FORMATION FLIGHT OF TWO AUTONOMOUS BLIMPS

THE ATALANTA WINGMAN PROJECT

van de Loo, J.

Award date:
2007
Formation Flight of Two Autonomous Blimps
The Atalanta Wingman Project

Jasper van de Loo
DCT 2007.130

Master thesis

Coach: Prof. dr. H. Nijmeijer
ir. W. Hendriksen

Supervisor: Prof. dr. H. Nijmeijer

Committee: Prof. dr. H. Nijmeijer
Prof. dr. H. Corporaal
dr. ir. N. van de Wouw
ir. W. Hendriksen

Technische Universiteit Eindhoven
Department Mechanical Engineering
Dynamics and Control Group

DevLab Development Laboratories

Eindhoven, October, 2007
Summary

The Atalanta Wingman Project is part of DevLabs Atalanta Project, which focuses on the development of an Atalanta, a mechatronic butterfly at the same scale as a real butterfly. The Atalanta Wingman Project should find an answer to the question *What is needed to make a formation flight with two or more "Atalantas"?* This thesis finds an answer to this question from a mechanical point of view with the development of a control module which is able to make an autonomous formation flight with another module.

A non-rigid airship, also called a blimp, is selected as a demonstrator platform which carries the control module, because it has some major advantages with respect to robustness, safety, stability and energy usage in comparison with the other flying vehicles. A mathematical model of the blimp is made in order to study its behavior. The model can be used to perform simulations, design and test different control strategies and to evaluate the design of the blimps before building a real one.

A tracking controller is developed, because the blimp should be able to fly a predefined trajectory at a fixed height. The tracking controller consists of a nonlinear computed torque controller which linearizes the input-output behavior of the blimp and the resulting linear system is controlled using a PD controller. The computed torque controller is build from the inverse of the dynamic model. Because there may be parameter uncertainties, unmodeled dynamics and other uncertainties which may degrade the performance of the computed torque controller, a $H_{\infty}$ compensation method is added to the tracking controller which compensates for these uncertainties. Finally a velocity observer is constructed which estimates the velocity and position of the blimp, using only position information from the positioning system. Simulations show the effect of the different steps on the tracking performance.

A synchronization controller is developed in order to be able to make a formation flight with two blimps. Two synchronization strategies are discussed; master-slave synchronization, where the master blimp follows the desired trajectory while the slave blimp should follow the master and mutual synchronization, where essentially both blimps are the same and together try to stay in formation. Simulations show the mutual synchronization strategy, combined with a velocity observer, appears to be the best choice for a formation flight between two Atalanta’s.

The parameters of the dynamic model are estimated using a Continuous-Discrete Extended Kalman Filter (CDEKF), which used experimental data, gathered with a webcam, to estimate the parameters.

Finally different experimental schemes are described which can be used for future work to validate the theoretical results on a real blimp.
Samenvatting

Het Atalanta Wingman Project is onderdeel van Devlabs Atalanta Project, welke zich richt op de ontwikkeling van een Atalanta, een mechatronische vlinder met de afmetingen van een echte vlinder. Het Atalanta Wingman Project probeert een antwoord te vinden op de vraag: Wat is er nodig om een formatie vlucht met twee of meer Atalanta’s te maken? Dit rapport beantwoordt deze vraag vanuit een werktuigbouwkundig oogpunt met de ontwikkeling van een regelmodule die in staat is een autonome formatie vlucht te maken met een andere module.

Een blimp, een niet stijf luchtschip dat zijn vorm behoudt door overdruk van het gas waarmee het is gevuld, is gekozen als demonstratie platform om de regelmodule te dragen, omdat deze een aantal voordelen heeft wat betreft robuustheid, veiligheid, stabilité en energiegebruik in vergelijking met andere vliegende platformen. Een wiskundig model is gemaakt van de blimp zodat zijn gedrag bestudeerd kan worden. Het model kan gebruikt worden voor simulaties, voor de ontwikkeling van verschillende regelstrategieën en voor het evalueren van het ontwerp van de echte blimp.

Een *tracking* regelaar is ontwikkeld, waarmee de blimp een gewenst traject op een vaste hoogte kan volgen. De *tracking* regelaar bestaat uit een niet-lineaire *computed torque* regelwet die de relaties tussen de in- en uitgangen van de blimp lineariseert, door gebruik te maken van de inverse van het dynamische model. Het resulterende lineaire systeem wordt vervolgens geregeld met een PD regelaar. Een $H_{\infty}$ compensatie methode is vervolgens toegepast om te compenseren voor verstoringen als gevolg van onzekerheden in de parameters, niet en/of onjuist gemodelleerde dynamica en andere verstoringen die de prestatie van de *computed torque* regelaar verminderen. Tot slot is een waarnemer geconstrueerd die de snelheid en positie schat aan de hand van informatie van het positiemeetsysteem. Het effect van de verschillende stappen op de prestatie van de *tracking* regelaar is getoond door middel van simulaties.

Om een formatievlucht met twee blimps te kunnen maken, is een synchronisatiestrategie ontwikkeld. Twee synchronisatiestrategieën zijn beschreven; master-slave synchronisatie, waarbij de master blimp zijn gewenste traject vliegt terwijl de slave de master volgt en wederzijdse synchronisatie, waarin principe beide blimps gelijk zijn en samen proberen in formatie te blijven. Simulaties tonen aan dat de wederzijdse synchronisatie strategie, gecombineerd met een snelheidswaarnemer, de beste optie is als een formatievlucht met twee Atalanta’s gemaakt moet worden.

De parameters van het dynamische model zijn geschat met behulp van een *Continuous - Discrete Extended Kalman Filter* (CDEKF), dat gebruik maakt van experimentele data, verkregen met een webcam, om de parameters te schatten.

Tot slot is er een opzet gemaakt voor verschillende experimenten die in de toekomst gedaan kunnen worden om de theoretische resultaten te toetsen op een echte blimp.
# Contents

## Summary

### Samenvatting

1. **Introduction**

2. **Atalanta Wingman Project**
   - The "Atalanta"
   - Requirements
     - Requirements Blimp
     - Requirements Trajectory and Formation Flight
   - Atalanta Wingman Project Divided

3. **Blimp Model**
   - Kinematic Model
   - Dynamic Model
     - Mass and Inertia
     - Coriolis Effect
     - Gravity and Buoyancy Forces
     - Aerodynamic Damping
     - Propulsion
   - Full Model

4. **Tracking Control**
   - Simplified Dynamic Model
   - Computed Torque Method
   - Tracking Controller
     - Tracking Simulation
   - Computed Torque Plus $H_{\infty}$ Compensation Method
     - Formulation of Uncertainty
     - Constructing an Augmented plant with Internal Weightings
     - Design Procedures
     - Simulation Robust Computed Torque Controller
   - Position Measurement

---

Summary: iii
Samenvatting: v

1. Introduction: 1

2. Atalanta Wingman Project: 5
   2.1 The "Atalanta": 6
   2.2 Requirements: 8
     2.2.1 Requirements Blimp: 8
     2.2.2 Requirements Trajectory and Formation Flight: 9
   2.3 Atalanta Wingman Project Divided: 10

3. Blimp Model: 13
   3.1 Kinematic Model: 13
   3.2 Dynamic Model: 15
     3.2.1 Mass and Inertia: 16
     3.2.2 Coriolis Effect: 18
     3.2.3 Gravity and Buoyancy Forces: 18
     3.2.4 Aerodynamic Damping: 19
     3.2.5 Propulsion: 20
   3.3 Full Model: 21

4. Tracking Control: 23
   4.1 Simplified Dynamic Model: 24
   4.2 Computed Torque Method: 25
   4.3 Tracking Controller: 26
     4.3.1 Tracking Simulation: 26
   4.4 Computed Torque Plus $H_{\infty}$ Compensation Method: 28
     4.4.1 Formulation of Uncertainty: 29
     4.4.2 Constructing an Augmented plant with Internal Weightings: 30
     4.4.3 Design Procedures: 35
     4.4.4 Simulation Robust Computed Torque Controller: 36
   4.5 Position Measurement: 39
### CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.5.1</td>
<td>Velocity Observer</td>
<td>39</td>
</tr>
<tr>
<td>4.5.2</td>
<td>Simulation Velocity Observer</td>
<td>40</td>
</tr>
<tr>
<td>4.6</td>
<td>A Worst Case Scenario Tracking Simulation</td>
<td>40</td>
</tr>
<tr>
<td>5</td>
<td>Formation Flight: Synchronization</td>
<td>45</td>
</tr>
<tr>
<td>5.1</td>
<td>Synchronization Controller</td>
<td>46</td>
</tr>
<tr>
<td>5.2</td>
<td>Master-Slave Synchronization</td>
<td>47</td>
</tr>
<tr>
<td>5.2.1</td>
<td>Master-Slave Synchronization Simulation</td>
<td>48</td>
</tr>
<tr>
<td>5.3</td>
<td>Mutual Synchronization</td>
<td>50</td>
</tr>
<tr>
<td>5.3.1</td>
<td>Mutual Synchronization Simulation</td>
<td>51</td>
</tr>
<tr>
<td>5.4</td>
<td>Synchronization with uncertainties</td>
<td>51</td>
</tr>
<tr>
<td>5.4.1</td>
<td>Synchronization Controller Based On Estimated Variables</td>
<td>53</td>
</tr>
<tr>
<td>5.4.2</td>
<td>Comparison Master-Slave vs. Mutual Synchronization</td>
<td>53</td>
</tr>
<tr>
<td>5.5</td>
<td>A Worst Case Scenario Synchronization Simulation</td>
<td>55</td>
</tr>
<tr>
<td>6</td>
<td>Parameter Identification</td>
<td>59</td>
</tr>
<tr>
<td>6.1</td>
<td>Continuous-Discrete Extended Kalman Filter</td>
<td>59</td>
</tr>
<tr>
<td>6.1.1</td>
<td>Continuous System- and Discontinuous Measurement model</td>
<td>60</td>
</tr>
<tr>
<td>6.1.2</td>
<td>Propagation Between Measurements</td>
<td>60</td>
</tr>
<tr>
<td>6.1.3</td>
<td>Measurement Update</td>
<td>61</td>
</tr>
<tr>
<td>6.2</td>
<td>Implementation of the CDEKF</td>
<td>61</td>
</tr>
<tr>
<td>6.2.1</td>
<td>Identification subsystem I</td>
<td>62</td>
</tr>
<tr>
<td>6.2.2</td>
<td>Identification subsystem II</td>
<td>63</td>
</tr>
<tr>
<td>6.3</td>
<td>Measurements for the CDEKF</td>
<td>63</td>
</tr>
<tr>
<td>6.3.1</td>
<td>Measurements for subsystem I</td>
<td>64</td>
</tr>
<tr>
<td>6.3.2</td>
<td>Measurements for subsystem II</td>
<td>64</td>
</tr>
<tr>
<td>6.4</td>
<td>Settings for the CDEKF</td>
<td>67</td>
</tr>
<tr>
<td>6.5</td>
<td>Identification Results</td>
<td>68</td>
</tr>
<tr>
<td>6.6</td>
<td>Discussion</td>
<td>71</td>
</tr>
<tr>
<td>7</td>
<td>Experimental Scheme</td>
<td>73</td>
</tr>
<tr>
<td>7.1</td>
<td>Hardware</td>
<td>74</td>
</tr>
<tr>
<td>7.2</td>
<td>Parameter Identification</td>
<td>75</td>
</tr>
<tr>
<td>7.2.1</td>
<td>Experiments for Parameter Identification</td>
<td>76</td>
</tr>
<tr>
<td>7.2.2</td>
<td>Data Processing</td>
<td>77</td>
</tr>
<tr>
<td>7.3</td>
<td>Tracking Experiment</td>
<td>77</td>
</tr>
<tr>
<td>7.3.1</td>
<td>Controller Settings</td>
<td>78</td>
</tr>
<tr>
<td>7.3.2</td>
<td>Comparing the Tracking Simulations with the Experiments</td>
<td>78</td>
</tr>
<tr>
<td>7.4</td>
<td>Synchronization Experiment</td>
<td>78</td>
</tr>
<tr>
<td>7.4.1</td>
<td>Master-Slave Synchronization Experiment</td>
<td>79</td>
</tr>
<tr>
<td>7.4.2</td>
<td>Mutual Synchronization Experiment</td>
<td>79</td>
</tr>
<tr>
<td>8</td>
<td>Conclusions and Recommendations</td>
<td>81</td>
</tr>
<tr>
<td>8.1</td>
<td>Future Work</td>
<td>83</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

DevLab, a collaboration initiative between twelve technological Small and Medium Enterprises, carries out applied research in the area of embedded systems. They work together with universities and other higher education institutes on two major research projects, which both study the development of new high-tech products. These products require multiple first-class technologies, stimulate the imagination with many new opportunities and eventually can be used widespread.

One of the long term research projects is called the "Atalanta Project", which studies the development of an "Atalanta", an autonomous mechatronic butterfly at the same scale as a real butterfly. This very complex product requires many new technologies and a lot of research, because it can not be build at this moment. The final Atalanta should be a device that weights only 5 grams and has a maximum wing span of 10 cm for its flapping wings. Also it should be able to navigate and communicate with other Atalanta’s.

The Eindhoven University of Technology (TU/e), with the departments of Mechanical Engineering and Electrical Engineering, is participating in the Atalanta Project. Both departments corporate together on the different aspects of the butterfly. One of these aspects is formation flight between a group of "Atalantas". This research project, called the Atalanta Wingman Project, will focus on that aspect with the following main question: What is needed to make a formation flight with two or more "Atalantas"? To answer this question a theoretical study will be carried out and a demonstrator will be build in order to prove the different aspects that are involved with this research project. The main objective hereby is to design and build a control module which is able to control an "Atalanta" in such a way that it can make an autonomous formation flight with another "Atalanta". This control module should be able to determine its position in 3D-space, fly a predefined trajectory, make a formation flight with another module, communicate with other modules and deal with small disturbances. The different requirements will be given a closer look in Chapter 2.2.

The Atalanta Wingman Project is carried out by the author of the thesis in cooperation with a MSc student from the department of Electrical Engineering. There-
fore this thesis only covers the aspects related to Mechanical Engineering. The aspects related to Electrical Engineering, like the electrical hardware and software for the control module and the positioning system, come from the department of Electrical Engineering and are assumed to be known or available for this project. In Chapter 2.3 a more extensive comparison is made between the two sub-projects in relation to the full project.

This thesis should contribute to the Atalanta Wingman Project by fulfilling the following main goals. First a suitable choice has to be made for a demonstrator platform which is able to carry the control module, because different flying platforms appear to be suitable. Secondly a mathematical model of this demonstrator platform is needed in order to be able to study its behavior and to carry out simulations. The model parameters should be estimated, so a good representation of the reality is available for the platform. According to the specifications the control module should be able to follow a certain desired trajectory, while it also has to deal with small disturbances. A tracking controller has to be designed which fulfills those requirements. The final goal of this thesis is to develop a synchronization strategy, which is suitable to perform a formation flight with two Atalanta’s.

A non-rigid airship, also called a blimp, is used as a demonstrator platform which carries the control module. Also other options have been investigated, like a fixed wing aircraft, a conventional helicopter and a quadrotor helicopter, but the blimp appears to be the most suitable platform. Although at first sight there seems to be a big difference between a large blimp and a very small butterfly, there are many similarities. These similarities are described, together with a comparison between the blimp and the other flying platforms, in Chapter 2.1.

The first step in the development of the control module is the development of a mathematical model, which describes the dynamics of the blimp. This mathematical model is made in order to study the behavior of the blimp. It can be used to perform simulations, design and test different control strategies and to evaluate the design of the blimps before building a real one. The mathematical model is described in Chapter 3.

In Chapter 4 the mathematical model is used to design a tracking controller for the blimp. This tracking controller is needed to be able to follow a predefined trajectory with the blimp. The first part of the tracking controller consists of a nonlinear computed torque controller, based on the dynamic model of the blimp, which linearizes the input-output relations of the system. In the second part of the tracking controller, the resulting linear system is further controller with a linear PD-controller. Because the tracking controller is based on the dynamic model of the blimp, model uncertainties, like unmodeled dynamics and parameter uncertainties will degrade the performance of the tracking controller. Therefore the tracking controller is extended with a $H_{\infty}$ compensation method, which compensates for these uncertainties. Finally the tracking controller is completed with the implementation of a velocity observer, which finds an estimate of the position and velocity, based on the position information from the positioning system. This observer is used, because only the position and orientation of the blimp is measured,
while also velocity information is needed. Furthermore the velocity observer reduces measurement noise in the position information from the positioning system. The final tracking controller is verified by means of simulations, which also show the effect of the different steps.

Formation flight, or synchronization between two blimps, is discussed in Chapter 5. Two different synchronization strategies are studied, master-slave synchronization and mutual synchronization. With master-slave synchronization the master blimp has to follow the predefined trajectory without paying any attention to the slave, while the slave has to follow the master, without knowing its own desired trajectory. With mutual synchronization essentially both blimps are equal, so they both know the desired trajectory and together they try to stay in formation by exchanging their position and velocity information. Because both synchronization controllers need information about the position and velocity of the blimp, the velocity observer is also implemented here. Finally the preferred synchronization strategy is chosen by comparing their simulation results.

The dynamic model of the blimp contains many physical parameters. It is important to know these parameters, because they are needed to do simulations with the dynamic model. Even more important is that some of the parameters are required for the tracking controller and also the synchronization controllers make use of them. Most of these parameter can not be measured directly, however it is possible to estimation them. In Chapter 6 a Continuous-Discrete Extended Kalman Filter (CDEKF) is implemented which uses experimental data, gathered using a webcam, to estimate the parameters.

Finally in Chapter 7 different recommended experimental schemes are described, which can be carried out for future work. First an experiment is discussed which may improve the parameter estimation. By using the positioning system of the blimps, instead of a webcam, longer runs can be carried out, which contain more information for the CDEKF. Secondly a tracking experiment is described which can be used to evaluate the tracking controller and the experimental results can be compared with the results from the tracking simulation. The last experimental scheme which is described is a synchronization experiment, where both blimps make a formation flight.

Finally in Chapter 8 conclusions are drawn with respect to the main questions and some recommendations are made for future work.
Chapter 2

Atalanta Wingman Project

The Atalanta Wingman Project is part of the long term project called The Atalanta Project, which aims for the development of an autonomous mechatronic butterfly at the same scale as a real butterfly. This major DevLab project is a combination of disciplines that are very rich of technological aspects, so many sub-projects can be generated. Some examples are; (wireless) communication, energy usage, sensing, micro mechanical drivetrains, network architectures and protocols, polymers, positioning, et cetera.

The goal of the Atalanta Wingman project is to investigate: What is needed to make a formation flight with two or more "Atalantas"? In order to answer this question many of the mentioned aspects have to be treated, like:

- Is there a need for communication?
- Is a positioning system required, for all, or only for some of them?
- How can the energy usage be limited?
- What kind of sensing do they need?
- What kind of synchronization strategy is suitable?

Because the main question of the Atalanta Wingman Project is very broad, a more specified goal is set. The main objective hereby is to design and build a control module that is able to make an autonomous formation flight with two "Atalantas". This control module should be able to determine its position in 3D-space, fly a predefined trajectory, make a formation flight with another module, communicate with other modules and deal with small disturbances. The design of this control module will give answers for most of the question which arise from the main question.

This chapter describes the Atalanta Wingman Project in more detail. First in Section 2.1 a suitable choice is made for the demonstrator platform, which carries the control module. Section 2.2 describes the exact specifications, for both the control module as for the desired trajectory which has to be followed.
The Atalanta Wingman Project contains aspects of both Mechanical Engineering and Electrical Engineering. Section 2.3 describes which parts are carried out by the author of this thesis and which parts are carried out in a companion project done by a MSc student from Electrical Engineering.

2.1 The "Atalanta"

A flying platform is needed to carry the control module. A flapping wing aircraft appears to be most obvious to use, because the final Atalanta will also be a flapping wing device. However at this moment flapping wing aircrafts have some disadvantages which make it unsuitable to use as platform. Flapping wing aircrafts are relatively new and although there are some examples that perform quite well (for example [1]), a lot of research is needed to refine their design. Because the Atalanta Wingman Project focuses on the design of a control module which is capable of performing an autonomous formation flight and not on the design of a flying platform, it is decided not to use a flapping wing aircraft for the Atalanta Wingman Project.

Another flying platform has to be chosen which can carry the control module. Different flying platforms have been considered, e.g. a fixed wing aircraft, a conventional helicopter, a quadrotor and a blimp. A conventional fixed wing aircraft can be very small and also it can be controlled easily, however it can not hover and fly in a confined area, making it unsuitable, because the Atalanta should be able to fly indoors in a relatively small room. A conventional helicopter can fly indoors and carry a high payload, but it is an unstable system, making it difficult to control. Also the exposed rotor blades can be extremely dangerous in practice. A quadrotor helicopter, which is depicted in Figure 2.1 does have the same advantages, but can more safely be used indoors. However the other disadvantages of the helicopter also hold for the quadrotor, it has a high onboard power requirement and it is an unstable system because of its low damping rate [2].

Eventually the blimp is chosen to be used as demonstrator platform. A blimp, depicted in Figure 2.2 is a non-rigid airship, which differs from a rigid airship (e.g. a Zeppelin) in that it does not have a rigid structure that holds the airbag in shape. Rather, blimps rely on a higher pressure of the gas (usually helium) inside the envelope. A blimp is chosen as platform because it has some major advantages with respect to robustness, safety, stability, energy usage and complexity in comparison with the other flying vehicles. It can be used safely indoors and it can crash without having a high chance of causing damage to itself or its surrounding. Also a blimp does not require power to stay in the air, because it is a buoyant vehicle which stays in the air because it is lighter than the air it displaces. This also makes it possible to hover with the blimp. Finally a slow moving blimp can be controlled relatively easy. A full list of advantages and disadvantages of the different platforms can be

---

1 Another DevLab project focuses on the development of a flapping wing aircraft, in the development of an autonomous mechatronic butterfly.
At first sight there seems to be a big difference between a blimp and a flapping wing device like a butterfly, but there are quite a lot of similarities. The small scale blimp that is used for this project can, like a flapping wing aircraft, only lift a very small amount of payload, in order to keep its size relatively small. The blimp is still quite large, because one liter of helium can only lift approximately one gram, of which the balloon weight has to be extracted in order to find the netto lift available for payload. Therefore the used blimp can only lift approximately 120 grams of payload while it is a 1.6 m long blimp with a maximum diameter of 0.8 m, carrying about 400 liters of helium. Another similarity between a blimp and a flapping wing aircraft is their low flight speed, which is for the used blimp about...
2.2 Requirements

There are many requirements set for this project. A summary of these requirements is already given in the introduction of this chapter. In this section a closer look will be given to the different requirements of this project. The requirement can be divided in requirements for the blimp, c.q. control module and requirements for the trajectory and formation flight.

### 2.2.1 Requirements Blimp

The most important requirement for the blimp is that is has to fly autonomously without external control. So all hardware and software required for flight has to be onboard. Another important requirement is that the blimp should be able to determine its absolute position in 3D space with a precision of 0.1 m. Also it has

<table>
<thead>
<tr>
<th>Platform</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blimp</td>
<td>- flies in a confined area</td>
<td>- slow reacting</td>
</tr>
<tr>
<td></td>
<td>- statically stable</td>
<td>- not possible to control all degrees of freedom</td>
</tr>
<tr>
<td></td>
<td>- lowest onboard power requirements</td>
<td>- they are big</td>
</tr>
<tr>
<td></td>
<td>- simple linear control systems</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- can be used indoors</td>
<td></td>
</tr>
<tr>
<td>Fixed wing aircraft</td>
<td>- statically stable</td>
<td>- can not fly in a confined area</td>
</tr>
<tr>
<td></td>
<td>- simple linear control</td>
<td>- can not hover</td>
</tr>
<tr>
<td></td>
<td>- moderate power requirements</td>
<td></td>
</tr>
<tr>
<td>Conventional</td>
<td>- flies in a confined area</td>
<td>- extremely dangerous in practice due to the exposed rotor blades</td>
</tr>
<tr>
<td>helicopter</td>
<td>- high payload to power ratio</td>
<td>- statically unstable</td>
</tr>
<tr>
<td></td>
<td>(gas powered helicopters)</td>
<td>- high onboard power requirements</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- non-linear control system</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- complex mechanics</td>
</tr>
<tr>
<td>Quadrotor helicopter</td>
<td>- flies in a confined area</td>
<td>- high onboard power requirements</td>
</tr>
<tr>
<td></td>
<td>- can be used indoor</td>
<td>- because of low damping rate, electronic stability augmentation is required</td>
</tr>
<tr>
<td></td>
<td>- can have simple control mechanisms [10]</td>
<td>for stable flight</td>
</tr>
<tr>
<td></td>
<td>- it is controlled by only changing the speed of rotation for the four motors</td>
<td>- the dynamics of the quadrotor can make the vehicle difficult to control [2]</td>
</tr>
<tr>
<td></td>
<td>- highly maneuverable</td>
<td></td>
</tr>
</tbody>
</table>

Table 2.1: Advantages and disadvantages of four different platforms

1 m/s. Finally the power requires are low, for the flapping wing aircraft because of its small scale and for the blimp because it is a buoyant vehicle, which does not require power to stay in the air.
to measure its heading or yaw angle. The other two angles, the roll and pitch, are not required for operation, so they do not have to be measured.

Another requirement is that it has to be able to control the measured degrees of freedom. This also implies the blimp should be able to deal with small disturbances, such as airflow. It is assumed that the DevLab office, where the experiments will be carried out, contains only small disturbances.

The blimp has to have an onboard energy source, which is large enough to provide power for at least ten minutes of flight, without recharging. And finally the blimp shall be able to communicate wireless with a PC for debugging and demonstration purposes.

2.2.2 Requirements Trajectory and Formation Flight

At least one blimp (the master blimp) should be able to fly a predefined desired trajectory. This trajectory is a rectangular figure of $3 \times 5$ m at a predefined constant altitude. The blimp shall not deviate more than 0.5 m from this preprogrammed route and the deviation in height has to be less than 0.1 m. During the flight of this figure the blimp has to make a full rotation about its yaw-axis. Finally one lap has to be completed within two minutes and the trajectory will not contain any obstacles at flight level. A schematic representation of the desired trajectory is depicted in Figure 2.3.

![Figure 2.3: Desired trajectory with tolerance.](image)

The blimps are able to fly in formation with each other. One blimp, the master, will fly the desired trajectory as discussed in Section 2.2.2. The other blimp, the slave, will fly along side the master blimp with a constant predefined distance between both blimps and their headings are kept equal. Each blimp is controlled by its own control module during formation flight and formation flight will only start if both blimps are within the preprogrammed range/boundaries of the formation, at a predefined point.
2.3 Atalanta Wingman Project Divided

The Atalanta Wingman Project is a multidisciplinary project where aspects related to both Mechanical Engineering and Electrical Engineering are present. For this reason the Atalanta Wingman Project is carried out by the author of this thesis in cooperation with a MSc student from the department of Electrical Engineering, Rolf van de Burgt. This thesis only describes the aspects related to Mechanical Engineering and it assumes that the aspects related to Electrical Engineering are known or available.

This section gives a short summary how the different components of the Atalanta Wingman Project, of which many are related with each other, are divided between the two disciplines.

This thesis starts in Chapter 3 with the design of a mathematical model, which describes the dynamics of the blimp. Continuing in Chapter 4 with the design of a tracking controller based on the dynamic model. Simulations show the tracking controller works well in controlling the simulated blimp, however in order to do the same experiment in real world, an actual blimp is needed. This blimp, together with a gondola containing the four thrusters is bought. Normally this blimp is controlled using manual remote control, but for this project it has to be made autonomous. The electrical hardware required to realize this consists of many components, like the electric motors with their motor controllers, the power supply, a microcontroller, ultrasonic receivers and a compass for navigation. The design and construction of the electrical hardware for the gondola is done by the MSc student from the department of Electrical Engineering. For this thesis it is assumed that the gondola can be build according to its specifications.

The blimp requires a positioning system to determine its absolute position and orientation, so it is possible to perform a tracking maneuver with the tracking controller or to do a formation flight using the synchronization controller. The positioning system is designed and build by the department of Electrical Engineering and it makes use of ultrasonic beacons and a compass. This thesis assumes the position and orientation can be measured according to the specifications, although some measures have been taken to deal with larger disturbances.

Although it appears that the aspects of both disciplines are separated, there is a lot of overlap. For example the tracking controller requires position and velocity information, gathered by the positioning system. However the positioning system only measures position and there is always a certain amount of measurement noise present, which is not known in advance. In order to deal with this, a velocity observer (see Chapter 4.5.1) is implemented, which estimates the velocity from only the position information. The other way around, the positioning system has to work with a certain sample rate. The minimal sample rate required to satisfy the accuracy requirements is found by performing tracking simulations using different sample times.

Finally the blimp has an onboard ARM 7 microcontroller. To be able to perform a tracking or synchronization experiment with the real blimp, software is needed for
this microcontroller. Unfortunately it is not possible to load the Matlab Simulink controllers, used for the simulations, directly into the microcontroller. Therefore the controllers have to be rewritten in the programming language C, which is done by the Electrical Engineering department.
Chapter 3

Blimp Model

A mathematical model of a blimp is made in order to study its behavior. This model can be used to perform simulations, design and test different control strategies and to evaluate the design of the blimps before building a real one.

First a kinematic model is made in Section 3.1 which describes the relation between the velocities of the blimp with respect to its body fixed reference frame and the earth fixed inertial reference frame. In Section 3.2 the dynamic equations of motion are discussed, which describe the full nonlinear six degree of freedom motion of the blimp.

Finally the complete mathematical model, by combining the kinematic model with the dynamic mode, is given in Section 3.3.

3.1 Kinematic Model

Two reference frames are considered in the derivation of the kinematics and dynamical equations of motion. One earth fixed inertial frame \( F_0 \) and a body fixed frame \( F_B \). Both reference frames form a right handed orthogonal frame.

The origin of \( F_B \) coincides with the center of gravity \( C_g \) of the blimp, as can be seen in Figure 3.1. The center of buoyancy \( C_b \), which coincides with the center of volume \( C_v \), is also often used as the origin of the body fixed frame, but this leads to a more complicated dynamic model together with a controller that is more complex. Despite of a higher complexity of the model, the center of buoyancy is used when modeling a large size blimp, which can fly at high altitudes. In that case the total mass of the airship can change considerably in a very short time, due to deflation or inflation of the ballonet during a climb or descent maneuver respectively.

Because of the constantly changing mass of the airship, the position of the center of gravity is also constantly changing, which makes it unsuitable as the origin of a body fixed frame. An example of a dynamical model of an airship which uses the center of buoyancy as the origin of the body fixed frame can be found in for example [9], [17] and [24]. Because of its small size and small altitude variation of the blimp which is described here, the position of the center of gravity \( C_g \) will not change.
considerably and can therefore be used as the origin of the body fixed frame $F_B$.

The relation between the orientation of the airship frame $F_B$ and the inertial reference frame $F_0$ can be expressed with the direction cosine matrix $R^{0B}$ [21]:

$$F_0 = R^{0B} F_B.$$  \hspace{1cm} (3.1)

The direction cosine matrix obeys

$$R^{0B} R^{0B T} = R^{0B} R^{B0} = I_{3 \times 3} \text{ and } \det(R^{0B}) = 1,$$  \hspace{1cm} (3.2)

where $I_{3 \times 3}$ represents the $3 \times 3$ identity matrix. Tait-Bryant angles have been used for a parametric representation of the direction cosine matrix, because this formulation is often used in flight dynamics [21]. The direction cosine matrix is given by:

$$R^{0B} = \begin{pmatrix}
c\psi c\theta & s\psi c\phi + c\psi s\theta s\phi & s\psi s\phi - c\psi s\theta c\phi \\
-s\psi c\theta & c\psi c\phi - s\psi s\theta s\phi & c\psi s\phi + s\psi s\theta c\phi \\
s\theta & -c\theta s\phi & c\theta c\phi
\end{pmatrix},$$  \hspace{1cm} (3.3)

where $s\phi = \sin(\phi)$ and $c\phi = \cos(\phi)$. This description is valid in the region $-\frac{\pi}{2} < \theta < \frac{\pi}{2}$. A singularity of this transformation exists for $\theta = \frac{\pi}{2} \pm k\pi; k \in \mathbb{Z}$. 

Figure 3.1: Inertial reference frame $F_0$ and body fixed frame $F_B$
Two reference frames are needed, because the translational and rotational velocities of the blimp

\[ \mathbf{v} = (v_x, v_y, v_z, \omega_x, \omega_y, \omega_z)^T \]  

are described in the body fixed reference frame \( \mathcal{F}_B \), whereas the position and orientation of the airship

\[ \mathbf{q} = (x, y, z, \phi, \theta, \psi)^T \]  

are expressed in the earth inertial reference frame \( \mathcal{F}_0 \). The angles \( \phi \), \( \theta \) and \( \psi \) represent the roll, pitch and yaw rotation of the blimp respectively.

The relations between the derivative of \( \mathbf{q} \), \( \dot{\mathbf{q}} \) and \( \mathbf{v} \) is described by the kinematic model of the blimp, which can be expressed as \[22\]:

\[ \dot{\mathbf{q}} = \begin{pmatrix} R_{0B}^{0B} & 0_{3 \times 3} \\ 0_{3 \times 3} & \mathbf{J} \end{pmatrix} \mathbf{v}, \] 

where

\[ \mathbf{J} = \begin{pmatrix} 1 & s\phi\tan\theta & c\phi\tan\theta \\ 0 & c\phi & -s\phi \\ 0 & s\phi/c\theta & c\phi/c\theta \end{pmatrix}. \] 

In the following section the dynamic equations of motion are discussed, which describe the full nonlinear six degree of freedom motion of the blimp.

### 3.2 Dynamic Model

The dynamic model describes the full nonlinear six degrees of freedom motion of the blimp. It gives the relation between the forces and moments acting on the blimp and the acceleration of the blimp in six degrees of freedom. It describes for example the effect of the thrust produces by the rotors on the movement of the blimp. A few assumptions have been made in the design of a full nonlinear mathematical model of the blimp, in order to reduce the complexity of the model:

- The blimp is assumed to form a rigid body, such that aerelastic effects can be ignored.
- The blimp is symmetric about the \( v_x - v_z \) plane, such that both the center of volume \( C_v \) and the center of gravity \( C_g \) lie in the plane of symmetry.
- The center of gravity \( C_g \) lies below the center of buoyancy \( C_b \), so that the blimp is stabilized about the roll and pitch axes. This assumption is valid because the gondola, which is attached under the hull, contains the majority of the equipment for actuation, sensing and power.

By taking the above assumptions into account, the dynamic model can be stated as:

\[ \mathbf{M}\ddot{\mathbf{v}} = \mathbf{F}_c + \mathbf{F}_g + \mathbf{F}_a + \mathbf{F}_p, \] 

where the five components are listed below:
3.2 Dynamic Model

\( M \) mass and inertia matrix (described in Section 3.2.1) with added mass effects;

\( F_c \) force vector as a result of the Coriolis effect (Section 3.2.2), containing fictitious forces appearing in non-inertial frames such as \( F_B \);

\( F_g \) includes gravitational and buoyancy induced forces and moments (Section 3.2.3);

\( F_a \) includes aerodynamic forces and moments (Section 3.2.4) arising from the flow of air around the hull of the blimp;

\( F_p \) vector containing the propulsion forces and moments (Section 3.2.5) generated by propeller thrust.

In the following sections the different components of (3.8) will be given a closer look.

3.2.1 Mass and Inertia

The mass and inertia matrix \( M \) is build of the rigid-body inertia matrix, which can be written as

\[
M_{RB} = \text{diag}(m, m, I_x, I_y, I_z),
\]

(3.9)

where \( m \) is the total mass of the blimp and \( I_x, I_y, I_z \) are the different inertia terms about the center of gravity \( C_g \).

Only rigid body terms in the mass and inertia matrix however is not sufficient for a buoyant vehicle like this blimp, because when moving, the large body of the blimp displaces a large amount of air particles, which generate resistance. This effect is important when dealing with vehicles which have a similar density as their surrounding fluid, like an airship or submarine. This additional effect is modeled by including added mass (sometimes also called "virtual mass") and inertia terms to the inertia matrix. A simple model of the added-mass effect, according to [7] and [24], is given below:

\[
M_A = \text{diag}(m_A, m_A, m_A, I_A, I_A, I_A),
\]

(3.10)

The off-diagonal elements can be neglected, because the blimp moves at very low speeds and it is assumed that the ellipsoidal hull has three planes of symmetry. The diagonal elements can not be neglected, because they are responsible for an intrinsic instability of airships, as will be explained in Section 3.2.2

The total mass and inertia matrix including the rigid-body and added-mass effects can now be calculated by combining (3.9) and (3.10):

\[
M = M_{RB} + M_A = \begin{pmatrix}
m'_x & 0 & 0 & 0 & 0 & 0 \\
0 & m'_y & 0 & 0 & 0 & 0 \\
0 & 0 & m'_z & 0 & 0 & 0 \\
0 & 0 & 0 & I'_x & 0 & 0 \\
0 & 0 & 0 & 0 & I'_y & 0 \\
0 & 0 & 0 & 0 & 0 & I'_z \\
\end{pmatrix},
\]

(3.11)
where \( m'_x = m + m_{A_{x}} \), \( m'_y = m + m_{A_{y}} \), \( m'_z = m + m_{A_{z}} \), \( I'_x = I_x + I_{A_{x}} \), \( I'_y = I_y + I_{A_{y}} \), and \( I'_z = I_z + I_{A_{z}} \) are respectively the apparent masses and inertias.

The shape of the envelope, an ellipse, already suggests \( m_{A_{x}} \approx m_{A_{y}} \approx 0 \) and \( I_{A_{y}} = I_{A_{z}} \). An estimation of the added mass effect can be made using a geometrical method based on the kinetic energy of an ideal unbounded liquid around the ellipsoid in motion [24]. The kinetic energy and the force necessary to accelerate the blimp can be computed by adding to the actual mass of the solid a fictitious mass. This added-mass is equal to the density of the fluid multiplied by a volume depending on the geometry of the airship only. The Lamb’s \( k \)-factors [11] are a result of this consideration, where \( k_1 \) and \( k_2 \) are the inertia coefficients representing the fraction of the mass displaced by the hull and \( k' \) is the ratio of apparent moment of inertia to the moment of inertia of the displaced air \( I_{zh} \).

Lamb’s \( k \) factors are used as follows to calculate the added mass terms

\[
\begin{align*}
    m_{A_{x}} &= k_1 m, \\
    m_{A_{y}} &= m_{A_{z}} = k_2 m, \\
    I_{A_{x}} &= 0, \\
    I_{A_{y}} &= I_{A_{z}} = k'I_{zh},
\end{align*}
\]

where the inertia of the displaced air \( I_{zh} \) can be calculated in case of an ellipsoid with semi-axes \( a \) and \( b \) (see Figure 3.2) as

\[
I_{zh} = \frac{4}{15} \pi \rho ab^2 (a^2 + b^2).
\]

The Lamb’s \( k \)-factors can be defined according to [11], [24], as

\[
\begin{align*}
    k_1 &= \frac{\alpha}{2 - \alpha}, \\
    k_2 &= \frac{\beta}{2 - \beta}, \\
    k' &= \frac{e^4(\beta - \alpha)}{(2 - e^2)(2e^2 - (2 - e^2)(\beta - \alpha))},
\end{align*}
\]

where the two constants \( \alpha \) and \( \beta \) are given as:

\[
\begin{align*}
    \alpha &= \frac{2(1 - e^2)}{e^3} \left( \frac{1}{2} \ln \left( \frac{1 + e}{1 - e} \right) - e \right), \\
    \beta &= \frac{1}{e^2} - \frac{1 - e^2}{2e^3} \ln \left( \frac{1 + e}{1 - e} \right),
\end{align*}
\]

and \( e \) denotes the eccentricity of the ellipsoid:

\[
e = \sqrt{1 - \left( \frac{b}{a} \right)^2}.
\]
3.2.2 Coriolis Effect

The Coriolis effects give a fictitious force exerted on a body in motion when the referential frame is not inertial, which is the case for $F_B$. The Coriolis force occurs when a motion is composed of linear and rotational velocities and can be expressed as $\omega \times v$. It results in a force perpendicular to the linear velocity vector and rotation axis. The Coriolis force tends to maintain the initial direction of motion without taking care of the body rotation.

If the center of buoyancy $C_b$ was chosen as the origin of the body fixed reference frame $F_B$, instead of the center of gravity $C_g$, there would be another fictitious force acting on the body, namely the centripetal force which can be expressed as $\omega \times (\omega \times r)$. Because in the presented model the origin of the body fixed frame $F_B$ lies in the center of gravity $C_g$, the centripetal effect will be zero.

The Coriolis force can be expressed as

$$F_c = C(v)v,$$  \hspace{1cm} (3.23)

where $C(v)$ is the so-called Coriolis matrix. The Coriolis matrix can be derived directly from the inertia matrix (see [18]):

$$C(v) = \begin{pmatrix} 0_{3\times 3} & S(M_{11}v + M_{12}\omega) \\ S(M_{11}v + M_{12}\omega) & S(M_{21}v + M_{22}\omega) \end{pmatrix},$$ \hspace{1cm} (3.24)

where $0_{3\times 3}$ is the null matrix, $S$ is the skew-symmetric matrix operator and $M_{ij}$ ($i, j = 1, 2$) are the four $3 \times 3$ sub matrices of the global inertia matrix $M$ \(\text{(3.11)}\).

The full expression for $C(v)$ is given below:

$$C(v) = \begin{pmatrix} 0 & 0 & 0 & 0 & -m'_z v_z & m'_y v_y \\ 0 & 0 & 0 & m'_z v_z & 0 & -m'_x v_x \\ 0 & 0 & 0 & 0 & -m'_y v_y & m'_x v_x \\ -m'_y v_y & m'_x v_x & 0 & -I'_z \omega_z & I'_y \omega_y \\ m'_z v_z & 0 & -m'_x v_x & I'_z \omega_z & 0 \\ -m'_x v_x & -m'_z v_z & I'_y \omega_y & I'_x \omega_x & 0 \end{pmatrix}$$ \hspace{1cm} (3.25)

Because the inertial matrix $M$ includes the added-mass terms, $C(v)$ will include them automatically. This explains why an axial motion of hull shaped solids, like a blimp, is intrinsically unstable. Any small angle between the $x$-axis and the direction of motion will tend to increase, because the difference between $m'_x$ and $m'_y$ is responsible for (not canceling) the yaw moment induced by the Coriolis effects [24].

3.2.3 Gravity and Buoyancy Forces

The aerostatic lift force, or buoyancy is explained by the principle of Archimedes and is equal to the weight of the air displaced by the airship. The gravity force together with buoyancy keep the blimp upright, because the gravity force, which
is pointing downwards, works below the buoyant force, directed upwards. The amplitudes of the gravity- and buoyant forces can be expressed by

\[ F_g = mg \quad \text{and} \quad F_b = \rho V g, \quad \text{with} \quad V = \frac{2}{3} \pi ab^2, \quad (3.26) \]

where \( m \) is the mass of the airship, \( g \) is the Earth gravitational acceleration, \( \rho \) is the density of air and \( V \) the volume of the blimp.

The vector \( F_g \) containing the gravitational and buoyancy induced forces and moments is now given by

\[ F_g = \begin{pmatrix}
-(F_g - F_b) \sin(\theta) \\
(F_g - F_b) \cos(\theta) \sin(\phi) \\
-(F_g - F_b) \cos(\theta) \cos(\phi) \\
-b_z F_b \sin(\theta) \\
-b_z F_b \sin(\phi) \\
0
\end{pmatrix}, \quad (3.27)\]

where \( r_z \) is the distance between the center of gravity \( C_g \) and center of buoyancy \( C_b \).

### 3.2.4 Aerodynamic Damping

Aerodynamic damping, or air friction, is depending on the velocity of the blimp, so the aerodynamic damping vector can be expressed as:

\[ F_a = D(v) v. \quad (3.28) \]

The damping matrix \( D(v) \) contains the linear damping coefficients \( D_{v_x}, D_{v_y}, D_{v_z}, D_{\omega_x}, D_{\omega_y}, D_{\omega_z} \) and the quadratic damping coefficients \( D_{v_x^2}, D_{v_y^2}, D_{v_z^2}, D_{\omega_x^2}, D_{\omega_y^2}, D_{\omega_z^2} \) to account for both linear friction due to laminar boundary layers and quadratic friction as a result of turbulent boundary layers [24]:

\[ D(v) = \text{diag} \begin{pmatrix}
D_{v_x} + D_{v_x^2} |v_x| \\
D_{v_y} + D_{v_y^2} |v_y| \\
D_{v_z} + D_{v_z^2} |v_z| \\
D_{\omega_x} + D_{\omega_x^2} |\omega_x| \\
D_{\omega_y} + D_{\omega_y^2} |\omega_y| \\
D_{\omega_z} + D_{\omega_z^2} |\omega_z|
\end{pmatrix}. \quad (3.29) \]

The presented friction model is an approximation, but it works sufficiently well in case of a low speed operation and a highly symmetrical ellipsoid hull [7], which is the case for the blimp.
3.2.5 Propulsion

The blimp contains four identical thrusters. Two lateral thrusters, attached to the gondola, provide thrust in the forward direction, one located on starboard side $F_{x,s}$ and one on port side $F_{x,p}$. These thrusters are also used to steer the blimp. Another thruster $F_z$, also attached to the gondola, provides thrust in the vertical direction. The fourth propeller $F_y$, placed at the back of the gondola, provides thrust in the sideward direction and it also produces torque about the vertical $z$-axis to steer the blimp. The exact location of the different thrusters is given in Figure 3.2.

The thrusters are assumed to be ideal thrusters, whose effects are directly proportional to the motor commands. They only produce a force, the torque generated by the electric motors is not taken into account in the model. The propulsion vector $F_p$ can now be derived:

$$
F_p = \begin{pmatrix}
F_{x,s} + F_{x,p} \\
F_y \\
F_z \\
0 \\
0 \\
F_y r_y + (F_{x,s} - F_{x,p}) r_x
\end{pmatrix}, \tag{3.30}
$$

where $r_y$ is the distance between thruster $F_y$ and the center of gravity $C_g$ along the $x$-axis and $r_x$ is the distance between the lateral thruster $F_{x,s}$, $F_{x,p}$ and $C_g$ along the $y$-axis.

![Figure 3.2: Location of the thrusters](image)
3.3 Full Model

The full mathematical model of the blimp, by combining the kinematic model from Section 3.1 with the dynamic model from Section 3.2, is given below:

\[
\begin{align*}
\dot{x} &= cv\psi\theta v_x + (sv\psi\phi + cv\psi\theta s\phi)v_y + (sv\psi\phi - cv\psi\theta c\phi)v_z, \\
\dot{y} &= -sv\psi\theta v_x + (cv\psi\phi - sv\psi\theta s\phi)v_y + (sv\psi\phi + sv\psi\theta c\phi)v_z, \\
\dot{z} &= s\theta v_x - c\theta s\phi v_y + c\theta c\phi v_z, \\
\dot{\phi} &= \omega_x + s\phi tan\theta \omega_y + c\phi tan\theta \omega_z, \\
\dot{\theta} &= c\phi \omega_y - s\phi \omega_z, \\
\dot{\psi} &= s\phi / c\theta \omega_y + c\phi / c\theta \omega_z, \\
\dot{v}_x &= \frac{1}{m'_x} \left( -m'_y v_y v_z + m'_y v_y \omega_z - (F_g - F_b) \sin(\theta) - (D_{v_x} + D_{v_z} v_x) v_x + (F_{x,s} + F_{x,p}) \right), \\
\dot{v}_y &= \frac{1}{m'_y} \left( m'_x v_x \omega_x - m'_x v_x \omega_z + (F_g - F_b) \cos(\theta) \sin(\phi) - (D_{v_y} + D_{v_z} v_y) v_y + F_y \right), \\
\dot{v}_z &= \frac{1}{m'_z} \left( -m'_y v_y \omega_x + m'_x v_x \omega_y + (F_g - F_b) \cos(\theta) \cos(\phi) - (D_{v_z} + D_{v_y} v_z) v_z + F_z \right), \\
\dot{\omega}_x &= \frac{1}{I'_x} \left( (m'_y - m'_z) v_y v_z + (I'_y - I'_z) \omega_y \omega_z - b_z F_b \cos(\theta) \sin(\phi) - (D_{\omega_x} + D_{\omega_z} \omega_x) \omega_x \right), \\
\dot{\omega}_y &= \frac{1}{I'_y} \left( -(m'_x - m'_z) v_x v_z - (I'_x - I'_z) \omega_x \omega_z - b_z F_b \sin(\theta) - (D_{\omega_y} + D_{\omega_z} \omega_y) \omega_y \right), \\
\dot{\omega}_z &= \frac{1}{I'_z} \left( (m'_x - m'_y) v_x v_y + (I'_x - I'_y) \omega_x \omega_y - (D_{\omega_z} + D_{\omega_y} \omega_z) \omega_z + \left( F_y r_y + (F_{x,s} - F_{x,p}) r_x \right) \right),
\end{align*}
\]

where \( s\phi = \sin(\phi) \) and \( c\phi = \cos(\phi) \).
Chapter 4

Tracking Control

The blimp is a six degrees of freedom system, which has four actuators as can be seen in Chapter 3, so the blimp is an underactuated system. However, a few assumptions can be made which reduce the complexity of the dynamic equations of motion and make it easier to control. For example, two rotations, the roll $\phi$ and the pitch $\theta$, are mechanically stabilized, because the center of gravity lies below the center of buoyancy. Because those two degrees of freedom are mechanically stabilized, there is no need for active stabilization. Also, it is chosen not to control the roll $\phi$ and pitch $\theta$ of the blimp, so it can be assumed that $\dot{\phi} = \dot{\theta} = 0$ and also $\phi = \theta = 0$. Also, some more assumptions have been made that reduce the complexity of the equations of motion. These assumptions, together with the simplified dynamic model, are given in Section 4.1.

The dynamic model of the blimp can be simplified to a four degrees of freedom system as is shown in Section 4.1. There are also four inputs, so the blimp is now fully actuated. However, this does not guarantee that the blimp can be controlled using only simple linear control strategies, because that is not the case for this system. One of the reasons is that the different inputs and outputs are coupled, i.e., if a thrust is generated with the tail rotor, the blimp not only rotates, but it also moves in the lateral direction.

The coupling between the different inputs and outputs is not the only problem in the control of the blimp. Other nonlinearities in the system also make it impossible to design a linear controller that achieves satisfactory performance of the closed loop system. For example, when performing a high-speed motion, the Coriolis and centrifugal effects become active, making the performance of the linear controllers no longer acceptable.

To deal with the nonlinearities and coupling effects in the system, the so-called computed torque method can be used. This method utilizes a feedback linearization control strategy, which decouples and linearizes the input-output relations of the system, making the nonlinear dynamical system linear [12], [19]. This technique is described and applied to the dynamic model of the blimp in Section 4.2. After the implementation of the computed torque method, the resulting linear
system can be used to design other control strategies for the blimp. For example
a tracking controller is developed which can be used to follow a certain desired
trajectory with the blimp. This tracking controlled is described in Section 4.3.
The computed torque method assumes the dynamics of the system is exactly known
and there are no uncertainties. However in real life this is not the case, because
there are many uncertainties. For example the parameters of the dynamic model
of the blimp are not known exactly, also there may exist modeling errors, un-
modeled dynamics and unknown disturbances. These uncertainties will degrade
the performance of the computed torque method. Therefore the computed torque
controller should be made robust with respect to these uncertainties. With the
implementation of the $H_\infty$ compensation method from \cite{12} and \cite{23} it is possible
to achieve this goal. In Section 4.4 the $H_\infty$ compensation method is described and
implemented with respect to the blimp.
The tracking controller, but also the computed torque controller, requires the po-
sition and velocity of the blimp, so they have to be measured. However only
the position of the blimp is measured and this measurement contains a signifi-
cant amount of measurement noise, therefore the derivative of the position can
not be used directly to find the velocity of the blimp. So the final step in the
design of a robust tracking controller is the implementation of a velocity observer
as presented in \cite{3}. This observer finds an estimate of the position and velocity
signals and reduces the amount of measurement noise significantly as can be seen
in Section 4.5.

\section{4.1 Simplified Dynamic Model}

A few assumptions have been made which reduce the complexity of the dynamic
model of the blimp and make it easier to design a suitable control strategy. The
first assumption that can be made is that the blimp stays in a horizontal position,
so the roll $\phi$ and the pitch $\theta$, equal zero and also $\dot{\phi} = \dot{\theta} = 0$. This assumption
is valid, because the center of gravity lies below the center of buoyancy, so both
rotations are mechanically stabilized. Also it is chosen not to control the roll $\phi$ and
pitch $\theta$ of the blimp. This assumptions makes is possible to reduce the six degrees
of freedom system as given in Chapter 3 to a four degrees of freedom system,
having three translations and one rotation. There are also four control inputs, so
each degree of freedom is actuated. The design of a control strategy for this fully
actuated system is less complicated than for the original underactuated system.
Because of the previous assumptions, the vertical thruster $F_z$ can only generate a
force in the vertical $z$-direction. Also the other three thrusters can not generate a
force in this direction and there are no disturbances acting in this direction from
movements in other directions. This makes that the $z$-dynamics can be uncoupled
from the resulting system and the altitude of the blimp can be controlled directly
by the vertical thruster. So it is not needed to include the $z$-dynamics in the
computed torque controller from Section 4.2, also because it is stated in Chapter 2.
that the height is kept constant.

The following assumptions are used to reduce the complexity of the model for use in the development of the computed torque controller in Section 4.2:

- The roll and pitch are zero; $\phi = \theta = 0$ and $\dot{\phi} = \dot{\theta} = 0$.
- The altitude is constant; $z = \text{constant}$ and $\dot{z} = 0$.

After implementation of these assumptions and some rewriting, the simplified model of the equations of motion as given in Chapter 3, become

$$
\text{M}(\text{q}(t), \text{p})\ddot{\text{q}}(t) + \text{h}(\text{q}(t), \dot{\text{q}}(t), \text{p}) = \text{u},
$$

(4.1)

where the state $\text{q} = [x \ y \ \psi]^T$ and $\text{p}$ represents the parameter vector containing the parameters of the blimp. The $3 \times 3$ matrix $\text{M}(\text{q}(t), \text{p})$ contains the mass and inertia components. The $3 \times 1$ vector $\text{h}(\text{q}(t), \dot{\text{q}}(t), \text{p})$ consists of the Coriolis, centripetal, gravity, buoyancy and aerodynamic forces and moments and $\text{u}$ contains the generalized forces from the inputs. The full expressions for $\text{M}(\text{q}(t), \text{p})$, $\text{h}(\text{q}(t), \dot{\text{q}}(t), \text{p})$ and $\text{p}$ can be found in Appendix A.1.

The relation between $\text{u}$ and the actual propulsion from the rotors $\text{F}_p = [F_{x,s} \ F_{x,p} \ F_y]^T$ is given as

$$
\text{F}_p = \text{Q}(\text{p})\text{u},
$$

(4.2)

where the full expression for $\text{Q}(\text{p})$ can also be found in Appendix A.1. Combining (4.1), (4.2) gives the final simplified model of the blimp

$$
\text{Q}(\text{p}) (\text{M}(\text{q}(t), \text{p})\ddot{\text{q}}(t) + \text{h}(\text{q}(t), \dot{\text{q}}(t), \text{p})) = \text{F}_p,
$$

(4.3)

which is used in Section 4.2 to design the computed torque controller.

### 4.2 Computed Torque Method

The computed torque method can be used to decouple and linearize the input-output relations of the system, making the initial nonlinear system linear. The computed torque method utilizes a feedback linearization control strategy, by using the inverse dynamics as a feedforward for the system. A schematic representation of this technique is depicted in Figure 4.1.

The goal of the computed torque method is to find a nonlinear control law $\text{F}_p$ such that the resulting dynamic system is linear. By applying the feedback linearization technique, $\text{F}_p$ may be chosen as

$$
\text{F}_p = \text{Q}(\text{p}) (\text{M}(\text{q}(t), \text{p})\nu + \text{h}(\text{q}(t), \dot{\text{q}}(t), \text{p})).
$$

(4.4)

Combining (4.3) and (4.4) results in the linear system

$$
\ddot{\text{q}} = \nu.
$$

(4.5)
4.3 Tracking Controller

To be able to let the state \( \mathbf{q} = (x, y, \psi)^T \) follow a certain desired trajectory \( \mathbf{q}_d = (x_d, y_d, \psi_d)^T \) a tracking control law \( \nu \) has to be designed for the linear system (4.5). A PD controller is chosen, resulting in

\[
\nu = \ddot{\mathbf{q}}_d - \mathbf{K}_d (\mathbf{q} - \dot{\mathbf{q}}_d) - \mathbf{K}_p (\mathbf{q} - \mathbf{q}_d),
\]

where \( \mathbf{K}_p \) and \( \mathbf{K}_d \) are positive definite diagonal matrices, containing the linear feedback gains. The tracking error \( \tilde{\mathbf{q}}_1 \) is defined by

\[
\tilde{\mathbf{q}}_1 = (\tilde{x}, \tilde{y}, \tilde{\psi})^T = (x - x_d, y - y_d, \psi - \psi_d)^T
\]

and the tracking error dynamics now become

\[
\dddot{\tilde{\mathbf{q}}}_1 + \mathbf{K}_d \ddot{\tilde{\mathbf{q}}}_1 + \mathbf{K}_p \tilde{\mathbf{q}}_1 = 0.
\]

These tracking error dynamics are stable as long as \( \mathbf{K}_p, \mathbf{K}_d > 0 \).

A schematic representation of the computed torque plus tracking controller is depicted in Figure 4.2.

4.3.1 Tracking Simulation

A tracking simulation with the tracking controller from Section 4.3 is performed using Matlab Simulink. The parameters for the computed torque controller (4.4) are assumed to be known exactly and can be found in Chapter 6.5. The gains for the PD controller (4.6) are set to be \( \mathbf{K}_p = (0.3, 0.3, 0.3)I_{3x3} \) and \( \mathbf{K}_p = (0.5, 0.5, 0.5)I_{3x3} \), resulting in good time domain response. The gains can be
chosen higher, but this would result more often in a saturation of the inputs, because the amount of thrust delivered by the propellers is very limited. With the selected gains, if the blimp deviates more than 0.3 m from its desired trajectory the rotors will deliver their maximum thrust. For the simulations the initial position of the blimp is $q_d(0) = (-0.3, 0.2, -0.25 \pi)^T$. The desired trajectory is depicted in Figure 4.3 together with the result from the tracking simulation. The tracking error is depicted in Figure 4.4 and the delivered thrust by the rotors is given in Figure 4.5.

As can be seen in Figure 4.4 the tracking error is very small compared to the...
specified maximum allowable tracking error of 0.5 m, so the tracking controller works well. The tracking error does not completely converge to zero, because sometimes the inputs become saturated as can be seen in Figure 4.5 where \( F_{x,s} \) reaches its maximum when the blimp is cornering. Even though the tracking controller is working well it is impossible to get the same results in real life, because using the inverse dynamics in the control strategy requires that the parameters of the dynamic model are known exactly. This requirement is difficult to satisfy in practice, so in the following section the current control strategy is extended to compensate for this.

### 4.4 Computed Torque Plus \( H_\infty \) Compensation Method

The computed torque controller from Section 4.2 does not take uncertainties into account, but there are many uncertainties that can degrade the performance of the computed torque controller. For example uncertainties in the parameter vector \( p \) in (4.3), unknown disturbances, but also uncertainties as a result of modeling errors and unmodeled dynamics. There are many different methods that can be used to make the controller more robust. In general these can be divided into two major groups, the adaptive control scheme and the robust control scheme. The first method makes use of an adap-
4 Tracking Control

Figure 4.5: Control inputs $F_{x,s}$, $F_{x,p}$, $F_y$, $F_z$ with respect to time.

tation mechanism, which adjusts the estimated model parameters based on the tracking error, on an online basis. The advantage of this method is that it can deal with large uncertainties, because of the self-adjustment of the model parameters, but the drawback of this method is that it requires a lot of on-line computational time. The robust control scheme results in a simple and fixed structure, which does not require a lot of computational time. So this last method is preferred over an adaptation mechanism, because computational power is very limited.

In this section a robust compensation scheme based on $H_\infty$ control theory is used to compensate for the disturbances and uncertainties. More information about this method can be found in [12], [20] and [23].

4.4.1 Formulation of Uncertainty

Before the $H_\infty$ compensation method can be implemented the uncertainties have to be formulated. To include uncertainties, the current control configuration (Figure 4.2) has to be modified, so instead of (4.4), the computed torque control law now becomes

$$F_p = \hat{Q}(\hat{p}) \left( \hat{M}(q(t), \hat{p}) \nu + \hat{h}(q(t), \dot{q}(t), \dot{\hat{p}}) \right),$$

(4.9)

where $\hat{M}$, $\hat{h}$, $\hat{Q}$ represent the nominal values of $M$, $h$, $Q$ respectively and $\hat{p}$ contains the nominal values of the parameters. Substituting (4.9) into (4.5) results
in
\[
\ddot{q} = M^{-1}Q^{-1}\dot{Q}\dot{M}\nu + M^{-1}(q^{-1}\dot{Q}\dot{h} - h), \tag{4.10}
\]
\[
\dot{\nu} = (M^{-1}Q^{-1}\dot{Q}\dot{M} - I)\nu + M^{-1}(Q^{-1}\dot{Q}\dot{h} - h), \tag{4.11}
\]
\[
\nu = \nu + \eta, \tag{4.12}
\]
where
\[
\eta = (M^{-1}Q^{-1}\dot{Q}\dot{M} - I)\nu + M^{-1}(Q^{-1}\dot{Q}\dot{h} - h) \tag{4.13}
\]
is called the uncertainty which can not be handled by the computed torque method.

### 4.4.2 Constructing an Augmented plant with Internal Weightings

The $H_\infty$ technique, which is based on linear systems, can not be used directly, because the original system dynamics are coupled and nonlinear. To be able to use the $H_\infty$ technique an "augmented plant" has to be designed. This augmented plant is based on the remaining linear system, which results after implementation of the computed torque method, together with the implementation of some weightings. These weightings are used to penalize some control signals in order to achieve the performance requirements. The augmented plant is designed according to the procedure described in [12].

A schematic representation of the computed torque controller plus $H_\infty$ compensation method is depicted in Figure 4.6. The new outer loop control law $\nu$ is a combination of the tracking controller $\nu'$ from Section 4.3 (where $\nu'$ equals (4.6)) plus an additional compensation control law $\nu_H$ which compensates for the uncertainty $\eta$:
\[
\nu = \nu' + \nu_H. \tag{4.14}
\]

The compensation control law $\nu_H$ can be represented by
\[
\nu_H = K_c \begin{bmatrix} q - q_d \\ \dot{q} - \dot{q}_d \end{bmatrix}, \tag{4.15}
\]
where $K_c$ is a dynamic controller, designed by $H_\infty$ control theory.

The error dynamics from Section 4.3 have to be extended, because the uncertainty $\eta$ and uncertainty compensation control law $\nu_H$ have to be added, so (4.8) now becomes
\[
\ddot{\tilde{q}}_1 + K_d \dot{\tilde{q}}_1 + K_p \tilde{q}_1 = \nu_H + \eta. \tag{4.16}
\]
If $\tilde{q}_1 = q - q_d$ and $\tilde{q}_2 = \dot{q} - \dot{q}_d$ then (4.16) can be written as
\[
\begin{align*}
\dot{\tilde{q}}_1 &= \tilde{q}_2, \\
\dot{\tilde{q}}_2 &= -K_p \tilde{q}_1 - K_d \tilde{q}_2 + \nu_H + \eta. \tag{4.17}
\end{align*}
\]

In matrix form (4.17) becomes
\[
\begin{align*}
\dot{\tilde{q}} &= A\tilde{q} + B(\nu_H + \eta), \\
\tilde{q} &= C\tilde{q} + D(\nu_H + \eta), \tag{4.18}
\end{align*}
\]
with

\[
A = \begin{bmatrix} O & I \\ -K_p & -K_d \end{bmatrix}_{2n \times 2n} ; \quad B = \begin{bmatrix} O \\ I \end{bmatrix}_{2n \times n} ; \\
C = \begin{bmatrix} I & O \\ O & I \end{bmatrix}_{2n \times 2n} ; \quad D = \begin{bmatrix} O \\ O \end{bmatrix}_{2n \times n} ; \quad \tilde{q} = \begin{bmatrix} \tilde{q}_1 \\ \tilde{q}_2 \end{bmatrix}_{2n \times 1},
\]

where \( I, O \) are respectively the \( n \times n \) identity and zero matrices. The system (4.18) can be represented as an open-loop linear system \( P \), as shown in Figure 4.7, where \( P \) can be written in state space realization:

\[
P(s) \equiv \begin{bmatrix} A & B \\ C & D \end{bmatrix} \equiv C(sI - A)^{-1}B + D \equiv \begin{bmatrix} O & I \\ -K_p & -K_d \\ I & O \\ O & I \end{bmatrix}.
\] (4.19)

To be able to deal with the uncertainties and guaranty stability of (4.18), these uncertainties have to be bounded. So the control law \( \nu_H \) can only be determined if a worst case bound can be found for the unknown uncertainty \( \eta \). According to (4.17) the uncertainty \( \eta \) depends on both the error \( \tilde{q} \) and \( \tilde{v}_H \). To be able to construct a worst case bound on its effect on the system tracking performance, the following assumptions have to be made [12]:

Figure 4.6: Schematic representation computed torque plus \( H_\infty \) compensation method.

Figure 4.7: Open-loop control configuration.
4.4 Computed Torque Plus $H_\infty$ Compensation Method

- $\text{sup} \|\dot{q}_d(t)\| \leq \nu_{\text{max}} < \infty$ for all $t$
- $\|M^{-1}Q^{-1}\dot{Q}\tilde{M} - I\| \leq \lambda < 1$ for all $t$
- $\|Q^{-1}\dot{Q}\tilde{h} - h\| \leq \sigma$ (bounded value), for all $t$
- $\|M^{-1}(q(t), p(t))\| \leq M$ (bounded value), for all $t$

Also suppose a suitable combination of tracking controller $K = [K_p, K_d]$ and uncertainty compensation controller $\nu_H$ can be found that asymptotically stabilize the tracking error $\tilde{q}$ to zero. These stable controllers have to be bounded with $\|\nu_H\| \leq \rho$ and $\|K\| \leq \delta$, where $\rho$ and $\delta$ are bounded values.

Using the assumptions a worst case bound on the uncertainty $\eta$ can be derived as

$$\|\eta\| = \|M^{-1}Q^{-1}\dot{Q}\tilde{M} - I\| (\nu' + \nu_H) + M^{-1}Q^{-1}\dot{Q}\tilde{h} - h\|,$$

$$\leq \lambda \nu_{\text{max}} + \lambda \|K\tilde{q}\| + \lambda \|\nu_H\| + M\sigma,$$

$$\leq \lambda \nu_{\text{max}} + \delta \|\tilde{q}\| + \lambda \rho + M\sigma. \quad (4.20)$$

Since a worst-case bound can be estimated for the uncertainty $\eta$, the control law $\nu_H$ may be designed to guarantee stability of (4.18). The closed loop control system is depicted in Figure 4.8, where the $2n \times 1$ vector $m$ represents additional measurement noise and $K_c$ the dynamic controller, which consists of an internal weighting $W_c$ and stabilizing controller $K_\infty$.

![Figure 4.8: Closed-loop control configuration with internal weighting.](image)

The internal weighting $W_c$ can be designed using the so called "loop shaping design procedure" developed in [13]. This method uses two steps to select a proper $W_c$. First loop shaping is used to shape the nominal plant singular values to give desired open-loop properties. So a weighting is selected that achieves sufficiently high open-loop gain at low frequencies and low open-loop gain at high frequencies, in order to achieve good closed-loop performance, respectively robust stability. A suitable weighting can be for example a simple PI compensator of the form

$$W_c(s) = \frac{s + \alpha}{s}I. \quad (4.21)$$
where $\alpha$ is a constant and $I$ the identity matrix. Another choice for $W_c$ can be of the form of a lag compensator:

$$W_c(s) = \alpha \left( \frac{s + \beta}{s + \gamma} \right) I,$$

where $\beta \gg \gamma$.

The second step is to find a stabilizing controller and verify for robust stability. Verification for robust stability can be carried out by investigating the stability margin of the uncertain dynamic system defined in [12] and [13]

$$\left\| \begin{pmatrix} (I - PW_cK_\infty)^{-1} & (I - PW_cK_\infty)^{-1}P \\ K_\infty(I - PW_cK_\infty)^{-1} & K_\infty(I - PW_cK_\infty)^{-1}P \end{pmatrix} \right\|_\infty =$$

$$\left\| \begin{bmatrix} I \\ K_\infty \end{bmatrix} (I - PW_cK_\infty)^{-1} \begin{bmatrix} I & P \end{bmatrix} \right\|_\infty \leq \epsilon^{-1}, \quad (4.23)$$

where $\| \cdot \|_\infty$ is the $H_\infty$ norm which is the supremum of the largest singular value over all frequencies. The closed-loop transfer functions $(I - PW_cK_\infty)^{-1}$, $(I - PW_cK_\infty)^{-1}P$, $K_\infty(I - PW_cK_\infty)^{-1}$ and $K_\infty(I - PW_cK_\infty)^{-1}P$ represent the gains from $m$ to $y$, $\eta$ to $\tilde{q}$, $m$ to $\nu_H$ and $\eta$ to $\nu'_H$ respectively.

A stabilizing controller $K_\infty$ can now be designed in such a way that it minimizes the reciprocal of the associated stability margin $\epsilon$ for the shaped plant $PW_c$. If it results in $\epsilon \ll 1$, then the specified $W_c$ is incompatible with the robust stability requirements and $W_c$ has to be adjusted accordingly, otherwise a proper $W_c$ is found [13].

In order to refine the design process of the dynamic control law $\nu_H$, Figure 4.8 is modified by adding weightings for $\eta$, $m$ and $\tilde{q}$. The new control scheme is depicted in Figure 4.9. This scheme can be rearranged in order to get a standard $H_\infty$ control configuration, which is depicted in Figure 4.10, where $G$ represents the augmented plant and $K_\infty$ is the controller to be designed.

Figure 4.9: Weighted $H_\infty$ control configuration.
4.4 Computed Torque Plus $H_\infty$ Compensation Method

![Diagram](image)

Figure 4.10: Standard $H_\infty$ control scheme.

The signals $w$, $u$, $z$ and $y$ in Figure 4.10 are defined with respect to Figure 4.9 as

$$ w = \begin{bmatrix} \eta' \\ m' \end{bmatrix}, \quad u = \nu_H' $$

$$ z = \begin{bmatrix} \dot{q}' \\ \nu_H' \end{bmatrix}, \quad y = y $$

(4.24)

(4.25)

The augmented plant $G$ can now be constructed in matrix form

$$ \begin{bmatrix} z \\ y \end{bmatrix} = G \begin{bmatrix} w \\ u \end{bmatrix}, $$

(4.26)

with

$$ G = \begin{bmatrix} W_\dot{q}PW_\eta & O_{2n \times 2n} & W_\eta PW_c \\ O_{n \times n} & O_{n \times 2n} & I_{n \times n} \\ PW_\eta & W_m & PW_c \end{bmatrix}. $$

(4.27)

By substituting (4.19) in (4.28), the state space realization of the augmented plant $G$ can be derived as

$$ G(s) \triangleq \begin{bmatrix} A & B_1 & B_2 \\ C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & D_{22} \end{bmatrix}, $$

(4.28)

with

$$ A = \begin{bmatrix} O & I \\ -K_p & -K_d \end{bmatrix}_{2n \times 2n}, \quad B_1 = \begin{bmatrix} O & O & O \\ W_\eta & O & O \end{bmatrix}_{2n \times 3n}, \quad B_2 = \begin{bmatrix} O \\ W_c \end{bmatrix}_{2n \times n} $$

$$ C_1 = \begin{bmatrix} W_\dot{q} \\ O & O \end{bmatrix}_{3n \times 2n}, \quad C_2 = \begin{bmatrix} I & O \\ O & I \end{bmatrix}_{2n \times 2n}, \quad D_{11} = \begin{bmatrix} O & O & O \\ O & O & O \\ O & O & O \end{bmatrix}_{3n \times 3n} $$

$$ D_{12} = \begin{bmatrix} O \\ O & I \end{bmatrix}_{3n \times 2n}, \quad D_{21} = \begin{bmatrix} O & W_m \end{bmatrix}_{2n \times 3n}, \quad D_{22} = \begin{bmatrix} O \\ O \end{bmatrix}_{2n \times n} $$

where $W_m$ and $W_\dot{q}$ are weighting matrices with dimension $2n$ and $W_\eta$, $W_c$ with dimension $n$.

The following assumptions have to be made in order to obtain a stabilizing controller $K_\infty$ according to $H_\infty$ control theory [20]:
• \((A, B_2, C_2)\) is stabilizable and detectable.

• \(D_{12}\) and \(D_{21}\) have full rank.

• \[
\begin{bmatrix}
A - j\omega I & B_2 \\
C_1 & D_{12}
\end{bmatrix}
\]
has full column rank for all \(\omega\).

• \[
\begin{bmatrix}
A - j\omega I & B_1 \\
C_2 & D_{21}
\end{bmatrix}
\]
has full row rank for all \(\omega\).

In order to fulfill those assumptions the weighting matrices \(W_m\), \(W_\tilde{q}\), \(W_\eta\) and \(W_c\) have to be designed as full rank.

### 4.4.3 Design Procedures

Now the augmented plant \(G\) (4.28) is constructed, the dynamic controller \(K_c\) (4.15) can be designed using the following design process. First the weightings have to be selected, after which the state space realization of the augmented plant is evaluated. If the augmented plant satisfies the design criteria it can be used to find a stabilizing controller using the Robust Control Toolbox from Matlab. If a satisfactory solution cannot be found, reselect the weightings repeat the steps. This procedure can be described in a more systematic way as [12]:

1. Select \(K_p\) and \(K_d\) and evaluate the linear plant \(P\).

2. Design the internal weighting \(W_c\) and check that the condition for robust stability is met using the intermediate \(H_\infty\) stabilizing controller \(K_\infty\) which does not consider \(W_m\), \(W_\tilde{q}\) and \(W_\eta\) yet.

3. Select proper values for \(W_m\), \(W_\tilde{q}\) and \(W_\eta\) for fine tuning, according to the importance of each signal.

4. Evaluate a state space realization of the augmented plant \(G\) by (4.28).

5. Find the final stabilizing controller \(K_\infty\) using the program HINFSYN from the Robust Control Toolbox in Matlab.

6. Calculate the dynamic controller \(K_c\) from \(K_c = W_c K_\infty\).

7. Re-select the weightings and go back to step 2 if the performance requirements are not met.

The implementation of the above design procedure with respect to the blimp can be found in the following section.
4.4.4 Simulation Robust Computed Torque Controller

The theory behind robust control using the computed torque plus \( H_\infty \) compensation method as discussed in Section 4.4 is now implemented with respect to the blimp. So the computed torque controller from Section 4.2 is extended with a stabilizing \( H_\infty \) controller. The \( H_\infty \) controller is developed by traversing the steps given in Section 4.4.3.

**Step 1:** The gains \( K_p \) and \( K_d \) are already selected in Section 4.3.1, so \( K_p = 0.3I_{3 \times 3} \) and \( K_d = 0.5I_{3 \times 3} \). The linear plant \( P \) can be evaluated using (4.19):

\[
P(s) = \begin{bmatrix}
O & I & O \\
-K_p & -K_d & I \\
I & O & O \\
O & I & O \\
\end{bmatrix} \begin{bmatrix}
1/s^2 + 0.5s + 0.3 & 0 & 0 \\
0 & 1/s^2 + 0.5s + 0.3 & 0 \\
0 & 0 & 1/s^2 + 0.5s + 0.3 \\
0 & 0 & 0 \\
\end{bmatrix}
\]

(4.29)

**Step 2:** For the internal weighting \( W_c \), a lag compensation is selected according to (4.22):

\[
W_c(s) = \alpha \left( \frac{s + \beta}{s + \gamma} \right) I_{3 \times 3},
\]

(4.30)

with \( \alpha = 2, \beta = 2 \) and \( \gamma = 0.1 \). The corresponding closed-loop frequency response measures (4.23), are depicted in Figure 4.11 where \( \bar{\sigma}() \) represents the maximum singular value. The stability margin \( \epsilon \) is about 0.65 (1/1.54, see Figure 4.11(a)), so the specified \( W_c \) is compatible with the robust stability requirements.

**Step 3:** The weightings \( W_m, W_\tilde{q} \) and \( W_\eta \) have to be selected according to the importance of each signal. The main goal for the \( H_\infty \) controller is to compensate for the uncertainty \( \eta \), so \( W_\eta \) should be larger than \( W_m \) and \( W_\tilde{q} \). Another demand that is made is that the tracking performance in case of no disturbances should be equal or better when using the \( H_\infty \) controller. Finally \( W_m, W_\tilde{q} \) and \( W_\eta \) should be designed as full rank in order to satisfy the assumption made in Section 4.4.2.

The following weightings are selected that satisfy all requirements:

\[
W_\eta = 0.4I_{3 \times 3},
W_m = 0.2I_{6 \times 6},
W_\tilde{q} = 0.2I_{6 \times 6}.
\]

**Step 4** The state space realization of the augmented plant \( G \) can be evaluated using (4.28).

**Step 5** The final stabilizing controller \( K_\infty \) can be found by using the HINFSYN program from the Robust Control Toolbox in Matlab. The program returns a transfer function with six inputs and three outputs.

**Step 6** The dynamic controller \( K_c \) can now be calculated by

\[
K_c = W_c K_\infty \triangleq \begin{bmatrix}
A_c & B_c \\
C_c & D_c \\
\end{bmatrix},
\]

(4.31)
where the expressions for $A_c$, $B_c$, $C_c$ and $D_c$ can be found in Appendix A.3. Note that the order of the state space model is reduced by removing negligible states.

**Step 7** The designed $H_\infty$ controller is tested by performing a tracking simulation as in Section 4.3.1. To introduce a certain amount of uncertainty, an error has been modeled in the estimated parameter values. The parameters used in the computed torque controller are given in Table 4.1, together with the parameters used in the dynamic model of the blimp. Also at $t = 60$ more uncertainty is introduced by adding a step of 0.2 by the uncertainty $\eta$ for the $x$-direction. The results of this simulations are depicted in Figure 4.12.

Figure 4.12 shows the $H_\infty$ compensation method works well. The mean square error over the complete simulation without the $H_\infty$ compensation method in the $x$, $y$, $\psi$ direction is respectively 0.1505, 0.0054, 0.1379, where the mean square error for the simulation with the $H_\infty$ compensation method is 0.0229, 0.0014, 0.0393, which indicates a substantial improvement in performance. Also the maximum er-

Figure 4.11: Frequency domain measures of the maximum singular value $\tilde{\sigma}(\cdot)$.
4.4 Computed Torque Plus $H_\infty$ Compensation Method

Table 4.1: Parameter difference between dynamic model and computed torque controller.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Dynamic model</th>
<th>Computed torque controller</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m_x'$</td>
<td>0.5563</td>
<td>1.0743</td>
</tr>
<tr>
<td>$m_y'$</td>
<td>0.8066</td>
<td>0.8156</td>
</tr>
<tr>
<td>$I_z$</td>
<td>0.0979</td>
<td>0.0347</td>
</tr>
<tr>
<td>$D_{e_x}$</td>
<td>0.1904</td>
<td>0.3438</td>
</tr>
<tr>
<td>$D_{e_y}$</td>
<td>0.78</td>
<td>0.9606</td>
</tr>
<tr>
<td>$D_{\omega_z}$</td>
<td>0.09</td>
<td>0.0792</td>
</tr>
</tbody>
</table>

Figure 4.12: Error $q - q_d$ with and without $H_\infty$ compensation.
ror in the $x$, $y$, $\psi$ direction for the simulation without compensation for the uncertainties is larger compared to the simulation with the $H_\infty$ compensation method, 0.7317, 0.1503, 0.7181 against 0.2523, 0.0839, 0.4141.

It should be noted that although the presented $H_\infty$ compensation controller performs well according to the simulation results, it does not guarantee that the same controller also performs well when implemented on the real blimp. The weightings $W_c$, $W_\eta$, $W_m$ and $W_\tilde q$ are selected in such a way the simulation performs well and all requirements are met. However when the $H_\infty$ controller is implemented on the real blimp, it is likely that the weightings have to be adjusted in order to satisfy the requirements or to improve the tracking results.

4.5 Position Measurement

The position measurement of the blimp is not very accurate, because the measured signals contain a lot of noise. This can result in a bad tracking performance, but there is another major problem. Only the position of the blimp is measured, but also the velocity is needed for the computed torque controller. The derivative of the position signal can be taken to find the velocity, but the derivative of a noisy signal results in a velocity signal that contains even more noise with very high velocity peaks. This velocity signal can not be used directly because is will result in an unstable system. There are many methods that can be implemented to filter this signal. For example by calculating a moving average (low-pass filter), but the drawback of this method is that it will introduce lag in the closed loop system, which may result in a bad performance.

A more effective way to deal with the measurement noise is by using velocity observers. By using these observers a direct derivative of the position is not needed anymore, making the system controllable by using only position measurements. This method has been developed in [3] and the implementation with respect to the blimp is described in the following section.

4.5.1 Velocity Observer

Because the blimp is a nonlinear system, a nonlinear observer appears to be most obvious, however after implementation of a computed torque controller (see Section 4.2) the remaining system becomes linear:

\[ \ddot q = \nu, \] (4.32)

where $q = [x \ y \ \psi]^T$. For this system a tracking controller has already be determined in Section 4.3 as

\[ \nu = \ddot q_d - K_d(\dot q - \dot q_d) - K_p(q - q_d). \] (4.33)

An observer for this system would be of the form [3]:

\[ \dot \hat q = z + L_d(q - \hat q) \] (4.34)
\[ \dot z = u + L_p(q - \hat q), \] (4.35)
where $\hat{q}$, $\dot{\hat{q}}$ represents the estimate of the position $q$, respectively the velocity $\dot{q}$ and the observer gains $L_d$, $L_p$ are symmetric, positive definite matrices.

The computed torque controller (4.4), including tracking controller (4.33), combined with the linear observer (4.34), (4.35) results in

$$F_p = Q(M(q)\ddot{q} + h(q, \dot{q}))$$  \hspace{1cm} (4.36)

$$\ddot{\nu} = \ddot{q}_d - K_d(\dot{\hat{q}} - \dot{q}_d) - K_p(\hat{q} - q_d)$$  \hspace{1cm} (4.37)

$$\dot{\hat{q}} = \dot{z} + L_d(q - \hat{q})$$  \hspace{1cm} (4.38)

$$\dot{z} = \ddot{\nu} + L_p(q - \hat{q})$$  \hspace{1cm} (4.39)

The proof for exponential stability of the closed loop system (4.36)-(4.39) can be found in [3], [4].

### 4.5.2 Simulation Velocity Observer

To test the effectiveness of the velocity observer a tracking simulations has been performed as in Section 4.3.1. For this simulation the measured states $x$, $y$, $z$, $\psi$ have been discretized with a sample rate of 10 Hz and quantization interval of 0.04 m in order to simulate the discrete behavior of the actual position measurement. Also zero mean white noise, with a variance of 0.04, has been added to the position signal to simulate the measurement noise. The observer gains $L_p$, $L_d$ have been found experimentally and are set to be $L_p = [1 \ 1 \ 2]I_{3x3}$, $L_d = [0.3 \ 0.3 \ 0.6]I_{3x3}$. These settings gave the best estimation of the position and velocity, while the amount of lag is limited. The simulation results are depicted in Figures 4.13 and 4.14.

As can be seen in Figure 4.13 the observer is working well for filtering the noise in the position signal. The mean square error of the actual measured position is 0.04 m, the observer reduces this error to about 0.01 m. Even more important the maximum error is reduced from 0.5 m to 0.15 m. The velocity measurement is also improved by using the observer, as can be seen in Figure 4.14. The mean square error of the actual velocity compared to the derivative of the unfiltered position signal is 8 m/s, with a maximum error of 113 m/s. A low-pass filter using 20 samples improves this result to 0.02 m/s and 0.4 m/s respectively, but introduces a lag in the velocity signal. By using the velocity observer the mean square error is reduced to 0.008 m/s and the maximum error is reduced to 0.17 m/s, which is a vast improvement over the filtered signal. Also the introduced lag is small compared to the filtered signal.

### 4.6 A Worst Case Scenario Tracking Simulation

To show the effect of the velocity observer and the $H_\infty$ compensation method on the performance of the final tracking controller, a concluding simulation is performed. This simulation represents a worst case scenario to show that the presented tracking controller still performs well in case the actual blimp can not
be build according to the specifications. So this simulations shows for example the effect in case the actual positioning system of the blimp is not able to meet the accuracy requirements, because for example the position measurement contains a large amount of measurement noise. At the same time the positioning system requires time to calculate the position, which introduces a lag in the position information that can not be neglected. Not only there is a problem with the positioning system, but also the controller requires a relatively large amount of computational time, which also introduces a lag between the controller and the actual blimp. Finally the parameters of the computed torque controller could be estimated incorrectly, so this is also taken into account.

The sketched situation is simulated for different configurations of the controller. The first configuration of the controller consists of only the tracking controller from Section 4.3. The second configuration is a combination of the tracking controller with the $H_\infty$ compensation method from Section 4.4. For the third configura-
A Worst Case Scenario Tracking Simulation

To simulate the different problems which can arise in case the tracking controller is implemented on the real blimp, the following disturbances are added to the Matlab Simulink model. Zero mean white noise with a variance of 0.04 m is added to the position signal to simulate measurement noise, together with a transport delay of 0.5 seconds to introduce lag in the position signal. This signal is then quantized to a 6 Hz signal with a quantization interval of 0.03 m. The lag between the controller and the blimp is modeled with a transport delay of one sample time, so $\frac{1}{6}$ second.

To simulate uncertainty in the parameters of the computed torque controller, a random fraction is added to each parameter, obtained from a normal distribution with a standard deviation of 0.5. The used parameters are given in Table 4.2.

For this final simulation the same gains are used for the controllers as used earlier.

Figure 4.14: Comparison between the low-pass filtered velocity signal, the estimated velocity from the observer and the actual velocity.
4 Tracking Control

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Dynamic model</th>
<th>Computed torque controller</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m'_x$</td>
<td>0.624</td>
<td>0.4404</td>
</tr>
<tr>
<td>$m'_y$</td>
<td>0.805</td>
<td>1.6837</td>
</tr>
<tr>
<td>$I'_z$</td>
<td>0.275</td>
<td>0.2562</td>
</tr>
<tr>
<td>$D_{\nu x}$</td>
<td>0.184</td>
<td>0.1945</td>
</tr>
<tr>
<td>$D_{\nu y}$</td>
<td>0.472</td>
<td>0.7238</td>
</tr>
<tr>
<td>$D_{\omega z}$</td>
<td>0.028</td>
<td>0.0288</td>
</tr>
</tbody>
</table>

Table 4.2: Parameter difference between dynamic model and computed torque controller.

for the separate simulations. The simulations results are depicted in Figure 4.15.

| mse($\cdot$) | $x - x_d$ | $y - y_d$ | $\psi - \psi_d$ | $|x - x_d|$ | $|y - y_d|$ | $|\psi - \psi_d|$ |
|--------------|-----------|-----------|-----------------|-----------|-----------|-----------------|
| no observer, no $H_\infty$ | 3.7938 | 35.9292 | 1.6782 | 3.9445 | 9.9864 | 2.8795 |
| no observer, $H_\infty$ | 6.9584 | 7.3202 | 0.9617 | 4.6690 | 4.3109 | 2.0193 |
| observer, no $H_\infty$ | 0.6781 | 0.7861 | 0.1905 | 1.9723 | 2.0177 | 0.8629 |
| observer, $H_\infty$ | 0.3216 | 0.3734 | 0.2071 | 1.2420 | 1.5509 | 1.1763 |

Table 4.3: The mean square error and maximum error for different configurations of the tracking controller.

Figure 4.15(a) shows the desired trajectory for $x$, $y$ and $\psi$ respectively and Figure 4.15(b) shows the tracking errors for the different configurations of the tracking controller. In Table 4.3 the mean square errors (mse) over the complete simulation are given together with the maximum errors for the four controller configurations. These results show the configuration without $H_\infty$ compensation and without the observer performs worst and is not very useful when dealing with this kind of disturbances. By adding the $H_\infty$ compensation method the results show some improvement, but this controller configuration is still not very useful. A huge improvement is made when the velocity observer is used. Both the mean square error and the maximum error are small compared to a simulation without the observer. By also adding the $H_\infty$ compensation method to this last configuration the results improve approximately with a factor two.

It can be concluded that the velocity observer is required with these kind of disturbances in order to get satisfying results. Adding the $H_\infty$ compensation improves the results even further, but the specified accuracy requirement (maximum error of 0.5 m) is still not met in this situation. It should be noted here that the introduced disturbances in the simulation are a large exceeding of the specification and represent a worst case scenario. It may therefore be concluded that the presented tracking controller still performs well in this situation, so it is likely that it will also perform well when implemented on the real blimp, which uncertainties should be much lower.
Figure 4.15: Final tracking simulation, representing a worst case scenario for different configurations of the tracking controller.
Chapter 5

Formation Flight:
Synchronization

In Chapter 2 it is stated that two blimps should be able to make a formation flight with each other. In other words they should be able to synchronize together. Different synchronization strategies are possible in order to make the formation flight. In this chapter the different synchronization strategies are discussed and simulations are performed to be able to test their effectiveness in order to choose the most suitable synchronization strategy.

First a general synchronization strategy is discussed which can be used for both master-slave and mutual synchronization. This general synchronization controller is described in Section 5.1. In Section 5.2 the general synchronization controller is used for master-slave synchronization between two blimps, where the master blimp has to follow a desired trajectory and the slave blimp has to follow the master. In this situation the master blimp does not have any notion of the slave and the slave does not know the desired trajectory the master blimp has to fly.

In Section 5.3 a mutual synchronization strategy is discussed, where each blimp should follow the desired trajectory, while at the same time they should mutually synchronize. The difference with the master-slave synchronization is that both blimps know the desired trajectory and both blimps also try to stay in formation, so basically the blimps are identical.

Finally in Section 5.4 both synchronization controllers are extended with a velocity observer and compared again. This velocity observer is implemented, because both synchronization controllers require position and velocity information of the blimps, while only the position is measured and also the position measurement contains measurement noise. By using observers for the different states the velocity can be estimated and the measured position can be filtered. Simulations are used to test the effectiveness of the synchronization controllers based on estimated variables and finally the most suitable controller to be used for this project is selected.
5.1 Synchronization Controller

A synchronization controller $\nu_i$ is designed for blimp $i = 1, \ldots, p$, where blimp $i$ can be described after implementation of the computed torque controller from Chapter 4.2 as

$$\ddot{q}_i = \nu_i.$$  \hfill (5.1)

The synchronization controller is defined according to [16] as

$$\nu_i = \ddot{q}_{r,i} - K_{d,i} \dot{s}_i - K_{p,i} s_i,$$  \hfill (5.2)

where $K_{p,i}$, $K_{d,i}$ are positive definite diagonal gain matrices and $s_i$, $\dot{s}_i$ are the synchronization errors defined by

$$s_i = q_i - q_{r,i} \quad \dot{s}_i = \dot{q}_i - \dot{q}_{r,i}.$$  \hfill (5.3)

The reference signals $q_{r,i}$, $\dot{q}_{r,i}$, $\ddot{q}_{r,i}$ define the mutual interaction between the blimps and with respect to their desired trajectory and are selected such that they guarantee the synchronous behavior:

$$q_{r,i} = q_d - p \sum_{j=1, j \neq i}^p K_{cp_{i,j}} (q_i - q_j),$$

$$\dot{q}_{r,i} = \dot{q}_d - p \sum_{j=1, j \neq i}^p K_{cv_{i,j}} (\dot{q}_i - \dot{q}_j),$$

$$\ddot{q}_{r,i} = \ddot{q}_d - p \sum_{j=1, j \neq i}^p K_{ca_{i,j}} (\ddot{q}_i - \ddot{q}_j),$$  \hfill (5.4)

where $K_{cp_{i,j}}$, $K_{cv_{i,j}}$, $K_{ca_{i,j}}$, $i$, $j = 1, \ldots, p$, are positive semi-definite diagonal matrices that define the interactions between the blimps. They allow to weigh the synchronization error between the blimps and the desired common trajectory, so a priority to synchronization between the blimps, or with respect to the common desired trajectory can be assigned. This is particularly important during transients when large errors can cause instability and/or compromise the synchronous behavior of the complete system existing of multiple blimps [16].

The partial synchronization errors between the $i$th and the $j$th blimp is defined by

$$e_{i,j} = q_i - q_j, \quad \dot{e}_{i,j} = \dot{q}_i - \dot{q}_j,$$  \hfill (5.5)

for all $i, j = 1, \ldots, p$, $i \neq j$ and for $j = i$ as

$$e_{i,i} = q_i - q_d, \quad \dot{e}_{i,i} = \dot{q}_i - \dot{q}_d.$$  \hfill (5.6)

The synchronization errors $s_i$, $\dot{s}_i$ defined by (5.3) can now be written as

$$s_i = e_{i,i} + p \sum_{j=1, j \neq i}^p K_{cp_{i,j}} e_{i,j}, \quad \dot{s}_i = \dot{e}_{i,i} + p \sum_{j=1, j \neq i}^p K_{cv_{i,j}} \dot{e}_{i,j}.$$  \hfill (5.7)
In [16] global-asymptotic stability of $s_i$, $\dot{s}_i$ is proven, which results in global-asymptotic synchronization between the systems. The presented general synchronization controller is developed in [16] as a mutual synchronization controller, but it can also be used for partial and master-slave synchronization by setting some $K_{ca_{i,j}}$, $K_{cv_{i,j}}$, $K_{cp_{i,j}} = 0$. First the presented controller is implemented for a master-slave synchronization problem, discussed in Section 5.2. In Section 5.3 the synchronization problem is extended with a full mutual synchronization controller.

5.2 Master-Slave Synchronization

A master-slave synchronization problem between two blimps is studied, where blimp 1 is the master blimp which has to follow a predefined trajectory and blimp 2 acts as a slave which has to follow the master. In this case, the master is not aware of the slave, so it will not act in case the slave can not keep up with the master.

As described in Chapter 2.2.2 during formation flight one blimp, in this case the master, has to follow the desired rectangular trajectory, while the second blimp, the slave, should fly along side, maintaining a preprogrammed distance between the two blimps. So the blimps must maintain a fixed distance $a$ between their centers of gravity $C_g$ and also maintain an equal heading $\psi$ as depicted in Figure 5.1. The relation between the states of blimp $1$ and $2$ can be written as

$$
\begin{align*}
x_2 &= x_1 + \sin(\psi_1)a, \\
y_2 &= y_1 + \cos(\psi_1)a, \\
\psi_2 &= \psi_1.
\end{align*}
$$

(5.8)

The synchronization controller from Section 5.1 is adapted first for the master blimp. There is no interaction with the slave, so $K_{ca_{i,j}} = K_{cv_{i,j}} = K_{cp_{i,j}} = 0$ in (5.7). The control law $\nu_1$ can now be defined by combining (5.2), (5.4) and (5.7):

$$
\nu_1 = \ddot{q}_{r,1} - K_{d,1}\dot{s}_1 - K_{p,1}s_1,
= \ddot{q}_d - K_{d,1}(q_1 - \dot{q}_d) - K_{p,1}(q_1 - q_d),
$$

(5.9)

where $K_{d,1} > 0$, $K_{p,1} > 0$ and $q_d$ is the desired trajectory defined in Chapter 2.2.2. The synchronization controller for the master blimp (5.9) equals the tracking controller from Chapter 4.3 which does make sense, because the master only has to follow the desired trajectory.

The control law for the slave blimp is also based on the general synchronization controller from Section 5.1. The difference with the controller for the master is that this time the slave has to follow the master, so the interaction between the blimps defined by $K_{ca_{i,j}}$, $K_{cv_{i,j}}$, $K_{cp_{i,j}}$ in (5.4) should be chosen as $K_{ca_{i,j}}$, $K_{cv_{i,j}}$, $K_{cp_{i,j}} > 0$. 


5.2 Master-Slave Synchronization

The slave should follow the master blimp along side with a fixed distance between them, without knowing the desired trajectory of the master. So the slave has to follow the master according to (5.8): \( q_2 = q_1 + \Delta q \), where \( \Delta q = [\sin(\psi_1)a, \cos(\psi_1)a, 0]^T \).

The control law \( \nu_2 \) for the blimp can now be defined as

\[
\nu_2 = \dot{q}_{r,2} - K_{d,2} \ddot{q}_2 - K_{p,2} \dot{q}_2,
\]

\[
= -K_{d,2}(\dot{q}_2 - (\dot{q}_1 + \Delta \dot{q})) - K_{p,2}(q_2 - (q_1 + \Delta q)).
\] (5.10)

Note that \( K_{cp,2,1} = K_{cv,2,1} = I \), because the slave only has to follow the master and is not aware of the global desired trajectory. The gain \( K_{ca,2,1} \) is set to be \( K_{ca,2,1} = 0 \), so no acceleration information is needed in order to reduce the complexity of the control strategy. This is a simplification of the synchronization controller from Section 5.1 and it will result in a slight loss of performance, because \( K_{ca,2,1} \) originally should be chosen as \( K_{ca,2,1} > 0 \).

5.2.1 Master-Slave Synchronization Simulation

To test the effectiveness of the controllers for the master-slave synchronization a simulation is performed using Matlab Simulink. No uncertainties are modeled for this simulation, so the parameters for the computed torque controller are known exactly and there is no measurement noise. The gains \( K_{p,1}, K_{d,1} \) for the tracking controller of the master blimp are chosen equal to the gains from the tracking simulation from Chapter 4.3.1 so \( K_{p,1} = 0.3I_{3x3}, K_{d,1} = 0.5I_{3x3} \). The gains for the synchronization controller from the slave \( K_{p,2}, K_{d,2} \) are chosen to be \( K_{p,1} = \)
$0.5I_{3\times3}, \mathbf{K}_{d,1} = 1.5I_{3\times3}$ in order to get sufficient synchronization between the master and the slave. The distance $a$ between the blimps in (5.8) is set to be $a = 2$. For the simulation, the initial position of the master is $[x, y, \psi] = [0.5, -0.5, 0]$ and for the slave $[x, y, \psi] = [0, 2, 0.25\pi]$. The simulation results are depicted in Figure 5.2.

![Figure 5.2](image)

**Figure 5.2**: Master-slave synchronization, comparison between blimp 1 (master) and blimp 2 (slave).

Figure 5.2 shows the master-slave synchronization is working well. The tracking performance of the master blimp equals the results from Chapter 4.3.1. The error does not converge to zero completely, because sometimes the required input for the master blimp becomes saturated. Also the synchronization controller of the slave performs well. The initial synchronization error is reduced relatively quick and the overall error stays well below the maximum allowable tolerance of 0.5m. The
error can be reduced further by increasing the gains $K_{p,2}$, $K_{d,2}$, but this would result in saturations of the inputs more often. Because with the original gains, the inputs sometimes already exceed the maximum available thrust, as can be seen in Figure 5.3 where the force produced by the thrusters of the slave is depicted.

Figure 5.3: Force produced by the thrusters of the slave blimp, together with their maximum thrust.

5.3 Mutual Synchronization

In this section the master-slave synchronization problem from Section 5.2 is extended to a mutual synchronization between the two blimps. This means the master blimp is also paying attention to the slave, instead of only looking at its desired trajectory. The advantage of this approach is that it is possible to minimize the synchronization error even in case the slave can not keep up with the master. For the mutual synchronization flight, the blimps still have to fly along side each other with a distance of 2 m between them. Blimp 1 will follow the rectangular desired trajectory $q_d$ and the desired trajectory for blimp 2 $q_{d,2}$ is given by

$$q_{d,2} = q_d + \Delta q,$$

(5.11)

where $\Delta q = [\sin(\psi_1)a, \cos(\psi_1)a, 0]^T$.

In order to get mutual interaction between the two blimps, the tracking controller for the master blimp, used for the master-slave synchronization from Section 5.2 has to be extended. This is done by choosing the gains $K_{ca,2}$, $K_{cv,2} > 0$ which define the level of interaction between blimp 1 and blimp 2 in (5.4). So the complete synchronization controller, developed in Section 5.1 is implemented for blimp 1,
resulting in
\[
\nu_1 = \ddot{q}_{r,1} - K_{d,1}\dot{s}_1 - K_{p,1}s_1,
\]
\[
= \ddot{q}_d - K_{d,1}(\dot{q}_1 - \dot{q}_d + K_{cv_{1,2}}(\dot{q}_1 - (\dot{q}_2 - \Delta \dot{q})))
- K_{p,1}(q_1 - q_d + K_{cp_{1,2}}(q_1 - (q_2 - \Delta q))),
\] (5.12)

Note that $K_{ca_{1,2}} = 0$ in (5.4), which reduces the complexity of the control strategy, because there is no need for acceleration information and only the position and velocity of the blimps has to be exchanged.

The controller $\nu_2$ for blimp 2 is almost identical to the controller of the slave as presented in Section 5.2, except for the mutual synchronization blimp 2 also has its own desired trajectory given by (5.11). The controller for blimp 2 is now given by
\[
\nu_2 = \ddot{q}_{d,2} - K_{d,2}(\dot{q}_2 - \dot{q}_{d,2} + K_{cv_{2,1}}(\dot{q}_2 - (\dot{q}_1 + \Delta \dot{q})))
- K_{p,2}(q_2 - q_{d,2} + K_{cp_{2,1}}(q_2 - (q_1 + \Delta q))).
\] (5.13)

### 5.3.1 Mutual Synchronization Simulation

Also for the mutual synchronization situation a simulation is performed using Matlab Simulink. The performed simulation is comparable to the simulation performed for the master-slave synchronization, so again there are no uncertainties and the initial conditions are the same as in Section 5.2. The gains for the mutual synchronization controllers for blimp 1 and 2 are chosen in the same order of magnitude as the gains used for the master-slave synchronization, to be able to compare the simulation results. The gains $K_{p,1}, K_{d,1}, K_{cp_{1,2}}, K_{cv_{1,2}}$ for the controller $\nu_1$ (5.12) are therefore set to be $K_{p,1} = 0.3I, K_{d,1} = 0.5I, K_{cp_{1,2}} = I, K_{cv_{1,2}} = I$. The gains for $\nu_2$ are chosen to be $K_{p,2} = 0.5I, K_{d,2} = 1.5I, K_{cp_{2,1}} = I, K_{cv_{2,1}} = I$. The weighting gains $K_{cp_{i,j}}, K_{cv_{i,j}}$ equal the identity matrix, so following the desired trajectory is as important as following the other blimp.

The simulation results, depicted in Figure 5.4 show the mutual synchronization works well. The results are comparable with the results from the master-slave synchronization simulation, although the mutual synchronization performs a little bit better. The error between blimp 1 and blimp 2 is a bit smaller compared to the results from the master-slave simulation and also the mutual synchronization responds a little bit faster to the initial error and there is less overshoot. On the other hand, the tracking performance of blimp 1 with respect to the desired trajectory $q_d$ is a little bit worse. Because blimp 1 also tries to minimize the error between the two blimps, the required thrust for blimp 2 is less often saturated, as can be seen in Figure 5.4(c).

### 5.4 Synchronization with uncertainties

For both synchronization controllers, the master-slave synchronization controller from Section 5.2 and the mutual synchronization controller from Section 5.3 it is
assumed all states can be measured and there are no uncertainties. As is stated already in Chapter 4 these assumptions cannot be made when dealing with the actual blimps. It is for example not possible to measure all states, because only the position of the blimp is measured and this measurement also contains measurement noise. Another uncertainty it is dealt with is the uncertainty in the parameters present in the synchronization controller. In this section both synchronization controllers are extended and based on estimated variables instead of the actual states. These new controllers are compared again by means of simulations to find which controller is most suitable to achieve synchronization in case uncertainties are present in the system.
5.4.1 Synchronization Controller Based On Estimated Variables

The synchronization controller from Section 5.1 is extended according to [16] so it only requires position measurement, which can include measurement noise. So the controller $\nu_i$ for the $i$th blimp, (5.2), can be modified to

$$\nu_i = \hat{q}_{r,i} - K_{d,i} \hat{s}_i - K_{p,i} s_i,$$

(5.14)

where $\hat{s}_i$ denotes the estimate of the velocity synchronization error $s_i$ (5.3) and $\hat{q}_{r,i} \hat{q}_{r,i}$ and $\hat{q}_{r,i}$ are estimates of the velocity $q_i$ and the reference signals $\hat{q}_{r,i}, \hat{q}_{r,i}$ (5.4) respectively. The reference signals are given by

$$\hat{q}_{r,i} = \hat{q}_d - \sum_{j=1, j \neq i}^{p} K_{cv,i,j} (\hat{q}_i - \hat{q}_j),$$

(5.15)

$$\hat{q}_{r,i} = \hat{q}_d - \sum_{j=1, j \neq i}^{p} K_{ca,i,j} (\hat{q}_i - \hat{q}_j),$$

(5.16)

where $\hat{q}_i$ corresponds to the derivative of the velocity estimate of the $i$th blimp, i.e.

$$\hat{q}_i = \frac{d}{dt} \hat{q}_i.$$  

(5.17)

For the estimate of the velocity synchronization error $\hat{s}_i$ is follows that

$$\hat{s}_i = \dot{\hat{q}}_i - \dot{\hat{q}}_{r,i},$$

(5.18)

or written in terms of the estimates for the partial synchronization error:

$$\hat{s}_i = \dot{\hat{e}}_{i,i} + \sum_{j=1, j \neq i}^{p} K_{cv,i,j} \dot{\hat{e}}_{i,j},$$

(5.19)

where $\dot{\hat{e}}_{i,j}$, $\dot{\hat{e}}_{i,i}$ are given by

$$\dot{\hat{e}}_{i,j} = \hat{q}_i - \hat{q}_j, \quad \text{for all } i, j = 1, ..., p, \quad i \neq j$$

$$\dot{\hat{e}}_{i,i} = \hat{q}_i - \hat{q}_d, \quad \text{for } i = j.$$  

An observer for estimates of the position and velocity of blimp $i$, $\hat{q}_i$ and $\hat{q}_i$ respectively, is already developed in Chapter 4.5.1 and can also be used in combination with the synchronization controller. More information about observers in combination with the presented synchronization controller can be found in [16].

5.4.2 Comparison Master-Slave vs. Mutual Synchronization

The synchronization controller based on estimated variables from Section 5.4.1 can be used for both master-slave synchronization and for mutual synchronization.
Those two synchronization strategies are now compared by means of simulations in order to find the most effective strategy in case only positions measurements are present which also posses some measurement noise. The simulations for both the master-slave synchronization controller and the mutual synchronization controller are performed again using Matlab Simulink. The gains for both controllers and the initial position of the blimps are kept equal to the gains and initial position as used in Section 5.2.1 and 5.3.1. The gains for the velocity observers equal the gains used for the observer from the tracking simulation (Chapter 4.5.2). To simulate measurement noise, zero mean white noise with a variance of 0.04 m is added to the discretized position signal of the blimps. The results from both simulations are depicted in Figure 5.5 and in Table 5.1 the mean square errors (mse) over the complete simulation of 120 seconds are given.

<table>
<thead>
<tr>
<th>mean square error</th>
<th>master-slave</th>
<th>mutual</th>
</tr>
</thead>
<tbody>
<tr>
<td>$mse(x_1 - x_d)$</td>
<td>0.0936</td>
<td>0.2938</td>
</tr>
<tr>
<td>$mse(y_1 - y_d)$</td>
<td>0.0681</td>
<td>0.2014</td>
</tr>
<tr>
<td>$mse(\psi_1 - \psi_d)$</td>
<td>0.0076</td>
<td>0.0392</td>
</tr>
<tr>
<td>$mse(x_2 - x_d, 2)$</td>
<td>2.3828</td>
<td>0.1322</td>
</tr>
<tr>
<td>$mse(y_2 - y_d, 2)$</td>
<td>0.4515</td>
<td>0.1088</td>
</tr>
<tr>
<td>$mse(\psi_2 - \psi_d, 2)$</td>
<td>0.3011</td>
<td>0.0598</td>
</tr>
<tr>
<td>$mse(x_2 - x_1)$</td>
<td>1.1318</td>
<td>1.4562</td>
</tr>
<tr>
<td>$mse(y_2 - y_1)$</td>
<td>3.0690</td>
<td>2.4842</td>
</tr>
<tr>
<td>$mse(\psi_2 - \psi_1)$</td>
<td>0.3011</td>
<td>0.0598</td>
</tr>
</tbody>
</table>

Table 5.1: The mean square errors (mse) for the master-slave and mutual synchronization simulations.

The results of the two different synchronization strategies, as depicted in Figure 5.5, are very comparable. There are only minor differences between the two methods. As can be expected, the tracking performance of blimp 1 with respect to the desired trajectory is better in case a master-slave synchronization strategy is used, which is also supported by the mean square error values given in Table 5.1. On the other hand, the mean square error of the difference between blimp 1 and blimp 2 is better in case a mutual synchronization strategy is used, which also gives better results when looking at the mean square error of the difference between blimp 2 and its desired trajectory.

Whether to use a master-slave or a mutual synchronization strategy depends on the requirements of the synchronization between the two blimps. If more importance is laid to the overall tracking performance, a master-slave strategy should be used. On the other hand, if the synchronization between the blimps is more important a mutual synchronization strategy should be used.

With respect to the requirements defined for this project a mutual synchronization strategy is the better choice, because synchronization is more important than following the desired trajectory. For example if blimp 2 can not keep up with blimp 1, blimp 1 should slow down and deviate from its desired trajectory, so blimp 2 can catch
Figure 5.5: Comparison between the master-slave and the mutual synchronization controller based on estimated variables.

Another reason to prefer mutual synchronization in this case is that eventually all blimps, c.q. Atalanta’s, should be identically and when using a master-slave synchronization strategy this will not be possible.

5.5 A Worst Case Scenario Synchronization Simulation

To investigate how the mutual synchronization controller works for the worst case scenario sketched in Chapter 4.6, a final synchronization simulation is performed comparable with the final tracking simulation from Chapter 4.6. This simulation shows how the mutual synchronization controller performs in case the real blimp
can not be build according to its specifications and if that is the case, is the pre-
sented synchronization controller still usable? This is investigated for the situation
where the velocity observer from Section 5.4.1 is not implemented and for the sit-
uation where the velocity observer is added to the synchronization controller.
The same disturbances with respect to measurement noise, signal lag and para-
meter uncertainties, are modeled as in Chapter 4.6 plus a transport delay of 0.5
seconds is added for the communication between blimp 1 and blimp 2 and the
other way around. The controller gains are selected equal to the gains used for
the other simulations in this chapter. The results of this simulation are depicted
in Figure 5.6
Figure 5.6(a) shows the trajectory of blimp 1 for both the synchronization con-
troller without the velocity observer and with the velocity observer respectively.
Figure 5.6(b) shows the same picture, except for blimp 2. The synchronization
error between blimp 1 and blimp 2 is depicted for both cases in Figure 5.6(c).
The simulation results show the synchronization controller without the velocity
observer is not usable when dealing with these large disturbances as the synchro-
nization error exceeds the 5 m most of the time. The synchronization controller
combined with the velocity observer performs much better. Although the maxi-
mum allowable synchronization error is exceeded, it stays below 2 m. It can be
concluded that the synchronization controller combined with the velocity observer
is usable for very large disturbances, so it should also work on the real blimp as
the uncertainties should be much lower.
Figure 5.6: Final mutual synchronization simulation, representing a worst case scenario.
Chapter 6

Parameter Identification

The dynamic model of the blimp (see Chapter 3) contains many physical parameters, for example the mass and inertia of the blimp and the different friction coefficients. Most of these parameters can not be measured directly, so they have to be estimated. It is important to have a good estimation of the parameters, because most of them are used for the computed torque controller from Chapter 4.2.

A Kalman filter technique is selected for the identification of the different parameters. Because the model of the nonlinear system is continuous and measurements of the states are discrete, the Continuous-Discrete Extended Kalman Filter (CDEKF) [8] is used. The CDEKF algorithm is described in Section 6.1 and the implementation of the CDEKF algorithm with respect to the blimp is discussed in Section 6.2.

Measurements are needed for the CDEKF algorithm. To be able to estimate all parameters, twelve different states (position, orientation and their derivatives) have to be measured during motion of the blimp. Measurements of the states are taken using a webcam and the images are processed with a Matlab algorithm. The results of the measurements are given in Section 6.3.

In Section 6.4 the different settings and initial values for the CDEKF algorithm are discussed. The final results of the parameter identification are given in Section 6.5 and in Section 6.6 a few comments are given with respect to these results.

6.1 Continuous-Discrete Extended Kalman Filter

To describe the Continuous-Discrete Extended Kalman Filter (CDEKF) algorithm, a notation for the continuous system model is given together with the discrete measurement model. Secondly the algorithm for the propagation of the estimated state between the measurements is discussed and finally the measurement update process will be described. The CDEKF algorithm is elaborately described in [8].
6.1 Continuous-Discrete Extended Kalman Filter

6.1.1 Continuous System- and Discontinuous Measurement model

In this chapter the nonlinear system as described in Chapter 3 is written according to the following continuous system model

\[ \dot{x} = f(x(t), u(t)) + w(t); \quad w(t) \sim N(0, Q(t)). \] (6.1)

The vector \( f \) is a nonlinear function of the state \( x(t) \) and the input \( u(t) \). Model uncertainties are represented by \( w(t) \) which is assumed to be zero mean Gaussian noise having covariance matrix \( Q(t) \).

The discrete measurement model is given as

\[ z_k = h_k(x(t_k)) + v_k; \quad k = 1, 2, \ldots; \quad v_k \sim N(0, R_k), \] (6.2)

where the vector \( h_k(x(t_k)) \) contains the measurement values, which depend upon both the index \( k \) and the state at each sampling time. Measurement noise is represented by \( v_k \) and is assumed to be zero mean Gaussian noise with associated covariance matrix \( R_k \). The noises \( w(t) \) and \( v_k \) are assumed to be uncorrelated, so \( E[w(t)v_k^T] = 0 \) for all \( k \) and \( t \).

6.1.2 Propagation Between Measurements

The CDEKF algorithm starts with initial condition

\[ x(0) \sim N(\hat{x}_0, P_0), \] (6.3)

where \( \hat{x}_0 \) and \( P_0 \) are the initial conditions for the estimated state \( x_k \) and error covariance matrix \( P_k \). Since measurements are taken at discrete times, no information from measurements is available between \( t_k \) and \( t_{k+1} \) for the state of the continuous model. So between this interval a continuous time propagation is made for the estimated state \( \hat{x}(t_k) \):

\[ \dot{\hat{x}}(t) = f(\hat{x}(t), u(t)). \] (6.4)

Similarly, the propagation between measurements of the matrix \( P(t) \), representing an estimate of the error covariance matrix, is governed by

\[ \dot{P}(t) = F(\hat{x}(t) , t)P(t) + P(t)F^T(\hat{x}(t), t) + Q(t), \] (6.5)

where the Jacobian matrix \( F(\hat{x}(t), t) \) is defined as

\[ F(\hat{x}(t), t) = \frac{\partial f(x(t), t)}{\partial x(t)} \bigg|_{x(t) = \hat{x}(t)}. \] (6.6)
6.1.3 Measurement Update

When a new measurement is available the state and error covariance matrix will be updated. Prior to the update, at time $t_k$, the discrete state estimate and estimate of the error covariance matrix are denoted as $\hat{x}_k(-)$ and $P_k(-)$. After a measurement is taken, the discrete time estimate of the state and estimate of the error covariance matrix are updated to $\hat{x}_k(+)\) and $P_k(+)\) respectively, according to

$$\hat{x}_k(+) = \hat{x}_k(-) + K_k \left[ z_k - h_k(\hat{x}_k(-)) \right] \quad (6.7)$$

and

$$P_k(+)= \left[ I - K_k H_k(\hat{x}_k(-)) \right] P_k(-). \quad (6.8)$$

The Kalman gain matrix $K_k$ is given by

$$K_k = P_k(-) H_k^T(\hat{x}_k(-)) \left[ H_k(\hat{x}_k(-)) P_k(-) H_k^T(\hat{x}_k(-)) + R_k \right]^{-1}, \quad (6.9)$$

where the Jacobian $H_k(\hat{x}_k(-))$ is defined as

$$H_k(\hat{x}_k(-)) = \left. \frac{\partial h_k(x(t_k))}{\partial x(t_k)} \right|_{x(t_k)=\hat{x}_k(-)}. \quad (6.10)$$

6.2 Implementation of the CDEKF

It is very complicated to do the identification of the complete blimp in one step. Firstly, because this will result is very large matrices and very long computational time, with the risk that the matrix $P(t)$ will be ill-conditioned. Secondly it is difficult to measure all states at once. So first it is assumed that the blimp flies in a vertical 2D plane along the longitudinal axis of the blimp. In this plane three degrees of freedom are present, two translations $x$ and $z$ and one rotation $\theta$. The corresponding equations of motion of this subsystem (called subsystem I from now on) contain ten parameters as is shown in Section 6.2.1. In Section 6.2.2 subsystem II is given, which is used to estimate the remaining parameters. For subsystem II a second measurement is carried out where the blimp flies in a 2D plane in the lateral direction of the blimp, so the states $y$, $z$ and $\phi$ and their derivatives can be measured.
6.2 Implementation of the CDEKF

6.2.1 Identification subsystem

The equations of motion of subsystem I, containing the states $x$, $z$, $\theta$, $\dot{x}$, $\dot{z}$, $\dot{\theta}$ are given as (see Chapter 3)

\[
\begin{align*}
\dot{x} &= \cos(\theta) v_x - \sin(\theta) v_z, \\
\dot{z} &= \sin(\theta) v_x + \cos(\theta) v_z, \\
\dot{\theta} &= \omega_y, \\
\dot{v}_x &= \frac{1}{m'_x} (-m'_z v_z \omega_y - (F_g - F_b) \sin(\theta) - (D_{v_x} + D_{v_x^2} |v_x|) v_x + u_1), \\
\dot{v}_z &= \frac{1}{m'_z} (m'_x v_x \omega_y + (F_g - F_b) \cos(\theta) - (D_{v_z} + D_{v_z^2} |v_z|) v_z + u_3), \\
\dot{\omega}_y &= \frac{1}{I'_y} ((m'_x - m'_z) v_x v_z - b_z F_b \sin(\theta) - (D_{\omega_y} + D_{\omega_y^2} |\omega_y|) \omega_y).
\end{align*}
\]

As can be seen subsystem I contains 12 parameters. It is not necessary to estimate all parameters, because some parameters can be measured directly, or can be simply derived. It is preferred to find a parameter in such a way, because it simplifies the estimation process for the remaining parameters. For example the gravity force $F_g$ and buoyancy force $F_b$ can be derived directly, because the volume of the blimp is known, together with the air density. Also the distance between the center of buoyancy and center of gravity $b_z$ is measured directly, so 9 parameters remain to be estimated.

During the first estimation attempts it appeared that nine parameters are too much to find a solution for the estimation problem, so it is decided to simplify the friction model by removing the second order friction terms $D_{v_x^2}$, $D_{v_z^2}$ and $D_{\omega_y^2}$. The following parameters remain to be estimated

\[
p_I = [m'_x, m'_z, I'_y, D_{v_x}, D_{v_z}, D_{\omega_y}].
\]

The state $x(t)$ of the filter in (6.1) containing the states of the model (6.11) and the vector $p_I$ with the parameters can now be constructed as

\[
x(t) = [x, z, \theta, \dot{x}, \dot{z}, \dot{\theta}, p_I]^T.
\]
6.2.2 Identification subsystem II

The equations of motion for subsystem II, which describe motion in the vertical 2D plane along the lateral direction of the blimp, are given as (see Chapter 3)

\[
\begin{align*}
\dot{y} &= \cos(\phi) v_y + \sin(\phi) v_z, \\
\dot{z} &= -\sin(\phi) v_y + \cos(\phi) v_z, \\
\dot{\phi} &= \omega_x, \\
\dot{v}_y &= \frac{1}{m_y^I} \left( m_y^I v_y \omega_x + (F_g - F_b) \sin(\phi) - (D_v y + D_v^2 |v_y|) v_y + u_2 \right), \\
\dot{v}_z &= \frac{1}{m_z^I} \left( -m_z^I v_y \omega_x + (F_g - F_b) \cos(\phi) - (D_v z + D_v^2 |v_z|) v_z + u_3 \right), \\
\dot{\omega}_x &= \frac{1}{I_x^I} \left( (m_y^I - m_z^I) v_y v_z - b_z F_b \sin(\phi) - (D_v \omega_x + D_v^2 |\omega_x|) \omega_x \right).
\end{align*}
\] (6.14)

Subsystem II also contains 12 parameters. Again, \(F_g, F_b\) and \(b_z\) can be measured directly and the second order friction terms \(D_v^2\) are assumed to be zero in order to reduce the complexity of the identification. So the remaining six parameters are estimated

\[
p_{II} = [m_y^I, m_z^I, I_x^I, D_v y, D_v z, D_v \omega_x].
\] (6.15)

It should be noted that \(m_z^I\) and \(D_v z\) are also estimated using subsystem I in Section 6.2.1. They can be left out in the identification of subsystem II (by using the identification results from subsystem I), but using them gives a possibility to verify the results from the CDEKF.

The state \(x(t)\) of the filter in (6.1) containing the states of the model (6.14) and the vector \(p_{II}\) with the parameters can now be constructed for subsystem II as

\[
x(t) = [y, z, \dot{y}, \dot{z}, \dot{\phi}, p_{II}]^T.
\] (6.16)

6.3 Measurements for the CDEKF

Measurements are needed for the CDEKF. Six states have to be measured for subsystem I (see Section 6.2.1), two translations, one rotation and their corresponding velocities and another six states for subsystem II (see Section 6.2.2). It is not necessary to measure the twelve states, for both subsystem I and II, at the same time, because each subsystem is treated separately, but the six states for each subsystem do have to be measured at once.

It is not possible to use the positioning system of the blimp for the identification measurements, because it gives only three translations and one rotation \(\psi\) about the vertical \(z\)-axis. The rotation \(\phi\) and \(\theta\), about the \(x\)-axis and \(y\)-axis respectively, are not measured with the positioning system, because they are not needed to control the blimp. But these rotations are needed for the identification process,
so in order to take measurements for the CDEKF another measurement method is adopted.

For the identification of one subsystem the blimp only has to move in a 2D plane. The position and orientation of the blimp in that plane can easily be measured using a webcam. Two red LEDs are attached to the blimp, one at front and one at the tail, exactly at the middle line of the blimp. The position of these LEDs is recorded with a webcam and the camera images are processed using a Matlab algorithm in order to determine the position and orientation of the blimp. An example of a webcam image being processed is depicted in Figure 6.1. The images are almost completely black, because the brightness is turned down. This way only the red dots of the LEDs are recorded (and also some TL light as can be seen in the sample image in Figure 6.1). The red dots are the recorded position of the LEDs and the green cross indicates the position found by the Matlab algorithm. The measurements that were taken this way for subsystem I and II are discussed in Section 6.3.1 and 6.3.2 respectively.

![Webcam image](image1.png) ![Processed webcam image](image2.png)

Figure 6.1: Webcam image with the two recorded red LEDs, together with their determined position represented by the green crosses.

### 6.3.1 Measurements for subsystem I

For subsystem I translations in both x and z direction and rotation \( \theta \) about the y-axis have to be measured. The results from a measurement are depicted in Figure 6.2. For this measurement no actuation was applied, but the blimp was given an initial orientation and velocity. In Figure 6.2 also the filtered measurement data is depicted, which is obtained from the estimation algorithm of the CDEKF.

### 6.3.2 Measurements for subsystem II

For subsystem II translations in the y and z direction have to be measured and the rotation \( \phi \) about the x-axis. The result from a measurement where the blimp
Figure 6.2: Measurement result for subsystem $I$.

is given an initial velocity and orientation and again with no actuation applied to the blimp, are depicted in Figure 6.3. Again the estimated values for the position and velocity are also depicted as the filtered signals.
Figure 6.3: Measurement result for subsystem $II$. 

(a) $y$ measured vs $y$ filtered
(b) $z$ measured vs $z$ filtered
(c) $\phi$ measured vs $\phi$ filtered
(d) $\dot{y}$ measured vs $\dot{y}$ filtered
(e) $\dot{z}$ measured vs $\dot{z}$ filtered
(f) $\dot{\phi}$ measured vs $\dot{\phi}$ filtered
6.4 Settings for the CDEKF

The extended Kalman filter from Section 6.1 requires some settings for initialization. The initial estimation of the state $x_0$ has to be specified, together with an initial estimation of the error covariance matrix $P_0$. Also the model covariance matrix $Q$ and measurement covariance matrix $R_k$ have to be chosen.

Because the filter is used off-line, initial estimations of the position and orientation of the blimp can be retrieved from the measurements. The initial estimation for the parameters is based on an estimation of dimension and characteristics of the blimp. The used initial estimation of the state $x_0$, for both subsystem $I$ and $II$ is given in Table 6.1.

<table>
<thead>
<tr>
<th>Subsystem $I$</th>
<th>State</th>
<th>Initial estimation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x$</td>
<td>0.9746</td>
<td></td>
</tr>
<tr>
<td>$z$</td>
<td>1.4605</td>
<td></td>
</tr>
<tr>
<td>$\theta$</td>
<td>0.5229</td>
<td></td>
</tr>
<tr>
<td>$\dot{x}$</td>
<td>0.0377</td>
<td></td>
</tr>
<tr>
<td>$\dot{z}$</td>
<td>0.0024</td>
<td></td>
</tr>
<tr>
<td>$\dot{\theta}$</td>
<td>-0.0645</td>
<td></td>
</tr>
<tr>
<td>$m_x'$</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>$m_z'$</td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>$I_y'$</td>
<td>0.15</td>
<td></td>
</tr>
<tr>
<td>$D_{v_x}$</td>
<td>0.19</td>
<td></td>
</tr>
<tr>
<td>$D_{v_z}$</td>
<td>0.78</td>
<td></td>
</tr>
<tr>
<td>$D_{\omega_y}$</td>
<td>0.3</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Subsystem $II$</th>
<th>State</th>
<th>Initial estimation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y$</td>
<td>1.0704</td>
<td></td>
</tr>
<tr>
<td>$z$</td>
<td>1.2032</td>
<td></td>
</tr>
<tr>
<td>$\phi$</td>
<td>-0.3268</td>
<td></td>
</tr>
<tr>
<td>$\dot{y}$</td>
<td>0.0169</td>
<td></td>
</tr>
<tr>
<td>$\dot{z}$</td>
<td>0.0769</td>
<td></td>
</tr>
<tr>
<td>$\dot{\phi}$</td>
<td>-0.1558</td>
<td></td>
</tr>
<tr>
<td>$m_y'$</td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>$m_z'$</td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>$I_x'$</td>
<td>0.15</td>
<td></td>
</tr>
<tr>
<td>$D_{v_y}$</td>
<td>0.78</td>
<td></td>
</tr>
<tr>
<td>$D_{v_z}$</td>
<td>0.78</td>
<td></td>
</tr>
<tr>
<td>$D_{\omega_x}$</td>
<td>0.016</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.1: Initial estimation of the states and parameters.

The initial estimation of the error covariance matrix $P_0$, indicating the uncertainty of the initial estimation of the state $x_0$, can be divided into three parts. The variance in the position measurement of the blimp $P_{0_q}$, the uncertainty in the velocity measurement $P_{0_\dot{q}}$ and the expected uncertainty of the initial parameter values $P_{0_p}$. The position of the blimp is measured using a webcam with a resolution of approximately 0.01 m per pixel, so the uncertainty of the initial position is assumed to be 0.01 m, giving $P_{0_q} = 0.01I_{1\times3}$. The variance of the initial estimation of the velocity of the blimp is set to be 0.1 m/s, so $P_{0_{\dot{q}}} = 0.1I_{1\times3}$. The initial value of the parameters is very inaccurate, so an uncertainty of 25% is supposed, giving $P_{0_p} = 0.25I_{6\times6}$. The used error covariance matrix $P_0$ now is

$$P_0 = diag \left( \begin{bmatrix} P_{0_q} & P_{0_\dot{q}} & P_{0_p} \end{bmatrix} \right).$$

The model covariance matrix $Q$ defines the level of model uncertainty. There are no uncertainties in the first three equations of (6.11) and (6.14), so their corresponding elements in $Q$ are set to be zero. The last three equations of (6.11) and (6.14) do
have some modeling errors, so their corresponding elements in $Q$ are chosen to be 0.01. Because only average parameter values are required, zero parameter noise is selected. The covariance matrix $Q$ now equals

$$Q = 0.01 \text{diag}([0_{1 \times 3} \ 1_{1 \times 3} \ 0_{1 \times 6}]).$$

Finally the measurement covariance matrix $R_k$, defining the amount of measurement noise, has to be selected. As stated before the position is measured using a webcam with a resolution of 0.01 m per pixel, so $R_{k_q} = 0.01I_{1 \times 3}$ is selected for the position measurement noise. The amount of measurement noise in the derivative of the position is set to be 0.1 m/s, so $R_{k_{\dot{q}}} = 0.1I_{1 \times 3}$.

All required settings for the CDEKF algorithm are selected and now the CDEKF can be used for the estimation of the states and parameters. The results of the identification procedure are discussed next.

### 6.5 Identification Results

The CDEKF algorithm, as discussed in Section 6.1, is implemented in Matlab for both subsystem $I$ and $II$, as described in Section 6.2.1 and 6.2.2 respectively, in order to find the parameters of the blimp. To be able to get reliable identification results, a relatively large amount of data is needed from measurements. The measurement results from Section 6.3 are for this reason not very good, because the measured time is relatively short, about 16 seconds. Because it was very difficult to do a measurement that gave useful results, only one data set per subsystem was available. This data set is used iteratively for about 35 times, because one data set appeared to be too short in order to get convergence of the parameters. The results of the runs of the CDEKF for subsystem $I$ and $II$, with the settings from Section 6.4 are depicted in Figures 6.4 and 6.5. The final estimation of the parameters is given in Table 6.2.

Figures 6.4 and 6.5 show all parameter eventually converge to a solution. The peaks visible in the graphs are resulting from the fact that the CDEKF is used iteratively and for every iteration step the position information is lost, resulting in an increasing position error.

Figure 6.4(c) and 6.5(c) show the identification results from both subsystem $I$ and $II$. As can be seen the final parameter value are almost the same, so it can be concluded that the correct parameter values have been found, at least for $m'_z$ and $D_{\omega_z}$.

Finally note that the values for $I'_z$ and $D_{\omega_z}$ in Table 6.2 are not the result from the performed identification, but from the assumption that $I'_z \approx I'_y$ and $D_{\omega_z} \approx D_{\omega_y}$. It is also possible to estimate $I'_z$ and $D_{\omega_z}$ by doing a third experiment and identification, but this would probably lead to the same results.
Figure 6.4: Estimated mass and inertia parameters.
6.5 Identification Results

Figure 6.5: Estimated friction parameters.
6 Parameter Identification

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Estimated value</th>
<th>SI unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m'_x$</td>
<td>0.624</td>
<td>Kg</td>
</tr>
<tr>
<td>$m'_y$</td>
<td>0.805</td>
<td>Kg</td>
</tr>
<tr>
<td>$m'_z$</td>
<td>0.740</td>
<td>Kg</td>
</tr>
<tr>
<td>$I'_{x}$</td>
<td>0.092</td>
<td>Kgm$^2$</td>
</tr>
<tr>
<td>$I'_{y}$</td>
<td>0.275</td>
<td>Kgm$^2$</td>
</tr>
<tr>
<td>$I'_{z}$</td>
<td>0.275*</td>
<td>Kgm$^2$</td>
</tr>
<tr>
<td>$D_{v_x}$</td>
<td>0.184</td>
<td>Ns/m</td>
</tr>
<tr>
<td>$D_{v_y}$</td>
<td>0.472</td>
<td>Ns/m</td>
</tr>
<tr>
<td>$D_{v_z}$</td>
<td>0.476</td>
<td>Ns/m</td>
</tr>
<tr>
<td>$D_{\omega_x}$</td>
<td>0.0066</td>
<td>Ns/m</td>
</tr>
<tr>
<td>$D_{\omega_y}$</td>
<td>0.0280</td>
<td>Ns/m</td>
</tr>
<tr>
<td>$D_{\omega_z}$</td>
<td>0.0280*</td>
<td>Ns/m</td>
</tr>
</tbody>
</table>

Table 6.2: Estimated parameter values (*not estimated, but from the assumption $I'_z \approx I'_y$ and $D_{\omega_z} \approx D_{\omega_y}$).

6.6 Discussion

In this chapter the parameters of the blimp have been identified using the CDEKF. The results from the previous section indicate the CDEKF worked well in order to find the parameters, but some questions arise with respect to these results and also some comments have to be made in order to improve the results.

It is hard to determine whether the estimated parameter correspond with the real values of the blimp, because the CDEKF algorithm could have found a local optima instead of the global optimum. So although the results from the CDEKF converged to a value that appears to be feasible, earlier runs with other settings also gave stable results, but those results were in most cases not possible (resulting in a negative mass for example). So despite that the presented results appear to be good, it is unknown how reliable they are. Although it has to be mentioned that the real parameter values will not differ very much from the presented results, because the estimated parameters are comparable with estimations done for a larger model blimp in [6].

There are many different ways to improve the identification procedure and obtain a more reliable estimation of the parameters. The main improvement that can be made is by performing better measurements for the CDEKF. The measurements can be improved in several ways. The measurement time of one run can be made longer, so instead of only measuring for about 16 seconds, it is better to do a measurement which lasts for example 100 seconds. This way more information is gathered for the CDEKF. Another way to introduce more variety in the gathered data is by using an input for the thrusters during a measurement. For example the thrusters can be fed with band limited white noise, or a block function. Also, the presented identification uses one short measurement iteratively, however it is better
to use results from multiple measurements, because they contain more information for the CDEKF, which improves the reliability of the identification results. Finally another improvement that can be made is taking measurements of all states at the same time, instead of dividing the states over two subsystems and measure them separately. This creates the possibility of estimating all parameters in one run and maybe even some more parameters that were not taken into account in the presented identification.
Chapter 7

Experimental Scheme

No experiments could be carried out for this thesis. However it is important to validate the theoretical results, not only by means of simulations, but also by performing experiments. This chapter describes several experiments, in line with the theoretical work in this thesis, which can be used as a guideline to validate the presented work in an experimental framework. So it should provide a future DevLab student a running start.

To improve the parameter identification, better measurements are needed as discussed in Chapter 6.6. Improved measurement can be taken by carrying out the experiment as described in Section 7.2, where also the implementation of the CDEKF algorithm is discussed. It is advised to start with this experiment, because it should deliver a more reliable estimation of the parameters, which successively can be used for the tracking experiment.

A better estimation of the parameters should improve the performance of the tracking controller when implemented on the real blimp. How this tracking experiment should be carried out is described in Section 7.3. The final goal of the tracking experiment is to investigate whether the designed tracking controller is working well on the real blimp, but also it is possible to verify the dynamic model of the blimp by comparing the experimental results with the results from the tracking simulations.

The third experiment, described in Section 7.4, is a synchronization experiment, where a formation flight between two blimps is performed. If the developed synchronization controller is working well for the real blimps and the formation flight satisfies all requirements, the final answer to the main question is found, because it is then known what is needed to make a formation flight with two blimps.

Before the described experiments can be carried out, some hardware requirements should be satisfied. These hardware requirements are given first in Section 7.1, together with a brief description of the current blimps.
7.1 Hardware

At this moment two blimps are available at DevLab. The envelope of the current blimps is made out of vinyl, is about 1.67 m long and contains about 430 liters of helium. The envelope itself weighs 229 grams, so the maximum amount of payload that is can carry is about 200 grams (one liter of helium produces around 1 gram of lift). Note this only holds if the blimp is filled with pure helium, however the used helium contains a small amount of other (heavier) gasses, so when the blimp is just filled the produced lift will be slightly less. Also the amount of lift decreases rapidly over time, because the blimp deflates as helium diffuses through the envelope. When the blimp is now refilled the generated lift will be less compared to the situation where the blimp is filled for the first time. This problem arises, because only the helium can diffuse though the envelope and not the heavier gasses, so every time the blimp is refilled the amount of pollution will increase, resulting in a decreased lift capacity. Because of this it is advised to keep the weight of the gondola below 120 grams in order to have some margin.

The current gondola is depicted in Figure 7.1. This version of the gondola only contains three thrusters, two forward and one vertical thruster. However the final gondola should also contain a fourth propeller which could produce thrust in the lateral ($y$) direction. This forth thruster is in most cases placed in the tail fin, because it is used to change the heading of the blimp. However for this blimp the two forward thrusters are used to rotate the blimp, so a tail rotor is not needed here. The fourth propeller is therefore used to generate thrust in the lateral direction, so the blimp can also move sideways. It should be attached to the gondola as close to the center of gravity as possible to minimize the undesired torque about the vertical axis. It is also advisable to put both forward thruster further away from each other in comparison with the current situation as depicted in Figure 7.1. This way it is easier for the blimp to change its heading. The exact location of the four thruster is depicted in Figure 3.2.

During all experiments the blimp has to fly at a fixed height. This is already realized at the moment of writing, an ultrasonic ranger measures the distance to the ground and any deviation with respect to the selected height is compensated by using the vertical thruster. A PD controller is used to control this vertical thruster.

For all experiments the $x$ and $y$ position of the blimp should be measured, together with its heading $\psi$. A positioning system, working with ultrasonic beacons and an ultrasonic receiver on the blimp is used at this moment to measure the position and a compass is used to measure the heading. It is also possible to use the ultrasonic beacons to measure the heading, however this method is more complex to implement, but is eventually a better option, because the compass reacts relatively slow. According to the specifications the position should be measured with an accuracy of 0.1 m. However in practice it appears that this accuracy will not be met with the current positioning system. This should not be a problem, because the developed controllers can deal with a much lower accuracy, as is shown by simulations. The
positioning system should work with a minimum sample frequency of 6 Hz, as it appears to be sufficient in order to satisfy the tracking requirements, according to simulations presented in Appendix C. However if possible, it is preferable to use a sample frequency of 7 Hz or higher, because simulations show the tracking error decreases if the sample rate is increased. More information about the positioning system can be found in the thesis of Rolf van de Burgt.

For this thesis it is assumed that the thrusters generate a force in Newton. In order to be able to implement the controllers on the real blimp the relation between the input, an integer number and the output, a force in Newton should be taken into account. This relation is almost linear, except at very low inputs, so a linear relation can be assumed at first.

The final requirement for the hardware of the blimp is that it should be possible to use data logging on an external computer in order to be able to compare the experimental results with the results from simulations. The position and heading of each blimp should be logged, together with the inputs for the thrusters.

### 7.2 Parameter Identification

The parameter identification procedure as discussed in Chapter requires experimental data in order to find good estimates of the different parameters of the blimp. It is important to have good estimates, especially of the parameters used for the
computed torque controller described in Chapter 4, because the effectiveness of this controller depends partly on how well the estimated parameters represent the actual parameters of the blimp. In Chapter 6, some experiments have already been carried out, by using a webcam, but as mentioned in Chapter 6.6 these experiments are of limited use in order to find reliable estimates of the parameters. Therefore it is preferable to do some more experiments, which can be used to find better estimates of the parameters. This section describes which experiments should be carried out and how the experimental data can be processed to get better results from the identification procedure.

It is not necessary to follow the estimation process as described in Chapter 6, because it is not required to estimate all parameters again. The current estimation of the parameters of the dynamic model, which are not used for the computed torque controller, are good enough to be able to carry out simulations with the dynamic model, so they do not have to be estimated again. It is only preferable to find better, more reliable, estimations of the parameters which are used for the computed torque controller, because they are not only important for the simulations, but also for the tracking and synchronization experiments. Therefore it is advisable to complete the identification procedure with better experiment before doing the tracking and synchronization experiments.

### 7.2.1 Experiments for Parameter Identification

Because it is not necessary to estimate all parameters again, it is also not required to measure all states of the blimp, which was done in the first estimation attempt. Only the states which can be measured with the positioning system of the blimp are required, so only the $x$ and $y$ position and the heading $\psi$ of the blimp and their derivatives should be measured.

In order to improve the measurements for the identification procedure, the new experiments should differ from the previous measurements from Chapter 6.3. Not only the positioning system of the blimps will be used this time, instead of the less accurate measurements with the webcam, but also (multiple) longer runs should be carried out. To retrieve more information from each run it is advisable to have an input for both sideboard thrusters $F_{x,s}$, $F_{x,p}$ and also for the lateral thruster $F_y$. As input can be used band limited white noise or a block function for example. One run should be as long as possible, but note that two good short runs is better than one bad long run. So for example, if it is not possible to perform a long run without hitting a wall, it is better to use a shorter run. The amount of runs that is necessary depends on the convergence of the CDEKF algorithm as given in the following section. Also it is possible to use one run iteratively as is done in Chapter 6.5, but using different runs should give a better result.
7 Experimental Scheme

7.2.2 Data Processing

The data can be processed the same way as is done in Chapter 6, so by using the CDEKF algorithm. It is not necessary to use subsystems, because all required states can be measured at the same time and also the resulting equations of motion are not very complicated. For the initialization of the CDEKF some settings are required as described in Chapter 6.4. For the initial estimation of the parameters the results from the previous parameter identification can be used as given in Chapter 6.5. The initial estimation of the error covariance matrix $P_0$ depends on the variance of the position and velocity measurement and the uncertainty in the initial estimation of the parameters. If the position measurement meets its requirements with respect to accuracy the same $P_0$ can be used as used in Chapter 6.4. Because the amount of model uncertainty is not changed also the model covariance matrix $Q$ can be selected equally as in Chapter 6.4. Finally the covariance matrix $R_k$ can be chosen depending on the amount of measurement noise. If the positioning system meets its requirement the same covariance matrix can be used as used before. Note that the suggested initial settings can be changed freely and do not have to be used as given. It is difficult to select the best settings beforehand and they probably have to be fine tuned, which may result in a better or faster convergence.

The Matlab files with the CDEKF algorithms are given in Appendix B. After measurements have been taken the given files can be used directly. Note that the CDEKF requires a lot of computational time, also depending on the amount of measurement data.

7.3 Tracking Experiment

To validate the simulations results from the tracking simulation in Chapter 4 an experiment is required. For the tracking experiment one blimp should follow the desired trajectory described in Chapter 2.2.2. The computed torque controller from Chapter 4.2 should be implemented, together with the tracking controller from Chapter 4.3. The use of an observer as described in Chapter 4.5.1 is advised in order to obtain velocity information for the controller. Note that the velocity can also be calculated by:

$$\dot{q}_k \approx \frac{q_k - q_{k-1}}{T_s},$$  \hspace{1cm} (7.1)

where $\dot{q}_k$ is the derivative of $q_k$ at sample $k = 1, 2, \ldots$ and $T_s$ is the sample time. This method, or a more extensive equivalent using more samples, requires less computational time than the velocity observer, but the velocity observer produces better results with less lag as is shown in Chapter 4.5.2.
7.3.1 Controller Settings

The computed torque controller requires parameters of the model. The parameters presented in this thesis (Chapter 6.5) can be used, but it is preferred to use the parameters from the new identification, as described in Chapter 7.2.1, at least if the new identification gives good results.

The initial gains for the tracking controller can be set as $K_p = (0.3, 0.3, 0.3)I_{3 \times 3}$ and $K_p = (0.5, 0.5, 0.5)I_{3 \times 3}$, equal to the gains used for the tracking simulation. Also the observer gains $L_p$ and $L_d$ can be set equal to the gains used for the simulation: $L_p = [1 1 2]I_{3 \times 3}$, $L_d = [0.3 0.3 0.6]I_{3 \times 3}$. Note that some fine tuning is probably needed to tune the gains for both the tracking controller and also the observer.

7.3.2 Comparing the Tracking Simulations with the Experiments

In order to be able to compare the tracking simulation with the tracking experiment the same gains for the controllers should be used. So if for the experiment some gains have been changed, because they gave better results, a new simulation should be carried out which uses the same gains.

For the first tracking experiment it is not advisable to implement the $H_\infty$ compensation method as described in Chapter 4.4, because it makes the overall controller rather complex. However if the tracking controller performs well, or if it does not work properly according to the specification even though it is implemented correctly, it has to be investigated if the implementation of the $H_\infty$ compensation method improves the tracking performance.

If the tracking experiment described above is working well, it could be interesting to do some more experiments to investigate the effect of different extra disturbances on the blimp. For example some wind could be added, by using a fan, or some parameters for the computed torque controller can be changed. Because the same disturbances can be added to the simulation model, the results of the experiments could be compared again with the simulations results.

7.4 Synchronization Experiment

The experiment which should complete the answer to the question: What is needed to make a formation flight with two or more "Atalantas"?, is the synchronization experiment. For this experiment two equal blimps are needed. At least the hardware should be equal, but there is a small difference in the software depending on the synchronization strategy which is chosen. Two synchronization strategies are discussed in Chapter 5: a master-slave synchronization strategy and a mutual synchronization strategy. Using simulations it is shown in Chapter 5.4.2 that the mutual synchronization strategy performed best, so it is preferred to implement that strategy on the real blimps. However the mutual synchronization strategy is
a little bit more difficult to implement, because more information has to be exchanged between the two blimps. Both blimps should exchange their position and velocity information, where only the slave should receive this information from the master when using the master-slave strategy. Also when using the mutual synchronization strategy both blimps have to know the desired trajectory, where in case of master-slave synchronization only the master should know the desired trajectory. The implementation of both synchronization strategies is described below, however it is preferred to implement the mutual synchronization strategy, because it performed best during simulations, as discussed in Chapter 5.4.2. Nevertheless it can be interesting to investigate if the mutual synchronization also performs best during a real experiment, by comparing it with a master-slave synchronization experiment.

7.4.1 Master-Slave Synchronization Experiment

For the master blimp, the blimp used for the tracking experiment can be used without any modification, because for the master blimp exactly the same controller is used (see Chapter 5.2) as used for the tracking experiment. The controller for the slave is given in Chapter 5.2 and it differs from the controller for the master. The slave needs position and velocity information of the master and it should measure its own position and velocity. Furthermore it is advised to implement the velocity observer, discussed in Chapter 5.4.1 in order to obtain the velocity information and filter the position signals. The same observer can be used, as used for the tracking experiment.

The gains for the synchronization controller of the slave blimp can be set equal to the gains used for the synchronization experiment, so $K_{p,1} = 0.3I_{3\times3}$, $K_{d,1} = 1.5I_{3\times3}$. For the master the same gains can be used as used for the tracking experiment.

7.4.2 Mutual Synchronization Experiment

For the mutual synchronization experiment the controllers for both blimps are equal and given in Chapter 5.3. Blimp I differs from the master blimp from the previous experiment in that it has to know the position of blimp II and it reacts in case blimp II can not keep up with blimp I. Blimp II differs from the slave blimp in the previous experiment, because this time it also knows its desired trajectory. The gains for the controller $\nu_1$ (5.12) of blimp I, $K_{p,1}$, $K_{d,1}$, $K_{cp1,2}$, $K_{cv1,2}$ can be set equal to the gains used for the simulation, $K_{p,1} = 0.3I$, $K_{d,1} = 0.5I$, $K_{cp1,2} = I$, $K_{cv1,2} = I$. The gains for the controller $\nu_2$ (5.13) of blimp II can be chosen to be $K_{p,2} = 0.5I$, $K_{d,2} = 1.5I$, $K_{cp2,1} = I$, $K_{cv2,1} = I$, equally to the gains used for the simulation. For these gains the same holds as for the gains of the tracking controller, probably some fine-tuning is needed, because there is always a difference between a simulation and an experiment.

If the mutual synchronization experiment performs well and if the experimental
7.4 Synchronization Experiment

results are comparable with the simulation results from Chapter 5.4.2 the answer to the main question is found, because it is known what is needed to do a formation flight with two Atalanta’s, not only in theory, but also in practice. Also if the simulation model can be used to describe the actual blimp it could be used to investigate a formation flight between many Atalanta’s.
Chapter 8

Conclusions and Recommendations

The Atalanta Wingman Project is set up to find an answer for the question "What is needed to make a formation flight with two or more Atalanta’s?", in order to get a step further with DevLabs long term Atalanta Project, which aims for the development of an autonomous mechatronic butterfly, the Atalanta. This thesis mainly gives a theoretical answer to that question, or at least it tries to give an answer to a part of that question. To give a full answer to the question, a demonstrator, consisting of a control module and a flying platform, has to be build which is capable of making an autonomous formation flight. This thesis describes the theory with respect to the mechanical aspects behind the control module and the choices which had to be made. The actual control module, consisting of electrical hardware and software, is mainly developed by Rolf van de Burgt and will therefore be described in his thesis.

Different goals have been set for this project. For the first goal, the investigation for a suitable experimental platform, different options have been considered, but the blimp appears to be the most suitable platform. A slow moving blimp is relatively easy to control and can be safely used indoors. Also its power requirements are low, while it can still generate enough lift to carry the control module.

In order to study the behavior of a blimp a mathematical model of the blimp is made which consists of the dynamic equations of motion. The model describes the effect of mass and inertia including added mass effects, because a blimp is a buoyant vehicle. Also the coriolis effect is included in the model, which gives a fictitious force, resulting from the fact that the blimp has a non-inertial reference frame. Furthermore the effects of gravity and buoyancy is described, together with the aerodynamic friction acting on the blimp. Finally the effect of the thrusters on the movement of the blimp is modeled. The model is made to be able to perform simulations, but it is also used to design and evaluate different control strategies.

A tracking controller has been developed, because the blimp should be able to fly a predefined trajectory while it also has to deal with small disturbances. This
Tracking controller consists of a nonlinear part and a linear part. With the implementation of a nonlinear computed torque controller, based on a simplification of the inverse dynamics of the blimp, the original nonlinear system is made linear. The resulting linear system is further controlled using a PD controller. The performance of this closed loop system is affected by disturbances like measurement noise, but also uncertainties from modeling errors, unmodeled dynamics and uncertainties in the parameters degrade the performance of the tracking controller. To compensate for those uncertainties a $H_\infty$ compensation algorithm is added to the tracking controller. To finalize the design of the tracking controller a velocity observer is implemented. This velocity observer estimates the velocity and position of the blimp, required for the tracking controller, based on only the position information from the positioning system of the blimp. Simulations have been performed to show the effect of the different steps in the design of the tracking controller and they give an insight in how the tracking controller will perform when implemented on the real blimp. According to those simulations a velocity observer is required for the actual blimp, while the $H_\infty$ compensation method can be added to gain some more accuracy.

To be able to make a formation flight with two blimps a synchronization controller is developed. Two synchronization strategies are discussed, both based on the same general synchronization controller. This general controller is based on the nonlinear computed torque controller, while a linear controller ensures synchronization between the blimps. Master-slave synchronization is studied first, where the master blimp follows its desired trajectory without paying any attention to the slave, while the slave has to follow the master. Secondly, mutual synchronization between the blimps is investigated, where essentially both blimps are equal and together try to stay in formation while following their desired trajectory. The velocity observer is implemented for both synchronization strategies, because velocity information is needed again, while only the position is measured. Simulations have shown both synchronization controllers work well, but the mutual synchronization controller appears to be the best option for the actual blimps.

The dynamic model of the blimp contains many physical parameters. It is important to know these parameters in order to be able to simulate the actual blimp, but even more important, some of the parameters are used for the computed torque controller. If the values of these parameters are estimated incorrectly this will result in a performance loss for both the tracking controller and the synchronization controller, when implemented on the real blimp. Because most of the parameters can not be measured directly, they have been estimated using a Continuous-Discrete Extended Kalman Filter, which uses measurement data from a webcam to estimate the parameters. Although an estimation of the parameters has been found, the estimation is not very reliable. Better measurements are needed to improve the identification results.
8.1 Future Work

This thesis alone does not answer the main question, not only because also the thesis of Rolf van de Burgt is needed, but also because no experiments have been carried out, which validate the presented theoretical work. So the main recommendation for future work would be to carry out those experiments. In order to give a future student at DevLab a running start, an experimental scheme is presented in this thesis. In the final chapter several experiments are described, in line with the presented theoretical work, which can be used as a guideline to validate the presented work in an experimental framework. To be able to carry out those experiments, some hardware requirements have to be met, which are also given. It is recommended to start with an experiment which can be used to improve the identification of the parameters. Secondly a tracking experiment is described which can be used to validate the tracking controller, with and without the implementation of the $H_\infty$ compensation method. Finally a synchronization experiment is described, where the presented synchronization controller is used to make a formation flight with two blimps. This experiment should give a final answer to the main question.

If the presented synchronization strategy appears to work well in practice, this does not mean that the final Atalanta should make use of the same strategy. There are other synchronization strategies that can be used, of which some of them may be a better option for the final Atalanta which should be able to fly in a large group of other Atalanta’s. For example a synchronization strategy based on swarm behavior may be an interesting option. So if a formation flight between a large group of Atalanta’s is considered, more research is needed to find out if the current synchronization strategy is still suitable, or if other synchronization strategies appear to be a better option.

Because the main question of the Atalanta Wingman Project is very broad, the goal is set to perform a formation flight at a fixed height without any obstacles. An interesting addition to this goal is to be able to fly at a varying height, so obstacles can be avoided for example. It is relatively easy to control the height by extending the computed torque controller. However obstacle detection is more complex, because it probably requires additional sensing and extra computational power, which results in an unwanted weight increase of the gondola. Because the final Atalanta may only weigh five grams, the current weight of the gondola, around 100 grams, should be decreased dramatically to operate in the same region. The current gondola is already designed to have a low weight, so an interesting question would be what is possible to decrease the weight of the blimp significantly?

During the design of the current controllers, the power efficiency of the controllers is not taken into account and they are especially designed to have a high accuracy. However the final Atalanta’s have a very limit power supply, so their controllers should be very power efficient. Therefore the final recommendation for future work is to design controllers for the blimp which are optimized with respect to energy usage and computational requirements.
Bibliography


Appendix A

Control Strategies

A.1 Simplified Dynamic Model

The full expressions of the different components in the simplified dynamic model of the blimp as discussed in Chapter 4.1, is given below:

\[
M(q(t),\dot{p}) = \begin{bmatrix}
m_x' \cos(\psi) & -m_x' \sin(\psi) & 0 \\
m_y' \sin(\psi) & m_y' \cos(\psi) & 0 \\
0 & 0 & I_z'
\end{bmatrix},
\]

\[
h(q(t),\dot{q}(t),\dot{p}) = \begin{bmatrix}
(-\sin(\psi) m_x' + m_y') + \cos(\psi) Dv_x - \frac{2 m_x' \cos(\psi) \sin(\psi)^2 D\omega_z}{m_y'} \\
(-\cos(\psi)^3 m_x' - \cos(\psi) \sin(\psi) Dv_x) \\
(m_x' \cos(\psi) m_x' + m_y' \cos(\psi) m_y' + \sin(\psi) - 2 \sin(\psi)^3 Dv_y) \dot{x} + \\
(-\sin(\psi) m_x' + (\sin(\psi)^3 - \sin(\psi)) m_y' + \cos(\psi) Dv_y) \dot{y} \\
-(m_x' - m_y') (\cos(\psi) \sin(\psi) \dot{x}^2 - \sin(\psi) \cos(\psi) \dot{y}^2 + \cos(2\psi) \dot{x} \dot{y}) + D\omega_z \psi
\end{bmatrix},
\]

\[
Q(p) = \begin{bmatrix}
\frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\
\frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \\
\frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\
0 & 1 & 0
\end{bmatrix},
\]

\[
p = \begin{bmatrix}
m_x' \\
m_y' \\
I_z' \\
Dv_x \\
Dv_y \\
D\omega_z \\
r_x \\
r_y
\end{bmatrix}.
\]
The full expression for the dynamic controller $F_p = [F_{x,s} F_{x,p} F_y]^T$ after implementation of the computed torque method as described in Chapter 4.2 are

$$F_{x,s} = -\frac{E_1}{2} - \frac{m_e^2}{2} \left( -\cos(\psi)u_x + \sin(\psi)u_y + \cos(\psi)v_y + \sin(\psi)v_x \right) - \frac{r x m_e^2}{2 y} (F_3 - I_x^r u_y) -$$

$$\frac{r x m_e^2}{2 y} \left( -\frac{F_p}{m_y} + \sin(\psi)u_x + \cos(\psi)u_y - \sin(\psi)v_y + \cos(\psi)v_x \right), \tag{A.1}$$

$$F_{x,p} = -\frac{E_1}{2} - \frac{m_e^2}{2} \left( -\cos(\psi)u_x + \sin(\psi)u_y + \cos(\psi)v_y + \sin(\psi)v_x \right) + \frac{1}{2 y} (F_3 - I_x^r u_y) +$$

$$\frac{r x m_e^2}{2 y} \left( -\frac{F_p}{m_y} + \sin(\psi)u_x + \cos(\psi)u_y - \sin(\psi)v_y + \cos(\psi)v_x \right), \tag{A.2}$$

$$F_y = -F_2 + m_e (\sin(\psi)u_x + \cos(\psi)u_y - \sin(\psi)v_y + \cos(\psi)v_x) \tag{A.3}$$

where

$$F_1 = -m_e v_z \omega_y + m_e v_y \omega_z - (D_{v_x} + D_{v_y} |v_x|) v_x \tag{A.4}$$

$$F_2 = m_e v_z \omega_x - m_e v_x \omega_z - (D_{v_x} + D_{v_y} |v_y|) v_y \tag{A.5}$$

$$F_3 = -(m_e - m_e) v_x v_y - (I_y - I_x^r) \omega_x \omega_y - (D_{\omega_x} + D_{\omega_y} |\omega_x|) \omega_z. \tag{A.6}$$

A.3 State Space Model Dynamic Controller

The full expression for the dynamic controller $K_c$ found by using the $H_\infty$ control technique as described in Chapter 4.4.3 is given by

$$K_c = W_c K_\infty = \begin{bmatrix} A_c & B_c \\ C_c & D_c \end{bmatrix}, \tag{A.7}$$

where

$$A_c = \begin{bmatrix}
-10.61 & 0 & 0 & -0.4254 & -0.9591 & -1.062 & 0.2027 & 0.9078 & -1.354 \\
0 & -10.61 & 0 & 0.2747 & -1.142 & 0.9214 & -0.4056 & 1.349 & 0.8424 \\
0 & 0 & -10.61 & -1.405 & 0.06711 & 0.5019 & 1.579 & 0.2391 & 0.3907 \\
-0.05659 & 0.03654 & -0.1869 & -0.8073 & 0 & 0 & 1.802 & 0.2527 & -0.1932 \\
-0.1276 & -0.152 & 0.008928 & 0 & -0.8073 & 0 & -0.2796 & 1.788 & -0.2688 \\
-0.1413 & 0.1226 & 0.06977 & 0 & 0 & -0.8073 & -0.1517 & -0.2942 & -1.799 \\
0.1994 & -0.399 & 1.553 & -0.05891 & 0.009143 & 0.004959 & -2.563 & 0 & 0 \\
0.8931 & 1.327 & 0.2264 & -0.009264 & -0.05846 & 0.090962 & 0 & -2.563 & 0 \\
-1.332 & 0.8904 & 0.3843 & 0.006317 & 0.008789 & 0.08883 & 0 & 0 & -2.563
\end{bmatrix}, \tag{A.8}$$

$$B_c = \begin{bmatrix}
0 & -0.04619 & -1.759 & 0 & -0.07573 & -2.883 \\
0 & 1.759 & -0.04619 & 0 & 2.883 & -0.07573 \\
0 & 2.883 & 0 & 0 & 2.883 & 0 \\
0.212 & -0.04314 & -0.03631 & -0.07713 & 0.01569 & 0.02296 \\
-0.01013 & 0.1686 & -0.1493 & 0.00685 & -0.06133 & 0.0543 \\
-0.07577 & -0.1433 & -0.1567 & 0.02756 & 0.05211 & 0.05099 \\
-0.1707 & 0.04442 & 0.02076 & -0.1932 & 0.05028 & 0.0235 \\
-0.02488 & -0.1433 & 0.102 & -0.02816 & -0.1622 & 0.1154 \\
-0.04225 & -0.09511 & -0.144 & -0.04782 & -0.1076 & -0.1629
\end{bmatrix}, \tag{A.9}$$

$$C_c = \begin{bmatrix}
0 & 0 & -3.377 & -0.2256 & 0.01078 & 0.08063 & 0.2578 & 0.03758 & 0.06381 \\
0.0887 & -3.377 & 0 & 0.0459 & -0.1794 & 0.1524 & -0.06709 & 0.2164 & 0.1436 \\
3.377 & 0.0887 & 0 & 0.06716 & 0.1588 & 0.1667 & -0.03136 & -0.154 & 0.2174
\end{bmatrix}, \tag{A.10}$$
\[ D_e = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \] (A.11)

Note that the order of the state space model is reduced by removing negligible states.
A.3 State Space Model Dynamic Controller
Appendix B

Matlab Program CDEKF

The Matlab program required to do the parameter identification as described in Chapter 7.2 is given below. This program also uses another Matlab function file which is needed to calculate the propagation of the states between two measurements. This function file is also given.

%%%%%% Continuous-Discrete Extended Kalman Filter %%%%%%
% Jasper van de Loo
% 23-08-2007
% This program can be used to estimate the parameters of the blimp.
% This program has to be used together with the function file
% calc_propagation_CDEKF
%%%% general %
clear all;close all;clc

warning off

global Fk Q N1 N2 rx ry;

% load data %

% Load the measurement data. It should give 7 columns with:
% time, x, y, psi, xdot, ydot, psidot
load t_xypsi_xdotydotpsidot

% Create matrix z containing 6 columns: [x y psi vx vy wz]:
% Relation between xdot ydot psidot and vx vy wz
vx = cos(psi).*xdot-sin(psi).*ydot;
vx = sin(psi).*xdot+cos(psi).*ydot;
wz = psidot;

z = [x y psi vx vy wz]; % states x y psi xdot ydot psidot

tspan = time;

% Load input from the thrusters. It should give 3 columns:
% Fxs, Fxp, Fy:
load FxsFxpFy
F=[Fxs Fxp Fy];

% initialization %
% initial estimation of the parameters:
% eta0=[mxa mya Iza Dvx Dvy Dwz]
eta0 = [0.624 0.805 0.275 0.184 0.472 0.028]';

N1 = 6; % # states
N2 = length(eta0); % # parameters

Pk_min = diag([0.01*ones(1,3) 0.1*ones(1,3) 0.25*eta0']);

Q = 0.01*diag([zeros(1,3) ones(1,3) zeros(1,6)]);

Rk = diag([0.01*ones(1,3) 0.1*ones(1,3) 1e-8*ones(1,6)]);

Hk = [eye(N1) zeros(N1,N2)
      zeros(N2,N1) zeros(N2,N2)];

% Distance thrusters from center of gravity:
% These values may change depending on the final design of the blimp
rx=0.2;
ry=0.1;

%% Kalman filter %
% If the measurements are used iteratively this for loop should be used:
iter=1;% # of iterations
for h=1:1:iter
    Xk_hat = [];
    xk_hat_min = [z(1,:).';eta0 

    tic
    % Show temporary results: plot a figure every 10 iterations
    for k = 1:length(tspan)-1
        if mod(k,10) == 0
            k
toc
            tic
            if mod(k,100) == 0
                figure(1)
                plot(tspan(1:k),z(1:k,:), tspan(2:k),Xk_hat);
                shg
                save temp_results_ident
            end
        end
    zk = [z(k,:).';zeros(N2,1)];
    Fk = [F(k,:).'];
    % Calculate updates
    Kk = Pk_min*Hk.'*inv(Hk*Pk_min*Hk.' + Rk);
    Pk_plus = (eye(size(Pk_min))-Kk*Hk)*Pk_min;
    xk_hat_plus = xk_hat_min + Kk*(zk - xk_hat_min);
    % Calculate propagation
    Pk_plus = Pk_plus.';
    x0 = [xk_hat_plus;
          reshape(Pk_plus,(N1+N2)^2,1)]
    tstart = tspan(k);
tend = tspan(k+1);

[T,X] = ode23s('calc_propagation_CDEKF',[tstart tend], x0);
X = X(end,:);
xk_hat_min = X(1:(N1+N2));
Xk_hat = [Xk_hat xk_hat_min];
Pk_min = reshape(X(N1+N2+1:N1+N2+(N1+N2)^2),N1+N2,N1+N2).
end

%% plot results
% final values parameters:
Xk_hat([7 8 9 10 11 12],end)

% Plot trajectories:
plot(tspan(1:k),z(1:k,:), tspan(2:k+1),Xk_hat([1 2 3 4 5 6],:),'--');
legend(['m_x','y_m','\psi_m','xd_m','yd_m','\omega_x_m','x_f','y_f','\psi_f','xd_f','yd_f','\omega_z_f'])
figure
plot(par); legend('m_x_a','m_y_a','I_z_a','D_{v_x}','D_{v_y}','D_{\omega_z}')
figure
plot(par(:,1),'k'); ylabel('m_x_a')
figure
plot(par(:,2),'k'); ylabel('m_y_a')
figure
plot(par(:,3),'k'); ylabel('I_z_a')
figure
plot(par(:,4),'k'); ylabel('D_{v_x}')
figure
plot(par(:,5),'k'); ylabel('D_{v_y}')
figure
plot(par(:,6),'k'); ylabel('D_{\omega_z}')

% Save identification results
save results_ident_CDEKF

Function file calc_propagation_CDEKF.m:

function dx = calc_propagation_CDEKF(t,xp)
% function file to calculate propagation between measurements
% Jasper van de Loo
% 23-08-2007

global Fk Q N1 N2 rx ry
dx = zeros(N1+N2*(N1+N2)^2,1);

% States
x = xp(1);
y = xp(2);
psi = xp(3);
vy = xp(5);
wz = xp(6);
mx = xp(7);
my = xp(8);
Iza = xp(9);
Dvx = xp(10);
Dvy = xp(11);
Dwz = xp(12);
% Inputs; delivered thrust
Fxs = Fk(1);
Fxp = Fk(2);
Fy = Fk(3);

% Derivatives of the position and velocity of the blimp
dx(1) = cos(psi)*vx+sin(psi)*vy;
dx(2) = -sin(psi)*vx*cos(psi)*vy;
dx(3) = wz;
dx(5) = 1/mxa*(mya*vy*wz-Dvx*vx+Fxs+Fxp);
dx(4) = 1/mya*(-mxa*vx*wz-Dvy*vy+Fy);
dx(6) = 1/Iza*((mxa-nya)*vx*vy-Dwz*wz+Fy*ry+(Fxs-Fxp)*rx);

P = reshape(xp(N1+N2+1:N1+N2+(N1+N2)^2),N1+N2,N1+N2).';

% Partial derivative: d(dx)/d(xp):
F = [0 0 -sin(psi)*vx*cos(psi)*vy cos(psi)*sin(psi) 0 0 0 0 0 0; 0 0 -cos(psi)*vx*sin(psi)*vy -sin(psi)*cos(psi) 0 0 0 0 0 0; 0 0 0 1 0 0 0 0 0; 0, 0, 0, -1/mxa*Dvx, 1/mxa*nya*wz, 1/mxa*nya*vy, -1/mxa^2*(mya*vy*wz-Dvx*vx+Fxs+Fxp),...

...1/mxa*vy*wz, 0, -1/mxa*vx, 0, 0; 0, 0, 0, -1/nya*mxa*wz, -1/nya*Dvy, -1/nya*mxa*vx, -1/nya*vx*wz,...

...-1/nya^2*(-mxa*vx*wz-Dvy*vy+Fy), 0, 0, -1/nya*vy, 0;
0, 0, 0, 1/Iza*(mxa-nya)*vy, 1/Iza*(mxa-nya)*vx, -1/Iza*Dwz, 1/Iza*vy, -1/Iza*vx, y, ...

...-1/Iza^2*((mxa-nya)*vy*vx-Dwz*wz+Fy*ry+(Fxs-Fxp)*rx), 0, 0, -1/Iza*wz];

F = [F;zeros(N2,N1+N2)];
dX = F*P + P'*F + Q;
dX = dX.';
dx(N1+N2+1:N1+N2+(N1+N2)^2) = reshape(dX,(N1+N2)^2,1);
Appendix C

Sample Frequency

To investigate which sample frequency for the position measurement system and controller is needed. A tracking simulation is performed according to the tracking simulation in Chapter 4.3.1. Uniforme random noise with a minimum of −0.1 m and a maximum of 0.1 m has been inserted in the measured signals. Also the velocity observer is used. The position signals including the noise have been discretized with a quantization interval of 0.02 m, based on a 8 bit signal. Also the input for the thrusters is discretized, with a quantization interval of 0.003 N, also based on a 8 bit signal. Table C.1 gives the mean square error over the complete tracking simulation, together with the maximum error, for different sample frequencies.

<table>
<thead>
<tr>
<th>sample frequency</th>
<th>mse(·)</th>
<th>max(·)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x</td>
<td>y</td>
</tr>
<tr>
<td>10 Hz</td>
<td>0.009</td>
<td>0.016</td>
</tr>
<tr>
<td>9 Hz</td>
<td>0.009</td>
<td>0.016</td>
</tr>
<tr>
<td>8 Hz</td>
<td>0.008</td>
<td>0.016</td>
</tr>
<tr>
<td>7 Hz</td>
<td>0.013</td>
<td>0.016</td>
</tr>
<tr>
<td>6 Hz</td>
<td>0.022</td>
<td>0.018</td>
</tr>
<tr>
<td>5 Hz</td>
<td>0.029</td>
<td>0.030</td>
</tr>
<tr>
<td>4 Hz</td>
<td>0.035</td>
<td>0.039</td>
</tr>
<tr>
<td>3 Hz</td>
<td>0.029</td>
<td>0.029</td>
</tr>
<tr>
<td>2 Hz</td>
<td>0.093</td>
<td>0.037</td>
</tr>
<tr>
<td>1 Hz</td>
<td>0.287</td>
<td>0.359</td>
</tr>
</tbody>
</table>

Table C.1: Tracking errors for different sample frequencies.

As can be seen in Table C.1 7 Hz should be the minimum sample frequency in order for the maximum position tracking error to stay within the specified 0.5 m. However in case 7 Hz should give implementation problems 6 Hz could also be used, but lowering the sample frequency even more is not advisable.