Faculty of Aerospace Engineering

Vision-based automatic landing of a quadrotor UAV on a floating platform

A new approach using incremental backstepping

A.S. Mendes

March 7, 2012

Delft University of Technology

Challenge the future

Vision-based automatic landing of a quadrotor UAV on a floating platform

A new approach using incremental backstepping

Master of Science Thesis

For obtaining the degree of Master of Science in Aerospace Engineering
at Delft University of Technology

A.S. Mendes

March 7, 2012

Faculty of Aerospace Engineering · Delft University of Technology

Delft University Of Technology
Department Of
Control and Simulation

The undersigned hereby certify that they have read and recommend to the Faculty of Aerospace Engineering for acceptance a thesis entitled **“Vision-based automatic landing of a quadrotor UAV on a floating platform”** by **A.S. Mendes** in partial fulfillment of the requirements for the degree of **Master of Science**.

Dated: March 7, 2012

Readers:

Ir. W. Berkelaar

Ir. B.D.W. Remes

Dr. ir. E. van Kampen

Dr. Q.P. Chu

Prof. dr. ir. J.A. Mulder

Abstract

The development of systems that allow unmanned aerial vehicles, known as UAVs, to perform tasks autonomously is a current trend in aerospace research. The specific aim of this thesis is to study and achieve vision-based automatic landing of a quadrotor UAV on a floating platform, a known target that possesses oscillatory behavior. The research contributions to be taken from this study can be divided into two perspectives, as described below.

From a theoretical point of view, a design solution is proposed which includes GPS navigation to enable the quadrotor to find the target, and vision-based control to approach and land upon it. From this design, several control-related issues must then be solved, mainly the development of a controller for the autoland mission. To accomplish this control task, an incremental backstepping control law is developed. Additionally, linear and standard backstepping controllers are designed for comparison. The derived control laws require knowledge of the states to close the feedback loops; therefore, state estimation algorithms are designed for complete state reconstruction. The approach selected is modular, thus separating position/velocity estimation from attitude determination. The former is performed using an extended Kalman filter, and the latter using a complementary filter. Furthermore, an augmented Kalman filter formulation is developed for estimation of the platform's vertical motion. The combination of control and state estimation algorithms is tested in a simulated environment using a simulation tool developed in this study for Monte-Carlo analysis. This tool allows for evaluation of the design not only for the nominal case, but also for random combinations of external conditions. Results show that successful performance is obtained for the nonlinear controllers since the desired criteria is met and the risk of crashing is demonstrated to be residual. Additional tests show that incremental backstepping is, in general, more robust than standard backstepping in the case of model mismatch, even in the presence of state estimation errors.

From a practical perspective, the findings are twofold. First, this thesis presents a procedure to experimentally determine the moments of inertia of the quadrotor by using a two-axis motion simulator and a six-component force/torque sensor. The inertia properties are also determined analytically using two modeling approaches: point mass analysis and assumption of simple geometric shapes. The results show that point mass analysis can lead to erroneous inertia estimation (deviation of 20-30% from the real value), thus resulting in a significant model mismatch. The experimental and simple shapes assumption methods render similar results, which strongly indicates not only that the experimental method proposed is valid, but also that the assumption of simple geometric shapes can be used as a reliable and cost-effective method to determine moments of inertia of small UAVs. Second, in this thesis the system is tested in real time using an actual quadrotor. Flight tests are performed for hovering above a target with known characteristics, and to achieve this end, a vision system is developed to obtain relative position measurements from images captured by an on-board camera. A Kalman filter is implemented for real-time integration of vision with IMU data, and a linear controller with reference command filters is used. Tuning procedures are then carried out until satisfactory performance is achieved.

Acknowledgments

I would like to thank the following people for their invaluable contributions to this thesis:

First, I would like to acknowledge Professors Ping Chu and Erik-Jan van Kampen for their guidance, orientation and support throughout this project. Not only did they share their knowledge and experience, but they did so with sincere enthusiasm that resulted in very interesting discussions.

I would also like to thank Professor Bob Mulder for giving me this assignment and trusting in me for this challenging project. His initial guidelines were the perfect head start to developing this thesis.

This research would not have been possible without Bart Remes and Christophe de Wagter, whose help was crucial for solving real-time implementation and practical-lab issues. Their contributions were key to achieving the final results, as developing real-time code and running flight tests with the actual helicopter are very challenging tasks. I would also like to acknowledge Gonçalo Bernardo for his valuable assistance in the lab.

Initiating this thesis would not have been possible without Professor Max Mulder. I owe him a great deal of thanks for all his help in my process of transferring to TUDelft.

I would like to extend my gratitude to Wiebe Berkelaar for joining the graduation committee. In fact, his Master's thesis served as a guiding reference for my project.

I would also like to thank the other students and staff from the Control and Simulation division of the TUDelft Aerospace Faculty, who helped to create a very enjoyable work environment.

Finally, I am grateful to my family and friends, and especially to my parents, my brother and Amber for always supporting and encouraging me in everything I do.

Delft, University of Technology
February 2012

Alexandre Santos Mendes

Acronyms

AHRS	Attitude Heading Reference System
ARMA	Autoregressive Moving Average
ARX	Autoregressive Exogenous
CCW	Counter Clockwise
CLF	Control Lyapunov Function
CM	Center of Mass
CoG	Center of Gravity
CSV	Comma-Separated Values
CTFT	Continuous-Time Fourier Transform
CW	Clockwise
DC	Direct Current
DFT	Discrete Fourier Transform
DTFT	Discrete-Time Fourier Transform
DUT	Delft University of Technology
EKF	Extended Kalman Filter
ENU	East North Up
FFT	Fast Fourier Transform
FOV	Field Of View
GCS	Ground Control Station
GPS	Global Positioning System
HSL	Hue Saturation Lightness
HSV	Hue Saturation Value
IEKF	Iterated Extended Kalman Filter
IGE	In Ground Effect
IMU	Inertial Measurement Unit
INDI	Incremental Nonlinear Dynamic Inversion
INS	Inertial Navigation System

INU	Inertial Navigation Unit
IR	Infrared
ISA	International Standard Atmosphere
Lisa	Lost Illusions Serendipitous Autopilot
LQG	Linear Quadratic Gaussian
LQR	Linear Quadratic Regulator
LTI	Linear Time-Invariant
MAV	Micro Areal Vehicle
MAVlab	Micro Areal Vehicle Lab
MEMS	Micro Electrical Mechanical Systems
MSL	Mean Sea Level
NDI	Nonlinear Dynamic Inversion
NED	North East Down
OGE	Out of Ground Effect
PID	Proportional Integral Derivative
RGB	Red Green Blue
RLS	Recursive Least Square
SISO	Single Input Single Output
TD	Touchdown
UAV	Unmanned Aerial Vehicle
VTOL	Vertical Take-Off and Landing

List of symbols

Greek symbols

ϵ	Modeling uncertainty
ϕ,θ,ψ	Attitude Euler angles
Φ	Discrete system transition matrix
η	Motor efficiency
λ_*	Sensor bias
$\underline{\nu}$	Innovations
ϑ	Two-axis motion simulator generic angle
$\underline{\Theta}$	Attitude Euler angles vector $(\phi,\theta,\psi)^T$
ρ	Air density
σ_*	Sensor noise standard deviation
τ_*	Time constant
$\underline{\omega}$	Angular rate vector $(p,q,r)^T$
ω_n	Natural frequency
ω_{plat}	Wave's frequency-dependent parameter
Ω	Generic rotor angular velocity
Ω_m	Rotor angular velocity from DC motors
Ω_r	Residual rotational rate of the rotors
ζ	Damping coefficient
Ψ	Input distribution matrix

Roman symbols

A	Rotor disc area
A_D	Aerodynamic area
A_*	To-be-sensed acceleration
b	Thrust factor
B	Input matrix for linear model

C_D	Aerodynamic dimensionless drag coefficient
C_{MQ}	Dimensionless moment coefficient
C_T	Dimensionless thrust coefficient
d	Drag factor (for quadrotor model)
d	Vertical distance from sensor to CM (for moments of inertia experiment)
D_*	IMU distance to quadrotor's CM
f	Frequency
f	Focal distance
f	Generic nonlinear function
$\underline{f}$	Nonlinear system equation
F	Linear system matrix
F_x	Linearized system matrix
F_*	Force
G	Input matrix for nonlinear model
g	Gravitational acceleration
h	Height (altitude)
$\underline{h}$	Nonlinear observer equation
H	Linear observation equation
H_x	Linearized observation equation
I_*	Inertia component
J	Inertia matrix
J_t	Total rotor inertia
J_r	Inertia of all for rotors
k_m	Motor torque constant
K_*	Controller gain (scalar or matrix)
l	Quadrotor's arm length
m	Quadrotor mass
M_*	Moment
p	Body rotational rate about x-axis
$\underline{p}$	Inertial position vector $(x,y,z)^T$
q	Body rotational rate about y-axis
Q	State covariance matrix
Q_d	Discrete state covariance matrix
r	Body rotational rate about z-axis
r	Reduction ratio (for actuator model)
R	Propeller radius (for quadrotor model)
R	Measurement covariance matrix
R_m	Motor internal resistance
s_*	Camera's scaling factor
t	Time variable
T_{plat}	Wave period
T_*	Torque (for moments of inertia experiment)
u	Body velocity component in the x-direction
u_m	Motor input
u	System input
v	Body velocity component in the y-direction

V_*	Control Lyapunov function
$\underline{V}$	Velocity vector expressed in the body-fixed reference frame $(u,v,w)^T$
w	Body velocity component in the z-direction
w_*	White noise
$\underline{w}_*$	Process noise
W	Quadrotor weight
x, y, z	Inertial position variables
x_p, y_p	Image plane coordinates
$\underline{x}$	System state vector
z_{plat}	Floating platform vertical position
z_*	Tracking error

Contents

Chapter 1

Introduction

An Unmanned Aerial Vehicle (UAV) is an aircraft that is flown without a human crew on board. It can either be piloted remotely or fly autonomously with the aid of preprogrammed flight plans. UAVs are often used in military operations. However, this type of craft is also used in civil applications, such as firefighting. In general, UAVs are preferred for missions that are considered too dull, dirty or dangerous for manned aircraft.

There are a wide variety of UAV geometric configurations, means of control and general design features. Rotorcraft UAVs, which function with rotating blades, have several advantages over most fixed-wing aircraft. These primarily include their ability to take off and land vertically and the capacity to hover over a specific fixed point. Furthermore, rotorcraft can operate in the low airspeed range where most fixed-wing aircraft cannot.

One of the possible configurations for rotorcraft is the so-called quadrotor. A quadrotor, also known as a quadrotor helicopter or quadrocopter, is a helicopter with four horizontal rotors and no tail rotor. All rotors are placed in the same plane and control of the vehicle's motion can be achieved by varying the relative speed of each propeller.

1-1 Autoland project description

1-1-1 Mission overview

There are multiple advantages to having automated machines perform surveillance tasks at sea. One example is the fact that certain types of such missions might be too dull to be performed by humans. Researchers from the Micro Areal Vehicle Lab (MAVlab) at the Delft University of Technology (DUT) are taking this problem into consideration and studying a solution involving several platforms floating in the water with one quadrotor UAV operating autonomously in the area. In this scenario, the helicopter would then be capable of navigating within the zone covered by the buoys, and could return safely to one of the floating platforms after completing a mission. A visual representation is shown in Figure 1-1.

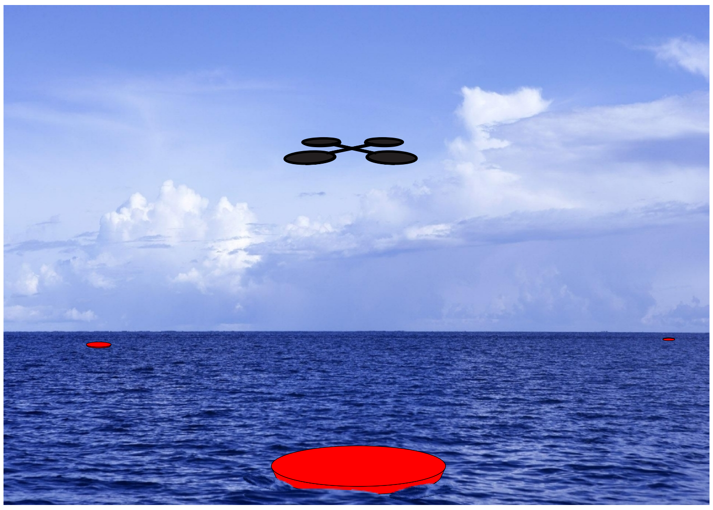


Figure 1-1: Visual representation of a quadrotor UAV returning to a buoy after performing a surveillance mission (quadrotor and platforms are not to scale)

As part of this problem, it is clear that an autopilot system that will enable a quadrotor UAV to land on a floating platform is needed. The main function of this system would be to track a moving device that possesses its own dynamics, and allow the air vehicle to safely approach and land upon it. To facilitate that function, several approaches can be considered for the estimation of the quadrotor's relative motion with respect to the floating platform. For this thesis, a vision-based approach will be used in the form of a camera installed underneath the UAV.

The project can then be summarized as follows:

There are several platforms floating in the sea and one quadrotor operating autonomously in the area covered by these platforms. Starting from any initial position, the quadrotor must be able to land on these buoys under given external conditions such as sea states and wind profiles. After landing, the helicopter must dock with the platform in order to recharge batteries and perform desired functionalities.

1-1-2 Development phase

The MAVlab at DUT makes it possible to put theory into practice with real flying UAVs. Therefore, this project is developed at DUT during its investigation phase such that it may be implemented later in real life. A test quadrotor (see Figure 1-2) is available and includes equipment such as an Inertial Measurement Unit (IMU), a Global Positioning System (GPS) receiver, a barometric altimeter and a camera that can be connected to the autopilot.

The real-time implementation of the control algorithms must be coded using Paparazzi, which is a free and open-source project intended to create exceptionally powerful and versatile autopilot systems by allowing and encouraging input from the community. Therefore, the design solution for this automatic landing mission must be compatible with existing software from this platform. Furthermore, it is expected that the design of the system can be easily extrapolated and applied to other quadrotors. In other words, one particular rotorcraft is used

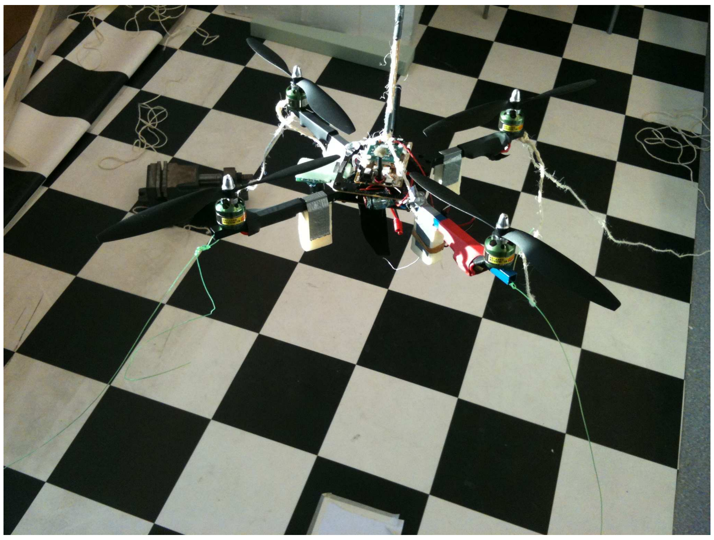


Figure 1-2: Test quadrotor

for this project, but the system should be easily transferable to another assembled helicopter with different features.

A Caspa VL camera from Gumstix will be used as the vision sensor (see Figure 1-3). This sensor is considered small, light and cheap, at the cost of $75.

Figure 1-3: Caspa VL from Gumstix

1-2 Problem statement

The goal of this thesis is to design a fully automated system for vision-based landing of a quadrotor UAV on a floating platform.

Since this project includes a real-time implementation component, it is also a secondary goal to analyze the feasibility of real-time implementation using cheap and light sensors.

1-3 Research challenges

Quadrotors are inherently unstable platforms and require constant compensation to maintain stable flight. Additionally, their highly nonlinear behavior makes the controller design tasks more complex. Furthermore, the automatic landing control problem under consideration brings the added challenge of a floating platform that possesses its own dynamics.

The set of sensors used to estimate the rotorcraft's states is limited and the control task becomes even more difficult when lower quality sensors are used. For this thesis, the use of cheap and light sensors and, in particular, the camera used for the vision system, creates challenges which are explained below.

- The update rate from the vision system is not sufficient for use in the control loop and therefore this problem must be addressed.
- Feedback latency is also a crucial factor for vision-based helicopter control. Time delays, mostly due to frame acquisition and image processing, significantly affect stability properties.
- Attitude variations are expected during flight. One of the main drawbacks of using a downward-looking camera fixed to the quadrotor is the fact that it is difficult to distinguish between rotation and translation motions of the flying vehicle in images under perspective projection.

It is also clear that sensor calibration and alignment is important for achieving high performance (or even just stable flight). These tasks may be time-consuming, and it is sometimes very difficult to obtain accurate measurements.

Another issue is that helicopters are dangerous due to their spinning rotor blades. Flight tests cannot be performed without taking the necessary safety precautions.

The final goal is to implement the autoland system in an outdoor environment. However, during its development phase, the system is tested indoors. This alters the nature of the problem since indoor conditions are, in general, different from those outdoors. The tests must be adapted to cope with the environment. One of the greatest challenges is the corruption of magnetometer information indoors, which leads to false yaw measurements.

Furthermore, the model of the quadrotor changes during the development of the thesis work. For example, different batteries with different shape and weight are used, rotor blades and other components of the structure might bend, etc.

1-4 Literature review

The topic of vision-based automatic landing of a quadrotor UAV on a floating platform can be addressed from multiple viewpoints. It is thus necessary to cover different aspects of the problem that are considered pertinent to the research being carried out.

As a starting point, recent research on quadrotor UAVs performed at DUT was examined. (Wierema, 2008) designed an indoor navigation system using Infrared (IR), and (Berkelaar

& Oonk, 2009) later designed a system for altitude and attitude control using laser. For both these theses, the authors tested their controllers with real quadrotors in order to validate their algorithms in real-time implementations.

This study is then another step with respect to control research performed at DUT. What follows is a summary of relevant topics that were researched for this thesis.

1-4-1 Quadrotor control

Control allocation

From (Wierema, 2008) and (Berkelaar & Oonk, 2009), it is possible to understand the basic concept of how to control a quadrotor. A brief description of the control allocation follows. In a regular helicopter configuration, the torque produced by the main rotor is counteracted by the tail rotor. For a quadrotor configuration, there are four control inputs, which are the inputs given to each of the four motors. The control forces and moments are then generated by varying the relative speeds of the different rotors. For the most common layout of quadrotors, the two pairs of rotors (1-2) and (3-4) spin in opposite directions as illustrated in Figure 1-4. Control allocation is then performed as follows:

- increasing or decreasing the four rotors' speed altogether generates changes in the total lift produced, thus creating motion in the vertical direction (top left in Figure 1-4);
- changing the relative speed of propellers 1 and 2 will induce a pitch motion, causing a tilt of the thrust vector and thus forward/backwards movements (down right in Figure 1-4);
- the same concept applies for propellers 3 and 4, but for roll motion, which results in lateral translations (down left in Figure 1-4);
- yaw rotation is the result of the difference in the counter-torque between the two pairs of rotors (top right in Figure 1-4).

Control techniques

Control of quadrotors is a challenging task due to the fact that these vehicles are underactuated and present strongly nonlinear behavior. Different control design methods have been investigated over the years to address this challenge. (Hua, 2009) presents a comprehensive overview of such methods in the context of linear as well as nonlinear control systems. In terms of linear systems, multiple references are given to Single Input Single Output (SISO) Proportional Integral Derivative (PID) control as well as optimal control strategies such as Linear Quadratic Regulator (LQR), Linear Quadratic Gaussian (LQG) and H_2 and H_∞. The author states that such approaches have limitations when applied to highly nonlinear systems, and presents a survey on nonlinear control techniques that should, in principle, allow most of these limitations to be overcome. These nonlinear control strategies encompass feedback linearization, backstepping, model-based predictive control, sliding mode and neural-network-based adaptive control.

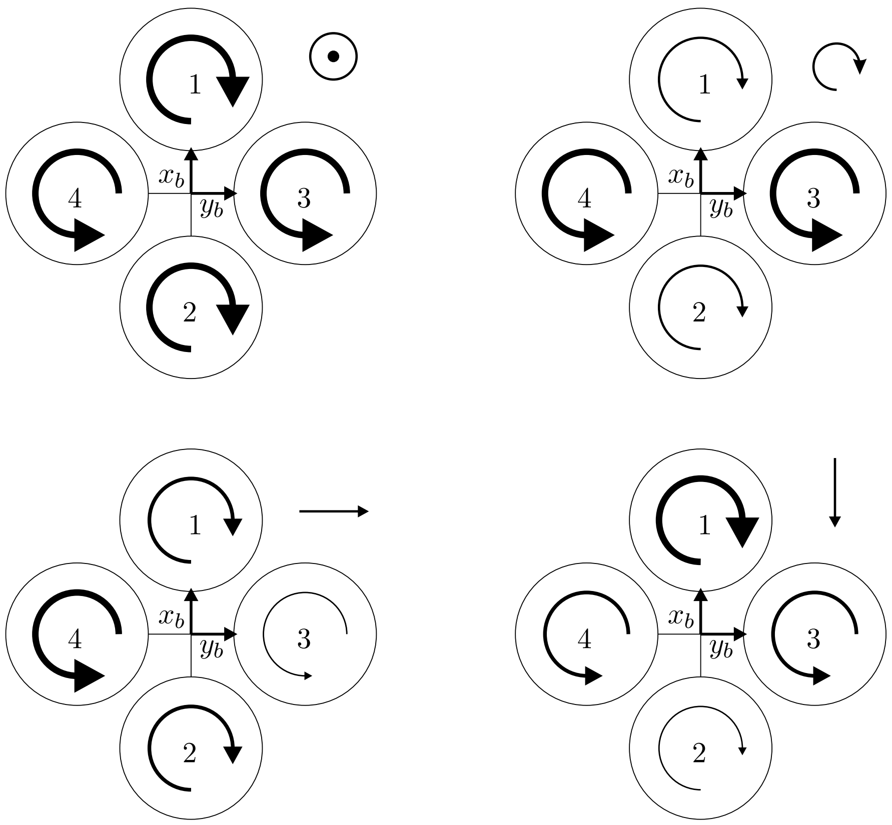


Figure 1-4: Quadrotor control concept

A special case of feedback linearization is the so-called Nonlinear Dynamic Inversion (NDI), which suffers from the major drawback that performance is lost in the case of model mismatch. At DUT, a so-called incremental nonlinear dynamic inversion technique has been studied in which angular accelerations are fed back to the controller (Sieberling, 2009), (Wedershoven, 2010) and (Simplicio, 2011). In theory, this approach eliminates sensitivity to model mismatch, thus increasing the robust performance of the system in comparison with conventional nonlinear dynamic inversion. The problem of estimating angular acceleration has also been addressed, and one proposed solution is a linear predictive filter (Sieberling, Chu, & Mulder, 2010).

There is no theoretical proof that the use of nonlinear dynamic inversion results in a stable closed-loop system. Therefore, a new step in research has been taken in which incremental backstepping techniques are investigated. Note that backstepping guarantees the stability of the system during the design process. (Acquatella, 2011) applies this control strategy to spacecraft control and suggests investigation of application to quadrotor control. These studies on incremental-based control laws are examples of state-of-the-art research in the area of generic robust flight control system design.

1-4-2 Quadrotor modeling

A model of the quadrotor UAV is necessary to facilitate controller design tasks and allow for validation of the design concepts through simulations. The topic of helicopter flight dynamics has been extensively investigated by (Padfield, 2007). Additionally, a comprehensive overview on helicopter aerodynamics has been presented by (Seddon & Newman, 1990) in which an explanation of both the Momentum Theory and Blade Element Theory is provided. (Leishman, 2002) also presents an overview on principles of helicopter aerodynamics.

In more recent years, further approaches to modeling such platforms have been investigated and validated with real flight test data. (Fay, 2001) described the derivation of aerodynamic forces used in the equations of motion for stability and control analysis of a rotorcraft (the mesicopter). (McKerrow, 2004) gave a theoretical analysis of the dynamics of a four-rotor helicopter (the Draganflyer) in order to develop a model of it. An interesting aspect of their work was a method to determine the moments of inertia in which the authors assumed masses with known geometric shapes attached to a center of rotation by thin rods. (Pounds, Mahony, & Corke, 2006) developed the X-4 Flyer, a quadrotor robot, using custom-built chassis and avionics with off-the-shelf motors and batteries. These researchers found the X-4 Flyer to be a highly reliable experimental platform and they provided modeling techniques for controller design. The work presented by (Hoffmann, Huang, Waslander, & Tomlin, 2007) sought to address issues that arise when deviating significantly from the hover flight regime. Three separate aerodynamic effects were investigated as they pertained to quadrotor flight, namely total thrust produced, blade flapping and airflow disruption. They showed that these effects caused moments that affected attitude control, and thrust variation that affected altitude control.

Some of the most crucial effects to be modeled entail the dynamics of the motors. In (Franklin & Emami-Naeini, 2006), a model of Direct Current (DC) motors is provided. From this model two main aspects should be highlighted: first, the model is nonlinear, since it includes a quadratic term; second, it includes a time constant parameter, meaning that there will be a delay between input requested and actual rotor speed.

At DUT, (Wierema, 2008) and (Berkelaar & Oonk, 2009) used the model described by (Bouabdallah, 2007). This model was quite comprehensive and was validated with flight test data. From the research of the aforementioned authors, it becomes clear that the most challenging components to be modeled are the aerodynamic effects and, more specifically for controller design, the mapping between rotor input and force/torque produced.

Taking into account all prior research mentioned in this section, the following is a brief summary of important aspects to consider when modeling a quadrotor:

- Rigid body dynamics (mass and inertia)
- Aerodynamic forces and moments of a rotor
- Structural friction
- Rotor gyro effect
- Motor dynamics

- Blade flapping
- Ground effect

1-4-3 Landing on a floating platform

To investigate possible approaches to solving the automatic landing problem, a more expansive literature survey was needed in order to include contributions that shed light on various aspects of the issue, even if they did not directly pertain to automatic landing of a quadrotor UAV on a floating platform. This section contains a summary of different contributions that are considered relevant to this project.

Ship motion estimation

A floating platform can surge, sway and heave as well as roll, pitch and yaw, making it a six degree of freedom moving target as shown in Figure 1-5.

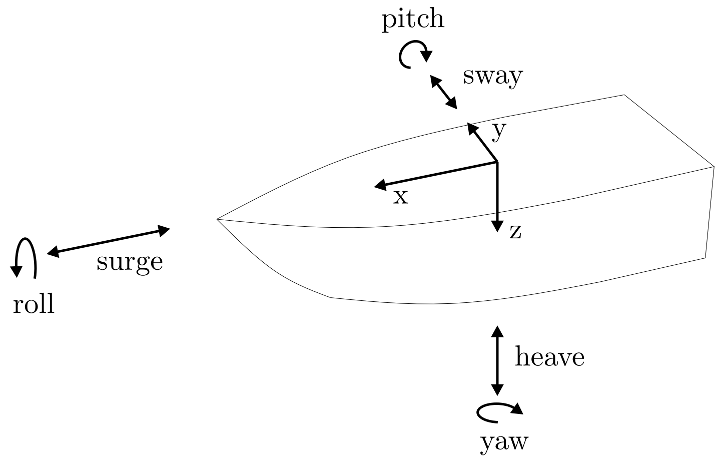


Figure 1-5: Floating platform motion

One of the possible ways of looking at the problem is by investigating how manned operations are executed for landing on the deck of a ship. In fact, this topic has been extensively studied since for these types of operations pilots usually need the aid of automated systems. More specifically, much research has been done regarding prediction of lull opportunities for landing.

Note that the work mentioned in this subsection concerns mostly design solutions for finding calm opportunities for landing. However, for this thesis, these contributions are important as they describe different ways of building models of a platform floating in the sea, and therefore can be used to help obtain a better state estimation of the relative motion between the quadrotor and the buoy.

One of the earliest papers on this topic was published in 1965, when Dalzell wrote a note on carrier deck motion using a Wiener filter linear prediction technique (Dalzell, 1965). In 1969, Kaplan proposed a Kalman filter as an alternative (Kaplan, 1969), tackling the implementation complexities of Dalzell's Wiener filter. Throughout the 1970s and 1980s, many other studies were carried out in which researchers used Kalman filtering techniques to study the ship motion estimation problem.

In 1977, Weiss and Devries designed a filter based on the state space modeling of ship motion dynamics (Weiss & DeVries, 1977). The authors claimed that the design of Kalman filters which model ship motion spectra in order to measure linear and angular ship motion parameters is superior to the design of Kalman filters which ignore this information. The measurement system consisted of linear and angular accelerometers in a strapdown mode (with a position reset at the time of interest), and bounding filter techniques were used to minimize the effects of mismodeling.

In 1982, Triantafyllou and Bodson published their findings on predicting a vessel's motion in real time using Kalman filtering techniques, claiming that the prediction time was accurate to between two and ten seconds (Triantafyllou & Bodson, 1982). In 1983 the same authors studied the estimation of heave, pitch, roll, sway and yaw motion of a DD-963 destroyer for application to the landing of Vertical Take-Off and Landing (VTOL) aircraft (Triantafyllou, Bodson, & Athans, 1983). The governing equations for modeling purposes were obtained from hydrodynamic considerations in the form of linear differential equations with frequency-dependent coefficients.

In 1983 Sidar and Doolin also showed quantitatively that, based upon the power spectrum data for pitch and heave measured for various ships and sea conditions, motion could be accurately predicted for a duration of up to fifteen seconds (Sidar & Doolin, 1983). Furthermore, the authors claimed that the zero crossover times for both pitch and heave motions could be predicted with high accuracy. The predictor was also based on Kalman's optimum filtering theory.

Following these studies, (Lainiotis, Charalampous, Giannakopoulos, & Katsikas, 1992) proposed a different approach. They claimed that since the designs of the Kalman filters were based on the assumption of complete knowledge of the model describing the ship motion dynamics, there would be a degradation in the estimate quality in the case of mismatch between the mathematical model used and the actual ship's model. The authors addressed the estimation problem as a nonlinear adaptive estimation problem for partially unknown, time-varying linear systems with a non-Gaussian initial state vector. The filter used was designed based on an adaptive Lainiotis partitioning approach.

In 1981, Yumori published his findings on predicting heave motions using Autoregressive Moving Average (ARMA) Models (Yumori, 1981). This concept was later extended to include pitch and roll motion predictions (Broome & Hall, 1998). (Yang, Pota, Garratt, & Ugrinovskii, 2008) studied a prediction method using an Autoregressive Exogenous (ARX) model, with the aid of the Bayes Information Criterion to obtain the optimal system order. The model coefficients identified from a Recursive Least Square (RLS) method were employed to predict vertical motion of a floating vessel.

Other approaches have also been considered. (Lainiotis, Plataniotis, Penon, & Charalampous, 1993) explored the real time estimation of ship motion using a neural estimator based on a dynamic recurrent neural network. (Fleischmann, 2000) proposed a forward-looking method using radar as a range-measuring sensor to predict future motions. In this approach, the measurements were subjected to Fast Fourier Transforms in order to determine periods of quiescence. (Riola, Diaz, & Giron-Sierra, 2011) investigated the possibility of using wavelets in the prediction of calm opportunities for landing a helicopter on a ship.

Additional important factors to be taken into account in this type of research are the definitions of the envelope of operability and limits for the floating platform. (Ferrier, Bradley, & Blackwell, 2001) defined these limits as 2 degrees in pitch, 6 degrees in roll (8 degrees at night), 8 ft/s heave rate and 3 ft/s sway rate. Note that these limits were defined for manned operations; however, they also serve as guidelines for automatic landing procedures.

Helicopter landing on a ship deck

It is now important to investigate what others have done regarding actual methods for landing a helicopter on a ship deck, as well as examine the corresponding controller designs.

(McMuldroch, Stein, & Athans, 1979) proposed a control algorithm for landing a VTOL type aircraft on a small ship in rough seas. The design included an aircraft-tracking-ship-motion controller as well as a specification of the actual landing control algorithm.

(Bodson & Athans, 1985) contributed a solution for this problem by designing a controller based on linear quadratic optimal theory. Among others, one of the most interesting aspects of this design was the fact that the ship motion was estimated using a Kalman filter (with ship sensor measurements), and a feedforward gain matrix was used to enhance performance.

(Storvik, 2003) proposed a guidance system for automatic approach and descent onto a wave-excited ship. The author designed a way-point generator on the basis of the initial position, course and minimum turning radius. In order to connect the different way points and generate a tracking path, spline interpolation methods were used.

(Oh, Pathak, Agrawal, Pota, & Garratt, 2006) addressed the design of an autopilot for autonomous landing of a helicopter on a ship. In their design, a tether is used for landing and securing the helicopter to the deck of the ship in rough weather. The controller is based on the time-scale separation between rotation and translation, and it was shown that the tether tension could be used to alter the coupling between these two motions. Although this solution showed promising results, it is clear that extra equipment would be necessary to perform the autoland mission.

(Saghafi & Esmailifar, 2007) studied the problem of automatic helicopter landing on a four degree of freedom platform. Later, (Esmailifar & Saghafi, 2009) extended this work to a six degree of freedom platform. In the control system, the landing phase is divided into two stages: approach and touchdown. In the first stage, the helicopter tries to attenuate the initial position and direction errors, and in the next stage the platform's attitude is tracked for a safe touchdown. This is achieved by means of a State Dependent Riccati Equation method, in which an objective function containing penalizing terms for states and inputs is minimized in order to generate suboptimal control feedback. The state matrix of this objective function is a state-dependent positive-definite square matrix, which is the same size as the helicopter states. The strength of the states and their interactions can be tuned in the objective function with the components of this gain matrix (for a higher effort to track a state, the related component is increased). This solution is relevant for the problem since both position and attitude are tracked for the touchdown moment.

(Ford & Boloye, 2010) proposed a combination of GPS and Inertial Navigation System (INS) for the task of following a moving ship. The approach consisted of updating the Inertial Navigation Unit (INU) by using data from the GPS receivers. Both aircraft and carrier would possess such systems, and the shipboard would transmit the INU and GPS data to the aircraft. With the information gathered, the aircraft could then determine a vector to the landing site.

Contributions on related problems

The previously mentioned work regards the landing of helicopters on ship decks, which is a very similar problem to the one being considered for this thesis. However, the literature survey would not be complete without looking into solutions presented in similar projects. With this in mind, other related problems were investigated.

In (Saripalli, 2009), a downward-looking camera is used to track a target with known characteristics. The motion of the target is also known. A template-matching algorithm is used for acquiring the target in the images and is integrated with a trajectory controller for landing the helicopter. The position of the target in the image is used as the input to a robust Kalman filter. A linear controller based on a kinematic model of the helicopter is used to perform trajectory following and landing.

(Wenzel, Masselli, & Zell, 2010) presented a system that would allow a UAV to land autonomously on a carrier moving on the ground (in a horizontal plane only). The authors used a Wii remote IR camera for tracking a known pattern of IR lights installed in the moving carrier. PID control techniques were used to perform the landing mission, augmented by second-order derivative terms (taking into account the acceleration of the aircraft).

At the University of Pennsylvania, a controller was designed for aggressive maneuvering of a quadrotor (Mellinger, Michael, & Kumar, 2010) and (Mellinger, Shomin, & Kumar, 2010). Among other possibilities, the helicopter could perch on a steady inclined surface. This was achieved through a method that allowed the quadrotor to fly through any position in space with reasonable velocity and pitching (3D trajectory control). This approach may be useful, but a trajectory would have to be computed and validated on board for autoland on a moving platform with unknown motion. In addition, the flight tests were performed indoors in protected environments with expensive measurement systems.

(Dalamagkidis, Ioannou, Valavanis, & Stefanakos, 2006) discussed a mobile landing platform for miniature VTOL vehicles. The system consisted of an unmanned ground vehicle with a landing platform mounted on top. A gimbaled subsystem design was proposed and the necessary equations were derived to level the platform regardless of the pose of the ground vehicle. This idea was developed for operation on hard surfaces, but can be transferred to a platform on the sea, as shown in (Ampelmann, 2009).

1-4-4 Vision in the loop

For autonomous landing, the aerial robot must be capable of performing navigation. This task depends not only on the control strategy applied, but also on the set of sensors used (Castillo, Lozano, & Dzul, 2005). The design of UAVs capable of performing navigation with precise path tracking usually involves a trade-off between performance, price, weight and payload, which is not easy to achieve. In general, UAV orientation measurement is achieved by using an IMU (composed by gyros, accelerometers and magnetometers) while outdoor position and velocity measurements are usually obtained from GPS. The main drawback of the common GPS receivers used in Micro Areal Vehicles (MAVs) is the fact that position measurements may be inaccurate by up to a few meters. Therefore, the GPS solution for navigation is not applicable for missions in which precision landing is required. Furthermore, the exact GPS coordinates of a floating platform are not known due to the fact that the buoy tends to drift within a certain radius.

One of the alternative ways to measure position and velocity for navigation purposes is based on computer vision, which is the solution to be implemented in this thesis. It is therefore necessary to analyze the implications of utilizing a vision sensor (a single camera) that appears in the control loop. Vision-based navigation techniques have been and continue to be developed by various research teams. What follows is an overview of important issues related to vision in the loop.

Using vision as a measuring system has multiple advantages over employing other sensors, as shown in (Nordberg et al., 2002), (Saripalli, Montgomery, & Sukhatme, 2003) and (Wu, Johnson, & Proctor, 2005). For example, cameras are passive (i.e., they do not emit external signals) and in general are also light and cheap. Furthermore, they can be used for both indoor and outdoor applications.

Visual servoing is the use of feedback from a camera or a set of cameras, and this approach has been used in many applications for the development of control algorithms such as motion control of mobile robots (Ma, Kosecka, & Sastry, 1997). (Lozano, 2010) presents an overview on different visual servoing techniques, differentiating between two categories:

- direct visual servoing, in which the visual control is directly responsible for giving actuator commands to the robot, and
- indirect visual servoing, in which the visual control gives a control input to a lower level control loop that computes actuator commands to the robot.

In this study, three types of strategies are also indicated according to the space of control used:

- position-based, using 3D information from the scene expressed in a well-known Euclidean reference;
- image-based, using 2D measurements extracted from the images;
- position-image based, a combination of the previous two.

Figure 1-6 depicts a simplified schematic representation of how a real target is projected in the image plane of a camera. This method of modeling computer vision position estimates has been widely used, for example in (Smith, Sridhar, & Hussien, 1992), (Hintze, 2004), (Wu et al., 2005), (Xu, Qiu, Liu, Kong, & Ge, 2006), (Dobrokhodov, Kaminer, Jones, & Ghabcheloo, 2006), (Daquan & Hongyue, 2007) and (Kendoul, Nonami, Fantoni, & Lozano, 2009).

The basic equations for image projection are then given by

$$\begin{aligned} x_p &= x\frac{f}{z} \\ y_p &= y\frac{f}{z} \end{aligned} \tag{1-1}$$

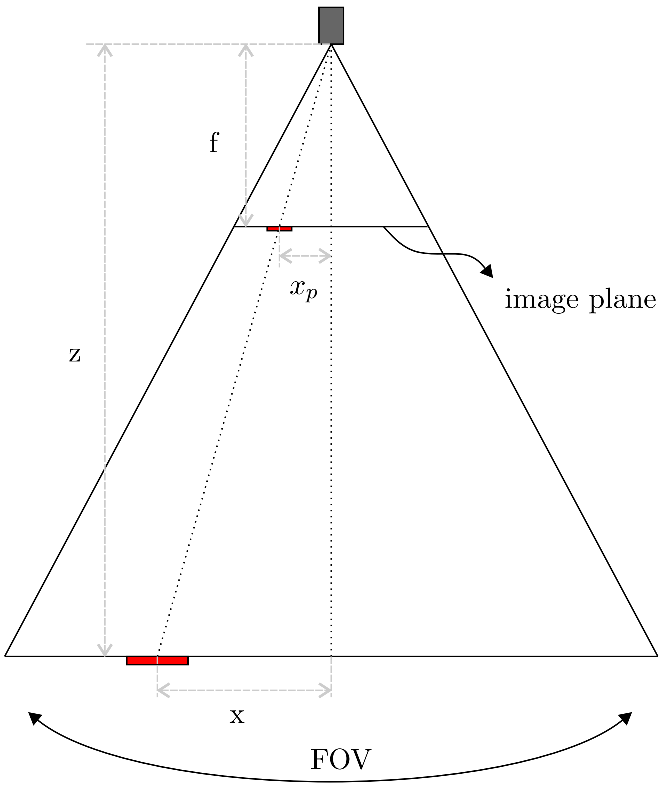


Figure 1-6: Schematic representation of the image projection of a target in a camera frame

where f is the focal distance of the camera. Note that x_p corresponds to the image projection in the x-direction and y_p in the y-direction. Note also that Eqs. (1-1) can only be applied if the target is in the camera's Field Of View (FOV).

Another important concept is that of optical flow, which is the time derivative of the image projection, and can be expressed mathematically by (Smith et al., 1992):

$$\begin{aligned}\dot{x}_p &= \frac{(-fV_{xE} + x_pV_{zE})}{z} + \frac{x_py_p}{f}\omega_{xE} - f\left(1 + \frac{x_p^2}{f^2}\right)\omega_{yE} + y_p\omega_{zE} \\ \dot{y}_p &= \frac{(-fV_{yE} + y_pV_{zE})}{z} - \frac{x_py_p}{f}\omega_{yE} + f\left(1 + \frac{y_p^2}{f^2}\right)\omega_{xE} - x_p\omega_{zE}\end{aligned} \tag{1-2}$$

where $\{V_{xE}, V_{yE}, V_{zE}\}$ and $\{\omega_{xE}, \omega_{yE}, \omega_{zE}\}$ are the camera's translational and rotational velocities expressed in the Earth reference frame. There are several algorithms for measuring optical flow, including correlation-based techniques, features-based approaches, differential techniques, etc., as shown in (Barron, Fleet, & Beauchemin, 1994).

From Eqs. (1-1), it is possible to observe that when the measurements of x_p and y_p are known, there are only two equations available to estimate three variables (x, y and z), which correspond to the 3D position of the camera with respect to the target. The estimation of z (distance between camera and target plane) is a type of problem known as range finding. Different contributions for solving this problem exist. For example, (Smith et al., 1992) proposes a recursive method for estimating range using a Kalman filter with a monocular sequence of images, (Kendoul et al., 2009) uses a recursive least squares approach to estimate z assuming that height remains approximately constant above a certain target, and (Dunbabin, Corke, & Buskey, 2004) uses multiple cameras (stereo vision) to determine the distance to the target plane. A survey on range-finding techniques can be found in (Jarvis, 1983).

The combination of vision with other sensor information has multiple advantages. (Lozano, 2010) combines the use of a camera with an IMU, which allows for full state estimation. In this study, vision is used to obtain position, velocity and yaw measurements while IMU is used to measure pitch and roll as well as angular rates. It is also possible to perform sensor integration and combine vision with inertial measurements in a filter as shown by (J. Chen & Pinz, 2004) and (Chroust & Vincze, 2004). Other approaches using different combinations of sensors can also be found: (Chatterji, Menon, & Sridhar, 1997) combines vision with GPS, (Wang et al., 2008) combines vision with INS and GPS and (Hubbard, Morse, Theodore, Tischler, & McLain, 2007) uses camera, laser range and IMU as sensors for full navigation.

Finally, there are two additional concepts that should be mentioned. First, it is important to stress that one of the key features of successful vision-based helicopter control is the frequency at which camera images are sampled and processed. Helicopters can move quickly and it is not guaranteed that the update rate of a vision system is high enough for control purposes. According to (Amidi, 1996), on-board image processing must be performed at a frame rate of 30 Hz or higher for effective vision-based object tracking. Second, for cameras that are fixed to the body-fixed reference frame, attitude compensation is required. This correction should be made using information about the vehicle's orientation (Amidi, 1996).

1-5 Research approach

After presenting the project and investigating pertinent literature, this section aims to describe the research approach for this thesis.

1-5-1 Detailed thesis objectives

The general intent of this thesis is as follows:

Design a vision-based system for automatic landing of a quadrotor UAV on a floating platform. Evaluate feasibility of real-time implementation with cheap and light sensors.

Having performed a literature search, it is possible to break this goal down into more specific sub-goals, and thus a more detailed set of objectives:

- Design the overall solution for the automatic landing mission. This will include all steps from the initial position of the quadrotor until it lands and docks with the platform. This solution must include an autonomous system based on computer vision using a single camera for the last stage of the landing;
- Select the characteristics of the to-be-tracked target and design algorithms to process camera images in order to measure relative motion between the quadrotor and the target. Filters must to be designed in order to provide full state estimation for use in the control laws;

- Develop control laws in order to accomplish the end goal of the mission: the autonomous landing. Regarding real-time implementation, as Paparazzi will be used, it is necessary to develop the control algorithms in such a way that they will be compatible with existing code. Furthermore, one of the most important goals is to design a system that can be applied to any quadrotor platform and not just that of the test quadrotor.

1-5-2 Contributions

To achieve the end goal of this thesis and accomplish the objectives set out above, several steps are taken as described in this section. These can be seen as contributions given in different control-related fields.

Before solving specific design issues, an overview of the design solution for the autoland mission is required. In short, the system will perform GPS-based navigation to approach the area of the buoy, and will switch to camera vision once the platform is in sight. Vision-based landing is then performed in multiple stages from initial correction of the lateral error to final touchdown with desired sink rate. Several options for the docking system are considered and explained in this thesis.

A detailed description of contributions corresponding to the steps taken is given below.

Modeling

A quadrotor model is needed to facilitate controller design tasks and to allow validation of the design concepts.

- For this thesis, a Matlab/Simulink simulation tool is created, in which models built by previous students at DUT are used. This simulation tool is extended to include a Monte-Carlo scheme, thus allowing evaluation of the control laws not just for the nominal case, but for a set of combinations of external conditions.
- In general, calculation of the moments of inertia is performed assuming mass components at a certain distance from the quadrotor's center of mass. In this thesis, an experiment to determine moments of inertia is designed and implemented. The general approach is to make use of the relation between torque, angular acceleration and inertia. For this purpose, a turn table is used as well as a force/torque sensor.

Computer vision

In terms of the vision measurement system, a blob-tracking approach is considered most suitable. Since red is not a color that can be easily found in nature (specifically at sea), the target is designed as a red blob. As this target has known characteristics, different sizes, shapes and other features must be considered and analyzed. An image processing algorithm will also be defined in order to obtain relative 3D position measurements. After evaluating the different possibilities, one design solution will be selected which corresponds to a good compromise/trade-off between computation time and measurement accuracy.

Filtering

It is expected that position measurements will be accurate at the centimeter level. However, the update rate is expected to be low. Therefore, a Kalman filter formulation is proposed in which vision data is integrated with IMU data. In this thesis, a model of the floating platform is assumed and incorporated into the filter for better state estimation. In particular, the vertical motion of the wave-excited buoy is estimated using an augmented Kalman filter, containing a frequency-dependent sea state parameter.

The complete state reconstruction is performed in a modular approach, in which the filter to estimate position and velocity is separated from that which estimates attitude. Note, however, that both filters use information from one another. This approach was chosen taking into consideration the fact that if less complex algorithms are employed, substitution or improvement of subsystems is easier.

Control

Development of the control laws is approached from two angles: on the one hand, advanced control laws based on state-of-the-art research are considered from a theoretical point of view; on the other hand, due to the time frame established for this thesis, practical solutions are adopted for real-time implementation.

- At DUT, nonlinear control laws using information of angular accelerations have been studied. In this thesis, an incremental backstepping design is analyzed in simulations. The controller must cope with the requirements of the design solution.

- Regarding real-time implementation, it is necessary to study the control loops that are already implemented in Paparazzi and design the new automatic landing control laws in such a way that they are compatible with existing code. Since Paparazzi uses PID for attitude control as the inner loop and PID for position control as the outer loop using GPS measurements, the proposed design solution for real-time implementation is to replace GPS with filtered vision position and velocity estimations. As it is a non-model-based approach, this strategy avoids redesigning the entire control loop when applying the controller to a different quadrotor; the only aspect that must be verified is tuning the PID gains.

1-6 Thesis outline

The structure of this thesis is presented below. Each of the core chapters addresses a different aspect of the research corresponding to a specific step towards the final goal.

- **Chapter 2: Design solution.** In this chapter, an overview of the design solution for the automatic landing problem is presented. This includes the different phases of the landing and design choices for safe approach and docking.

- **Chapter 3: Quadrotor and environment models.** In this chapter, the quadrotor is modeled to facilitate the controller design process. The environment model is also given for simulation purposes.
- **Chapter 4: Moments of inertia experiment.** In this thesis, a new method to determine the quadrotor's moments of inertia is proposed and implemented. In this chapter the concept of the experiment is explained and the results are shown.
- **Chapter 5: Vision system.** Camera images obtained on board are used to determine a 3D position vector of the quadrotor with respect to the target. In this chapter, the design choices for the vision system are presented and the corresponding algorithms are explained.
- **Chapter 6: State estimation.** In this chapter, the filters used for full state reconstruction are presented. Specifically, the augmented Kalman filter design for estimation of the platform's vertical motion is given and the sensor integration approach for estimation of position and velocity (also based on Kalman filtering) is explained.
- **Chapter 7: Controller design.** In this chapter, the proposed controller designs are presented. The backstepping design is first introduced, followed by incremental backstepping. A linear controller is also included. Finally, an autoland controller mode is designed to cope with the desired system requirements.
- **Chapter 8: Monte-Carlo simulations.** The overall system is simulated and the results are presented in this chapter, including simulations in which perfect state knowledge is assumed, as well as simulation performed with state estimation algorithms in the loop. Tests are performed for both nominal and uncertain cases.
- **Chapter 9: Real-time implementation.** In this chapter, the equipment and software used for flight tests are presented along with the results. Also included are the different steps taken towards stabilization of the quadrotor over a target.
- **Chapter 10: Conclusion.** Conclusions are drawn and lessons learned are shared in this final chapter. A list of recommendations for future work is also given.

Chapter 2

Design solution

Following the project introduction and literature survey, an overview of the design solution is now presented in this chapter. It is first assumed that the quadrotor can only operate under certain conditions. A set of performance requirements is thus defined and will serve as a guideline for portions of the remainder of this thesis, primarily with regard to controller synthesis. The design proposed in this chapter is an overview of the entire system and sets the stage for the implementation of subsystems required to accomplish the mission's goal.

2-1 Assumptions

Several assumptions must be made before providing a design solution for this vision-based automatic landing mission. First, it is assumed that the quadrotor cannot operate under stormy conditions; therefore heavy rain, strong turbulence, gusty winds and extreme rough sea states are considered unsuitable conditions for operation. To be consistent, the Beaufort Wind Force Scale is used. The Beaufort Scale is an empirical measure that relates wind speed to observed conditions at sea. The modern scale numbers range from zero to twelve, with zero being calm conditions (less than 0.3 m/s wind speed and flat sea surface) and twelve being hurricane force (wind speed greater than 32.7 m/s and wave height greater than 14 m). For this thesis, conditions up to level four are considered; that is, wind speed up to 8 m/s and wave height up to 2 m.

The floating platform must be anchored, otherwise it could drift away with sea currents. Subsequently, the anchor's GPS coordinates will be known and the buoy will always be located within a certain radius at the water surface. Furthermore, it is assumed that the platform's motion is constrained. More specifically, it has negligible motion in the horizontal plane and variations in pitch and roll angles are very small. This assumption is valid provided that the platform is sufficiently large and possesses a mechanism to maintain the landing area leveled with the horizon. As shown in the literature survey, such systems already exist with the inverted six degrees of freedom motion simulator concept. However, even more cost-effective solutions could be implemented based on other maritime engineering applications, such as anti-roll tanks.

The final assumption is that the vertical motion of the platform is similar to the motion of a water particle at the sea surface. Therefore, high frequency motion due to the buoy's flotation is not present. Again, this approximation is valid for a large and well-stabilized floating platform.

2-2 Performance requirements

Certain requirements must be defined to ensure proper performance. In order to accomplish the mission, the vehicle must land smoothly on the buoy as opposed to crashing against it or even landing in the water. The following performance requirements are thus defined:

- Pitch and roll angles should be zero at the landing moment; the desired values are then
  - $\theta_{TD}^{des} = 0$ deg $\quad \phi_{TD}^{des} = 0$ deg
- Touchdown (TD) point should be at the center of the designated landing area; the desired values are then
  - $x_{TD}^{des} = 0\ m \quad y_{TD}^{des} = 0\ m$
- At TD, vertical velocity (sink rate) relative to the platform should be small; the desired value is designed as
  - $\dot{z}_{TD}^{des} = 0.5\ m/s$

For all the above requirements, thresholds are defined to establish the limits of the safety region of operation. For example, if the attitude angles are too large at TD, the quadrotor could flip, and this cannot be allowed. Additionally, a time constraint exists due to the fact that batteries run out after a certain period of flying time.

2-3 Step-by-step landing procedure

The general goal of the mission is to bring a quadrotor back to a platform after performing a pre-specified task at sea. The vehicle might be in a random position between buoys and must navigate to one of them and land upon it. As stated before in this chapter, the anchor's GPS coordinates are known. Therefore, the first controller mode is GPS navigation. Once the platform is in sight, the system should then switch to vision-based control. Figure 2-1 shows a block diagram with the different controller modes required for the procedure. What follows are detailed explanations of each of these modes.

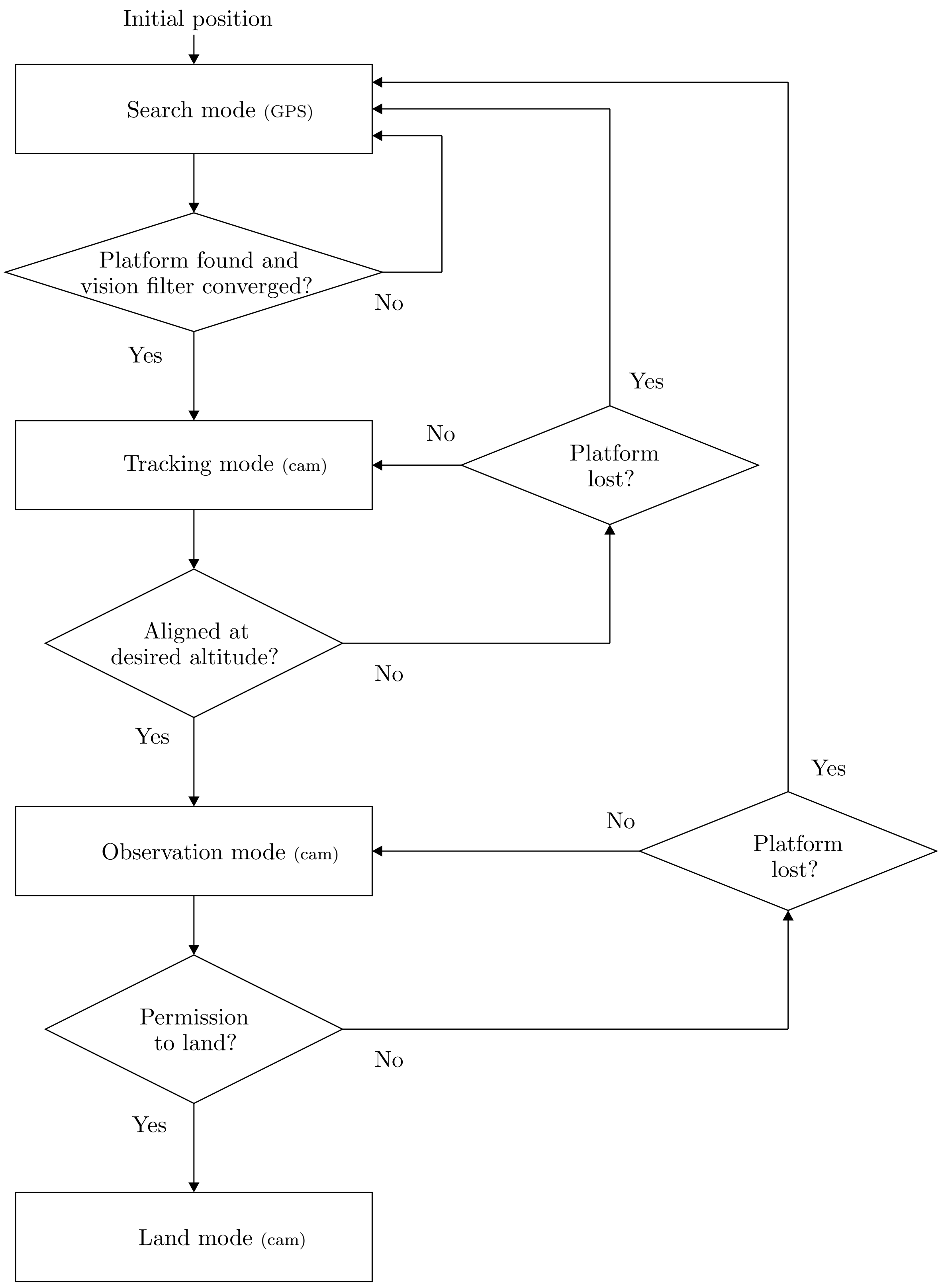


Figure 2-1: Block diagram of the autoland modes

Search mode

When flying at a random position above the sea, the quadrotor can only rely on the GPS coordinates of the platform's anchor location. Therefore, the first mode must be GPS navigation (Search mode) to return the quadrotor to the platform's general vicinity. Because of sea currents, the buoy is not expected to remain afloat above the anchor's exact coordinates. Nevertheless, since the platform's motion is restrained by a tether, a maximum range can be expected for the floating device's location. Based on the camera's FOV, the necessary altitude above Mean Sea Level (MSL), h_{search}, can then be estimated for this mode so that when the quadrotor is at the desired GPS coordinates, the platform is in sight. This altitude can be easily calculated as

$$h_{search} = \frac{R_p}{\tan\left(\min\{FOV\}/2\right)} \tag{2-1}$$

where R_p is the radius within which the platform can be found. Note that the FOV may differ in the horizontal and vertical directions. Therefore, the minimum value must be considered. Eq. (2-1) is for the ideal case that GPS measurements are very accurate. However, with the small and cheap receivers commonly used for these types of UAVs, the position measurement may be off by a few meters. This error must be taken into account and an altitude increment is necessary. Such an increment can be computed as

$$\Delta h_{search} = \frac{P_e}{2\tan\left(\min\{FOV\}/2\right)} \tag{2-2}$$

where P_e corresponds to the maximum window of position measurement error. For clarity, a visual representation of the scenario under consideration is presented in Figure 2-2. As an example, if $\min\{FOV\} = 60\,\text{deg}$, $R_p = 10\ m$ and $P_e = 2\ m$ then $h_{search} = 17.3\ m$ and $\Delta h_{search} = 1.73\ m$, meaning that the quadrotor would have to search for the buoy at an altitude of approximately 20 m.

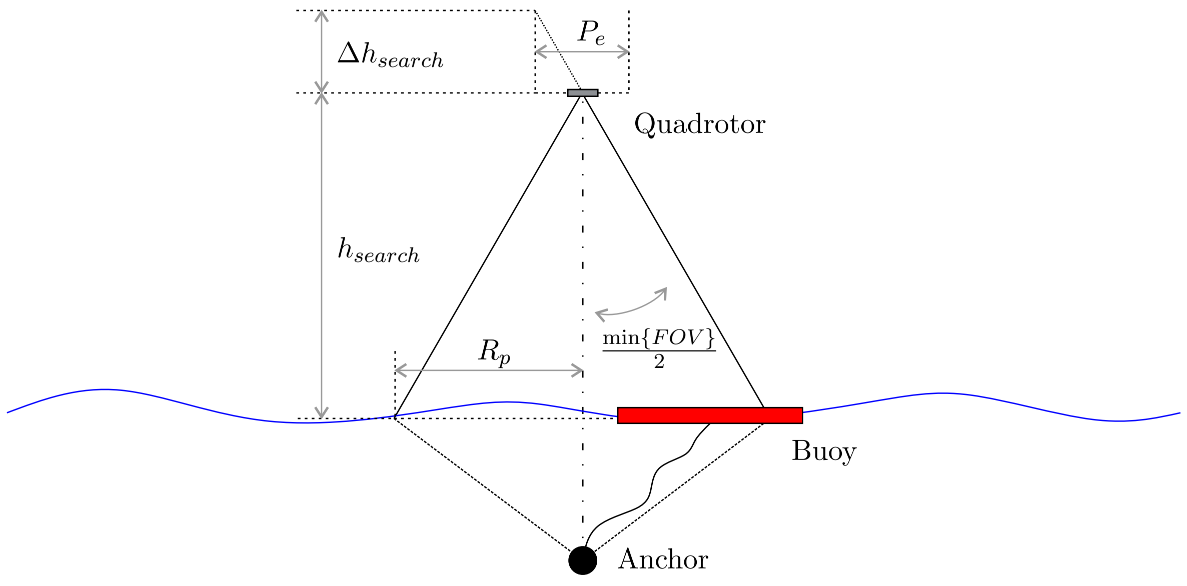


Figure 2-2: Search mode schematic representation

Note that the altitude obtained with Eqs. (2-1) and (2-2) could be too large to obtain clear images. In that case, this mode must be executed at a lower altitude and once the anchor's GPS coordinates have been reached, a search algorithm must be performed to find the buoy. As an example, the quadrotor could follow a predefined trajectory that would ensure complete coverage of the area.

It is important to stress that the altitude pressure should be reset every time a mission begins. Before releasing the quadrotor to perform a certain task, the pressure for that moment must be measured and used as the reference for further altitude calculations.

The switch to vision-based control is only possible when the platform is in sight. However, there is one key aspect to take into account: since the vision-based control requires state estimation through filtering, it is necessary to wait until the vision filter has converged and the estimations are reliable for use in the control loops. This allows for smooth transient behavior after switching from GPS to camera.

Tracking mode

Once the platform is in the camera's FOV and the vision-filter has converged, the Tracking mode is initiated. In this mode, the quadrotor should align with the platform's center (or designated landing area) and descend to a desired altitude. At this point, this altitude should be based on pressure (seen as inertial altitude) and should not yet be relative to the buoy. When aligned at the desired altitude above MSL, the quadrotor should hold its position and enter the Observation mode.

An algorithm to determine whether the platform has been lost is required in this mode. It is not realistic to assume that the platform is lost as soon as it has disappeared from one image frame. In fact, with rough weather conditions, it is expected that the target may disappear several times in the tracking process. Therefore, the platform is considered lost only when the target is not in sight after several consecutive frames (the exact number of frames should be tuned based on real-time testing). As illustrated in Figure 2-1, the natural step would be to return to Search mode and rely on GPS measurements to find the platform once again.

Observation mode

In the literature review, it was found that motion estimation of a moving target can be useful when the intention is to land a flying vehicle upon it. This concept of motion estimation is pursued in this thesis. A model of the floating platform must be constructed, and the Observation mode aims at finding the parameters of such a model online. An algorithm must also be developed to ensure that permission to land is only granted when all necessary requirements are fulfilled, that is, when the floating platform's motion estimation is producing reliable estimates to be used in the control laws.

Note that the algorithm to determine whether the platform has been lost is also run for this mode. Recalling the literature review, this solution poses an interesting alternative for finding lull opportunities for landing. If the quadrotor can manage to stay aligned with the platform during the Observation mode, then there is a high likelihood of landing upon it. If the quadrotor cannot maintain alignment, then the procedure must restart until a quiescent opportunity for landing arrives. This mostly concerns the buoy's horizontal motion, which is assumed as negligible for this study. Therefore, this method for finding lull opportunities would be more applicable in the case where lateral motion is not neglected.

Land mode

The Land mode is the final stage of the controller. This mode should ensure a TD sink rate as specified in the requirements, and the quadrotor should land as close to the desired point as possible.

Note that when the quadrotor is far from the buoy it is not necessary to control vertical velocity with respect to the target. In fact, it makes more sense to control vertical descent in the inertial frame, i.e., keeping MSL as the reference height. When the quadrotor is closer to the platform, the sink rate should then be controlled with respect to the target.

The algorithm for determining if the platform as been lost was intentionally excluded for this mode. When operating closer to the platform, there is a high probability of aborting the mission too frequently. Note that the camera used has a narrow FOV and some turbulence can be expected due to the platform's proximity; this in turn could conflict with the time constraint imposed by the battery levels. Nevertheless, such design choice can only be made after testing with the real quadrotor.

For the Land mode as well as for the Tracking and Observation modes, it is clear that the target cannot be the same size for when the quadrotor is far and when it is close to the platform. Therefore, the target must be designed in such a way that different shapes, colors, sizes or other features can be used depending on the quadrotor's height. For example, the buoy could have a circular shape forming a ring around its border and a square in the middle. A different solution to this problem could be to use lights to illuminate different parts of the platform as necessary. This solution would require minimal communication between the flying machine and the floating platform, which is expected since the quadrotor must be secured after TD and must also recharge batteries.

2-4 Required subsystems

From the strategy outlined to accomplish the autoland mission, a controller with several modes must be designed. First, a position controller is required; given desired coordinates (either from GPS or camera), the quadrotor must be able to navigate to a desired location and hold its position. Then, a mode to control the vertical velocity is needed; this mode must be adapted to ensure that the sink rate controlled can be relative to MSL or relative to the platform. To summarize, the following controller stages are required:

- **Inertial altitude hold.** This stage is required for the Tracking and Observation modes. The quadrotor should regulate its horizontal position to zero (with the target being the reference) and hold its altitude in the navigation frame (with respect to MSL).
- **Inertial descent.** This stage is required for landing purposes, for the case in which the quadrotor is not yet close to the buoy. Vertical velocity is then controlled, but with

respect to MSL. The control objective is still to regulate horizontal position to zero, but no vertical position is controlled.

- **Relative descent.** This stage is required for the final landing phase when the platform is close. Since the objective is to control relative sink rate with respect to the target, the inertial vertical velocity must be adjusted to cope with the buoy's dynamics. As for the previous stage, the vertical position is not controlled to zero and the objective is to regulate the horizontal position to zero.

The quadrotor is equipped with an IMU, an altitude pressure sensor and a camera. With the information gathered from this set of sensors, it is necessary to reconstruct all states required for control purposes. Therefore, a state estimation block based on filtering techniques must be implemented.

These two components - controller and state estimation blocks - are two major research topics for this thesis, and set the stage for developing solutions to control-related issues.

2-5 Docking system

To start, the best approach to designing a docking system is to study previous solutions implemented in projects from different fields. Perhaps the most state-of-the-art docking mechanism is implemented in the Automated Transfer Vehicle (ATV) project from the European Space Agency (ESA). The Integrated Cargo Carrier (ICC) is a spacecraft module that contains a system for docking with the International Space Station (ISS), and from the ESA website the following information can be found (http://www.esa.int/esaMI/ATV/index.html):

> The front cone of the ICC accommodates the 235 kg Russian docking system with its 80 cm-diameter hatch, its alignment mechanism and its one-metre-long extendible probe. During rendezvous with ISS, the ATV is the active spacecraft and is equipped with an arrow-shaped probe mechanism. The Space Station has receiving-cone mechanisms at the docking ports which are routinely used for Soyuz and Progress dockings. The Russian docking system, which has been continuously refined since it was originally developed in the late 1960s for the Salyut space station programme, remains the worldwide state-of-the-art in docking mechanisms.

A similar concept can be used as a solution for this quadrotor landing project. By designing the landing area with a receiving-cone-shaped geometry (see Figure 2-3), there is no demand for extremely precise landing (which is unrealistic for the system under consideration). The quadrotor could then slide to the middle of the platform where a docking mechanism would be prepared to secure the vehicle.

It should also be stressed that the quadrotor should be equipped with landing mechanisms to enhance TD safety. More specifically, the rotorcraft should possess legs in such a way that the actual TD point is far from the quadrotor's Center of Mass (CM). This will reduce the probability of flipping after one of the legs has touched the landing area. Finally, it should be mentioned that the height of these legs should permit landing even when the camera has

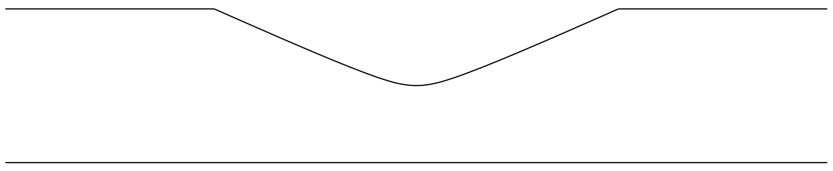


Figure 2-3: Section of the buoy corresponding to the cone-shaped landing area

a certain distance from the target. This is mainly due to the fact that the camera's FOV is limited and the chance of loosing the target is high when the quadrotor is extremely close to it.

Chapter 3

Quadrotor and environment models

A model of the quadrotor is necessary in order to facilitate controller design tasks as well as to validate the design concepts through simulations. In this thesis, the model used is based on those described by (Wierema, 2008) and (Berkelaar & Oonk, 2009). The environment is also modeled, including description of the atmosphere, wind profiles and motion of the floating platform.

3-1 Assumptions

Several assumptions were made for this model:

- The quadrotor is a rigid structure, possessing a very stiff frame.
- The quadrotor is symmetrical, and therefore the cross products in the inertia matrix are zero.
- Blade flapping effects of the rotor are neglected, as expected velocities are small.
- The thrust of the rotors is in the vertical direction of the quadrotor.
- The quadrotor has a constant mass.
- A flat, non-rotating Earth is also assumed, given that the flight times are short and velocities are small.

3-2 Reference frames

As is common practice in the aerospace field, the modeling procedure begins with the definition of reference frames used to describe the motion of the craft in space:

- The Earth-fixed reference frame $\mathcal{N}$, a right-handed orthogonal axis-system with origin at a pre-specified convenient point. This reference frame is fixed to the earth's local tangent plane and is considered as the inertial frame of reference under simplifying conditions. Its x-axis points North, its y-axis points East and its z-axis points down, making it a North East Down (NED) frame.
  - Alternatively, the Earth-fixed reference frame $\mathcal{E}$. This reference frame is similar to that described above, except for the fact that its x-axis points East, its y-axis points North and its z-axis points up, making it a East North Up (ENU) frame.
- The body-fixed reference frame $\mathcal{B}$, a right-handed orthogonal axis-system with origin at the quadrotor's center of gravity. This reference frame is fixed to the quadrotor's body. Its x-axis points forward, y-axis points to the right and z-axis points down.

A schematic representation of the reference frames $\mathcal{N}$ and $\mathcal{B}$ is given in Figure 3-1.

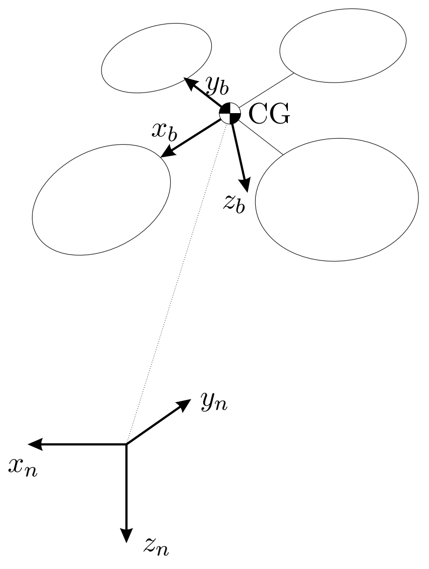


Figure 3-1: Representation of the inertial and body-fixed reference frames

The transformation between two reference frames is composed by

- translation of the origin from the first reference frame to the second, and
- a set of rotations that defines the difference in orientation of the two frames.

3-3 Attitude representation and rotations

In order to adequately build the quadrotor model, the angular attitude of the vehicle must be defined. For this project, the Euler angles representation is used. This representation, as implemented in this model, presents a singularity for the pitch angle at ±90 deg. However, since no acrobatic maneuvers are desired (or expected) during the approach and touchdown phases of the automatic landing procedure, this choice is suitable for modeling purposes.

Furthermore, the Euler angles have the advantage of being intuitive to interpret. Nevertheless, it is clear that the computational load using Euler angles is higher when compared to other representations such as the quaternion of rotation, which presents an extra element to fully describe the attitude, but is less nonlinear.

Now consider a right-handed orthogonal reference frame $\mathcal{A}$ and another denoted by $\mathcal{B}$, also right-handed and orthogonal. It is possible to rotate from $\mathcal{A}$ to $\mathcal{B}$ by performing three single-axis rotations. Each of these rotations can be described by the rotation matrix R as presented in Eqs. (3-1), (3-2) and (3-3), depending on the rotation axis.

For a rotation about the x-axis by φ_1, the rotation matrix is given by

$$R_x(\varphi_1) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi_1 & \sin\varphi_1 \\ 0 & -\sin\varphi_1 & \cos\varphi_1 \end{bmatrix} \tag{3-1}$$

For a rotation about the y-axis by φ_2, the rotation matrix is given by

$$R_y(\varphi_2) = \begin{bmatrix} \cos\varphi_2 & 0 & -\sin\varphi_2 \\ 0 & 1 & 0 \\ \sin\varphi_2 & 0 & \cos\varphi_2 \end{bmatrix} \tag{3-2}$$

And finally, for a rotation about the z-axis by φ_3, the rotation matrix is given by

$$R_z(\varphi_3) = \begin{bmatrix} \cos\varphi_3 & \sin\varphi_3 & 0 \\ -\sin\varphi_3 & \cos\varphi_3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3-3}$$

Note that these matrices are orthogonal, meaning that $R(\varphi)^{-1} = R(\varphi)^T$. The full transformation between reference frames can be broken down into the multiplication of these three basic matrices, which also results in an orthogonal matrix. It is possible to choose different sequences, but in aerospace (and flight dynamics in particular), one of the most common choices is to perform the so-called 3-2-1 sequence of rotation. For this rotation, the angles are defined as $[\varphi_1, \varphi_2, \varphi_3] = [\phi, \theta, \psi]$ (roll, pitch and yaw, respectively) and the sequence is described as follows:

- First, rotation about the z-axis by yaw angle (ψ);
- Second, rotation about the intermediate y-axis by pitch angle (θ);
- Third, rotation about the intermediate x-axis by roll angle (ϕ).

According to this formulation, the transformation $T^{\mathcal{NB}}$ from the NED frame $\mathcal{N}$ to the body-fixed frame $\mathcal{B}$ is given by:

$$T^{\mathcal{NB}} = R_x(\phi)R_y(\theta)R_x(\psi) = \begin{bmatrix} \cos\psi\cos\theta & \sin\psi\cos\theta & -\sin\theta \\ \cos\psi\sin\theta\sin\phi - \sin\psi\cos\phi & \sin\psi\sin\theta\sin\phi + \cos\psi\cos\phi & \cos\theta\sin\phi \\ \cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi & \sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi & \cos\theta\cos\phi \end{bmatrix} \tag{3-4}$$

The inverse transformation $T^{\mathcal{BN}} = T^{\mathcal{NB}^{-1}}$ (from the body-fixed to the NED reference frame) corresponds to $T^{\mathcal{NB}^{-1}} = T^{\mathcal{NB}^{T}}$, following the orthogonality property of the rotation matrix obtained.

3-4 Kinematic relations

The kinematic relations describe the motion of objects without considering the forces that cause the motion. The time derivatives of the Euler angles as a function of the angular rates expressed in the body-fixed reference frame are presented in this section. A sequence of three rotations (first yaw, then pitch and finally roll) is applied as follows:

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + R_x(\phi) \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + R_x(\phi) R_y(\theta) \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} \tag{3-5}$$

which leads to the following relation

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & -\sin\phi & \cos\phi\cos\theta \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{3-6}$$

from which the Kinematic Differential Equations (expressed in terms of the Euler Angles) can be obtained:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi/\cos\theta & \cos\phi/\cos\theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{3-7}$$

3-5 Dynamic equations of motion

The equations of motion of the quadrotor UAV can be derived using Newton's second law of motion, which can be expressed in the inertial, Earth-fixed reference frame by the two vector equations (J. A. Mulder, Staveren, Vaart, & Weerdt, 2006)

$$\underline{F} = \frac{d}{dt}(m\underline{V})\bigg|_E \tag{3-8}$$

$$\underline{M} = \frac{d\underline{H}}{dt}\bigg|_E \tag{3-9}$$

By expanding these two equations, it follows that

$$\begin{bmatrix} F_{x_{ext}} \\ F_{y_{ext}} \\ F_{z_{ext}} \end{bmatrix} = \begin{bmatrix} -mg\sin\theta \\ mg\cos\theta\sin\phi \\ mg\cos\theta\cos\phi \end{bmatrix} + \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} m(\dot{u} + qw - rv) \\ m(\dot{v} + ru - pw) \\ m(\dot{w} + pv - qu) \end{bmatrix} \tag{3-10}$$

$$\begin{bmatrix} M_{x_{ext}} \\ M_{y_{ext}} \\ M_{z_{ext}} \end{bmatrix} = \begin{bmatrix} L \\ M \\ N \end{bmatrix} = \begin{bmatrix} I_{xx}\dot{p} + (I_{zz} - I_{yy})qr \\ I_{yy}\dot{q} + (I_{xx} - I_{zz})rp \\ I_{zz}\dot{r} + (I_{yy} - I_{xx})pq \end{bmatrix} \tag{3-11}$$

where F_{xext}, $F_{y_{ext}}$ and F_{zext} correspond to the external forces acting on the quadrotor (expressed in the body-fixed reference frame), and M_{xext}, $M_{y_{ext}}$ and M_{zext} correspond to the external moments acting on the quadrotor (also expressed in the body-fixed reference frame).

To construct an accurate model from Eqs (3-11) and (3-10), the mass and inertia properties of the quadrotor must be known. The mass of the quadrotor can be accurately measured using a scale, and it is expected that fluctuation in this parameter will be negligible (unless changes are applied to the layout of the quadrotor). Determining the moments of inertia is a slightly more complex task. In this thesis, a new method to determine I_{xx}, I_{yy} and I_{zz} is presented in Chapter 4. Note the assumption that the cross products of the inertia matrix J can be neglected, due to the layout of the quadrotor.

$$J = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{3-12}$$

External forces

The external forces applied to the quadrotor can be summarized as follows:

- along the x_b-axis (X): hub forces and friction;
- along the y_b-axis (Y): hub forces and friction;
- along the z_b-axis (Z): thrust and friction.

External moments

The external torques applied to the quadrotor can be summarized as follows:

- about x_b (L): roll control action, rotor gyro effect, moment due to hub force and rolling moment due to forward flight;
- about y_b (M): pitch control action, rotor gyro effect, moment due to hub force and moment due to forward flight;
- about z_b (N): yaw control action, rotor gyro effect and hub force unbalance.

These forces and moments are explained in the next section.

3-6 External forces and moments

There are two types of aerodynamic effects considered for this model: those resulting from the rotors (rotating blades) and those associated with friction in the whole frame when the velocity of the quadrotor is non-zero. The external forces and moments generated by these aerodynamic effects are explained in this section. Furthermore, since for landing purposes the quadrotor will have to operate close to the platform, ground effect is also modeled. Finally, the torques caused by the rotor gyro effect are also given.

3-6-1 Rotor aerodynamics

Blade element theory (Seddon & Newman, 1990) is used to model the aerodynamic forces acting parallel and perpendicular to the rotor shaft, as well as the moments about the rotor shaft and the hub.

Before introducing the forces and moments generated by the rotors, it is first necessary to outline a few important concepts related to rotating blades.

- The rotor disc area is defined as
$$A = \pi R^2 \tag{3-13}$$
where R is the radius of the rotor.
- The rotor solidity is defined as the ratio between the total blade area and the rotor disc area. For a rotor composed by N blades with an equivalent chord c_e, this parameter is given by
$$\sigma = \frac{Nc_e}{\pi R} \tag{3-14}$$
- The inflow velocity, v_1, of the rotor (in hovering mode) can be described by
$$v_1 = \sqrt{-\frac{V^2}{2} + \sqrt{\left(\frac{V^2}{2}\right)^2 + \left(\frac{W}{2\rho A}\right)^2}} \tag{3-15}$$
where $V = \sqrt{u^2 + v^2}$ is defined as the lateral velocity, W is the weight of quadrotor and ρ is the air density.
- The total inflow ratio is given by
$$\lambda = \frac{v_1 - \dot{w}}{\Omega R} \tag{3-16}$$
where Ω is rotational speed of the rotor and $\dot{w}$ is the vertical speed of the quadrotor (in the z-body-axis).
- The advance ratio is
$$\mu = \frac{V}{\Omega R} \tag{3-17}$$
- The lift coefficient varies linearly with the angle of attack
$$C_L = a\alpha \tag{3-18}$$
where a is the lift slope.
- The twist of the rotor varies linearly with the radial position
$$\theta_{twist} = \theta_0 - \theta_{tw}\,(r/R) \tag{3-19}$$
where θ_0 is the rotor pitch at the hub and θ_{tw} is the linear pitch constant of the rotor.

The different forces and moments are now introduced as functions of these parameters.

Thrust force

The thrust force is the resultant force acting on all blade elements in the vertical direction.

$$F = C_T \rho A \left(\Omega R\right)^2 \tag{3-20}$$

$$\frac{C_T}{\sigma a} = \left(\frac{1}{6} + \frac{\mu^2}{4}\right)\theta_0 - \left(1 + \mu^2\right)\frac{\theta_{tw}}{8} - \frac{\lambda}{4} \tag{3-21}$$

Hub force

The hub force is the resultant force acting on all blade elements in the horizontal plane, as represented by

$$H = C_H \rho A \left(\Omega R\right)^2 \tag{3-22}$$

$$\frac{C_H}{\sigma a} = \frac{\mu \overline{C}_d}{4a} + \frac{1}{4}\lambda\mu\left(\theta_0 - \frac{\theta_{tw}}{2}\right) \tag{3-23}$$

where $\overline{C}_d$ is the airfoil drag coefficient at the 70 % radial station.

Drag moment

The drag moment is the resulting moment from the drag force on the rotor.

$$M_Q = C_{MQ} \rho A \left(\Omega R\right)^2 R \tag{3-24}$$

$$\frac{C_{MQ}}{\sigma a} = \frac{1}{8a}\left(1 + \mu^2\right)\overline{C}_d + \lambda\left(\frac{\theta_0}{6} - \frac{\theta_{tw}}{8} - \frac{\lambda}{4}\right) \tag{3-25}$$

Rolling moment

The rolling moment depends on the lateral velocity of the quadrotor. The advancing blade will produce more lift than the retracting blade, which will cause a moment.

$$M_R = C_{MR} \rho A \left(\Omega R\right)^2 R \tag{3-26}$$

$$\frac{C_{MR}}{\sigma a} = \mu\left(\frac{\theta_0}{6} - \frac{\theta_{tw}}{8} - \frac{\lambda}{8}\right) \tag{3-27}$$

3-6-2 Ground effect

When the quadrotor is flying near the ground, the thrust produced by the rotors increases. The distribution of airflow from the rotor is modified as a result of the cushion of air formed under the rotorcraft due to the proximity of the ground. This in turn leads to an increase in lift when compared to operation in Out of Ground Effect (OGE) conditions. This effect is known as the ground effect, and should be included in this research project since the landing procedure depends strongly upon it just before touchdown. In Ground Effect (IGE) conditions are usually found within heights about 0.5 to 1.0 times the diameter of the rotor. The ground effect is very difficult to model and it depends on many parameters such as the type of rotor, the slope and nature of the ground and any prevailing winds.

The model used to describe the ground effect relates the IGE thrust with the OGE thrust as

$$\frac{F_{IGE}}{F_{OGE}} = \frac{1}{1 - \left(\frac{R}{4Z_{ag}}\right)^2} \tag{3-28}$$

where Z_{ag} is the height above the ground (Cheeseman & Bennett, 1957).

Figure 3-2 shows the increase in thrust divided by the OGE thrust, as a function of height above the ground.

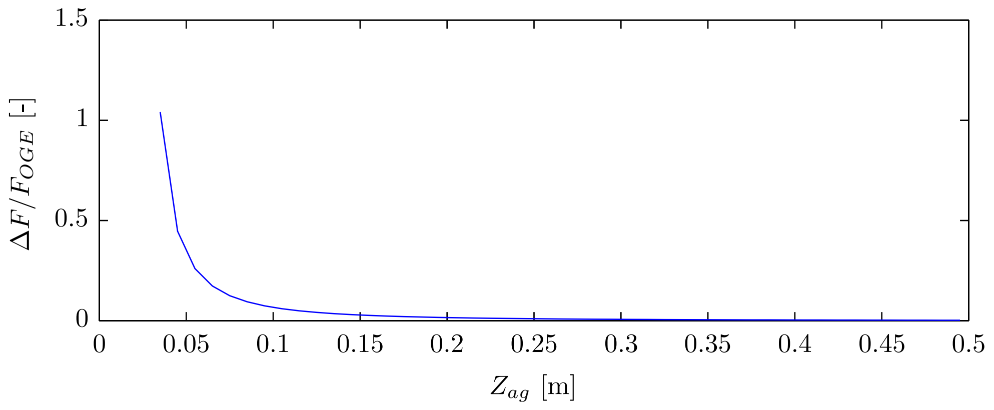


Figure 3-2: Ratio between increased thrust over F_{OGE} as a function of height above ground, for a rotor radius $R = 0.1\ m$

3-6-3 Structural friction

When the quadrotor is airborne, friction can be expected. This force is modeled according to

$$D = -\frac{1}{2} C_D A_D \rho \cdot V_b \cdot |V_b| \tag{3-29}$$

where C_D denotes the aerodynamic dimensionless drag coefficient, A_D is the aerodynamic area and V_b is the velocity expressed in the axis under consideration.

3-6-4 Rotor gyro effect

As the quadrotor has four rotating masses, a gyro effect is expected. This effect is not present when all rotors rotate at the same speed. However, the Clockwise (CW) and Counter Clockwise (CCW) rotors will rotate at different speeds for yaw angle changes. The moments generated due to the gyro effect are given in Eq. (3-30) where J_r is the inertia of all four rotors and Ω_r is defined as the residual rotational rate of the rotors:

$$\begin{bmatrix} M_{xge} \\ M_{yge} \\ M_{zge} \end{bmatrix} = J_r \begin{bmatrix} q\Omega_r \\ -p\Omega_r \\ \dot{\Omega}_r \end{bmatrix} \tag{3-30}$$

with

$$\Omega_r = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4 \tag{3-31}$$

3-7 Gathering the equations of motion

The equations of motion presented in the previous sections are now collected. The vector nomenclature is defined as presented below:

- inertial position: $\underline{p} = (x, y, z)^T$
- body velocity: $\underline{V} = (u, v, w)^T$
- attitude Euler angles: $\underline{\Theta} = (\phi, \theta, \psi)^T$
- body angular rates: $\underline{\omega} = (p, q, r)^T$

The differential equations describing the system are then given in vector form as:

- position loop:
$$\dot{\underline{p}} = T^{\mathcal{BN}}\underline{V} \tag{3-32}$$
- velocity loop:
$$\dot{\underline{V}} = -\underline{\omega} \times \underline{V} + m^{-1}\left(\underline{W} + \underline{F} + \underline{F}_d\right) \tag{3-33}$$
- attitude loop:
$$\dot{\underline{\Theta}} = N\underline{\omega} \tag{3-34}$$
- rate loop:
$$\dot{\underline{\omega}} = J^{-1}(-\underline{\omega} \times J\underline{\omega} + \underline{M} + \underline{M}_d) \tag{3-35}$$

In the above equations $\underline{F}$ contains the control forces, $\underline{F}_d$ contains the disturbance forces, $\underline{W}$ contains the gravity components, $\underline{M}$ contains the control moments and $\underline{M}_d$ contains the disturbance moments.

By expanding the above equations, twelve scalar first-order differential equations are then obtained:

$$\begin{aligned}
\dot{x} =&(\cos\psi\cos\theta)u + (\cos\psi\sin\theta\sin\phi - \sin\psi\cos\phi)v \\
&+ (\cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi)w \\
\dot{y} =&(\sin\psi\cos\theta)u + (\sin\psi\sin\theta\sin\phi + \cos\psi\cos\phi)v \\
&+ (\sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi)w \\
\dot{z} =&(-\sin\theta)u + (\cos\theta\sin\phi)v + (\cos\theta\cos\phi)w \\
\dot{u} =& rv - qw - g\sin\theta + \frac{1}{m}F_{xd} \\
\dot{v} =& pw - ru + g\cos\theta\sin\phi + \frac{1}{m}F_{yd} \\
\dot{w} =& qu - pv + g\cos\theta\cos\phi - \frac{1}{m}F_z + \frac{1}{m}F_{zd} \\
\dot{\phi} =&p + \tan\theta(q\sin\phi + r\cos\phi) \\
\dot{\theta} =&q\cos\phi - r\sin\phi \\
\dot{\psi} =&(q\sin\phi + r\cos\phi)/\cos\theta \\
\dot{p} =& \frac{1}{I_{xx}}\left[(I_{yy} - I_{zz})qr + M_x + M_{xd}\right] \\
\dot{q} =& \frac{1}{I_{yy}}\left[(I_{zz} - I_{xx})rp + M_y + M_{yd}\right] \\
\dot{r} =& \frac{1}{I_{zz}}\left[(I_{xx} - I_{yy})pq + M_z + M_{zd}\right]
\end{aligned} \tag{3-36}$$

Finally, to allow for testing of the controllers' responses to disturbances, the following forces and moments due to rotor gyro effect and structural drag are considered. Force disturbances are modeled as

$$\begin{aligned}
F_{xd} &= -\frac{1}{2}C_D A_D \rho \cdot u \cdot |u| \\
F_{yd} &= -\frac{1}{2}C_D A_D \rho \cdot v \cdot |v| \\
F_{zd} &= -\frac{1}{2}C_D A_D \rho \cdot w \cdot |w|
\end{aligned} \tag{3-37}$$

and the moment disturbances are modeled as

$$\begin{aligned}
M_{xd} &= J_r q \Omega_r \\
M_{yd} &= -J_r p \Omega_r \\
M_{zd} &= J_r \dot{\Omega}_r
\end{aligned} \tag{3-38}$$

Note that the hub forces, and consequently the corresponding moments, have been neglected, as well as moments due to sideways velocities.

3-8 Motor dynamics

For this model, DC motors are used. They are modeled using the known equations

$$\dot{\Omega}_m = -\frac{1}{\tau}\Omega_m - \frac{d}{\eta r^3 J_t}\Omega_m^2 + \frac{1}{k_m \tau}u_m \tag{3-39}$$

$$\frac{1}{\tau} = \frac{k_m^2}{R_m J_t} \tag{3-40}$$

where d is the drag factor, τ is the motor time constant, k_m is the motor torque constant, R_m is the motor internal resistance, η is the motor efficiency, J_t is the total rotor inertia seen by the motor, r is the reduction ratio and u_m is the motor input (Franklin & Emami-Naeini, 2006).

3-9 Sensor modeling

For on-board state estimation, Micro Electrical Mechanical Systems (MEMS) sensors are used to measure accelerations, angular rates and magnetic fields. A list of different errors associated with these types of sensors is given below.

- **Constant bias**
  The bias is an offset of the sensor's output from the true value. As the name indicates, it possesses a constant value and can be estimated by calculating the mean of a large sample of data. Note that this property is dependent upon external conditions such as temperature, meaning that bias drifts (explained below) can be expected.
- **Measurement noise**
  - white noise: the measurements are perturbed by a sequence of zero-mean random values.
  - flicker noise (bias drift): the bias of a sensor wanders over time due to flicker noise. This type of noise is mostly noticeable at low frequencies. At higher frequencies the white noise will, in general, be stronger than the flicker noise, thus explaining why it can be seen as a bias drift.
  - other types of noise such as correlated noise, quantization error and dither noise.
- **Nonlinearities**
  Sensors present nonlinear behaviors. Such effects are difficult to model and thus it is difficult to compensate for them. In short, the output of a sensor is nonlinear with respect to the quantity it is sensing. However, since the nonlinear effects are small, they can generally be neglected.
- **Misalignments**
  Given that the sensors are mounted on the vehicle, misalignments can be expected between the sensor's sensing axes and the vehicle's body axes. Furthermore, it is possible that the sensor axes are not orthogonal with respect to each other.

- **Scale factor error**
  The scale factor error is defined as a non-constant sensor gain that can result from aging or manufacturing tolerances.

What follows are the mathematical descriptions of the sensors used. The values used for simulation purposes are given in Appendix A.

Accelerometers

Accelerometers measure specific forces and it is assumed that the measurements have a constant bias and a normally-distributed white noise. It is not always possible to place the IMU at the Center of Gravity (CoG); therefore, a correction must be made. The distance from the IMU to CoG is defined as D_x, D_y and D_z. The acceleration caused by a non-zero displacement is given by Eq. (3-41)

$$\underline{A}_D = \underline{\omega} \times (\underline{\omega} \times \underline{D}) + \underline{\dot{\omega}} \times \underline{D} \tag{3-41}$$

By extending this equation it follows that

$$\begin{bmatrix} A_{xD} \\ A_{yD} \\ A_{zD} \end{bmatrix} = \begin{bmatrix} q(pD_y - qD_x) - r(rD_x - pD_z) + \dot{q}D_z - \dot{r}D_y \\ r(qD_z - rD_y) - p(pD_y - qD_x) + \dot{r}D_x - \dot{p}D_z \\ p(rD_x - pD_z) - q(qD_z - rD_y) + \dot{p}D_y - \dot{q}D_x \end{bmatrix} \tag{3-42}$$

and the total to-be-sensed acceleration is given by

$$\begin{bmatrix} A_x \\ A_y \\ A_z \end{bmatrix} = \begin{bmatrix} \dot{u} + qw - rv + g\sin\theta + A_{xD} \\ \dot{v} + ru - pw - g\cos\theta\sin\phi + A_{yD} \\ \dot{w} + pv - qu - g\cos\theta\cos\phi + A_{zD} \end{bmatrix} \tag{3-43}$$

The measurement equations are then

$$\begin{bmatrix} A_{xm} \\ A_{ym} \\ A_{zm} \end{bmatrix} = \begin{bmatrix} A_x + \lambda_{A_x} + w_{A_x} \\ A_y + \lambda_{A_y} + w_{A_y} \\ A_z + \lambda_{A_z} + w_{A_z} \end{bmatrix} \tag{3-44}$$

with

$$\begin{bmatrix} \dot{\lambda}_{A_x} \\ \dot{\lambda}_{A_y} \\ \dot{\lambda}_{A_z} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{3-45}$$

where λ_i represent constant biases and w_i are white noises.

Gyroscopes

Similarly, the gyroscopes' measurements are modeled as

$$\begin{bmatrix} p_m \\ q_m \\ r_m \end{bmatrix} = \begin{bmatrix} p + \lambda_p + w_p \\ q + \lambda_q + w_q \\ r + \lambda_r + w_r \end{bmatrix} \tag{3-46}$$

with

$$\begin{bmatrix} \dot{\lambda}_p \\ \dot{\lambda}_q \\ \dot{\lambda}_r \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{3-47}$$

where, again, λ_i represent constant biases and w_i are white noises.

Magnetometer

Magnetometers measure magnetic fields. Specifically, for this thesis a model of a two-axis magnetometer is used to measure the Earth's magnetic field, which for a specific location has a certain inclination and declination. It is assumed that the exact values of these two properties are known.

The on-board IMU measures magnetic components in the body-fixed reference frame. Therefore, the Earth's magnetic field is sensed in the body axes, from which a measurement of the ψ angle can be obtained. For details see (Wierema, 2008) and (Berkelaar & Oonk, 2009).

It is assumed that a yaw measurement is then available containing only white noise as

$$\psi_m = \psi + w_\psi \tag{3-48}$$

Note that it is assumed that the measurements can be perfectly corrected for all sensor biases.

Barometric altimeter

It is assumed that the barometric altimeter provides a measurement of height, $h = -z$, based on local static pressure measured. It is assumed that no bias exists and that the sensor's output is contaminated by zero-mean random noise as

$$h_m = -z + w_h \tag{3-49}$$

Camera

Finally, the camera must be modeled. A detailed description of this sensor will be given in Chapter 5. In the meantime, a short introduction is presented here for convenience. Initially, the vision system measures the quadrotor's position relative to the target. These measurements are in the body-fixed reference frame and must be corrected using attitude estimation. The final measurements are then given by

$$\begin{bmatrix} x_m \\ y_m \\ z_m \end{bmatrix} = T^{\mathcal{BN}}(\phi, \theta, \psi) \begin{bmatrix} x_{cam} \\ y_{cam} \\ z_{cam} \end{bmatrix} \tag{3-50}$$

where x_{cam}, y_{cam} and z_{cam} represent the initial camera measurements expressed in the body-fixed reference frame and are assumed as perfectly known (no bias and no white noise). The rotation matrix $T^{\mathcal{BN}}$ has been explained in Section 3-3.

It should be stressed that the camera has a limited FOV; this means that when the target does not appear in the image, it follows that no measurement is available. Note that the FOV is different in the horizontal and vertical directions.

Furthermore, it is assumed not only that the update rate from this sensor is lower when compared to the other sensors, but also that the measurements contain a significant time delay.

3-10 Environment model

To simulate conditions at sea, a model of the environment is required. This includes the atmosphere model, wind profiles and floating platform motion.

3-10-1 Atmosphere model

To describe the atmosphere, the International Standard Atmosphere (ISA) model is used. In this model, the Earth's atmosphere parameters are computed as a function of the height above the MSL as follows:

$$\rho = \rho_0 \left(1 + \lambda_{trop} \frac{h}{T_0}\right)^{\left(-\frac{g}{R\lambda_{trop}} - 1\right)} \qquad \left[kg/m^3\right] \tag{3-51}$$

where

$$\begin{aligned} \rho_0 &= 1.225\ kg/m^3 \\ \lambda_{trop} &= -0.0065\ K/m \\ T_0 &= 288.15\ K \\ R &= 287.05 m^2/s^2\ K \end{aligned} \tag{3-52}$$

The computation of the magnitude of gravity as a function of height is determined according the following expression

$$g = 9.80665 \left(\frac{R_e}{R_e + h}\right)^2 \tag{3-53}$$

where R_e is the average radius of the Earth ($R_e = 6367434m$). Note that Eq. (3-51) is valid for $h \leq 11km$, which will always be the case for this specific automatic landing mission.

3-10-2 Wind profiles

The vertical distribution of the wind, or wind shear, over the water surface is an important factor to consider for the simulation being performed. A power-law wind profile (Hsu, Meindl, & Gilhousen, 1994) is employed to simulate the atmospheric surface boundary layer extending to 100 m above the sea surface. This law states that

$$\frac{V_{wind}}{V_{wind,ref}} = \left(\frac{h}{h_{wind,ref}}\right)^P \tag{3-54}$$

where V_{wind} is the wind speed at height h, $V_{wind,ref}$ and $h_{wind,ref}$ are the wind speed and height already known, respectively, at a reference height, and the exponent P is a function of both the atmospheric stability in the layer over which P is determined to be valid and the underlying surface characteristics. For the open sea, the exponent P is chosen as 0.10 (Hsu et al., 1994). In Figure 3-3, a representation of the Earth boundary layer model is given for $h_{wind,ref} = 100\ m$. Turbulence and wind gusts are not modeled in this study. However, it

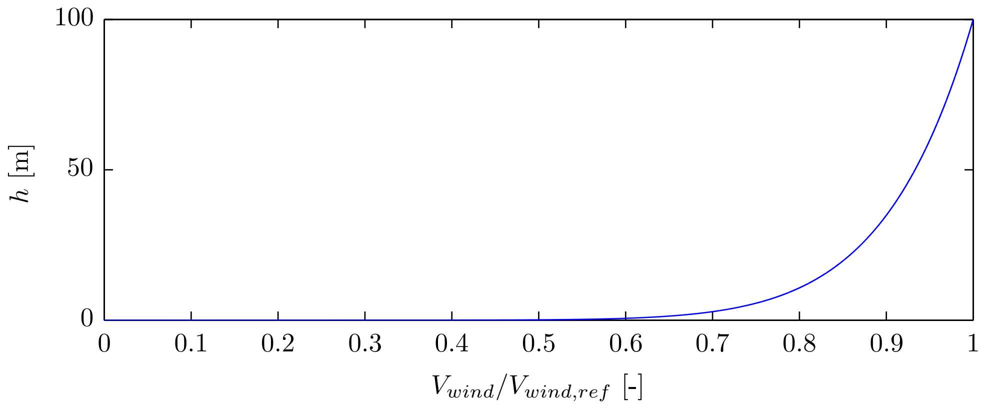


Figure 3-3: Earth boundary layer model for $h_{wind,ref} = 100\ m$

would be interesting to analyze the quadrotor's response to atmospheric stochastic turbulence. Models can be derived for this type of analysis, and this could be suggested as a topic for further research.

3-10-3 Floating platform motion

The motion of a floating platform is similar to ship motion. Although accurate models describing ship motion dynamics are widely available, developing a highly accurate model of the floating platform is outside the scope of this thesis. A simple model is therefore developed to allow for testing of the state estimation algorithms and control laws developed. Note that the most important aspect to be modeled is the fact that a floating platform is a moving target with vertical oscillatory behavior.

A common model of the vertical motion of a floating platform in high seas is given by (Marconi, Isidori, & Serrani, 2002) and (Herisse, Hamel, Mahony, & Russotto, 2012)

$$z_{plat}(t) = \sum_{i=1}^{n} A_i \cos\left(2\pi/T_i \cdot t + \phi_i\right) \tag{3-55}$$

where A_i, T_i and ϕ_i are unknown constants. As the platform is anchored, it is assumed that it has no motion in the x and y directions. An example is shown in Figure 3-4.

3-11 Modeling uncertainties

No system can be described mathematically with absolute certainty. In this section, different uncertainties are modeled in order to allow for evaluation of the control laws in the case of

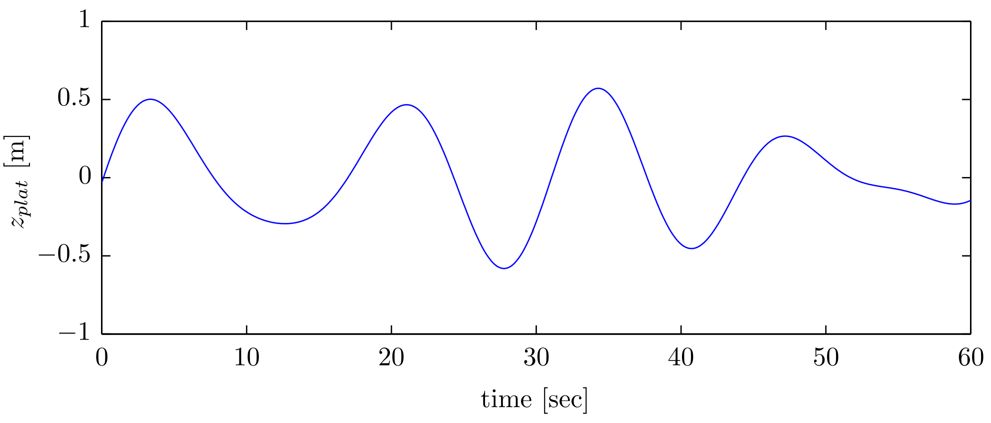


Figure 3-4: Example of the vertical motion of a floating platform

model mismatches. Each uncertainty is introduced via a multiplicative term ϵ, meaning that no uncertainty exists when $\epsilon = 1$. As an example, if a parameter is modeled with a deviation of 25% from its real value, it follows that $\epsilon = 1.25$.

3-11-1 Mass and inertia uncertainties

The real mass m of the quadrotor is different from the modeled mass m_0 by a multiplicative term. This relation is expressed mathematically by

$$m = m_0 \epsilon \tag{3-56}$$

Similarly, the real inertia properties of the quadrotor, present in the matrix J, differ from those modeled, J_0, by an multiplicative term. This relation is given by

$$J = J_0 \epsilon \tag{3-57}$$

3-11-2 Actuator uncertainties

One of the greatest difficulties in quadrotor modeling is to accurately describe the complex process in the motor/rotor that generates the aerodynamic forces and moments. The corresponding model uncertainties are simulated by introducing inaccuracies in the actuator parameters using the formulas

$$k_m = k_{m0} \epsilon \tag{3-58}$$

and

$$\tau = \tau_0 \epsilon \tag{3-59}$$

The first corresponds to an uncertainty in the motor torque constant, and the second to an uncertainty in the motor time constant.

3-11-3 Rotor gyro effect uncertainty

The gyro effect produces extra moments in all three axes as a function of the residual rotational rate of the motors. However, since these torques are not measured, nor are the rotors' speeds, it is interesting to analyze the system's response to increased gyro effect torques. The uncertainty model is given as

$$\underline{M}_{ge} = \underline{M}_{ge_0}\epsilon \tag{3-60}$$

Note that this model change may be introduced, for instance, when replacing the rotor blades. Furthermore, it should be stressed that although this uncertainty corresponds to a model mismatch, it can be seen as an external disturbance torque that must be rejected by the controller.

3-12 Matlab/Simulink implementation

The model described in this chapter is implemented in the Matlab/Simulink environment. The simulation tool is designed in a modular form, which allows for easy changes in different parts. The parameters chosen are identical to those used by (Wierema, 2008) and (Berkelaar & Oonk, 2009); however, since certain parameters of the test quadrotor used for this thesis are known (or relatively known), adjustments were made in order to build a more realistic model.

Chapter 4

Moments of inertia experiment

This chapter presents a procedure to experimentally determine the moments of inertia of a small quadrotor, using a two-axis motion simulator with predefined rotational motion and a six-component force/torque sensor. The inertia properties of the flying vehicle were also determined analytically using two modeling approaches: point mass analysis and assumption of simple geometric shapes.

4-1 Introduction

Calculation of a small quadrotor's moments of inertia is often performed using point mass analysis, as for instance in (Altuğ, Ostrowski, & Taylor, 2005) and (Pounds et al., 2006). The heaviest components, the motors, are modeled as point masses lying at a fixed distance from the quadrotor's center of mass. Alternatively, the entire structure can be seen as an assembly of various mass components with known geometric shapes (McKerrow, 2004). The arms can be modeled as slender rods, the motors as cylinders and the battery as a rectangular parallelepiped. These types of strategies can be improved when using computer-aided design (CAD) software (Pegram & Anemaat, 2000).

Using experimental methods to calculate the inertia properties of flying vehicles is not a new concept. In fact, in 1930, Miller published a technical note offering an apparatus and procedure based on pendulum theory to accurately determine moments of inertia of an airplane (Miller, 1930). In recent years, many researchers have proposed and tested similar experimental methods or variants to calculate moments of inertia not only of flying machines, but also of different kinds of objects with unknown mass distribution (Da Lio, Doria, & Lot, 1999; Storozhenko, 2003; Previati et al., 2004; Genta & Delprete, 1994; Podhorodeski & Sobejko, 2005; Bussamra, Vilchez, & Santos, 2009). These studies were mostly based on pendulum theory using multifilar or torsional pendula, and used measurements of the oscillation period of a suspended object. While the aforementioned techniques drew upon time-domain analysis, some frequency-domain techniques have also been implemented based on the measurements of frequency response functions (Lamontia, 1982; Almeida, Urgueira, & Maia, 2007; Mucchi, Fiorati, Di Gregorio, & Dalpiaz, 2011).

In this thesis, an innovative method is developed to determine the inertia properties experimentally. A two-axis motion simulator is used to generate rotational motion and a six-component sensor is used to measure forces and torques. A detailed description of the experiment is presented, including explanations of the concept, test results and data analysis. The research carried out is innovative in the sense that it takes a different approach from the classical methods; rather than measurements of periodic oscillation, it utilizes measurements of forces and torques directly.

4-2 Mathematical framework

4-2-1 Mathematical formulation for experimental inertia determination

The rotational dynamic equations of motion become linear when the coupling terms are eliminated. For example, for the yaw motion, if the body rotational rates p and q are zero, then

$$I_{zz}\dot{r} = M_z \tag{4-1}$$

where I_{zz} is the inertia component around the vertical axis, r is the rotational rate around the same axis and M_z is the external torque applied. This situation is difficult to achieve in flight, but can be created using the predefined rotation motion of an Acutronic two-axis motion simulator (see Figure 4-1).

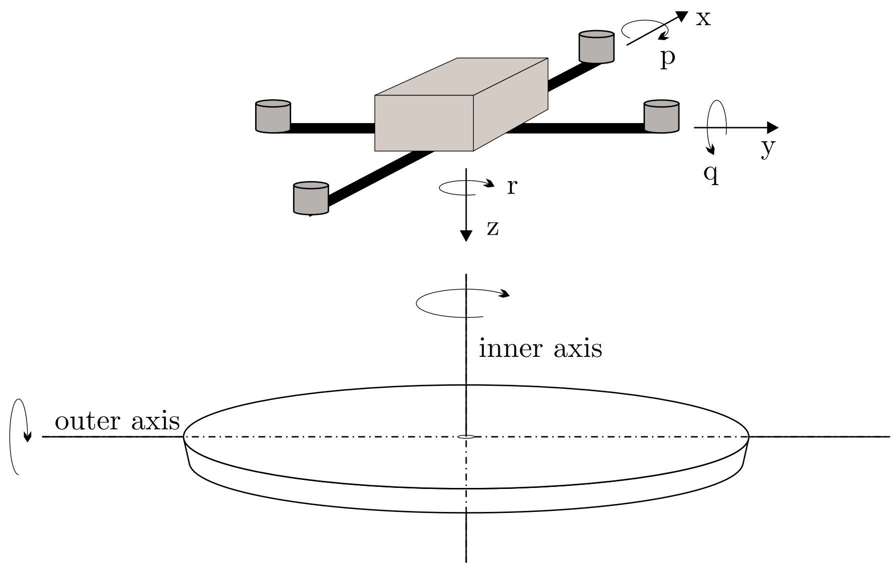


Figure 4-1: Schematic representations of the quadrotor and two-axis motion simulator

By placing a force/torque sensor between the two-axis motion simulator and the quadrotor, it is possible to measure the forces and moments applied at the body frame's center of mass, which result from the rotation of the table. Note that it is assumed that it is possible to align the quadrotor's vertical axis that passes through its center of mass with the two-axis motion simulator's vertical axis. Eq. (4-1) can be generalized for all three axes and is formulated for this experiment as

$$T_{rot} = I\ddot{\vartheta} \tag{4-2}$$

where T_{rot} denotes, for convenience, the rotational component of the torque to be measured, I is the moment of inertia of the corresponding axis and $\ddot{\vartheta}$ is the angular acceleration of the table (specified as input). Note that Eq. (4-2) is valid for the quadrotor's center of mass.

All torque sensor axes can be aligned in such a way that they are parallel to the body axes. However, it is not possible to place the sensor at the center of mass of the quadrotor. The z-axis of the sensor can be aligned with the z-axis of the body frame, but the same does not hold for x and y. These two sensor axes are at a certain distance from the x and y axes of the quadrotor. As represented in Figure 4-2, the sensor will thus measure a torque, T_{disp}, due to a vehicle weight component when the angle of rotation of the outer axis of the table (that is, the horizontal axis) is non-zero. It follows that

$$T_{disp} = d \cdot (W + W_{plate}) \sin\vartheta = d \cdot W_T \sin\vartheta \tag{4-3}$$

where d is the distance from the sensor axes to body axes (distance to center of mass) and $W_T = W + W_{plate}$ is the sum of the weights of the vehicle and the top mounting plate.

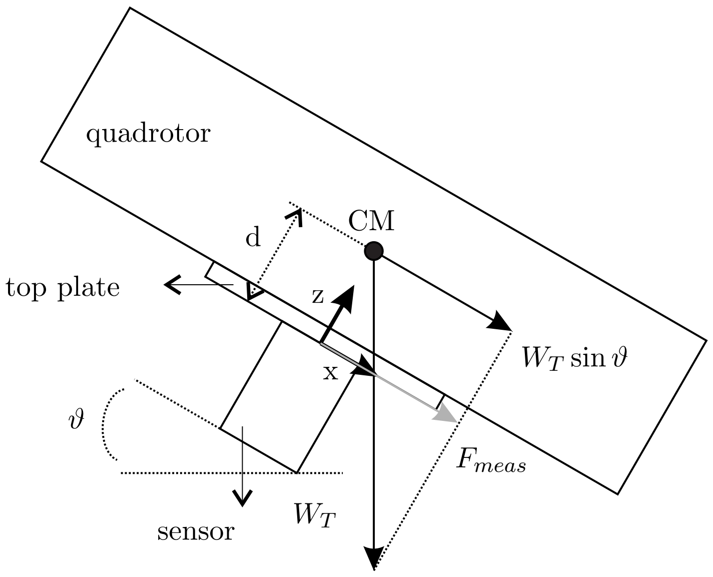


Figure 4-2: Measured torque due to weight component in the horizontal direction of the sensor

Note that $W_T \sin\vartheta$ can be obtained experimentally, F_{meas}, since the sensor used can measure forces in all three axes. The only unknown parameter is d. The total torque measured for this case is then given by

$$T_{meas} = T_{rot} + T_{disp} = I\ddot{\vartheta} + d \cdot F_{meas} \tag{4-4}$$

Estimations of the quadrotor's moments of inertia can then be obtained as follows:

- For the z-axis (I_{zz}):

$$I_{est} = \frac{T_{meas}}{\ddot{\vartheta}} \quad , \quad for \quad \ddot{\vartheta} \neq 0 \tag{4-5}$$

- For the x-axis (I_{xx}) and y-axis (I_{yy}):

$$I_{est} = \frac{T_{meas} - d_{est} \cdot F_{meas}}{\ddot{\vartheta}} \quad , \quad for \quad \ddot{\vartheta} \neq 0 \tag{4-6}$$

These two equations summarize this paper, as they provide the basic concept for estimation of the quadrotor's moments of inertia I_{xx}, I_{yy} and I_{zz}. F_{meas} and T_{meas} are measured forces and torques, $\ddot{\vartheta}$ is the known angular acceleration of the two-axis motion simulator and d_{est} is the to-be-estimated vertical distance between the sensor and the quadrotor's center of mass.

4-2-2 Mathematical formulation for theoretical moment of inertia analysis

The moment of inertia of a body about an axis of rotation is defined by the following equation(Török, 2000):

$$I = \int_M r^2 dM \tag{4-7}$$

The elements of mass, dM, are located at a certain distance r from the axis of rotation, and integration is performed over the entire mass of the body, M. For a body that can be broken down into an assemblage of n bodies (parts), the total moment of inertia can be determined by summing the inertia contributions of each part:

$$I = \sum_{k=1}^{n} I_k \tag{4-8}$$

There are multiple resources available which give the mass moments of inertia of simple body shapes about their own centroid axes, $\overline{I}$. To determine the inertia about the body axes, the parallel axis theorem must be used as follows

$$I_k = \overline{I}_k + M_k d_k^2 \tag{4-9}$$

where d_k is the perpendicular distance from the centroidal axis of the part to the body rotation axis.

Combining Eqs. (4-8) and (4-9) leads to:

$$I = \sum_{k=1}^{n} \left(\overline{I}_k + M_k d_k^2 \right) \tag{4-10}$$

Assuming point masses

The heaviest parts of the quadrotor are the motors. With this method, the helicopter is modeled as four point masses with mass M_{motor}. The body frame's center of mass is therefore at the center of the four point masses, at the height corresponding to the middle point of the motors. A point mass has no inertia around its center of mass. Thus, as each of the motors is at a distance l_{arm} from the quadrotor's center of mass, the inertia properties are given by

$$I_{xx} = I_{yy} = 2 \left(M_{motor} \cdot l_{arm}^2 \right) \quad , \quad I_{zz} = 4 \left(M_{motor} \cdot l_{arm}^2 \right) \tag{4-11}$$

Assuming geometric shapes

Alternatively, the quadrotor can be seen as an assembly of three main groups of components, namely: motors, arms and the core, which contains the autopilot board, sensors, cables and other hardware. Note that the battery is excluded in this configuration. All components can be modeled as simple geometric shapes with known inertia (Meriem & Kraige, 1998), as shown in the schematic representation in Figure 4-1.

- The four motors are modeled as cylinders with mass M_{cyl}, radius r_{cyl} and length l_{cyl}. The moments of inertia at the center of mass of each of these components are then given as
$$I_{c,c.a.} = \frac{1}{2}M_{cyl} \cdot l_{cyl}^2 \quad , \quad I_{c,c.d.} = \frac{1}{4}M_{cyl} \cdot r_{cyl}^2 + \frac{1}{12}M_{cyl} \cdot l_{cyl}^2 \tag{4-12}$$
respectively about the central axis and central diameter.

- The two arms are modeled as slender rods, each with mass M_{rod} and length l_{rod}. The moments of inertia at their centers of mass are then given by
$$I_r = \frac{1}{12}M_{rod} \cdot l_{rod}^2 \tag{4-13}$$
about the axis that passes through the center, perpendicular to the length. The inertia about the length axis is negligible.

- The core is modeled as a parallelepiped with mass M_{core}, dimensions a_{core}, b_{core} (for the base) and h_{core} (height). The moments of inertia at the center of mass are then given by
$$I_p = \frac{1}{12}M_{core}\left(a_{core}^2 + b_{core}^2\right) \tag{4-14}$$
about the vertical axis. For the other axes, the same formula is used, but with different length parameters.

The quadrotor's center of mass is calculated assuming a geometric configuration. Due to the body symmetry, x_{CM} and y_{CM} are at the center point of the body frame, while z_{CM} is given by
$$z_{CM} = \frac{4z_{cm,cyl} \cdot M_{cyl} + 2z_{cm,rod} \cdot M_{rod} + z_{cm,core} \cdot M_{core}}{4M_{cyl} + 2M_{rod} + M_{core}} \tag{4-15}$$
With the estimation of the distance from each component's center of mass to the center of mass of the entire structure, Eq. (4-10) can be used to determine the quadrotor's moments of inertia.

The quadrotor parameters to perform the above calculations are given in Appendix A.

4-3 Materials and equipment

The materials and equipment used for this experiment are listed below:

- AC2266L two-axis motion simulator from Acutronic

- ATI Force/Torque sensor: NANO 17
- Data acquisition device and connecting cables
- PCMCIA card
- Data storage device
- Top mounting metallic plate
- Bottom mounting metallic plate
- M2 bolts for mounting plates
- Bolts and blocks for clamping

4-4 Experiment measurement procedure

4-4-1 Mounting

The mounting procedure is explained in this section. First, the force/torque sensor is screwed to the top and bottom plates. Then, the bottom plate is clamped to the surface of the two-axis motion simulator. To conclude the mounting procedure, the quadrotor is screwed to the top plate. Figure 4-3 illustrates the mounted setup of all components. Note that different types of screws are used depending on their functionality (either countersunk or cheese head, as illustrated).

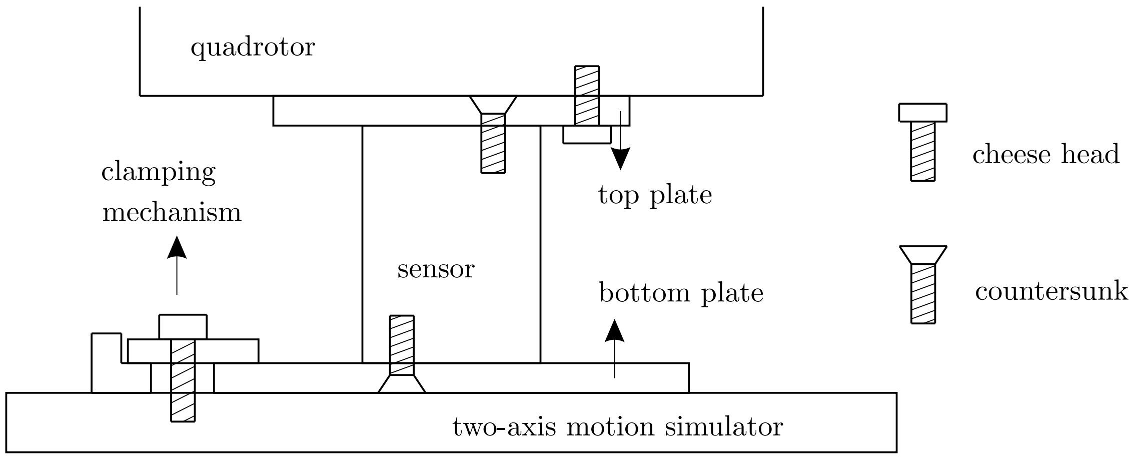


Figure 4-3: Scheme of mounting procedure

To obtain more reliable results, it is necessary to attempt full alignment of the z-axis of the quadrotor with the inner (vertical) axis of the two-axis motion simulator and y-axis (or x-axis) of the quadrotor with the outer (horizontal) axis of the two-axis motion simulator. Furthermore, the body axes of the quadrotor must be parallel to the axes of the sensor. This can be achieved by designing the holes in the top mounting plate in such a way that a perfect match is obtained. Note also that since the two-axis motion simulator will rotate during the

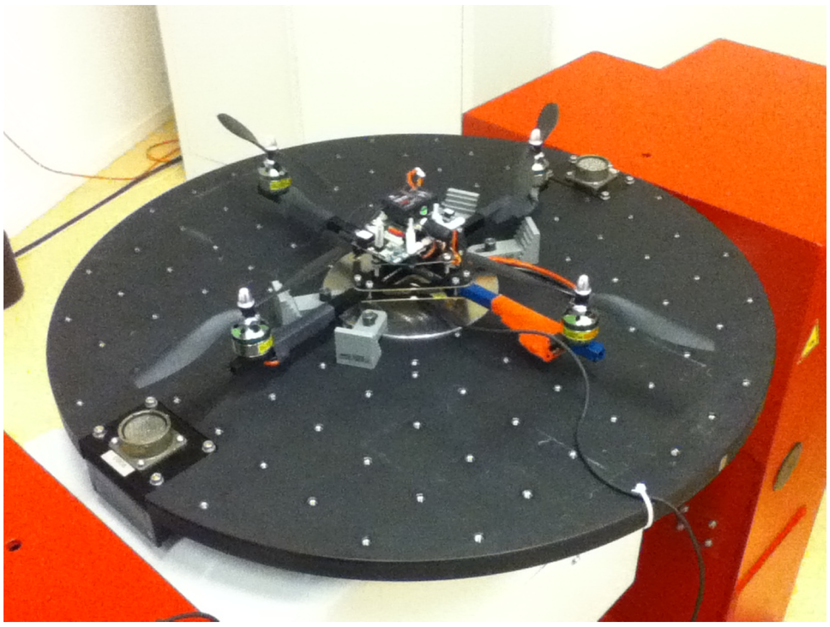


Figure 4-4: Quadrotor mounted on the turn table ready for testing

experiment, the sensor cable must be safely positioned within the mounting setup. The real setup ready for testing is shown in Figure 4-4.

The sensor is connected to the data acquisition device, which sends data to a laptop through a PCMCIA card. All logs are saved for post-data treatment as text files in the format of comma-separated values.

4-4-2 Determining sensor properties

Measurements are performed while the system is not in motion. This allows sensor properties such as biases and noise characteristics to be studied. Furthermore, it is possible to observe whether the sensor has a drift; that is, a small change of the biases over a certain period of time. The sensor has a resolution of 1/32 N mm for torques and 1/160 N for forces and samples were obtained at a frequency of 1000 Hz. From Figure 4-5 it can be determined that the sensor exhibits non-zero biases for all three directions. In addition, the measurements resemble white noise signals. Table 4-1 presents mean values (biases) as well as standard deviations of these signals. A drift in the biases is evident between the beginning and end of the experiment (a duration of a few hours). This implies that the biases must be estimated separately for each run.

4-4-3 Experiment runs

After mounting all components and testing the system with no motion, the experiment can be run. The procedure is as follows.

- Measure torques in the z-axis. Input signals are first given for the inner axis of the

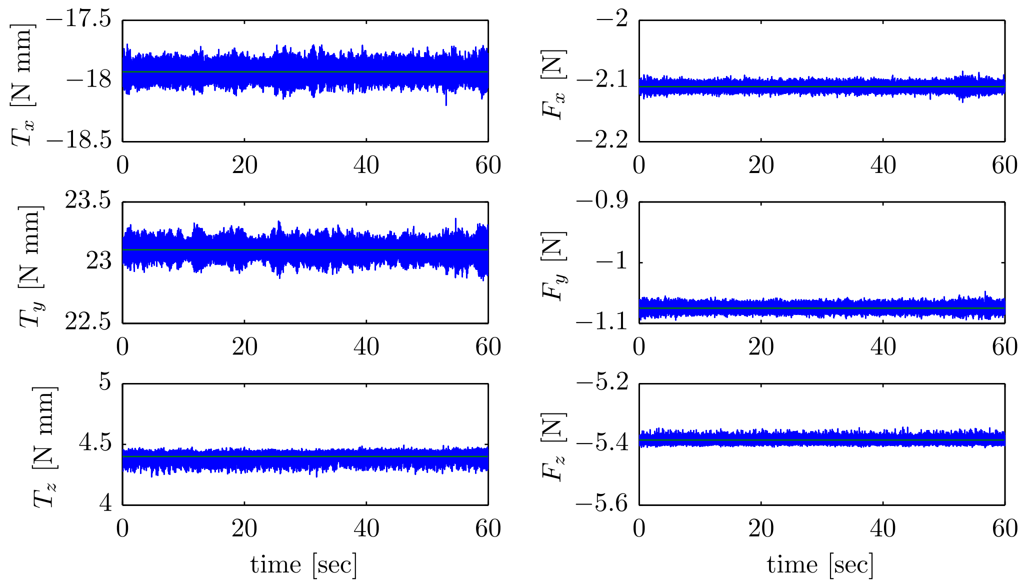


Figure 4-5: Sensor measurements with no motion at the beginning of the experiment

Table 4-1: Biases and standard deviations of the measured signals when turn table has no motion; 'beginning' and 'end' indicate values from before and after the experiment was run

	Mean (in N mm)			Mean (in N)		
	T_x	T_y	T_z	F_x	F_y	F_z
beginning	-17.9243	23.1063	4.4001	-2.1093	-1.0748	-5.3857
end	-18.5475	25.2510	6.5631	-1.9805	-1.1425	-5.4136
	Std (in N mm)			**Std (in N)**		
	T_x	T_y	T_z	F_x	F_y	F_z
beginning	0.0033	0.0035	0.0005	$2.14 \cdot 10^{-5}$	$2.36 \cdot 10^{-5}$	$5.85 \cdot 10^{-5}$
end	0.0045	0.0071	0.0008	$6.01 \cdot 10^{-5}$	$2.80 \cdot 10^{-5}$	$8.88 \cdot 10^{-5}$

two-axis motion simulator, leading to a rotation over its vertical direction with a known sinusoidal angular velocity.

- Measure torques in the y-axis. Input signals are then given for the outer axis of the two-axis motion simulator, leading to a rotation over its horizontal direction with a known sinusoidal angular velocity. A possible misalignment between the outer axis of the motion simulator and the y-axis of the quadrotor (and consequently of the sensor as well) results in a measured sinusoidal component on the x-axis. Therefore, it is necessary to rotate the two-axis motion simulator over its vertical axis by small angle increments and repeat the measurements until the sinusoidal component on the x-axis has vanished and only noise is measured. Forces in the x-direction are also measured for center of mass correction.

- Measure torques in the x-axis. Since the best alignment is encountered in the previous step, it is only necessary to rotate the two-axis motion simulator 90 degrees over its inner axis and perform measurements for the x-axis. Forces in the y-direction are also measured for center of mass correction.

- Remove quadrotor to measure moments of inertia of the top plate. The moments of inertia measured in the previous runs include the inertia of the top mounting plate. The testing procedure is then repeated, after removing the quadrotor. The experiment is performed for both the vertical and horizontal axes. Note that since the plate is circular, the inertia for x and y will be identical.

It is important to emphasize that data is only logged once the rotation is steady, which means that no data is acquired while the two-axis motion simulator is not stable on the final desired sinusoidal angular velocity profile.

4-4-4 Test signals

The rotation angle of the two-axis motion simulator is defined as

$$\vartheta = \vartheta_{max} \sin\left(2\pi f \cdot t + \theta_0\right) \tag{4-16}$$

where ϑ_{max} and f are the input parameters of the Acutronic two-axis motion simulator. Taking the derivative of Eq. (4-16) with respect to time yields

$$\dot{\vartheta} = \vartheta_{max} 2\pi f \cos\left(2\pi f \cdot t + \theta_0\right) \tag{4-17}$$

and taking the second derivative

$$\ddot{\vartheta} = -\vartheta_{max} \left(2\pi f\right)^2 \sin\left(2\pi f \cdot t + \theta_0\right) \tag{4-18}$$

From the previous equations, it is necessary to determine which combinations of ϑ_{max} and f will be suitable for the experiment. The maximum torque expected to be measured will be

$$|T|_{max_{expected}} = |\ddot{\vartheta}|_{max} I_{ub} + d_{ub} \cdot W_T | \sin\vartheta|_{max} \tag{4-19}$$

where I_{ub} and d_{ub} are the upper bounds of the moment of inertia along the desired axis and distance to center of mass, respectively. These quantities can be predicted mathematically. Note that for z-axis measurements, the last term of Eq. (4-19) does not appear. For sinusoidal signals, the modulus becomes irrelevant since the maximum absolute values will be the same for the positive and negative sides. Furthermore, the torque sensor has a maximum value that it is capable of measuring, $T_{max_{sensor}}$. Therefore, to avoid saturation or even damage to the sensor, the parameters are chosen according to the criterion

$$\frac{T_{max_{expected}}}{T_{max_{sensor}}} = \frac{\vartheta_{max} \left(2\pi f\right)^2 I_{ub} + d_{ub} \cdot W_T \sin\vartheta_{max}}{T_{max_{sensor}}} \leq f_s \tag{4-20}$$

for x-axis and y-axis measurements and

$$\frac{T_{max_{expected}}}{T_{max_{sensor}}} = \frac{\vartheta_{max} \left(2\pi f\right)^2 I_{ub}}{T_{max_{sensor}}} \leq f_s \tag{4-21}$$

for z-axis measurements. For these criteria, f_s is a safety factor that corresponds to a safe design ratio between maximum expected torque and maximum torque that can be measured by the sensor. f_s must be smaller than one. No saturation is expected in the measurements of the force components since these measurements will correspond to fractions of the vehicle's weight along with that of the top plate. Following are the test signals selected.

- Measurements for Z-axis
  - Run 1: $\vartheta_{max} = 45\,\text{deg}, f = 0.6Hz$
  - Run 2: $\vartheta_{max} = 45\,\text{deg}, f = 0.5Hz$
  - Run 3: $\vartheta_{max} = 30\,\text{deg}, f = 0.6Hz$
- Measurements for Y-axis (and X-axis)
  - Run 1: $\vartheta_{max} = 30\,\text{deg}, f = 0.6Hz$
  - Run 2: $\vartheta_{max} = 15\,\text{deg}, f = 0.9Hz$

4-5 Data collection and analysis

4-5-1 Observations

The data collected is presented in this section in Figures 4-6, 4-7 and 4-8. Long runs of one minute each were taken, but only five-second samples are shown.

The force and torque measurements used to calculate inertia of the top mounting plate presented sinusoidal signals with very low amplitude, even when the two-axis motion simulator was excited with large values of maximum angle and frequency. This indicated that the plate's inertia was small. The calculations were performed and moments of inertia in the magnitude of 10^{-6} were obtained. As will be shown later, these values have a negligible influence on the previous measurements.

The data collected was analyzed in order to determine the moments of inertia. To eliminate noise and other unwanted contents present in the measurements, signals were first filtered. Subsequently, the necessary calculations were performed.

4-5-2 Filtering

From the observation of the plots shown, it is clear that unwanted effects such as motion-induced vibrations and aeroelastic modes are perturbing the measurements. These effects manifest as high-frequency contents that are added to the signals. All signals of interest obtained during the course of the experiment should have a pure sinusoidal behavior and are expressed mathematically as

$$F(\underline{p}) = F(A, \beta_0, \mu) = A \sin(2\pi f \cdot t + \beta_0) + \mu \tag{4-22}$$

where the amplitude A, initial phase β_0 and bias μ are the unknown parameters, grouped in a vector $\underline{p}$. Note that the frequency f is known since it is an input parameter of the two-axis motion simulator. For each signal, the objective is to find the optimum set of parameters, $\underline{p}^*$, that minimizes the difference between the modeled and measured signals. This problem can be formulated as an optimization problem

$$\min_{\underline{p}} ||F(\underline{p}) - F_{meas}||_2^2 = \min_{\underline{p}} ||e(\underline{p})||_2^2 = \min_{\underline{p}} \sum_{i=1}^{N} e_i^2(\underline{p}) \tag{4-23}$$

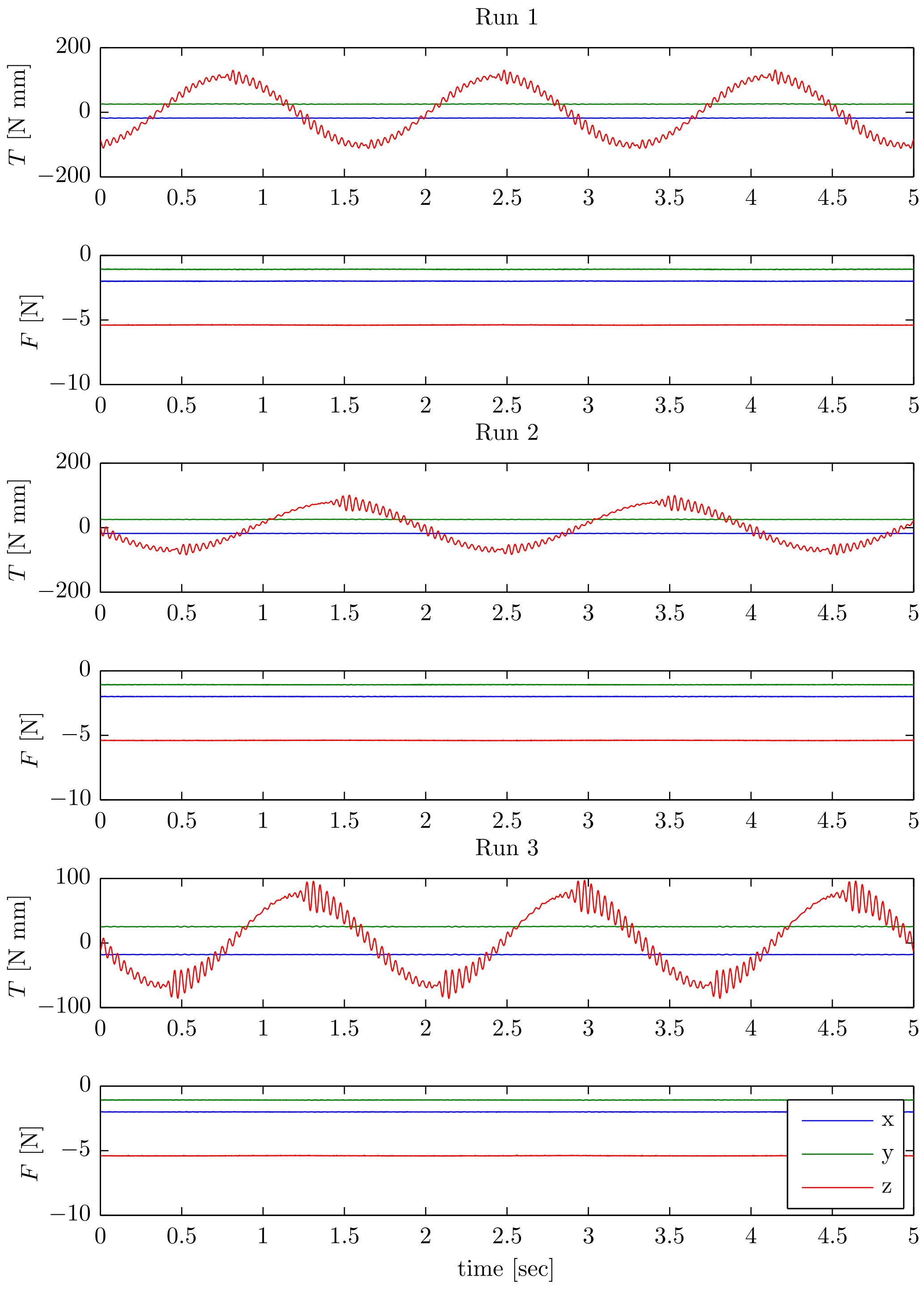


Figure 4-6: Measurements for the z-axis; rotation over the inner axis of the two-axis motion simulator

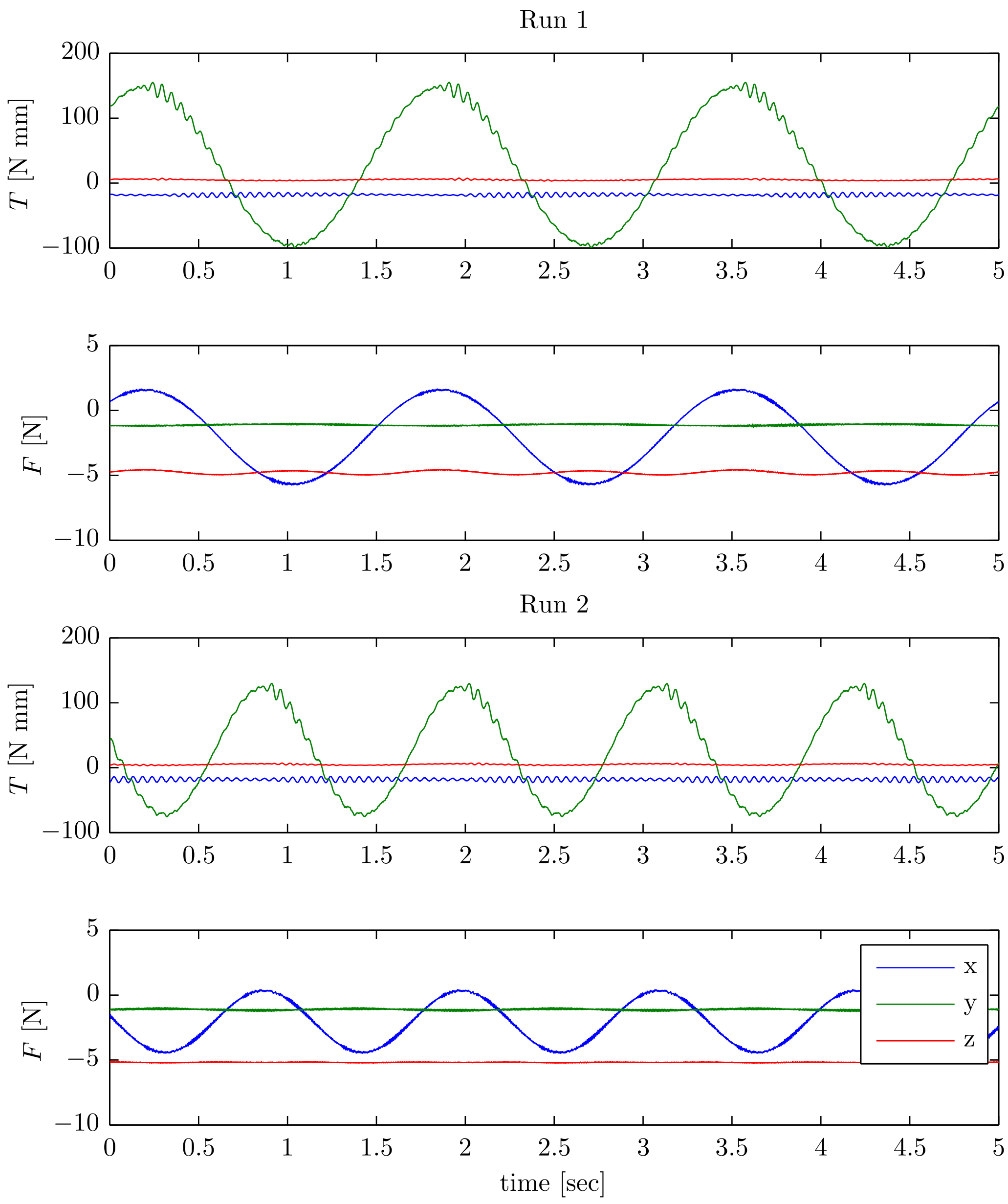


Figure 4-7: Measurements for the y-axis; rotation over the outer axis of the two-axis motion simulator

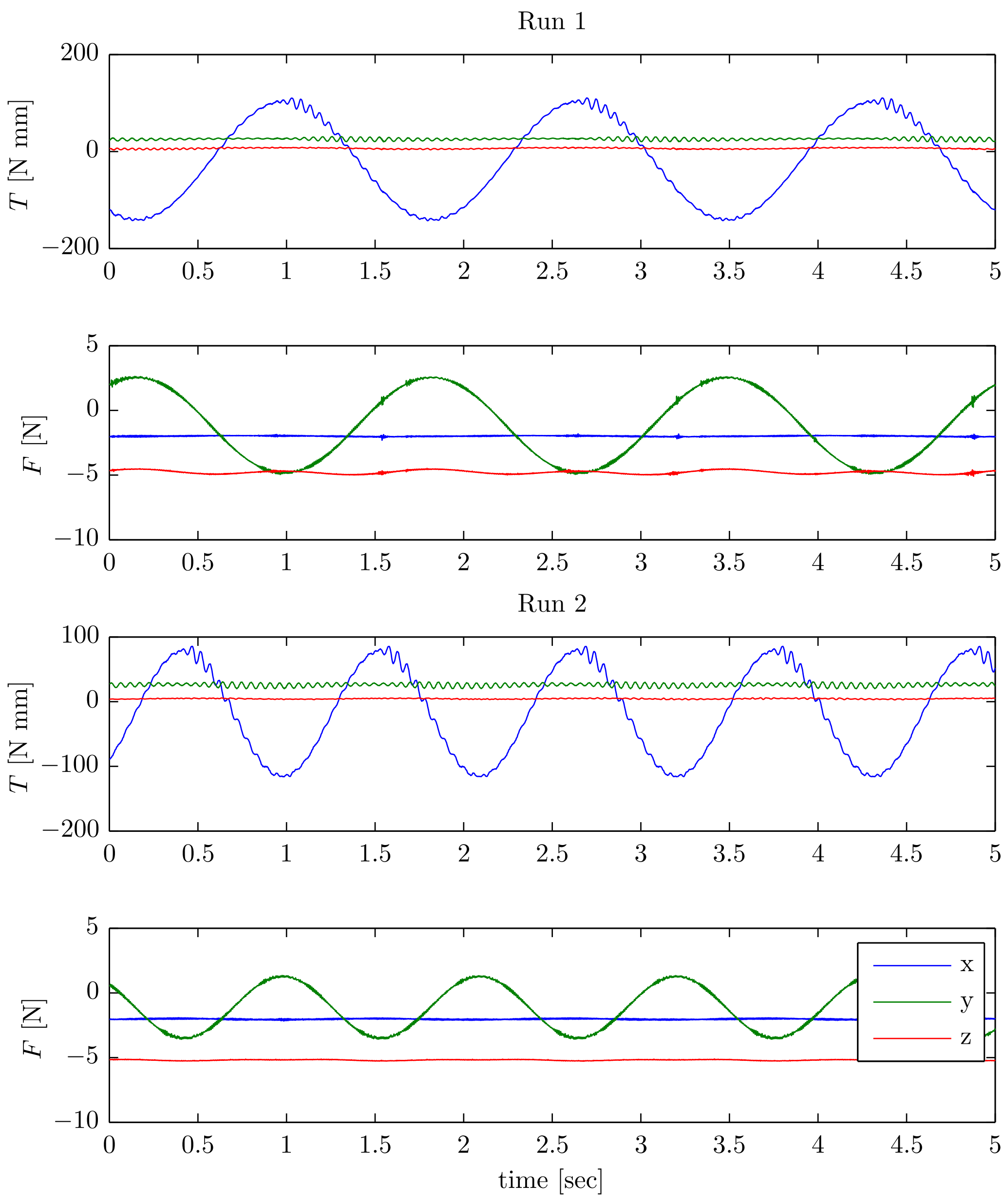


Figure 4-8: Measurements for the x-axis; rotation over the outer axis of the two-axis motion simulator

subjected to the following inequality constraints

$$\begin{aligned} A &> 0 \\ -\pi < \beta_0 &\leq \pi \end{aligned} \tag{4-24}$$

To obtain the optimum solution to this nonlinear least squares problem, the Matlab toolbox 'Curve Fitting Tool' was used in which a Trust-region algorithm was selected. All optimization procedures converged and rendered optimal solutions with root-mean-square error in the expected order of magnitude. In other words, the residual, $e(\underline{p}^*)$, contained only the unwanted high-frequency oscillations and noise. Plots of filtered measurements are presented in Figures 4-9, 4-10 and 4-11.

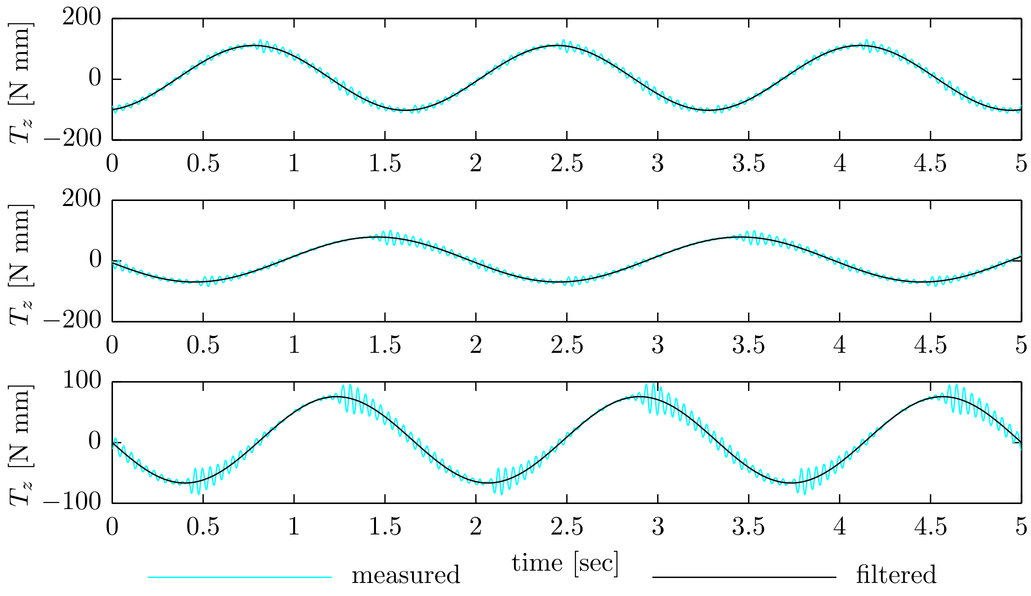


Figure 4-9: Filtered data versus non-filtered data for measurements in the z-axis

4-5-3 Determining height of the center of mass

Recall Eq. (4-15), from which a theoretical value of 0.021 m for the height of the center of mass, z_{CM}, was obtained. This parameter can be determined by calculating the moments of inertia for the x- and y-axis for different values of d. Note that for these two axes, Eq. (4-6) is used and can be rewritten as

$$I_{est} = \frac{T_{rot} + d_{real} W_T \sin\vartheta - d_{est} W_T \sin\vartheta}{\ddot{\vartheta}} + \epsilon \quad , \quad \ddot{\vartheta} \neq 0 \tag{4-25}$$

where the subscript *est* indicates estimate and *real* stands for the real value, and ϵ corresponds to a very small estimation error. Eq (4-25) can be rearranged as follows:

$$I_{est} = I_{real} + (d_{real} - d_{est}) \frac{W_T \sin\vartheta}{\ddot{\vartheta}} + \epsilon \quad , \quad \ddot{\vartheta} \neq 0 \tag{4-26}$$

It becomes clear that if $d_{est} \neq d_{real}$ then the second term of the right hand side of this equation will not disappear and the moment of inertia estimation will depend wrongly on the

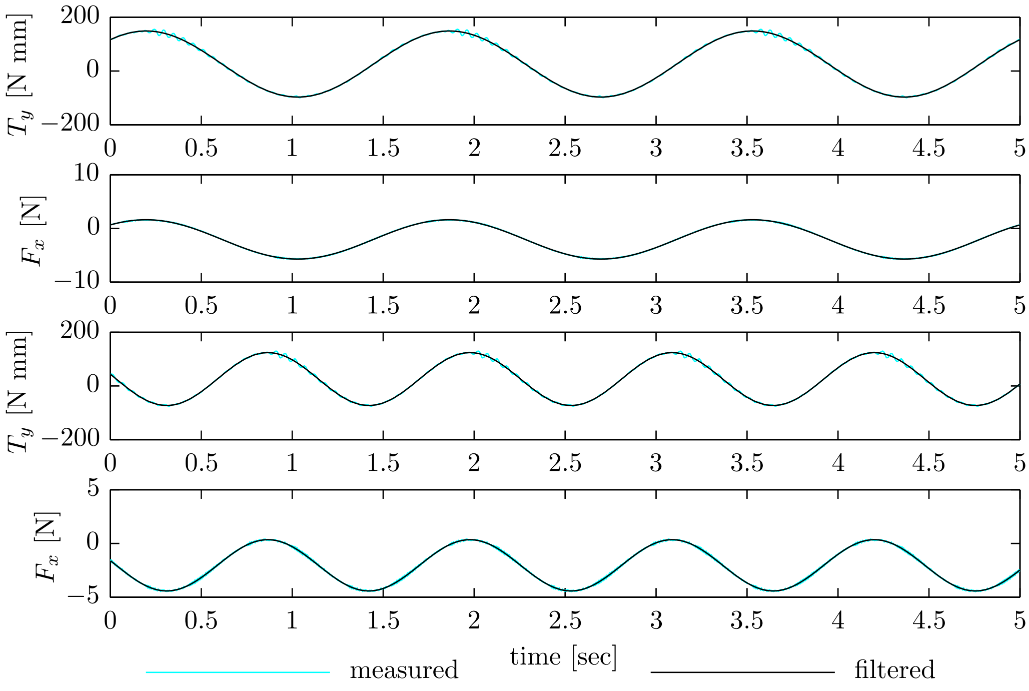


Figure 4-10: Filtered data versus non-filtered data for measurements in the y-axis

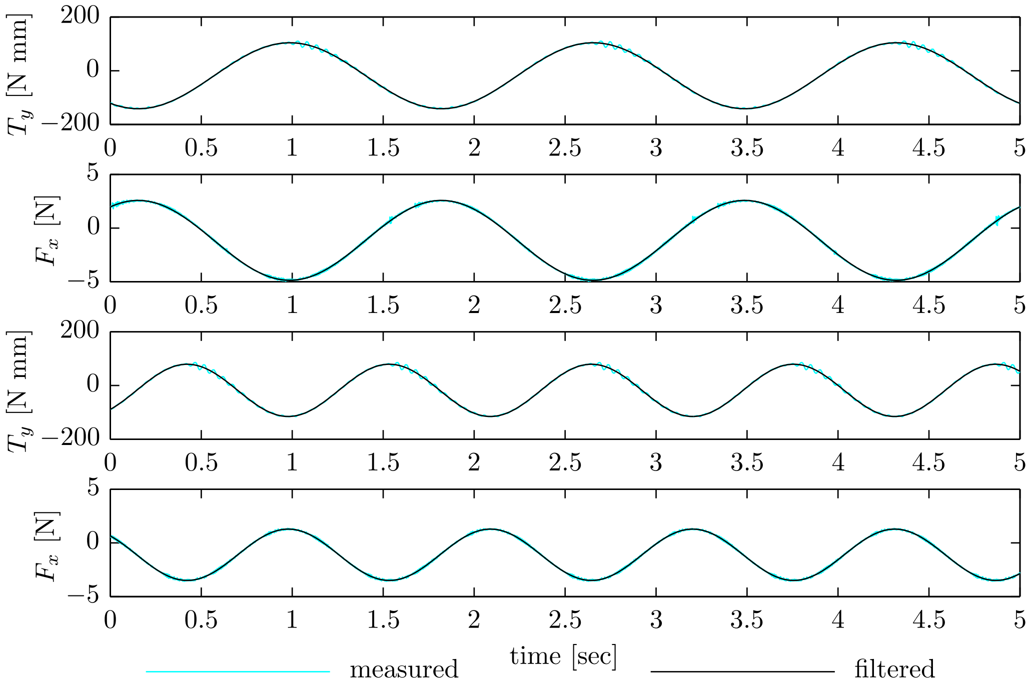


Figure 4-11: Filtered data versus non-filtered data for measurements in the x-axis

Table 4-2: Calculation of the moments of inertia for different values of d. The test signals parameters are as follows. Run 1: $\vartheta_{max} = 30\,\text{deg}, f = 0.6Hz$. Run 2: $\vartheta_{max} = 15\,\text{deg}, f = 0.9Hz$

	I_{xx} (in $Kg \cdot m^2$)		I_{yy} (in $Kg \cdot m^2$)	
	Run		Run	
	1	2	1	2
$d = 0.021m$	0.0060	0.0056	0.0062	0.0058
$d = 0.022m$	0.0055	0.0053	0.0057	0.0055
$\mathbf{d = 0.023m}$	**0.0050**	**0.0051**	**0.0052**	**0.0052**
$d = 0.024m$	0.0045	0.0048	0.0047	0.0049
$d = 0.025m$	0.0040	0.0045	0.0042	0.0046

Table 4-3: Comparison of experimental with theoretical results

	I_{xx} (in $Kg \cdot m^2$)		I_{yy} (in $Kg \cdot m^2$)		I_{zz} (in $Kg \cdot m^2$)		
	Run		Run		Run		
	1	2	1	2	1	2	3
Experimental	0.0050	0.0051	0.0052	0.0052	0.0096	0.0096	0.0096
Geometrical shapes	0.0050		0.0050			0.0095	
Point masses	0.0036		0.0036			0.0071	

two-axis motion simulator's input parameters. Since the thickness of the top mounting plate is known ($h_{plate} = 2\ mm$), it follows that $z_{CM} = d - h_{plate}$. Table 4-2 presents the inertia obtained for different values of d (in increments of 1 mm). The results are most consistent for $d = 0.023\ m$ and consequently $z_{CM} = 0.021\ m$, which is in accordance with the value obtained theoretically.

Alternatively, the calculation of z_{CM} could have been performed independently by measuring forces and torques for when the two-axis motion simulator is not in motion, but has a certain angle about its horizontal axis. The sensor was no longer available after the experiment, so it was not possible to perform such measurements. Nevertheless, the method used rendered highly accurate results.

4-5-4 Experiment results and discussion

The results of the moments inertia calculations are presented in Table 4-3 for each experiment run. For comparison, the theoretical predictions are presented as well for both the point masses assumption and the geometric shapes assumption.

It is clear that when only the motors are assumed as point masses, the inertia will be underestimated as part of the mass of the quadrotor is not taken into account. Upon analyzing the results of the other two methods it becomes clear that the theoretical results are in accordance with the experimental values. This indicates that the experimental method proposed is valid for determining moments of inertia.

4-6 Final remarks

The experimental method proposed in this chapter is more expensive and time-consuming than a more mathematical and theoretical approach, where only a scale and a ruler are needed. In this study, the same results are also obtained. Nevertheless, the results presented serve as empirical proof that the concept of the experiment conducted is valid and can be applied to multiple platforms where the analytical method is not feasible; for example, when it is not possible to disassemble the system, or when the shapes of the objects are not entirely known or do not have known inertia formulas. Moreover, this experimental method is useful when extremely precise knowledge of the inertia properties is required.

The quadrotor's moments of inertia were also calculated using the commonly used point mass analysis, in which the vehicle was modeled as four point masses (corresponding to the four motors) lying at a certain distance from the body's center of mass. Such calculations lead to erroneous inertia estimation (deviation of 20-30% from the real value) thus resulting in a significant model mismatch.

Further details were considered in order to enhance the accuracy of the results. However, they were considered irrelevant. For example, for measured torques with and without the quadrotor, the mean values without motion were different, indicating that the center of mass is not exactly located in the center of the vehicle. The order of magnitude of the observed difference was around 10 N mm, which leads to a shift of the center of mass of about 1 mm. Such correction has a negligible influence on the final results, meaning that the axes alignment assumption made previously was valid.

The errors resulting from the analytical calculations are due to the assumptions of simplified geometry of the quadrotor's components. From the experimental side, the error sources include noise and biases of the sensor, misalignments between motion simulator, sensor and quadrotor axes, vibrations and aeroelastic modes. It should be noted that accurate measurements were nonetheless obtained, and these sources of error did not hinder the final results. In fact, with the proposed filtering procedure, the influence of these effects was minimized using a mathematical optimal approach based on physical knowledge.

To conclude, the results are summarized in Table 4-4. The experimental results were averaged in order to obtain the final values of the moments of inertia.

Table 4-4: Summary of the results

	I_{xx} (in $Kg \cdot m^2$)	I_{yy} (in $Kg \cdot m^2$)	I_{zz} (in $Kg \cdot m^2$)	z_{CM} (in m)
theoretical	0.0050	0.0050	0.0095	0.021
experimental	0.0050	0.0052	0.0096	0.021

Chapter 5

Vision System

An on-board downward-looking single camera is used in this project as a means to determine position of the quadrotor relative to a target with known characteristics. This chapter presents an overview of the vision-based tracking algorithm developed for autonomous flight. The vision system proposed in this thesis computes 3D position measurements from images captured by a camera.

5-1 Tracking algorithm

5-1-1 Characteristics of the target

The characteristics of a target can be multiple and varied. The design of the geometry and other features of the target should take into account the fact that such characteristics will influence the choice of the image processing algorithms. For this project, different options have been considered. The target can be composed of either one or several solid shapes, and for each of those shapes the following three characteristics must be selected:

- Geometric shape. Simple shapes such as circles or squares are preferred, as they possess known properties that can be used by the visual odometer.
- Dimensions. On the one hand, it is clear that the target must be small enough to appear completely in the image when the quadrotor is in the vicinity of the landing platform; on the other hand, if the target is too small, it will not fill enough pixels in the image at higher altitudes, resulting in less robust measurements.
- Color. A color that does not exist abundantly in nature is preferable. Specifically for application at sea, red is a reasonable choice. Furthermore, if a Red Green Blue (RGB) color mode is used, it is possible to evaluate the intensity of the first component in comparison to the other two to determine if a particular pixel has the desired color.

5-1-2 Image processing

From the aforementioned target characteristics it is clear that the target will be composed of one or more red blobs. The image processing algorithm to detect these blobs and extract their properties is therefore designed accordingly. During the development phase, the design is performed off-line using the 'Image Processing' toolbox from Matlab. Afterward, the system is implemented on board for real-time applications. The steps used for this algorithm are listed below.

1. Red pixels search
2. Conversion to a binary image
3. Extraction of blob properties

Red pixels search

The detection of the blob can be performed by scanning all pixels from a grabbed image. When using the RGB color mode, every pixel is described as a vector with three parameters (R - red,G - green,B - blue), each assuming a value ranging from 0 to 255. Pure red exists when the values of the pixels are [R,G,B]=[255,0,0], but finding such a combination cannot be expected in real applications. A certain tolerance should be added and red-like combinations should be accepted. Several methods are thus possible for determining whether or not a pixel is accepted as red. Among other options, the 'red condition' can be selected as follows:

- Condition 1: $D = \sqrt{(R-255)^2 + (G-0)^2 + (B-0)^2} < threshold_1$
- Condition 2: $R > threshold_2 \cdot G \quad \& \quad R > threshold_2 \cdot B$
- Condition 3: $R > threshold_3 \cdot (G + B)$

Similar results were obtained for all approaches tested during the off-line design process, and it became evident that the most important factor was a proper choice of the threshold, regardless of the method.

Alternative techniques can also be applied, including using a Hue Saturation Lightness (HSL) or Hue Saturation Value (HSV) color mode scheme, fuzzy logic image segmentation, etc.

Conversion to a binary image

A value of one is attributed to all pixels for which the 'red condition' is satisfied; otherwise, a value of zero is given. With this technique, a binary image can be constructed. A blob is then defined as the solid shape formed by clustering all adjacent black pixels. For clarity, Figure 5-1 presents the result of the image processing algorithm up to this step.

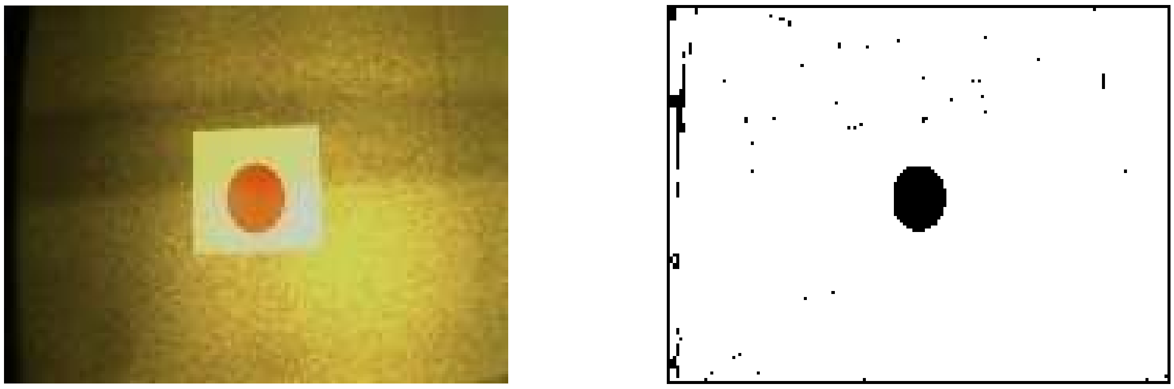


Figure 5-1: Image grabbed by the camera (left) and corresponding output after searching for red pixels and converting to a binary image (right). These images were obtained during design phase. For the binary image, condition 2 was used and the *threshold* parameter was not optimized.

Extraction of blob properties

The next step consists of extracting information from the binary image obtained. First, it is important to be aware that multiple blobs can be expected. In Figure 5-1, the threshold from the 'red condition' was purposely selected as not optimal in order to show that false blobs that do not correspond to the target might appear. An integrity monitor, explained in Section 5-3, is employed to filter such blobs and avoid erroneous measurements (among other functions).

Given that the correct blobs are detected, the extraction of properties is then performed. For each blob, the following parameters can be obtained: area, perimeter, minimum and maximum in the horizontal and vertical direction and centroid. Note that in this step all properties are given in pixels (or pixels squared).

5-1-3 Computation of relative position

The known equations for projection of a target in the camera frame are given by

$$\begin{aligned} x_{proj} &= f\frac{x}{z} \\ y_{proj} &= f\frac{y}{z} \end{aligned} \tag{5-1}$$

Note, however, that to be usable by the vision system with desirable units, these equations must be adapted to the image coordinate frame as

$$\begin{aligned} x_p &= \frac{s_x f}{z}x + x_o \\ y_p &= \frac{s_y f}{z}y + y_o \end{aligned} \tag{5-2}$$

where s_x and s_y are scaling factors that take into account pixel size, and x_o and y_o represent the origin of the image frame coordinate system. This origin is selected as the center point of the image frame, corresponding to half of the camera frame length in each direction.

Range finding

The first parameter to be estimated is the distance between the camera and the target. It is assumed that all points of the target's solid shapes are at the same distance from the camera. If the real distance between two points of the target is known, z can be estimated as follows:

$$\begin{aligned} x_{p2} - x_{p1} &= \frac{s_x f}{z} x_2 + x_o - \left(\frac{s_x f}{z} x_1 + x_o \right) \\ &= \frac{s_x f}{z} \left(x_2 - x_1 \right) \end{aligned} \tag{5-3}$$

In these equations z must be positive, as the camera will always be above the target. Therefore, by taking the modulus

$$|x_{p2} - x_{p1}| = \frac{s_x f}{z} |x_2 - x_1| \tag{5-4}$$

the equation can be rearranged to obtain

$$z = \frac{s_x f |x_2 - x_1|}{|x_{p2} - x_{p1}|} \tag{5-5}$$

where $|x_2 - x_1|$ is a known real distance characteristic of the target and $|x_{p2} - x_{p1}|$ is the corresponding distance in the projected image along the x-direction. The same derivation holds for the y-direction.

If the real distance known appears in the projection image with components in both the x- and y-directions, the projected length would be expressed by

$$d_p = \sqrt{\left(x_{p2} - x_{p1}\right)^2 + \left(y_{p2} - y_{p1}\right)^2} \tag{5-6}$$

and given that the scaling factors, defined as s_c, are equal for x and y, it follows that

$$z = \frac{s_c f d_{real}}{d_p} \tag{5-7}$$

where d_{real} is the real distance. Note that for this case, the orientation of the image must be known. For example, if a square is used with known diagonal length, the image processing algorithm must find the points in the image corresponding to the opposite corners of the square.

Horizontal position

Once the height is calculated, the lateral position can be computed from Eqs. (5-2) as follows

$$\begin{aligned} x &= \frac{z \left(x_p - x_o \right)}{s_c f} \\ y &= \frac{z \left(y_p - y_o \right)}{s_c f} \end{aligned} \tag{5-8}$$

where x_p and y_p are the image frame coordinates of the target's center. It should be noted that these coordinates are in the body-fixed reference frame.

5-2 Algorithm selection for real-time implementation

From the generic algorithm sketched in the previous section, a more specific implementation is chosen for real-time testing. It was observed that the time delays due to image acquisition and processing were quite large. Therefore, the algorithm was selected based on the fact that the position measurement update rate should be as high as possible.

A single red circle with diameter of 10 cm was chosen as the target. This length parameter was consistent with the desired starting height for testing as will be seen in Section 5-5. Note that the image projection of the circle will always be the same, independent of the camera's orientation (provided that the image frame is parallel to the target). This feature facilitates the extraction of the circle's diameter under projection that can be compared to the real known diameter as follows

$$diameter = 2\sqrt{\frac{Area}{\pi}} \tag{5-9}$$

or simply

$$diameter = max(blob_x) - min(blob_x) \tag{5-10}$$

where $blob_x$ is the representation of the blob in the x-direction. The same can be applied in any other direction.

The center of the target is then given by

$$target_{center} = (min(blob) + max(blob))/2 \tag{5-11}$$

It should be pointed out that by using only one solid shape in the target, measurement accuracy and robustness will suffer. The selection of the tracking algorithm is then a compromise between accuracy/robustness and speed; in general, a more complex algorithm results in more accurate and robust measurements, but with larger time delays.

5-3 Integrity monitor

The blobs detected in an image frame are not always reliable for position calculations. An integrity monitor is therefore developed to avoid erroneous measurements as follows:

- The area of the largest blob should be larger than a certain threshold th_A. This avoids misidentification of a blob that does not correspond to the red circle.
- A roundness score of the blob is computed as (Berkelaar & Oonk, 2009)

  $$Roundness\ score = |\frac{perimeter}{2\pi} - \frac{diameter}{2}| \tag{5-12}$$

  To consider the blob reliable, this score should be below a threshold th_R.

If one of the two tests fails, then no measurement is available. Otherwise, position measurement calculations are performed. For every two consecutive samples, finite differences are computed. If the absolute value of the calculated speed is higher than the maximum speed at which the airframe can fly, then no measurement is available; if it is lower, then the a relative position measurement is obtained. Figure 5-2 clarifies the integrity monitor.

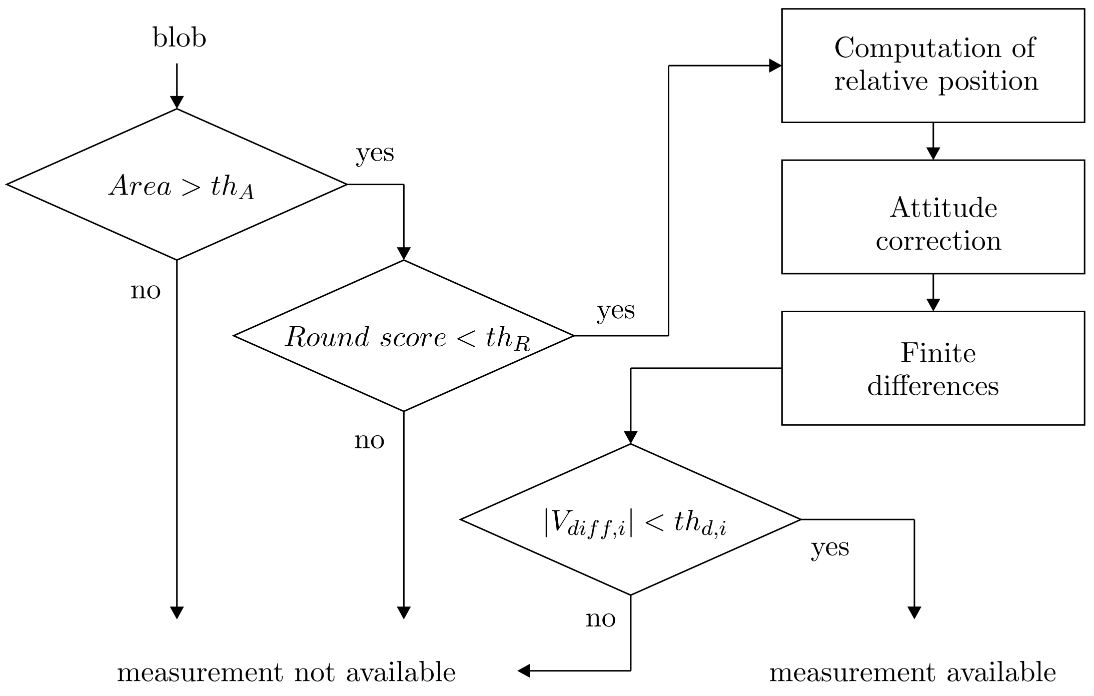


Figure 5-2: Integrity monitor scheme

5-4 Attitude compensation

Tracking the quadrotor's translational motion with respect to the target is essential for achieving autonomous stable position control. However, tracking helicopter translation alone is complicated since image displacements also occur with helicopter rotation. As the camera is fixed to the rotorcraft, it will rotate with the body-fixed reference frame. It is therefore extremely difficult to distinguish between rotation and translation of the vehicle when merely analyzing a sequence of images under perspective projection. Rolling motion will appear very similar to sideways translation, and pitching motion to forward/backward movements. The aforementioned effects of rotation must be eliminated from the measured image displacements in order to determine the rotorcraft's position with respect to the target. This can be achieved by accurately measuring the attitude of the helicopter (and consequently of the camera as well) and performing the necessary corrections. This compensation is only valid provided that the attitude data is precisely synchronized with the camera information (Amidi, 1996).

First, since the camera-fixed reference frame has its own orientation (ϕ_{cam}, θ_{cam}, ψ_{cam}) with respect to the body-fixed reference frame, a rotation between these two frames is necessary. This rotation is described by the rotation matrix $R^{\mathcal{CB}}(\phi_{cam}, \theta_{cam}, \psi_{cam})$. Then, the rotation from the body-fixed reference frame to the inertial NED reference frame can be obtained with the attitude of the quadrotor (ϕ, θ, ψ). This rotation is described by $R^{\mathcal{BN}}(\phi, \theta, \psi)$. The two rotation matrices are obtained as explained in Section 3-3, and the full rotation from camera to inertial frame is performed as follows:

$$R^{\mathcal{CN}} = R^{\mathcal{BN}} \cdot R^{\mathcal{CB}} \tag{5-13}$$

Figure 5-3: Caspa VL from Gumstix

5-5 Real-time implementation

5-5-1 Overview of the camera properties

The camera used in this project is a CASPA VL from Gumstix, for which the main features are listed below:

- 1/3-Inch wide-VGA CMOS digital image sensor
- Active pixels: 752 horizontal $\times$ 640 vertical
- Pixel size: 6.0μm $\times$ 6.0μm
- Frame rate: 60 fps at full resolution
- 3.6mm fixed focal length lens with IR cut filter (receives only visible spectrum light)

This camera requires a Gumstix OVERO COM for computation, and connection to the OVERO board is possible with an 80mm 27-pin flex ribbon cable. The camera is mounted in a downwards-facing position on the quadrotor's frame.

5-5-2 Validation with real camera data

To test whether the vision system is working properly, a validation procedure is required. For this purpose, the vision measuring system output data was logged and compared to expected values (note that the experiments were carried out in a laboratory, where conditions are known).

Motionless tests

For each height above the target, two measurements were performed with lateral displacement.

From Table 5-1, it is clear that this size of target (circle with 10 cm of diameter) is suitable for when the camera is in the near vicinity of the target (less than 1 m). At a height of approximately 2 m, the measurements are off by several centimeters. This could easily be corrected by using a larger circle for the target.

Table 5-1: No motion tests for camera validation

	Vision measurements (in cm)								
	Test 1			Test 2			Test 3		
	horiz. Δx	vertical 1^{st}	vertical 2^{nd}	horiz. Δy	vertical 1^{st}	vertical 2^{nd}	horiz. Δx	vertical 1^{st}	vertical 2^{nd}
measured	10.34	28.70	31.58	21.99	81.47	81.72	140.30	261.66	284.38
expected	11	30		20	82		100	200	

Note that the measurements for heights of 30 and 82 centimeters were still not perfect. This is not only caused by indoor light conditions that affect the colors seen by the camera, but also due to the fact that the camera's vertical axis is not perfectly aligned with the inertial vertical direction.

In-flight tests

To conclude these tests, the system was validated in flight with manual control. The maneuver performed resembles a situation that might occur in an autonomous landing procedure. Starting in a grounded position directly on top of the target, the helicopter first climbs to 1 m and navigates around the target for a few seconds. During this stage, the pilot purposely loses the target to test the integrity monitor. The results are shown in Figure 5-4.

From an hardware/software perspective, it was not possible to synchronize vision data with attitude information. Since the update rate of the vision system is asynchronous (i.e., measurements do not have a fixed time step), finding the correct attitude for the delayed vision measurements is quite challenging. One possible solution would be to introduce a delay in the attitude estimation, but for that, the vision system would need to produce synchronous measurements with a known time delay.

5-5-3 Noise properties

Determining the noise properties of this vision measuring system is a very difficult task. For accelerometers and gyros, a fairly reasonable estimation of the noise characteristics can be obtained by performing a statistical analysis of the IMU measurements for when the quadrotor is motionless. This noise can be approximated to a zero-mean white noise. The same does not hold for the camera, however; in a no-motion state with perfect external conditions, it is expected that the output of the vision system will not vary around a specific value. The noise thus appears from vibration of the camera due to the rotating blades and other external conditions such as shadows.

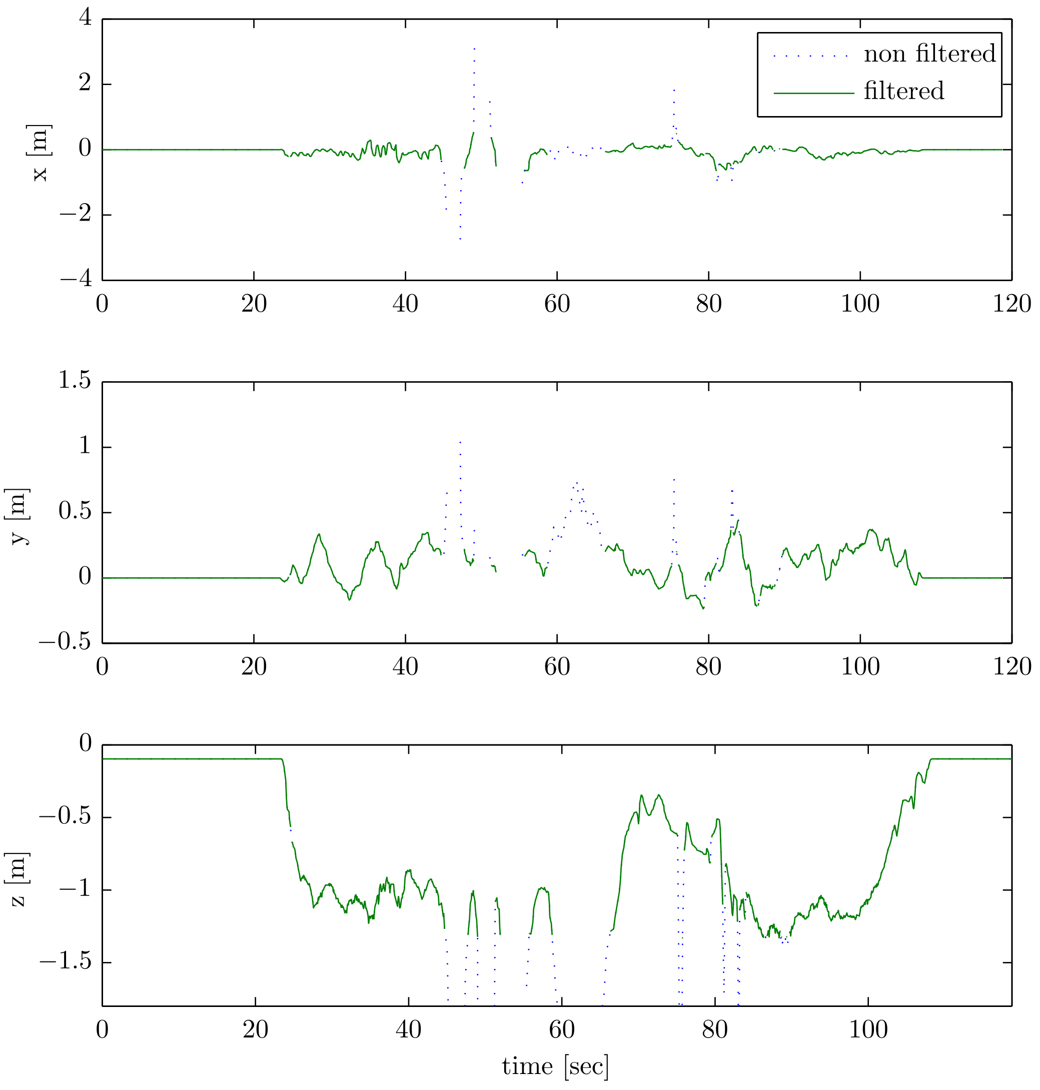


Figure 5-4: Vision measurements from manual test flight

Chapter 6

State estimation

Perfect sensor measurements are an unrealistic concept in the engineering world. In fact, measurements are contaminated by noise and biases, and in some instances may even be unavailable. It is therefore necessary to use state estimation. For this mission, there are two main stages to be considered in terms of state estimation; when the landing spot is visible from the camera perspective, and when it is not. When the quadrotor is several meters away from the landing platform, it is expected that the camera will not be able to obtain an image of the landing spot. For this phase, the autopilot must rely on GPS measurements to determine its position with respect to the platform (which has its own GPS coordinates). GPS measurements, however, are not reliable in the critical phase of landing. Therefore, the camera images must be used to determine the relative position of the camera with respect to the platform. This chapter covers the Kalman filter state estimation algorithms for such vision-based approach and landing. One of the main research contributions to be taken from this study is a new approach for estimating the vertical motion of a floating platform.

6-1 Kalman filtering

6-1-1 Theory of the Kalman filter

The widely known Kalman filter was named after Rudolph Emil Kálmán, who in 1960 published his famous paper describing a recursive solution to the discrete-data linear filtering problem (Kalman, 1960). This filter is a tool in the form of a set of equations that provides an efficient computational method to optimally estimate the variables of a wide range of processes. In mathematical terms, it can be said that a Kalman filter estimates the states of a linear system by theoretically minimizing the estimation error.

After its introduction, the Kalman filter has been subjected to extensive research. Currently, many variations of this filter can be found, including continuous and discrete-time implementations as well as linear and nonlinear variants. The different versions of the Kalman filter have been widely applied in many technological fields, particularly in the area of navigation. In this thesis, this tool will be used for the quadrotor's state estimation and sensor integration.

6-1-2 The discrete Kalman filter

Consider the general linear, time-varying state-variable model given by

$$\dot{\underline{x}}(t) = F(t)\underline{x}(t) + B(t)\underline{u}(t) + G(t)\underline{w}(t), \quad \underline{x}(t_0) = \underline{x}_0 \tag{6-1}$$

and output (measurements) model given by

$$\underline{z}(t) = H(t)\underline{x}(t) + D(t)\underline{u}(t) + \underline{v}(t), \quad t = t_i, \;\; i = 1, 2, ... \tag{6-2}$$

where the variables are defined as

$\underline{x}(t)$	State vector (dimension n)
$\underline{x}_0$	Initial condition of the state
$\underline{u}(t)$	Control input vector (dimension l)
$\underline{w}(t)$	System/process noise vector (dimension m)
$\underline{z}(t)$	Measurement vector (dimension p)
$\underline{v}(t)$	Measurement noise vector (dimension p)
$F(t)$	System matrix (dimension $n \times n$)
$B(t)$	Input matrix (dimension $n \times l$)
$G(t)$	System/process noise matrix (dimension $n \times m$)
$H(t)$	Observation matrix (dimension $p \times n$)
$D(t)$	Feedforward matrix (dimension $p \times l$)

Note that $\underline{w}(t)$ is a continuous-time white noise process and $\underline{v}(t_i)$ is a discrete-time white noise sequence (the sensors' output is discrete). Furthermore, $\underline{w}(t)$ and $\underline{v}(t_i)$ are mutually uncorrelated for all $t = t_i$, $i = 1, 2, ...$ Therefore, the following holds:

$$\begin{aligned} E\{\underline{w}(t)\} &= 0 \\ E\{\underline{w}(t)\underline{w}^T(\tau)\} &= Q(t)\delta(t-\tau) \end{aligned} \tag{6-3}$$

$$\begin{aligned} E\{\underline{v}(t_i)\} &= 0 \\ E\{\underline{v}(t_i)\underline{v}^T(t_j)\} &= R(t_i)\delta_{ij} \end{aligned} \tag{6-4}$$

$$E\{\underline{w}(t)\underline{v}^T(t_i)\} = 0, \qquad \text{for } t = t_i, \; i = 1, 2, ... \tag{6-5}$$

where $Q(t)$ and $R(t_i)$ are the system and measurement noise covariance matrices, respectively. The symbol $\delta(\cdot)$ denotes the Dirac delta function and the symbol δ_{ij} denotes the Kronecker delta function.

After discretization of the linear system given by Eqs. (6-1) and (6-2), one obtains the discrete-time model given by

$$\underline{x}(k+1) = \Phi(k)\underline{x}(k) + \Psi(k)\underline{u}(k) + \underline{w}_d(k) \tag{6-6}$$

and

$$\underline{z}(k+1) = H(k+1)\underline{x}(k+1) + D(k+1)\underline{u}(k+1) + \underline{v}(k+1) \tag{6-7}$$

where

$\Phi(k)$	System transition matrix (dimension $n \times n$)
$\Psi(k)$	Input distribution matrix (dimension $n \times l$)

Similar to the continuous-time case, the system and measurement noises have the following characteristics:

$$\begin{aligned} E\{\underline{w}_d(i)\} &= 0 \\ E\{\underline{w}_d(i)\underline{w}_d^T(j)\} &= Q_d(i)\delta(kj) \end{aligned} \tag{6-8}$$

$$\begin{aligned} E\{\underline{v}(i)\} &= 0 \\ E\{\underline{v}(i)\underline{v}^T(j)\} &= R(i)\delta_{ij} \end{aligned} \tag{6-9}$$

$$E\{\underline{w}_d(i)\underline{v}^T(j)\} = 0 \tag{6-10}$$

One way of deriving the Kalman filter mathematical equations can be found in (Chu, 2009). The formulation of the estimation problem starts by defining a mean vector $\hat{\underline{x}}(k+1|k)$ and the associated error covariance matrix of the state $P(k+1|k)$ as follows:

$$\hat{\underline{x}}(k+1|k) = E\left\{\underline{x}(k+1)\right\} \tag{6-11}$$

$$P(k+1|k) = E\left\{\left[\underline{x}(k+1) - \hat{\underline{x}}(k+1|k)\right]\left[\underline{x}(k+1) - \hat{\underline{x}}(k+1|k)\right]^T\right\} \tag{6-12}$$

The filter is then derived by minimizing a quadratic value function defined as:

$$\begin{aligned} J =& \frac{1}{2}\left[\underline{x}(k+1) - \hat{\underline{x}}(k+1|k)\right]^T P^{-1}(k+1|k)\left[\underline{x}(k+1) - \hat{\underline{x}}(k+1|k)\right] \\ &+ \frac{1}{2}\left[\underline{z}(k+1) - H(k+1)\underline{x}(k+1)\right]^T R^{-1}\left[\underline{z}(k+1) - H(k+1)\underline{x}(k+1)\right] \end{aligned} \tag{6-13}$$

The Kalman filter can be summarized into five standard steps for each iteration as follows. Note that the notation $k+1|k$ describes a prediction estimate.

- **Time Update (Prediction)**
  1. One step ahead state prediction (time propagation)
  $$\hat{\underline{x}}(k+1|k) = \Phi(k)\underline{x}(k|k) + \Psi(k)\underline{u}(k) \tag{6-14}$$
  2. Prediction of the covariance matrix of the state error vector
  $$P(k+1|k) = \Phi(k)P(k|k)\Phi^T(k) + Q_d(k) \tag{6-15}$$
- **Measurement Update (Correction)**
  3. Computation of the Kalman gain
  $$K(k+1) = P(k+1|k)H^T(k+1)\left[H(k+1)P(k+1|k)H^T(k+1) + R(k+1)\right]^{-1} \tag{6-16}$$
  4. Measurement update step
  $$\hat{\underline{x}}(k+1|k+1) = \hat{\underline{x}}(k+1|k) + K(k+1)\left[\underline{z}(k+1) - H(k+1)\hat{\underline{x}}(k+1|k)\right] \tag{6-17}$$
  5. Update of the error covariance matrix
  $$\begin{aligned} P(k+1|k+1) =& \left[I - K(k+1)H(k+1)\right]P(k+1|k)\left[I - K(k+1)H(k+1)\right]^T \\ &+ K(k+1)R(k+1)K^T(k+1) \end{aligned} \tag{6-18}$$

Note that the filter must be initialized by assigning values to $\underline{\hat{x}}(0|0)$ and $P(0|0)$. Q_d, the discrete covariance matrix of the process noise, will be discussed later.

Tuning parameters

To implement a Kalman filter, knowledge of the noise statistical information for both the process and measurements is required. Such information is introduced in the Kalman filter in the matrices Q and R (Chu, 2009), which should be properly tuned to improve the filter's performance. The measurement noise can be determined by taking offline samples from the sensors used in order to determine the variance of the noise. This approach is valid for most cases, but it should be stressed that for some sensors, it may not work. For instance, the noise characteristics of the vision system developed in this thesis are very difficult to determine. Therefore, an initial guess can be made for R, but it will most likely be necessary to change its values after flight testing. The other tuning parameter is the matrix Q, which is the covariance matrix related to the process noise and, as expected, is more challenging to define. If the model is accurate, then lower values can be selected for Q. However, this scenario may not always be realistic; higher values can be chosen for Q, thus introducing uncertainty in the model.

Discretization issues

The Kalman filter presented is in a discrete-time form. For the cases in which the system dynamics are represented in continuous time, continuous-to-discrete transformations are required to obtain the system transition $\Phi(k)$ and input distribution $\Psi(k)$ matrices. These are computed as follows:

$$\Phi(k) = e^{F(t_k)\Delta t} \tag{6-19}$$

$$\Psi(k) = \int_{t_k}^{t_{k+1}} \Phi(t_{k+1}, \tau)B(\tau)\mathrm{d}\tau \tag{6-20}$$

Furthermore, the matrix Q_d is given by (Lopes, 2011)

$$Q_d(k) = \int_{t_k}^{t_{k+1}} \Phi(t_{k+1,\tau})G(\tau)Q(\tau)G^T(\tau)\Phi^T(t_{k+1}, \tau)\mathrm{d}\tau \tag{6-21}$$

and an approximation can be obtained with the following equation

$$Q_d(k) \approx \frac{1}{2}\left[\Phi(k)G(t_k)Q(t_k)G^T(t_k)\Phi^T(k) + G(t_{k+1})Q(t_{k+1})G^T(t_{k+1})\right]\Delta t \tag{6-22}$$

6-1-3 Extension to nonlinear systems

The Kalman filter algorithm shown in the previous section has been designed for linear systems. However, system and measurement equations are nonlinear most of the time, particularly in aerospace applications. Therefore, the Extended Kalman Filter (EKF) can be used to estimate the state vector when nonlinearities are present. As the name indicates, the EKF is an extension of the basic Kalman filter to nonlinear dynamical systems.

Consider the general nonlinear state space model given by

$$\begin{aligned}\dot{\underline{x}}(t) = \underline{f}\left[\underline{x}(t), \underline{u}(t), t\right] + G\left[\underline{x}(t), t\right]\underline{w}(t), \quad \underline{x}(0) = \underline{x}_0 \\ \underline{z}_n(t) = \underline{h}\left[\underline{x}(t), \underline{u}(t), t\right] \\ \underline{z}(t_k) = \underline{z}_n(t_k) + \underline{v}(t_k), \qquad k = 1, 2, ...\end{aligned} \tag{6-23}$$

where $\underline{f}\left[\cdot\right]$ is the nonlinear system equation and $\underline{h}\left[\cdot\right]$ is the nonlinear observer equation that relates the states to the measurements. It is assumed that both these functions are continuous and continuously differentiable with respect to all elements of $\underline{x}$ and $\underline{u}$. The noise properties are the same as those given for the linear case.

The five standard steps from Section 6-1-2 remain, but with some additional changes such as the linearization (based on the Taylor series expansion) of the nonlinear state and observation equations about the nominal values of $\underline{x}(t)$ and $\underline{u}(t)$, which are denoted as $\underline{x}^*(t)$ and $\underline{u}^*(t)$, respectively. These steps are as follows:

- **Time Update (Prediction)**

  1. One step ahead state prediction (time propagation): for nonlinear systems this can be performed by integration of Eq. (6-23)

$$\hat{\underline{x}}(k+1|k) = \hat{\underline{x}}(k|k) + \int_{t_k}^{t_{k+1}} \underline{f}\left(\underline{x}(t|t_k), \underline{u}^*(t), t\right) \mathrm{d}t \tag{6-24}$$

  2. Prediction of the covariance matrix of the state error vector: the calculation is executed in the same way as for the linear Kalman filter, but first the following linearization must be performed

$$F_x(t_k) = \frac{\partial \underline{f}\left(\underline{x}(t), \underline{u}(t), t\right)}{\partial \underline{x}(t)}|_{\underline{x}(t)=\underline{x}^*(t_k), \underline{u}(t)=\underline{u}^*(t_k), t=t_k} \tag{6-25}$$

  where $\underline{x}^*(t_k) = \hat{\underline{x}}(k|k)$. Subsequently, the discretization is performed to obtain the discrete-time linearized system dynamics matrix $\Phi(k)$ and the discrete-time system noise covariance matrix Q_d. The error covariance matrix prediction is then obatined as

$$P(k+1|k) = \Phi(k)P(k|k)\Phi^T(k) + Q_d(k) \tag{6-26}$$

- **Measurement Update (Correction)**

  3. Computation of the Kalman gain: a new linearization is performed, now for the observation equation, as

$$H_x(k+1) = \frac{\partial \underline{h}\left(\underline{x}(t), \underline{u}(t), t\right)}{\partial \underline{x}}|_{\underline{x}(t)=\underline{x}^*(t_{k+1}), \underline{u}(t)=\underline{u}^*(t_{k+1}), t=t_{k+1}} \tag{6-27}$$

  where $\underline{x}^*(t_{k+1}) = \hat{\underline{x}}(k+1|k)$. Then, the calculation is carried out as before

$$K(k+1) = P(k+1|k)H^T(k+1)\left[H(k+1)P(k+1|k)H^T(k+1) + R(k+1)\right]^{-1} \tag{6-28}$$

4. Measurement update step: using the nonlinear observation equation

$$\underline{\hat{x}}(k+1|k+1) = \underline{\hat{x}}(k+1|k) + K(k+1)\left[\underline{z}(k+1) - \underline{h}\left(\underline{\hat{x}}(k+1|k), \underline{u}^*(k+1)\right)\right] \quad (6\text{-}29)$$

5. Update of the error covariance matrix: using the linearized observation equation

$$\begin{aligned} P(k+1|k+1) = &\left[I - K(k+1)H_x(k+1)\right] P(k+1|k) \left[I - K(k+1)H_x(k+1)\right]^T \\ &+ K(k+1)R(k+1)K^T(k+1) \end{aligned} \quad (6\text{-}30)$$

Once again, the filter must be initialized by assigning values to $\underline{\hat{x}}(0|0)$ and $P(0|0)$. Q_d is the discrete covariance matrix of the process noise.

The aforementioned steps are described as used in this thesis, although different versions are also possible.

Iterated extended Kalman filter

The nonlinear Kalman filter outlined in this section can be further extended. When the observation model presents high nonlinearities, the Iterated Extended Kalman Filter (IEKF) can be used to improve the convergence of the EKF. In the IEKF formulation, local iterations are used in the measurement update steps; the procedure is explained below (Chu, 2009).

The stage ahead prediction step is the same as in the EKF formulation. However, the nominal state vector is defined as an iterator:

$$\underline{\eta}_i = \underline{x}^*(k+1) \quad (6\text{-}31)$$

The idea now is to update the nominal states using the current measurement and relinearize the system using the improved nominal states. As expected, the iterator starts with the first state prediction and the measurement update equation becomes

$$\begin{aligned} \underline{\eta}_{i+1} = &\underline{\hat{x}}(k+1|k) + \\ &K(k+1)\left\{\underline{z}(k+1) - \underline{h}\left[\underline{\eta}_i, \underline{u}^*(k+1)\right] - H_x(k+1)\left[\underline{\hat{x}}(k+1|k) - \underline{\eta}_i\right]\right\} \\ &\text{for } i = 1, \dots, l \end{aligned} \quad (6\text{-}32)$$

Iterations are performed until

$$\frac{\left\|\underline{\eta}_l - \underline{\eta}_{l-1}\right\|}{\left\|\underline{\eta}_{l-1}\right\|} \leq \varepsilon \quad (6\text{-}33)$$

where ε is a design threshold. The subsequent steps are the same as for the regular EKF, but now using the last state vector obtained after l iterations. Note that when the iterator is updated, matrices related to it must be updated as well. These matrices are the Jacobean matrix H_x and the Kalman gain matrix. The nonlinear predicted observation function should also be updated.

The use of this iterated algorithm brings the main advantage of improved convergence. However, due to the necessary local iterations, the computational load increases as well.

6-1-4 Reliability tests

Verifying whether or not a Kalman filter is operating correctly is key. While in simulations the true state values are known, the same does not hold true for real flights. Therefore, for the former, the estimated states can be compared directly with the true states; for the latter, however, it is more difficult to verify if the estimations are reliable due to the fact that the real states are not available.

One way to test the Kalman filter is to perform a statistical analysis of the innovations. The innovation $\underline{\nu}$ is defined as the difference between the actual observation and the predicted observation:

$$\underline{\nu}(k+1) = \underline{z}(k+1) - H(k+1)\hat{\underline{x}}(k+1|k) \tag{6-34}$$

While the Kalman filter is running, it is possible to compute the mean and covariance of the innovation; if the innovation sequence has a zero-mean white noise characteristic with covariance given by Eq. (6-35), then the Kalman filter is working properly (Lopes, 2011).

$$\sigma_\nu(k+1) = H(k+1)P(k+1|k)H^T(k+1) + R(k+1) \tag{6-35}$$

If the innovation sequence presents a colored-noise behavior, nonzero mean value, or has an incorrect covariance, then something is functioning improperly in the filter. The error might be due to incorrect modeling or incorrect assumption of noise statistics.

6-2 Platform vertical motion estimation

One of the main contributions of this thesis is the design of a vertical motion estimator for an oscillatory platform. To land a quadrotor on a floating platform, knowledge of the platform's motion is beneficial in the sense that such information might be helpful in the control laws. The sea spectrum usually contains one peak for a given frequency. A common model used to describe the vertical motion of a wave-excited floating platform is written as

$$z_{plat}(t) = \sum_{i=1}^{n} A_i \cos\left(\frac{2\pi}{T_i}\cdot t + \phi_i\right) \tag{6-36}$$

where A_i, T_i and ϕ_i are unknown constants. By using a sum of sinusoids, the oscillatory motion is still obtained, but presents a more stochastic behavior in which periods of quiescence may occur. Note that to ensure one peak in the spectrum, the different period constants T_i must be close to one another.

A measurement of z_{plat} can be obtained by subtracting the quadrotor's vertical inertial motion from the relative motion (between the rotorcraft and the floating platform). The quadrotor's inertial motion can be measured using GPS or altitude pressure information. Using current GPS receivers has the problem that the measurements may be off by several meters. Altitude pressure is then the best choice, especially when combined with inertial measurements such as accelerometers, as explained in Section 6-3. The relative motion can be accurately determined using the vision system explained in Chapter 5.

It should be stressed that the approach selected for obtaining a measurement of z_{plat} may be subjected to criticism, specifically since it may not be possible to obtain reliable altitude measurements from a barometric altimeter. As will be seen, the filter's performance was very satisfactory even when tested in the lab with real vision data and large artificial measurement noise. However, this issue can only be properly addressed when testing the actual system in the real sea environment. Nevertheless, sensor quality is constantly improving and if the sensors of today are not adequate, the future will bring better measuring equipment.

The z_{plat} measurements will be contaminated by noise. Three main options are then available for obtaining time derivatives of such measurements: differentiation techniques, low-pass filtering and Kalman optimal filtering. The first choice will lead to amplification of noise while the second will introduce delay. Kalman filtering is then chosen, and the internal model used for the prediction step is given below.

The vertical motion of a floating platform can be approximated locally by linear non-damped oscillatory behavior, which can be described mathematically by

$$\ddot{z}_{plat} = -\omega_{plat} z_{plat} \tag{6-37}$$

where ω_{plat} is a frequency-related parameter corresponding to the location of the two imaginary poles that describe the system. The solution to Eq. (6-37) is given by

$$z_{plat} = A_{plat} \sin\left(2\pi/T_{plat} \cdot t + \phi_{plat}\right) \tag{6-38}$$

which, when differentiated twice with respect to time, yields

$$\ddot{z}_{plat} = -A_{plat} \left(\frac{2\pi}{T_{plat}}\right)^2 \sin\left(2\pi/T_{plat} \cdot t + \phi_{plat}\right) \tag{6-39}$$

This means that the frequency-related parameter ω_{plat} is obtained as a function of the main platform's period T_{plat} as

$$\omega_{plat} = \left(\frac{2\pi}{T_{plat}}\right)^2 \tag{6-40}$$

An EKF is then formulated from the above equations.

6-2-1 Process model

The state vector is defined as:

$$\underline{x} = \begin{bmatrix} z_{plat} & \dot{z}_{plat} & \omega_{plat} \end{bmatrix}^T \tag{6-41}$$

The process model is given by

$$\underline{f}(\underline{x}) = \begin{bmatrix} \dot{z}_{plat} \\ -\omega_{plat} z_{plat} \\ 0 \end{bmatrix} \quad G(\underline{x}) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6-42}$$

and the process noise is

$$\underline{w} = \begin{bmatrix} w_{z_{plat}} & w_{\dot{z}_{plat}} & w_{\omega_{plat}} \end{bmatrix}^T \tag{6-43}$$

From Eq. (6-42) it follows that

$$F_x = \begin{bmatrix} 0 & 1 & 0 \\ -\omega_{plat} & 0 & -z_{plat} \\ 0 & 0 & 0 \end{bmatrix} \tag{6-44}$$

6-2-2 Measurement model

The measurement equation is given as

$$\underline{h}(\underline{x}) = z_{plat} \tag{6-45}$$

from which it holds that

$$H_x = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \tag{6-46}$$

6-2-3 Augmented Kalman filtering and observability issues

Note that the formulation proposed for this estimation procedure is an augmented form of the regular Kalman filter since it estimates not only the system states, but also a system parameter. For that reason, this filter is denoted an augmented Kalman filter.

One of the main issues that must be addressed when designing an observer is the observability (see Appendix B). Specifically for this filter, determining whether or not the system parameter ω_{plat} is observable or not is key. To analyze this problem, a test was performed, which showed that the observability matrix has full rank provided that sufficient excitation is given to the buoy; i.e., if the wave amplitude is zero, then it is not possible to estimate the frequency-related parameter. It is clear that this result will have no affect on the final results.

6-2-4 Initial condition through frequency-domain analysis

It was observed that the Kalman filter proposed for estimation of the platform's vertical motion performs significantly better when the frequency of the oscillatory buoy is relatively known. In fact, in reality, the sea spectrum contains a range of frequencies. However, this range is narrow and there is one peak for non-stormy conditions, corresponding to the frequency for which the signal has the most power. A frequency analysis, using Fourier transforms, can then be performed to determine such peak using a buffered time history of the platform's motion.

The theoretical concepts gathered for this analysis were obtained from (M. Mulder, 2007).

A Continuous-Time Fourier Transform (CTFT) of a continuous-time signal $x(t)$ is given as

$$X(\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t}\mathrm{d}t \tag{6-47}$$

with ω defined as the circular frequency.

Let $x[n]$ be a discrete-time sequence of $x(t)$ such as

$$x[n] = x(n\Delta t) \tag{6-48}$$

where Δt is the sampling time.

The Discrete-Time Fourier Transform (DTFT) of $x[n]$ is given as

$$X(\mathit{\Omega}) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\mathit{\Omega} n} \tag{6-49}$$

where Ω is the discrete-time frequency.

This Fourier transform is of great interest from a theoretical perspective. For practical applications, a Discrete Fourier Transform (DFT) can be used.

Assume now that the signal $x(t)$ is sampled with a sampling frequency $1/\Delta t$, resulting in a discrete signal with N samples. Then, the DFT is defined as

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-jk\frac{2\pi}{N}n} \tag{6-50}$$

or, by considering $W = e^{j\frac{2\pi}{N}}$, written simply as

$$X[k] = \sum_{n=0}^{N-1} x[n] W^{-kn} \tag{6-51}$$

Note that the DFT and DTFT are different.

Finally, the Fast Fourier Transform (FFT), which is a very efficient algorithm to compute DFTs, can be used to perform all the necessary calculations.

6-2-5 Improvement of the estimation algorithm

Several improvements are considered when taking into account real implementation issues. Note that the second-order system (6-37) describes oscillations around zero. It is expected that an offset between vision height and pressure altitude exists, which would hinder the filter's results. Removing such an offset can be accomplished by adding an extra state to the filter. However, simpler solutions can be applied such as computing the mean between the difference of the two measurements from a buffered signal. Since the expected signal is approximately sinusoidal, the expected mean value should be approximately zero.

Furthermore, the sea spectrum frequency range is known, as is maximum wave height. It is thus possible to add state saturations including maximum wave amplitude and maximum rate expected, as well as minimum and maximum wave frequency.

6-2-6 Examples

Frequency-domain analysis example with simulated data

As an example, consider a platform's vertical motion described by a sum of five sinusoids as in the following equation:

$$\begin{aligned} h_{plat} =& 0.4\sin(\frac{2\pi}{5}t) + 0.25\sin(\frac{2\pi}{5.4}t+1) + 0.25\sin(\frac{2\pi}{4.6}t-0.2) \\ &+ 0.1\sin(\frac{2\pi}{5.8}t-2.5) + 0.1\sin(\frac{2\pi}{3.8}t+1.5) \end{aligned} \tag{6-52}$$

Note that this signal contains frequencies f around 0.2 Hz (corresponding to a period T_{plat} of 5 seconds), and the peak in the spectrum lies at exactly 0.2 Hz. The phases of each sinusoid are assigned randomly. White noise with standard deviation equal to 0.01 m is then introduced to simulate measurement errors. A sample of the simulated measurement signal along with a spectral analysis is presented in Figure 6-1, from which a maximum is observed for $f = 0.1953\ Hz$ (meaning that $T_{plat} = 5.12\ sec$). Such a result can be directly used in the Kalman filter as an initial condition.

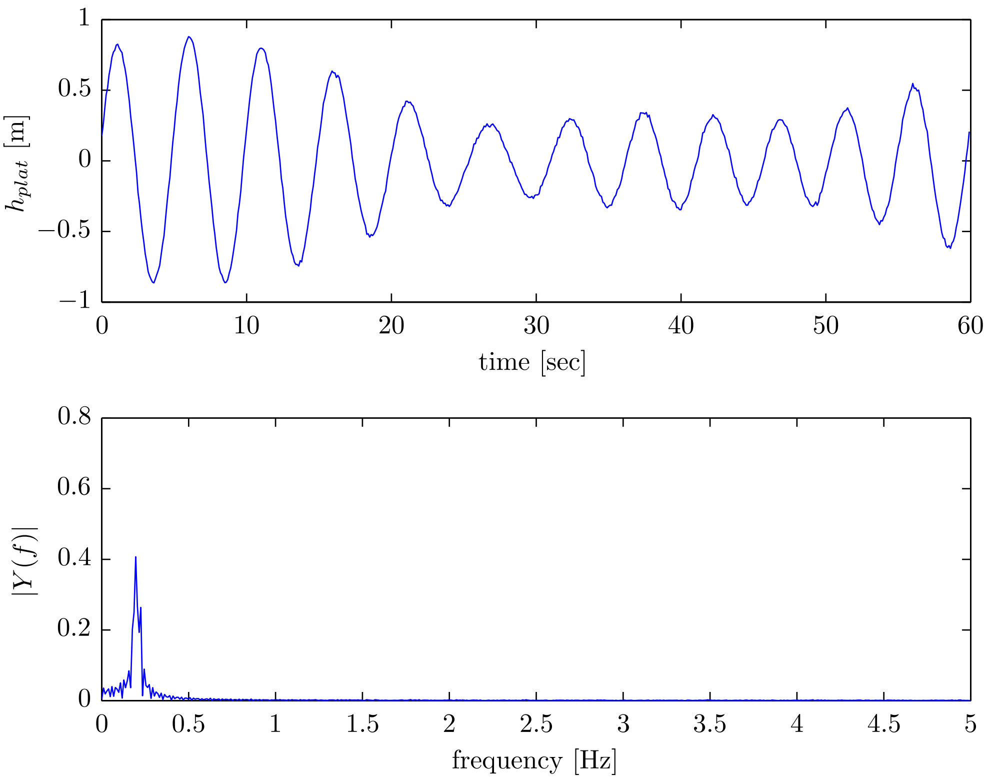


Figure 6-1: 60-second sample of simulated platform height measurement and single-sided amplitude spectrum of h_{plat}

Augmented Kalman filter example with simulated data

An example of simulated measurements is now presented. Consider a to-be-estimated wave-excited motion with peak frequency corresponding to a period of 20 seconds. The filter's initial condition was selected after performing the frequency-domain analysis as demonstrated above. Figure 6-2 shows the true versus estimated states, while Figure 6-3 presents the frequency-related parameter estimation. Finally, a reliability test is carried out to check whether the innovations present a zero-mean white noise behavior. The results are shown in Figure 6-4, from which it is possible to conclude that the estimation procedure was successful.

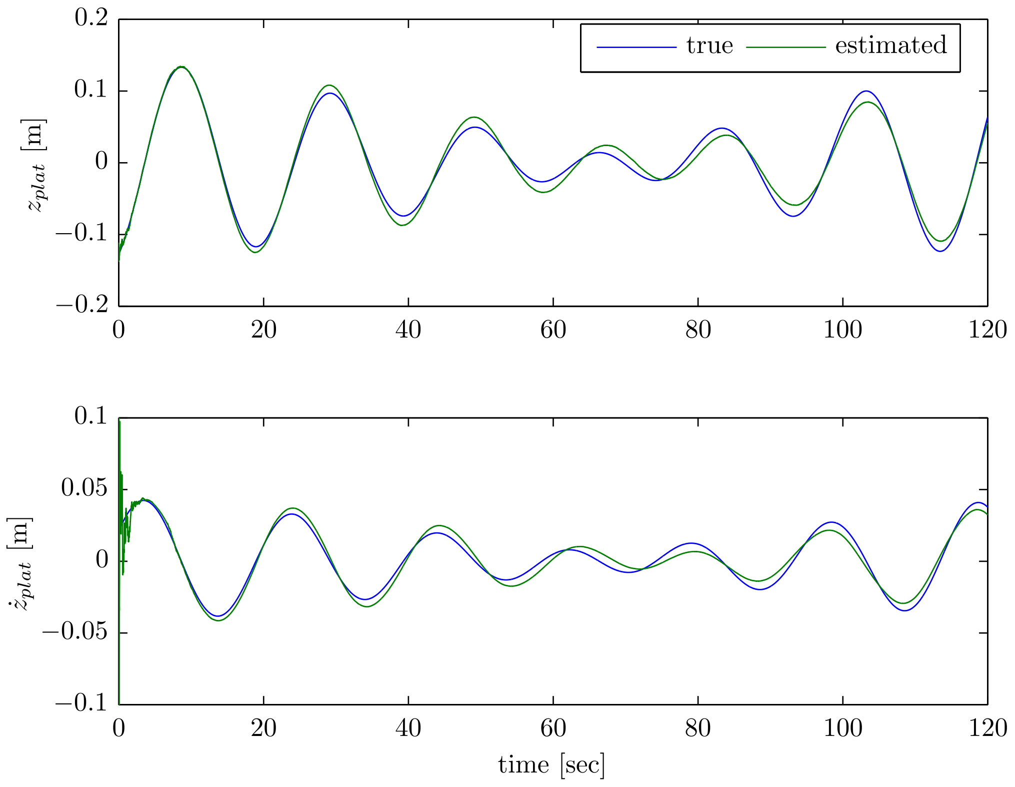


Figure 6-2: Augmented Kalman filter simulation example: true versus estimated states

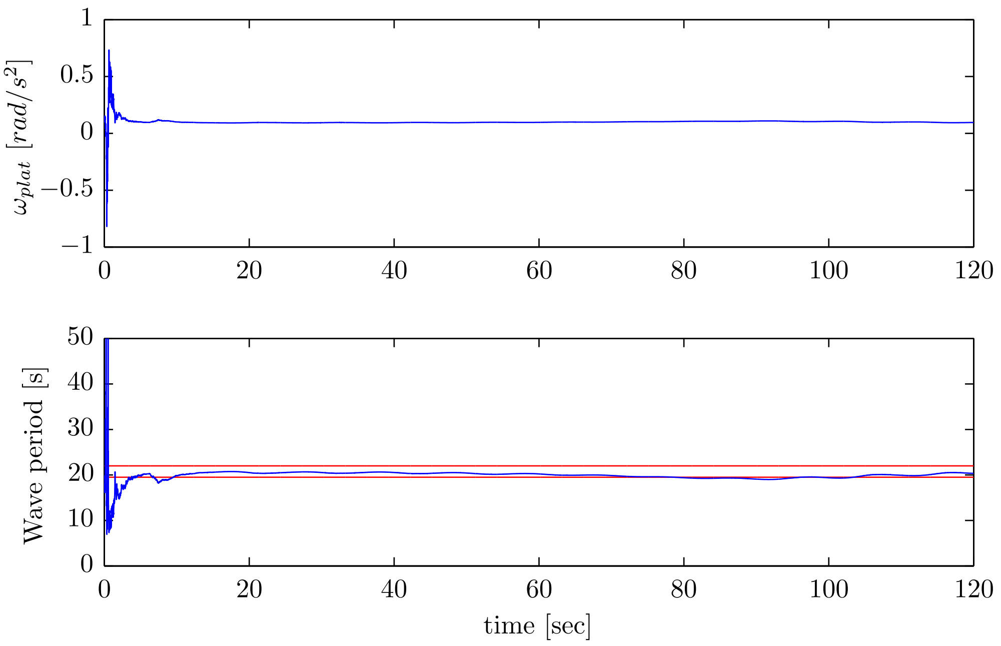


Figure 6-3: Augmented Kalman filter simulation example: frequency parameter estimation; the red lines correspond to minimum and maximum expected wave periods

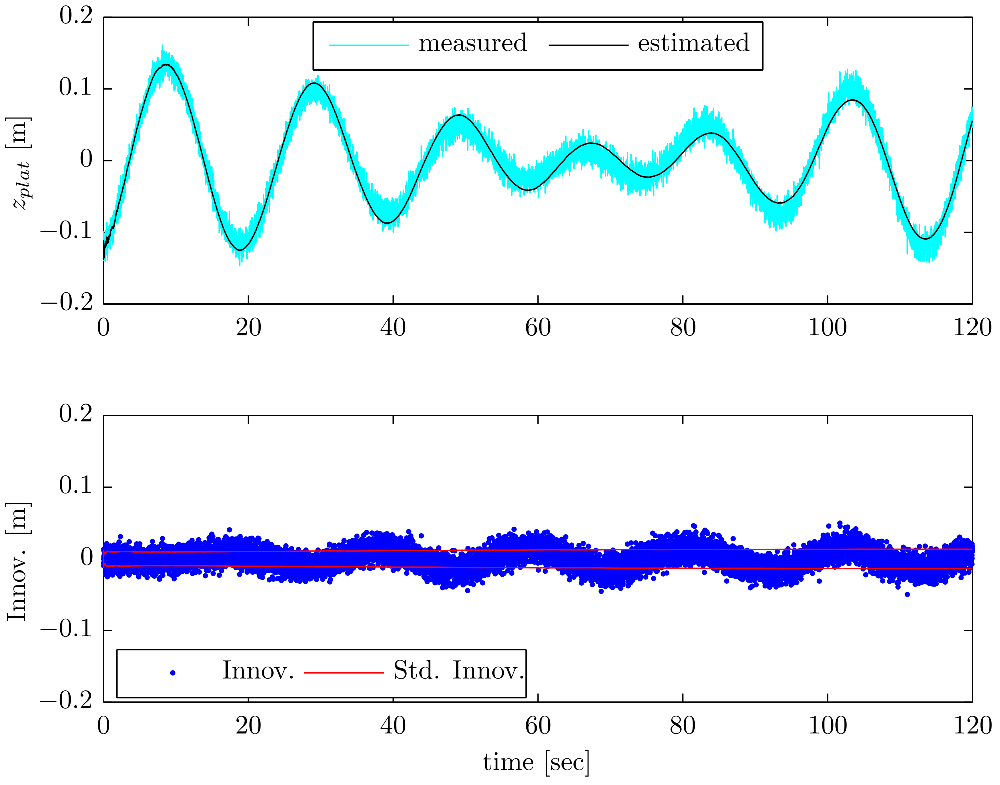


Figure 6-4: Augmented Kalman filter simulation example: innovation sequence ($mean = -2.9 \cdot 10^{-5}\ m$ and $std = 0.0135\ m$); the red lines correspond to the standard deviation of the innovation

Augmented Kalman filter example with real vision data

The estimation algorithm was then tested with real data. For that purpose, camera images were captured while introducing an oscillatory motion to the target. Note that pure sinusoidal behavior cannot be achieved for such tests, meaning that the results obtained here are useful to validate the system considering a more realistic case in which the target's motion presents more stochastic characteristics.

Two examples are considered:

- Example 1: low-noise case, for which the platform's motion contains mostly one low frequency component;
- Example 2: high-noise case, for which the platform's motion contains not only a low frequency component, but also a higher frequency to simulate noise and other effects.

The single-sided amplitude spectra of the signals are presented in Figure 6-5. The stars indicate the frequencies for which the maximum values occurred. Note that for Example 2, although not visible, a second peak was observed for a higher frequency. However, as expected, the magnitude of this peak is lower than the one shown. The results gathered from this frequency analysis are used for the state's initial condition.

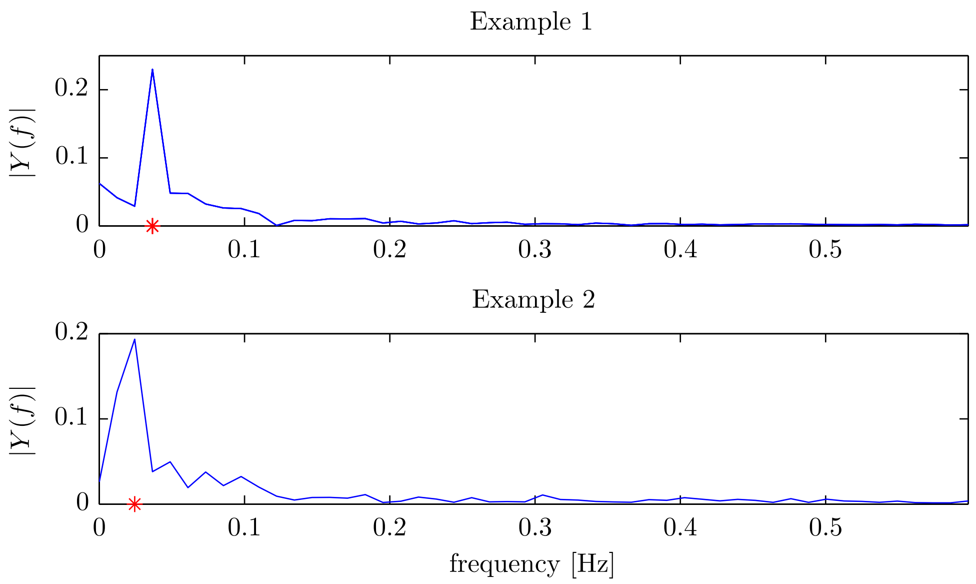


Figure 6-5: Single-sided amplitude spectra of the test signals for Example 1 and Example 2; the stars indicate the frequencies for which the maximum values occurred

Table 6-1 presents the mean values and standard deviations of the innovation sequences for both examples. Furthermore, Figures 6-6 and 6-7 show the measured versus estimated signals, as well as innovation sequences and wave period estimations. Note that for Example 1, the filter was capable of propagating its states correctly even when the target disappeared from the camera image (time period marked with the vertical lines). The results shown clearly indicate that, given certain conditions, the proposed method can be used for such applications.

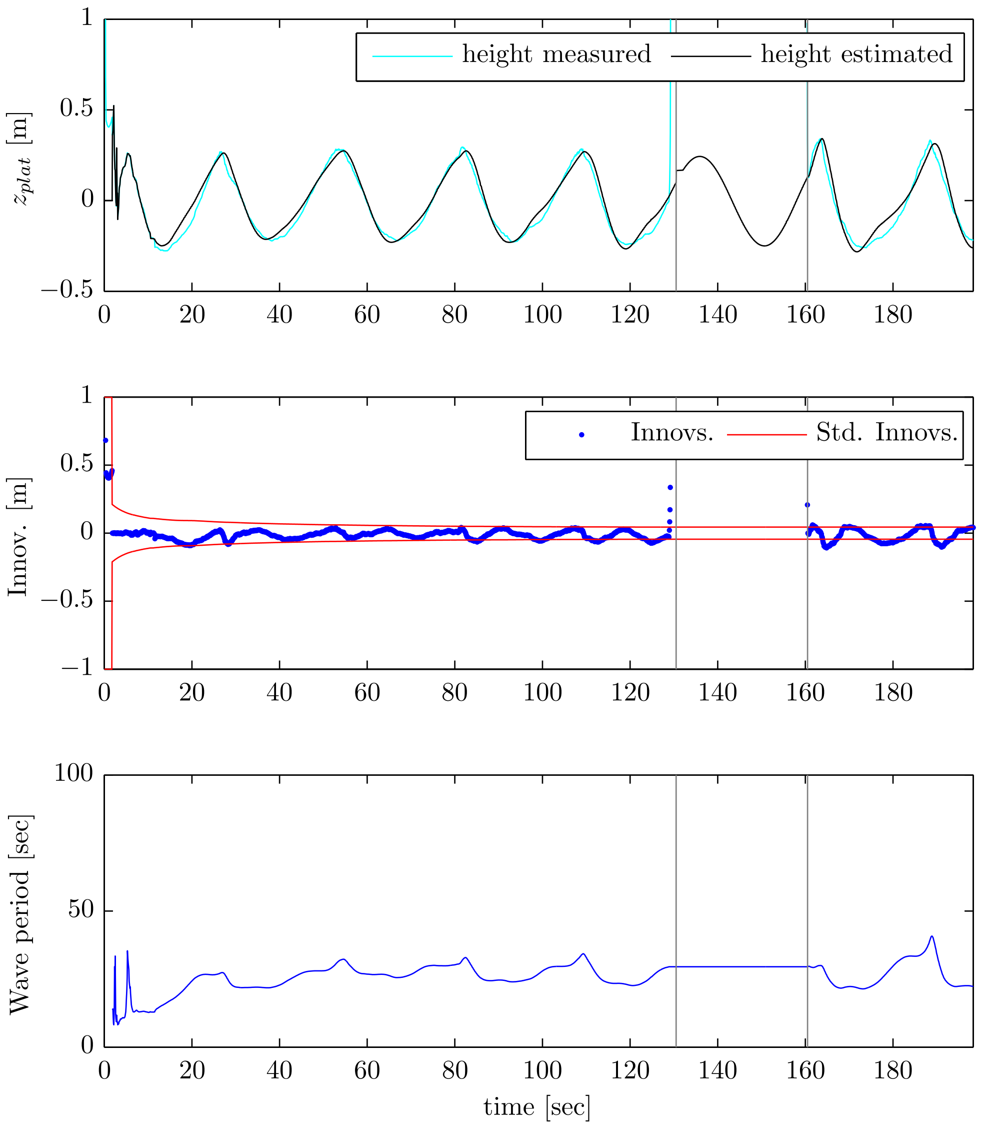


Figure 6-6: Augmented Kalman filter with real vision data for Example 1: estimation results for wave height and period

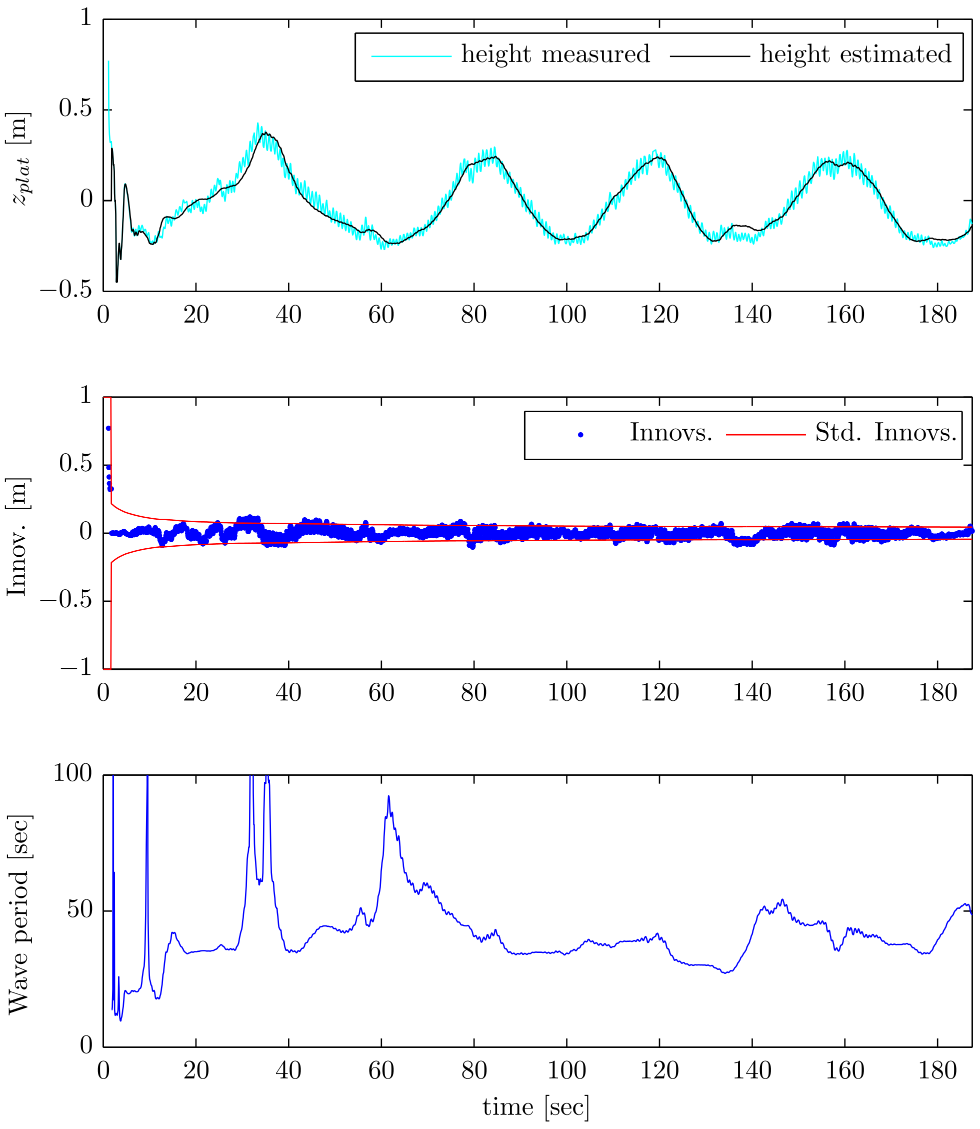


Figure 6-7: Augmented Kalman filter with real vision data for Example 2: estimation results for wave height and period

Table 6-1: Statistical data of the innovation sequences for the Augmented Kalman filter tests with real vision data

Innovations for	Mean (m)	Std. deviation (m)
Example 1	$-4.35\cdot10^{-5}$	0.0445
Example 2	$6.93\cdot10^{-4}$	0.0445

6-3 Position and velocity estimation

The update rate from the camera system is low, and by itself does not produce velocity measurements. Therefore, there are multiple advantages to integrating vision with IMU data, which can be achieved using the Kalman filter theory explained in this chapter. The sensor integration concept is first explained, followed by the process and observation (measurement) models used for position/velocity estimation.

6-3-1 Sensor integration principle

Recall that the Kalman filter steps are divided into two main phases: prediction and correction. The prediction phase propagates the state using the process model, while the correction phase updates the filter's states using measurements. Note that there is no requirement to execute these two phases one after the other. In fact, it is possible to implement the filter in such a way that the prediction and correction steps are performed asynchronously.

Moreover, the Kalman filter equations can be constructed such that the process model uses IMU information as input (more specifically, accelerations obtained from accelerometer measurements). The filter then runs with different frequencies. The main filter's frequency represents the prediction steps using IMU data; meanwhile, correction steps are taken whenever a new vision measurement becomes available.

Note that the same principle can be used for GPS, barometric altimeters or other position measurements.

6-3-2 Process model

The objective of this filter is to estimate position and velocity. However, as will be seen, the accelerometer biases are also estimated in the process and are therefore considered as states. The final state vector is then given by:

$$\underline{x} = \begin{bmatrix} x & y & z & u & v & w & \lambda_{A_x} & \lambda_{A_y} & \lambda_{A_z} \end{bmatrix}^T \tag{6-53}$$

Let $T_{ij}^{\mathcal{BN}}$ be the element corresponding to the i^{th} row and j^{th} column of $T^{\mathcal{BN}}$. Then, the system equations are given as:

$$\begin{aligned}
\dot{x} &= T_{11}^{\mathcal{BN}} u + T_{12}^{\mathcal{BN}} v + T_{13}^{\mathcal{BN}} w \\
\dot{y} &= T_{21}^{\mathcal{BN}} u + T_{22}^{\mathcal{BN}} v + T_{23}^{\mathcal{BN}} w \\
\dot{z} &= T_{31}^{\mathcal{BN}} u + T_{32}^{\mathcal{BN}} v + T_{33}^{\mathcal{BN}} w \\
\dot{u} &= A_x - g\sin\theta - qw + rv \\
\dot{v} &= A_y + g\cos\theta\sin\phi - ru + pw \\
\dot{w} &= A_z + g\cos\theta\cos\phi - pv + qu
\end{aligned} \tag{6-54}$$

The three accelerations A_x, A_y and A_z are measured by the accelerometers. A model of this measurement was given in Eq. (3-44), from which it follows that:

$$\begin{aligned}
A_x &= A_{x_m} - \lambda_{A_x} - w_{A_x} \\
A_y &= A_{y_m} - \lambda_{A_y} - w_{A_y} \\
A_z &= A_{z_m} - \lambda_{A_z} - w_{A_z}
\end{aligned} \tag{6-55}$$

By substituting Eqs. (6-55) into Eqs. (6-54), the following is obtained

$$\begin{aligned}
\dot{x} &= T_{11}^{\mathcal{BN}} u + T_{12}^{\mathcal{BN}} v + T_{13}^{\mathcal{BN}} w \\
\dot{y} &= T_{21}^{\mathcal{BN}} u + T_{22}^{\mathcal{BN}} v + T_{23}^{\mathcal{BN}} w \\
\dot{z} &= T_{31}^{\mathcal{BN}} u + T_{32}^{\mathcal{BN}} v + T_{33}^{\mathcal{BN}} w \\
\dot{u} &= A_{x_m} - \lambda_{A_x} - g\sin\theta - qw + rv - w_{A_x} \\
\dot{v} &= A_{y_m} - \lambda_{A_y} + g\cos\theta\sin\phi - ru + pw - w_{A_y} \\
\dot{w} &= A_{z_m} - \lambda_{A_z} + g\cos\theta\cos\phi - pv + qu - w_{A_z}
\end{aligned} \tag{6-56}$$

and given that the biases are assumed as constants

$$\begin{aligned}
\dot{\lambda}_{A_x} &= 0 \\
\dot{\lambda}_{A_y} &= 0 \\
\dot{\lambda}_{A_z} &= 0
\end{aligned} \tag{6-57}$$

The above equations can be used to formulate the process model using the Kalman filter formalism. By writing the process model equations as in Eq. (6-23), it follows that

$$\underline{f}[\underline{x}(t), \underline{u}(t)] = \begin{bmatrix}
T_{11}^{\mathcal{BN}} u + T_{12}^{\mathcal{BN}} v + T_{13}^{\mathcal{BN}} w \\
T_{21}^{\mathcal{BN}} u + T_{22}^{\mathcal{BN}} v + T_{23}^{\mathcal{BN}} w \\
T_{31}^{\mathcal{BN}} u + T_{32}^{\mathcal{BN}} v + T_{33}^{\mathcal{BN}} w \\
A_{x_m} - \lambda_{A_x} - g\sin\theta - qw + rv \\
A_{y_m} - \lambda_{A_y} + g\cos\theta\sin\phi - ru + pw \\
A_{z_m} - \lambda_{A_z} + g\cos\theta\cos\phi - pv + qu \\
0 \\
0 \\
0
\end{bmatrix} \qquad G = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0
\end{bmatrix} \tag{6-58}$$

with

$$\underline{w} = \begin{bmatrix} w_{A_x} & w_{A_y} & w_{A_z} \end{bmatrix}^T \tag{6-59}$$

and finally

$$F_x = \begin{bmatrix} 0 & 0 & 0 & T_{11}^{\mathcal{BN}} & T_{12}^{\mathcal{BN}} & T_{13}^{\mathcal{BN}} & 0 & 0 & 0 \\ 0 & 0 & 0 & T_{21}^{\mathcal{BN}} & T_{22}^{\mathcal{BN}} & T_{23}^{\mathcal{BN}} & 0 & 0 & 0 \\ 0 & 0 & 0 & T_{31}^{\mathcal{BN}} & T_{32}^{\mathcal{BN}} & T_{33}^{\mathcal{BN}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & r & -q & -1 & 0 & 0 \\ 0 & 0 & 0 & -r & 0 & p & 0 & -1 & 0 \\ 0 & 0 & 0 & q & -p & 0 & 0 & 0 & -1 \end{bmatrix} \tag{6-60}$$

Note that these equations require attitude and angular rate information, which are obtained with different estimation procedures. Attitude estimations are determined with a complementary filter (see Appendix C), while angular rates are gathered from gyro measurements.

6-3-3 Measurement model

The horizontal inertial position can be obtained from vision measurements corrected for attitude, while the vertical position can be obtained from altitude pressure. The measurement equation is then given as:

$$\underline{h}(\underline{x}) = \begin{bmatrix} x & y & z \end{bmatrix}^T \tag{6-61}$$

$$H_x(\underline{x}) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{6-62}$$

The Kalman filter results for both inertial position and body velocity states are presented in Figure 6-8. The mean values and standard deviations of the differences between real and estimated states are given below.

Table 6-2: Means and standard deviations of the differences between real and estimated states

state	Mean	Std. deviation
x [m]	-0.0205	0.0045
y [m]	-0.0127	0.0090
z [m]	0.0010	0.0513
u [m/s]	-0.0020	0.0392
v [m/s]	-0.0011	0.1135
w [m/s]	0.0034	1.7935

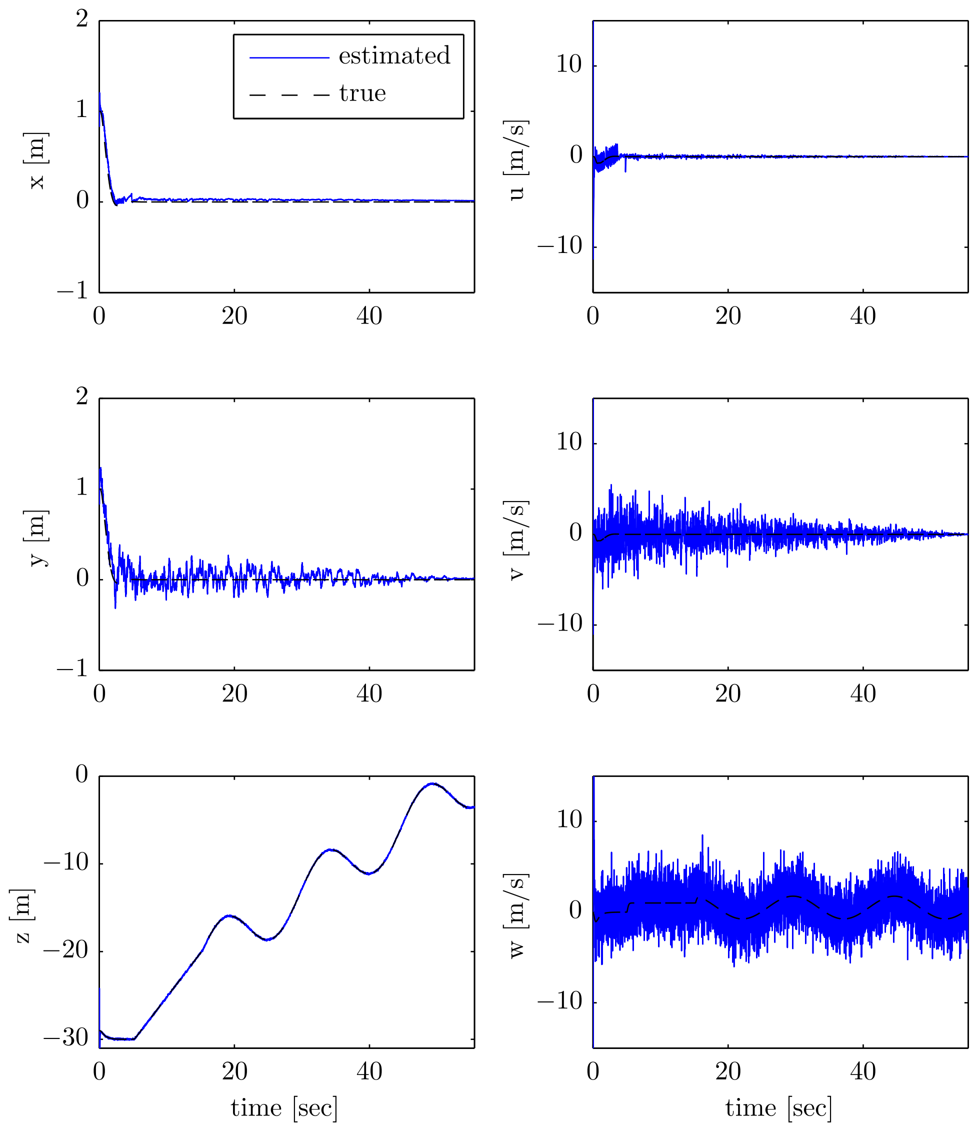


Figure 6-8: Kalman filter results for position/velocity estimation

Chapter 7

Controller design

Quadrotors are dynamically unstable systems, meaning that feedback control is necessary to control such vehicles. Furthermore, these types of rotorcraft are underactuated, making the controller design task very challenging. In this chapter, the controller designs developed for this thesis are presented. The backstepping basics are first introduced to allow for the design of a backstepping controller for the position tracking problem. The necessary fundamentals for incremental-based control design are then given, followed by the controller adaptation to obtain the incremental backstepping control law. The classical (linear) approach is also explained and used for tuning the control gains of all obtained controllers. Then, an autoland mode is developed in accordance with the requirements listed in Chapter 2. The topic of control allocation is covered in the end, and a preliminary simulation of the controllers is shown.

7-1 Backstepping

Backstepping is a Lyapunov-based method to design controllers for nonlinear systems (Kokotović & Arcak, 2001). Lyapunov theory and stability concepts are first outlined here, followed by a trajectory backstepping control design.

7-1-1 Lyapunov stability concepts

First, several important definitions must be introduced for the controller design. Since this aspect of the research pertains to new theoretical developments in the field of control, rigorous mathematical definitions must first be formally presented.

Consider the nonlinear dynamical system described by:

$$\dot{\underline{x}} = f\left(\underline{x}(t), t\right), \quad \underline{x}(t_0) = \underline{x}_0 \tag{7-1}$$

where $\underline{x}(t) \in \mathbb{R}^n$ and $f : \mathbb{R}^n \times \mathbb{R}^+ \mapsto \mathbb{R}^n$ is locally Lipschitz in $\underline{x}$ and piecewise constant in t.

Definition 7.1 (Lipschitz condition)
A function $f(\underline{x}, t)$ satisfies a Lipschitz condition on $\mathcal{D}$ with Lipschitz constant L if

$$\left|f(\underline{x}, t) - f(\underline{y}, t)\right| \leq L \left|\underline{x} - \underline{y}\right|$$

for all points $(\underline{x}, t)$ and $(\underline{y}, t)$ in $\mathcal{D}$.

Note that the Lipschitz continuity is a stronger condition than continuity. For example, the function $f(x) = \sqrt{x}$ is continuous on $\mathcal{D} = [0, \infty)$, but it is not Lipschitz continuous on $\mathcal{D}$ since its slope approaches infinity as x approaches zero.

An equilibrium point $\underline{x}_e \in \mathbb{R}^n$ of (7-1) is such that $f(\underline{x}_e) = 0,\ \forall t \geq 0$. For simplicity, suppose that $\underline{x}_e$ is the origin of $\mathbb{R}^n$. This assumption does not result in any loss of generality since any equilibrium point can be shifted to the origin by simple coordinate transformation. For example, if $\underline{x}_e \neq 0$ the change of variables $\underline{y} = \underline{x} - \underline{x}_e$ would place the equilibrium point at the origin.

Contrary to linear systems, nonlinear systems can have multiple equilibrium points. Therefore, it does not make sense to refer to system stability, but instead to the stability of each equilibrium point. What follows is a definition concerning stability of an equilibrium point at the origin.

Definition 7.2 (Stability in the sense of Lyapunov)
The equilibrium point $\underline{x}_e = 0$ of system (7-1) is considered

- **stable** if, for each $\epsilon > 0$ and $t_0 \leq 0$, there is $\delta(\epsilon, t_0) > 0$ such that
$$|\underline{x}(t_0)| < \delta(\epsilon, t_0) \Rightarrow |\underline{x}(t)| < \epsilon,\ \forall t \geq t_0;$$
- **unstable** if it is not stable
- **attractive** if, for each $\epsilon > 0$ and $t_0 \leq 0$, there is both $\delta(t_0) > 0$ and $T(\epsilon, t_0)$ such that
$$|\underline{x}(t_0)| < \delta(t_0) \Rightarrow |\underline{x}(t)| < \epsilon,\ \forall t \geq t_0 + T;$$
- **uniformly stable** if, for each $\epsilon > 0$ and $t_0 \leq 0$, there is $\delta(\epsilon) > 0$ such that
$$|\underline{x}(t_0)| < \delta(\epsilon) \Rightarrow |\underline{x}(t)| < \epsilon,\ \forall t \geq t_0;$$
- **asymptotically stable** if it is stable, and for any $t_0 > 0$, there exists a $\delta(t_0) > 0$ such that
$$|\underline{x}(t_0)| < \delta(t_0) \Rightarrow \lim_{t\to\infty} \underline{x}(t) = 0;$$
- **uniformly asymptotically stable** if it is uniformly stable, and there exists a $\delta > 0$ for all $t_0 \geq 0$ such that
$$|\underline{x}(t_0)| < \delta \Rightarrow \lim_{t\to\infty} \underline{x}(t) = 0;$$
- **exponentially stable** if, for $\epsilon > 0$, there exists a pair $\delta(\epsilon) > 0$ and $\lambda > 0$ that satisfies
$$|\underline{x}(t_0)| < \delta(\epsilon) \rightarrow |\underline{x}(t)| < \epsilon e^{-\lambda(t-t_0)},\ \forall t \geq t_0 \geq 0.$$

In this definition, $|\cdot|$ denotes any p-norm. Note that the stability properties of the equilibrium point do not depend on the type of norm used in the definition (Vukić, Kuljača, Donlagić, & Tešnjak, 2003).

In the sense of Lyapunov, stability means that all state trajectories starting close enough to the origin (equilibrium point) will remain in the vicinity of that equilibrium state. The main difference between stability and uniform stability is that in the latter case, the stability condition is independent of the initial time t_0. Asymptotic stability is a more strict property as it requires solutions to converge to the origin, while exponential stability requires the converge rate to be exponential.

In some cases, it may not be possible to prove stability of $\underline{x}_e = 0$, but it may still be possible to prove boundedness of the solution using Lyapunov analysis (Khalil, 2002).

Definition 7.3 (Boundedness)
An equilibrium state $\underline{x}_e = 0$ of system (7-1) is

- **uniformly ultimately bounded** if there exist positive constants R, $T(R)$ and b such that $|\underline{x}(t_0)| \leq R$ implies that

$$|\underline{x}(t)| < b, \qquad \forall t > t_0 + T$$

- **globally uniformly ultimately bounded** if it is uniformly ultimately bounded and $R = \infty$.

The constant b is referred to as the ultimate bound.

7-1-2 Lyapunov's direct method

The stability properties of the equilibrium point of the system (7-1) can be studied by using the so-called Lyapunov's direct method (also known as Lyapunov's second method) (Khalil, 2002). This method allows for the study of stability properties without explicitly solving Eq. (7-1). This method is a generalization of the idea that if there is some measure of energy in a system, then stability analysis can be performed by studying the rate of change of such energy.

Consider a ball $R(r)$ of radius r around the origin, $B(r) = \{\underline{x} \in \mathbb{R}^n : |\underline{x}| < r\}$.

Definition 7.4
A continuous function $V(\underline{x})$ is

- **positive definite** on $B(r)$ if $V(\underline{x}) > 0$ and $V(0) = 0$, $\forall \underline{x} \in B(r)$ such that $\underline{x} \neq 0$;
- **positive semi-definite** on $B(r)$ if $V(\underline{x}) \geq 0$ and $V(0) = 0$, $\forall \underline{x} \in B(r)$ such that $\underline{x} \neq 0$;
- **negative (semi-)definite** on $B(r)$ if $-V(\underline{x})$ is positive (semi-)definite;
- **radially unbounded** if $V(0) = 0$, $V(\underline{x}) > 0$ on $\mathbb{R}^n$ $\{0\}$, and $V(\underline{x}) \to \infty$ as $|\underline{x}| \to \infty$

A continuous function $V(\underline{x}, t)$ is

- **positive definite** on $\mathbb{R} \times B(r)$ if there exists a positive definite function $\alpha(\underline{x})$ on B(r) such that
$$V(0,t) = 0,\ \forall t \geq 0 \text{ and } V(\underline{x},t) \geq \alpha(\underline{x}),\ \forall t \geq 0,\ \underline{x} \in B(r);$$
- **radially unbounded** if there exists a radially unbounded function $\alpha(\underline{x})$ such that
$$V(0,t) = 0,\ \forall t \geq 0 \text{ and } V(\underline{x},t) \geq \alpha(\underline{x}),\ \forall t \geq 0,\ \underline{x} \in \mathbb{R}^n;$$
- **decrescent** on $\mathbb{R} \times B(r)$ if there exists a positive definite function $\alpha(\underline{x})$ on B(r) such that
$$V(\underline{x},t) \leq \alpha(\underline{x}),\ \forall t \leq 0,\ \underline{x} \in B(r).$$

The above defined properties are said to be global if they hold true for the entire state space, i.e., $r \to \infty \Rightarrow \underline{x} \in \mathbb{R}^n$.

Based on these definitions, the following theorem can be used to determine stability of a system by studying an appropriate function $V(\underline{x},t)$ (a Lyapunov function) and its time derivative along the trajectories of (7-1), given by

$$\dot{V}(\underline{x},t)|_{\dot{\underline{x}}=f(\underline{x},t)} = \frac{\partial V}{\partial t} + \frac{\partial V}{\partial \underline{x}} f(\underline{x},t) \tag{7-2}$$

Theorem 7.5 (Lyapunov's Direct Method)
Let $V(\underline{x},t) : \mathcal{D} \times \mathbb{R}^+ \mapsto \mathbb{R}^+$ be a continuously differentiable and positive definite function, where $\mathcal{D}$ is an open region containing the origin. The equilibrium point $\underline{x}_e = 0$ is

- **stable** if $\dot{V}(\underline{x},t)|_{\dot{\underline{x}}=f(\underline{x},t)}$ is negative semi-definite for $\underline{x} \in \mathcal{D}$;
- **uniformly stable** if $V(\underline{x},t)$ is decrescent and $\dot{V}(\underline{x},t)|_{\dot{\underline{x}}=f(\underline{x},t)}$ is negative semi-definite for $\underline{x} \in \mathcal{D}$;
- **asymptotically stable** if $\dot{V}(\underline{x},t)|_{\dot{\underline{x}}=f(\underline{x},t)}$ is negative definite for $\underline{x} \in \mathcal{D}$;
- **uniformly asymptotically stable** if $V(\underline{x},t)$ is decrescent and $\dot{V}(\underline{x},t)|_{\dot{\underline{x}}=f(\underline{x},t)}$ is negative definite for $\underline{x} \in \mathcal{D}$;
- **exponentially stable** if there exist constants c_1, c_2 and c_3 such that
$$c_1|\underline{x}|^2 \leq V(\underline{x},t) \leq c_2|\underline{x}|^2 \text{ and } \dot{V}(\underline{x},t)|_{\dot{\underline{x}}=f(\underline{x},t)} \leq -c_3|\underline{x}|^2,\ \forall t \geq 0,\ \underline{x} \in \mathcal{D}.$$

Proof: the proof for this theorem can be found in Chapter 4 of (Khalil, 2002).

The requirement for negative definiteness of the derivative of the Lyapunov function $\dot{V}(\underline{x},t)$ to guarantee asymptotic stability is quite stringent. It is possible to prove asymptotic stability with negative semi-definiteness alone by using LaSalle's invariance theorem (Khalil, 2002). However, this theorem only applies to time-invariant systems. For time-varying systems, Barbalat's lemma can be used (Krstič, Kanellakopoulos, & Kokotović, 1995).

Lemma 7.6 (Barbalat's Lemma)
Let $\phi : \mathbb{R}^+ \mapsto \mathbb{R}$ be a uniformly continuous function on $[0,\infty)$. Assuming that $\lim_{t\to\infty} \int_0^t \phi(\tau)d\tau$ exists and is finite, then

$$\lim_{t\to\infty} \phi(t) = 0$$

Proof: the proof for this lemma can be found both in Chapter 8 of (Khalil, 2002) and in Appendix A of (Krstič et al., 1995).

Combining this lemma with Lyapunov's direct method leads to the powerful theorem by LaSalle and Yoshizawa.

Theorem 7.7 (LaSalle-Yoshizawa)
Let $\underline{x}_e = 0$ be an equilibrium point of (7-1) and assume that $f(\underline{x}, t)$ is locally Lipschitz in $\underline{x}$ uniformly in t. Let $V : \mathbb{R}^n \mapsto \mathbb{R}^+$ be a continuously differentiable, positive definite and radially unbounded function $V(\underline{x})$ such that

- $\gamma_1(\underline{x}) \leq V(\underline{x}, t) \leq \gamma_2(\underline{x})$
- $\dot{V}(\underline{x}) = \frac{\partial V}{\partial \underline{x}}(\underline{x}) f(\underline{x}, t) \leq -W(\underline{x}) \leq 0$

$\forall t \geq 0$, $\forall \underline{x} \in \mathbb{R}^n$, where γ_1 and γ_2 are continuous positive definite functions and where W is a continuous function. Then, all solutions of (7-1) are globally uniformly bounded and satisfy

$$\lim_{t \to \infty} W\left(\underline{x}(t)\right) = 0$$

Additionally, if $W(x)$ is positive definite, then the equilibrium $\underline{x}_e = 0$ is globally uniformly asymptotically stable.

Proof: the proof for this theorem can be found in Appendix A of (Krstič et al., 1995).

This theorem is very useful because it can be applied without the explicit solution of (7-1), which can be extremely difficult (or impossible) to obtain analytically. However, the determination of an appropriate function $V(\underline{x}, t)$ is not prescribed by this theorem. Hence, it is still necessary to find a convenient Lyapunov function to perform the analysis.

7-1-3 Lyapunov-based control design

The tools presented so far in this chapter are meant for performing stability analysis. What follows is an extension of such concepts to obtain control laws that yield closed-loop systems with desired stability properties. Consider now the nonlinear time-invariant system

$$\dot{\underline{x}} = f(\underline{x}, u), \ \underline{x} \in \mathbb{R}^n, \ u \in \mathbb{R}, \ f(0,0) = 0 \tag{7-3}$$

where u is the control input to the system. The objective is to design a feedback control law $\alpha(\underline{x})$ for u such that the equilibrium $\underline{x}_e = 0$ for the closed-loop system

$$\dot{\underline{x}} = f(\underline{x}, \alpha(\underline{x})) \tag{7-4}$$

is globally asymptotically stable. To obtain stability, a function $V(\underline{x})$ is needed as a Lyapunov candidate, and its derivative along the solutions of (7-4) must satisfy $\dot{V}(\underline{x}) \leq -W(\underline{x})$, where $W(\underline{x})$ is a positive semi-definite function. The straightforward approach is to select a positive definite, radially unbounded function $V(\underline{x})$ and, subsequently, choose $\alpha(\underline{x})$ such that

$$\frac{\partial V}{\partial \underline{x}}(\underline{x}) f(\underline{x}, \alpha(\underline{x})) \leq -W(\underline{x}) \ \forall \underline{x} \in \mathbb{R}^n \tag{7-5}$$

Careful selection of $V(\underline{x})$ and $W(\underline{x})$ is required. For a certain stabilizing control law, it is possible that the condition (7-5) is not satisfied. This problem motivated the definition of the Control Lyapunov Function (CLF).

Definition 7.8 (Control Lyapunov Function)
A smooth positive definite and radially unbounded function $V(\underline{x}) : \mathbb{R}^n \mapsto \mathbb{R}^+$ is called a CLF for (7-3) if:

$$\inf_{u\in\mathbb{R}} \left\{ \frac{\partial V}{\partial \underline{x}}(\underline{x}) f(\underline{x}, u) \right\} < 0, \quad \forall \underline{x} \neq 0$$

Note that given a CLF for a system, a globally stabilizing control law can thus be found. In fact, the existence of a CLF proves global asymptotic stability, as shown in (Artstein, 1983), since it is sufficient condition for the existence of a feedback law that satisfies (7-5).

Various examples showing application of this control theory to simple nonlinear systems can be found in (Sonneveldt, 2010), (Trigo, 2011), and (Acquatella, 2011).

7-1-4 Generic backstepping controller design

The generic backstepping design method is now presented. This method treats the design in a recursive manner using CLFs.

Integrator backstepping

Consider the nonlinear system

$$\dot{x}_1 = f(x_1) + g(x_1)x_2 \tag{7-6}$$

$$\dot{x}_2 = u \tag{7-7}$$

where $(x_1, x_2) \in \mathbb{R}^2$ are the states, $u \in \mathbb{R}$ and $g(x_1) \neq 0$. The control objective is to make x_1 track a smooth reference signal y_r, for which all derivatives are known and bounded. This tracking problem can be converted into a regulation problem by defining the tracking error

$$z_1 = x_1 - y_r \tag{7-8}$$

and writing the dynamics of the new coordinate z_1 as

$$\dot{z}_1 = f(x_1) + g(x_1)x_2 - \dot{y}_r \tag{7-9}$$

To steer this subsystem, x_2 can be regarded as its virtual input. By proper selection x_2, the z_1-subsystem can be made globally asymptotically stable. Take the desired value of x_2 as $x_2^{des} \doteq \alpha$. A CLF must be found such that the stabilizing virtual control law renders its time-derivative along the solutions of (7-9) negative (semi-)definite, i.e.,

$$\dot{V}_1 = \frac{\partial V_1}{\partial z_1} \left[f(x_1) + g(x_1)\alpha - \dot{y}_r \right] \leq W_1(z_1) \tag{7-10}$$

where z_1 is positive definite. V_1 can be selected, for example, as a quadratic function of z_1 such as

$$V_1 = \frac{1}{2} z_1^2 \tag{7-11}$$

yielding

$$\dot{V}_1 = z_1 \dot{z}_1 = z_1[f(x_1) + g(x_1)\alpha - \dot{y}_r] \tag{7-12}$$

from which desired virtual control input laws can be derived, for example

$$\alpha = \frac{1}{g(x_1)}[-c_1 z_1 - f(x_1) + \dot{y}_r], \quad c_1 > 0 \tag{7-13}$$

The subsequent step is to define a second error variable z_2 using the signal x_2^{des} as

$$z_2 = x_2 - x_2^{des} \doteq x_2 - \alpha \tag{7-14}$$

The complete error dynamics are then written as

$$\dot{z}_1 = f(x_1) + g(x_1)x_2 - \dot{y}_r \tag{7-15}$$
$$\dot{z}_2 = u - \dot{\alpha} \tag{7-16}$$

where α can be computed analytically from (7-13) and depends not only on the state x_1, but also on $\dot{y}_r$ and $\ddot{y}_r$.

The regulation of the error z_2 is achieved by way of a second CLF. One possibility is to augment V_1 with a quadratic term that penalizes this tracking error as

$$V_2 = V_1 + \frac{1}{2}z_2^2 \tag{7-17}$$

The time derivative of this CLF is given by

$$\begin{aligned}
\dot{V}_2 &= \dot{V}_1 + z_2\dot{z}_2 \\
&= \frac{\partial V_1}{\partial z_1}\left(f(x_1) + g(x_1)x_2 - \dot{y}_r\right) + z_2(u - \dot{\alpha}) \\
&= \frac{\partial V_1}{\partial z_1}\left[f(x_1) + g(x_1)(z_2 - \alpha) - \dot{y}_r\right] + z_2(u - \dot{\alpha}) \\
&= \frac{\partial V_1}{\partial z_1}\left[f(x_1) + g(x_1)\alpha - \dot{y}_r\right] + z_2\left(\frac{\partial V_1}{\partial z_1}g(x_1) + u - \dot{\alpha}\right) \\
&\le -W_1(z_1) + z_2\left(\frac{\partial V_1}{\partial z_1}g(x_1) + u - \dot{\alpha}\right)
\end{aligned} \tag{7-18}$$

from which the control input u can be designed as

$$u = -c_2 z_2 - \frac{\partial V_1}{\partial z_1}g(x_1) + \dot{\alpha}, \quad c_2 > 0 \tag{7-19}$$

yielding

$$\dot{V}_2 \le -W_1(z_1) - c_2 z_2^2 = -W_2(z_1, z_2) \tag{7-20}$$

Hence, according to the LaSalle-Yoshizawa theorem, the equilibrium $(z_1, z_2) = 0$ is globally asymptotically stable. Furthermore, the tracking problem is solved since $\lim_{t\to\infty} x_1 \to y_r$. Note that selecting the CLFs with quadratic forms is usually the most straightforward approach. However, it should be stressed that other choices are also possible, and in some cases may even result in a more efficient control law; for example, avoiding the cancellation of stabilizing nonlinearities.

Extension to higher order systems

The backstepping procedure outlined for second-order systems can be extended to higher-order systems. The main difference is that there are more virtual states to 'backstep' through. Starting from the state that is furthest from the actual control input, each step of the backstepping technique can be divided into three parts (Sonneveldt, 2010):

1. introduce a virtual control and an error state, and rewrite the current state equation in terms of these;
2. choose a CLF for the system, treating it as a final stage;
3. choose a stabilizing feedback term for the virtual control that makes the CLF stabilizable.

The CLF is augmented at subsequent steps to reflect the presence of new virtual states, but the same three stages are followed at each step.

Consider a nonlinear system of n^{th} order

$$\begin{aligned}
\dot{x}_1 &= f_1(x_1) + g_1(x_1)x_2 \\
\dot{x}_2 &= f_2(x_1, x_2) + g_2(x_1, x_2)x_3 \\
&\vdots \\
\dot{x}_i &= f_i(x_1, x_2, \dots, x_i) + g_i(x_1, x_2, \dots, x_i)x_{i+1} \\
&\vdots \\
\dot{x}_{n-1} &= f_{n-1}(x_1, x_2, \dots, x_{n-1}) + g_{n-1}(x_1, x_2, \dots, x_{n-1})x_n \\
\dot{x}_n &= f_n(x_1, x_2, \dots, x_{n-1}, x_n) + g_n(x_1, x_2, \dots, x_{n-1}, x_n)u
\end{aligned} \tag{7-21}$$

with $x_i \in \mathbb{R}$, $u \in \mathbb{R}$ and $g_i \neq 0$, $\forall i \in \{1, 2, \dots, n\}$.

The objective is to make x_1 track a smooth signal y_r whose first n derivatives are known and bounded. The procedure starts by defining the tracking error as

$$z_i = x_i - \alpha_{i-1}, \quad i = 1, 2, \dots, n \tag{7-22}$$

where $\alpha_0 \doteq y_r$. With the change of coordinates, it is possible to build CLFs recursively for each design step as

$$V_i(z_1, z_2, \dots, z_{i-1}, z_i) = V_{i-1}(z_1, z_2, \dots, z_{i-1}) + \frac{1}{2}z_i^2 \tag{7-23}$$

The time derivative of (7-23) is given by

$$\dot{V}_i(z_1, z_2, \dots, z_{i-1}, z_i) = \dot{V}_{i-1}(z_1, z_2, \dots, z_{i-1}) + z_i\dot{z}_i \tag{7-24}$$

where the dynamics of the error coordinates are

$$\dot{z}_i = \dot{x}_i - \dot{\alpha}_{i-1} = f_i + g_i z_{i+1} + g_i\alpha_i - \dot{\alpha}_{i-1} \tag{7-25}$$

with $x_{n+1} \doteq u$. The virtual control inputs α_j and control input u are then given, for example, by

$$\begin{aligned}\alpha_1 &= \frac{1}{g_1}(-c_1 z_1 - f_1 + \dot{y}_r)\\ \alpha_j &= \frac{1}{g_j}(-c_j z_j - f_j - g_{j-1} z_{j-1} + \dot{\alpha}_{j-1}), \quad j = 2, 3, \ldots, n-1\\ u &= \frac{1}{g_n}(-c_n z_n - f_n - g_{n-1} z_{n-1} + \dot{\alpha}_{n-1})\end{aligned} \tag{7-26}$$

where $c_i > 0$.

Asymptotic tracking of y_r by x_1 is then achieved since the derivative of the n^{th} CLF is

$$\dot{V}_n = -\sum_{i=1}^{n} c_i z_i^2 \tag{7-27}$$

which is negative definite for $z_i \neq 0$. This means that global asymptotic stability of the equilibrium $z_1 = 0$ is guaranteed, resulting in

$$\lim_{t\to\infty} x_1 = y_r \tag{7-28}$$

Command filtering backstepping

Finally, the command filtering approach is introduced. The backstepping design extended to higher order systems presented previously permits the designing of control laws for strict-feedback systems of any order. However, as a system's order becomes greater, there is an increase in complexity for the analytical computation of the virtual control laws' time derivatives ($\dot{\alpha}_i$). An alternative method involving command filters is then used to simplify the problem (Farrell, Polycarpou, Sharma, & Dong, 2009). This command filtering approach not only reduces the derivation load, but also allows for the inclusion of magnitude and rate limits for the each state and control input (Farrell, Polycarpou, & Sharma, 2004).

Consider a second-order system

$$\begin{aligned}\dot{x}_1 &= f_1(x_1) + g_1(x_1)x_2\\ \dot{x}_2 &= f_2(x_1, x_2) + g_2(x_1, x_2)u\end{aligned} \tag{7-29}$$

where $g_1 \neq 0$ and $g_2 \neq 0$, $\forall(x_1, x_2, u) \in \mathbb{R}^3$. Once again, the control objective is to make x_1 track a smooth reference signal y_r. The error coordinates are defined as

$$\begin{aligned}z_1 &= x_1 - y_r\\ z_2 &= x_2 - x_{2,c}\end{aligned} \tag{7-30}$$

where $x_{2,c}$ will be defined later in the design. Consider the virtual control input obtained via the regular backstepping procedure

$$\alpha_1 = \frac{1}{g_1}(-f_1 - c_1 z_1 + \dot{y}_r), \quad c_1 > 0 \tag{7-31}$$

Instead of applying this feedback law directly, this signal is corrected

$$x_{2,c}^0 = \alpha_1 - \chi_2 \tag{7-32}$$

where χ_2 will be defined later in this section. The signal $x_{2,c}^0$ is fed to the command filter which gives $x_{2,c}$ and $\dot{x}_{2,c}$ as outputs.

Let the compensated error coordinates now be defined as

$$\begin{aligned} \overline{z}_1 &= z_1 - \chi_1 \\ \overline{z}_2 &= z_2 - \chi_2 \end{aligned} \tag{7-33}$$

and the control input signal be

$$u^0 = \frac{1}{g_2}(-f_2 - c_2 z_2 - g_1 \overline{z}_1 + \dot{x}_{2,c}), \quad c_2 > 0 \tag{7-34}$$

The signals χ_1 and χ_2 can now be defined as

$$\begin{aligned} \dot{\chi}_1 &= -c_1\chi_1 + g_1(x_{2,c} - x_{2,c}^0) \\ \dot{\chi}_2 &= -c_2\chi_2 + g_2(u - u^0) \end{aligned} \tag{7-35}$$

Let a CLF be given by

$$V = \frac{1}{2}\overline{z}_1^2 + \frac{1}{2}\overline{z}_2^2 \tag{7-36}$$

Its derivative is then

$$\begin{aligned} \dot{V} &= \overline{z}_1(f_1 + g_1 x_2 - \dot{y}_r - \dot{\chi}_1) + \overline{z}_2(f_2 + g_2 u - \dot{x}_{2,c} - \dot{\chi}_2) \\ &= \overline{z}_1[f_1 + g_1 z_2 + g_1(x_{2,c} - x_{2,c}^0) + g_1(\alpha_1 - \chi_2) - \dot{y}_r - \dot{\chi}_1] \\ &+ \overline{z}_2[f_2 + g_2(u - u_0) + g_2 u^0 - \dot{x}_{2,c} - \dot{\chi}_2] \\ &= \overline{z}_1[-c_1 z_1 + g_1\overline{z}_2 + c_1\chi_1] + \overline{z}_2[-c_2 z_2 - g_1\overline{z}_1 + c_2\chi_2] \\ &= -c_1\overline{z}_1^2 - -c_2\overline{z}_2^2 \leq 0 \end{aligned} \tag{7-37}$$

meaning that the origin of the compensated error coordinates is globally asymptotically stable. Furthermore, if the signals χ_1 and χ_2 are small, by proper selection of the command filter parameters, it then follows that the error coordinates have an attractive origin. The complete stability proof, which uses singular perturbation theory, can be found in (Farrell et al., 2009).

Further designs

The backstepping control design can be further extended in order to allow the controller to deal with model uncertainties. A so-called robust backstepping controller can be obtained by adding nonlinear damping terms. Alternatively, it is possible to explore the concept of adaptability to cope with model mismatches and design adaptive backstepping controllers. These designs, however, are beyond the scope of this research. For reference, the interested reader can find studies on these topics in (Sonneveldt, 2010), (Trigo, 2011), and (Acquatella, 2011).

7-1-5 Backstepping position controller design

The theoretical framework for designing backstepping-based control laws given in the previous sections can now be applied to the quadrotor model described in Chapter 3. The control objective is to track a desired reference trajectory. More specifically, the goal is to regulate the horizontal position to zero and track an altitude set point. In this backstepping controller design, force and torque disturbances are not considered. Recall the quadrotor model, given here again for convenience:

$$\dot{\underline{p}} = T^{\mathcal{BN}}\underline{V} \tag{7-38}$$

$$\dot{\underline{V}} = m^{-1}\left(\underline{F} + \underline{W}\right) - \underline{\omega} \times \underline{V} \tag{7-39}$$

$$\dot{\underline{\Theta}} = N\underline{\omega} \tag{7-40}$$

$$\dot{\underline{\omega}} = J^{-1}(-\underline{\omega} \times J\underline{\omega} + \underline{M}) \tag{7-41}$$

The backstepping controller is then designed using a cascaded four-loop scheme as depicted in Figure 7-1. The complete derivation of the backstepping control laws for position tracking is given below.

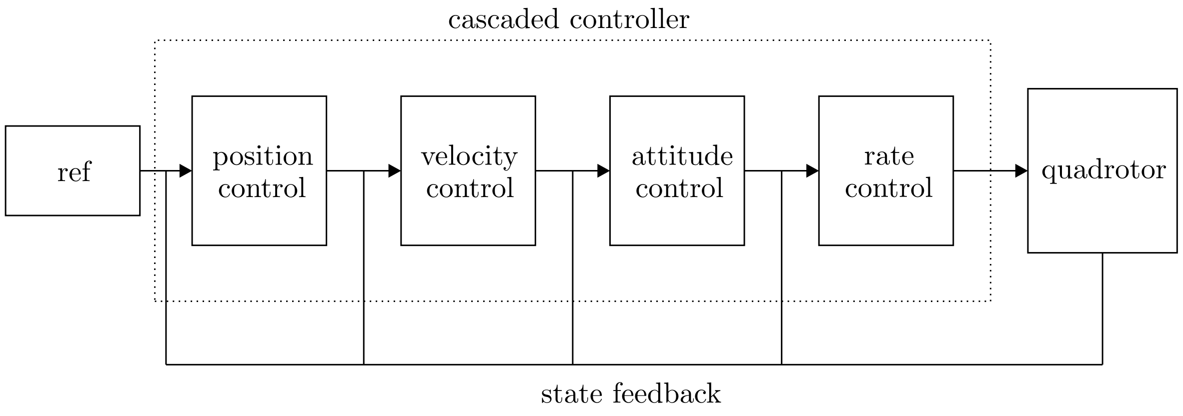


Figure 7-1: Generic representation of the cascaded control structure

Position control

The design process starts by defining the error

$$\underline{z_p} = \underline{p} - \underline{p}^{des} \tag{7-42}$$

from which it follows

$$\dot{\underline{z_p}} = \dot{\underline{p}} - \dot{\underline{p}}^{des} \tag{7-43}$$

A CLF can then be constructed as

$$V_{\underline{p}} = V_1 = \frac{1}{2}\underline{z_p}^T\underline{z_p} \tag{7-44}$$

and by differentiating with respect to time it yields

$$\begin{aligned}\dot{V}_{\underline{p}} &= \underline{z}_{\underline{p}}^T \dot{\underline{z}}_{\underline{p}} \\ &= \underline{z}_{\underline{p}}^T (\dot{\underline{p}} - \dot{\underline{p}}^{des}) \\ &= \underline{z}_{\underline{p}}^T (T^{\mathcal{BN}} \underline{V} - \dot{\underline{p}}^{des})\end{aligned} \tag{7-45}$$

One possibility to make this derivative negative is to ensure

$$T^{\mathcal{BN}} \underline{V} - \dot{\underline{p}}^{des} = -K_{\underline{p}} \underline{z}_{\underline{p}} \tag{7-46}$$

where $K_{\underline{p}}$ is a diagonal matrix such that $K_{\underline{p}} > 0$.

The desired velocity vector can then be derived as

$$\underline{V}^{des} = T^{\mathcal{NB}}(-K_{\underline{p}} \underline{z}_{\underline{p}} + \dot{\underline{p}}^{des}) \tag{7-47}$$

Velocity control

With the desired velocity command computed in the previous step, the velocity control loop design begins with the definition of the error

$$\underline{z}_{\underline{V}} = \underline{V} - \underline{V}^{des} \tag{7-48}$$

followed by

$$\dot{\underline{z}}_{\underline{V}} = \dot{\underline{V}} - \dot{\underline{V}}^{des} \tag{7-49}$$

The first CLF is then extended to penalize this new error as

$$\begin{aligned}V_2 =& V_1 + V_{\underline{V}} \\ =& V_1 + \frac{1}{2} \underline{z}_{\underline{V}}^T \underline{z}_{\underline{V}}\end{aligned} \tag{7-50}$$

The time derivative of V_2 yields

$$\begin{aligned}\dot{V}_2 =& \dot{V}_1 + \underline{z}_{\underline{V}}^T \dot{\underline{z}}_{\underline{V}} \\ =& \underline{z}_{\underline{p}}^T (T^{\mathcal{BN}} \underline{V} - \dot{\underline{p}}^{des}) + \underline{z}_{\underline{V}}^T (\dot{\underline{V}} - \dot{\underline{V}}^{des}) \\ =& \underline{z}_{\underline{p}}^T \left(T^{\mathcal{BN}} (\underline{z}_{\underline{V}} + \underline{V}^{des}) - \dot{\underline{p}}^{des} \right) + \underline{z}_{\underline{V}}^T (\dot{\underline{V}} - \dot{\underline{V}}^{des}) \\ =& \underline{z}_{\underline{p}}^T \left(T^{\mathcal{BN}} \underline{V}^{des} - \dot{\underline{p}}^{des} \right) + \underline{z}_{\underline{V}}^T (\dot{\underline{V}} - \dot{\underline{V}}^{des} + (T^{\mathcal{BN}})^T \underline{z}_{\underline{p}})\end{aligned} \tag{7-51}$$

where the relation $\underline{V} = \underline{z}_{\underline{V}} + \underline{V}^{des}$ has been used.

From the position control step, it follows that the term $\underline{z}_{\underline{p}}^T \left(T^{\mathcal{BN}} \underline{V}^{des} - \dot{\underline{p}}^{des} \right)$ is already negative. Therefore, the desired commands can be obtained from

$$\dot{\underline{V}} - \dot{\underline{V}}^{des} + (T^{\mathcal{BN}})^T \underline{z}_{\underline{p}} = -K_{\underline{V}} \underline{z}_{\underline{V}} \tag{7-52}$$

with $K_{\underline{V}}$ being a diagonal matrix such that $K_{\underline{V}} > 0$.

By expanding Eq. (7-52), the following is obtained

$$\begin{bmatrix} -g\sin\theta + rv - qw \\ g\cos\theta\sin\phi + pw - ru \\ -F_z/m + g\cos\theta\cos\phi + qu - pv \end{bmatrix} - \underline{\dot{V}}^{des} + (T^{\mathcal{BN}})^T \underline{z_p} = -K_{\underline{V}}\underline{z_V} \tag{7-53}$$

and, as is known, control of the body longitudinal velocity u is achieved through θ and lateral velocity v through ϕ. Therefore, the desired attitude Euler angle equations for pitch and roll are given as

$$\begin{aligned} \theta^{des} &= \arcsin\left\{-\frac{1}{g}\left[-K_u z_u + \dot{u}^{des} + qw - rv - \left((T^{\mathcal{BN}})^T \underline{z_p}\right)_1\right]\right\} \\ \phi^{des} &= \arcsin\left\{\frac{1}{g\cos\theta}\left[-K_v z_v + \dot{v}^{des} + ru - pw - \left((T^{\mathcal{BN}})^T \underline{z_p}\right)_2\right]\right\} \end{aligned} \tag{7-54}$$

where $\left((T^{\mathcal{BN}})^T \underline{z_p}\right)_1$ and $\left((T^{\mathcal{BN}})^T \underline{z_p}\right)_2$ are the first and second elements of $\left((T^{\mathcal{BN}})^T \underline{z_p}\right)$, respectively. The yaw angle can also be controlled as desired with ψ^{des}.

Moreover, the desired thrust command is given by

$$F_z = -m\left[-K_w z_w + \dot{w}^{des} - g\cos\theta\cos\phi + pv - qu - \left((T^{\mathcal{BN}})^T \underline{z_p}\right)_3\right] \tag{7-55}$$

where $\left((T^{\mathcal{BN}})^T \underline{z_p}\right)_3$ is the third element of $\left((T^{\mathcal{BN}})^T \underline{z_p}\right)$.

Attitude control

Again, the attitude error is defined as

$$\underline{z_\Theta} = \underline{\Theta} - \underline{\Theta}^{des} \tag{7-56}$$

and

$$\underline{\dot{z}_\Theta} = \underline{\dot{\Theta}} - \underline{\dot{\Theta}}^{des} \tag{7-57}$$

V_2 is now extended to penalize the attitude error as

$$\begin{aligned} V_3 &= V_2 + V_{\underline{\Theta}} \\ &= V_2 + \frac{1}{2}\underline{z_\Theta}^T \underline{z_\Theta} \end{aligned} \tag{7-58}$$

with time derivative

$$\begin{aligned} \dot{V}_3 &= \dot{V}_2 + \dot{V}_{\underline{\Theta}} \\ &= \underline{z_p}^T\left(T^{\mathcal{BN}}\underline{V}^{des} - \underline{\dot{p}}^{des}\right) + \underline{z_V}^T(\underline{\dot{V}} - \underline{\dot{V}}^{des} + (T^{\mathcal{BN}})^T \underline{z_p}) + \underline{z_\Theta}^T \underline{\dot{z}_\Theta} \\ &\le \underline{z_V}^T(\underline{\dot{V}} - \underline{\dot{V}}^{des} + (T^{\mathcal{BN}})^T \underline{z_p}) + \underline{z_\Theta}^T \underline{\dot{z}_\Theta} \end{aligned} \tag{7-59}$$

Note, again, that the term $\underline{z_p}^T\left(T^{\mathcal{BN}}\underline{V}^{des} - \underline{\dot{p}}^{des}\right)$ is already negative from the position control step. Furthermore, the $\underline{\dot{V}}$-equation is not linear in the attitude parameters, meaning that the procedure followed for the previous step cannot be directly used. Assuming small attitude angles, the velocity equations are then approximated as

$$\begin{aligned}\dot{u} &= -g\sin\theta + rv - qw \approx -g\cdot\theta + rv - qw \\ \dot{v} &= g\cos\theta\sin\phi + pw - ru \approx g\cos\theta\cdot\phi + pw - ru\end{aligned} \tag{7-60}$$

and the following equation is constructed

$$\underline{\dot{V}} = \underline{f}_{\underline{V}} + G_{\underline{V}}\underline{\Theta} \tag{7-61}$$

with

$$G_{\underline{V}} = \begin{bmatrix} 0 & -g & 0 \\ g\cos\theta & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{7-62}$$

and $f_{\underline{V}}$ containing all the remaining terms.

With this approximation it follows that

$$\begin{aligned}\dot{V}_3 &\leq \underline{z}_{\underline{V}}^T\left(\underline{f}_{\underline{V}} + G_{\underline{V}}\underline{\Theta} - \underline{\dot{V}}^{des} + (T^{\mathcal{BN}})^T\underline{z}_{\underline{p}}\right) + \underline{z}_{\underline{\Theta}}^T\underline{\dot{z}}_{\underline{\Theta}} \\ &= \underline{z}_{\underline{V}}^T\left[\underline{f}_{\underline{V}} + G_{\underline{V}}(\underline{z}_{\underline{\Theta}} + \underline{\Theta}^{des}) - \underline{\dot{V}}^{des} + (T^{\mathcal{BN}})^T\underline{z}_{\underline{p}}\right] + \underline{z}_{\underline{\Theta}}^T(N\underline{\omega} - \underline{\dot{\Theta}}^{des}) \\ &= \underline{z}_{\underline{V}}^T\left(\underline{f}_{\underline{V}} + G_{\underline{V}}\underline{\Theta}^{des} - \underline{\dot{V}}^{des} + (T^{\mathcal{BN}})^T\underline{z}_{\underline{p}}\right) + \underline{z}_{\underline{\Theta}}^T(N\underline{\omega} - \underline{\dot{\Theta}}^{des} + G_{\underline{V}}^T\underline{z}_{\underline{V}}) \\ &\leq \underline{z}_{\underline{\Theta}}^T(N\underline{\omega} - \underline{\dot{\Theta}}^{des} + G_{\underline{V}}^T\underline{z}_{\underline{V}})\end{aligned} \tag{7-63}$$

Note that the last inequality can be guaranteed by the velocity control step.

A stabilizing control law can be computed from

$$N\underline{\omega} - \underline{\dot{\Theta}}^{des} + G_{\underline{V}}^T\underline{z}_{\underline{V}} = -K_{\underline{\Theta}}\underline{z}_{\underline{\Theta}} \tag{7-64}$$

with $K_{\underline{\Theta}}$ once again being a diagonal control matrix such that $K_{\underline{\Theta}} > 0$.

The desired angular rate is then calculated as

$$\underline{\omega}^{des} = N^{-1}(-K_{\underline{\Theta}}\underline{z}_{\underline{\Theta}} + \underline{\dot{\Theta}}^{des} - G_{\underline{V}}^T\underline{z}_{\underline{V}}) \tag{7-65}$$

where N^{-1} is the transformation from Euler angle rates to body rates given in Eq. (3-6).

Rate control

Finally, the angular rate error is defined as

$$\underline{z}_{\underline{\omega}} = \underline{\omega} - \underline{\omega}^{des} \tag{7-66}$$

meaning that

$$\underline{\dot{z}}_{\underline{\omega}} = \underline{\dot{\omega}} - \underline{\dot{\omega}}^{des} \tag{7-67}$$

The CLF is augmented one last time to penalize the angular rate errors as

$$\begin{aligned}V_4 &= V_3 + V_{\underline{\omega}} \\ &= V_3 + \frac{1}{2}\underline{z}_{\underline{\omega}}^T\underline{z}_{\underline{\omega}}\end{aligned} \tag{7-68}$$

with time derivative

$$\begin{aligned}
\dot{V}_4 &= \dot{V}_3 + \underline{z}_{\underline{\omega}}^T \underline{\dot{z}}_{\underline{\omega}} \\
&\leq \underline{z}_{\underline{\Theta}}^T (N\underline{\omega} - \underline{\dot{\Theta}}^{des} + G_{\underline{V}}^T \underline{z}_{\underline{V}}) + \underline{z}_{\underline{\omega}}^T (\underline{\dot{\omega}} - \underline{\dot{\omega}}^{des}) \\
&= \underline{z}_{\underline{\Theta}}^T \left[N(\underline{z}_{\underline{\omega}} + \underline{\omega}^{des}) - \underline{\dot{\Theta}}^{des} + G_{\underline{V}}^T \underline{z}_{\underline{V}} \right] + \underline{z}_{\underline{\omega}}^T (J^{-1}(-\underline{\omega} \times J\underline{\omega} + \underline{M}) - \underline{\dot{\omega}}^{des}) \\
&= \underline{z}_{\underline{\Theta}}^T (N\underline{\omega}^{des} - \underline{\dot{\Theta}}^{des} + G_{\underline{V}}^T \underline{z}_{\underline{V}}) + \underline{z}_{\underline{\omega}}^T (J^{-1}(-\underline{\omega} \times J\underline{\omega} + \underline{M}) - \underline{\dot{\omega}}^{des} + N^T \underline{z}_{\underline{\Theta}})
\end{aligned} \tag{7-69}$$

Note that the term $\underline{z}_{\underline{\Theta}}^T (N\underline{\omega}^{des} - \underline{\dot{\Theta}}^{des} + G_{\underline{V}}^T \underline{z}_{\underline{V}})$ is already negative from the attitude control step. Therefore, a stabilizing control law is obtained from the following equation

$$J^{-1}(-\underline{\omega} \times J\underline{\omega} + \underline{M}) - \underline{\dot{\omega}}^{des} + N^T \underline{z}_{\underline{\Theta}} = -K_{\underline{\omega}} \underline{z}_{\underline{\omega}} \tag{7-70}$$

as

$$\underline{M} = J(-K_{\underline{\omega}} \underline{z}_{\underline{\omega}} + \underline{\dot{\omega}}^{des} - N^T \underline{z}_{\underline{\Theta}}) + \underline{\omega} \times J\underline{\omega} \tag{7-71}$$

Control law simplification

The desired signals (virtual inputs) and actual actuator inputs are then summarized as follows:

$$\begin{aligned}
\underline{V}^{des} &= T^{\mathcal{NB}}(-K_{\underline{p}} \underline{z}_{\underline{p}} + \underline{\dot{p}}^{des}) \\
\theta^{des} &= \arcsin\left\{ -\frac{1}{g} \left[-K_u z_u + \dot{u}^{des} + qw - rv \right] \right\} \\
\phi^{des} &= \arcsin\left\{ \frac{1}{g\cos\theta} \left[-K_v z_v + \dot{v}^{des} + ru - pw \right] \right\} \\
F_z &= -m \left[-K_w z_w + \dot{w}^{des} - g\cos\theta\cos\phi + pv - qu \right] \\
\underline{\omega}^{des} &= N^{-1}(-K_{\underline{\Theta}} \underline{z}_{\underline{\Theta}} + \underline{\dot{\Theta}}^{des}) \\
\underline{M} &= J(-K_{\underline{\omega}} \underline{z}_{\underline{\omega}} + \underline{\dot{\omega}}^{des} - N^T \underline{z}_{\underline{\Theta}}) + \underline{\omega} \times J\underline{\omega}
\end{aligned} \tag{7-72}$$

Note that some terms have been removed. This is explained as follows. The design starts by solving the attitude tracking problem. By selecting a CLF as $V_{\underline{\Theta}} = \frac{1}{2}\underline{\Theta}^T\underline{\Theta}$, a control law is obtained for $\underline{\omega}^{des}$ as shown. Subsequently, the controller for the rate loop is obtained by expanding the given CLF to penalize the rate error.

The outer loops are obtained with regular state feedback. However, terms for canceling the nonlinearities and the stabilizing terms corresponding to the derivatives of the desired signals are added.

This modification not only permits reduction of the controller's complexity, but also facilitates the gain tuning procedure, which is less straightforward in backstepping designs. This is all achieved while maintaining stability properties as will be seen in Chapter 8. With classical control theory, pole-placement analysis is then used to select the controller gains as shown in Section 7-3. Note that by proper selection of these gains, and more specifically, by proper pole placement, the position/velocity outer loops will have slower dynamics than the attitude/rate loops.

7-2 Incremental backstepping

7-2-1 Time-scale separation

Time-scale separation exists in many nonlinear dynamical systems and can be used in flight dynamics problems to simplify complexity by separating the fast from the slow dynamics. A comprehensive survey on time scales in guidance and control of aerospace systems can be found in (Naidu & Calise, 2001), in which singular perturbation theory is used for the analysis. A brief overview of this concept is presented below.

Consider the nonlinear system

$$\begin{aligned} \dot{x}_s &= f_s(x_s, x_f, u, \epsilon) \\ \epsilon \dot{x}_f &= f_f(x_s, x_f, u, \epsilon) \end{aligned} \tag{7-73}$$

where x_s and x_f are the system's states, u is the control input, f_s and f_f are the nonlinear system equations and ϵ (small value) is known as singular perturbation. Note that when the perturbation is suppressed ($\epsilon = 0$), it results in a reduction of the system's order. In this case, the fast dynamics are so rapid that the fast state x_f reaches a quasi-steady situation in the slow time scale. The system is then described by the slow dynamics of x_s only, subjected to $f_f(x_s, x_f, u, 0) = 0$. Thus, singular perturbation theory allows for simplification of the mathematical model representation via order reduction. The analysis can even be extended to systems with multiple time scales, treating each loop individually.

The time scale separation principle can be also be applied to control systems design. Fast dynamics are then associated with variables with a higher control effectiveness, while slow dynamics correspond to the states that are more weakly affected by the control inputs. Specifically in aerospace control applications, the time-scale separation analysis is used in both attitude and trajectory control problems. For example, in attitude control problems, the attitude parameters are the slow states, while the angular velocities are the fast states.

7-2-2 Incremental form of the system dynamics

Incremental-based control laws are of interest due to their robustness properties. Specifically applied to NDI, it can be shown that a pure linearization cannot be achieved in the presence of model uncertainties and inaccuracies (Sieberling et al., 2010). A more robust version can be obtained with the incremental version of the same controller (Incremental Nonlinear Dynamic Inversion (INDI)). As stated in the literature review, although the advantages of using INDI are clear, it is still not possible to prove closed-loop stability. Therefore, the next step in research is to introduce backstepping.

The concept of incremental backstepping is based on the incremental form of the system dynamics (H. Chen & Zhang, 2008). Instead of computing the complete control input vector, it is possible to calculate only the required variation from the previous input. What follows is an explanation of how to determine such a control input increment. Consider the system

$$\dot{\underline{x}} = \underline{f}(\underline{x}) + G(\underline{x})\underline{u} \tag{7-74}$$

Consider now the Taylor series expansion of Eq. (7-74) to obtain a first-order approximation of $\dot{\underline{x}}$ around the system's current solution $(\underline{x}_0, \underline{u}_0)$

$$\begin{aligned} \dot{\underline{x}} &\approx \dot{\underline{x}}_0 + \frac{\partial}{\partial \underline{x}}[\underline{f}(\underline{x}) + G(\underline{x})\underline{u}]_{\underline{x}_0,\underline{u}_0}(\underline{x} - \underline{x}_0) + \frac{\partial}{\partial \underline{u}}[\underline{f}(\underline{x}) + G(\underline{x})\underline{u}]_{\underline{x}_0,\underline{u}_0}(\underline{u} - \underline{u}_0) = \\ &= \dot{\underline{x}}_0 + \frac{\partial}{\partial \underline{x}}[\underline{f}(\underline{x}) + G(\underline{x})\underline{u}]_{\underline{x}_0,\underline{u}_0}(\underline{x} - \underline{x}_0) + G(\underline{x}_0)(\underline{u} - \underline{u}_0) \end{aligned} \tag{7-75}$$

Eq. (7-75) can be further simplified since the variation $x - x_0$ can be neglected for very small time increments, yielding

$$\dot{\underline{x}} \approx \dot{\underline{x}}_0 + G(\underline{x}_0)(\underline{u} - \underline{u}_0) \tag{7-76}$$

This simplification is explained as follows. Let Eq. (7-75) be a representation of the rotational dynamics, meaning that $\underline{x} = \underline{\omega}$. A change in the control input results in a change in moment, which directly affects the angular accelerations. In turn, angular rates change only after integrating the angular accelerations, hence integrating the control input component. This makes the term $x - x_0$ (specifically to this example, change in angular rates $\underline{\omega} - \underline{\omega}_0$) negligible for small time increments. Note that for this assumption, fast actuators are required.

Finally, the system dynamics can be rewritten as a function of the input increment as

$$\dot{\underline{x}} = \dot{\underline{x}}_0 + G(\underline{x}_0)\mathrm{d}\underline{u} \tag{7-77}$$

in which the total control input is given by

$$\underline{u} = \underline{u}_0 + \mathrm{d}\underline{u} \tag{7-78}$$

Eq. (7-77) can then be utilized in the backstepping analysis for determination of a stabilizing control law. Note, however, that only the input increment can be determined and the complete input is obtained with Eq. (7-78).

7-2-3 Control adaptation to incremental form

To obtain the incremental backstepping control law, the angular rates equation is first expressed in the incremental form as

$$\dot{\underline{\omega}} = \dot{\underline{\omega}}_0 + J^{-1}\mathrm{d}\underline{M} \tag{7-79}$$

and the vertical velocity equation as

$$\dot{w} = \dot{w}_0 - \frac{1}{m}\mathrm{d}F_z \tag{7-80}$$

The angular rate backstepping loop is adapted to render the incremental control law. Recall the time derivative of the CLF V_4 given in (7-69), from which the following relationship was selected to obtain a stabilizing control law:

$$\dot{\underline{\omega}} - \dot{\underline{\omega}}^{des} + N^T \underline{z}_{\Theta} = -K_{\underline{\omega}}\underline{z}_{\underline{\omega}} \tag{7-81}$$

By expressing the rate dynamics in the incremental form, it follows that

$$\dot{\underline{\omega}}_0 + J^{-1}\mathrm{d}\underline{M} - \dot{\underline{\omega}}^{des} + N^T \underline{z}_{\Theta} = -K_{\underline{\omega}}\underline{z}_{\underline{\omega}} \tag{7-82}$$

and consequently, the torque commands are given as increments computed from

$$\mathrm{d}\underline{M} = J(-\underline{\dot{\omega}}_0 - K_{\underline{\omega}}\underline{z}_{\underline{\omega}} + \underline{\dot{\omega}}^{des} - N^T \underline{z}_{\underline{\Theta}}) \tag{7-83}$$

The final controller command is then given by

$$\underline{M} = \underline{M}_0 + \mathrm{d}\underline{M} \tag{7-84}$$

where $\underline{M}_0$ is a vector containing the previous torque inputs given to the system.

The vertical velocity backstepping loop is adapted in a similar way. By treating the CLF for velocity control as a final stage, the following equation was selected to obtain a stabilizing control law:

$$\dot{w} - \dot{w}^{des} = -K_w z_w \tag{7-85}$$

Again, by expressing the dynamics in the incremental form, it follows that

$$\dot{w}_0 - \frac{1}{m}\mathrm{d}F_z - \dot{w}^{des} = -K_w z_w \tag{7-86}$$

meaning that the thrust increment input commands are computed as

$$\mathrm{d}F_z = -m(-\dot{w}_0 - K_w z_w + \dot{w}^{des}) \tag{7-87}$$

and the total input is given by

$$F_z = F_{z0} + \mathrm{d}F_z \tag{7-88}$$

where F_{z0} is the previous thrust input given to the system.

Several considerations are examined when taking a closer look into the control laws obtained. First, one of the main advantages of using incremental backstepping is already clear; as this controller relies less on the model and more on measurements (or estimations), it should in principle be more robust to model mismatches when compared to the standard backstepping controller. However, note that estimation of both angular accelerations and vertical kinematic acceleration is required (these estimations are carefully analyzed below). Furthermore, the inputs $\underline{M}_0$ and F_{z0} cannot be measured in this quadrotor control problem. Thus, an actuator model must be used to estimate such inputs.

Angular accelerations

Incremental forms of control require information about the angular accelerations of the vehicle. In the previously-derived incremental backstepping control law, it was assumed that angular accelerations can be accurately measured. However, the issue of estimating such physical quantities must be carefully addressed. Despite the fact that angular acceleration sensors exist, they are still not commonly used in practice. In previous research at DUT, this problem has been studied and several methods have been proposed to derive angular accelerations from IMU measurements. A brief summary is presented here.

One of the most intuitive methods is to use simple finite differences (Simplicio, 2011). In this case, information about angular velocities is used to derive angular accelerations. More specifically, two consecutive samples of angular velocities are used to perform the necessary calculations. (Sieberling et al., 2010) presents a predictive filter that uses the decoupling properties of an INDI closed-loop. Also in (Sieberling et al., 2010), a five-point scheme is presented. The latter is based on a numerical scheme that makes use of five samples of angular velocities.

In this study, a different method is used; the gyro measurements are filtered with a low-pass first-order filter and the time-derivative of the smoothed signals are obtained as explained in Appendix D. It should be stressed that the filters' time constants should be very small to avoid introducing significant time delays.

Vertical kinematic acceleration

The linear acceleration required to compute the thrust command increment can be obtained as follows

$$\dot{w} = A_z + g\cos\theta\cos\phi + qu - pv \tag{7-89}$$

where A_z is obtained from accelerometer measurements, the local gravity g is assumed as constant with known magnitude, p and q are obtained from gyro measurements and the states θ, ϕ, u and v are estimated.

A simpler version is also possible as

$$\dot{w} = A_z + g\cos\theta\cos\phi \tag{7-90}$$

by neglecting the last two nonlinear terms of (7-89). Note that this approximation should be valid for non-aggressive maneuvers and has the advantage of not introducing estimation errors of u and v into the estimation of $\dot{w}$.

7-3 Classical control

7-3-1 Basic principle

In classical control theory, PID controllers are used as feedback mechanisms to control a system. By adjusting the process control inputs, this type of controller attempts to minimize an error value calculated as the difference between a measured process variable and a desired setpoint. The PID controller is composed of three components: proportional, derivative and integral actions. Proportional action provides the basic state feedback and mostly influences the speed of the closed-loop system, derivative action adds damping to the closed-loop response and integral action is responsible for removing the steady-state errors. The generic control law is then given by

$$u = K_p e(t) + K_d \dot{e}(t) + K_i \int e(t)dt \tag{7-91}$$

where $e(t)$ represents the to-be-controlled error. Note that some applications may require the use of only one or two actions.

Tuning

When a mathematical model is available, the control design and tuning can be achieved via linearization. The process starts by defining an equilibrium point $\underline{x}_e$ around which the model is linearized. Consider a generic nonlinear model of the form

$$\dot{\underline{x}} = f(\underline{x}, \underline{u}) \tag{7-92}$$

where the function $\underline{f}(\underline{x}, \underline{u})$ is continuously differentiable in a domain that contains the equilibrium point $\underline{x}_e$. The linearization around this point gives

$$\dot{\underline{x}} = A\underline{x} + B\underline{u} \tag{7-93}$$

with

$$A = \left.\frac{\partial \underline{f}(\underline{x}, \underline{u})}{\partial \underline{x}}\right|_{\underline{x}=\underline{x}_e, \underline{u}=\underline{u}_e} \quad , \quad B = \left.\frac{\partial \underline{f}(\underline{x}, \underline{u})}{\partial \underline{u}}\right|_{\underline{x}=\underline{x}_e, \underline{u}=\underline{u}_e} \tag{7-94}$$

where $\underline{u}_e$ is the trim control input. If there is a controller that can stabilize the linearized system (7-93), then it can be proven that it also stabilizes the nonlinear system (7-92) in the vicinity of the equilibrium point.

The system described in Chapter 3 is linearized as follows.

$$\begin{array}{llll} \dot{p} = M_p/I_{xx} & \dot{\phi} = p & \dot{u} = -g\theta & \dot{x} = u \\ \dot{q} = M_q/I_{yy} & \dot{\theta} = q & \dot{v} = g\phi & \dot{y} = v \\ \dot{r} = M_r/I_{zz} & \dot{\psi} = r & \dot{w} = -F_z/m + g & \dot{z} = w \end{array} \tag{7-95}$$

With pure proportional action all SISO loops can be controlled, leading to first-order closed loop systems. The dynamics of a first-order system are described by the known generic equation

$$\tau\dot{y} + y = Ku \tag{7-96}$$

meaning that the proportional gain for each control loop can be selected to obtain the desired closed-loop time constant τ.

Integral action is then added to remove undesired possible steady-state errors, with integral gains that are initially very small and are increased progressively until unstability is observed. Finally, these integral gains are reduced to render the desired closed-loop performance.

7-3-2 Cascaded linear trajectory controller design

The controller developed here is based on classical control theory. A cascaded system is designed for which time-scale separation between outer and inner loops is used. The controller structure is the same as depicted in Figure 7-1. All variables are controlled using a SISO scheme as given below. For tuning purposes the design now starts with the fastest loops.

- Angular rate control

$$\begin{aligned} M_p &= K_{p_p}(p_{ref} - p) + K_{p_i}\int(p_{ref} - p)\mathrm{d}t \\ M_q &= K_{q_p}(q_{ref} - q) + K_{q_i}\int(q_{ref} - q)\mathrm{d}t \\ M_r &= K_{rp}(r_{ref} - r) + K_{ri}\int(r_{ref} - r)\mathrm{d}t \end{aligned} \tag{7-97}$$

- Attitude control

$$
\begin{aligned}
p_{ref} &= K_{\phi_p}(\phi_{ref} - \phi) + K_{\phi_i}\int(\phi_{ref} - \phi)\mathrm{d}t \\
q_{ref} &= K_{\theta p}(\theta_{ref} - \theta) + K_{\theta i}\int(\theta_{ref} - \theta)\mathrm{d}t \\
r_{ref} &= K_{\psi_p}(\psi_{ref} - \psi) + K_{\psi_i}\int(\psi_{ref} - \psi)\mathrm{d}t
\end{aligned}
\tag{7-98}
$$

- Velocity control

$$
\begin{aligned}
\phi_{ref} &= K_{vp}(v_{ref} - v) + K_{vi}\int(v_{ref} - v)\mathrm{d}t \\
\theta_{ref} &= K_{up}(u_{ref} - u) + K_{ui}\int(u_{ref} - u)\mathrm{d}t \\
F_z &= K_{wp}(w_{ref} - w) + K_{wi}\int(w_{ref} - w)\mathrm{d}t + mg
\end{aligned}
\tag{7-99}
$$

- Position control

$$
\begin{aligned}
u_{ref} &= K_{xp}(x_{ref} - x) + K_{xi}\int(x_{ref} - x)\mathrm{d}t \\
v_{ref} &= K_{y_p}(y_{ref} - y) + K_{y_i}\int(y_{ref} - y)\mathrm{d}t \\
w_{ref} &= K_{zp}(z_{ref} - z) + K_{zi}\int(z_{ref} - z)\mathrm{d}t
\end{aligned}
\tag{7-100}
$$

The integral terms are neglected initially for the tuning procedure. By inserting the above equations into (7-95) and rearranging to obtain all final equations in the form of (7-96), the proportional gains are selected as a function of the desired closed-loop time constants as follows:

$$
\begin{array}{llll}
K_{p_p} = \frac{I_{xx}}{\tau_p} & K_{\phi_p} = \frac{1}{\tau_\phi} & K_{up} = -\frac{1}{g\tau_u} & K_{xp} = \frac{1}{\tau_x} \\
K_{q_p} = \frac{I_{yy}}{\tau_q} & K_{\theta p} = \frac{1}{\tau_\theta} & K_{vp} = \frac{1}{g\tau_v} & K_{y_p} = \frac{1}{\tau_y} \\
K_{rp} = \frac{I_{zz}}{\tau_r} & K_{\psi_p} = \frac{1}{\tau_\psi} & K_{wp} = -\frac{m}{\tau_w} & K_{zp} = \frac{1}{\tau_z}
\end{array}
\tag{7-101}
$$

The time constants and corresponding gains used for simulation purposes are given in Appendix E.

7-4 Autoland controller mode

The autoland controller mode implemented for simulation purposes is designed according to Chapter 2 (see Figure 2-1 for details). This mode is only valid once the target has been found. To accomplish the mission successfully, three main controller stages are required as shown in Section 2-4: inertial altitude hold, inertial descent and relative descent. What follows is an explanation of how these stages are represented in terms of controller design.

- **Inertial altitude hold.** The quadrotor should regulate its horizontal position to zero (with the target being the reference), and hold its altitude in the navigation frame (with respect to MSL). Then
$$\underline{p}^{des} = (0, 0, -h^{des})^T \tag{7-102}$$
and
$$\dot{\underline{p}}^{des} = (0, 0, 0)^T \tag{7-103}$$

- **Inertial descent.** This stage is required for landing purposes in cases where the quadrotor is not yet close to the buoy. Vertical velocity is then controlled, but with respect to MSL. No vertical position is controlled, meaning that
$$K_z = 0 \text{ i.e., } K_{\underline{p}} = \begin{bmatrix} K_x & 0 & 0 \\ 0 & K_y & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{7-104}$$
Furthermore, the objective is still to regulate the horizontal position to zero, and
$$\dot{\underline{p}}^{des} = (0, 0, \dot{z}^{des})^T \tag{7-105}$$

- **Relative descent.** This stage is required for the final landing phase, when the platform is close. Since the objective is to control relative sink rate with respect to the target, the inertial vertical velocity must be adjusted to cope with the buoy's dynamics. As in the previous stage, the vertical position gain is set to zero and the objective is to regulate the horizontal position to zero. Moreover, since the relative motion between the quadrotor and the platform is given by
$$\dot{z}_{rel} = \dot{z}_{quad} - \dot{z}_{plat} \tag{7-106}$$
the following command derivative is designed to obtain the desired relative TD sink rate:
$$\dot{\underline{p}}^{des} = (0, 0, \dot{z}^{des}_{TD} + \dot{z}_{plat})^T \tag{7-107}$$
From this equation, it follows that an estimation of the platform's vertical motion is required. For that reason, an augmented Kalman filter has been developed in this thesis and is explained in Section 6-2.

7-5 Control allocation

To generate the desired control forces and moments, the necessary actuator inputs must be computed. There are two main options to perform such calculations: nonlinear control allocation or linear control allocation around a trim operation point. For this section, the rotors are numbered as follows: rotors 1 and 2 are attached to the front and back motors, respectively, and rotors 3 and 4 are attached to the left and right motors, respectively.

7-5-1 Nonlinear

From Eqs. (3-20) and (3-24), the control forces and moments produced by each rotor can be approximated by

$$F \approx b\Omega^2 \quad , \quad M \approx d\Omega^2 \tag{7-108}$$

where b and d are thrust and moment factors for the hovering condition. Therefore, assuming a distance l from each rotor to the quadrotor's center of mass, the following relation holds true:

$$\begin{bmatrix} M_x \\ M_y \\ M_z \\ F_z \end{bmatrix} = \begin{bmatrix} 0 & 0 & -lb & lb \\ lb & -lb & 0 & 0 \\ -d & -d & d & d \\ b & b & b & b \end{bmatrix} \begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} \tag{7-109}$$

By inverting Eq. (7-109), the desired squared blade rotational speeds can be obtained as:

$$\begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{2bl} & \frac{-1}{4d} & \frac{1}{4d} \\ 0 & \frac{-1}{2bl} & \frac{-1}{4d} & \frac{1}{4d} \\ \frac{-1}{2bl} & 0 & \frac{1}{4d} & \frac{1}{4d} \\ \frac{1}{2bl} & 0 & \frac{1}{4d} & \frac{1}{4d} \end{bmatrix} \begin{bmatrix} M_x \\ M_y \\ M_z \\ F_z \end{bmatrix} \tag{7-110}$$

Finally, the required actuator inputs can be computed using the steady-state values of Eq. (3-39). For $\dot{\Omega}_m = 0$, the control input is then computed as

$$u_m = k_m \left(\Omega_m + \frac{d\tau}{\eta r^3 J_t} \Omega_m^2 \right) \tag{7-111}$$

where Ω_m^2 is obtained directly from Eq. (7-110), and subsequently $\Omega_m = \sqrt{\Omega_m^2}$.

7-5-2 Linear

Linear control allocation starts by defining an operating point Ω_0 for each rotor velocity. The sum of all forces produced by the motors is

$$F_z = 4b\Omega_m^2 \tag{7-112}$$

Therefore, to counteract the weight of the vehicle W the trim point for each rotor is given as

$$\Omega_0 = \sqrt{\frac{W}{4b}} \tag{7-113}$$

Note that the trim input moments are all zero. The total velocity is then given by

$$\Omega_m = \Omega_0 + \Omega_c \tag{7-114}$$

where Ω_c is the linear control velocity assumed as a small disturbance for design purposes. The product of two small disturbances is negligible, meaning that the approximation

$$\Omega_m^2 = (\Omega_0 + \Omega_c)^2 = \Omega_0^2 + 2\Omega_0\Omega_c + \Omega_c^2 \approx \Omega_0^2 + 2\Omega_0\Omega_c \tag{7-115}$$

can be used. Thus, Eq. (7-109) in the linear form becomes

$$\begin{bmatrix} M_x \\ M_y \\ M_z \\ F_z \end{bmatrix} = \begin{bmatrix} 0 & 0 & -2lb\Omega_0 & 2lb\Omega 0 \\ 2lb\Omega_0 & -2lb\Omega_0 & 0 & 0 \\ -2d\Omega_0 & -2d\Omega_0 & 2d\Omega_0 & 2d\Omega_0 \\ 2b\Omega_0 & 2b\Omega_0 & 2b\Omega_0 & 2b\Omega_0 \end{bmatrix} \begin{bmatrix} \Omega_{1c} \\ \Omega_{2c} \\ \Omega_{3c} \\ \Omega_{4c} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 4b\Omega_0 \end{bmatrix} \tag{7-116}$$

and can be written simply as

$$\underline{u}_m = T_{lca}\underline{\Omega}_c + \underline{u}_0 \tag{7-117}$$

which, when inverted, leads to

$$\underline{\Omega}_c = T_{lca}^{-1}\left(\underline{u}_m - \underline{u}_0\right) = T_{lca}^{-1}\underline{u}_c \tag{7-118}$$

The linear dynamics of the actuators can be obtained by linearizing Eq. (3-39) around the operating point Ω_0. The necessary input u_0 that yields Ω_0 is obtained using Eq. (7-111):

$$u_0 = k_m\left(\Omega_0 + \frac{d\tau}{\eta r^3 J_t}\Omega_0^2\right) \tag{7-119}$$

The actuator input becomes

$$u_m = u_0 + u_c \tag{7-120}$$

where u_c is the linear control input, assumed also as a small disturbance for design purposes. Substituting Eq. (7-120) and (7-114) into (3-39) yields

$$\left(\dot{\Omega}_0 + \dot{\Omega}_c\right) = -\frac{1}{\tau}\left(\Omega_0 + \Omega_c\right) - \frac{d}{\eta r^3 J_t}\left(\Omega_0 + \Omega_c\right)^2 + \frac{1}{k_m\tau}\left(u_0 + u_c\right) \tag{7-121}$$

Note that since Ω_0 is a constant, it holds that $\dot{\Omega}_0 = 0$. Expanding Eq. (7-121) results then in the first-order linear actuator dynamics described by

$$\dot{\Omega}_c = \left(-\frac{1}{\tau} - \frac{2d\Omega_0}{\eta r^3 J_t}\right)\Omega_c + \frac{1}{k_m\tau}u_c \tag{7-122}$$

with the linear time constant

$$\tau_{act,linear} = \left(-\frac{1}{\tau} - \frac{2d\Omega_0}{\eta r^3 J_t}\right)^{-1} \tag{7-123}$$

from which the following steady-state relation is obtained

$$u_c = k_m\left(1 + \frac{2\Omega_0 d\tau}{\eta r^3 J_t}\right)\Omega_c \tag{7-124}$$

where Ω_c is obtained from Eq. (7-118).

By inserting Eq. 7-124 into Eq. 7-120, the final desired command inputs can be determined as a function of the desired command forces and moments.

7-6 Preliminary controller simulations

In this section, the designs of the controllers outlined previously in this chapter are tested. Three types of controllers were presented/derived (linear cascaded, backstepping and incremental backstepping), and a simulation will now be shown for one scenario under nominal conditions. The rotorcraft has an initial condition and must regulate both longitudinal and lateral position to zero while climbing to a reference altitude. After, it should continue to hover before starting to descend with constant velocity. The results of this simulation are presented in Figure 7-2. For convenience, the reference signals are shown with dashed lines. It is possible to observe that for the nominal condition and under the assumption that the model is well-known, all controllers led to a similar result. Note that although the position loop was designed to render first-order closed-loop behavior, a small overshoot is observed. This is due to the fact that to obtain a faster response, the difference between loop time constants was reduced. The result is nonetheless a well-damped response with higher speed and therefore better overall performance.

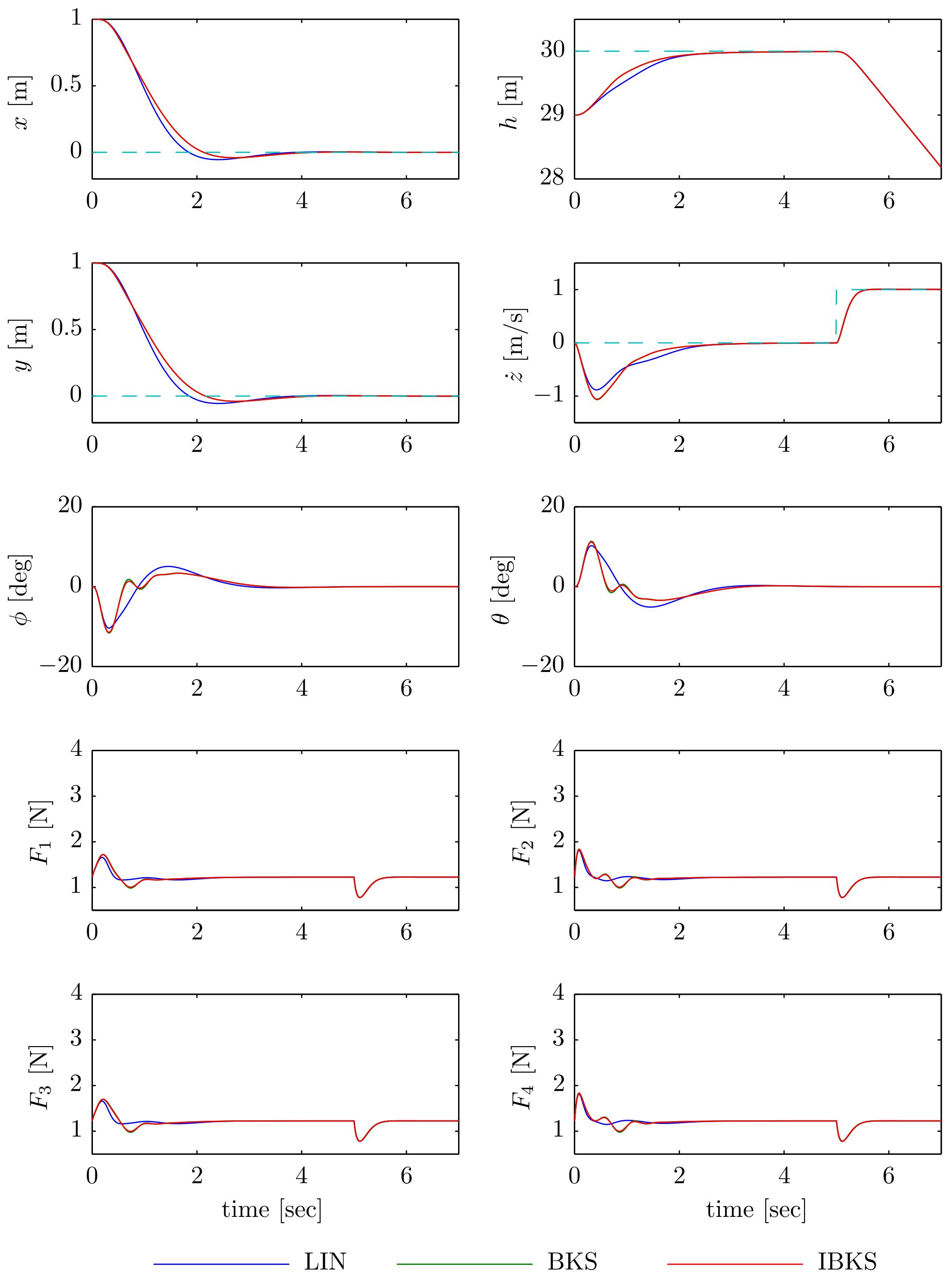


Figure 7-2: Preliminary controllers test; F_1, F_2, F_3 and F_4 denote the control forces for each motor

Chapter 8

Monte-Carlo simulations

In order to evaluate the performance of the control laws and state estimation algorithms developed in this thesis, a simulation tool is designed. The problem being studied is the automatic landing of a quadrotor UAV on a floating platform. A model of the environment has been given in Chapter 3, including a description of the floating platform motion and the wind profiles above the sea surface. A range of possible external variables are then combined in a Monte-Carlo scheme for conditions up to the upper bound of level 4 in the Beaufort scale. Simulations are performed not only for the case in which the rotorcraft model is known, but also for uncertain cases. Finally, a comparative analysis of the different controllers developed in this thesis is presented.

8-1 Monte-Carlo scheme

The following parameters are varied with a uniform distribution (see Appendix F) in the Monte-Carlo scheme:

- Initial position
  - Longitudinal deviation from platform's center
    (min: -2.5 m, max: 2.5 m)
  - Lateral deviation from platform's center
    (min: -2.5 m, max: 2.5 m)
  - Initial altitude above MSL
    (min: 30 m, max: 31 m)
- Platform motion (sum of three sinusoids)
  - Wave height components
    (min: 0 m, max: 1 m)

  - Wave period components
    (min: 5 sec, max: 15 sec)
  - Wave initial phase components
    (min: 0 deg, max: 360 deg)

- Wind conditions
  - Wind speed at reference altitude
    (min: 0 m/s, max: 8 m/s)
  - Wind direction
    (min: 0 deg, max: 360 deg)
- Temperature at MSL
  (min: 0 Celsius, max: 30 Celsius)

8-2 Performance assessment criteria

The following parameters are used as assessment criteria:

- PHITD (in degrees): defined as the roll angle at touchdown;
- TTATD (in degrees): defined as the pitch angle at touchdown;
- XTD (in meters): defined as the longitudinal touchdown position in the NED reference frame centered at the buoy's midpoint;
- YTD (in meters): defined as the lateral touchdown position in the NED reference frame centered at the buoy's midpoint;
- ZDTD (in meters per second): defined as the relative touchdown velocity (sink rate) between the quadrotor and the floating platform expressed in the NED reference frame fixed with the buoy.

The attitude angles should be as close to zero as possible at the touchdown moment. In fact, a crash landing can be expected if the vehicle has a non-zero attitude angle larger than a certain threshold when it touches the platform. Therefore, the statistical distribution obtained for PHITD and TTATD should be similar to a zero-mean normal distribution with low standard deviation. A risk analysis should prove that the probability of landing with more than a certain designed threshold angle (in absolute value) would be residual. Note that the touchdown yaw angle is irrelevant.

For touchdown position, it is important to always land very close to the center of the buoy. XTD and YTD should then present a statistical distribution that can be approximated by a zero-mean normal distribution with low standard deviation. As for the angles, it should be possible to prove that the probability of landing outside the designated landing area is residual.

Finally, the relative touchdown velocity ZDTD should not cross a certain design maximum value. Note that if touchdown sink rate is too high, then a crash landing could occur resulting in structural damage or other unwanted consequences. An approximately non-zero-mean normal distribution should be obtained with low touchdown sink rate dispersion. Recall that the design value for touchdown sink rate was 0.5 m/s.

Three controllers have been developed in this thesis based on linear control theory (LIN), backstepping (BKS) and incremental backstepping (IBKS). All three have been tested for the Monte-Carlo simulations.

Two main sets of simulations were performed, corresponding to simulations

- with perfect state knowledge, and
- with state estimator in the loop.

What follows are the results obtained.

8-3 Simulations with perfect state knowledge

Preliminary results of the controller performances have already been shown in Chapter 7. However, such results were obtained for a simple maneuver under nominal conditions. A 200-sample Monte-Carlo simulation was performed to analyze the response of the controllers for a set of combinations of external conditions for the complete mission. The results are shown in Table 8-1 and Figure 8-1. Note that no large disturbances are introduced and it is assumed that the model is completely known. It is observed that the autoland mission is always successfully executed. Both roll and pitch angles at TD always have mean values that are very close to zero and present very low dispersion (low standard deviations). The same holds true for the longitudinal and lateral positions at TD. In terms of relative sink rate, mean values of approximately 0.5 m/s (the design value) are always obtained and a very low sink rate TD dispersion is also observed.

Perfect state knowledge: no uncertainties

Table 8-1: Monte-Carlo simulation results for 200 samples assuming perfect state knowledge and no uncertainties

	PHITD	TTATD	XTD	YTD	ZDTD
	mean $\overline{x}$				
LIN	-0.0324	-0.0219	-0.0012	0.0021	0.5070
BKS	-0.0296	-0.0201	-0.0012	0.0020	0.5063
IBKS	-0.0347	-0.0213	-0.0013	0.0025	0.5105
	std σ				
LIN	0.7822	0.7087	0.0483	0.0527	0.0841
BKS	0.7847	0.7090	0.0457	0.0502	0.0839
IBKS	0.8336	0.7529	0.0498	0.0547	0.0851

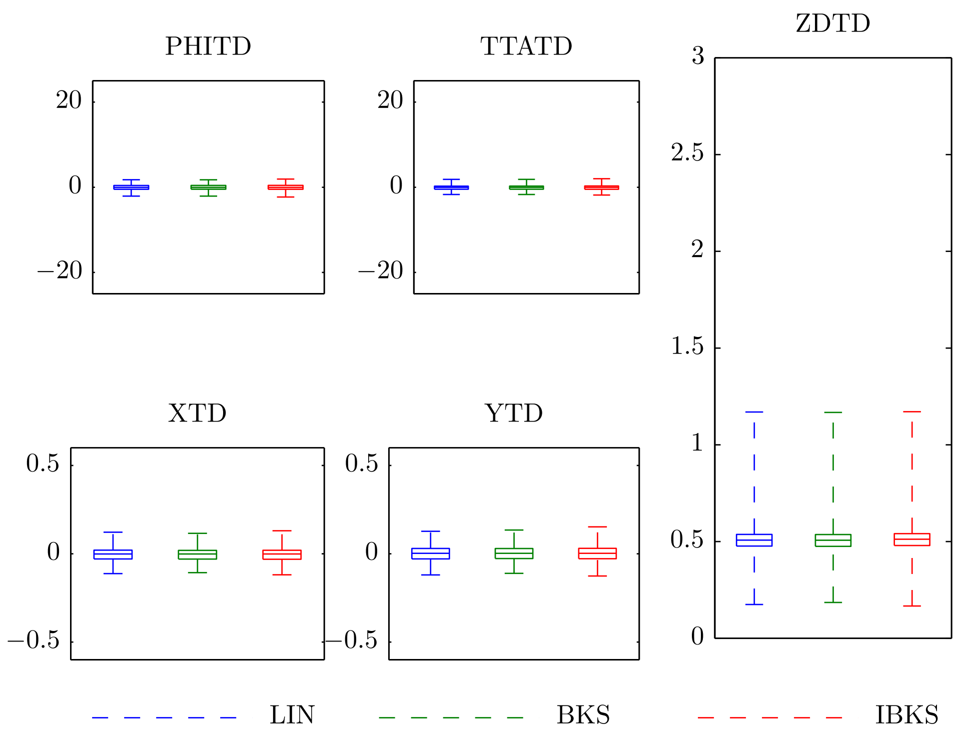


Figure 8-1: Boxplots for 200-sample Monte-Carlo simulation assuming perfect state knowledge and no uncertainties

Figures 8-2 and 8-3 show the complete mission and final landing phase, respectively, for one case scenario.

No difference between the controllers is observed for the perfect state knowledge case with no uncertainties. Robustness tests are then performed to assess the performance in the case of model mismatches.

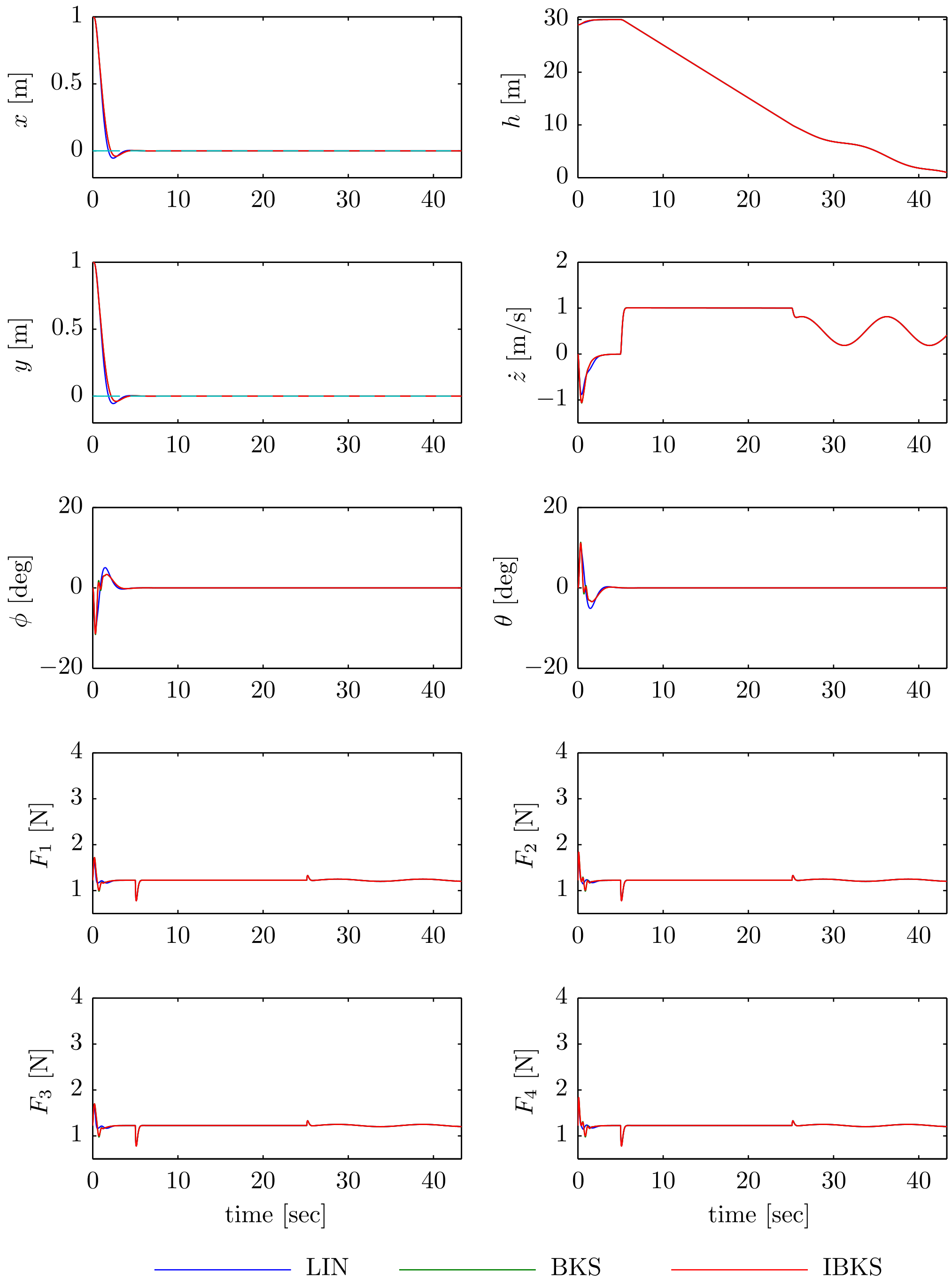


Figure 8-2: Simulation of the complete mission; F_1, F_2, F_3 and F_4 denote the control forces for each motor

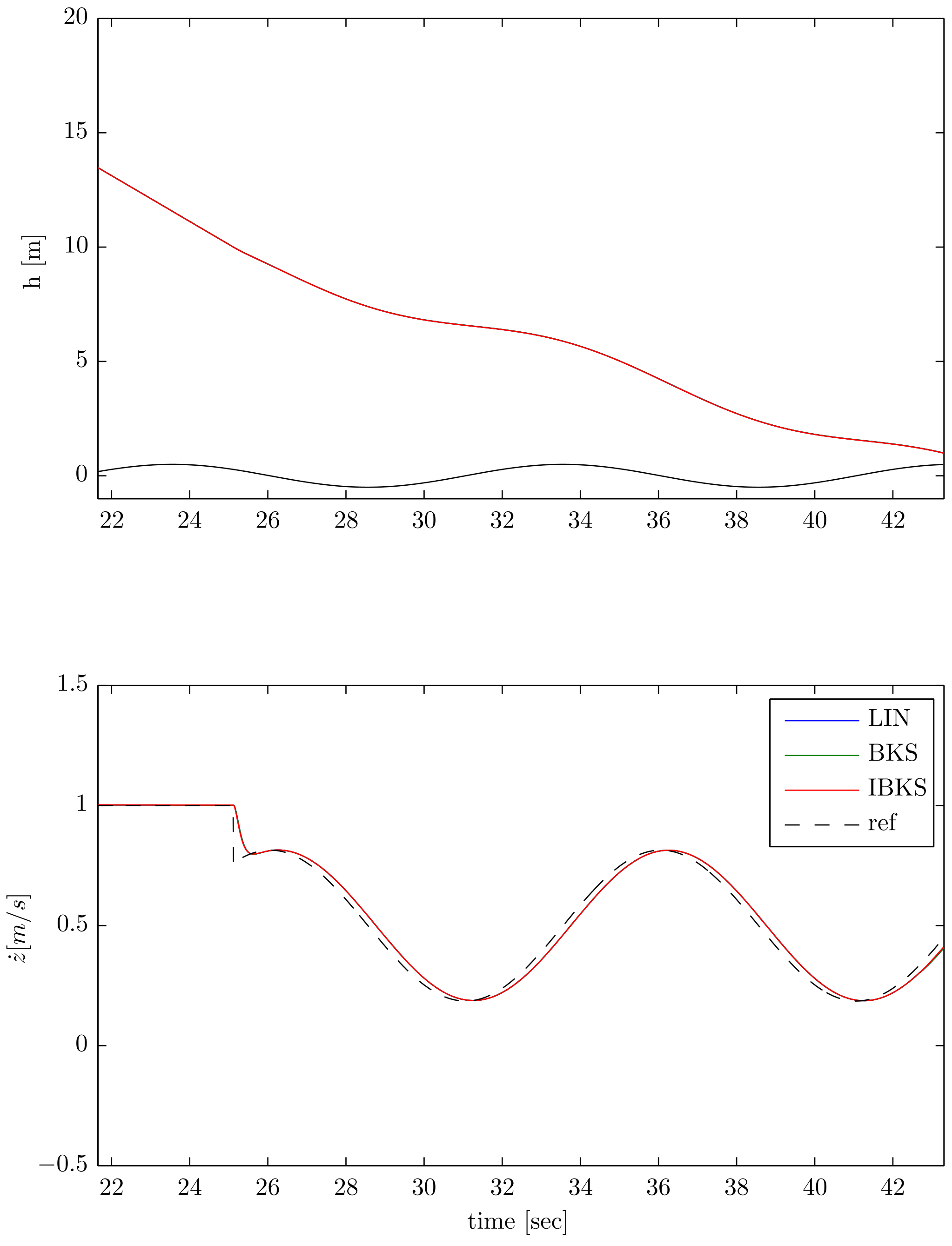


Figure 8-3: Final phase of the landing mission before touchdown

8-3-1 Robustness to uncertainties

Perfect state knowledge: mass uncertainties

Table 8-2: Monte-Carlo simulation results for 200 samples assuming perfect state knowledge; 25% uncertainty in mass ($\epsilon = 1.25$)

	PHITD	TTATD	XTD	YTD	ZDTD
			mean $\overline{x}$		
LIN	0.0315	0.0626	0.0042	-0.0021	1.0126
BKS	0.0323	0.0628	0.0042	-0.0021	1.0102
IBKS	0.0201	0.0310	0.0023	-0.0012	0.5153
			std σ		
LIN	1.3992	1.2279	0.0834	0.0948	0.1323
BKS	1.3832	1.2199	0.0814	0.0920	0.1322
IBKS	0.9162	0.8159	0.0536	0.0603	0.0916

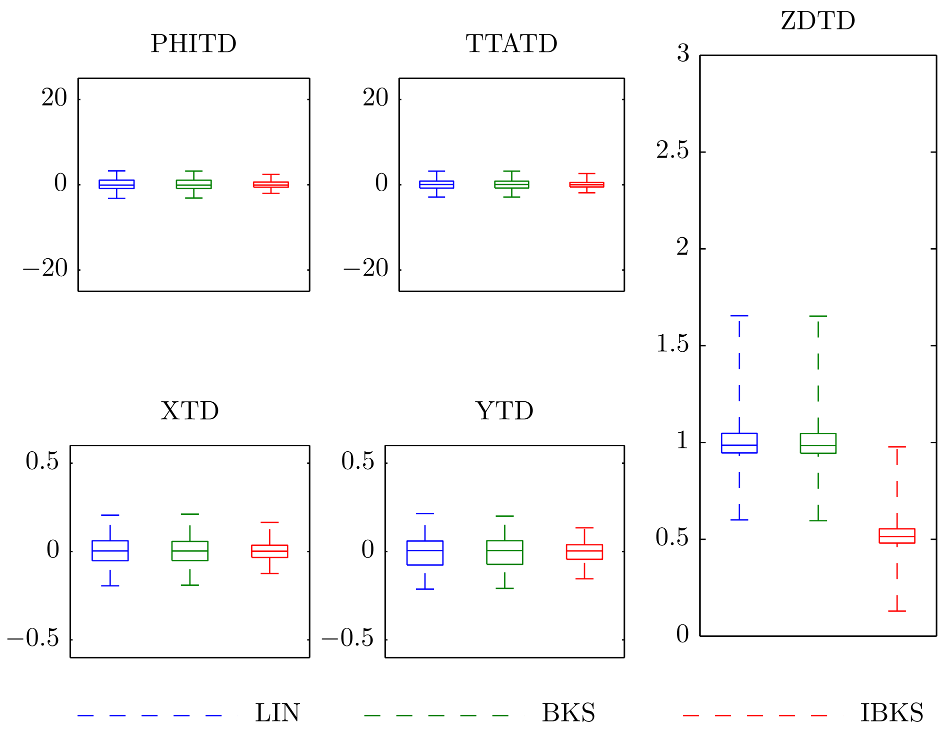


Figure 8-4: Boxplots for 200 sample Monte-Carlo simulation assuming perfect state knowledge; uncertainty in mass ($\epsilon = 1.25$)

The desired mean sink rate value is obtained for incremental backstepping, while for linear control and backstepping a steady-state error occurs due to incorrect mass assumption. After analyzing Figure 8-5, the effects of this uncertainty become clear. Note that the initial seconds of the mission are sufficient to understand robustness issues.

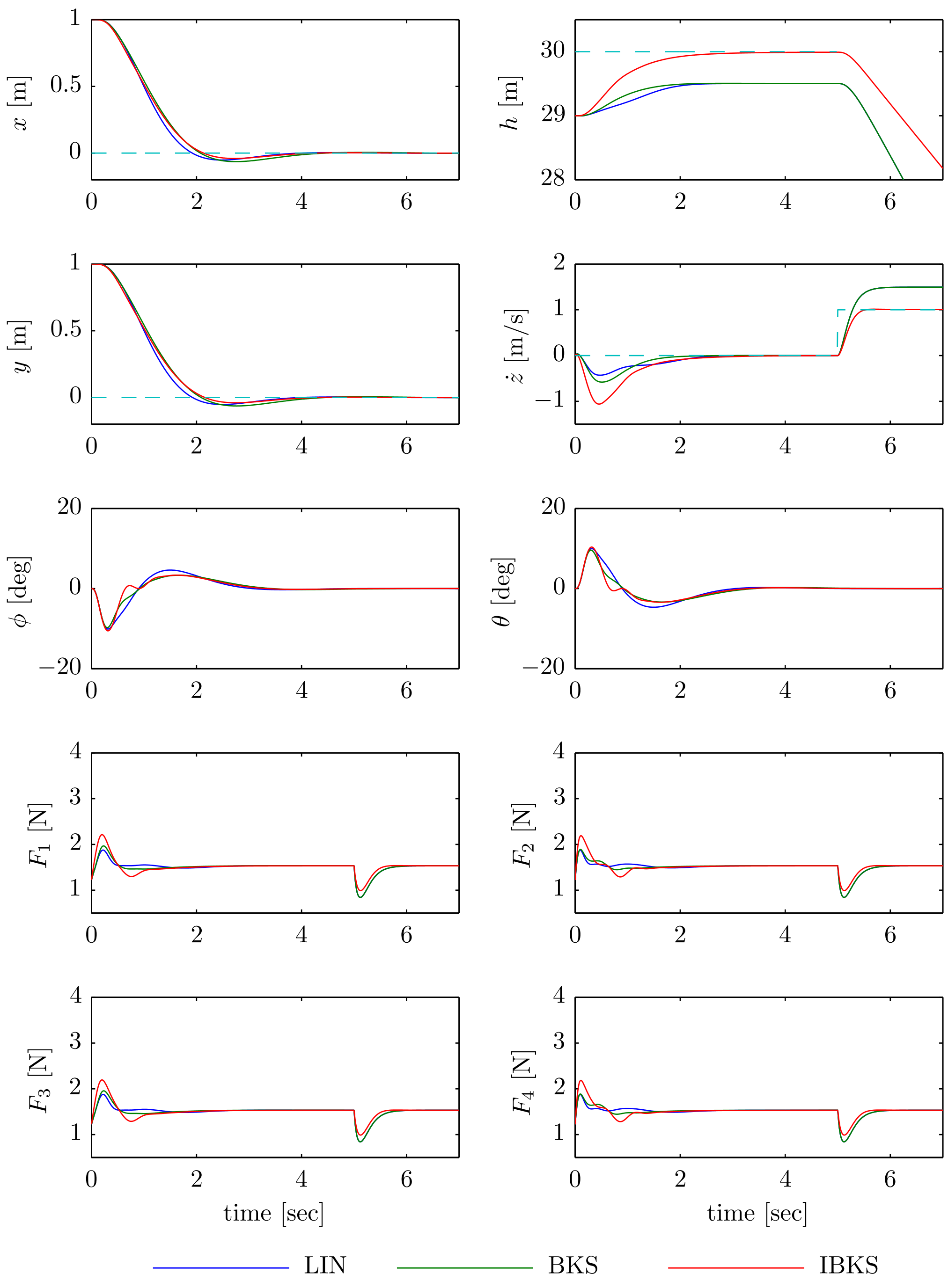


Figure 8-5: Controller performances for uncertainty in mass ($\epsilon = 1.25$); F_1, F_2, F_3 and F_4 denote the control forces for each motor

Perfect state knowledge: inertia uncertainties

Table 8-3: Monte-Carlo simulation results for 200 samples assuming perfect state knowledge; 30% uncertainty in inertia ($\epsilon = 1.3$)

	PHITD	TTATD	XTD	YTD	ZDTD
	mean $\overline{x}$				
LIN	0.0723	0.0416	0.0027	-0.0048	0.5072
BKS	0.0728	0.0417	0.0027	-0.0047	0.5063
IBKS	0.0766	0.0454	0.0031	-0.0050	0.5106
	std σ				
LIN	0.8610	0.7601	0.0514	0.0586	0.0685
BKS	0.8596	0.7588	0.0490	0.0550	0.0690
IBKS	0.9172	0.8040	0.0530	0.0602	0.0686

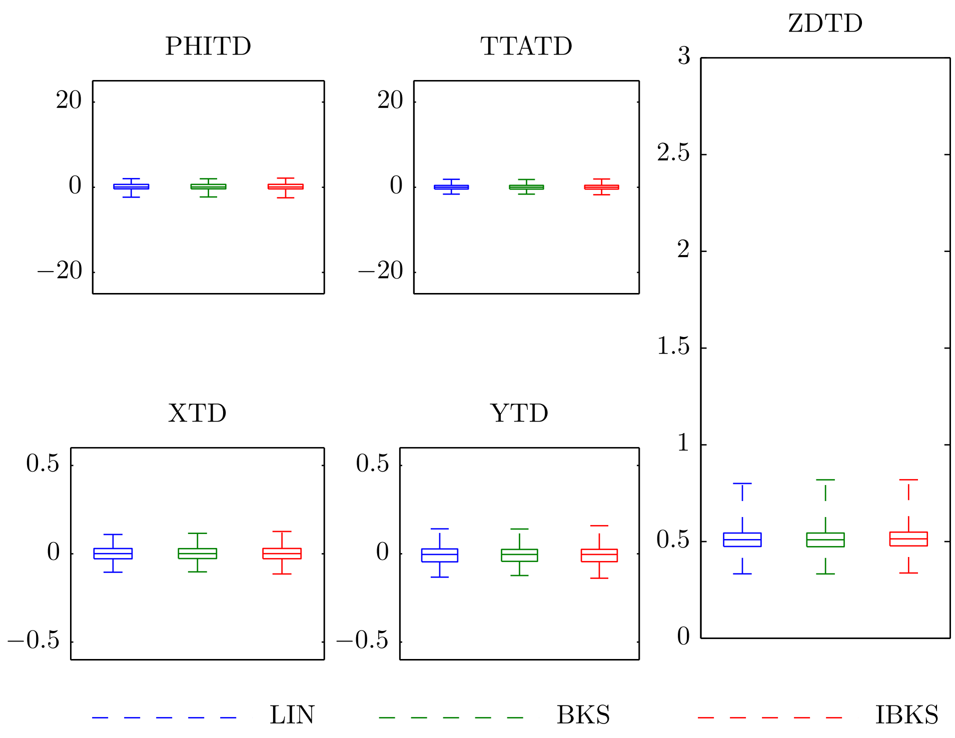


Figure 8-6: Boxplot for 200-sample Monte Carlo simulation assuming perfect state knowledge; uncertainty in inertia ($\epsilon = 1.3$)

The differences between the controllers are negligible for this uncertainty. For that reason, a larger mismatch is introduced by selecting $\epsilon = 4$. The results are presented in Figure 8-7.

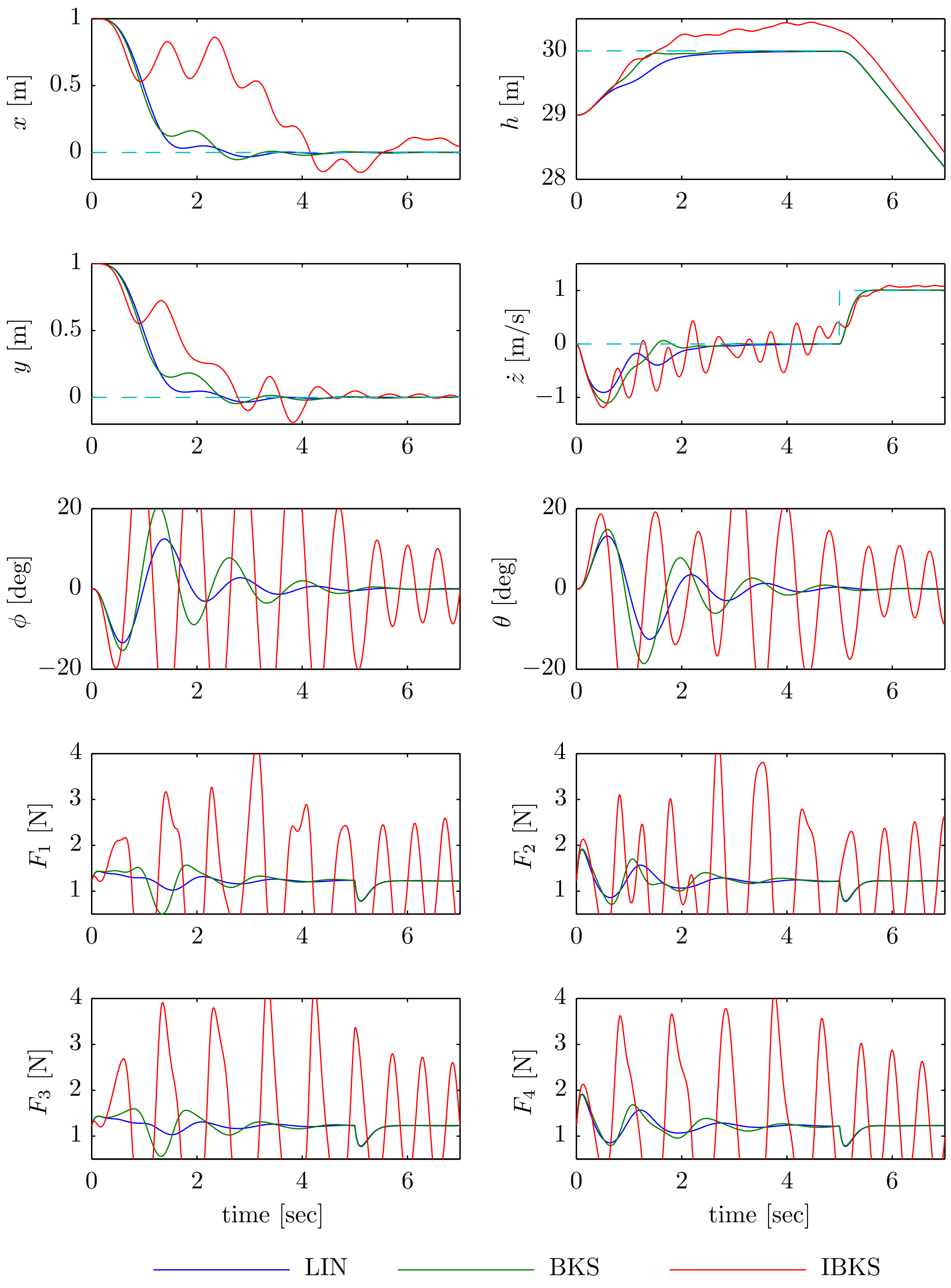


Figure 8-7: Controller performances for extreme case of uncertainty in inertia ($\epsilon = 4$); F_1, F_2, F_3 and F_4 denote the control forces for each motor

Perfect state knowledge: motor constant uncertainties

Table 8-4: Monte-Carlo simulation results for 200 samples assuming perfect state knowledge; 20% uncertainty in motor torque constant ($\epsilon = 1.2$)

	PHITD	TTATD	XTD	YTD	ZDTD
	mean $\overline{x}$				
LIN	0.0959	-0.0170	-0.0014	-0.0072	1.1285
BKS	0.0947	-0.0194	-0.0016	-0.0066	1.1257
IBKS	0.0677	-0.0179	-0.0011	-0.0044	0.5211
	std σ				
LIN	1.3978	1.4272	0.0970	0.0943	0.1147
BKS	1.3813	1.4146	0.0957	0.0932	0.1154
IBKS	0.8913	0.9171	0.0602	0.0584	0.0801

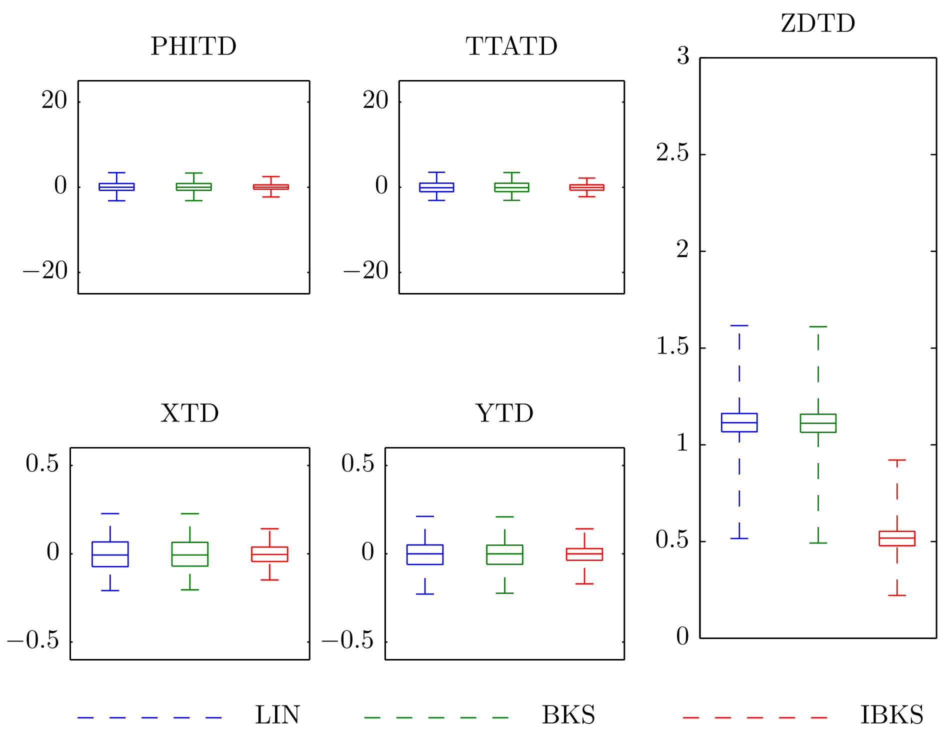


Figure 8-8: Boxplot for 200-sample Monte Carlo simulation assuming perfect state knowledge; uncertainty in motor torque constant ($\epsilon = 1.2$)

A steady-state error is again observed for linear control and backstepping regarding vertical velocity tracking. The same does not hold true for incremental backstepping. Figure 8-9 shows the time histories of states and control action for this uncertainty scenario.

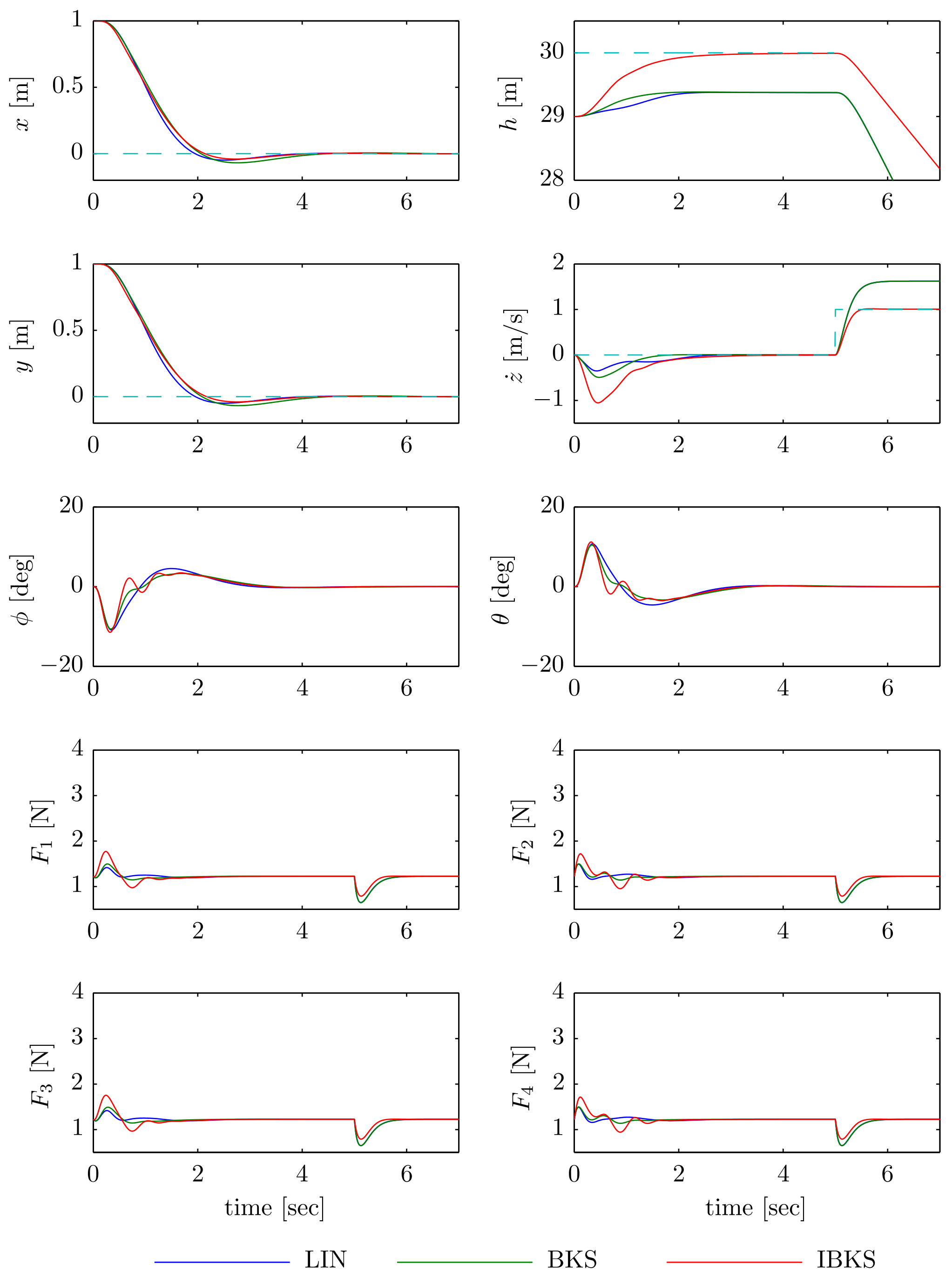


Figure 8-9: Controller performances for uncertainty in motor constant ($\epsilon = 1.2$); F_1, F_2, F_3 and F_4 denote the control forces for each motor

Perfect state knowledge: motor time constant uncertainties

Table 8-5: Monte-Carlo simulation results for 200 samples assuming perfect state knowledge; 30% uncertainty in actuator time constant ($\epsilon = 1.3$)

	PHITD	TTATD	XTD	YTD	ZDTD
	mean $\overline{x}$				
LIN	-0.1074	0.1344	0.0082	0.0074	0.8332
BKS	-0.1076	0.1335	0.0084	0.0070	0.8311
IBKS	-0.0954	0.0949	0.0062	0.0062	0.5021
	std σ				
LIN	1.2008	1.2704	0.0840	0.0804	0.1190
BKS	1.1869	1.2553	0.0811	0.0771	0.1189
IBKS	0.9015	0.9592	0.0630	0.0592	0.0924

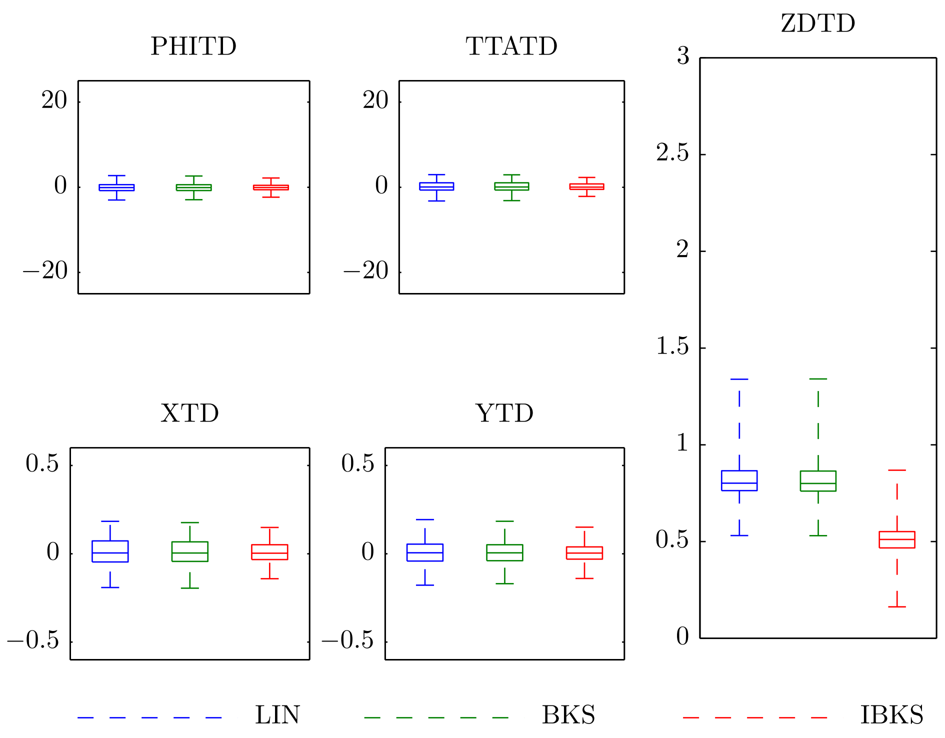


Figure 8-10: Boxplot for 200-sample Monte Carlo simulation assuming perfect state knowledge; uncertainty in motor time constant ($\epsilon = 1.3$)

To clearly show the negative effects of mismatches in modeled actuator delays, Figure 8-11 shows time histories of the states and control action for an uncertainty in the motor time constant for $\epsilon = 2$.

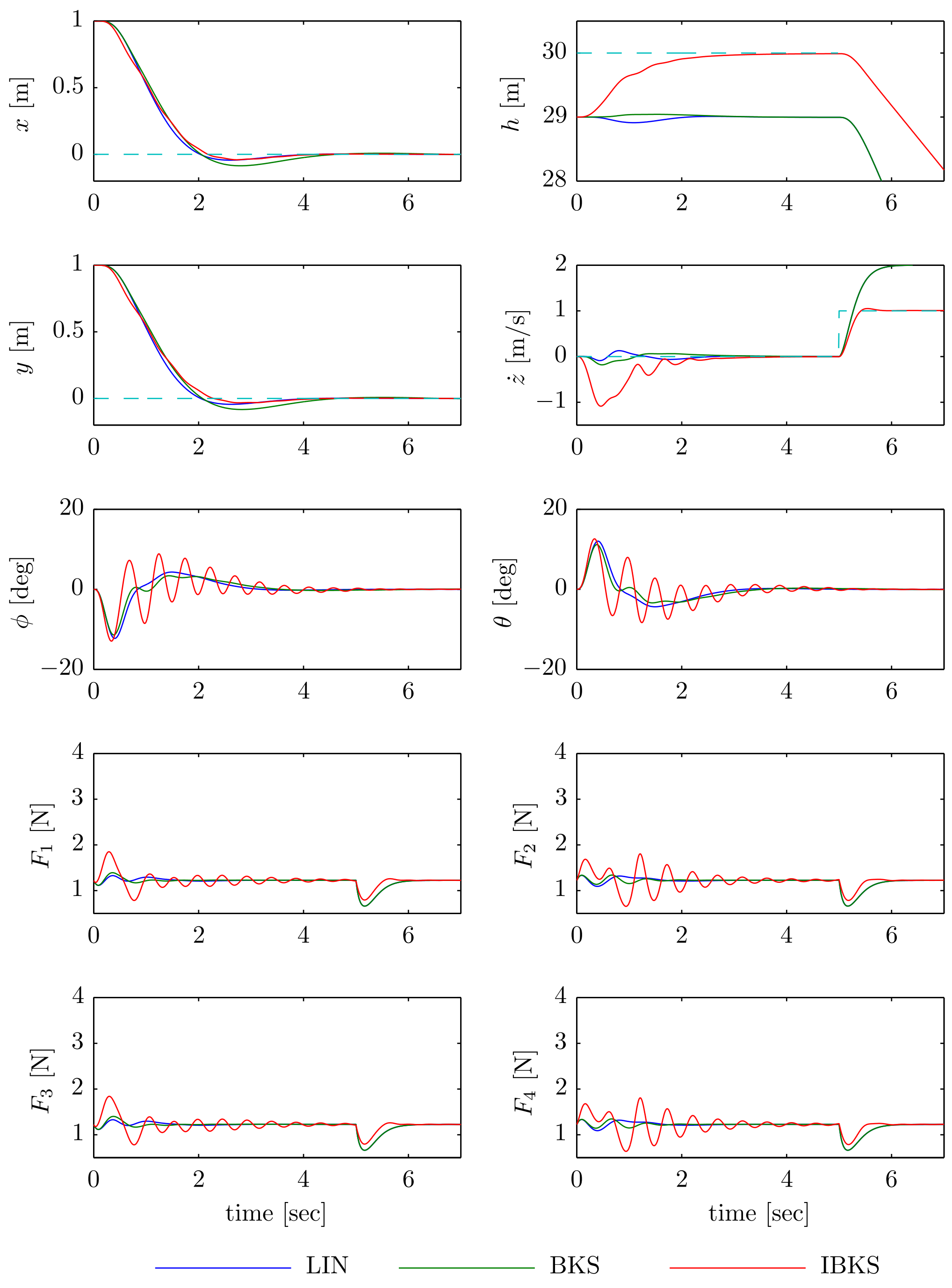


Figure 8-11: Controller performances for uncertainty in motor time constant ($\epsilon = 2$); F_1, F_2, F_3 and F_4 denote the control forces for each motor

Perfect state knowledge: rotor gyro effect uncertainties

Table 8-6: Monte-Carlo simulation results for 200 samples assuming perfect state knowledge; uncertainty in rotor gyro effect ($\epsilon = 100$)

	PHITD	TTATD	XTD	YTD	ZDTD
	mean $\overline{x}$				
LIN	0.0137	-0.0232	-0.0017	-0.0008	0.5134
BKS	0.0144	-0.0204	-0.0011	-0.0008	0.5123
IBKS	0.0185	-0.0242	-0.0015	-0.0012	0.5170
	std σ				
LIN	0.9474	0.8544	0.0575	0.0635	0.0847
BKS	0.9421	0.8526	0.0547	0.0599	0.0847
IBKS	1.0052	0.9078	0.0597	0.0657	0.0848

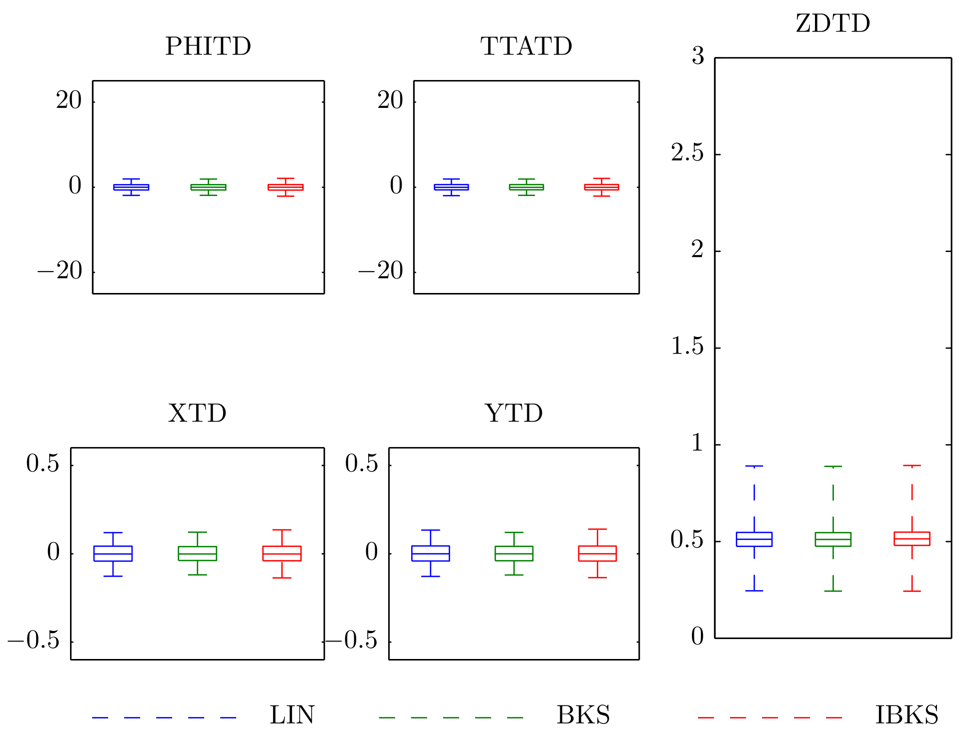


Figure 8-12: Boxplot for 200-sample Monte Carlo simulation assuming perfect state knowledge; uncertainty in rotor gyro effect ($\epsilon = 100$)

No difference between the control laws is observed. Therefore, an exaggerated uncertainty with $\epsilon = 6500$ is introduced and the results are presented in Figure 8-13. The effects of this last uncertainty, seen as a disturbance, will be explained later.

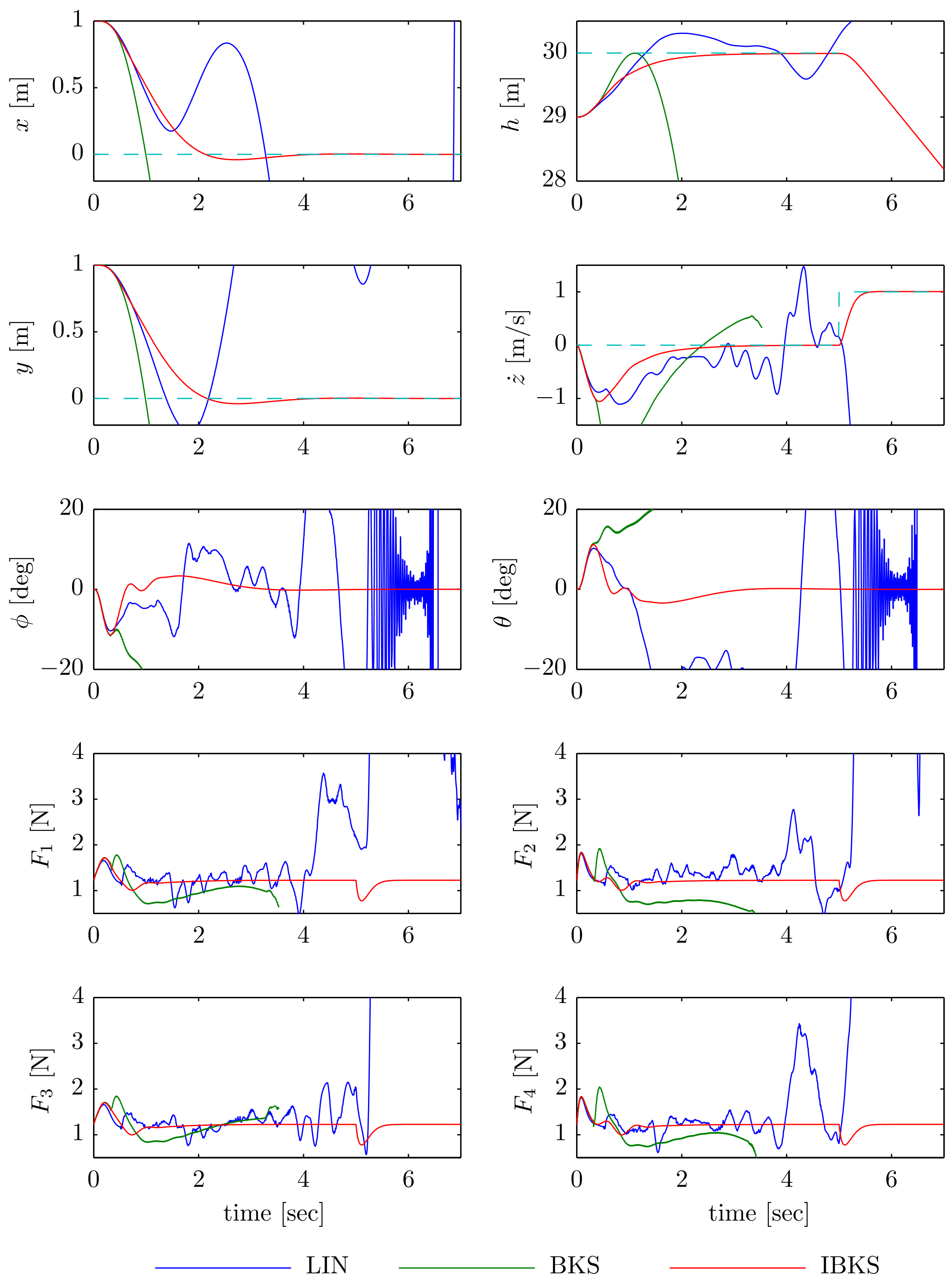


Figure 8-13: Controller performances for extreme case of uncertainty in rotor gyro effect ($\epsilon = 6500$); F_1, F_2, F_3 and F_4 denote the control forces for each motor

8-4 Simulations with state estimator in the loop

With state estimator in the loop: no uncertainties

Table 8-7: Monte-Carlo simulation results for 1000 samples using state estimator in the loop; no uncertainties

	PHITD	TTATD	XTD	YTD	ZDTD
	mean $\overline{x}$				
BKS	-0.1409	-0.0950	0.0409	-0.0443	0.7705
IBKS	-0.1660	0.0085	0.0064	-0.0849	0.7009
	std σ				
BKS	2.0923	1.8025	0.0550	0.0847	0.2563
IBKS	1.9672	1.8131	0.0579	0.0914	0.2493

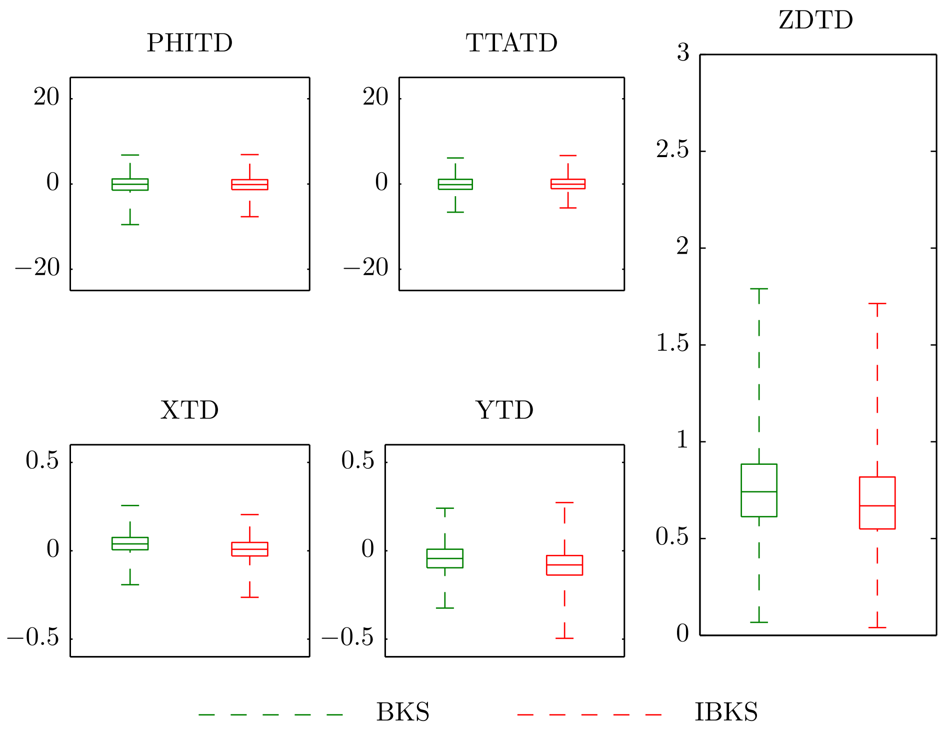


Figure 8-14: Boxplot for 1000-sample Monte-Carlo simulation using state estimator in the loop; no uncertainties

The complete mission is shown in Figure 8-15 and the landing phase is presented in Figure 8-16. The robustness tests carried out were identical to the perfect state knowledge case, with the exception of the rotor gyro effect.

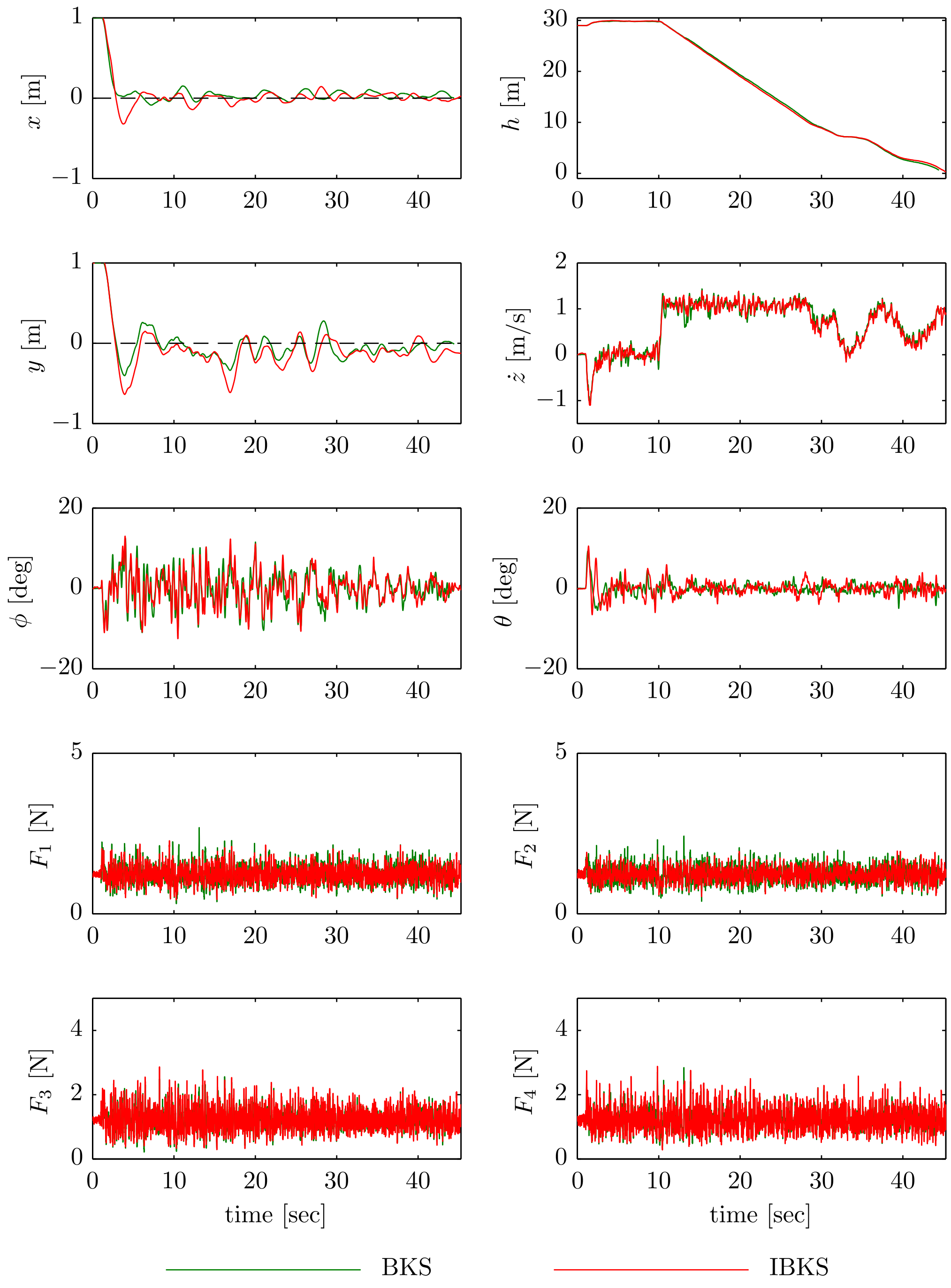


Figure 8-15: Simulation of the complete mission; F_1, F_2, F_3 and F_4 denote the control forces for each motor

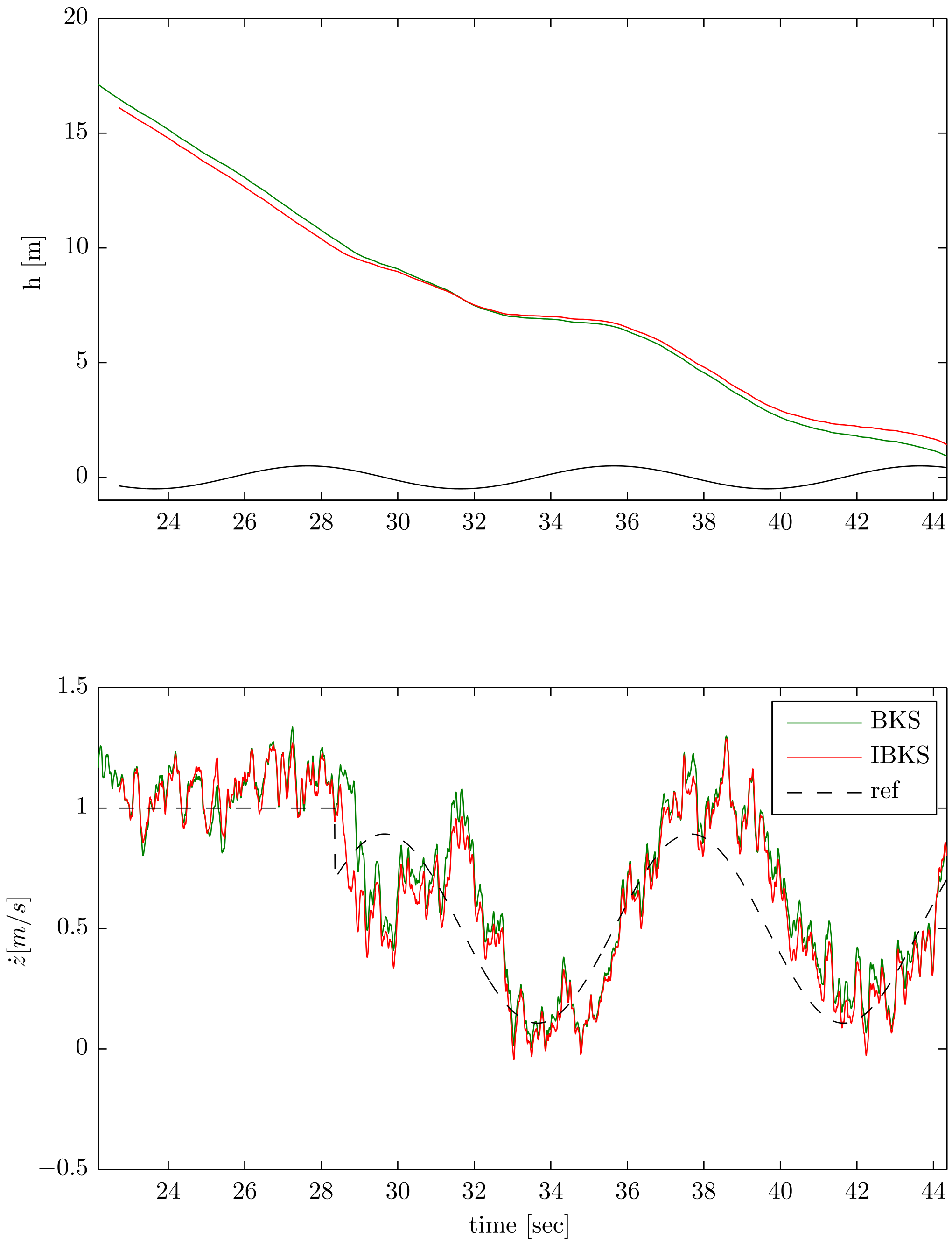


Figure 8-16: Final phase of the landing mission before touchdown

8-4-1 Robustness to uncertainties

With state estimator in the loop: mass uncertainties

Table 8-8: Monte-Carlo simulation results for 200 samples using state estimator in the loop; uncertainty in mass ($\epsilon = 1.25$)

	PHITD	TTATD	XTD	YTD	ZDTD
			mean $\overline{x}$		
BKS	-0.1375	-0.0802	0.0285	-0.0257	1.2299
IBKS	-0.2608	-0.0397	0.0062	-0.0571	0.7607
			std σ		
BKS	1.8089	1.6634	0.0694	0.0946	0.2867
IBKS	2.0074	1.8449	0.0548	0.0931	0.2492

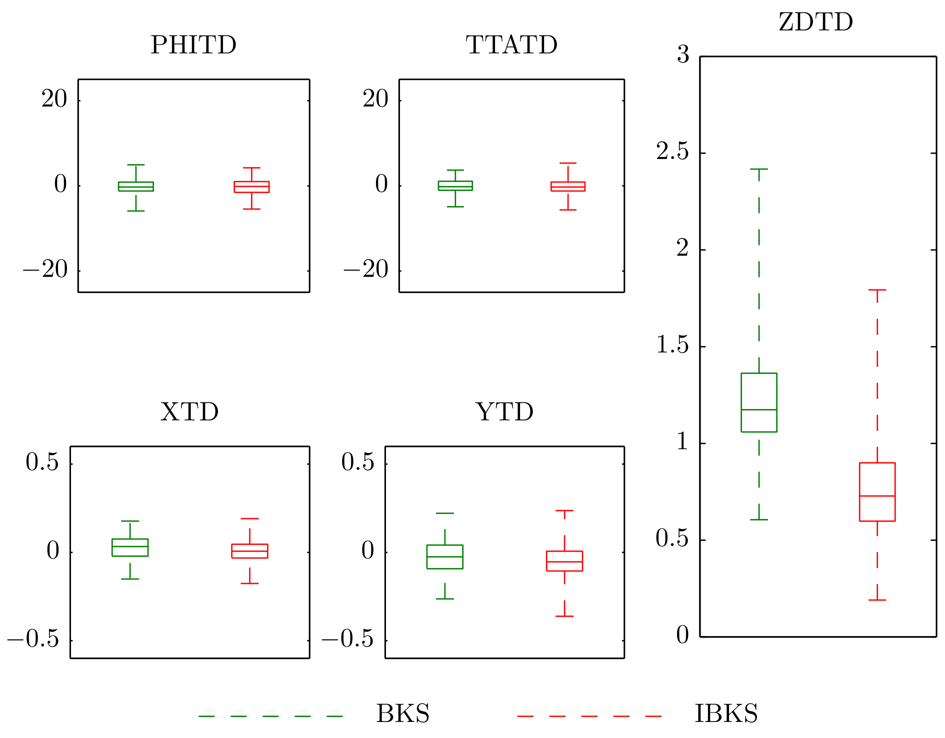


Figure 8-17: Boxplots for 200-sample Monte-Carlo simulation using state estimator in the loop; uncertainty in mass ($\epsilon = 1.25$)

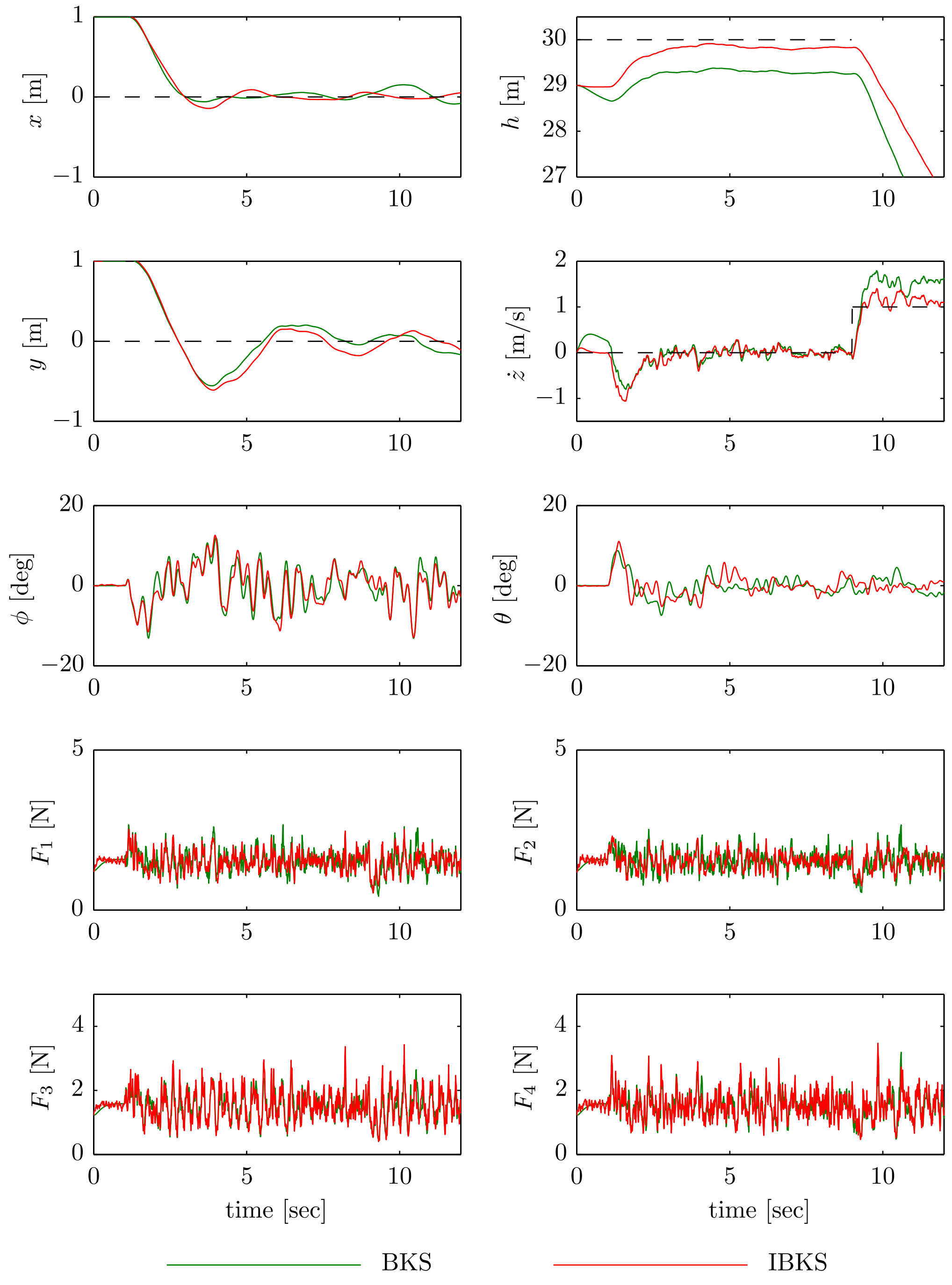


Figure 8-18: Controller performances for uncertainty in mass ($\epsilon = 1.25$); F_1, F_2, F_3 and F_4 denote the control forces for each motor

With state estimator in the loop: inertia uncertainties

Table 8-9: Monte-Carlo simulation results for 200 samples using state estimator in the; 30% uncertainty in inertia ($\epsilon = 1.3$)

	PHITD	TTATD	XTD	YTD	ZDTD
	mean $\overline{x}$				
BKS	-0.3128	0.0387	0.0455	-0.0475	0.7847
IBKS	-0.2308	0.0780	0.0056	-0.0652	0.7200
	std σ				
BKS	2.0144	1.5930	0.0521	0.0774	0.2525
IBKS	2.0607	2.1296	0.0547	0.0845	0.2255

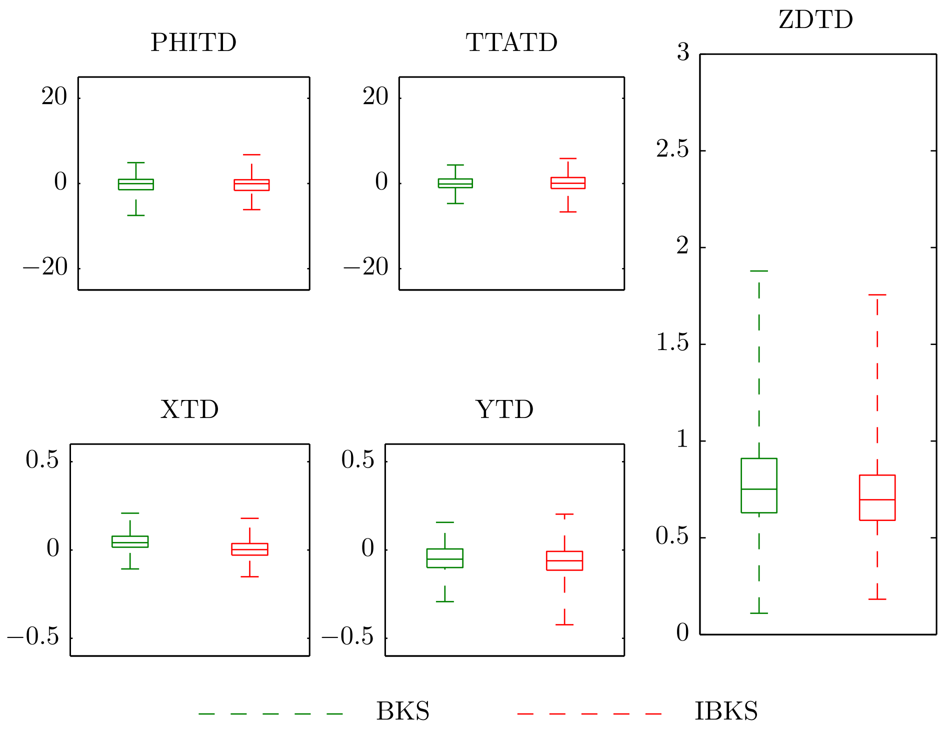


Figure 8-19: Boxplots for 200-sample Monte-Carlo simulation using state estimation; uncertainty in inertia ($\epsilon = 1.3$)

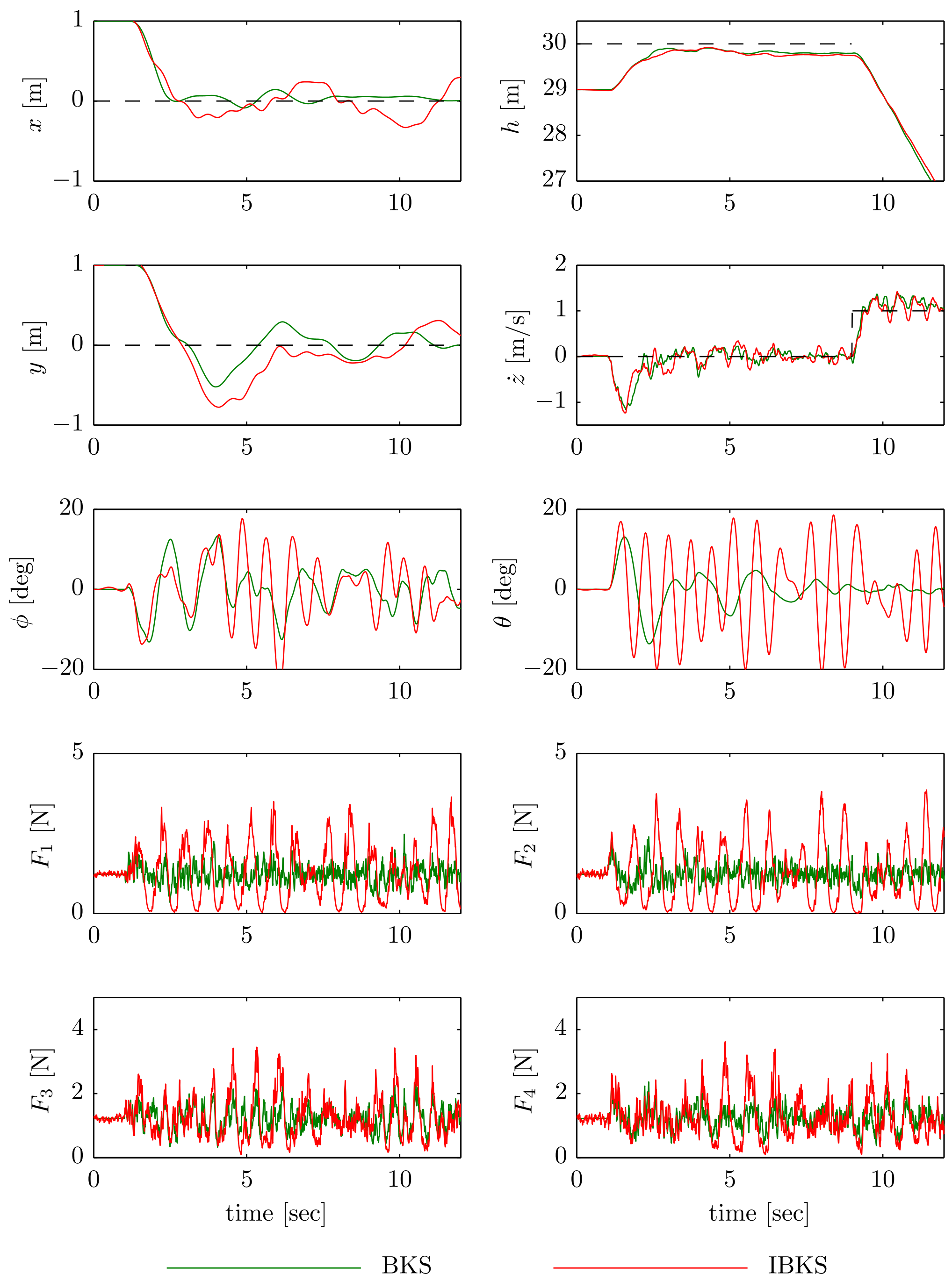


Figure 8-20: Controller performances for extreme case of uncertainty in inertia ($\epsilon = 4$); F_1, F_2, F_3 and F_4 denote the control forces for each motor

With state estimator in the loop: motor constant uncertainties

Table 8-10: Monte-Carlo simulation results for 200 samples using state estimator in the loop; uncertainty in motor torque constant ($\epsilon = 1.2$)

	PHITD	TTATD	XTD	YTD	ZDTD
			mean $\overline{x}$		
BKS	-0.1583	-0.0878	0.0278	-0.0317	1.3110
IBKS	0.0376	-0.3397	0.0074	-0.0593	0.7432
			std σ		
BKS	1.6255	1.5991	0.0677	0.0934	0.2642
IBKS	2.4437	2.2761	0.0519	0.1012	0.2459

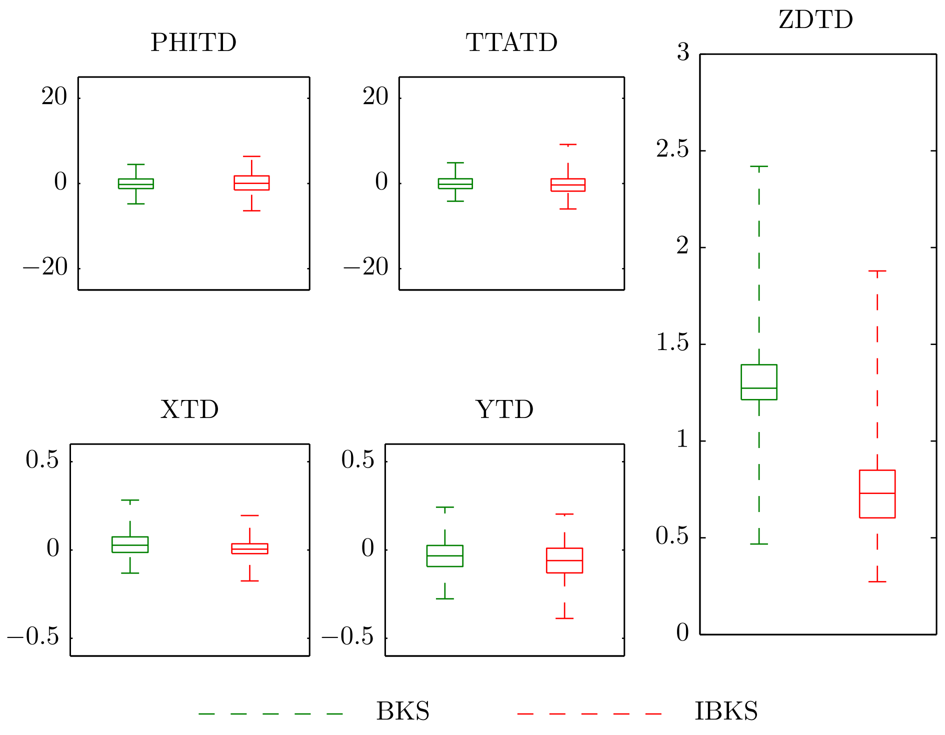


Figure 8-21: Boxplots for 200-sample Monte-Carlo simulation using state estimator in the loop; uncertainty in motor torque constant ($\epsilon = 1.2$)

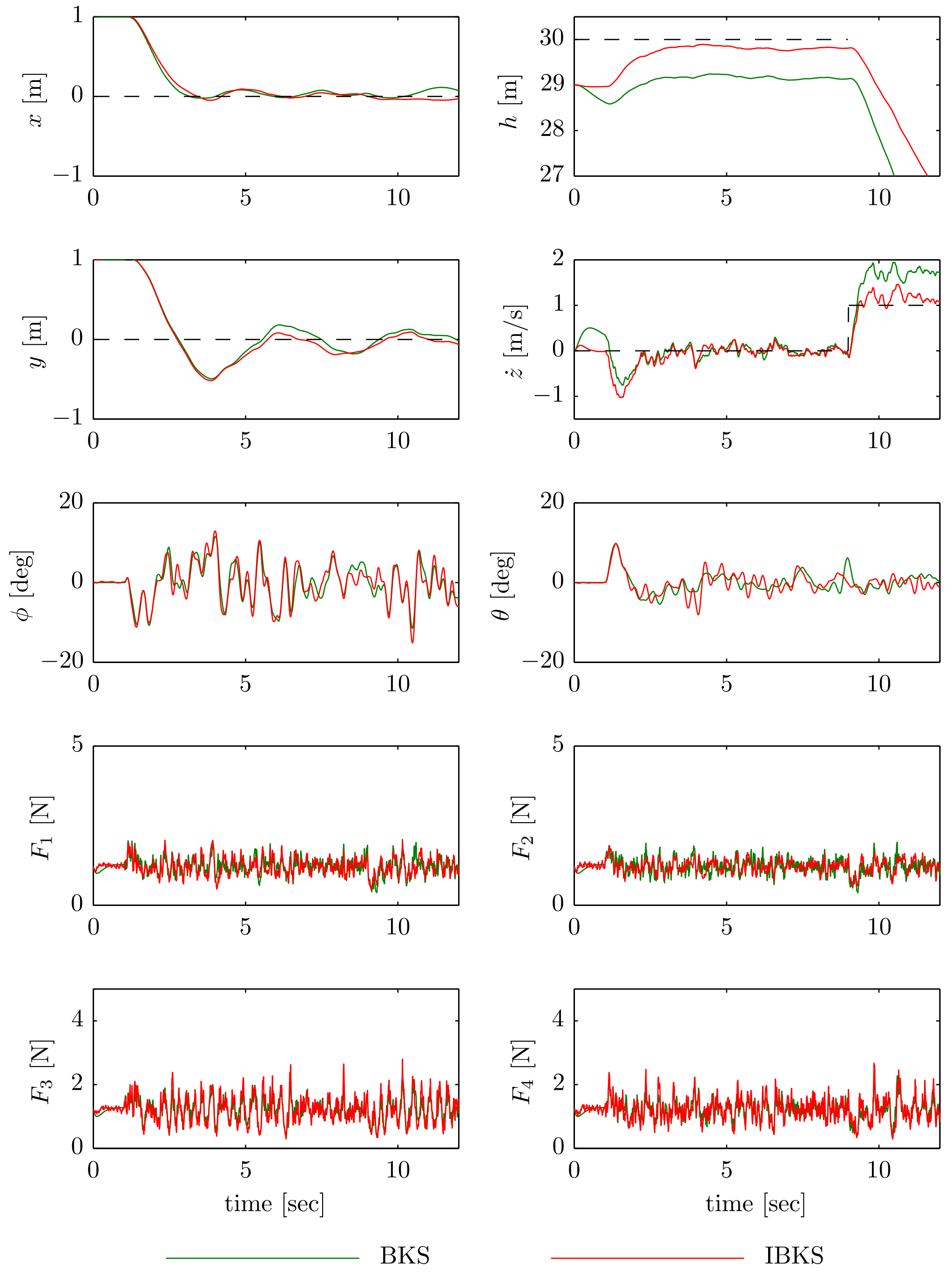


Figure 8-22: Controller performances for uncertainty in motor constant ($\epsilon = 1.2$); F_1, F_2, F_3 and F_4 denote the control forces for each motor

With state estimator in the loop: motor time constant uncertainties

Table 8-11: Monte-Carlo simulation results for 200 samples using state estimator in the loop; uncertainty in motor time constant ($\epsilon = 1.3$)

	PHITD	TTATD	XTD	YTD	ZDTD
	mean $\overline{x}$				
BKS	0.0731	-0.1861	0.0426	-0.0731	1.0459
IBKS	-0.1015	-0.1196	0.0011	-0.0702	0.7949
	std σ				
BKS	1.9261	1.7071	0.0566	0.0992	0.3102
IBKS	2.1829	2.0336	0.0577	0.0946	0.2893

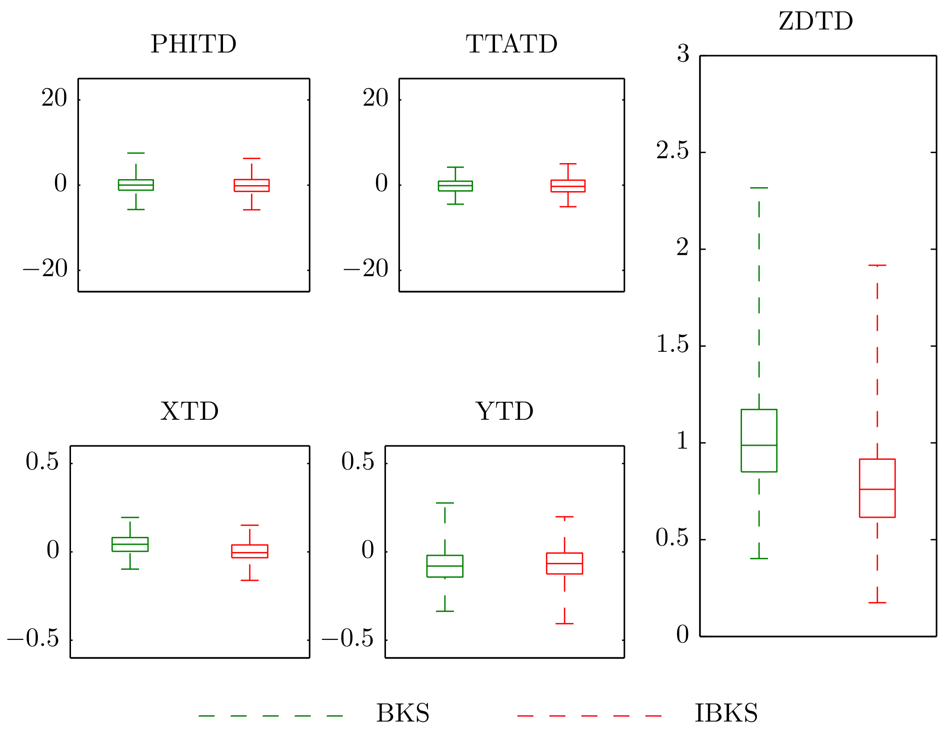


Figure 8-23: Boxplots for 200-sample Monte-Carlo simulation using state estimation; uncertainty in motor time constant ($\epsilon = 1.3$)

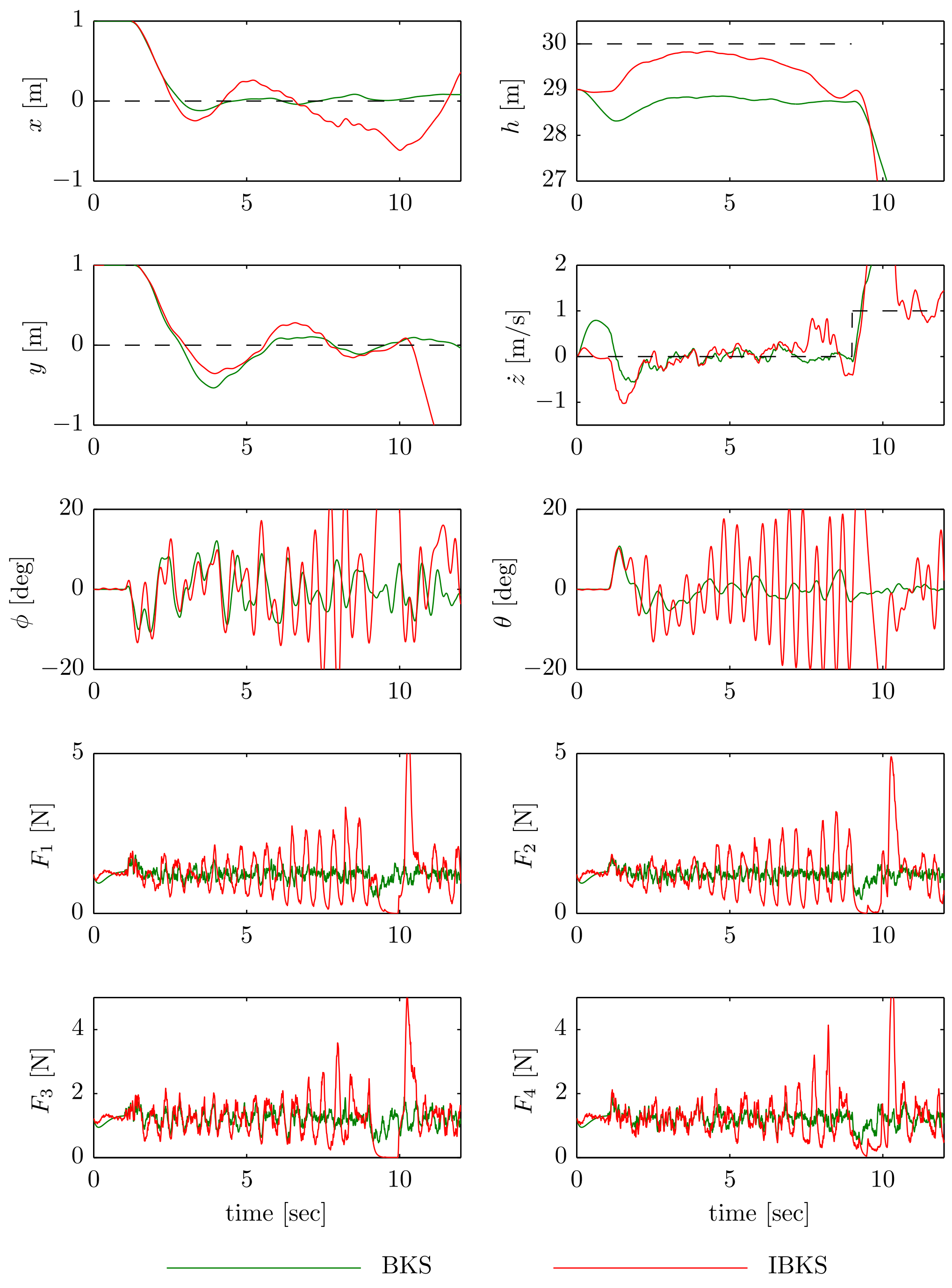


Figure 8-24: Controller performances for uncertainty in motor time constant ($\epsilon = 2$); F_1, F_2, F_3 and F_4 denote the control forces for each motor

With state estimator in the loop: rotor gyro effect uncertainties

Table 8-12: Monte-Carlo simulation results for 200 samples using state estimator in the loop; uncertainty in rotor gyro effect ($\epsilon = 100$)

	PHITD	TTATD	XTD	YTD	ZDTD
	mean $\overline{x}$				
BKS	-0.1266	0.0360	0.0407	-0.0538	0.7278
IBKS	0.0651	-0.1104	0.0313	-0.1068	0.7064
	std σ				
BKS	1.9969	1.7259	0.0538	0.0870	0.2451
IBKS	1.8667	1.6247	0.0554	0.0923	0.2374

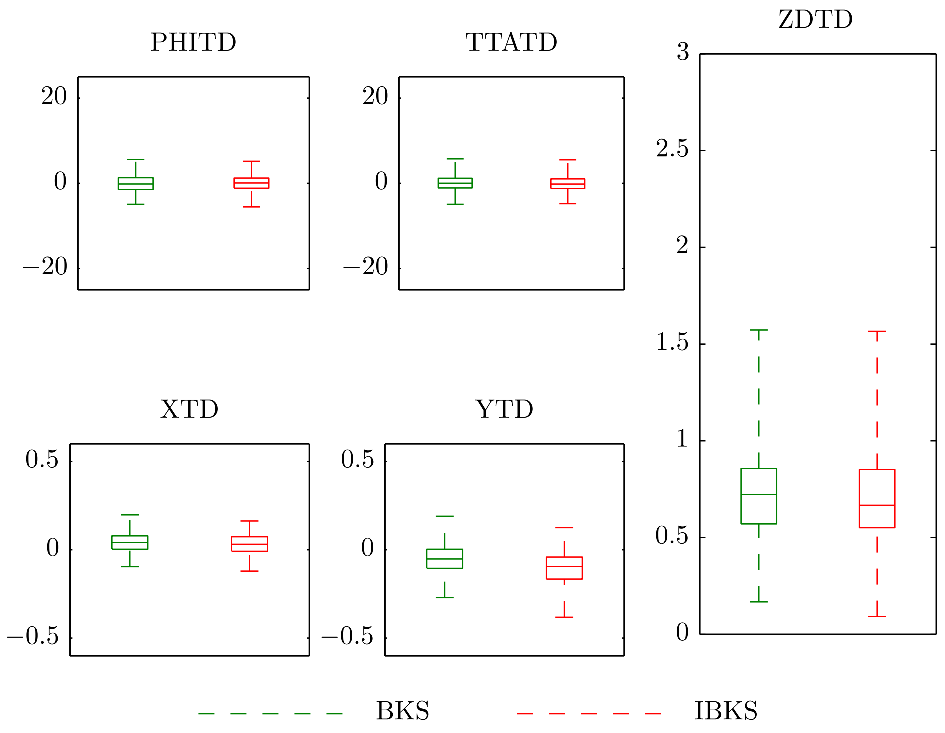


Figure 8-25: Boxplots for 200-sample Monte-Carlo simulation using state estimator in the loop; uncertainty in rotor gyro effect ($\epsilon = 100$)

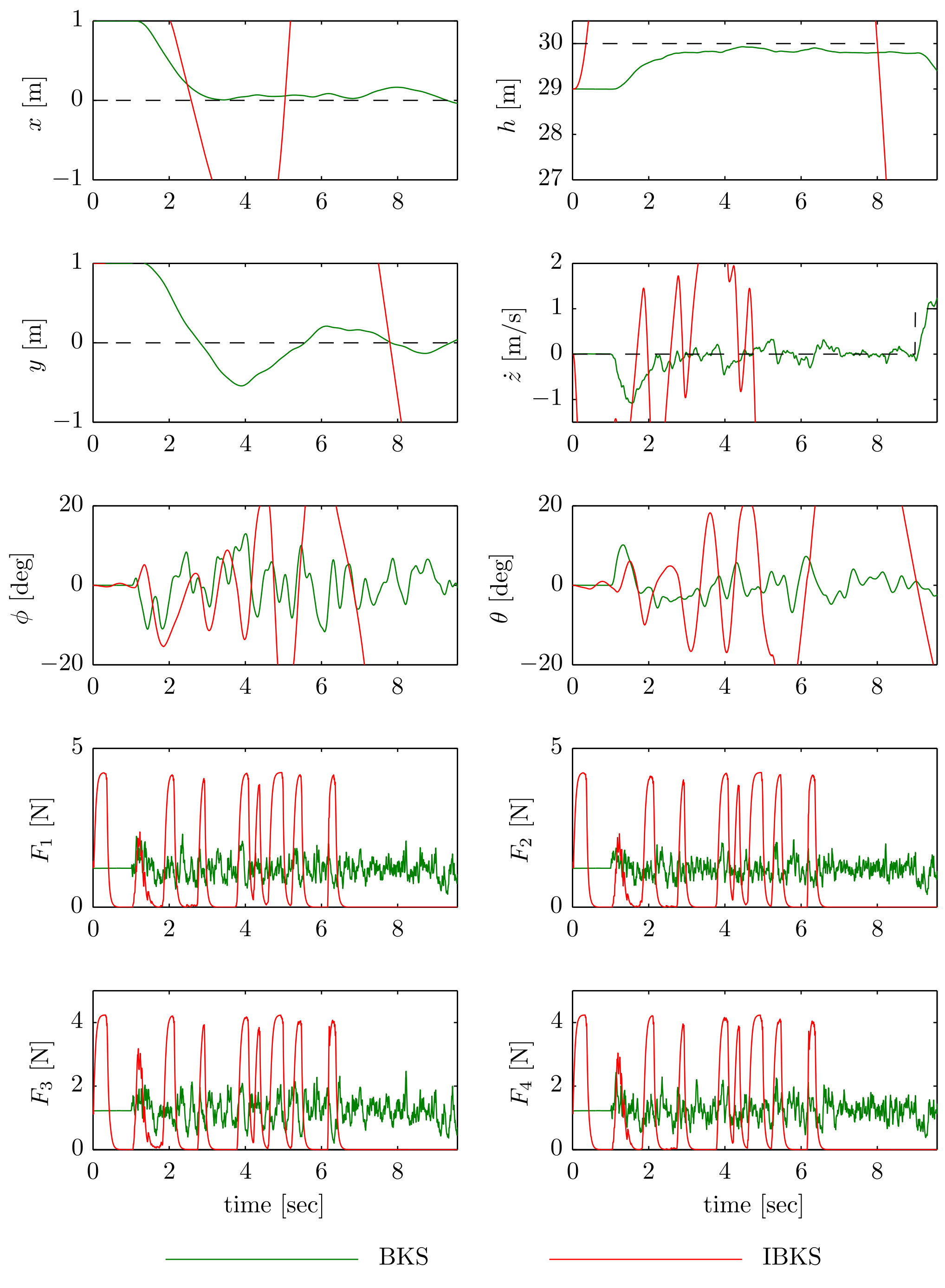


Figure 8-26: Controller performances for extreme case of uncertainty in rotor gyro effect ($\epsilon = 650$); F_1, F_2, F_3 and F_4 denote the control forces for each motor

8-5 Discussion

Two sets of simulations were performed: one without any state estimator in the loop (perfect state knowledge) and another with the state estimator in the loop. For each set, the controllers were tested for the case in which the model is completely known and also for model mismatch cases. For the latter, five robustness tests were performed for uncertainties in mass, inertia, motor torque and time constants and gyro effect. The results obtained are now critically analyzed and a discussion is given separately for perfect state knowledge and with the estimator in the loop.

8-5-1 Critical analysis of results with perfect state knowledge

200-sample sets of external conditions were used to test the controllers for the case in which perfect state knowledge is assumed. Note that this assumption is unrealistic, but it is important to evaluate the control laws without estimation errors in order to allow for a fair comparison between them. For instance, incremental backstepping uses feedback signals of angular accelerations, while the linear controller and standard backstepping do not. An erroneous estimation of such quantities may lead to poorer performance of the incremental-based control law when, in fact, the estimation block may be the possible cause of worse performance. The control laws are then first examined without the inclusion of the estimator to allow for the evaluation of the control law alone.

The first simulation shown concerns the best case scenario: **no uncertainties** with perfect state knowledge. All controllers allowed the quadrotor to successfully land on the buoy. From Figure 8-3, note that when further away from the target, the helicopter descends without taking into consideration the platform's motion; however, when closer to it, the vertical control is adjusted to cope with the buoy's dynamics. It is clear that the differences between the results obtained for the different controllers are very small. This shows that for such optimal conditions, no major conclusions can be drawn. The robustness tests, however, present conclusive results. The new findings are explained below.

For a **mass** uncertainty of 25%, a steady-state error is observed in sink rate for the linear and backstepping controllers. This result was expected, since no integral action was added. However, even without integral gains the incremental backstepping controller was capable of following the command signals, and the desired mean value of approximately 0.5 m/s was achieved. Furthermore, an improvement is observed for the incremental-based control law regarding the dispersions of all assessment criteria parameters, which are lower compared to those of the linear and backstepping controllers.

As for a 30% deviation in **inertia**, no noticeable differences between controllers are observed. In fact, the results are very similar to those obtained for no uncertainties. An exaggerated and unrealistic mismatch is tested considering $\epsilon = 4$. Note that while for the linear and backstepping controllers satisfactory performance is still observed, the same cannot be said for incremental backstepping, for which large oscillations in attitude angles are observed. This is due to an increase in the control activity. This last case is unrealistic and should never be expected in controller design; however, it is included to demonstrate the mentioned control law properties.

The performance for an uncertainty in **motor torque constant** was also tested. The statistical data shows that the effects of this uncertainty are very similar to the effects of mass uncertainty (see above for details). However, one important conclusion can be drawn. Recall that incremental-based control laws require the system's current input. Since the actual actuator output cannot be measured, it could be expected that the performance for incremental backstepping would suffer when dealing with actuator modeling uncertainties. However, a motor torque constant deviation of 20% did not hinder the controller performance, showing that incremental backstepping is somehow robust to actuator gain uncertainities.

The next test is a continuation of the previous. Actuators have gain factors as well as time delays; therefore, a **motor time constant** uncertainty is introduced. The statistical data analysis is identical to that explained for the mass and motor time constant uncertainties. The effects of incorrect assumption of the actuator time delay are explained as follows. When computing the system's current input, if the real actuator time delay is larger, then the control law will compute the wrong input. This may in turn affect stability properties. It is observed, however, that incremental backstepping is somehow robust for a deviation of 30% in this parameter. A larger deviation is also shown for $\epsilon = 2$ for which oscillations in the attitude angles are observed for the incremental-based control law.

The **rotor gyro effect** uncertainty was introduced to test how the controllers would reject disturbance torques. Since the moments generated by this effect are not measured (or estimated), and therefore are not taken into account in the controller design process, they are seen as disturbances. For the case in which the gyro effect is 100 times stronger, the differences are not significant. However, to show the advantage of using incremental backstepping, an extreme case is tested assuming an uncertainty factor of 6500 in gyro effect. The closed-loop response with the incremental-based controller was similar to the nominal response, while the system became unstable for the linear and backstepping cases. Note that when using incremental backstepping, the information from this disturbance is contained in the angular accelerations that are fed back to the controller. It should be stressed that this extreme case is unrealistic, but the order of magnitude of such disturbance could be the result of other effects in a realistic simulation. The main conclusion to be drawn is that incremental backstepping possesses better large torque disturbance rejection properties.

Overall, it can then be concluded that incremental backstepping is more robust to model uncertainties. One of its main advantages is its integral-like action without the drawbacks of using integrators. Additionally, although the incremental-based control law requires an actuator model in the controller, it is robust to realistic actuator uncertainties. Furthermore, it was observed that this control law possesses very powerful rejection capabilities for strong disturbance torques.

8-5-2 Critical analysis of results with state estimator in the loop

More realistic simulations were performed by including the state estimator in the loop. It was already shown in Chapter 6 that although it is possible to reconstruct the states with the sensors used, the estimation was not perfect. It is then interesting to evaluate the performance of the different controllers in the presence of state estimation errors.

First, when testing the linear controller with the state estimator in the loop, very poor performance was observed (note that low-quality sensors are used). In fact, without any adjustments, the controller could not follow the target and divergent responses were obtained. The controller was then subjected to further tuning until satisfactory performance was observed. Although it could then successfully land on the buoy for the nominal case, crashes were still frequent in Monte-Carlo analyses. Further improvement of the linear controller was considered irrelevant for the research being carried out, as the main focus lies on incremental backstepping. Nevertheless, one main conclusion can already be drawn. Recall that the first controller to be tuned was the linear controller, and the exact same gains were used for backstepping and incremental backstepping. Furthermore, the backstepping-based control laws presented stable behavior without any adjustments, while the controller based on linear theory required further tuning to enhance performance. From these findings, it can be concluded that the stabilizing terms obtained in the backstepping designs are helpful when estimation errors are present.

The first simulation shown for the estimator in the loop is in fact the main simulation: **no uncertainities** and state estimation block in the loop. As it pertains to the case that determines whether the control laws serve their purpose, 1000 samples were used for the Monte-Carlo analysis. The results show that both backstepping and incremental backstepping are suitable control laws for the mission under consideration (to validate this statement even further, a risk analysis is carried out as presented in Section 8-6). An increase in parameter dispersion is observed compared to the perfect state knowledge case and is naturally due to state estimation errors. Furthermore, it is observed that no major differences exist between the controllers when analyzing the statistical data. Incremental backstepping, however, shows a slight improvement with respect to TD sink rate characteristics. Taking a closer look into the final landing phase for one case scenario (see Figure 8-16), it is possible to observe that the desired vertical velocity command is tracked. This means that the controllers are working properly in combination with the augmented Kalman filter proposed for estimation of the platform's vertical motion. Note that the peak frequency of the sea spectrum was obtained on board (from a buffered signal) with the frequency analysis method explained in Section 6-2-4.

For the robustness tests, 200 samples were used for each uncertainty.

For a 25% **mass** uncertainity, vertical velocity control was not properly achieved for backstepping; this was expected after analyzing the perfect state knowledge case. Furthermore, the dispersion for this parameter is larger when compared to that of incremental backstepping. The results observed present very similar trends to those obtained for the mass uncertainity with perfect state knowledge.

A test with **inertia** uncertainty has also been presented, showing that no major differences were observed for the two control laws. The presence of state estimation errors produced similar effects for both controllers; nevertheless, a larger deviation of the pitch angle at TD is observed for incremental backstepping. Furthermore, a slight improvement is observed for the incremental-based control law for sink rate characteristics at TD. Oscillations in attitude are also observed for incremental backstepping in the case of large inertia mismatch.

The trends observed for actuator uncertainties (**motor torque constant** and **motor time constant**) were identical to those observed for the case in which perfect state knowledge was assumed. Note, however, that incremental backstepping now presents larger deviations for the attitude angles at TD. For a larger time constant deviation, attitude oscillations are again observed for incremental backstepping.

When introducing the **rotor gyro effect** uncertainty with $\epsilon = 100$, no major differences between the two control laws were observed. However, the same cannot be said for a larger uncertainty regarding this effect. Recall the factor used ($\epsilon = 6500$) for the perfect state knowledge case, for which incremental backstepping showed significantly better robustness properties. When using the state estimator for $\epsilon = 650$, the incremental-based control law did not render stability while the backstepping control law did. This means that the incremental backstepping disturbance rejection properties are strongly affected by angular acceleration estimations.

It is concluded that although incremental backstepping presents better robustness properties overall, even with the state estimator in the loop, the performance sometimes deteriorates because of estimation errors. Specifically, the estimation of angular accelerations is key for the problem analysis.

8-6 Risk analysis

A risk analysis, (Looye, 2007), can be performed by calculating the probability of crashing. It only makes sense to do so for the case in which state estimation algorithms are used in the loop, as this situation is closer to reality.

First, since the design of the landing platform is open in this phase of the project, the radius of the landing area can be designed by using the Monte-Carlo data for the worst case scenario. Figure 8-27 shows the risk (probability) of a failed landing as a function of the to-be-designed landing area radius. It is clear that for small radius values, the probability is unacceptably high. By selecting a risk threshold of 10^{-6}, a radius of 0.52 m is obtained. This means that the designated area allowed for landing would have a diameter of approximately 1 m.

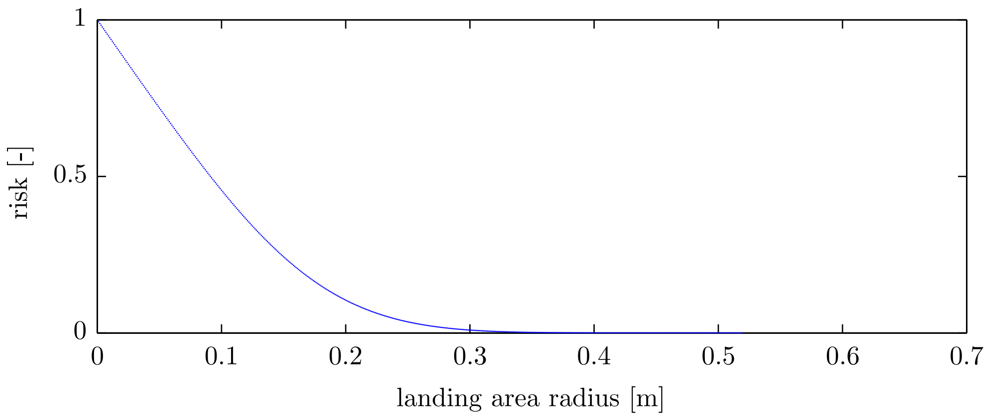


Figure 8-27: Risk (probability) of failed landing as a function of the landing area radius

The parameter thresholds for risk analysis demonstrations are defined as follows:

- $|PHITD| > 10\ deg$
- $|TTATD| > 10\ deg$

- $|XTD| > 0.52\ m$
- $|YTD| > 0.52\ m$
- $ZDTD > 2\ m/s$

It should then be proven that the probabilities of the above conditions are residual.

Table 8-13 shows the results of the risk analysis, from which it is possible to conclude that the probability of crashing is, in fact, residual.

Table 8-13: Risk analysis from Monte-Carlo simulation: 1000 samples, state estimator in the loop and no uncertainties

	PHITD	TTATD	XTD	YTD	ZDTD
			risk		
BKS	$1.8534 \cdot 10^{-6}$	$3.0203 \cdot 10^{-8}$	$1.0101 \cdot 10^{-24}$	$9.7684 \cdot 10^{-9}$	$8.0484 \cdot 10^{-7}$
IBKS	$4.0663 \cdot 10^{-7}$	$3.4808 \cdot 10^{-8}$	$4.8824 \cdot 10^{-20}$	$9.5568 \cdot 10^{-7}$	$9.3917 \cdot 10^{-8}$

Chapter 9

Real-time implementation

For this study, a single test quadrotor with an on-board camera is used. In this chapter, an overview of the real-time platform is presented from both a hardware and software perspective. Implementation of the algorithms was tested in the MAVlab, using a safe setup which included ropes to restrain the helicopter's movement. A description of the real-time implementation of the Kalman filter for estimation of position and velocity is given, as well as of the controller as implemented on board. The various steps taken in order to achieve stable flight are also described, and the results observed are discussed.

9-1 Hardware overview

The quadrotor possesses a cross-shaped frame. Self-designed base plates are the core of the assembly and include a clamping system for the beams. These hold the engines and propellers at their outer ends. All cables run through the beams (instead of outside of them). In the center, the autopilot is attached to the rest of the quadrotor through four pins which damp vibrations. Finally, the battery is placed below the core of the assembly and the camera is positioned close to the CM, pointing downwards.

The base plates and beams together weigh 187 grams while each motor weights 61 grams. The total weight of the vehicle without battery is 589 grams, meaning that the remaining components together weigh 244 grams. Depending on the battery used, the total weight of the quadrotor is then between 700 grams and 1 kg.

A short list of the most important components used for real-time implementation is given below.

- Frame: Glass fiber base plates hold aluminum beams, that are hollow square tubes with outside dimensions 10x10x180 mm, and 1 mm thickness.
- Motors: Robbe Roxxy 2827-34 brushless engines.

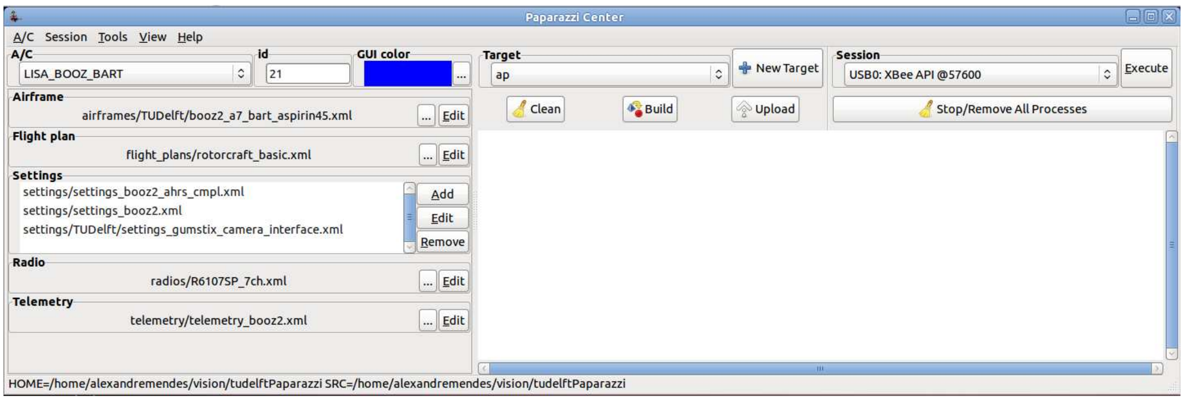
- Rotor: 12 cm radius propellers.
- Autopilot board: Lisa/L. Lost Illusions Serendipitous Autopilot (Lisa) is a new range of autopilots based on STM32 microcontrollers designed to run Paparazzi (software introduced in the next section). Lisa/L is a dual processor board autopilot designed to allow Linux to be used for Paparazzi airborne code.
- Battery: Lithium polymer with capacity of 3300 mAh (another one of 1800 mAh is also available). With these types of batteries, the quadrotor has a flying time endurance of approximately 10 minutes.
- Camera: CASPA VL from Gumstix (all details are given in Section 5-5).

All components mentioned are available at the MAVlab facilities.

9-2 Software overview

The software used for real-time implementation and testing is from the free and open-source project Paparazzi. This software suite contains everything needed for an unmanned airborne system to fly reliably. Highly effective stability and navigation code is also included; all algorithms implemented for this thesis were then additions or modifications to existing software.

The Paparazzi Center (see Figure 9-1) is a graphical user interface that can be used for real-time testing and includes: a section containing a set of configurations boxes, a section for building the program and uploading code to the quadrotor and a section for execution.



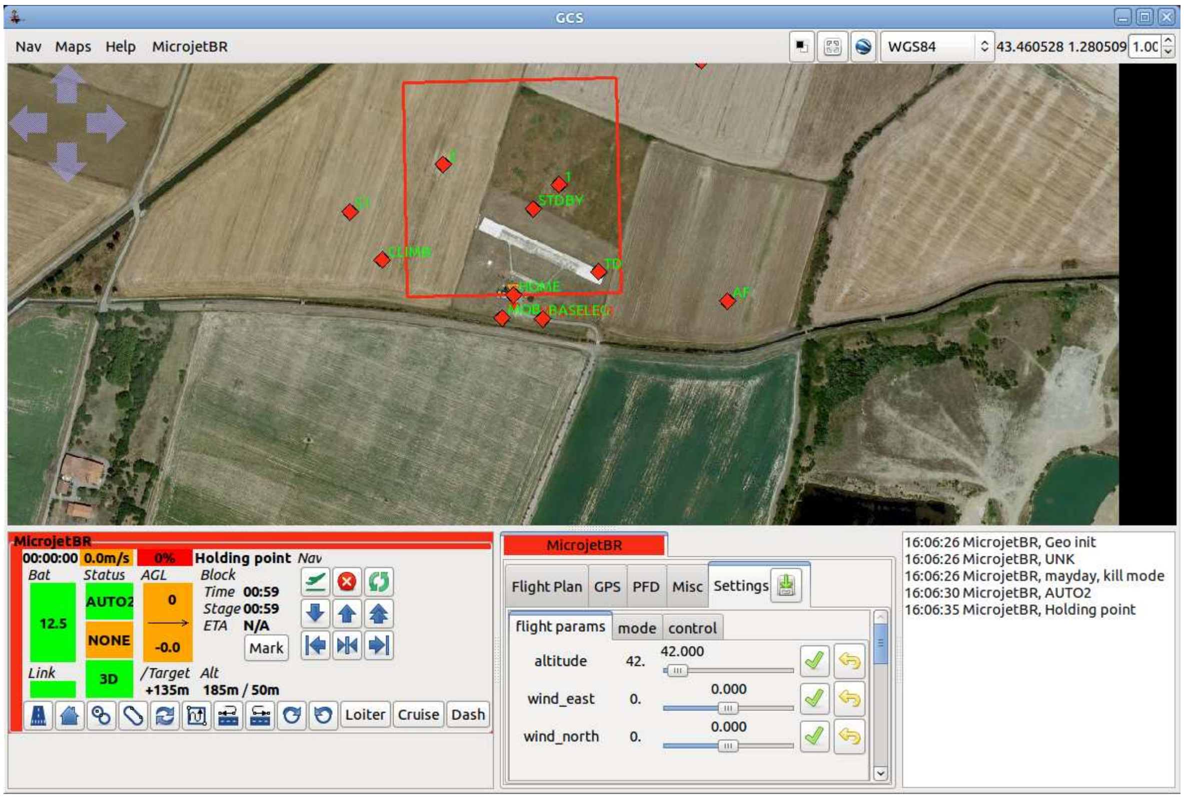
A/C: LISA_BOOZ_BART
id: 21
GUI color: 🟦
Target: ap
Session: USB0: XBee API @57600
Airframe: airframes/TUDelft/booz2_a7_bart_aspirin45.xml
Flight plan: flight_plans/rotorcraft_basic.xml
Settings:
settings/settings_booz2_ahrs_cmpl.xml
settings/settings_booz2.xml
settings/TUDelft/settings_gumstix_camera_interface.xml
Radio: radios/R6107SP_7ch.xml
Telemetry: telemetry/telemetry_booz2.xml

```
HOME=/home/alexandremendes/vision/tudelftPaparazzi SRC=/home/alexandremendes/vision/tudelftPaparazzi
```

Figure 9-1: Paparazzi center

A set of tools is also available to facilitate real-time testing tasks. What follows is a list and short description of the most commonly used tools. As an example, a simulated flight of a microjet is shown for altitude change maneuvers.

- Ground Control Station (GCS), Figure 9-2, which allows the operator to monitor important flight information such as battery level, datalink status, modes available, etc. A settings tab is available that permits the operator to change variable values during flight, including controller parameters and desired set points.



altitude	42.	42.000
wind_east	0.	0.000
wind_north	0.	0.000

```
16:06:26 MicrojetBR, Geo init
16:06:26 MicrojetBR, UNK
16:06:26 MicrojetBR, mayday, kill mode
16:06:30 MicrojetBR, AUTO2
16:06:35 MicrojetBR, Holding point
```

Figure 9-2: Paparazzi GCS

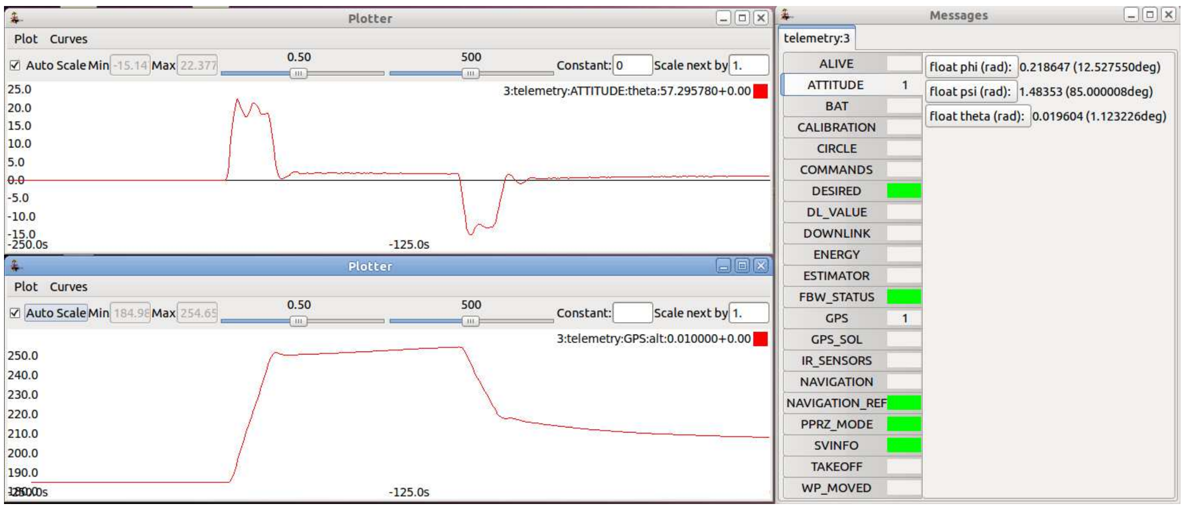
- Messages, which allows for visualization of the variables being sent in the telemetry. With this tool, the operator has access to the measurements' numerical values.

- Real-time plotter, which is a tool to plot real-time data broadcast on the Paparazzi network bus. The main features are: drag and drop from the messages window, plot multiple curves in single window including constant lines and scale variables.

  Figure 9-3 shows an example of how the microjet maneuver can be monitored in real time with this tool. The real-time plots show the pitch angle and altitude of the aircraft. At any time, new messages can be dragged to the window for visualization of other variables.

- Log plotter, which is a tool to plot data recorded in a log file. After a flight test, all values can be exported in Comma-Separated Values (CSV) format to a text file using this tool. The main features are similar to those of the real-time plotter.

  For the example shown, it could be useful to perform a close analysis of the pitch response with respect to the desired value commanded by the controller. This situation is illustrated in Figure 9-4.



Auto Scale Min -15.14 Max 22.377 0.50 500 Constant: 0 Scale next by 1.

3:telemetry:ATTITUDE:theta:57.295780+0.00 🟥

Auto Scale Min 184.98 Max 254.65 0.50 500 Constant: Scale next by 1.

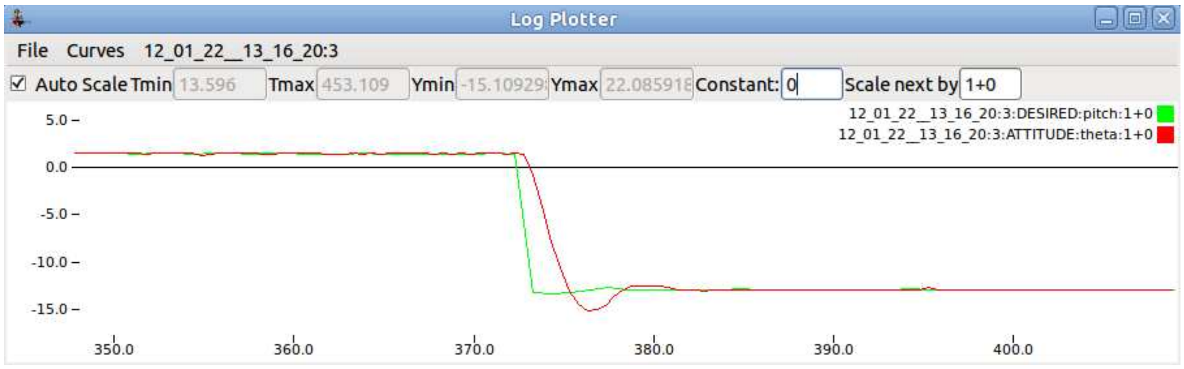
3:telemetry:GPS:alt:0.010000+0.00 🟥

telemetry:3

ALIVE	
ATTITUDE	1
BAT	
CALIBRATION	
CIRCLE	
COMMANDS	
DESIRED	🟩
DL_VALUE	
DOWNLINK	
ENERGY	
ESTIMATOR	
FBW_STATUS	🟩
GPS	1
GPS_SOL	
IR_SENSORS	
NAVIGATION	
NAVIGATION_REF	🟩
PPRZ_MODE	🟩
SVINFO	🟩
TAKEOFF	
WP_MOVED	

float phi (rad):	0.218647 (12.527550deg)
float psi (rad):	1.48353 (85.000008deg)
float theta (rad):	0.019604 (1.123226deg)

Figure 9-3: Paparazzi messages and real-time plotter



Auto Scale Tmin 13.596 Tmax 453.109 Ymin -15.10929 Ymax 22.085918 Constant: 0 Scale next by 1+0

12_01_22__13_16_20:3:DESIRED:pitch:1+0 🟩
12_01_22__13_16_20:3:ATTITUDE:theta:1+0 🟥

Figure 9-4: Paparazzi log plotter

9-3 Testing setup

Helicopters are dangerous due to their spinning rotor blades. Furthermore, in the event of a crash, repair of the quadrotor could take up to several weeks. Therefore, to avoid accidents a test setup was prepared in order to achieve maximum safety. Four ropes were attached to the end of each arm and one rope was attached to the quadrotor's core. The first four were tied to heavy objects on the floor, and the other core stabilizing rope was secured to the ceiling. By allowing these ropes to remain loose, a volume of operation is created. The movements' limits of operation are reached when one or more ropes are pulled taut.

Note that it is important to ensure that these ropes are not caught in the blades. For this reason, a new feature was introduced to avoid such a situation. Stiff components were added to the portions of rope nearest to the rotors, positioned to point away from the blades as much as possible. A schematic representation is presented in Figure 9-5, followed by the real lab setup ready for testing in Figure 9-6.

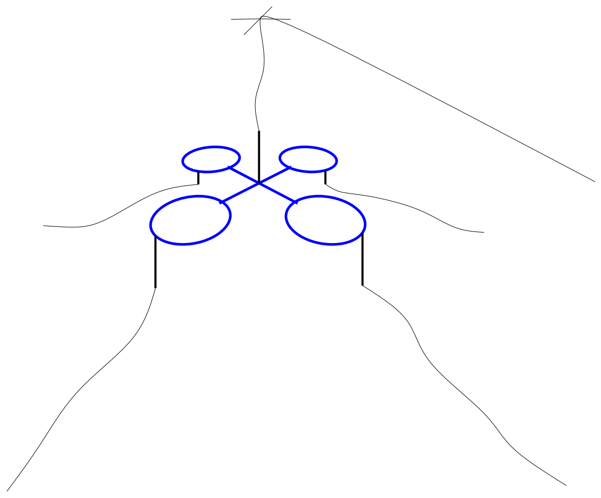


Figure 9-5: Test setup

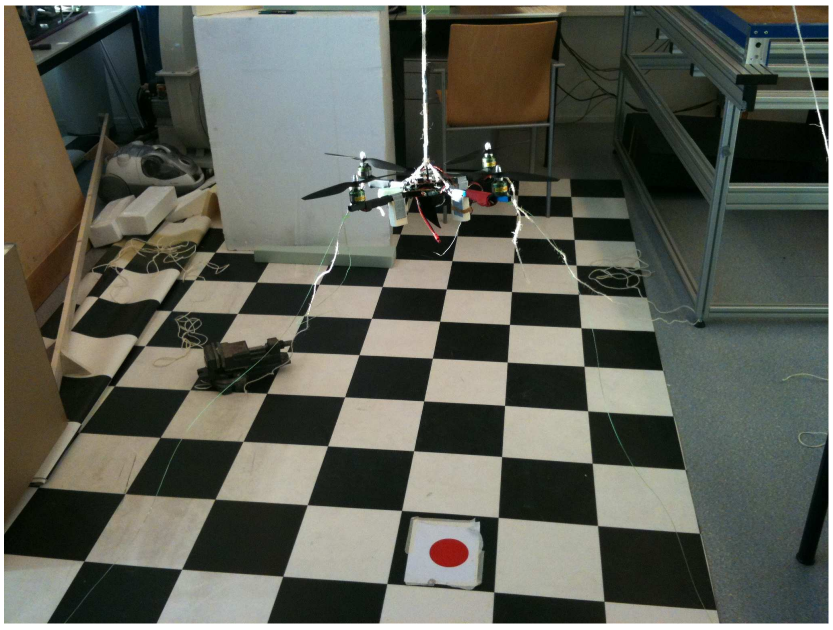


Figure 9-6: Quadrotor ready for testing

9-4 System overview

The real-time system overview is presented in Figure 9-7. What follows is an explanation of the different system blocks.

- Quadrotor. This block describes the helicopter's behavior and includes actuators and sensors.
- State estimation. Some states are directly available as measurements and can be used readily in the controller. However, some other states must be estimated. In this study,

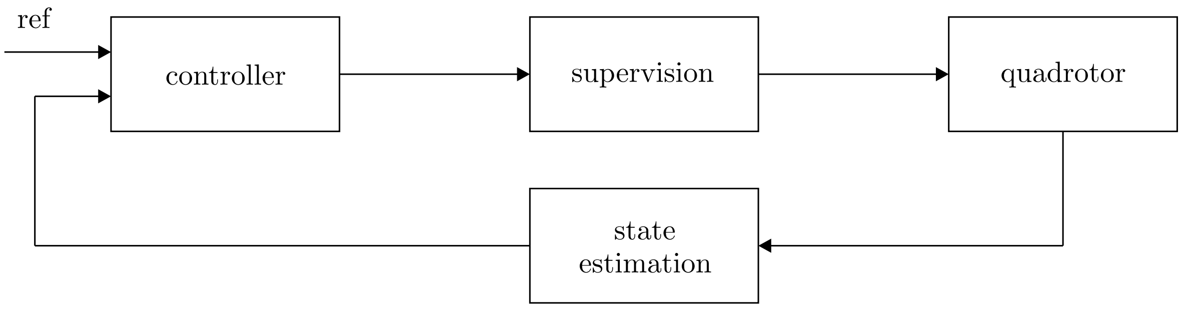


Figure 9-7: Real-time system overview as implemented in Paparazzi

a Kalman filter for integration of IMU with vision data was implemented in real time (presented in Section 9-5). An Attitude Heading Reference System (AHRS) was already implemented in Paparazzi for attitude estimation and was used without any modification.

- Controller. The control algorithm was divided into
  - horizontal loop, concerning the vehicle's longitudinal and lateral inertial position variables (x and y);
  - vertical loop, concerning the vehicle's vertical inertial position variable (z).

  These control loops are explained in detail in Section 9-6.

- Supervision. This block is equivalent to the linear control allocation explained in Section 7-5-2. However, since the actuator model is not known, a simpler version is implemented as explained next. The controller produces three torque commands (cmd_{roll}, cmd_{pitch} and cmd_{yaw}) and one thrust command (cmd_{thrust}). Therefore, a transition matrix is applied to obtain the required input U_i for each motor as follows

$$\begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & -1 & 1 \\ 0 & -1 & -1 & 1 \\ -1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} cmd_{roll} \\ cmd_{pitch} \\ cmd_{yaw} \\ cmd_{thrust} \end{bmatrix} \tag{9-1}$$

  The input is then saturated by the minimum and maximum allowed values, before a command is given to the actuators.

  Note that since the control allocation is done without physical knowledge of the motor characteristics, it is necessary to tune the controller accordingly, i.e., manual tuning is required.

9-5 Kalman filter implementation

The filter implemented in real time is a linearized version of the Kalman filter developed in Section 6-3 and, for convenience, is briefly explained here. Kinematic acceleration is the second-time derivative of position

$$\ddot{x} = a_x \tag{9-2}$$

Let $x_1 = x$ and $x_2 = \dot{x}$ and consider a measurement of acceleration given by

$$a_m = a + \lambda_a + w_a \tag{9-3}$$

where λ_a is a bias and w_a is measurement noise. Furthermore, let $x_3 = \lambda_a$ and assume that the bias is constant such that

$$\dot{x}_3 = \dot{\lambda}_a = 0 \tag{9-4}$$

Assume now that a measurement of position is available and is described by the following equation

$$x_m = x + v_x \tag{9-5}$$

The following state space description is then obtained

$$\begin{aligned}
\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\end{bmatrix} &= \begin{bmatrix}0 & 1 & 0\\ 0 & 0 & -1\\ 0 & 0 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix} + \begin{bmatrix}0\\ 1\\ 0\end{bmatrix} a_m + \begin{bmatrix}0\\ -1\\ 0\end{bmatrix} w_a\\
y &= \begin{bmatrix}1 & 0 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix} + v_x
\end{aligned} \tag{9-6}$$

which is in the same format as the system (6-1), meaning that the discrete linear Kalman filter algorithm can be applied.

Acceleration measurements are available from accelerometers and position measurements are available from the vision system. Therefore, the formulation described above can be used for estimation of the quadrotor's position and velocity. Note, however, that to obtain kinematic acceleration the accelerometer measurements must be compensated for gravity. Furthermore, both accelerometers and the camera measurements are expressed in the body-fixed reference frame; this means that rotations to the navigation frame are required, based on the estimated vehicle's attitude. It should be stressed that since no synchronization between IMU and vision data was achieved, rotation of camera measurements was not implemented.

The position/velocity filter for integration of vision with IMU data was implemented in the autopilot board, and the filter output was then used for control purposes. Accordingly, the objective of the first tests conducted was to tune the Kalman filter parameters. Recall the Kalman filter formulation explained in Chapter 6, from which it follows that the tuning parameters are: the initial state conditions, the initial state covariance matrix and the covariance matrices of the process noise (Q_d) and measurement noise (R). In Section 6-1, an explanation is given on how to select the values for these parameters. However, when dealing with the actual system, manual tuning is required based on the filter's observed behavior.

What follows is an example concerning the aspect that is most relevant to achieving satisfactory flight results. The camera measurements are expressed in the body-fixed reference frame and no attitude correction is made in order to rotate these to the navigation frame. The consequences of using non-compensated measurements are explained in Section 9-8. To tackle this issue, it is possible to tune the Kalman filter to rely more on twice-integrated accelerometer measurements. By increasing the value of R, artificial measurement noise is introduced in the filter making the camera measurements less reliable. To illustrate this tuning procedure, Kalman filter results for $R = 0.1$ and $R = 8$ are shown in Figures 9-8 and 9-9, respectively. Note that for a greater R value, the filtered output follows the camera measurements less strictly. With this strategy, the vision system is then mostly used for accelerometer bias estimation. It should be stressed that while this solution is far from ideal, it certainly is functional. Nevertheless, camera measurements are still more accurate, meaning that better performance is expected when attitude compensation is performed.

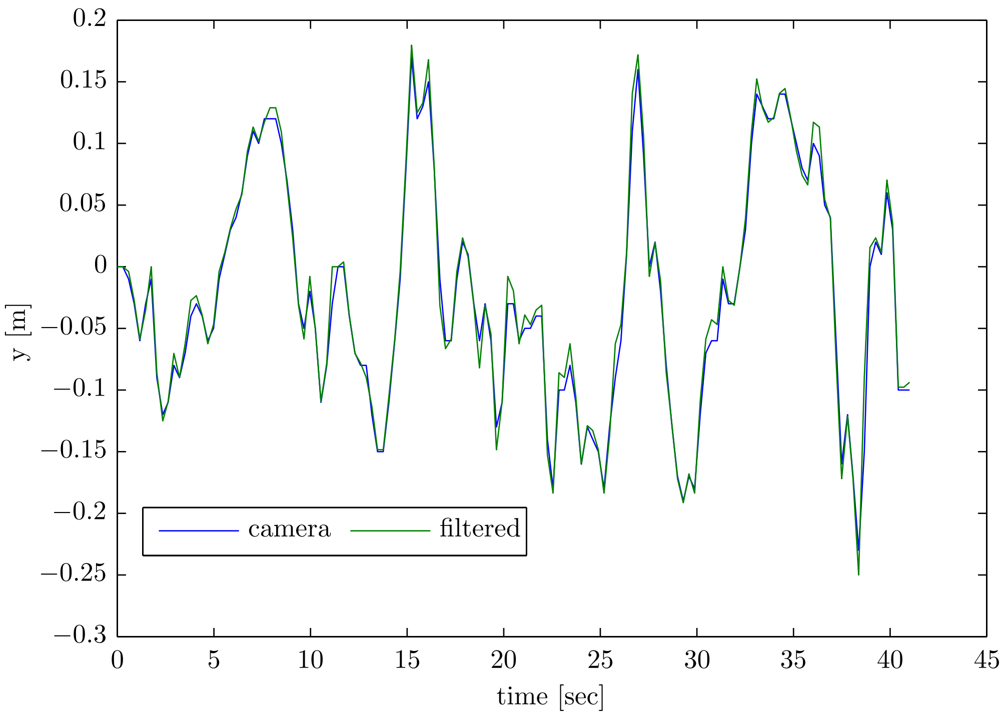


Figure 9-8: Kalman filter example for the y coordinate; $R = 0.1$

After the tuning procedure, reliability tests were performed to check the correctness of the Kalman filter implemented on board, while functioning in real time. An example is presented in Figure 9-10. Table 9-1 presents mean values and standard deviations of the innovation sequences shown.

Table 9-1: Statistical data of the innovation sequences for the Kalman filter operating in real time

Innov. sequence	Mean (m)	Std. deviation (m)
x	-0.0020	0.0357
y	0.0053	0.0273
z	-0.0020	0.0115

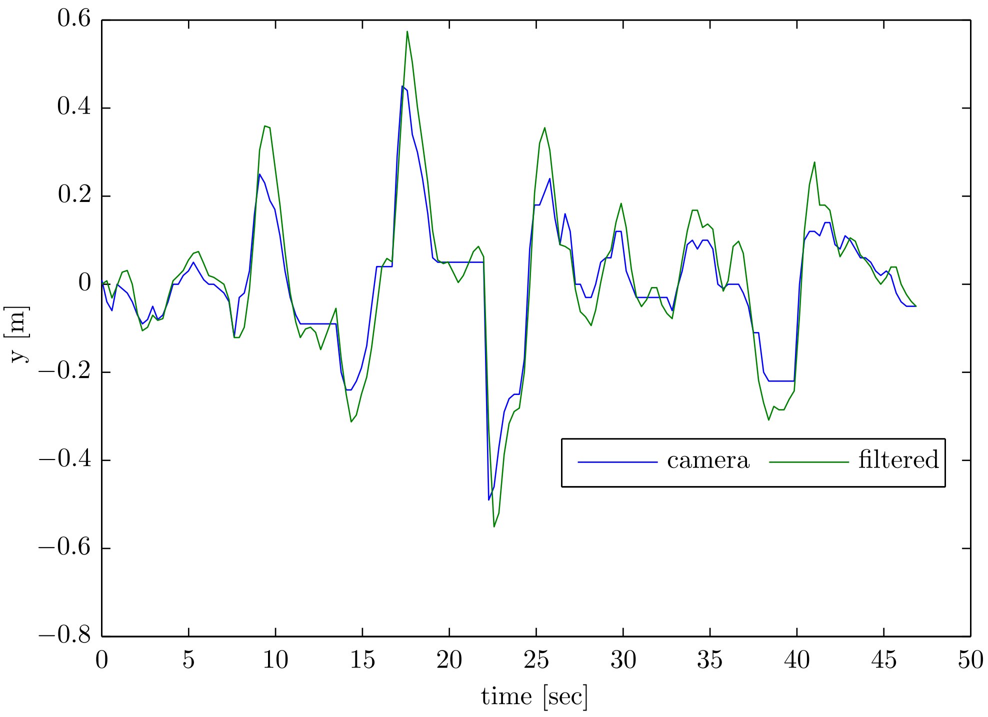


Figure 9-9: Kalman filter example for the y coordinate; $R = 8$

9-6 Real-time controller

One of the contributions of this study was a complete description the multi-rotor controller as implemented in Paparazzi. Since Paparazzi is an open-source project, all control loops are available online from http://paparazzi.enac.fr/wiki/Control_Loops (Multi-rotor section).

A modified version of the hover loop was utilized for the flight tests and, for convenience, is presented here. As mentioned previously, the controller is divided into two main loops: horizontal and vertical. These are now explained. It should be mentioned that all errors can be bounded before being multiplied by their respective gains. Furthermore, the command filters presented are second-order filters as described in Appendix D. Finally, the subscript *dd* stands for double-derivative.

9-6-1 Horizontal loop

Horizontal hover

The control laws for horizontal hovering are described according to the following equations

$$\begin{aligned} x_c &= K_{xp}(x - x_{sp}) + K_{xd}(\dot{x}) + K_{xi}\int(x - x_{sp})\mathrm{d}t \\ y_c &= K_{y_p}(y - y_{sp}) + K_{y_d}(\dot{y}) + K_{y_i}\int(y - y_{sp})\mathrm{d}t \end{aligned} \tag{9-7}$$

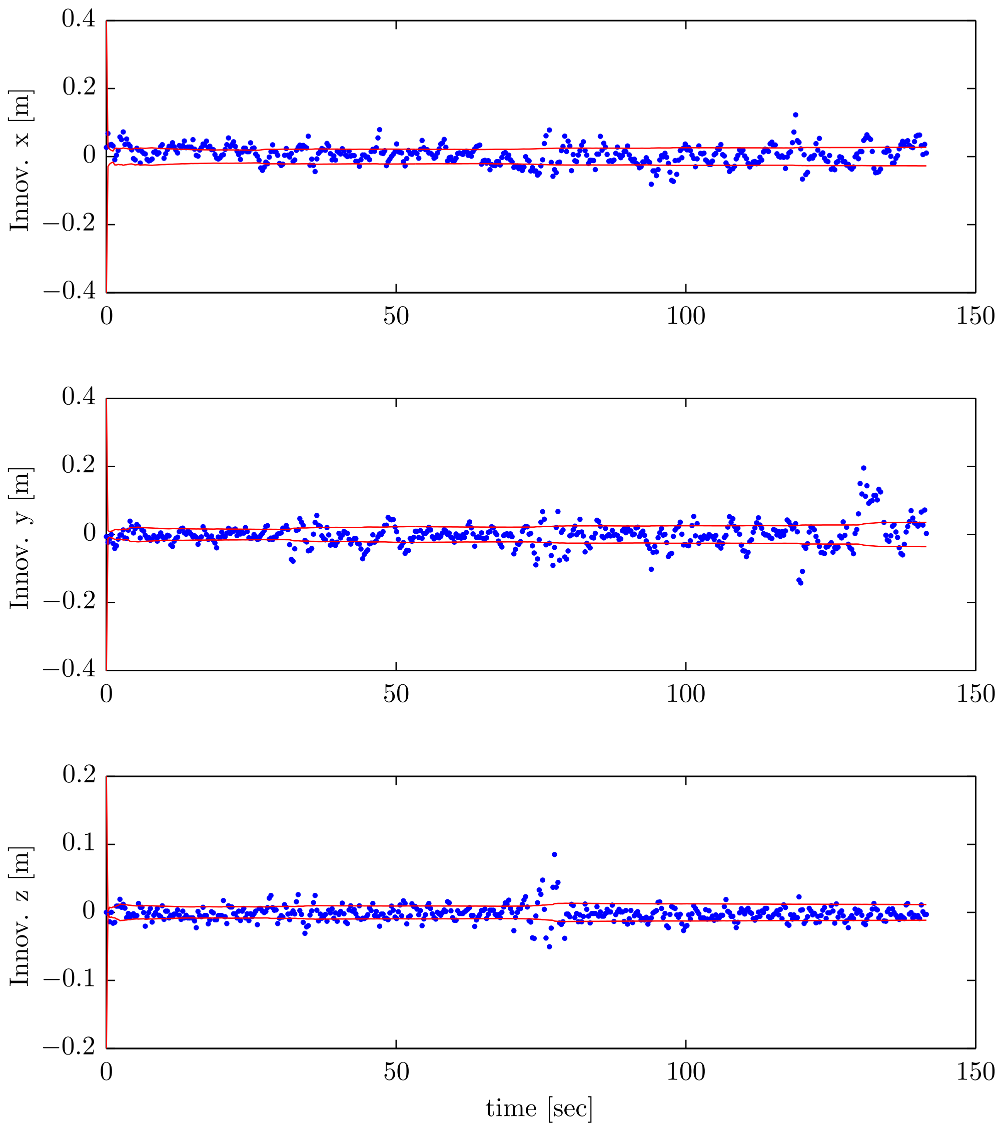


Figure 9-10: Innovation sequences from the Kalman filter functioning in real time; the red lines represent the standard deviation of the innovations

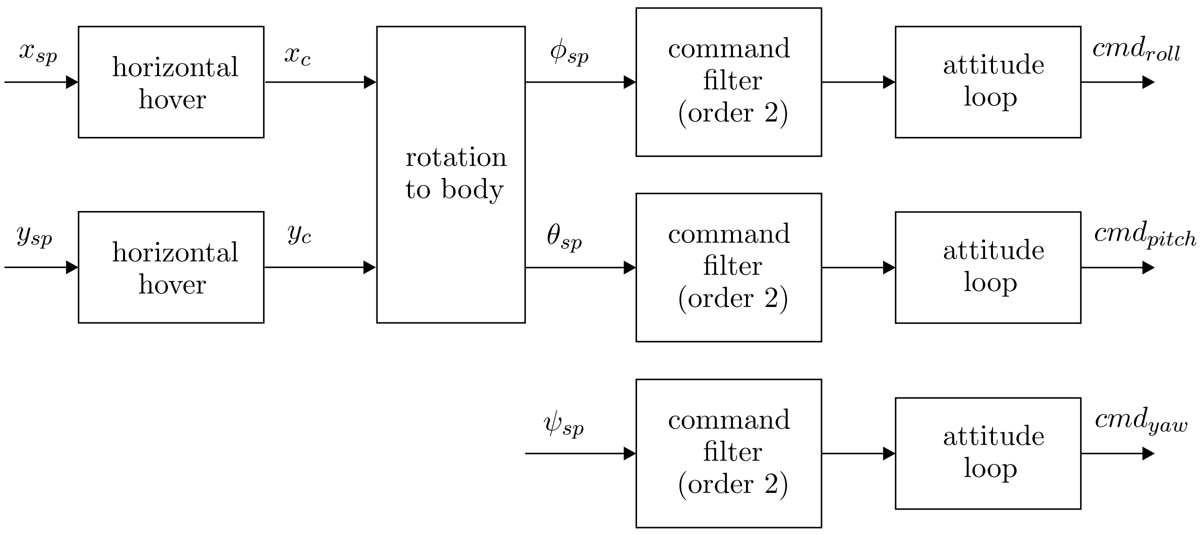


Figure 9-11: Horizontal loop

and the rotation to body is given as

$$\begin{aligned} \phi_{sp} &= -\sin\psi x_c + \cos\psi y_c \\ \theta_{sp} &= -(\cos\psi x_c + \sin\psi y_c) \end{aligned} \tag{9-8}$$

Attitude (inner) loops

The attitude loops as implemented in the real-time platform are described according to the following equations

$$\begin{aligned} cmd_{roll} &= K_{\phi_p}(\phi - \phi_c) + K_{\phi_d}(p - \dot{\phi}_c) + K_{\phi_i}\int(\phi - \phi_c)\mathrm{d}t + K_{\phi_{dd}}\ddot{\phi}_c \\ cmd_{pitch} &= K_{\theta p}(\theta - \theta_c) + K_{\theta d}(p - \dot{\theta}_c) + K_{\theta i}\int(\theta - \theta_c)\mathrm{d}t + K_{\theta dd}\ddot{\theta}_c \\ cmd_{yaw} &= K_{\psi_p}(\psi - \psi_c) + K_{\psi_d}(p - \dot{\psi}_c) + K_{\psi_i}\int(\psi - \psi_c)\mathrm{d}t + K_{\psi_{dd}}\ddot{\psi}_c \end{aligned} \tag{9-9}$$

9-6-2 Vertical loop

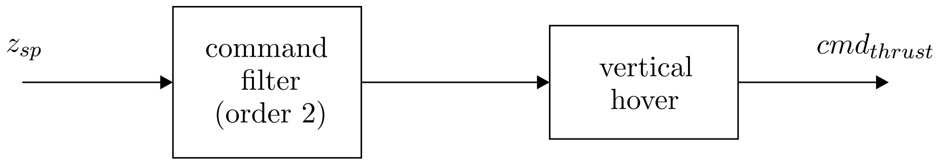


Figure 9-12: Vertical loop

Vertical hover

The vertical hover control loop is described by

$$cmd_{thrust} = K_{zp}(z - z_c) + K_{zd}(\dot{z} - \dot{z}_c) + K_{zi}\int(z - z_c)\mathrm{d}t + K_{zdd}\ddot{z}_c + trim \tag{9-10}$$

9-7 Towards stable flight

Since no model of the motors was available, it was not possible to select the controllers gains offline. Furthermore, as the control loops are coded in Paparazzi using fixed point arithmetic (integer representation) to improve speed, it is difficult to assign a physical meaning to the controller gains. Consequently, these gains were thus tuned manually. The key is making sure to tune the gains in the correct order.

9-7-1 Inner loop

The gains were first tuned to allow for manual control in the attitude mode (inner loop). When holding the quadrotor with the motors spinning, the rotors can be felt pushing the helicopter in a certain direction. By tilting the quadrotor at a certain angle to either side, the proportional gain should cause the quadrotor to attempt to regain zero-angle position and restore the trim condition. Also, the derivative gain should lead to damping of the velocity during transient periods. To tune this last gain, it is necessary to introduce fast rotation motions and see whether the controller is able to attenuate angular velocity. The integral gain is added in the end and increased by small increments until oscillations are observed. Then it is reduced to permit proper flying qualities.

The previous tuning should permit disturbance rejection. However, to follow reference orders commanded by an outer loop (either a human or an autopilot), two more parts of the controller must be tuned; first the reference commands, and then the double derivative gain. For the first, is it necessary to ensure high speed with large damping of the attitude mode. To achieve such performance the natural frequency is selected as a large value and the damping coefficient between 0.7 and 1. Finally, the double derivative gain is increased until fast response to attitude commands is reached.

9-7-2 Outer loop

The algorithms implemented for autonomous flight were tested with low gains in a first flight, in which several real-time implementation issues became evident. Namely, the corruption of magnetometer information indoors, which lead to false yaw measurements, caused problems not only for yaw stabilization but also for calculation of rotation matrices. This issue was tackled by faking magnetometer measurements ($\psi \approx 0$). Note that it is irrelevant whether the nose of the vehicle points to North or any other direction. Nevertheless, it is still important to ensure damping of the yaw motion, which is obtained using gyro information. All rotations after this step did not take into account the ψ angle.

Horizontal damping

To achieve stable flight, one of the most important steps is to obtain damping. The setup with the ropes was ideal for tuning this parameter, since it was possible to let the quadrotor hang on the rope attached to the ceiling.

The rope hanging from the ceiling worked as a sort of proportional gain for the horizontal motion. By letting the quadrotor swing from an initial position, it was observed that with no controller, the top rope would make the quadrotor swing around the equilibrium point with very little to low damping. When the autopilot was activated, the motion was damped by the derivative action of the horizontal outer loop. As an example, Figure 9-13 shows a damped oscillation observed during the tuning phase. The gain was increased until high frequency oscillations were observed. These oscillations were caused due to the fact that no attitude corrections were made for the visions measurements.

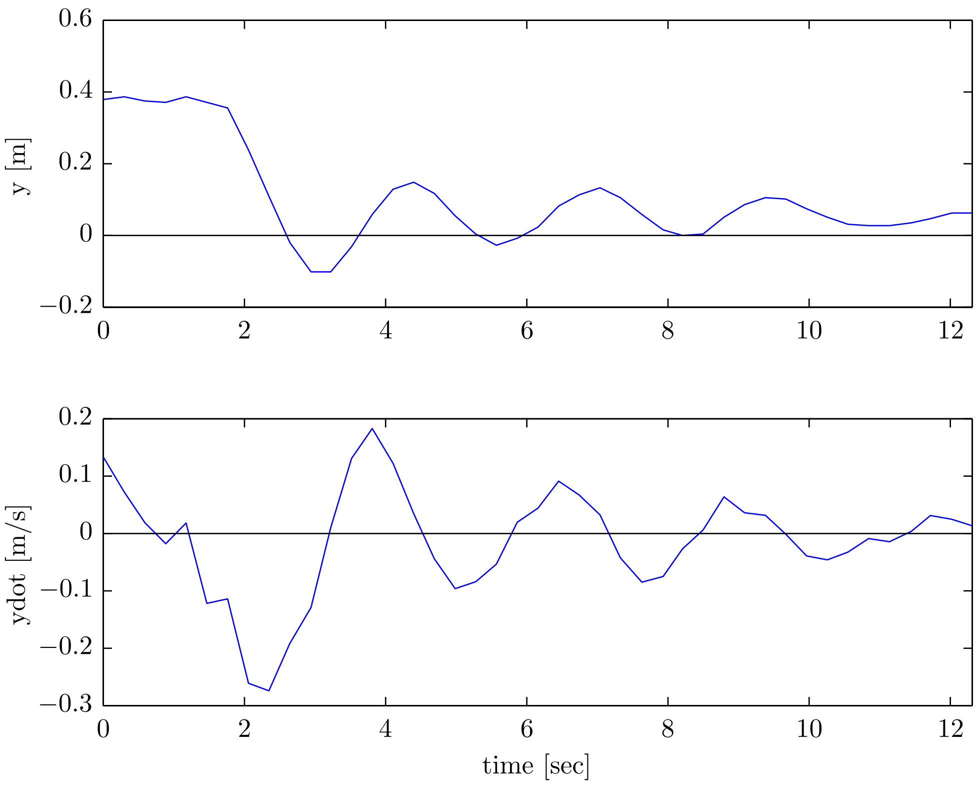


Figure 9-13: Damped horizontal motion; result obtain during tuning procedure

Vertical control

After achieving the highest damping possible for the vehicle's translational horizontal motion, it was necessary to tune the vertical loop. First the derivative gain was increased, and to test whether the corresponding action was effective, fast vertical motions were introduced by hand. When pushing the quadrotor down, all motors' power should increase; contrarily, when pulling the quadrotor up, all motors' power should decrease. Since this was only tested with derivative gain, changes were only observed during the transient periods.

Proportional gain was then added to ensure not only velocity but also vertical position control. This new action could be tested by changing the desired altitude set point and observing whether the quadrotor would try to climb up or descend as desired. Finally, a very low integral gain (compared to proportional and derivative of this mode) was included. The effects of this gain are only visible by analyzing whether the actual desired altitude is achieved or if a steady-state error is present.

Figure 9-14 shows an example of vertical motion control with all three actions implemented. The quadrotor starts at a resting altitude of 70 cm and a desired altitude of 95 cm is commanded from the GCS. Note that to climb, a negative peak in vertical velocity (upwards direction in the NED frame) is observed. Furthermore, the velocity while hovering is always close to zero.

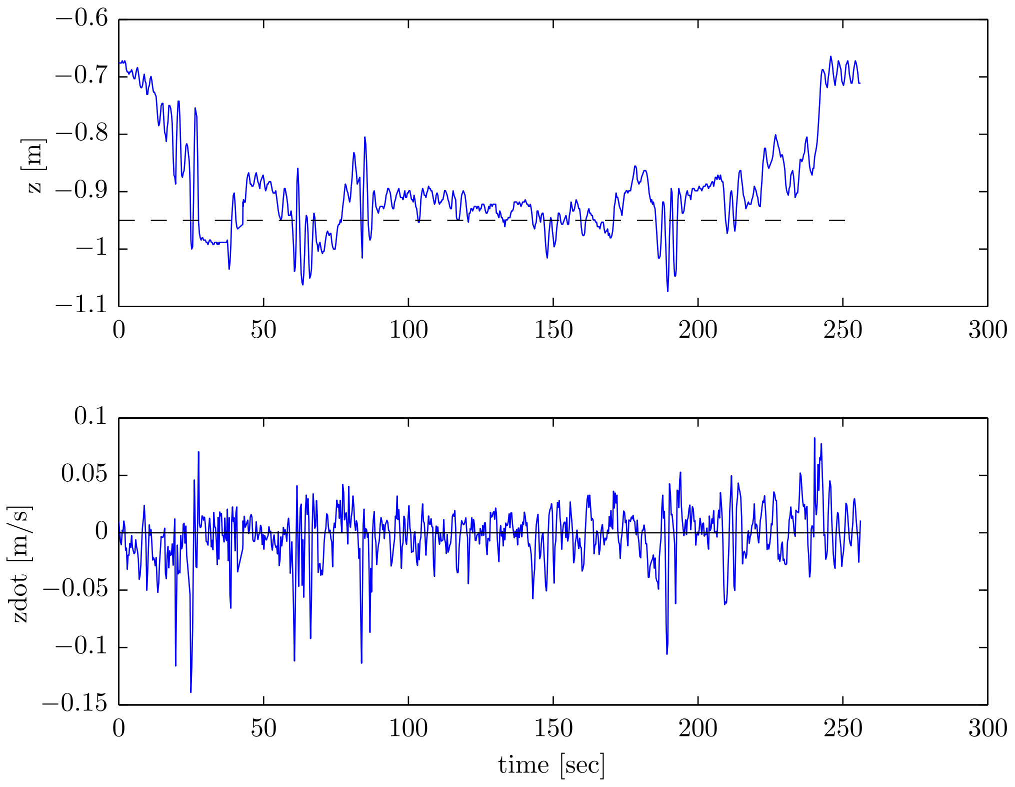


Figure 9-14: Vertical control in real time

9-7-3 Full free flight

With the control loops tuned as explained previously, a complete free flight can be performed. For this case, the ropes are made loose to allow more freedom of movement for the quadrotor. Contrary to the previous steps, at this point the safe testing setup is only used for extreme cases in which the vehicle could crash.

For these tests, the quadrotor starts at a resting position and an altitude change is commanded from the GCS. Since the system already possesses damped behavior for lateral motion, the quadrotor can hover above the target at the desired altitude. Results from one of these flight tests were logged and are shown in Figure 9-15. It is possible to observe that the quadrotor is not exactly aligned with the target; this was expected since no proportional and integral actions were added for the horizontal translational motion.

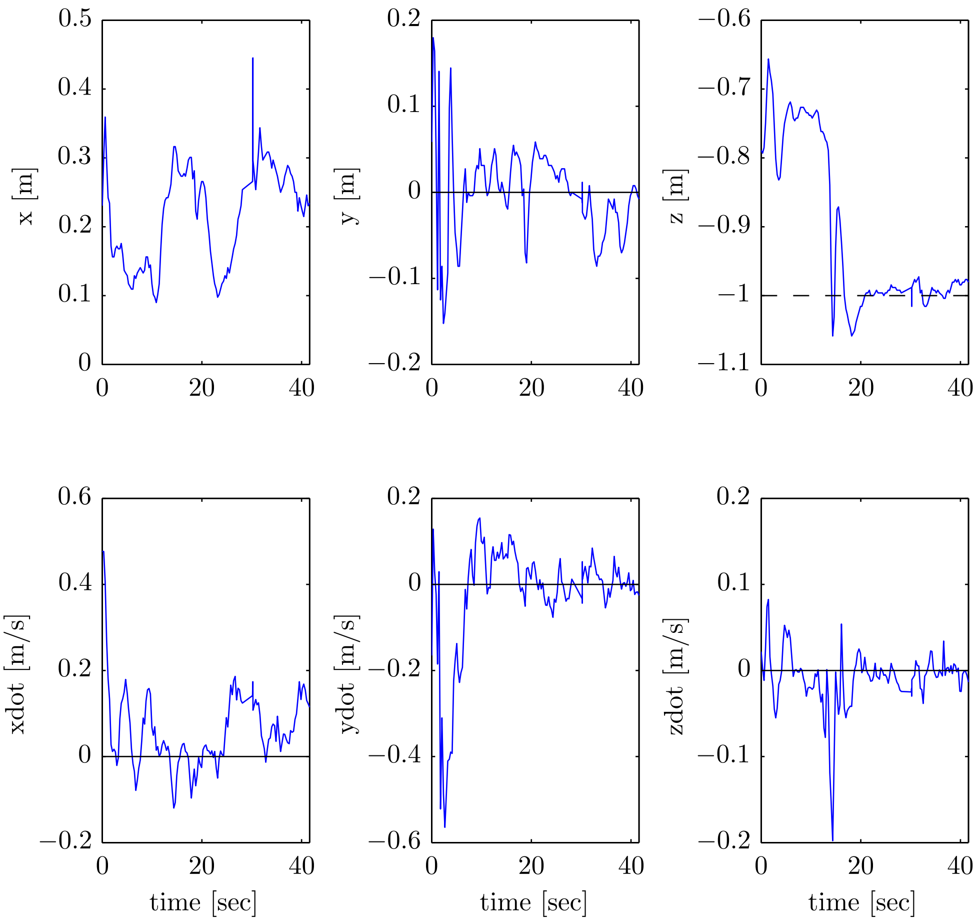


Figure 9-15: Position and velocity plots for free flight in real time

9-8 Discussion

Following the tuning procedure and flight tests, a critical analysis is now given of the results obtained for the real-time implementation. First, it should be stressed that although all concepts designed and tested in simulated flights are valid, some adjustments are required when implementing the algorithms on board a real quadrotor. The challenges predicted in Section 1-3, associated with this component of the project, were encountered during the implementation phase. Furthermore, other unforeseen difficulties arose during testing.

One of the first issues that should be discussed is the fact that delayed measurements, aggravated by other lags such as actuator delays, affect the closed-loop stability. Specifically, the vision system output has a delay of about 0.06 to 0.2 seconds. A simple analysis in the frequency domain is performed to explain this issue. Since the quadrotor's exact model is not available, a simple system is used to clarify this stability problem. Assume that the roll angle is already stabilized with an inner loop such that second-order linear dynamics are obtained between a reference command and the actual Euler angle (for the example, a natural frequency of 3 rad/s and a damping coefficient of 0.7 are considered to describe this inner loop). To control lateral velocity, an outer loop is designed with a single negative feedback proportional gain (for the example, the gain is taken as -0.2 rad$(m/s)^{-1}$). A bode plot of the open-loop system is shown in Figure 9-16 to analyze the closed-loop stability margins. From this plot, it is possible to determine that the stability delay margin is of 0.338 seconds. This example does not match exactly with reality, but serves as a perfect example to show how the camera delays affect stability properties. Note that velocity feedback is obtained through vision measurements.

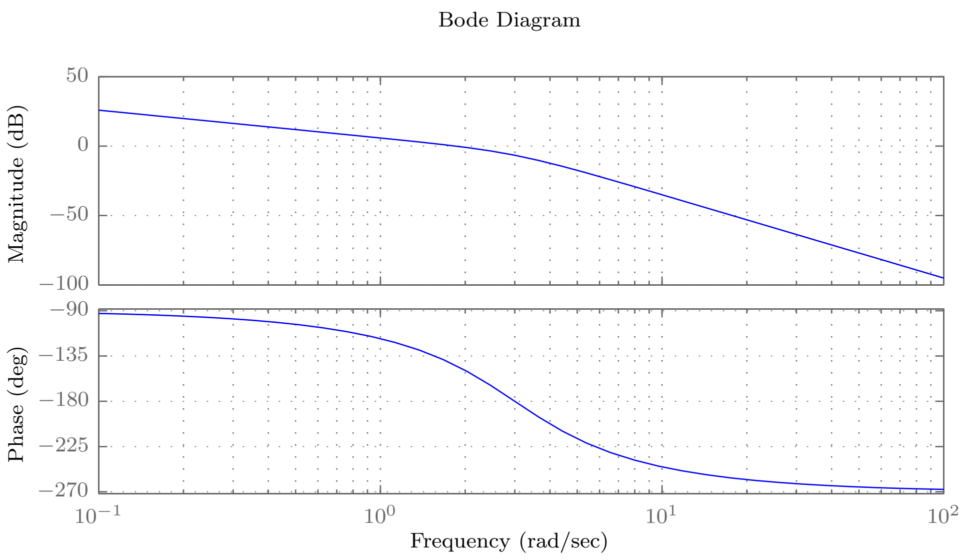


Figure 9-16: Bode plot for stability margins analysis

The position measurements obtained from the vision system are expressed in the body-fixed reference frame. To obtain the corresponding position expressed in the navigation frame, a coordinate-transformation method was proposed and validated offline. For this method, attitude measurements extracted from IMU data were used. As mentioned before, synchronization of camera with IMU is then required to perform the necessary rotations. The delayed vision measurements are obtained at an asynchronous update rate, which adds an extra challenge to the task. For this reason, the attitude compensation was not implemented in this study and the consequences are explained as follows. Consider as an example a rolling motion with no translation. If the quadrotor tilts to the right, the target will shift to the right side of the camera's image. Furthermore, if this motion is perceived as a translational motion (the apparent movement would be a translation to the left), then the control system would command a roll angle to the right. This will cause the quadrotor to rotate even more to the right, thus aggravating the stability condition. The strategy adopted to solve this issue was to tune the Kalman filter so that it would rely less upon camera information by increasing the R matrix as illustrated in Section 9-5. Despite the fact that better results were obtained after this modification, it should be noted that this attitude compensation is a major issue compromising the controller's performance.

The magnetometer measurement was another issue that posed some difficulties. Since the tests were run indoors, yaw measurements were corrupted by the local magnetic field. The main implications of this issue can be viewed from two perspectives. First, yaw attitude estimation is required for yaw motion control. In fact, for this specific application, it does not matter if the quadrotor is pointing in a certain direction or not. However, it was observed that yaw control is necessary to achieve an overall stable flight. Second, yaw-angle estimation is necessary for coordinate transformations. Note that the accelerations measured in the body-fixed reference frame must be rotated to the navigation frame. Thus, if the estimation is incorrect, integrations of the accelerations will then be performed in the wrong directions. Moreover, if the yaw measurements possess too much noise, this noise will affect position/velocity estimations when rotations are performed.

One of the major unforeseen issues in simulation was the convergence of the states corresponding to the accelerometer biases. The real biases change over time, therefore it is necessary to estimate these parameters for each flight test. Once the filter is activated along with the large initial state covariance matrices, the states converge very quickly. This situation represents when the quadrotor is not yet in flight, meaning that the attitude angles may differ from those when hovering, and consequently the bias estimations may be incorrect. After a certain period of time, the state covariance values decrease, meaning that once the quadrotor begins to fly the convergence of the biases is slower. A resetting procedure was implemented to guarantee fast convergence when required. This procedure was based on a re-initialization of the covariances to their initial large values (Lombaerts, 2010).

The accelerometer bias estimation affected not only position/velocity estimations, but also attitude estimation. In addition, the Euler angles determination had a strong effect on the overall stability and performance observed. In fact, it was noted that these parameters could not be estimated properly. A human pilot can adapt the control strategy to deal with large attitude offsets and fly the rotorcraft manually, but the same does not hold for the automated system unless necessary adjustments are performed.

The aforementioned issues were addressed to discuss the real-time implementation results obtained. It should be stressed that other minor issues were also encountered. To finalize this discussion, the use of the selected cheap and light sensors makes the control task more challenging. However, one of the main conclusions to be drawn from this study is that with proper software and hardware solutions, stable flight with satisfactory performance is possible as shown by the results obtained.

Chapter 10

Conclusion

A design solution for a vision-based automatic landing problem has been proposed in this thesis. Specifically, contributions have been made regarding modeling, nonlinear robust control, computer vision and state estimation. This chapter is a conclusion to the study, where final remarks are made including a critical analysis of the design choices and results obtained, a list of lessons learned and recommendations for future work.

10-1 Critical analysis of design choices and results obtained

10-1-1 Modeling

Modeling the highly nonlinear dynamics of a four-rotor helicopter is a challenge that has been faced by many research teams in the recent past. In this thesis a new contribution is given regarding calculation of moments of inertia, as an innovative experimental procedure was proposed and implemented. For the experiment, a two-axis motion simulator and a six-component force/torque sensor were used. For comparison, the inertia properties were also determined using two modeling techniques: point mass analysis and simple geometric shapes assumption. The results show that the experimental method is valid, as identical results were obtained when using the simple shapes assumption technique. Since the theoretical method is more cost-effective and less time-consuming, it can be concluded that modeling components as known geometric shapes is a reliable strategy for determining the moments of inertia of small UAVs. Furthermore, one of the most important conclusions is the fact that point mass analysis can result in erroneous calculations (deviation of 20-30% from the real value) leading to a significant model mismatch.

10-1-2 Computer vision

Computer vision has a been a topic of increasing interest among UAV-related research. This thesis continues the trend by proposing a vision-based algorithm for position estimation. The general idea was to use a target with known characteristics to calculate relative position between the quadrotor and the target. The target was designed as one or multiple red circles and the vision algorithm was tested with a Gumstix camera operating on board of a quadrotor. The results of this approach showed accurate position estimation.

10-1-3 Control and state estimation

This thesis proposes a design solution for vision-based automatic landing of a quadrotor UAV on a floating platform. The proposed solution includes GPS navigation to find the buoy and vision-based control for autonomous landing. In this study, the platform's motion was used in the control laws to enhance performance. What follows are conclusions drawn regarding control and state estimation.

Recent studies at DUT have shown that so-called incremental-based control laws present better robustness properties. In this thesis, a new contribution towards advanced certifiable nonlinear controllers is given. The technique used is based on incremental backstepping applied to quadrotor position (or velocity) control. The simulation results show that this control law is more robust to external disturbances and model uncertainties when compared to linear or even conventional backstepping controllers. Such robustness properties were demonstrated for modeling mismatches in mass and inertia properties, actuator dynamics and rotor gyro effect (seen as an external disturbance).

A modular approach for state estimation was also adopted. An AHRS algorithm based on complementary filtering was implemented for attitude estimation using IMU measurements, and a Kalman filter was developed for position/velocity estimation, providing integration of IMU with vision data. Additionally, a new contribution was given with an augmented Kalman filter formulation for estimation of a floating platform's vertical motion. Despite the fact that sensor noises were high and the camera had a low update rate, the algorithms allowed for proper state reconstruction. It is concluded that the modular approach adopted is a reasonable approach for the state estimation problem being studied.

One of the main conclusions to be drawn is the fact that the backstepping's stabilization properties are advantageous when using the controller in combination with the proposed state estimation algorithms. In fact, with the simplest form of linear control (SISO loops controlled by simple proportional gains), the closed loop was stable for the case of perfect state knowledge, but was unstable when using the state estimation algorithms. The same did not happen for the backstepping or incremental backstepping control laws; stable responses were obtained for both, regardless of whether full state knowledge was assumed or state estimation was used. It should be stressed that, in principle, the linear controller should work with state estimation. To achieve satisfactory performance, the linear controller would have to be further tuned. However, this step was considered irrelevant for this study.

Monte-Carlo simulation analysis was used to test the system for the autoland mission. Statistically, the results show that backstepping and incremental backstepping are suitable control laws to accomplish the goal with desired performance, and a risk analysis shows that the probability of crashing is residual. Furthermore, it is concluded that incremental backstepping presents better robustness properties overall.

10-1-4 Real-time implementation

Real-time implementation was performed to test the algorithms proposed, with the goal of investigating the feasibility of implementation using cheap and light sensors. In fact, the challenges faced posed major difficulties in the process of real-time implementation. While stability in velocity for hovering over a red blob was achieved, this final step was only possible after several modifications of the initial implementation. The main contributions given were: an explanation of the existing control loops already implemented in Paparazzi, adaptation of these to the hovering task and implementation of a Kalman filter for real-time integration of vision with IMU data. No modifications were made to the AHRS already implemented in Paparazzi, which presented erroneous pitch and roll estimations due to the accelerometer biases. Furthermore, since the tests were performed indoors, the yaw angle estimation was corrupted by the local magnetic field. The most important conclusion to draw from this study is the fact that unwanted effects and modeling details that are not easily predictable create a gap between simulated a real flight. Nevertheless, such issues sometimes require creative solutions, and stability can be achieved in the end.

10-2 Next steps in the development phase

The study presented in this dissertation is the initial step toward achieving the final objective of the autoland project stated in Section 1-1. During the development phase of this project at DUT, and specifically at the MAVlab, several steps are still required to meet the end goal. These are outlined below.

The real system should be tested outdoors in order to evaluate the controller and state estimation algorithms for external environment conditions. Note that magnetometer information should then be included in the loop.

The controller implemented in real-time is capable of hovering at a desired altitude. However, this was achieved only with velocity control, i.e., without horizontal position control. Adding this last component is then a step that must be taken. When the system is capable of performing longitudinal and lateral control, then flight tests should be performed to analyze whether the quadrotor can follow a moving target.

Finally, test landings are required to assess the design. Such tests should eventually be performed with a target excited with vertical oscillatory motion. When this step is accomplished, then the final goal is met, since the quadrotor could then perform vision-based automatic landing on a floating platform.

It should be stressed that these steps may require adaptations of the current algorithms implemented on board.

10-3 Lessons learned

Several useful lessons were learned during the course of this study. Some could be considered too obvious, but these points can easily be forgotten when performing research. A short list of lessons learned is shared below:

- Literature research should always be well-organized and structured from the beginning.
- Experimental procedures should be well-prepared. Nevertheless, the testing engineer should also be prepared to repeat measurements. When analyzing data, small errors in the experimental procedure can often be found.
- In simulation, it is not so difficult to achieve desired performance for the nominal case. The most challenging task is designing a controller that is robust for various combinations of external conditions. Such analysis should be carried out for the nominal model as for model uncertainties.
- Control and state estimation algorithms may work well independently; however, there is no guarantee that they work well together.
- Real-time implementation should be done step-by-step. Simple tests should always be performed before attempting to implement complex algorithms.

10-4 Recommendations for future work

A list of recommendations for future work is given below:

- Regarding the inertia experiment, it is possible to use the same concept to determine the inertia cross-coupling terms and experimentally obtain complete knowledge of the entire inertia matrix. Note that in this thesis, single axis rotations were used as they render linear relationships between torques and angular accelerations. However, this concept can be expanded since the motion simulator can rotate over both its axes at the same time. A wobbling motion would then permit estimation of the other inertia parameters. It would be interesting to analyze whether the assumption that the inertia matrix is diagonal is, in fact, valid.
- Modeling of the attitude representation was based on a Euler-angle parametrization. Alternatively, Quaternions, Rodriguez Parameters or Modified Rodriguez Parameters representation of the rotational kinematics could be tested. On a similar note, the overall model could also be improved. Specifically, the rotor dynamics were assumed and do not correspond to the real dynamics of the actuators used for real-time testing. Modeling the real dynamics would be useful for tuning the controller and enhancing performance, particularly for implementation of incremental-based control laws. Experimental procedures could thus be implemented to build an accurate model.
- Regarding computer vision, different algorithms can be investigated for relative position estimation. These include not only target features recognition, but also techniques based on optical flow.
- Low-pass filtering was used in this thesis for estimation of angular accelerations from gyro measurements. Despite the fact that satisfactory results were obtained, it is recommended that more efficient methods be explored. These might include Kalman filtering, predictive filters or even the introduction of angular acceleration sensors. This estimation problem must first be investigated independently; however, as was learned in this

study, the estimation method can only be accepted when the controller works well in its presence.

- The controller gains were designed using pole placement. As a recommendation, multi-objective optimization could be applied to obtain optimal gains as shown in (Looye, 2007). With this alternative approach, the Monte-Carlo results regarding assessment criteria could be improved.
- An AHRS could be designed for accurate attitude estimation in real time. Attitude is crucial not only for control purposes, but also for coordinate transformations. On a similar note, a method for IMU/camera synchronization could be developed. This would improve the controller performance significantly provided that the attitude estimation was accurate.
- On the one hand, for this thesis, detailed descriptions of the Paparazzi control loops were given from the designer perspective (as opposed to that of the user). On the other hand, following other research contributions from DUT, it was shown once again that incremental-based control laws are, in principle, more robust. With the real-time platform ready for testing with code explanations, the perfect conditions are gathered for testing incremental-based control laws in real time.

Appendix A

Quadrotor parameters

The quadrotor parameters used in this thesis are listed below. This appendix includes quadrotor model constants, actuator parameters and sensor characteristics used for simulation purposes, as well as values used for the theoretical moment of inertia calculations from Chapter 4.

A-1 Simulation parameters

The simulation parameters are given below. These parameters correspond to the model constants introduced in Chapter 3. Note that some of the variables mentioned in the chapter are not included in this appendix, as they are not needed for simulation purposes.

Table A-1: Quadrotor parameters used in the simulation model

parameter	symbol	value
mass (kg)	m	0.5
roll moment of inertia ($kg \cdot m^2$)	I_{xx}	0.0050
pitch moment of inertia ($kg \cdot m^2$)	I_{yy}	0.0050
yaw moment of inertia ($kg \cdot m^2$)	I_{zz}	0.0095
product moment of inertia ($kg \cdot m^2$)	I_{xy}	0.0000
product moment of inertia ($kg \cdot m^2$)	I_{xz}	0.0000
product moment of inertia ($kg \cdot m^2$)	I_{yz}	0.0000
inertia of all four rotors ($kg \cdot m^2$)	J_r	$6{\cdot}10^{-7}$
aerodynamic dimensionless drag coefficient (-)	C_D	0.5
aerodynamic area (m^2)	A_D	0.0025
dimensionless thrust coefficient (-)	C_T	0.08
dimensionless moment coefficient (-)	C_{MQ}	0.02
propeller radius (m)	R	0.1
arm length (from CM to center of the rotor) (m)	l	0.15

Table A-2: Actuator parameters used for the simulation model

parameter	symbol	value
drag factor $(N \cdot m \cdot s^2)$	d	$7.7 \cdot 10^{-7}$
motor time constant (sec)	τ	0.12
torque constant $(N \cdot m/A)$	k_m	$7.2 \cdot 10^{-4}$
motor internal resistance (Ω)	R_m	0.1
motor efficiency (-)	η	1
total rotor inertia $(kg \cdot m^2)$	J_t	$6 \cdot 10^{-7}$
reduction ratio (-)	r	4

The sensor characteristics are assumed as:

- accelerometers
  - IMU distance to CM $[m]$: $D_x = 0$, $D_y = 0$, $D_z = 0$
  - biases $[m/s^2]$: $\lambda_{A_x} = 0.01$, $\lambda_{A_y} = 0.01$, $\lambda_{A_z} = 0.01$
  - noise standard deviations $[m/s^2]$: $\sigma_{A_x} = 0.025$, $\sigma_{A_y} = 0.025$, $\sigma_{A_z} = 0.025$
  - update rate: 100 Hz
- gyroscopes
  - biases $[deg/s]$: $\lambda_p = 0.001$, $\lambda_q = 0.001$, $\lambda_r = 0.001$
  - noise standard deviations $[deg/s]$: $\sigma_p = 0.5$, $\sigma_q = 0.5$, $\sigma_r = 0.5$
  - update rate: 100 Hz
- magnetometer
  - bias of ψ_m $[deg/s]$: $\lambda_\psi = 0$
  - noise standard deviation of ψ_m $[deg/s]$: $\sigma_\psi = 0.1$
  - update rate: 100 Hz
- barometric altimeter
  - bias of h_m $[m]$: $\lambda_h = 0$
  - noise standard deviation of h_m $[m]$: $\sigma_h = 0.01$
  - update rate: 100 Hz
- vision system
  - biases $[m]$: $\lambda_{x_{cam}} = 0$, $\lambda_{y_{cam}} = 0$, $\lambda_{z_{cam}} = 0$
  - noise standard deviations $[m]$: $\sigma_{x_{cam}} = 0$, $\sigma_{y_{cam}} = 0$, $\sigma_{z_{cam}} = 0$
  - field of view [deg]: $FOV_x = 60$, $FOV_y = 80$
  - update rate: 10 Hz
  - camera time delay: 0.1 seconds

A-2 Parameters for theoretical moment of inertia determination

Given below are the values measured (or assumed) for the moments of inertia analytical calculations. For clarity, the geometric shapes assumed are shown in Figure A-1.

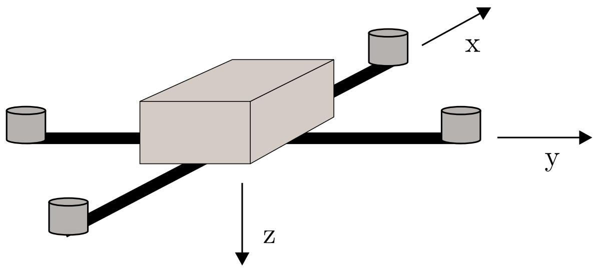


Figure A-1: Schematic representation of the quadrotor modeled with simple geometric shapes

For the point mass analysis, the following was assumed:

- $M_{motor} = 0.061\ Kg$
- $l_{arm} = 0.171\ m$

For simple geometric shape analysis, the following was assumed:

- Motors: four cylinders
  - $M_{cyl} = 0.061\ Kg$
  - $z_{cm,cyl} = 0.0275\ m$
  - $l_{cyl} = 0.035\ m$
  - $r_{cyl} = 0.014\ m$
- Arms: two slender rods
  - $M_{rod} = 0.0935\ Kg$
  - $z_{cm,rod} = 0.005\ m$
  - $l_{rod} = 0.342\ m$
- Core: one parallelepiped
  - $M_{core} = 0.158\ Kg$
  - $z_{cm,core} = 0.03\ m$
  - $a_{core} = 0.09\ m$
  - $b_{core} = 0.05\ m$
  - $h_{core} = 0.04u\ m$

Appendix B

Controllability and observability

Controllability and observability are two fundamental concepts in mathematical system theory, as they play an essential role in the design and control of systems. The theory presented here has been obtained from (Olsder & Woude, 2005).

Consider a Linear Time-Invariant (LTI) state-space model given in the form

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du \end{aligned} \tag{B-1}$$

with $x \in \mathbb{R}^{\ltimes}$, $x \in \mathbb{R}^{\gtrdot}$ and $y \in \mathbb{R}^{\lessdot}$. The matrices A, B, C and D are constant and have the appropriate dimensions. Note that a nonlinear system can generally be represented by its linearized form around an operating point. For brevity, system (B-1) is referred to as system (A,B,C,D).

Definition B.1
The system (A,B,C,D) is called **controllable** if for any two states $x_0, x_1 \in \mathbb{R}^n$, a finite time $t_1 > 0$ and an admissible input function u exist such that $x(t_1, x_0, u) = x_1$.

Hence, a system is controllable if an arbitrary state $x_1 \in \mathbb{R}^n$ can be reached starting from an arbitrary state $x_0 \in \mathbb{R}^n$, in finite time t_1, by means of the application of a suitable admissible input function u.

Controllability is characterized in terms of the matrices A and B. Let the so-called controllability matrix be defined as

$$R = \begin{bmatrix} A & AB & A^2B & \cdots A^{n-1}B \end{bmatrix} \tag{B-2}$$

which is an $n \times nm$ matrix. Furthermore, $im\{R\}$ denotes the controllable subspace. The following lemma is useful in the development of conditions for the controllability of a system:

Lemma B.2
$im\{A^kB\} \subset im\{R\}$, for all $k \geq 0$

Theorem B.3
The following statements are equivalent:

- The system (A,B,C,D) is controllable.
- R has full rank.
- $im\{R\} = \mathbb{R}^n$.

The observability concept is now presented.

Definition B.4
The system (A,B,C,D) is **observable** if a finite time $t_1 > 0$ exists such that for each admissible input function u, it follows from $y(t, x_0, u) = y(t, x_1, u)$ for all $t \in [0, t_1]$, that $x_0 = x_1$.

A system is then called observable if the initial state x_0 can be constructed from the knowledge of u and y on the interval $[0, t_1]$ for some finite time $t_1 > 0$.

Observability can be completely characterized by the matrices A and C. Let the $np \times n$ matrix W, called the observability matrix, be defined as

$$W = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix} \tag{B-3}$$

Lemma B.5
Let the vector $x \in \mathbb{R}^n$ be such that $Cx = CAx = \cdots = CA^{n-1}x = 0$. Then $CA^k x = 0$ for all $k \geq 0$.

Let the subspace $ker\{W\}$ be called the non-observable subspace.

Theorem B.6
The following statements are equivalent.

- System (A,B,C,D) is observable.
- W has rank n.
- $ker\{W\} = 0$.

All proofs for the lemmas and theorems enunciated are given in Chapter 4 of (Olsder & Woude, 2005).

Appendix C

Attitude complementary filter

Attitude determination is done by means of a complementary filter. The estimations of pitch and roll angles are then obtained using an algorithm as depicted in Figure C-1.

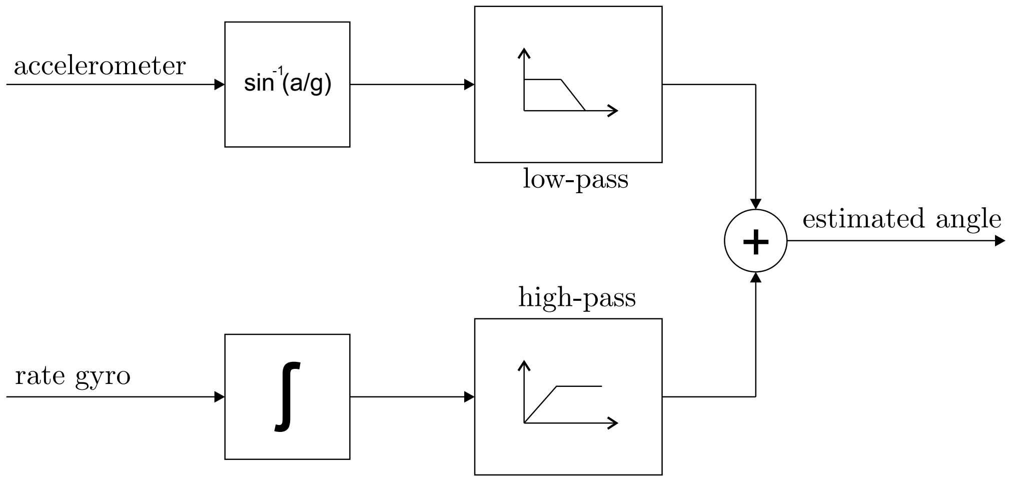


Figure C-1: Complementary filter for attitude angles (pitch and roll) estimation

This approach permits combining the low-frequency components of angle calculation through accelerometer measurements and high-frequency components of integrated rate gyro data. Note that due to noise contents and especially sensor biases, pure integration of angular rate measurements is not possible. The optimal approach would be to use a Kalman filter for estimation of the gyro biases. However, this complementary filter approach was selected for this thesis. The complementary filter is simpler since it involves less computations (Higgins, 1975).

The integrated gyro measurements are filtered by the low-pass filter

$$G_l(s) = \frac{1}{\tau_{cf}s + 1} \tag{C-1}$$

and the data obtained from accelerometers is filtered by the high-pass filter

$$G_h(s) = 1 - G_l(s) = 1 - \frac{1}{\tau_{cf}s + 1} = \frac{\tau_{cf}s}{\tau_{cf}s + 1} \tag{C-2}$$

The yaw angle is estimated in a similar way. For the high-frequency path, integration of gyro measurements is performed. However, note that instead of using accelerometer information, the signal for the low-frequency path is obtained from magnetometer measurements.

The result of this combination of high with low frequency contents is then an estimation of the Euler angles ϕ, θ and ψ. This information can then be readily used in other estimation blocks where these parameters are required.

Appendix D

Command filters

The filters used in the control laws as reference command generators are presented in this appendix. The first-order filter was also used for estimation of the angular accelerations required for the incremental form of backstepping.

D-1 First order

The transfer function of a first-order low pass filter is given by the following equation

$$\frac{X_c(s)}{X_c^0(s)} = \frac{1}{\tau s + 1} \tag{D-1}$$

which, in the time domain, can be represented as

$$\dot{x} = \frac{1}{\tau}(x_c^0 - x_c) \tag{D-2}$$

The implementation of this filter is depicted in Figure D-1. Note that magnitude and rate limiters can be introduced as shown.

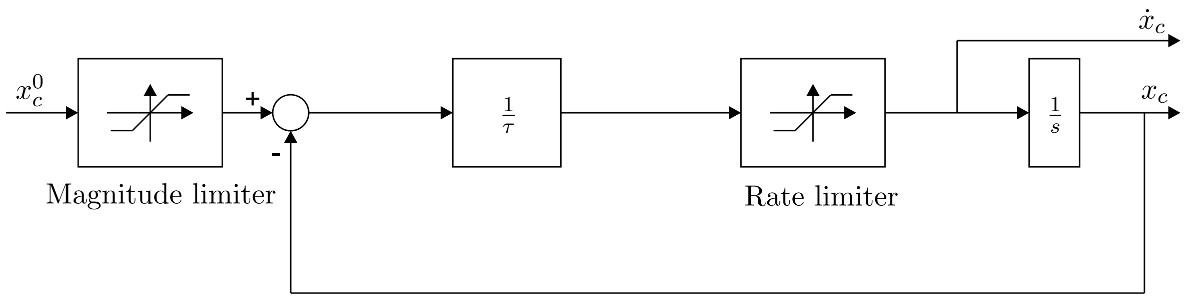


Figure D-1: First-order low pass filter with magnitude and rate limiters

D-2 Second order

The transfer function of a first-order low pass filter is given by the following equation

$$\frac{X_c(s)}{X_c^0(s)} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{D-3}$$

which, in the time domain, can be represented as

$$\ddot{x} = -2\zeta\omega_n\dot{x} + \omega_n^2(x_c^0 - x_c) \tag{D-4}$$

The implementation of this filter is depicted in Figure D-2. Note that magnitude and rate limiters can be introduced as shown.

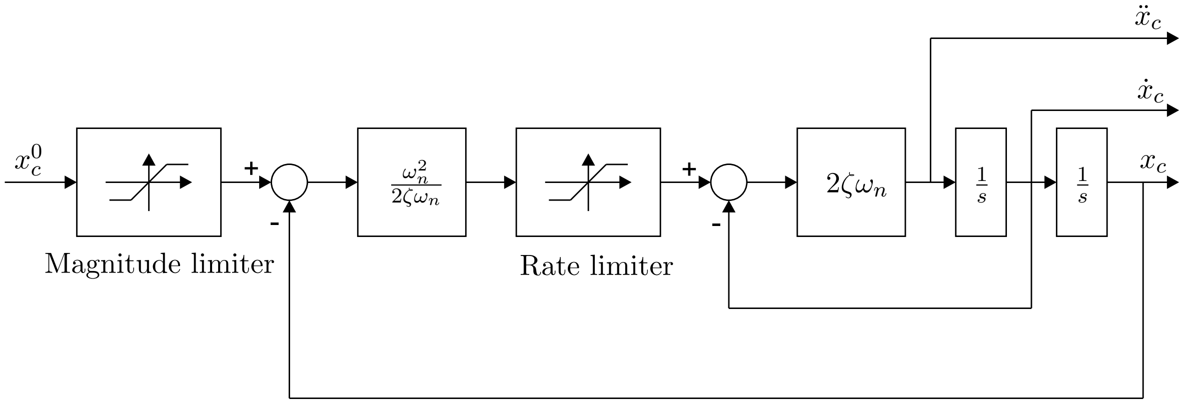


Figure D-2: Second-order low pass filter with magnitude and rate limiters

Appendix E

Controller gains used for simulations

The controller gains used for simulation purposes are given in Table E-1 (all controllers used the same gains). As explained in Chapter 7, these gains were obtained using a pole-placement technique by selecting adequate closed-loop time constants, which are given as well.

Table E-1: Controller gains used for the simulation model

control variable	loop time constant τ [sec]	gain K_p	gain K_i
	rate loops		
p	0.05	0.1	0
q	0.05	0.1	0
r	0.5	0.019	0
	attitude loops		
ϕ	0.2	5	0
θ	0.2	5	0
ψ	4	0.25	0
	velocity loops		
u	0.5	-0.2039	0
v	0.5	0.2039	0
w	0.2	-2.5	0
	position loops		
x	1	1	0
y	1	1	0
z	1	1	0

Appendix F

Additional statistical data from Monte-Carlo simulations

Additional statistical data from the Monte-Carlo simulations performed in Chapter 8 is presented on the following pages. Specifically, examples of histograms of external conditions are shown, as well as empirical cumulative distribution functions of the assessment criteria parameters with fitted normal probability density functions.

The fitted lines show how well the empirical distributions can be approximated by normal distributions. As expected, a better fit is obtained when using a larger sample size.

The distributions are only shown for the cases in which no model uncertainties are considered, for both perfect state knowledge and state estimator in the loop. For the former, 200-sample Monte-Carlo simulations were performed, while for the latter, 1000 were used.

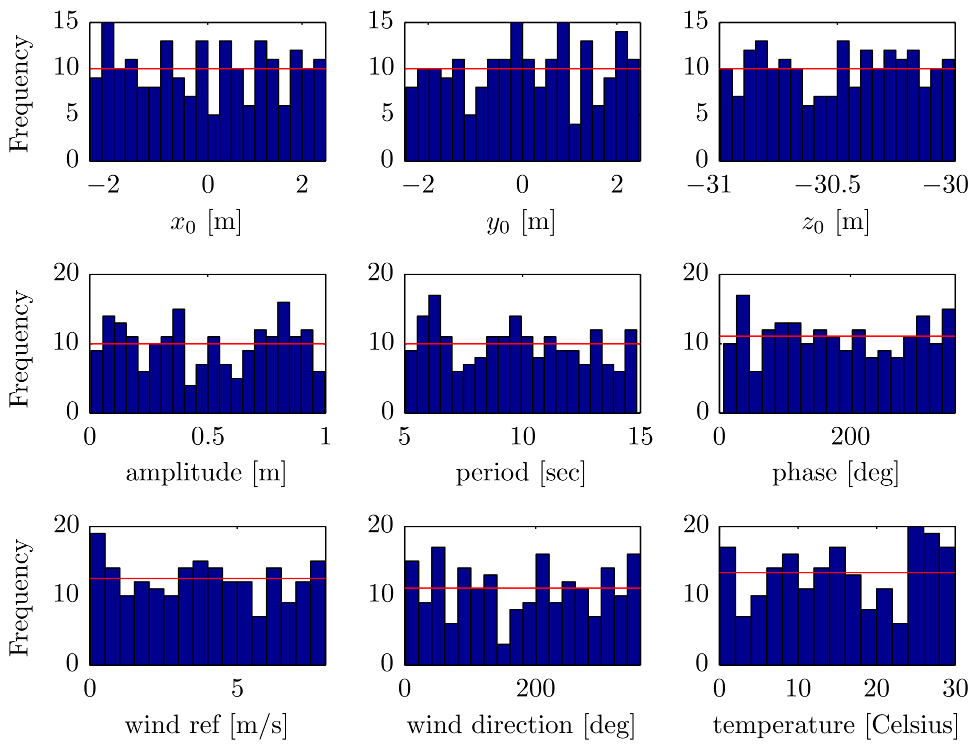


Figure F-1: Histograms of external conditions for 200 samples

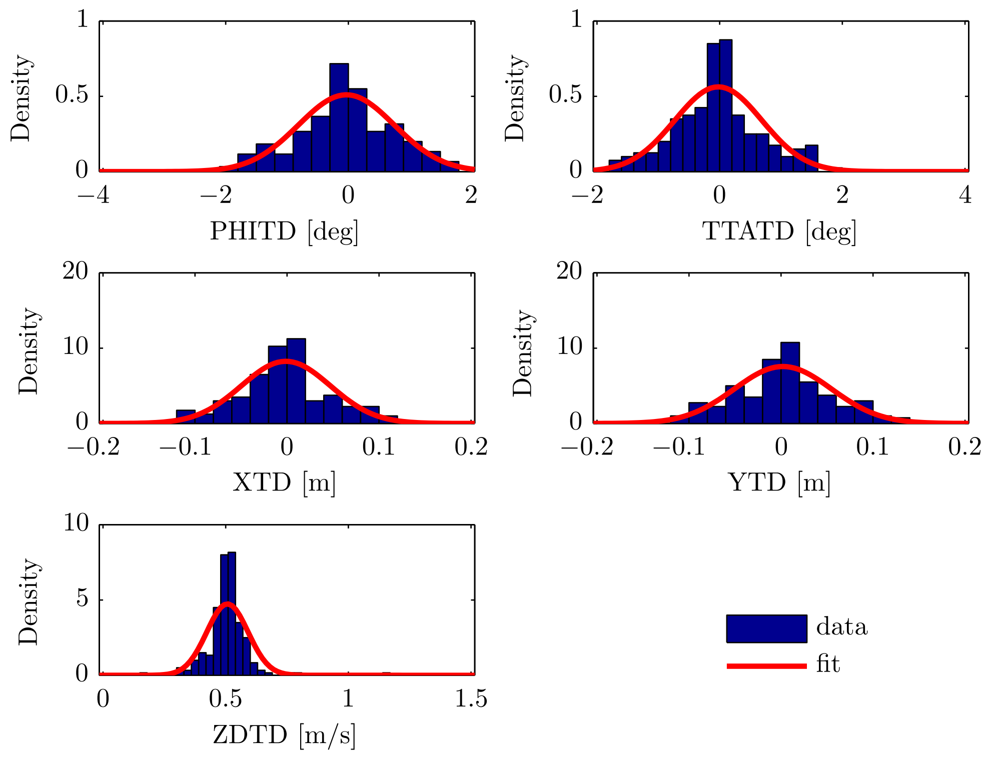


Figure F-2: Statistical distribution of assessment criteria parameters for linear controller with perfect state knowledge (200 samples)

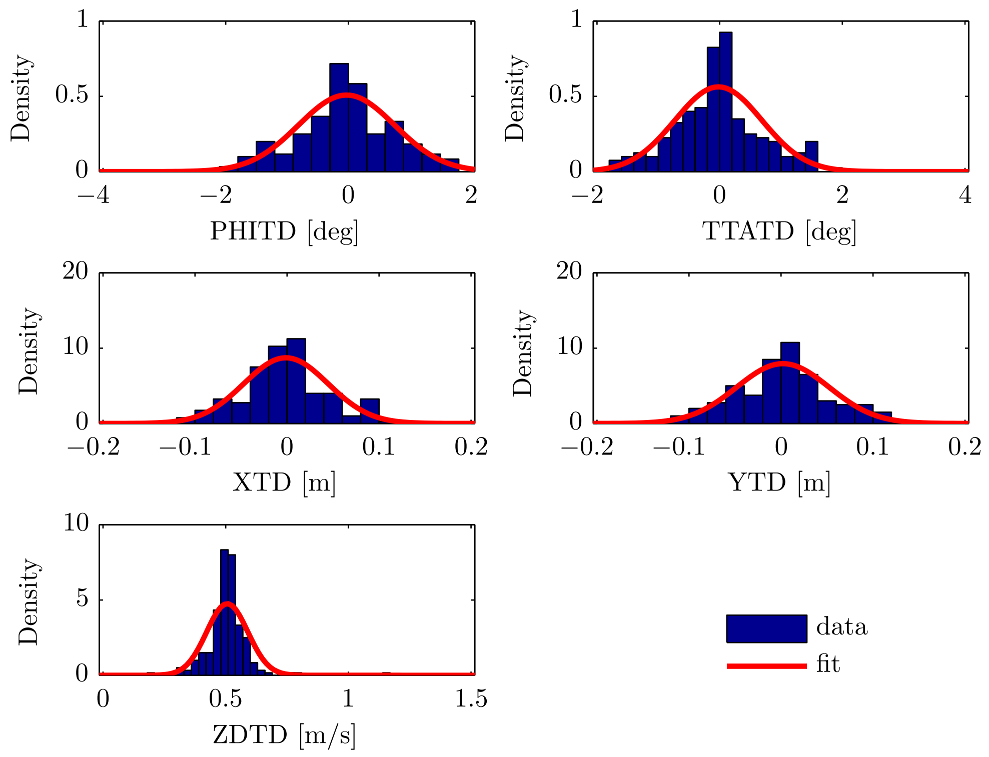


Figure F-3: Statistical distribution of assessment criteria parameters for backstepping controller with perfect state knowledge (200 samples)

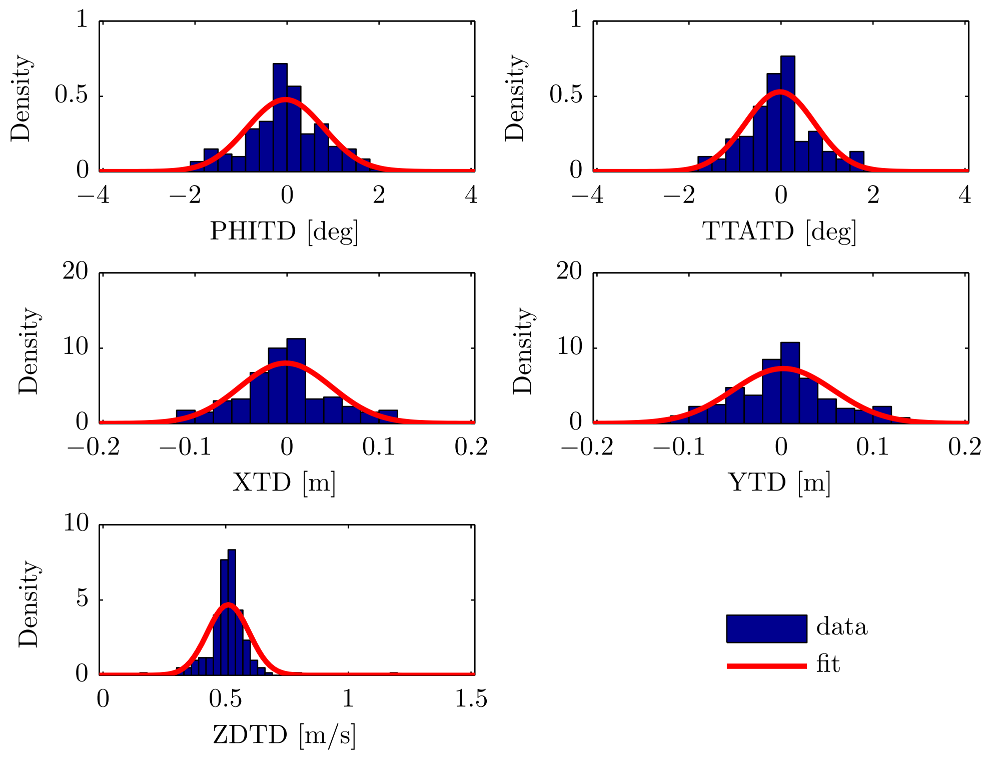


Figure F-4: Statistical distribution of assessment criteria parameters for incremental backstepping controller with perfect state knowledge (200 samples)

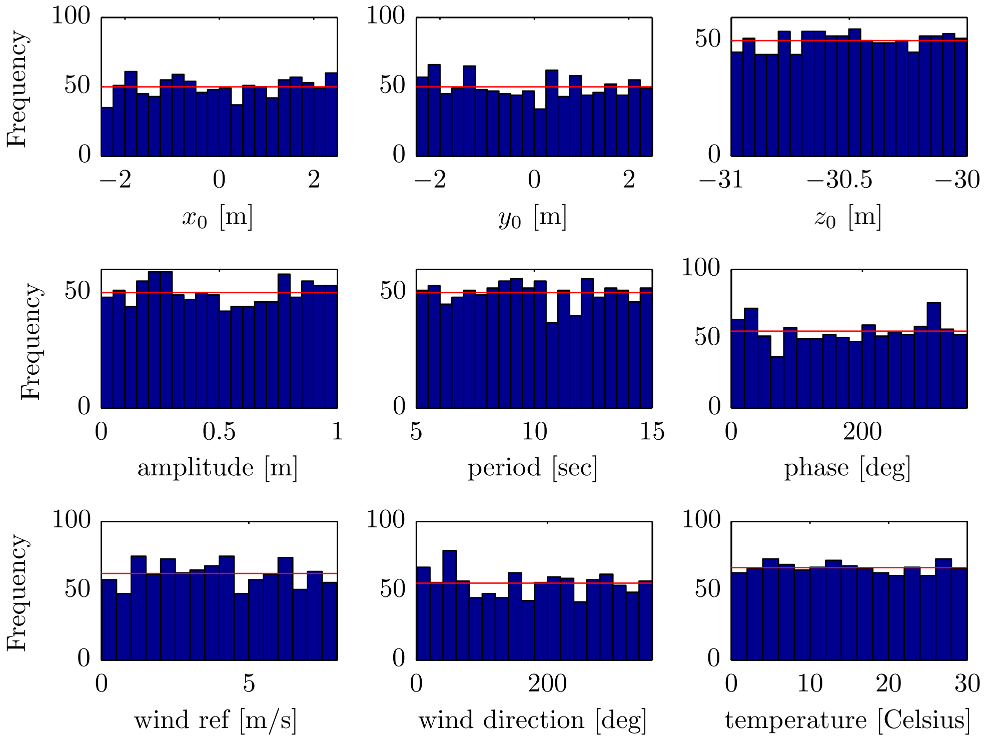


Figure F-5: Histograms of external conditions for 1000 samples

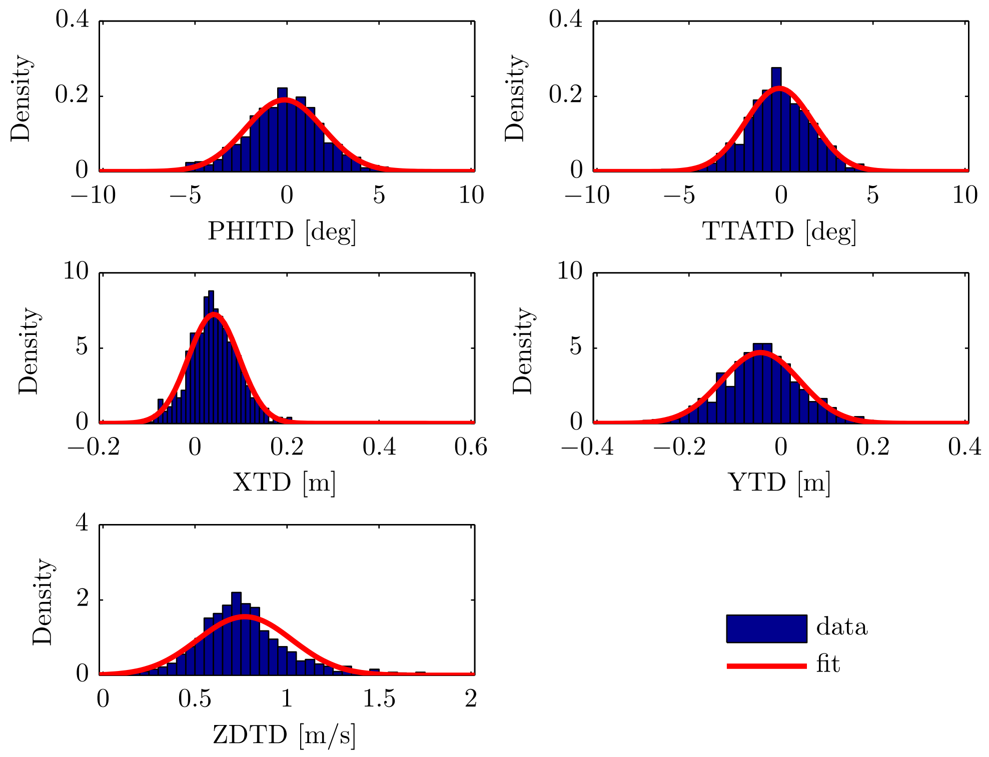


Figure F-6: Statistical distribution of assessment criteria parameters for backstepping controller with state estimator in the loop (1000 samples)

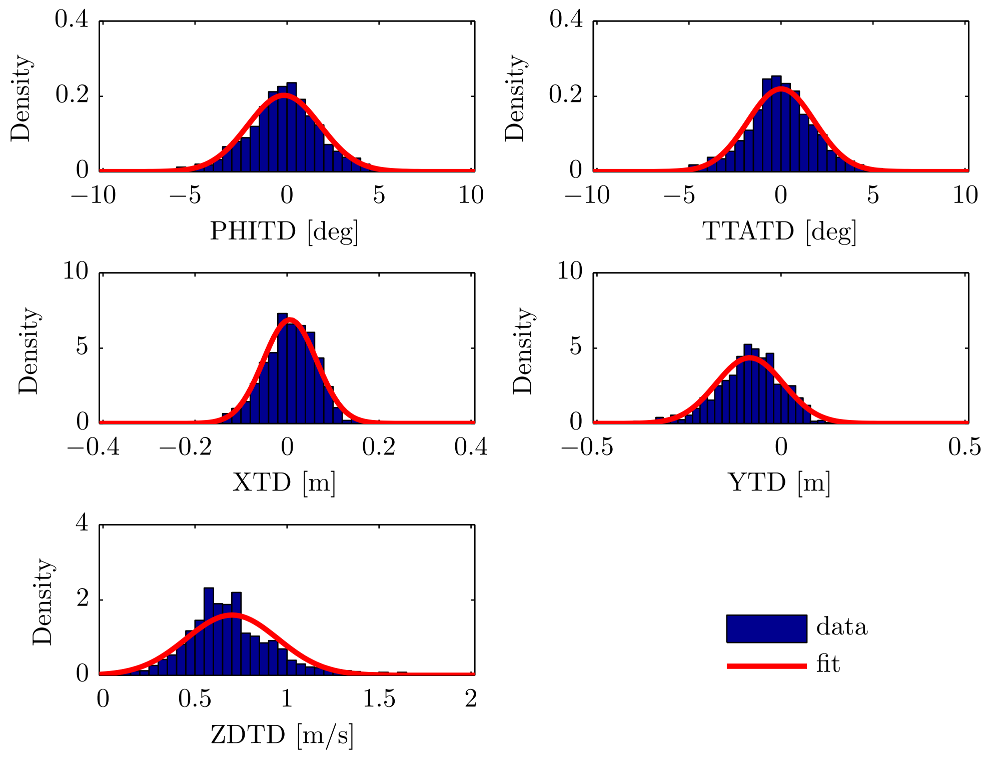


Figure F-7: Statistical distribution of assessment criteria parameters for incremental backstepping controller with state estimator in the loop (1000 samples)

Bibliography

Acquatella, P. (2011). *Robust Nonlinear Spacecraft Attitude Control, an Incremental Backstepping Approach.* Unpublished M.Sc. Thesis, Faculty of Aerospace Engineering, Delft University of Technology.

Almeida, R., Urgueira, A., & Maia, N. (2007). Identification of rigid body properties from vibration measurements. *Journal of Sound and Vibration*, *299*, 884-899.

Altuğ, E., Ostrowski, J. P., & Taylor, C. J. (2005, May). Control of a quadrotor helicopter using dual camera visual feedback. *Int. J. Rob. Res.*, *24*, 329–341.

Amidi, O. (1996). *An Autonomous Vision-Guided Helicopter.* PhD thesis, Robotics Institute, Carnegie Mellon University.

Ampelmann. (2009, October). *Ampelmann, making offshore access as easy as crossing the street.* Available from `http://www.ampelmann.nl/index.php?id=149`

Artstein, Z. (1983). Stabilization with relaxed controls. *Nonlinear Analysis*, *7*(11), 1163–1173.

Barron, J. L., Fleet, D. J., & Beauchemin, S. S. (1994). Performance of optical flow techniques. *International Journal of Computer Vision*, *12*, 43–77.

Berkelaar, W., & Oonk, A. (2009). *Design of a laser Attitude and Altitude System for quadrotor UAV control.* Unpublished M.Sc. Thesis, Faculty of Aerospace Engineering, Delft University of Technology.

Bodson, M., & Athans, M. (1985). Multivariable control of VTOL aircraft for shipboard landing. In *1985 AIAA Guidance and Control Conference.*

Bouabdallah, S. (2007). *Design and control of quadrotors with application to autonomous flying.* PhD thesis, EPFL.

Broome, D. R., & Hall, M. S. (1998). Application of ship motion prediction I. *Transactions - Institute of Marine Engineers*, *110*(1), 77-93.

Bussamra, F., Vilchez, C., & Santos, J. (2009, September). Experimental determination of unmanned aircarft inertial properites. In *2009 brazilian symposium on aerospace eng. and applications, 3rd cta-dlr workshop on data analysis and flight control.*

Castillo, P., Lozano, R., & Dzul, A. E. (2005). *Modelling and control of mini-flying machines.* Advances in Industrial Control, Springer-Verlag, London.

Chatterji, G., Menon, P., & Sridhar, B. (1997). GPS/machine vision navigation system for aircraft. *Aerospace and Electronic Systems, IEEE Transactions on*, *33*(3), 1012 - 1025.

Cheeseman, I. C., & Bennett, W. E. (1957). *The Effect of the Ground on a Helicopter Rotor in Forward Flight* (Technical Report). Aeronautical Research Council.

Chen, H., & Zhang, S. (2008). Robust dynamic inversion flight control law design. In *The 2nd International Symposium on Systems and Control in Aerospace and Astronautics* (pp. 1–6).

Chen, J., & Pinz, A. (2004). Structure and motion by fusion of inertial and vision-based tracking. In *Proceedings of the 28th OAGM/AAPR Conference* (Vol. 179, p. 55-62).

Chroust, S., & Vincze, M. (2004). Fusion of vision and inertial data for motion and structure estimation. *J. Field Robotics*, *21*(2), 73-83.

Chu, Q. (2009). *Modern flight test technologies and system identification* (AE4-3494 Course Notes). Faculty of Aerospace Engineering, Delft University of Technology.

Dalamagkidis, K., Ioannou, S., Valavanis, K., & Stefanakos, E. (2006). A Mobile Landing Platform for Miniature Vertical Take-Off and Landing Vehicles. In *Control and Automation, 2006. MED '06. 14th Mediterranean Conference on* (p. 1 - 6).

Da Lio, M., Doria, A., & Lot, R. (1999, March). A spatial mechanism for the measurements of the inertia tensor: theory and experimental results. *Journal of dynamic systems, measurements, and control*, *121*, 111-121.

Dalzell, J. F. (1965). A note on short-time prediction of ship-motions. *Journal of Ship Research*, *9*(2), 118-121.

Daquan, T., & Hongyue, Z. (2007). Vision Based Navigation Algorithm for Autonomic Landing of UAV without Heading & Attitude Sensors. In *Proceedings of the 2007 Third International IEEE Conference on Signal-Image Technologies and Internet-Based System* (pp. 972–978). Washington, DC, USA.

Dobrokhodov, V., Kaminer, I., Jones, K., & Ghabcheloo, R. (2006). Vision-based tracking and motion estimation for moving targets using small UAVs. In *American Control Conference, 2006.*

Dunbabin, M., Corke, P., & Buskey, G. (2004). Low-cost vision-based AUV guidance system for reef navigation. In *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on* (Vol. 1, pp. 7–12).

Esmailifar, S. M., & Saghafi, F. (2009). Autonomous Unmanned Helicopter Landing System Design for Safe Touchdown on 6DOF Moving Platform. In *Autonomic and Autonomous Systems, 2009. ICAS '09. Fifth International Conference on* (p. 245-250).

Farrell, J., Polycarpou, M., & Sharma, M. (2004). On-Line Approximation Based Control of Uncertain Nonlinear Systems with Magnitude , Rate and Bandwidth Constraints on the States and Actuators. In *Proceeding of the 2004 American Control Conference.*

Farrell, J., Polycarpou, M., Sharma, M., & Dong, W. (2009). Command filtered backstepping. *IEEE Transactions on Automatic Control*, *54*, 1391–1395.

Fay, G. (2001). *Derivation of the aerodynamic forces for the mesicopter simulation* (Technical Report). EPFL.

Ferrier, B., Bradley, D., & Blackwell, J. (2001). Development and Test Evaluation of Ship Motion Based Helicopter Recovery Monitoring System. In *American Helicopter Society International Annual Forum (57th).* Washington, D.C..

Fleischmann, D. S. (2000). *Method and System for Predicting Ship Motion or the Like to Assist in Helicopter Landing.* U. S. P. Office. USA, Lockheed Martin Corporation. 6064924.

Ford, T., & Boloye, M. (2010). *Helicopter ship board landing system.* United State Patent Application Publication. US 2010/0228408A1.

Franklin, P., & Emami-Naeini. (2006). *Feedback control of Dynamic Systems.* Pearson Prentice Hall.

Genta, G., & Delprete, C. (1994). Some considerations on the experimental determination of moments of inertia. *Meccanica*, *29*, 125-141.

Herisse, B., Hamel, T., Mahony, R. E., & Russotto, F. (2012). Landing a VTOL Unmanned Aerial Vehicle on a Moving Platform Using Optical Flow. *Robotics, IEEE Transactions on*, *28*(1), 77–89.

Higgins, W. (1975). A Comparison of Complementary and Kalman Filtering. *Ieee Transactions On Aerospace And Electronic Systems*, *AES-11*(3), 321–325.

Hintze, J. (2004). *Autonomous landing of a rotary unmanned aerial vehicle in a non-cooperative environment using machine vision.* Unpublished M.Sc. Thesis, Brigham Young University.

Hoffmann, G. M., Huang, H., Waslander, S. L., & Tomlin, C. J. (2007). Quadrotor Helicopter Flight Dynamics and Control: Theory and Experiment. In *Proc. of the AIAA Guidance, Navigation, and Control Conference.*

Hsu, S., Meindl, E., & Gilhousen, D. (1994). Determining the power-law wind-profile exponent under near-neutral stability conditions at sea. *Journal of Applied Meteorology*, *33*(6), 757–765.

Hua, M. D. (2009). *Contributions to automatic control of aerial vehicles.* PhD thesis, Universit de Nice-Sophia Antipolis.

Hubbard, D., Morse, B. S., Theodore, C., Tischler, M., & McLain, T. W. (2007). Performance Evaluation of Vision-based Navigation and Landing on a Rotorcraft Unmanned Aerial Vehicle. In *Applications of Computer Vision, 2007. WACV '07. IEEE Workshop on.*

Jarvis, R. A. (1983). A Perspective on Range Finding Techniques for Computer Vision. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *PAMI-5*(2), 122 - 139.

Kalman, R. E. (1960). A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME–Journal of Basic Engineering*, *82*(Series D), 35–45.

Kaplan, P. (1969). A study of prediction techniques for aircraft carrier motions at sea. *Journal of Hydronautics*, *3*(3), 121-131.

Kendoul, F., Nonami, K., Fantoni, I., & Lozano, R. (2009). An adaptive vision-based autopilot for mini flying machines guidance, navigation and control. *Autonomous Robots*, *27*.

Khalil, H. (2002). *Nonlinear Systems* (3rd ed.). Prentice Hall.

Kokotović, P., & Arcak, M. (2001). Constructive nonlinear control: a historical perspective. *Automatica*, *37*(5), 637–662.

Krstič, M., Kanellakopoulos, I., & Kokotović, P. (1995). *Nonlinear and Adaptive Control Design.* John Wiley and Sons, Inc.

Lainiotis, D., Charalampous, C., Giannakopoulos, P., & Katsikas, S. (1992). Real Time Ship Motion Estimation. In *Oceans '92. 'Mastering the Oceans Through Technology'. Proceedings.*

Lainiotis, D., Plataniotis, K., Penon, D., & Charalampous, C. (1993). Neural network application to ship position estimation. *Oceans '93*.

Lamontia, A. (1982). On the determination and use of residual flexibilities, inertia restraints, and rigid-body modes. In *Proceedings of the international modal analysis conference exhibit.*

Leishman, J. G. (2002). *Principles of helicopter aerodynamics.* Cambridge University Press.

Lombaerts, T. (2010). *Fault Tolerant Flight control, A Physical Model Approach.* PhD thesis, Faculty of Aerospace Engineering, Delft University of Technology.

Looye, G. (2007). *An Integrated Approach to Aircraft Modelling and Flight Control Law Design.* PhD thesis, Faculty of Aerospace Engineering, Delft University of Technology.

Lopes, H. (2011). *Attitude Determination of Highly Dynamic Fixed-wind UAVs, A MEMS-AHRS/GPS Integration.* Unpublished M.Sc. Thesis, Faculty of Aerospace Engineering, Delft University of Technology.

Lozano, R. (2010). *Unmanned Aerial Vehicles Embedded Control.* Wiley.

Ma, Y., Kosecka, J., & Sastry, S. (1997). Vision guided navigation for a nonholonomic mobile robot. In *Decision and Control, 1997., Proceedings of the 36th IEEE Conference on.*

Marconi, L., Isidori, A., & Serrani, A. (2002). Autonomous vertical landing on an oscillating platform: an internal-model based approach. *Automatica*, *38*(1), 21–32.

McKerrow, P. (2004). Modelling the Draganflyer four-rotor helicopter. In *Proceedings of the 2004 IEEE International Conference on Robotics and Automation.*

McMuldroch, C. G., Stein, G., & Athans, M. (1979). VTOL control for shipboard landing in high sea states. In *Decision and Control including the Symposium on Adaptive Processes, 1979 18th IEEE Conference on* (Vol. 18, p. 626-629).

Mellinger, D., Michael, N., & Kumar, V. (2010). Trajectory Generation and Control for Precise Aggressive Maneuvers with Quadrotors. In *Proceedings of the International Symposium on Experimental Robotics.*

Mellinger, D., Shomin, M., & Kumar, V. (2010). Control of Quadrotors for Robust Perching and Landing. In *Proceedings of the International Powered Lift Conference.*

Meriem, J., & Kraige, L. (1998). *Engineering mechanics* (4th ed., Vol. 2: Dynamics). John Wiley and Sons, Inc.

Miller, M. (1930, October). *An accurate method of measuring the moments of inertia of airplanes* (Tech. Rep.). Washington, National Advisory Committee for Aeronautics, Technical Notes, No. 351.

Mucchi, E., Fiorati, S., Di Gregorio, R., & Dalpiaz, G. (2011). Determining the rigid-body inertia properties of cumbersome systems: comparison of techniques in time and frequency domain. *Experimental Techniques*, *35*(3), 36–43.

Mulder, J. A., Staveren, W. H. van, Vaart, J. C. van der, & Weerdt, E. de. (2006). *Flight Dynamics, Lecture Notes AE3-302* (Technical Report). Delft University of Technology.

Mulder, M. (2007). *Atmospheric Flight Dynamics* (AE4-304 Course Notes). Faculty of Aerospace Engineering, Delft University of Technology.

Naidu, D. S., & Calise, A. J. (2001). Singular Perturbations and Time Scales in Guidance and Control of Aerospace Systems: A Survey. *Journal of Guidance, Control and Dynamics*, *24*(6), 1057–1078.

Nordberg, K., Doherty, P., Farnebäck, G., Forssén, P., Granlund, G., Moe, A., et al. (2002, October). Vision for a UAV helicopter. In *Proceedings of IROS'02, workshop on aerial robotics.* Lausanne, Switzerland.

Oh, S., Pathak, K., Agrawal, S., Pota, H., & Garratt, M. (2006). Approaches for a tether-guided landing of an autonomous helicopter. *Robotics, IEEE Transactions on*, *22*(3), 536 - 544.

Olsder, G., & Woude, J. (2005). *Mathematical Systems Theory* (3rd ed.). VSSD.

Padfield, G. D. (2007). *Helicopter flight dynamics: The theory and application of flying qualities and simulation modelling.* Blackwell Publishing.

Pegram, J., & Anemaat, W. (2000, May). Preliminary estimation of airplane moments of inertia using cad solid modeling. In *General avaition technology conference and exposition.*

Podhorodeski, R. P., & Sobejko, P. (2005, October). A project in the determination of the moment of inertia. *International Journal of Mechanical Engineering Education.*

Pounds, P., Mahony, R., & Corke, P. (2006). Modelling and Control of a Quad-Rotor Robot. In *Proceedings of the Australasian Conference on Robotics and Automation* (p. 27-29).

Previati, G., Mastinu, G., Gobbi, M., Piccardi, C., Bolzoni, L., & Rinaldi, S. (2004, November). A new test rig for measuring the inertia properties of vehicles and their subsystems. In *Asme 2004 international mechanical engineering congress and exposition (imece2004).*

Riola, J., Diaz, J., & Giron-Sierra, J. (2011). The prediction of calm opportunities for landing on a ship: Aspects of the problem. *OCEANS, 2011 IEEE - Spain.*

Saghafi, F., & Esmailifar, S. M. (2007). Automatic landing of small helicopters on 4DOF moving platforms. *JAST*, *4*(4), 37-46.

Saripalli, S. (2009). Vision-based Autonomous Landing of an Helicopter on a Moving Target. In *AIAA Guidance, Navigation and Control Conference.*

Saripalli, S., Montgomery, J., & Sukhatme, G. (2003). Visual-Guided Landing of an Unmanned Aerial Vehicle. *In IEEE Transactions on Robotics and Automation*, *19*(3), 371-381.

Seddon, J., & Newman, S. (1990). *Basic Helicopter Aerodynamics.* BSP Professional Books.

Sidar, M., & Doolin, B. (1983). On the feasibility of real-time prediction of aircraft carrier motion at sea. *Automatic Control, IEEE Transactions on*, *28*(3), 350-356.

Sieberling, S. (2009). *Design of a Robust Generic Flight Control System using Incremental Nonlinear Dynamic Inversion.* Unpublished M.Sc. Thesis, Faculty of Aerospace Engineering, Delft University of Technology.

Sieberling, S., Chu, Q. P., & Mulder, J. A. (2010). Robust Flight Control Using Incremental Nonlinear Dynamic Inversion and Angular Acceleration Prediction. *Journal of Guidance Control and Dynamics*, *33*(6), 1732-1742.

Simplicio, P. (2011). *Helicopter Nonlinear Flight Control, An Acceleration Measurements-based Approach using Incremental Nonlinear Dynamic Inversion.* Unpublished M.Sc. Thesis, Faculty of Aerospace Engineering, Delft University of Technology.

Smith, P. N., Sridhar, B., & Hussien, B. (1992). Vision-based range estimation using helicopter flight data. In *Computer Vision and Pattern Recognition, 1992. Proceedings CVPR '92., 1992 IEEE Computer Society Conference on* (p. 202 - 208).

Sonneveldt, L. (2010). *Adaptive Backstepping Flight Control for Modern Fighter Aircraft.* PhD thesis, Faculty of Aerospace Engineering, Delft University of Technology.

Storozhenko, V. (2003). A technique for identification of the principal central axis of inertia in an inhomogeneous rigid body. *International Applied Mechanics*, *39*, 1464-1472.

Storvik, M. (2003). *Guidance System for Automatic Approach to a Ship.* Unpublished M.Sc. Thesis, Norwegian University of Science and Technology.

Török, J. (2000). *Analytical mechanics: with an introduction to dynamical systems.* John Wiley and Sons, Inc.

Triantafyllou, M., & Bodson, M. (1982). Real time prediction of marine vessel motions using kalman filtering techniques. In *Proceedgins of the 14th Annual Offshore Technology Conference.* Houston, Texas.

Triantafyllou, M., Bodson, M., & Athans, M. (1983). Real time estimation of ship motions using kalman filtering techniques. *Oceanic Engineering, IEEE Journal of*, *8*(1), 9-20.

Trigo, G. (2011). *A Robust and Adaptive Nonlinear Attitude Control of a Spacecraft, A Comparison of Backstepping-based Designs.* Unpublished M.Sc. Thesis, Faculty of Aerospace Engineering, Delft University of Technology.

Vukić, Z., Kuljača, L., Donlagić, D., & Tešnjak, S. (2003). *Nonlinear Control Systems.* Marcel Dekker.

Wang, J., Garratt, M., Lambert, A., Wang, J., Han, S., & Sinclair, D. (2008). Integration of GPS/INS/Vision Sensors to Navigate Unmanned Aerial Vehicles. In *ISPRS08* (p. B1: 963 ff).

Wedershoven, J. A. (2010). *Ananlysis of Nonlinear Dynamic Inversion Based Control Law Designs.* Unpublished M.Sc. Thesis, Faculty of Aerospace Engineering, Delft University of Technology.

Weiss, I., & DeVries, T. (1977). Ship motion measurement filter design. *Oceanic Engineering, IEEE Journal of*, *2*(4), 325-330.

Wenzel, K. E., Masselli, A., & Zell, A. (2010). Automatic take off, tracking and landing of a miniature uav on a moving carrier vehicle. *Journal of Intelligent and Robotic Systems*, *61*.

Wierema, M. (2008). *Design, implementation and flight test of indoor navigation and control system for a quadrotor UAV.* Unpublished M.Sc. Thesis, Faculty of Aerospace Engineering, Delft University of Technology.

Wu, A. D., Johnson, E. N., & Proctor, A. A. (2005). Vision-Aided Inertial Navigation for Flight Control. In *Journal of Aerospace Computing, Information, and Communication.*

Xu, C., Qiu, L., Liu, M., Kong, B., & Ge, Y. (2006). Stereo Vision based Relative Pose and Motion Estimation for Unmanned Helicopter Landing. In *Information Acquisition, 2006 IEEE International Conference on* (p. 31 - 36).

Yang, X., Pota, H., Garratt, M., & Ugrinovskii, V. (2008). Prediction of vertical motions for landing operations of UAVs. In *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on.*

Yumori, I. (1981). Real time prediction of ship response to ocean waves using time series analysis. In *Oceans 81.*