Doctoral Thesis Reviewed
by Ritsumeikan University

Vision based localization, positioning and movement on the ceiling using a multirotor UAV

(マルチロータ型 UAV を用いた天井面での視覚情報に基づく位置特定，位置決めおよび移動)

March 2018

2018 年 3 月

Doctoral Program in Advanced Mechanical Engineering and Robotics

Graduate School of Science and Engineering

Ritsumeikan University

立命館大学大学院理工学研究科

機械システム専攻博士課程後期課程

LADIG, Robert Oliver

ラディック　ロバート　オリバー

Supervisor：Professor HIRAI Shinichi

研究指導教員：平井　慎一　教授

Acknowledgment

I would like to especially thank my mentor and main adviser: Prof. Kazuhiro Shimonomura of the Ritsumeikan University. Without his ongoing support, patience, kindness and guidance, I would probably not have been able to proceed with my Ph.D study. I would also like to thank my other advisor, Prof. Shinichi Hirai, for taking me as a Ph.D student and always offering advice. I would also like to thank former Guest Prof. Takeharu Goji Etoh for his advice, support and guidance in times when I needed it most and Prof. Shigeru Takayama for his lectures and for enabling me to produce my custom circuit boards for this project. I would also like to thank Prof. Sadao Kawamura for his lectures and his advice and acknowledgement when it was most needed and Prof. Ryuta Ozawa for his excellent lectures and always constructive and insightful input.

Furthermore I would like to thank all the people responsible for giving me the incentive and opportunity to study in Japan. Prof. Dr. Oliver Brock of the Technical University Berlin for introducing me to the field of Robotics and sharing his fascination of the topic with me. Jutta Borchert of the Humboldt University Berlin who believed in an engineer who was motivated to learn the Japanese language, accepted me in one of her excellent courses and for offering me a seat in a language exchange program to Japan. The whole staff and all lecturers of the Konan University Kobe who gave me an unforgettable, unique insight into Japan during my exchange program. Prof. Ralf Bebenroth of the Kobe University for always offering advice and help during my stay here in Japan. Dr. Mortiz Marutschke, currently lecturer at the Ritsumeikan University, for giving me incentive to stay and do my master's degree here in Japan after I finished my language exchange program, and for always being a source of advice.

I am incredibly grateful to have received financial support during my Ph.D study from the Heiwa-Nakajima Foundation Scholarship for International Students and from the Ritsumeikan's Kenkyu-Shorei Scholarship. Not only did they highly motivate me, but also made studying and daily life in Japan possible in the first place.

I also want to thank all my friends for supporting and motivating me even when I am often thousands of kilometers away from them. Especially I would like to thank Mr. Christopher Lamm for his invaluable advice, for always asking the right questions and for keeping my spirits up.

Lastly I want to thank my family, not only for the immense financial support and sacrifices that needed to be made because of my decision to continue my study in Japan, but also for showing me nothing but love and affection throughout my whole life.

In loving memory of my grandfather, Herman Zuther, who gifted me with the fascination of exploring the insides of electronic devices and without whom I might not have learned which side of a soldering iron is "up".

Abstract

Multirotor unmanned aerial vehicles (UAVs), especially small sized UAVs, continue to be popular in the consumer market and have become a constant factor in our daily life. There are some applications that require physical interaction with the environment by using an effector, such as a robotic hand attached to the airframe of the aerial robot, which is called aerial manipulation. Aerial manipulation is expected to extend the range of possible applications, in particular, for inspection and maintenance of infrastructures. In this study, a force-sensing, top-mounted omni-wheel drive system fit for a multirotor system has been proposed. This work attempts to explore and solve two main requirements for a small size UAV platform for aerial work to be viable as an industrial robot. One requirement that is nowadays taken for granted in ground robotics is the need for high precision movement. In the scope of this project, this is achieved by designing, building and implementing an aerial robotic system that is able to establish contact with the ceiling with a force sensing omni-wheel drive system, creating a system state towards the ceiling similar to a conventional ground robot towards the ground. The second big requirement, high precision localization, is explored and achieved by utilizing a dual camera AR marker system that uses sensor fusion for high precision absolute indoor localization on an on-board ARM processor. The feasibility of this system is successfully tested in various experiments in controlled and real-life environments.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Since the reduction in price and widespread availability of high precision positional sensors such as digital gyroscopes and accelerometers since the late 2000s, we can observe the rise in popularity of small size, unmanned aerial vehicles (UAVs), also called micro air vehicles (MAVs) or often simply drones, in the hobbyist industry [4]. They are used primarily for recreational flying and occasionally are used privately or in the photo/film industry to take aerial photos or videos. But UAVs also have enormous potential to be used in industrial manipulation and observation tasks due to their high mobility, low mechanical complexity, high speed and relatively low cost. Apart from widespread military use and very rare use in search and rescue operations as documented by Murphy [5], there is little utilization of MAVs as a robotic assistance platform in current industrial environments. There have been many studies on MAVs for manipulation, surveillance and transport ([6],[7],[8],[9],[10],[11],[12],[13]). However, to our knowledge, aerial robotics platforms have yet to be specifically developed and deployed regularly for a specific industrial task. This approach has been common practice for years in regards to other robotics platforms such as specialized manipulators or ground based robotic vehicles [14]. The goal of this research is therefore not only to develop a novel aerial robotics platform, but also to design and test this autonomous aerial platform following the specifications of real-life environments

and evaluate the results against real-life industrial requirements.

1.2 Related Works

The decision to solve this task with an aerial vehicle came after considering the characteristics of the work space, a ceiling, and previous works on developing a robot that can traverse a ceiling. Most popular here is the use of biomimetics and design orientations towards animals that are able to walk on wall and ceiling as e.g. the robot developed by Unver and Sitti [15]. While the precision that can be achieved by this method is excellent, one main problem with this approach is the low speed and the low payload capacity. There are also several works of aerial robots that are able to achieve contact with the wall or ceiling e.g. by creating a sphere shaped protection cage developed by Briod et al. or using passive rolls and a motor swivel system shown by Kawasaki *et al.* [16] [17].

Although they are able to establish contact with the ceiling, a problem those methods cannot solve is the tight movement precision specifications for the task we are aiming for. For all of these solutions the main propulsion device is the combined thrust generated by the MAV's propellers which is prone to disturbance by air turbulences, especially when being close to the ceiling and when the MAV is operating in a closed, confined space. Furthermore, a multirotor, which because of its simple blueprint and low price is the most feasible type of MAV design in our scenario, is an underactuated system i.e. we have to tilt in a certain direction to achieve translational movement along the horizontal axis. If we want to work in close proximity with the ceiling, either for ink-marker painting or eventually for measuring, we lose the ability to tilt and therefore translate the position of the multicoper. This problem that has been mechanically complex but interestingly addressed by [17]. While there is the possibility to choose a higher distance between an end effector that is interacting with the ceiling (in our work a print head) and the MAV by attaching a manipulator to the MAV, there is the high risk of tipping the platform due to a very high center of gravity. This has been shown and explored by Jimenez *et al.* [18].

There are also several previous examples of research into aerial robot manipulation. Kim *et al.* and Lucia *et al.* presented an aerial robot with an attached two-DOF manipulator [19] [20]. An example of a more complex manipulator, mounted under a drone, is presented by Orsag *et al.* [21]. They used the manipulator to accomplish the task of grasping an object midair. Another example of research into aerial robot manipulation is the work of Thomas, Loianno *et al.*, who used a novel method inspired by biomimicry [22]. Since perching is a fundamental ability of many birds and insects, many researchers exploring this topic are using the principle of biomimicry as well. Jiang *et al.* describe an aerial robotics platform which uses micro-spines to perch on walls, which is heavily inspired by the physiology of insects [23]. Another similar approach, especially useful for small-sized micro air vehicles, uses specially shaped landing gear to maximize perching success, as shown by Fukushima and Kawaguchi [24]. However, these methods are not suitable for approaching an overhead workspace in an uncontrolled environment. The reason being that they either consider only the workspace under the airframe or rely on a controlled environment by using motion capturing equipment to extract the rigid body dynamics of the robot, which requires extensive setup time. One work of note regarding the workspace above the aerial robot is the work of Kumar *et al.*. They attempted to approach a bar-like object, and in a subsequent paper describe how they managed to perch on a bar-like object from the bottom [25] [26].

1.3 Scope of Dissertation

Chapter 2 gives an overview of the base dynamics and aerodynamics of a multirotor system followed by a short overview of the reason for the current rise in popularity of multirotor in the public and also in the research community. Furthermore, current and potential applications for aerial robotics and multirotor technology and a manifold of different industries are explored. To close the chapter, a short overview of the commercially available flight controllers used in this work is given. Chapter 3 gives an overview of early approaches that lead towards this project. This chapter was

added to show other potential solutions towards the precise movement of a multirotor system that have been explored for a certain amount of time, but dismissed because they have proven to be unfeasible or did not show enough potential to be pursued further. The chapter was added because the solutions shown here arise frequently as questions during QA sessions regarding this project and explains why exactly these solutions don't show enough potential. Chapter 4 explores different solutions for an embedded vision system for localization. Since the system's potential deployment area is indoors or in other GPS denied locations, a GPS-less, on-board localization method is explored in this chapter. Three potential solutions are explored. Two solutions, one marker based and one image feature based, are conceptualized, explained in detail and then implemented on a FPGA board. The third method is an AR marker based dual camera localization method, implemented on a ARM processing board. This will become the final localization method of the system, proposed in this work. Chapter 5 explains the concept and hardware behind positioning on the ceiling with an omni-wheel drive system. Due to the length of the projects development several different versions of the hardware exist and have been tested separately. To account for this, the chapter has a separate section for each version of the hardware. At the end of the chapter, the robot-human interface for simultaneously controlling the 7DOF of this system, a custom remote controller, is shortly introduced. Chapter 6 explores how a force feedback control can be implemented in a multirotor aerial robotic system. Chapter 7 gives an overview of the experiments performed with this system. Again, the chapter is divided into experiments for each version of the hexrotor system since the hardware differs in between the different version and is only comparable within limits. Conclusions are given for each of the system versions. At the end of the chapter, real-life applications, like paint marker setting on the ceiling are explored and tested in experiments. Chapter 8 gives a short overview of potential future work and a summary of this thesis.

Chapter 2

Multirotor UAV

2.1 Basic Dynamics

To explain basic multirotor dynamics, we will look at one of the simplest designs of this kind of platform, the quadrotor. As seen in Fig. 2.1 roll and pitch can be archived by controlling the thrust generated by each motor. Two motors spin clockwise and two spin counter clockwise. To control the yaw of the platform, the speed of the clockwise or counterclockwise spinning must be adjusted respectively. The system is unteractuated. According to Newton's law of motion, we can describe the dynamics of a mechanical system as:

$$\ddot{\vec{q}} = f(\vec{q}, \dot{\vec{q}}, \vec{u}, t), \tag{2.1}$$

with $\vec{q} \in \mathbb{R}^n$ as the position vector, $\vec{u} \in \mathbb{R}^m$ as the control input vector and t as time. Looked at independently from the control inputs, we can write:

$$\ddot{\vec{q}} = f_1(\vec{q}, \dot{\vec{q}}, t) + f_2(\vec{q}, \dot{\vec{q}}, t)\vec{u}. \tag{2.2}$$

When in this form, a system is underactuated when:

$$\operatorname{rank}\big[f_2(\vec{q}, \dot{\vec{q}}, t)\big] < \dim\big[\vec{q}\big], \tag{2.3}$$

which is the case for all standard multirotor designs. Because the system is underactuated, the remaining DoF, which is the translation along the x-y plane, must be controlled through the system dynamics.

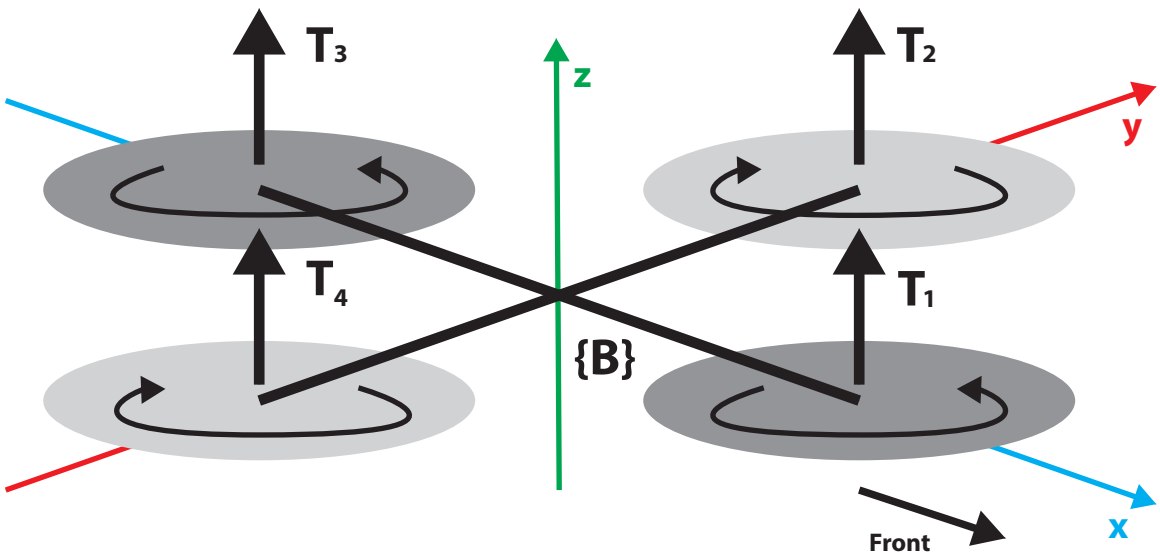

Figure 2.1: Sketch diagram of a quadrotor.

2.1.1 Rigid Body Dynamics of a Multirotor

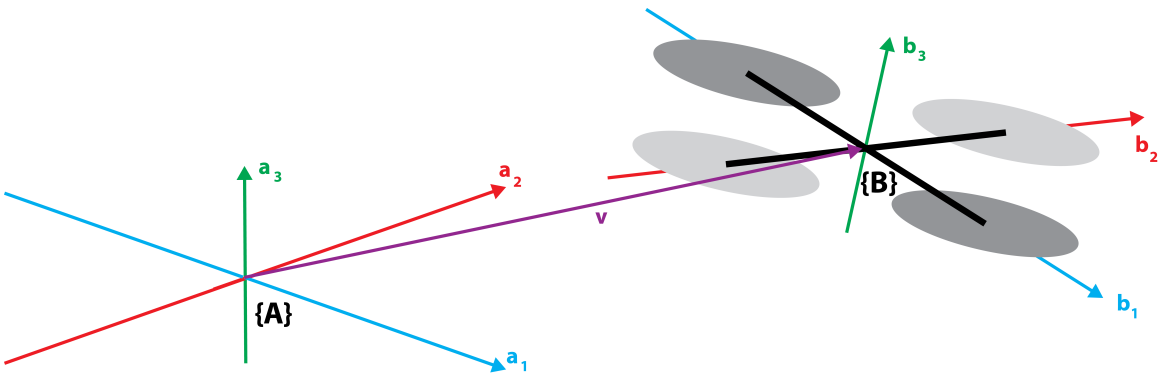

Figure 2.2: Sketch diagram of a quadrotor regarding rigid body dynamics. $\{A\}$ denotes the inertial frame and $\{B\}$ the body fixed frame.

Let $\{A\}$ be a right hand inertial frame with unit vectors named $\{\vec{a_1}, \vec{a_2}, \vec{a_3}\}$, where $\vec{a_1} = \vec{x}, \vec{a_2} = \vec{y}, \vec{a_3} = \vec{z}$ (Fig. 2.2). The position of the center of the multirotor would be $r = (x, y, z) \in \{A\}$. Let $\{B\}$ be the right hand body fixed frame of the multirotor frame with the unit vector $\{\vec{b_1}, \vec{b_2}, \vec{b_3}\}$. Now we construct a rotation matrix R, so that

$\vec{b_1} = R\vec{x}, \vec{b_2} = R\vec{y}, \vec{b_3} = R\vec{z}$. Using $Z - X - Y$ Euler angles to get from $\{A\}$ to $\{B\}$, we first rotate around a_3 with the yaw angle ψ , followed by a rotation around a_1 with the roll angle ϕ and lastly a rotation around the pitch angle θ:

$$R = \begin{pmatrix} c\psi c\theta - s\phi s\psi s\theta & -c\phi s\psi & c\psi s\theta + c\theta s\phi s\psi \\ c\theta s\psi + c\psi s\phi s\theta & c\psi c\phi & s\psi s\theta - c\psi c\theta s\phi \\ -c\phi s\theta & s\phi & c\phi c\theta \end{pmatrix}, \tag{2.4}$$

where c and s stand for cosine and sine respectively.

With $\vec{v} \in \{A\}$ donating the linear velocity of $\{B\}$ with respect to $\{A\}$, $\vec{\Omega} \in \{B\}$ the angular velocity of $\{B\}$ with respect to $\{A\}$, m the mass of the multirotor and $I \in \mathbb{R}^{3\times 3}$ the constant inertia matrix, we can write down the rigid body dynamics of a multirotor as:

$$m\dot{\vec{v}} = mg\vec{a_3} + R\vec{F} \tag{2.5}$$

$$\dot{R} = R\Omega_\times \tag{2.6}$$

$$I\dot{\vec{\Omega}} = -\vec{\Omega} \times (I\vec{\Omega} + \vec{\tau}), \tag{2.7}$$

with $\vec{F}, \vec{\tau} \in \{B\}$ the combined vectors of the principle non conservative forces and moments applied to the airframe by the aerodynamics of the rotors as well as $\Omega_\times$ being the skew-symmetric matrix, so that $\Omega_\times \vec{v} = \vec{\Omega} \times \vec{v}$ for all $\vec{v} \in \mathbb{R}^3$.

2.1.2 Multirotor Aerodynamics

Multirotor aerodynamics depend heavily on the aerodynamics of the rotors. Rotor aerodynamics have been extensively studied and detailed modes can be found in other recent studies [27] [28]. For this work we are using a simpler model and concentrate on the steady state thrust generated by a hovering rotor as shown by Mahony *et al.* [29]:

$$T_i := C_T p A_{ri} {r_i}^2 {\omega_i}^2, \tag{2.8}$$

where for rotor i the rotor disk area is A_{ri}, r_i the radius, ω_i the angular velocity, p the air density and C_T the thrust coefficient. The thrust coefficient depends on the

shape and profile of the rotor. We can simplify this by using the constant c_T, which can be determined from a simple static thrust test and:

$$T_i := c_T {\omega_i}^2. \tag{2.9}$$

Equally, we can determine the reaction torque due to motor drag and describe it as:

$$Q_i := c_Q {\omega_i}^2. \tag{2.10}$$

Assuming the trust is exactly in direction of b_3, the total thrust at hover applied to the airframe is, in the case of a quadrotor:

$$T_{\sum} = \sum_{i=1}^{4} |T_i| = c_T(\sum_{i=1}^{4} {\omega_i}^2). \tag{2.11}$$

The hover thrust is the primary component of the exogenous force:

$$\vec{F} = T_{\sum}\vec{z} + \Delta, \tag{2.12}$$

where Δ are other aerodynamic forces introduced by more complex rotor behaviour like rotor flapping.

The net moment $\tau = (\tau_1, \tau_2, \tau_3)$ of the aerodynamics is a combination of rotor forces and air resistance:

$$\begin{aligned} \tau_1 &= c_T(\sum_{i=1}^{4} d_i s\phi_i {\omega_i}^2) \\ \tau_2 &= c_T(\sum_{i=1}^{4} d_i c\phi_i {\omega_i}^2) \\ \tau_3 &= c_Q(\sum_{i=1}^{4} \sigma_i {\omega_i}^2), \end{aligned} \tag{2.13}$$

with d_i as distance of the rotor from the center of the frame, ϕ_i the angle between the multirotor arms and $\sigma_i \in -1, +1$ depending on if the rotor is turning counterclockwise or clockwise. We can combine formula 2.11 and formula 2.13 exemplary in a matrix

as for a quadrotor with equal distances $d_i = d$ and $\Phi_n = n\frac{\pi}{2}$ as follows:

$$\begin{bmatrix} T_\sum \\ \tau_1 \\ \tau_2 \\ \tau_3 \end{bmatrix} = \begin{bmatrix} c_T & c_T & c_T & c_T \\ dc_T c(\Phi_1) & dc_T c(\Phi_2) & dc_T c(\Phi_3) & dc_T c(\Phi_4) \\ dc_T s(\Phi_1) & dc_T s(\Phi_2) & dc_T s(\Phi_3) & dc_T s(\Phi_4) \\ -c_Q & c_Q & -c_Q & c_Q \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix}$$

$$\begin{bmatrix} T_\sum \\ \tau_1 \\ \tau_2 \\ \tau_3 \end{bmatrix} = \Gamma \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix}. \tag{2.14}$$

For a quadrotor to hover, we must choose a suitable ω_i by inverting the constant matrix Γ, so that $\tau_1, \tau_2, \tau_3 = 0$ and $T_\sum = mg$. This hover state is defined in this work as the "neutral state" of the aerial platform.

2.2 Reasons for the Current Popularity of Multirotors

While there are historic reports of quadrotors or similar VTOLs being developed as early as 1907, e.g. the Breguet-Richet Gyroplane shown in Fig. 2.3 [30], one could argue that the most rapid developments in regards to VTLOs happened in the late 1990s to early 2000s. There are several important reasons why we see a recent increase in interest from the academic and the public in regards to quadrotors, low cost unmanned drones and VTOLs in general. In this chapter we will look at three reasons in detail: The development and refinement of brushless motors in the late 1990s, the introduction and widespread availability of small size digital gyroscopes in the late 1990s and the introduction of high precision GPS to the civil public in 2000. These technologies are key technologies in regards to multirotors, showing that recent rise in popularity of multirotor in consumer electronics and research is not only a trend or a fad of our time, but a logical and consequent result of the technology available.

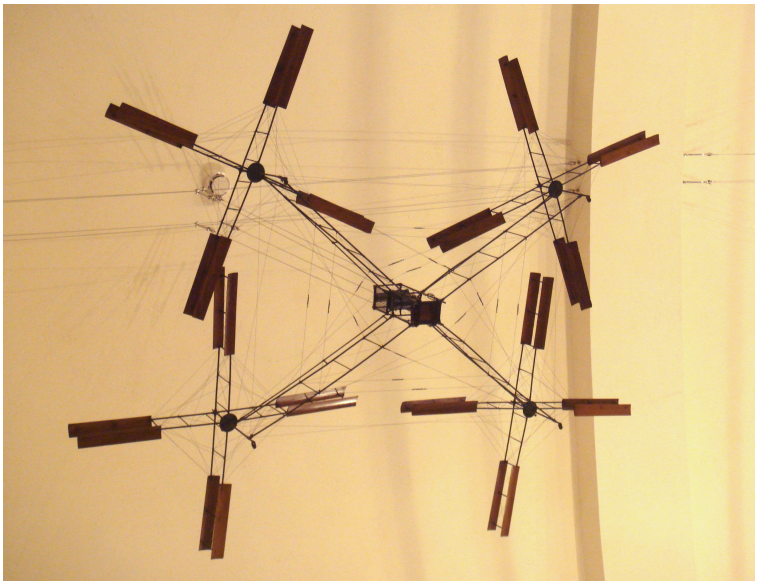

Figure 2.3: Photo of the Breguet-Richet Gyroplane 1, constructed in 1907. The construction made from fabric covered four-blade biplane rotors with one pair turning clockwise and one pair turning counter-clockwise. This makes it very similar to quadrotors used today. © PHGCOM, CC BY-SA 3.0

2.2.1 Brushless Motors

One reason for the rising popularity of small sized aerial vehicles is the development and widespread availability of low-cost brushless motors. While brushed motors have been around for nearly a century, brushless motors are a relatively young invention developed in the 1960s [31]. Continuous development and improvements in size and efficiency were made during the 1980s due to the increased availability of better permanent magnets. This development continued through the 90s to today. Brushless motor, in comparison to their brushed counterpart, have a fixed armature, with permanent magnets rotating around this fixed armature (Fig. 2.4). Instead of the mechanical solution that is the brush-communicator assembly of a brushed motor, a brushless motor uses the windings in its assembly to constantly change its phase and keep the motor running. Very important for the use case that is looked upon in this work is the lack of centrifugal forces in a brushless motor, since the windings

are not located in the rotor. Other advantages of a brushless motor compared to brushed motors are the higher efficiency, high weight to torque ratio, lower noise, better reliability and lifetime due to the lack of touching parts between brush and communicator, less electromagnetic interference (which is important in our use-case since any electromagnetic interference will affect the performance of the measurements of the inertial measurement unit mounted on our multirotor), as well as the lack of a spark within the electro motor which can be relevant in hazardous, easily inflammable environments. One of the major drawbacks of brushless motors is that due to the phase shift principle they are driven by, they require a micro controller to control said phase shift and cannot be driven directly via voltage as their brushed counterparts. In case of a multirotor design, those controllers are called electronic speed controllers (ESC) and are required for each motor in a multirotor. While this raises the electronic complexity and points of failure in the overall design, the many advantages of brushless motors over brushed motors overshadow these small drawbacks. There are three performance parameters that are commonly used for brushless motors which are K_t, K_e and the from K_e derived Kv:

$$K_t = \frac{\text{NewtonMeter}}{\text{Amp}} \tag{2.15}$$

$$K_e = \frac{\text{VoltSecond}}{\text{Radian}} \tag{2.16}$$

$$\text{Kv} = \frac{1}{K_e}. \tag{2.17}$$

2.2.2 Small Size Digital Gyroscopes

With the invention of the electric motor and thus the ability to spin indefinitely when driven by a motor, the gyroscope has become an increasingly important component since the late 19th century. Particularly when used by naval forces to measure changes in heading during the first and second world war, the gyroscope and subsequently the gyrocompass have played a large role in aerial and nautical navigation [32]. After the second world war, the basic technology and principle was kept, but there was a strong motivation to miniaturize the overall mechanism to make it more feasible for

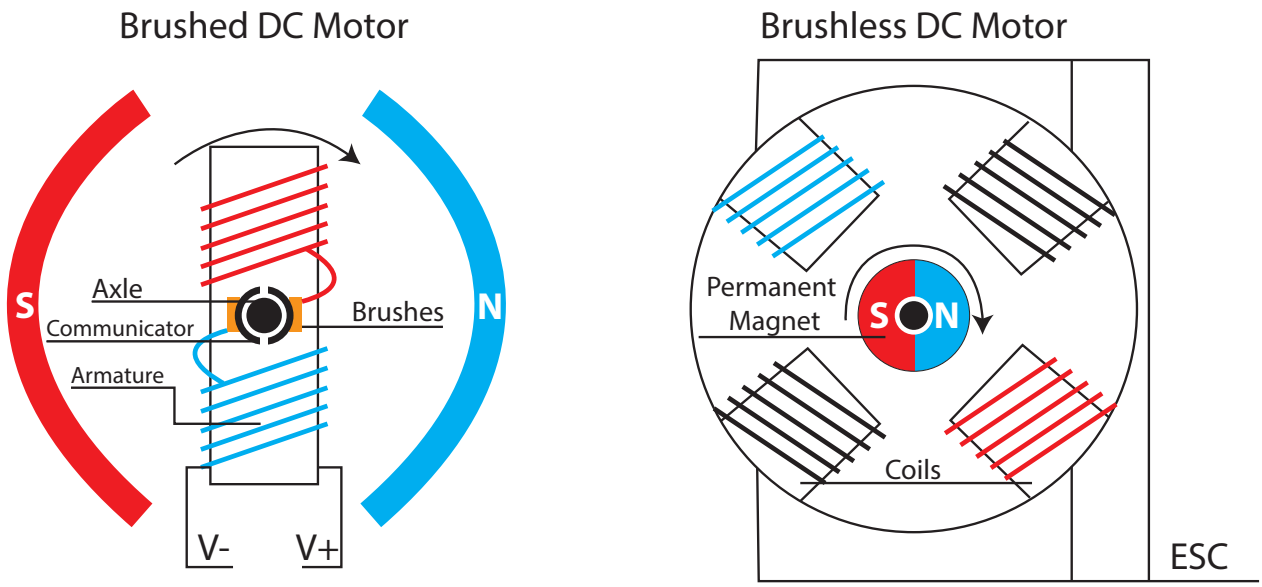

Figure 2.4: General schematic of a brushless and a brushed DC motor.

rocket and missile technology. To this day, a lot of military hardware still employs a mechanical gyroscope for flight control, radar stabilization or fire control [33]. Since a full-fledged gyroscope is not affordable nor possible with the limited payload of small size aerial vehicles there have been alternatives in the RC community, such as a simpler mechanical gyroscope for RC helicopters, often as big as 10×5×5cm [34]. While this construction managed to control the tail blade speed and thereby helped the operator control the general heading of the helicopter (on one axis only), it failed when there were major corrections to be made and had a relatively long response time. Due to the two disks that needed to constantly spin, it posed a constant drain on the battery. The overall size required for the mechanism and its weight were also a major drawback. Nonetheless, this was one of the main components for autonomous stabilization used by small sized RC controlled aerial vehicle hobbyists in the 1980s and 90s. With the rise of affordable, widely available microelectromechanical system (MEMS) gyroscopes, based on the vibrating gyroscope principle perfected in the late 90s, this technology of small size gyroscopes for autonomous stabilization became open for the civil industry and research to build, explore and experiment with [35].

Especially in customer electronics, this lead to a wide implementation of digital gyroscopes in various products such as video games, cameras for image stabilization and cellphones to use as digital compasses. It also revolutionized the RC helicopter scene and helped to make one of the main components of a multirotor IMU affordable, small and lightweight [34]. At the time of writing, the digital gyroscope has become so affordable and widespread, that is has become a standard part within every cellphone and basically every commercially available flight controller for small size aerial vehicles on the market today.

2.2.3 The Opening of GNSS to the Public

One key technology in the use of small sized aerial vehicles, especially in the context of widespread consumer electronics, is the use of a global navigation satellite system (GNSS). While the overall stability of a small size multirotor system can be provided by the onboard IMU, the overall position can be very easily disturbed since the neutral state of a multirotor or similar VTOL system is the hover state. Since the multirotor is an under-actuated system by design, we are not able to counter any disturbance for the translational axis x and y without correctional manoeuvres. The systems translational position is also very unstable, since there is only a very small amount of air friction:

$$f_{counter_{x,y}} = f_{drag} = \frac{1}{2} C \rho A v^2, \tag{2.18}$$

with C being the drag coefficient, ρ being the air density, v the speed of the UAV (in the hover state this becomes the wind speed) and A the cross section area of the UAV, which is by design traditionally very small, acting against these disturbances.

Or, in other words, the system is unable to withstand disturbance in the horizontal direction, due to the lack of counter-forces, except for air resistance (which is usually very low due to the overall lightweight construction of an average multirotor system). To make the control of a multirotor more enjoyable and safe without fighting against the constant influence of disturbance and risking the loss of control of the vehicle, a GNSS system such as the United States Global Positioning System (GPS) or Russia's

Global'naya Navigatsionnaya Sputnikovaya Sistema (GLONASS) is needed. GPS, the standard mainly used for all high quality commercial drones currently available, was first exclusively developed for the military starting in 1974. It became fully operational in 1995 with the launch of the 24th satellite added to this system in 1993 [36].

Noteworthy is the change from an exclusive military application of GPS to a limited civilian usage in 1983 by U.S. President Ronald Reagan in order to improve safety in the civil flight industry. This happened after a passenger plane bound for South Korea was shot down by a Soviet interceptor fighter after accidentally flying though a restricted airspace just north of Hokaido, Japan [37]. The GPS standard allowed for civilian use was much less accurate than the capabilities the military, since many in the US worried that enemies would use this open technology against them in the Cold War. Due to this so-called "selective availability" of GPS for civil use, which had a maximum accuracy of about 50 m (164 ft) horizontally and 100 m (328 ft) vertically, GPS was not feasible to be used in commercial products nor in research [38].

This however changed after the May 1st 2000 "Statement by the President regarding the United States' Decision to Stop Degrading Global Positioning System Accuracy", given by US president Bill Clinton [39]. It stated that the error introduction for public available GPS signals should be set to 0 by 2006. It was actually enforced by the end of the year of the announcement, in 2000 [40]. The second biggest standard of GNSS, GLONASS, followed this example and became available to the public in 2007 [41]. With the availability of highly accurate GNSS data began the rise of consumer products which are ubiquitous in our daily life today. GNSS navigation for cars quickly became a standard, mobile phones and even watches use GNSS technology now (often a chip able to receive signals from both of the biggest GNSS: GPS and GLONASS). GNSS has also become a main component in outdoor autonomous navigation and control for small size aerial vehicles. It plays a large role in making an aerial vehicle easier and safer to fly and this development contributed tremendously towards the popularity of small sized aerial vehicles in the leisure in-

dustry.

2.3 Use of Aerial Robotics in Different Industries.

One of the central questions that this work is exploring is the question of how to make aerial robotics feasible to use in an industrial environment. It is therefore important to have a look at different industries and the current use of aerial robotics in these industries at the time of writing. In the following chapters, the focus will be on the military and recreational use of UAVs, as well as the use of UAVs in the commercial industry. This includes applications in construction, film, agriculture, and search and rescue.

2.3.1 Military

The military has a long history of developing and using autonomous aerial vehicles. One of the earliest concepts of an autonomous aerial vehicle is the so-called "Hewitt-Sperry Automatic Airplane" (Fig. 2.5), also known as the "flying bomb" developed in 1917 during the first world war by the United States navy. It is considered the predecessor of cruise missiles that are used on many ships in the United States today. The Hewitt-Sperry Automatic Airplane failed in the conception phase and was never actually deployed due to several unsolved problems with its auto stabilization system. Because of these issues, the airplane always needed continuous remote inputs from an operator. [42].

One of the first vehicles that included all of the necessary components to be considered a full fledged autonomous aerial vehicle was the V2 rocket used by Nazi Germany in 1944 during the second world war [43]. Although ultimately this weapon did not have any significant impact on Germany's war efforts, Nazi Germany spent a significant amount of their war budget on the development of what is considered by many as the first truly autonomous aerial vehicle [44].This was the first implementation of feedback stabilization and gyroscopic guidance into an aerial vehicle. The V2 had a separate mechanical gyro for roll and yaw and one for pitch. The output signals

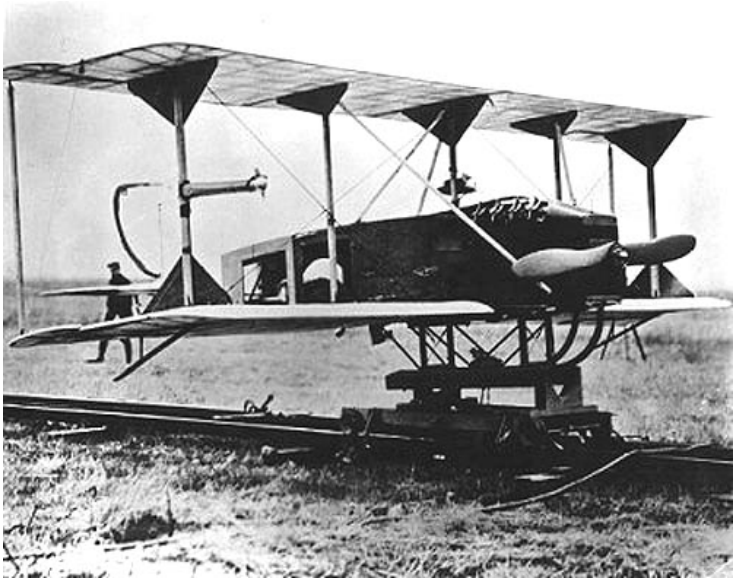

Figure 2.5: Historic photo of the Hewitt-Sperry Automatic Airplane, or the "flying bomb". It is one of the earliest concepts of an unmanned aerial vehicle and precursor to modern rocket technology.

from those gyros were forwarded to control vanes within a feedback loop to keep the rocket on its predetermined flight path. The capture of intact V2 models and the capture of researchers working on the V2 laid the foundation for the development of intercontinental ballistic missiles (ICBMs) in the US and USSR and eventually the carrier rockets that lead to the first manned space missions [45]. This period in which autonomous vehicles were synonymous with missiles could be called the "first era of autonomous vehicles" [43].

It is hard to get an insight into the development process of autonomous vehicles during the cold war crisis, since most of the development was confidential and not available to the public. One of the first times the general public was able to see the development of unmanned aerial vehicles in the military was during the military operation "Desert Storm" in 1990, where the RQ-2 Pioneer, developed by the AAI Corporation and the Israel Airforce Industries in the 1980s, was successfully used for reconnaissance tasks by the US Army. It reportedly collected a total of 1500 flight

hours during this operation alone. In contrast to conventional autonomous aerial vehicles, which basically consisted of cruise missiles at that time, the goal of this fixed wing vehicle was to stay in the air as long as possible.

The deployment of fixed wing UAVs in the 1990s was so successful that it lead to one of the first armed unmanned vehicles, the General Atomics MQ-1 Predator. It saw its first flight in 1994 and is probably the most well-known autonomous aerial vehicle due to its wide and at times controversial coverage by diverse media outlets [46] [47] [48]. It was first deployed in the United States "War on Terror" after the attack on the World Trade Center in September 2001. The controversy surrounding armed autonomous aerial vehicles, even within government branches, becomes apparent when looking at the fact that, while there were early tests conducted with firing anti-tank "Hellfire" missiles from the Predator in the early 2000s, the CIA only gave their approval to use weapons on this platform after the 9/11 attacks. This led to a completely new era of warfare, dubbed by many critics as the so-called "Drone wars" [49]. The strategy in this kind of warfare is to use remote controlled or autonomous vehicles to collect intelligence data or engage enemy forces and to completely avoid possible friendly human losses as far as possible by not deploying any human forces locally at all.

In context of this work, it is important to look at the development of autonomous VTOLs by the military. But apart from the usual helicopter designs e.g. the Boeing A-160 Hummingbird or coaxial rotor driven vehicles like the Swiss KOAX X-240, there have only been a few VTOL developments by the military in the mid 2000s and none had any significant deployment [50] [51].

It is also noteworthy that there have been no significant developments and deployments of unmanned VTOL in the military from the mid 2000s to today. A possible reason to that is the relatively limited range of VTOL drones in comparison to fixed wing drones due to limitations in the current battery technologies. It would mean deploying troops close to potential targets of reconnaissance, which is a strategy less favoured by most modern military troops [52]. Since small size VTOLs only have a limited use for the military industry in the conflicts today, it creates an opportunity

for the civil industries and research to make impacts in the field of autonomous UAVs. Research in the area on small size unmanned vehicles has shifted noticeably in recent years from military focused applications to a focus on civil applications (see chapter 1.2).

Due to the controversial connotation of the word "drone", as well as the enormous physical and psychological damage the deployment of armed aerial vehicles can have on the population of a country [47], this work will try to distance itself from the military use of aerial vehicles by trying to refrain from using the word "drone" as a simple equivalent to UAVs, multirotors or VTOLs as far as possible, except when specifically talking about the use of unmanned aerial vehicles in the military industry.

2.3.2 Leisure Industry

The history of commercially available remote controlled VTOL began in 1969 with the first remote controllable helicopter model kit developed by the German engineer Dieter Schueter [53]. Due to limited export capabilities, RC helicopters became a big trend in the early 70s West-Germany, and after the sale of model licenses to the Japanese company "Kalt", it also became a trend within Japan. Because of the limitations in gyroscope technology already discussed in chapter 2.2.2, apart from advances in the materials used for remote controlled helicopters, there have not been significant changes in the general structure of these devices. Also, because of the high initial costs of RC helicopters, as well as the space required for safe operation and the complex steering ability demanded of the operator, the RC helicopter stayed a very niche product exclusive to dedicated hobbyists.

With the development of affordable digital gyros and brushless motors (see chapter 2.2.1) there was a rapid change in the RC VTOL industry. Not only were a manifold of open source flight controllers developed (see chapter 2.4), but also the parts to build a working RC multirotor reached a price of under $1000 in the early 2000s, making it affordable for a wide array of people. The first widely successful commercial multirotor was the AR.Drone (Fig. 2.6a) by the French company Parrot. It was heavily marketed by Parrot and Apple as a companion toy to the Apple iPhone that was introduced

only 3 years before. While mainly marketed for leisure and taking pictures with the low resolution onboard camera, the AR.Drone and its predecessor were also a big revelation for academia. It eliminated the need to build a vehicle that can reliably navigate 3d space from the ground up and thus incited numerous studies involving 3d navigation [54] [55] [56].

Another company that influenced modern perceptions of autonomous aerial vehicles is the Chinese company DJI. While their first commercial model in January 2013, the Phantom 1, was only a minor success, they had major commercial and critical success with the introduction of the Phantom 2 in December 2013 (Fig. 2.6b) [57]. It was the first widely available commercial drone with built in GPS capabilities like "hold position" and "return to home", making it significantly easier to control even for absolute newcomers in this hobby. Positive reviews and continuing success with predecessors of this model made DJI the market leader in the area of small sized aerial vehicles, which caused the number of commercial competitors and open source solutions on the market to shrink dramatically [58].

(a) Photo of the Parrot AR.Drone. © Syahlevi, CC BY-SA 4.0

(b) Photo of the DJI Phantom 2. © DJI technologies, CC BY-SA 3.0

Figure 2.6: Two of the most commercial successful MAVs.

2.3.3 Commercial UAV industries

In 2014, the United States Federal Aviation Association (FAA) began lifting the active ban on the flight of commercial unmanned aerial vehicle which led to a large rise in the use of UAV in several industries. Data is available from the time the ban was lifted (Table 2.1), and the table shows the first 1000 exemptions from the FAA ban. It is clear that the most prominent industries requesting the use of UAVs were aerial photography, survey and real estate industry, the construction and inspection industry as well as the film industry [59]. As of today, the FAA has issued over 5500 exemptions from the commercial UAV ban.

Aerial Photography, Survey and Real Estate Industry

While aerial photography has been possible before the widespread use of small size UAVs, it has always had a tremendous cost. Making even one aerial picture usually meant chartering a plane or helicopter and a professional pilot, as well as managing the flight date and flight route. For firms with a small budget, this method has not been viable at all, making the availability of aerial pictures of small communities very uncommon. This changed so drastically with the widespread availability of UAVs that the National Association of Realtors are calling it a similar technological revolution to the change from analog to digital photography [3]. According to them, aerial pictures will be mainly used in the future for promotional reasons in this industry, but also as a way to provide potential homeowners with videos and pictures of their potential property and to assess the entirety of the area surrounding. Affordable UAVs are also used for property appraisals, facility management, roof inspection, insurance evaluation and thermal imaging evaluations. Another reason given to the FAA exemption committee was the use of UAVs to assess the damage of properties after a flood or other catastrophic event, speeding up the process for property owners and companies to assess and fix the damage.

Table 2.1: List of the first 1000 exemptions the FAA made from the ban on the use of the flight of commercial UAVs. Data taken from [3]. Note that for some exemptions, two or more categories overlap.

Industry/Operation	Nr. of Exemptions
Aerial Photography	512
Real Estate	350
Aerial Survey	301
Aerial Inspection	242
Agriculture	164
Construction	134
Infrastructure Inspection	102
Film and TV	91
Utility Inspection	78
Environmental	61
Training	55
Search and Rescue	52
Research and Development	24
Emergency Management	38
Demos	26
Insurance	25
Mining	25
Oil and Gas	24
Advertising	22
Newsgathering	20
Landscaping	15
Flare Stack Inspection	13
Sports	13
Education	9
Market Research	8
Security	6
Railroad Inspection	4
Aerial Communications Services	1
Maritime Operations	1
Mill Operations	1
Paving	1
Risk Management	1

Construction and Inspection Industry

In a discussion paper held by twenty six thousand members of the Associated General Contractors (AGC) of America, a US wide trade association of construction companies and related firms, many merits of small size aerial vehicles in regard to the construction and inspection industries have been identified. The claim of the AGC of America is that by using UAVs, the cost and speed of the planning and design of construction projects can be improved since the planning team is able to assess the construction zone faster, cheaper and with higher frequency [3]. Furthermore, the safety aspect of using a UAV is highlighted, since there is no need to send workers to high altitude, potentially dangerous inspection tasks. Other merits mentioned in the report name higher efficiency due to the opportunity to analyze the workflow from an aerial point of view, better and more frequent quality control, since small size UAVs are able to reach positions that would be impossible to reach without building a scaffold first, and the ability to make it easier to follow environmental regulations and regular checks required by the U.S. Environmental Protection Agency. As an outlook, the report mentions that UAVs may be used to completely replace even long-established machines in the construction industry like cranes and to transport tools, equipment and construction material in the future.

Film Industry

The introduction of UAVs had a big influence on the film industry starting in 2013, the year the FAA decided to allow exemptions to the ban of small size commercial UAVs. Before then, the U.S. film industry, the biggest worldwide, was either forced to film sequences that required the use of UAVs outside of the U.S. (as has happened with the 2012 James Bond movie "Skyfall" that was filmed in large parts in Turkey due to the less restricting regulations regarding UAVs) or movies needed to fall back to other techniques to film action sequences like rigging cable cameras and camera cranes. Apart from the money saved on the rental of helicopters and pilots for aerial shots, one of the biggest merits given by industry insiders in using UAVs over helicopters

that were routinely used in the film industry until 2013 in the U.S., is the higher degree of safety when using UAVs. Reasons given for this are the elimination of the need to store flammable liquids close to the film set (gasoline for the helicopter) and the high number of fatal accidents in the film and TV industry before 2013 that involved utilizing helicopters. Since the safety regulations in so-called "closed-sets" in the U.S. film industry were already higher than the safety regulations required by the FAA, the U.S. film industry was one of the first industries to receive exemptions from the FAA commercial UAV flight ban [3].

Agricultural Industry

In the 2010s, the agricultural industry saw a sharp rise in the utilization of digital data to manage crop production and for livestock monitoring. The use of UAVs for live aerial imaging of the crop fields led to higher efficiency in planting methods, as well as better adjustment to seasonal changes in weather patterns and the environment. Since, much like in the film industry, traditionally aerial photography was done using planes or helicopters, the agricultural industry has improved the safety of people on the ground, livestock and pilots by using UAVs for aerial imaging tasks. First deployments of small sized aerial vehicles in the agricultural industry involve e.g. the Yamaha RMAX for crop spraying in Napa Valley [60]. The main advantage of these deployments is the speed that crops can be sprayed when compared to more traditional methods, like spray-tractors, which were less than 1/4 as fast in spaying crops than the UAV. The improved safety and efficiency of spray material has also been well highlighted in comparison to spraying the crops by plane [3].

Search and Rescue Industry

The history of remote controlled vehicles (or robotics in general) in search and rescue (SAR) is a relatively young one. According to one of the researchers most deeply involved in the field of disaster robotics, Prof. Dr. Robin Murphy, the first documented large scale real life deployment of remote controlled vehicles in a search and rescue environment was the search efforts at ground zero after the terrorist attacks on the

World Trade Center in September 2001 [1]. The overall conclusion of this deployment was rather negative. Casper and Murthy describe in [61] many of the challenges, the responders on ground zero faced during their deployment of search and rescue robots. All of the robots in this specific deployment were ground based robots. Due to the high mechanical complexity of the robots, joints and wheels broke or became unusable due to dust that got into the enclosure of robots, and were hard to repair on-site. The size of the robots also played a large role. The most frequently deployed robots were the small size robots that where able to be easily transported to the disaster site. Therefore, some of the large size ground based robots did not see any deployment at all (Fig. 2.7). Also, because of the instability of the rubble surrounding the destroyed World Trade Center, light weight robots posed less of a threat to the SAR personal. Lack of training of the SAR personal in remote controlling a SAR robot resulted in confusion, frustration and ultimately less use of robotics than anticipated.

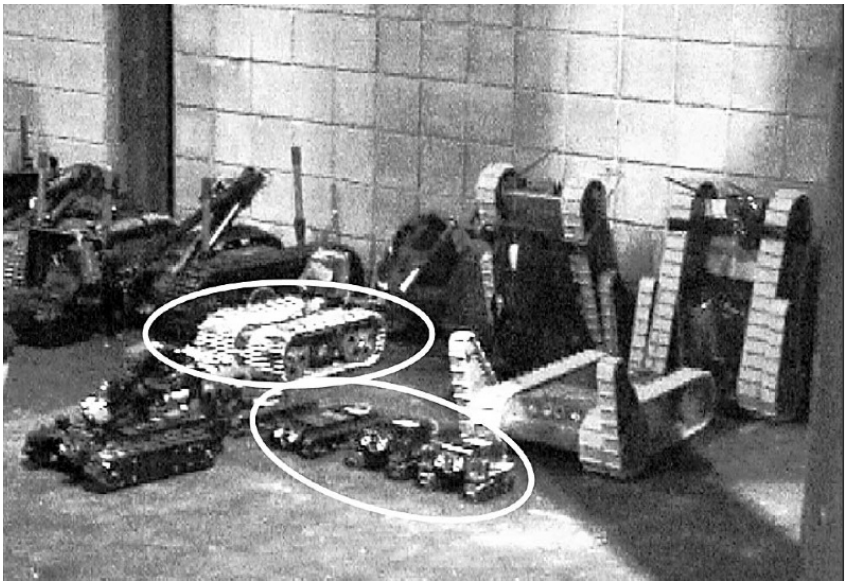

Figure 2.7: Photo of the robots that were considered for deployment in the search and rescue efforts after the 9/11 terrorist attacks on the world trade center. Only the white circled ones were actually deployed. (picture taken from [1]) © 2004 IEEE

Even 10 years after the 9/11 disaster relief efforts, the use and development of

search and rescue robotics was very limited. This unfortunately became very apparent in 2011 after the Tohoku earthquake and the resulting Tsunami, which lead to the destruction of the Fukushima Daiichi power plant and the following reconnaissance efforts. No machines or robots were immediately available in Japan for a deployment in such a disaster area, so the first robots sent to assess the damage of the power plant were supplied by the U.S. military (Fig. 2.8) [2]. Since the robots deployed were intended to be used in a military battlefield and were ill-equipped for the half-destroyed, industrial environment that could be found at the nuclear power plant in Fukushima, many deployments resulted in a loss of the devices and only limited intelligence of the area [5]. Numerous reconnaissance operations of the destroyed nuclear power plant in Fukushima Daiichi (e.g. [62][63]) have been conducted in the weeks and months after the earthquake. The most immediate responses documented were the deployment of the KOHGA3, a tracked ground vehicle, by the International Rescue System Institute (IRS) to inspect a partially collapsed building at Hachinohe Institute [5]. Other operations (e.g. Huang *et al.* [64]), while very resourceful, can be considered recovery, and not search and rescue operations as they were started at least 1 week after the occurrence of the earthquake.

Figure 2.8: Photo of iRobot Packbot (left) and Warrior (right) during training exercises for a deployment in the destroyed Fukushima Daiichi power plant. Although specifically developed for use in a military environment, they were the first robots deployed in the Fukushima Daiichi reconnaissance efforts. (images taken from [2])

Today, disaster robotics enjoys a greater focus in the research community than in 2011, which is visible in the rise of robotic challenges focused on search and rescue, such as the DARPA Robotics Challenge in 2013 and 2015. With the rising popularity of small sized aerial vehicles such as multirotors, we can also see a rise in UAV deployment after disasters such as floods (like the flood in Yuyao, China in 2013 [65]). The use of multirotors solved several problems that have been unsolved since the introduction of robotics to the field of search and rescue. Since the only moving parts on a multirotor are the brushless motors, the mechanical complexity of the system is much lower that of any ground based robot. This makes repair and exchange of parts in the field very manageable. Additionally, due to their naturally small size, aerial SAR robots are easy to transport to the deployment zone. Lastly, since multirotors are moving while airborne, there is no risk involved for any personnel on the ground. They are also mobile regardless of what kind of surrounding they are deployed into.

According to Murphy one of the biggest unsolved problems aerial SAR robots face is the need for a human controller. Since the control of a multirotor requires the control of the vehicle within 6 degrees of freedom simultaneously, the deployment of these machines requires a trained operator to avoid collision with the surroundings. Murphy also mentions that one third of documented uses of aerial robots in a disaster environment until 2014 resulted in failure due to human error [5]. With the further development of collision avoidance systems and autonomously flying multirotors in research, and even in the commercial area [66], this problem can be solved in the near future, making the aerial robot a reasonable tool for future search and rescue efforts.

2.4 Flight Controller

The flight controller is the main hardware of a UAV platform and responsible for sensor processing and flight operations. While some research projects are focused on designing their own flight controllers [67] [68], many research projects in the field of aerial robotics make use of readily available flight controllers. In recent years, the number of available flight controllers has grown dramatically, making the decision of

which one to choose more difficult.

In this chapter four flight controllers, the Arudupilot Mega, the Pixhawk, the DJI Naza-M-V2 and the DJI A2 (Fig. 2.9), are briefly introduced and put into context. All flight controllers have been used in several stages throughout this work in diverse flight experiments (see Table 2.2).

Table 2.2: Overview of the flight controller used in this work.

	Ardupilot	Pixhawk	DJI Naza-M-V2	DJI A2
Open Source	x	x		
Architecture	Atmel	ARM	ARM	ARM
Clock Rate (MHz)	16	168	?	?
RAM (KB)	4	256	?	?
Integrated IMU	x	x	x	x
Ports	UART,I2C	PWM, UART, SPI, I2C, CAN	CAN (proprietary)	CAN (proprietary)
Chapters this flight controller was used	4.1.1	5	3.2 4.1.2 4.2	4.1.1

Ardupilot Mega

The Ardupilot Mega (APM) is the successor of a previous, abandoned project: the Ardupilot flight controller (AP). As the name suggests, the flight controller is based on the Arduino Mega platform. It is entirely compatible with the boot loader of the standard Arduino platform and even uses the identical Atmel ATMEGA2560 amd ATMEGA32U-2 chips. The only difference between it and a normal Arduino Mega is the added 3-axis gyro, 3-axis accelerometer and a high resolution altimeter that has been added on board in a compact form factor. It was designed by an open source community as a controller for multirotors, fixed wing aircrafts, helicopters, ground rovers and also antenna trackers. The project was distributed under the GPLv3 open source license.

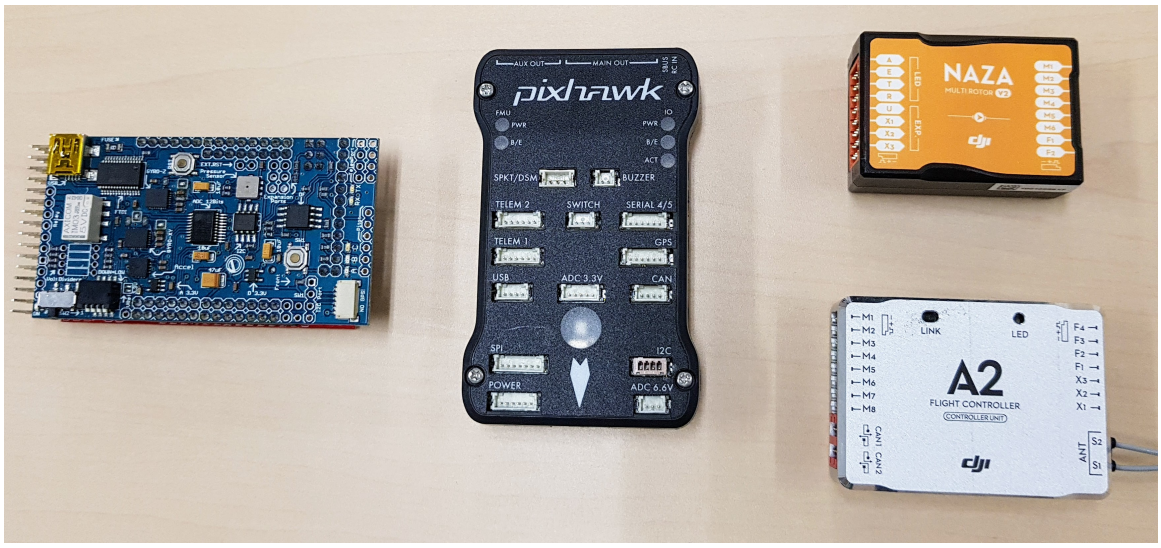

Figure 2.9: (left) The Ardupilot Mega flight controller (center) The Pixhawk flight controller (right) The DJI Naza-M-V2 and the A2 flight controller

The APM was one of the first all-in-one open source flight controller solutions that was widely available. Because of the popular open source community gathering around the Ardupilot Group website [69], the APM was able to receive nearly weekly updates. Due to the constant maintenance of its source code, it was generally known as being a reliable flight controller. But one of the major issues it faced was its reliance on the Arduino platform and its Atmel chips. Around 2014, the capability of the only 16MHz Atmega 2560, 4KB RAM and its 16MB of dataflash memory was simply not enough to implement more advanced algorithms beyond simple altitude control. The low clock speed also proved to be a problem when implementing algorithms for interfacing with high speed sensors or sensor noise reducing algorithms. It also had a severe problem connecting to external hardware since the only interfaces available were either UART or I2C.

While it still had enough features to be used as a flight controller for leisure, the project was quickly abandoned by many researchers in favor of other open source hardware solutions, like the PixHawk project.

Note that while the Ardupilot hardware has basically been abandoned, the Ardupi-

lot Group source code is still being actively developed and is popular in many research environments. This is due to the fact that the source code has been made compatible with more capable flight controllers, such as the PixHawk.

PixHawk

The PixHawk flight controller emerged from the researcher group around Lorenz Meier *et al.* of the ETH Zurich and their node-based open source robotics framework, called PX4 [70]. Development on the PX4 framework started in 2009, but it wasn't until 2012 that its popularity sharply rose with the development of the Pixhawk flight controller. This flight controller was a joint effort from the Computer Vision and Geometry Lab of ETH Zurich and the Autonomous Systems Lab, and used the PX4 framework as base component. It started being distributed in 2012 as the most advanced open source flight controller of the time, by 3D Robotics, a company based in the US. In contrast to the APM, the Pixhawk is focused on the control of multirotor platforms and thus has many capabilities the APM does not. The project has been made available under the Berkeley Software Distribution (BSD) open source license.

The PixHawk hardware has become one of the most frequently used open source controllers in academia. As with the APM, it has an integrated inertial measurement unit (IMU), consisting of a 3-axis gyro, 3-axis accelerometer and a high resolution altimeter, on board in a compact form factor. Due to its use of a STM32F427 with an ARM based Cortex-M4F core of 168MHz and 256KB of RAM it is significantly more capable than the APM. That means more complex algorithms for noise filtering and sensors with higher clockrate can be used. This results in much better flight characteristics and response times in comparison to the APM. It also provides more capabilities for communication with external modules, since it has PWM, UART, SPI, I2C and CAN communication ports on board and integrated into its default firmware.

Although, as mentioned, the Pixhawk is the reference hardware in most open source flight controllers, the project has seen some problems recently. The main supplier, 3D Robotics, parted ways with the original developer group and the Arudupi-

lot Group due to disagreements over the license of the open source code which 3D Robotics products are based upon [71]. Although still an open source project, 3D Robotics abandoning the production of this flight controller makes it hard to find a supplier for the hardware and has left the open source community unsure about the future of open source UAV controllers.

DJI Flight Controller

Da-Jiang Innovations Science and Technology Co., or in short DJI is a Chinese company located in Shenzhen, Guangdong. It develops and sells small size UAVs, flight controllers and camera gimbal related products. The company was founded in 2009 and is the biggest company in the civilian aerial robotics industry, making up to 70% of the global consumer UAV market [72].

Since the DJI flight controller has been built for the mass market, not much is known about the detailed specifications. Teardowns show that some DJI flight controllers, e.g. the DJI Naza-M use a LPC176x Series ARM Cortex-M3 MCU with 64KB RAM with a maximum of 100 to 120MHz clock rate, in the case of the LPC1769 [73]. This would place the capability of this particular flight controller in between the APM and the Pixhawk. In contrast to the open source solutions mentioned above, it is entirely unknown what algorithms are used to accomplish this stability.

About the flight controllers used in this work, the Naza-M-V2 and the A2 flight controllers, no information regarding clock speed or memory is known. Despite this lack of information, the flight controllers were chosen in several aerial platforms throughout this work to simulate a standard off-the-shelf flight controller of unknown properties and because of their outstanding flight performance and stability.

In 2015, DJI started its initiative to become more relevant in academics with the introduction of the DJI Matrice 100, which was the first DJI UAV with support for their API and also the first with a UART port interface (instead of the proprietary CAN-bus all other DJI flight controller use). Unfortunately, the support and capabilities of the API are far behind open source solutions, making DJI hardware often not the first choice in an academic environment. Furthermore, some controversies, like

the forced firmware updates or the internal guidance paper by the US Army calling for a ban of the use of DJI UAV because of security problems found, raise concerns about the reliability of these commercial flight controllers in an academic setting [74].

Chapter 3

Early Concepts

As discussed in chapter 2.1.2, one of the big differences between ground based and aerial robots is the lack of a stable "neutral state" for aerial robots. The goal of this work is to find a way to make the neutral state of a multirotor less prone to disturbance and thus make the multirotor platform more feasible in applications where a high degree of precision in control is necessary. This chapter gives an overview of concepts and designs that have been tested, but eventually discarded in the course of the development process. The reason for their discontinuation is given with the hope that it will add to the overall understanding of the final design of the aerial platform developed in this work.

3.1 Adding More Actuators

As described in chapter 2.2.3, a multirotor design is very prone to horizontal disturbances. One of the main reasons we are not able to counter horizontal forces in the neutral state is the fact that the multirotor is an under actuated system (see formula 2.3), and thus requires active exiting of the neutral state to counter these horizontal forces. The most trivial solution would be to add more actuators (Fig. 3.1) until

$$\operatorname{rank}\big[f_2(\vec{q}, \dot{\vec{q}}, t)\big] = \dim\big[\vec{q}\big]. \tag{3.1}$$

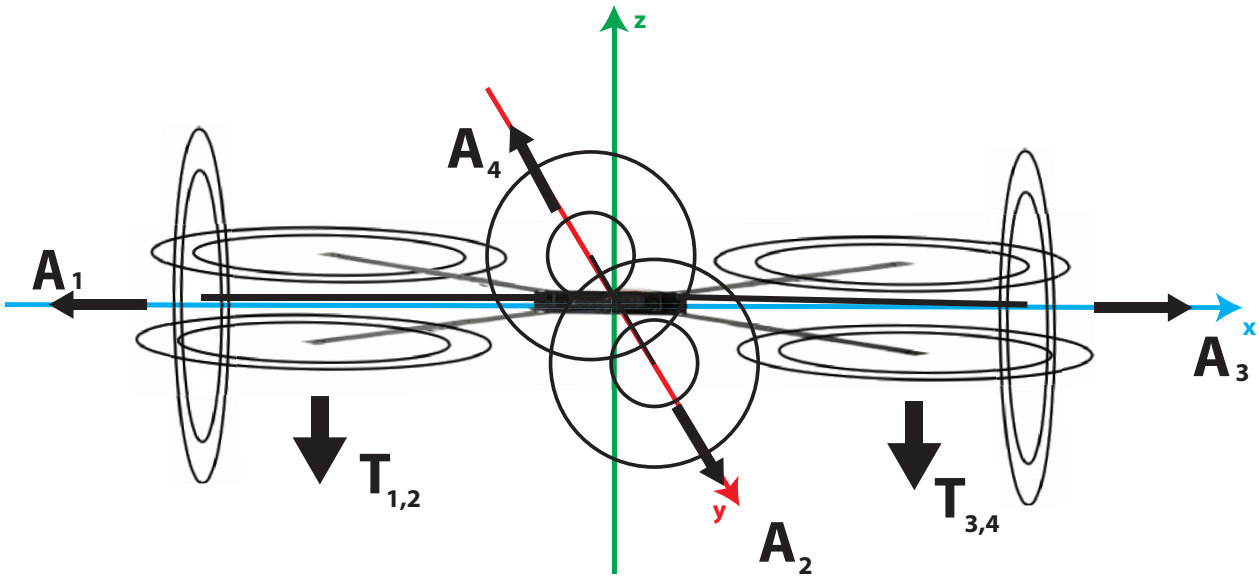

Figure 3.1: Sketch diagram of a quadrotor concept with additional actuators ($A_{1,2,3,4}$) that are added to the usual actuators of a quadrotor that provide thrust $T_{1,2,3,4}$. This theoretically enables the control along the horizontal x,y axis without tipping and thus without disturbing the neutral state of the device.

While this would be a valid solution to the problem, a few problems became apparent very quickly in the planning phase of this concept. First of all is the problem of practicality. Additional motors (preferably mounted with 2 pusher and 2 puller propellers) would add not only additional weight, but also would add additional strain on the overall power consumption. Another concern would be the placement of these motors. To prevent tipping of the multirotor, these motors would have to be placed along the horizontal axis of the center of gravity of the platform. Since on a conventional multirotor, the main propulsion motors are already mounted on this axis, this would lead to the blades touching each other. While this may be countered with some sort of synchronized propulsion method, it would require a much more mechanically complex mechanism than desired. Apart from the issue of practicality, there is also the problem of basic inertial control. When spinning one of the motors that are mounted orthogonal to the horizontal axis, the motors apply torque τ_m over

time:

$$\int \tau_m dt = I\vec{\Omega}, \tag{3.2}$$

equal to the change in angular momentum that would be necessary to be countered by the main propulsion, which is already controlling the attitude. This would require knowledge of the exact speed of the motors to estimate their produced torque and hence more complex motors with encoders. Apart from this, the compensational control of this counter-motion needs to happen almost instantaneously, or further disturbances are introduced. This would result in an infinite feedback loop. At this point it is clear why the whole concept was considered unfeasible and the idea was shelved. To our knowledge there has been no other research group that has tried tackling the complex problems which arise when pursuing the concept of redundant actuation when applied to a multirotor base.

3.2 Rope Based Refined Control

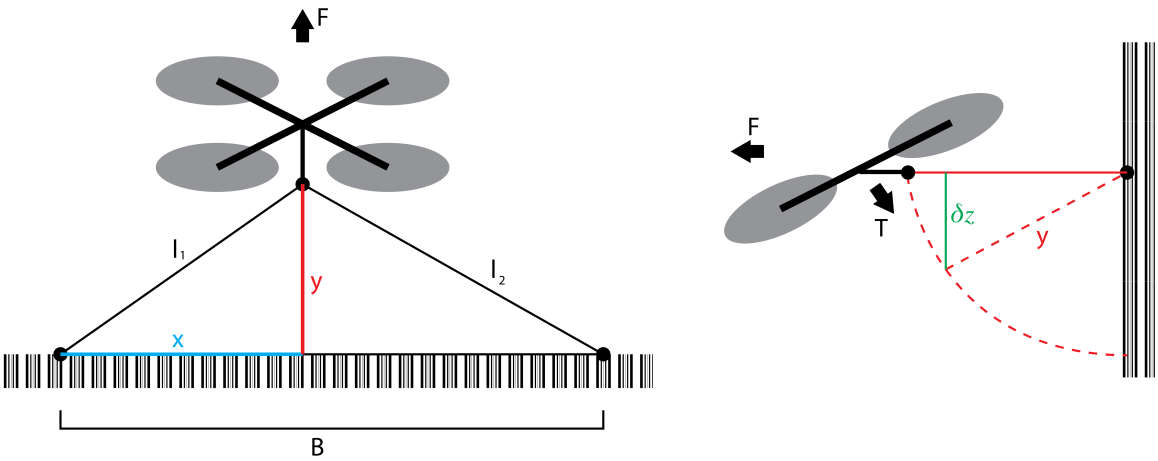

Figure 3.2: Sketch diagram from the top (left) and side (right) of a quadrotor concept with ropes connected to a wall for precise manoeuvring.

Another concept that we pursued initially is using multiple ropes or strings fixed to a wall and a pulley system mounted on the multirotor to archive a more stable

neutral hover state. By adjusting the length $l_{1,2}$ of the rope that has been mounted to the wall at a certain, known baseline B, we can create tension by steering the multirotor away from the wall with a forward force F_f large enough to achieve very precise changes in horizontal position x, y relative to the wall (see Fig. 3.2). The calculation of the required rope length l_1 and l_2 was done using basic Pythagorean theorems:

$$l_1 = \sqrt{x^2 + y^2} \tag{3.3}$$

$$l_2 = \sqrt{(B - x)^2 + y^2}. \tag{3.4}$$

While we are now able to precisely control the position of the multirotor along one plane, the multirotor is still able to change the altitude (δz) by regulating the thrust T which in turn will influence the distance d to the wall with:

$$d = \sqrt{y^2 - \delta z^2}. \tag{3.5}$$

The assumed application scenario was an inspection task of a wall, where a previous setup (the fixation of the rope to the wall) was possible. After a successful initial experiment to test the feasibility of this concept with fixed length ropes (Fig. 3.3), an early prototype of a mechanism was built, consisting of a custom MDF 3D printed base with two pulleys being driven by two small 9V stepper motors, as well as an Arduino micro controller and motor drivers (Fig. 3.4).

Figure 3.3: Photo of a test flight of the rope-based refined control prototype. In this test flight, the ropes were non-retractable. The position of the rope was marked with dots by hand to make them better visible.

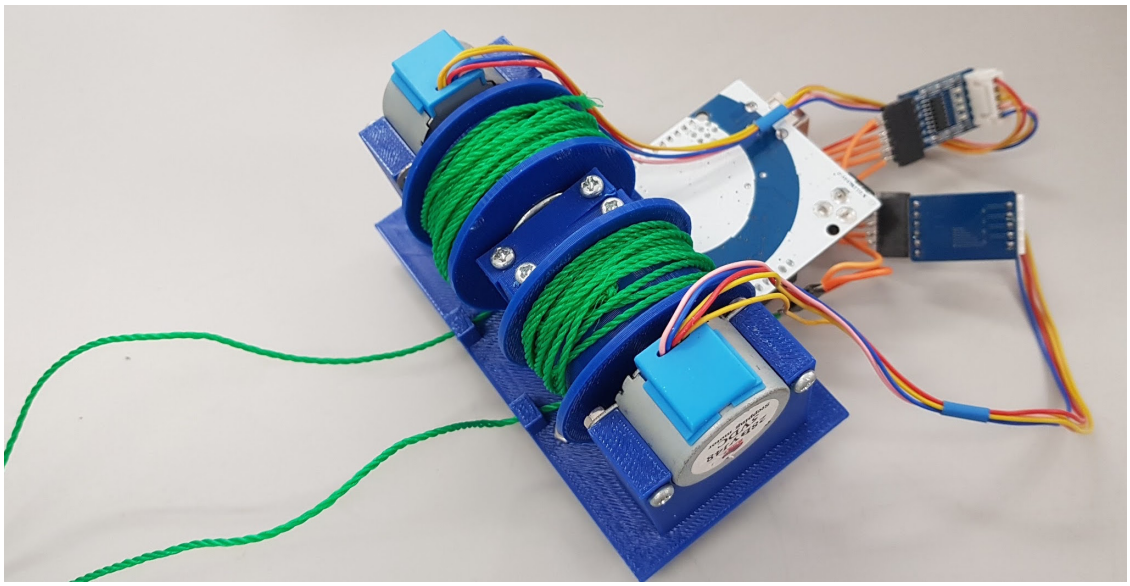

Figure 3.4: Photo of the pulley mechanism prototype for retracting the ropes for enabling a varying length $l_{1,2}$

This control scheme worked in early experiments. But there were two major problems that were discovered during the testing. First, there was the problem of required torque from the pulley system. Since the multirotor used for this experiment, a quadrotor, is able to generate around 9N thrust per blade, the required torque τ_P per 5cm radius pulley needed to be at least (when considering the worst case scenario in which the rope is parallel to the thrust and without considering eventual impact forces) $\tau_P = 2 \cdot 9\text{N} \cdot 0.05\text{m} = 0.9\text{Nm}$ just to keep the multirotor platform from moving. This was too much for the small size motors on the prototype, so a test in-flight with the pulley prototype could not be conducted. When planning a bigger pulley system, another problem became clear; when applying force away from the wall, the multirotor needs to angle itself perpendicular to the wall to generate force 3.2. Ideally, the most force to keep the rope under tension is generated when the multirotor reaches an angle of 45 degrees towards the ground, since the thrust is then equally divided between generating enough downward thrust to keep the current altitude (mg) while simultaneously generating F_f. When calculating the downward

force F_d remaining from the thrust $T_{\sum}$ that is acting downwards when the quadrotor is tilted by 45 degree, we get

$$F_d = \cos(\frac{\pi}{4})T_{\sum} = \frac{1}{\sqrt{2}}T_{\sum}. \tag{3.6}$$

This means that only around 71% thrust can be used in this state to prevent the quadrotor from dropping to the ground and keep a stable position. Considering that a larger system of pulleys and stepper motors would have been required to test this concept in full, the additional payload for devices to actually observe the wall for an eventual inspection task would have been significantly decreased. After further consideration whether or not to further pursue this concept, the choice was made to explore other options mainly because of the lack of payload efficiency and lack of ability to reliably control the altitude of the multirotor system. To my knowledge, a multirotor system with a previously mounted rope system has not been pursued by other research teams at the time of writing.

Chapter 4

Embedded vision system for localization

4.1 FPGA Based Vision System

While many of the methods shown in this FPGA based vision system chapter can be considered trivial when implementing them in a traditional imperative programming language like C, the complexity of FPGA architecture makes even simple algorithms challenging to implement properly. Nonetheless due to the advantages of an FPGA implementation such as very low response time, highly parallel processing and a small system footprint, it is still valuable to consider such an implementation.

The algorithms presented in this chapter have mostly been prototyped in Matlab to test their function, then adapted and implemented using VHDL to create a task specific logic circuit that is able to solve a certain computer vision task and handle any necessary outputs. An interface like Simulink or methods to convert C or Matlab code to VHDL code was not used, mainly because the resulting VHDL code would often be unreadable to a human and hence would make debugging harder or impossible. An additional advantage is that unnecessary use of the already limited resources, e.g. the number of available gates and memory of the FPGA has been avoided.

4.1.1 Target Tracking with Active Camera

Concept

The goal in this part of the project was to create a computer vision algorithm that is able to solve a hold position and altitude task for a flying mobile platform, simple enough to be implemented into an FPGA while still maintaining sufficient robustness.

To simplify the design of this algorithm, it was assumed that the object that is targeted for tracking is round and a certain distinguishable color (in this case red). We also assume the target's size is known beforehand. We do not have to specifically calibrate the camera, but nonetheless need certain information about the lens used (focal length) and the camera sensor (pixel size on image sensor) to later get a sufficient approximation of the distance between camera and target.

First, we need to extract the target object from the received image. In conservative programming language this would be a simple blob extraction. Traditional blob extraction is mostly a two-pass algorithm. This is where we reach an important limitation when using an FPGA. Since the memory is very limited we can not store a lot of information, hence making two pass algorithms, or algorithms relying on storing numerous information in general, not feasible to be implemented in this kind of architecture.

There are algorithms dealing with this limitation and also algorithms specifically designed for FPGA blob detection, like the method by Trein, Schwarzbacher and Hoppe [75]. But since for this task the properties are highly defined (round target, certain color), a one-pass algorithm for round targets is computationally cheaper and uses only basic addition and division.

Due to the low complexity of the algorithm, it can not be used to counter an eventual image distortion due to the tilting of an aerial platform, which occurs in all multirotors, due to the need to roll and pitch when flying in the horizontal plane. To counter this, the camera for receiving the marker image is mounted on a 2-axis active camera mount. The tilt and roll measurements are taken from the internal IMU of the flight controller and are used to keep the camera mount perpendicular to

the marker on the ground.

Tracking Algorithm

Since the target is round, if we create a histogram of only the target, we see a histogram in a certain shape, gradually increasing in size in the horizontal and vertical direction each. Since we are interested in the center of the target, we can assume that the most important area of our target has the largest dimensions in both the horizontal and vertical directions, seeing a gradual growth beforehand.

To take advantage of this observation, we must first extract the target by comparing the Euclidean distance of each pixel's color in RGB color space with the desired color to a certain threshold:

$$dist_{RGB} = \left| \begin{bmatrix} des_r \\ des_g \\ des_b \end{bmatrix} - \begin{bmatrix} pixel_r \\ pixel_g \\ pixel_b \end{bmatrix} \right| . \tag{4.1}$$

When creating this binary image, we add each valid pixel to a histogram and give it a certain weight. This weight is dependent on the pixels that have already been added to the corresponding row or column in each histogram. The formula for calculating the estimated target center position in the image plane $(\overline{u}, \overline{v})$ is:

$$\begin{aligned} \overline{u} &= \frac{\sum_{i=1}^{x_{hits}} w_i x_i}{\sum_{i=1}^{x_{hits}} w_i} \\ \overline{v} &= \frac{\sum_{i=1}^{y_{hits}} w_i y_i}{\sum_{i=1}^{y_{hits}} w_i}. \end{aligned} \tag{4.2}$$

This method should remain stable against noise and even other non-round objects in the image, but we can assume it will give false positives if there are other large, circular objects in the picture.

The algorithm works well for round targets since it weighs lines and rows with a continuous target pixel, like the middle of the target, very high while weighting lines and rows in the image with less continuous target pixels, like the edge of the round target significantly lower.

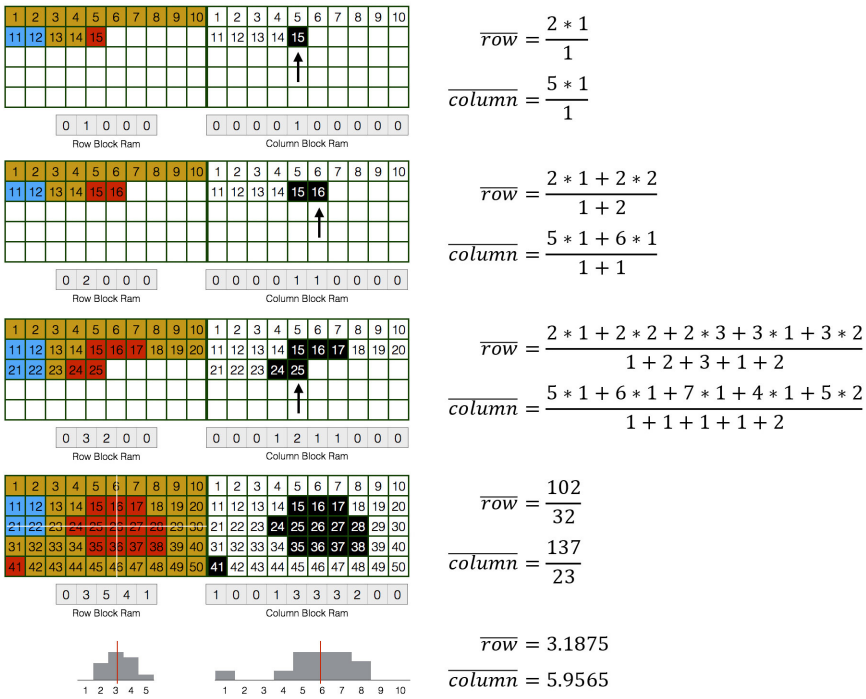

Figure 4.1: A visualization of the creation of a weighted average mean via FPGA.

Overall, while not being overly complex, the algorithm gives satisfying results for the task at hand without being prone to noise and other objects in the picture, while still remaining a one-pass approach. Furthermore the use of the FPGA's memory was limited to $width*$(bit needed for binary($height$))$+height*$(bit needed for binary($width$)) bit. In case of the 320×240px picture that is processed in this example, we used 4720bit (590byte) of memory to extract the center position of the target. The memory can even be reused to visualize the histogram of the target detection for visualization or debugging purposes. A representation of the detection process can be found in Fig. 4.1 and the pseudo-code to help understand this approach in Algorithm 1.

After determining the estimated center of the round target, the size of the target in the image plane can be extracted by simply making note of the histogram's build-up around the estimated target center. To smooth the error that may result when estimating the target size, we make use of the fact that the target is round and calculate the target diameter D with the horizontal histogram H_h and the vertical

```
Data: Binary image of target
Result: Target center
initialization;
for x to frame_width do
    for y to frame_height do
        read current pixel from memory;
        if color within range then
            u-pos-numerator = u-pos-numerator + u-histogram[x] * x;
            v-pos-numerator = v-pos-numerator + v-histogram[y] * y;
            u-pos-denominator = u-pos-denominator + u-histogram[x];
            v-pos-denominator = v-pos-denominator + v-histogram[x];
            u-histogram[x] +1;
            v-histogram[y] +1;
        end
    end
end
/* end of frame                                                    */
estimated u position = u-pos-numerator/ u-denominator;
estimated v position = v-pos-numerator/ v-denominator;
```

Algorithm 1: How to calculate estimated center of target

histogram H_v:

$$D = \frac{H_h(\overline{u}) + H_v(\overline{v})}{2}. \tag{4.3}$$

A picture of the FPGA output after this process can be seen in Fig. 4.2 Problems may occur if the tracked object's round shape degenerates in the image plane. Although the camera mount prevents any degeneration of the shape due to angular distortion, if the object is leaving the image plane, the center of the target cannot be tracked reliably.

Edge proximity detection is used to prevent losing the tracked target. When the target is tracked and is moving towards the edge of the image plane, it is assumed

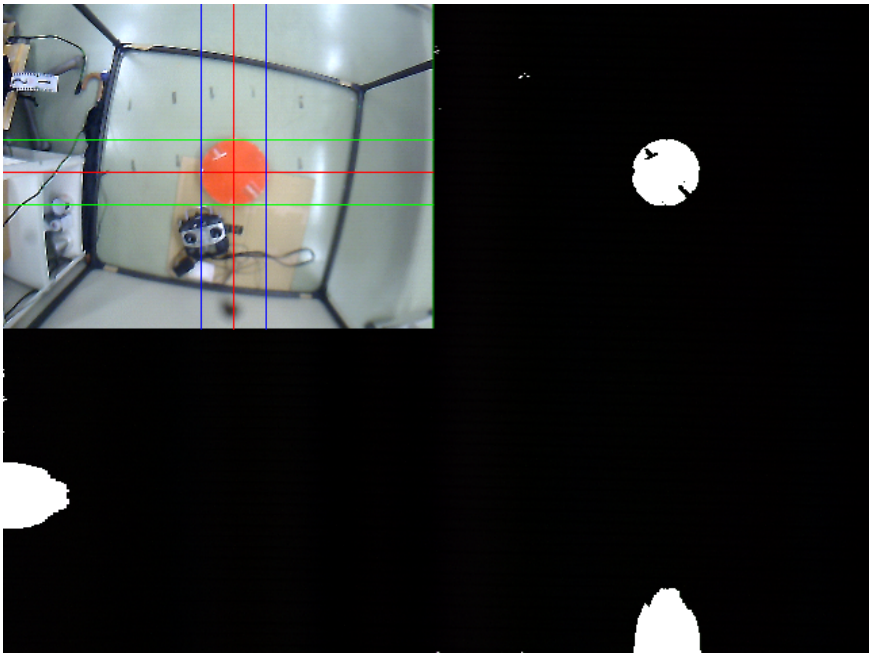

Figure 4.2: The FPGA debug output. Upper left: Camera perspective with target visualizations (target estimated position: red, target estimated width: blue, target estimated height: green). Upper right: binary image. Lower part: Extracted vertical and horizontal histogram.

that in the next frames received from the camera, the object in the image plane will likely be degenerated and the center will not be estimated correctly anymore. To counter this, the last known correct image size and position is recalled and depending on where the object left the image plane, the size and the coordinate axis where the object left the image plane is frozen.

Since the algorithm is used on a mobile flying platform and its task is to center the tracking object in the middle of the image plane, we can assume that the camera taking the picture will move very quickly back into a position where the target is not degenerated anymore. During that time, the size in the image plane (the distance of the camera from the target) should not change significantly, making this strategy feasible to use. Pictures showing partial vacating of the image plane are shown in Fig. 4.3.

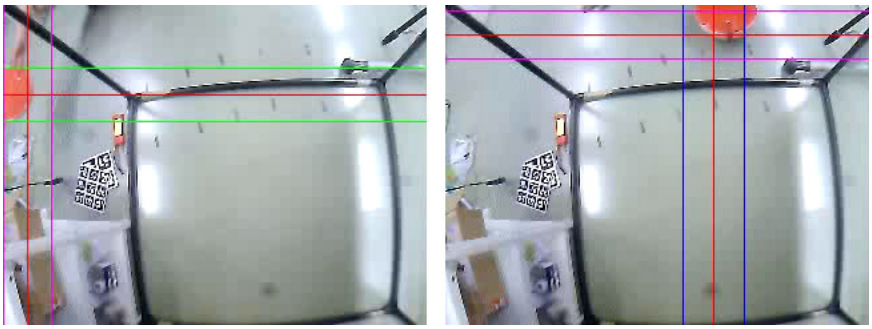

Figure 4.3: Output of the FPGA showing the partial image plane vacating of the target and the visualization of the estimated target size and position. Green and blue show the outer edge of the target; they become pink if the target vacates the image plane. Red shows the last position before vacating.

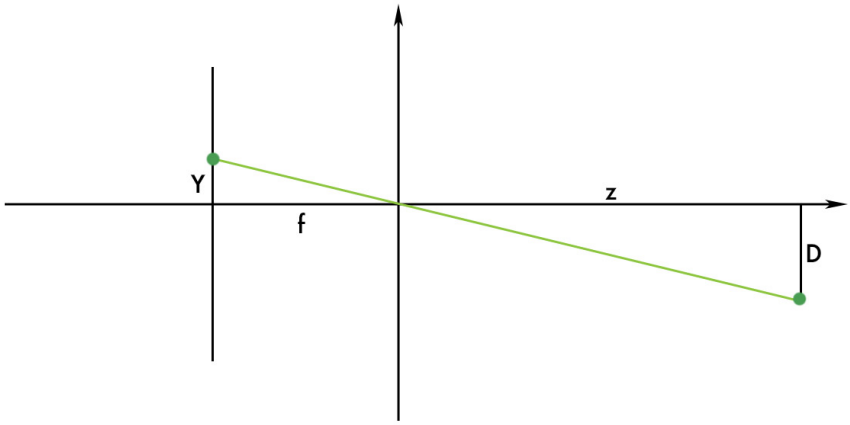

Figure 4.4: Pinhole camera principle.

If the algorithm is not able to find enough target pixels in the frame or the target found does not reach a certain expansion threshold in the horizontal and vertical directions, the algorithm returns a "target not found" signal to avoid tracking a wrong target and endangering the task at hand.

Height Estimation

After the position of the center of the target on the image plane has been extracted, the distance of the target needs to be approximated to not only hold the position, but also to adjust the altitude of the flying mobile vehicle over the tracking target. To do that, we can either use the thin lens approximation

$$\frac{1}{f} = \frac{1}{s} + \frac{1}{s'}, \tag{4.4}$$

as was done in previous papers of this project [76] (s and s' refer to the image distance and object distance, f to the focal length) or, since $f << D$, where D is the object diameter, we can further simplify the problem and formula to a pinhole camera model ,(Fig. 4.4) we can see two similar triangles which share the same line as their hypotenuses. We can assume the relationship:

$$\frac{Y}{f} = \frac{D}{z}. \tag{4.5}$$

In our approach, Y corresponds to the size of the projected target on the image sensor, D to the actual diameter of the target and z to the distance of the camera to the target. Since we extracted the size of the target on the image plane in pixels in the earlier stages, we can calculate

$$Y = d * p, \tag{4.6}$$

where d is the target size on the image plane in pixels and p the size of one pixel on the image sensor. Given this knowledge, we can assume:

$$z = \frac{f * D}{d * p} \tag{4.7}$$

and from this we can determine the approximate distance between the camera and the tracking target.

Conclusion

While this method has been implemented and tested successfully, some limitations apply for this kind of approach. The biggest limitation for this method is the fact

that the target's color and size must be known beforehand. If the color is not known beforehand, this algorithm will not be able to stabilize the horizontal position nor the flight height. Similarly, if the size of the target is not known, this algorithm is not able to calculate the distance to the target using the pinhole camera model. Both of these problems are addressed in a marker-based approach in chapter 4.2. Nonetheless, an FPGA based visual navigation system showed potential as a fast response on-board logic circuit flight controller add-on. Since the marker in this case was unfit to control the yaw orientation of the UAV (yaw orientation in this approach was exclusively done with the IMU compass), we went on, in a consecutive work presented in the next chapter, to solve this orientation control with a single FPGA flight controller add-on module. This FPGA module receives control over the 4DOF that are controlling the position and heading of the UAV: x,y,z axis and yaw.

4.1.2 Target Tracking with Orientation Measurement

Concept

In this chapter, we will to look at another way to extract the position and orientation of a UAV toward an object using an on board logic circuit visual navigation system. In the last chapter we considered a simple, round, colored target and tracking marker. In this chapter, we focus on a more practical approach and detect bar-like objects and make extended use of the parallel processing capabilities of an FPGA by using a type of parallel Hough Transform specialized for FPGA implementation.

The goal of this algorithm is to extract the distance, position and angle of the quadrotor in relation to a bar like object. This information can then be used to fly a UAV close to a bar-like object like a light pole, bridge steel beam or other similar object and would then, given that the UAV is equipped with a manipulator, give the UAV the ability to grab this object. Afterwards the UAV would be able to shut down its propeller to conserve energy, while maintaining a high altitude position. Therefore the conceptualized flight assistance system is required to detect and analyze a bar-like object in an unstructured environment.

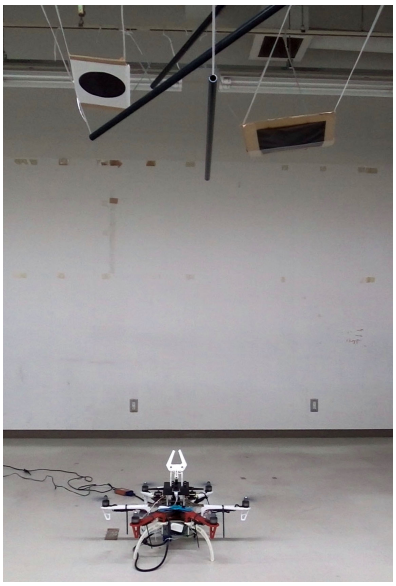

Figure 4.5: Test scenario of the autonomous yaw adjustment system. Several bar-like objects as well as multiple shapes printed on cardboard are hung at different heights from the ceiling.

In situations like Fig. 4.5, a bar-like target is indistinguishable from other objects in the image plane. To identify it as a desired target, certain criteria must be met. First, the distance from the neutral axis of the object and the center of the camera must be within a certain range. This is indicated by a black circle which defines the valid region in Fig. 4.6. Second, the object length needs to be above a certain threshold and must not have large gaps when compared with the longest object within the black circle. Third, the object diameter, projected flat onto the camera image plane, has to be within a certain pre-determined width. If there is more than one object detected, the system will select the closest object, chosen by finding the largest diameter of the detected objects, then finding the shortest distance from the neutral axis of all detected objects to the center of the camera.

The overall process of image processing is shown in Fig. 4.7. Initially, the Y channel of the recorded image is thresholded, assuming that the target's luminance is lower than that of the background. The whole thresholded image is then transformed to Hough space in order to distinguish between bars and paired lines. This approach

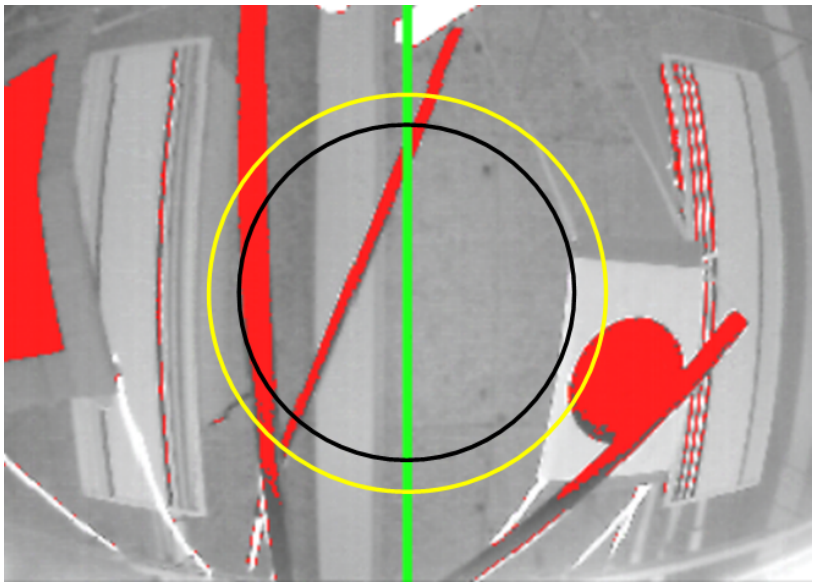

Figure 4.6: Debug output image of the FPGA. The original picture recorded from a monocular camera on the top of the drone in gray-scale. The red color indicates the result of binalization detecting the object. The green line indicates the angle of the nearest bar-shaped object after processing. The black circle defines our valid region, in which targets are considered. The yellow circle shows where we split the image for inner and outer Hough transforms.

is practical in parallel Hough transform (PHT) since the number of valid object pixels make no difference in term of processing time. However, when applying this method, Hough transform (HT) can interpret non-bar-like objects or discontinuous bars as valid objects. We propose to overcome this problem by introducing a novel method, performing Hough transform in an inner and an outer area of the thresholded image. To our knowledge, this is the first recorded example of HT being used in this way. This inner area is located inside a certain region, shown as the yellow circle in Fig. 4.6 and rest of the image is considered as the outer area. The diameter of this region equals the diameter of the region where objects are considered for detection (shown as the black circle in Fig. 4.6) plus a carefully defined, predetermined maximum acceptable bar-like object diameter.

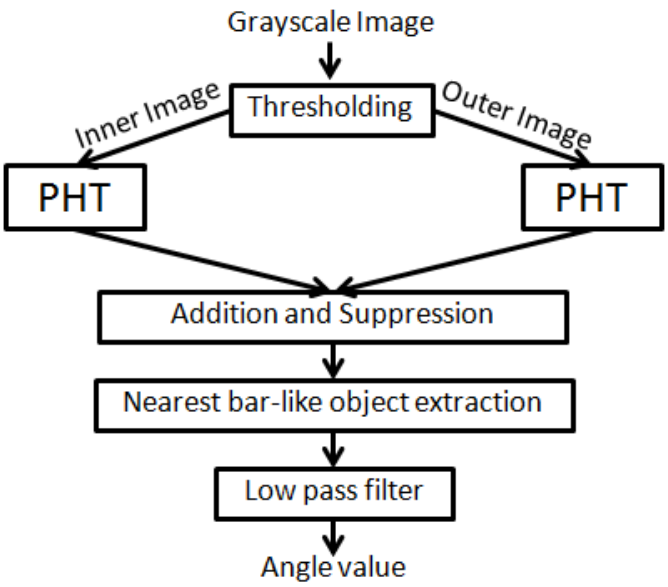

Figure 4.7: Flowchart of the computational processing, from received image to automatic yaw control.

Measurement of the Angle of the Object

For detecting a bar-like object in most environments, Hough Transform (HT) is an essential algorithm for classification due to its ability to act as a line detector and its resistance to noise. However, HT is not suitable for direct FPGA implementation due to its heavy reliance on parametric equations and the necessity of look-up tables and large multiplier blocks [77]. Fast Increment Hough Transform 2 (FIHT2) however, developed by Koshimizu and Numada, has proved to be a comprehensive alternative to usual HT [78]. This is because it does not require any trigonometric calculations to generate the Hough distributions but still works as a line detector. In the work of Tagzout *et al.* [77], an implementation of FIHT2 in an FPGA is presented . We derive our algorithm from some of their work to avoid using parametric expressions by using the following:

$$p_{n+1} = p_n + \varepsilon p_{\frac{K}{2}+n}, \qquad 0 \leq n < \frac{K}{2} \tag{4.8}$$

$$p_{\frac{K}{2}+n+1} = p_{\frac{K}{2}+n} - \varepsilon p_n. \tag{4.9}$$

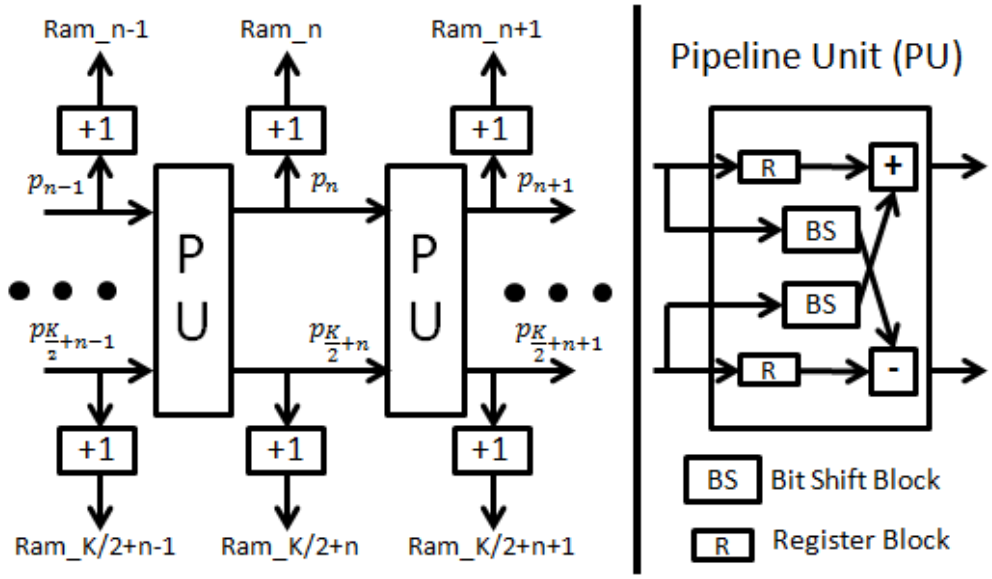

Figure 4.8: Parallel Hough transform architecture.

With $p_0 = x$, $p_{\frac{K}{2}} = y$, where n, ε and K are indices, the resolution and the number of quantizations in the image space. With this algorithm, the length of Hough generator pipeline is divided by two, because for each specific value in image space this algorithm can generate p values, two by two. Moreover, when ε is selected carefully, the multiplication in Formula 4.8 could be replaced by a shift operation. When implemented in an FPGA, we model parallel Hough transform (PHT) as shown in Fig. 4.8. In the figure, each RAM contains Hough space in both inner and outer areas, because there is no requirement to access these RAM locations at the same time. By doing this, the number of FPGA slices used to construct the RAMs is significantly decreased, while at the same time the number of RAM accesses is increased. Each pipeline unit requires only two registers, two bit shift blocks, one adder and one subtractor. The calculated value of p_{index} is treated as the corresponding memory address in Ram_index with the accumulator of the value in this address incremented by one. The number of ram is related to the number of quantizations in the image space, K.

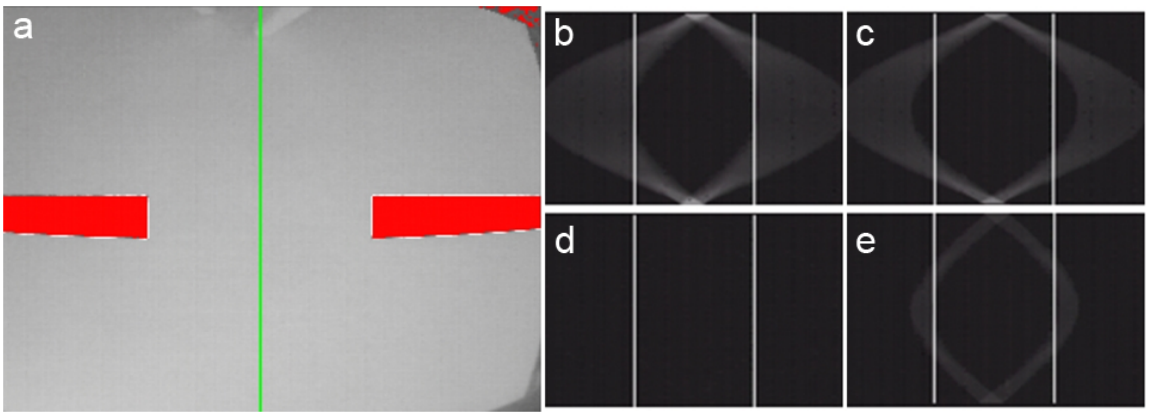

Figure 4.9: a): Output image after analyzing a discontinuous bar-like object. b): Hough space of the whole image. c): Hough space of the outer-area-image. d): The result of addition and suppression processing Hough space. e): Hough space of the inner-area-image.

Addition and Suppression

The image is separated into an inner and outer area. Each area is separately transformed to Hough space. The inner area Hough space verifies that only bars passing the valid region are detected. An addition and suppression process is applied to produce the final Hough result from each Hough space. The procedure scans the whole inner Hough space and if the vote value is more than a specific value, proportional to the diameter of the yellow circle in Fig. 4.6, it adds the value in outer Hough space. Otherwise the value at the same position in outer Hough space is replaced by zero. The effect of this method is shown in Fig. 4.9 and Fig. 4.10.

In Fig. 4.9a), no bar passes the valid region which results in blank Hough space as shown in Fig. 4.9d). On the other hand, In Fig. 4.10d), the main components in the Hough space still remain while insignificant values are eliminated. At the same time, the algorithm searches for a maximum vote value within an acceptable distance range, indicated by a black circle in image space in Fig. 4.6 and by white lines in Hough space as seen in Fig. 4.9 and 4.10.

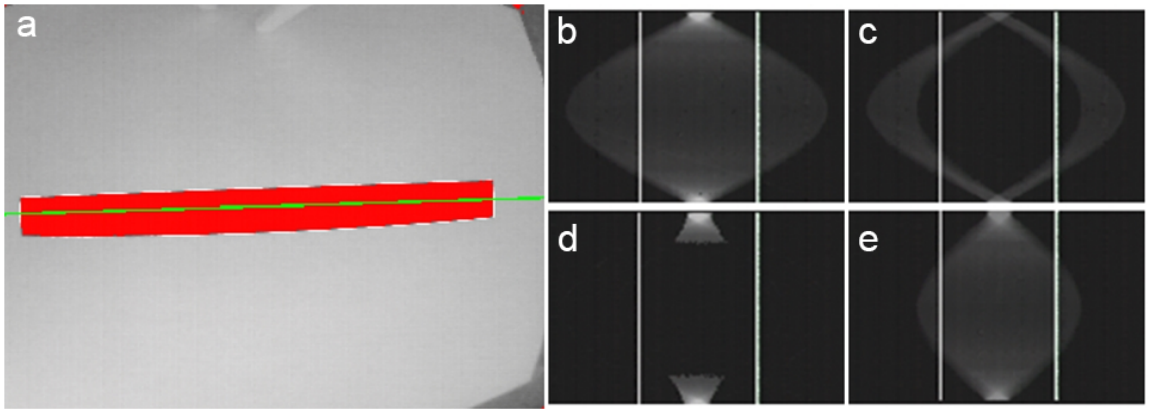

Figure 4.10: a): Output image after analyzing a continuous bar-like object. b): Hough space of the whole-image. c): Hough space of the outer-area-image. d): The result of addition and suppression processing Hough space. e): Hough space of the inner-area-image.

Nearest Bar-like Object Extraction

In order to extract essential information from Hough space, thresholding is a fundamental step. The threshold value is equal to our selected ratio multiplied by the maximum vote from the previous process. If the maximum vote is lower than our specific constant, the threshold value is set to a certain minimum constant. Due to the characteristics of Hough space, each bar-like object in the image is transformed into a diamond, eclipse or hexagonal shaped object in Hough space. We further visualize how the bar shaped object correlates to a diamond shaped object in Hough space in Fig. 4.11. We designed tip template images based on this principle, as shown in Fig. 4.12, to find tips of every object in Hough space. For tail searching, we simply check for blank pixels. The pseudo code shown in Algorithm.2 demonstrates how nearest bar-like objects are extracted. The function of the algorithm is to find the largest acceptable diameter, bar-like object from Hough space. If there are more than one largest diameter objects i.e. two bars with exactly the same diameter, it will select the object which is closest to the vertical center. If the recognized tip is blunt, the result will be calculated by averaging blunt tips.

```
Data: Hough space
Result: Nearest bar-like object
initialization;
for each angle index n do
    start tip matching;
    for each distance p do
        if tip is found then
            Save the tip, stop tip matching and start tail searching;
        end
        if tail is found then
            if length is in acceptable range AND (length > local length OR
              (length = local length AND nearer to the vertical center) ) AND
              (tip+tail)/2 is in acceptable range then
                set local length to length and set local tip to tip ;
                // (tip+tail)/2 range is shown as white lines in output
            end
            start tip matching and stop tail searching;
        end
    end
    stop tip matching and tail searching;
    if local tip = global tip AND local length < P*global length AND previous
      angle index = current index-1 then
        if local length> global length then
            set global length to local;
        end
        calculate average angle index and average distance, set global tip to
          local and set previous angle index to current;
    else
        if local length > global length then
            set global length to local, set global tip to local and set new average
              angle index and distance;
        end
    end
    set local length to zero;
end
```

Algorithm 2: Nearest bar-like object extraction

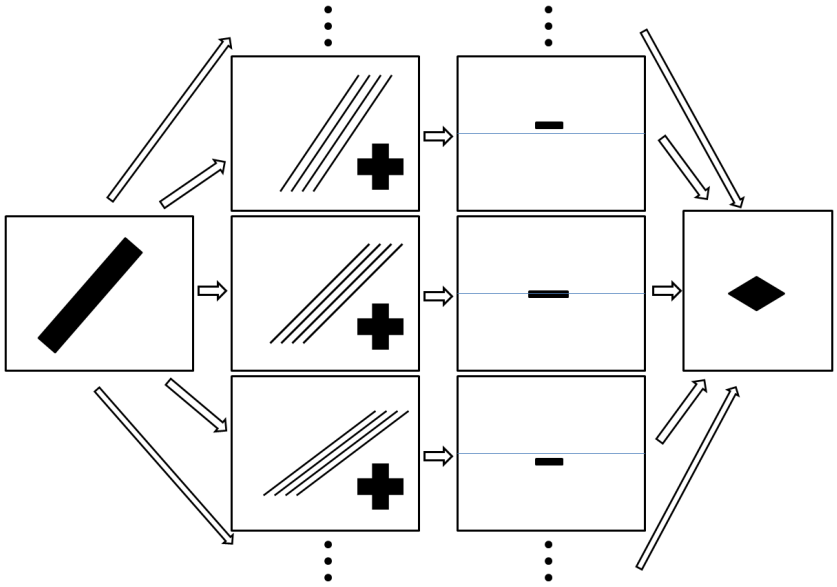

Figure 4.11: Visualization of how a bar like object transforms into a diamond or eclipse shape object in Hough space.

0	0	X	0	0	0	0	0	X
0	X	1	0	0	0	0	X	1
X	1	1	X	1	1	X	1	1
0	X	1	0	X	1	0	0	0
0	0	X	0	0	X	0	0	0
0	1	1	0	0	0	0	1	1
0	1	1	0	1	1	0	1	1
X	1	1	X	1	1	X	1	1
0	1	1	0	1	1	0	1	1
0	1	1	0	1	1	0	0	0

Figure 4.12: Tip template images. X indicates irrelevant values.

In our algorithm, *length* is equal to $tip - tail + 1$. *Local length* is the maximum *length* in each angle index n. *Global length* is the maximum *length* from the start of the process. *Local* and *global tip* are used for calculating the average of *angle index* and *distance* in case the tip is blunt. P is gain in range 0 to 1.

Conclusion

Although in our experiment the algorithm was used to successfully orient a UAV towards a bar, there are some situations in which it is difficult for the robot to accomplish the task. Those situations include a fast moving bar or environments where strong side wind makes the robot rotate around yaw axis in hover state. This is because currently, only P-control is used in the visual feedback control system, which is insufficient to counter these problems. One solution would be to implement a proper PID or PD-control, which is unfortunately, not a trivial implementation in FPGA hardware. In this research, we also assume that the object has low brightness and that we will perform static thresholding to classify objects from background. In poor light situations, such as at night or in a place with high intensity light, this assumption can mislead the classification. The solution would be to use a better thresholding method or apply preprocessing before thresholding. There are a few suitable adaptive threshold methods such as Otsu-, Kittler-Lllingworth- and Local Entropy thresholding. Unfortunately implementing these methods within an FPGA environment would require an immense degree of work. Which brings us to the final conclusion that, while there are definitely merits of using an FPGA based platform, due to the fact the implementation of even the simplest algorithms is non-trivial, using an FPGA for a vision based navigation UAV platform might not be desirable. Using the knowledge collected in the FPGA based approaches, the next goal was to find a navigation and localization method for a small size UAV of higher complexity that would overcome the limitations of the FPGA algorithms presented here. An approach that was identified as feasible was to use AR-markers for localization, size they give us information about the position and distance to the marker (like in the on-board target tracking with an active camera) but also information about rotation

and attitude towards the marker (like the on-board target tracking with orientation measurement)

4.2 AR-Marker Based Localization

Camera pose estimation is a very common problem in the field of robotics. A typical approach is to find image features in the environment and then extract the movement of the camera by observing the movement of these image features on the image plane. Other solutions towards localization include the use of laser range finder or RGB-D cameras (e.g. Stowers *et al.* [79]). The problem with these approaches is that they often pose a heavy strain on the available payload due to their weight or are computationally too expensive to be considered for an on board system. While all these approaches can be a very useful when searching for the relative position, the position of the camera relative to its own coordinate system, these methods fail when the cameras absolute position is required.

When the absolute position of the camera, meaning the position of the camera in relation to an external coordinate system is required, this approach is less feasible. In this case we either have to rely on outside observation, as with a Vicon or similar motion capture system [80] or make use of an inside-out observation system, like using fiducial marker.

There are many fiducial marker systems as can be seen in Fig. 4.13. One that became popular outside of research environments and is used for so-called argumented reality (AR) for cell phones or other mobile devices, is the AR-marker system. The most popular AR-marker system in the academic environment for the last decade has been the ARToolKit marker system. This is certainly because of its good performance but also partly because it was included as a standard package in OpenCV, with an open source computer vision library, widely used in academia.

Two major problems with AR-markers are occlusion and illumination. For most markers (as with the ARToolKit marker) if part of the marker is occluded, the detection of the base shape of the marker (in the case of the ARToolKit marker, a square)

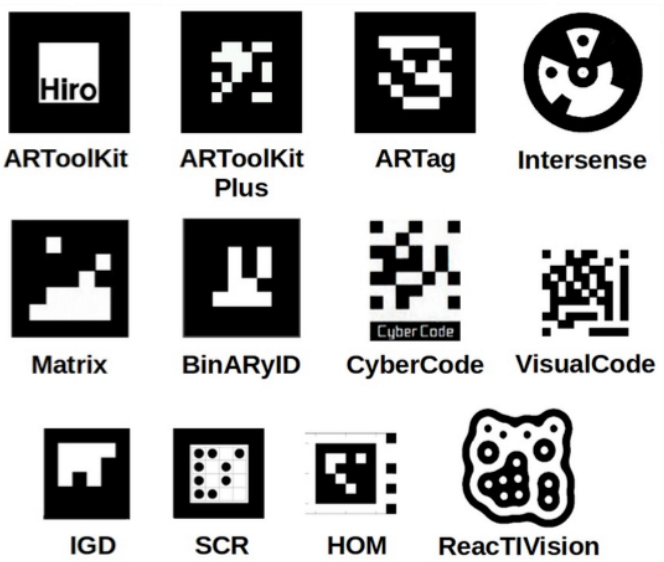

Figure 4.13: An example of various fiducial marker systems. One of the markers most commonly used in the last decade is the ARToolKit marker. (picture taken from Garrido-Jurado et al [81])

completely fails and the reconstruction of the camera position becomes impossible. A similar problem occurs when there are reflections of surrounding light sources on the marker. While a detection of the base shape might be possible in these cases, the information that is visible on the marker itself might be over- or underexposed in the camera plane, and hence the reconstruction of the exact camera rotation or direction fails.

It was therefore decided to use a marker system that is more reliable in avoiding occlusion and illumination errors than the commonly used ARToolKit marker system. Instead of building an AR-marker system from the ground up, the decision was made to use the ArUco marker library, developed by Garrido-Jurado *et al.* [81]. It also employs the ability to generate markers with a high dictionary distance, making multiple markers easier to distinguish from each other [82].

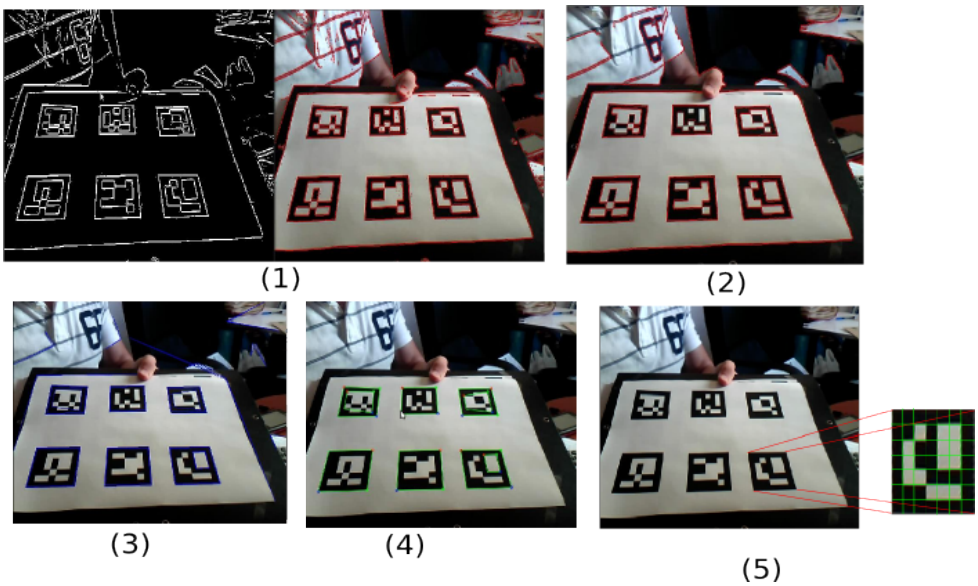

Figure 4.14: Sequential explanation of how the ArUco library extracts the camera position and orientation from the ArUco marker board (image taken from the official ArUco website [83])

4.2.1 ArUco Marker Library

The following chapter briefly introduces the ArUco marker library, its approach and function. A more detailed description can be found in the proceedings from Garrido-Jurado *et al.* [81].

The ArUco marker system was released as an open source library in 2014. Similar to its predecessor, the ARToolKit, the ArUco marker system has been added as a standard library in OpenCV. In contrast to ARToolKit, the ArUco maker library anticipates occlusion and tries to handle it accordingly. This is achieved by using a cluster of AR markers to extract the camera position, instead of using only a single marker. The following is the sequence in which the camera position extraction takes place (see Fig. 4.14 taken from [81]):

- An adaptive thresholding is applied to extract the borders of the individual markers (Fig. 4.14.1)

- To prevent the detection of undesired borders, borders with a only a small number of points are removed (Fig. 4.14.2)
- Polygonal approximation is applied to find concave contours with exactly 4 corners (Fig. 4.14.3)
- Rectangles which are too close to eachother are removed. This results in the removal of rectangles the are detected within the marker itself (Fig. 4.14.4)
- After that, the actual marker detection commences:
  - Projection perspective is removed to obtain a frontal view (Fig. 4.14.5)
  - A threshold is applied over the area according to an algorithm by Otsu [84].
  - The marker is divided into an $n + 1 \times n + 1$ grid where the internal n x n cells contain the marker id information and the outer cells consist of a black border (in the Fig. 4.14 example, $n = 5$)
  - Marker id is checked for validity against a previously constructed, custom dictionary
  - For valid markers, the corners are refined using subpixel interpolation
  - If the intrinsic camera parameters are provided, the camera position and orientation are extracted.

Feasibility Test

To analyze whether or not the ArUco AR-marker system is feasible for use in an on-board visual navigation system for a small sized UAV platform, a feasibility test was performed (see Fig. 4.15). A square marker board of 0.15m size with a marker set of 8 individual markers (Fig. 4.16) was placed under good lighting conditions in different distances (1m, 2m, 3m) from two calibrated USB cameras mounted on a UAV platform set to a resolution of 640×480. The ArUco library was running on a ARM system and the marker position was saved for further analysis. The marker was

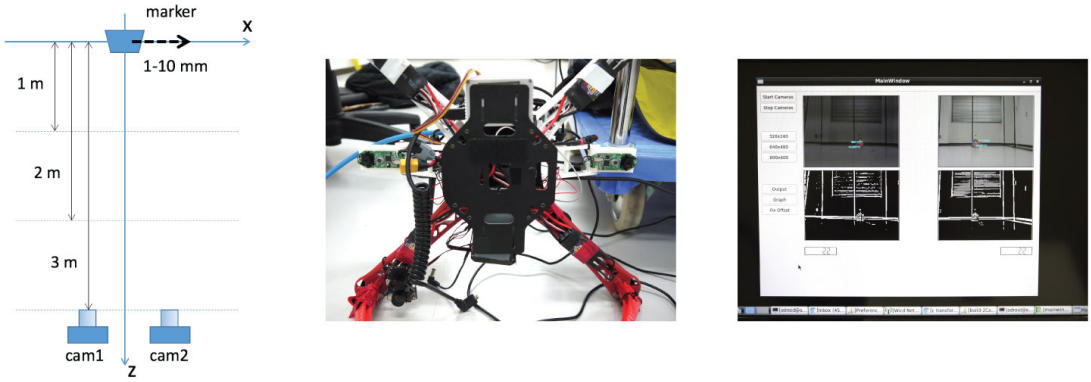

Figure 4.15: (left) Schematic of how the feasibility test for the ArUco marker system was conducted (middle) Dual camera system mounted to a quadrotor (right) Dual camera marker system interface

then moved by 0.001m and then several measurements were taken with the ArUco library approach for each camera. This was repeated 10 times, until the marker moved a total of 1cm. The results can be found in Fig. 4.17.

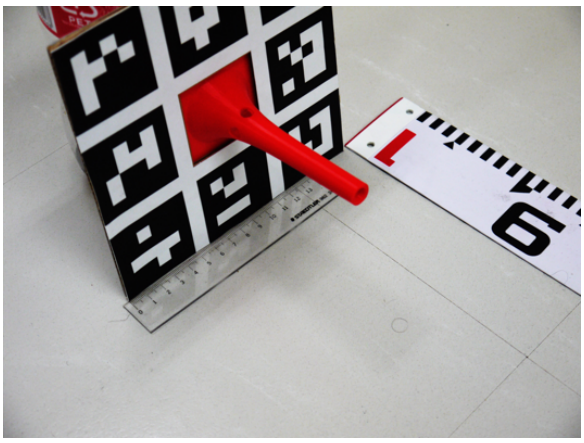

Figure 4.16: The marker used in the ArUco marker system feasibility test.

The deviation of error if the marker is 1m from the cameras, is within 0.001m. At 2m distance we see a bigger deviation of up to 0.004m, but considering that the behaviour of the measurements is still linear and the average of the values corresponds with the ground truth, the results were satisfying enough to further pursue

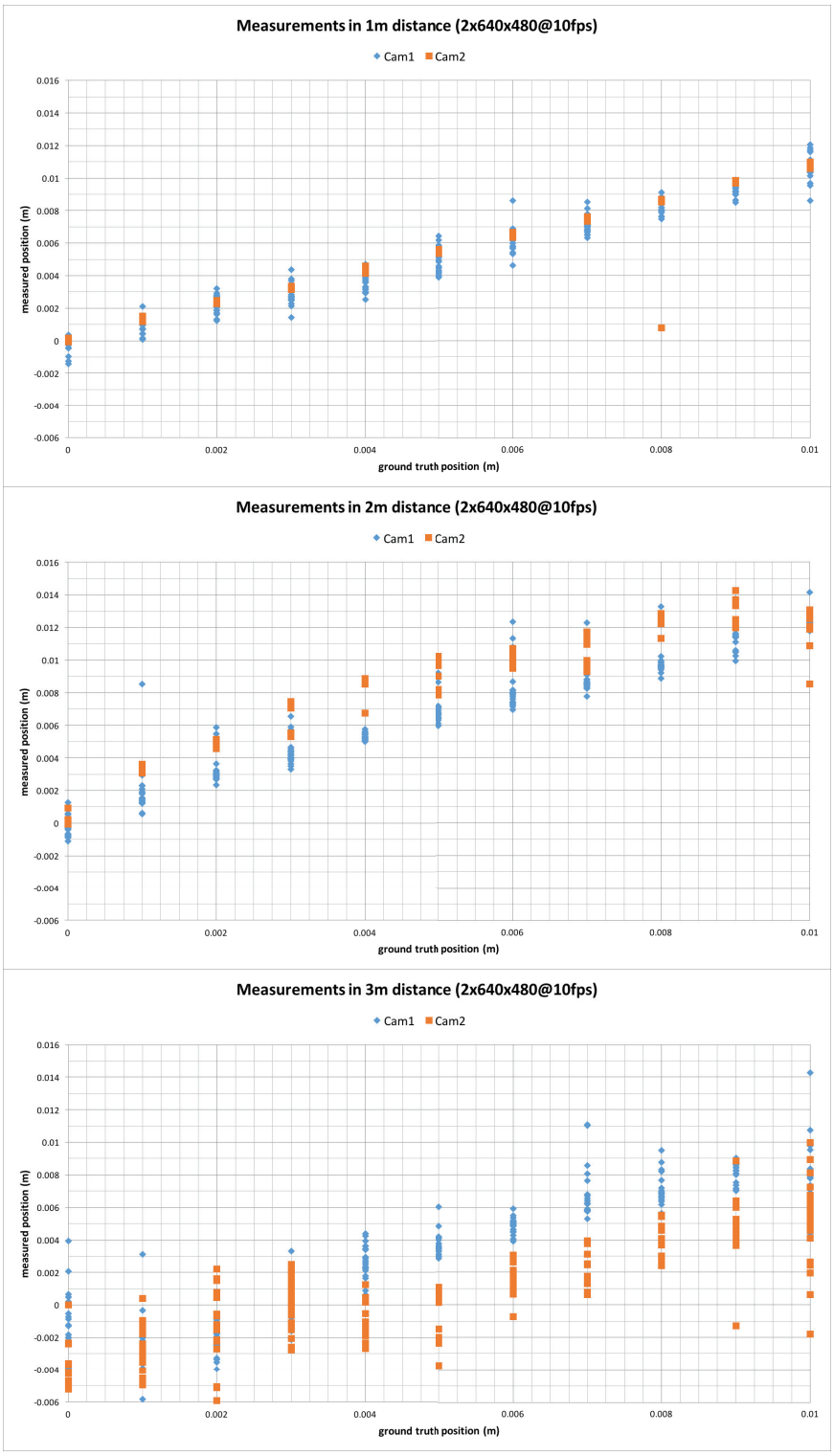

Figure 4.17: Results of the feasibility test of the ArUco marker system. Measurements were taken at 1m, 2m and 3m distances from the a dual camera setup.

this method. For 3m distance we can see a further decrease to quality of measurements. The main reason for this behaviour was the relatively small resolution of 640x460 pixel, which did not provide enough detail for the ArUco marker system to reliably extract the marker position. To mitigate this, the marker size or camera resolution needs to be increased. Since an increase in camera resolution would impact computational expense, it was decided to instead slightly increase the marker size in further experiments.

4.2.2 Dual Camera AR-Marker System

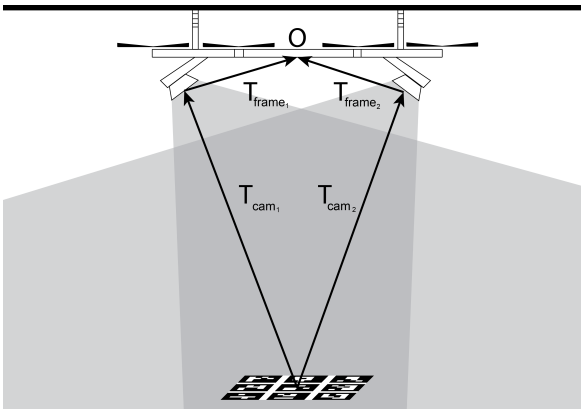

Figure 4.18: Schematic of the dual camera system. The area under the hexrotor is covered by two cameras simultaneously and enables very precise positioning over the target with the utilization of sensor fusion.

The ArUco marker library is used with two standard USB cameras with a resolution of 320x240 pixel at a frame-rate of 60FPS. After the initial camera calibrating process, in which the intrinsic camera parameters are found, we are able to analyze the received camera feed with the help of the ArUco library, and get an estimate about the current AR-marker board position and orientation relative to the camera, as described in the last chapter. We use this information to calculate the current position Pos and rotation Rot of the camera and create the translation matrix T

with:

$$T = \begin{bmatrix} Rot & Pos \\ 0 & 1 \end{bmatrix}. \tag{9}$$

To get the position of the drone O_n extracted from analysing camera picture n we calculate:

$$O_n = T_{cam_n} T_{frame_n}, \tag{10}$$

where T_{frame} is the known homogeneous transformation matrix between the camera and the center of the drone.

Kalman Filter

To further reduce the redundancy and thus accuracy of this system, the ArUco marker library system is extended by adding a secondary camera. In the initial approach, both cameras are tracking the same single target. Naturally, both camera position and orientation estimations $O_{1,2}$ suffer from noise in the current calculated position data. To slightly reduce the noise, we take advantage of having multiple sensors for the same data set and apply the Kalman filter for sensor fusion on the received data to get a more refined estimate $\hat{O}$ of the current drone position and orientation. First, a prediction is made with:

$$\hat{O}(k) = A\hat{O}(k-1) + Bu(k) \tag{11}$$

$$P(k) = AP(k-1)A^T + Q \tag{12}$$

u being the measurements, P the error covariance matrix and Q the noise covariance matrix of the measurements. That is followed by an update process with:

$$K(k) = P(k)C^T(CP(k)C^T + R)^{-1} \tag{13}$$

$$\hat{O}(k) = \hat{O}(k) + K(k)(Z(k) - C(\hat{O}(k))) \tag{14}$$

$$P(k) = (I - K(k)C)P(k) \tag{15}$$

K being the Kalman gain, Z the state measurements and R the process noise covariance matrix.

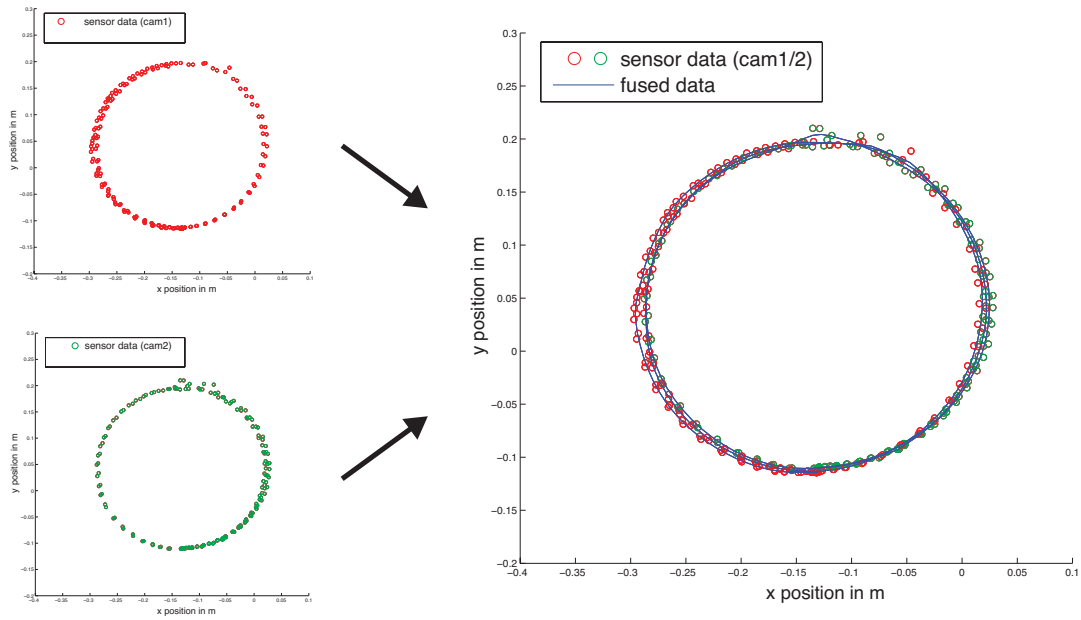

Figure 4.19: Result of applying the implemented Kalman filter on one AR-marker at 2m distance that has been moved in a circular motion with a radius of 0.15m with 20 RPM. Although the cameras already give detailed positional data, the filter helps to smoothing out the noise present in the location data.

In Fig. 4.19 we show the exemplary results of tracking an AR-marker as it moves in a circular motion of 0.15m radius in a distance of two meters with 20 RPM, after Kalman filter sensor fusion. These results enable us to precisely track a marker and calculate the exact position of the hexrotor relative to the recognized marker within a relative small error margin of 0.001-0.002 m.

Chapter 5

Positioning on the ceiling with an omni-wheel drive system

5.1 Hardware

After initial trials that failed to find a good solution for precise movement of a multi-rotor in its neutral hover state (see chapter 3) a novel concept was designed. Instead of relying only on the propeller for propulsion, an idea was formulated to add another method of propulsion that would have a higher degree of accuracy than the propeller, which are easily disturbed by external disturbance such as wind or turbulence.

This led to the idea to add a drive system on top of the UAV and establish force contact with the ceiling. As discussed in chapter 1.2 there have been several research projects that have dealt with establishing force contact between a UAV and a ceiling or wall. The goal of this project is not only to establish force contact with the ceiling to stabilize the UAV, but also to enable precise force control and movement on the ceiling.

In this chapter, the development and design process of a UAV prototype with an upwards directed omni-wheel drive system is explained and documented.

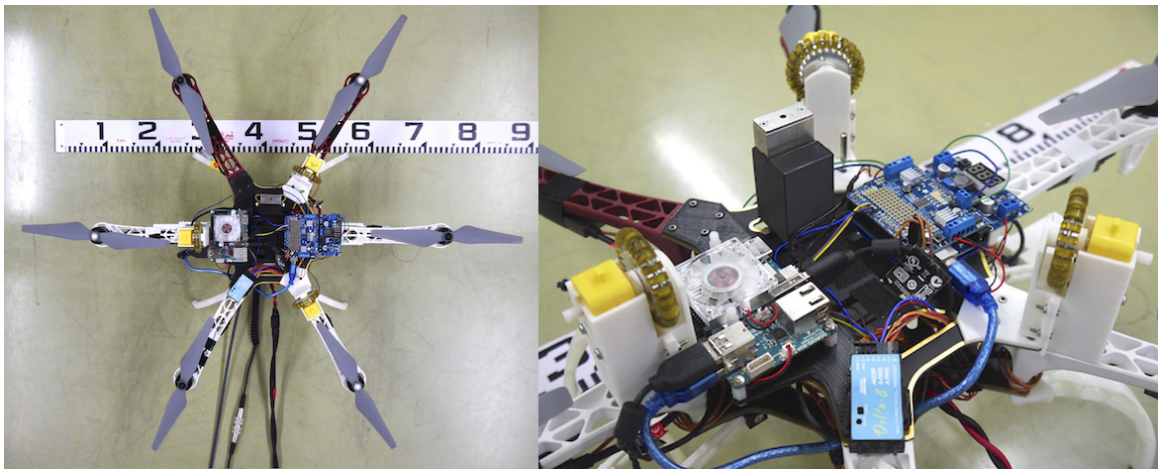

Figure 5.1: Total- and detail-picture of the hexrotor platform prototype. This is the initial Version 1 presented at IROS 2016 (CITE).

5.1.1 Version 1

To allow for high manoeuvrability and precision, while still keeping the main body of the MAV as close as possible to the ceiling and thus avoiding an unfavourable distribution of the center of gravity, we introduce a MAV with a top mounted secondary omni wheel propulsion system (See Fig. 5.1). We use an off-the-shelf multirotor hexrotor platform, the DJI F550 frame with a DJI Naza-M-V2 flight controller with all its necessary components (electronic speed controller (ESC), remote receiver, power distribution board), as a base. The maximum span from propeller to propeller is 80cm. Additionally, we mounted three no brand geared 9V DC-motors with a maximum stall torque of 40mNm with thin, aluminum omni-wheels (20g) attached to three of the six hexrotor arms (Fig. 5.2). For this purpose, a custom motor mount was CAD designed and 3d-printed with a MDF-printer out of ABS plastic (Fig. 5.3). We also mounted an industry-grade precision print head (REA-JET DS) to the frame, at a slightly lower height than the omni wheels. In accordance with the print head specifications, this leaves a gap between the wheel contact plane and print head of 1.5 cm.

Since we aim for the system to be used as a completely independent platform without any need for a remote operator or wireless base station connection, we also

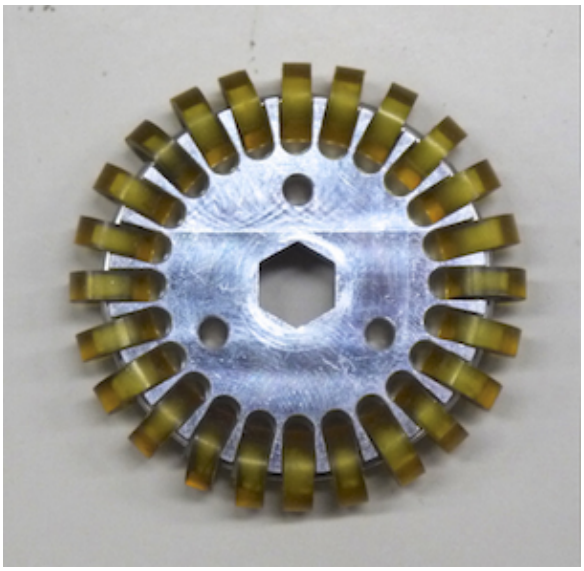

Figure 5.2: The omni wheels used in this work have a radius of 2.25cm and a weight of 20g.

mounted an ARM-based processing unit with a Samsung Exynos5422 Cortex-A15 at 2Ghz and Cortex-A7 Octa core CPU with 2 GB of RAM to the frame. This processing unit is used to compute the high precision localization with the help of AR-markers and dual camera sensor fusion. It is connected to an Arduino-based, Atmega328-driven motor driver via Serial-USB as well as two 60fps USB web cams with variable resolutions. A schematic of the connections between the different parts can be seen in Fig. 5.4. An ARM-compatible Linux build serves as an operating system for the processing unit. We power the system either by Li-Po battery or via tethered external power supply (16V at 80A max). The power draw of all components apart from the main hexrotor system is 15W at maximum omni wheel motor speed and simultaneous dual camera AR-marker analysis. Considering the overall maximum power draw is 1280W, this amount is nearly insignificant.

Pre-flight setup requires a 5mm diameter tube to feed the print head with ink, as well as a controller cable to connect the print head. The system can be powered by a Li-Po battery or a ground-based power station. When powered via tether, the system uses a total of one tube and two cables. The weight of the whole system is

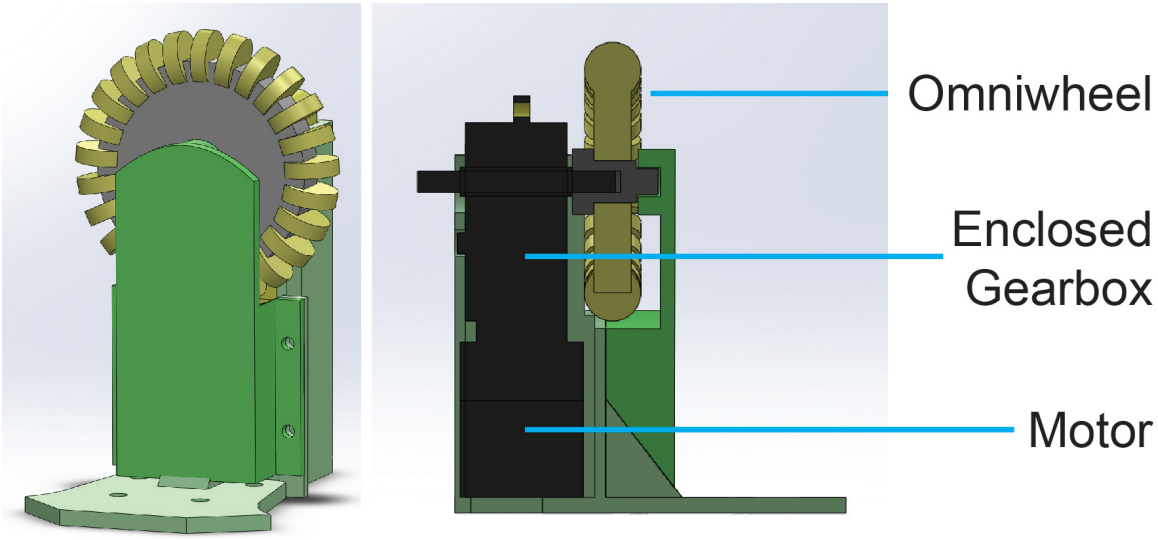

Figure 5.3: Cut schematic through the Version 1 motor mount.

2.1kg. Based on the specifications of the motor and propeller used in this project, one motor/blade combination is able to generate enough thrust to lift around 700g. According to these specifications, the system is still able to lift an additional payload of around 2.1kg.

5.1.2 Version 2

Since the motors used in Version 1 of the prototype did not provide the drive system with a a high enough torque to reliably move along the ceiling (as seen in the flight tests discussed in 7.1.1), the decision was made to exchange the simple no-brand 9V DC motors from Version 1 with higher torque Maxon 215341 motors, which are usually used for robotic applications, in Version 2 of the platform (Fig. 5.5). The maximum torque for these motors is 67mNm. Since they also come with a built-in encoder with a resolution of 360degree/2100, they can feed the current turning speed of each wheel back to the system. While we cannot rely on odometry alone in a holonomic drive system due to the large amount of slipping that is common in such a system, in combination with the dual camera system it provides a potent tool for precise localization.

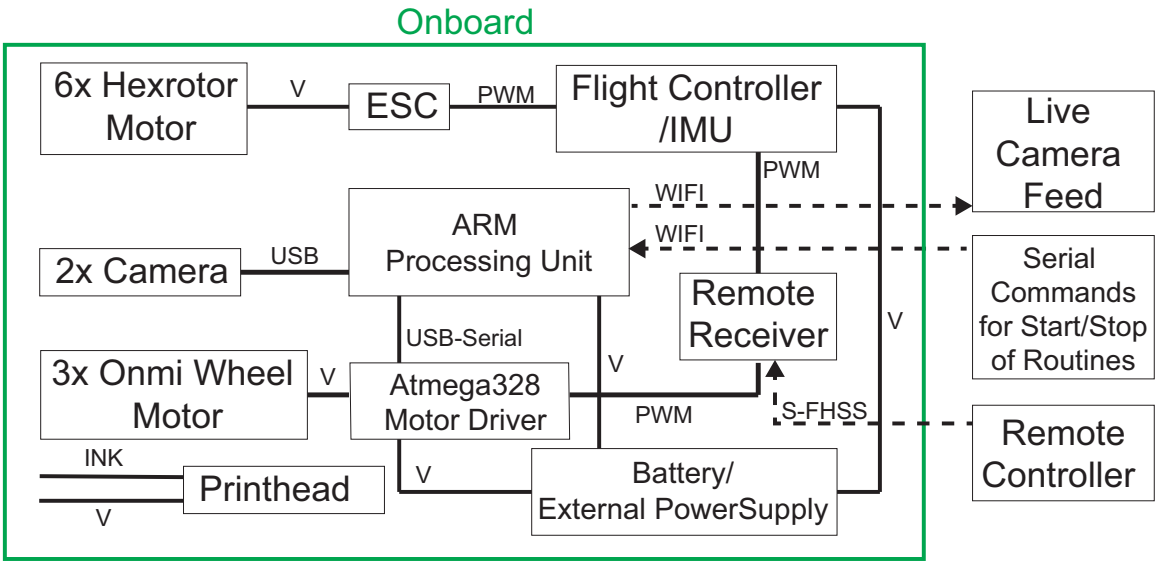

Figure 5.4: Connection chart of all parts mounted to the frame in Version 1 of the aerial platform.

Additionally the decision was made to integrate a force-sensing capability into the system. After initial fight tests (see chapter 7.1.1, it was deemed necessary to measure the applied force to the ceiling to avoid exerting unnecessary force on the ceiling which would put too much strain on the motor torque. To achieve this a small size load cell with a resolution of 0.01N was added to the motor mount design.

To avoid damage to the motor on impact and to reliably establish contact force with the ceiling, even if it is not perfectly planar, the motor is mounted in a spring loaded design.

All larger scale motor mount parts were again printed with a MDF 3d-printer using ABS plastic. Due to the precision required, the axle, as well as the gear mounts and axle nuts, were printed from acrylic in a stereolithography process.

The complete design of the Version 2 motor mount can be seen in Fig. 5.6.

A custom, Arduino compatible motor mount add-on module was designed to drive the motor, enable the Atmega chip to interface with the load cell and to enhance the cable management. It integrates the cable connector for voltage supply by the motor shield and also passes through signals from the encoder to the Atmega microchip,

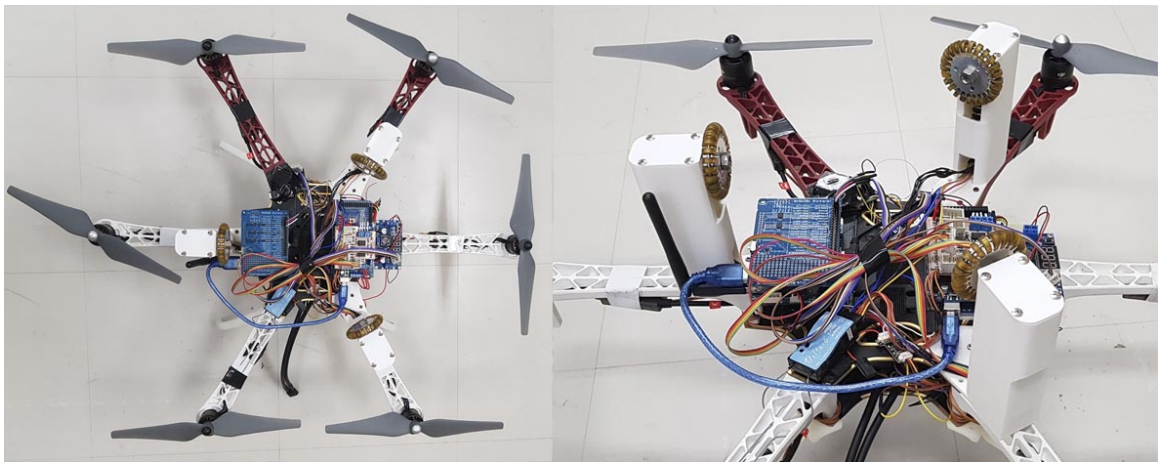

Figure 5.5: Total- and detail-picture of the hexrotor platform prototype. This is Version 2 of the aerial platform with stronger motors, encoders and force sensors, as well as a spring mounted drive system.

that is handling the AD conversion and forwarding the digital encoder signal to the ARM-based processing unit. It also acts as a load cell operation amplifier. All these functions are fit into one single board(see Fig. 5.7). The circuit board was custom designed and made with a CNC drilling machine. A picture of the circuit board and its schematic can be found in Fig. 5.8.

Since we lost access to the print head, at this stage of the project no additional tool was mounted on the aerial platform. Due to the added weight of the new motors as well as additional plastic and metal parts required for mounting the motor and the motor shield add on module, the weight of the whole system did not change significantly in comparison with Version 1 of the prototype and came in at 2kg. This left a theoretical payload of 2.2kg. One thing that significantly changed in comparison to the Version 1 design was the distance between the propeller and ceiling in the docking state. The distance minimum distance between ceiling and propeller of Version 2 increased to 7 cm, while the original distance of Version 1 was 4.5 cm.

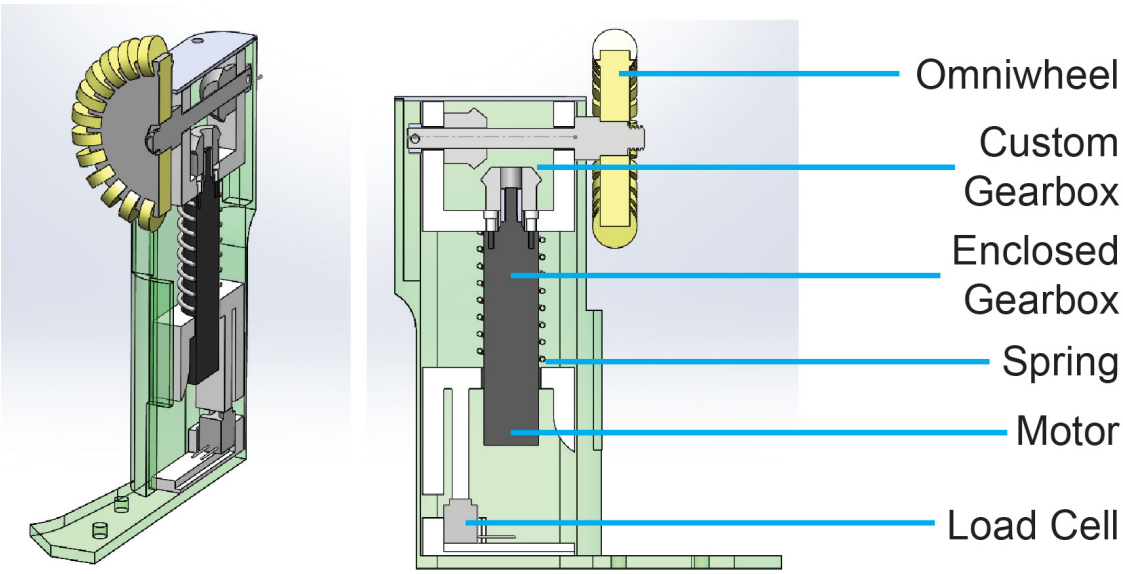

Figure 5.6: Cut schematic through the Version 2 motor mount.

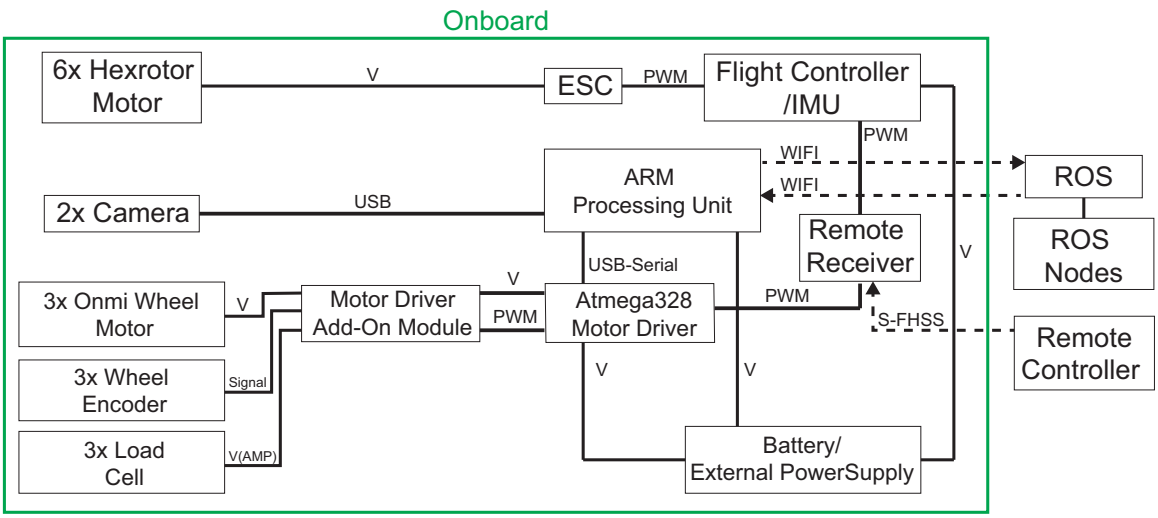

Figure 5.7: Connection chart of all parts mounted to the frame in Version 2 of the aerial platform. A motor mount add-on module was added to drive the new motors, encoders and load cells. The connection diagram stayed the same in Version 3.

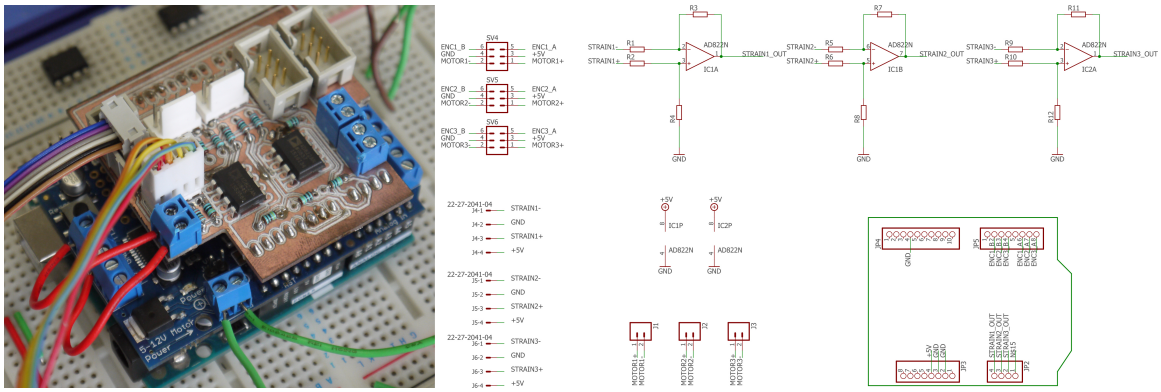

Figure 5.8: (left) Picture of the motor mount add-on. (right) A schematic of the motor mount add-on.

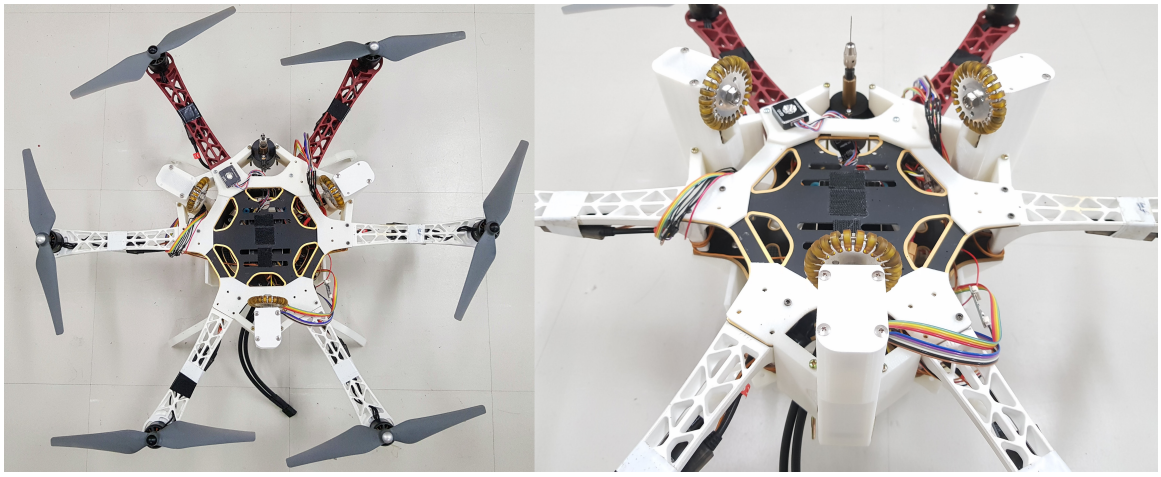

Figure 5.9: Total- and detail-picture of the hexrotor platform prototype. This is Version 3 of the aerial platform. The drive system has been lowered, and the electronics has been moved to the bottom of the aerial platform.

5.1.3 Version 3

The Version 3 (Fig. 5.9) is the latest version of the prototype aerial platform. The overall setup of electronics and motors stayed the same between Version 2 and 3 (Fig. 5.7). The motor mounts (Fig. 5.10) have been redesigned to reduce the gap between the ceiling and the hexrotor propellors, as shown in the comparison image between the motor mounts of all three versions in Fig. 5.11. Furthermore, the electronics have

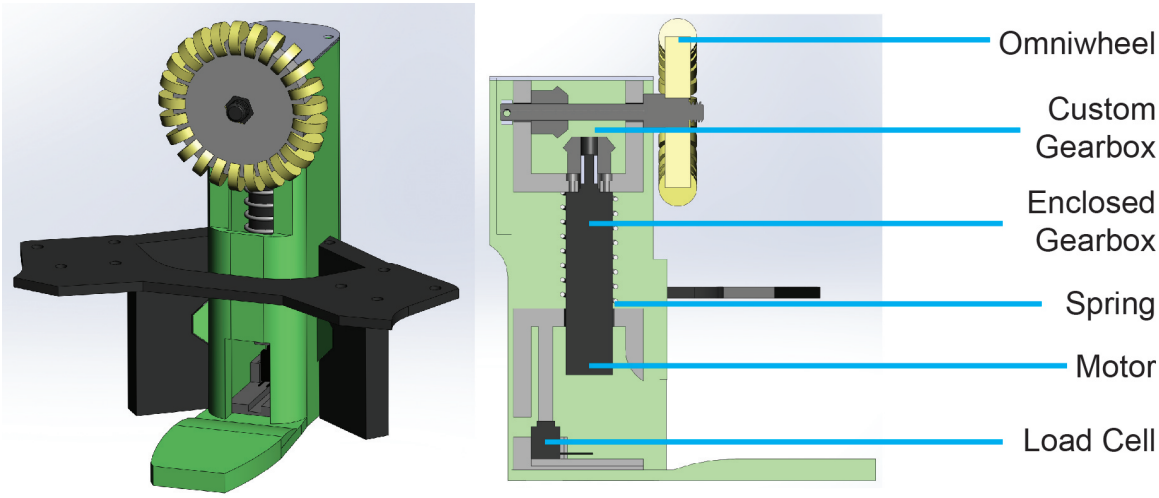

Figure 5.10: Cut schematic through the Version 2 motor mount.

been moved to the bottom of the hexrotor frame to avoid accidental collisions with the ceiling and also to lower the center of gravity of the platform.

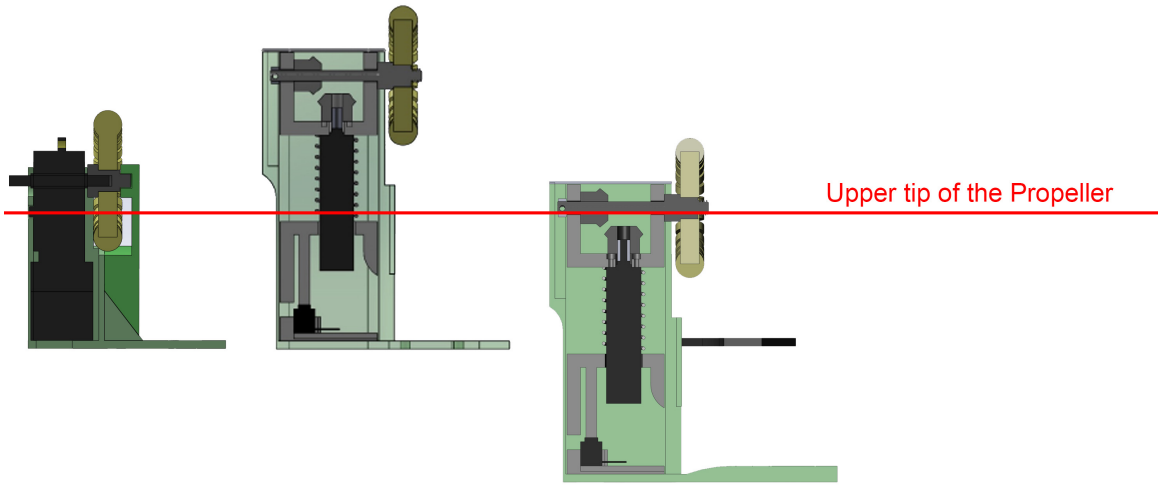

Figure 5.11: Caparison between the motor mounts of all three versions.

5.2 Omni-wheel Control

The hexrotor can be controlled mid flight via an off-the-shelf flight controller using typical multirotor kinematics [29]. After establishing contact with the ceiling, the

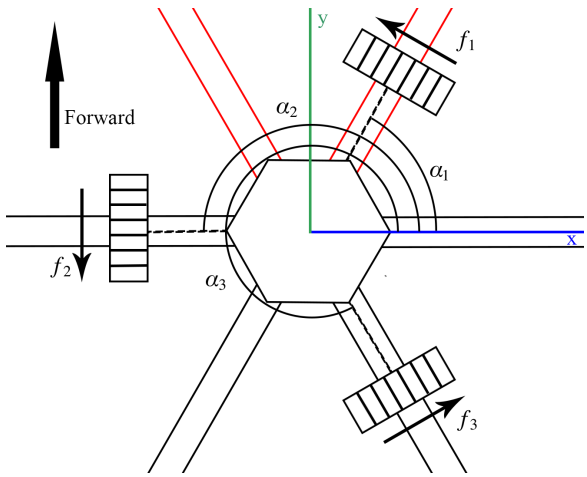

Figure 5.12: Schematic of the 3-wheel omni wheel drives.

drive is switched to the top mounted omni wheel system. The decision to use a three wheel omni drive system was made for several reasons. The strongest rationale is that an omni-wheel drive is a holonomic drive system, enabling movement in all directions on the ceiling plane without needing to change orientation. Not only does this make the manual control for an operator simpler, it also enables continuous drawing on the ceiling with the print head, without the need to pause the drawing and reorient as would be necessary in a non-holonomic drive system.

Another reason is the fact that a multirotor frame is constantly exposed to heavy vibration due to the flapping phenomenon of the fast turning propeller [85]. This causes extemely noisy inertia measurement unit (IMU) data readings mid-flight and is a general problem for the flight stability of multirotors. The three omni wheels create a contact plane, making contact with the ceiling very stable and reducing micro vibrations. Since three points in a three-dimensional space create a plane, adding an additional fourth wheel potentially creates a small gap between the wheel and the ceiling, which would cause strong vibrations throughout the system. This makes a three wheel solution more suitable for our project than a four- or more-wheel solution.

In Version 1 and 2 a configuration of $\alpha_1 = \frac{\pi}{3}, \alpha_2 = \pi, \alpha_3 = \frac{5\pi}{3}$ was chosen; however, in Version 3 we use the configuration of $\alpha_1 = \frac{\pi}{6}, \alpha_2 = \frac{5\pi}{6}, \alpha_3 = \frac{3\pi}{2}$ to make room to

lower the motor mounts (Fig. 5.12, 5.5, 5.9).

For Version 1 of the system we used very simple geared DC motors, only having control over the voltage V and thus the speed and direction of the wheels. Fore Versions 2 and 3, we used motors with a built-in encoder, therefore can directly measure the rotational speed ω. Since we used wheels of the same radii and same motors, with the same distance to the center of the frame, we can safely assume that for all wheels the applied voltage V, rotational speed ω and force F_w transfered from the wheels to the ceiling,

$$V \propto \omega \propto F_w \tag{5.1}$$

holds true. We can then derive from Fig. 5.12 that, given we want to move in a certain direction x_d, y_d and rotate around the center of the hexrotor ω_d:

$$x_d = c(\alpha_1 + \frac{\pi}{2})f_1 + c(\alpha_2 + \frac{\pi}{2})f_2 + c(\alpha_3 + \frac{\pi}{2})f_3 \tag{5.2}$$

$$y_d = s(\alpha_1 + \frac{\pi}{2})f_1 + s(\alpha_2 + \frac{\pi}{2})f_2 + s(\alpha_3 + \frac{\pi}{2})f_3 \tag{5.3}$$

$$\omega_d = f_1 + f_2 + f_3, \tag{5.4}$$

with c and s denoting cosine and sine respectively. For clarity, we can write as matrices:

$$\begin{bmatrix} x_d \\ y_d \\ \omega_d \end{bmatrix} = \begin{bmatrix} c(\alpha_1 + \frac{\pi}{2}) & c(\alpha_2 + \frac{\pi}{2}) & c(\alpha_3 + \frac{\pi}{2}) \\ s(\alpha_1 + \frac{\pi}{2}) & s(\alpha_2 + \frac{\pi}{2}) & s(\alpha_3 + \frac{\pi}{2}) \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} \tag{5.5}$$

or, if simplified:

$$D = JF_w. \tag{5.6}$$

By creating the inverse, we can now calculate

$$V \propto \omega \propto F_w = J^{-1}D. \tag{5.7}$$

To avoid damage to the motor in cases where there is no encoder data available, V has to be limited. We normalize the result:

$$V_{norm} = \frac{V}{max(V)} v_{range}, \tag{5.8}$$

with $max(V)$ being the maximum desired Voltage a single motor is supposed to be driven.

Parallel to the encoder-less approach, we limit the maximum turning speed by:

$$\omega_{norm} = \frac{\omega}{max(\omega)}\omega_{max} \tag{5.9}$$

Using (5.7) an operator can steer the hexrotor with

$$D = \begin{bmatrix} x_d \\ y_d \\ \omega_d \end{bmatrix} \tag{5.10}$$

using three analog channels of a customized radio remote control shown in Fig. 5.13.

To provide a stable autonomous control, a PID controller was implemented. The PID controller uses formula 5.7 with $D = D_{PID}$ as

$$D_{PID} = \begin{bmatrix} k_p x_\epsilon + k_i \int_0^t x_\epsilon dt + k_d \dot{x}_\epsilon \\ k_p y_\epsilon + k_i \int_0^t y_\epsilon dt + y_d \dot{x}_\epsilon \\ l_p \omega_\epsilon + l_i \int_0^t \omega_\epsilon dt + l_d \dot{\omega}_\epsilon \end{bmatrix}, \tag{5.11}$$

with $k_p, k_i, k_d, l_p, l_i, l_d$ all being different PID coefficients that need to be manually tuned and x_ϵ, y_ϵ being the error towards the desired position in m and ω_ϵ being the error in direction towards the desired position in rad ($\tan^{-1}(y_\epsilon^2/x_\epsilon^2)$).

The Operator Remote

For the remote operation of a hexrotor with an upside omni-wheel we need at least 7 different channels of control. Roll, Pitch and Yaw (ϕ θ ψ) while in mid-flight, thrust while in mid-flight and while engaged with the ceiling (T), movement on the ceiling (x_d, y_d) and lastly the heading (ω_d). Each of these channels have 1024 states, a limit given by the 10-bit values the S-FHSS protocol of the remote control uses. To switch between 1024 states per channel, an analog controller in needed. A typical RC-remote has two joysticks which provide the user with a total of 4 analog channels. These are normally used for the roll, pitch, yaw and thrust of a multirotor. Some RC-remotes

Figure 5.13: The custom remote used by the operator. Since generally radio remotes come with a maximum of four analog channels, we added two joysticks and a micro controller to the remote, making the simultaneous control of a total of eight analog channels possible.

add knobs for additional channels, but this would be unfeasible for the control of x_d, y_d and ω_d.

One solution for this problem is the use of another operator with a second remote. While this would certainly be possible, requiring a secound operator would significantly rise the operational cost of the system. Additionally both operators would need to have very good coordination since thrust is controlled in free flight mode as well as when the system has established contact with the ceiling. Since operator two

would have no control over the thrust, this could lead to complications in controlling the UAV on the ceiling or abort contact with the ceiling if there are any problems.

Another solution is the addition of more joysticks. When searching the market for a consumer solution there were no compatible solutions available. The closest solution is a custom made joystick add on by the company Aerialpixel [86]. This device is glued to the side of the remote and adds 2 more analog channels. This option still leaves the system short of one necessary channel with no possibility to add another device, since a single one of these add-ons completely blocks the trainer port of the remote. It also requires the operator to take his thumbs off one of the other joysticks in order to operate the add-on. While roll, pitch and yaw are not required in docked mode, it could be dangerous to release the control in case there is an unforeseen disengage.

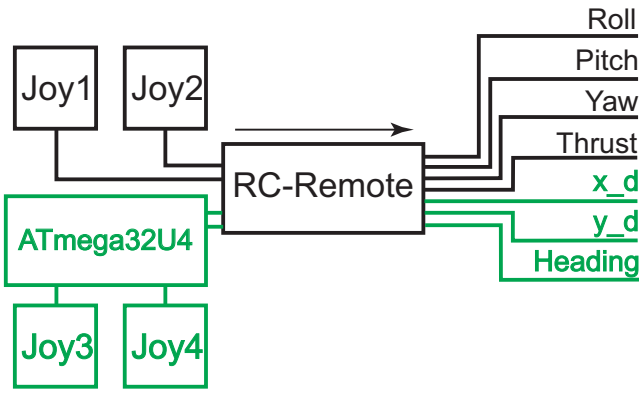

Figure 5.14: Connection schematic of the remote. Green marks indicate the components added to the standard RC-remote.

Since there was no satisfying solution readily available, it was decided to design and build a novel RC-remote controller add on. The backplate of a standard RC-remote was drilled open and two small joysticks were inserted in the back. The joysticks are positioned to be easy to reach with the index finger while holding the remote. An ATmega32U4 microcontroller was inserted into the case of the remote and programmed to perform a null-position calibration when the remote is switched on as well as a analog digital conversion of the additional joysticks analog signals to

PWM signals readable by the remote's internal control chip. The PWM signal lines were soldered internally to the trainer port of the remote, making the microcontroller behave like a second remote connected as a so-called "trainer remote" internally.

This solution now enables one single operator to control roll, pitch, yaw and thrust with his two thumbs in free flight mode. When moving on the ceiling thrust is controlled with one thumb, movement on the ceiling and heading with one thumb and two index fingers. A picture of the finished remote can be seen in Fig. 5.13 and a connection schematic in Fig. 5.14.

Chapter 6

Force Feedback

One of the major improvements to Version 2 of the aerial platform was the addition of load cells into the motor mount design (Fig. 5.6).

The decision to add a force sensing capability was made after tests with the Version 1 showed that too much thrust is applied towards the ceiling without the force sensing capability (see chapter 7.1.1). This led to two major problems. A very high power consumption and the stalling of the motors driving he omni-wheels.

6.1 Power Consumption in Multirotor Systems

When we want to raise thrust on a multirotor system, we need raise the turning speed of the motors driving the propellers of the multirotor. Naturally this also raises the power consumption. For a multirotor system, generated thrust and required current have an exponential relationship, as can be seen in Fig. 6.1.

When we press uncontrolled against the ceiling, we generate maximum thrust and the maximum amount of current the system's motors are able to draw in their maximum speed. This draws an unnecessary high amount of current, significantly reducing the flight time of the system. This was one reason to implement a more refined approach for applying force towards the ceiling.

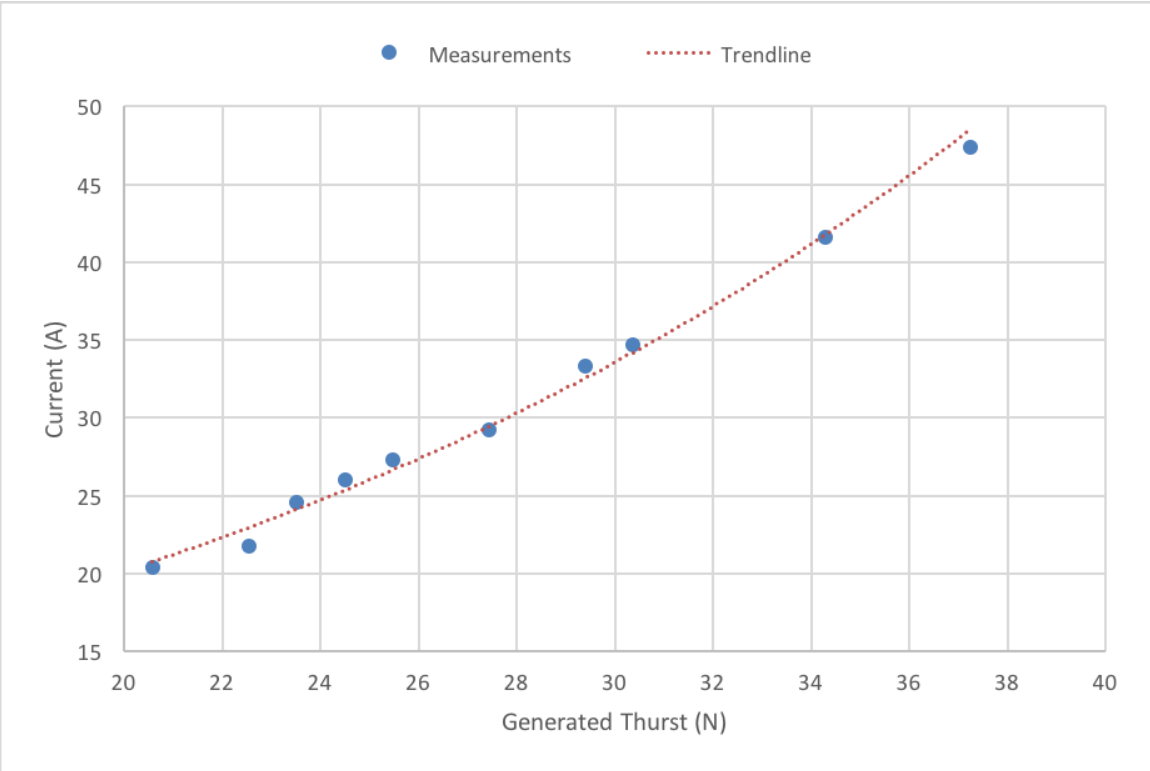

Figure 6.1: Graph showing the relationship between current consummation and generated thrust, measured in an experiment.

6.2 Avoiding Stall Torques for the Omni-wheel Drive System

Another reason for implementing force control into the system is to avoid stall torques in the motors driving the omni-wheel drive system.

Applying force towards the ceiling also raises the normal force F_n exerted on the axle of each wheel towards the mount. We can describe the impact of this force on the internal friction with the following equation:

$$F_f \leq \mu F_n, \tag{6.1}$$

where μ is the friction coefficient between the materials of the axle and frame, in our case ABS plastic and acrylic. This in turn raises the torque τ required by the

wheel motor to make the omni-wheel spin. When the required torque is larger than the motor's stall torque τ_s, the motor driving the omni-wheel will stop turning. It is important to avoid this stall torque state since, if the system remains in this state for a prolonged period of time, it can cause irreparable damage to the DC motor.

6.3 Adding Force Control

To control the amount of F_n it is important to measure how much F_n is applied towards the axle at any given time. To do this, a load cell with a maximum resolution of 0.01N was added to the design. It was decided to have a total F_{n_c} applied to the ceiling, equal to the force F_{n_g} applied to the system when it is driving on the ground. Not only does this make it possible to test with the conditions of driving on the ceiling, on the ground, but coincidentally also showed a good balance of grip and quick responding movement when doing tests on the ground. Since the F_n applied on one wheel when driving on the ground is

$$F_{n_g} = mg/3, \tag{6.2}$$

with a total system weight of 2.1kg, this leads to $F_{n_g} = 6.86N$ per wheel when driving on the ground. Therefore, the desired F_{n_c} chosen was slightly rounded up to 7N per wheel for convenience. The desired behaviour of the force sensing wheel mount is designed around the specs of the spring that was used (maximum load 14.71N, load when halfway compressed 8.83N, $k = 0.706N/mm$) to assist the system to compensate for vibration and dampen the initial impact when engaging with the ceiling, protecting the load cell. The comparison between the desired wheel mount force sensor behaviour and the behaviour of the final design can be seen in Fig. 6.2.

F_{n_c} is controlled by the Thrust of the aerial system. Looking again at Formula (2.14) from chapter 2.1.2 we can create a matrix for the neutral state, this time for a hexrotor, who's geometry is very similar to the quadrotor, except it possesses six motors as actuators instead of four:

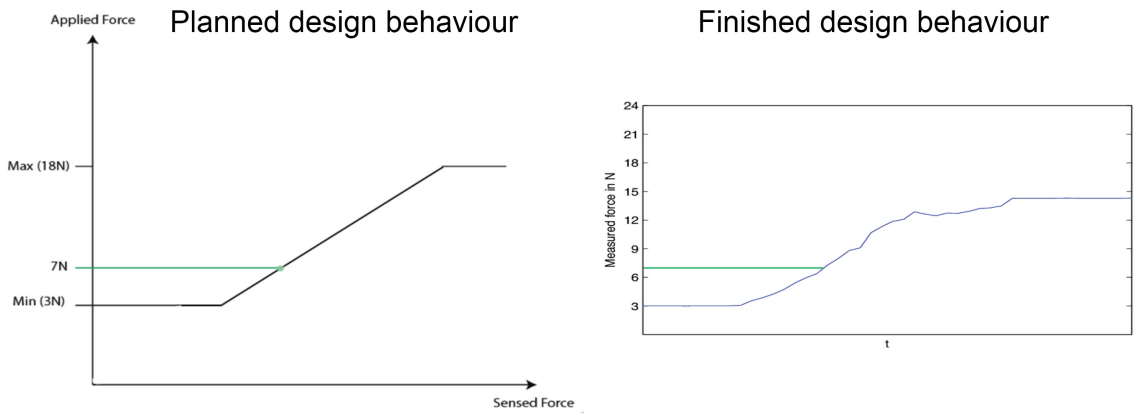

Figure 6.2: Comparison between the initially planned behaviour and the behaviour of the finished force sensing motor mount. Note that the test for the finished design was done by manually pressing the wheel downwards as linear as possible, which explains small deviations from a perfect line in the graph of the finished design.

$$\begin{bmatrix} T_{\sum} \\ \tau_1 \\ \tau_2 \\ \tau_3 \end{bmatrix} = \begin{bmatrix} c_T & c_T & c_T & c_T & c_T & c_T \\ dc_T c(\Phi_1) & dc_T c(\Phi_2) & dc_T c(\Phi_3) & dc_T c(\Phi_4) & dc_T c(\Phi_5) & dc_T c(\Phi_6) \\ dc_T s(\Phi_1) & dc_T s(\Phi_2) & dc_T s(\Phi_3) & dc_T s(\Phi_4) & dc_T s(\Phi_5) & dc_T s(\Phi_6) \\ -c_Q & c_Q & -c_Q & c_Q & -c_Q & c_Q \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \\ \omega_5^2 \\ \omega_6^2 \end{bmatrix}$$

$$\begin{bmatrix} T_{\sum} \\ \tau_1 \\ \tau_2 \\ \tau_3 \end{bmatrix} = \Gamma \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \\ \omega_5^2 \\ \omega_6^2 \end{bmatrix}. \tag{6.3}$$

In the neutral state of the aerial platform, when τ_1, τ_2 and $\tau_3 = 0$, F_{n_c} can be entirely controlled by $T_{\sum}$.

$$F_{n_c} \propto T_{\sum} = mg + \sum_{i=1}^{3} F_{des_i}, \tag{6.4}$$

with F_{des_i} the desired F_n for each wheel (7N). As with Formula 2.14, we invert the constant matrix Γ and solve for ω_i .

This becomes more complicated if either the system is not in the neutral state (τ_1, τ_2, $\tau_3 \neq 0$) when establishing contact with the ceiling, or if the ceiling is not perfectly horizontal, forcing the aerial platform out of its neutral state (τ_1, $\tau_2 \neq 0$).

τ_3 is the current yaw movement of the system. Let's assume that $\tau_3 \approx 0$. This assumption can be made because yaw is normally controlled with the help of the digital compass of the IMU, which gives the absolute position of the earth's magnetic field at all times. Yaw position, and therefore yaw speed, can therefore be corrected most of the time with relative ease. Yaw is also less important when contact is established with the ceiling, because as long as τ_3 is not strong enough to overcome the friction force between ceiling and wheels, the system will not move in direction of τ_3.

Since even a slight imbalance in the distribution of F_{n_c}, when close to the ceiling, leads to a contact of the blades with the ceiling, it is essential that these values are tightly controlled.

To guarantee the equal distribution of force over all wheels, we have to control the distribution of force over the wheels by controlling the roll rate τ_1 and pitch rate τ_2 of the system. To do this we create a force distribution vector $\vec{V}$, with its origin at the center of the aerial system (Fig. 6.3). The direction of this vector is:

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \epsilon_1 \begin{bmatrix} c(\alpha_1) \\ s(\alpha_1) \end{bmatrix} + \epsilon_2 \begin{bmatrix} c(\alpha_2) \\ s(\alpha_2) \end{bmatrix} + \epsilon_3 \begin{bmatrix} c(\alpha_3) \\ s(\alpha_3) \end{bmatrix}, \tag{6.5}$$

with ϵ_i the measured force at the motor mount M_i.

Since the force between the wheels can be distributed evenly with slight adjust-

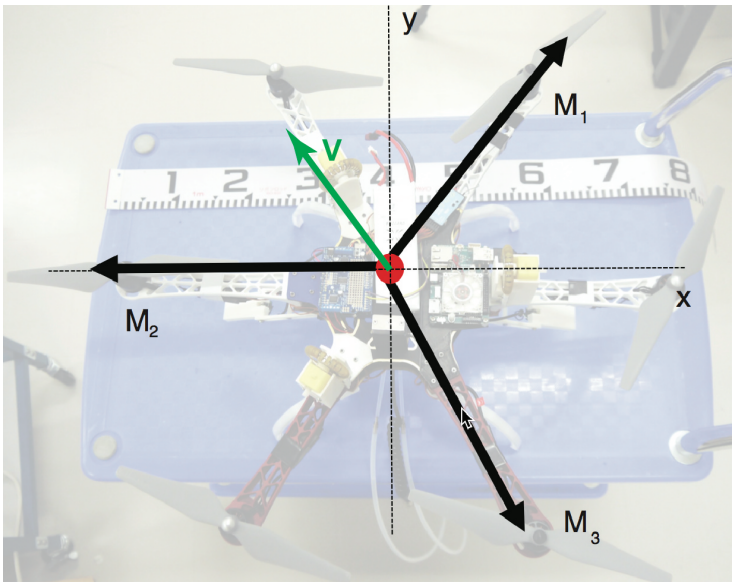

Figure 6.3: Graph showing the force vector (green) required to compensate the force applied to the the motor mounts M_i. The speed of the motor is tuned so that the system rolls and pitches slightly (τ_1, τ_2), reducing $\vec{V}$ to 0.

ments to the roll and pitch of the system:

$$\begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} = k \begin{bmatrix} V_x \\ V_y \end{bmatrix}, \tag{6.6}$$

with k as a certain gain coefficient.

We can now control the force between each wheel individually by controlling the individual motor speed of each wheel, as shown in Formula (6.3) and:

$$T_{\sum} = mg + \sum_{i=1}^{3} F_{des_i} \tag{6.7}$$

$$\begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} = k \left(\epsilon_1 \begin{bmatrix} c(\alpha_1) \\ s(\alpha_1) \end{bmatrix} + \epsilon_2 \begin{bmatrix} c(\alpha_2) \\ s(\alpha_2) \end{bmatrix} + \epsilon_3 \begin{bmatrix} c(\alpha_3) \\ s(\alpha_3) \end{bmatrix} \right), \epsilon_i = F_{des_i} - F_i \tag{6.8}$$

and again, we invert Γ and solve for ω_i.

Chapter 7

Experiments

This chapter describes the experiments conducted with the aerial robot prototypes. Since there have been more than one model of prototypes with different characteristics and capabilities, the fist section of this chapter, as well as chapter 5, has been divided into three different sections, each corresponding to one of the three versions described in chapter 6 respectively.

7.1 Docking ,Driving and Disengage

7.1.1 Version 1

In the very first experiments, we tested the feasibility of closely approaching the ceiling with the multirotor. After setting up the upwards-directed drive system in the hexrotor system, the system was tested without dual camera system on a roughly structured ceiling of 3m height (see Fig. 7.1). These first tests of docking, driving and disengage from the ceiling showed that the docking and disengage from the ceiling could be done consistently without damage to the system. The tests also showed that driving on the ceiling is possible and that it is possible to precisely maneuver a multirotor at high altitudes.

After testing the feasibility of the dual camera system (Chapter 4.2.1) and testing the performance of the AR marker based localization system (Chapter 4.2.2) the

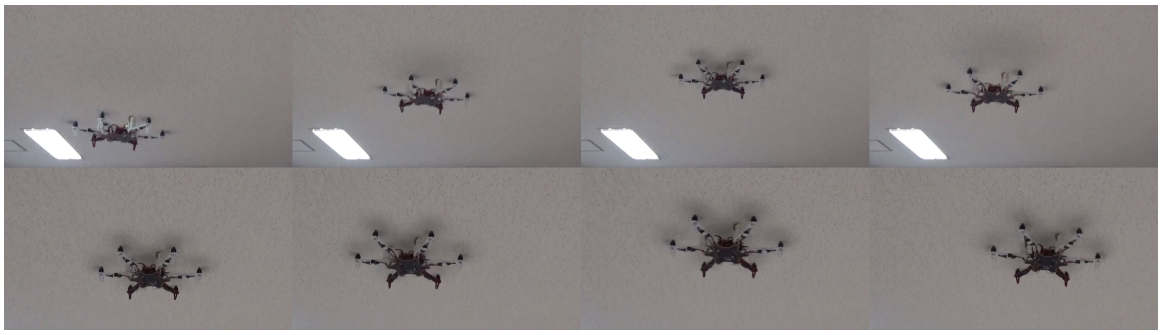

Figure 7.1: Video-sequence of an early ceiling docking, driving and disengage test. The test was performed on a 3m high ceiling. In the upper row, ceiling engage and disengage is shown. In the lower row, an open loop drive sequence driven by the Atmega micro controller on board is tested, where the system drives in a 30cm diameter circle. (`https://youtu.be/gEDIJqR9pS8`)

camera system was mounted on the hexrotor platform and a visual servoing task was tested with the hexrotor on the ground (see Fig. 7.2). The hexrotor is placed upside-down on the ground and a AR marker is held above the system to be visible by at least one of the cameras. The marker is then moved by hand with several quick motions to see if the system is able to follow the marker. As seen in the provided video (linked in Fig. 7.2), the system is able to quickly follow the marker and compensate for eventual overshoot due to quick movement. The experiment showed, that the dual camera marker system processed by the on-board mounted ARM system is capable of performing a visual servoing task with a moving AR marker target. It furthermore showed that the weight of the hexrotor itself (2.1kg) provides the omni-wheels with enough grip while simultaneously not creating too much friction on the axle to provide smooth motion. Hence the $\approx (2.1 \cdot 9.8)/3$N chosen as desired force value per wheel on the ceiling in Chapter 6.

An artificial room was constructed to securely test autonomous movement of the hexrotor system. The artificial room consists of a metal fame with the dimensions of 3×3×2m (w×l×h). The artificial ceiling was built out of 1.5×0.9m chipboard tiles. Manual approach and disengage were successfully tested even with relatively heavy payloads, like the paint marking device used in chapter 7.2 (See Fig. 7.3).

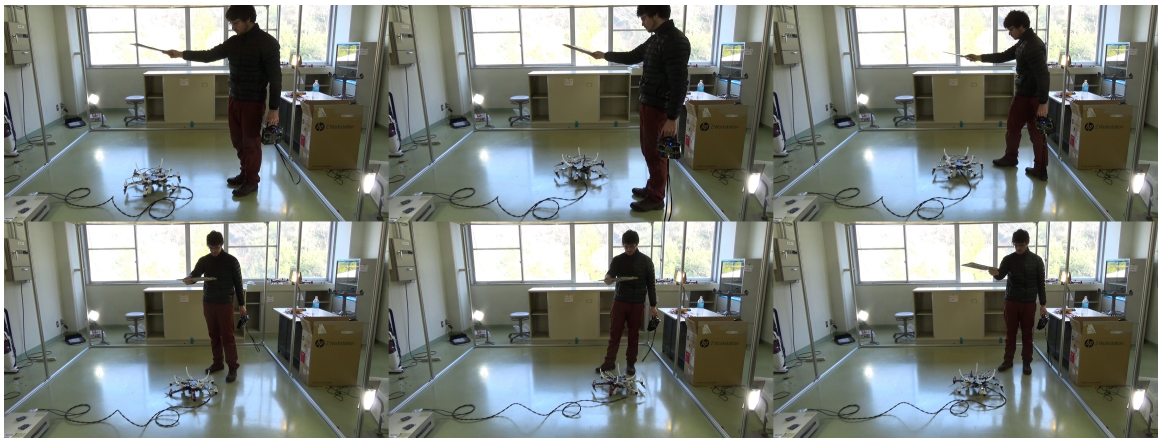

Figure 7.2: Video-sequence of testing of the dual camera system mounted on the hexrotor with a visual servoing task. The system successfully follows the marker held above it and correctly centers directly under the marker, even if it's held at an angle. (`https://youtu.be/Rp3i9mW-yBE`)

The image servoing of the hexrotor system on the ceiling was tested while tethered to a power supply as well as an ink-tube for the print head and its power and control cable. An AR marker is placed on the floor in the artificial room setup. Approach and contact with the ceiling is controlled by a remote operator. After a short hovering phase in which the balance of the system is visually confirmed, the operator carefully increases the thrust until contact with the ceiling is established. The thrust is then further increased by around 10% total thrust. When the propeller continues to push the hexrotor against the ceiling, its position on the ceiling plane does not change until the omni-wheel drive is utilized. We are able to manipulate its position either manually by using dedicated joysticks mapped to $\begin{bmatrix} x_d & y_d & \omega \end{bmatrix}^T$ or, if at least one camera of the dual camera system has successfully recognized the AR-marker placed on the floor, the hexrotor autonomously moves to the desired position O_des within the range of the combined camera angle. In the case shown here, the desired position is directly over the marker.The image processing and signal generation is handled on board by the ARM computing unit. The operator is then able to visually confirm the correct position and set an ink-marker (result of this experiment is presented in chapter 7.2). For disengaging from the ceiling the control is switched back to manual

Figure 7.3: Video-sequence of testing ceiling engage, manual movement and disengage with paint marking payload. (`https://youtu.be/BVQZBh6Zw04`)

and the thrust slowly lowered. After lowering the thrust by around 30%, the hexrotor disengages from the ceiling and can manually be steered to a safe landing spot. In Fig. 7.4 a video-sequence of this process can be seen.

Figure 7.4: Video-sequence of testing the authonomous tracking of a target on the ceiling. (`https://youtu.be/FwOYFaVgGUI`)

Conclusion Version 1

While overall the experiments for Version 1 were successful, it is visible in Fig. 7.4 is that the hexrotor does not always successfully position itself completely over the

target. The reason for this is the fact that there is no feedback loop from regarding the wheel speed. The controller still provides the wheel motors with voltage equal to V described in formula 5.8, but since there is no way to control the normal force, that the hexrotor is applying on the ceiling in this version of the prototype, the friction on the axle is too high and the motor stalls when a certain low amount of V is provided. This hinders the goal of small, precise movements on the ceiling, which was set as a goal at the beginning of this work and is addressed in Version 2 of the prototype by using force sensing actuators and a feedback loop to measure the current speed of the motors.

Another observation made in these experiments was that due to the low pressure area generated between the propeller and ceiling, there is significantly less thrust necessary to keep the multirotor engaged with the ceiling than keeping it in a hover position. While this phenomenon is not part of this work, it could potentially be used to lower energy consumption when engaged with the ceiling and lead to longer operation times of similar systems.

7.1.2 Version 2

Experiments with the Version 2 of the prototype focus on the force sensing capabilities of the holonomic drive system and the ability to control the force as described in Chapter 6.

To safely and repeatably test this, the hexrotor was attached to a mount, that is extendable in the z-direction. The mount rests on the hexrotor at a height of 1.7m. While the mount consists of a metal bar and a heavy metal stand to provide stability against tipping, the part that extends and the part that the hexrotor is mounted on are made out of abs plastic and are relatively light. The extendable plastic bar adds 130g to the system setup. A picture of the setup can be seen in Fig. 7.5.

An experiment was conducted to confirm the correct implementation and feasibility of the force control methods described in Chapter 6. In contrast to Version 1, where the engagement with the ceiling was done manually and then the operator needed to switch to the autonomous processing via a switch on the remote, the whole

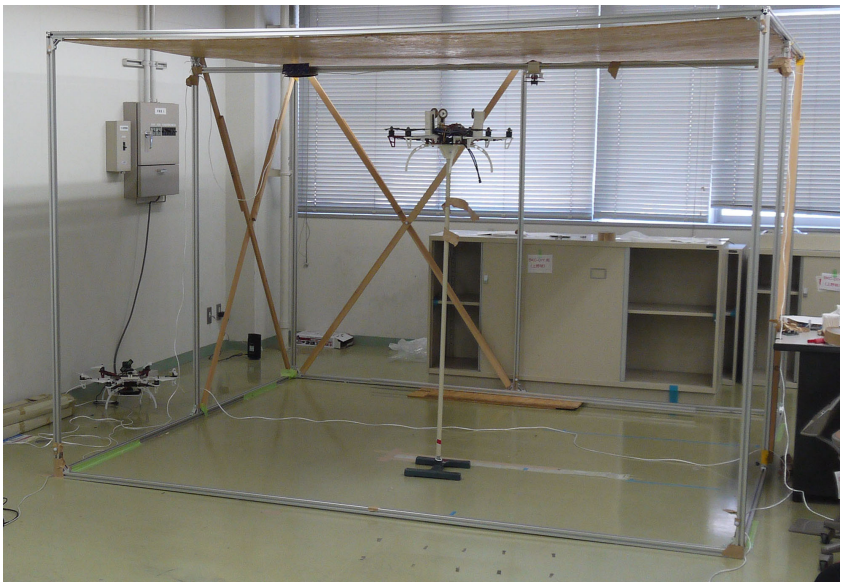

Figure 7.5: Setup of experiments performed with Version 2 of the prototype.

process of ceiling engagement, pushing against the ceiling with a certain force and disengage are done autonomously now. The ARM system is running a node of the robotics middleware called Robot Operating System (ROS) (see Fig. 5.7). It broadcasts via WIFI from the ARM processing board and can receive parameters such as desired engagement length and desired force towards the ceiling and broadcasts the raw and processed data received from the force sensors. One can interface with this node from a nearby base station, also running ROS. All processing for force control is done onboard the hexrotor's ARM processing unit.

To test the implementation of the force controller, the following sequence (Algorithm 3) was broadcast via ROS towards the system.

A video-sequence of the experiment can be seen in Fig. 7.6.

The experiment has been performed with two different implementations. One implementation is using the force sensing data received by the actuators and controlling the normal force F_n by regulating the thrust of the system $T_{\sum}$ (as described in formula 6.4). The graph for this implementation can be seen in Fig. 7.7. Another implementation is using the method described in formula 6.7 and formula 6.8. Here, the system tries to distribute the forces acting on the actuators equally, by slightly

```
start;
if impact with ceiling then
    for x=1 to 5 do
        while time < x*20sek do
            desired force on wheel1 to wheel3: x+4N;
        end
    end
end
land;
```

Algorithm 3: Experiment sequence

tilting the hexrotor by tuning the parameters τ_1 and τ_2 of formula 6.3.

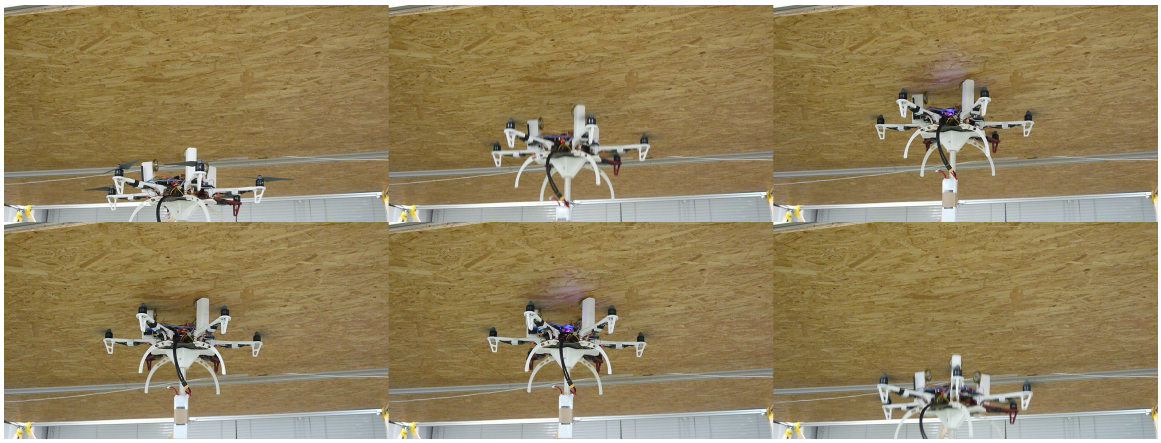

Figure 7.6: Video-sequence of testing the autonomous approach towards the ceiling as well as force application. (`https://youtu.be/4X1wQ4zUTVQ`)

Conclusion Version 2

The equal distribution of forces across all actuators, has been successfully implemented and tested in this version of the prototype. This enables equal and well adjusted force contact with the ceiling, which should lead to better performance when moving on the ceiling with the proposed top mounted omni-wheel drive system.

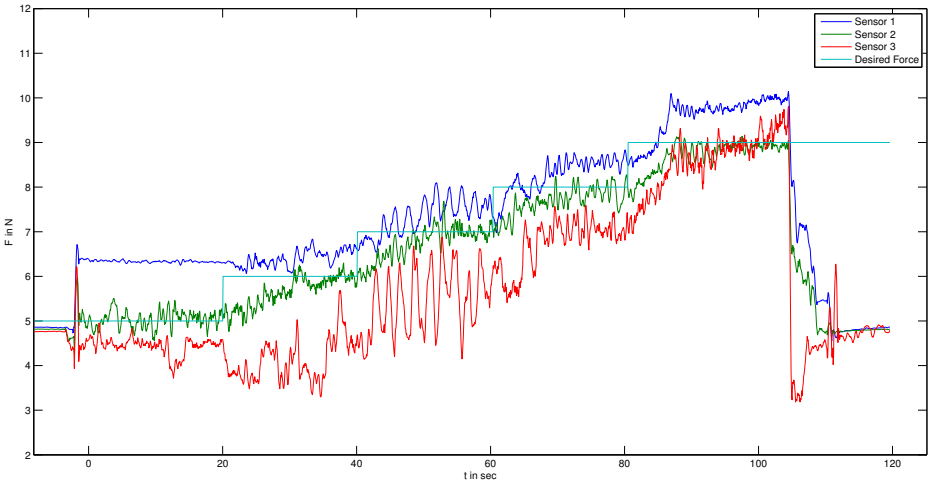

Figure 7.7: Graph of testing algorithm 3 with P control and controlling F_n over $T_{\sum}$.

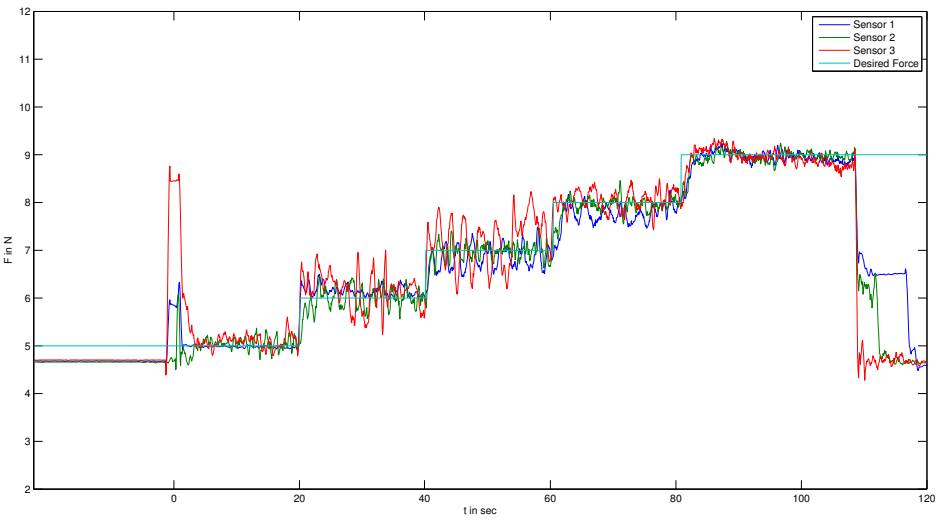

Figure 7.8: Graph of testing algorithm 3 with PID control and controlling F_n over $T_{\sum}$, τ_1 and τ_2.

One observation that was made in experiments in Version 1 was that a negative pressure area seemed to form between the ceiling and aerial platform which might have positive effects on the overall performance of the system (see chapter 7.1.1). Unfortunately, due to the significantly larger gap between the propeller and ceiling in Version 2, this behaviour was not visible anymore. The decision was made to slightly redesign the motor mount and position of the electronics to reduce the gap between propeller and ceiling when engaged with the ceiling to reestablish this behaviour.

7.1.3 Version 3

Due to the lack of a dedicated room to perform a localization and movement experiment, a provisional experimental setup is done. An artificial ceiling is created using stacked tables. To ensure safety during the experiment and prevent damage to the system, the operational height is relatively low at 50cm. Since ground effects only come into play when a rotor of a VTOL is less than a propeller diameter away form the ground, they should not play a role in evaluating the movement performance on the ceiling.

An AR marker is placed on the ground to be tracked by the dual camera setup. The system then receives commands to move towards certain positions relative to the AR marker.

Ground truth is measured by a Vicon system that has been set up to be able to perform measuring under the table surface. Reflective markers are attached on the system as a tracking marker for the motion capture system. The measurements of this system are processed on a different system and are neglected when calculating the current position of the hexrotor.

Conclusion Version 3

Unfortunately the experiment could not be performed yet due to unforeseen problems regarding the setup of the above mentioned experiment. The final experiment and results will be performed as soon as possible and will be added in the final version of

this work.

7.2 Marking

An experiment was performed to test the feasibility of using the system to print markers on the ceiling with the industrial grade print head seen in Fig. 7.9. The test was performed outside within a frame of 5×5m and a ceiling height of 2m (Fig. 7.10). This setup was chosen to simulate a deployment of the system in a inspection task environment. Similar environments can be seen e.g. in maintenance areas under bridges (see Fig. 7.11).

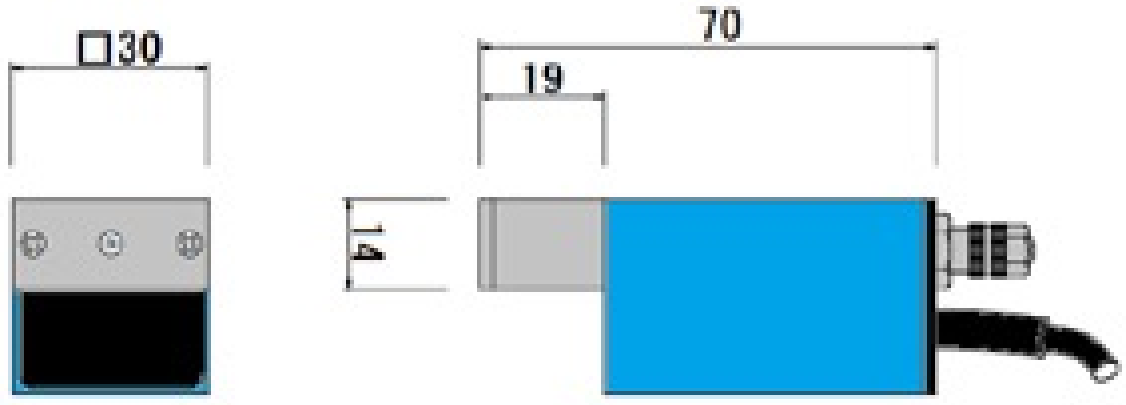

Figure 7.9: The industrial grade print head module (REA-JET DS) used in all marking experiments.

Version 1 of the system was used at the time of the experiment. This means the tracking system was not designed to provide reliable go-to tasks on the ceiling. To test the possibility of putting markers and certain shapes on the ceiling, an open loop sequence was implemented into the Atmega micro controller. This sequence let the system drive a certain time frame to its right, front, left then back while deploying paint with the print head; forming a square on the ceiling.

A video sequence of this process can be found in Fig. 7.12 with a photo of the finished print produced on the ceiling.

Figure 7.10: The place where the marking experiment was performed.

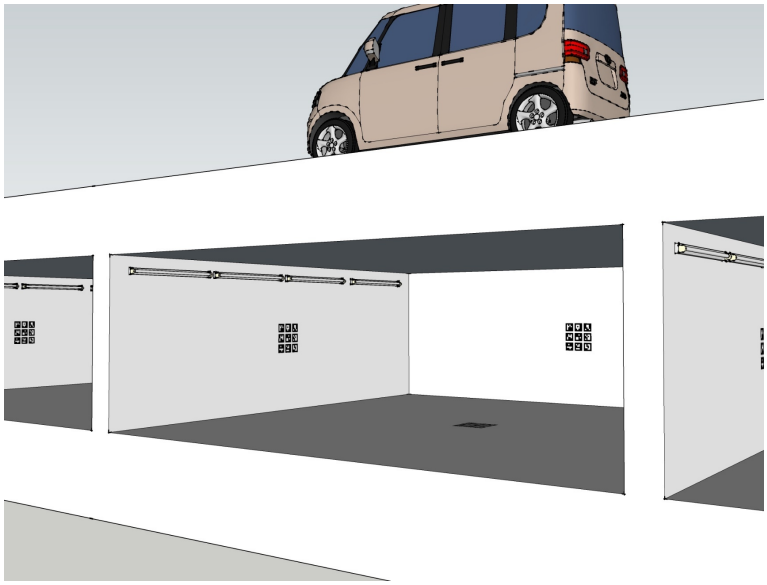

Figure 7.11: A potential deployment area targeted by the introduced system: Maintenance areas under bridges. The system could be used to quickly and precisely print markers on the ceiling to mark drill and inspection positions for maintenance worker.

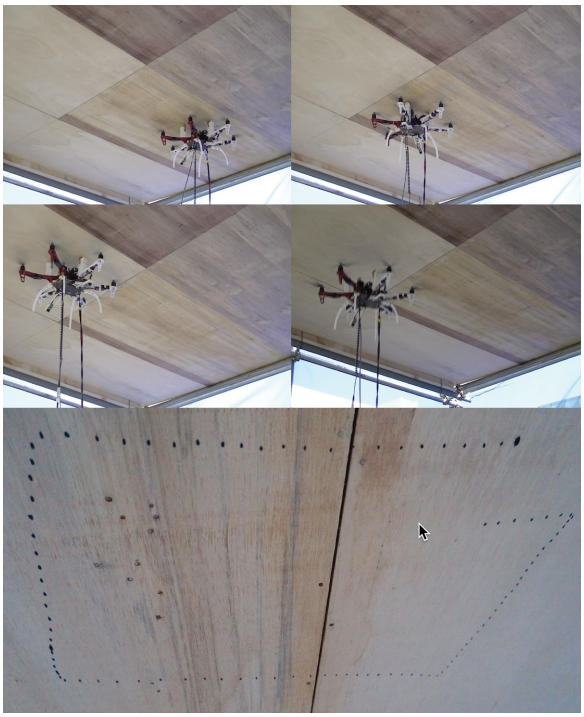

Figure 7.12: (Up) Video sequence showing the use of an industrial print head on the ceiling. The result can be seen in the lower picture.

Better results were achieved when the dual camera system was operational and used. In tests with the Version 1 of the system (described at the end of chapter 7.1.1) we were able to successfully put ink markers scattered within a circle of 1cm diameter (see Fig. 7.13). However, these results were not perfectly consistent due to the tendency of the motors driving the omni-wheel drive system to stall. This sometimes resulted in a failure to reach the target destination.

Conclusion Marking

In conclusion the marking experiment can be considered a success. It was successfully shown that a marker painting task on the ceiling, performed with a multirotor is feasible.

During the testing several problems became apparent. As seen in Fig. 7.12 without proper tracking, it is unlikely to move on the ceiling with a high degree of

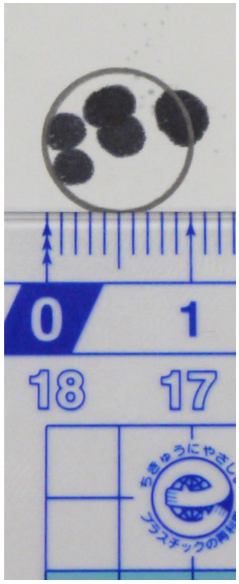

Figure 7.13: Result of the experiment described at the end of chapter 7.1.1. Several ink markings are put on a sheet of paper in 2m height.

precision due to frequent slipping when using a omni wheel drive system.

7.3 Tracking

To test the performance of the tracking system, Version 1 of the system is put upside down on the ground, with the cameras directed upwards. An AR marker is then placed directly above the it, acting as the starting position.

Due to lack a Vicom system, the ground truth had to be measured by hand. The command is given via a base station to move the desired position by 0.1m from the starting point in one direction. After the system completes its visual servoing movement and comes to a rest, the ground truth position of the system is measured. This is repeated 10 times. The results of these two measurements can be seen in Fig. 7.14

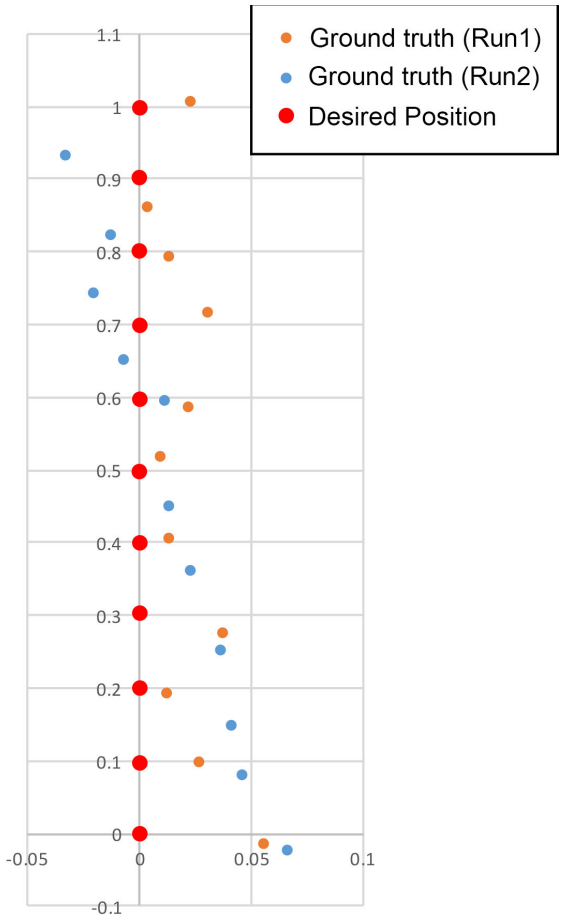

Figure 7.14: The result of the experiment described in this chapter (measurements in m)

Conclusion Tracking

From the Figure it is visible that during run one and run two the desired positions could not be reached with a high degree of accuracy. The tracking and localization in general is working, but since there is a deviation of up to 0.07m from the desired target point, the accuracy was deemed as not sufficient. While the expectation was to be able to move to a position at least within 0.02m diameter, experiments like this showed that Version 1 of the system was not able to perform visual servoing tasks sufficiently well. One problem that was observed was the stalling of the DC motors when moving at low speed. This was the deciding factor behind completely renewing

the drive system for Version 2.

Another observation was that the measured AR marker position deteriorates when the marker comes closer to the edge of the image frame. This could be explained by distortion due to the lens. Another reason might have been insufficient or faulty calibration and thus faulty intrinsic values. This was the reason why a Kalman filter see chapter 4.2.2 was implemented to create a certain degree of redundancy and thus improve the overall performance of the system.

7.4 Drilling

As a substitute for the print head that became unavailable, a drill tool was mounted on the hexrotor.

The tool was built from spare parts taken from an off-the-shelf dremel fit into a CAD designed and MDF 3d printed mount (see Fig. 7.15).

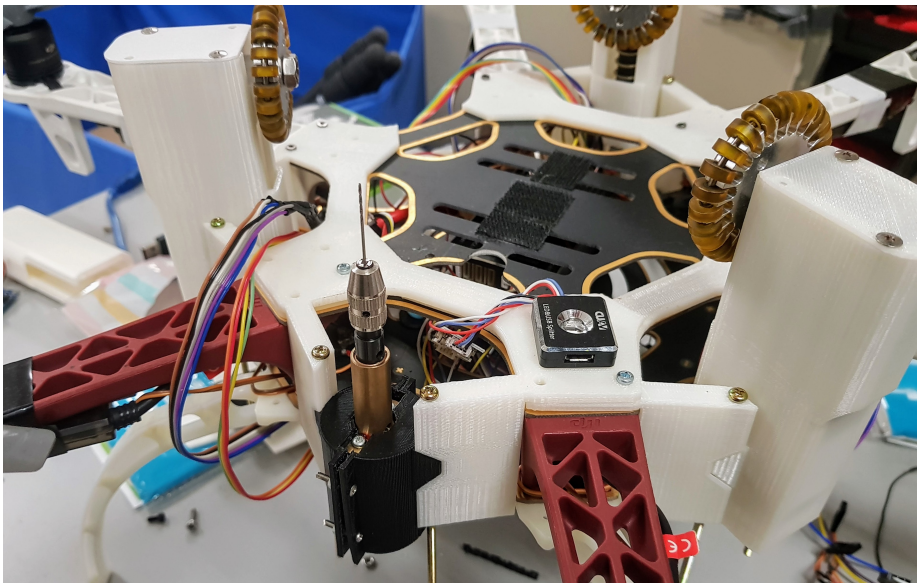

Figure 7.15: Drill tool mounted to Version 3 of the system.

Chapter 8

Conclusions

8.1 Future Work

While the system in itself is operational and provided good results, there is still a manifold of potential projects that can be done in future work.

One of the most interesting observations was that the system tended to require less thrust when close to the ceiling when compared to normally hovering. While this can be potentially explained with a low pressure area that is created between propellers and ceilings, there is no detailed literature analyzing this phenomenon for multirotors. While the "ground effect" has been analyzed in detail by a manifold of literature, the "ceiling effect" that undeniably exists as show in experiments performed in this work and e.g. by Powers *et al.* [87] still requires detailed analysis. While for this work, a detailed analysis has been out of our scope, doing an analysis of this phenomenon with an additional mapping of the air flow could benefit future projects and multirotor design in general.

Another issue that became apparent when working on this project is the issue of a better human-robot interface for aerial robots. Even with the remote controller especially customized only for this project, operating the aerial robot proved to be very complex. If a system like this should be deployed in an actual industrial environment, a better, simpler human-robot interface would be necessary. Since for this project AR markers are already deployed, an augmented reality application for a mobile de-

vice where the operator could point on a certain area of the ceiling, followed by a go-to task of the system would be a human robotic interface that is implementable with toady's technology. Another big potential is the use of VR and its tracking capabilities to give the operator very fine control over the system when working on the ceiling. This is an area that still needs to be explored in the area of robotics in general and promises great potential for scientific contributions.

While the marker-based approach has great benefits regarding setup time and computational costs, a big issue arises when the cameras lose track of the marker. One potential solution is the implementation of an active camera control as done in a previous project [76]. This active camera could systematically hunt for a marker when tracking is lost, making the complete loss of a localization feature point unlikely. A potential application would be freely moving within a complex indoor environment, provided enough identifiable markers are present within visual distance in this environment. The AR marker system used in this work is able to identify and differentiate markers, making the system able to know its exact position within the indoor environment a priori map knowledge, by only receiving a single image of the environment that has a marker present.

8.2 Summary

In this work the gradual development of an aerial robotic system able to reliably drive on the ceiling has been shown. The work has been put in context by exploring current and potential industrial applications of aerial robotics and multirotor technology. A force-sensing, top-mounted omni-wheel drive system fit for a multirotor system has been designed and tested. The system has been made made manually operable via a custom remote control as well as interfaceable via the standard Robotic Operating System. A suitable method for on-board indoor localization has been explored and a dual camera AR marker based system has been implemented and successfully tested in various experiments. In experiments, the potential of this system for real life industrial tasks such as inspection and marker printing on the ceiling have been

explored and successfully shown.

References

[1] R. R. Murphy, "Trial by fire," *Robotics & Automation Magazine, IEEE*, vol. 11, no. 3, pp. 50–61, 2004.

[2] E. Guizzo, "Fukushima robot operator writes tell-all blog," *IEEE Spectrum*, 2011.

[3] "The first 1,000 commercial uas exemptions," Association for unmanned vehicle systems international, 2017.

[4] R. Buechi, *Fascination quadrocopter*. BoD–Books on Demand, 2011.

[5] R. R. Murphy, *Disaster robotics*. MIT Press, 2014.

[6] M. Bernard and K. Kondak, "Generic slung load transportation system using small size helicopters," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, 2009, pp. 3258–3264.

[7] D. Mellinger, Q. Lindsey, M. Shomin, and V. Kumar, "Design, modeling, estimation and control for aerial grasping and manipulation," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*. IEEE, 2011, pp. 2668–2673.

[8] N. Michael, J. Fink, and V. Kumar, "Cooperative manipulation and transportation with aerial robots," *Autonomous Robots*, vol. 30, no. 1, pp. 73–86, 2011.

[9] P. E. Pounds, D. R. Bersak, and A. M. Dollar, "Grasping from the air: Hovering capture and load stability," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2491–2498.

[10] C. M. Korpela, T. W. Danko, and P. Y. Oh, "Mm-uav: Mobile manipulating unmanned aerial vehicle," *Journal of Intelligent & Robotic Systems*, vol. 65, no. 1, pp. 93–101, 2012.

[11] E. W. Hawkes, D. L. Christensen, E. V. Eason, M. A. Estrada, M. Heverly, E. Hilgemann, H. Jiang, M. T. Pope, A. Parness, and M. R. Cutkosky, "Dynamic surface grasping with directional adhesion," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on.* IEEE, 2013, pp. 5487–5493.

[12] F. Huber, K. Kondak, K. Krieger, D. Sommer, M. Schwarzbach, M. Laiacker, I. Kossyk, S. Parusel, S. Haddadin, and A. Albu-Schäffer, "First analysis and experiments in aerial manipulation using fully actuated redundant robot arm," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on.* IEEE, 2013, pp. 3452–3457.

[13] F. Ruggiero, M. A. Trujillo, R. Cano, H. Ascorbe, A. Viguria, C. Peréz, V. Lippiello, A. Ollero, and B. Siciliano, "A multilayer control for multirotor uavs equipped with a servo robot arm," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on.* IEEE, 2015, pp. 4014–4020.

[14] J. D. Tedford, "Developments in robot grippers for soft fruit packing in new zealand," *Robotica*, vol. 8, no. 04, pp. 279–283, 1990.

[15] O. Unver and M. Sitti, "A miniature ceiling walking robot with flat tacky elastomeric footpads," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on.* IEEE, 2009, pp. 2276–2281.

[16] A. Briod, P. Kornatowski, J.-C. Zufferey, and D. Floreano, "A collision-resilient flying robot," *Journal of Field Robotics*, vol. 31, no. 4, pp. 496–509, 2014.

[17] K. Kawasaki, Y. Motegi, M. Zhao, K. Okada, and M. Inaba, "Dual connected bi-copter with new wall trace locomotion feasibility that can fly at arbitrary tilt

angle," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on.* IEEE, 2015, pp. 524–531.

[18] A. Jimenez-Cano, J. Braga, G. Heredia, and A. Ollero, "Aerial manipulator for structure inspection by contact from the underside," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on.* IEEE, 2015, pp. 1879–1884.

[19] S. Kim, S. Choi, and H. Kim, "Aerial manipulation using a quadrotor with a two dof robotic arm," *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference*, pp. 4990–4995, 2013.

[20] S. D. Lucia, G. D. Tipaldi, and W. Burgard, "Attitude stabilization control of an aerial manipulator using a quaternion-based backstepping approach," *Mobile Robots (ECMR), 2015 European Conference*, pp. 1–6, 2015.

[21] M. Orsag, C. Korpela, M. Pekala, and P. Oh, "Stability control in aerial manipulation," *American Control Conference (ACC)*, pp. 5581–5586, 2013.

[22] J. Thomas, G. Loianno, K. Sreenath, and V. Kumar, "Toward image based visual servoing for aerial grasping and perching," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on.* IEEE, 2014, pp. 2113–2118.

[23] H. Jiang, M. T. Pope, E. W. Hawkes, D. L. Christensen, M. A. Estrada, A. Parlier, R. Tran, and M. R. Cutkosky, "Modeling the dynamics of perching with opposed-grip mechanisms," in *2014 IEEE International Conference on Robotics and Automation (ICRA).* IEEE, 2014, pp. 3102–3108.

[24] A. Fukushima and Y. Kawaguchi, "Insect leg inspired friction attachment for miniature quadcopter," in *Proceedings of the 12th International Conference on Advances in Computer Entertainment Technology.* ACM, 2015, p. 34.

[25] K. Mohta, V. Kumar, and K. Daniilidis, "Vision-based control of a quadrotor for perching on lines," in *2014 IEEE International Conference on Robotics and Automation (ICRA).* IEEE, 2014, pp. 3130–3136.

[26] J. Thomas, G. Loianno, K. Daniilidis, and V. Kumar, "Visual servoing of quadrotors for perching by hanging from cylindrical objects," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 57–64, 2016.

[27] R. W. Prouty, *Helicopter performance, stability, and control.* Krieger Publishing Company, 1995.

[28] G. J. Leishman, *Principles of helicopter aerodynamics with CD extra.* Cambridge university press, 2006.

[29] R. Mahony, V. Kumar, and P. Corke, "Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor," *Robotics & Automation Magazine, IEEE*, vol. 19, no. 3, pp. 20–32, 2012.

[30] D. Mondey, *The illustrated encyclopedia of aircraft.* Hamlyn, 1978.

[31] T. Wilson and P. Trickey, "Dc machine with solid-state commutation," *Electrical Engineering*, vol. 81, no. 11, pp. 879–884, 1962.

[32] D. A. MacKenzie, *Inventing accuracy: A historical sociology of nuclear missile guidance.* MIT press, 1993.

[33] http://usdynamicscorp.com/literature/general/precision_instruments.pdf, last visited 2017-08-28.

[34] http://www.3dheliupgrades.com/our-blog, last visited 2017-08-28.

[35] C. Acar and A. Shkel, *MEMS vibratory gyroscopes: structural approaches to improve robustness.* Springer Science & Business Media, 2008.

[36] E. Kaplan and C. Hegarty, *Understanding GPS: principles and applications.* Artech house, 2005.

[37] S. Pace, G. Frost, I. Lachow, D. Frelinger, D. Fossum, D. K. Wassem, and M. Pinto, "The global positioning system, chapter gps history, chronology and budgets," *RAND Coorporation*, pp. 237–270, 1995.

[38] M. S. Grewal, L. R. Weill, and A. P. Andrews, *Global positioning systems, inertial navigation, and integration.* John Wiley & Sons, 2007.

[39] https://www.faa.gov/about/office_org/headquarters_offices/ato/service_units/techops/navservices/gnss/gps/policy/presidential/#1, last visited 2017-08-28.

[40] *Frequently Asked Questions About Selective Availability.* National Coordination Office for Space-Based Positioning, Navigation, and Timing (PNT), 2001.

[41] "Russia to launch glonass satellite on feb. 24," 2011.

[42] L. Pearson, "Developing the flying bomb," *Naval Aviation in World War I*, pp. 70–73, 1969.

[43] L. R. Newcome, *Unmanned aviation: a brief history of unmanned aerial vehicles.* Aiaa, 2004.

[44] F. I. Ordway, *The rocket team.* Mit Press, 1979.

[45] P. I. Casey, *Apollo: A Decade of Achievement*, A. M. Dorsey, Ed. JS Blume Publishing, 2013.

[46] S. Ackerman, "41 men targeted but 1,147 people killed: Us drone strikes – the facts on the ground," *the Guardian*, 2014.

[47] J. Cavallaro, S. Sonnenberg, S. Knuckey *et al.*, "Living under drones: Death, injury and trauma to civilians from us drone practices in pakistan," *International Human Rights and Conflict Resolution Clinic at Stanford Law School and Global Justice Clinic at NYU School of Law*, pp. 1–165, 2012.

[48] MSNBC, "Pakistan says u.s. drones in its air space will be shot down," 12 2011.

[49] "Movie screening: 'unmanned: American drone wars'," The Express Tribune, 10 2013.

[50] G. Golightly, "Boeing's concept exploration pioneers new uav development with the hummingbird and the maverick," *Boeing Frontiers*, vol. 3, no. 8, 2004.

[51] A. Chuter, *Saab, Swiss UAV Team on Rotorcraft.* Defense News, 2009.

[52] B. Ghosh and M. Thompson, "The cia's silent war in pakistan," *TIME Magazine*, pp. 40–41, 2009.

[53] D. Schlüter, *Die Geschichte des Modellhubschraubers und andere Erinnerungen.* Neckar-Verlag, 2007.

[54] T. Krajník, V. Vonásek, D. Fišer, and J. Faigl, "Ar-drone as a platform for robotic research and education," *Research and Education in Robotics-EUROBOT 2011*, pp. 172–186, 2011.

[55] M. Saska, T. Krajník, J. Faigl, V. Vonásek, and L. Přeučil, "Low cost mav platform ar-drone in experimental verifications of methods for vision based autonomous navigation," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on.* IEEE, 2012, pp. 4808–4809.

[56] A. Hernandez, C. Copot, R. De Keyser, T. Vlas, and I. Nascu, "Identification and path following control of an ar. drone quadrotor," in *System Theory, Control and Computing (ICSTCC), 2013 17th International Conference.* IEEE, 2013, pp. 583–588.

[57] A. Fitzpatrick, "Finally, a drone you can own," *Time*, 2014.

[58] "Dji's biggest competition in drones is itself," The Verge, 2016.

[59] B. Canis, *Unmanned aircraft systems (UAS): Commercial outlook for a new industry.* Congressional Research Service Washington, 2015.

[60] M. Redbond, "Robots-the future of agriculture," *International Pest Control*, vol. 57, no. 6, p. 314, 2015.

[61] J. Casper and R. R. Murphy, "Human-robot interactions during the robot-assisted urban search and rescue response at the World Trade Center," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 33, no. 3, pp. 367–385, 2003.

[62] K. Nagatani, S. Kiribayashi, Y. Okada, K. Otake, K. Yoshida, S. Tadokoro, T. Nishimura, T. Yoshida, E. Koyanagi, M. Fukushima *et al.*, "Emergency response to the nuclear accident at the Fukushima Daiichi nuclear power plants using mobile rescue robots," *Journal of Field Robotics*, vol. 30, no. 1, pp. 44–63, 2013.

[63] S. Kawatsuma, M. Fukushima, and T. Okada, "Emergency response by robots to fukushima-daiichi accident: summary and lessons learned," *Industrial Robot: An International Journal*, vol. 39, no. 5, pp. 428–435, 2012.

[64] Y.-W. Huang, Y. Sasaki, Y. Harakawa, E. F. Fukushima, and S. Hirose, "Operation of underwater rescue robot Anchor Diver III during the 2011 Tohoku earthquake and tsunami," in *OCEANS 2011*. IEEE, 2011, pp. 1–6.

[65] Q. Feng, J. Liu, and J. Gong, "Urban flood mapping based on unmanned aerial vehicle remote sensing and random forest classifier—a case of yuyao, china," *Water*, vol. 7, no. 4, pp. 1437–1455, 2015.

[66] https://www.dji.com/guidance, last visited 2017-08-28.

[67] A. L. Salih, M. Moghavvemi, H. A. Mohamed, and K. S. Gaeid, "Flight pid controller design for a uav quadrotor," *Scientific Research and Essays*, vol. 5, no. 23, pp. 3660–3667, 2010.

[68] S. Kurnaz, O. Cetin, and O. Kaynak, "Adaptive neuro-fuzzy inference system based autonomous flight control of unmanned air vehicles," *Expert Systems with Applications*, vol. 37, no. 2, pp. 1229–1234, 2010.

[69] http://ardupilot.org/ardupilot/, last visited 2017-08-28.

[70] L. Meier, D. Honegger, and M. Pollefeys, "Px4: A node-based multithreaded open source robotics framework for deeply embedded platforms," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 6235–6240.

[71] L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, "Pixhawk: A system for autonomous flight using onboard computer vision," in *Robotics and automation (ICRA), 2011 IEEE international conference on.* IEEE, 2011, pp. 2992–2997.

[72] http://www.chinadaily.com.cn/business/tech/2017-04/25/content_29068931.htm, last visited 2017-08-28.

[73] https://www.rcgroups.com/forums/showthread.php?1676150-DJI-ZeroUAV-hardware-3-in-1-technical-review-teardown, last visited 2017-09-24.

[74] https://www.suasnews.com/2017/08/us-army-calls-units-discontinue-use-dji-equipment/, last visited 2017-08-28.

[75] J. Trein, A. T. Schwarzbacher, and B. Hoppe, "FPGA implementation of a single pass real-time blob analysis using run length encoding," in *MPC-Workshop, February*, 2008.

[76] R. Ladig and K. Shimonomura, "Fpga-based fast response image analysis for autonomous or semi-autonomous indoor flight," *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 682–687, 2014.

[77] S. Tagzout, K. Achour, and O. Djekoune, "Hough transform algorithm for fpga implementation," *Signal Processing Systems, 2000. SiPS 2000. 2000 IEEE Workshop*, pp. 384–393, 2000.

[78] H. Koshimizu and M. Numada, "Fiht2 algorithm: a fast incremental hough transform," *IEICE TRANSACTIONS on Information and Systems*, pp. 3389–3393, 1992.

[79] J. Stowers, M. Hayes, and A. Bainbridge-Smith, "Altitude control of a quadrotor helicopter using depth map from microsoft kinect sensor," in *Mechatronics (ICM), 2011 IEEE International Conference on.* IEEE, 2011, pp. 358–362.

[80] M. Windolf, N. Götzen, and M. Morlock, "Systematic accuracy and precision analysis of video motion capturing systems—exemplified on the vicon-460 system," *Journal of biomechanics*, vol. 41, no. 12, pp. 2776–2780, 2008.

[81] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014.

[82] S. Garrido-Jurado, R. Muñoz-Salinas, F. Madrid-Cuevas, and R. Medina-Carnicer, "Generation of fiducial marker dictionaries using mixed integer linear programming," *Pattern Recognition*, vol. 51, pp. 481–491, 2016.

[83] https://www.uco.es/investiga/grupos/ava/node/26, last visited 2017-08-28.

[84] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE transactions on systems, man, and cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[85] P. Pounds, R. Mahony, and P. Corke, "Modelling and control of a quad-rotor robot," in *Proceedings Australasian Conference on Robotics and Automation 2006*. Australian Robotics and Automation Association Inc., 2006.

[86] "Add-on joystick for futaba transmitter," https://aerialpixels.com/shop/transmitters-and-receivers/transmitters/add-on-joystick-for-futaba-transmitter/, last visited 2017-08-28.

[87] C. Powers, D. Mellinger, A. Kushleyev, B. Kothmann, and V. Kumar, "Influence of aerodynamics and proximity effects in quadrotor flight," in *Experimental robotics*. Springer, 2013, pp. 289–302.