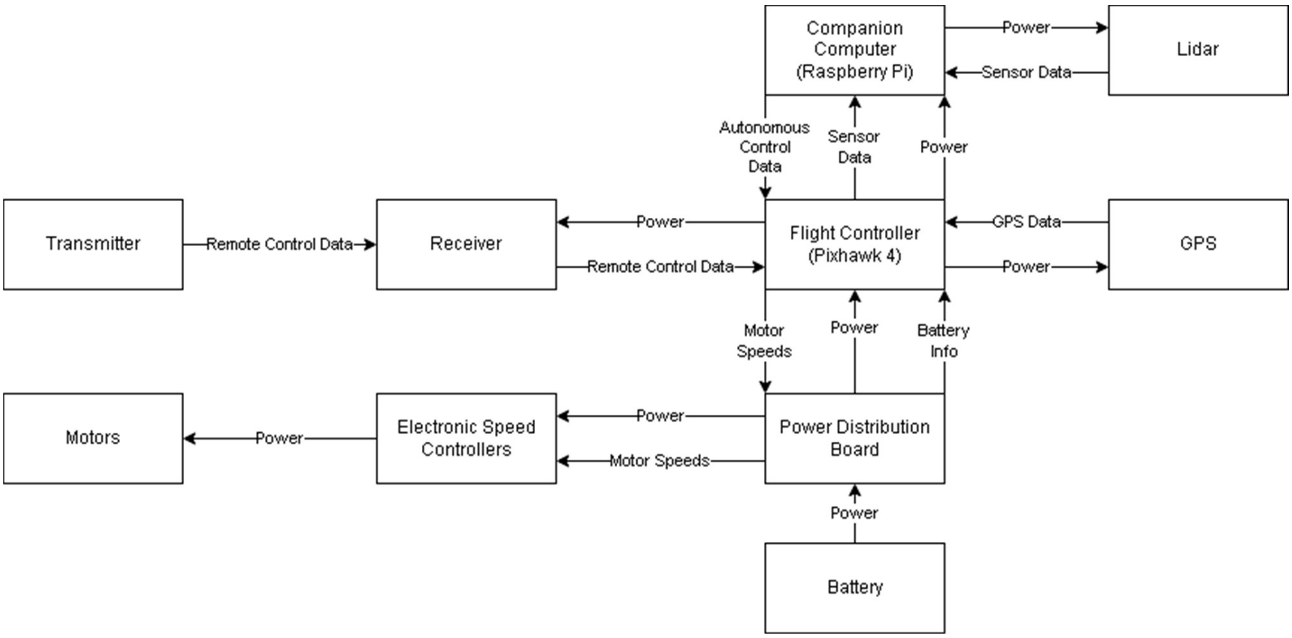


Figure 2: Relations between individual parts

3.1.3 Calculations

As previously stated, flight time was a major selection criterion and the most important variable to consider. The second important variable was the load capacity, which specifies how much extra weight the drone can carry. It is integral to all use cases of drones. Therefore, calculations with different configurations were necessary.

The weight of the drone is important for all other calculations and should be kept as low as possible. Adding up the specified weight of all the parts selected in 3.1.2 results in

[16] Eric, "RC LiPo Battery Guide: Explanation, Safety and Care". http://learningrc.com/LiPo-battery/ (17.08.2022).

2.401 kilograms when weighing my drone, the result was found to be 2.435 kilograms. The discrepancy is due to the additional hardware needed: the measured value includes the modifications made to the frame, cables, and the mounting of certain electronics, while the calculated weight does not.

Load capacity refers to how much additional weight a vehicle can carry. The load capacity is determined by the maximal thrust and the weight of the drone. For the drone to fly, the maximum thrust must always be greater than the weight of the drone. To determine the load capacity, we also have to define how much greater the maximum thrust should be. As per the widespread rule of thumb[17], **the drone has to be able to produce twice as much thrust as the total weight of the drone including the payload**. This leads to the following formula to calculate the load capacity:

$$\text{Load Capacity(kg)} = 0.5 \cdot \text{Maximal Thrust(kg)} - \text{Weight of drone(kg)}$$

$$\mathbf{-0.135\ kg} = 0.5 \cdot 4.6\ \text{kg} - 2.435\ \text{kg}$$

According to this rule the drone is too heavy, however, given it is a rule of thumb it is not precise and will not be true for every given scenario. It is important to understand why the maximum thrust should nonetheless be significantly greater than the total weight of the drone: The thrust-to-weight ratio has a substantial effect on the agility of the drone. Not only does it determine the acceleration the drone can achieve, but also heavily influences at what angle the drone can fly without losing altitude as the vertical component of the thrust vector must be equal to the drone's entire weight. Flying at an angle is the only way the drone can change direction horizontally. If we want our drone to be able to fly at an angle of 45 degrees while maintaining altitude our formula changes to:

$$\text{Load Capacity(kg)} = \cos 45° \cdot \text{Maximum Thrust(kg)} - \text{Weight of drone (kg)}$$

Another thing to consider is motor overheating. Drone motors typically cannot provide full thrust for longer periods. Combining these new findings leads to the following conclusion: **The thrust produced by the drone must be greater than the total weight of the drone including the payload while the motors operate at a safe temperature by a factor of $(\cos 45°)^{-1}$. A temperature is considered safe if the motor can operate at said temperature for a full battery charge.** The manufacturer of the motor will usually specify how long the motor can operate at a specific temperature without being damaged.

[17] Droneomega.com, "The Beginner's Guide to Drone Motor Essentials". https://droneomega.com/DRONE-MOTOR-ESSENTIALS/ (14.8.2022).

$$\text{Load Capacity(kg)} = \cos 45° \cdot \frac{\text{Safe thrust \%}}{100} \cdot \text{Thrust(kg)} - \text{Weight of drone(kg)}$$

$$\mathbf{0.8177\ kg} = 0.707 * \frac{100}{100} * 4.6\ \text{kg} - 2.435\ \text{kg}$$

Using my chosen configuration, the motor will not overheat even at 100% thrust as the 11 inch-propellers create too little resistance[18], and thus the drone should be able to carry **0.818 kilograms** of additional weight. A third value can be obtained from the manufacturer, which specifies the recommended take-off weight for the four motors to be 2.8 kilograms[19]. Subtracting the weight of my drone results in **0.365 kilograms** of load capacity. It is noteworthy that the motor manufacturer provides this value independent of the propeller choice.

To determine which value is closer to the true load capacity, experiments would need to be performed. But these values allow one to conclude that despite the rule of thumb the drone is not too heavy and possesses the ability to carry additional weight.

Flight time refers to how long a drone can fly. This however is hard to calculate as there are too many variables, such as wind, temperature, and acceleration. To get an approximate idea of the drone's performance, it is reasonable to calculate the hover time without wind instead. To find out how long the drone can hover, simply divide the battery capacity by the power consumption, which is present when the drone produces the same amount of thrust as it weighs. As I am using a lithium-polymer battery it is worth noting that over-discharging will permanently damage the battery, so it is advisable to start your landing manoeuvre well in advance.

$$\text{Hover time(h)} = \frac{\text{Battery capacity(Wh)}}{\text{Power consumption (W)}}$$

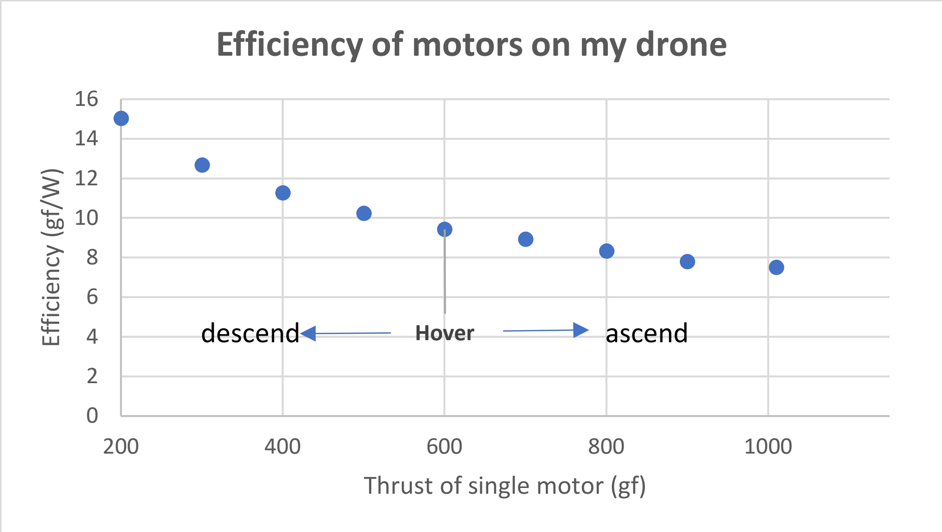
The most difficult part of this passage is to determine the power consumption. On my drone there are seven components that draw power. The flight controller, the electronic speed controllers, the motors, the companion computer, the receiver, the power management board, and the lidar sensor.

[18] SunnySky v3508 580kv performance table: https://sunnyskyusa.com/collections/v-motors/products/sunnysky-v3508-motor?variant=45680383759 (5.8.2022).
[19] SunnySky v3508 580kv Specifications: https://sunnyskyusa.com/collections/v-motors/products/sunnysky-v3508-motor?variant=45680383759 (5.8.2022).

Flight controller	2.5 W[20]
Electronic speed controllers	Insignificant
4 Drone motors	256 W (hovering) [21]
Companion computer	2.5 W[22]
Receiver	0.5 W[23]
Power management board	Insignificant
Lidar	0.3 W[24]
Total	**261.8 W**

Table 1: Power consumption of onboard parts

To optimise hover time, we must look at how the motor efficiency changes based on how much thrust it has to produce.

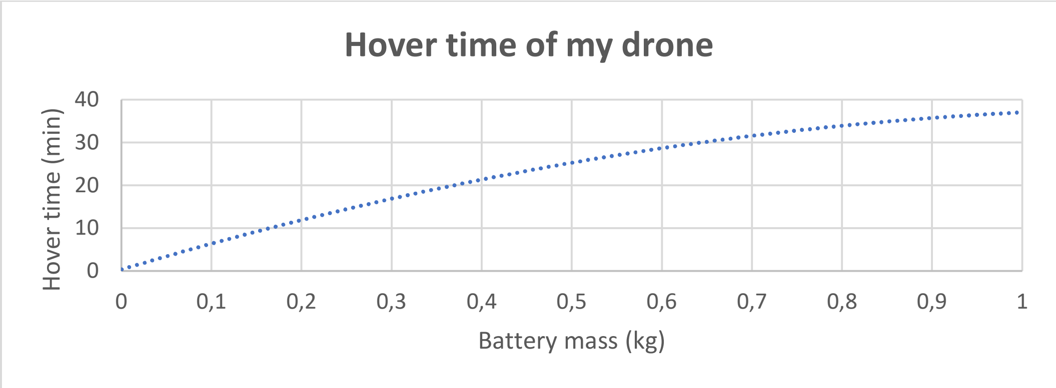


Graph 1: Relation of motor efficiency and produced thrust of my chosen propeller/motor combination 25

Motor efficiency decreases with increasing thrust produced. Therefore, if we take a smaller battery the motors are more efficient when hovering, but a bigger battery has a better energy-to-weight ratio. Using curve fitting we can derive an equation to express battery capacity[26] and power consumption[25] in terms of battery mass and create an optimization problem (see Graph 2) to reach the conclusion that **the increase in capacity matters more**

[20] Stephen, Dade, "Pixhawk (and APM) Power Consumption". https://diydrones.com/profiles/blogs/pixhawk-and-apm-power-consumption (20.8.2022).
[21] SunnySky v3508 580kv Performance table: https://sunnyskyusa.com/collections/v-motors/products/sunnysky-v3508-motor?variant=45680383759 (5.8.2022).
[22] Matt, "Raspberry Pi Power Consumption Data". https://www.raspberrypi-spy.co.uk/2018/11/raspberry-pi-power-consumption-data/ (20.8.2022).
[23] FrSky X8R Specifications: https://www.frsky-rc.com/product/x8r/ (13.8.2022).
[24] LDRobot LD06 Specifications: https://www.ldrobot.com/product/en/98 (5.8.2022).
[25] SunnySky v3508 580kv Performance table: https://sunnyskyusa.com/collections/v-motors/products/sunnysky-v3508-motor?variant=45680383759 (5.8.2022).
[26] Swaytronic Data source to derive battery mass to battery capacity function: https://www.swaytronic.ch/RC-Akkutechnik/Akkus-nach-Spannung/LiPo-Akku-4S-14.8V/?order=price-asc&p=1&properties=f257c03cbb114d90b253f34c34c17863 (20.8.2022).

than the decrease in efficiency of the motors that comes with a bigger battery. Therefore, the biggest battery that the drone can carry will equal the longest hover time. *The Swaytronic 10'000mah 4S battery*[27] is the biggest 4S battery that *Swaytronic* offers and lasts 34 minutes per charge when neglecting take-off and landing.



Graph 2: Relation of battery mass and hover time on my drone

In an attempt to measure the hover time two tests were performed under extreme and mild conditions respectively. For the first test, the drone held its position against wind in 4-degree weather while the second test was performed indoors at room temperature and the position of the drone was controlled manually. The battery's voltage was measured before and after the flight using an external device and a projection was made to get an approximation of how long 100% battery capacity would last. This resulted in 30 minutes under mild conditions and 19 minutes under extreme conditions. Even under mild conditions the drone performed significantly worse than the calculations, this can however be partly attributed to the manual controlling as unsteady flight consumes more power. As for the test under extreme conditions, the low temperature is presumably the main cause for the underwhelming result, since LiPo batteries are notoriously sensitive to low temperatures.

[27] Swaytronic LiPo 4S 10000mah: https://www.swaytronic.ch/Swaytronic-LiPo-Akku-4S1P-14.8V-10-000mAh-35C-70C-XT90 (17.8.2022).

3.2 Assembly

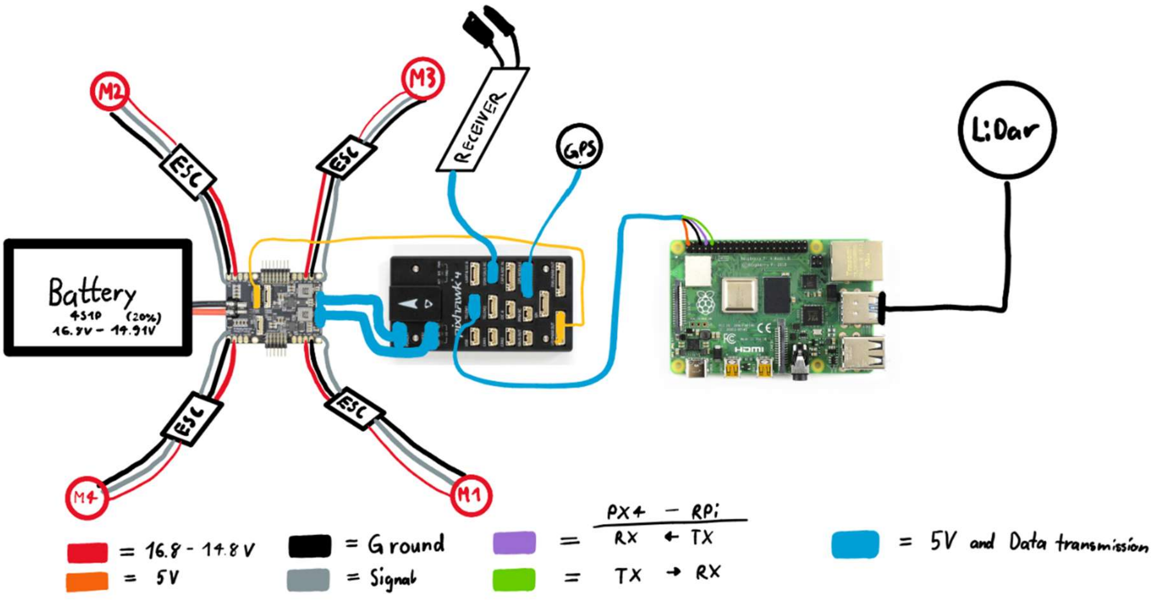
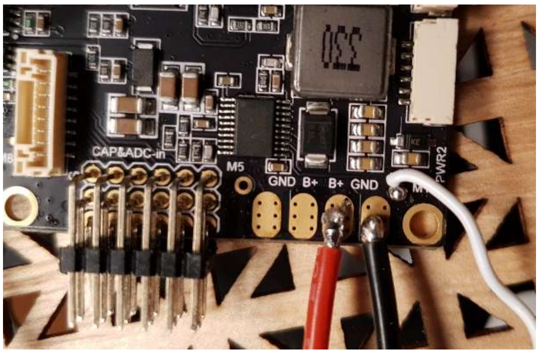


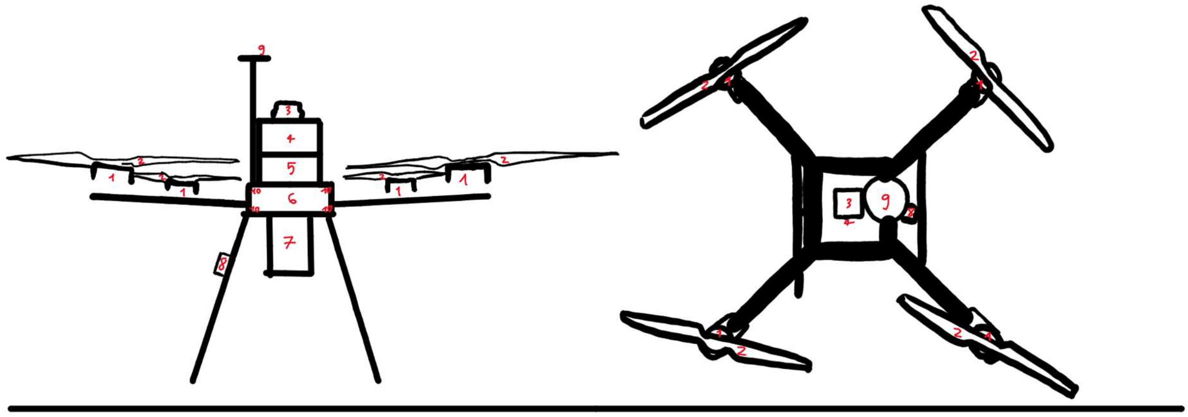
Figure 3: Overview of onboard electronics and wiring

Note that the blue cables can simply be plugged in, whereas the others must be soldered. As for the connection between the *Pixhawk* and the *Raspberry Pi* the single wires in the cable must be separated and then soldered.



Picture 2: Signal wire, Ground wire, and live wire soldered onto the power management board

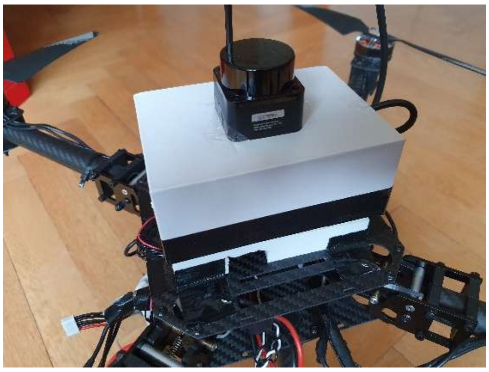
As the *ZD550* frame is not a prebuilt frame but a building kit, it allows for easy modifications, such as an enlargement of the battery compartment – which I had to perform as it was too small for my battery. The battery is placed at the lowest part of the vehicle to keep the centre of mass as low as possible. This reduces the risk of the drone tipping over on take-off or landing. The power management board is in the centre of the drone to allow for easy wiring.



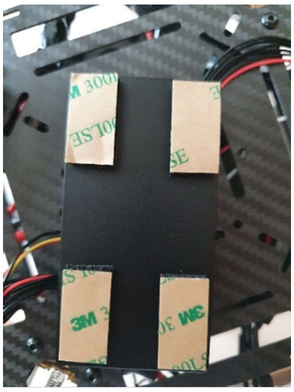
1: Motor, 2: Propeller, 3: Lidar, 4: Raspberry Pi, 5: Pixhawk 4, 6: PMB, 7: Battery, 8: Receiver, 9: GPS, 10: ESC

Figure 4: Overview of parts onboard

The battery compartment had to be enlarged using longer screws. The lidar and the motors needed to be fixed using a spirit level. The lidar stands on a tower made of polystyrene. The flight controller is glued directly onto the frame using vibration isolation tape. This is necessary to reduce noise for the integrated sensors and obtain more accurate readings.



Picture 3: Polystyrene tower housing Raspberry Pi and Pixhawk



Picture 4: Vibration isolation tape on the back of the Pixhawk

4. Programming

4.1 Introduction

The extent of my programming encompasses the code executed on the *Raspberry Pi*. The *Pixhawk* on the other hand runs an open-source software called *PX4*[28]. This software allows the *Pixhawk* to react to both the remote control and the *Raspberry Pi*. The program that runs on the *Raspberry Pi* is written in C++ and communicates with the *Pixhawk* and the lidar. The following pages will discuss the implementation of the exploration mode and the waypoint mode mentioned in section 2.3.

```cpp
void waypoint_mode(Autopilot_Interface &api, ldlidar::LiPkg* lidar, std::vector<mavlink_local_position_ned_t>* waypoints)
{
    std::cout << "[Waypoint Mode] start" << std::endl;

    for (int i = 0; i < waypoints->size(); i++) {

        //initialize mavlink_set_position_target_local_ned_t struct
        mavlink_set_position_target_local_ned_t sp;

        //altitude correction
        if (abs(api.current_messages.local_position_ned.z - waypoints->at(i).z) > 0.3) {…

        //fly to next waypoint
        std::vector<mavlink_local_position_ned_t> intermediate_waypoints = exploration_mode(api, lidar, waypoints->at(i).x, waypoints->at(i).y);

        //save path adjustment
        if (intermediate_waypoints.size() > 2)…

        std::cout << "[Waypoint Mode] waypoint reached" << std::endl;
    }

    std::cout << "[Waypoint Mode] end" << std::endl;
}
```

Picture 5: Code snippet of waypoint mode

4.2 Sensors

4.2.1 Lidar

The avoidance algorithm has access to a list of all the measuring points that the lidar collected in one full rotation. This list is not sent by the lidar instead it must be constructed out of the data packets, which the lidar continuously sends via a port to the *Raspberry Pi*. A single data packet has the following structure[29]:

Header	VerLen	Speed	Start angle	Data	End angle	Timestamp	CRC check

Table 2: Lidar packet fields

[28] PX4, "Open-Source Autopilot For Drone Developers". https://px4.io/ (4.11.2022)
[29] LDRobot, "Development Manual". https://www.ldrobot.com/download/en/98 (4.11.2022)

The header marks the beginning of a packet and always has the same value. When the *Raspberry Pi* receives a byte with this specific value, it can assume that the next byte corresponds to VerLen, the following byte to speed, and so forth.

The VerLen field holds information about how many measuring points are contained within the data field but as the data field always contains twelve measuring points there is no use in reading the value of this field.

The speed field, as the name suggests, indicates the speed at which the lidar turns. By default, the lidar has a rotation frequency of 12Hz. As the speed should only change on request, this field is also not used.

The data field holds the distance and the intensity of twelve measuring points.

The start angle corresponds to the angle of the first measuring point, while the end angle corresponds to the angle of the last measuring point. The angles of the ten remaining points are calculated with the assumption that the speed of the lidar was constant.

The cyclic redundancy check field contains a value that was calculated based on all the other values in the packet. The calculation is repeated by the *Raspberry Pi*, and the result is used to check this field's value. If the values do not match up, the whole packet is discarded. This is done to prevent corrupted packets from providing wrong data to the avoidance algorithm.

If the *Raspberry* has collected enough packets to create a list that represents a full revolution, unreasonable[30] measurements are removed, and the list is created. As radians are used throughout the program, the angles are converted to radians in this list as well.

4.2.2 Inertial Measuring Unit

The *Pixhawk* has an integrated inertial measuring unit, a GPS, and a compass to stabilise the flight on its own. But the data, which is collected by said sensors, is also crucial to the *Raspberry Pi* to avoid obstacles. The drone's attitude angles are an example of this. Therefore, the *Pixhawk* needs to send the data to the *Raspberry Pi*. Once again, this data

[30] The reasonability of a measuring point is determined by the intensity of the reflection and other measuring points in close proximity

is continuously sent in data packets via a port. The packets follow the *MAVLink 2* packet format[31]:

STX	LEN	INC Flags	CMP Flags	SEQ	SYS ID	COMP ID	MSG ID	payload	check sum	Signature

Table 3: MAVLink 2 packet fields

Although the fields at the beginning and the end of the packet are called differently, they function very similarly to the fields of the lidar packets. The STX or start-of-text field corresponds to the header and the checksum, and the signature ensure the integrity of the packet. The biggest difference is the payload field compared to the data field. While the data field always contains twelve measurement points with two values per point, the payload contains different information based on the value of the MSG ID field. Therefore, the MSG ID field is particularly important to correctly interpret the payload data. Upon receival of a packet, the *Raspberry Pi* checks the MSG ID against a few of the IDs that the *MAVLink* website provides[32]. If the IDs match the *Raspberry Pi* can decode the payload accordingly and use the sensor data within the program.

4.3 Movement Commands

The *MAVLink 2* packet format is used for all communication between the *Pixhawk* and the *Raspberry Pi*. This includes the commands that are sent from the *Raspberry Pi* to the *Pixhawk*. There are several options for the *Raspberry Pi* to control the drone, such as sending a target position, velocity, acceleration, or speed for each motor. While the high-level commands are attractive for their ease of use, they leave the low-level decisions up to interpretation, which disrupts the flow of the program. Using the target position command will result in the drone flying to the specified position at an unreasonably high speed. The target velocity command makes the drone accelerate aggressively, which tilts the drone enough to render the lidar useless in crucial moments. For those reasons, it would be best to send acceleration commands but unfortunately, that did not work. To achieve the same effect, the desired velocity is not sent directly but instead, an intermediate velocity is

[31] MAVLink, “Packet Serialization”. https://mavlink.io/en/guide/serialization.html (4.11.2022)
[32] MAVLink, “Messages”. https://mavlink.io/en/messages/common.html (4.11.2022)

determined. The *Raspberry Pi* calculates the x and y components separately using the following formula:

$$\text{intermediate velocity} = \text{current velocity} + (\text{target velocity} - \text{current velocity}) \cdot 0.5$$

The *Pixhawk* then enforces the intermediate velocity in a **n**orth **e**ast **d**own (NED) coordinate frame and adjusts the speed of each motor accordingly.

4.4 Exploration Mode (EM)

Throughout the development of the exploration mode, the ideas, and the implementations of said ideas changed numerous times ranging from solutions that require low processing power to solutions requiring high processing power. At this point, **s**imultaneous **l**ocalization, **a**nd **m**apping (abbr: SLAM) must be mentioned. SLAM is a method that would allow the drone to create a map of the environment and simultaneously localise itself on said map. It does so by recognizing multiple measuring points of the lidar which were scanned from various positions as the same points in space thus allowing the drone to deduct its change of location. Lidars that scan 360 degrees are suitable for SLAM as they collect measuring points in all directions. However, using SLAM on outdoor drones is challenging as it cannot localise the drone when no obstacles are in range of the sensor, hence other sensor data must assist in localising the drone. This, combined with the limited time frame of the project had given me reasons to disregard SLAM. Instead, a position estimate using GPS and accelerometers helps the drone navigate. To avoid obstacles the drone continuously calculates the distance to the next collision. It does so by checking how far away the first obstacle within a 3 metres wide path in a given direction is. Although the drone is not three metres wide the extra space provides safety and time for the drone to react to gusts of wind that can push the drone closer to the obstacle. A higher path width causes the drone to be more cautious, however, if the path width is too high the drone will always choose to fly over the obstacle, as it will see no path to avoid the obstacle. The path width of three metres was found experimentally. If a measuring point lies within a path the following statement is true:

$$\text{path width} \cdot 0.5 > \sin(\alpha) \cdot \text{distance to measuring point}$$

$$\text{where, } \alpha = |\text{flight direction} - \text{angle of measuring point}|$$

After the detection of a potential collision, the drone must decide the order in which it attempts its avoidance approaches. This is meant to prevent the drone from flying an unnecessarily lengthy detour. This decision is made from one position and therefore if the edge of an obstacle that is perpendicular to the desired flight direction is not in range of the lidar, the drone has no way of determining the better avoidance path as it would have to try to avoid the obstacle on both sides before making a verdict.

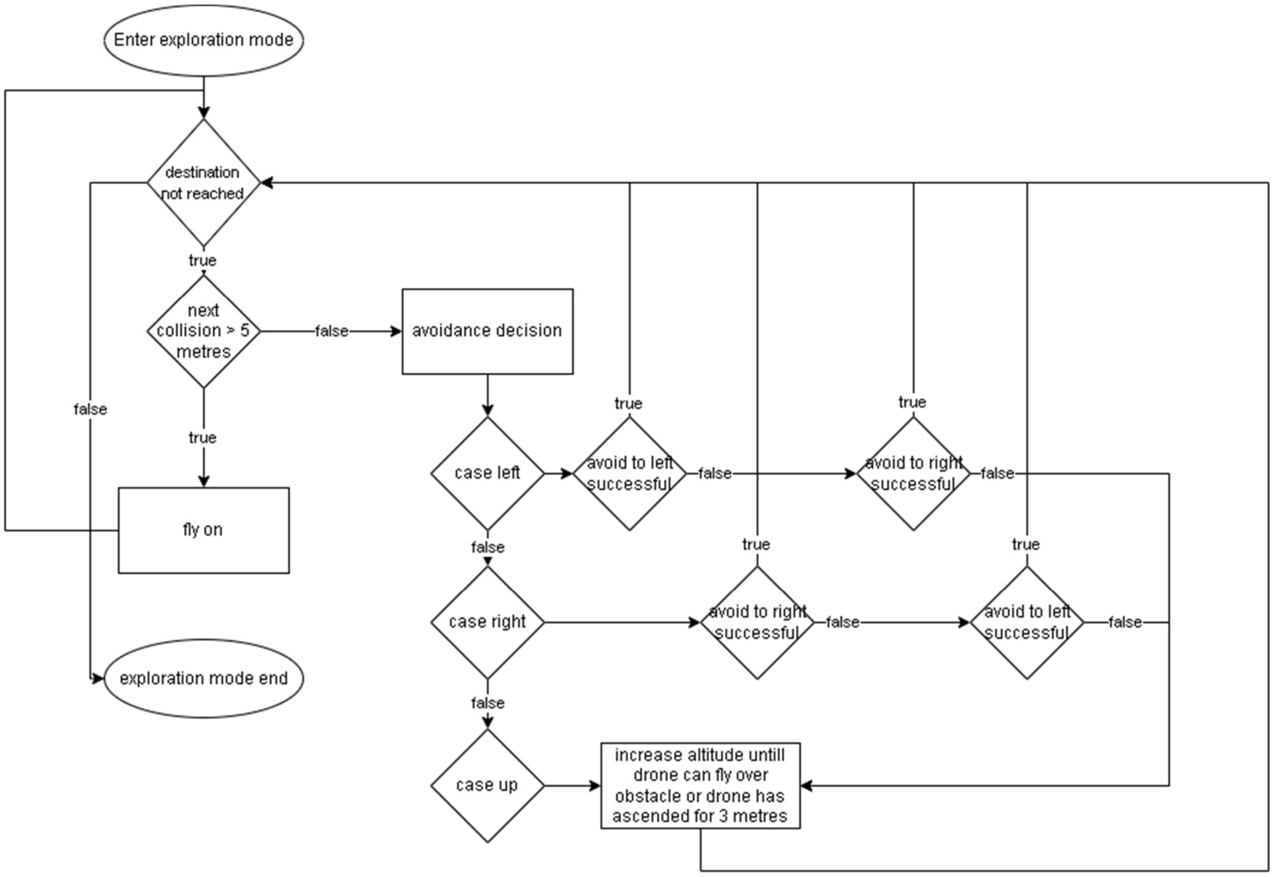


Figure 5: Flowchart of exploration mode algorithm

To make the avoidance decision the drone uses the next collision function in directions that deviate from the desired flight direction to the right and left. After a path without collisions has been found on one side, said side is chosen to avoid the obstacle. If no path without obstacles is found, the drone increases its altitude until it is able to fly over the obstacle.

As visible in the flowchart, the drone can also abandon an avoidance approach and switch to the other side. This happens when the obstacle demands the drone to fly backwards since the detour is then considered too lengthy. As a last resort, the drone will attempt to fly over the obstacle. If this too fails after ascending by 3 metres, the whole process is repeated until a path has been found.

Whenever the drone is actively avoiding an obstacle, the coordinates of the drone's position are saved to a list at regular intervals. A low interval translates to a high resolution of the recorded flight path with the disadvantage of occupying a lot of memory. In this regard, the list can be compared to a video where the single frames are not pictures but coordinates. These coordinates act as waypoints and enable the drone to have an approximation of a previously flown path.

One instance where these waypoints are used is when a horizontal avoidance approach is aborted. The drone must return to the point where the avoidance approach was first initiated before continuing with the procedure shown in the flowchart. It does so by reversing the waypoint list collected during the avoidance approach and passing it on to the waypoint mode.

4.5 Waypoint Mode (WM)

The waypoint mode allows the drone to fly a route that has been previously flown in exploration mode. As new obstacles could have appeared since the creation of the waypoints, the drone flies from one waypoint to the next in exploration mode. If obstacles have remained stationary since the creation of the waypoint list, exploration mode will fly the drone in a straight line from one waypoint to the next.

If new obstacles obstruct the straight line between two waypoints, the obstacle will be avoided according to the exploration mode procedure. The new waypoints that are collected during exploration mode will be inserted between those two waypoints.

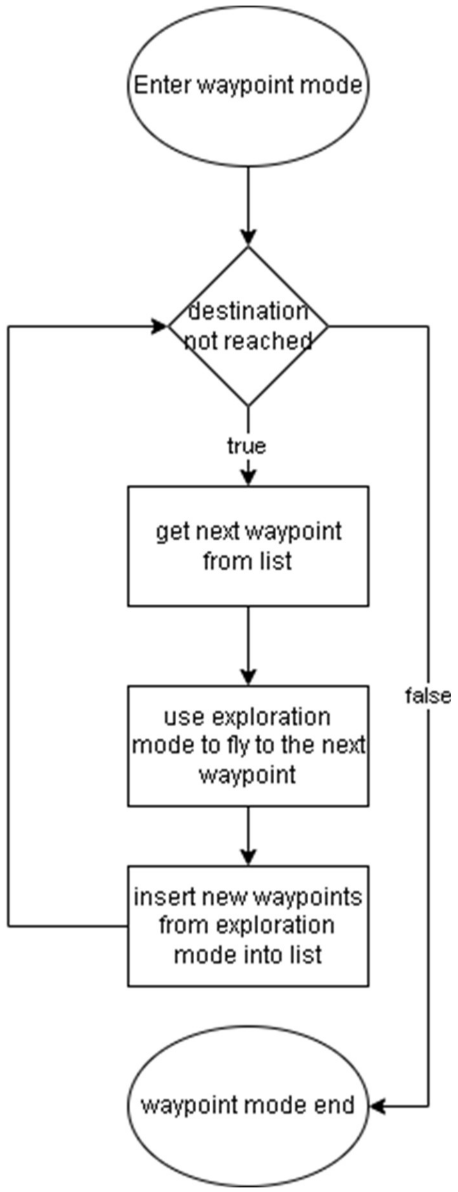


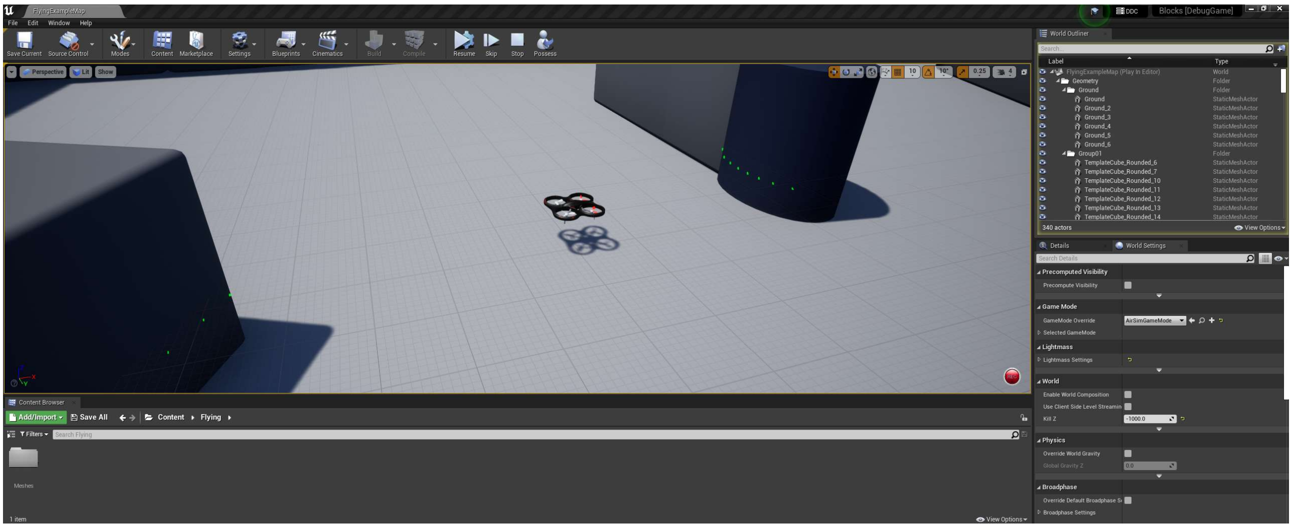
Figure 6: Flowchart of waypoint mode algorithm

4.6 Simulation

Testing the program in a real-world environment during development was time-consuming and weather dependent, which meant that even small changes could be tedious as they required test flights. Furthermore, there is always a certain risk that something does not work as intended, which could result in the drone crashing. Thus, the necessity of a simulation became noticeably clear.

AirSim[33] is a simulator built on *Unreal Engine*[34], which offers the possibility to customise both environment and vehicle. It allowed for quick test flights whenever needed without any risks. However, when implementing the new improvements to the actual drone, the shortcomings of the simulation become evident.

In the simulation, the GPS and the lidar are perfectly accurate and there is no wind. During test flights, this became a problem as the wind pushed the drone, which flew close to the obstacles already, even closer to the obstacles. Nevertheless, the simulation has benefited the development immensely.



Picture 6: AirSim in UnrealEngine

4.7 Safety Measures

As programming errors have much greater consequences in this project, safety measures and precautions are all the more important. As already mentioned, manual intervention with the remote control is possible at all times, but to ensure control of the drone, the *Pixhawk* must be informed when to ignore and when to obey input from the *Raspberry Pi*. To do this the *Pixhawk* has a dedicated flight mode called offboard control, which when disabled causes the *Pixhawk* to ignore all movement input from the *Raspberry Pi*. This mode is automatically disabled when the frequency of movement commands drops too low or stops altogether, which makes a crash of the *Raspberry Pi* harmless.

The most important safety measure is the flight direction correction vector. The flight direction correction vector is added to every velocity vector during active obstacle avoidance

[33] AirSim, “Documentation”. https://microsoft.github.io/AirSim/ (4.11.2022)
[34] UnrealEngine, “Official Website”. https://www.unrealengine.com/ (4.11.2022)

to ensure a safe distance from the obstacles. Its components are calculated using the following formula but only when the closest point is closer than five metres:

$$\text{x component} = \text{-cos (closest point's angle)} \cdot \text{(5 – closest point's distance)}$$

$$\text{y component} = \text{-sin (closest point's angle)} \cdot \text{(5 – closest point's distance)}$$

The resultant flight direction correction vector points away from the closest point and has a length such that adding the vector to the drone's position vector leads to a point that is 5 metres away from the closest point. This safety feature was originally meant to compensate for wind that would push the drone closer to an obstacle however it has allowed for a lower required path width when checking for collisions and is now active during every evasion manoeuvre.

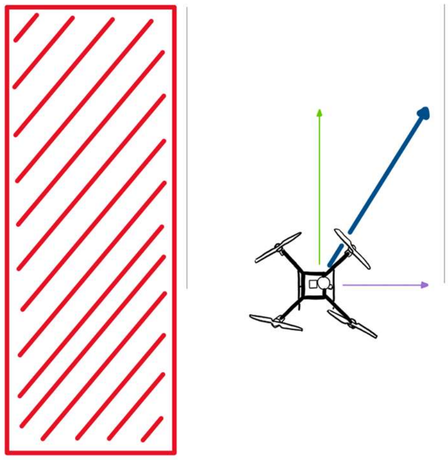


Figure 7: Obstacle (red), flight direction without correction vector (green), flight direction correction vector (purple), resultant vector (blue)

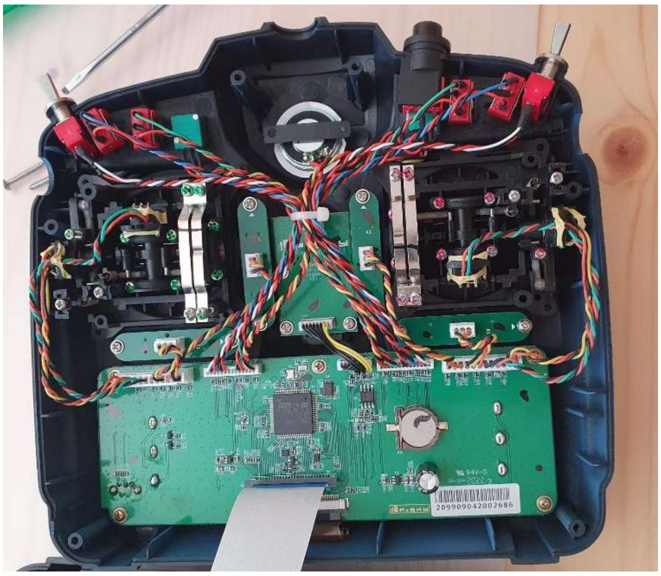
5. Challenges

5.1 Durability

Despite mindful treatment of the drone its resistance to damage was put to the test multiple times. Especially during the first flights, which included rough landings. The hardest of these tests occurred while controlling the motors with commands from the *Raspberry Pi* for the first time. The drone tipped over, which caused the floor to block the rotation of a propeller. When the motors spin, they experience a change in flux and, according to Lenz's law a current is induced such that the current creates a magnetic field that opposes the direction of the change in flux. Therefore, the induced current reduces the current that causes the motor to spin as they flow in opposite directions. If the motor is powered but does not spin, there is no change in flux and thus no reduction of the current causing the motor to overheat. However, the impact caused a solder joint to break, which saved the motor as it was not powered anymore. After inspection, no further damage was found, and the drone was able to fly again after resoldering the connection.

5.2 Broken Remote Control

Fortunately, the majority of purchased parts never malfunctioned. There was however one exception when the throttle stick of the remote control started picking up inputs that were never made. Since there was no time to have the remote control replaced or sent back for reparation, it had to be done at home. This was possible as replacement throttle sticks could be found in Europe.



Picture 7: The inside of the remote control

If the problem is caused solely by the throttle stick, swapping it with the stick that controls the yaw and roll should cause the problem to swap sides as well. But oddly enough the problem disappeared altogether, and the remote control has not caused any issues since.

5.3 Angle Normalisation

The coordinates of a measuring point, which the lidar has collected, can be calculated using trigonometry. However, the resulting coordinates belong to a reference frame that is fixed to the drone’s body, while movement commands and waypoints all operate in a NED frame. Coordinate transformation would solve this issue, but it is not a viable solution, as it would require various calculations for each and every measuring point. A more efficient approach is to add the drone's yaw value to the angles of the measuring points.

This however causes a different problem to arise, as now angles can exceed a full revolution. When determining whether the desired path is blocked, the angle difference between the measuring angles and the desired flight direction is calculated. If this difference is bigger than one-fourth of a revolution, the measuring point is not further considered for potential collisions. Therefore, measuring points that indicate a collision will mistakenly be ignored in situations like the following: desired flight direction (blue arrow) of 165 degrees, yaw (purple arrow) of 270 degrees, measuring point (red) at an angle of 270 degrees with respect to the drone's front.

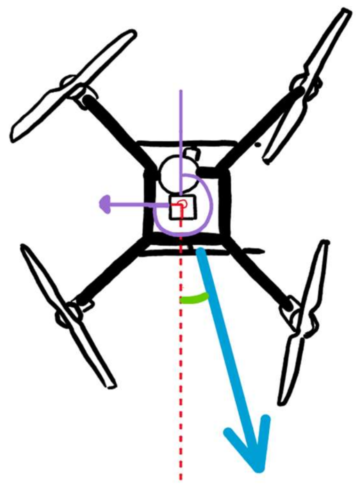


Figure 8: Drone seen from above, flight direction (blue), yaw (purple), lidar beam which detected measuring point in question (red), calculated angle (green)

measuring point angle (270°) + yaw (270°) – desired flight direction (165°) = 375°

To prevent these situations the angles must be normalised, which then results in the actual angle difference of 15 degrees.

5.4 Ground detection

During the first test flights of the avoidance algorithm, the drone seemed to detect non-existent obstacles. After careful observation, a pattern could be observed where the false detections occurred at an increased frequency while the drone was accelerating. From this, it became evident that the drone was detecting the ground as an obstacle. Therefore, it was necessary to accelerate less aggressively. This could however be done intentionally to determine the distance to the ground by multiplying the measured distance with the sine of the drone’s inclination towards the ground. This would provide a safe way for the drone to descend.

6. Conclusion

The drone has technically fulfilled all objectives which is demonstrated to a certain extent using the actual drone. The program’s potential is shown to the full extent in the simulation as it allows for the creation of an optimal environment to show all features over a short distance. Despite the achievement of the objectives, the exploration mode could be significantly improved by implementing SLAM and a better algorithm to decide the avoidance approach or a way for the drone to descend safely. Adding more distance sensors is also a must as a single lidar is not capable of offering industrial-grade obstacle detection. Nevertheless, it is remarkable how much information about a three-dimensional world can be gathered using a single two-dimensional sensor along with its orientation in space. Therefore, using multiple lidars appears to be a working alternative to the use of cameras. However, the economic viability remains questionable, as they are rather expensive especially when multiple lidars are required for a single drone. It is thus improbable that any enterprise that develops autonomous delivery drones will abstain from using cameras and resort to lidars instead.

Videos showing the code in action can be found here:

Simulation

https://www.youtube.com/watch?v=uRNTako1OuE

Test Flight

https://www.youtube.com/watch?v=K-50ah2NfNc

The code described throughout the paper is open-source and can be found on GitHub: https://github.com/matteogiger/AutoDrone

The scope of this project has expanded across more fields than I initially expected, all bringing their own challenges to the table. I now better understand why developing a vehicle is done by a team of experts from different fields. Tenacity and perseverance proved to be fundamental in developing a successful model. I am glad to have gotten the opportunity to work on a project that combines my various passions for technology.

7. References

AirSim, "Documentation". https://microsoft.github.io/AirSim/ (4.11.2022)¨

Arduino: https://www.arduino.cc/ (30.11.2022)

BAZL, "Flugeinschränkungen in der Schweiz". https://www.bazl.admin.ch/bazl/de/home/drohnen/drohnen-und-flugmodelle/drohnenkarte.html (29.10.2022)

BAZL, "Update: EU-Drohnenregulierung in der Schweiz". https://www.bazl.admin.ch/bazl/de/home/drohnen/drohnen-und-flugmodelle/Europaeische_Drohnenregulierung_uebernommen.html (29.10.2022)

Bundesrecht, "Verordnung des UVEK über Luftfahrzeuge besonderer Kategorien". https://www.fedlex.admin.ch/eli/cc/1994/3076_3076_3076/de - fn-d39694e131 (29.10.2022)

Bundesrecht, "Verordnung des UVEK über Luftfahrzeuge besonderer Kategorien". https://www.fedlex.admin.ch/eli/cc/1994/3076_3076_3076/de - fn-d39694e131 (29.10.2022)

Charles Tytler, "Modeling vehicle dynamics". https://charlestytler.com/modeling-vehicle-dynamics-euler-angles/ (20.7.2022)

Charles Tytler, "Quadcopter Equations Motion". https://charlestytler.com/quadcopter-equations-motion/ (20.7.2022)

Charles Tytler, "Modeling vehicle dynamics 6dof non-linear simulation". https://charlestytler.com/modeling-vehicle-dynamics-6dof-nonlinear-simulation/ (20.7.2022)

David Pierce, "Delivery drones are coming: Jeff Bezos promises half-hour shipping with Amazon Prime Air". https://www.theverge.com/2013/12/1/5164340/delivery-drones-are-coming-jeff-bezos-previews-half-hour-shipping (29.10.2022)

Dennis Baldwin, “Connecting Raspberry Pi w/ Pixhawk and Communicating via MAVLink Protocol”. https://www.youtube.com/watch?v=DGAB34fJQFc (9.5.2022)

Droneomega.com, “The Beginner’s Guide to Drone Motor Essentials”. https://droneomega.com/DRONE-MOTOR-ESSENTIALS/ (14.8.2022).

Eric, “RC LiPo Battery Guide: Explanation, Safety and Care”. http://learningrc.com/LiPo-battery/ (17.08.2022).

FrSky Taranis X9D+ SE 2019: https://www.frsky-rc.com/product/taranis-x9d-plus-se-2019/ (8.8.2022).

FrSky X8R: https://www.frsky-rc.com/product/x8r/ (8.8.2022).

FrSky X8R Specifications: https://www.frsky-rc.com/product/x8r/ (13.8.2022).

LDRobot LD06: https://www.ldrobot.com/product/en/98 (5.8.2022).

LDRobot, “Development Manual”. https://www.ldrobot.com/download/en/98 (4.11.2022)

Matt, “Raspberry Pi Power Consumption Data”. https://www.raspberrypi-spy.co.uk/2018/11/raspberry-pi-power-consumption-data/ (20.8.2022).

MAVLink, “Packet Serialization”. https://mavlink.io/en/guide/serialization.html (4.11.2022)

MAVLink, “Messages”. https://mavlink.io/en/messages/common.html (4.11.2022)

Oscar Liang, “How to choose FPV drone motors”. https://oscarliang.com/quadcopter-motor-propeller/ (14.4.2022)

Oscar Liang, “FPV drone flight controller explained”. https://oscarliang.com/flight-controller-explained/ (14.4.2022)

Pixhawk 4: https://docs.px4.io/main/en/flight_controller/pixhawk4.html (2.8.2022).

PX4, “Open-Source Autopilot For Drone Developers”. https://px4.io/ (4.11.2022)

Raspberry Pi 3B+: https://www.raspberrypi.com/products/raspberry-pi-3-model-b-plus/ (2.8.2022).

Red Bull UAE, “Drone Racing: The Sport of the Future”. https://www.redbull.com/mea-en/drone-racing-the-sport-of-the-future (20.8.2022).

Spencer Soper and Matt Day, “Amazon drone crashes hit Jeff Bezos’ delivery dreams”. https://www.bloomberg.com/news/features/2022-04-10/amazon-drone-crashes-delays-put-bezos-s-delivery-dream-at-risk?sref=VV6WcQFL (29.10.2022)

Stephen, Dade, “Pixhawk (and APM) Power Consumption”. https://diydrones.com/profiles/blogs/pixhawk-and-apm-power-consumption (20.8.2022).

SunnySky v3508 580kv: https://sunnyskyusa.com/collections/v-motors/products/sunnysky-v3508-motor?variant=45680383759 (5.8.2022).

SunnySky EOLO 11inch propeller: https://sunnyskyusa.com/collections/polymer-propellers/products/eolo-cn11-4-5-prop (5.8.2022)

SunnySky v3508 580kv Specifications: https://sunnyskyusa.com/collections/v-motors/products/sunnysky-v3508-motor?variant=45680383759 (5.8.2022).

SunnySky v3508 580kv Performance table: https://sunnyskyusa.com/collections/v-motors/products/sunnysky-v3508-motor?variant=45680383759 (5.8.2022).

SunnySky v3508 580kv Performance table: https://sunnyskyusa.com/collections/v-motors/products/sunnysky-v3508-motor?variant=45680383759 (5.8.2022).

Swaytronic Data source to derive battery mass to battery capacity function: https://www.swaytronic.ch/RC-Akkutechnik/Akkus-nach-Spannung/LiPo-Akku-4S-14.8V/?order=price-asc&p=1&properties=f257c03cbb114d90b253f34c34c17863 (20.8.2022).

Swaytronic LiPo 4S 10000mah:https://www.swaytronic.ch/Swaytronic-LiPo-Akku-4S1P-14.8V-10-000mAh-35C-70C-XT90 (17.8.2022).

UnrealEngine, “Official Website”. https://www.unrealengine.com/ (4.11.2022)

8. List of Illustrations

8.1 List of Pictures

8.2 List of Figures

8.3 List of Tables

8.4 List of Graphs

9. Acknowledgements

I would like to thank my supervisor Michael Oettli for his guidance and for believing in my project. I would also like to thank my family for their support, especially to my father for recording my test flights and helping me safely perform them.