NTNU
Norwegian University of
Science and Technology

A vertical stack approach to cooperative drone lifting

Stian Låte

Master of Science in Cybernetics and Robotics
Submission date: June 2018
Supervisor: Tor Arne Johansen, ITK

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Problem description:

Describe and investigate an extendable vertical stack system for cooperative drone lifting. Modeling, simulation, and implementation should all be part of the consideration.

The following three aspects should be emphasized in the analysis.

1. For the vertical stack, what will be a suitable length for the connecting links to minimize vertical thrust loss while maintaining system compactness?
2. How can a formation control system be designed to make any number of drones reach the desired vertical stack formation, and move as a group in a coordinated manner?
3. To which degree is uneven thrust a problem for the vertical stack, and how can the system be made to distribute the payload mass equally among the drones?

Abstract

In the Pyeongchang 2018 Winter Olympics, over 1200 coordinated drones displayed an impressive lightshow for the opening ceremony. This is a testimony of the fact that the field of drone swarm technology has advanced rapidly in recent years. In future development, moving from coordinated to cooperative utilization of drones will lead to a more commercially viable way of transportation and delivery through drone lifting. Inherent challenges of established methods for cooperative drone lifting are limited scalability, and complicated collision avoidance. This report presents a novel approach to cooperative drone lifting which aims to alleviate these challenges.

In the new approach, multiple drones are interconnected in a vertical formation. The physical connections are made as two rigid, lightweight rods connected by freely rotating joints. In this way, the gravitational pull of a hanging payload can be propagated upwards, and distributed among the cooperating drones. The benefit of this is extended lifting capabilities and system redundancy. The examination of this novel concept was divided into two main parts, one was a derivation of a system model with subsequent simulations, and the other was a real-world implementation.

The implementation involved two drones, equipped with a Pixracer autopilot, RTK positioning equipment, and an onboard companion computer, providing a framework for custom control design. The companion computer was configured to communicate with a stationary system which sends reference positions and operation plans, and which gives the user a system level overview. Sensor accuracy and aerodynamic influence between drones were measured using the physical platform.

A new model for simulating the dynamics of the lifting scheme was developed. By making conservative simplifications and by finding constraint forces from solving the Udwadia-Kalaba equation, simulations including several connected drones were made possible. System control was reached by an implementation of three control layers: single drone, formation and formation guidance. An approximation of the aerodynamic influence between the cooperating drones, found from the realized system, was included, and its implications studied.

Single drone flight was achieved using the realized system. With the simulated system including aerodynamic influence, a vertical formation containing ten drones successfully traversed the desired path.

Keywords: Cooperative Drone Lifting, Quad-Copter, Udwadia-Kalaba Equation, Slung Load, Rotor Downwash, UAV, LSTS, DUNE, Arducopter

Sammendrag

I Pyeongchang 2018 Vinter-OL viste over 1200 koordinerte droner et imponerende lysshow for åpningsseremonien. Dette er et eksempel på at drone-sverm feltet har utviklet seg svært raskt de siste årene. I fremtidig utvikling vil flytting fra koordinert til kooperativ utnyttelse av droner føre til en mer kommersielt forsvarlig måte å tilby transport og levering ved bruk av droner. Utfordringer med etablerte metoder for kooperativ løft ved bruk av droner inkluderer begrenset skalerbarhet, og komplisert kollisjons ungåelse. Denne rapporten presenterer en ny tilnærming til kooperative løft ved bruk av droner som har som mål å lette disse utfordringene.

I den nye tilnærmingen er flere droner koblet sammen i en vertikal formasjon. De fysiske forbindelsene er laget som to stive lettvektstenger forbundet med fritt roterende skjøter. På denne måten kan tyngdekraften av en hengende nyttelast spres oppover og fordeles blant de samarbeidende dronene. Fordelen med dette er utvidet løfteevne og en grad av redundans. Undersøkelsen av dette nye konseptet ble delt inn i to hoveddeler, først utledning av system-modell og utførelse av simuleringer, og deretter en fysisk implementasjon av konseptet.

Implementasjonen involverte to droner, utstyrt med en Pixracer autopilot, RTK posisjoneringsutstyr og en assisterende computer som tilbyr et rammeverk for tilpasset kontrolldesign. Den assisterende computeren ble konfigurert til å kommunisere med et stasjonært system som sender referanseposisjoner og operasjonsplaner, og som også gir brukeren oversikt på systemnivå. Sensor nøyaktighet og aerodynamisk påvirkning mellom droner ble målt ved hjelp av den fysiske plattformen.

En ny modell for simulering av løfteplanens dynamikk ble utviklet. Ved å gjøre utvalgte forenklinger og ved å finne kreftene for fysisk sammenkobling ved å løse Udwadia-Kalaba-ligningen, ble simuleringer med flere sammenkoblede droner muliggjort. Systemkontroll ble oppnådd gjennom implementering av tre kontrolllag: enkelt-drone, formasjon og formasjons navigasjon. En approksimasjon til den aerodynamiske innflytelsen mellom de samarbeidende dronene, funnet fra det realiserte systemet ble inkludert.

Flyvning med drone ble oppnådd ved hjelp av det realiserte systemet. Tet simulerte systemet, inkludert aerodynamisk innflytelse, muliggjorde at en vertikal formasjon som inneholdt 10 droner traverserte en planlagt løype.

Preface

This thesis was written for the mandatory master course, TTK4900, at the department of engineering cybernetics at the Norwegian University of Science and Technology.

This is a continuation of a project which was first initiated at the Sampei Lab at Tokyo Institute of Technology in the spring of 2017, under the supervision of Prof. Mitusji Sampei and Ibuki Tatsuya. In this initial project, a small two-dimensional analytical model containing two connected drones was considered. Next, as the subject of a project thesis under the guidance of Tor Arne Johansen in the fall semester 2017, a three-dimensional extension of the model including two drones and a payload was developed. I am sincerely grateful for their support and guidance. A more detailed summary of the background, support, and preceding work can be found in appendix E.

I would also like to thank my brother, mother, and father, for their unwavering support.

Trondheim, 2018-06-09

Stian Låte

Contents

Abstract ii
Sammendrag iii
Preface iv

1 Introduction **1**
1.1 Background And Motivation 2
1.1.1 The Established Scheme 2
1.2 The Vertical Stack Approach 3
1.2.1 Related work 5
1.3 Thesis Outline 7

2 Background Theory **11**
2.1 Notation 11
2.1.1 State Variables 11
2.2 Transformations 12
2.3 Vertical Stack Dynamics 12
2.4 Constrained Systems 13
2.4.1 The Udwadia-Kalaba Equation 14
2.4.2 Moore Penrose Pseudo-Inverse 17
2.5 Positioning 18

3 Drone Technology **23**
3.1 System Architecture 23
3.2 Single Drone System 23
3.2.1 Drone Onboard Overview 23
3.2.2 Ground Control Overview 24
3.3 Component Description 25
3.3.1 Pixracer Autopilot 25

3.3.2 Beaglebone Black Wireless . . . 26
3.3.3 Battery and Power distribution board . . . 27
3.3.4 Frame And Motors . . . 28
3.3.5 RTK And GPS System . . . 28
3.3.6 Physical Interconnection . . . 29
3.4 System Specification Summary . . . 31
3.5 Software And Frameworks . . . 31
3.5.1 DUNE . . . 31
3.5.2 IMC . . . 32
3.5.3 Neptus . . . 32
3.5.4 Ardupilot . . . 33
3.5.5 RTKLIB . . . 33
3.6 Evaluating The Test Platform . . . 33
3.6.1 Measuring Downwash Severity . . . 33
3.6.2 Positioning And Sensors . . . 34
3.6.3 UAV Progress . . . 35

4 Simulation Methodology **39**
4.1 System modeling . . . 39
4.2 System Control . . . 42
4.2.1 Inner Control System . . . 42
4.2.2 Formation Control . . . 46
4.3 Simulation . . . 50
4.4 Downwash Model Extension . . . 52
4.4.1 Downwash Compensation . . . 54
4.4.2 Efficiency Estimate . . . 54
4.4.3 Payload Distribution . . . 55

5 Results **57**
5.1 Real Platform Testing . . . 57
5.1.1 Susceptibility To Drone Tilt . . . 58
5.1.2 Height . . . 59
5.1.3 Fix Precision . . . 60
5.1.4 DUNE Estimate . . . 61
5.1.5 UAV Flight . . . 62
5.2 Simulation Results . . . 63
5.2.1 Formation Reach . . . 64
5.2.2 Traversing A Path . . . 66

5.2.3 Downwash Effect And Compensation . . . 67
5.2.4 Determining Link Length . . . 70

6 Discussion **73**
6.1 Realized Drone System . . . 73
6.1.1 Downwash Effect Measurement . . . 73
6.1.2 RTK Position Evaluation . . . 74
6.1.3 Flight Behavior . . . 77
6.1.4 Realized Drone System - Closing Discussion . . . 78
6.2 Simulation And Model . . . 79
6.2.1 Single Drone Behavior . . . 79
6.2.2 Multi-UAV Behavior . . . 81
6.2.3 Downwash Influence . . . 83
6.2.4 Payload Distribution . . . 85
6.2.5 Choice Of Link Length . . . 86
6.2.6 Model Realism . . . 87
6.2.7 Potential System Extensions . . . 89
6.2.8 Simulation And Model - Closing Discussion . . . 91
6.3 Problem Description - Closing Discussion . . . 92

7 Closing Summary And Conclusions **95**

8 Recommendations For Further Work **99**

A Acronyms **101**

B Project Thesis Mathematical Model **103**

C Project Thesis Guidance System **113**

D Passivity Based Formation Control **115**

E Background And Preceding Work **121**

List of Tables

2.1 Overview of frequently used notation and variables. 12
2.2 Overview of system model variables from the previous model. 13

3.1 Physical specifications of the implemented UAV system. 31

5.1 Overview of a population of 1250 position solutions found using **patch** antenna for increasing angles. 58
5.2 Overview of a population of 1250 position solutions found using **helix** antenna for increasing angles. 58

6.1 Plot specification table to Fig. 5.11. 79
6.2 Plot specification table to Fig. 5.12 and Fig. 5.13. 80
6.3 Plot specification table to figures 5.14, 5.15 and 5.16. 81
6.4 Plot specification table to Fig. 5.17. 82
6.5 Plot specification table to Fig. 5.18. 83
6.6 Plot specification table to Fig. 5.20. 83
6.7 Plot specification table to Fig. 5.21. 84
6.8 Plot specification table to Fig. 5.22 84
6.9 Plot specification table to Fig. 5.23 85
6.10 Plot specification table to Fig. 5.24. 86
6.11 Plot specification table to Fig. 5.26, and Fig. 5.27. 87

B.1 System control inputs. 104

List of Figures

1.1 Four drones showing two variations of the flat lifting schemes, differentiated by the payload shape and connection points. . . . 2
1.2 Multiple individual drone units with two attached links can be interconnected to form a *vertical stack* to distribute the payload burden. . . . 4
1.3 In situations where the conventional flat lifting schemes are applied, replacing individual drones with vertical stacks would improve lifting capacity. . . . 5

2.1 A simple pendulum illustrating the concept of constrained systems. . . . 14
2.2 The outer edge of a square body c is attached to a fixed point-mass j. . . . 15
2.3 Comparison showing constrained and unconstrained behavior of cube c and point-mass j. . . . 18
2.4 The unconstrained acceleration of the square body is altered. . . . 19
2.5 Conventional GNSS systems only include the satellites and the original receiver (UAV in this case). RTK improves the position estimate using a reference base station. . . . 20
2.6 The signal sent from the satellites is a combination of multiple signals, modulated onto the $1572.42MHz$ carrier signal. . . . 21

3.1 Overview of the modules of the drone platform. . . . 24
3.2 The base station contains a router which acts as the central node for the entire stack system. . . . 24
3.3 The Pixracer autopilot contains most of the sensors, and performs the low-level control. . . . 25
3.4 Image of the Beaglebone Black Wireless (BBBW). . . . 26
3.5 Images of an assembled drone, highlighting the location of the components. . . . 27
3.6 Images of the components assembled into a single drone. . . . 27
3.7 Images of the two included GNSS receivers for the drone platform. . . . 28

3.8 Images of the two antennae considered for the system. . . . 29
3.9 The base station components resemble the drone components, and provides the RTK GNSS correction data. . . . 30
3.10 Picture of the two UAVs with the connecting links, and a small dense 0.3$[kg]$ payload. 30
3.11 Test setup to evaluate the aerodynamic influence between drones. . . . 34
3.12 Comparison of PID parameters before and after tuning. . . . 36

4.1 Block diagram overview of the inner control system. . . . 42
4.2 S bodies are included in the vertical stack, including UAVs, connecting joints, and a payload. . . . 44
4.3 Graph for three UAVs communicating over three communication links. . . . 47
4.4 Block diagram showing the structure of the formation control system. . . . 50
4.5 Variable overview to model downwash intensity. . . . 53
4.6 Block diagram illustrating the agreement problem of equal payload distribution between the drones. . . . 55

5.1 The change of maximum thrust measured, with the curve-fitting line used to approximate the thrust efficiency reduction as a function of vertical drone distance. . 57
5.2 Position solutions for various antenna angles using a patch antenna. . . . 58
5.3 Position solutions for various antenna angles using a helix antenna. . . . 59
5.4 Height estimate from DUNE using patch antenna measurements. . . . 59
5.5 Height estimate from DUNE using helix antenna measurements. . . . 59
5.6 Grouping of the fix solutions from the angle test are seen to be very dense. . . . 60
5.7 Estimate of position away from average position fetched from DUNE using the helix antenna data. . . . 61
5.8 Height estimate from DUNE over time using helix antenna measurements. . . . 61
5.9 One of the UAVs maintaining altitude. . . . 62
5.10 Simultaneous logging of RTK receiver measurements and DUNEs internal position estimate. . . . 62
5.11 The single drone case given desired input velocity as a spiral, slight delay is seen between the commanded force and actual force realized by the quad-rotor. . . . 63
5.12 Simulation showing a single UAV with a 0.3$[kg]$ payload reaching 3$[m/s]$, then having desired velocity suddenly inverted. . . . 63
5.13 Plots corresponding to Fig. 5.12, showing constraint force $\boldsymbol{Q}_c$, and the constraint numerical slip. . . . 64
5.14 Three unconnected UAVs attempt to reach the desired formation topology and velocity from random starting points without integral action. . . . 64

5.15 Three unconnected UAVs approach the desired formation topology and velocity from random starting points with integral action. . . . 65
5.16 Three unconnected UAVs approach the desired formation topology and speed this time using the gradual integral system. . . . 65
5.17 Simulation showing the two drone stack with a $0.4[kg]$ load traversing a 7×7 meter square, with reasonably small payload swing. . . . 66
5.18 The simulation successfully extends to 10 drones, traversing the same square with a $2[kg]$ payload. . . . 66
5.19 Two plots showing how downwash is represented as a function of UAV position and orientation, from section 4.4. . . . 67
5.20 Three drones carrying a $0.4[kg]$ load traversing a path before including the effect of downwash. . . . 67
5.21 Similar setup as Fig. 5.20, only this time downwash appears problematic for the altitude tracking. . . . 68
5.22 Including the downwash feed-forward resolves most of the downwash influence, and the path is followed with reasonable precision. Notable difference between expended energy is seen. . . . 68
5.23 Similar setup, only this time with downwash, downwash compensation and the load sharing system implemented. . . . 69
5.24 Ten drone stack with $2.0[kg]$ payload, and link length $0.8[m]$, falling due to the downwash force. . . . 70
5.25 Total downwash from above, as experienced by UAVs at $t = 0$ in ten drone stack. . . . 70
5.26 Ten drone stack with $2.0[kg]$ payload executing single turn, with link length $1.0[m]$, maintaining altitude but showing some oscillation. . . . 71
5.27 Ten drone stack with $2.0[kg]$ payload executing single turn, with link length $1.2[m]$ successfully. . . . 71

6.1 Hexarotors with rotated motor arms to redirect downwash and gain sideways actuation. . . . 89
6.2 Illustration of the *Y extension* in the case of one of the top UAVs failing. . . . 91

B.1 Illustration of the *minimal stack* showing the index order of rigid bodies. . . . 104
B.2 The separate components included in the simplified two drone model. . . . 105
B.3 Illustration of the control input forces and torques. . . . 106
B.4 The rigid bodies have been modelled as simple shapes with uniform density, drone A and link 1 are displayed as an example. . . . 107
B.5 The holonomic constraints express how the mass centers of the rigid bodies are positioned relative to one another. . . . 109

C.1 Concept illustration for the Line-Of-Sight guidance steering law. 113

Chapter 1

Introduction

In congested urban areas, and inaccessible locations the use of unmanned aerial vehicles (UAV)s could prove to be one of the remedies used to alleviate the clutter of existing transportation infrastructure. There are, however, a number of challenges currently facing transportation by UAV, such as limited lifting capabilities, battery lifetime and safety. The rapid improvement in performance and cost of hardware has motivated academia and commercial giants like Amazon [1] and Google [2] to invest in further research of the prospect of UAV transport and parcel delivery. The focus of this thesis is to analyze a new multi-UAV cooperative lifting scheme to meet these challenges and to investigate the advantages and disadvantages which this new lifting strategy could present.

Single UAV systems have become widespread, and they are suitable for tasks such as mapping and observation. By implementing very large UAVs as has been done by *Griff Aviation* [3], improved lifting capacity can also be achieved. Two main problems burden single UAV systems at a conceptual level. The first is *scalability*. If the task is not near identical each time and exposed to the same constraints, the single drone systems will often be too small, or unnecessarily large for the task at hand. The second is *modularity*, a single drone system is in most cases either operational or disabled, unlike the multi-UAV systems where parts of the system can be subtracted, inspected, repaired and added back to the drone swarm system without unnecessary downtime. These two terms will reappear in the thesis, and an explicit definition as they are interpreted in this work is appropriate.

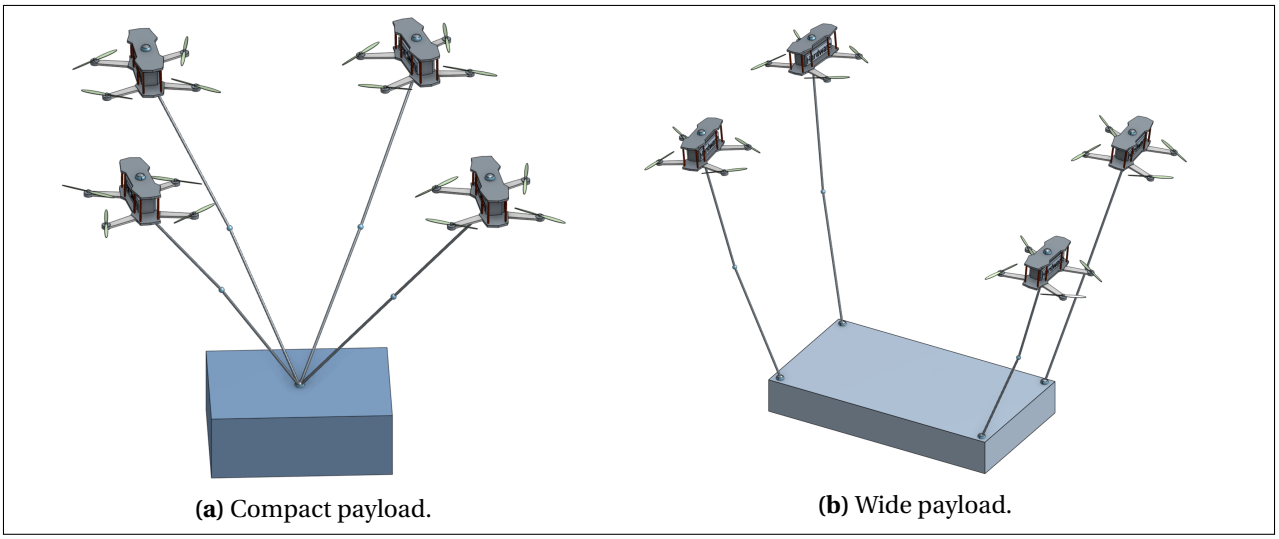
- *Scalability* - The ability of a system to adapt to the magnitude of a task.
- *Modularity* - The degree to which a system's components may be separated and recombined.

1.1 Background And Motivation

1.1.1 The Established Scheme

Before introducing the cooperative lifting approach which is the main contribution of this thesis, the most common lifting formation which has received widespread attention will be described and briefly evaluated. This is to provide context, as well as a conceptual understanding of the difference between the systems. It should also be mentioned regarding terminology that this thesis refers to a single UAV as either a drone or a UAV interchangeably.

The most common formation for cooperative drone lifting is illustrated by Fig. 1.1a and Fig. 1.1b, differentiated by the payload attributes. This drone formation is structured so that each UAV is connected directly to the payload while maintaining a safe distance to the other connected UAVs. For the subsequent work, we refer to these types of cooperative lifting schemes as *flat* lifting schemes, because the formation is usually characterized by the UAVs operating in the same flat plane.



(a) Compact payload.

(b) Wide payload.

Figure 1.1: Four drones showing two variations of the flat lifting schemes, differentiated by the payload shape and connection points.

Advantages

The flat cooperative lifting schemes can increase the total payload capacity, as one would want from a cooperative system. For payload shapes such as Fig. 1.1b, it also offers control of the orientation of the payload, which could be beneficial. The tilted orientation of the drones also means that the formation is able to actuate sideways. Another advantage is the fact that identical UAVs can be used in this approach, this would simplify production and design in the long

term, and keep costs down.

Disadvantages

There are however challenges posed by the flat lifting schemes. First, the close proximity which the UAVs are likely to operate within makes collision avoidance a demanding and risky problem which must be considered. Second, the minimum acceptable distance between drones means that the airspace directly above the payload will become clogged up when including a high number of UAVs. Third, in a flat scheme, only a few drones can operate at a near zero degree tilt away from the payload. This is also because of the need to maintain a safe distance from the other drones cooperating in the lifting operation. Finally, for another drone to be added into the lifting formation during operation, it is necessary that the other drones accommodate by giving the new drone direct access to the payload to connect.

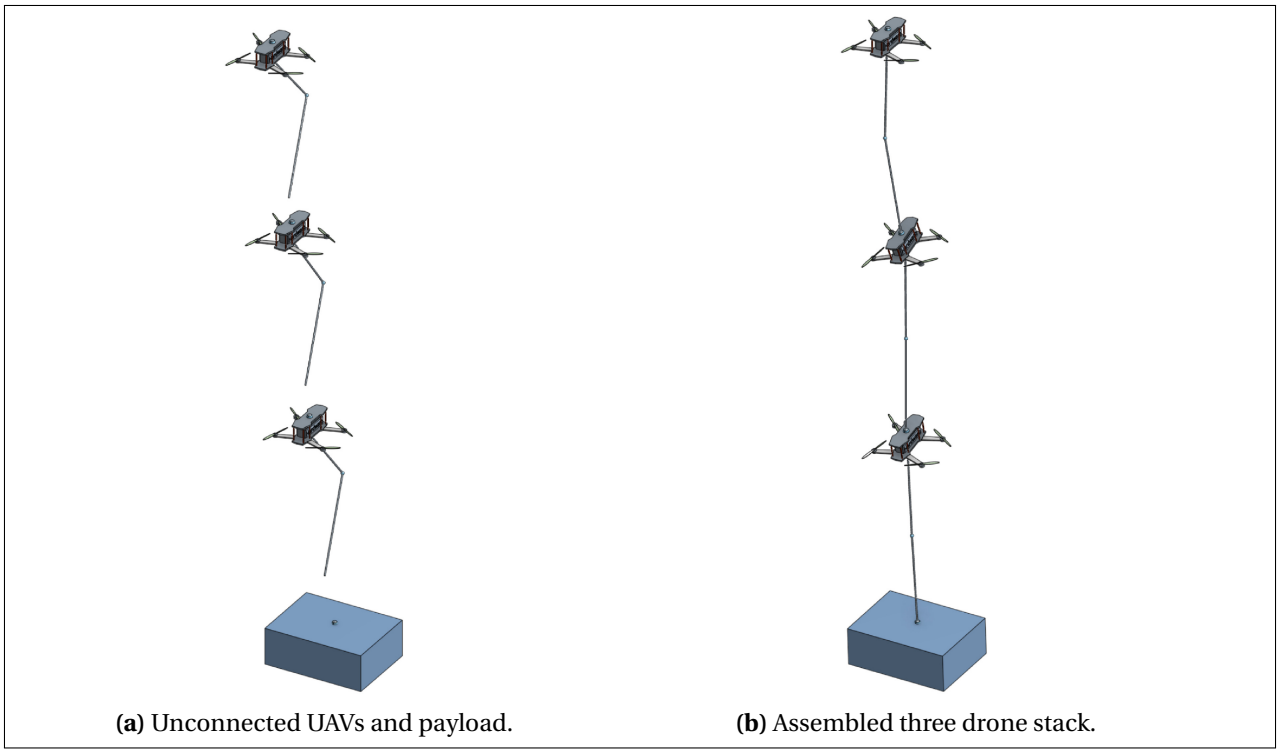
This is by no means a complete evaluation, but the observations provide motivation for proposing an alternative lifting approach to alleviate these shortcomings.

1.2 The Vertical Stack Approach

With this brief summary of the strengths and weaknesses of the flat lifting schemes discussed, the cooperative lifting approach which this thesis is to analyze can be introduced. In this approach each drone is equipped with two freely rotating rigid links as depicted on the left-hand side in Fig. 1.2. The links and drones are designed to allow for interconnection, forming a chain of drones referred to by this thesis as a *vertical stack*, or just *stack*. The drones used for modeling and development in this project are chosen to be standard quad-copters due to cost, accessibility and adoptance in academia. This is not to rule out the use of other multi-copters in the future, only to propose a convenient platform alternative for this work. Now we proceed with a short evaluation of the vertical stack, similar to the one presented for the flat lifting schemes.

Advantages

First and perhaps most significant of the assumed benefits the vertical stack could give is scalability. While the flat lifting scheme is constrained by limited space, the vertical stack might allow a very large drone swarm to cooperate in a single task. Even in cases where the flat lifting scheme could be used, each individual drone might be replaced by a vertical stack as is illustrated by Fig. 1.3. Another advantage is the simplicity of the collision avoidance. As the drones in a single stack will operate at different heights, avoiding collisions between drones when the



(a) Unconnected UAVs and payload. **(b)** Assembled three drone stack.

Figure 1.2: Multiple individual drone units with two attached links can be interconnected to form a *vertical stack* to distribute the payload burden.

stack has been assembled should be significantly more straightforward for the vertical stack, than for its flat counterpart.

A third argument for the vertical stack is that the drones in the formation should be able to operate at near zero degrees tilt for a steady hover. This is because all drones in the formation are placed almost directly above the payload. The result could mean that no force is wasted from constant drone tilt. Finally, new drones can be added to the stack at the top with relative ease, allowing for rapid adaptation to meet the problem at hand.

Disadvantages

The vertical stack formation does present new obstacles as well as these potential benefits. First, multi-rotors rely on forcing air downwards to generate thrust; this will in effect mean that lower drones in the stack will operate in downwards "wind" likely causing effect loss. To use terminology applied in a related research paper [4], this is referred to as **downwash** in this thesis.

Second, the sudden failure of the uppermost drone could perhaps result in a complete system failure, by the failing drone pulling on and colliding with the remaining stack. This requires

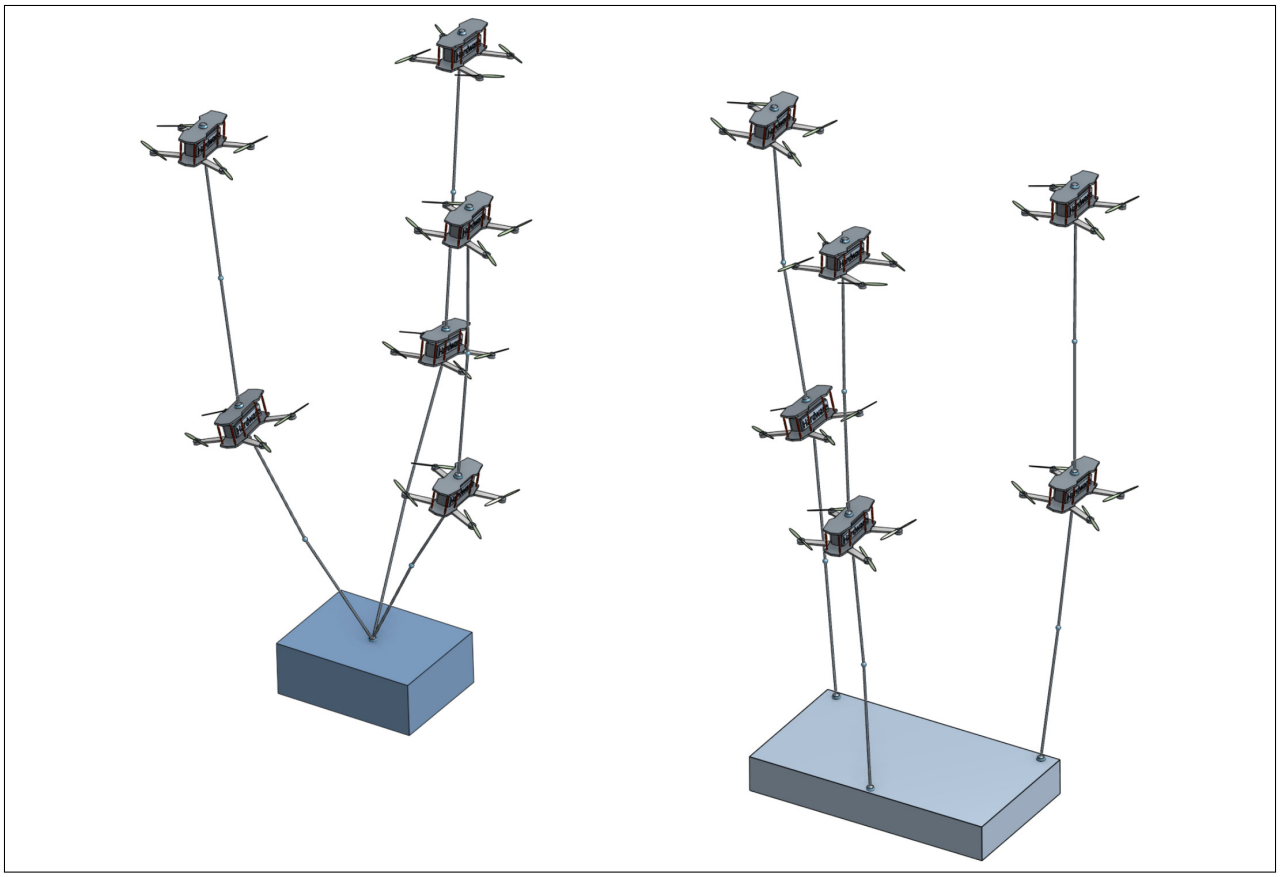


Figure 1.3: In situations where the conventional flat lifting schemes are applied, replacing individual drones with vertical stacks would improve lifting capacity.

attention, as this type of single point failure, would likely not be acceptable for a real-world application. There are other aspects to the vertical stack to be discussed; however, this is only intended as a first evaluation from which the remaining thesis elaborates.

1.2.1 Related work

At the time of writing, identical or similar research on vertical drone stacks had not been found through the literature searches. However, some indirectly related work and technical aspects to the vertical stack approach to cooperative drone lifting discovered throughout the searches are summarized here.

Multi-Drone Management

A prerequisite for cooperative drone movement is a functioning system for multi-drone management. By this, we mean a framework for multiple cooperating vehicles to work and communicate within. A tool-chain for communication and control of multiple autonomous vehicles called the LSTS toolchain described in [5] was chosen for the work in this thesis. Other alterna-

tives do exist, such as a collaborative mission system discussed in [6], or the open-source work done by the Arducopter community [7].

Multi-Drone Movement

For multiple drones or vehicles to move in a coordinated manner a suitable control approach is needed. A system for coordinated multi-rotor movement was proposed in [8], and was a helpful guide to practical implementation of the passivity-based control theory from [9]. An alternative formation approach would be the geometric control system for cooperative quad-rotor lifting which was researched in [10].

Aerodynamic Interactions

The degree to which quad-copters flying in close proximity to each other interact will be central to the work in this thesis, as mentioned before it was suspected to be one of the two main problems facing the vertical stack as a formation lifting scheme. Closely related to the interaction between drones is the empirical models for various rotorcraft downwash found in [4]. These empirical models are found for a quad-rotor as well as other rotor-craft and was used in downwash approximations in this thesis. Another project which touches upon analysis of the aerodynamic interaction between quad-copters is presented in [11], where a cylindrical region below an operating drone significantly influences drones below it is defined.

Quad-Rotor Dynamics

Numerous papers have studied the development of quad-rotor platforms putting emphasis on different aspects. A model for quad-rotor dynamics used as inspiration was *Quad Rotorcraft Control: vision-based hovering and navigation*, [12]. This was also of interest because of the vision-based control detailed, as that might be of interest in future variants of the new formation. A system can be seen where a single rigid rod is stabilized as an inverted pendulum by using a bi-linear control method was found here [13], and is noted due to the modeling of a rigid rod upon a quad-rotor, making it somewhat related to the vertical stack.

Suspended Load

The payload of the formation in this thesis is suspended using two rigid links below the vertical stack. For this reason, it classifies as a suspended load control problem. Multi-drone suspended payload carrying is researched in [14], where multiple UAVs cooperate in following a common path with a shared load. This was closely related to the constraint modeling done in this study on vertical stack systems and is therefore included in this related works section. Other consider-

ations must also be made, such as payload swing. This has been researched for the single drone case, with swing dampening efforts made with passivity-based control in [15].

Physically Connected Flight Systems

There are several difficulties with interconnecting flying vehicles physically. One example of this kind of problem being tackled is a distributed flight array [16] proposed by ETH Zurich, wherein identical hexagon shaped UAV units are connected to form a complete flying structure. While this did not directly influence the work on the vertical stack concept, it is noted as an interesting approach to achieve modularity for a flight system.

Flat Lifting Schemes

The flat lifting schemes have been studied in depth in various scenarios, examples of the study of flat lifting schemes applied to wide payloads such as illustrated in Fig. 1.1b was found in [17], [18], and [19]. They detail characteristics such as control of the orientation of a payload using multiple flying vehicles. Other, less conservative approaches to cooperative drone lifting exists, discussing possibilities such as direct attachment to the payload or clamping mechanisms would be [20], [11] or [21], but these are less prevalent, and are only mentioned to show that other alternatives than the conventional flat lifting schemes are being considered in the field.

1.3 Thesis Outline

Background Theory

The theoretical foundation upon which the rest of the report rests is given in this chapter. The notation used in the report is summarized, together with frames of reference, and applied mathematical transformations. The two subjects given most attention in this chapter is the modeling of constraints and the principles behind the applied positioning sensor technology. With this basis, it should be possible for a reader with some background in control theory to appreciate the discussions and findings of the thesis.

Drone Technology

This chapter is intended to explain what components were assembled and used to implement the drone platform, it also includes a description of the software and protocols in use. This explanation is given to describe the function of each component and software module, as well as the way that these components and modules communicate to form a test platform for the ver-

tical stack formation. The chapter ends by summarizing the specifications of the implemented drones, and a step by step development overview for the realized platform.

Simulation Methodology

The model from which a simulation system was derived is detailed in this chapter. The approximations made at this stage are stressed so that the models and the result that can be adequately discussed in later chapters. An inner control system for the drones in this model is detailed first, leading up to the formation control scheme with a short summary of what the formation system is expected to achieve. After introducing the method of simulation, extensions are made to increase the realism of the model by approximating drone downwash. Finally, control system adaptations to respond to the downwash are detailed. The order of these subjects was chosen such that the model and control system could be segmented into understandable parts for later discussion.

Results

The results from the preceding technology and methodology chapters chosen to clearly illustrate the properties of the vertical stack are presented here. This includes sensor measurements, operation images and simulation plots. The chapter is kept as brief as possible, reserving the discussion of the data, and the drawing of conclusions to later chapters.

Discussion

Details from the results chapter, and how they relate to the work described in the drone technology and simulation methodology is discussed in this chapter. The reader's attention is directed to points of interest in the plots, and interpretations of the discovered results is stated with their implications to the vertical stack. Thereafter potential extensions which can or should be made to the system are shown and briefly evaluated. Finally, a broad consideration of the vertical stack concept is given, such that the findings and observations are put into perspective.

Closing Summary And Conclusions

The contributions of the thesis are summarized, and a few selected concluding paragraphs about the vertical stack are presented. The intention here is to give a concise closing examination of the proposed concept.

Recommendations Of Future Work

Suggestions for future research efforts continuing the work of this thesis are given. This includes short-term goals and practical matters, and a few long-term considerations that would be beneficial to research, for the vertical stack to become a reality.

Chapter 2

Background Theory

The purpose of this chapter is to provide a theoretical basis for this thesis. This is done by first defining the notation to be used. Thereafter, the mathematical model derived in a preceding project to model the vertical stack using Lagrangian constraints is summarized. Next, after a brief commentary on system constraints, an explanation of the Udwadia-Kalaba equation with a corresponding example is presented. Finally, an introduction to the theory behind the positioning systems used onboard the physical drones is given.

2.1 Notation

Two frames of reference will be used in this report, both of which are fairly conventional in the modeling of rigid bodies.

- $\{n\}$, North-East-Down frame, well established in vehicle modeling.
- $\{b\}$, Body-fixed frames are also used, referred to as $\{b\}$, and fixed to each of the bodies in the system. Body frame $\{b\}$ has orthogonal axis b_x-b_y-b_z.

2.1.1 State Variables

For both vectors and matrices, **boldface** characters are used to emphasize that they are not scalar values. To the extent possible and practical the notation used in *Handbook of Marine Craft Hydrodynamics and Motion Control* [22] has been applied, some adjustments have been made to suit a multi-vehicle system. The motivation for these adjustments was to avoid unnecessarily bloated notation.

Variable	Description
$\dot{a}$	Time derivative $\frac{\partial a}{\partial t}$ of example variable a.
a_{cmd}	Read $a_{command}$, is the commanded reference for example variable a.
a_d	Read $a_{desired}$, is the desired state for example variable a.
$\boldsymbol{p}^n_{b,i}$	Position of body i relative to $\{n\}$ frame, $\boldsymbol{p}^n_{b,i} = [x_i, y_i, z_i]^T$.
$\boldsymbol{\Theta}_{n,i}$	Orientation of rigid body i relative to $\{n\}$ frame, $\boldsymbol{\Theta_{n,i}} = [\phi_i, \theta_i, \psi_i]^T$.
$\boldsymbol{v}^b_{b,i}$	Velocity of rigid body i relative to $\{b\}$ frame.
$\boldsymbol{\omega}^b_{b,i}$	Pitch, Yaw and Roll rate $\boldsymbol{\omega}^b_{b,i} = [p_i, q_i, r_i]^T$ around b_{iy}, b_{ix} and b_{iz}, respectively.
$\boldsymbol{\eta}_i$	State vector for body i, $\boldsymbol{\eta}_i = \begin{bmatrix} \boldsymbol{p}^n_{b,i} \\ \boldsymbol{\Theta}_{n,i} \end{bmatrix}$.
$\boldsymbol{\nu}_i$	Velocity vector for body i in body frame $\{b\}$, $\boldsymbol{\nu}_i = \begin{bmatrix} \boldsymbol{v}^b_{b,i} \\ \boldsymbol{\omega}^b_{b,i} \end{bmatrix}$

Table 2.1: Overview of frequently used notation and variables.

2.2 Transformations

Two useful transformations have been introduced in order to move between frames of reference. First, a rotational matrix, with s_a and c_a representing sine and cosine of a respectively. This was used to rotate from frame $\{b\}$ to frame $\{n\}$.

$$R(\boldsymbol{\Theta}) = R_{z,\psi} R_{y,\theta} R_{x,\phi} = \begin{bmatrix} c_\theta c_\psi & c_\psi s_\theta s_\phi - c_\phi s_\psi & c_\phi s_\psi + c_\phi c_\psi s_\theta \\ c_\theta s_\psi & c_\phi c_\psi + s_\theta s_\phi s_\psi & c_\phi s_\theta s_\psi - c_\psi s_\phi \\ -s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{bmatrix} \tag{2.1}$$

The second transformation introduced relates angle-rate relative to frame $\{n\}$, $\dot{\phi}, \dot{\theta}, \dot{\psi}$, to body-fixed angular velocity $\boldsymbol{\omega}^b_b$. This transformation came from equation (2.28) in [22].

$$T^{-1}_{\boldsymbol{\Theta}} = \begin{bmatrix} 1 & 0 & -s_\theta \\ 0 & c_\phi & c_\theta s_\phi \\ 0 & -s_\phi & c_\theta c_\phi \end{bmatrix} \implies T_{(\boldsymbol{\Theta})} = \begin{bmatrix} 1 & s_\phi t_\theta & c_\phi t_\theta \\ 0 & c_\phi & -s_\phi \\ 0 & \frac{s_\phi}{c_\theta} & \frac{c_\phi}{c_\theta} \end{bmatrix} \tag{2.2}$$

$$\boldsymbol{\omega}^b_b = \begin{bmatrix} p \\ q \\ r \end{bmatrix} = T^{-1}_{(\boldsymbol{\Theta})} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{2.3}$$

$$\tag{2.4}$$

2.3 Vertical Stack Dynamics

A highly detailed model for describing the dynamics of a vertical stack containing a number of drones was presented in the preceding project thesis. This subsection gives a very compact

Variable	Description
$\boldsymbol{\lambda}$	Lagrange multiplier, determining constraint force magnitude.
$\boldsymbol{q}$	State variable containing position and angle of each rigid body in the system.
$\boldsymbol{H}(\boldsymbol{q})$	Constraint force matrix, determining constraint force direction.
$\boldsymbol{M}$	Inertia matrix.
$\boldsymbol{F}(\boldsymbol{q})$	Input force vector, resulting from drone rotor forces.
$\boldsymbol{S}(\boldsymbol{q})$	Air resistance vector, counteracting velocity and angular velocity.
$\boldsymbol{N}(\boldsymbol{q})$	Gravity force vector.
$\boldsymbol{C}(\boldsymbol{q})$	Coriolis force vector.

Table 2.2: Overview of system model variables from the previous model.

summary of the previous vertical stack model. For the interested reader, the complete details on how the model was reached are included in appendix B. The intention of displaying this is to be able to highlight the similarities and the differences between the modeling approaches.

In the aforementioned model, each link and drone was approximated to be solid rigid bodies with uniform densities. The connecting forces maintaining the vertical stack structure, where the drones remain interconnected with links were modeled using Lagrange constraints, with [23] as a theoretical basis. The mathematical model of the system dynamics could finally be gathered into the following two equations.

$$\boldsymbol{\lambda} = (\boldsymbol{H}(\boldsymbol{q})\boldsymbol{M}^{-1}(\boldsymbol{q})\boldsymbol{H}^{T}(\boldsymbol{q}))^{-1}(\boldsymbol{H}(\boldsymbol{q})\boldsymbol{M}^{-1}(\boldsymbol{q})(\boldsymbol{F}(\boldsymbol{q}) - \boldsymbol{S}(\dot{\boldsymbol{q}}) - \boldsymbol{N}(\boldsymbol{q})) + \dot{\boldsymbol{H}}(\boldsymbol{q})\dot{\boldsymbol{q}}) \tag{2.5}$$

$$\ddot{\boldsymbol{q}} = \boldsymbol{M}^{-1}(\boldsymbol{q})(\boldsymbol{F}(\boldsymbol{q}) - \boldsymbol{S}(\dot{\boldsymbol{q}}) - \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{N}(\boldsymbol{q}) + \boldsymbol{H}^{T}(\boldsymbol{q})\boldsymbol{\lambda}) \tag{2.6}$$

A compact explanation of the variables used is given in Table 2.2.

This model was successfully used to simulate two connected drones with a slung load. There were, however, issues with the model being unnecessarily comprehensive and slow.

2.4 Constrained Systems

In the field of mechanics, constrained systems is a well-researched subject. Aspects of it are included into many introductory dynamical systems, such as the point-mass pendulum, or the rolling wheel on a flat surface. Usually, in these types of introductory systems, the choice of variables to describe the system state is chosen such that the assignment becomes more compact

and readily understandable, an example being the pendulum in Fig. 2.1. The pendulum can be described as being located in position $\boldsymbol{p}_e(x,y) = [x,\ y]$. However, the knowledge that the center of the pendulum is a constant distance ℓ from the axis of rotation means we can also describe the system fully, by only using the angle β, as $\boldsymbol{p}_g(\beta) = [sin(\beta)\ell,\ -cos(beta)\ell]$.

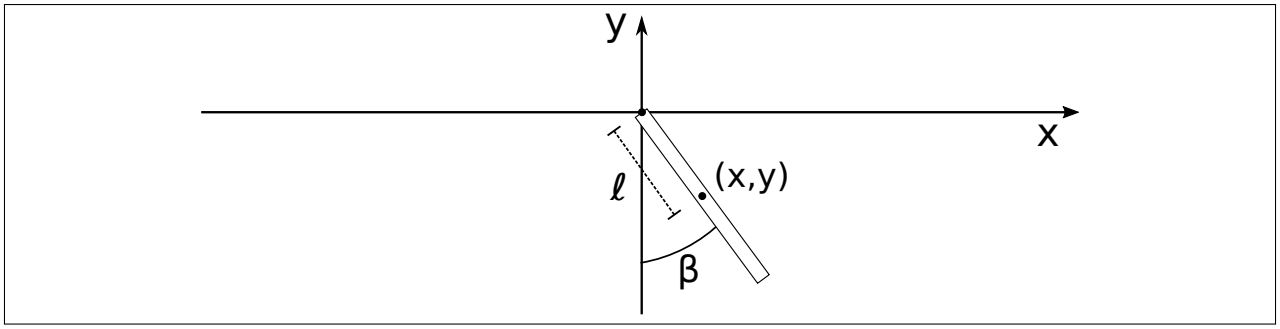


Figure 2.1: A simple pendulum illustrating the concept of constrained systems.

This illustrates the difference between excessive and generalized coordinates, being $\boldsymbol{p}_e(x,y)$ and $\boldsymbol{p}_g(\beta)$ respectively. The constraint of the distance ℓ has reduced the Degrees of Freedom (DoF) of the system by 1, when going from $\boldsymbol{p}_e(x,y)$ to $\boldsymbol{p}_g(\beta)$.

2.4.1 The Udwadia-Kalaba Equation

Solving the Udwadia-Kalaba equation can be used as an alternative to the Lagrangian mechanics method in order to find constrained system dynamics. It is a method which according to [24] applies to both holonomic and nonholonomic constraints, as long as long as they are linear with respect to their accelerations. Consider Eq. 2.7:

$$\boldsymbol{M}\ddot{\boldsymbol{q}} = \boldsymbol{Q} + \boldsymbol{Q_c} \tag{2.7}$$

Where $\boldsymbol{Q}$ refers to the generalized forces acting upon the system, and $\boldsymbol{Q}_c$ represents the constraint force needed to satisfy the imposed constraints. This is closely related to Eq. 2.6, where the role of $\boldsymbol{Q}_c$ is analogous to that of the constraint force term $\boldsymbol{H}^T(\boldsymbol{q})\boldsymbol{\lambda}$ found using Lagrangian mechanics. The remaining forces dictating the system behavior as if it were not subject to constraints can be found in $\boldsymbol{Q}$, thus being related to the terms in Eq. 2.6 excluding $\boldsymbol{H}^T(\boldsymbol{q})\boldsymbol{\lambda}$.

Among the main benefits from transitioning to using the Udwadia-Kalaba Equation is that we avoid the intermediate step of calculating the Lagrange multipliers $\boldsymbol{\lambda}$ as was shown in Eq. 2.5. Constraint forces $\boldsymbol{Q}_c$ are found instead by expressing the constraints in the following form.

$$\boldsymbol{A}\ddot{\boldsymbol{q}} = \boldsymbol{b} \tag{2.8}$$

From this form, the following equation for which a derivation can be found here [25], the constraint force vector $\boldsymbol{Q}_c$ can be found.

$$\boldsymbol{M}\ddot{\boldsymbol{q}}_u = \boldsymbol{Q} \tag{2.9}$$

$$\boldsymbol{Q}_c = \boldsymbol{M}^{1/2}(\boldsymbol{A}\boldsymbol{M}^{-1/2})^{+}(\boldsymbol{b} - \boldsymbol{A}\ddot{\boldsymbol{q}}_u) \tag{2.10}$$

The newly introduced vector $\boldsymbol{q}_u$ refers to acceleration before accounting for constraint forces. The notation $^{+}$ refers to the Moore Penrose pseudo-inverse. In essence, what the Moore Penrose pseudo-inverse does is find the closest direction of acceleration which doesn't violate the applied constraints. The general process is perhaps best illustrated by an example, which is included in part because it is closely tied to the dynamics in focus in this thesis.

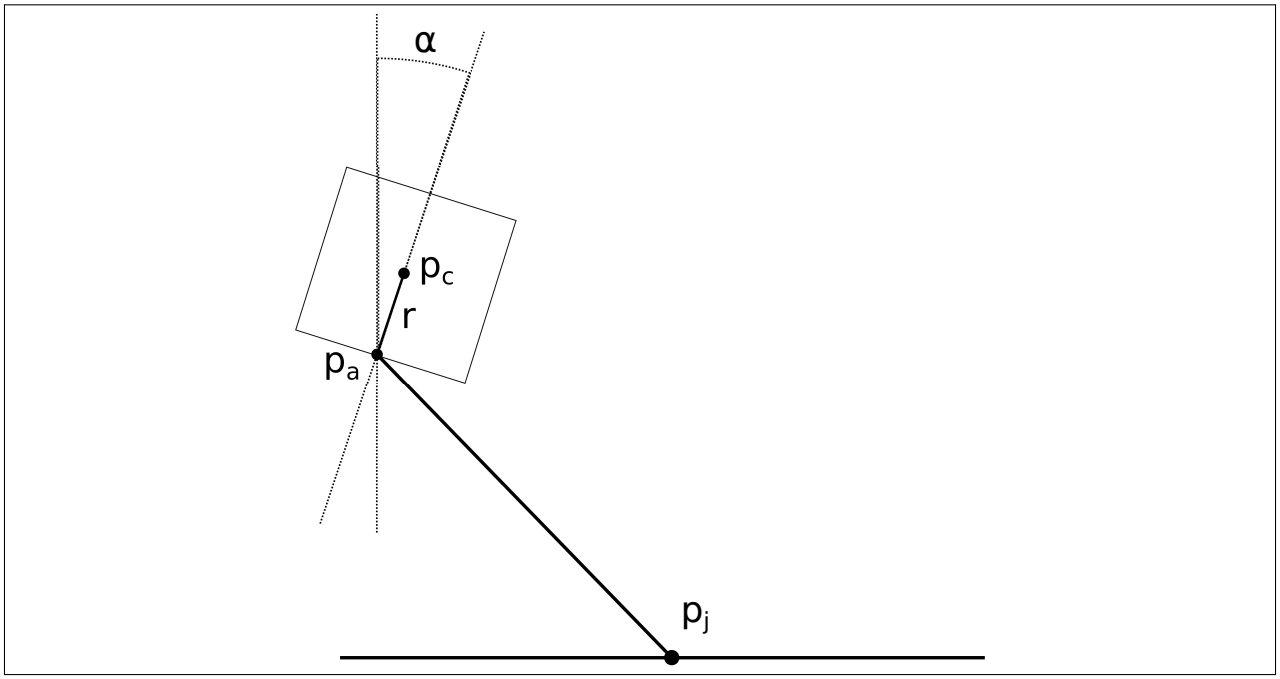


Figure 2.2: The outer edge of a square body c is attached to a fixed point-mass j.

Observing Fig. 2.2, we have a square body c and a fixed point-mass j. The dynamics in the unconstrained case are relatively trivial, with only gravity influencing the square. The system state and unconstrained dynamics are found with mass and inertia matrix $\boldsymbol{M}$ as:

$$\boldsymbol{q} = \begin{bmatrix} \boldsymbol{p}_c^T & \alpha_c & \boldsymbol{p}_j^T \end{bmatrix}^T \quad \in \mathbb{R}^5 \tag{2.11}$$

$$\boldsymbol{M}\ddot{\boldsymbol{q}} = \begin{bmatrix} 0 & -m_c g & 0 & 0 & -m_j g \end{bmatrix}^T \quad \in \mathbb{R}^5 \tag{2.12}$$

We apply the constraint of constant length between the attachment point on the square, and point-mass j. We define a constraint vector $\boldsymbol{L}$ for this constraint, and express it as a function of the position of the two points, as well as the angle of the square.

$$\boldsymbol{L} = \boldsymbol{p}_a - \boldsymbol{p}_j \tag{2.13}$$

$$\boldsymbol{L} = \boldsymbol{p}_c + \boldsymbol{R}(\alpha)\boldsymbol{r} - \boldsymbol{p}_j = \boldsymbol{p}_c + \begin{bmatrix} cos(\alpha) & -sin(\alpha) \\ sin(\alpha) & cos(\alpha) \end{bmatrix} \begin{bmatrix} 0 \\ r \end{bmatrix} - \boldsymbol{p}_j \tag{2.14}$$

The constraint demands that the link vector $\boldsymbol{L}$ keeps constant length d, this constraint is denoted as g. We define this length and its derivatives, as they too are implicitly constrained.

$$g = ||\boldsymbol{L}^2|| - d^2 = \boldsymbol{L}^T\boldsymbol{L} - d^2 = 0 \tag{2.15}$$

$$\dot{g} = 2\dot{\boldsymbol{L}}^T\boldsymbol{L} = 0 \tag{2.16}$$

$$\ddot{g} = 2\ddot{\boldsymbol{L}}^T\boldsymbol{L} + 2\dot{\boldsymbol{L}}^T\dot{\boldsymbol{L}} = 0 \tag{2.17}$$

The first and second order derivatives of $\boldsymbol{L}$ are found next:

$$\dot{\boldsymbol{L}} = \dot{\boldsymbol{p}}_c + \dot{\boldsymbol{R}}\boldsymbol{r} - \dot{\boldsymbol{p}}_j \tag{2.18}$$

$$\ddot{\boldsymbol{L}} = \ddot{\boldsymbol{p}}_c + \ddot{\boldsymbol{R}}\boldsymbol{r} - \ddot{\boldsymbol{p}}_j \tag{2.19}$$

Then to determine the first and second derivative of $\boldsymbol{R}(\alpha)\boldsymbol{r}$:

$$\dot{\boldsymbol{R}}(\alpha)\boldsymbol{r} = \frac{d}{dt}\begin{bmatrix} -sin(\alpha)r \\ cos(\alpha)r \end{bmatrix} = \begin{bmatrix} -cos(\alpha)\dot{\alpha}r \\ -sin(\alpha)\dot{\alpha}r \end{bmatrix} \tag{2.20}$$

$$\ddot{\boldsymbol{R}}(\alpha)\boldsymbol{r} = \frac{d}{dt}\begin{bmatrix} -cos(\alpha)\dot{\alpha}r \\ -sin(\alpha)\dot{\alpha}r \end{bmatrix} = \begin{bmatrix} (sin(\alpha)\dot{\alpha}^2 - cos(\alpha)\ddot{\alpha})r \\ (-cos(\alpha)\dot{\alpha}^2 - sin(\alpha)\ddot{\alpha})r \end{bmatrix} \tag{2.21}$$

Inserting the result from 2.21 into Eq. 2.19 we get the following:

$$\ddot{g} = 2\ddot{\boldsymbol{L}}^T \boldsymbol{L} + 2\dot{\boldsymbol{L}}^T \dot{\boldsymbol{L}} \tag{2.22}$$

$$= 2\dot{\boldsymbol{L}}^T \dot{\boldsymbol{L}} + 2\boldsymbol{L}^T (\ddot{\boldsymbol{p}}_c + \ddot{\boldsymbol{R}}(\alpha)\boldsymbol{r} - \ddot{\boldsymbol{p}}_j) \tag{2.23}$$

$$= 2\dot{\boldsymbol{L}}^T \dot{\boldsymbol{L}} + 2\boldsymbol{L}^T (\ddot{\boldsymbol{p}}_c + \begin{bmatrix} -cos(\alpha)r \\ -sin(\alpha)r \end{bmatrix} \cdot \ddot{\alpha} - \ddot{\boldsymbol{p}}_j) + 2\boldsymbol{L}^T \begin{bmatrix} sin(\alpha)\dot{\alpha}^2 r \\ -cos(\alpha)\dot{\alpha}^2 r \end{bmatrix} \tag{2.24}$$

By factorizing this into the form 2.8 from knowing system state $\boldsymbol{q} = \begin{bmatrix} \boldsymbol{p}_c^T & \alpha_c & \boldsymbol{p}_j^T \end{bmatrix}^T$, this yields:

$$\boldsymbol{A}\ddot{\boldsymbol{q}} = -2\boldsymbol{L}^T \begin{bmatrix} \boldsymbol{I}_{2x2} & \begin{bmatrix} -cos(\alpha)r \\ -sin(\alpha)r \end{bmatrix} & -\boldsymbol{I}_{2x2} & \boldsymbol{0}_{2x1} \end{bmatrix} \tag{2.25}$$

$$\boldsymbol{b} = 2\dot{\boldsymbol{L}}^T \dot{\boldsymbol{L}} + 2\dot{\boldsymbol{L}}^T \begin{bmatrix} sin(\alpha)\dot{\alpha}^2 r \\ -cos(\alpha)\dot{\alpha}^2 r \end{bmatrix} \tag{2.26}$$

With matrix $\boldsymbol{A}$ and vector $\boldsymbol{b}$ defined Eq. 2.10 can be used to find $\boldsymbol{Q}_c$. Then included to alter the unconstrained dynamics to satisfy the applied constraints. The final system dynamics can then be expressed by Eq. 2.28:

$$\boldsymbol{Q}_c = \boldsymbol{M}^{1/2} (\boldsymbol{A}\boldsymbol{M}^{-1/2})^+ (\boldsymbol{b} - \boldsymbol{A}\ddot{\boldsymbol{q}}_u) \tag{2.27}$$

$$\boldsymbol{M}\ddot{\boldsymbol{q}} = \boldsymbol{Q} + \boldsymbol{Q}_c \tag{2.28}$$

The left-hand side of Fig. 2.3 show the simulated system for the case where the described constraint is applied, and the trajectory is marked by the green dotted line. It can be compared to the unconstrained case on the right-hand side where the cube simply falls straight down. It should be mentioned that the acceleration of point $\boldsymbol{p}_j$ was kept to zero at all time, to represent a stationary point. Also, note from Eq. 2.25 and Eq. 2.26 that most of the complexity would be removed if $\boldsymbol{r}$ was zero, this is pointed out since it is exploited in chapter 4.

2.4.2 Moore Penrose Pseudo-Inverse

The Moore Penrose pseudo-inverse is a generalization of the inverse matrix. It has the convenient property that the pseudo-inverse is well defined for all matrices whose entries are real or complex numbers. A matrix $\boldsymbol{A}$ satisfying the following four criteria, known as the Moore Penrose conditions is by definition a pseudo-inverse matrix.

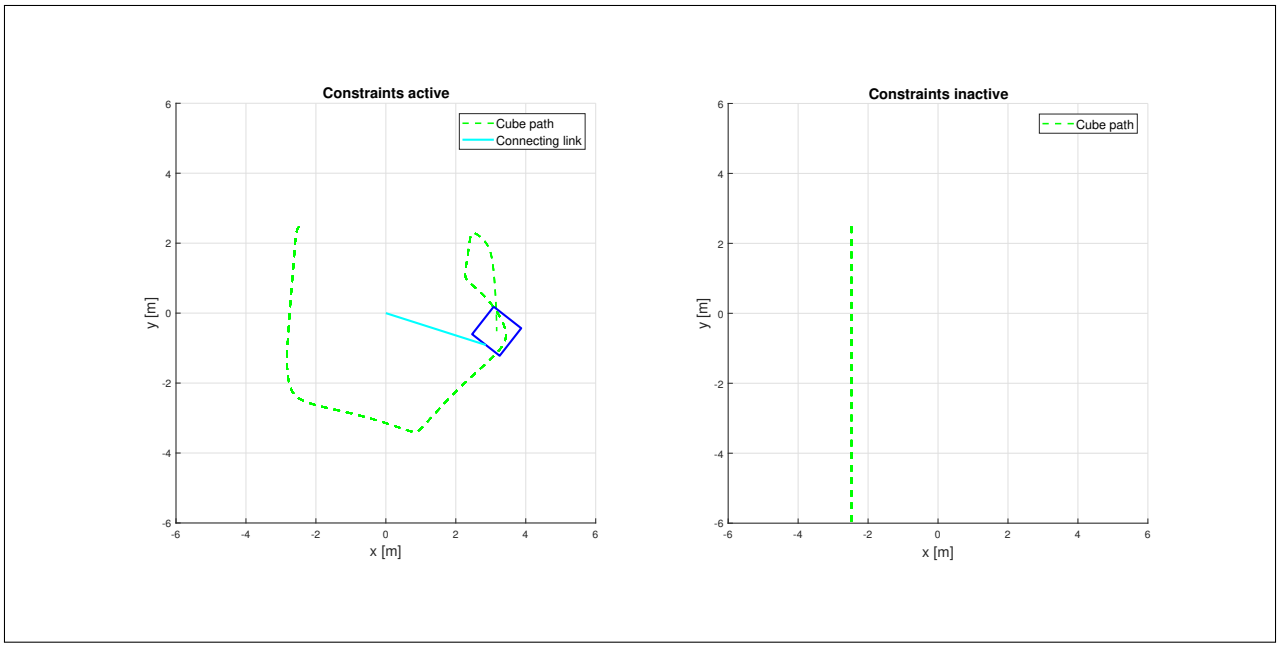


Figure 2.3: Comparison showing constrained and unconstrained behavior of cube c and point-mass j.

$$\boldsymbol{A}\boldsymbol{A}^{+}\boldsymbol{A} = \boldsymbol{A} \tag{2.29}$$

$$\boldsymbol{A}^{+}\boldsymbol{A}\boldsymbol{A}^{+} = \boldsymbol{A}^{+} \tag{2.30}$$

$$(\boldsymbol{A}\boldsymbol{A}^{+})^{*} = \boldsymbol{A}\boldsymbol{A}^{+} \tag{2.31}$$

$$(\boldsymbol{A}^{+}\boldsymbol{A})^{*} = \boldsymbol{A}^{+}\boldsymbol{A} \tag{2.32}$$

Where the * notation denotes the conjugate transpose. The Moore Penrose pseudoinverse can be used to find a least square solution for the Udwadia-Kalaba equation. This is equivalent to solving Gauss's principle of least constraints. An intuitive way of understanding this is that it finds the acceleration vector closest to the unconstrained acceleration vector which does not violate the constraints. A simple illustration of this can be seen in Fig. 2.4.

More in-depth explanations of the inner workings and applications of the Moore Penrose pseudo-inverse can be found here [26].

2.5 Positioning

To achieve decent navigation and system behavior, autonomous systems such as the vertical stack proposed here require decent position measurements. Cooperative UAV systems need a higher degree of precision than most other vehicle systems, as the close proximity flight has a low tolerance for error. For conventional vehicle navigation, the most widespread solution

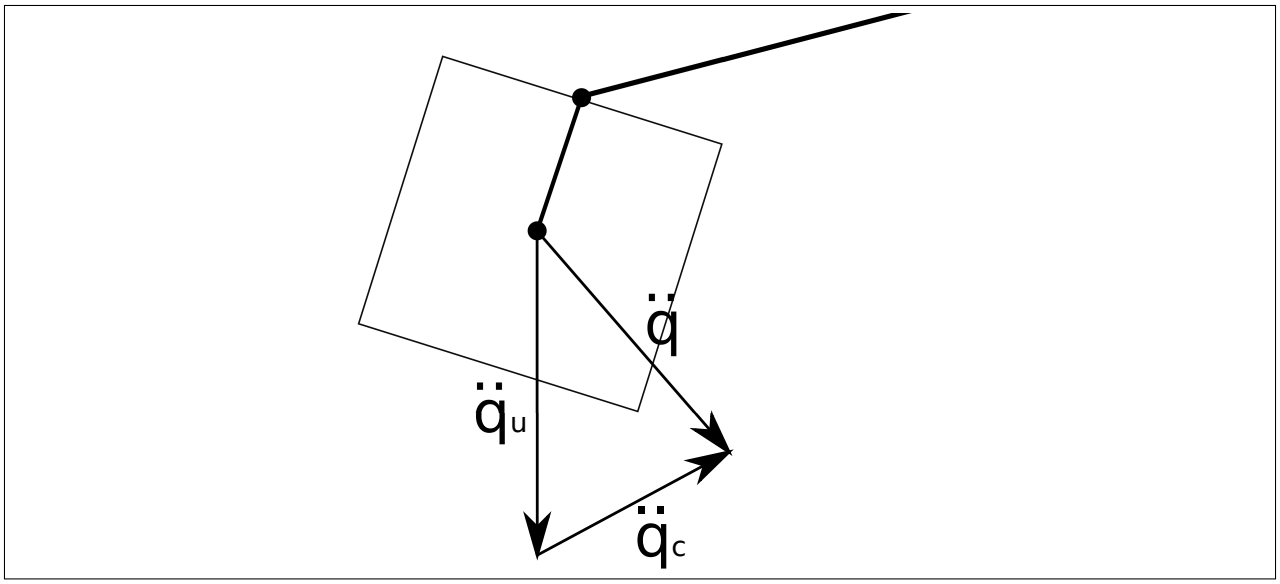


Figure 2.4: The unconstrained acceleration of the square body is altered.

is a global navigation satellite system (GNSS), examples of such systems being the well known global positioning system (GPS), and others such as the European Galileo, Russian GLONASS or the Chinese BeiDou. The high precision demand inherent in multi-UAV systems make it important, or necessary to consider alternatives to the conventional positioning systems. For this reason, a brief explanation of the concept of GNSS systems will be given here, and in turn, a technology known as Real-Time Kinematic (RTK) positioning will be detailed as well.

Fig 2.5 illustrates how the position of a vehicle can be estimated by interpreting data received from four or more satellites. The GNSS receiver onboard the vehicle receives the satellite signal in order to determine what is known as a pseudorange between itself and the satellite. By pseudorange it is underscored that it is an approximate range, as there are numerous sources of error making exact ranges difficult to find. Examples of these errors are:

- Ionospheric delay - A result of X and UV rays from solar Radiation is a layer in the atmosphere with a high ion and free electron density. This causes an irregular propagation delay which depends on the time of day and solar cycles and is hard to account for entirely.
- Satellite clock bias - Although the GNSS satellites are equipped with multiple atomic clocks which are very accurate, a small amount of clock drift can occur, this influences the pseudo range calculation.
- Multi-path issues - Depending on the receiver environment, be it valleys, trees or buildings, some GNSS signals will reflect off these objects, giving readings which don't represent the actual distance between the UAV and the sending satellite.

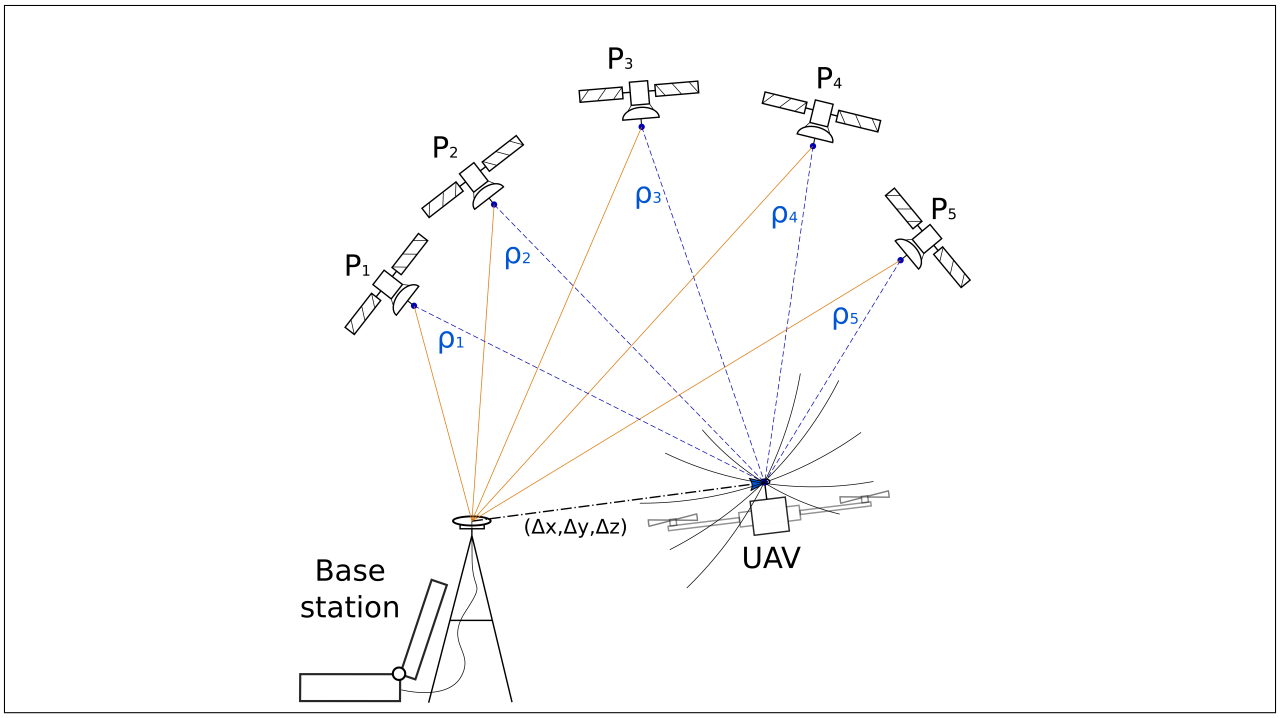


Figure 2.5: Conventional GNSS systems only include the satellites and the original receiver (UAV in this case). RTK improves the position estimate using a reference base station.

- Tropospheric delay - Varying levels of humidity, temperature, and pressure in this layer of the atmosphere contribute to the uncertainty of the travel distance of the signal.

Despite multiple systems being in place to reduce these problems, the conventional GNSS systems accurately measure position to approximately 3 – 5 meters according to GNSS receiver provider SwiftNAV [27].

The satellites in the GPS system using the L1 frequency band, emit a signal which will be used as an explanatory example. Fig. 2.6 is an altered figure copied from this whitepaper [28] by Ublox and displays a block diagram showing how the satellite signal is composed. The carrier signal is modulated to carry the signal from the two other blocks, first of which is the pseudo range noise (PRN) code generator which attaches a unique satellite identifier code to the signal. Next is the data generator block which transmits navigation messages such as satellite ephemerides, almanacs and the time of transmission. The ephemerides and almanacs are adjustment tables containing data such as the week number, satellite health status, clock corrections, and time-stamps. This explanation is of course very simplified, but from the description thus far we have the rough prerequisites to explain how the receiver estimates its position.

From the satellite signal content described in the previous paragraph, the receiver's task is to determine the pseudoranges, illustrated by Fig. 2.5. Using the knowledge of when the signal was

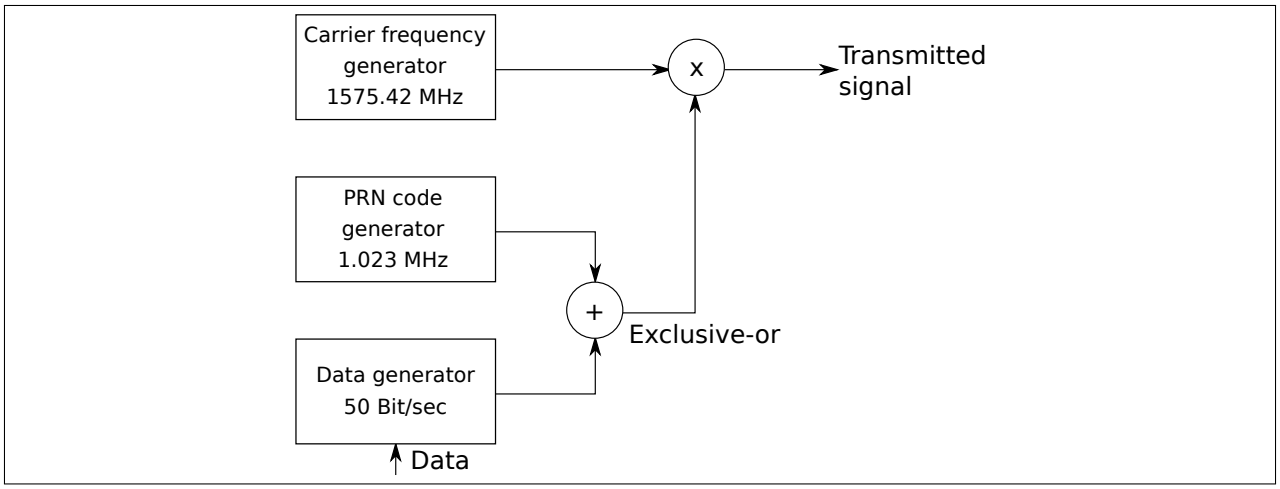


Figure 2.6: The signal sent from the satellites is a combination of multiple signals, modulated onto the $1572.42 MHz$ carrier signal.

sent, and its own internal clock it can determine the signal travel time. The book *GNSS–global navigation satellite systems: GPS, GLONASS, Galileo, and more* [29] gives the following model for the pseudo range.

$$Rc\Delta t + c\Delta\delta = \rho + \Delta\rho \tag{2.33}$$

In Eq. 2.33, c is the speed of light, the time of signal propagation is denoted by Δt, δ refers to the time difference between the satellite and receivers clocks. The result is the actual distance between the receiver and the satellite ρ, and range correction $\Delta\rho$. By finding this range between the receiver and enough satellites, the position estimate is in turn calculated by using a process known as trilateration [30]. Eq. 2.33 is included to illustrate the process of finding pseudo-ranges, as the work of trilateration and pseudo-range estimation is done using existing software and systems in this project.

By introducing Real-Time Kinematic differential corrections, the uncertainty surrounding the position estimate from the trilateration can be reduced greatly. This requires another receiver, which we refer to as the base station to be within range of the vehicle's receiver, and for the two to have a robust communication link. A rough maximum range is $20[km]$ for the L1 band, according to [31]. A very precise position estimate can be found by accounting for the individual signal cycles in the carrier signal, to do what is known as an integer ambiguity resolution (IAR). When the receiver knows the number of signal cycles between itself and the satellite, the pseudorange can be found with much greater accuracy. To summarize, the receiver can attain three levels of position estimates using the RTK system:

- **Single** - The communication with the base station does not yield useful improvement of the signal, and the position is estimated in the conventional GNSS manner.
- **Float** - The position estimate is improved through the differential correction with the base station, giving higher accuracy than the single solution.
- **Fix** - Integer ambiguities are resolved, and a high precision position estimate can be found.

Chapter 3

Drone Technology

The hardware components, system software and design choices made for the implemented drones is the focus of this chapter. The aim is to give a description of the UAVs which were created and explain the modules making up the complete system. Details on the interconnections between the various components are also offered, then the specifications of the drones are summarized so that they can be used in later simulations and discussions. Finally, a summary of the work done with and on the UAVs is given.

3.1 System Architecture

The experimental platform has been chosen to consist of two quad-rotor drones, as well as a ground control system. This was meant to be the minimal implementation of a multi-drone system which will give valuable insight into the dynamics of the vertical stack lifting scheme.

3.2 Single Drone System

A detailed description of the inner workings of the drones is in order. An overview of the modules which each drone relies upon is depicted in Fig. 3.1, and can be revisited to gain a good complete system level overview.

3.2.1 Drone Onboard Overview

The right side of Fig. 3.1 shows onboard components for a single drone. The Beaglebone Black Wireless (BBBW) is a companion computer where most of the modifications made to realize the proposed vertical stack formation are done. It is responsible for communication with the ground station and base station. The distinction between *ground station* being a personal com-

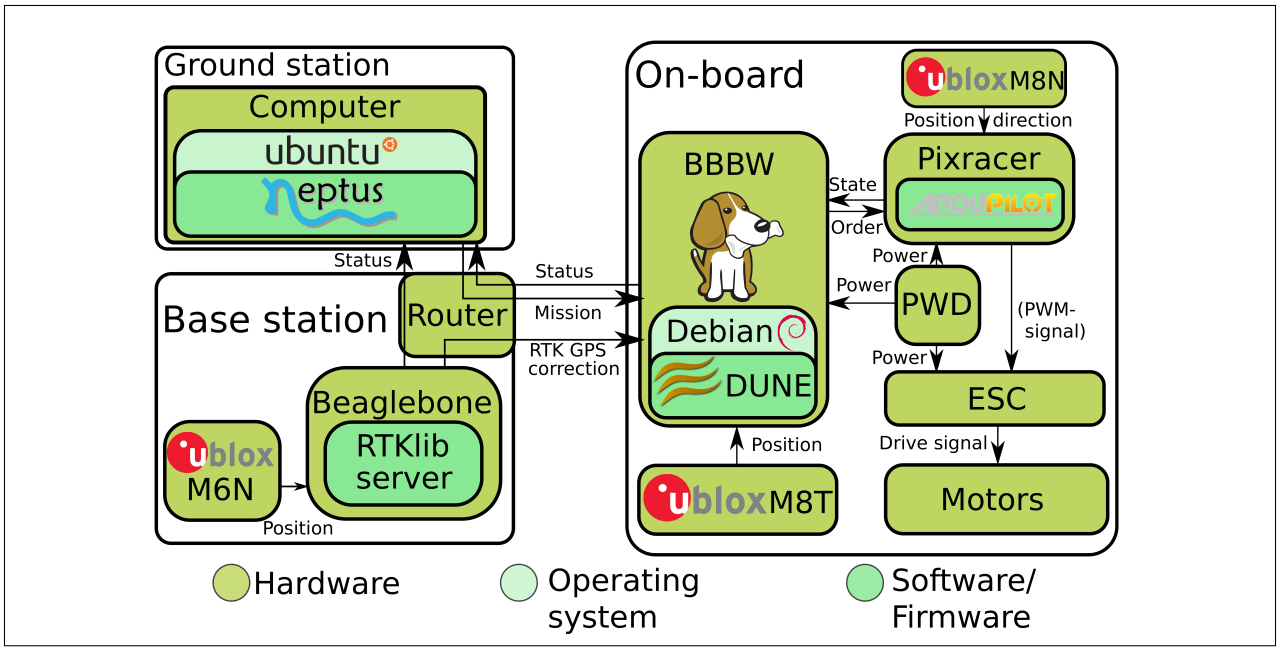


Figure 3.1: Overview of the modules of the drone platform.

puter, and the *base station* being the RTK reference point is noted to avoid confusion. The BBBW also fetches satellite data from the onboard Ublox RTK receiver and dispatches appropriate orders to the Pixracer. The Pixracer is a versatile autopilot, which attempts to execute the orders it receives from the Beaglebone. It actuates by sending control signals to the electronic speed controllers (ESC). More detail on each of these components will be given in the next sections.

3.2.2 Ground Control Overview

In the left part of Fig. 3.1, the two main parts of the stationary equipment are shown. The ground station acts as both a mission planner and an interface to keep system overview over all drones and is connected to the base station over WiFi. The general system can, therefore, be thought as a centralized network with the base station as the center, as illustrated by Fig. 3.2:

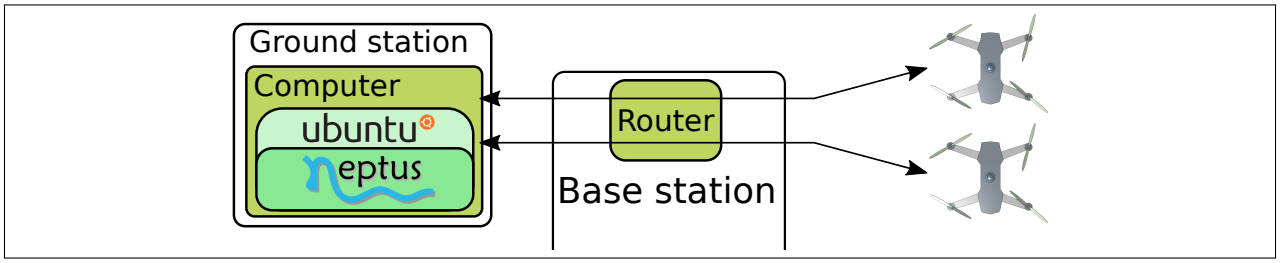


Figure 3.2: The base station contains a router which acts as the central node for the entire stack system.

Each individual drone in the stack is communicating through the base station router. At

this early experimental stage, this centralized communication was deemed sufficient, although modularity and robustness requirements are likely to demand a distributed communication solution if a larger stack was implemented. The base station is not only a communication node but also a necessary part of the RTK GNSS system, which makes it possible for the RTKLIB software running on the BBBW to compensate for various disturbances and uncertainties, as was explained in section 2.5.

3.3 Component Description

A more detailed explanation of the system components will now be given, to provide a more complete picture of what protocols and hardware was used to realize the system.

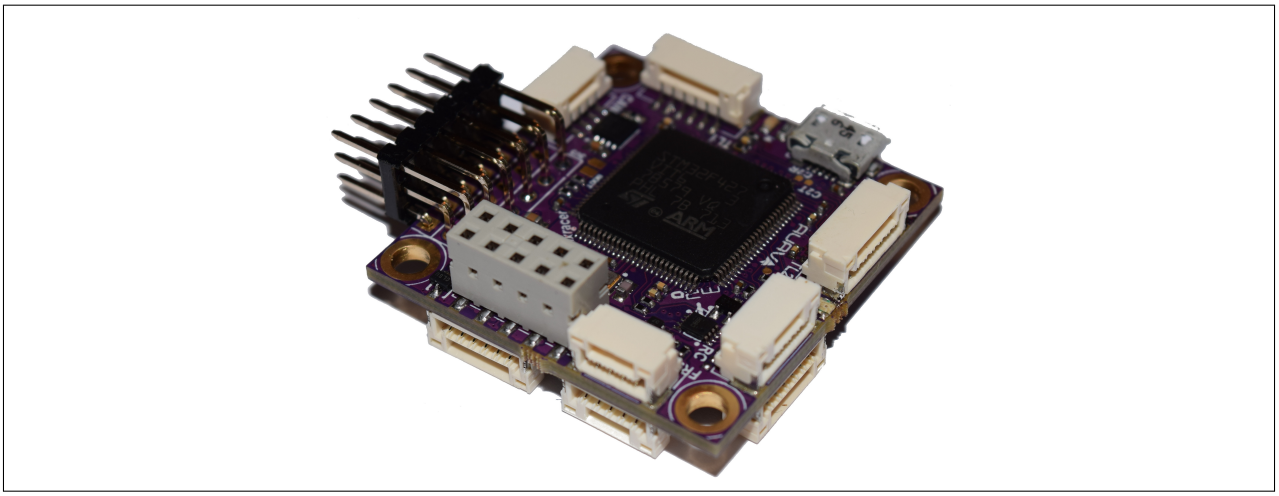


Figure 3.3: The Pixracer autopilot contains most of the sensors, and performs the low-level control.

3.3.1 Pixracer Autopilot

Acting as a low-level autopilot for the system is a Pixracer shown in Fig. 3.3, which was purchased from mRobotics [32]. It is very compact measuring $36 \times 36[mm]$, and contains several sensors which are absolutely necessary for autonomous flight, such as a gyroscope, accelerometer, magnetometer, and barometer. The Pixracer controls motor thrust by using a pulse width modulated (PWM) signal sent to the electronic speed controllers. The ESCs will, in turn, apply an appropriate level of power to the motors to generate the commanded thrust. A multi-rotor build called Arducopter [33], from the open-source software Ardupilot, runs on the Pixracer.

With the Pixracer in charge of low-level control, we can move control concerns up an abstraction level. The Arducopter software is open-source, and customizable, however for implementing the vertical stack system it was decided to follow precedence from the UAV-lab at

NTNU. For this reason, a companion computer Beaglebone Black Wireless was included. Communciation between the Pixracer and BBBW is done using a serial line (UART) from the Telem 2 port on the Pixracer to GPIO pins 22 and 24 on the BBBW. The communication over this UART connection is done using the micro air vehicle link (MAVLINK) protocol, offering many useful predefined packet structures. Further details on the MAVLINK protocol can be found in here [34].

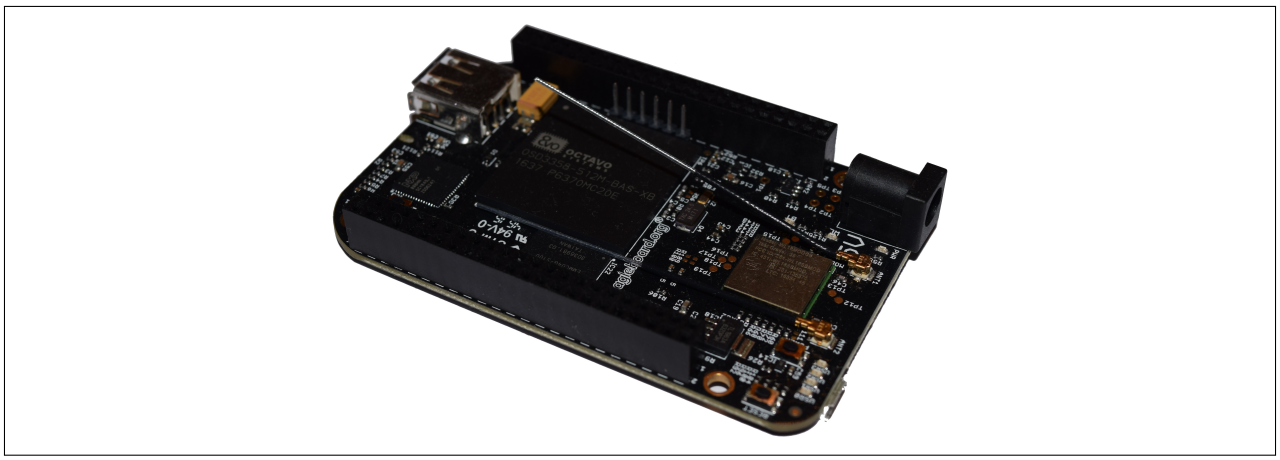
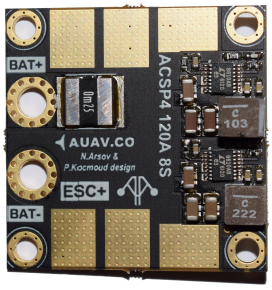


Figure 3.4: Image of the Beaglebone Black Wireless (BBBW).

3.3.2 Beaglebone Black Wireless

The Beaglebone Black Wireless [35] has dimensions $48.4 \times 63.6[mm]$, and the compact computer also includes two $50[mm]$ long antennae. The BBBW is a tiny development platform which in this case will run the operating system Debian 8.3. The main software which is used on the BBBW is a navigation and communication software with the recursive name DUNE (DUNE Unified Navigation Environment), explained further shortly. Also running on the BBBW is an open-source piece of software called RTKLIB, responsible for interpreting the data from the RTK GNSS receiver and the base station, and passing on a corrected position estimate to a corresponding DUNE task.

The Beaglebone also has wireless capabilities which are used to communicate with a standard commercial router in the base station over Wi-Fi 2.4 GHz. Also connected to this network is the ground station computer running the software Neptus, created to interact with DUNE using a protocol called the Inter-Module Communication (IMC) briefly explained later in this chapter. All devices on the network that is the ground station, base station and two UAVs were made to request static IP-addresses. This made it easy to access the devices over a secure shell terminal (SSH) as well as fetching of operation logs, and device inspection, by use of implemented shell scripts.



(a) Power distribution board acsp4.

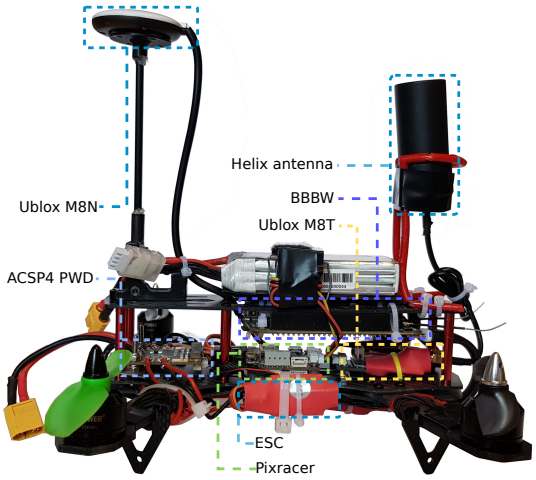
(b) A 1400 mAh 3 cell LiPo battery provides power to the system.

Figure 3.5: Images of an assembled drone, highlighting the location of the components.

3.3.3 Battery and Power distribution board

The drones are also equipped with a power distribution board, bought from mRobotics [36]. This changes the battery 12.6$[V]$ input voltage to the 5$[V]$ required by the Pixracer. It also supplies the input voltage for the ESCs, and like the Pixracer it also has the dimensions $36 \times 36[mm]$. The battery size of 1400$[mAh]$ was chosen to be lightweight, and sufficient for short test flights such as the ones planned for this system, thus the relatively small capacity should not be problematic.

(a) Image of an assembled drone.

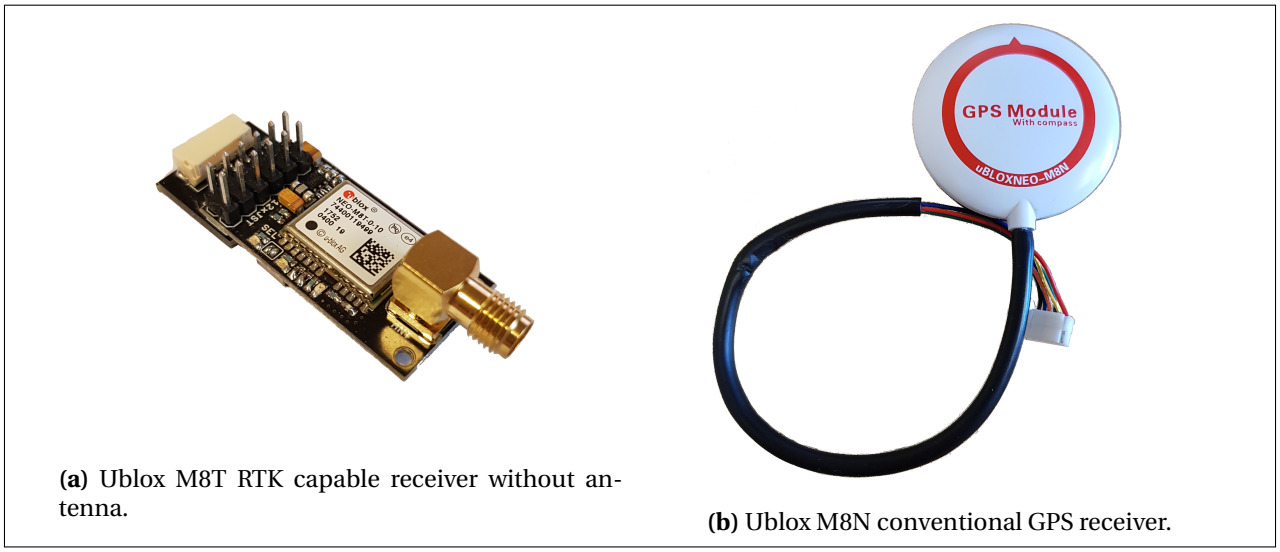


(b) Component overview.

Figure 3.6: Images of the components assembled into a single drone.

3.3.4 Frame And Motors

The implemented UAVs were intended to be as compact and practical as possible, yet sufficient for a proof of concept. Therefore a small commercial racing drone frame with model name QAV ZMR 250 as seen in 3.6a was chosen. It is a compact racing drone frame, which was delivered with four MT1806 2280kV Brushless Motors and four 12A Simonk ESCs. While its lifting capabilities are fairly limited, it gives an affordable and functional platform to explore the vertical stack concept.



(a) Ublox M8T RTK capable receiver without antenna.

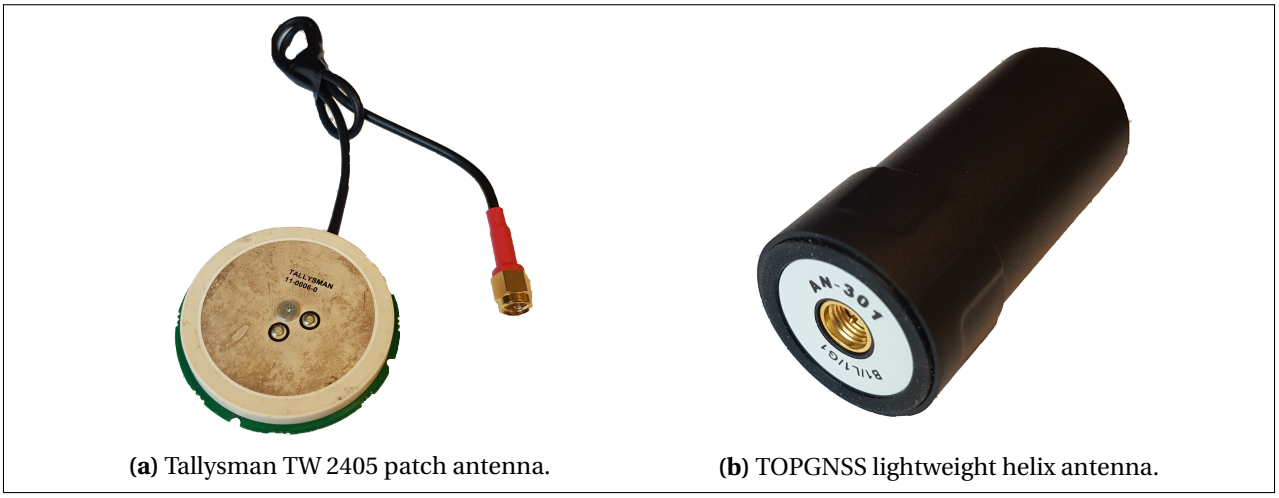
(b) Ublox M8N conventional GPS receiver.

Figure 3.7: Images of the two included GNSS receivers for the drone platform.

3.3.5 RTK And GPS System

Two separate GNSS systems are used for the drone system. The primary position sensor which we intend to rely most on is a Ublox NEO-M8T RTK receiver, as shown in Fig. 3.7a. The other receiver is a Ublox NEO-M8N conventional UAV GPS with included compass and is depicted in Fig. 3.7b. The reason for having both is twofold, first this provides a degree of redundancy as the NEO-M8N GPS is connected directly into the Pixracer, and second, the M8T offers RTK capabilities while the M8N includes an additional magnetometer.

In the case where the Beaglebone or RTK-receiver fails, the Pixracer should be able to rely on the connected Ublox NEO-M8N displayed in Fig. 3.7b to return to the original launch location and land, reducing or removing the harm in such an eventuality. Regarding position measurements, it should also be mentioned that DUNE merges the 10$[Hz]$ RTK position measurements with more frequent IMU data to enhance the precision of the position estimate further. This sensor data merging is done using an Extended Kalman Filter.



(a) Tallysman TW 2405 patch antenna. **(b)** TOPGNSS lightweight helix antenna.

Figure 3.8: Images of the two antennae considered for the system.

As precise positioning is exceedingly important for the system, two different antennae for the RTK receiver were considered. In Fig. 3.8a a Tallysman TW 2405 patch antenna is displayed, this has the benefit of being very efficient within a reasonably tight arc. The alternative shown in Fig. 3.8b is a TOPGNSS helix antenna designed to support GPS, GLONASS, and BEIDOU. It was acquired in order to consider an antenna with a wider signal reception arc. As the UAVs are quad-rotors and need to tilt to actuate horizontally, the wider arc might prove very useful or even necessary for the positioning system.

The base station depicted in Fig. 3.9 contains very similar components to that found on-board the drones. An exception is the antenna, which is significantly larger for the base station than the drones to improve signal reception. The third-party software RTKLIB runs on the Beaglebone Black in the base station too, but specifically a part of the software adapted to act as a reference point in an RTK system.

3.3.6 Physical Interconnection

The choice of using two rigid links instead of wire or a single rigid link as means of interconnection had two primary motivations. The system maintains the maneuverability of each drone relative to each other, and it avoids problems such as rope entanglement. Ideally, the links would be completely rigid, as well as very lightweight. Further the joints at which the links are connected should ideally give completely free rotation, the real implementation is in fact reasonably close to achieving these points.

Figure 3.9: The base station components resemble the drone components, and provides the RTK GNSS correction data.

The links themselves are hollow plastic cylinders which are quite stiff and durable, through these hollow cylinders a thin but strong wire is drawn. An image of the setup is shown in Fig. 3.10. The plastic cylinders are allowed to rotate around freely in all points of connection, by simply having the wire attached without too much tension. No automatic latching or connection mechanism for the links has been proposed at this stage of development due to time constraints, although it would absolutely be of interest in a future application.

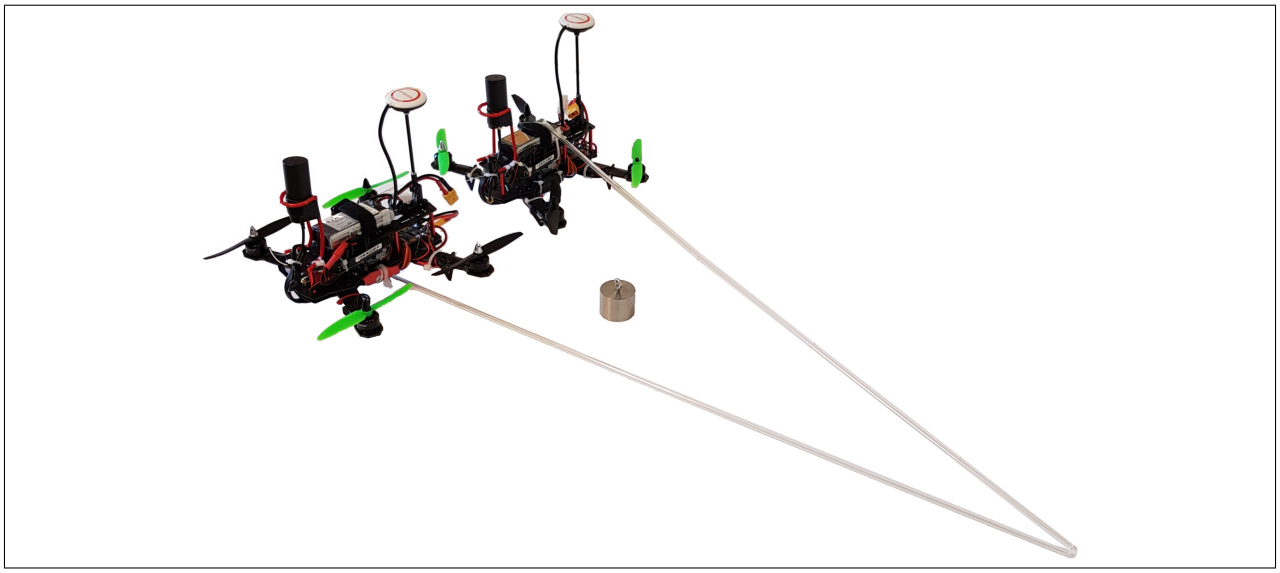


Figure 3.10: Picture of the two UAVs with the connecting links, and a small dense 0.3$[kg]$ payload.

The wire used is commercial fishing line, as it is very durable and has high tensile strength. The plastic rods are made of the material PMMA, a lightweight and relatively sturdy type of plastic. From arguments presented later in the report, two plastic rods were cut to $0.8[m]$, making link mass $m_{link} = 0.017[kg]$. In Fig. 3.10 the assembled system is displayed, together with the $0.3[kg]$ weight intended as a payload is showcased. Discussion regarding the properties of this completed system will be provided in chapter 6.

3.4 System Specification Summary

The complete system was measured and weighed, and its component specifications was found, such that these values can be used as starting parameters for the simulations performed. Table 3.1 can be inspected for a specification overview for the complete system. The width and length are measured between rotor centers, and the height is measured from lower frame plate, up to battery top. Details on how the maximum thrust was found are given later in this chapter.

Parameter	Variable name	measurement
Total mass	m_{drone}	$0.620[kg]$
Total max thrust	F_{max}	$10.32[N]$
Drone length	$2 \cdot l_x$	$0.155[m]$
Drone width	$2 \cdot l_y$	$0.203[m]$
Drone height	h	$0.060[m]$
Link mass	m_{link}	$0.017[kg]$
Link length	l_{link}	$0.8[m]$
Link mass per meter	ρ	$0.0213[kg]$

Table 3.1: Physical specifications of the implemented UAV system.

3.5 Software And Frameworks

3.5.1 DUNE

DUNE: Unified Navigation Environment is a C++ based software framework. It bridges the gap between the high-level formation orders which could be provided by Neptus on the ground station, to the relatively low-level autopilot desired state commands. The framework is centered around tasks which are separate program threads running concurrently to achieve specific desired purposes. Dune tasks can be roughly segmented into two main parts where the majority of the functionality of the framework can be found. The first of these are the consumer part, responsible in the task to receive data from external sources or other tasks, and interpret it accordingly. The second is the producer part, which is in charge of sending data to other tasks,

external hardware, or logging files. A few illustrative examples of tasks are given below.

- EstimatedState - Estimate the current state of the UAV.
- Voltage - Request battery voltage.
- AbortOnRadio - Abort maneuver if radio contact is lost.
- RTKGPS - Receiver, parse and filter the positions from RTKLIB.
- Ardupilot - Interface task to communicate with the Pixracer.

To reiterate the concept of producing and consuming parts, consider the EstimateState task. Its consumer segment takes in barometer data, GNSS data and IMU data, and its producer task makes the resulting state estimate available to other tasks and the logging functionality. When the DUNE tasks have been compiled, they can be initiated by a configuration file with appropriate parameters. This makes DUNE a modular and adaptable framework to implement real-time systems. In addition to this, DUNE offers user-friendly logging and can be made to communicate with a number of sensors. Further, it has also been used in multiple projects before within the UAV-lab at NTNU, also contributing in the choice of making use of this for the vertical stack system.

3.5.2 IMC

The Inter-Module Communication protocol (IMC) is a protocol used to communicate between the various DUNE-tasks running. This allows, for instance, an abort message from the aforementioned AbortOnRadio to be sent and picked up by the navigation and formation tasks running, so that they may properly abort their processes. IMC opens for user-defined messages as well, but for the vertical stack system, the predefined messages were found to be sufficient. IMC is also used to communicate with the Neptus running on the ground station.

3.5.3 Neptus

Neptus is a Java-based program which runs on the ground station, as is indicated by Fig. 3.1. It was created in conjunction with DUNE and gives the user a high-level mission based control system to use in real-time. It gives an overview of the status of all drones which make up the vertical stack and allows, for instance, the user to abort the current formation plan if needed. Neptus also offers review of the DUNE logs and exportation of flight logs to Matlab's *.mat* format among others.

3.5.4 Ardupilot

Ardupilot is an open-source autopilot software, which has multiple branches for a number of different autonomous vehicles. The one used for the implemented test platform is Arducopter and is flashed onto the Pixracer memory. Arducopter handles the low-level control and allows the user to change operational modes for the Pixracer with a switch. This switch can, for instance, change the Arducopter mode between *guided* where DUNE should be in control, to *alt-hold*, where the user can guide a drone with relative ease to a safe landing using a manual radio controller. Ardupilot offers detailed logging of the sensors housed within the Pixracer, and other functionality such as Emergency stop of motors, return to launch, and control parameter auto-tuning.

3.5.5 RTKLIB

RTKLIB is an open source program package for implementation of RTK positioning; it contains functionality which is of interest to our system. The program rtkrcv is one part of RTKLIB and is a program running on the onboard BBBW which continuously "listens" to data on a fixed port with the Transmission Control Protocol (TCP). The data received over TCP comes from the RTK receiver in the ground station, as has been touched upon earlier. Rtkrcv also fetches data from the onboard Ublox RTK receiver, attempts to improve the position estimate as detailed in section 2.5, and makes the position solutions available to DUNE, together with time stamp, solution type and a number of other tags.

3.6 Evaluating The Test Platform

3.6.1 Measuring Downwash Severity

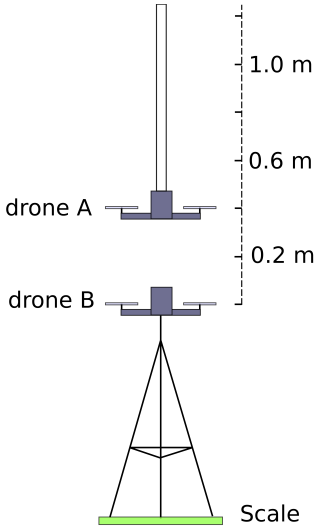
One of the central problems introduced in the introduction expected to be encountered with the vertical stack is the effect of the downwash from the upper drones. While several different approaches to minimize this influence will be discussed later in this thesis, the first and perhaps most crucial design choice is the vertical distance between the drones in the stack. To make a well-founded decision for the length of the links interconnecting the drones, an experimental setup was created to roughly map the influence of the downwash.

A conceptual drawing and an actual image of the experimental setup can be seen in Fig. 3.11a and Fig. 3.11b respectively. After having reset the weight measurement scale such that initial weight upon the scale is zero, the target is to measure how much thrust the lower drone can generate under various conditions. With the displayed setup, the test procedure was as

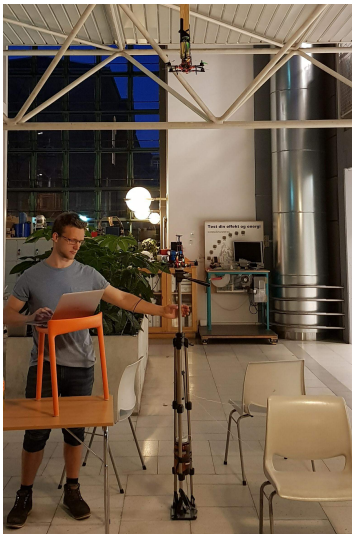
follows:

- 1: Measure maximum thrust for the lower drone, without external influence.
- 2: With top drone 0.4m above the lower drone, measure maximum thrust during 5-second motor burst of both drones.
- N: Repeat step 2 with 0.2m greater distance between drones.

A few more details should be pointed out regarding this test setup. First of all the lower drone is placed 1.5m above ground level, in order to reduce the ground effect. Second that the lower tripod, as well as the scale itself, is sure to influence the thrust of the lower drone, as airflow is hindered. Regardless the approach was deemed acceptable because the goal is to map the relative efficiency loss from the influence of the upper drone, not the absolute force loss. The results will be shown in chapter 5 and discussed further in later chapters.



(a) Concept illustration of downwash test setup.



(b) Image showing the implemented test setup.

Figure 3.11: Test setup to evaluate the aerodynamic influence between drones.

3.6.2 Positioning And Sensors

The effectiveness of the RTK receivers position measurements and its susceptibility to change of drone angle with various antennas was also tested. First, a drone was placed with the antenna of interest in a location with unobstructed access to satellites, and measurements were made. The location chosen was the rooftop of the department of engineering cybernetics NTNU. The drone was then tilted to 0°, 15°, 30° and 45° northwards and a 5-minute session of gathering position solutions was done for each angle.

One the values used to evaluate and compare the results was the variance of the distance from the average sample position. The sample variance was calculated for N samples, with the following equation, where the $\bar{bar}$ notation indicates average and n, e, u is north, east, up respectively.

$$D_i = \sqrt{(n_i - \bar{n})^2 + (e_i - \bar{e})^2 + (u_i - \bar{u})^2} \tag{3.1}$$

$$var = \frac{\sum (D_i - \bar{D})^2 {}_{i=1}^{i=N}}{N-1} \tag{3.2}$$

After determining which antenna gave superior behavior, the position estimate reached with the Extended Kalman Filter in DUNE was also evaluated. Also of interest to the project was the altitude estimate reached by the system. Because the accuracy of altitude measurements relying on barometer readings might be significantly affected by the rotor downwash from the drone above, meaning a greater reliance on GNSS measurements could be needed.

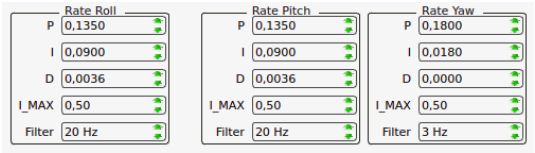
3.6.3 UAV Progress

In preparation for the two drone vertical stack to be realized, a number of intermediary goals were set as milestones on the way to a complete functioning vertical stack system. This section is intended to briefly summarize how progress was made with regards to the various goals, and provide a basis for later discussion of the implemented UAVs.

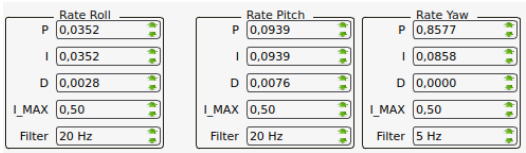
- Assembly of two drones.
  The completed drone systems were assembled, containing all of the components detailed in this chapter.

- Manual flight and platform tuning.
  Manual flight was tested in multiple modes, *altitude-hold* was found to be most convenient and pilot-friendly. By making use of Arducopters autotune functionality, the UAVs were made to observe multiple impulse responses acting on pitch, yaw and roll angles, and adapt their control parameters accordingly, as seen from comparing Fig. 3.12a with Fig. 3.12b.

- Establishing Pixracer and BBBW communication.
  Custom cables were made to connect the Pixracer with the BBBW, and settings such as



(a) PID parameters before the autotune process.



(b) PID parameters after the autotune process.

Figure 3.12: Comparison of PID parameters before and after tuning.

General Purpose In/Out (GPIO) mode, UART baud rate, and Telem port mode were adjusted making data transmission using UART possible.

- Make Ublox RTK receiver communicate with BBBW.
  Preexisting DUNE and RTKLIB systems had been made to function within the Glued operative system (OS), and had to be adapted, for running on the Debian OS on the BBBW. The Glued operative system is a slim variant of Linux, only containing the bare necessities for vehicle operation. However, no preexisting work showing the use of BBBW wireless functionality in the Glued OS was found during the project. It was therefore seen as risky for communication to proceed with Glued, making Debian the OS of choice, and making adaptations necessary. The configuration of the RTK receiver also had to be done.

- Implement and test log fetching scripts.
  Bash scripts for acquiring the logs from onboard the UAVs were written, so as to gather, fetch and store logs onto the base station computer and then to remove the logs onboard the drones in an orderly manner. Corresponding data parsing and plotting programs were also written.

- Verify decent position with both GNSS systems.
  Susceptibility to change of antenna type and drone angle was tested as was described in section 3.6.2, the results can be found in chapter 5.

- Attempt complete system running, and subsequently, position loiter.
  All systems illustrated in the overview in Fig. 3.1 were activated simultaneously and were found to function during flight as well. Various difficulties were encountered at this point which will be given proper attention in the discussion in chapter 6.

- Send and execute missions from Neptus.

Separate vehicle definitions had to be implemented, and fixed ip-addresses had to be arranged for the UAVs to be recognizable for the ground station. Missions were successfully sent, and executing was started by the UAV.

- Activate and test geofence RTL.
  For safer autonomous flight, a strict geofence of the minimum radius acceptable for Arducopter, 25$[m]$ was activated and tested, with varying degree of success.

Chapter 4

Simulation Methodology

In this chapter, the aim is to explain how an efficient mathematical model for the vertical stack system was derived by defining unconstrained dynamics, and then apply constraints to the model. After this, it is shown how multiple levels of control systems have been designed to give reliable behavior for the stack. The chapter also aims at presenting the method of simulation, with extensions to the model and controller complexity. The extensions are included so that the downwash influence between drones can be considered, together with approaches to compensate against this influence.

4.1 System modeling

The motivation for having a model describing the dynamics of the vertical stack is initially to be able to safely test formation control schemes, as well as single drone control within the stack. For future research, a reasonable system model would be essential for tasks such as path planning as well as robustness analysis for the formation.

Unconstrained Dynamics

Although the previous model which was summarized in section 2.3 showed decent behavior for two drones, a central goal of the vertical stack approach is to become a modular and scalable solution. For this reason, modeling of a high number of connected drones must be made possible for the model to be genuinely applicable. Simply extending the previous model shown in section 2.3 turned out to be very cumbersome, as the increasing complexity quickly slowed down simulations. In response to this issue, the Udwadia-Kalaba approach to constraint system motion detailed in section 2.4.1 was chosen. This alternative approach was inspired by the slung load work which can be found in a doctoral dissertation written by Kristian Klausen [37], concerning cooperative multi-drone lifting.

The complete previous mathematical derivation can be seen in B for comparison. Derivation of the vertical stack model used for this thesis begins with a finding the unconstrained dynamics of the system, much like what was shown in the example in section 2.4.1. First, we consider UAV i, which is seen as a rigid body with uniform density and has dynamics which can be described using Eq. 4.2, based on Eq. (3.42) in the book *Handbook of Marine Craft Hydrodynamics and Motion Control* [22], concerning the equations of motion of rigid bodies.

$$\boldsymbol{\nu}_i = \begin{bmatrix} \boldsymbol{v}_{b,i}^b \\ \boldsymbol{\omega}_{b,i}^b \end{bmatrix} \tag{4.1}$$

$$\boldsymbol{M}_i \dot{\boldsymbol{\nu}}_i + \boldsymbol{C}(\boldsymbol{\nu}_i)\boldsymbol{\nu}_i = \boldsymbol{\tau}_i \tag{4.2}$$

It should be pointed out that $\boldsymbol{\nu}_i$ is body relative velocity for drone i. For this reason, the transformation which was described in Eq. 2.4 is used when transitioning to Euler angle rates and velocities in $\{n\}$ frame.

$$\boldsymbol{M}_i = \begin{bmatrix} \boldsymbol{M}_{translation,i} & \\ & \boldsymbol{I}_{rotation,i} \end{bmatrix} = \begin{bmatrix} m_i & & & & & & \\ & m_i & & & & & \\ & & m_i & & & & \\ & & & I_{x,i} & & & \\ & & & & I_{y,i} & & \\ & & & & & I_{z,i} & \end{bmatrix} \tag{4.3}$$

The matrix $\boldsymbol{M}_i$ in Eq. 4.2 is the inertia and mass matrix for UAV i, as shown in Eq. 4.3. As for the Coriolis matrix $\boldsymbol{C}(\boldsymbol{\nu})_i$, since we operate with respect to inertial frames, the term becomes zero. Next we split the external force vector $\boldsymbol{\tau}_i$ into multiple terms to make its origin clear. This yields Eq. 4.4

$$\boldsymbol{M}_i \dot{\boldsymbol{\nu}}_i = \boldsymbol{\tau}_g + \boldsymbol{\tau}_d + \boldsymbol{\tau}_{in} \tag{4.4}$$

Where $\boldsymbol{\tau}_g$ is the gravity force vector, $\boldsymbol{\tau}_d$ is the air resistance vector, and $\boldsymbol{\tau}_{in}$ is the input force vector, essentially created by the UAV actuating using its rotors. Eq. 4.4 can also be written in the following form.

$$\boldsymbol{M}\dot{\boldsymbol{\nu}}_i = \begin{bmatrix} \boldsymbol{M}_{translation,i} & \\ & \boldsymbol{I}_{rotation,i} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{v}}^b_{b,i} \\ \dot{\boldsymbol{\omega}}^b_{b,i} \end{bmatrix} = \begin{bmatrix} \boldsymbol{F}_{g,i} \\ \boldsymbol{0}_{3\times 1} \end{bmatrix} - diag(\boldsymbol{d}_i) \cdot diag(\boldsymbol{\nu}_i)|\boldsymbol{\nu}_i| + \begin{bmatrix} \boldsymbol{f}^b_{b,i} \\ \boldsymbol{m}^b_{b,i} \end{bmatrix} \tag{4.5}$$

Where the input torque and force vectors are $\boldsymbol{m}^b_{b,i}$ and $\boldsymbol{f}^b_{b,i}$ respectively. The function $diag(\boldsymbol{a})$ creates a square matrix with the elements of vector $\boldsymbol{a}$ along the diagonal, and 0 elsewhere. The vector $\boldsymbol{d}_i$ corresponds to the air resistance of UAV i in $\{b\}$ frame.

$$\boldsymbol{f}^b_{b,i} = \boldsymbol{R}^n_b(\boldsymbol{\Theta}_{n,i}) \begin{bmatrix} 0 & 0 & F_i \end{bmatrix}^T \tag{4.6}$$

$$\boldsymbol{m}^b_{b,i} = \begin{bmatrix} K_i & M_i & N_i \end{bmatrix}^T \tag{4.7}$$

Where K_i, M_i, N_i are the torque components for drone i around b_x-b_y-b_z respectively, and F_i is the force magnitude resulting from the total drone thrust being rotated to have effect in $\{b_z\}$ direction. Next the gravity force is rotated to body frame to suit Eq. 4.7.

$$\boldsymbol{F}_{g,i} = \boldsymbol{R}^n_b(\boldsymbol{\Theta}_{n,i}) \begin{bmatrix} 0 & 0 & m_i \cdot g \end{bmatrix}^T \tag{4.8}$$

$$\tag{4.9}$$

The model also makes use of point-masses as the points of connection between the rigid links. The payload is also represented by a point-mass, only with adjusted air resistance and mass. The dynamics of these point-masses is simple, but for completeness stated explicitly in Eq. 4.10, before any constraint forces have been accounted for in the dynamics.

$$m\ddot{\boldsymbol{p}}^n_{b,i} = \boldsymbol{F}_{g,i} - diag(\boldsymbol{d}_i) \cdot diag(\boldsymbol{v}^b_{b,i})|\boldsymbol{v}^b_{b,i}| \tag{4.10}$$

Since the UAV dynamics are stated in Eq. 4.5 and point-mass dynamics in Eq. 4.10, all bodies in the vertical stack have well defined unconstrained behavior, only the input vector $\boldsymbol{\tau}_{in}$ for the drones remains undefined.

4.2 System Control

4.2.1 Inner Control System

With a compact model for quadrotor and point-mass dynamics in place, an examination the inner controller implemented to determine the input vector $\boldsymbol{\tau}_{in}$ to control the UAV behavior can be made. The inner controller has three segments in addition to the UAV plant, as illustrated by Fig. 4.1. A commanded force, in the form of the vector $\boldsymbol{\tau}_{cmd}$ is entered into the controller, which the controller then actuates to achieve. As is inherent in the quad-rotor platform, force is only applied in $\{b_z\}$ direction, thus the command force is translated to a desired angle $\boldsymbol{\Theta}_{cmd}$. This desired angle is passed onto the angle controller, and subsequently angular velocity controller. Both of these are conventional PID controllers. In parallel to this, the magnitude of the quad-rotor thrust F is set as the norm of the commanded thrust, and clamped to account for the quad-rotor limitations:

$$F = \begin{cases} |F_{max}| & if \quad |\boldsymbol{\tau}_{cmd}| > F_{max} \\ 0 & if \quad |\boldsymbol{\tau}_{cmd}| < 0 \\ |\boldsymbol{\tau}_{cmd}| & otherwise \end{cases} \tag{4.11}$$

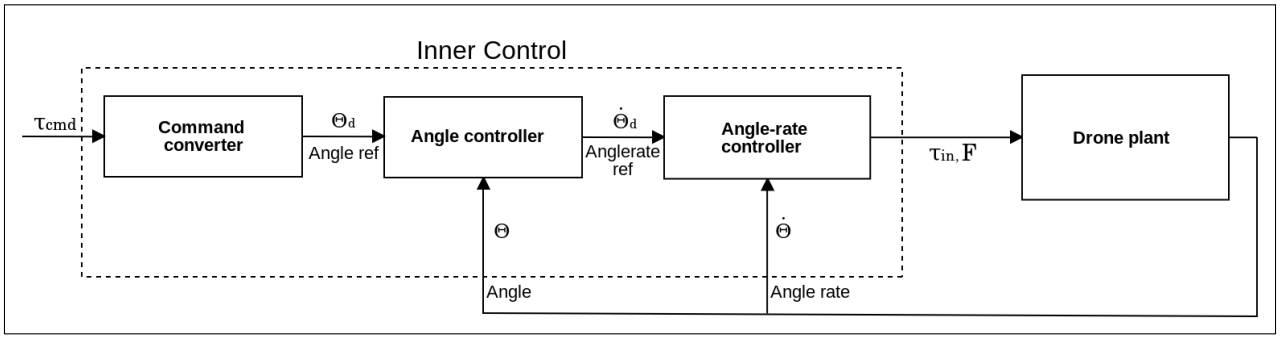


Figure 4.1: Block diagram overview of the inner control system.

The conversion from command force to the desired angle is done as shown in Eq. 4.12 - 4.15:

$$\boldsymbol{\tau}_{cmd} = \begin{bmatrix} x_{cmd} & y_{cmd} & z_{cmd} \end{bmatrix}^T \tag{4.12}$$

$$\theta_{cmd} = -acos(\frac{-z_{cmd}}{|\boldsymbol{\tau}_{cmd}|}) \tag{4.13}$$

$$\psi_{cmd} = atan2(y_{cmd}, x_{cmd}) \tag{4.14}$$

$$\phi_{cmd} = 0 \tag{4.15}$$

This gave decent behavior, yet the desired yaw angle ψ_{cmd} presented a problem when going from $-\pi$ to π, an inherent issue from the use of the $atan2$ function. This was resolved by a simple modulo based function, essentially choosing the shortest path for the yaw aspect of the angle controller.

$$\psi_{error} = mod(\psi_{cmd} - \psi + \pi,\ 2\pi) - \pi; \tag{4.16}$$

$$\theta_{error} = \theta_{cmd} - \theta \tag{4.17}$$

$$\phi_{error} = \phi_{cmd} - \phi \tag{4.18}$$

$$\tag{4.19}$$

The inner control could likely be made more elegant and given a greater area of validity by using quaternion based control as is described in for instance [38]. However, the described controller was found to behave reasonably well for all situations during this research, as will be shown in chapter 5, and was therefore not altered further.

Vertical Stack Model

Part of the unnecessary model complexity in the previous model described in appendix B came from the two connecting links being modeled as separate rigid bodies. For this reason, the links are in the new model replaced by a single point-mass joint, where the links meet. The point-mass joint was given the mass of the two links combined. This simplification was deemed appropriate as the links are light-weight compared to the drones. Not only does this reduce the number of states needed to describe the system fully, but also makes the constraint calculations much faster as will be shown in the results.

The model can, therefore, be illustrated by Fig. 4.2. There are a total of S bodies, N of which are UAVs, in the illustrated case, this means $S = 5$ and $N = 2$. The bodies are indexed from top to bottom as shown. For the reader's convenience we reiterate some notation, for body i in the stack, the position and angle relative to $\{n\}$ frame are referred to as $\boldsymbol{p}^n_{b,i}$ and $\boldsymbol{\Theta}_{n,i}$ respectively.

Constrained System Dynamics

For later use, we gather states and define a composite state vector $\boldsymbol{q}$, and a composite position vector $\boldsymbol{q}_p$ for all S bodies in the vertical stack as follows in Eq. 4.21:

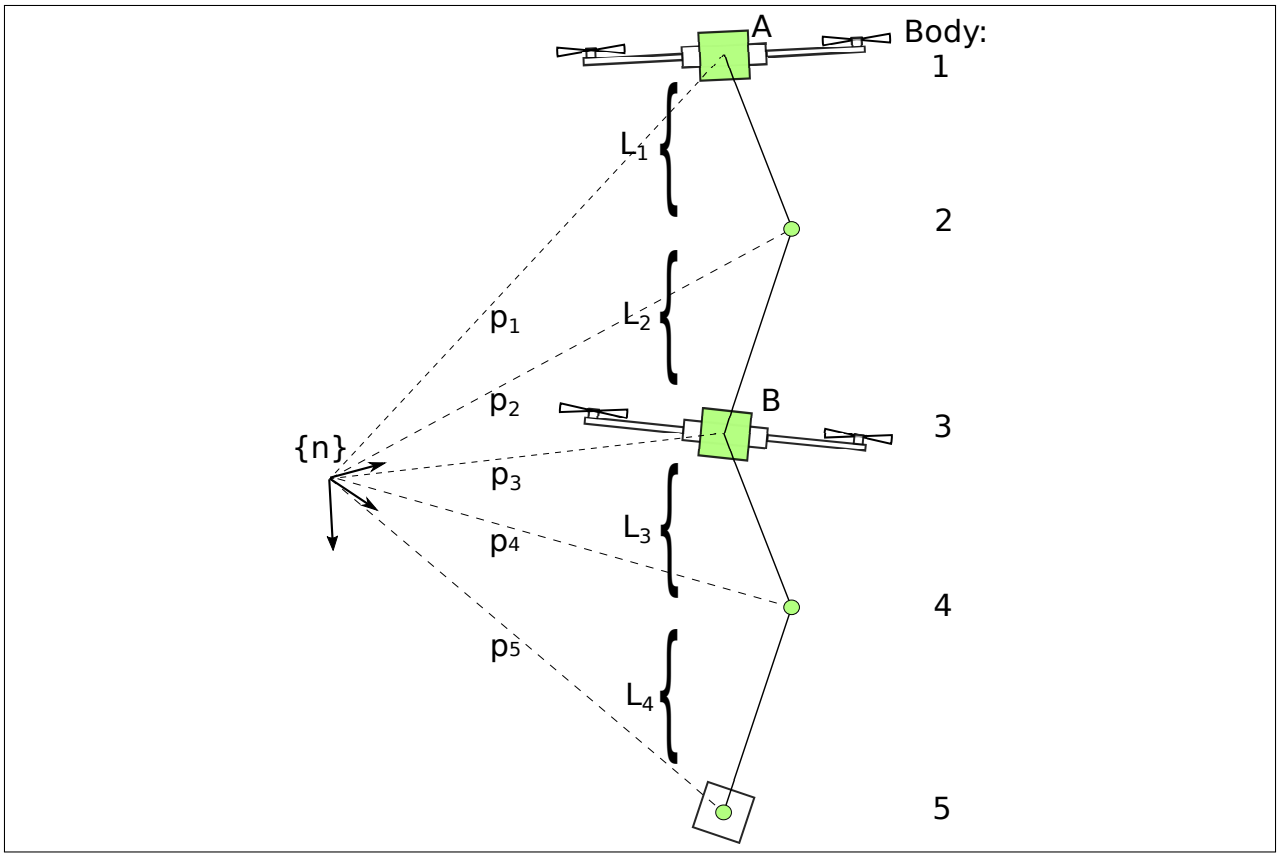


Figure 4.2: S bodies are included in the vertical stack, including UAVs, connecting joints, and a payload.

$$\boldsymbol{\eta}_i = \begin{cases} \begin{bmatrix} \boldsymbol{p}^n_{b,i} \\ \boldsymbol{\Theta}_{n,i} \end{bmatrix}, & if \quad UAV \\ \boldsymbol{p}^n_{b,i}, & if \quad point-mass \end{cases} \tag{4.20}$$

$$\boldsymbol{q} = \begin{bmatrix} \boldsymbol{\eta}_{n,1} \\ \boldsymbol{\eta}_{n,2} \\ \vdots \\ \boldsymbol{\eta}_{n,S} \end{bmatrix} \in \mathbb{R}^{S\cdot 3+N\cdot 3} \qquad \boldsymbol{q}_p = \begin{bmatrix} \boldsymbol{p}_{n,1} \\ \boldsymbol{p}_{n,2} \\ \vdots \\ \boldsymbol{p}_{n,S} \end{bmatrix} \in \mathbb{R}^{S\cdot 3} \tag{4.21}$$

Next, we intend to solve the Udwadia-Kalaba equation to find the constrained system motion of these S bodies. The Udwadia-Kalaba equation was introduced and explained in section 2.4.1. As the simulation system was intended to be efficient, the constraint model places all points of connection in the mass center of the UAVs. We index the constraint vectors of connecting links from top to bottom as link $\boldsymbol{L}_j$, from $j = 1$ up to $j = S-1$. Also note from Fig. 4.2

how the two bodies interconnected by $\boldsymbol{L}_j$ can be indexed by $i = j$ and $i = j+1$. This yields the following constraint calculations.

$$\boldsymbol{L}_j = \boldsymbol{p}^n_{b,j} - \boldsymbol{p}^n_{b,j+1} \tag{4.22}$$

$$\dot{\boldsymbol{L}}_j = \boldsymbol{v}^n_{b,j} - \boldsymbol{v}^n_{b,j+1} \quad j = 1,2,\ldots,S-1 \tag{4.23}$$

$$\ddot{\boldsymbol{L}}_j = \dot{\boldsymbol{v}}^n_{b,j} - \dot{\boldsymbol{v}}^n_{b,j+1} \tag{4.24}$$

In a similar fashion as the constraint calculations presented in chapter 2, Eq. 2.17, the distance between the points are kept constant, leading to the constraint equation g_j for link j,.

$$\ddot{g}_j = 2\ddot{\boldsymbol{L}}_j^{\,T}\boldsymbol{L}_j + 2\dot{\boldsymbol{L}}_j^{\,T}\dot{\boldsymbol{L}}_j = 0 \tag{4.25}$$

This yields the matrix $\boldsymbol{A}_j$ and vector $\boldsymbol{b}_j$ after insertion and factorization to be quite simple, due to the choice of attaching links in drone mass-centers.

$$\boldsymbol{A}_j = 2\boldsymbol{L}_j^T \begin{bmatrix} \boldsymbol{0}_{3\times 3\cdot(j-1)} & \boldsymbol{I}_{3\times 3} & -\boldsymbol{I}_{3\times 3} & \boldsymbol{0}_{3\times 3\cdot(S-j-2)} \end{bmatrix} \tag{4.26}$$

$$\boldsymbol{b}_j = -2\dot{\boldsymbol{L}}_j^T\dot{\boldsymbol{L}}_j \tag{4.27}$$

As this only relates to a single constraint, it is necessary to iterate over all $S-1$ constraints, constructing matrix $\boldsymbol{A}$ and vector $\boldsymbol{b}$ one row at the time.

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_1 \\ \boldsymbol{A}_2 \\ \vdots \\ \boldsymbol{A}_{S-1} \end{bmatrix} \in \mathbb{R}^{3\cdot(S-1)\times 3\cdot S} \qquad \boldsymbol{b} = \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \\ \vdots \\ \boldsymbol{b}_{S-1} \end{bmatrix} \in \mathbb{R}^{3\cdot(S-1)} \tag{4.28}$$

These are inserted into Eq. 4.29, where $\boldsymbol{q}_p$ is the gathered position vector from Eq. 4.21. Which in turn yields the effect of the constraint forces $\boldsymbol{Q}_t$.

$$\boldsymbol{Q}_t = \boldsymbol{M}^{1/2}(\boldsymbol{A}\boldsymbol{M}^{-1/2})^+(\boldsymbol{b} - \boldsymbol{A}\ddot{\boldsymbol{q}}_p) \in \mathbb{R}^{S\cdot 3} \tag{4.29}$$

Currently $\boldsymbol{Q}_t$ only concerns translation as indicated by the t subscript, thus it is necessary to change the vector by adding $\boldsymbol{0}_{1\times 3}$ after the constraint force vectors acting upon UAVs to rep-

resent zero effect on the UAV angles. The result is a stretched constraint force vector $\boldsymbol{Q}_c$, which has the same dimensions as state vector $\boldsymbol{q}$ and can be used to alter the previous unconstrained force vector $\boldsymbol{Q}_u$. Finally this gives the constrained vertical stack dynamics by insertion into Eq. 4.30:

$$\boldsymbol{M}\ddot{\boldsymbol{q}} = \boldsymbol{Q}_u + \boldsymbol{Q}_c \in \mathbb{R}^{(S+N)\cdot 3} \tag{4.30}$$

The example of Udwadia-Kalaba shown in section 2.4.1 illustrates how the method could be applied if the links were not attached in the drone mass centers. The effect is a torque acting upon the angle of the attached object. The reasons for not including this aspect of the model was to narrow the focus of the assignment and keep the model efficient and extendable to a high number of cooperating UAVs.

4.2.2 Formation Control

The need for a formation control system for the vertical stack is in some sense obvious, but the requirements should be explicitly stated in order to clarify what the formation system is expected to achieve.

- Target formation reach - The system should reach and maintain the target relative UAV positions, from a wide array of starting conditions.
- Formation velocity control - It should be possible to command the formation to reach target velocities, while maintaining the overall formation structure.
- Formation communication robustness - To provide a system robust in terms of communication, the formation system should ideally allow for not all UAVs communicating directly with all other UAVs but still giving acceptable behavior.

To achieve these goals, a passivity-based approach to cooperative control design presented in *Cooperative control design: a systematic, passivity-based approach* [9] was used as theoretical framework. Further, previous work done with this design procedure in [37] and [8] was used as inspiration for how to proceed. To avoid unnecessarily reiterating preexisting work, a short excerpt from [8] was included in appendix D explaining two steps to passivity-based control design originating from [9].

While all three of the listed requirements are important, the focus in this thesis centers around the first two, while keeping the possibility of communication robustness research a possibility for future research. A potential future system should have the UAVs only depend on their local states and the information received from their communication links, giving a decentralized control structure. With the intention of further focusing the work of this thesis, we assume perfect communication between all UAVs for all cases in this thesis. For the remainder of this section, it is assumed that the reader is familiar with graph theory and its nomenclature.

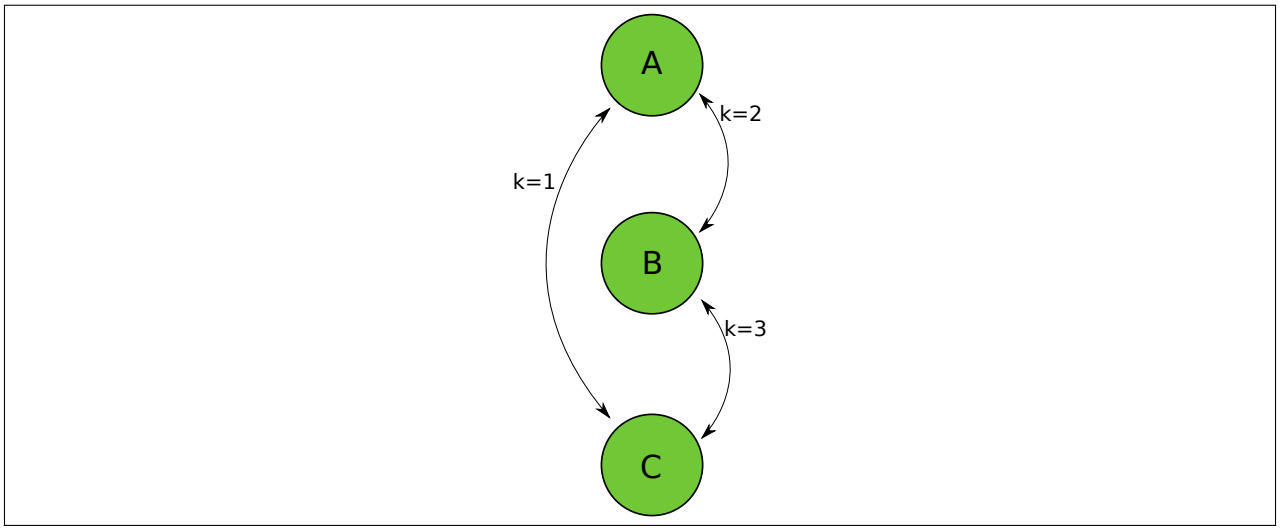


Figure 4.3: Graph for three UAVs communicating over three communication links.

We begin by considering N UAVs, and P communication links, as illustrated by figure 4.3 showing only $N = 3$ drones. The links are seen as undirected, as the communication links provide two-way communication. The communication between the N drones can then be represented by the incidence matrix $\boldsymbol{D}$:

$$\boldsymbol{D} = \begin{bmatrix} d_{11} & \cdots & d_{1P} \\ \vdots & \ddots & \vdots \\ d_{N1} & \cdots & d_{NP} \end{bmatrix} \quad \in \mathbb{R}^{N \times P} \tag{4.31}$$

In Eq. 4.31, the index i refers to UAV i, and k refers to communication link k. The elements d_{ik} of matrix $\boldsymbol{D}$ are defined in [9] as follows.

$$d_{ik} := \begin{cases} +1 & if\ k \in \mathscr{L}_i^+ \\ -1 & if\ k \in \mathscr{L}_i^- \\ 0 & otherwise \end{cases} \tag{4.32}$$

$$sending\ side\colon \mathscr{L}_i^+ \quad receiving\ side\colon \mathscr{L}_i^- \tag{4.33}$$

The communication links which the incidence matrix represents are used to determine the difference-variables $\boldsymbol{z}_k$ and $\dot{\boldsymbol{z}}_k$, which are central to the formation control scheme. For two drones i and j communicating over link k, the difference variable, and its derivative can be found as:

$$\boldsymbol{z}_k = \sum_{l=1}^{N} d_{lk} \boldsymbol{p}_{b,l}^n = \begin{cases} \boldsymbol{p}_{b,i}^n - \boldsymbol{p}_{b,j}^n & if\ k \in \mathscr{L}_i^+ \\ \boldsymbol{p}_{b,j}^n - \boldsymbol{p}_{b,i}^n & if\ k \in \mathscr{L}_i^- \end{cases} \tag{4.34}$$

$$\dot{\boldsymbol{z}}_k = \sum_{l=1}^{N} d_{lk} \dot{\boldsymbol{p}}_{b,l}^n = \begin{cases} \boldsymbol{v}_{b,i}^n - \boldsymbol{v}_{b,j}^n & if\ k \in \mathscr{L}_i^+ \\ \boldsymbol{v}_{b,j}^n - \boldsymbol{v}_{b,i}^n & if\ k \in \mathscr{L}_i^- \end{cases} \tag{4.35}$$

$$\tag{4.36}$$

By concatenating the UAV positions and difference-variables into two composite vectors, it is possible to rewrite the difference-variable vector $\boldsymbol{z}$ as shown in Eq. 4.38. The $\otimes$ notation in Eq. 4.38 is the Kronecker product.

$$\boldsymbol{p} = \begin{bmatrix} \boldsymbol{p}_{b,1}^n \\ \vdots \\ \boldsymbol{p}_{b,N}^n \end{bmatrix} \in \mathbb{R}^{3\cdot N} \quad \boldsymbol{v} = \begin{bmatrix} \boldsymbol{v}_{b,1}^n \\ \vdots \\ \boldsymbol{v}_{b,N}^n \end{bmatrix} \in \mathbb{R}^{3\cdot N} \quad \boldsymbol{z} := \begin{bmatrix} \boldsymbol{z}_1 \\ \vdots \\ \boldsymbol{z}_p \end{bmatrix}^T \in \mathbb{R}^{3\cdot P} \quad \dot{\boldsymbol{z}} := \begin{bmatrix} \dot{\boldsymbol{z}}_1 \\ \vdots \\ \dot{\boldsymbol{z}}_p \end{bmatrix}^T \in \mathbb{R}^{3\cdot P} \tag{4.37}$$

$$\boldsymbol{z} = (\boldsymbol{D}^T \otimes \boldsymbol{I}_{P\times P})\boldsymbol{p} \qquad \dot{\boldsymbol{z}} = (\boldsymbol{D}^T \otimes \boldsymbol{I}_{P\times P})\boldsymbol{v} \tag{4.38}$$

As is indicated by the name, the difference variable $\boldsymbol{z}$ and its derivative $\dot{\boldsymbol{z}}$ indicate the position difference and velocity difference respectively between communicating UAVs. The goal now is first to generate a desired difference vector $\boldsymbol{z}_{k,d}$. We set the desired velocity difference to be the zero vector, as we currently desire the drones not to move relative to each other when they have reached the formation. Next, we need to choose a function $\Psi_k(\boldsymbol{z}_k - \boldsymbol{z}_{k,d})$ such that we drive the difference vector $\boldsymbol{z}_k$ to the desired value $\boldsymbol{z}_{k,d}$. To define $\boldsymbol{z}_{k,d}$, we use the same approach as shown

to generate $\boldsymbol{z}_k$ as was shown in Eq. 4.38 only this time with vector $\boldsymbol{p}_d$ as desired relative position.

$$\boldsymbol{z}_{k,d} = (\boldsymbol{D}^T \otimes \boldsymbol{I}_{P\times P})\boldsymbol{p}_d \qquad \boldsymbol{p}_d = \begin{bmatrix} [0,0,0]^T \\ [0,0,-2\cdot l_{link}]^T \\ \vdots \\ [0,0,-2\cdot N\cdot l_{link}]^T \end{bmatrix} \tag{4.39}$$

$$\dot{\boldsymbol{z}}_{k,d} = \boldsymbol{0}_{P\times 1} \tag{4.40}$$

The constant l_{link} in Eq. 4.39 is the length of one of the physical links. The $\boldsymbol{p}_d$ vector forms a vertical line with N points at regular intervals. These points are the goal positions for the UAVs to be relative to each other in the formation. The function $\Psi_k(\boldsymbol{z}_k - \boldsymbol{z}_{k,d})$ used was simply the difference-variable error, with a weighting constant δ_k for each link k:

$$\Psi_k(\boldsymbol{z}_k - \boldsymbol{z}_{k,d}) = \delta_k \cdot (\boldsymbol{z}_k - \boldsymbol{z}_k^d) \qquad \Psi_k(\dot{\boldsymbol{z}}_k - \dot{\boldsymbol{z}}_{k,d}) = \delta_k \cdot \dot{\boldsymbol{z}}_k \tag{4.41}$$

Gathering the $\Psi_k(\boldsymbol{z}_k - \boldsymbol{z}_{k,d})$ terms in a single vector is done with a similar Kronecker matrix multiplication, this gives us the vector $\boldsymbol{u}$, representing the force the UAVs are to apply to reach the formation configuration.

$$\boldsymbol{u} = (\boldsymbol{D} \otimes \boldsymbol{I}_{2P\times 2P})\boldsymbol{\Psi} \tag{4.42}$$

The UAV-to-UAV control is done by formation control vector $\boldsymbol{u}$. As stated at the beginning of this section, the formation is also intended to reach a shared commanded velocity $\boldsymbol{v}_{cmd}$. This is done by a conventional PI controller:

$$\boldsymbol{\tau}_v = K_p(\boldsymbol{v}(t) - \boldsymbol{v}_{cmd}(t)) + K_i \int_0^t (\boldsymbol{v}(a) - \boldsymbol{v}_{cmd}(a))\,da \tag{4.43}$$

Finally, the work can be brought together with the internal control system to create the control loop illustrated by block diagram 4.4. The transfer functions $\mathscr{H}_i$ in the feedback loop are the transfer functions of the inner control system of the UAVs, taking in commanded force $\boldsymbol{\tau}_{cmd,i}$, and giving out velocity and position.

The $\boldsymbol{\tau}_{aux}$ term in the top right corner of Fig. 4.4 is the gathering term for various other inputs included to change system behavior. Initially, it only contains gravity compensation for the

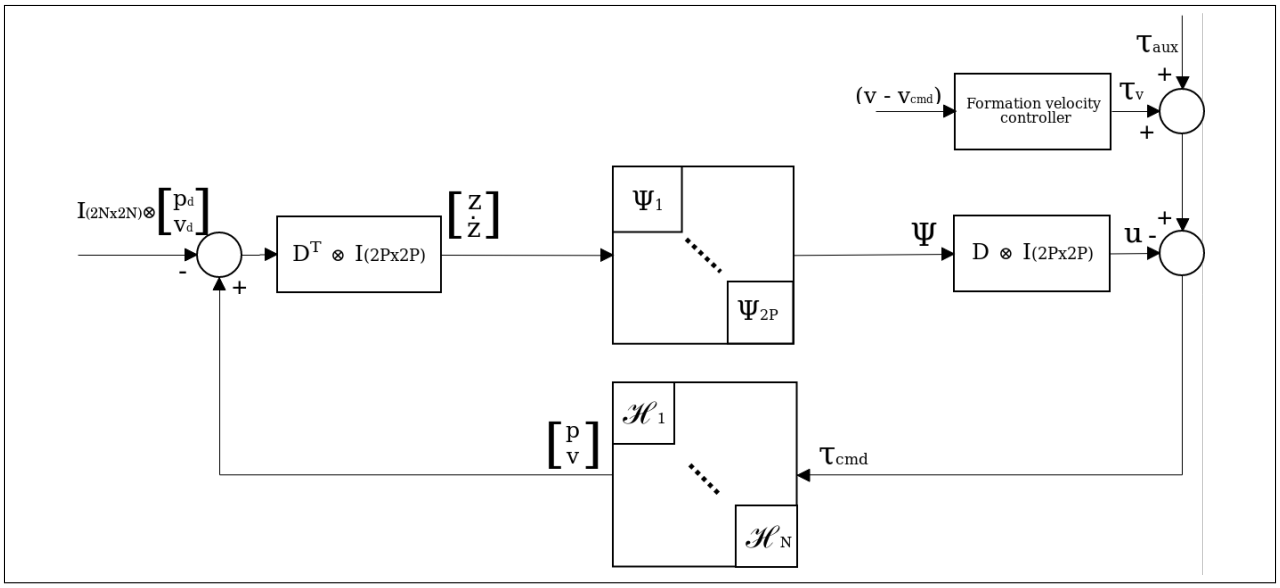


Figure 4.4: Block diagram showing the structure of the formation control system.

system. In the case where the vertical stack is fully connected the following was used for UAV i, essentially dividing the total mass of the stack equally among the drones:

$$\boldsymbol{\tau}_{aux,i} = \frac{1}{N}(\sum_{j=1}^{j=S} m_j \cdot g) \cdot \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}^T \tag{4.44}$$

Finally, the formation control system shown in Fig. 4.4 was implemented in the simulation system, and the results can be found in chapter 5.

Guidance System

In order to make the formation capable of following a number of predefined waypoints, a guidance system was implemented. This was the same type of Line-Of-Sight (LOS) based guidance system as was used in the preceding project thesis, and its properties are described in appendix C. One significant adaptation has been made to the guidance system. The position, velocity, and orientation which the guidance system relies on as the current state of the formation are found as the average of all the UAVs in the stack, this means that the formation will attempt to follow the waypoint path with its centroid, meaning the average UAV position.

4.3 Simulation

So far in this chapter, we have introduced the method with which the new approach models the vertical stack dynamics, and subsequently how the inner UAV control system actuates according

to the command from the formation control system. Before adding more complexity, it would be appropriate to detail how the simulations are performed. By gathering the contributions of the previously detailed models and controllers we are able to find the state double derivatives for all bodies in the system from Eq. 4.2 and Eq. 4.10 in the *unconstrained case*. The constraint forces $\boldsymbol{Q}_c$ are found next, finally giving us the state double derivatives $\ddot{\boldsymbol{q}}$ for the *constrained* complete system. The system behavior is intended to be as close to that of the real physical system as possible, thus the specifications which were summarized in table 3.1 were used.

The simulations were performed using MATLAB where the state double derivatives are integrated up yielding velocity and position subsequently in the conventional numerical manner using the Runge-Kutta 4th order method, courtesy of the MSS GNC toolbox [39]. A step size of $h = 0.01[s]$, was used for all simulations, and regarding the payload, the weight spec is indicated by the payload color.

- Blue: $0.3[kg]$, 48% of single drone weight.
- Green: $0.4[kg]$, 64% of single drone weight.
- Cyan: $1.0[kg]$, 161% of single drone weight.
- Red: $2.0[kg]$, 323% of single drone weight.

Finally, for the interested reader, the simulation system has been made accessible at this location [40]. Effort has been directed into making the simulation programs user-friendly with regards to high-level adaptations such as desired plots and system specifications.

Stack Initialization

The only aspect missing from the simulation setup described thus far is an initialization approach which makes sure that a user can start the system in the intended state, while not violating the constraints. The fact that constraints such as the rigid links connecting the drones and joints reduce the degrees of freedom of a system can be used to simplify the initialization and ensure satisfied constraints. In practice, this means that we begin by defining the position of the top drone, and iteratively define the initial state of the bodies below as a function of the position of the previous body. An angle $\boldsymbol{\Gamma}$ is used to describe the orientation of the connecting rigid link above the body to be placed. This can be summarized by Eq. 4.46, for body i in the stack.

$$\boldsymbol{p}^n_{b,1} = \boldsymbol{p}_{topdrone} \tag{4.45}$$

$$\boldsymbol{p}^n_{b,i} = \boldsymbol{p}^n_{b,i-1} + \boldsymbol{R}(\boldsymbol{\Gamma}^{\boldsymbol{b}}_{\boldsymbol{n,i}}) \cdot [0,\ 0,\ -l_{link}], \quad i = 2,3,\cdots,S \tag{4.46}$$

Another way of stating this is that the user defines the generalized, rather than the excessive coordinates.

4.4 Downwash Model Extension

The influence of downwash upon the lifting capabilities turned out to be significant when conducting the downwash experiments described in section 3.6.1 with the real drones. For this reason, an extension of the mathematical model was created to make the simulations more realistic. In the development of this model, the results of the downwash experiments, as well as a detailed study on rotorcraft downwash from the paper [4] was used. It is not the target of this thesis to provide a detailed aerodynamic simulation, but instead suggest a model matching the results from section 3.6.1 reasonably well.

By using curve-fitting functionality from the python Scipy library [41], we proceed to find suitable values for variables a, b and c in the expression Eq. 4.48. The curve-fitting is done with respect to the downwash intensity readings from the experiment described in section 3.6.1. This yields an approximation of the thrust reduction as a function of distance, and with some further adjustment should be usable in the simulations. The downwash reduction curve and the curve-fitting approximation can be seen in Fig. 5.1, discussion of the findings, and choices made for the approximation are given in chapter 6.

$$a = -0.7692 \quad b = -0.6026 \quad c = 1.0526 \tag{4.47}$$

$$m(o_z) = a \cdot e^{b \cdot o_z} + c \tag{4.48}$$

Inspecting Fig. 4.5, the distance o_{xy} away from the center of the downwash is accounted for in the model. The change of downwash effect is represented first by the bell curve shaped function around the center with height 1 as seen in Eq. 4.51. Then a cosine term is included to account for the directional limit of rotor thrust. The motor intensity i_{motor} of the upper drone responsible for the downwash is accounted for by approximating it to be linear with respect to thrust.

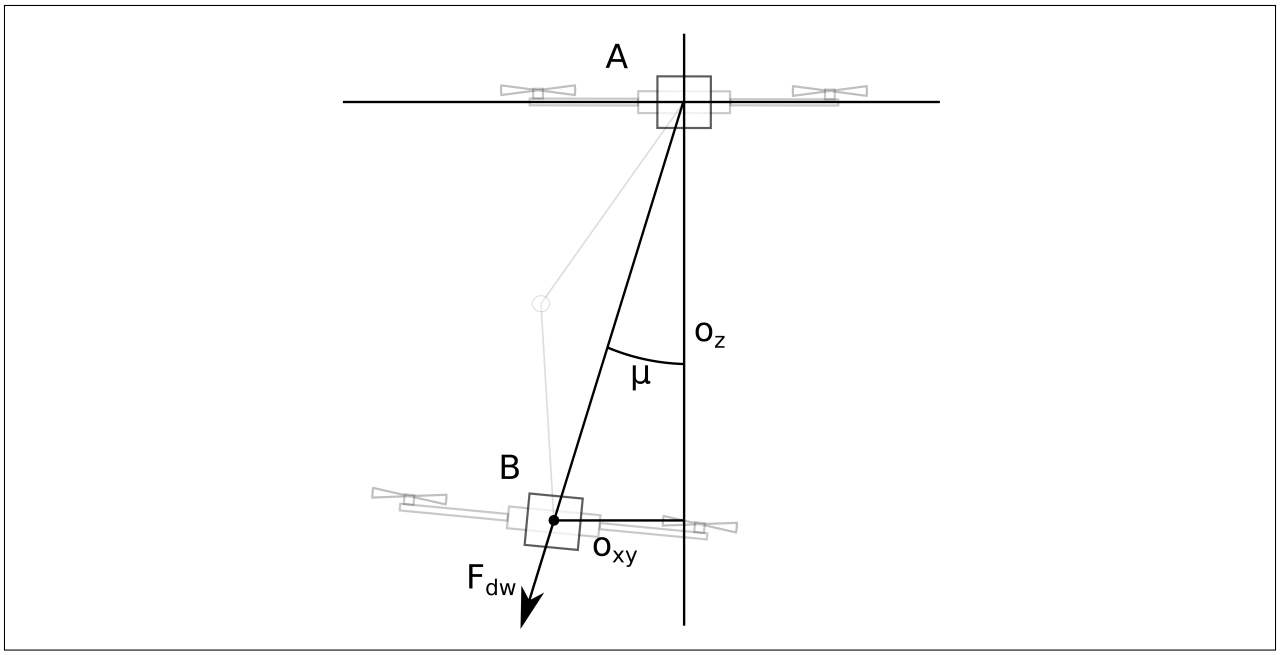


Figure 4.5: Variable overview to model downwash intensity.

$$\mu = \arcsin\left(\frac{o_{xy}}{|\boldsymbol{p}^n_{b,A} - \boldsymbol{p}^n_{b,B}|}\right) \tag{4.49}$$

$$\boldsymbol{F}_{dir} = \frac{\boldsymbol{p}^n_{b,A} - \boldsymbol{p}^n_{b,B}}{|\boldsymbol{p}^n_{b,A} - \boldsymbol{p}^n_{b,B}|} \tag{4.50}$$

$$g(o_{xy}) = e^{-|o_{xy}|\cdot\delta} \tag{4.51}$$

$$i_{motor} = \frac{F}{F_{max}} \qquad 0 \leq i_{motor} \leq 1 \tag{4.52}$$

The function $g(o_{xy})$ was adapted to correspond to the findings from [4]. The paper describes downwash stream velocity for various rotor-craft. It includes a study of the DJI Phantom commercial quad-rotor, and while it is not identical to our drone platform, it can be expected that the downwash behavior is similar. They show that approximately $\frac{1}{6}$ airstream velocity of the downwash remains at $0.4[m]$ from the center. By using this, a value for δ in $g(o_{xy})$ was found, giving the same bell curve shaped reduction in intensity as was presented in [4].

$$\delta = -\frac{1}{0.4}\ln(\frac{1}{6}) = 4.4794 \tag{4.53}$$

Finally, a downwash force vector F_{dw} could be found.

$$\boldsymbol{F}_{dw} = \boldsymbol{F}_{dir} \cdot (F_{max} - m(o_z)) \cdot g(o_{xy}) \cdot \cos(\mu) \cdot i_{motor} \tag{4.54}$$

This model for downwash force was implemented into the simulation such that each UAV applies the downwash model force to those below it, in order to explore the challenges presented by this effect. The force F_{dw} is added as an additional external force into Eq. 4.7. The resulting downwash plotted in 2d below a UAV can be found in result Fig. 5.2.3, and the approximations done to create the model are discussed in depth in section 6.2.6.

4.4.1 Downwash Compensation

After including the effect of downwash upon the system, as it was described in the previous section, it was quickly seen from simulations as shown later in result Fig. 5.21 that there would be a need to compensate for the downwash effect. The approaches to alleviate the issues are detailed here.

Downwash Feedforward

We consider a strategy for resolving the downwash issue where a feedforward term is included into $\boldsymbol{\tau}_{aux}$, relying on the same downwash approximation $\boldsymbol{F}_{dw}$ as detailed above to compensate. A feedforward term could eliminate rapidly changing influence if it is precise enough, and the drones are able to actuate sufficiently fast. The feedforward term included into $\boldsymbol{\tau}_{aux}$ is:

$$\tau_{dw-comp} = -\boldsymbol{F}_{dw} \tag{4.55}$$

The behavior of the system with feed-forward included into the controllers of the simulation is presented in the next chapter.

4.4.2 Efficiency Estimate

During simulations, it became clear that a metric was needed to evaluate the efficiency of the system. The intention was to estimate the influence of the downwash and the downwash compensation in terms of energy consumption. Not only the total energy consumption of the stack as a whole is of interest. Since the drones do not share an energy reserve, the battery-life of the stack can only be said to be as long as that of its shortest cooperating UAV. This means that one target for the formation is to maximize the battery life of the "worst" drone, through all UAVs

having an approximately equal energy consumption.

The energy consumption is assumed to scale linearly with the thrust intensity. We are in fact not interested in approximating the energy consumption of the test-platform with respect to real-world units, but rather the workload put on the UAVs as a result of the formation and control schemes. With this in mind, the energy consumed by UAV i, $E_i(t)$ up to time t, and the ratio $E_{ri}(t)$ to the average consumed energy is defined to be:

$$E_i(t) = \int_0^t |\tau_i(a)| da \tag{4.56}$$

$$E_{ri}(t) = \frac{E_i(t)}{(\sum_{j=0}^{N} E_j(t))\frac{1}{N}} \tag{4.57}$$

4.4.3 Payload Distribution

Observation of the ratio $E_{ri}(t)$ from the simulations affected by downwash revealed inconsistent energy consumption between the drones in the vertical stack. A formation feedback system similar to the one applied for the formation control system was implemented, only this time with the vertical force contribution in focus to try to alleviate the inconsistency between the energy expenditure of the drones.

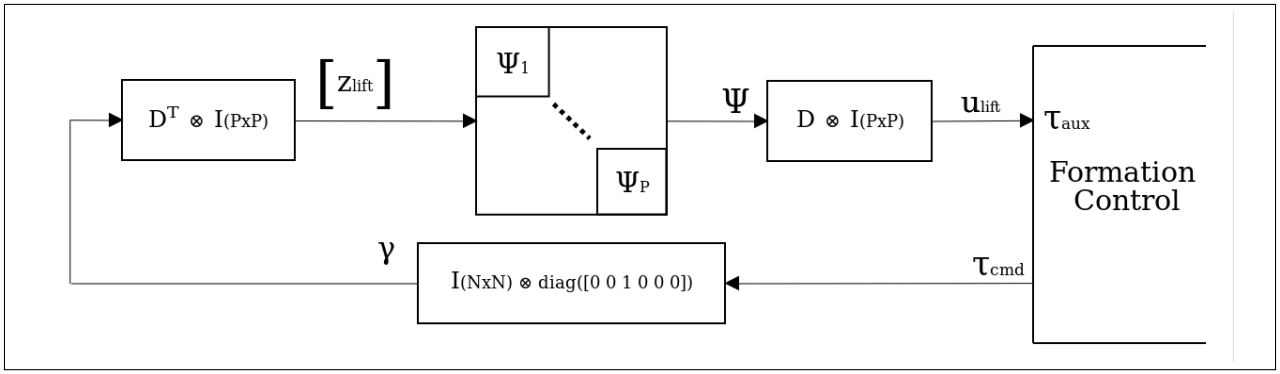


Figure 4.6: Block diagram illustrating the agreement problem of equal payload distribution between the drones.

Putting this into mathematical terms, we intend to drive the difference variables $\boldsymbol{z}_{lift,k}$ to zero. The $\boldsymbol{z}_{lift,k}$ terms reflect the difference in vertical lift between two UAVs communicating over link k. We define $\boldsymbol{z}_{lift}$, by first finding the vector of the vertical applied forces $\boldsymbol{\gamma}$:

$$\boldsymbol{\gamma_i} = \boldsymbol{I}_{N\times N} \otimes (diag([0\,0\,1\,0\,0\,0\,]) \boldsymbol{\tau}_{cmd,i}) \in \mathbb{R}^N \tag{4.58}$$

$$\boldsymbol{z}_{lift} = (\boldsymbol{D}^T \otimes \boldsymbol{I}_{P\times P})\boldsymbol{\gamma} \in \mathbb{R}^N \tag{4.59}$$

The same function Ψ is used in this case as was applied in section 4.2.2, only with an adjusted weight δ_{lift}. As was mentioned, the intention is to force $\boldsymbol{z}_{lift}$ to zero, which yields:

$$\Psi_k(\boldsymbol{z}_{lift}) = \delta_{lift} \cdot \boldsymbol{z}_{lift} \tag{4.60}$$

$$\tag{4.61}$$

The resulting terms were combined as was done in 4.2.2, and was included as an additional term into $\boldsymbol{\tau}_{aux}$ with appropriate zero spacing to affect only the vertical force commanded of the drones. Finally, the system performance was evaluated for various values of δ_{lift}. This feedback derivation was kept very brief since the process is largely an adaption of the previous formation system, which was described in greater detail.

Chapter 5

Results

This chapter aims to concisely present the results of the work detailed in chapters 3 and 4, extensive discussion of the results will be left to the next chapter. First, the thrust reduction in downwash is shown, next the evaluation of RTK susceptibility to angle. Finally a selected set of simulation results, the figures included are chosen to give a basis for the system discussion.

5.1 Real Platform Testing

The maximum thrust of the lowermost drone without external influence in the displayed experimental setup was found as described in section 3.6.1 to be $1.052[kg] = 10.320[N]$. Furthermore, the scale measurement as a function of the distance between the two drones can be seen plotted below in Fig. 5.1.

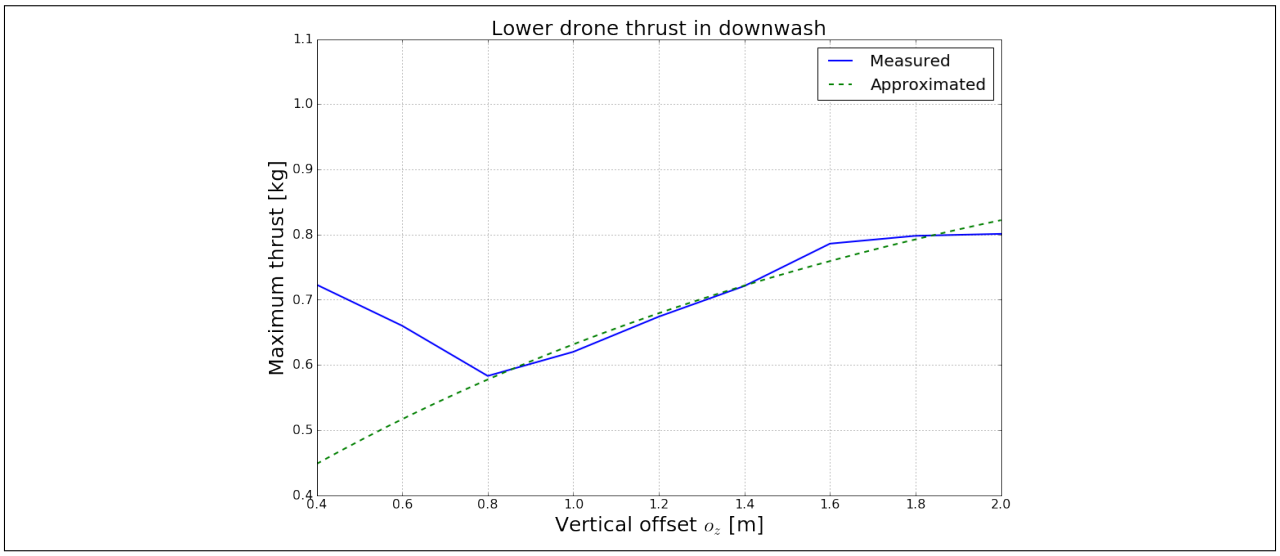


Figure 5.1: The change of maximum thrust measured, with the curve-fitting line used to approximate the thrust efficiency reduction as a function of vertical drone distance.

5.1.1 Susceptibility To Drone Tilt

Tables 5.1 and 5.2 show the distribution of position solution types among the solution types. The UAVs were kept stationary in an open area with decent satellite coverage, as described in section 3.6.2.

	Single	Float	Fix	Variance
Patch 0°	960	290	0	0.973
Patch 15°	921	327	2	50.182
Patch 30°	865	385	0	0.821
Patch 45°	871	379	0	21.684

Table 5.1: Overview of a population of 1250 position solutions found using **patch** antenna for increasing angles.

	Single	Float	Fix	Variance
Helix 0°	49	850	351	0.733
Helix 15°	62	860	328	4.262
Helix 30°	299	860	118	35.995
Helix 45°	333	792	125	10.268

Table 5.2: Overview of a population of 1250 position solutions found using **helix** antenna for increasing angles.

Figures 5.2 and 5.3 correspond to tables 5.1 and 5.2 respectively. The figures visualize how the solutions are spread out in North-East direction. The difference of the solution spread for changes of antenna angle for the two choices of antennas is of interest here.

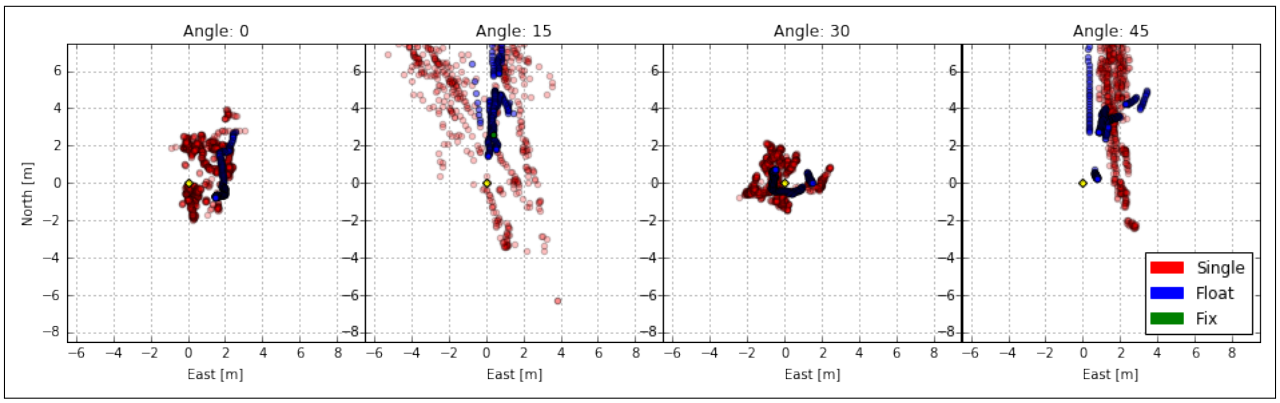


Figure 5.2: Position solutions for various antenna angles using a patch antenna.

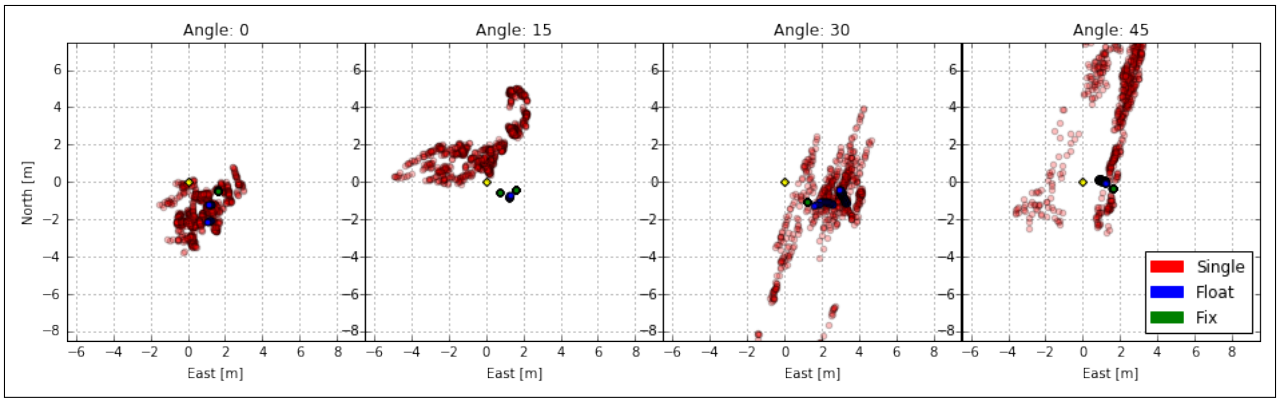


Figure 5.3: Position solutions for various antenna angles using a helix antenna.

5.1.2 Height

Box-plots 5.4 and 5.5 show the height aspect of the measurements summarized in tables 5.1 and 5.2. Separate boxes are used for each of the solution types, to underline the difference in sample grouping between the different types.

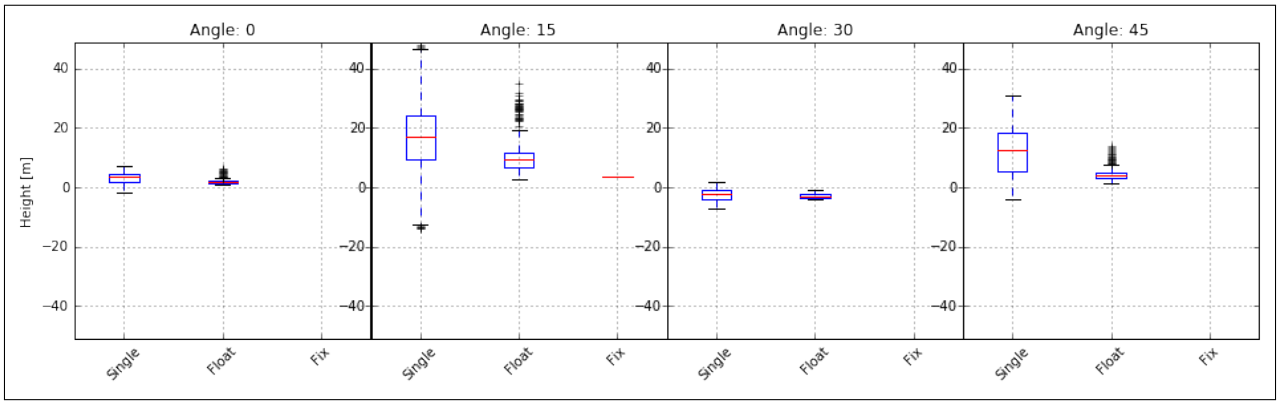


Figure 5.4: Height estimate from DUNE using patch antenna measurements.

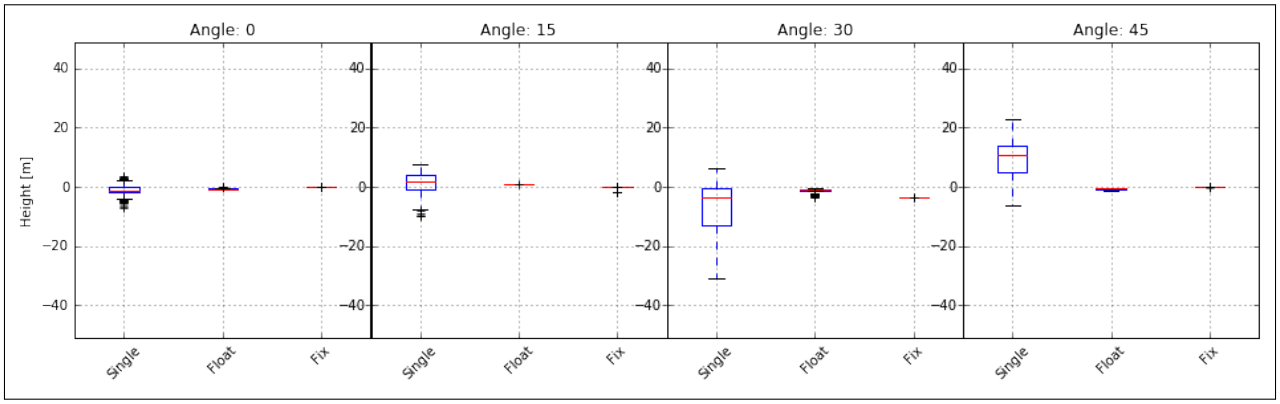


Figure 5.5: Height estimate from DUNE using helix antenna measurements.

5.1.3 Fix Precision

A visualization of the fix solutions from the helix antenna is shown in Fig. 5.6. In addition to an overall dense grouping of solutions, the split of solution grouping for the 15° session, and the altitude bias in the 30° case are noted.

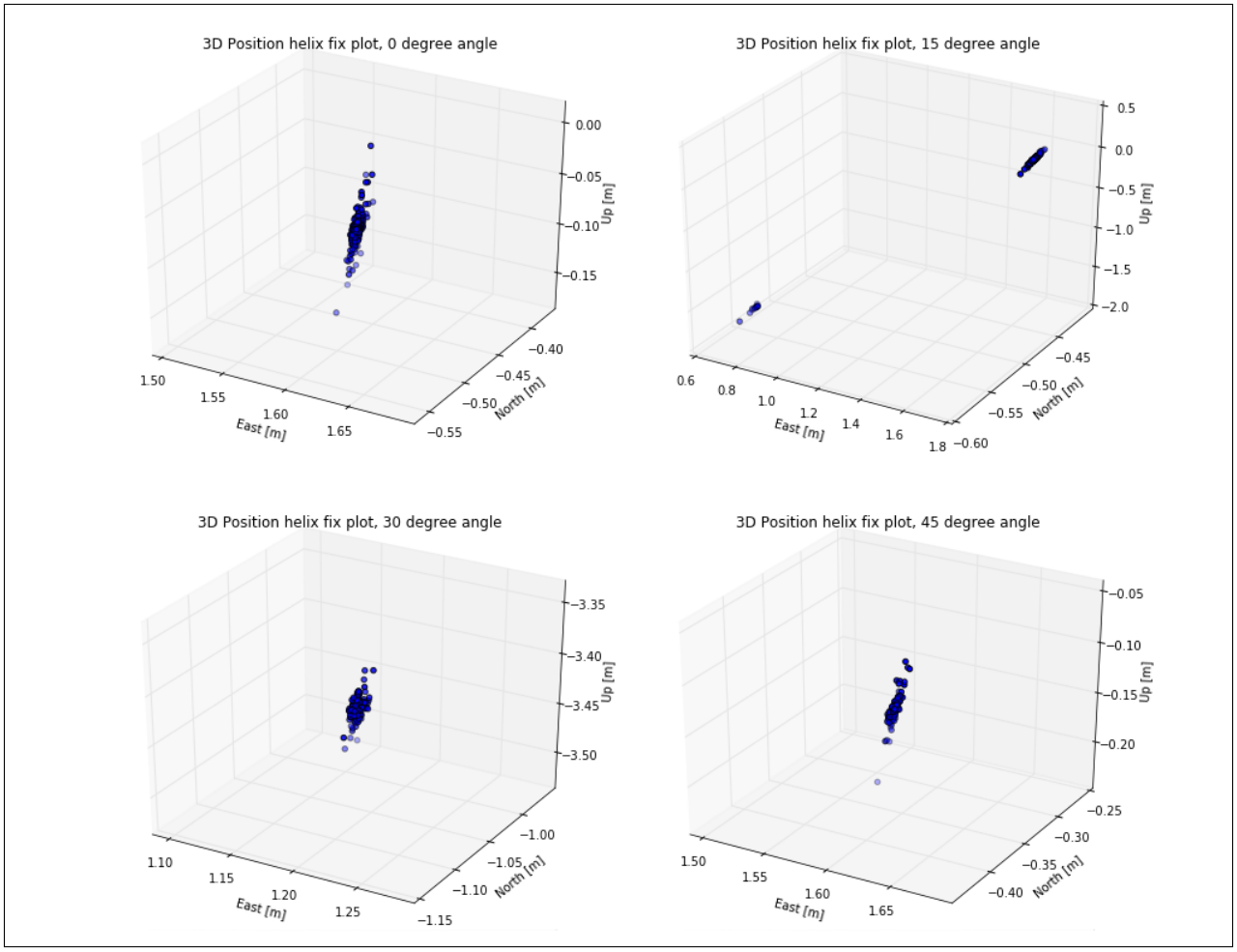


Figure 5.6: Grouping of the fix solutions from the angle test are seen to be very dense.

5.1.4 DUNE Estimate

The position estimated by DUNE from the helix antenna data, as mentioned in section 3.3.5 is illustrated in Fig. 5.7. An almost continuous, but not stationary position estimate is seen for all of the angles.

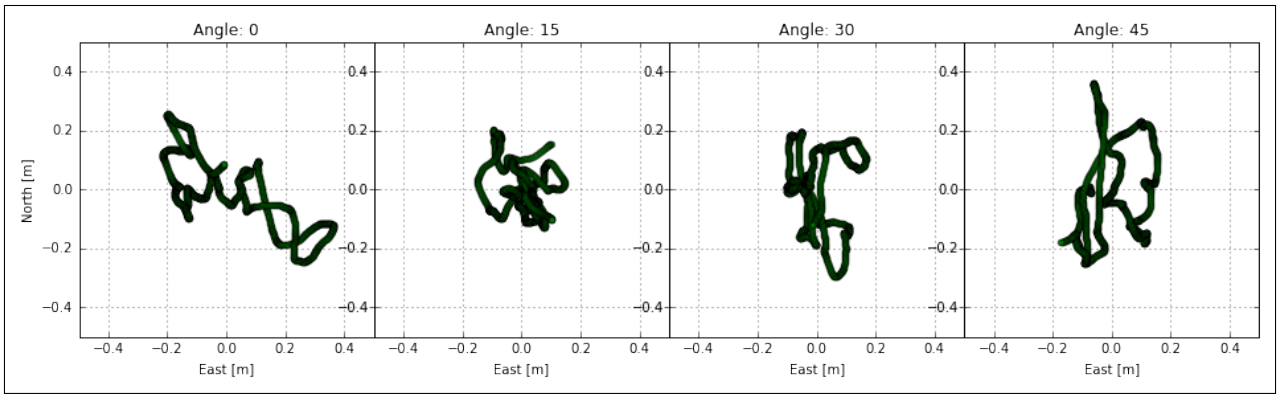


Figure 5.7: Estimate of position away from average position fetched from DUNE using the helix antenna data.

From the same DUNE estimation session as displayed by Fig. 5.7, the height estimate over time is shown in Fig. 5.5. The estimate is seen to be drifting slightly for all angles.

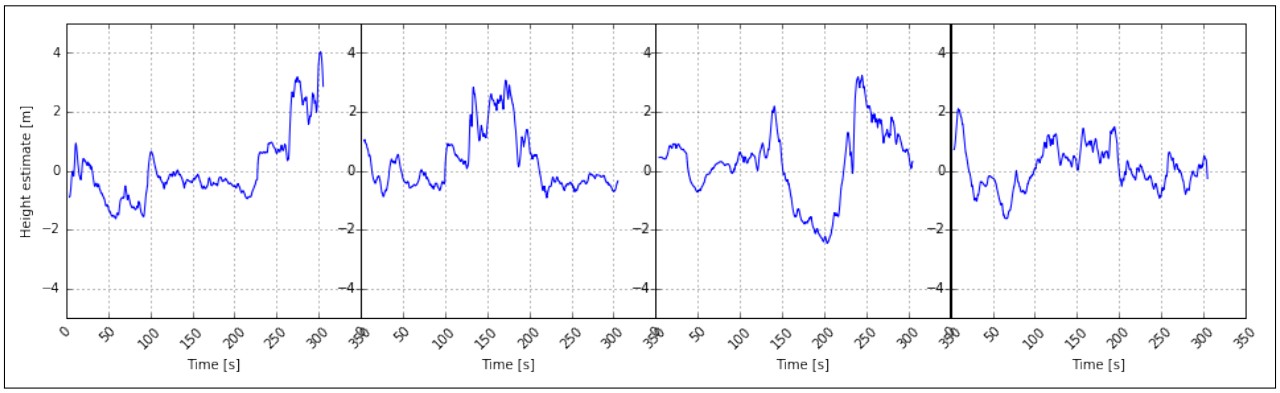


Figure 5.8: Height estimate from DUNE over time using helix antenna measurements.

5.1.5 UAV Flight

An image of a drone loitering, during a 60[s] flight session is displayed in Fig. 5.9, with its corresponding RTK position solutions and DUNE position estimates shown in Fig. 5.10. Some movement is seen in the DUNE estimate, and there is notable solution scattering of the RTK solutions of Fig. 5.10.

Figure 5.9: One of the UAVs maintaining altitude.

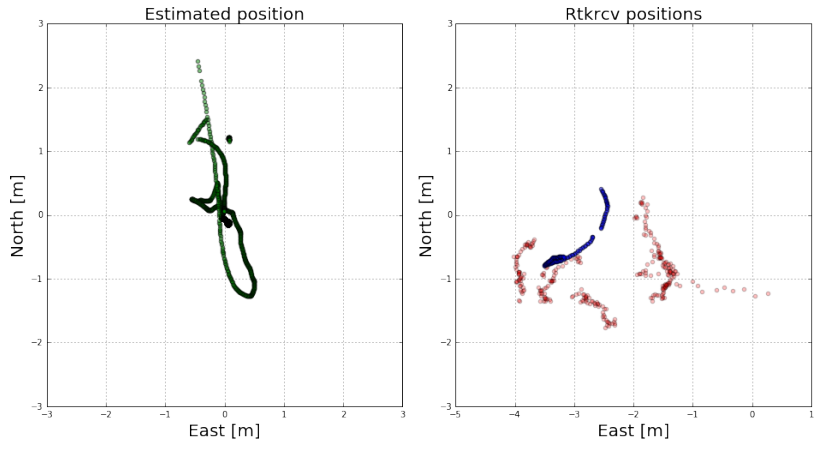


Figure 5.10: Simultaneous logging of RTK receiver measurements and DUNEs internal position estimate.

5.2 Simulation Results

The first simulation setup involves a single drone modeled and controlled as was detailed in section 4.2.1, ordered to spiral gradually upwards. Fig. 5.11 shows adequate gravity compensation and velocity tracking from the right-hand side plots.

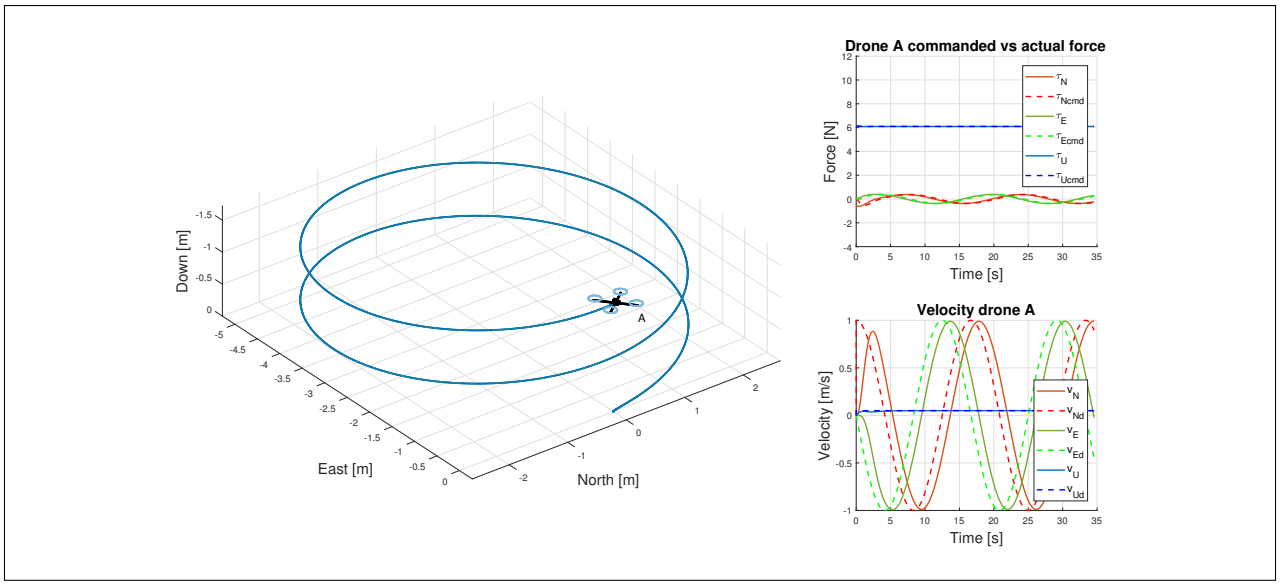


Figure 5.11: The single drone case given desired input velocity as a spiral, slight delay is seen between the commanded force and actual force realized by the quad-rotor.

A single drone, this time with a $0.3[kg]$ payload is seen executing an abrupt turn at time $10[s]$ in Fig. 5.12. Payload swing and increased thrust needed at time $10[s]$ can be seen.

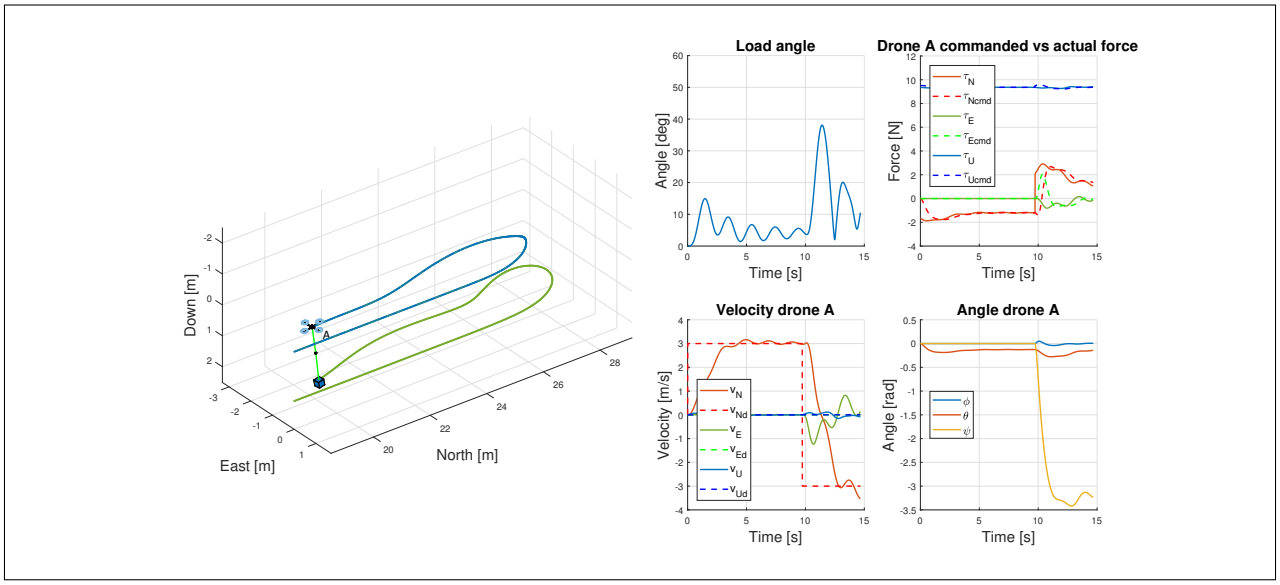
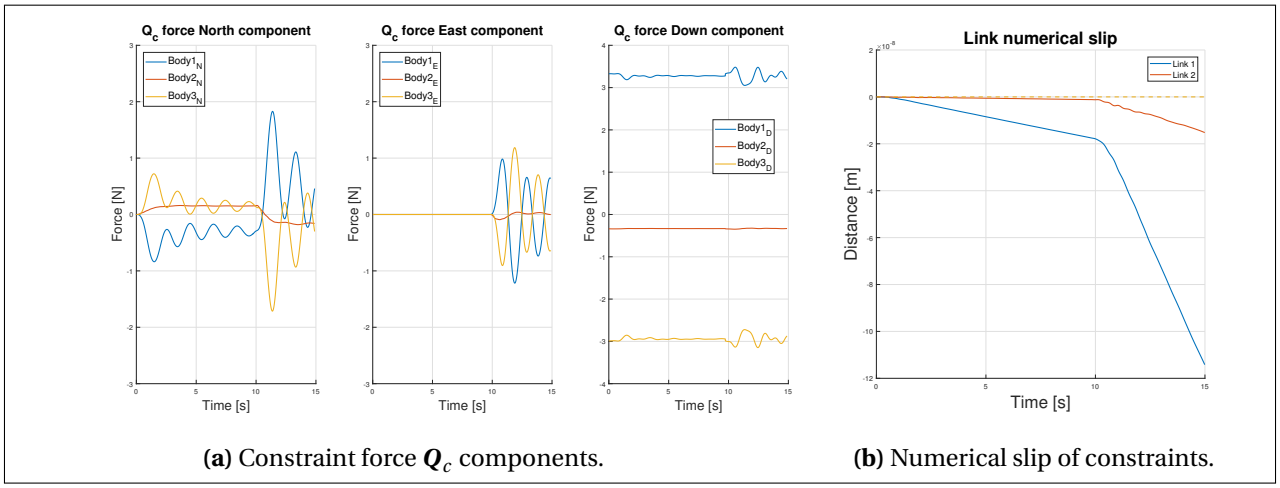


Figure 5.12: Simulation showing a single UAV with a $0.3[kg]$ payload reaching $3[m/s]$, then having desired velocity suddenly inverted.

Four additional plots related to the abrupt turn simulation in Fig. 5.12 can be inspected in Fig. 5.13. The turn is seen to strain the constraint greater from Fig. 5.13a, and increase rate of numerical error from Fig. 5.13b



(a) Constraint force $\boldsymbol{Q}_c$ components. **(b)** Numerical slip of constraints.

Figure 5.13: Plots corresponding to Fig. 5.12, showing constraint force $\boldsymbol{Q}_c$, and the constraint numerical slip.

5.2.1 Formation Reach

The formation control system detailed in section 4.2.2 was tested for variations of the integral action. The case where zero integral effect was included is displayed in Fig. 5.14, showing overall good behavior, but constant offset of the formation velocity.

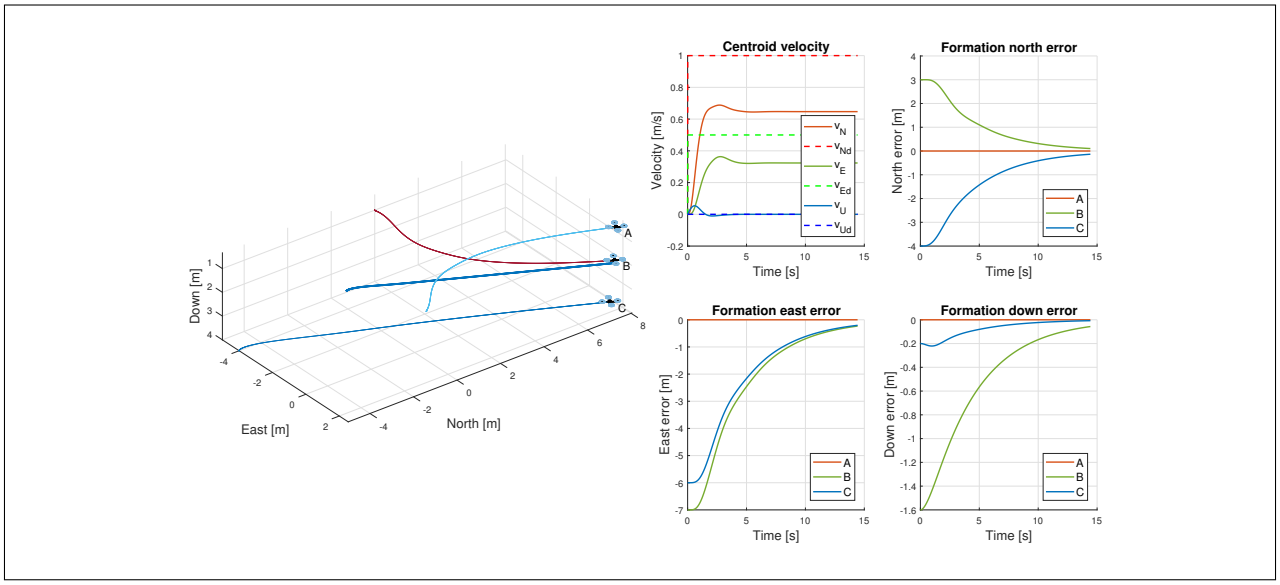


Figure 5.14: Three unconnected UAVs attempt to reach the desired formation topology and velocity from random starting points without integral action.

The velocity offset from the zero integral case is seen to be eliminated in Fig. 5.15, but the target formation topology is not reached with the integral effect.

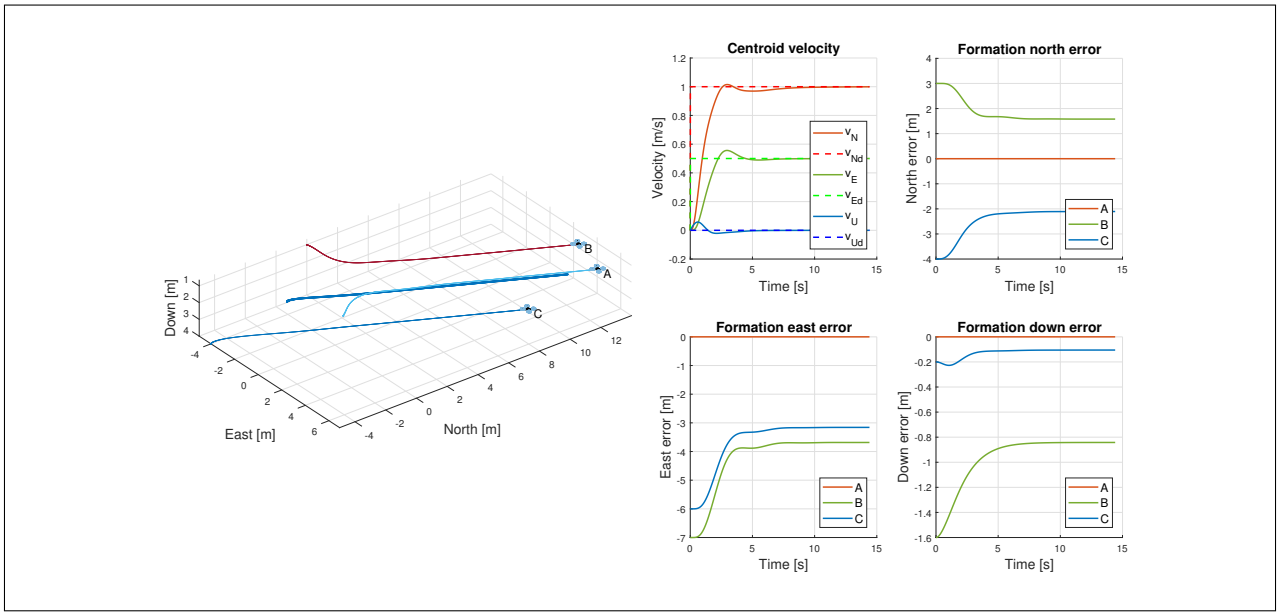


Figure 5.15: Three unconnected UAVs approach the desired formation topology and velocity from random starting points with integral action.

The result of applying the gradual integral effect discussed in section 6.2.2 is presented in Fig. 5.16. The drones first reach the target formation, and then the integral is seen to take effect from the centroid velocity plot.

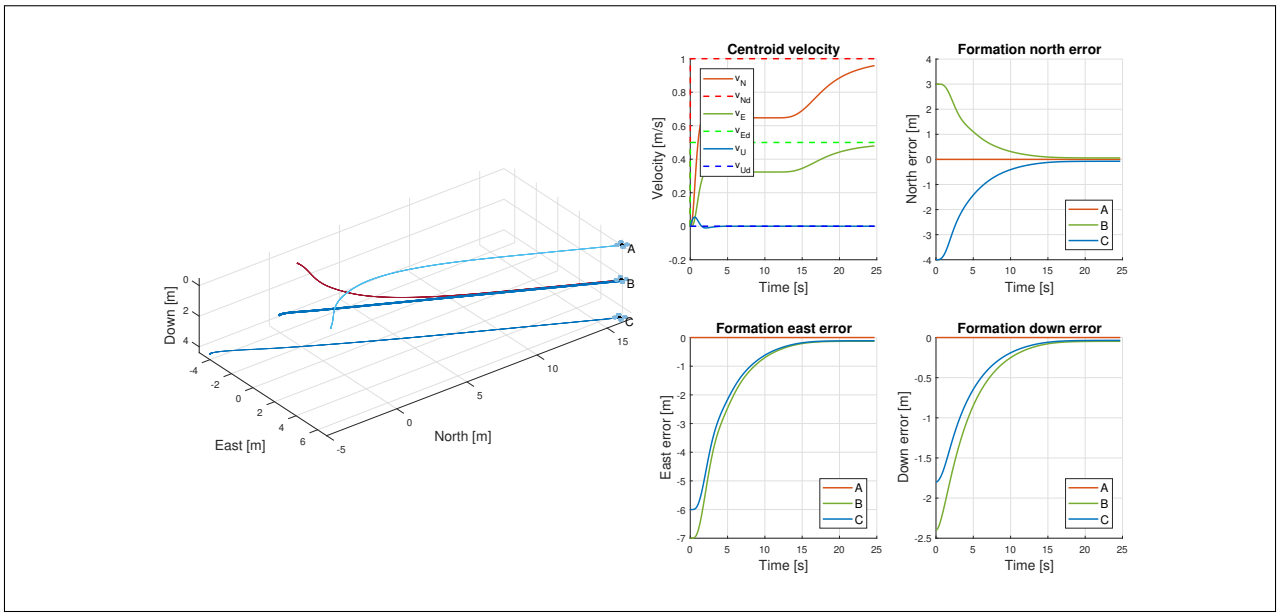


Figure 5.16: Three unconnected UAVs approach the desired formation topology and speed this time using the gradual integral system.

5.2.2 Traversing A Path

A two-drone stack traversing a square trajectory lifting a $0.4[kg]$ payload is seen in Fig. 5.17. The distribution of the payload is seen from the lower right plot, where A applies $8.2[N]$.

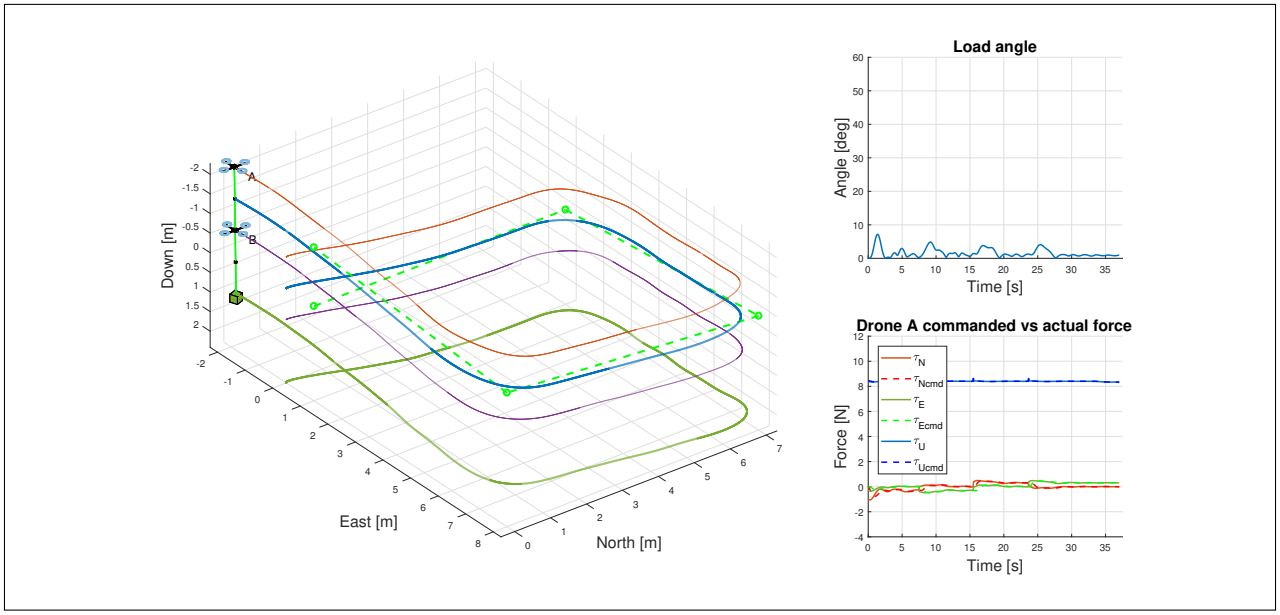


Figure 5.17: Simulation showing the two drone stack with a $0.4[kg]$ load traversing a 7×7 meter square, with reasonably small payload swing.

The same setup as in Fig. 5.17 was extended to include ten UAVs and a $2.0[kg]$ payload. The same trajectory is successfully traversed, but more time is needed to traverse the path.

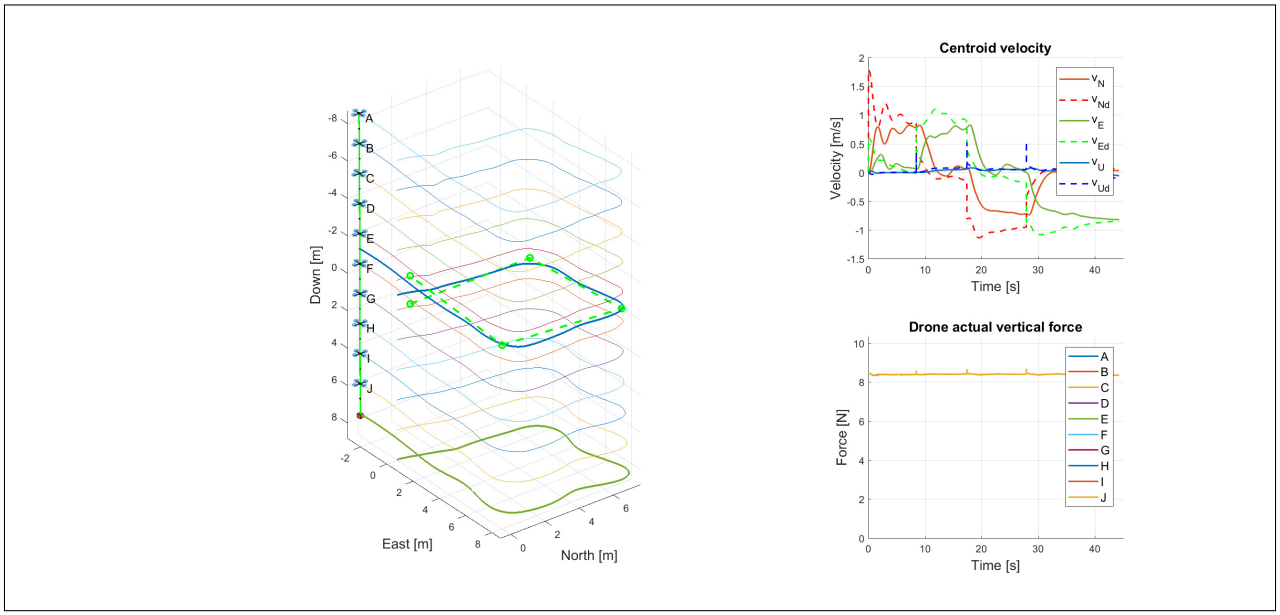
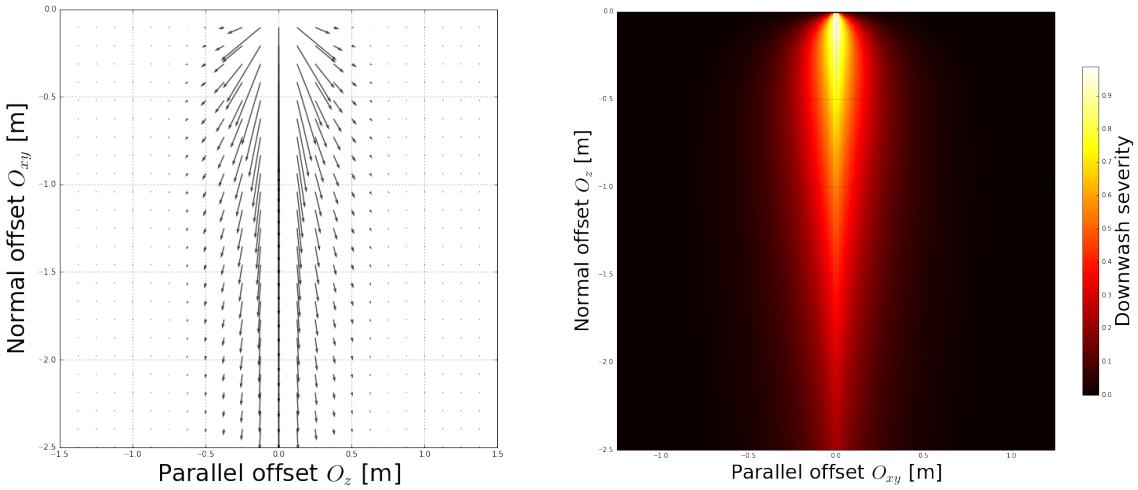


Figure 5.18: The simulation successfully extends to 10 drones, traversing the same square with a $2[kg]$ payload.

5.2.3 Downwash Effect And Compensation

From the downwash approximation function derived in section 4.4 a visualization of the downwash magnitude and direction was made, and is shown in Fig. 5.2.3. The severity of the downwash approximation is seen to be most intense in the center of the airstream.



(a) Quiver plot illustrating downwash force $\boldsymbol{F}_{dw}$.

(b) Plot showing the intensity plot of the downwash.

Figure 5.19: Two plots showing how downwash is represented as a function of UAV position and orientation, from section 4.4.

In Fig. 5.20 the case of three drones cooperatively carrying a 0.4$[kg]$ payload is seen, before adding downwash to the system.

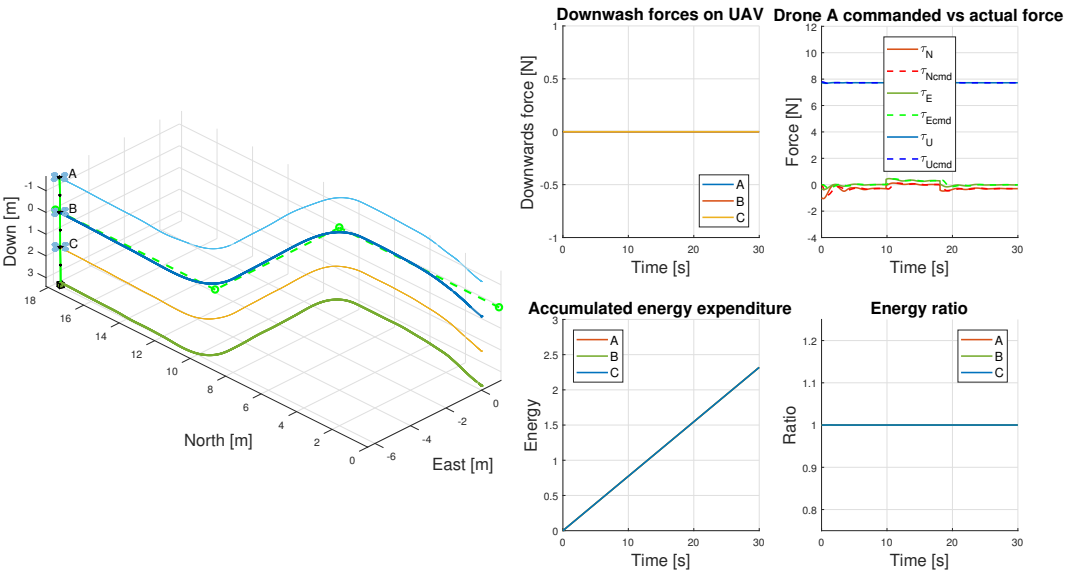


Figure 5.20: Three drones carrying a 0.4$[kg]$ load traversing a path before including the effect of downwash.

The same path was traversed, this time with downwash included in Fig. 5.21. The contrast in both the waypoint tracking and drone actuation to Fig. 5.20 is substantial.

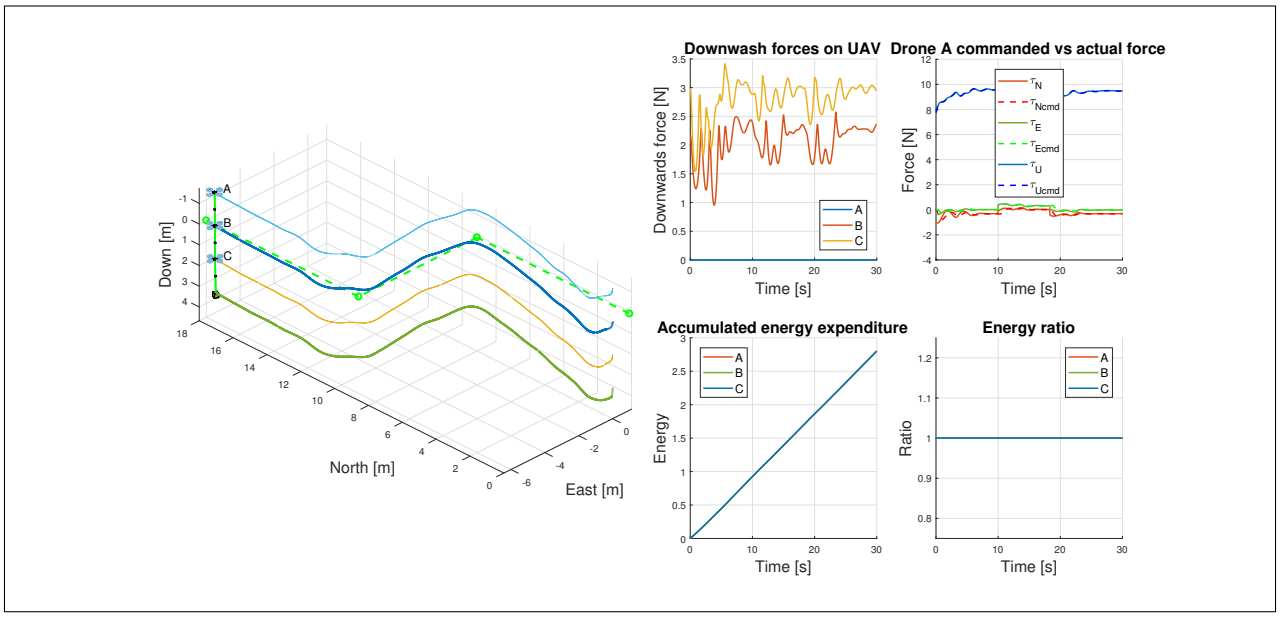


Figure 5.21: Similar setup as Fig. 5.20, only this time downwash appears problematic for the altitude tracking.

Feedforward downwash compensation included into the same three drone stack system as was detailed in 4.4.1 appears to alleviate the issues caused by the downwash, although uneven energy expenditure by the drones can be seen by the right-hand side plot in Fig. 5.22.

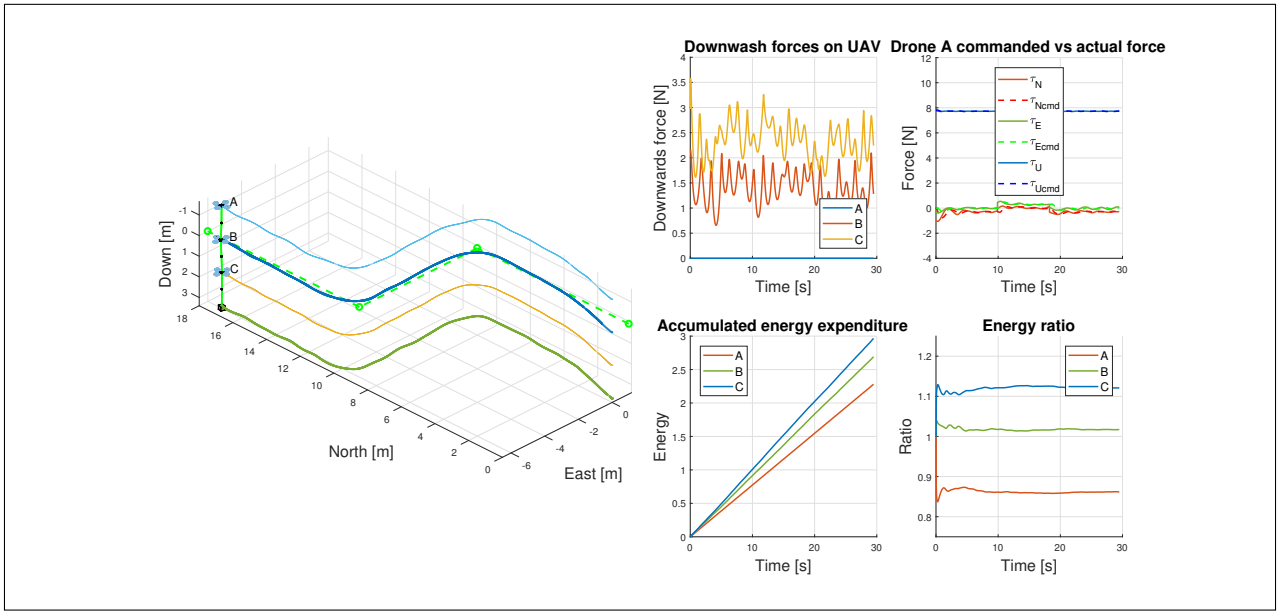


Figure 5.22: Including the downwash feed-forward resolves most of the downwash influence, and the path is followed with reasonable precision. Notable difference between expended energy is seen.

An additional control layer to distribute the workload more evenly as explained in section 4.4.3 gives the behavior seen in Fig. 5.23. The energy ratio appears to have been brought closer to an equal distribution, without increasing the total energy expenditure.

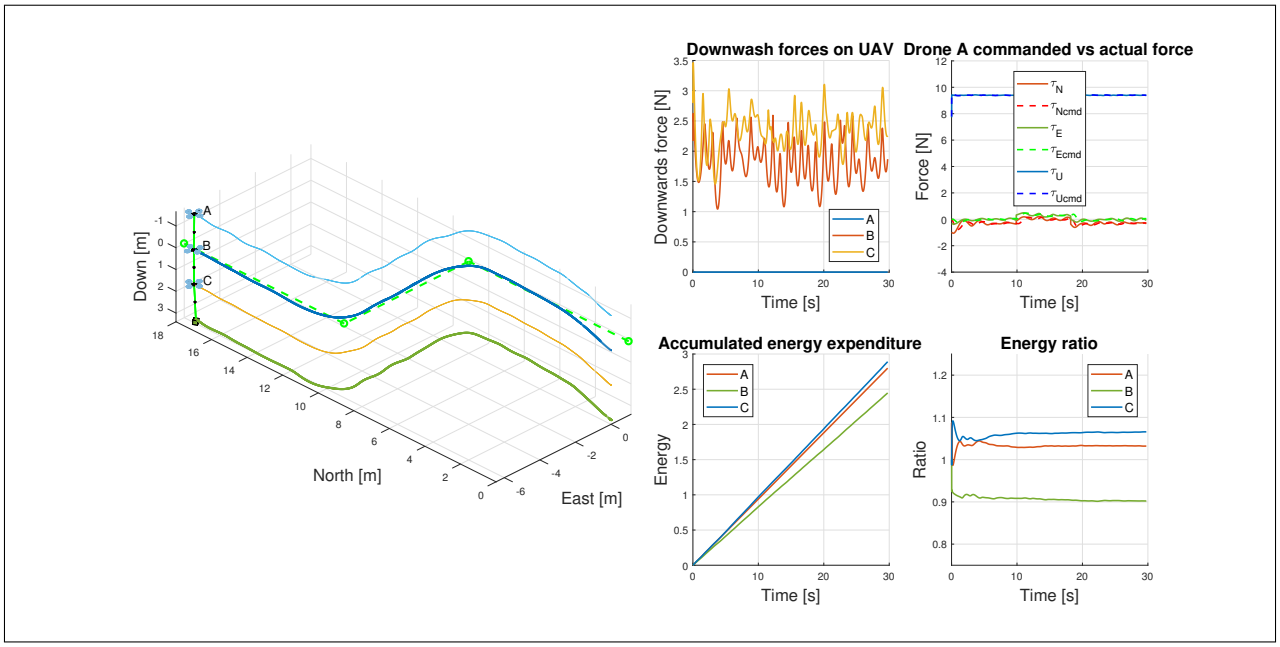


Figure 5.23: Similar setup, only this time with downwash, downwash compensation and the load sharing system implemented.

5.2.4 Determining Link Length

The ten-drone stack, with downwash, and feedforward compensation was ordered to do a left turn carrying a 2.0$[kg]$ payload in Fig. 5.24. The 0.8$[m]$ link length meant the downwash was too much for the UAVs to maintain altitude, causing the UAV thrust to become saturated.

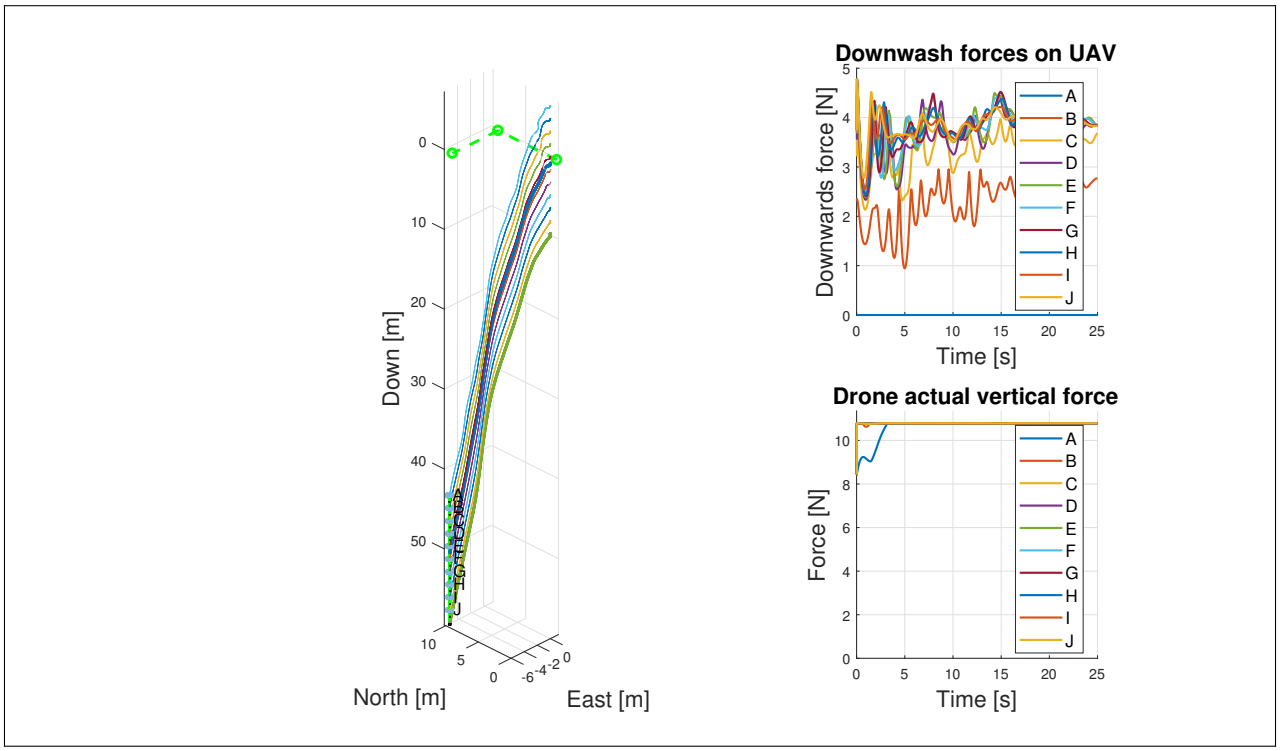


Figure 5.24: Ten drone stack with 2.0$[kg]$ payload, and link length 0.8$[m]$, falling due to the downwash force.

Downwash force for on drones in the ten drone stack at 0° degree angle is displayed in Fig. 5.25, for various choices of link length.

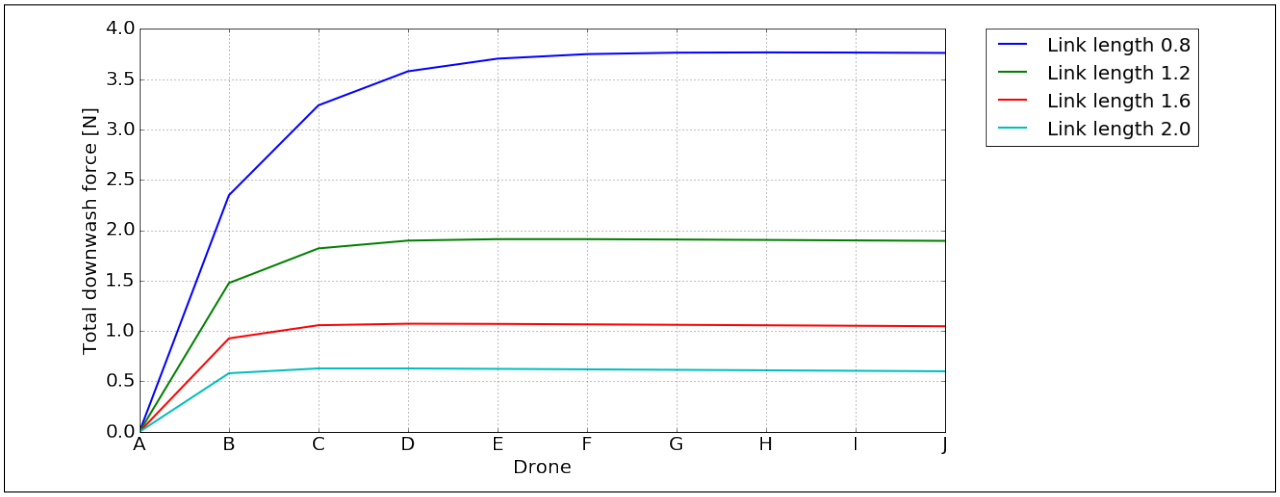


Figure 5.25: Total downwash from above, as experienced by UAVs at $t = 0$ in ten drone stack.

Increasing the link length to 1.0$[m]$ as shown in Fig. 5.26, appears to give better behavior. The change of link length enables the stack to traverse the planned trajectory, but with some drones still showing saturated thrust and oscillatory behavior.

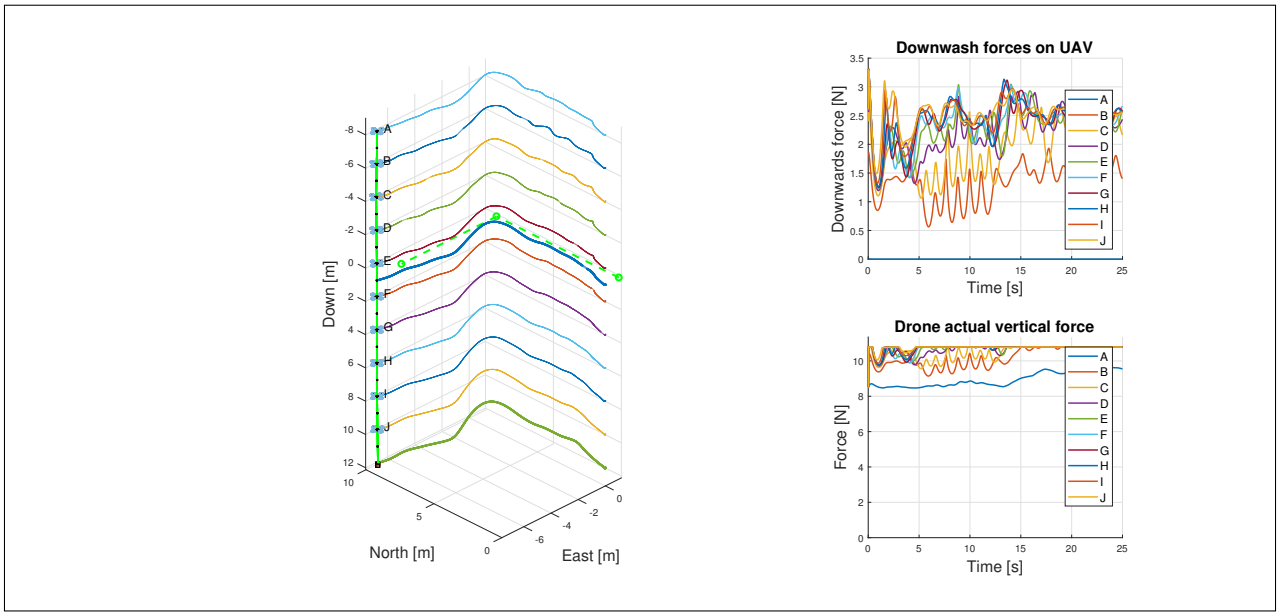


Figure 5.26: Ten drone stack with 2.0$[kg]$ payload executing single turn, with link length 1.0$[m]$, maintaining altitude but showing some oscillation.

By increasing the link length further to 1.2$[m]$ as shown in Fig. 5.27. The stack executes the turn smoothly without saturating thrust for the cooperating UAVs.

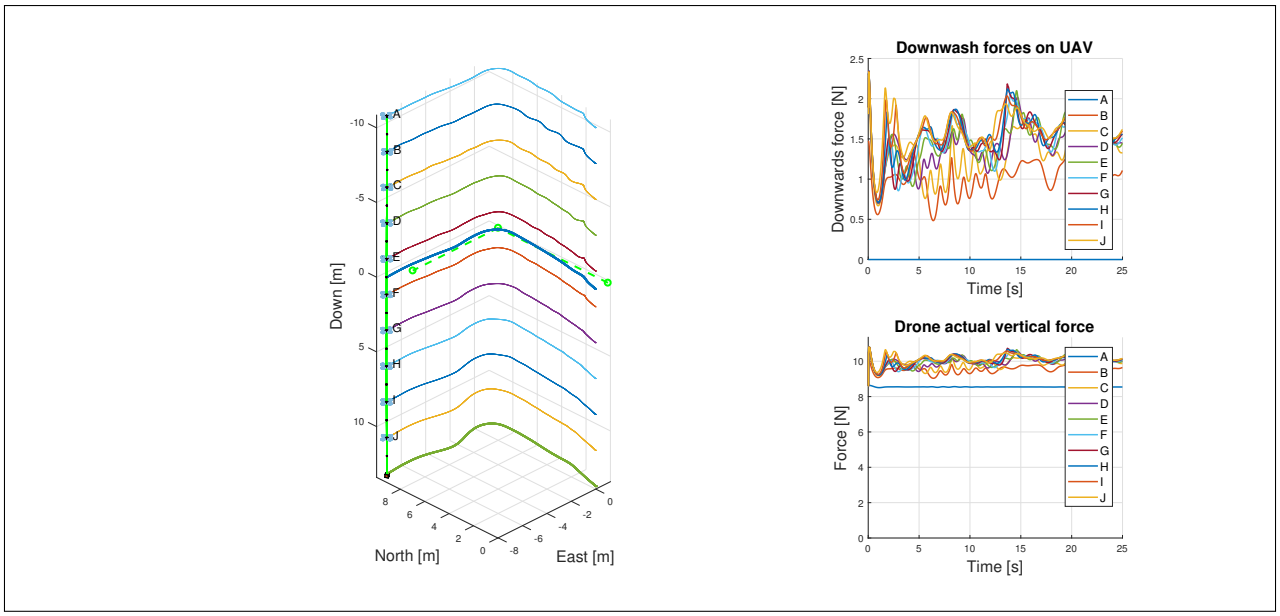


Figure 5.27: Ten drone stack with 2.0$[kg]$ payload executing single turn, with link length 1.2$[m]$ successfully.

Chapter 6

Discussion

A discussion surrounding the findings presented in chapter 5 will be given in this chapter, as well as a review of the reliability of the experiments conducted. This discussion will begin with the drone technology chapter 3 as the center of attention, and thereafter move on to the simulation methodology chapter 4. The realism of the model and validity of the simulation results will also be discussed. Finally, this leads to a broader discussion of the vertical stack concept, and the implications of the findings of this thesis.

6.1 Realized Drone System

6.1.1 Downwash Effect Measurement

Fig. 5.1 presents the data points found through the downwash effect measurements described in section 3.6.1. The first aspect noted from this plot is the apparent exponential decrease of downwash severity as the distance is increased from $0.8[m]$ to $2.0[m]$. This observation was the motivation for the curve-fitting for the downwash model implemented in section 4.4. The tendency fits well with intuition regarding air resistance, where turbulent air stream flow is approximated to experience air resistance squared with its velocity.

A completely satisfactory explanation to the peculiar tendency noted from $O_z = 0.4[m]$ to $O_z = 0.8[m]$, showing reduced downwash when moving closer to the UAV has not been found. The tendency was repeatedly measured over multiple samples for the same distance. It could be theorized that the air stream had not yet become turbulent at this distance, providing a more efficient medium of operation for the lower rotors. Regardless of the actual cause of the tendency, a connection between the UAVs closer than 1 meter would be hazardous, meaning this distance is not considered an option for choice of link length.

As was briefly mentioned in the experiment description, the intention of the setup was not to exactly measure the maximum lift for the lower UAV. The purpose was to reveal how the downwash intensity is a function of vertical distance O_z, so that it could be applied to the simulations, in order to root the simulation deeper in the real world system. There are several issues with the downwash measurement approach which prevent it from being an exact experiment, such as the existence of the tripod below the lower drone, and the connecting bar above, as illustrated in Fig. 3.11a. The height $1.5[m]$ of the lower from floor level is also a source of error, as some of the ground effect likely remains and influences the maximum achievable thrust. Despite this, from the measurements, $F_{max} = 10.32[N]$ was included into the model, as it seemed like a conservative limit compared to the $F_{max} = 18[N]$ specification reported by the motor supplier with ideal rotors and power supply.

An initial choice of link length was made from observation of the severity measurement in Fig. 5.1. The length was chosen such that the downwash still would be notable and its effect upon the system could be studied and addressed. The intention was not to make it too intense for the UAVs to actuate and maintain altitude initially either, for this reason, a suitable compromise $l_{link} = 0.8[m]$ was chosen as an initial value.

6.1.2 RTK Position Evaluation

To start the comparison of the two antenna alternatives, consider table 5.1 and table 5.2. First it should be pointed out that 1250 solutions were chosen from each of the measurement sessions, however, the helix antenna was able to gather around 2500 solutions during the $300[s]$ duration, compared to the approximately 1400 solutions gathered by the patch antenna setup. To compare roughly equal population sizes, the first 1250 solutions of both were extracted, giving the presented tables. With this in mind, the table entries can be discussed.

North-East Solutions

Table 5.1, corresponding to the patch antenna shows very uneven levels of variance when changing angle. This corresponds well to the solution plots in Fig. 5.2 showing wide spreads. From the table, it can also be seen that almost no RTK-fix solutions are found, but a decent number of float solutions are found for all choices of angle. The helix antenna as indicated by table 5.2 has a lower variance in general, with the exception of angle 30° showing a spike from the otherwise very low numbers. Observing Fig. 5.3 the difference does in fact not seem very sizable to the previous patch antenna results. However, the table data reveals that the majority of the

solutions are in the float class, and a substantial number in the fix class. This means that the solution concentration of the helix antenna is overall significantly better than that of the patch antenna, as shown by the stable position of the dense clusters of both blue float solutions and green fix solutions in Fig. 5.3.

Height Solutions

The height estimates of the measurements also need to be evaluated. Comparison between Fig. 5.4 and Fig. 5.5 shows the same general tendency of notably less variance for the samples from the helix antenna. Relatively few outliers are seen in the helix boxplots, even in the part of the sample population in the single class. The same uneven accuracy of the patch antennas measurements for varying angles is reiterated in these plots, where both 0° and 30° show acceptable spread, but the opposite is true for 15° and 45°. This corresponds well to the description of patch antennas given earlier, where it was stated that they are usually effective at relatively small arcs, meaning small variations in angle can cause large changes in solution quality. The float and fix solutions of the helix antenna test appeared reliable and concentrated, this highlights the effect that the RTK correction has on the height estimate, comparing to the widespread single solutions.

From these observations, the helix antenna was chosen to be used on the antennas for the following reasons:

- The number of solutions gathered during the 300$[s]$ session was almost twice as high using the helix antenna.
- The helix antenna offered fix solutions and a large number of float solutions.
- The variance appeared lower in general with the helix antenna.

Now we intend to consider the effective precision of the drones with this setup.

RTK-Fix Precision

The primary motivation for including Fig. 5.6 into the report is to show what level of precision was achieved for the fix type solutions. The points are seen to be densely grouped, within approximately $\pm 0.15[m]$, which is excellent. The variation between the angle of the samples is seen to influence the fix measurement little. The exception of the four angles is the 15° degree plot,

showing two clusters, spaced about 1[m] apart. Since the measurements were done entirely stationary, this error is likely an initial position fix error, perhaps from multi-path issues where the satellite signal bounces off the floor. A substantial constant error is seen in the 30° degree plot, estimating the drone to be 3.4[m] higher than it actually is. These measurements were all done while the drone was completely still, this has the benefit of making the circumstances for the RTK receiver constant, and it is uncertain if equally tight grouping could be achieved mid-air.

DUNE Position Estimates

Next, DUNE was used to estimate UAV position using the helix data which was summarized in table 5.2, and illustrated in Fig. 5.3 and Fig. 5.5. The result was the filtered $North-East$ position displayed in Fig. 5.7 and height plot in Fig. 5.8. The much smaller scale on the $North-East$ axis compared to the direct measurements is the first aspect of the plots to be noted. The number of estimates reached by DUNE was around 7800, showing that the more frequent IMU and barometer sensor readings are relied upon in between the measurements to estimate position. This level of high-frequency measurement is necessary for a real-time system such as the described UAVs.

Observing the position, the Kalman-filter prevents the sudden long hops from one estimate to another. The overall behavior with all angles is about equally decent. The reason might be that the IMU data is also being included into the Kalman-filter, making the variance from the change of angle matter less. Continuing to the height estimate of Fig. 5.8 over time, the estimate does not seem very precise. The height estimate is however affected by barometer data, meaning variations in pressure could account for this unsteady change over time. The measurement location at the time of measurement was exposed to wind, which might explain changes in pressure measurements. Established practice in the field of drone flight is that the barometer is covered by a layer of foam to prevent overly sensitive readings, however, at time of the measurements the barometer was already covered by foam and had shown reasonably stable altitude holding properties. As an intermediary summary, the estimated position from DUNE stayed within about ± 0.25[m] in North-East direction, and the height estimate within ± 3.0[m] of the actual position, while the drone was kept stationary and with abundant satellite cover.

From the discussion of sensor data, and state estimates thus far, part of the challenge of implementation of the vertical stack becomes clear. The high degree of precision required for realization is difficult to attain, although a few redeeming points working to the systems advantage should be pointed out. From the error sources noted in section 2.5 one would expect roughly the same pseudo-range errors for UAVs in such close proximity as the vertical stack formation demands. This fact would mean that despite the entire stacks position estimate drifting slightly

as seen in Fig. 5.7, the relative drone-to-drone position would not necessarily be influenced to the same degree, as both UAVs measurements are affected by the error sources almost equally.

It would be rash not to point out some difficulties which are likely to appear in a vertical stack implementation, drawn from the experience of these sensor experiments. Barometer sensitivity could prove a significant issue, as it could also be influenced by the UAV downwash altering its pressure measurements. For the UAVs to effectively interconnect mid-air, a precise height estimate would be vital, and downwash influencing barometer readings might add to the difficulty of this process.

6.1.3 Flight Behavior

Fig. 5.9 shows a single UAV in loiter mode, where it attempts to hold its current position in a steady hover. Fig. 5.10 shows a comparison between the RTK receiver measurements and the DUNE estimated state for the same time period. The tendency that the filtered results of the DUNE provide a compact and steady line, compared to the RTK scattered solutions remain as in the previous section. This loiter flight and position measurement session was successfully executed over approximately 1 minute, in gentle wind. As illustrated by the DUNE estimates, the position was not kept entirely constant during the session. Numerous flights were conducted with either drone and based on these flights a discussion can be conducted regarding the qualities and shortcomings of the implemented platform.

The constructed UAVs both actuate correctly, and the flight characteristics of the system were usually responsive and quick. All onboard components were powered properly and did not show problems of voltage drops as one might encounter during intense motor current draw. The compact form factor made them convenient for transportation, and the open frame structure provided access to the components for debugging and inspection readily. Despite three separate low-altitude crashes on grass, the system remained undamaged, showing the drones to be quite sturdy. All segments illustrated in the system architecture shown in Fig. 3.1 communicated successfully, without problems in this respect.

There were, however, multiple difficulties involved with the drones as well. The first and foremost proved to be unreliable autonomous flight. By unreliable we mean that the transient problems surrounding the navigation were encountered multiple times, but not necessarily in a reproducible manner. Time and effort was put towards finding the sources of these issues with varying degrees of success. The problems could manifest as either drone flyaway, meaning the UAV simply began to move off the desired course, or the drone spiraling outwards which also resulted in drone flyaway after some time. As intended, manual pilot takeover could usually bring

the UAV to a safe landing when problems became visible. The main suspect for these problems was the magnetometers, due to Arducopter logs frequently warning about yaw realignment during operation. For this reason, various alternatives for magnetometers were tried:

- Original: Both Pixracer internal magnetometer and Ublox M8N magnetometer
- Alternative: Only Pixracer internal
- Alternative: Only Ublox M8N magnetometer

The fact that the problems were not possible to reliably reproduce made them very difficult to resolve, but since they reappeared at times they could not be ignored. Redoing all available sensor calibrations, and attempting hardware improvements such as moving compass farther away from main power wires, as illustrated by the tall rods upon which the Ublox M8N was placed in drone Fig. 3.6a did not improve conditions.

6.1.4 Realized Drone System - Closing Discussion

Despite the implemented drones showing unwanted behavior, their construction and operation have given a better understanding of the challenges involved in the vertical stack system implementation, which a future implementation might benefit from.

The drone size, although compact, convenient and safe for small-scale testing should ideally have been larger. A larger UAV would not only provide a platform with a better thrust to weight ratio than what was implemented here, but also allow for customizability in the event that sensors were found to be unreliable, or if new components ought to be added. The tight space constraint of the drones as illustrated by the component overview in Fig. 3.6b, resulted in intrusive antenna placement which might have interfered with the physical link connections if connected formation flight had been performed. The UAVs of a future vertical stack should have a thrust to weight ratio of about 2.5 or better, as this might allow one UAV to support the weight of one other connected UAV in the event of sudden system failure of the lower UAV, with some margin for actuation and controlled landing.

The positioning system and antennas showed overall decent behavior, but if the precision is sufficient without further adaption for a real vertical stack implementation is debatable. The height estimates found in Fig. 5.8 don't appear satisfactory if the precision requirement is within $\pm 0.2[m]$. Again, this is a filtered estimate including data from the Pixracers internal sensors, and

with the RTK data gathered using the helix antenna, so it would be of interest to attempt alternative sensors, such as an external IMU, or external barometer to see the effect upon the estimate.

The close grouping of the fix solutions in Fig. 5.6 indicate that an RTK system could likely be sufficiently precise if an adequate number of fix solutions were reliably found. Exactly what precision is necessary for an effective vertical stack system is difficult to ascertain, as illustrated by comparing the direct RTK solutions in Fig. 5.3 with the filtered DUNE positions of Fig. 5.7, proper filtering from multiple sensor types can give far more consistent position estimates.

The ground station and base station both operated as intended, positional data was received from the base station and successfully included into RTKLIB onboard, giving a decent percentage of fix and float solutions as was shown in Table 5.2 without problems. The ground station, that is the computer running Neptus was successfully used to send mission orders to the UAVs, and also continuously received the drone states, thereby fulfilling its purpose.

6.2 Simulation And Model

The simulation results will be discussed beginning with the low-level systems, such as the inner control, and proceeding upwards to the constraints, and formation control subsequently. Thereafter, the downwash model, and the compensation approaches work will be the focus of the discussion.

6.2.1 Single Drone Behavior

Drones	Payload	Link	Mission	DW	Controllers
1	No payload	No link	Spiral	No	Default

Table 6.1: Plot specification table to Fig. 5.11.

To make the distinction between the various simulations clearer, tables such as 6.1 are included whenever there is a change of simulation configurations. The column **DW** refers to downwash and **Controllers** to the controllers in use for the simulation. By the table entry Default in controllers column, we mean that the inner control and the formation control are active as they were described by section 4.2.2 in chapter 4.

The overall behavior of the single UAV displayed in Fig. 5.11 where the UAV was ordered to spiral is smooth and regular. From the spiral path, it can be seen that the issue concerning the

yaw angle ψ transitioning between $\pm\pi$ was indeed resolved by the modulo based solution described in section 4.2.1. It can also be observed that the inner controller did indeed apply the commanded input force, albeit with a slight delay. The delay can be seen from the solid line lagging slightly behind the dotted in the top right plot, illustrating applied force $\boldsymbol{\tau}_{in}$ versus commanded force $\boldsymbol{\tau}_{cmd}$.

The formation system described in section 4.2.2 was in use in this single drone case, and from that, it can be seen that the special case of only a single UAV using the formation scheme was handled successfully. One discrepancy can be seen in the bottom right plot, where the UAV despite having integral effect in its velocity controller, never exactly reaches the desired velocity. This was a result of the continuously changing reference velocity. One final detail from Fig. 5.11 should be highlighted, the system did compensate for gravity as is seen from the top right plot. The weight of the UAVs means a constant force of about $6[N]$ has to be applied in upwards direction to prevent falling.

Drones	Payload	Link	Mission	DW	Controllers
1	$0.3[kg]$	$0.8[m]$	Abrupt turn	No	Default

Table 6.2: Plot specification table to Fig. 5.12 and Fig. 5.13.

Proceeding to Fig. 5.12, a simulation showing a single drone with a suspended payload is seen. This is the minimal case where the constraint calculations are in use, due to the hanging payload. The UAV was given a desired velocity of $3[m/s]$, which at $t = 10[s]$ was changed to $-3[m/s]$. The simulation was done carrying a medium payload of $0.3[kg]$, as indicated by the payload color.

The motivation for creating and including this plot was to inspect the constraint calculations detailed in section 4.2.1. From the main plot, it appears that the link length was kept constant and that the system allowed the payload and joint to swing as we might expect from the real system. The UAV was able to turn rapidly at the intended time, but the momentum of the payload made it swing out in a wider arc. The angle reached about 38°, and was dampened quite quickly.

Fig. 5.13a corresponds to the simulation in 5.12, and from inspecting the plot the properties of the $\boldsymbol{Q}_c$ term found by solving the Udwadia-Kalaba equation can be discussed. There were three bodies which were interconnected, the drone, joint, and payload. It can be seen that in downwards direction the drone experienced about a $4.3[N]$ force from the constraints. As intended, this force magnitude is the sum of the force acting from gravity on the joint and payload

pulling on the UAV. An increase of the constraint forces is displayed in Fig. 5.13a, this corresponds the UAV having to accelerate itself and the payload to reverse velocity.

From observation of Fig 5.13b a plot of the discrepancy between the original $0.8[m]$ link length, and the distance of the connected bodies over time can be inspected. $\boldsymbol{Q}_c$ roughly maintains the connecting constraints as is illustrated in Fig. 5.12, however, a slight discrepancy in this calculation is to be expected from numerical errors. These numerical errors were accumulated over time by the integration. However, over the $15[s]$ simulation, the accumulated error was minimal, as can be seen from the 10^{-8} unit of the y axis. Also interesting is how the rate of numerical error increased as the UAV abruptly turned at time $10[s]$. The exact reason for this is unknown, however it is suspected that the pseudo-inverse calculation becomes more complicated when the payload strains constraints in multiple directions, increasing the numerical error.

The model for the individual drone with inner control and constrained payload appears to function properly. With the observations from the preceding discussion, the inner control system for the single UAV was regarded as sufficient to proceed to more involved problems, namely the multi-UAV systems.

6.2.2 Multi-UAV Behavior

Drones	Payload	Link	Mission	DW	Controllers
3	No payload	No link	Formation reach	No	Default with variations of integral effect.

Table 6.3: Plot specification table to figures 5.14, 5.15 and 5.16.

By temporarily disabling the connecting link forces, and giving three drones random starting positions with a formation goal velocity of $\boldsymbol{v}_d = [1,\ 0.5,\ 0]^T$, the plots displayed in Fig. 5.15 and 5.14 were created. The velocity controllers integral action was active and inactive for Fig. 5.15 and Fig. 5.14 respectively. By comparing the two figures, an issue related to the formations velocity controllers integral action is shown. The formation as a whole did not reach the target velocity $\boldsymbol{v}_d$ without integral action active, as can be seen by the centroid velocity plotted in Fig. 5.14. If integral action is in effect, the UAVs did not converge to the desired formation topology, seen from the constant formation errors in Fig. 5.15.

In section 4.2.2 it was stated that both the ability to reach formation topology and the desired formation velocity are requirements for the system; thus a compromise must be made to resolve

this issue. Intuition from the field of control theory would suggest that another layer of integral action should be added to the formation control system, however, this was found to worsen the system behavior overall, likely due to violating the passivity requirements of the formation control system. For this reason, an approach which gradually activates the integral effect as the vertical stack reaches the desired topology was implemented. This was done by multiplying the integrand term within the formation velocity PI controller by the weight $b(\boldsymbol{z}, \boldsymbol{z}_d)$, yielded by Eq. 6.2. In essence, this only activates integral effect gradually once the formation is adequately close to its desired topology.

$$c(\boldsymbol{z} - \boldsymbol{z}_d) = (\boldsymbol{z} - \boldsymbol{z}_d)^T (\boldsymbol{z} - \boldsymbol{z}_d) \tag{6.1}$$

$$b(\boldsymbol{z}, \boldsymbol{z}_d) = 1 - max(0, min(c(\boldsymbol{z} - \boldsymbol{z}_d), 1)) \tag{6.2}$$

The result of this change can be seen in Fig. 5.16, where the integral effect is gradually activated as the total error between desired formation topology and current topology is below 1 meter total near time $25[s]$. There remains a slight position error, and the system requires more time to converge to target state compared to the always active integral case, but the system was deemed acceptable to proceed. This formation reach problem was tested for a wide range of initial positions, and the stack formation and velocity was reached with relative ease in all cases. This shows the robustness of the implemented formation system.

Drones	Payload	Link	Mission	DW	Controllers
2	$0.4[kg]$	$0.8[m]$	Traverse square	No	Default with gradual integral.

Table 6.4: Plot specification table to Fig. 5.17.

Next, we consider the situation where a two-drone stack has reached target formation and been interconnected, then picked up a $0.4[kg]$ payload. Next, the stack is given the task of traversing the vertices of a $7[m] \times 7[m]$ square, with a gradual $1.5[m]$ climb, and the resulting behavior can be seen in Fig. 5.17. The main motivation for including this plot was to showcase that the constraint calculations allow the payload burden to be shared by multiple UAVs. This is indeed the case as seen from the lower right plot. The vertical force applied by the UAVs is steadily right above $8[N]$, meaning both UAVs contribute with approximately half of the payload burden in addition to compensating for their own weight.

More observations about the system can be made from Fig. 5.17, for instance, that the UAVs handle the climb imposed by the trajectory well. It can also be seen from the payload angle plot included in the top right corner that the gradual turns from the guidance system, coupled with

the relatively slow speed of $1[m/s]$ gives only minor payload swing.

Drones	Payload	Link	Mission	DW	Controllers
10	$2.0[kg]$	$0.8[m]$	Traverse square	No	Default with gradual integral.

Table 6.5: Plot specification table to Fig. 5.18.

Presented in Fig 5.18 is a ten-drone vertical stack given the same trajectory as the previous two-drone stack. In this situation, a heavy payload of $2.0[kg]$ is attached to the bottom UAV. A slightly oscillating trajectory can be seen for the UAVs higher up in the stack, however, the system does traverse its intended path with reasonable precision. This gives significant contrast to the previous model, which was detailed in appendix B, as a drone stack this large would be very computationally demanding to simulate.

The model has at this point has not been extended to include the effects of drone downwash, which would very likely make the behavior worse and less effective than what is shown here. The attached payload is currently about 5 times heavier than the maximum lifting capacity of a single drone, giving testimony to the increase in lifting capacity resulting from the cooperative drone use. Each UAV can be seen to contribute with about $2[N]$ to the payload from looking at the lower right plot. In a real system, this would mean that the drones were operating at a far more sustainable intensity-level than five drones lifting at maximum capacity.

6.2.3 Downwash Influence

From this point on, we begin to consider the effects of rotor downwash upon the system dynamics. The magnitude and direction of the downwash force calculated as described in Eq. 4.54 is plotted in Fig. 5.2.3. Along the line where $o_{xy} = 0$, meaning the middle of the downwash stream, the intensity can be seen to gradually decrease with greater distance O_z. This was expected from the $(F_{max} - m(o_z))$ term from section 4.4. The decrease in severity can also be seen to be as intended as the lower UAV moves along the o_{xy} direction.

Drones	Payload	Link	Mission	DW	Controllers
3	$0.4[kg]$	$0.8[m]$	Z turn	No	Default with gradual integral.

Table 6.6: Plot specification table to Fig. 5.20.

Fig 5.20 displays a three drone UAV stack traversing a two-turn path, and is intended as a control for comparison before downwash is added to the system. This can be seen from the zero downwash force. Two new types of plots have also been added, giving an estimate for how much energy the UAVs have expended, and the ratio compared to the average energy spent. It should be kept in mind that the energy is simply an accumulated value for thrust intensity, as was described in Eq. 4.57, which is also the reason for the lack of a proper unit along the y axis. All the connected UAVs appear to use the same amount of energy during the path being traversed in this case.

Drones	Payload	Link	Mission	DW	Controllers
3	$0.4[kg]$	$0.8[m]$	Z turn	Yes	Default with gradual integral.

Table 6.7: Plot specification table to Fig. 5.21.

Comparing the previous plot with Fig. 5.21 gives insight into how the downwash influences the system. The main 3D plot shows how the stack drops significantly in the very beginning of executing the waypoint tracking. This is because the downwash force has not been accounted for yet. The stack recovers over time due to the integral effect, and despite showing worse behavior than the zero downwash case in Fig. 5.20 it successfully traverses the path. Interestingly the UAVs still expend more energy, but equal amounts, indicating that the efficiency loss is distributed among the cooperating drones. This makes sense as the integral effect of the centroid velocity controller affects all the UAVs equally.

The two upper plots on the right hand side show how the downwash forces have the highest impact on the lowermost drone C, as it is subjected to the downwash of both drone A and B. The downwash can be seen to apply a total force of about $3[N]$ downwards on C, meaning that its lifting capability, in this case, would be substantially reduced. This is clearly an issue, and coupled with the worsened behavior it is easily perceived that system changes should be made to alleviate the problems.

Drones	Payload	Link	Mission	DW	Controllers
3	$0.4[kg]$	$0.8[m]$	Z turn	Yes	Default with gradual integral, and downwash feedforward.

Table 6.8: Plot specification table to Fig. 5.22

Solutions to the poor behavior are considered. The feedforward approach described in section 4.4.1 is applied, and the implementation gives the behavior shown in Fig. 5.22. In the main

3D plot, the problem of the immediate drop in vertical direction seen before has been resolved. The overall trajectory traversing appears to be improved compared to the previous setup. It can also be seen that the downwash force has about equal magnitude this time and that it still varies with high frequency. The rapid variation of $|\boldsymbol{F}_{dw}|$ likely comes from the lower UAVs moving in and out of the center of the downwash stream, and from the upper UAVs changing their orientation.

Also of interest in this plot is the accumulated energy and energy ratio plots in the lower right-hand side. They indicate that there is a significant difference in the energy expenditure of the UAVs with the feedforward compensation implemented. UAV C which was most prone to the downwash effect has to spend about 10% more energy for operation, which could be problematic. A cooperative scheme such as the vertical stack is in a sense limited by its "worst" cooperating agent, as was mentioned in section 4.4.2. In practical terms, this would mean that the lifetime of the formation as a whole would be decreased by this difference in energy expenditure.

6.2.4 Payload Distribution

Drones	Payload	Link	Mission	DW	Controllers
3	$0.4[kg]$	$0.8[m]$	Z turn	Yes	Default with gradual integral, downwash feedforward, and payload distribution.

Table 6.9: Plot specification table to Fig. 5.23

In response to the problem revealed from the feedforward implementation causing uneven energy expenditure, an additional control layer was added. The idea is to see the vertical force applied as the difference variable in an agreement problem, somewhat like the position in the formation system. This should pull the drones to an agreement on UAV vertical thrust as well as position. Results from the implementation can be viewed in Fig. 5.23 where the system now include downwash, feedforward downwash compensation, and the payload distribution system. Then is ordered to traverse the same trajectory as before.

The benefits of the distribution system can be seen in the energy expenditure plots. Compared to the previous system with only downwash compensation implemented, the energy expenditure has in fact brought drone energy expenditure notably closer, without increasing the actual maximum energy spent. From the top right corner plot, the vertical force demanded by drone A is indeed increased comparing to Fig. 5.22, meaning that drone A is contributing more,

as was the intention for the system. Unfortunate effects of the implementation were noted in the main 3D plot, where the stack rises slightly in the beginning, before tracking the trajectory sensibly.

In a real-world implementation, a similar system might be necessary such that the UAVs intelligently share the payload burden. The reason for this statement is the positioning evaluation, even if the system was to reach the $\pm 15[cm]$ precision displayed by the RTK fix solutions in Fig. 5.6, equal contribution as was seen in the early simulations before downwash would be very difficult. This position error would likely mean that the interconnected drones would actuate to correct their perceived error in the formation, thereby unevenly contributing to the cooperative lift. A variant of the payload distribution system in place should reduce this problem, by having the UAVs effectively agree upon a more equal distribution.

6.2.5 Choice Of Link Length

So far the link length $0.8[m]$ has been used, giving a $1.6[m]$ distance between the drones. As can be seen from Fig. 5.2.3, the downwash is substantial for this distance, however, the length was chosen such that the issues which originate from downwash could be highlighted as mentioned earlier. At this point, it is clear that the effect of downwash is significant, and that reducing its influence is among the chief concerns, in order to make the vertical stack a viable lifting scheme.

Drones	Payload	Link	Mission	DW	Controllers
10	$2.0[kg]$	$0.8[m]$	C turn	Yes	Default with gradual integral, and downwash feedforward.

Table 6.10: Plot specification table to Fig. 5.24.

Fig. 5.24 shows how the downwash hinders a ten drone stack system from maintaining altitude while carrying a heavy $2.0[kg]$ payload. Both the downwash plots from Fig. 5.2.3, and intuition about rotorcraft point to extending the connecting links as a decent remedy.

In Fig. 5.25 the **total downwash** force upon the various drones of the ten drone stack when all drones are kept at 0° degree angle, and links are fully extended is displayed. The plot includes the result for various choices of link length. The interesting property here is how the downwash force acting upon the lower drones becomes the same constant, this is due to the exponential decrease of the downwash. By increasing link length the downwash effect is greatly reduced, and almost all UAVs can operate in the same downwash intensity.

Drones	Payload	Link	Mission	DW	Controllers
10	2.0[kg]	1.0[m] and 1.2[m]	C turn	Yes	Default with gradual integral, and downwash feedforward.

Table 6.11: Plot specification table to Fig. 5.26, and Fig. 5.27.

The same setup as the ten drone stack which was unable to maintain altitude earlier in Fig. 5.24 was tried with link length 1[m], and 1.2[m]. The increase in link mass is automatically accounted for by the simulation, the mass per meter $\rho = 0.02125[kg/m]$ from table 3.1 was used. From the results in Fig. 5.26, we see that link length 1.0[m] has the actuation of the drones borderline saturated, the turn is executed successfully albeit with some oscillations likely due to the majority of the actuation being directed upwards to compensate for weight and downwash. Fig 5.27 shows far better behavior with link length 1.2[m], none of the UAVs appear saturated, and the maximum total downwash experienced by any single drone is at around 1.5[N].

Of course, there are practical aspects to consider with the stack becoming too tall, depending on the environment which the system is to operate within, and problems long connecting links could pose for interconnection becoming difficult. The observations and the discussed downwash research, a simple, yet essential property inherent to the vertical stack is underscored. The **downwash properties** and **environmental constraints** of the drone system used in a vertical stack formation will dictate what will be a suitable length of the connecting links. An ideal implementation of the vertical stack system would have the drones themselves designed such that the downwash thrust is not directed directly at the cooperating drones below.

6.2.6 Model Realism

The approximations and simplifications made in the development of the model necessitate some justification to argue for the overall realism of the simulated results. The areas to which the most noteworthy approximations have been made will be discussed point by point.

- Attachment point in the mass center
  The model assumes that the connecting links are attached to the drones in their mass centers. This means that no torques are created through the constraint forces, and depending on the situation this could make the model differ significantly from reality. The lack of torque from constraints is especially clear when inspecting the drone angle plot in Fig. 5.12. The payload was shown to pull quite heavily on the drone in the turn from Fig. 5.13a, but the roll angle ϕ of the UAV only shows a very small change. If the payload were attached 0.1[m] below mass center, this would likely result in a pronounced roll and pitch

angle spike. This was however deemed to be an acceptable approximation, as shown by the other 3D plots throughout the results, little payload swing is seen. This means that the distance vector of the torque would be near zero in $\{b_x\},\{b_y\}$ direction, making the effective torque small.

- Downwash model

  For the downwash model, there are a few points which ought to be mentioned. First of all the effect of downwash is simply summarized in the force vector $\boldsymbol{F}_{dw}$ acting upon lower drones. This was a conjecture formed from two points, the measured reduction in maximum thrust of the lower drone as described in section 3.6.1, and the empirical data of a quad-rotors downwash air-stream velocity from [4] for a slightly different quad-rotor. The realities of aerodynamics are more complicated than this, although the end result should be possible to represent by a force vector such as $\boldsymbol{F}_{dw}$. Distinctions such as turbulent or laminar flow are likely to mean much for the downwash. For this reason, a deeper analysis of the aerodynamic interaction, either through close proximity coordinated flight, or detailed numerical modeling of the airstream should be performed. If sufficient control had been achieved with the physical drones, this effect would have been studied for the implemented drones in greater detail for this thesis.

- Downwash feedforward

  The feedforward control approach described in section 4.4.1 depends upon knowledge of how severe incoming downwash acting upon the drone is. This means that either a precise enough model estimating the incoming downwash force from the angle and position of the upper UAVs is needed, or sensors capable of measuring the airstream reliably, and from that estimate effective downwash. Either way, the controller might be demanding to realize.

- Connecting link point-mass

  The physical links interconnecting the drones are represented by a single point-mass, located at their point of connection as was illustrated in Fig. 4.2. It might be more precise to model the links as rigid bodies, as was done in the preceding project thesis [42]. This approximation was applied based upon two main arguments, first that the connecting links are not rotating much in the constrained case, meaning that rotational inertia is not of great interest in this case. The second argument is that the connecting point-mass is given the combined mass of the links, and also a degree of air-resistance, thereby representing the links effect upon the system while making simulation considerably more efficient.

- Payload point-mass

  The payload was also represented by a point-mass with its own adjusted air resistance.

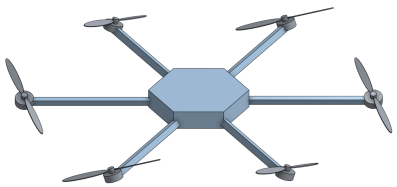
The reality would rather be a body with its own attributes of rotational inertia and air resistance. The gravitational pull was the most important aspect from the payload, meaning that a point-mass was seen as an acceptable substitute in the model.

6.2.7 Potential System Extensions

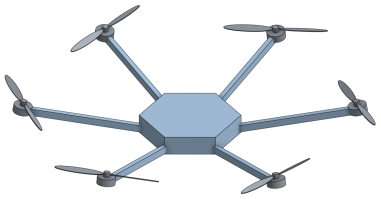
This section is included to describe a few system-level changes which might have significant benefits to the system. It is included here, rather than in future works because it is seen as a useful part of the discussion on the vertical stack.

Changing Motor Orientation

The disadvantages of downwash have been discussed at length, it reduced overall system efficiency, maximum payload, and it appears to make smooth movement of the vertical stack more difficult. For this reason, drone platform changes should be considered to lessen the downwash overall impact on the system. Research has been conducted on alterations to the conventional multi-rotor systems, where the orientation of the drone motors is changed to give interesting properties to the system. Examples of this research is [43], [44] and [45] all showing interesting variations and analysis of the concept. The idea is perhaps best illustrated by Fig. 6.1a and Fig. 6.1b.



(a) Motors angled sidewards, detailed in [45], [43].



(b) Motors angled outwards as studied by [44].

Figure 6.1: Hexarotors with rotated motor arms to redirect downwash and gain sideways actuation.

While some of the lifting capability is lost through altering orientation of the motor arms as indicated, there might be two major advantages to this approach.

- Directing downwash away from lower UAVs.
- Possibility to actuate sideways without changing UAV orientation.

Including Visual Feedback

The RTK positioning system discussed and described earlier has the disadvantage that the base station is needed as a reference point, and continuous communication with the base station is

needed. The range of the compensation is limited, and one might require multiple base stations if operation was to be extended past this range. An alternative to the RTK positioning system might be a visual-based control system, somewhat like what is described in this article [46] and this paper [47]. The book *Quad Rotorcraft Control: Vision-Based Hovering and Navigation* [12] was also found to be promising for the concept. The general idea being that by using visual data from a camera mounted below each UAV, three things might be possible:

- Precise relative positioning of the UAVs, without dependence on RTK precision.
- Detecting position and orientation of the connecting links, for swing dampening of the hanging links before and during interconnection.
- Reliable payload detection and pickup.

All of these points are matters which demand more research to realize, but they would make for interesting extensions to most UAV lifting systems.

The Y Extension

One potential strength of the vertical stack is the inherent redundancy properties of the formation. By redundancy, we mean that in the case where a UAV in the stack fails, it will simply become inactive and hang as a connecting joint in the stack. This is a central prerequisite for the modularity of the system. It was briefly touched upon earlier when discussing the thrust to weight ratio of the UAVs, that if the ratio is greater than 2.5, it could be supported by only one or more of the other drones in the stack. The disabled UAV would, of course, add weight to the stack, making the supporting UAVs have to lift a greater load, however, it would be a useful failsafe for the system. This redundancy property was part of the study done in the project thesis [42].

An issue with this redundancy property is the fact that if the top drone in the stack fails, it would likely be very hazardous for the formation as a whole. Due to the identical link lengths, it would very likely collide with other cooperating UAVs, and cause a fatal system failure. In an effort to resolve this problem, a draft to an extension to the vertical stack was planned as part of this work. We refer to this as the *Y extension* due to the general shape of the altered formation. The idea itself is quite straight-forward and is easily illustrated by Fig. 6.2a and Fig. 6.2b.

By making the lower of the top connecting links 150% longer than the two upper links, the failing UAV should avoid collision with the next UAV in the stack. The intention here is to eliminate the single point of failure where the top drone suddenly shuts down. The apparent disadvantage by this approach is that the system moves in the direction of a flat lifting scheme again, and in general a less modular system structure.

(a) Illustration of the *Y extension* during operation.

(b) Illustration of the intended behavior of the *Y extension* during operation.

Figure 6.2: Illustration of the *Y extension* in the case of one of the top UAVs failing.

We will not go into greater depth into this concept in this work to keep focus on the vertical stack itself as a formation.

6.2.8 Simulation And Model - Closing Discussion

Upon the discussions, and observations presented so far in this chapter, we should be able to broaden our perspective and discuss the vertical stack formation as a whole. While conclusions are intended to be largely left to the next chapter, a more general discussion about the vertical stack has not yet been given.

The implemented formation control scheme showed overall decent behavior, and with the gradual integral effect approach, we get the formation to track both target velocity and reach target topology. Aspects of the problem such as collision avoidance have not yet been accounted for. The formation control system seems apt to keep the vertical stack in order and to execute planned coordinated movements. In the preceding two-drone stack system introduced in [42] the cooperating UAVs were simply given the exact same commands and had individual altitude controllers keeping them at constant heights. The contrast is appreciable, as with the newly proposed system even large vertical stacks such as was shown in Fig. 5.27 are capable to maneuver well.

Downwash and its implications have been studied and considered in numerous ways, and while the model implemented for the downwash is not perfect, it shines light upon the problem itself, and potential solutions. Compensation against it appears to create uneven energy expenditure, but from the result of the load sharing system in Fig. 5.23, the severity of this prob-

lem was shown to be possible to reduce. One problem with the load sharing system was that it was sensitive to tuning, meaning that the constant δ_{lift} mentioned in section 4.4.3 had to be adjusted when changing the size of the drone stack, also it did not eliminate the difference in energy expenditure. Both of these facts seem to underline that the payload sharing concept has potential, but further development remains before that segment of the control structure works in combination with the remaining system.

6.3 Problem Description - Closing Discussion

Here a few closing paragraphs are offered in order to bind the findings discussed so far in this chapter even closer to the questions which this thesis has aimed to answer.

- For the vertical stack, what will be a suitable length for the connecting links to minimize vertical thrust loss while maintaining system compactness?

The ideal choice of link length is dependent on **downwash properties**, **environmental constraints**, as was stated in section 6.2.5. Increasing the length of the connecting links implies a linear increase in weight, but also a rapid reduction in downwash severity, as illustrated by Fig. 5.25. By environmental constraints, we mean issues such as space constraints where the stack, becomes too high. Downwash properties are determined by the choice of cooperating drone, its applied propellers and how it is designed to direct downwash air current.

- How can a formation control system be designed to make any number of drones reach the desired vertical stack formation, and move as a group in a coordinated manner?

The passivity based formation control system introduced in section 4.2.2, is an adequate approach to achieving a working control system. With the gradual integral effect adaption mentioned in section 6.2.2, it was found to give satisfactory behavior for a wide range of conditions. The system functioned as intended for 1, 2, 3, and 10 interconnected drones, and might work well for greater numbers as well.

- To which degree is uneven thrust a problem for the vertical stack, and how can the system be made to distribute the payload mass equally among the drones?

Initially, the drones cooperating in the multi-UAV simulations showed identical levels of thrust, however, with the inclusion of downwash, and downwash compensation the contribution of the cooperating UAVs began to differ, as seen in Fig. 5.22. The 10% increase in the energy expenditure of the lower drone relative to average in the stack illustrates how even in such a small

vertical formation, the difference could be significant. The payload distribution system detailed in section 4.4.3, was shown to reduce this discrepancy notably, as seen in Fig. 5.23. Payload distribution is a suitable problem for this type of formation agreement system, where the cooperating UAVs are made to contribute more evenly.

Chapter 7

Closing Summary And Conclusions

This thesis has continued the investigation into an alternative cooperative lifting approach for several multi-rotor drones. In order to resolve problems from a previously implemented modeling system [42], a new and significantly more efficient model for describing the vertical stack system dynamics has been derived. In the new model, any number of UAVs can be included to be part of the lifting scheme. In the lifting scheme, the cooperating drones are interconnected in the vertical formation by rigid links, and finally to a hanging payload forming what is referred to as a vertical stack. Subsequent extensions to the model including aspects such as rotor downwash from one drone on another have also been considered.

In parallel with the derivation of a model, development of a two-drone test system has also been done. The implemented drone system makes use of Real-Time Kinematic positioning technology to accurately estimate drone positions. The realized UAVs have also been designed to provide robustness and customizability, such that they could be used to implement and test the two-drone version of the vertical stack. This customizability is achieved by having an onboard companion computer that runs software known as DUNE, which is suitable for multi-drone systems and which communicates with a common base station. DUNE allows for formation schemes, fail-safes, communication and other functionality to be implemented as needed. Together with the base station software Neptus, a user should be able to have an overview over multiple drones at all times.

From the derived model, and the attributes, limitations, and measurements of the realized drone system a simulation program for the model was implemented. The efficient properties of the derived model allowed this simulation system to be extended to many more drones than that of a previous simulation system, without becoming impractically slow. The simulations were used to design and implement three levels of control systems in order to reach desirable behavior for the vertical stack formation as a whole. First an inner control system for the orientation

and thrust of each UAV. Thereafter a broader formation control system, steering the position of the UAVs in relation to each other. Finally, a guidance system was added to enable traversing of a number of planned waypoints. With the aforementioned model extension including rotor downwash into the simulation, new problems arose and were later met with changes to the formation control systems.

The implemented physical platform was flown successfully manually and with mixed success autonomously. The autonomous flights were found to present unpredictable behavior preventing further progress in this respect despite significant time and effort invested into achieving improved behavior. For this reason formation flight with the full vertical stack has not yet been performed. Despite this, a number of useful observations were made during the implementation. Measurements made using the realized drone platform was used as specifications for the simulations, this way the simulation results should closely reflect the behavior of the real system.

Finally a closing statement, first regarding the properties of the implemented drone system, and then the overall impression of the vertical stack as an approach for multiple UAVs to cooperatively lift a payload.

The implemented drones appeared insufficient for realizing the vertical stack formation. Despite the position measurement systems functioning well, an unknown amount of adjustments and adaptions remained to make the drones capable of flying in the precise and stable manner which was intended for the system. There are also challenges which are inherent with the chosen UAVs. Their small size, protruding antennas, and lack of an automatic latching mechanism for disconnection mean the unreliable behavior might result in a fatal system failure if connected formation flight was tested using the drones. However, these problems are most likely possible to resolve with an improved drone implementation. And further, the implemented drone systems turned out to be very useful in the sense that they gave real insight into the aerodynamic interactions between UAVs, and the challenges faced when implementing a vertical stack system using physical drones.

The overall evaluation of the vertical stack as a cooperative formation is good in most respects. The simulations performed provide a decent foundation from which we can make meaningful statements about the formation approach. We have shown that the novel formation enables multiple UAVs to cooperate when executing a common lifting assignment, thereby extending lifting capabilities. The UAVs were also shown to be able to move as a group in a coordinated manner, and still maintain the formation integrity while traversing a planned path. With suf-

ficient vertical distance between the cooperating drones, the aerodynamic influence between the UAVs gave the impression of being manageable. From this, the vertical stack was found to be promising as a modular and versatile multi-drone system, and worthy of further study and development.

Chapter 8

Recommendations For Further Work

The cooperative vertical stack system represents a promising concept for flexible drone lifting. It also reveals interesting control problems, and much remains to be investigated regarding the possibilities and shortcomings inherent in the system. A point by point summary regarding the work that remains will be given here, together with the motivation for each point, and additional commentary. The points are sorted by priority, starting with what was deemed most realistic and leading up to future extensions.

Formation flight implementation

Completing the DUNE program implementation of the vertical stack system would be recommended as the first priority for the continued development of the system. The two UAVs displayed in this thesis were intended to act as a minimal prototype for the vertical stack, and the fact remains that such a minimal prototype would be a good first step in development. A different choice of drone platform could be selected to give more stable behavior and further insight into more aspects of the influence that the UAVs have on each others and the payload.

Collision avoidance

The formation control system introduced in this thesis did not yet consider collision avoidance, this is a widely researched field and finding a suitable option would likely not prove too difficult. Adaptions to the formation flight system should be made to implement an appropriate option.

Autonomous interconnection

One of the central arguments for the vertical stack is modularity. Without the possibility to connect and disconnect automatically the modularity of the system is greatly reduced. A connection mechanism locking or unlocking the connecting link would make the interconnection possible. The challenge likely lies in accurately moving the link such that the connection can be made.

Fail-safe implementation

This is was placed after autonomous interconnection, as the possibility to disconnect selected UAVs would likely be among the most important prerequisites for the vertical stack to respond adequately to a partial system failure. The stack should be able to handle sudden failure of any of the connected UAVs, and respond in a sensible manner. With sufficient thrust to weight ratio, the physical interconnection could make it possible for connected UAVs to safely land a disabled UAV, such that system damage is kept to a minimum. In the case where the disabled UAV is the one tethered to the payload, it might have to be kept as dead weight in the stack until the operation is complete. These are just initial thoughts concerning fail-safes, but it is a vital aspect of the system, and needs proper consideration for safely developing the vertical stack concept.

Decentralizing the system

All parts of the system should be made to rely on only their own sensor data, and the data gathered from their connection links. This would make the system more robust with respect to communication with ground control and each other. Simulations of partially hindered communication links could be done by altering the incidence matrix $\boldsymbol{D}$ mentioned in section 4.2.2.

Scaling up the vertical stack

Increasing the size and number of drones to cooperate in the vertical stack formation should be done. This is likely to reveal new challenges upon the minimal two-drone case but is considered vital to realize the scalability expectation to the vertical stack formation.

Alternative platform investigation

As was found in this thesis, one of the most significant inherent challenges to the vertical stack is the downwash from one UAV on another, according to the simulation findings. Investigations should be made into UAVs having angled motors, as this would direct the airstream away from the UAVs below. This would reduce efficiency by a constant multiplier, but it might still prove to be an advantageous design choice. This could not only alleviate the downwash issue but also allow the stack to move sideways without the connected UAVs having to alter their orientation.

Hybrid formation research

As mentioned in the introduction, for situations where flat lifting schemes appear superior, individual drones might be replaced by vertical stacks. The next level of formation control research for the system could be how the stacks move relative to each other to make such a hybrid structure possible.

Appendix A

Acronyms

DoF Degrees of Freedom

DUNE DUNE Unified Navigation Environment

GNSS Global Navigation Satellite System

GPIO General-purpose input/output

GPS Global Positioning System

IAR Integer Ambiguity Resolution

IMC Intermodule Communication

IMU Inertial Measurement Unit

LOS Line-Of-Sight

LSTS Laboratório de Sistemas e Tecnologia Subaquática

MAVLINK Micro Air Vehicle Link

NED North-East-Down

NTNU Norwegian University of Science and Technology

PRN Pseduo-Random Noise

PWM Pulse Width Modulation

RTK Real Time Kinematic

SSH Secure Shell

UART Universal Asynchronous Receiver-Transmitter

UAV Unmaned aerial vehicle

Appendix B

Project Thesis Mathematical Model

Model Description

Describing the system dynamics is useful for two main reasons. Having the dynamics enables realistic simulation of the system, which in turn can avoid costly accidents on physical prototypes. The second reason is that it provides mathematical rigor, potential issues can be discovered and accounted for ahead of time with proper theoretical preparation.

The mathematical model quickly becomes large, thus in order to maintain simplicity and compactness a minimal case which still completely exemplifies the steps taken has been chosen as an example. This minimal system consists of two drones connected by two links. It is referred to as the *minimal stack*. An illustration of the minimal stack which will be modelled can be seen in Fig. B.1.
As seen in Fig. B.2 the minimal stack can be thought of as four rigid bodies, each with their respective states. Placing the bodies in the vertical stack formation introduced in chapter 1, they are referred to as Drone A, link 1, link 2 and Drone B from the top to the bottom respectively. All points of connection between the rigid bodies are thought of as being joints providing free rotation. These connections are from Drone A to link 1, from link 1 to link 2, and from link 2 to drone B.

As introduced in table 2.1, the following state vector can be used to describe the position and orientation of rigid body i.

$$\boldsymbol{x_i} = [x_i \quad y_i \quad z_i \quad \phi_i \quad \theta_i \quad \psi_i]^T \quad i \in A, 1, 2, B \tag{B.1}$$

Since $\boldsymbol{x_i} \in \mathbb{R}^6$ this means that the minimal stack has $4 \times 6 = 24$ degrees of freedom before the the holonomic constraints are included. The control inputs for drone j are introduced for the first

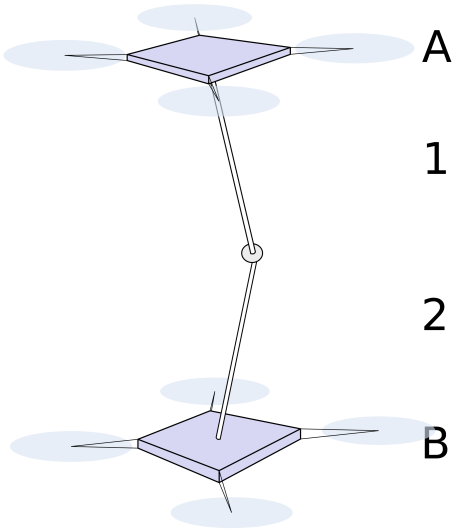


Figure B.1: Illustration of the *minimal stack* showing the index order of rigid bodies.

Input	Description
τ_{jx}	Torque applied to drone $j \in \{A, B\}$ around axis b_x
τ_{jy}	Torque applied to drone $j \in \{A, B\}$ around axis b_y
τ_{jz}	Torque applied to drone $j \in \{A, B\}$ around axis b_z
u_j	Total force in positive b_z direction, working on drone $j \in \{A, B\}$.

Table B.1: System control inputs.

time in table B.1.

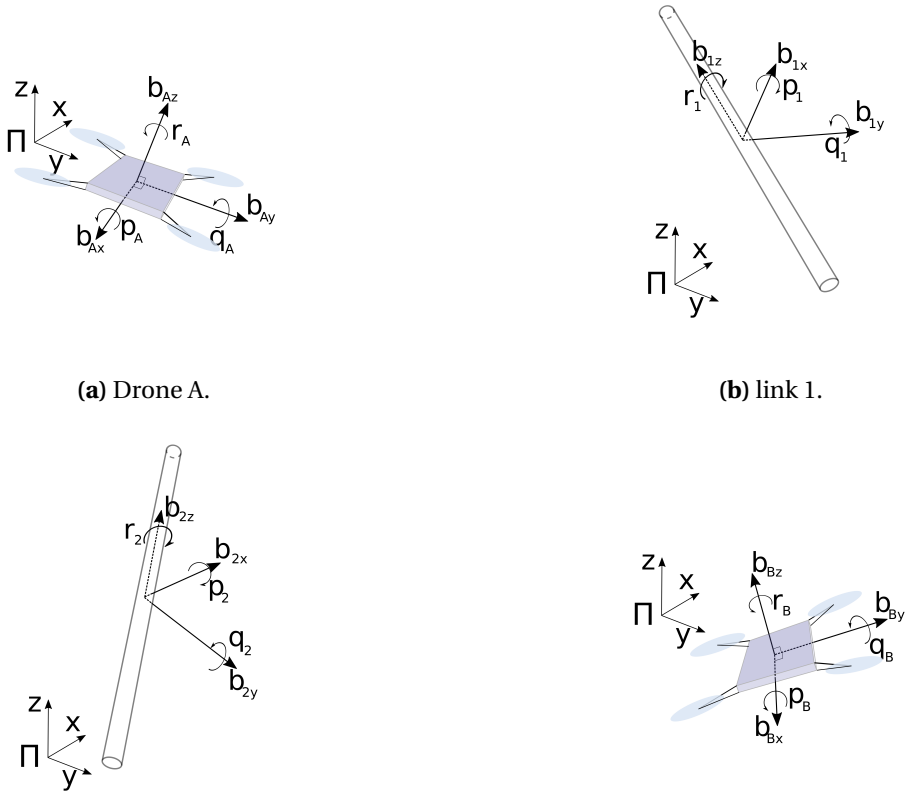
Using rotation matrix 2.1, the force components along x, y and z axis of frame Π from the total thrust u in positive b_{jz} direction have been found as follows.

$$\begin{bmatrix} F_{jx} \\ F_{jy} \\ F_{jz} \end{bmatrix} = R(\boldsymbol{\Theta_j}) \begin{bmatrix} 0 \\ 0 \\ u_j \end{bmatrix} \quad j \in \{A, B\} \tag{B.2}$$

Finally for compact notation, the complete input vector $F_j(\boldsymbol{x_j}) \in \mathbb{R}^6$ for a drone was defined by combining torques from table B.1 and force components from Eq. B.2. The elements of the force vector can be viewed for an example drone j in Fig. B.3.

$$\boldsymbol{F_j} = \begin{bmatrix} F_{jx} & F_{jy} & F_{jz} & \tau_{jx} & \tau_{jy} & \tau_{jz} \end{bmatrix}^T \quad j = \{A, B\} \tag{B.3}$$

The effect of air resistance was included together with the input forces, as both can be seen as



(a) Drone A.

(b) link 1.

(c) Link 2.

(d) Drone B.

Figure B.2: The separate components included in the simplified two drone model.

external influence upon the system. The approximation that air resistance scales quadratically with velocity was used. This resulted in the following vector $\boldsymbol{S_i} \in \mathbb{R}^6$ for rigid body i.

$$\boldsymbol{S_i} = \begin{bmatrix} -\rho \cdot c_{\dot{x}} \cdot \dot{x}_i |\dot{x}_i| \\ -\rho \cdot c_{\dot{y}} \cdot \dot{y}_i |\dot{y}_i| \\ -\rho \cdot c_{\dot{z}} \cdot \dot{z}_i |\dot{z}_i| \\ -\rho \cdot c_p \cdot p_i |p_i| \\ -\rho \cdot c_q \cdot q_i |q_i| \\ -\rho \cdot c_r \cdot r_i |r_i| \end{bmatrix} \quad i = \{A, 1, 2, B\} \tag{B.4}$$

The constants $c_{\dot{x}}, \cdots, c_r$ are coefficients that represent the severity of air resistance expected from certain states, constant ρ represents the density of air. In the real world system these would depend upon the aerodynamics of the rigid bodies, and the properties of the surrounding air.

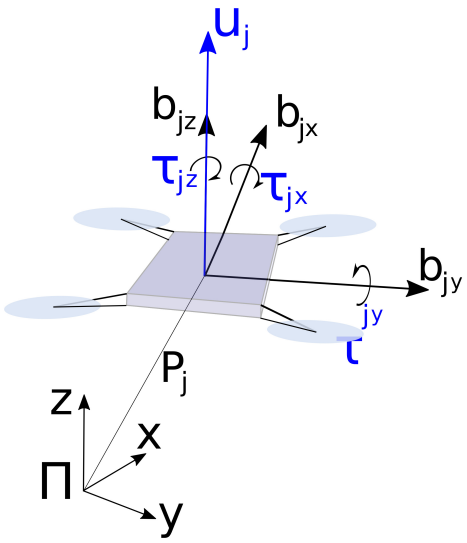


Figure B.3: Illustration of the control input forces and torques.

Physical Properties

Each rigid body i of the minimal stack was assigned a mass m_i and a diagonal rotational inertia matrix $I_i \in \mathbb{R}^3$. The elements along the diagonal of I_i are the rotational inertia around b_{ix}, b_{iz} and b_{iz}. Links 1 and 2 were modelled as cylinders with uniform density, and were given lengths $2 \cdot l_1$ and $2 \cdot l_2$ and radius r_1 and r_2. Drones A and B were modelled as cuboids with uniform density and a motor placed at each vertex. Next these cuboids were given side lengths $2 \cdot l_x$, width $2 \cdot l_y$ and height h as displayed in Fig. B.4. These choices gave the following inertia and mass matrices for the links and drones, relative to their respective body frames.

$$I_{link} = \begin{bmatrix} \frac{m_{link} l_{link}^2}{12} & 0 & 0 \\ 0 & \frac{m_{link} l_{link}^2}{12} & 0 \\ 0 & 0 & \frac{m_{link} r^2 l_{link}}{2} \end{bmatrix} \quad M_{link} = \begin{bmatrix} m_{link} & 0 & 0 \\ 0 & m_{link} & 0 \\ 0 & 0 & m_{link} \end{bmatrix} \tag{B.5}$$

$$I_{drone} = \begin{bmatrix} \frac{m_{drone}(4l_y^2+h^2)}{12} & 0 & 0 \\ 0 & \frac{m_{drone}(4l_x^2+h^2)}{12} & 0 \\ 0 & 0 & \frac{m_{drone}(4l_x^2+4l_y^2)}{12} \end{bmatrix} \quad M_{drone} = \begin{bmatrix} m_{drone} & 0 & 0 \\ 0 & m_{drone} & 0 \\ 0 & 0 & m_{drone} \end{bmatrix} \tag{B.6}$$

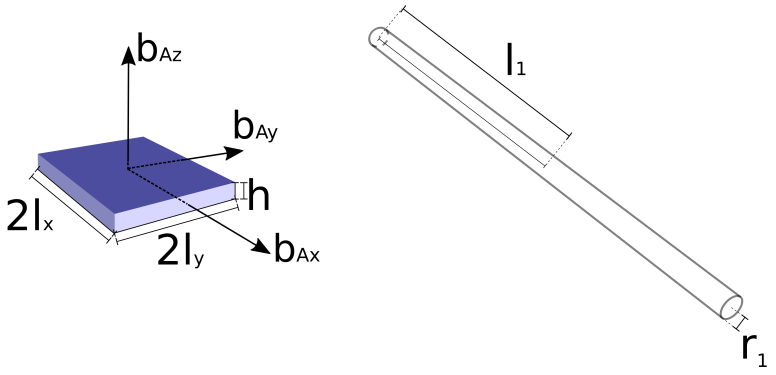


Figure B.4: The rigid bodies have been modelled as simple shapes with uniform density, drone A and link 1 are displayed as an example.

As the inertias in Eq. B.5 and Eq. B.6 are given in body frame, it was necessary to relate this to the generalized angular velocity $\boldsymbol{\Theta}$. This was done by applying the transformation introduced in chapter 2, Eq. 2.4.

$$I_j(\boldsymbol{\Theta_j}) = T^T(\boldsymbol{\Theta_j}) I_{drone} T(\boldsymbol{\Theta_j}), \quad j \in \{A, B\} \tag{B.7}$$

$$I_i(\boldsymbol{\Theta_i}) = T^T(\boldsymbol{\Theta_i}) I_{link} T(\boldsymbol{\Theta_i}), \quad i \in \{1, 2\} \tag{B.8}$$

For the complete system, these inertias and masses were gathered in the following positive definite matrix $M(\boldsymbol{q}) \in \mathbb{R}^{24\times 24}$, by gathering the properties from Eq. B.5, Eq. B.6 as shown in Eq. B.9.

$$M(\boldsymbol{q}) = \begin{bmatrix} M_A & & & & & & & & \\ & I_A(\boldsymbol{\Theta_A}) & & & & & & & \\ & & M_1 & & & & & & \\ & & & I_1(\boldsymbol{\Theta_1}) & & & & & \\ & & & & M_2 & & & & \\ & & & & & I_2(\boldsymbol{\Theta_2}) & & & \\ & & & & & & M_B & & \\ & & & & & & & I_B(\boldsymbol{\Theta_B}) & \end{bmatrix} \tag{B.9}$$

The $\boldsymbol{q}$ variable included in Eq. B.9 is a composite state vector $\boldsymbol{q} \in \mathbb{R}^{24}$ for the entire system, and it was also explicitly defined by combining the four rigid body state vectors $\boldsymbol{x_i} \in \mathbb{R}^6$ of the minimal

stack in order to enable compact notation.

$$\boldsymbol{q} = \begin{bmatrix} \boldsymbol{x_A} \\ \boldsymbol{x_1} \\ \boldsymbol{x_2} \\ \boldsymbol{x_B} \end{bmatrix} \tag{B.10}$$

Equations Of Motion

With the physical properties described and a state vector defined it was possible to proceed with modelling the system dynamics. In [48], it is stated and shown that in cases where the Lagrangian of a system can be written as kinetic minus potential energy, Eq. B.11 can be used to describe the system dynamics of a constrained Lagrangian system. As the vertical stack system simply consists of rigid bodies moving in 3D space, was found to hold for the vertical stack.

$$M(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{N}(\boldsymbol{q}) + H^T(\boldsymbol{q})\boldsymbol{\lambda} = \boldsymbol{F}(\boldsymbol{q}) - \boldsymbol{S}(\dot{\boldsymbol{q}}) \tag{B.11}$$

$M(\boldsymbol{q})$ was defined in Eq. B.9, and the matrices $C(\boldsymbol{q}, \dot{\boldsymbol{q}})$ and $H(\boldsymbol{q})$ as well as vectors $\boldsymbol{\lambda}$, $\boldsymbol{N}(\boldsymbol{q})$, $\boldsymbol{S}(\dot{\boldsymbol{q}})$ and $\boldsymbol{F}(\boldsymbol{q})$ had not yet been properly determined. First input vector $\boldsymbol{F}(\boldsymbol{q})$ for the minimal stack was defined, it represents the control input and external forces, and is a combination of the drone inputs as seen in Eq. B.12.

$$\boldsymbol{F}(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{F_A}^T & \boldsymbol{0}_{1\times 12} & \boldsymbol{F_B}^T \end{bmatrix}^T \tag{B.12}$$

Analogously, the vector $S(\dot{\boldsymbol{q}})$ is a combination of air resistance for the rigid bodies.

$$\boldsymbol{S}(\dot{\boldsymbol{q}}) = \begin{bmatrix} \boldsymbol{S_A}^T & \boldsymbol{S_1}^T & \boldsymbol{S_2}^T & \boldsymbol{S_B}^T \end{bmatrix}^T \tag{B.13}$$

The potential force vector $\boldsymbol{N}(\boldsymbol{q})$, which in this case only represents gravity, was defined from mass of rigid body i, m_i and the gravitational constant g.

$$\boldsymbol{N_i}(\boldsymbol{x_i}) = \begin{bmatrix} 0 & 0 & m_i \cdot g & 0 & 0 & 0 \end{bmatrix}^T \quad i \in \{A, 1, 2, B\} \tag{B.14}$$

$$\boldsymbol{N}(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{N_A}(\boldsymbol{x_A}) \\ \boldsymbol{N_1}(\boldsymbol{x_1}) \\ \boldsymbol{N_2}(\boldsymbol{x_1}) \\ \boldsymbol{N_B}(\boldsymbol{x_B}) \end{bmatrix} \tag{B.15}$$

The term $C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \in \mathbb{R}^{24}$ is the Coriolis term for the system, representing the gyroscopic and centrifugal effects. Coriolis terms arise because of the non-inertial frames which in some cases are

implicit in the use of generalized coordinates. A fortunate side effect of the use of the excessive state modelling approach is that since the states of all rigid bodies in this system were expressed relative to the inertial frame Π the Coriolis term became zero.

$$C(\boldsymbol{q}, \dot{\boldsymbol{q}}) = 0 \tag{B.16}$$

Finally the constraint force matrix $H(\boldsymbol{q})$, with multiplier vector $\boldsymbol{\lambda}$ which accounts for the forces which act upon the system as a result of the holonomic constraints were defined. First the holonomic constraints from which the $H(\boldsymbol{q})$ matrix were derived were specified. The holonomic constraints relate the position of the mass centers of the rigid bodies to one another as illustrated in Fig. B.5, these constraints are referred to as $h_1, h_2, \ldots, h_9$.

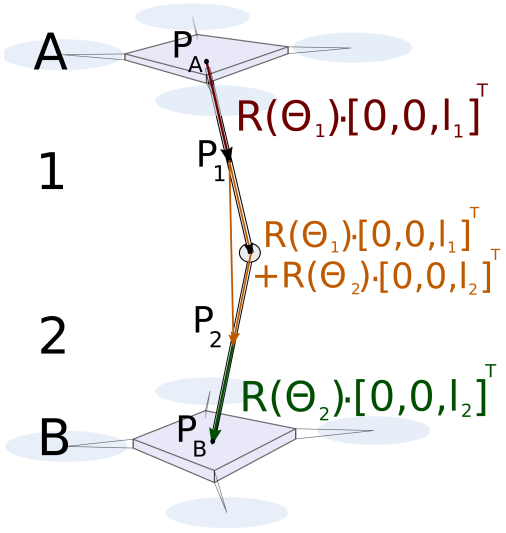


Figure B.5: The holonomic constraints express how the mass centers of the rigid bodies are positioned relative to one another.

$$\begin{bmatrix} h_1 \\ h_2 \\ h_3 \end{bmatrix} = \begin{bmatrix} x_A \\ y_A \\ z_A \end{bmatrix} - \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} + R(\boldsymbol{\Theta_1}) \begin{bmatrix} 0 \\ 0 \\ l_1 \end{bmatrix} \tag{B.17}$$

$$\begin{bmatrix} h_4 \\ h_5 \\ h_6 \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} - \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} + R(\boldsymbol{\Theta_1}) \begin{bmatrix} 0 \\ 0 \\ l_1 \end{bmatrix} + R(\boldsymbol{\Theta_2}) \begin{bmatrix} 0 \\ 0 \\ l_2 \end{bmatrix} \tag{B.18}$$

$$\begin{bmatrix} h_7 \\ h_8 \\ h_9 \end{bmatrix} = \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} - \begin{bmatrix} x_B \\ y_B \\ z_B \end{bmatrix} + R(\boldsymbol{\Theta_2}) \begin{bmatrix} 0 \\ 0 \\ l_2 \end{bmatrix} \tag{B.19}$$

These holonomic constraints were then used to calculate what can be intuitively understood as the direction of the constraint forces as a function of the system states. For the minimal stack this formed a constraint matrix $H(\boldsymbol{q}) \in \mathbb{R}^{9\times 24}$.

$$H(\boldsymbol{q}) = \begin{bmatrix} \frac{\partial h_1}{\partial q_1} & \frac{\partial h_1}{\partial q_2} & \cdots & \frac{\partial h_1}{\partial q_{24}} \\ \frac{\partial h_2}{\partial q_1} & \ddots & & \\ \vdots & & \ddots & \\ \frac{\partial h_9}{\partial q_1} & & & \frac{\partial h_9}{\partial q_{24}} \end{bmatrix} \tag{B.20}$$

Next the vector $\boldsymbol{\lambda}$ needed to be defined. $\boldsymbol{\lambda}$ is a vector containing the Lagrange multipliers which are found such that the holonomic constraints remain satisfied. From equation (6.6) in [48] these Lagrange multipliers are given as follows.

$$\boldsymbol{\lambda} = (HM^{-1}H^T)^{-1}(HM^{-1}(\boldsymbol{F} - C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{N} + \dot{H}\dot{\boldsymbol{q}}) \tag{B.21}$$

By inserting for the zero Coriolis term, and including the air resistance vector $\boldsymbol{S}$ the Lagrange multipliers for the minimal stack were found.

$$\boldsymbol{\lambda} = (H(\boldsymbol{q})M^{-1}(\boldsymbol{q})H^T(\boldsymbol{q}))^{-1}(H(\boldsymbol{q})M^{-1}(\boldsymbol{q})(\boldsymbol{F}(\boldsymbol{q}) - \boldsymbol{S}(\dot{\boldsymbol{q}}) - \boldsymbol{N}(\boldsymbol{q})) + \dot{H}\dot{\boldsymbol{q}}) \tag{B.22}$$

Subsequently the second order derivative of the state vector $\ddot{\boldsymbol{q}}$ was found by rewriting Eq. B.11 as follows.

$$\ddot{\boldsymbol{q}} = M^{-1}(\boldsymbol{q})(\boldsymbol{F}(\boldsymbol{q}) - \boldsymbol{S}(\dot{\boldsymbol{q}}) - C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{N}(\boldsymbol{q}) + H^T(\boldsymbol{q})\boldsymbol{\lambda}) \tag{B.23}$$

With equations B.22 and B.23, and all their terms determined it was possible to numerically simulate the system from a chosen initial state and observe behavior. To perform this numerical

simulation the equations of motion were first found, using a simple change of variables.

$$\tilde{\boldsymbol{q}}_{\mathbf{1}} = \boldsymbol{q}, \qquad \tilde{\boldsymbol{q}}_{\mathbf{2}} = \dot{\boldsymbol{q}} \tag{B.24}$$

Thus from using the change of variables on the the second order differential equations B.23, a first order differential equation with twice as many states was formed. Eq. B.25 was then the final equations of motion for the minimal stack system.

$$\begin{bmatrix} \dot{\tilde{\boldsymbol{q}}}_1 \\ \dot{\tilde{\boldsymbol{q}}}_2 \end{bmatrix} = \begin{bmatrix} \tilde{\boldsymbol{q}}_{\mathbf{2}} \\ M^{-1}(\tilde{\boldsymbol{q}}_{\mathbf{1}})(\boldsymbol{F}(\tilde{\boldsymbol{q}}_{\mathbf{1}}) - \boldsymbol{S}(\tilde{\boldsymbol{q}}_{\mathbf{2}}) - C(\tilde{\boldsymbol{q}}_1, \tilde{\boldsymbol{q}}_2)\tilde{\boldsymbol{q}}_{\mathbf{2}} + \boldsymbol{N}(\tilde{\boldsymbol{q}}_{\mathbf{1}}) + H^T(\tilde{\boldsymbol{q}}_{\mathbf{1}})\boldsymbol{\lambda}) \end{bmatrix} \tag{B.25}$$

Initialization

With the concepts of generalized coordinates, holonomic constraints and degrees of freedom introuduced and briefly explained in chapter 2. It is useful to note how this can be used to simplify state initialization of the minimal stack. The state vector $\boldsymbol{q}$ for the minimal stack contains 24 states, but because of the 9 holonomic constraints it should be possible to express the system state using only $24 - 9 = 15$ generalized coordinates. Hence the constrained minimal stack was found to have 15 degrees of freedom. By using the position of drone A, and the angle of all four rigid bodies, the following generalized state vector was used.

$$\boldsymbol{q}_{\boldsymbol{gen}} = \begin{bmatrix} \boldsymbol{x}_{\boldsymbol{A}} \\ \boldsymbol{\Theta}_{\mathbf{1}} \\ \boldsymbol{\Theta}_{\mathbf{2}} \\ \boldsymbol{\Theta}_{\boldsymbol{B}} \end{bmatrix} \quad \boldsymbol{q}_{\boldsymbol{gen}}, \in \mathbb{R}^{15} \tag{B.26}$$

The positions of link 1, 2 and drone B could then be found by their position relative to drone A, as a function of the angles. For instance the position of link 1 was given as:

$$\boldsymbol{P}_{\mathbf{1}} = \boldsymbol{P}_{\boldsymbol{A}} - R(\boldsymbol{\Theta}_{\mathbf{1}}) \begin{bmatrix} 0 \\ 0 \\ l_1 \end{bmatrix} \tag{B.27}$$

This is exceedingly useful in state initialization, as it does not require the explicit calculation of positions for link 1, 2 and drone B.

Extending The Model

As stated before deriving the equations of motion the method for finding the dynamics of the minimal stack exemplify an extension of the system. A mathematical model for longer chains of

drones and links can be reached by simply adding the states of two more links, and one drone, while including the holonomic constraints between those new rigid bodies in the exact same way as done in the implementation above. The addition of a point mass payload can be done by adding only the dynamics concerning position and velocity, while excluding rotation for the payload. The holonomic constraints are in this case identical to those of the minimal stack.

Appendix C

Project Thesis Guidance System

Path Following

The concept of Line-Of-Sight (LOS) steering has been used in this thesis, and is described in [22], but an explanation of the concept applied to a drone have been included here.

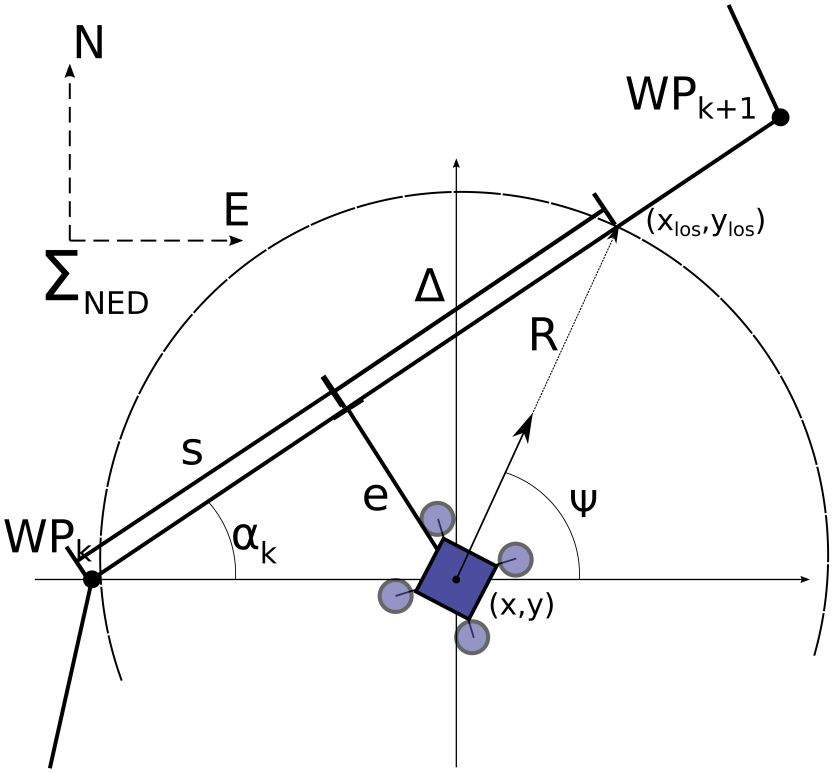


Figure C.1: Concept illustration for the Line-Of-Sight guidance steering law.

The LOS steering law determines the target yaw angle ψ_d of the drone, from a set of predefined waypoints. First the switching radius R is chosen, when the drone comes closer to WP_{k+1} than R, waypoints WP_k and WP_{k+1} become WP_{k+1} and WP_{k+2} respectively. This allows for smooth turning when approaching the waypoint path, and gradual turning when waypoints are switched as the drone is unable to turn instantly. Next a constant Δ is chosen, Δ is known as the look-ahead distance, the length to the look-ahead point (x_{los}, y_{los}) along the line between the waypoints, as illustrated in Fig. C.1.

The yaw ψ of the drone is controlled to direct the drone towards the look-ahead point, this makes the drone approach the line between waypoints with minimal overshoot, as (x_{los}, y_{los}) keeps moving ahead as the drone approaches. This ensures that the cross-track error e is reduced until it reaches zero when the drone reaches the waypoint line. Cross-track error e, angle α_k between waypoints, and finally target heading ψ_d are found using the following equations.

$$e = -[x - x_k]sin(\alpha_k) + [y + y_k]cos(\alpha_k) \tag{C.1}$$

$$\alpha_k = atan2(y_{k+1} - y_k, x_{k+1} - x_k) \tag{C.2}$$

$$\psi_d = \alpha_k + arctan(-\frac{e}{\delta}) \tag{C.3}$$

Appendix D

Passivity Based Formation Control

4.1 Problem Statement

Consider a group of N agents where the variables to be coordinated is represented by the vector $\boldsymbol{x}_i \in \mathbb{R}^p$ and $i \in \{1,\dots,N\}$ denotes each agent. This variable can for example be the position of each agent (in which case $p = 3$). Assuming symmetric information flow, let G be the undirected graph modelling the information topology between the agents. Furthermore, assume that G is connected and has ℓ undirected links. Because of symmetric information flow, the analysis can be simplified without changing the results by considering one of the agents at each link to be at the positive end. G and the orientations of each link k can then represented by the *incidence matrix*:

$$\boldsymbol{D} = \begin{bmatrix} d_{11} & \cdots & d_{1\ell} \\ \vdots & \ddots & \vdots \\ d_{N1} & \cdots & d_{N\ell} \end{bmatrix} \in \mathbb{R}^{N\times\ell} \tag{4.1}$$

where each element is defined as (Bai, Arcak, and Wen 2011):

$$d_{ik} := \begin{cases} +1 & \text{if } k \in \mathscr{L}_i^+ \\ -1 & \text{if } k \in \mathscr{L}_i^- \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

where the set $\mathscr{L}_i^+$ ($\mathscr{L}_i^-$) denotes the links for which agent i is at the positive (negative) end.

The overall goal is to implement distributed coordination laws, using only local information about neighboring agents, that guarantee the following two group behaviours:

A1) Each agent achieves a common velocity vector $\boldsymbol{v}(t) \in \mathbb{R}^p$ prescribed for the group; that is:

$$\lim_{t\to\infty} |\dot{\boldsymbol{x}}_i - \boldsymbol{v}(t)| = 0, \quad \forall\, i \in \{1,\dots,N\} \tag{4.3}$$

A2) If agents i and j are connected by link k, then the difference variable $\boldsymbol{z}_k$ defined as:

$$\boldsymbol{z}_k := \sum_{l=1}^{N} d_{lk}\boldsymbol{x}_l = \begin{cases} \boldsymbol{x}_i - \boldsymbol{x}_j & \text{if } k \in \mathscr{L}_i^+ \\ \boldsymbol{x}_j - \boldsymbol{x}_i & \text{if } k \in \mathscr{L}_i^- \end{cases} \tag{4.4}$$

converges to a prescribed compact set $\mathscr{A}_k \subset \mathbb{R}^p, \quad \forall\, k \in \{1,\dots,\ell\}$.

Note that $\boldsymbol{v}(t)$ can be seen as the common mission velocity to be achieved by the group. Furthermore, note that the target set $\mathscr{A}_k$ may change form depending on the application. For the purpose of this thesis, it can be designed such that the vehicles maintain a prescribed distance, hence keeping a formation. However, as discussed further in

Section 4.4, it is important that the target sets are feasible (Bai, Arcak, and Wen 2011). Introducing the concatenated vectors:

$$\boldsymbol{x} := \begin{bmatrix} \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_N \end{bmatrix} \in \mathbb{R}^{pN}, \quad \boldsymbol{z} := \begin{bmatrix} \boldsymbol{z}_1 \\ \vdots \\ \boldsymbol{z}_\ell \end{bmatrix} \in \mathbb{R}^{p\ell} \tag{4.5}$$

partitioning $\boldsymbol{D}$ in terms of column vectors:

$$\boldsymbol{D} = [\boldsymbol{D}_1 | \cdots | \boldsymbol{D}_\ell] \tag{4.6}$$

and using (4.4), $\boldsymbol{z}$ can be rewritten as:

$$\boldsymbol{z} = (\boldsymbol{D}^\mathsf{T} \otimes \boldsymbol{I}_p)\boldsymbol{x} \tag{4.7}$$

Hence, $\boldsymbol{z}$ is restricted to $\mathscr{R}(\boldsymbol{D}^\mathsf{T} \otimes \boldsymbol{I}_p)$ and thus, the target sets $\mathscr{A}_k$ are only feasible if (Bai, Arcak, and Wen 2011):

$$\{\mathscr{A}_1 \times \cdots \times \mathscr{A}_\ell\} \cap \mathscr{R}(\boldsymbol{D}^\mathsf{T} \otimes \boldsymbol{I}_p) \neq \emptyset \tag{4.8}$$

4.2 The Passivity-based Design Procedure

(Bai, Arcak, and Wen 2011) suggests a two-step design procedure with the objective to render the target sets $\mathscr{A}_k$ in (4.4) invariant and asymptotically stable, using passivity properties. This section first presents each step, before applying them to the system used in this thesis.

4.2.1 Step 1: Internal feedback

*For each agent $i \in \{1, \ldots, N\}$, design an internal feedback loop that renders its dynamics passive from an external feedback signal $\boldsymbol{u}_i$ (to be designed in **Step 2**) to the velocity error:*

$$\boldsymbol{y}_i := \dot{\boldsymbol{x}}_i - \boldsymbol{v}(t). \tag{4.9}$$

Step 1 assumes that the input-output dynamics of each agent are given by:

$$\boldsymbol{x}_i = \mathscr{H}_i^0\{\boldsymbol{\tau}_i\} \tag{4.10}$$

where $\mathscr{H}_i^0\{\boldsymbol{\tau}_i\}$ denotes the input-output dynamics of the system in agent i. Thus, we seek a feedback controller $\boldsymbol{\tau}_i$ for each agent, such that the agent dynamics may be expressed in the form:

$$\dot{\boldsymbol{x}}_i = \mathscr{H}_i\{\boldsymbol{u}_i\} + \boldsymbol{v}(t) \tag{4.11}$$

where $\mathscr{H}_i$ is a strictly passive system that is either *dynamic* or *static*. This transformation is shown in Figure 4.1

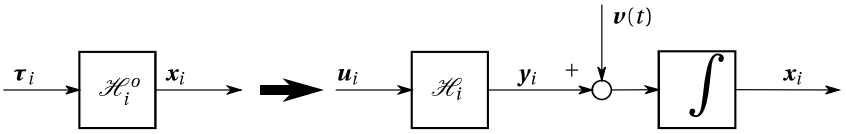


Figure 4.1: Transformation of the dynamics of agent i in Step 1, from (4.10) to (4.11).

If $\mathscr{H}_i$ is dynamic, it is assumed to have the form:

$$\mathscr{H}_i : \begin{cases} \dot{\boldsymbol{\xi}}_i = f_i(\boldsymbol{\xi}_i, \boldsymbol{u}_i) \\ \boldsymbol{y}_i = h_i(\boldsymbol{\xi}_i, \boldsymbol{u}_i) \end{cases} \tag{4.12}$$

where $\boldsymbol{y}_i$ and $\boldsymbol{\xi}_i \in \mathbb{R}^{n_i}$ is the velocity error and state variable of subsystem $\mathscr{H}_i$, respectively. Both $f_i(\cdot,\cdot)$ and $h_i(\cdot,\cdot)$ are assumed to be C^2 functions such that:

$$f_i(0, \boldsymbol{u}_i) = 0 \Rightarrow \boldsymbol{u}_i = 0 \tag{4.13}$$

and:

$$h_i(0,0) = 0 \tag{4.14}$$

As stated, $\mathscr{H}_i$ is designed to be strictly passive. By (Khalil 2002, Definition 6.3) this restricts (4.12) to have a C^1, positive definite, radially unbounded storage function, thus making the origin globally asymptotically stable.

If $\mathscr{H}_i$ is a static block, it is assumed to be of the form:

$$\boldsymbol{y}_i = h_i(\boldsymbol{u}_i) \tag{4.15}$$

where $h_i : \mathbb{R}^p \to \mathbb{R}^p$ is a locally Lipschitz function satisfying:

$$\boldsymbol{u}_i^\mathsf{T} \boldsymbol{y}_i = \boldsymbol{u}_i^\mathsf{T} h_i(\boldsymbol{u}_i) > 0, \quad \forall \boldsymbol{u}_i \neq 0 \tag{4.16}$$

thus making the system strictly passive by (Khalil 2002, Definition 6.1).

4.2.2 Step 2: External feedback

Design the external feedback signal $\boldsymbol{u}_i$ in the form:

$$\boldsymbol{u}_i = -\sum_{k=1}^{\ell} d_{ik}\Psi_k(\boldsymbol{z}_k) \tag{4.17}$$

where $\boldsymbol{z}_k$ is the difference variables defined in (4.4) and $\Psi_k(\boldsymbol{z}_k)$ is a multivariable nonlinearity designed such that the target set $\mathscr{A}_k$ is invariant and asymptotically stable.

Step 2 is basically applying a feedback-term consisting of an input $\boldsymbol{u}_i$ that is the sum of some function Ψ_k of the difference variable $\boldsymbol{z}_k$ for each link k. Note that since $d_{ik} \neq 0$ only if agent i is communicating on link k, this feedback is decentralized and implementable with local information.

By again introducing concatenated vectors:

$$\boldsymbol{y} := \begin{bmatrix} \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_N \end{bmatrix} \in \mathbb{R}^{pN}, \quad \boldsymbol{u} := \begin{bmatrix} \boldsymbol{u}_1 \\ \vdots \\ \boldsymbol{u}_N \end{bmatrix} \in \mathbb{R}^{pN}, \quad \boldsymbol{\Psi} := \begin{bmatrix} \Psi_i \\ \vdots \\ \Psi_\ell \end{bmatrix} \in \mathbb{R}^{p\ell} \tag{4.18}$$

and noting that $\boldsymbol{u}_i$ can be expressed as:

$$\boldsymbol{u}_i = -[d_{i1}\boldsymbol{I}_p|\cdots|d_{i\ell}\boldsymbol{I}_p]\boldsymbol{\Psi} \tag{4.19}$$

the external feedback $\boldsymbol{u}_i$ for all agents can be written as:

$$\boldsymbol{u} = -(D \otimes \boldsymbol{I}_p)\boldsymbol{\Psi}(\boldsymbol{z}) \tag{4.20}$$

Hence, with the applied external feedback and some rearranging, the closed-loop structure for all agents can be represented as shown in Figure 4.2. As mentioned, $\mathscr{H}_i$ is designed to be strictly passive, and pre- and post-multiplying with $\boldsymbol{D}^\mathsf{T}$ and $\boldsymbol{D}$ preserves this passivity. Therefore, the restrictions for the feedback in **Step 2** boils down to designing Ψ_k such that the passivity is preserved from $\dot{\boldsymbol{z}}$ to $\boldsymbol{\Psi}$, rendering the whole interconnected system in Figure 4.2 passive.

Appendix E

Background And Preceding Work

This section is included to provide the reader with information regarding what the contributions from this thesis are, in addition to an overview of which preexisting information, software and equipment the thesis draws from and relies upon. This is recommended to be treated as a reference for the reader to review during or after reading the thesis.

NTNU Uavlab

The Department of Engineering Cybernetics at NTNU provided access to, and some guidance surrounding the LSTS toolchain, which includes DUNE, Neptus and the IMC protocol. Helpful members of the UAVlab have made some of their previous experience regarding RTKLIB and procedures of operation for DUNE available. A github repository was also made accessible, providing software examples as a basis from which some of the programs for this thesis were written. The RTK basestation described in section 3.3.5 was preexisting equipment from previous NTNU Uavlab projects. Kristian Klausen is one of the members of the Uavlab which has done multi-UAV research which can be seen related to the work done in this thesis. His work modelling constraint forces with wires using the Udwadia Kalaba equation was the main inspiration for the system model implemented in this thesis. He also provided a relatively brief example simulation program illustrating use of this, from which some plotting functionality was reused for this project.

Project Thesis

In the project thesis, which was completed in the fall semester 2017, some preparatory work for this project was done. Outlines for a UAV platform were made, and some of the components to be used were ordered, namely the UAV frame with motors and ESCs, as well as batteries, two Beaglebone Black Wireless and two pixracer autopilots. One of the UAVs were partially assem-

bled intended for display in the project thesis report.

Bibliography

[1] Amazon, "Amazons proposed drone delivery system," https://www.amazon.com/Amazon-Prime-Air/b?node=8037720011, (Accessed on 20/11/2017).

[2] P. wing, "Googles project wing," https://x.company/wing/, (Accessed on 20/11/2017).

[3] Griff, "Griff aviation," http://griffaviation.com/the-griff-fleet/, (Accessed on 04/06/2018).

[4] D. Yeo, E. Shrestha, D. Paley, and E. Atkins, "An empirical model of rotorcraft uav downwash for disturbance localization and avoidance," in *Proc. AIAA Atmospheric Flight Mechanics Conference*, 2015.

[5] J. Pinto, P. S. Dias, R. Martins, J. Fortuna, E. Marques, and J. Sousa, "The lsts toolchain for networked vehicle systems," in *OCEANS-Bergen, 2013 MTS/IEEE*. IEEE, 2013, pp. 1–9.

[6] Y.-L. Chen, M. Peot, J. Lee, V. Sundareswaran, and T. Altshuler, "Autonomous collaborative mission systems (acms) for multi-uav missions," in *Defense Transformation and Network-Centric Systems*, vol. 5820. International Society for Optics and Photonics, 2005, pp. 152–160.

[7] Ardupilot, "Ardupilot multi-vehicle flying," http://ardupilot.org/copter/docs/common-multi-vehicle-flying.html, (Accessed on 01/06/2018).

[8] J.-H. B. Røli, "Cooperative control and rtk navigation system for multirotors," Master's thesis, NTNU, 2015.

[9] H. Bai, M. Arcak, and J. Wen, *Cooperative control design: a systematic, passivity-based approach*. Springer Science & Business Media, 2011.

[10] T. Lee, K. Sreenath, and V. Kumar, "Geometric control of cooperating multiple quadrotor uavs with a suspended payload," in *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*. IEEE, 2013, pp. 5510–5515.

[11] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The grasp multiple micro-uav testbed," *IEEE Robotics & Automation Magazine*, vol. 17, no. 3, pp. 56–65, 2010.

[12] L. R. G. Carrillo, A. E. D. López, R. Lozano, and C. Pégard, *Quad rotorcraft control: vision-based hovering and navigation*. Springer Science & Business Media, 2012.

[13] T. Ibuki, Y. Tadokoro, Y. Fujita, and M. Sampei, "3d inverted pendulum stabilization on a quadrotor via bilinear system approximations," in *Control Applications (CCA), 2015 IEEE Conference on*. IEEE, 2015, pp. 513–518.

[14] K. Klausen, T. I. Fossen, T. A. Johansen, and A. P. Aguiar, "Cooperative path-following for multirotor uavs with a suspended payload," in *Control Applications (CCA), 2015 IEEE Conference on*. IEEE, 2015, pp. 1354–1360.

[15] M. Weijers, "Minimum swing control of a uav with a cable suspended load," Master's thesis, University of Twente, 2015.

[16] R. Oung and R. D'Andrea, "The distributed flight array: Design, implementation, and analysis of a modular vertical take-off and landing vehicle," *The International Journal of Robotics Research*, vol. 33, no. 3, pp. 375–400, 2014.

[17] V. Parra-Vega, A. Sanchez, C. Izaguirre, O. Garcia, and F. Ruiz-Sanchez, "Toward aerial grasping and manipulation with multiple uavs," *Journal of Intelligent & Robotic Systems*, vol. 70, no. 1, pp. 575–593, Apr 2013. [Online]. Available: https://doi.org/10.1007/s10846-012-9743-0

[18] Q. Jiang and V. Kumar, "The inverse kinematics of cooperative transport with multiple aerial robots," *IEEE Transactions on Robotics*, vol. 29, no. 1, pp. 136–145, 2013.

[19] K. Sreenath and V. Kumar, "Dynamics control and planning for cooperative manipulation of payloads suspended by cables from multiple quadrotor robots," *rn*, vol. 1, no. r2, p. r3, 2013.

[20] M. Mohammadi, A. Franchi, D. Barcelli, and D. Prattichizzo, "Cooperative aerial tele-manipulation with haptic feedback," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*. IEEE, 2016, pp. 5092–5098.

[21] R. Ritz and R. D'Andrea, "Carrying a flexible payload with multiple flying vehicles," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE, 2013, pp. 3465–3471.

[22] T. I. Fossen, *Handbook of marine craft hydrodynamics and motion control*. John Wiley & Sons, 2011.

[23] H. Goldstein, *Classical mechanics*. Pearson Education India, 2011.

[24] F. E. Udwadia and R. E. Kalaba, "A new perspective on constrained motion," *Proceedings: Mathematical and Physical Sciences*, vol. 439, no. 1906, pp. 407–410, 1992. [Online]. Available: http://www.jstor.org/stable/52227

[25] ——, *Analytical dynamics: a new approach*. Cambridge University Press, 2007.

[26] M. Z. Nashed, *Generalized Inverses and Applications: Proceedings of an Advanced Seminar Sponsored by the Mathematics Research Center, the University of Wisconsin—Madison, October 8-10, 1973*. Elsevier, 2014, no. 32.

[27] SwiftNav, "Swiftnavs whitepaper pages," https://www.swiftnav.com/whitepaper, (Accessed on 28/05/2018).

[28] Ublox, "Ublox whitepaper pages," https://www.u-blox.com/en/white-papers, (Accessed on 27/05/2018).

[29] B. Hofmann-Wellenhof, H. Lichtenegger, and E. Wasle, *GNSS–global navigation satellite systems: GPS, GLONASS, Galileo, and more*. Springer Science & Business Media, 2007.

[30] GIS, "Gis trilateration vs triangulation," https://gisgeography.com/trilateration-triangulation-gps/, (Accessed on 01/06/2018).

[31] A. Rietdorf, C. Daub, and P. Loef, "Precise positioning in real-time using navigation satellites and telecommunication," in *Proceedings of The 3rd Workshop on Positioning and Communication (WPNC'06)*. Citeseer, 2006.

[32] mRobotics, "Official distributor of pixracer equipment," https://store.mrobotics.io/mRo-PixRacer-R14-Official-p/auav-pxrcr-r14-mr.htm, (Accessed on 1/11/2017).

[33] O. source project, "Ardupilot copter build," http://firmware.ardupilot.org/, (Accessed on 28/10/2017).

[34] Qgroundcontrol, "Mavlink qgroundcontrol homepage," http://qgroundcontrol.org/mavlink/start, (Accessed on 19/12/2017).

[35] B. Foundation, "Beaglebone black wireless," https://beagleboard.org/black-wireless, (Accessed on 23/10/2017).

[36] mRobotics, "Power distrubution board," https://store.mrobotics.io/product-p/auav-acsp4-mr.htm, (Accessed on 1/11/2017).

[37] https://brage.bibsys.no/xmlui/handle/11250/2448054, "Coordinated control of multirotors for suspended load transportation and fixed-wing net recovery," http://hdl.handle.net/11250/2448054, (Accessed on 04/05/2018).

[38] J. Carino, H. Abaunza, and P. Castillo, "Quadrotor quaternion control," in *Unmanned Aircraft Systems (ICUAS), 2015 International Conference on*. IEEE, 2015, pp. 825–831.

[39] T. I. Fossen, "Mss gnc tools," http://www.marinecontrol.org/, (Accessed on 30/05/2018).

[40] S. Låte, "Vertical stack simulation program," https://www.dropbox.com/s/sdj4gqeh3ibedn3/VerticalStackSimulation.zip?dl=0, (Accessed on 08/06/2018).

[41] Scipy, "Python library for numerical routines and optimization." https://scipy.org/scipylib/, (Accessed on 26/05/2018).

[42] S. Låte, "A vertical stack approach to cooperative drone lifting," https://www.dropbox.com/s/ivcpvxwgo6fc3gi/Project_thesis_stianlaa.pdf?dl=0, (Accessed on 27/05/2018).

[43] Y. Tadokoro, T. Ibuki, and M. Sampei, "Maneuverability analysis of a fully-actuated hexrotor uav considering tilt angles and arrangement of rotors," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 8981–8986, 2017.

[44] H. Mehmood, T. Nakamura, and E. N. Johnson, "A maneuverability analysis of a novel hexarotor uav concept," in *Unmanned Aircraft Systems (ICUAS), 2016 International Conference on*. IEEE, 2016, pp. 437–446.

[45] E. Kaufman, K. Caldwell, D. Lee, and T. Lee, "Design and development of a free-floating hexrotor uav for 6-dof maneuvers," in *Aerospace Conference, 2014 IEEE*. IEEE, 2014, pp. 1–10.

[46] M. Fujita, H. Kawai, and M. W. Spong, "Passivity-based dynamic visual feedback control for three-dimensional target tracking: Stability and l_{2}-gain performance analysis," *IEEE Transactions on Control Systems Technology*, vol. 15, no. 1, pp. 40–52, 2007.

[47] S. Nakano, T. Ibuki, and M. Sampei, "Visual feedback pose tracking control of two-wheeled vehicles with target vehicle motion models," in *Society of Instrument and Control Engineers of Japan (SICE), 2016 55th Annual Conference of the*. IEEE, 2016, pp. 1070–1075.

[48] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.