NTNU – Trondheim
Norwegian University of
Science and Technology

Design of Attitude Control System of a Double CubeSat

Gaute Bråthen

Master of Science in Engineering Cybernetics
Submission date: Januar 2013
Supervisor: Jan Tommy Gravdahl, ITK

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Design of Attitude Control System of a Double CubeSat

Master thesis

by

Gaute Bråthen

Supervisor:
Prof. J.T. Gravdahl
Co-supervisor:
Roger Birkeland, NTNU IET

Norwegian University of Science and Technology,
Institute of Engineering Cybernetics
Trondheim, January 25, 2013

NTNU
Norges teknisk-naturvitenskapelige universitet

Fakultet for informasjonsteknologi, matematikk og elektroteknikk
Institutt for teknisk kybernetikk

MSc thesis assignment

Name of the candidate: Gaute Bråthen

Subject: Engineering Cybernetics

Title: Attitude control for the Norwegian student satellite NUTS

Background

The NUTS (NTNU Test Satellite) project was started in September 2010. The project is part of the Norwegian student satellite program run by NAROM (Norwegian Centre for Space-related Education). The projects goal is to design, manufacture and launch a double CubeSat by 2014.

Assignment:

1) Evaluate different methods for numerical differentiation in order to find the time derivative of the measured magnetic field, $\dot{B}$. Implement the chosen method on the microcontroller.
2) Simulate detumbling of the spacecraft using feedback from $\dot{B}$. Add noise to the measurements. The noise should be consistent with what you measure using the real magnetometer. Use the IGRF-model in the simulations. Compare the results with the detumbling done in the thesis by Z.Tudor (2011)
3) Investigate if $\dot{B}$-detumbling is able to bring the angular velocity of the spacecraft to such low levels that LQ-control can be employed for attitude control.
4) Conclude on what detumble and attitude control algorithms to use on the real spacecraft
5) Design and specify the torque coils to be used on the spacecraft with respect to number of windings, resistance and physical size.

To be handed in by:

Trondheim,

Jan Tommy Gravdahl
Professor, supervisor

Abstract

The NTNU Test Satellite, NUTS, is a satellite being build in a student CubeSat project at the Norwegian University of Science and Technology. The project was started in September 2010 as a part of the Norwegian student satellite program run by NAROM (Norwegian Centre for Space-related Education). The NUTS project goals are to design, manufacture and launch a double CubeSat by 2014. As payload an IR-camera observing waves in the air-glow layer is planned, as well as a short-range RF experiment. The satellite will fly two transceivers in the amateur radio bands. Final year master students from several departments are the main contributors in the project and most of the system components are designed and built by students.

As the main payload is an IR-camera and one of the main goals once in orbit is to take pictures of the gravity waves in the atmosphere, a reliable Attitude Determination and Control System (ADCS) is important. In order to take reliable pictures, an absolute requirement is that the camera does not miss the Earth, and more accurately be able to point towards the Earth in such a matter that unwanted disturbances, such as stars and other light sources in space, is not in the camera's field of view.

In this thesis, a realistic simulation environment have been created in order to assess and design an Attitude Control System (ACS) to be implemented on the satellite. Magnetorquers as the satellite's actuators have been designed. Further, a detumbling control algorithm using an estimated derivative of the geomagnetic field measured with a magnetometer, is shown to be working satisfactory, even when being subjected to heavy environmental disturbances. Linear and nonlinear reference control algorithms subjected to realistic disturbances in the space environment have been tested and compared as well. Comparisons between a proportional derivative (PD) controller and a linear quadratic regulator (LQR) are presented, and the nonlinear controller proved to give best results and pointing control good to take pictures of the Earth atmposphere is achieved.

Sammendrag

Norges teknisk- naturvitenskapelige universitet (NTNU) startet høsten 2010 studentsatellittprosjektet NTNU Test Satellite, NUTS. Prosjektet er en del av det nasjonale ANSAT-programmet styrt av NAROM (Nasjonalt senter for romrelatert opplæring) og Norsk romsenter, som andre norske høyskoler og universiteter også er en del av. Målene til NUTS-prosjektet er å designe, bygge og sende opp en dobbel CubeSat i 2014. Hoved payload'en er et infrarødt kamer som skal ta bilde av jordas amtosfære, og en annen payload er et trådløs databus-eksperiment. Satellitten vil ha to radiobånd for kontakt med bakken. Masterstudenter fra mange forskjellige disipliner er med på prosjektet.

Siden hovedpayload'en er et IR-kamer, å få satellitten til å peke mot jorda er viktig, og et robust attitude kontrollsystem er å foretrekke.

Det er vist i denne rapporten at man kan detumble stelliten ved bruk av estimering av derivatet av en måling av det lokale geomagnetiske felt, også når store forstyrrelse er tilstede. Sammenligninger mellom en ulineær og en lineær kontroller har blitt gjort, der den ullineære ser ut til å gi best resultater. Peking av satellitten mot jorda for å ta bilder av dens atmosfære ser ut til å være oppnådd.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 The NUTS - NTNU Test Satellite project

The NUTS project is a student satellite project at the Norwegian university of science and technology (NTNU), which aims to design, build and launch a satellite of the CubeSat standard by the year of 2014. The project is part of a national satellite program, ANSAT, where also the Narvik University College and University of Oslo are bulding a satellite each, called HinCube [1] and CubeSTAR [2] respectively. Apart from building three student satellites, the ANSAT program also "has the intention to stimulate cooperation between educational institutions in Norway and with industry, and also to give the students experience in team work and hands-on training. The program also aims to increase the interest for science and technology in lower educational levels to secure future recruitment to higher education" [3]. The program was initiated in 2006 and started up in 2007 as a succession of the NCUBE-1 and NCUBE-2 projects. The NUTS project were initiated in the fall of 2010, while students started working on it in the spring of 2011. The project spans over a wide area of different discipilines, and since the start-up there's been more than 30 students involved, and with even more to come. The satellite will be built following the CubeSat standard developed by the California Polytechnic State University and Stanford University [4], as a way for schools and universities to be able to build and launch affordable satellites consisting of mostly commercial off-the-shelf components. Our satellite will be a double CubeSat measuring $10 \times 10 \times 20$ cm^3 and have a maximum weight of 2.66 kg. The main payload will be an infrared (IR) camera which will take pictures of gravity waves in the Earth's atmosphere, and as a secondary payload there's an internal wireless databus to be tested in space. As for the attitude control, magnetic torquer is the chosen actuator. A project statement can be found at [5], and for updated information see the project's webpage [6].

1.2 Small satellites

After the World War II, when both USSR and USA acquired advanced missile technology from Germany, the very first signs of the coming space age took place. On October 4th 1957 the space age begun with the launch of Sputnik 1 by the USSR. This ignited the space-race between the USSR and the United States of America, and over the years with ever more decreasing size of electrical components into micro- and nanoelectronics have made it possible to make smaller and smaller satellites at a low cost.

1.3 Small satellite magnetic attitude control systems overview

With the increasing possibilities of having advanced payloads in small satellites, also nano satellites, comes the need of an accurate attitude control system. There are numbers of different approaches one can use for ACS, both passive and active ones.
Passive control schemes includes gravity gradient boom and/or permanent magnet; the former mainly supporting pointing stability for active controlled satellites which already are nadir pointing, and the latter consists of a ferromagnet in order to make the sattelite's dipole vector follow the geomagnetic field lines. With the permanent magnetic solution our nadir pointing requirement would be compromised, and since the gravity gradient boom only supports other active controllers, none of them were chosen to be used in our satellite.
Active schemes are for example liquid or gas thrusters, moment wheels and/or electromagnetic torquers. As the CubeSat standard doesn't allow for liquid thrusters, only gas thrusters are an alternative. However, with the strict space and weight restrictions of our satellite and maybe more important, for practical mechanical reasons thrusters were not considered an option for our solution. Moment wheels also take up some space and weight in a satellite, and one would also need electromagnetic torquers (magnetorquers) for moment dumping of the wheels. For satellites at altitudes defined as a low Earth orbit (LEO), i.e. the locus from the Earth's surface up to an altitude of 2,000 km, and with a near polar orbit the geomagnetic field is strong enough and is varying enough to be able to use magnetorquers as control actuators. Magnetorquers are also relatively cheap and easy to manufacture. Hence for the NUTS, which will be in a LEO, an active scheme of magnetic actuators is suitable and an overview of this field will now be presented.

Since the beginning of the space age the use of magnetorquers for attitude control have been studied White et al. [7]. However, only since the early 90's the use of magnetic control for *small* satellites gained attention; in Wen & Kreutz-Delgado [8] a general framwork for the attitude tracking control problem for a rigid body was developed, and in Pittelkau [9] it was shown that roll/yaw dynamics with mag-

netic control are/can be seen as periodically time varying and an optimal periodic control law was developed. Nonlinear sliding mode control for satellites is the topic of the paper by Chen & Lo [10]. In Steyn [11], a rule-based fuzzy controller is presented in a low-earth-orbit small satellite attitude control system. Three-axis satellite control based only on magnetic control for a low earth orbit satellite with optional moments of inertia is shown in Wisniewski & Blanke [12] and is further revised in Wisniewski & Blanke [13]. In Wisniewski [14] a nonlinear sliding mode controller is developed for a small near polar LEO satellite without appendages. In Wang et al. [15], detumbling and attitude acquisition is made possible without e.g. gravity boom or reaction wheels, only magnetorquers. A control system with the satellite dynamics represented with an inner and outer loop is used, where the outer loop is regarded as a regulation system controlled by a virtual input produced by the inner. Magnetic attitude control using periodic linear systems theory were investigated in Wisniewski [16], based on the steady state and transient solutions of the Riccati equation, and it was shown that a LEO near polar with magnetic actuators can be described with a set of periodic differential equations. The periodicy being exploited is that of the geomagnetic field surrounding the Earth, which over 24 hours can be said to be periodic. Psiaki [17] also finds periodic linear controllers using asymptotic linear quadratic regulator (LQR) techniques, which also includes integral action and saturation logic. In Lovera and Astolfi [18] an optimal periodic controller for asymptotic stabilization is described and in addition, techniques for disturbance attenuation for periodic systems subject to periodic disturbances is described. For a good overview and survey of the small satellite magnetic attitude control field, the paper by Silani and lovera [19] is worth the read.

1.4 Previous work

The work done on attitude control in the NCUBE-1 and NCUBE-2 projects, with contributions from Kristiansen [20], Fauske [21] Busterud [22], Øverby [23] and Narverud et al. [24], and the thesis by Soglo [25], the makes up the basis of most of the work on attitude determination and control systems (ADCS) in the NUTS projects. Other Master theses worth mentioning are Makovec [26] of the HokieSat project, Andresen et al. [27] from the AAUSAT-II project, Giesselmann [28] of the Compass-1 project, and the two from the CubeSTAR project by Stray [29] and Rensel [30].

Since this project is very interdisciplenary, the work of the all members of the group is crucial to make the system as a whole to work altogether. The different subsystems of the satellite project are:

- On-board computer (OBC)
- Backplane

- Electrical power system (EPS)
- Tracking, telemetry & command (TT&C)
- Payload (IR Camera and wireless bus)
- Mechanical
- Attitude determination and control system (ADCS)
- Ground station (not part of the space segment/satellite)

Work on the OBC have been done by Volstad [31] with a focus on hardware design, and Holmstrøm [32] with software design focus. de Bruyn [33] developed the a system for the backplane amd EPS with further work done by Jacobsen [34]. An antenna system design were developed by Marholm [35], and uplink security and access control were the topics in Visockas [36]. Systems for taking, handling and sending infrared pictures with the IR camera have been done by Bakken and Rønning [37], while work on the wireless bus experiment is shown in Meland [38]. Rokstad [39] started analysis of the possibilities of making the frame structure of the satellite in composite material. Ground station work have been done by Stenhaug [40].

Earlier work on the ADCS part of the project have been done by Jenssen and Yabar [41], Rinnan [42] and Holberg [43], which all investigates and suggests solutions of the attitude determination problem. In [41] the Extended Kalman Filter (EKF) for nonlinear estimation of the satellite's attitude is investigated, and also a new method called the Extended QUaternion EStimation, abbreviated EQUEST, for nonlinear attitude estimation were found. This method utilizes vectorized and non-vectorized measurements for the attitude estimation. For satellites with magnetic actuation as ours, the EQUEST method have some advantages over the EKF-method. In [42] the EQUEST method were further developed, and it now makes use of quaternion products as opposed to earlier, which used subtractions between estimated and measured quaternions. This latter method does not result in a new quaternion, but the former with quaternion products does. As a result, it has a higher accuracy, but with the cost of more complexity and it's computationally more demanding.
Work done in the ADCS part of the project with a focuse on attitude control, are Tudor [44], and also Holberg [45] and Bråthen [46]. In [44] work on detumbling the satellite and reference control using a PD-like nonlinear controller were shown to give good results. using these control algorithms, three perpendicular magnetic coils (also called magnetic torquers/magnetorquers) were found to give sufficient actuation for reference control of a spacecraft. Also, a 8-bit, 16 MHz microcontroller were found to have the sufficient processing power to compute the geomagnetic field using the International Geomagnetic Reference Field (IGRF) model and simultaneously maintaining correct coil actuation. In [45] an optimal linear

quadratic regulator (LQR) for reference control were developed, while in [46] different nonlinear controllers were investigated, and a sliding mode and a PD controller similar to the one developed by Tudor [44] proved to work satisfactory.

On-going work: There are on-going work on all the subsystems except the

1.5 Thesis outline

Chapter 2 introduces some mathematical principles used throughout the thesis. Filters for magnetometer measurements are explained. It also includes Lyapunov stability theorems, and the concepts of observability, controllability and stabilizability is introduced.

Chapter 3 presents and explains the models used for simulating the dynamics of a rigid body in orbit around the Earth, including models for the space environment; the geomagnetic IGRF model, and gravitational, aerodynamic, solar radiation and internal dipole disturbance torques. Introduces the magnetic actuator model and derives the linearized satellite model.

Chapter 4 presents the magnetorquer design for the satellite.

Chapter 5 presents the control strategy and the controllers to be investigated: the so-called B-dot detumbling controller, the nonlinear PD-like reference controller and the optimal LQR controller. Some coments on the stability of the three control algorithms are provided.

Chapter 6 shows the results of the filters for the magnetometer measurements, and of the three control algorithms investigated. Some comparisons between the different filters and the different control algorithms subjected to dofferent conditions are made.

Chapter 7 presents the final conclusions and remarks on further work to be done.

Appendix A contains the Matlab code and scripts used to simulate the controllers and filters.

Appendix B presents some mechanical drawings of the frame used for the magnetorquers, provided by Christian E. Nomme and Kim Sandvik.

Chapter 2

Theory

Working with navigation from an engineering point of view, is in many ways the study and effort to try to determine and control the change of dynamics of an object/body, such as marine vessels, airplanes and spacecrafts or satellites. Different coordinate frames are used to model the dynamics of the different objects, and in order to work with and analyse these, one need mathematical tools to describe how these objects relate to each other.
Space navigation with respect to a small satellite is often restricted to change its attitude, which is one of this thesis' objectives, and is described in chapter 3. In this chapter, the different reference frames are defined and established, as well as the most commonly used parametrizations for attitude representation. For a more in-depth study of these topics, see (Gravdahl and Egeland, 2002) in general and (Hughes, 1986) for space attitude specifically.

2.1 Coordinate frames

First step in order to analyse and design an ACS control system for a satellite is to define the different coordinate frames to work with. A satellite's attitude is maybe most conveniently described as a deviation relative to a chosen reference coordinate frame. For this reason, and because the topic of attitude, reference frames and rotations can be a bit mind-boggling, it is important to have clear definitions of the different frames.

2.1.1 Earth centered inertial frame

The Earth centered inertial (ECI) frame, denoted $\mathcal{F}_i$, has its origin in the center of the Earth and is an inertial, non-rotational frame in which Newtons's laws of motion apply. It is defined by the the unit vectors $\vec{x}_i$, $\vec{y}_i$ and $\vec{z}_i$. The $\vec{z}_i$ axis points at a 90° angle relative to the Earth's equatorial plane where it coincides with the Earths rotational axis and continues through the celestial North Pole. The $\vec{x}_i$ axis points in the vernal equinox vector direction, which is the vector pointing from the centre of the Sun to the centre of the Earth at the vernal equinox. The vernal equinox is a time of the year when the Earths orbital plane as it rotates around the Sun coincides with the equatorial plane, i.e. the center of the Sun lies in the same plane as the Earths equator. Finally, the $\vec{y}_i$ axis completes the three axis orthonormal frame according to the right-hand rule.

2.1.2 Earth centered Earth fixed frame

The Earth centered Earth fixed frame (ECEF) frame, denoted $\mathcal{F}_e$, has as the ECI frame its origin at the Earth's centre, but it's different in the way that it rotates with the Earth. The frame is spanned out by the unit vectors $\vec{x}_e$, $\vec{y}_e$ and $\vec{z}_e$. Again, the $\vec{z}_e$ points along the Earth's rotational axis, while $\vec{x}_e$ points in the direction of 0° latitude and 0° longitude. This means the coordinate frame rotates around the $\vec{z}_e$ axis with the angular rate of $\omega_e = 7.2921 \cdot 10^{-5}$ rad/s. The $\vec{y}_e$ completes the right-handed orthonormal frame.

2.1.3 Orbit fixed frame

The orbit fixed frame, denoted $\mathcal{F}_o$, follows the orbit trajectory and has its origin at the satellite's center of mass. It is defined by the unit vectors $\vec{x}_o$, $\vec{y}_o$ and $\vec{z}_o$. The $\vec{z}_o$ axis points towards the center of the Earth, and the $\vec{x}_o$ axis points in the orbit normal direction, which is parallel to the orbital angular momentum vector. Again, the $\vec{y}_o$ axis completes the right-handed orthonormal frame.

2.1.4 Body fixed frame

The body fixed frame, denoted $\mathcal{F}_b$, is a moving coordinate system which also has it's origin at the satellite's center of mass. However, unlike the orbit frame it is fixed to the satellite, and so it rotates and moves with the satellite. It's defined

by the unit vectors $\vec{x}_b, \vec{b}$ and $\vec{z}_b$, and for simplifying further attitude comptutations it's commmon to let these vectors coincide with the satellite's axes of moments of inertia. We will define it by letting the $\vec{y}_b$ axis be that of the maximum inertia, the $\vec{z}_b$ axis be the minimum inertia, and the $\vec{x}_b$ axis completes the right-hand orthonormal system. This choice follows from the result of the stability analysis done in section 5.2. Further, the $\vec{x}_b$ axis is called the roll axis, $\vec{y}_b$ is the pitch axis and $\vec{z}_b$ is the yaw axis.

2.2 The Rotation Matrix

Our main mission in orbit is to take pictures of the Earth's gravity field waves in the upper atmosphere with an IR camera, and hence we want to orient the satellite's camera in the nadir direction towards the Earth. In order to compare and manipulate the satellite's attitude relative to a reference frame, one need to or express the attitude vector in several different frames. One *transforms* the attitude vector from one frame to another, and the tool used for doing this is the *rotation matrix* $\mathbf{R}$.
Another interpretation is that it can also describe the attitude of a rigid body.
The definitions and notation below follows mostly that of Gravdahl and Egeland [47].

Let $\mathcal{F}_a$ and $\mathcal{F}_c$ denote two coordinate frames, both $\in \mathbb{R}^3$. The frames are defined by the orthogonal unit vectors $\vec{a}_1$, $\vec{a}_2$, $\vec{a}_3$ and $\vec{c}_1$, $\vec{c}_2$, $\vec{c}_3$ respectively. Further, a vector $\vec{v}$ may be described in both systems $\mathcal{F}_a$ and $\mathcal{F}_c$:

$$\vec{v} = \sum_{i=1}^{3} v_i^a \vec{a}_i \quad \text{and} \quad \vec{v} = \sum_{i=1}^{3} v_i^c \vec{c}_i. \tag{2.1}$$

where $v_i^a = \vec{v} \cdot \vec{a}_i$ and $v_i^c = \vec{v} \cdot \vec{c}_i$ are the coordinates of the vector $\vec{v}$ in the frames $\mathcal{F}_a$ and $\mathcal{F}_c$, respectively. The column vectors in the two frames are then given by (for likt Tudor?)

$$\mathbf{v}^a = \begin{bmatrix} v_1^a \\ v_2^a \\ v_3^a \end{bmatrix} \quad \text{and} \quad \mathbf{v}^c = \begin{bmatrix} v_1^c \\ v_2^c \\ v_3^c \end{bmatrix}. \tag{2.2}$$

The relation between the coordinate vector represented in the two different coordinate frames can be found by combining the equations (2.1) and (2.2):

$$v_i^a = \vec{v} \cdot \vec{a}_i = (v_1^c \vec{c}_1 + v_2^c \vec{c}_2 + v_3^c \vec{c}_3) \cdot \vec{a}_i = \sum_{j=1}^{3} v_j \left(\vec{a}_i \cdot \vec{c}_j\right). \tag{2.3}$$

Further, one can now transform the coordinate vector from frame $\mathcal{F}_c$ to frame $\mathcal{F}_a$ via the rotation matrix $\mathbf{R}_c^a$ given by:

$$\mathbf{v}^a = \mathbf{R}_c^a \mathbf{v}^c \text{ where } \mathbf{R}_c^a = \{\vec{a}_i \cdot \vec{c}_j\}. \tag{2.4}$$

So $\mathbf{R}_c^a$ is the rotation matrix which takes a vector in the $\mathcal{F}_c$ frame and transforms it to a vector in the $\mathcal{F}_a$ frame. A simple way of explaining this is to describe the transformation as

$$\mathbf{v}^{to} = \mathbf{R}_{from}^{to} \mathbf{v}^{from}.$$

The elements of a rotation matrix $\mathbf{R}_{from}^{to}$ are called the *direction cosines*, and so the rotation matrix is also called the *direction cosine matrix*.

The two main properties of the rotation matrix are that its determinant equals unity and that it is orthogonal:

1. $\mathbf{R}_c^a = (\mathbf{R}_a^c)^{-1} = (\mathbf{R}_a^c)^T$.
2. $\det \mathbf{R} = 1$.

From this a special orthogonal group for the rotation matrix, the $SO(3)$ - *special group of order 3* is defined. See Egeland and Gravdahl [47] for proofs.

$$SO(3) = \left\{\mathbf{R} | \mathbf{R} \in R^{3\times 3},\ \mathbf{R}^T\mathbf{R} = \mathbf{I}_{3\times 3} \text{ and } \det \mathbf{R} = \mathbf{1}\right\} \tag{2.5}$$

where $\mathbf{I}_{3\times 3}$ is the 3 by 3 identity matrix.

One important property of the rotation matrix is called *composite rotations*. This means that a rotation from frame $\mathcal{F}_d$ to frame $\mathcal{F}_a$ can be done by, for instance first rotating from frame $\mathcal{F}_d$ via intermediate frame $\mathcal{F}_c$, and then finally from $\mathcal{F}_c$ to $\mathcal{F}_a$. Say one want to express a vector in the $\mathcal{F}_d$ frame, $\mathbf{v}^d$, in the $\mathcal{F}_a$ frame, and to do so one need to go via frame $\mathcal{F}_c$. To illustrate consider the following equation to obtain the rotation matrix $\mathbf{R}_d^a$ from $\mathcal{F}_d$ to $\mathcal{F}_a$:

$$\mathbf{v}^a = \mathbf{R}_c^a \mathbf{R}_d^c \mathbf{v}^d \Rightarrow \mathbf{R}_d^a = \mathbf{R}_c^a \mathbf{R}_d^c. \tag{2.6}$$

Finally, the rotation matrix has two interpetations:

1. One uses the rotation matrix to *transform* a vector from one coordinate frame to another.
2. One uses the rotation matrix to *rotate* a vector within the coordinate frame it is represented in.

2.3 Attitude representation

Attitude simply means how an object is rotated or oriented relative to a reference frame, and it is evident that the concept of attitude representation and rotation matrices are highly interconnected. With which parameters one wish to describe an object's attitude is what we mean by attitude representation. (Euler angles and Euler parameters will be presented below.)

2.3.1 Euler angles and roll-pitch-yaw

Euler angles are a classical and intuitive way of expressing the attitude. It consists of the roll angle ϕ, pitch angle θ and yaw angle ψ contained in the Euler angle vector $\mathbf{\Theta}$:

$$\mathbf{\Theta} = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix}. \tag{2.7}$$

In a three-axis orthonormal coordinate system, with $\vec{x}$, $\vec{y}$ and $\vec{x}$ spanning out the principal axes, roll describes rotation about the $\vec{x}$ axis, pitch is rotation about the $\vec{y}$ axis and yaw is rotation about the $\vec{z}$ axis.
From this we get that e.g. the matrix $\mathbf{R}_y(\theta)$ denotes a single rotation of θ radians around the $\vec{y}$ axis. The elements of the matrices for all the three axes becomes:

$$\mathbf{R}_x(\phi) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{pmatrix} \tag{2.8}$$

$$\mathbf{R}_y(\theta) = \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix} \tag{2.9}$$

$$\mathbf{R}_z(\psi) = \begin{pmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.10}$$

Say that a transformation of a vector from one frame to another consists of first a rotation ϕ around the x axis, followed by a rotation θ around the current y axis, and finally a rotation ψ around the current z axis. Then the resulting rotation matrix can be written as

$$\mathbf{R}_c^a = \mathbf{R}_x(\phi)\mathbf{R}_y(\theta)\mathbf{R}_z(\psi). \tag{2.11}$$

This formulation is useful and an easier way to interpret a rotation of e.g. a satellite as a sequence of single rotations rather than a rotation matrix.

Now, the concept of the rotation matrix when relating it to Euler angles can be taken one step further. In [47], it is shown that the rotation matrix $\mathbf{R}_c^a$ can be described as a rotation θ about an unit vector $\vec{k}$, and $\mathbf{R}_c^a$ is given by

$$\mathbf{R}_c^a = \cos(\mathbf{I}_{3\times 3}) + \sin\theta\, \mathbf{S}(\mathbf{k}) + (1-\cos\theta)\mathbf{k}(\mathbf{k})^T \tag{2.12}$$

where the $\mathbf{S}(\mathbf{k})$ is the *skew-symmetric form* of the coordinate vector $\mathbf{k}$ defined by

$$\mathbf{S}(\mathbf{k}) = -\mathbf{S}^T(\mathbf{k}) = \begin{bmatrix} 0 & -k_3 & k_2 \\ k_3 & 0 & -k_1 \\ -k_2 & k_1 & 0 \end{bmatrix}, \; \boldsymbol{k} = \begin{bmatrix} k_1 \\ k_2 \\ k_3 \end{bmatrix} \tag{2.13}$$

which gives that the cross product can be written on the form

$$\mathbf{a} \times \mathbf{b} = \mathbf{S}(\mathbf{a})\mathbf{b}. \tag{2.14}$$

In litterature the notation $\mathbf{k}^\times$ instead of $\mathbf{S}(\mathbf{k})$ is used as well.

The parametrization in equation (2.12) is called the *angle-axis* parametrization of the rotation matrix.

How intuitive Euler angles and the roll-pitch-yaw concept may be, one problem when using this representation is that one risk to get singularities for some angles, e.g. $\cos\frac{\pi}{2}$, when calculating the resulting rotation matrix. For this reason Euler *parameters*, also called *quaternions*, is used in the numerical computations in this project. However, for the ease of interpetation the Euler angles are used when presenting some of the results in this thesis.

2.3.2 Euler parameters and Quaternions

As mentioned above, since we can risk to get singularities when using Euler *angles* we are interested in another way to represent the satellite's attitude. This is where Euler *parameters* comes in handy. The Euler parameters is a four-parameter singularity free attitude representation. The Euler parameters and unit quaternions are basically the same, and, to not confuse more than necessary, the expression *quaternion* will be used from now on. The quaternion $\mathbf{q}$ is defined as a complex number with one real part η and three imaginary parts given in the vector $\boldsymbol{\varepsilon}$ defined by:

$$\eta = \cos\frac{\theta}{2}\;, \quad \boldsymbol{\varepsilon} = \mathbf{k}\sin\frac{\theta}{2} = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \end{bmatrix} \tag{2.15}$$

which, as in (2.12), corresponds to a rotation θ about an unit vector $\mathbf{k}$.
The quaternion becomes

$$\mathbf{q} = \begin{bmatrix} \eta & \varepsilon_1 & \varepsilon_2 & \varepsilon_3 \end{bmatrix}^T. \tag{2.16}$$

The quaternion also satisfies the constraint

$$\mathbf{q}^T\mathbf{q} = 1 \;\Rightarrow\; \eta^2 + \boldsymbol{\varepsilon}^T\boldsymbol{\varepsilon} = \eta^2 + \varepsilon_1^2 + \varepsilon_2^2 + \varepsilon_3^2 = 1. \tag{2.17}$$

When using the quaternion in practical implementations, equation (2.17) gives that an object is aligned with the reference frame of interest when $\mathbf{q} = [1\ 0\ 0\ 0]^T$.

One can say that the quaternion parameters express a rotation η about an unit vector $\boldsymbol{\varepsilon}$ in a similar manner as in (2.12), and we get the rotation matrix given by the quaternion parameters η and $\boldsymbol{\varepsilon}$:

$$\mathbf{R}_{k,\theta} = \mathbf{R}_e(\eta, \boldsymbol{\varepsilon}) = \mathbf{I}_{3\times 3} + 2\eta\,\mathbf{S}(\boldsymbol{\varepsilon}) + 2\mathbf{S}^2(\boldsymbol{\varepsilon}) \tag{2.18}$$

Further, once we have the quaternions for the satellite in the body frame and using (2.18), we can obtain the rotation matrix between the body frame of the satellite and the orbit frame:

$$\mathbf{R}_b^o = \mathbf{I}_{3\times 3} + 2\eta\,\mathbf{S}(\boldsymbol{\varepsilon}) + 2\mathbf{S}^2(\boldsymbol{\varepsilon}) \tag{2.19}$$

And finally we get

$$\mathbf{R}_o^b = (\mathbf{R}_b^o)^T \tag{2.20}$$

which will be made use of later.

While the main advantage when using quaternions, namely that it is singularity free, is a big one; a disadvantage is that its physical meaning isn't as intuitive as Euler angles.

2.4 Frame transformations

Now that the concept of the rotation matrix is established, the practice of transforming vectors between frames will be shown. Two assumptions are made; the satellite's orbit eccentricity is zero and the altitude is 600 km.

ECI to ECEF

ECEF to Orbit frame

Orbit to Body frame

2.5 The inertia matrix

In order to have a good model of the satellite, it is important to know as exactly as possible what the satellite's inertia matrix is. Mathematically, the inertia matrix $\mathbf{I_o} \, \epsilon \, \Re^{3\times3}$ about an arbitrary origin *O* is defined according to

$$\mathbf{I_o} := \begin{bmatrix} I_x & -I_xy & -I_xz \\ -I_yx & I_y & -I_yz \\ -I_zx & -I_zy & I_z \end{bmatrix}, \; \mathbf{I_o} = \mathbf{I_o}^T > 0. \tag{2.21}$$

where I_x, I_y and I_z are the moments of inertia about the x_b -, y_b - and z_b - axis respectively, and $I_{xy} = I_{yx}$, $I_{xz} = I_{zx}$ and $I_{yz} = I_{zy}$ are the products of inertia. If one choose to let the principal axes of inertia coincide with the axes of the body frame, the inertia matrix reduces to:

$$\mathbf{I} = \begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix}. \tag{2.22}$$

where I_x, I_y and I_z are given by:

$$\begin{aligned} I_x &= \int_V (y^2 + z^2)\rho_m dV \\ I_y &= \int_V (x^2 + z^2)\rho_m dV \\ I_z &= \int_V (x^2 + y^2)\rho_m dV \end{aligned} \tag{2.23}$$

and ρ_m is the density of the satellite and *V* is the volume.

Throughout this paper the inertia matrix will be referred to as **I**, while the identity matrix wil be referred to as $\mathbf{I}_{n\times n}$, as in equation (2.18) e.g.

2.6 Magnetometer measurements

As we will see in chapter 5, we want to use magnetometer mesurements in the detumbling algorithm, the so-called *B-dot* algorithm. The measurements can be expected to be noisy, and from the name of the algorithm it's apparent that the derivative of the local geomagnetic field, $\dot{\mathbf{B}}$, needs to be found.

In signal processing, and in control theory for that matter, derivating a noisy signal is generally not considered to be a very good idea. The reason for this is that any

noise in the original raw signal will be carried on and amplified in the process of finding its derivative. Different methods will be looked into in this chapter to find the smoothest $\dot{\mathbf{B}}$ values.

A suitable and sufficient sampling frequency of the magnetometer measurements have been found to be 1 Hz. With this frequency and with a measuring/actuating switching every 3 seconds, see section 5.1, we get efficiency and the estimated time derivative of the local geomagnetic field, $\hat{\dot{\mathbf{B}}}$, is stable.

For filter results see section 6.2.1.

2.6.1 Magnetometer filter

For some of the differentiation methods investigated here, the raw and noisy measurement signal is not preferred to use. Before the signal is differentiated we want to filter it in order to remove most of the high frequency noise, hence a digital low-pass filter is found. The measured magnetic field might also be used in other control laws, and in attitude determination schemes, so to filter these measurements is necessary. We use a standard low-pass filter:

$$H(s) = \frac{\hat{B}}{B} = \frac{\omega_c}{s + \omega_c} \tag{2.24}$$

which when written out in discrete form becomes:

$$\hat{\mathbf{B}}_k = a_1 \hat{\mathbf{B}}_{k-1} + b_1 \mathbf{B}_k \tag{2.25}$$

where $\hat{\mathbf{B}}_k$ is the filtered value, at sample k, of the measured value $\mathbf{B}_k$, and a_1 and b_1 are the filter coefficients:

$$a_1 = 0.8.$$

and

$$b_1 = 1 - a_1.$$

When simulating and testing this and other filters, the IGRF model discussed in section 3.4 is used, which calculates the local gemomagnetic field vector represented in the orbit frame, $\mathbf{B}^o$. Noise corresponding to about 1.5 times the expected noise floor given in the chosen magnetometer datasheet, see [48], is added to the computed IGRF value. Since the magnetometer we want to emulate is fixed to the satellite, we further transform the calculated local geomagnetic field vector $\mathbf{B}^o$ to the body frame, and we get $\mathbf{B}^b$. This vector is our emulated magnetometer measurement, i.e. the raw signal on which we want to test the filter found above, and also other filters presented below.

2.6.2 Numerical differentiation

We will here investigate a method of numerical differentiation of a digital signal, namely the central difference method.

Central difference
This method is a quite well-known and basic formula for numerical differentiation of a signal. The formula is a first-order finite difference method and takes basis in the definition of the time derivative of a function $f(x)$, which is:

$$\frac{d}{dt}f(x) = \frac{f(x+\Delta t) - f(x)}{\Delta t}. \tag{2.26}$$

where Δt is the sample time, i.e. the time between each sample. When implemented on the on-board microcontroller this will be fixed to 1 second, and we get a sampling frequency of 1 Hz.
The central difference method is the following:

$$\frac{d}{dt}f(x) = \frac{f(x+\Delta t) - f(x-\Delta t)}{2\Delta t} \tag{2.27}$$

When you have a sequence of samples of a signal, you take the difference between the next sample and the previous and divides by the change in the time. Now you get a smaller error then what you would've got using the standard derivative definition (called *backward difference* when used as a numerical differentiator). This can be directly implemented as a discrete numerical differentiator of the measured geomagnetic field vector **B**. We get the following

$$\hat{\dot{\mathbf{B}}}_k = \frac{\mathbf{B}_{k+1} - \mathbf{B}_{k-1}}{2\Delta t} \tag{2.28}$$

where subscript k, $k+1$ and $k-1$ demotes the current, next and the previous sample, respectively.
Since the measurements are noisy, the values $\mathbf{B}_{k-1}$, $\mathbf{B}_k$ and $\mathbf{B}_{k-1}$ are filtered values from the filter in equation 2.25 in section 2.6.1.

2.6.3 Digital estimator

Here, an estimation algorithm of $\dot{\mathbf{B}}$ is presented, based on (Andersen, 2005).

The estimator can be expressed as a Laplace domain filter:

$$H(s) = \frac{\hat{\dot{B}}}{B} = \frac{\omega_c s}{s + \omega_c} \tag{2.29}$$

with ω_c being the cut-off frequency. A cut-off frequency of $\omega_c = 0.7$ rad/s have been found to be valid.

We will implement this on a computer and microcontroller, so the estmator needs to be discrete. In the z-domain it is the following

$$H(z) = \omega_c \frac{z-1}{z - e^{-\omega_c T_s}} = \frac{b_2(1-z^{-1})}{1-a_2 z^{-1}} \tag{2.30}$$

where T_s is the sample time.
And thus

$$\hat{\dot{B}}(1-a_2 z^{-1}) = b_2(1-z^{-1})B. \tag{2.31}$$

Written out in the discrete notation used when implemented in the Matlab simulation script and eventually on the on-board microcontroller, we get the recurrent filter:

$$\hat{\dot{\mathbf{B}}}_k = a_2 \hat{\dot{\mathbf{B}}}_{k-1} + b_2 \left(\mathbf{B}_k - \mathbf{B}_{k-1}\right) \tag{2.32}$$

with *k* and *k-1* subscript denoting the current and previous sample, respectively, and with $\hat{\dot{\mathbf{B}}}_k$ and $\mathbf{B}_k$ being vectors. The filter coefficients is found to be

$$a_2 = e^{-\omega_c T_s} = 0.9324.$$

and

$$b_2 = 0.2028.$$

One can see that the estimated $\hat{\dot{\mathbf{B}}}$ is calculated by using almost all the information (about 93 %) of the previous estimated value, and adding a low average of the current and previously measured values of **B**. Some high-frequency noise will come through. See more in chapter 6.

2.7 Stability

In order to evaluate the controller strategies presented in chapter 5, we here look at some stability tools for linear and nonlinear systems.

First, we present Lyapunov stability theorems extracted from chapter Chapter 4.1 in Khalil [49].

Theorem 1: *Let $x = 0$ be an equilibrium point for $\dot{x} = f(x)$ and $D \subset R^n$ be a domain containing $x = 0$. Let $V : D \to R$ be a continuously differentiable function such that*

$$V(0) = 0 \text{ and } V(x) > 0 \text{ in } D - \{0\} \tag{2.33}$$

$$\dot{V}(x) \leq 0 \text{ in } D. \tag{2.34}$$

Then, $x = 0$ is stable. Moreover, if

$$\dot{V}(x) < 0 \text{ in } D - \{0\} \tag{2.35}$$

then $x = 0$ is asymptotically stable. $\diamond$

Theorem 2: *Let $x = 0$ be an equilibrium point for $\dot{x} = f(x)$. Let $V : R^n \to R$ be a continuously differentiable function such that*

$$V(0) = 0 \text{ and } V(x) > 0, \ \forall x \neq 0 \tag{2.36}$$

$$||x|| \to \infty \Rightarrow V(x) \to \infty \tag{2.37}$$

$$\dot{V}(x) < 0, \ \forall x \neq 0 \tag{2.38}$$

then $x = 0$ is globally asymptotically stable. $\diamond$

2.8 Controllability

A linear system

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)$$

is said to be controllable if for all initial conditions $\mathbf{x}(0) = \mathbf{x}_0$, final conditions $\mathbf{x}_f$ and $t_f > 0$, there exists an input $\mathbf{u}(t)$, $0 < t < t_f$, such that

$$\mathbf{x}(t) = \mathbf{x}_f.$$

In other words, a linear system is controllable if at any given time t with an input $\mathbf{u}(t)$, it's possible to make the system assume the state $\mathbf{x}_f$.

Further, with the controllability matrix $\mathbf{C}$ we have a mathematical definition of controllability:

Definition 1
For the linear system $\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t)$, and with n being the dimension of $\boldsymbol{A}$, the controllability matrix $\boldsymbol{C}$ is defined as

$$\mathbf{C} = [\mathbf{B}|\mathbf{A}\mathbf{B}|\mathbf{A}^1\mathbf{B}|\dots|\mathbf{A}^{n-1}\mathbf{B}]. \tag{2.39}$$

The linear system is controllable if and only if $\boldsymbol{C}$ has full row rank, i.e. $rank(\boldsymbol{C}) = n$.

2.9 Stabilizability

For a system which is uncontrollable in terms of the definition in 2.8, we have a tool for ensuring that a system with state-feedback control wil be stable, namely the concept of stabilizability:

Definition 2
For the a linear system $\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t)$ *with the state-feedback controller* $\boldsymbol{u} = \boldsymbol{K}\boldsymbol{x}(t)$*, the closed loop system becomes*

$$\mathbf{x}(t) = (\mathbf{A} + \mathbf{BK})\mathbf{x}(t) = \mathbf{A}_c\mathbf{x}(t). \tag{2.40}$$

The linear closed-loop system is said to be stabilizable if $\boldsymbol{A}_c$ *is Hurwitz, i.e. all the eigenvalues of* $\boldsymbol{A}_c$ *has strictly negative real parts.*

Chapter 3

Spacecraft Dynamics and Environment

In this section some basic space calculations will be presented, in addition to a mathematical model of the satellite, the actuators used to control it and the main surroundings influencing it's behaviour.

3.1 Basic orbit dynamics

Orbit dynamics are based on the laws of Newton and Kepler. Kepler's laws describes unperturbed orbital motions of planets around the Sun, while Newton's laws describes the physical laws governing the relationship between the forces acting on a body and its motion due to those forces. In other words, Newton gave us the mathematical explanation of why planets and satellites follow Kepler's laws.

Kepler's laws of planetary motion

1. The orbit of every planet is an ellipse with the Sun at one of the two foci.
2. The line joining a planet and the Sun sweeps out equal areas during equal intervals of time.
3. The square of the planetary period of revolution is proportional to the cube of the planet's mean distance to the Sun.

Newton's laws of motion

1. An object which experiences no net force will have constant velocity; it will

remain in a state of rest (zero velocity) or it will move in a straight line with constant velocity (non-zero velocity).

2. The acceleration **a** of a body is parallel and directly proportional to the net force **F** acting on the body, is in the direction of the net force and is inversely proportional to the mass *m* the body:

$$\mathbf{F} = m\mathbf{a}. \tag{3.1}$$

3. If a body exerts a force $\mathbf{F}_1$ on another body, the other body will exert an equal force with opposite direction $\mathbf{F}_2 = -\mathbf{F}_2$, to the first body. So $\mathbf{F}_1$ and $\mathbf{F}_2$ are equal in magnitude and opposite in direction.

4. The gravitational force between any two objects with mass m_1 and m_2 and distance r, is given by[1]:

$$\mathbf{F} = \frac{Gm_1m_2}{r^2}, \text{ where } G = 6.7638 \times 10^{-11}\left[\frac{m^3}{kgs^2}\right]. \tag{3.2}$$

To find a satellite's angular velocity while orbiting the Earth, we make use of Newton's universal gravitation law and his famous second law, with $\mathbf{a} = \omega_0^2 r$:

$$\begin{aligned} \omega_0^2 r &= \frac{Gm_2}{r^2} \\ \omega_0 &= \sqrt{\frac{Gm_2}{r^3}} \end{aligned} \tag{3.3}$$

We have assumed an orbital altitude of $R_0 = 600$ km. so the distance from the satellite to the centre of the Earth:

$$r = R_e + R_0 = 6371 \text{ km} + 600 \text{ km} = 6971 \text{ km} \tag{3.4}$$

where $R_e = 6371$ km is the Earth's mean radius.

3.2 Satellite dynamics

We assume we can treat the satellite as a rigid body. Lets start with the rotational equivalent of Newtons second law of motion, the law of angular momentum **h**:

$$\mathbf{h} = \mathbf{I}\boldsymbol{\omega} \tag{3.5}$$

[1]This last law is not one of Newton's three laws of motion, but since it's used in our reasoning to end up with the mathematical model which explains a body's orbital dynamics, we choose to list it here. It's name is *Newton's law of universal gravitation*.

where $\mathbf{I}$ is the body's moment of inertia matrix (see chapter 2.5) and $\boldsymbol{\omega}$ is the body's rotational velocity.
The torque $\boldsymbol{\tau} = [\tau_1, \tau_2, \tau_3]^T$ is the total torque acting on the body. The torque law is then given by

$$\boldsymbol{\tau} = \frac{d}{dt}\left(\mathbf{I}\boldsymbol{\omega}\right) \tag{3.6}$$

When this is expressed in the body frame, relative to an inertial frame, such as the ECI for our case, we get the following:

$$\boldsymbol{\tau}^b = \frac{d}{dt}\left(\mathbf{I}^b\boldsymbol{\omega}_{ib}^b\right) + \boldsymbol{\omega}_{ib}^b \times \left(\mathbf{I}^b\boldsymbol{\omega}_{ib}^b\right). \tag{3.7}$$

The moment of inertia matrix is constant in the body frame, for simplicity we denote it $\mathbf{I}$, and the torque law then becomes

$$\boldsymbol{\tau}^b = \mathbf{I}\dot{\boldsymbol{\omega}}_{ib}^b + \boldsymbol{\omega}_{ib}^b \times (\mathbf{I}\boldsymbol{\omega}_{ib}^b). \tag{3.8}$$

The last cross-product part of (3.8) comes from the fact that dynamics are described in the rotating body frame.

After some regrouping and mathematical manipulation we get the following:

$$\begin{aligned} \mathbf{I}\dot{\boldsymbol{\omega}}_{ib}^b &= -\mathbf{S}(\boldsymbol{\omega}_{ib}^b)\mathbf{I}\boldsymbol{\omega}_{ib}^b + \boldsymbol{\tau}^b \\ \dot{\boldsymbol{\omega}}_{ib}^b &= \mathbf{I}^{-1}\left(-\mathbf{S}(\boldsymbol{\omega}_{ib}^b)\mathbf{I}\boldsymbol{\omega}_{ib}^b + \boldsymbol{\tau}^b\right). \end{aligned} \tag{3.9}$$

Here we have ended up with the satellite's dynamics expressed by its angular velocity (w.r.t. the inertial frame) time differentiative.
Indeed, how we want to manipulate the satellite is by changing its angular velocity with the means of a control torque, in order to make it point in a certain direction. From this perspective, the model in (3.8) is convenient for our purpose.

In component form the model is

$$\dot{\omega}_1 = \frac{1}{I_x}\left((I_y - I_z)\omega_2\omega_3 + \tau_1\right) \tag{3.10}$$

$$\dot{\omega}_2 = \frac{1}{I_y}\left((I_z - I_x)\omega_1\omega_3 + \tau_2\right) \tag{3.11}$$

$$\dot{\omega}_3 = \frac{1}{I_z}\left((I_x - I_y)\omega_1\omega_2 + \tau_3\right) \tag{3.12}$$

The total torque, $\boldsymbol{\tau}^b$, in (3.9) consists of a sum of torques acting on the satellite. These are the magnetic control torque, $\boldsymbol{\tau}_m^b$, generated by the magnetorquers, and the total of the disturbance torques, $\boldsymbol{\tau}_{dist}^b$. All of these torques will be derived in later sections.

3.3 Satellite kinematics

The kinematic equations of motion of how the attitude quaternions change are given by

$$\dot{\eta} = -\frac{1}{2}\boldsymbol{\epsilon}^T \boldsymbol{\omega}_{ob}^b \tag{3.13}$$

$$\dot{\boldsymbol{\epsilon}} = \frac{1}{2}\left(\eta \mathbf{I}_{3x3} + \mathbf{S}(\boldsymbol{\epsilon})\right)\boldsymbol{\omega}_{ob}^b \tag{3.14}$$

as given in [47].

Since the attitude we are interested in altering is that of the satellite (body) relative the orbit frame, the corresponding angular velocity to the quaternions representing that attitude is $\boldsymbol{\omega}_{ob}^b$. In order to connect this to the angular velocity $\boldsymbol{\omega}_{ib}^b$ in the expression for the satellite dynamics in (3.9) we have the following relationship

$$\begin{aligned}\boldsymbol{\omega}_{ob}^b &= \boldsymbol{\omega}_{ib}^b - \boldsymbol{\omega}_{io}^b \\ &= \boldsymbol{\omega}_{ib}^b - \mathbf{R}_o^b \boldsymbol{\omega}_{io}^o \\ &= \boldsymbol{\omega}_{ib}^b + \omega_0 \mathbf{c}_2^b \end{aligned} \tag{3.15}$$

where $\boldsymbol{\omega}_{io}^o$ is the orbital angular velocity around the Earth w.r.t. the inertial frame, and it comes in the form of a rotation about the $\vec{y}_o$ axis:

$$\boldsymbol{\omega}_{io}^o = \begin{bmatrix} 0 & -\omega_0 & 0 \end{bmatrix}^T. \tag{3.16}$$

The vector $\mathbf{c}_2^b$ in (3.16) comes from the *column vector representation* of the rotation matrix $\mathbf{R}_o^b$:

$$\mathbf{R}_o^b = \begin{bmatrix} \mathbf{c}_1^b & \mathbf{c}_2^b & \mathbf{c}_3^b \end{bmatrix}, \tag{3.17}$$

where $\mathbf{c}_1^b$, $\mathbf{c}_2^b$ and $\mathbf{c}_3^b$ are the projection of the x_o -, y_o - and z_o - axis in the body frame. For example, the deviation of the z_b - axis and the z_o - axis is given by $\mathbf{c}_3^b$, and if it satisfy $\mathbf{c}_3^b = [0\ 0\ 1]^T$ the two axes are aligned.

For more on angular velocities between different frames in general, and its physical interpretations, see chapter 6.8 in [47].

3.4 The geomagnetic field model

The International Geomagnetic Reference Field (IGRF) is a global model of the geomagnetic field sourrounding the Earth, and it is estimated every five years by

the International Association of Geomagnetism and Aeronomy (IAGA). It consists of a set of Gaussian coefficients that is used to create a spherical harmonical approximation of the field. The latest version is the 11th generation (IGRF 11, revised 2009) model. The model and its application to satellite attitude control have already been well studied by Kristiansen, Makovec and Davis in [20; 26; 50], and a good summary of the model can be seen in Tudor [44]. I therefore refer to these sources, and only present the main IGRF equation here, which is obtained from Davis [50].
The magnetic field, **B**, is defined as the negative gradient of the scalar potential function *V*:

$$\mathbf{B} = -\nabla \mathrm{V} \tag{3.18}$$

$$V(R_c, \lambda', \theta) = R_e \sum_{n=1}^{k} \left(\frac{R_e}{R_c}\right)^{n+1} \sum_{m=0}^{n} (g_n^m \cos m\theta + h_n^m \sin m\theta) P_n^m(\lambda') \tag{3.19}$$

where R_e is the mean radius of the Earth, R_c is the distance from centre of Earth to the point of which one want to find the magnetic field values, λ' is the co-latitude ($\lambda' = 90° -$ latitude) and θ is the longitude. Further, $P(\lambda')$ is the Schmidt normalized associated Legendre polynomials, and g_n^m and h_n^m are the Gaussian coefficiants provided by the IAGA for the IGRF. Thus, the final magnetic field vector components in ECEF spherical coordinates are:

$$\begin{aligned} B_r &= -\frac{\partial V}{\partial R_c} \\ &= \sum_{n=1}^{k} \left(\frac{R_e}{R_c}\right)^{n+2} (n+1) \sum_{m=0}^{n} (g_n^m \cos m\theta + h_n^m \sin m\theta) P_n^m(\lambda') \end{aligned} \tag{3.20}$$

$$\begin{aligned} B_{\lambda'} &= -\frac{1}{R_c} \frac{\partial V}{\partial \lambda'} \\ &= -\sum_{n=1}^{k} \left(\frac{R_e}{R_c}\right)^{n+2} \sum_{m=0}^{n} (g_n^m \cos m\theta + h_n^m \sin m\theta) \frac{\partial P_n^m(\lambda')}{\partial \lambda'} \end{aligned} \tag{3.21}$$

$$\begin{aligned} B_\theta &= -\frac{1}{R_c \sin \lambda'} \frac{\partial V}{\partial \theta} \\ &= -\frac{1}{\sin \lambda'} \sum_{n=1}^{k} \left(\frac{R_e}{R_c}\right)^{n+2} \sum_{m=0}^{n} (-g_n^m \sin m\theta + h_n^m \sin m\theta) P_n^m(\lambda') \end{aligned} \tag{3.22}$$

3.5 Actuator model

The satellite's attitude will be controlled using magnetic coils as actuators, also called magnetorquers. When applying an electrical current through the windings

of the coils, a magnetic dipole field will be created. The magnetic dipole can be described as a vector, and the vector will be perpendicilar to the coil, pointing in the direction of the extended thumb of one's hand according to the right-hand rule. When this field reacts with the local geomagnetic field, a magnetic moment is generated. In a very intuitive way, the magnetic field generated by the magnetorquers will try to align itself with the local geomagnetic field surrounding the satellite at that particular place. Thus, the satellite will experience a torque acting on it; pushing it in order to align it with the geomagnetic field.
The magnetic dipole moment generated by the coils is given by the number of windings *N*, the current *i* and the coil area *A*. In our satellite we will have coils on the x^+-, y^+- and z^-- panels, so the total moment will be

$$\mathbf{m}^b = \mathbf{m}_x^b + \mathbf{m}_y^b + \mathbf{m}_z^b = \begin{bmatrix} N_x i_x A_x \\ N_y i_y A_y \\ N_z i_z A_z \end{bmatrix}. \tag{3.23}$$

The cross product of the magnetic dipole moment, $\mathbf{m}^b$, and the local geomagnetic field, $\mathbf{B}^b$, generates the magnetic control torque $\boldsymbol{\tau}_m^b$:

$$\boldsymbol{\tau}_m^b = \mathbf{m}^b \times \mathbf{B}^b = \mathbf{S}(\mathbf{m}^b)\mathbf{B}^b. \tag{3.24}$$

This torque is what will be used to maneuver the satellite.

One inherit problem with magnetic control is that the system at times is underactuated. This comes from the fact that the torque is generated by the cross product between the magnetic moment, $\mathbf{m}^b$, and the geomagnetic field, $\mathbf{B}^b$. If, for instance, we want to turn the satellite about the *x* axis, and this is axis is parallel to the local geomagnetic field vector; then control is lost and the system is in fact underactuated. Further, since when any of the magnetorquers is parallel to the local geomagnetic field we loose one degree of freedom, from definition 2.8 our system is uncontrollable.
Another problem is that the coils only can produce a moment perpendicular to the coil area. Since the desired moment rarely will coincide with one of the coil moments, one have to do some modifications to the control algorithms. More on this in chapter 5.

3.6 Environmental disturbances

There are several different environmental disturbances which will act on the satellite while orbiting the Earth, most of them will be disturbance torques. To list some of them, one should mention

- Gravitation

- Aerodynamic torque.
- Solar radiation torque.
- Internal magnetic dipole moments and other electrical noise which could lead to measurement errors.

The disturbances listed above will, without doubt, all act on the satellite. Some argue that since the size of a CubeSat is relatively small, some of the disturbances will be so small they can be neglected. In low Earth orbits the most influental disturbance torques are gravity, aerodynamic and possibly magnetic dipoles. In any case we choose to model and investigate them here.

3.6.1 Gravitation

The gravitation disturbance comes in form of a gravity-gradient torque, which will affect any non symmetric body in the Earth's gravity field.
When a non-symmetric satellite is orbiting the Earth, different gravitation forces from the Earth will act on different places on the satellite, which will lead to a gravity-gradient disturbance torque acting on the satellite. According to (Hughes, 1986), it's given by

$$\boldsymbol{\tau}_g^b = 3\omega_0^2 \mathbf{S}(\mathbf{c}_3^b)\mathbf{I}\mathbf{c}_3^b \tag{3.25}$$

where $\mathbf{I}$ is the satellites moment of inertia, ω_0 is the orbital angular velocity from (3.3) and $\mathbf{c}_3^b$ is given in (3.17).

The gravitational force will be slightly bigger on the part of the satellite being closer to the Earth, and because of this the gravity-gradient torque will actually have a small stabilizable effect on the satellite according to our desired attitude. It will in a way *pull* one of the z-facets towards the Earth, depending on the mass distribution of the satellite. From the stability analysis in section/chapter ??, it's shown that the satellite will be gravity-gradient stable if the satellite's moments of inertia are designed in a certain way:

$$I_y > I_z > I_z. \tag{3.26}$$

3.6.2 Aerodynamic torque

Even though there aren't many particles crashing into the satellite in LEO altitudes (the atmosphere by definition ended at 100 km altitude), there are in fact some areodynamic drag creating a torque on the satellite. In order to have an as accurate model as possible, the model obtained in (Hughes, 1986) takes the satellite's

rotating movements into account:

$$\boldsymbol{\tau}_a^b = \rho_a V_r^b \left(V_r^b A_p \mathbf{S}(\mathbf{c}_p) \hat{\mathbf{V}}_r^b - \left(\mathbf{I} + \mathbf{S}(\hat{\mathbf{V}}_r^b)\mathbf{J} \right) \boldsymbol{\omega}_{ob}^b \right) \tag{3.27}$$

where

- ρ_a is the atmospheric density.
- A_p is the total projected area the airflow "sees".
- $\mathbf{c}_p$ is the location of the atmospheric centre of pressure relative centre of gravity.
- V_r^b is the magnitude of the local atmospheric velocity vector, $V_r^b = \left|\mathbf{V}_r^b\right|$.
- $\hat{\mathbf{V}}_r^b$ is the *unit* velocity vector of the local atmosphere relative to the satellite surface, $\hat{\mathbf{V}}_r^b = \mathbf{V}_r^b / V_r^b$.
- $\mathbf{J}$ is the moment of inertia matrix with origin in the centre of pressure.
- $\boldsymbol{\omega}_{ob}^b$ is the angular velocity of the satellite relative the orbit frame.

For practical purposes, the local velocity vector is calculated in the orbit frame, so we need to transform it to the body frame:

$$\mathbf{V}_r^b = \mathbf{R}_o^b \mathbf{V}_r^o \tag{3.28}$$

Lastly, the projected area in (3.27) is constant, while in reality it will vary depending on the satellite's attitude. A worst-case area is used, i.e. the projected area seen when one of the corner sides between the x- and y-sides are pointing in the velocity direction:

$$A_p = z\sqrt{x^2 + y^2}. \tag{3.29}$$

3.6.3 Solar radiation torque

Electromagnetic radiation from the Sun may produce a pressure on the satellite surface and in this way and create a disturbance torque. The solar radiation torque expression is given by

$$\boldsymbol{\tau}_s^b = \left(\frac{F_s}{c} A_p (1+q) \cos i \right) c_{ps} \tag{3.30}$$

where

- F_s is the solar constant, $F_s = 1367\ W/m^2$.
- c is the speed of light, $c = 3 \times 10^8\ m/s$.
- A_p is the total projected area.
- q is the reflectance factor.
- i is the angle of incidence of the Sun.
- c_{ps} is the location of the solar centre of pressure relative centre of gravity.

As for the aerodynamic torque, a constant worst-case projected area is used. See equation 3.29.

3.6.4 Internal magnetic dipole

The way in which the dipole moments of the magnetorquers are created apply to any kind of conductor which have a current running through it. This leads to small internal dipole moments being created all the time when any electronics on board the satellite is being used. It is very difficult to have a model for this to include in simulation models, for the reason that different electronics have different characteristics and on the satellite thousands of different components are being used. To try to model all of this would be very time demanding and not very constructive. However, it is a matter that is worth being aware of, and one might want to include these effects as a bias in the control algorithms. This has not been done at this time, but some very rough estimations of the dipole effects is shown in chapter 6.
The resulting internal magnetic dipole disturbance torque will be created in the same manner as the control torque from section 3.5:

$$\boldsymbol{\tau}^b_{m,d} = \mathbf{m}^b_d \times \mathbf{B}^b = \mathbf{S}(\mathbf{m}^b_d)\mathbf{B}^b. \tag{3.31}$$

3.7 Linearized satellite model

A commonly used controller for attitude control of a small satellite using only magnetorquers is the Linear-Quadratic Regulator. The word *linear* makes it clear that it's based on linear dynamics, which will be presented here, while the rest of the controller as a whole will be further investigated in section 5.5. Two assumptions are made here; that in the equlibrium point the disturbance torques of solar radiation and internal dipole will have a negliable impact on the satellite. the linearization is done according to Busterud [22] and Øverby [23].

With the definition of the orbit reference frame in 2.1, the satellite have the desired attitude when it's aligned with the orbit frame, i.e. the linearization will be around the equilibrium points

$$\mathbf{q}_{eq} = \begin{bmatrix} \eta_{eq} \\ \boldsymbol{\varepsilon}_{eq} \end{bmatrix} = \begin{bmatrix} 1 \\ \mathbf{0} \end{bmatrix} \tag{3.32}$$

Kinematics

Looking at the kinematics equation in (3.14) and using the constraint property of (2.17), we have that since η is defined by the $\boldsymbol{\varepsilon}$-vector elements, η can be omitted in the linearization: $\boldsymbol{\varepsilon} = \boldsymbol{\varepsilon}_{eq} + \delta\boldsymbol{\varepsilon} = \delta\boldsymbol{\varepsilon}$, where $\delta\boldsymbol{\varepsilon}$ are small perturbations from the equlibrium. When linearizing the satellite kinematics in this equilibrium point we get:

$$\delta\dot{\boldsymbol{\varepsilon}} = \frac{1}{2}\boldsymbol{\omega}_{ob}^{b} \tag{3.33}$$

Rotation matrix

From (2.19) we get that the rotation matrix corresponding to the quaternion describing the satellite's attitude relative the orbit frame is

$$\begin{aligned} \mathbf{R}_{o}^{b} &= (\mathbf{R}_{b}^{o})^{T} \\ &= \mathbf{I}_{3\times 3} - 2\eta\mathbf{S}(\boldsymbol{\varepsilon}) - 2\mathbf{S}^{2}(\boldsymbol{\varepsilon}) \end{aligned} \tag{3.34}$$

which for the equilibrium point can be written as

$$\mathbf{R}_{o}^{b} = \mathbf{I}_{3\times 3} - 2\mathbf{S}(\delta\boldsymbol{\varepsilon}) \tag{3.35}$$

$$\mathbf{R}_{o}^{b} = \begin{bmatrix} 1 & 2\delta\varepsilon_3 & -2\delta\varepsilon_2 \\ -2\delta\varepsilon_3 & 1 & 2\delta\varepsilon_1 \\ 2\delta\varepsilon_2 & -2\delta\varepsilon_1 & 1 \end{bmatrix}. \tag{3.36}$$

Angular velocity

Using the relationship in (3.15) we get the following expression for the angular velocity of the satellite relative the ECI frame:

$$\boldsymbol{\omega}_{ib}^{b} = \mathbf{R}_{o}^{b}\boldsymbol{\omega}_{io}^{o} + \boldsymbol{\omega}_{ob}^{b} \tag{3.37}$$

And from (3.33) one can see that $\boldsymbol{\omega}_{ob}^{b} = 2\dot{\boldsymbol{\varepsilon}}$.
Inserting this and (3.36) into equation (3.37), we get the linearized model of$\boldsymbol{\omega}_{ib}^{b}$ to be

$$\boldsymbol{\omega}_{ib}^{b} = \begin{bmatrix} 2\delta\dot{\varepsilon}_1 - 2\omega_0\delta\varepsilon_3 \\ 2\delta\dot{\varepsilon}_2 - \omega_0 \\ 2\delta\dot{\varepsilon}_3 + 2\omega_0\delta\varepsilon_1 \end{bmatrix}, \tag{3.38}$$

with its time derivative being

$$\dot{\boldsymbol{\omega}}_{ib}^{b} = \begin{bmatrix} 2\delta\ddot{\varepsilon}_1 - 2\omega_0\delta\dot{\varepsilon}_3 \\ 2\delta\ddot{\varepsilon}_2 \\ 2\delta\ddot{\varepsilon}_3 + 2\omega_0\delta\dot{\varepsilon}_1 \end{bmatrix}. \tag{3.39}$$

Magnetic control torque

From (3.24) the magnetic control torque is

$$\boldsymbol{\tau}_m^b = \mathbf{S}(\mathbf{m}^b)\mathbf{B}^b. \tag{3.40}$$

Since our model of the local geomagnetic field, the IGRF, is expressed in the orbit frame we have that

$$\boldsymbol{\tau}_m^b = \mathbf{S}(\mathbf{m}^b)\mathbf{R}_o^b\mathbf{B}^o \tag{3.41}$$

which in the equilibrium point is the following

$$\boldsymbol{\tau}_m^b = \mathbf{S}(\mathbf{m}^b)\mathbf{B}^o = \begin{bmatrix} B_z^o m_y - B_y^o m_z \\ B_x^o m_z - B_z^o m_x \\ B_y^o m_x - B_x^o m_y \end{bmatrix}. \tag{3.42}$$

Gravity-gradient

In Busterud [22] it is shown that the expression for the gravity-gradient in (3.25) written out with quaternion parameters is

$$\boldsymbol{\tau}_g^b = 3\omega_0^2 \begin{bmatrix} 2(I_z - I_y)(\varepsilon_2\varepsilon_3 + \eta\varepsilon_1)(1 - 2(\varepsilon_1^2 + \varepsilon_2^2)) \\ 2(I_x - I_z)(\varepsilon_1\varepsilon_3 + \eta\varepsilon_2)(1 - 2(\varepsilon_1^2 + \varepsilon_2^2)) \\ 4(I_y - I_x)(\varepsilon_1\varepsilon_1 + \eta\varepsilon_2)(\varepsilon_2 + \eta\varepsilon_1) \end{bmatrix}, \tag{3.43}$$

and inserting for the equlibrium point we find

$$\boldsymbol{\tau}_g^b = 3\omega_0^2 \begin{bmatrix} (I_z - I_y)\delta\varepsilon_1 \\ (I_x - I_z)\delta\varepsilon_2 \\ 0 \end{bmatrix}. \tag{3.44}$$

Areodynamic torque

The aerodynamic torque expression from (3.27) with $\boldsymbol{\omega}_{ob_{eq}}^b = \mathbf{0}$ from (3.32) inserted is

$$\boldsymbol{\tau}_a^b = \rho_a (V_r^b)^2 A_p \mathbf{S}(\mathbf{c}_p)\hat{\mathbf{V}}_r^b. \tag{3.45}$$

Further, from (3.28) and the definition of $\hat{\mathbf{V}}_r$ we have that

$$\hat{\mathbf{V}}_r^b = \mathbf{V}_r^b / V_r^b = (\mathbf{R}_o^b \mathbf{V}_r^o)/V_r^b, \tag{3.46}$$

so that when inserting for $\mathbf{R}_o^b$ from (3.34) and using the equilibrium quaternion parameters, we get

$$\hat{\mathbf{V}}_r^b = \mathbf{V}_r^o / V_r^o. \tag{3.47}$$

Here we have also used the fact that the magnitude of the velocity vectors represented in different frames are the same in the equilibrium, $V_r^b = V_r^o$.
And finally we arrive at the linearized aerodynamic torque:

$$\boldsymbol{\tau}_a^b = \rho_a V_r^o A_p \begin{bmatrix} c_{p_2} V_{r_3}^o - c_{p_3} V_{r_2}^o \\ c_{p_3} V_{r_1}^o - c_{p_1} V_{r_3}^o \\ c_{p_1} V_{r_2}^o - c_{p_2} V_{r_1}^o \end{bmatrix}. \tag{3.48}$$

Linearized satellite dynamics

Putting together equations (3.9), (3.42), (3.44) and (3.48), the linearized satellite dynamics becomes

$$\dot{\boldsymbol{\omega}}_{ib}^b = \mathbf{I}^{-1} \left[-\mathbf{S}(\boldsymbol{\omega}_{ib}^b)\mathbf{I}\boldsymbol{\omega}_{ib}^b + \boldsymbol{\tau}_m^b + \boldsymbol{\tau}_g^b + \boldsymbol{\tau}_a^b \right] \tag{3.49}$$

We introduce

$$\sigma_x = \tfrac{I_y - I_z}{I_x}, \quad \sigma_y = \tfrac{I_z - I_x}{I_y}, \quad \sigma_y = \tfrac{I_x - I_y}{I_z}. \tag{3.50}$$

Now we can write the cross-product in (3.49) above as

$$-\mathbf{I}^{-1}\mathbf{S}(\boldsymbol{\omega}_{ib}^b)\mathbf{I}\boldsymbol{\omega}_{ib}^b = \begin{bmatrix} \sigma_x \boldsymbol{\omega}_{ib,y}^b \boldsymbol{\omega}_{ib,z}^b \\ \sigma_y \boldsymbol{\omega}_{ib,x}^b \boldsymbol{\omega}_{ib,z}^b \\ \sigma_z \boldsymbol{\omega}_{ib,x}^b \boldsymbol{\omega}_{ib,y}^b \end{bmatrix}, \tag{3.51}$$

which in the equilibrium can be approximated with

$$-\mathbf{I}^{-1}\mathbf{S}(\boldsymbol{\omega}_{ib}^b)\mathbf{I}\boldsymbol{\omega}_{ib}^b = \begin{bmatrix} \sigma_x(\delta\omega_y - \omega_0)\delta\omega_z \\ \sigma_y \delta\omega_x \delta\omega_z \\ \sigma_z \delta\omega_x(\delta\omega_y - \omega_0) \end{bmatrix} \approx \begin{bmatrix} -\sigma_x \omega_0 \delta\omega_z \\ 0 \\ -\sigma_z \omega_0 \delta\omega_x \end{bmatrix}. \tag{3.52}$$

Equation (3.49) written out becomes

$$\begin{aligned} \dot{\boldsymbol{\omega}}_{ib}^b = &\begin{bmatrix} -\sigma_x \omega_0 \delta\omega_z \\ 0 \\ -\sigma_z \omega_0 \delta\omega_x \end{bmatrix} + \begin{bmatrix} (B_z^o m_y - B_y^o m_z)/I_x \\ (B_x^o m_z - B_z^o m_x)/I_y \\ (B_y^o m_x - B_x^o m_y)/I_z \end{bmatrix} \\ &+ 3\omega_0^2 \begin{bmatrix} -\sigma_x \delta\varepsilon_1 \\ -\sigma_y \delta\varepsilon_2 \\ 0 \end{bmatrix} + \rho_a V_r^o A_p \begin{bmatrix} (c_{p_2} V_{r_3}^o - c_{p_3} V_{r_2}^o)/I_x \\ (c_{p_3} V_{r_1}^o - c_{p_1} V_{r_3}^o)/I_y \\ (c_{p_1} V_{r_2}^o - c_{p_2} V_{r_1}^o)/I_z \end{bmatrix}. \end{aligned} \tag{3.53}$$

Finally, when inserting for $\dot{\boldsymbol{\omega}}_{ib}^{b}$ from equation (3.39) and after manipulating[2] one get the linearized model on component form:

$$\begin{aligned}
\delta\ddot{\varepsilon}_1 &= (1-\sigma_x)\omega_0\delta\dot{\varepsilon}_3 - 4\sigma_x\omega_0^2\delta\varepsilon_1 + \frac{1}{2I_x}(B_z^o m_y - B_y^o m_z) \\
&\quad + \rho_a V_r^o A_p(c_{p_2}V_{r_3}^o - c_{p_3}V_{r_2}^o)/I_x \\
\delta\ddot{\varepsilon}_2 &= -3\sigma_y\omega_0\delta\dot{\varepsilon}_2 + \frac{1}{2I_y}(B_x^o m_z - B_z^o m_x) \\
&\quad + \rho_a V_r^o A_p(c_{p_3}V_{r_1}^o - c_{p_1}V_{r_3}^o)/I_y \\
\delta\ddot{\varepsilon}_3 &= -(1-\sigma_x)\omega_0\delta\dot{\varepsilon}_1 - \sigma_z\omega_0^2\delta\varepsilon_3 + \frac{1}{2I_z}(B_y^o m_x - B_x^o m_y) \\
&\quad + \rho_a V_r^o A_p(c_{p_1}V_{r_2}^o - c_{p_2}V_{r_1}^o)/I_z
\end{aligned} \tag{3.54}$$

3.7.1 Linearized equations of motion

For this LQR controller the ε-vector of the quaternion and its time derivative is chosen as the states, and we have the follwing state vector:

$$\mathbf{x}(t) = \begin{bmatrix} \delta\varepsilon_1 & \delta\dot{\varepsilon_1} & \delta\varepsilon_2 & \delta\dot{\varepsilon_2} & \delta\varepsilon_3 & \delta\dot{\varepsilon_3} \end{bmatrix}^T, \tag{3.55}$$

with the gain vector

$$\mathbf{u}(t) = \begin{bmatrix} m_x & m_y & m_z \end{bmatrix}^T, \tag{3.56}$$

and disturbance vector

$$\mathbf{v}(t) = \begin{bmatrix} V_r^o & V_r^o & V_r^o \end{bmatrix}^T. \tag{3.57}$$

The linearized system represented in the state-space model form becomes

$$\dot{\mathbf{x}}(t) = \mathbf{F}\mathbf{x}(t) + \mathbf{G}(t)\mathbf{u}(t) + \mathbf{D}(t)\mathbf{v}(t),$$

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t),$$

[2] See Appendix A.2 in [23] for complete deduction of the linearized model

where

$$\mathbf{F} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ -4\sigma_x\omega_0^2 & 0 & 0 & 0 & 0 & (1-\sigma_x)\omega_0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -3\sigma_y\omega_0^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & -(1-\sigma_z)\omega_0 & 0 & 0 & -\sigma_z\omega_0^2 & 0 \end{bmatrix}, \tag{3.58}$$

$$\mathbf{G}(t) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \frac{B_z^o(t)}{2I_x} & \frac{-B_y^o(t)}{2I_x} \\ 0 & 0 & 0 \\ \frac{-B_z^o(t)}{2I_y} & 0 & \frac{B_x^o(t)}{2I_y} \\ 0 & 0 & 0 \\ \frac{B_y^o(t)}{2I_z} & \frac{-B_x^o(t)}{2I_z} & 0 \end{bmatrix}, \tag{3.59}$$

$$\mathbf{D}(t) = \rho_a A_p \begin{bmatrix} (c_{p_2}V_{r_3}^o - c_{p_3}V_{r_2}^o)/I_x \\ (c_{p_3}V_{r_1}^o - c_{p_1}V_{r_3}^o)/I_y \\ (c_{p_1}V_{r_2}^o - c_{p_2}V_{r_1}^o)/I_z \end{bmatrix}. \tag{3.60}$$

Chapter 4

Magnetorquer design

In the work of Tudor [44] an early prototype of the magnetorquer was designed. Since then, a whole lot of different subsystems have been designed and developed for the NUTS satellite. Most important for the magnetorquer design have been the progress in the mechanical design of the satellite's frame. With the good help of my fellow students Christian E. Nomme and Kim Sandvik, who are currently working on the mechanical design of the frame, I've been able to design a more realistic magnetorquers which fit with the overall and mehanical satellite design.

There's two different designs of the magnetorquers; one for the magnetorquer placed on the z-facet and one for the two magnetorquers placed on the x- and y-facets.
When working with this, the main restriction proved to be that of space available on the satellite, with other being power and energy available from the EPS and to some degree weight of the magnetorquers. Further, when designing the magnetorquers there are a few tradeoffs. Using the expression for the magnetic moment generated by the magnetorquers in (3.23), essentially $m = niA$, we have that we want to maximize the number of turns N, current i and area A. Thinner copper wire increases number of turns, but also the thinner the wire, the higher its resistance per meter is, and thus the current decreases. In addition, with higher amount of turns the longer the wire, and the resistance increases as well. Theres also a limit of current available from the EPS system, so it will not help very much with a very low resistance since the current will meet a roof limit.

4.1 Design specification

In the EPS design done by Lars Erik Jacobsen in Jacobsen [34] each subsystem will have 300 mA accessible, which means 100 mA on each magnetorquer. How-

Parameters	Magnetorquer 1	Magnetorquer 2
AWG number	AWG 31	AWG 31
Conductor diameter [mm]	0.226	0.226
Mass [g]	36	25
Mean height [mm]	165	77
Mean width [mm]	63	77
Mean ara [mm^2]	10395	5929
Cross-section height [mm]	4	4
Cross-section width [mm]	3	3
Number of windings	221	221
Resistance [Ω]	43.03	23.78
Max current in conductor [mA]	113	113
Inductance [mH]	5.6	4.3
Time constant [ms]	0.13	0.18
Magnetic moment [Am^2]	0.2596	0.1481

Table 4.1: Table of magnetorquer parameters.

ever, this is merely a current regulator on the EPS module which may be changed. Further, the EPS provides a 5V power supply and there's 29 Wh of available energy from the battery package. The size specification are based on the mechanical drawings of the magnetorquer frames done by Christian E. Nomme, see appendix B. From this a mean area is calculated for the two different magnetorquers. The cross-sectional area specification is also found in the drawings in appendix B. In table 4.1 the magnetorquer constraints and resulting parameters are found.

4.2 Coil design

The magnetorquer is basically a simple coil of winded wire around a frame.

For the conductor in the coil, copper is the chosen material since it's the most available and most of its specifics are quite known, making the design process easier. The wire evaluation were done based on the American Wire Gauge (AWG) standard, see [51]. A few different wire diameters were evaluated, and the results can be found in table 4.2. The calculations resulting in table 4.2 are based on the size of the biggest magnetorquer (for x- and y-sides), and the chosen wire is AWG 31 with diameter 0.226 mm.

To find the coil's inductance we first need to know it's magnetic flux:

$$\Phi = BA = \mu_0 \frac{2\sqrt{2}nI}{\pi d} \cdot A = \mu_0 \frac{2\sqrt{2}nI}{\pi\sqrt{hw}} \cdot A, \tag{4.1}$$

	AWG 30	**AWG 31**	**AWG 32**	**AWG 33**
Diameter [mm]	0.305	0.226	0.252	0.230
Resistance [Ω/m]	0.339	0.427	0.538	0679
Max amps [mA]	142	113	91	72
Turns	165	221	266	352
Mass [g]	27	36	43	57
Total Resistance [Ω]	25.51	43.03	65.26	109
Magnetic moment [Am^2]	0.2436	0.2596	0.2119	0.1679

Table 4.2: AWG parameters.

where the coil length d is approximated with $\sqrt{hw}$, and h and w is the coil's height and width respectively. Now, the inductance of the biggest coil is found by the formula for a rectangular coil:

$$L = \frac{n\Phi}{I} = \frac{2\sqrt{2}n^2\mu_0 A}{\pi\sqrt{hw}} = \frac{2\sqrt{2}}{\pi}n^2\mu_0\sqrt{hw}$$
$$L \approx 5.6 \text{ mH}. \tag{4.2}$$

From this the coil's time constant is found to be

$$\tau = \frac{L}{R} = \frac{5.6}{43.03} \approx 0.13\text{ms}. \tag{4.3}$$

This is a very small inductance and thus a small time constant, and one may have reason to believe the coil's inductance to be a bit higher. Even if one were to find the inductance to be tenfold higher than in (4.2), the time constant would be small enough that doing switching with a few seconds, or even less, is achievable.

Because a magnetorquer frame hasn't been made yet, a coil prototype haven't been made so unfortunately measurements haven't been done in order to verify the calculated values. A prototype coil without any frame were however made by Tudor in [44], and it worked nicely.

Chapter 5

Control

The main payload of the satellite, the IR camera, needs to point in the nadir direction towards the Earth in order to take pictures of the gravity waves. With our definition of the reference frames in section 2.1, the satellite will be nadir-pointing when the body and reference frames are aligned. From the work done by Snorre S. Rønning in [37], a requirement for getting any pictures at all is that the satellite attitude doesn't deviate more 50 degrees. However, a higher accuracy is desirable. Further, the satellite while taking pictures should not drift much. Thus, even if the satellite were to deviate from the desired attitude, we would want it to be stable. This leads to the requirement that the satellite's desired angular velocity w.r.t. the orbit frame is close to zero. The ACS requirements are summed up in table 5.1.

In this chapter the control strategy and control algorithms are presented. In order to find a way of meeting the ACS requirements, different control algorithms are presented, evaluated and stability properties are found.

5.1 Control strategy

After being released from the P-POD on the launch vehicle, the satellite will spin arbitrarily around its three axes. In order to be able to point the satellite at the desired attitude, its spin needs to be slown down first, i.e. detumbling is necessary. After detumbling, the satellite's angular velocity will hopefully be close to zero

	Roll	**Pitch**	**Yaw**
Pointing accuracy [° deg]	± 25	± 25	-
Angular velocity [rad/s]	$\pm 1 \cdot 10^{-3}$	$\pm 1 \cdot 10^{-3}$	$\pm 1 \cdot 10^{-3}$

Table 5.1: ACS requirements.

and one can employ pointing control.

The detumble algorithm presented here makes use of a measurement of the Earth's geomagnetic field. Since the actuators are magnetic coils (magnetorquers), there's a conflict between measuring the magnetic field surrounding the satellite and using the actuators, which will influence the magnetometer measurements. Because of this a switching strategy is suggested. What we want to do is store a sequence of the measured magnetic field, and use the estimator in 2.6.3 to estimate the time derivative of the local geomagnetic field. This estimated $\hat{\dot{\mathbf{B}}}$ is the stored and used in the detumbling algorithm, while the magnetometer is turned off since a measurement at this stage might be disturbed and thus maybe useless. (Since the magnetometer may be a measurement being used in an attitude determination algorithm, the switching strategy is used while using all the control algorithms.) A method of emulating the switching has been implemented in the Matlab script used for simulation.
One can note that one could use a strategy of feed forward from the known actuator gain in order get a measurement while using the magnetorquers, but this is outside the scope of this thesis.

After detumbling the satellite, one can, as mentioned above, start pointing control. Dependent on how large the deviations away from the desired attitude are, different controller are used. Typically, with more than 20° deviation we want to use the nonlinear PD-like controller based on Soglo [25] and Tudor [44], and for smaller deviations the linear LQR controller based on Busterud [22], Øverby [23] and Wisniewski [14]) is preferred.

5.2 Stability requirements

By using Lyaounov theory with the concepts of stability presented in section 2.7, together with energy considerations of the satellite we can find some requirements for stability. The stability analysis shown here is heavily based on the Lyapunov analysis by Soglo [25] and Tudor [44], but also adding the energy contribution from the aerodynamic torque.

The energy of the system consists of kinetic and potential energy. Kinetic energy is the satellite's translational and rotatioinal energy, while the potential energy is mainly because of gravity-gradient effects.

Let's start with the kinetic energy:

$$T = \frac{1}{2} m_b \omega_o^2 R_c^2 + \frac{1}{2} (\boldsymbol{\omega}_{ib}^b)^T \mathbf{I} \boldsymbol{\omega}_{ib}^b \tag{5.1}$$

and when inserting for $\boldsymbol{\omega}_{ib}^{b}$ we get

$$T = \frac{1}{2}m_b\omega_o^2 R_c^2 + \frac{1}{2}\omega_o^2 \mathbf{c}_2^T \mathbf{I}\mathbf{c}_2 - \omega_o \mathbf{c}_2^T \mathbf{I}\boldsymbol{\omega}_{ob}^{b} + \frac{1}{2}(\boldsymbol{\omega}_{ob}^{b})^T \mathbf{I}\boldsymbol{\omega}_{ob}^{b} \tag{5.2}$$

And by grouping 5.2 into

$$T = T_0 + T_1 + T_2 \tag{5.3}$$

we arrive at

$$T_0 = \frac{1}{2}m_b\omega_o^2 R_c^2 + \frac{1}{2}\omega_o^2 \mathbf{c}_2^T \mathbf{I}\mathbf{c}_2 \tag{5.4}$$

$$T_1 = -\omega_o \mathbf{c}_2^T \mathbf{I}\boldsymbol{\omega}_{ob}^{b} \tag{5.5}$$

$$T_2 = \frac{1}{2}(\boldsymbol{\omega}_{ob}^{b})^T \mathbf{I}\boldsymbol{\omega}_{ob}^{b} \tag{5.6}$$

The potential energy is dominated by the gravitational force working on the satellite caused by the Earth. We make use of Newton's law of gravity and the following assumptions

- The Earth is the only object exerting a gravitational force on the satellite.
- The Earth is close to spherical and has a symmetric mass distribution.
- The satellite is small compared to distance to the centre of the Earth.

The potential energy exerted by the Earth on the satellite is then given by

$$U = -\frac{\mu M_b}{R_c} - \frac{1}{2}\omega_o^2(I_x + I_y + I_z) + \frac{3}{2}\omega_o^2 \mathbf{c}_3^T \mathbf{I}\mathbf{c}_3. \tag{5.7}$$

with M_b being the mass of the satellite.

From [25] the Lyapunov function candidate is given by

$$V = \frac{1}{2}(\boldsymbol{\omega}_{ob}^{b})^T \mathbf{I}\boldsymbol{\omega}_{ob}^{b} + \frac{3}{2}\omega_o^2 \mathbf{c}_3^T \mathbf{I}\mathbf{c}_3 - \frac{1}{2}\omega_o^2 \mathbf{c}_2^T \mathbf{I}\mathbf{c}_2 + \frac{1}{2}\omega_o^2(I_y - 3I_z), \tag{5.8}$$

which when evaluated in the equilibrium states $\boldsymbol{\omega}_{ob}^{b} = 0$ and $\mathbf{R}_b^o = 0$, and manipulated and written out on scalar form becomes

$$\begin{aligned} V = &\frac{1}{2}(\boldsymbol{\omega}_{ob}^{b})^T \mathbf{I}\boldsymbol{\omega}_{ob}^{b} + \frac{3}{2}\omega_o^2[(I_x - I_z)c_{13}^2 + (I_y - I_z)c_{23}^2] \\ &+ \frac{1}{2}\omega_o^2[(I_y - I_x)c_{12}^2 + (I_y - I_x)c_{32}^2]. \end{aligned} \tag{5.9}$$

If we now define a state vector $\mathbf{x} = \begin{bmatrix}(\boldsymbol{\omega}_{ob}^{b})^T & c_{13} & c_{23} & c_{12} & c_{32}\end{bmatrix}^T$, we can find the requirements which needs to be fulfilled in order to have Lyapunov stability. From

the Lyapunov function candidate in 5.9, we see that the following requirements have to be met:

$$I_y > I_x > I_z.$$

When this is fulfilled, the candidate function in 5.9 is zero in the equlibrium points when the state vector equals zero. This means the satellite needs to be designed and built accordingly.

To have Lyapunov stability we need to investigate the time derivative of the Lyapunov function candidate. Using the result in chapter 7 in Soglo [25], we have that the time derivative of V in 5.9 is

$$\dot{V} = (\boldsymbol{\omega}_{ob}^b)^T \boldsymbol{\tau}_m^b. \tag{5.10}$$

This result will be further used to evaluate stabilit of the different control algorithms.

5.3 Detumbling Controller

As already mentioned, one need to detumble the satellite in order to start pointing control. The detumbling controller proposed here is the B-dot controller. This is a very common detumbling controller in the small satellite litterature, and as the name implies, it makes use of the derivative of the magnetic field. This is only possible in LEO, since for higher altitudes the geomagnetic field is practically unmeasurable.
The controller was first proposed by Stickler and Alfriend [52], and later improved by Wisniewski [14]:

$$\mathbf{m}^b = -k\dot{\mathbf{B}}^b, \quad k > 0, \tag{5.11}$$

where $\mathbf{m}^b$ is the magnetic control output moment, k is the positive control gain and $\dot{\mathbf{B}}$ is the time derivative of the measured local magnetic field. The thought is, that when the satellite is spinning after launch from the P-POD, the surrounding geomagnetic field will change accordingly to the satellite's angular velocity. Therefore, by measuring the **B**-field, and caluclating its time derivative and using it in the negative feedback control-law, the satellite's energy will decrease, and thus it's spin or angular velocity will decrease as well. Based only on measurements of the magnetic field, it doesn't need any attitude information and is therefore quite straight forward and easy to use, hence it's popularity.

With the expression for the estimated $\dot{\mathbf{B}}$ found in 2.32, the detumbling control law becomes

$$\mathbf{m}^b = -k\hat{\dot{\mathbf{B}}}^b, \quad k > 0, \tag{5.12}$$

where $\hat{\dot{\mathbf{B}}}^b$ is the last calculated estimation of $\dot{\mathbf{B}}^b$ based on a sequence of measurements of the magnetic field $\mathbf{B}$, according to the switching strategy explained above.

When decreasing the angular velocity by means of decreasing the rate of change of the magnetic field, the satellite basically behaves like a compass with three degrees of freedom, and will therefore try to align itself and follow the local geomagnetic field. It follows that the slowest angular velocity one can expect from the B-dot controller is the change of the geomagnetic field. This is indeed not very fast.

When inserting this controller into the expression for the magnetic control torque $\boldsymbol{\tau}_m^b$, and further into the expression for the derivative of the Lyapunov finction we get

$$\dot{V} = -k(\boldsymbol{\omega}_{ob}^b)^T \left[\hat{\dot{\mathbf{B}}}^b \times \mathbf{B}^b\right], \quad k > 0 \tag{5.13}$$

This detumbling controller clearly takes energy out of the system, and one can argue that since this controller will always try to slow the system down, it is asymptotically stable in terms of angular velocity for our case. It is, however difficult to show analytically because we cannot guarantee that $\boldsymbol{\omega}_{ob}^b$, $\hat{\dot{\mathbf{B}}}^b$ or $\mathbf{B}^b$ to be positive semi-definite. Simulations show that it is stable and will slow the satellite down and keep it close to the equilibrium of $\boldsymbol{\omega}_{ob}^b = \mathbf{0}$.

5.4 Nonlinear controller

With our main goal of the mission being to take pictures of the Earth's atmosphere, we need to point the satellite's camera towards the Earth. To do this a reference controller is proposed, and in this section a nonlinear reference controller will be presented. First of all, with our definition of the satellites axes in section 2.1.4, the control goal is to align the *body* frame with the *orbit* frame. This, when expressed with quaternions, means that the quaternion describing the rotation of the satellite (body) relative the orbit frame is desired to be $\mathbf{q} = [1\ 0\ 0\ 0]^T$. From the constraint property of quaternions in equation 2.17, one can see that it will be enough to make sure the $\boldsymbol{\varepsilon}$-vector part of the quaternion is equal to zero, and we will get the desired alignment. This controller presented here is based on the works of Soglo [25] and Tudor [44].

The PD-like controller is given by

$$\boldsymbol{\tau}_d^b = -p\boldsymbol{\varepsilon} - k\boldsymbol{\omega}_{ob}^b, \quad p, k > 0 \tag{5.14}$$

which is found by extending the Lyapunov function candidate in equation (5.9) by using the $\boldsymbol{\varepsilon}$-vector of the quaternions as the error measurement. After some manipulation one arrives at the controller in 5.14. This is the *desired* control law, hence the subscript d. For more see Soglo [25].

This control law will always calculate a gain assuming the magnetorquers are perpendicular to the local geomagnetic field, but this willl seldom be the case. Because of this we project it down to a plane always being perpendicular to the local geomagnetic field. This way we make sure the calculated control gain is the best we can get, and we don't use more gain effort than necessary. Further, it's for the magnetorquers we want to calculate the gain, and the control law becomes

$$m^b = \frac{1}{||\mathbf{B}^b||}\left(-p(\mathbf{B}^b \times \boldsymbol{\varepsilon}) - k(\mathbf{B}^b \times \boldsymbol{\omega}_{ob}^b)\right) \tag{5.15}$$

Inserting the controller in equation (5.14) into the equation for the magnetic control torque we get

$$\boldsymbol{\tau}_m^b = \frac{1}{||\mathbf{B}^b||}\left(-p(\mathbf{B}^b \times \boldsymbol{\varepsilon}) - k(\mathbf{B}^b \times \boldsymbol{\omega}_{ob}^b)\right) \times \mathbf{B}^b. \tag{5.16}$$

This controller is shown in [25] to be uniformly asymptotically stable. [25] also gives a gain requirement

$$k > 8\omega_0^2(I_y - I_z). \tag{5.17}$$

5.5 Linear-Quadratic Regulator

The other reference controller to be investigated is a Linear-Quadratic Regulator (LQR). This controller makes use of the linearized satellite model found in section 3.7.1.

A LQR controller is an optimal linear controller which seeks to meet some control goal while minimizing the effort it takes to achieve this. This type of controller is a popular and well documented controller to use on spacecrafts with magnetic actuation. See for example the work of Wisniewski and Blanke [13], Psiaki [17], Giesselmann [28] and Stray [29]. From its name, it is based on linearized system dynamics while defining a cost function which is to be minimized. Further, this method calculates a gain matrix based on state feedback in order to minimize the cost function. The cost function is often defined as a sum of some key states one want to minimize the deviation of. In our case this is the attitude of the spacecraft, represented by the quaternions describing the deviation between the satellite's body frame and the orbit frame, which we want to be aligned.

The quadratic cost function for the infinite-horizon LQR controller is given by

$$J = \frac{1}{2}\int_0^\infty [\mathbf{x}^T(t)\mathbf{Q}\mathbf{x}(t) + \mathbf{u}^T(t)\mathbf{R}\mathbf{u}(t)]dt \tag{5.18}$$

where $\mathbf{Q}$ and $\mathbf{R}$ are positive semi-definite and diagonal weighting matrices. As mentioned above, in our case $\mathbf{x}$ is the $\boldsymbol{\varepsilon}$-vector of the attitude quaternion, and $\mathbf{u}$ is the magnetic control gain.

Using the linearized system dynamics found in section 3.7.1, the solution to the cost function J is

$$\mathbf{u}(t) = -\mathbf{K}\mathbf{x}(t), \tag{5.19}$$

where optimal gain matrix $\mathbf{K}$ is the solution of the optimal control problem, given by

$$\mathbf{K} = \mathbf{R}^{-1}\mathbf{B}(t). \tag{5.20}$$

Here, $\mathbf{B}(t)$ is the local geomagnetic field mesurement. Further, the soultion to equation (5.20) is found by solving the Riccati equation. The Riccati differential is used to minimalization problems with quadratic cost function, and is given by

$$\dot{\mathbf{P}}(t) = -\mathbf{P}(t)\mathbf{F} - \mathbf{F}^T\mathbf{P}(t) - \mathbf{Q} + \mathbf{P}(t)\mathbf{G}(t)\mathbf{R}^{-1}(t)\mathbf{G}(t)\mathbf{P}, \tag{5.21}$$

where $\mathbf{F}$ and $\mathbf{G}(t)$ is from the linearized model in section 3.7.1. $\mathbf{P}(t)$ converges to a stationary solution if the system is controllable or stabilizable, and we arrive at

$$\mathbf{0} = \mathbf{F}^T\mathbf{P} + \mathbf{P}\mathbf{F} - \mathbf{P}\mathbf{G}\mathbf{R}^{-1}\mathbf{G}^T\mathbf{P} + \mathbf{Q}. \tag{5.22}$$

Now we get the closed-loop system to be:

$$\dot{\mathbf{x}} = (\mathbf{F} - \mathbf{G}\mathbf{K})\mathbf{x}(t) = \mathbf{F}_c\mathbf{x}(t). \tag{5.23}$$

We also need to mention a few words about the weighting matrices $\mathbf{Q}$ and $\mathbf{R}$. With weighting matrices one can choose if one wishes to weigh the states $\mathbf{x}$ more with the matrix $\mathbf{Q}$, or if one wishes to weigh the control inputs $\mathbf{u}$ more with the matrix $\mathbf{R}$. By choosing a larger $\mathbf{Q}$, one will get the poles of the closed-loop system matrix $\mathbf{F}_c$ will have poles further left in the s-plane, and so one will have a quicker response, but with a higher control effort (more gain/power). With a larger $\mathbf{R}$ on the other hand, one will experience a cheaper controller in terms of control effort, but one will get a slower response.

When simulating the system with the LQR controller, the Matlab function *lqr()* will be used to find $\mathbf{K}$; $\mathbf{K} = lqr()$. This takes the linearized system matrices $\mathbf{F}$ and $\mathbf{G}$, and the weighting matrices $\mathbf{Q}$ and $\mathbf{R}$ as inputs. It will only return a solution if the closed loop matrix $\mathbf{F}_c$ is Hurwitz.

We have shown in section 3.5 that our system is uncontrollable. From definition 2.9 we have a stable controller if the closed-loop system matrix $\mathbf{F}_c$ is Hurwitz. Our closed-loop system with the optimal gain matrix $\mathbf{K}$ is indeed Hurwitz, so the controller is stable.

Chapter 6

Simulations and Results

6.1 Simulation

This chapter presents results of the numerical simulations done in Matlab. The MSS toolbox, which is a toolbox developed for marine control, have been used for all simulations because of it's functions for calculating rotation matrices and quaternions. The parameters used in the simulations can be seen in tables 6.1 and 4.1. Further, the matlab function *ode45* have been used to numerically integrate the system using the state matrix

$$\mathbf{x} = \begin{bmatrix} (\boldsymbol{\omega}_{ib}^{b})^T & \eta & \boldsymbol{\varepsilon}^T & \lambda & p \end{bmatrix}^T. \tag{6.1}$$

where λ is the satellite's latitude, and p is the satellite's power consumption. This last state p si not a state wwith any impact on the system, but since $p = \int Wattdt$ it is practical to include it as a state in order to monitor the different control algorithms power consumptions. From this, the wattage use relates to p with $Wh = \frac{p}{3600s}$.

Different initial states are chosen for the different control algorithms. As Tudor's and Soglo's approach, it's more intuitive to state the initial conditions with the satellites angular velocity relative the orbit frame and with the Euler angles, $[\boldsymbol{\omega}_{ib}^{b} \;\; \phi \;\; \theta \;\; \psi]$. These parameters are then rotated and transformed, respectively to become $\boldsymbol{\omega}_{ib}^{b}$, η and $\boldsymbol{\varepsilon}$, using the functions *euler2q* and *Rquat* from the MSS toolbox.

All the Matlab simulation files can be found in appendix A. The maximum internal dipole moment found is a *very* rough estimate based loosely on how much current each module can use at any time, from [34].

Parameters	Value
Mass [kg]	2.6
Size [mm]	$100 \times 100 \times 200$
Moments of inertia [kgm^2]	$I_x = 0.33$, $I_y = 0.33002$, $I_z = 0.11$
Voltage [V]	5
Maximum current [mA]	113
Maximum magnetic moment [Am^2]	0.2596
Maximum aerodynamic torque [Nm]	$1.977 \cdot 10^{-8}$
Maximum solar radiation torque [Nm]	$4.124 \cdot 10^{-9}$
Maximum internal dipole moment [Am^2]	$7.2 \cdot 10^{-4}$
Desired Euler angles [deg]	$\phi = 0$, $\theta = 0$, $\psi = 0$
Desired angular velocity [rad/s]	$\boldsymbol{\omega}_{ob}^{b} = \mathbf{0}$

Table 6.1: Table of simulation parameters.

6.2 Results

6.2.1 Magnetometer measurement filters

The results after simulating the different magnetometer measurement filters from section 2.6 are presented here. All the testing of the filters have been done in the detumbling phase, since the change in measured geomagnetic fiekd is expected to be greatest here.

Low-pass filter

Here are results of the low-pass filter from section 2.6.1, used for the measured local geomagnetic field.

$$\hat{\mathbf{B}}_k = a_1 \hat{\mathbf{B}}_{k-1} + b_1 \mathbf{B}_k. \tag{6.2}$$

The plots of the results can be seen in figures 6.1 and 6.2.

Numerical differentiator

Here are results of the numerical differentiator from section 2.6.2 used for the measured local geomagnetic field.

$$\hat{\dot{\mathbf{B}}}_k = \frac{\mathbf{B}_{k+1} - \mathbf{B}_{k-1}}{2\Delta t}. \tag{6.3}$$

The plots of the results can be seen in figures 6.3 and 6.4.

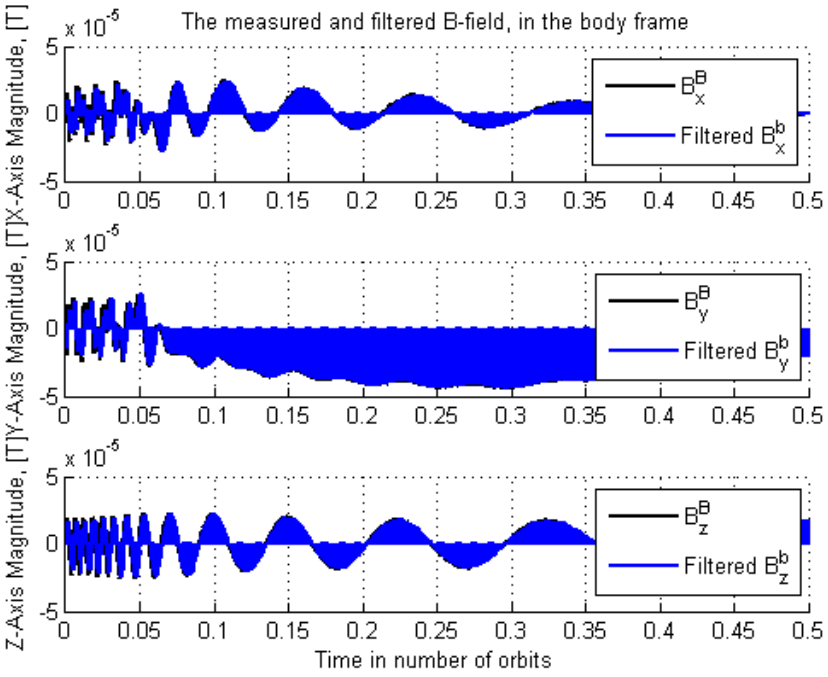

Figure 6.1: Plot of low-pass filter for geomagnetic measurements.

Estimator

Here are results of the estimator from section 2.6.3 used to estimate the derivative of the measured local geomagnetic field.

$$\hat{\dot{\mathbf{B}}}_k = a_2 \hat{\dot{\mathbf{B}}}_{k-1} + b_2 \left(\mathbf{B}_k - \mathbf{B}_{k-1} \right) \tag{6.4}$$

The plots of the results can be seen in figures 6.5 and 6.6.

6.2.2 Detumbling

Simulation of the detumbling controller, with all disturbance torques included. The detumble control law was found in section 5.3 and is given by:

$$\mathbf{m}^b = -k \hat{\dot{\mathbf{B}}}^b, \quad k > 0 \tag{6.5}$$

and is using the estimated value of the derivative of the measured local geomagnetic field. The control law results in the control torque

$$\boldsymbol{\tau}_m^b = -\frac{k}{||\mathbf{B}^b||} \left(\hat{\dot{\mathbf{B}}}^b \right) \times \mathbf{B}^b. \tag{6.6}$$

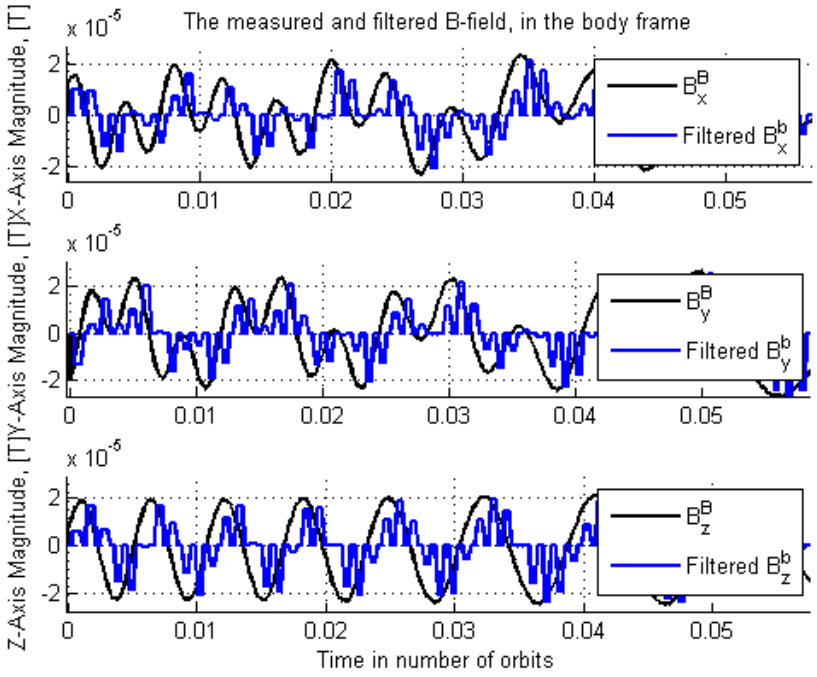

Figure 6.2: Zoomed plot of low-pass filter for geomagnetic measurements.

The control gain used in 6.6 is $d = 4 \cdot 10^{-5}$.

The initial values for detumbling can be seen in table 6.2.

The results of the simulations of the detumbling controller can be seen in figures 6.7 and 6.8, depicting the angular velocity of the satellite w.r.t. to the orbit frame and the power consumption of the controller, respectively. The total disturbance torques from section 3.6 are:

$$\boldsymbol{\tau}_d^b = \boldsymbol{\tau}_g^b + \boldsymbol{\tau}_a^b + \boldsymbol{\tau}_s^b + \boldsymbol{\tau}_{m,d}^b, \tag{6.7}$$

and can be seen in figure 6.9.

$$\begin{aligned} \boldsymbol{\omega}_{ob}^b &= [0.1 \quad -0.2 \quad 0.1]^T \\ \phi &= 180^\circ \\ \theta &= 30^\circ \\ \psi &= -75^\circ \\ \lambda &= 0^\circ \\ p &= 0\ J \end{aligned}$$

Table 6.2: Initial values for the detumbling simulations.

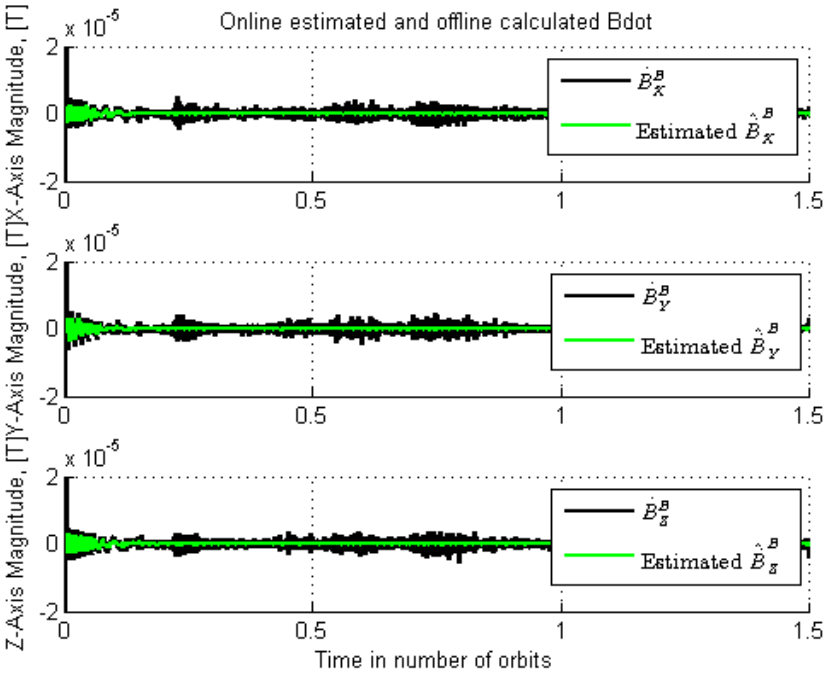

Figure 6.3: Plot of the numerical differentiation by central difference method for geomagnetic measurements

6.2.3 Nonlinear control

Here the simulations using the nonlinear reference controller from section 5.4 is presented. The control law is:

$$\boldsymbol{\tau}_m^b = \frac{1}{||\mathbf{B}^b||}\left(-p(\mathbf{B}^b \times \boldsymbol{\varepsilon}) - k(\mathbf{B}^b \times \boldsymbol{\omega}_{ob}^b)\right) \times \mathbf{B}^b. \tag{6.8}$$

In the following subsections the results of using the controller when subjected to different disturbance torques is presented.

Gravity gradient as only disturbance

The nonlinear reference controller simulated with gravity-gradient torques as only disturbance torque is presented here. In figures 6.10 and 6.11 one can see plots of the satellite's roll, pitch and yaw angles, and the controller's power consumption, respectively. The gravity-gradient torque can be seen in figure 6.12.

The control gains used in here is $p = 3 \cdot 10^{-9}$ and $k = 6 \cdot 10^{-6}$.

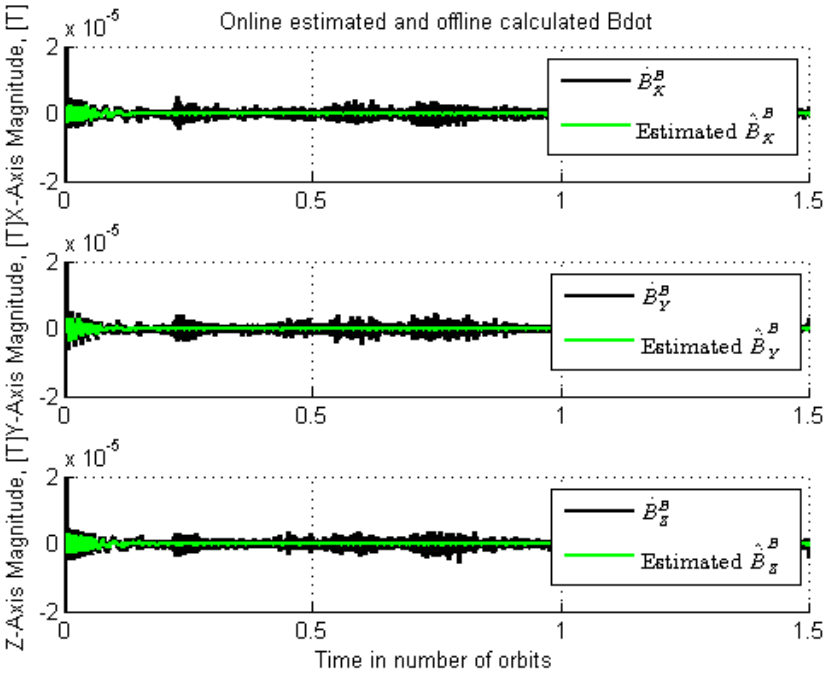

Figure 6.4: Zoomed plot of the numerical differentiation by central difference method for geomagnetic measurements

The initial values for the nonlinear reference controller simulations with only gravity-gradient disturbance can be seen in table 6.3. The angular velocity initial values are the final values from the detumble controller.

Gravity-gradient and aerodynamic disturbance torques

The nonlinear reference controller simulated with gravity-gradient and aerodynamic torques is presented here. In figures 6.13 and 6.14 one can see plots of the satellite's roll, pitch and yaw angles, and the controller's power consumption, respectively. The aerodynamic torque can be seen in figure 6.15.

The control gains used in here is $p = 1.5 \cdot 10^{-9}$ and $k = 6 \cdot 10^{-6}$.

The initial values for the nonlinear reference controller simulations with gravity-gradient and aerodynamic disturbances can be seen in table 6.4. The angular velocity initial values are the final values from the detumble controller.

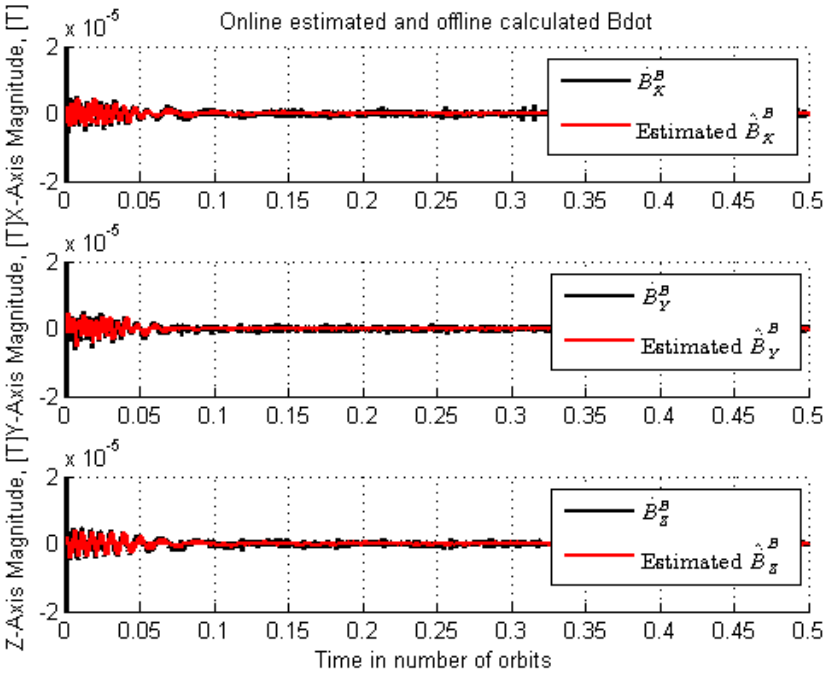

Figure 6.5: Plot of the digital estimator used for geomagnetic measurements

Gravity-gradient, aerodynamic and solar disturbance torques

The nonlinear reference controller simulated with gravity-gradient, aerodynamic and solar disturbance torques is presented here. In figures 6.16 and 6.17 one can see plots of the satellite's roll, pitch and yaw angles, and the solar radiation disturbance torque, respectively.

The control gains used in here is $p = 1.5 \cdot 10^{-9}$ and $k = 6 \cdot 10^{-6}$.

The initial values for the nonlinear reference controller simulations with gravity-gradient, aerodynamic and disturbances can be seen in table 6.5. The angular ve-

$$\begin{aligned} \boldsymbol{\omega}_{ob}^{b} &= [0.001056 \;\; 0.000722 \;\; -0.000339]^T \\ \phi &= 180^\circ \\ \theta &= 30^\circ \\ \psi &= -75^\circ \\ \lambda &= 0^\circ \\ p &= 0 \; J \end{aligned}$$

Table 6.3: Initial values for the nonlinear reference controller simulations with only gravity-gradient disturbance.

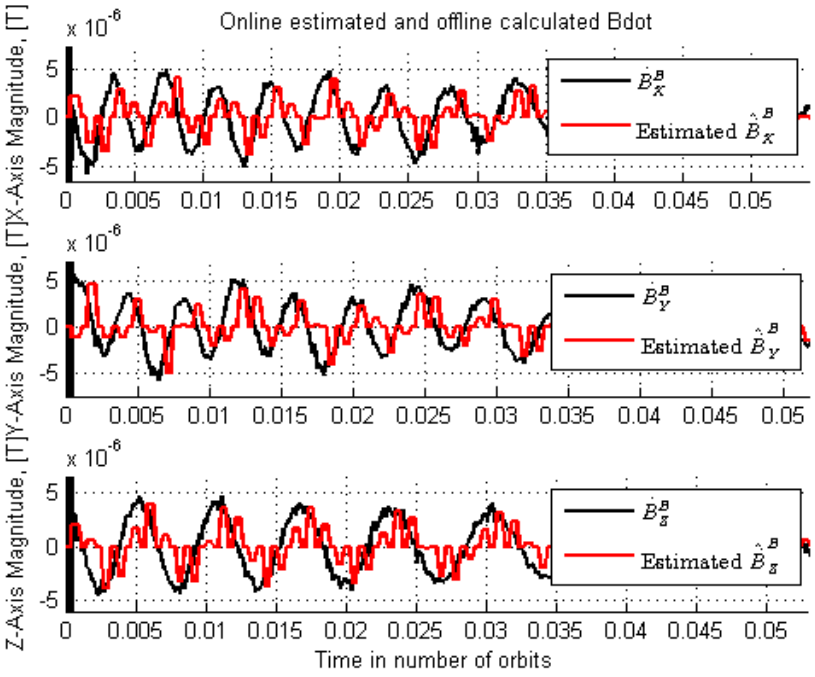

Figure 6.6: Zoomed plot of the digital estimator used for geomagnetic measurements

locity initial values are the final values from the detumble controller.

Nonlinear control with integral action

The simulation of the nonlinear reference controller with integral action is presented here. When adding more disturbance torques than the gravity-gradient it's apparent that we get a constant deviation. The reason for this is that when adding e.g. aerodynamic torque, after the satellite's spin is decreased we get a more or less

$$\begin{aligned} \boldsymbol{\omega}_{ob}^{b} &= [0.001056 \;\; 0.000722 \;\; -0.000339]^T \\ \phi &= 180^\circ \\ \theta &= 30^\circ \\ \psi &= -75^\circ \\ \lambda &= 0^\circ \\ p &= 0\ J \end{aligned}$$

Table 6.4: Initial values for the nonlinear reference controller simulations with gravity-gradient and aerodynamic disturbances.

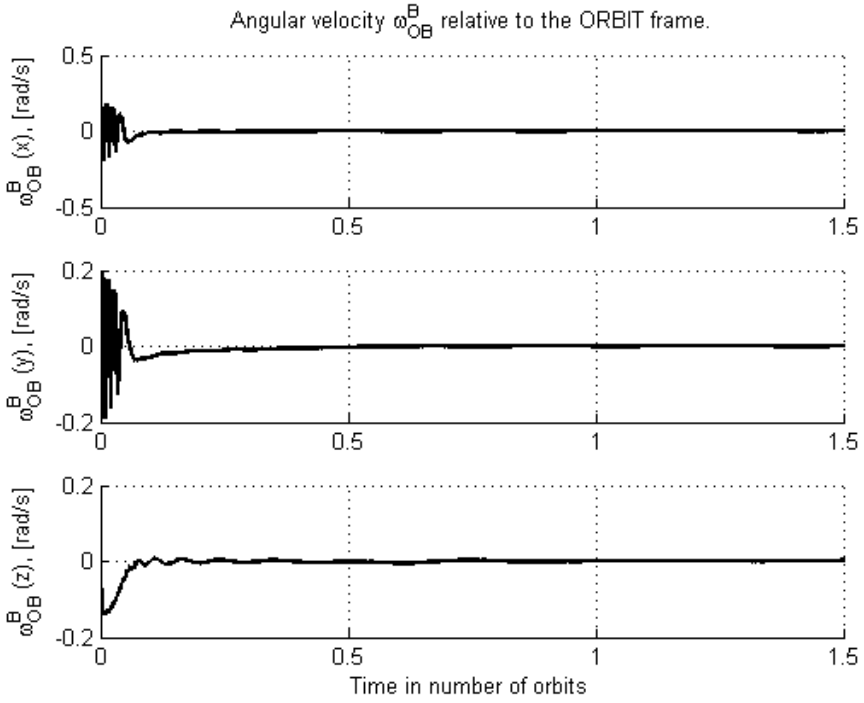

Figure 6.7: Plot of angular velocity $\boldsymbol{\omega}_{ob}^{b}$ after detumbling.

constant disturbance effect, see figure 6.15. Hence the equlibrium of the satellite is now shifted and we get a steady-state offset as a result. Because of this the nonlinear controller in 5.14 were augmented with integral action in an attempt to get rid of the offset.

$$\boldsymbol{\tau}_d^b = -p\boldsymbol{\varepsilon} - k\boldsymbol{\omega}_{ob}^b - i\boldsymbol{\varepsilon}_e, \;\; p, k, i > 0 \tag{6.9}$$

where $\boldsymbol{\varepsilon}_e$ is the accumulated error of the $\boldsymbol{\varepsilon}$-vector of the quaternion describing the satellites position w.r.t. to the orbit frame. Because of the quaternion properties from equation (2.17), and since the desired attitude quaternion is $q = [1 \;\; 0 \;\; 0 \;\; 0]^T$,

$$\begin{aligned}
\boldsymbol{\omega}_{ob}^b &= [0.001056 \;\; 0.000722 \;\; -0.000339]^T \\
\phi &= 20^\circ \\
\theta &= 20^\circ \\
\psi &= 20^\circ \\
\lambda &= 0^\circ \\
p &= 0 \; J
\end{aligned}$$

Table 6.5: Initial values for the nonlinear reference controller simulations with gravity-gradient, aerodynamic and solar disturbances.

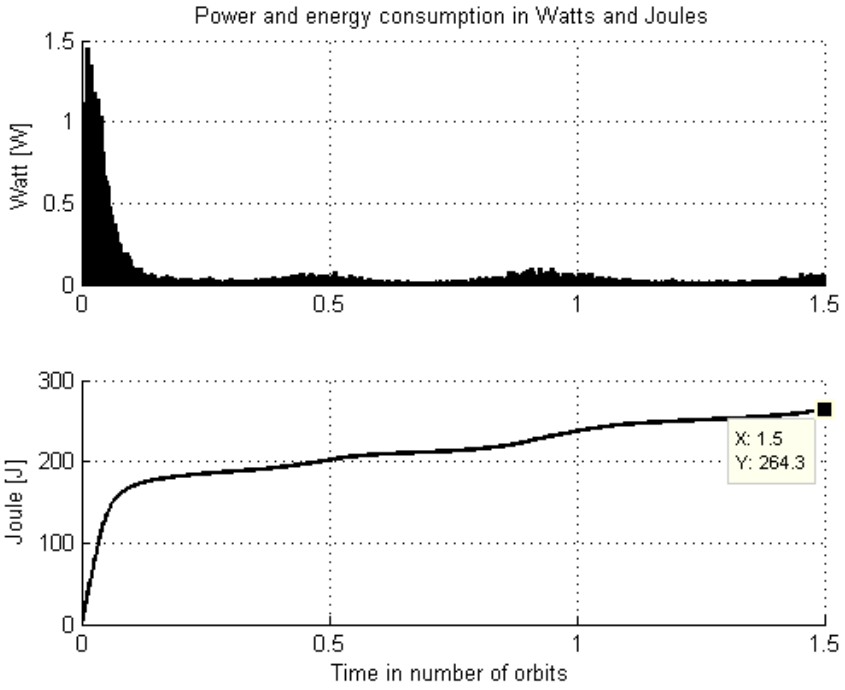

Figure 6.8: Plot of energy and power consumption after detumbling.

we get that the quaternion error (or rather the "epsilon"-error) is given by

$$\varepsilon_e = \int \varepsilon dt. \tag{6.10}$$

The system is simulated with the nonlinear reference controller augmented with integral action, and with gravity-gradient and aerodynamic disturbance torues, in order to have an offset to remove. In figures 6.18 one can see plots of the satellite's roll, pitch and yaw angles when the integral gain equals $i = 1 \cdot 10^{-12}$. In figure 6.19 one can see the same, but now with an integral gain of $i = 5 \cdot 10^{-11}$. The other two gains are $p = 1.5 \cdot 10^{-9}$ and $k = 6 \cdot 10^{-6}$.

The initial values for the nonlinear reference controller augmented with integral action, and with gravity-gradient and aerodynamic disturbances can be seen in table 6.6. The angular velocity initial values are the final values from the detumble controller.

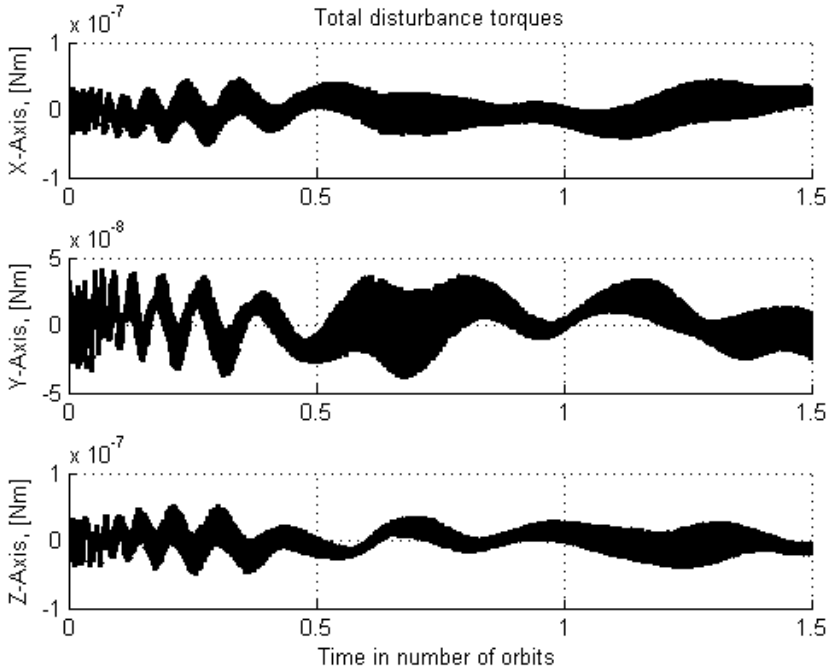

Figure 6.9: Plot of total disturbance torques while detumbling.

$$
\begin{aligned}
\boldsymbol{\omega}_{ob}^{b} &= [0.001056 \;\; 0.000722 \;\; -0.000339]^T \\
\phi &= 180^\circ \\
\theta &= 30^\circ \\
\psi &= -75^\circ \\
\lambda &= 0^\circ \\
p &= 0 \; J
\end{aligned}
$$

Table 6.6: Initial values for the nonlinear reference controller simulations with integral action.

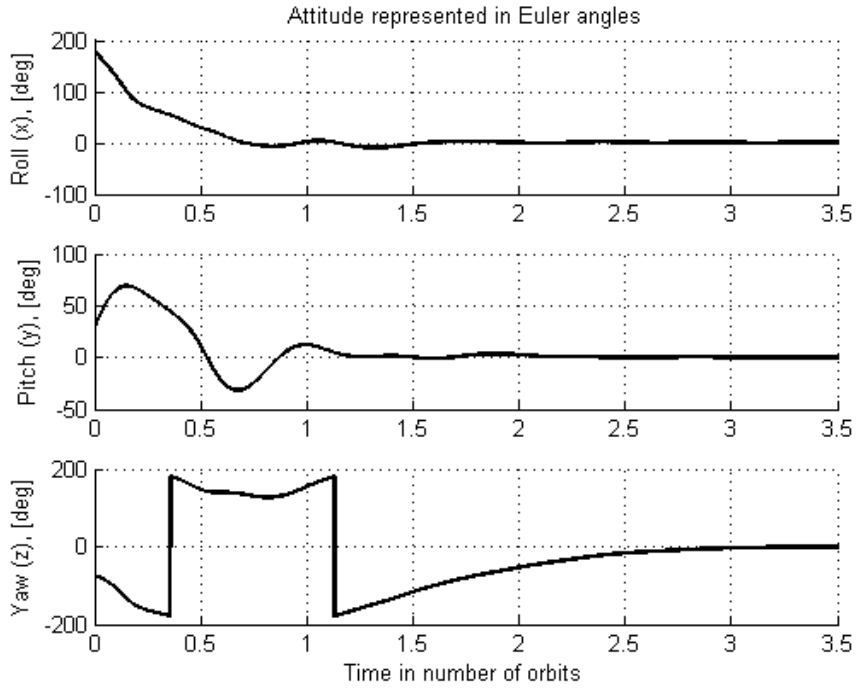

Figure 6.10: Plot of the Euler angles after nonlinear control, gravity as only disturbance.

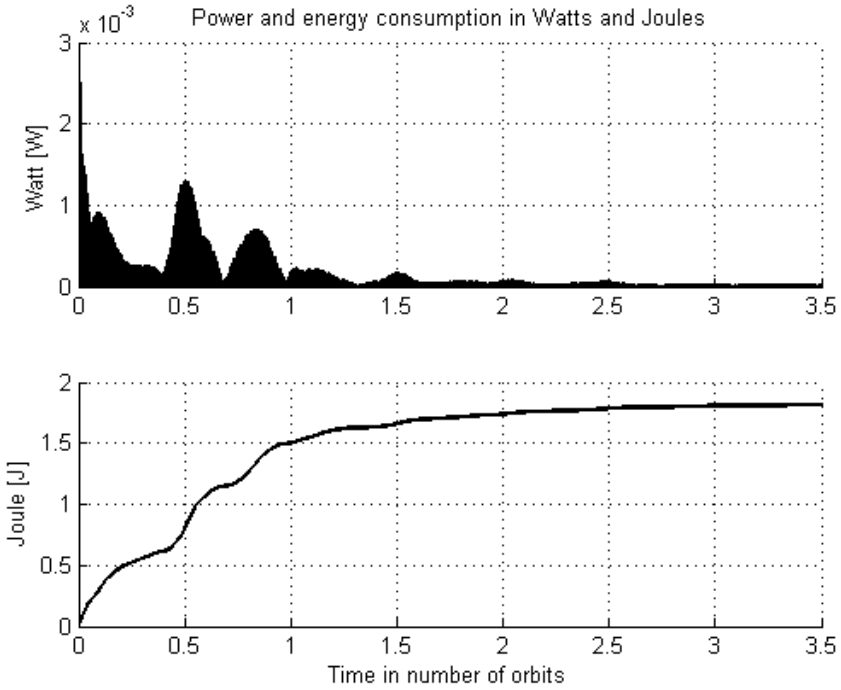

Figure 6.11: Plot of energy and power consumption after nonlinear control, gravity as only disturbance.

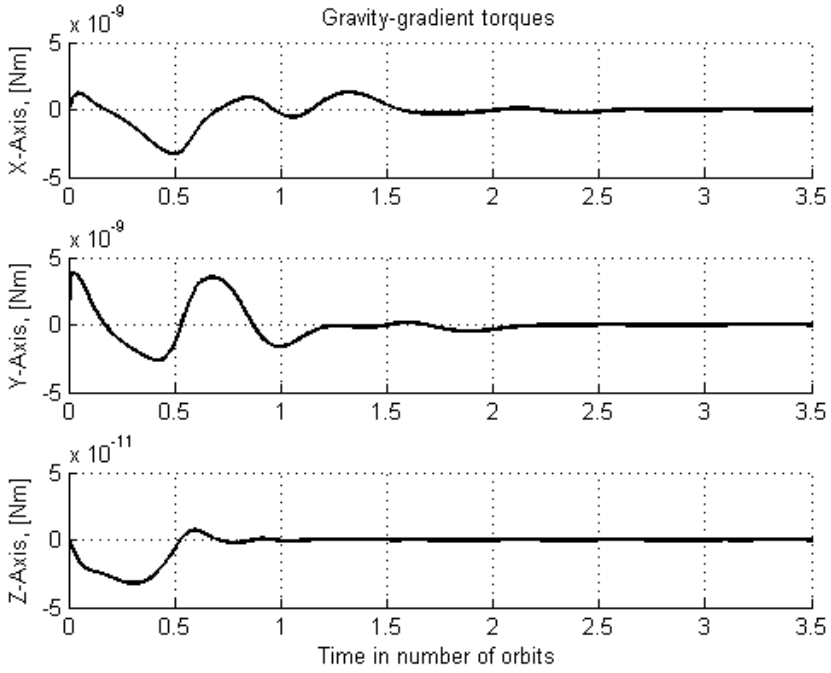

Figure 6.12: Plot of gravity-gradient torques while simulating nonlinear reference controller.

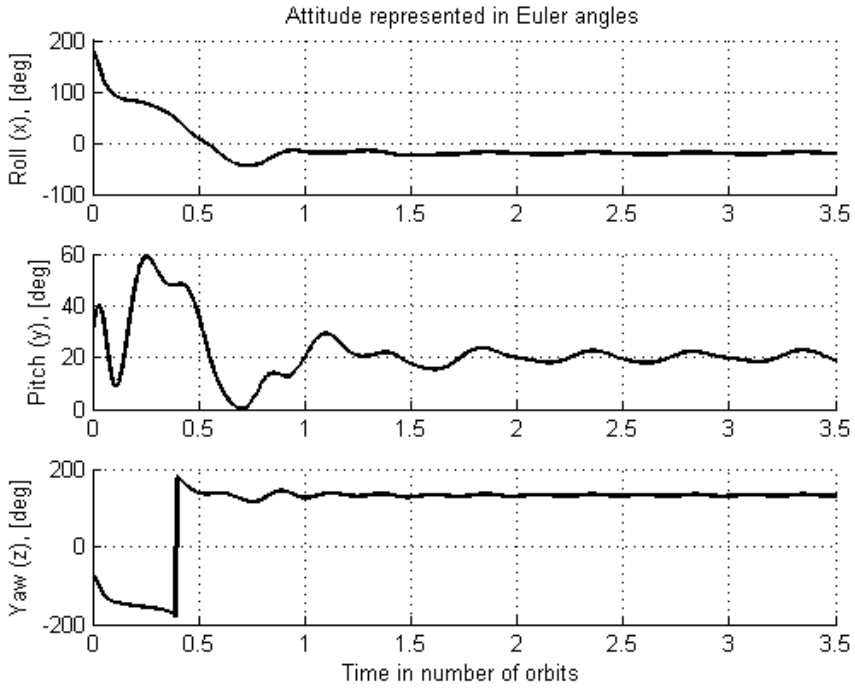

Figure 6.13: Plot of the Euler angles after nonlinear control, with gravity and aerodynamic disturbance torques.

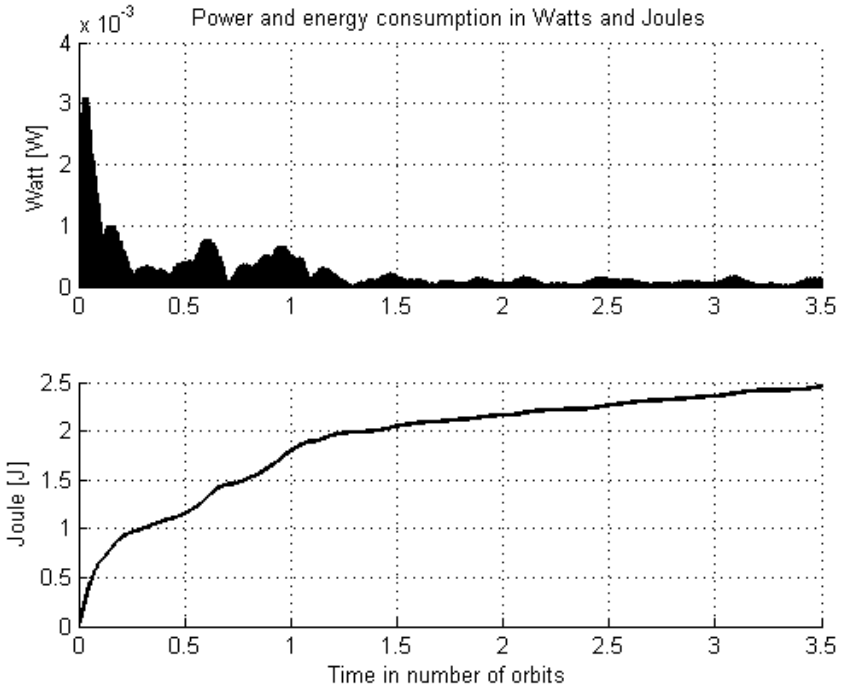

Figure 6.14: Plot of energy and power consumption after nonlinear control, with gravity and aerodynamic disturbance torques.

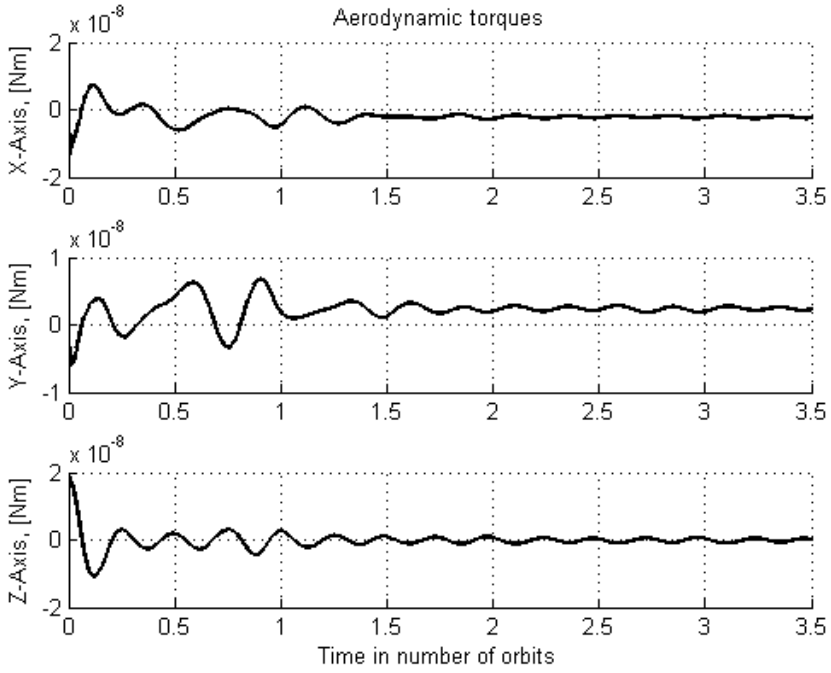

Figure 6.15: Plot of the aerodynamic disturbance torques while simulating nonlinear control.

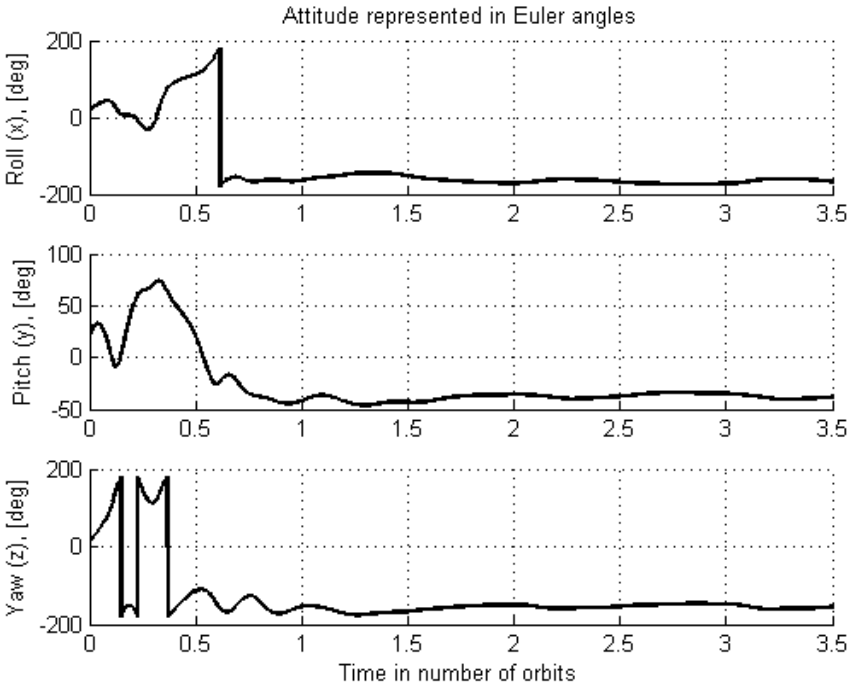

Figure 6.16: Plot of the Euler angles after nonlinear control, with gravity, aerodynamic and solar disturbance torques.

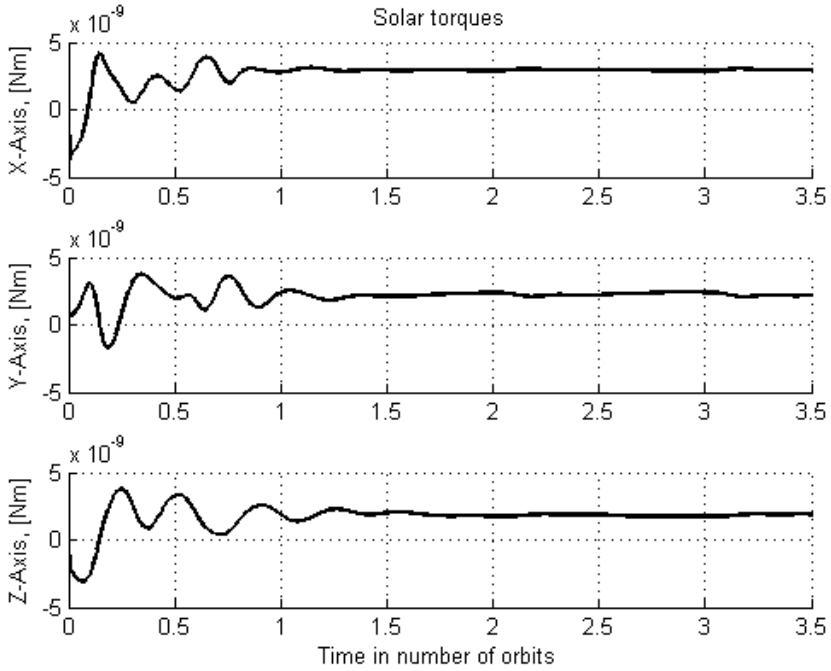

Figure 6.17: Plot of the solar disturbance torques while simulating nonlinear control, with gravity, aerodynamic and solar disturbance torques.

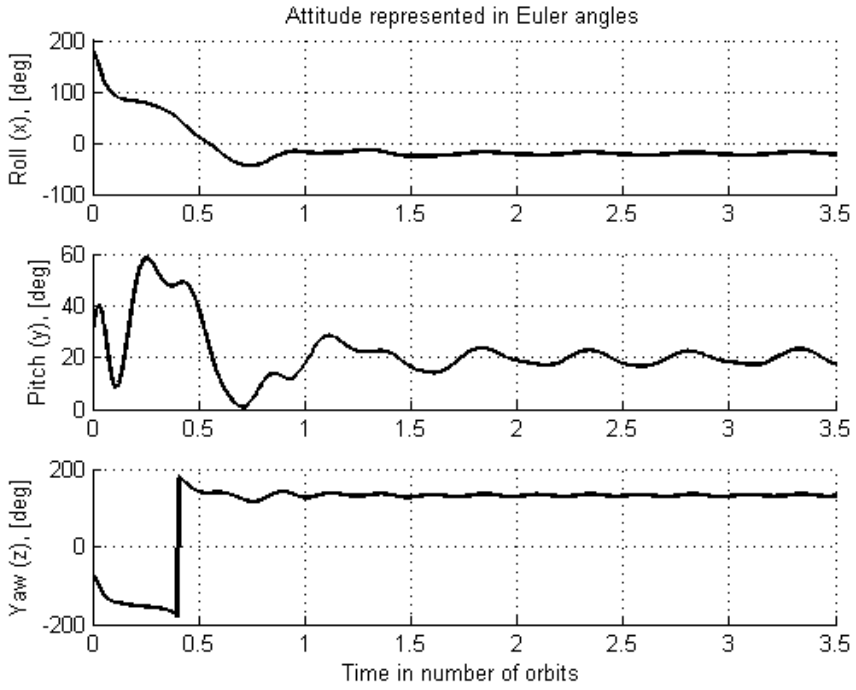

Figure 6.18: Plot of the Euler angles after nonlinear control with integral action, $i = 1 \cdot 10^{-12}$.

6.2.4 Optimal control

Here the simulations using the optimal LQR reference controller from section 5.5 is presented. In the following subsections the results of using the controller when subjected to different disturbance torques is presented.

Gravity-gradient as only disturbance

The LQR reference controller simulated with gravity-gradient torques as only disturbance torque is presented here.

The initial values for the optimal LQR controller with gravity-gradient as only disturbance can be seen in table 6.7. The angular velocity initial values are the final values from the detumble controller. The roll, pitch and yaw initial values are the final values after using the nonlinear controller with gravity-gradient and aerodynamic disturbance torques from figure 6.13 in section 6.2.3.

In figures 6.20 and 6.21 one can see plots of the satellite's roll, pitch and yaw angles, and the controller's power consumption, respectively.

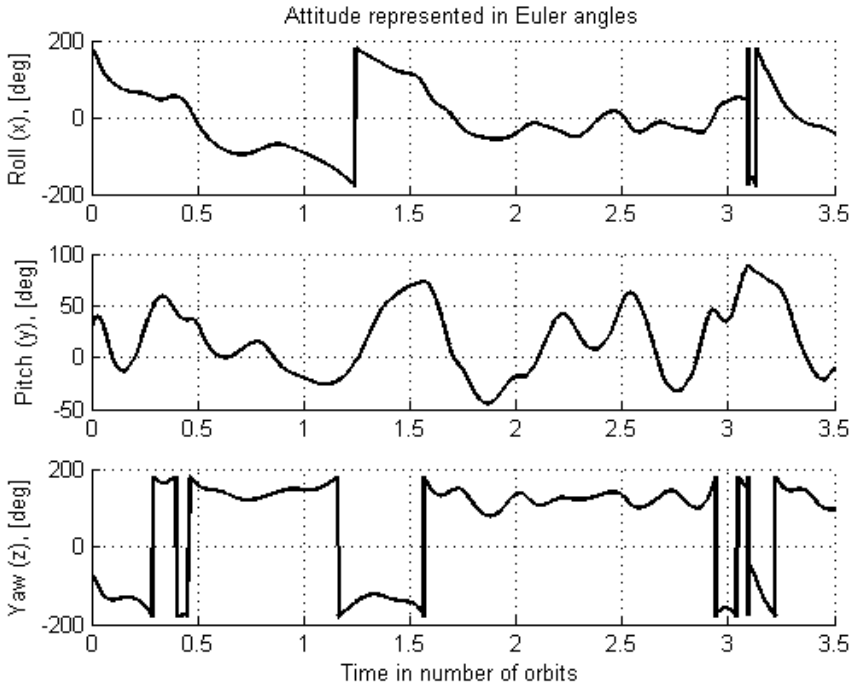

Figure 6.19: Plot of the Euler angles after nonlinear control with integral action, $i = 5 \cdot 10^{-11}$.

Gravity-gradient and aerodynamic disturbance torques

The LQR reference controller simulated with gravity-gradient and aerodynamic torques is presented here.

The initial values for the optimal LQR controller with gravity-gradient and aerodynamic disturbances can be seen in table 6.8. The angular velocity initial values are the final values from the detumble controller. The roll, pitch and yaw initial values are picked to be close to the equilibrium, this being a linear controller.

$$\begin{aligned} \boldsymbol{\omega}_{ob}^{b} &= [0.001056 \;\; 0.000722 \;\; -0.000339]^T \\ \phi &= -20^\circ \\ \theta &= 20^\circ \\ \psi &= 130^\circ \\ \lambda &= 0^\circ \\ p &= 0 \; J \end{aligned}$$

Table 6.7: Initial values for the optimal LQR reference controller simulations with gravity-gradient as only disturbance.

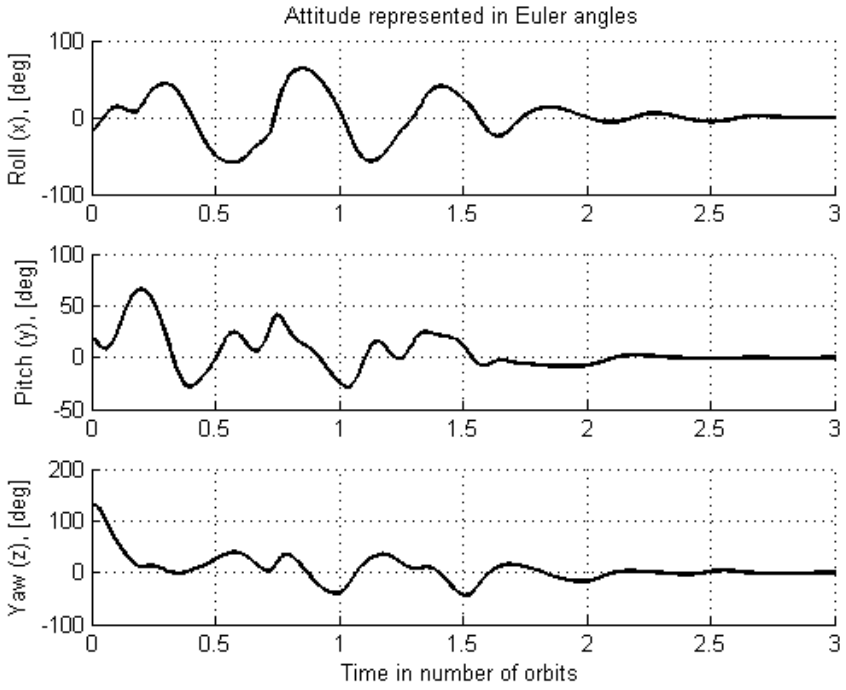

Figure 6.20: Plot of the Euler angles after optimal control, gravity as only disturbance.

In figures 6.22 one can see plot of the satellite's roll, pitch and yaw angles.

$$
\begin{aligned}
\boldsymbol{\omega}_{ob}^{b} &= [0.001056 \;\; 0.000722 \;\; -0.000339]^T \\
\phi &= 5^\circ \\
\theta &= -9^\circ \\
\psi &= 6^\circ \\
\lambda &= 0^\circ \\
p &= 0\ J
\end{aligned}
$$

Table 6.8: Initial values for the optimal LQR reference controller simulations with gravity-gradient and aerodynamic disturbances.

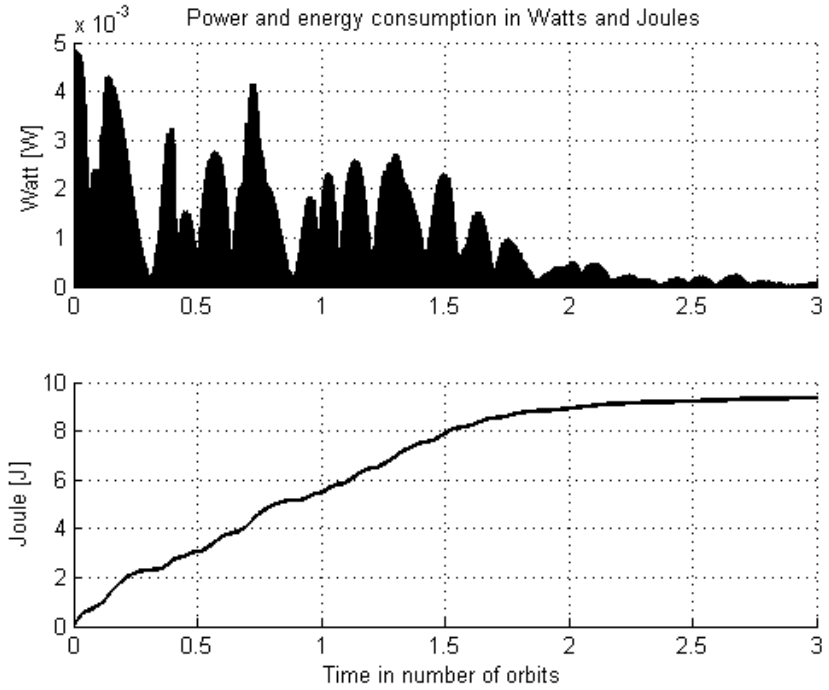

Figure 6.21: Plot of energy and power consumption after optimal control, gravity as only disturbance.

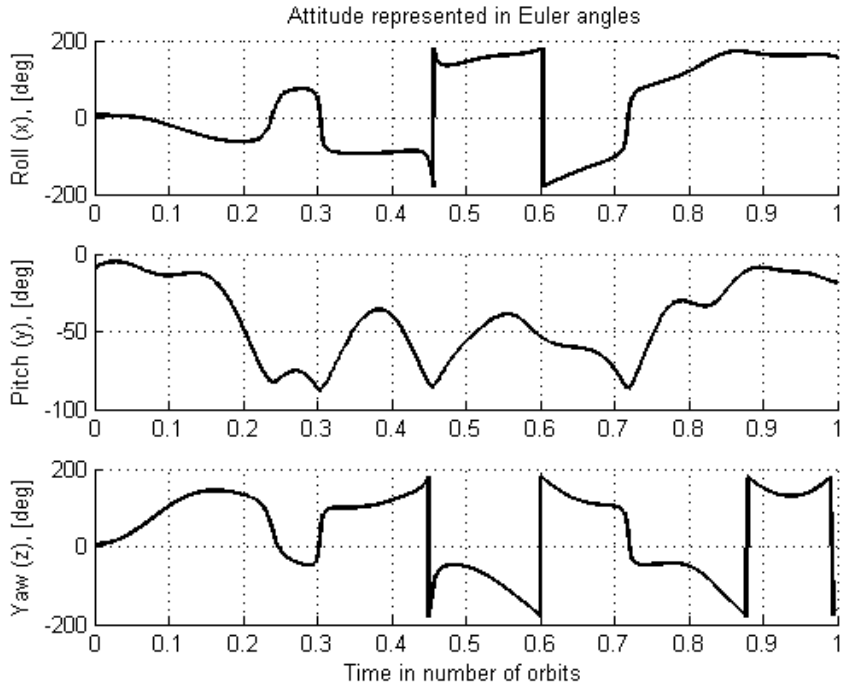

Figure 6.22: Plot of the Euler angles after optimal control, with gravity and aerodynamic disturbance torques.

Chapter 7

Discussion and Conclusion

7.1 Discussion

All the filters; the low-pass filter, the central difference method for numerical differentiation and the estimator worked nicely, and since they were being tested with a higher noise than can be expected from the magnetometer to be used, i think it is safe to say they all gave satisfactory results.

The detumbling controller worked satisfactory as well, and so did the nonlinear reference controller to a certain degree. When subjected to more than only the gravity-gradient torques, the nonlinear controller didn't meet the desired angles. However, with aerodynaic torques added it were able to settle the satellite with an offset for the roll, pitch and yaw angles of about -20, 20 and 130 degrees, respectively. Since the last angle, the yaw, is the angle about the z-axis, this doesn't have an impact on wether or not we're able to take a picture of the Earth, as long as it's stable at that angle - which it is. From [37] it is argued that with an offset of less than 50 degrees, one is still able to take good pictures of the Earth's atmosphere with the infrared camera. When adding more disturbances, the solar radiation torque, the controller didn't work and weren't able to stabilize the satellite at any angle. The attempt with integral action augmentation of the nonlinear controller proved to be unsatisfactory as well.

The optimal LQR regulator worked nicely when only being subjected to gravity-gradient torques, and also gave good results with rather high initial roll, pitch and yaw values. When adding aerodynamic torques, it failed.

7.2 Conlusion

A low pass filter for magnetometer measurements were found to be working satisfactory.
As for finding the derivative of the local geomagnetic field, the estimator is suggested. This is because if one were to use the numerical differentiator, its necessary to filter the measurement first, and this will take too much time.

The detumbling controller worked still when being subjected to all the disturbance torques, and managed to settle the satellite at an angular velocity very close to zero. The detumbling controller utilizing the estimated value of the derivative of the local geomagnetic field can be said to work as fine as the one found by Tudor [44].

The nonlinear controller worked well with gravity-gradient as the only disturbance, but with aerodynamic disturbance torques there were a ste, and even more when solar radiation torques were included. However, it can be used since one can still take pictures of the Earth's atmosphere with a 20 degrees deiation in roll and pitch, and yaw does not have an impact of the picture to be taken as long as it doesn't drift. Integral action were tried out in an attempt to remove the deviatioins, but it was unsuccessfull.

The optimal LQR controller worked well with gravity-gradient as the only disturbance as well, but with aerodynamic disturbance torques the results were unsatisfactory. To use the proposed detumbling algorithm first and then the nonlinear controller for big deviations in order to be able to use the optimal LQR controller, can not be said to guarantee good results.

The suggestion is to use the detumbling controller with the estimated B-dot as proposed, with the nonlinear PD-like controller, and maybe see if there can be done some mmore investigations on methods to remove the steady-state offset when subjected to disturbance torques.

If anything, it have been pointed out that no matter how small any disturbance torque is, since every force and torque acting on a satellite in space is very small, it has an impact and consequences.

Bibliography

[1] http://hincube.hin.no. The HiNCube project webpage.

[2] http://www.cubestar.no/. The CubeSTAR webpage.

[3] http://www.rocketrange.no/?page_id=254. The ANSAT program webpage.

[4] "Cubesat design specification rev. 12." http://www.cubesat.org/images/developers/cds_rev12.pdf, 2011.

[5] R. Brikeland, "NUTS-1 Mission Statement," 2011.

[6] http://nuts.cubesat.no/. The NUTS project webpage.

[7] J. White, F. Shigemoto, and K. Bourquin, "Satellite attitude control utilizing the earth's magnetic field," tech. rep., DTIC Document, 1961.

[8] J. Wen and K. Kreutz-Delgado, "The attitude control problem," *Automatic Control, IEEE Transactions on*, vol. 36, no. 10, pp. 1148–1162, 1991.

[9] M. Pittelkau, "Optimal periodic control for spacecraft pointing and attitude determination," *Journal of Guidance Control Dynamics*, vol. 16, pp. 1078–1084, 1993.

[10] Y. Chen and S. Lo, "Sliding-mode controller design for spacecraft attitude tracking maneuvers," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 29, no. 4, pp. 1328–1333, 1993.

[11] W. Steyn, "Fuzzy control for a non-linear mimo plant subject to control constraints," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 24, no. 10, pp. 1565–1571, 1994.

[12] R. Wisniewski and M. Blanke, "Three-axis satellite attitude control based on magnetic torquing," in *13th IFAC World Congress*, pp. 1–36, Citeseer, 1996.

[13] R. Wisniewski and M. Blanke, "Fully magnetic attitude control for spacecraft subject to gravity gradient," *Automatica*, vol. 35, no. 7, pp. 1201–1214, 1999.

[14] R. Wisniewski, "Satellite attitude control using only electromagnetic actuation," *Department of Control Engineering, Aalborg University*, 1996.

[15] P. Wang, Y. Shtessel, and Y. Wang, "Satellite attitude control using only magnetic torquers," in *AIAA Guidance, Navigation, and Control Conference and Exhibit, Boston, USA*, 1998.

[16] R. Wisniewski, "Linear time-varying approach to satellite attitude control using only electromagnetic actuation," *Journal of Guidance Control and Dynamics*, vol. 23, no. 4, pp. 640–647, 2000.

[17] M. Psiaki, "Magnetic torquer attitude control via asymptotic periodic linear quadratic regulation," *Journal of Guidance, Control, and Dynamics*, vol. 24, no. 2, pp. 386–394, 2001.

[18] M. Lovera and A. Astolfi, "Spacecraft attitude control using magnetic actuators," *Automatica*, vol. 40, no. 8, pp. 1405–1414, 2004.

[19] E. Silani and M. Lovera, "Magnetic spacecraft attitude control: a survey and some new results," *Control Engineering Practice*, vol. 13, no. 3, pp. 357–371, 2005.

[20] R. Kristiansen, "Attitude control of a microsatellite," Master's thesis, Department of Engineering Cybernetics (ITK), Norwegian University of Science and Technology (NTNU), Trondheim, Norway, 2000.

[21] K. Fauske, "Attitude stabilization of an underactuated rigid spacecraft," Master's thesis, Department of Engineering Cybernetics (ITK), Norwegian University of Science and Technology (NTNU), Trondheim, Norway, 2003.

[22] B. Busterud, "Orienteringsregulering av mikrosatellitter," Master's thesis, Norwegian University of Science and Technology, 2003.

[23] E. Øverby, "Attitude control for the norwegian student satellite ncube," *Norwegian University of Science and Technology. Department of Engineering Cybernetics*, 2004.

[24] E. Narverud, R. Birkeland, and E. K. Holm, "Design of small student satellite," Project work.

[25] P. Soglo, "3-aksestyring av gravitasjonsstabilisert satellitt ved bruk av magnetspoler," Master's thesis, Department of Engineering Cybernetics (ITK), Norwegian University of Science and Technology (NTNU), Trondheim, Norway, 1994.

[26] K. L. Makovec, "A nonlinear magnetic controller for three-axis stability of nanosatellites," Master's thesis, Faculty of the Virginia Polytechnic Institute and State University, 2001.

[27] B. Ø. Andresen, C. Grøn, R. H. Knudsen, C. Nielsen, K. K. Sørensen, and D. Taagaard, "Attitude Control System for AAUSAT-II," Master's thesis, Aalborg University, 2005.

[28] J. Giesselmann, "Development of an active magnetic attitude determination and control system for picosatellites on highly inclined circular low earth orbits," *Master of Engineering RMIT University*, 2006.

[29] F. Stray, "Attitude control of a nano satellite," Master's thesis, University of Oslo, 2010.

[30] K. Rensel, "An attitude detumbling system for the cubestar nano satellite," Master's thesis, University of Oslo, 2011.

[31] M. L. Volstad, "Internal data bus of a small student satellite," Master's thesis, Norwegian University of Science and Technology (NTNU), Trondheim, Norway, 2011.

[32] D. E. Holmstrøm, "Software and software architecture for a student satellite," 2011. Project report at the Norwegian University of Science and Technology (NTNU), Trondheim, Norway.

[33] D. de Bruyn, "Power distribution and conditioning for a small student satellite design of the nuts backplane & eps module," Master's thesis, Norwegian University of Science and Technology (NTNU), Trondheim, Norway, 2011.

[34] L. E. Jacobsen, "Electrical power system of the ntnu test satellite.," Master's thesis, Norwegian University of Science and Technology, Department of Electronics and Telecommunications, 2012.

[35] S. Marholm, "Antenna systems for nuts," Master's thesis, Norwegian University of Science and Technology (NTNU), Trondheim, Norway, 2012.

[36] V. Visockas, "Access control and securing of the nuts uplink," 2011. Project report at the Norwegian University of Science and Technology (NTNU), Trondheim, Norway.

[37] S. S. Rønning, "Optimizing an infrared camera for observing of atmospheric gravity waves from a cubesat platform," Master's thesis, Norwegian University of Science and Technology, 2012.

[38] E. Meland, "Internal wireless bus in student satellite experiment," 2011. Project report at the Norwegian University of Science and Technology (NTNU), Trondheim, Norway.

[39] K. I. M. Rokstad, "Investigation of using composite materials for the cubesat primary structure," in *European Cubesat Symposium, Brussels, February 2012*, 2012.

[40] B. H. Stenhaug, "Antenna system for a ground station communicating with the ntnu test satellite (nuts)," Master's thesis, Norwegian University of Science and Technology (NTNU), Trondheim, Norway, 2011.

[41] K. L. Jenssen and K. H. Yabar, "Development, implementation and testing of two attitude estimation methods for cube satellites," Master's thesis, Department of Engineering Cybernetics (ITK), Norwegian University of Science and Technology (NTNU), Trondheim, Norway, 2011.

[42] T. B. Rinnan, "Development and comparison of estimation methods for attitude determination," Master's thesis, Department of Engineering Cybernetics (ITK), Norwegian University of Science and Technology (NTNU), Trondheim, Norway, 2012.

[43] F. S. Holberg, "Design of attitude estimation and control system for a cube satellite," Master's thesis, Department of Engineering Cybernetics (ITK), Norwegian University of Science and Technology (NTNU), Trondheim, Norway, 2012.

[44] Z. Tudor, "Design and implementation of attitude control for 3-axes magnetic coil stabilization of a spacecraft," Master's thesis, Department of Engineering Cybernetics (ITK), Norwegian University of Science and Technology (NTNU), Trondheim, Norway, 2011.

[45] F. S. Holberg, "Optimal attitude control of a double cubesat using magnetorquers," Project report at the Norwegian University of Science and Technology (NTNU), Trondheim, Norway.

[46] G. Bråthen, "Nonlinear attitude control of a double cubesat using magnetorquers," 2011. Project report at the Norwegian University of Science and Technology (NTNU), Trondheim, Norway.

[47] O. Egeland and J. Gravdahl, *Modeling and simulation for automatic control*. Marine Cybernetics, 2002.

[48] "3-Axis Digital Compass IC HMC5983." http://www51.honeywell.com/aero/common/documents/myaerospacecatalog-documents/Defense_Brochures-documents/HMC5983_3_Axis_Compass_IC.pdf. HMC5983 Magnetometer Datasheet.

[49] H. Khalil, *Nonlinear systems*. Prentice hall New Jersey, 3rd ed., 2002.

[50] J. Davis, "Mathematical modeling of earth's magnetic field," *Technical Note, Virginia Tech, Blacksburg*, 2004.

[51] "American Wire Gauge." http://www.powerstream.com/Wire_Size.htm.

[52] A. Stickler and K. Alfriend, "An elementary magnetic attitude control system," in *American Institute of Aeronautics and Astronautics, Mechanics and Control of Flight Conference, Anaheim, Calif, Research supported by ITHACO, Inc. and NASA*, vol. 5, 1974.

[53] M. Bakken, "Signal processing for communicating gravity wave images from the ntnu test satellite," Master's thesis, Norwegian University of Science and Technology (NTNU), Trondheim, Norway, 2012.

[54] K. Fauske, "Ncube attitude control," 2002. Project report at the Norwegian University of Science and Technology (NTNU), Trondheim, Norway.

[55] B. Gregory, "Attitude control system design for ion, the illinois observing nanosatellite," Master's thesis, University of Illinois, 2004.

[56] P. Hughes and P. Carlisle, *Spacecraft attitude dynamics*. J. Wiley, 1986.

[57] T. Kane, P. Likins, and D. Levinson, *Spacecraft dynamics*, vol. 1. New York, McGraw-Hill Book Co, 1983.

[58] S. Marshall and G. Skitek, *Electromagnetic concepts and applications*, vol. 10. Prentice-Hall, 1990.

[59] K. Musser and W. Ebert, "Autonomous spacecraft attitude control using magnetic torquing only," in *NASA, Goddard Space Flight Center, Flight Mechanics/Estimation Theory Symposium, 1989 p 23-38(SEE N 90-13413 05-13)*, 1989.

[60] B. Vik, "Integrated satellite and inertial navigation systems," *Department of Engineering Cybernetics, NTNU*, 2009.

[61] J. Wertz and W. Larson, *Space mission analysis and design*. Microcosm, 1999.

Appendix A

Matlab Code

```
%*************************************************************************
% File main.m
% Main script. Contains the following sections:
%
% - Simulate (integrate) system
% - Estimate the derivative of the geomagnetic field (the B field)
%
% Written by Zdenko Tudor, 2011. Edited by Gaute Bråthen, 2012.
%*************************************************************************

clear all; close all; clc;
addpath('myIGRF');

% Initialize storage variables (For testing and plotting purposes only)
global VAR Btime Bcnt Bfilt Bdot part_sum tau_a tau_s tau_d R_B_O; % part_

% Bdot = [];

% Counters and global variables used to save the "measured" Bfield and Ba
% to be used as input to control law
cnt = 0;
Bcnt = [0 1];
Bfilt = zeros(3,1);
Bdot= zeros(3,1);
part_sum = zeros(1,3);
% part_sum2 = part_sum;
tau_a = zeros(3,1);
tau_s = zeros(3,1);
tau_d = zeros(3,1);
```

```
% Create parameters struct
P = parameters();

% Load neccessary parameters
orbitPeriod = P.orbitPeriod;          % For defining simulation length
scalePlot = P.scalePlot;              % For x-axes scaling when plotting

% Initial and simulation parameters
numOfOrbits = 3.5;                      % Simulation time in number of or
tSpan = [0,orbitPeriod*numOfOrbits]; % Time span for ODE45
% tSpan = [0,20];                     % For testing higher sampling rate
Btime = 0;

% w_B_OB = [0.1;-0.2;0.1];                    % Initial rotational veloci
w_B_OB = [0.001056;0.000722;-0.000339];   % 0.0019;0.0006;-0.0007 % 0.000
0.001056;0.000722;-0.000339
eulAng = [20;20;20]; %[180;30;-75]; %[20;20;20];   [-20;20;130];
% Initial orientation
% eulAng = [5;-9;6];
% [w_B_IB_tudor,~] = eul2qua(P,w_B_OB',eulAng')     % Transform initial pa
qua = euler2q(eulAng(1)*pi/180,eulAng(2)*pi/180,eulAng(3)*pi/180);
R = Rquat(qua);
w_B_IB = R*w_B_OB;
initLat = 0;                                % Initial latitude
initJoule = 0;                              % Always to be set to zero
init_e = [0 0 0]';                          % Initial error

% State vector x = [wx wy wz eta eps1 eps2 eps3 latitude totalJoule];
xInit = [w_B_IB;qua;initLat;initJoule;init_e];

% ODE45 simulaton settings
options = odeset('MaxStep',1,'OutputFcn',@outFcn,'Refine',1); %
% options = odeset('MaxStep',0.1,'OutputFcn',@outFcn,'RelTol',1e-5,'AbsTo

% Simulate system
tic
[tout,yout] = ode45(@(t,x)nonlinearSatellite_edit(t,x,P),tSpan,xInit,opti
toc


plots();

%*************************************************************************
```

```
% File nonlinearSatellite.m
%
% xDot = nonlinearSatellite(T,X,P)
%
% Simulates the earth-satellite system. To be used with an ordinary
% differential equation solver such as MATLAB's Runge Kutta(4,5) method,
% ODE45.
% The input:
% T - time vector (implicit)
% X - state vector, initial state has to be provided. The state vector is
% of the form X = [W^B_IB' Q' LAMBDA JOULE]. W^B_IB is a 3x1 vector
% containing the three rotational velocity components. Q is the quaternio
% vector of the form Q = [eta eps1 eps2 eps3]. LAMBDA is the latitude of
% the satellite and JOULE is only a testing paramemter which will return
% the total Joule consumption of the satellite during the simulation.
% Initial JOULE should always be set to zero.
% P - struct containg necessary parameters (w_o, I, Rc, coil data, voltag
% and current ratings).
%
% Example:
% [TOUT,YOUT] = ode45(@(t,x)nonlinearSatellite(t,x,P),tSpan,xInit,options
% this produces:
% TOUT - time vector for the integration
% YOUT - Integrated xDot vector for each timestep size(YOUT)=length(TOUT)
%
% Written by Zdenko Tudor, 2011. Edited by Gaute Bråthen, 2012.
%*******************************************************************

function xDot = nonlinearSatellite_edit(t,x,P)
global R_B_O tmpVAR VAR p Btime Bcnt Bfilt Bdot; % tau_a tau_s tau_d;
% tic

% Load necessary parameters from P struct.
w_o = P.w_o;
I = P.I;
J = P.J;
A_drag = P.A_drag;
rho_a = P.rho_a;
vel = P.vel;
s_const = P.s_const;
d_const = P.d_const;

% Normalization of quartenions
x(4:7) = x(4:7)./norm(x(4:7));
```

```
% State vector x = [];
w_B_IB = x(1:3);              % Angular velocity vector
eta = x(4);                   % Euler parameter eta
eps = x(5:7);                 % Euler parameter epsilon
lat = x(8);                   % Lambda
% e = x(10:12);                 % The integrated error of epsilon
% e_tmp = e;
% tmpVAR.e = e;                 % Storing integral of error

satLimit1 = 400;              % Saturation limit for integral action
satLimit2 = 400;              % Saturation limit for integral action

% eul = q2euler([eta eps']);

% Anti-wind up scheme. Set to a preset limit if error is over the limit,
% and set to zero if error is small (in effect, dont use integral action)
if p>1
    if abs(VAR.e(p,1)) > satLimit1
        VAR.e(p,1) = satLimit1*sign(VAR.e(p,1)); %zeros(size(VAR.e));
    end

    if abs(VAR.e(p,2)) > satLimit2
        VAR.e(p,2) = satLimit2*sign(VAR.e(p,2));
    end

    if abs(VAR.e(p,3)) > satLimit2
        VAR.e(p,3) = satLimit2*sign(VAR.e(p,3));
    end

    if abs(eps(1)) < 0.1
        VAR.e(p,1) = 0;
    end

    if abs(eps(2)) < 0.05
        VAR.e(p,2) = 0;
    end

    if abs(eps(3)) < 0.1
        VAR.e(p,3) = 0;
    end

    e_tmp = VAR.e(p,:)';
    tmpVAR.e = eps*(VAR.t(p)-VAR.t(p-1));
```

```
else
    e_tmp = zeros(3,1);
    tmpVAR.e = e_tmp;
end


% Epsilon crossmatrix, skew-symmetric
S_eps = [ 0, -eps(3), eps(2);
          eps(3), 0, -eps(1);
         -eps(2), eps(1), 0];

% Rotation matrices
R_O_B = eye(3) + 2*eta*S_eps + 2*S_eps^2;
R_B_O = R_O_B';

% Angular velocities of frames relative each other
w_O_IO = [0;-w_o;0];
w_B_OB = w_B_IB-R_B_O*w_O_IO;
% w_B_OB = w_B_OB + 200e-9.*randn(size(w_B_OB)); % Adding noise (Noise fl
tmpVAR.omega = w_B_OB; % Storing/saving the "measured" angular velocity

%% w_B_OB crossmatrix, skew-symmetric
% S_w_B_OB = [ 0, -w_B_OB(3), w_B_OB(2);
%              w_B_OB(3), 0, -w_B_OB(1);
%              -eps(2), w_B_OB(1), 0];

%% Filtering the angular velocity w_B_OB
% if(p>1)
%     part_sum2 = 0.95*part_sum2 + 0.05*VAR.omega(p,:);
%     w_B_OB = part_sum2';
% end
% tmpVAR.omegaFilt = w_B_OB; % Storing/saving the filtered angular veloci

[B_O, B_O_n] = IGRF(10,10,lat,0,P);
tmpVAR.B = B_O_n; % Storing/saving the "measured" B field
B_B = R_B_O*B_O;
B_B_n = R_B_O*B_O_n;
tmpVAR.B_B = B_B_n; % Storing/saving the "measured" B field rotated to th

% if p>1
%
%     e = (B_O-B_B);
%
```

```
%       if abs(eps(1)) < 0.05
%           VAR.e(p,1) = 0;
%       end
%
%       if abs(eps(2)) < 0.05
%           VAR.e(p,2) = 0;
%       end
%
%       if abs(eps(3)) < 0.05
%           VAR.e(p,3) = 0;
%       end
%
%       e_tmp = VAR.e(p,:)';
%       tmpVAR.e = e*(VAR.t(p)-VAR.t(p-1));
%
% else
%       e_tmp = zeros(3,1);
%       tmpVAR.e = e_tmp;
% end

% Here the usage of the actuators is imitated. Every iteration the
% calculated ("measured") B field are saved in outFcn.m. For every half
% second, retrieve the sequence of "measured" B fields since last time th
% actuators were used, and use this to filter it and calculate its
% derivative. Use the outputs from B_filterAndDot() in the control loop.
if (~any(Bfilt) && t-Btime>=3)
%       disp('Starting actuation at time')
    Btime = t;

    % Update B counters and retrieve "measured" B_B field and time vector
    Bcnt(1) = Bcnt(2);
    Bcnt(2) = p;
    B = VAR.B_B(Bcnt(1):Bcnt(2),:);
    timeSpan = VAR.t(Bcnt(1):Bcnt(2));

    % Compute filtered geomagnetic B field and the derivative Bdot
    [Bfilt,Bdot] = B_filterAndDot(B,timeSpan,'Estimator_1Hz');
%       [Bfilt,Bdot] = B_filterAndDot(B,timeSpan,'central');

end

% Here the measuring is imitated.
% After 0.5 sec stop the actuation, i.e. set Bfilt and Bdot to zero, and
% start measuring again.
```

```
if (any(Bfilt) && t-Btime>=3)
%     disp('Starting measurement at time')
    Btime = t;
    Bfilt = zeros(3,1);
    Bdot = zeros(3,1);
end

% Storing Bfilt and Bdot
tmpVAR.Bfilt = Bfilt;
tmpVAR.Bdot = Bdot;

% Picking controller
% [tau_m,W] = detumblingController3(P,B_B,Bdot);
[tau_m,W] = referenceController(P,w_B_OB,eps,Bfilt,e_tmp);
% [tau_m,W] = LQcontroller(P,eta,eps,w_B_OB,Bfilt,S_eps,e_tmp);

%% Switch controller after this many orbits
% orbitSwitch = 2;
%
%% Torque from controllers. Logic 'if' is for controller switching.
% if t<P.orbitPeriod*orbitSwitch*1000
%     % Pick detumbling controller
%     [tau_m,W] = detumblingController3(P,Bfilt,Bdot);
% else
%%     [tau_m,W] = referenceController(P,w_B_OB,eta,eps,Bfilt,S_eps);
%     [tau_m,W] = LQcontroller(P,eta,eps,w_B_OB,Bfilt,S_eps);
% end

%-----Storing variables used for testing purposes only-----Start-----
tmpVAR.torque = tau_m;
tmpVAR.W = W;
%-----Storing variables used for testing purposes only-------End-------

% Gravitational torque
c3 = R_B_O(:,3);
tau_g = 3*w_o^2*cross(c3,I*c3);
% tau_g = 0;

% Solar drag torque in ORBOT frame with direction in the orbital
% velocity plane
% if(mod(p,5)==0)
%     tau_s_O = [0 -s_const 0]';
%     tau_s = R_B_O*tau_s_O; % Rotated to the BODY frame
% else
```

```
%       tau_s = 0;
% end
tau_s_O = [-s_const 0 0]';
tau_s = R_B_O*tau_s_O; % Rotated to the BODY frame

% Aerodynamic drag torque in ORBOT frame with direction in the orbital
% velocity plane
c_p = [0.02 0.02 0.02]';
c_p_x = Smtrx(c_p);
Vr = R_B_O*[-1 0 0]';
Vr_x = Smtrx(Vr);
tau_a = rho_a*vel*( vel*A_drag*c_p_x*Vr - (I + Vr_x*J)*w_B_OB );

% Internal dipole torque
% tau_d = 0;
% if(mod(p,10)==0)
%       di = d_const*randn(3,1);
%       m_d = d_const*(di/norm(di));
%       tau_d = cross(m_d,B_B); % Already in the BODY frame
% end
% Di = 0.1386*ones(3,1);
% tau_d = cross(Di,Bfilt);

% All disturbance torques
tau_dist = tau_g+tau_a+tau_s;%+tau_d;
tmpVAR.distTorque = tau_g+tau_a+tau_s;%+tau_d;
tmpVAR.torque = tau_s;


% ************** Differential equations update **************

% Angular acceleration
wDot_B_IB=I\(tau_m+tau_dist-cross(w_B_IB,I*w_B_IB));
% wDot_B_IB=I\(tau_m+tau_g-cross(w_B_IB,I*w_B_IB));

% Eta
etaDot = -0.5*eps'*w_B_OB;

% Epsilon
epsDot = 0.5*(eta*eye(3)+S_eps)*w_B_OB;

% Variable assignment
xDot(1:3,1) = wDot_B_IB;
xDot(4,1) = etaDot;
```

```
xDot(5:7,1) = epsDot;
xDot(8,1) = w_o;
xDot(9,1) = W;
xDot(10:12) = eps;

% toc
end


% Bdot detumbling controller
% Controls the satellite to follow the geomagnetic field.
function [tau_m,W] = detumblingController3(P,B_B,Bdot_B)

% Sign of Bdot controller
% if (norm(w_B_OB)>0.001)
% %     disp('a')
%     k_s = -1;
% elseif (norm(w_B_OB)<0.001)
% %     disp('b')
%     k_s = 1;
% else
% %     disp('c')
%     k_s = 0;
% end

% Controller gain
k = 4e-5;
% k = 1;

% Moment set up by coils before scaling (saturating current)
m_B = (-k/norm(B_B,2)^2)*Bdot_B;

% Moment set up by coils after scaling
[m_B,W] = currentScaling(P,m_B);

% Torque set up by coils
tau_m = cross(m_B,B_B);
% end

end

% Satellite reference controller for large deviations from R_B_O = eye(3)
function [tau_m,W] = referenceController(P,w_B_OB,eps,B_B,e)
```

```
% Controller gains
p = 8e-9; %.4e-9; %2.5e-7 %5e-8
d = 12e-6; %.8e-6; %0.0004 %4e-5
i = 0; %5e-11;

% Moment set up by coils before scaling (saturating current)
m_B = (1/norm(B_B,2)^2)*( -p*cross(B_B,eps) - d*cross(B_B,w_B_OB) - i*cro

% Moment set up by coils after scaling
[m_B,W] = currentScaling(P,m_B);

% Torque set up by coils
tau_m = cross(m_B,B_B);
end

% Satellite reference LQ-controller for deviations from R_B_O = eye(3).
function [tau_m,W] = LQcontroller(P,eta,eps,w_B_OB,B_B,S_eps,e)
% Eta
% etaDot = -0.5*eps'*w_B_OB;

% Epsilon
epsDot = 0.5*(eta*eye(3)+S_eps)*w_B_OB;

% Check if 'measuring' or not
if any(B_B==0)
    m_B = zeros(3,1);
else
    % LQ gain matrix
    K = K2(P,B_B); %Time varying LQ

    % Moment set up by coils before scaling (saturating current)
    m_B = -K*[eps(1) epsDot(1) eps(2) epsDot(2) eps(3) epsDot(3)]'; %e(1)
%     m_B = cross(B_B,m_B);
end

% Moment set up by coils after scaling
[m_B,W] = currentScaling(P,m_B);

% Torque set up by coils
tau_m = cross(m_B,B_B);
end


% Scales (saturates) the power consumption if the maximum currents are
```

```
% exceeded
function [m_B,W] = currentScaling(P,m_B)
% Load coil parameters from parameter struct
N = P.N; % Ny = P.Ny; Nz = P.Nz;
Ax = P.Ax; Ay = P.Ay; Az = P.Az;
ix_max = P.ix_max; iy_max = P.iy_max; iz_max = P.iz_max;
V = P.V;

% If m_B is zero or NaN, i.e. one is in measurement mode, set the moment
% to zero. Also, to avoid error, set the ratio variable to ones.
if (isempty(m_B) || any(m_B==0) || any(isnan(m_B)==1))
%     disp('a')
    m_B = zeros(3,1);
    ratio = ones(3,1);

else
%     disp('b')
    % Current scaling
    % Creates ratio variable containing a coil's: max current/wanted curr
    % The lowest value in ratio variable, if below 1 is the highest curre
    % violation, thus all the currents should be scaled by this factor.
    ix = m_B(1)/(N*Ax); ratio(1)=ix_max/abs(ix);
    iy = m_B(2)/(N*Ay); ratio(2)=iy_max/abs(iy);
    iz = m_B(3)/(N*Az); ratio(3)=iz_max/abs(iz);

end

if min(ratio)<1
%     disp('c')
    m_B = m_B*min(ratio);
end

% Power consumption (Watt)
Wx = abs(V*m_B(1)/(N*Ax));
Wy = abs(V*m_B(2)/(N*Ay));
Wz = abs(V*m_B(3)/(N*Az));
W = Wx+Wy+Wz;

%-----Storing variables used for testing purposes only-----Start-----
global tmpVAR;
tmpVAR.moment=m_B;
% tmpVAR.W = W;
% tmpVAR.J = j;
%-----Storing variables used for testing purposes only-------End-------
```

```
end


% Function for filtering the "measured" B-field, and estimating the
% derivative of the B-field.
function [Bfilt_array, Bdot_array] = B_filterAndDot(B,t,flag)

    delta_t = zeros(size(t));
    Bfilt_array = zeros(size(B));
    Bdot_array = zeros(size(B));

    Bfilt_array(1,:) = B(1,:);

    switch flag

        case 'Estimator_10Hz'
            % First, LP filtering of B using filter approach gotten from
            % Trygve Utstumo:

            a1 = 0.8;
            b1 = 1-a1;
            part_sum = B(1,:);
            for k = 2:length(B)
                part_sum = a1*part_sum + b1*B(k,:);
                Bfilt_array(k,:) = part_sum;
            end

            % Estimating Bdot using discrete LPfilter
            K = 2.65;
            a2 = 0.932; %0.9324
            b2 = 0.7;
            b2 = K*b2;

            for i=2:length(B)
                Bdot_array(i,:) = a2*Bdot_array(i-1,:) + b2*(B(i,:) - B(i
            end

            Bdot_array(~any(Bdot_array,2),:) = []; % Cancel any zeros

            % Return last filtered/estimated value
            Bfilt_array = Bfilt_array(end,:)';
            Bdot_array = Bdot_array(end,:)';

        case 'Estimator_1Hz'
```

```
        % First, LP filtering of B using filter approach gotten from
        % Trygve Utstumo:

        a1 = 0.8;
        b1 = 1-a1;
        part_sum = B(1,:);
        for k = 2:length(B)
            part_sum = a1*part_sum + b1*B(k,:);
            Bfilt_array(k,:) = part_sum;
        end

        % Estimating Bdot using discrete LPfilter
        K = 3; %3;
        a2 = 0.9324; %0.03021; %0.9324;
        b2 = K*(1-a2); %(1-a2); %0.0676 %0.4156

        for i=2:length(B)
            Bdot_array(i,:) = a2*Bdot_array(i-1,:) + b2*(B(i,:) - B(i
        end

        Bdot_array(~any(Bdot_array,2),:) = []; % Cancel any zeros

        % Return last filtered/estimated value
        Bfilt_array = Bfilt_array(end,:)';
        Bdot_array = Bdot_array(end,:)';

    case 'central'
    % Numerical differentiation by central difference approximation

        a1 = 0.8;
        b1 = 1-a1;
        part_sum = B(1,:);
        for k = 2:length(B)
            part_sum = a1*part_sum + b1*B(k,:);
            Bfilt_array(k,:) = part_sum;
        end

        for i=2:length(B)-1
            delta_t(i) = (t(i+1)-t(i-1));
            Bdot_array(i,:) = ( Bfilt_array(i+1,:)-Bfilt_array(i-1,:)
        end

        Bdot_array(~any(Bdot_array,2),:) = []; % Cancel any zeros
```

```
            % Return last filtered/estimated value
            Bfilt_array = Bfilt_array(end,:)';
            Bdot_array = Bdot_array(end,:)';

    end

end

%**************************************************************************
% File parameters.m
%
% P = parameters()
%
% Creates a parameter struct P, which is used to pass the parameters
% between different functions.
%
% Written by Zdenko Tudor, 2011. Edited by Gaute Bråthen, 2012.
%**************************************************************************

function P = parameters()
    % Coil data
    % Battery voltage
    V = 5;

    % Number of coil windings (turns)
    N = 221; %196;
%     Ny = 221; %196;
%     Nz = 221; %196;

    % [m^2] Coil area
    Ax = 0.063*0.165; %0.079*0.184; %0.075*0.175;
    Ay = 0.063*0.165; %0.079*0.184; %0.075*0.175;
    Az = 0.077^2;     %0.079^2;     %0.075^2;

    % [Ohm] Coil resistance
    Rx = 43.03; %34.78; %43.03; %55.47; %47.18;
    Ry = 43.03; %34.78; %43.03; %55.47; %47.18;
    Rz = 23.78; %19.22; %23.78; %33.32; %28.31;

    % [A] Theoretical maximum current through coils
    ix_max = V/Rx;
    iy_max = V/Ry;
    iz_max = V/Rz;
```

```
    % [A] Real maximum current through coils for a given AWG number (see
    % excel-sheet 'Coil design'):
%     i_max = 0.2260; % Max amps for AWG# 28
%     i_max = 0.1420; % Max amps for AWG# 30
    i_max = 0.1130; % Max amps for AWG# 31
%     i_max = 0.0910; % Max amps for AWG# 32

    if(ix_max>i_max) % Check if theoretical max amps is higher than real
        ix_max = i_max;
        iy_max = i_max;
    end

    if(iz_max>i_max) % Same as aboce for the z-coil
        iz_max = i_max;
    end

    Re = 6371.2e3;                  % [m] Earth radius
    Rs = 600e3;                     % [m] Satellite altitude
    Rc = Re+Rs;                     % [m] Distance from earth center to satel
    m = 2.6;     % 2.66             % [kg] Satellite mass
    dx = 0.05;  % 0.05              % X-axis length
    dy = 0.049; % 0.049             % Y-axis length
    dz = 0.113;                     % Z-axis length
    Ix = (m/12)*(dy^2+dz^2);        % X-axis inertia
    Iy = (m/12)*(dx^2+dz^2);        % Y-axis inertia
    Iz = (m/12)*(dx^2+dy^2);        % Z-axis inertia
    jx = 0.03;  % 0.05              % X-axis length
    jy = 0.03; % 0.049              % Y-axis length
    jz = 0.093;                     % Z-axis length
    Jx = (m/12)*(jy^2+jz^2);        % X-axis inertia
    Jy = (m/12)*(jx^2+jz^2);        % Y-axis inertia
    Jz = (m/12)*(jx^2+jy^2);        % Z-axis inertia
    x = 0.1;
    y = 0.1;
    z = 0.2;
    A_xo = x*z;                     % Outer satellite area, x-panel
    A_yo = y*z;                     % Outer satellite area, y-panel
    A_zo = x*y;                     % Outer satellite area, z-panel
    A_drag = z*sqrt(x^2 + y^2); % Maximum area for calc disturbance drags

    I = diag([Ix Iy Iz]);           % Inertia matrix
    J = diag([Jx Jy Jz]);           % Inertia matrix

    G = 6.67428e-11;                % Earth gravitational constant
```

```
    M = 5.972e24;                % Earth mass

    w_o = sqrt(G*M/Rc^3);        % Satellite angular velocity relative Ear

    orbitPeriod = (2*pi)/(w_o); % For defining simulation length
    scalePlot = 1/orbitPeriod;  % For x-axes scaling when plotting

    vel = 2*pi*Rc/orbitPeriod;  % Orbit velocity
    rho_a = 4.89e-13;
    a_const = 0.5*rho_a*vel^2*2.5*A_drag*0.02;       % Aero drag const
    s_const = (1367/3e8)*A_drag*(1+0.6)*cos(0)*0.02;
% Solar drag const
    d_const = 0.00072;            % Calculated self, see notes
%     d_const = 4.59e-7; % Internal dipole const, Giesslmann, 2006 p. 44/

    % Store variables
    P.w_o = w_o;
    P.V = V; P.I = I; P.J = J;
    P.N = N; % P.Ny = Ny; P.Nz = Nz;
    P.Ax = Ax; P.Ay = Ay; P.Az = Az;
    P.Rx = Rx; P.Ry = Ry; P.Rz = Rz;
    P.ix_max = ix_max; P.iy_max = iy_max; P.iz_max = iz_max;
    P.Re = Re; P.Rc = Rc;
    P.orbitPeriod = orbitPeriod; P.scalePlot = scalePlot;
    P.A_xo = A_xo; P.A_yo = A_yo; P.A_zo = A_zo; P.A_drag = A_drag;
    P.vel = vel; P.rho_a = rho_a;
    P.a_const = a_const; P.s_const = s_const; P.d_const = d_const;
end

%*************************************************************************
% File IGRF.m
%
% Algorithm for finding the magnetic field vectors from the IGRF-11
% model for a given latitude LAT and longitude LON in radians, with
% order n and degree m. The output is the magnetic field vector
% represented in the orbit frame.
%
% Created by Raymond Kristiansen, 2000
% Edited by Zdenko Tudor, 2011
%*************************************************************************
function [B, B_n] = IGRF(n, m, lat, lon, P)
% tic
% Load IGRF11 coefficient variables
load('IGRF11_data')
```

```
i = n;
j = m;

% Fix latitude in case of possible singularity
mt = mod(lat,2*pi);
tol = 1e-9;
if (mt>(pi/2)-tol && mt<(pi/2)+tol)
    lat=(pi/2)+tol;
elseif(mt>(3*pi/2)-tol && mt<(3*pi/2)+tol)
    lat=(3*pi/2)+tol;
end

% Defining constants
Re = P.Re;
Rc = P.Rc;

% Calculating colatitude and longitude in radians
theta = (pi/2)-lat;
phi = lon;

% Zero offset
O = 1;

% Defining temporary variables
Bt2 = 0; Bp2 = 0; Br2 = 0;

% Calculating Legendre polynominals
[P, dP, S]=Pfunk(n, m, theta);

% Calculating field vectors
for n=1:i
    Bt1 = 0;
    Bp1 = 0;
    Br1 = 0;
    for m=0:j
        Bt1 = Bt1 + (S(O+n,O+m)*g_data(O+n,O+m)*cos(m*phi)+S(O+n,O+m)*...
            h_data(O+n, O+m)*sin(m*phi))*dP(O+n,O+m);
        Bp1 = Bp1 + (m*S(O+n,O+m)*h_data(O+n, O+m)*cos(m*phi)-m*...
            S(O+n,O+m)*g_data(O+n, O+m)*sin(m*phi))*P(O+n,O+m);
        Br1 = Br1 + (S(O+n,O+m)*g_data(O+n,O+m)*cos(m*phi)+S(O+n,O+m)*...
            h_data(O+n, O+m)*sin(m*phi))*P(O+n,O+m);
    end
    Bt2 = Bt2 + ((Re/Rc)^(n+2))*Bt1;
```

```
    Bp2 = Bp2 + ((Re/Rc)^(n+2))*Bp1;
    Br2 = Br2 + (n+1)*((Re/Rc)^(n+2))*Br1;
end

% In Cartesian coordinates
eps = 0;
X = Bt2*cos(eps)-Br2*sin(eps);
Y = -Bp2/(sin(theta));
Z = -Bt2*sin(eps)-Br2*cos(eps);

B = [X;Y;Z]*1e-9;
B_n = B + 300e-9.*randn(size(B)); % Adding noise (Noise floor from HMC598
% toc
end

%**************************************************************************
%
% Algorithm for calculating the associated Legendre polynominals
% for the given order n, degree m and co-latitude theta. The output
% is the Legendre polynominal P, its partial derivative dP and
% the Schmidt normalization matrix S.
%
% Created by Raymond Kristiansen, 2000
% Edited by Zdenko Tudor, 2011
%**************************************************************************
function [P, dP, S] = Pfunk(n, m, theta)
O = 1; % Zero index offset

% Defining zero matrices
P = ones(n+O, m+O);
dP = zeros(n+O, m+O);
S = ones(n+O, m+O);
i = n;
j = m;

% Calculating S matrix
for n = 0:1:i
    for m = 0:1:j
        if n > 0
            if m == 0
                S(O+n, O+m) = S(n-1+O, 0+O)*(2*n-1)/n;
            else
                if m == 1
                    deltaFun = 1;
```

```
                else
                    deltaFun = 0;
                end
                S(n+O, m+O) = S(n+O, m-1+O)*sqrt(((n-m+1)*(deltaFun+1))/(
            end
        end
    end
end

% Calculates the Legendre polynominal P and its partial derivative dP
for n = 0:1:i
    for m = 0:1:j
        if n==1
            dP(n+O, m+O) = cos(theta)*dP(n-1+O, m+O)-sin(theta)*P(n-1+O,m
            P(n+O, m+O) = cos(theta)*P(n-1+O, m+O);
        elseif n>1
            K=((n-1)^2-m^2)/((2*n-1)*(2*n-3));
            dP(n+O, m+O) = cos(theta)*dP(n-1+O, m+O)-sin(theta)*P(n-1+O,m
                K*dP(n-2+O, m+O);
            P(n+O, m+O) = cos(theta)*P(n-1+O, m+O) - K*P(n-2+O, m+O);
        end
        if n==m && n>0
            P(n+O, m+O) = sin(theta)*P(n-1+O, n-1+O);
            dP(n+O, m+O) = sin(theta)*dP(n-1+O, n-1+O)+cos(theta)*P(n-1+O
        end
    end
end

end

% %*************************************************************************
% File outFcn.m
%
% status = outFcn(t,~,flag)
%
% Is run when ODE45 successfully completes a step.
% This function is used implicitly by ODE45, when ODE settings are
% properly set.
%
% Example:
% options = odeset('OutputFcn',@outFcn,'Refine',1);
% Refine sets how many steps should be created for each successfull step.
% With this set to 1 length of all vectors created by the OutputFcn will
% the same as the length of ODE timevector TOUT.
```

```
%
% Written by Zdenko Tudor, 2011
%*************************************************************************

function status = outFcn(t,~,flag)
    persistent ite;
    global tmpVAR VAR p;
        switch flag
            case 'init'
                %initialize arrays etc.
                p = 1;
                ite = 1;
                VAR.e = zeros(20000,3);
                VAR.moment = zeros(20000,3);
                VAR.torque = zeros(20000,3);
                VAR.distTorque = zeros(20000,3);
                VAR.B = zeros(20000,3);
                VAR.B_B = zeros(20000,3);
                VAR.Bfilt = zeros(20000,3);
                VAR.Bdot = zeros(20000,3);
                VAR.omega = zeros(20000,3);
%                   VAR.omegaFilt = zeros(20000,3);
                VAR.W = zeros(20000,1);
                VAR.t = zeros(20000,1);

                status = 0;
            case 'done'
                VAR.e(ite+1:end,:) = [];
                VAR.moment(ite+1:end,:) = [];
                VAR.torque(ite+1:end,:) = [];
                VAR.distTorque(ite+1:end,:) = [];
                VAR.B(ite+1:end,:) = [];
                VAR.B_B(ite+1:end,:) = [];
                VAR.Bfilt(ite+1:end,:) = [];
                VAR.Bdot(ite+1:end,:) = [];
                VAR.omega(ite+1:end,:) = [];
%                   VAR.omegaFilt(ite+1:end,:) = [];
                VAR.W(ite+1:end,:) = [];
                VAR.t(ite+1:end,:) = [];

                status = 0;
            otherwise
                p = p+1;
                ite = ite+1;
```

```
                VAR.e(ite,:) = VAR.e(ite-1,:) + tmpVAR.e';
                VAR.moment(ite,:) = tmpVAR.moment';
                VAR.torque(ite,:) = tmpVAR.torque';
                VAR.distTorque(ite,:) = tmpVAR.distTorque';
                VAR.B(ite,:) = tmpVAR.B';
                VAR.B_B(ite,:) = tmpVAR.B_B';
                VAR.Bfilt(ite,:) = tmpVAR.Bfilt';
                VAR.Bdot(ite,:) = tmpVAR.Bdot';
                VAR.omega(ite,:) = tmpVAR.omega';
%                 VAR.omegaFilt(ite,:) = tmpVAR.omegaFilt';
                VAR.W(ite,:) = tmpVAR.W;
                VAR.t(ite,1) = t(end);

                status = 0;
        end
end


function K = K2(P,B_B)
%*************************************************************************
% LQ gain matrix calculation
%
% Written by Eli Overby 2004, modified by Fredrik Sola Holberg 2011.
% Modified by Gaute Bråthen 2012.
% Ntnu, Trondheim, December 20th 2012.
%
%*************************************************************************
% global I omega_0

% Initial values
w_o = P.w_o;
Ix = P.I(1,1);
Iy = P.I(2,2);
Iz = P.I(3,3);
kx = (Iy - Iz)/Ix;
ky = (Ix - Iz)/Iy;
kz = (Iy - Ix)/Iz;

% The geomagnetic field
Bx_0 = B_B(1);
By_0 = B_B(2);
Bz_0 = B_B(3);

% The linearized system matrix
```

```
A = [0 1 0 0 0 0;
-4*kx*w_o^2 0 0 0 0 (1 - kx)*w_o;
0 0 0 1 0 0;
0 0 -3*ky*w_o^2 0 0 0;
0 0 0 0 0 1;
0 -(1 - kz)*w_o 0 0 -kz*w_o^2 0];

% size(A)
% eig(A)

% The input matrix for the linearized system
B = [0 0 0;
0 Bz_0/(2*Ix) -By_0/(2*Ix);
0 0 0;
-Bz_0/(2*Iy) 0 Bx_0/(2*Iy);
0 0 0;
By_0/(2*Iz) -Bx_0/(2*Iz) 0];

% C = diag([1 1 1 1 1 1 1 1 1]);
% D = 0;
%
% SYS = ss(A,B,C,D)
% minSYS = minreal(SYS)

% SYSd = c2d(SYS,1);

% LQ-weighting matrices
% Qmat = diag( [1/((160*pi/180)^2) 0 1/((160*pi/180)^2) 0 1/((130*pi/180)
1e-7 1e-7 1e-7
% Qmat = blkdiag( diag([1 0 1 0 1 0])*1/((80*pi/180)^2), diag([1 0 1 0 1
Qmat = diag( [1 0 1 0 1 0] )*1/((60*pi/180)^2); %x
Pmat = diag([1 1 1])*1/((0.001)^2); %u      0.00091

% Calculate the gain matrix
K = lqr(A,B,Qmat,Pmat);

end
```

Appendix B

Mechanical Drawings and Data

Mechanical drawings for coil (magnetorquer) frames and mechanical data of the satellite. All privided by Christian E. Nomme.

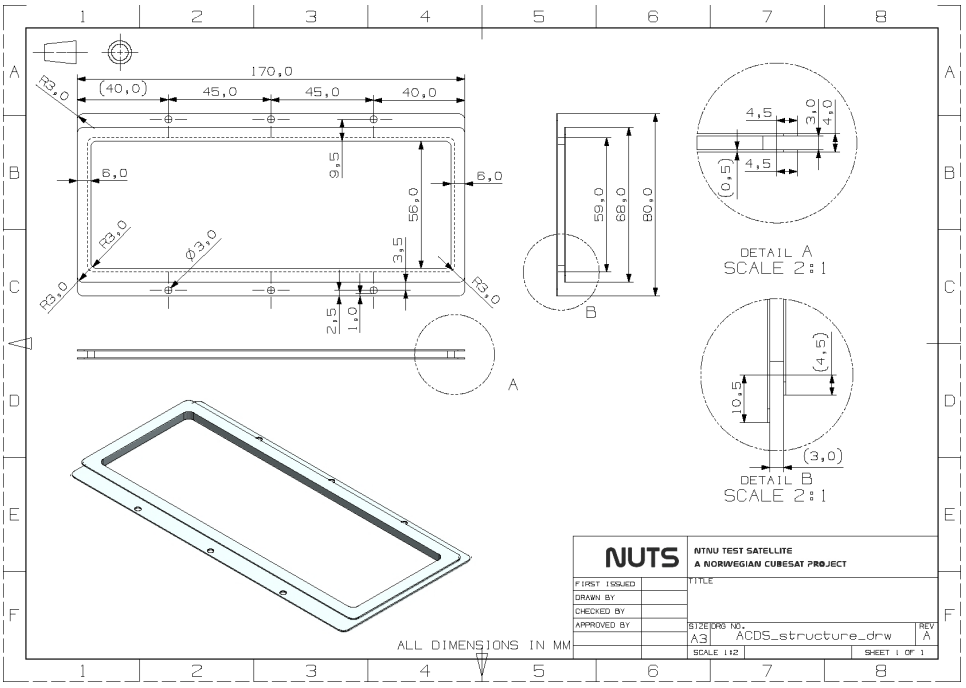

Figure B.1: Mechanical drawing of magnetorquer to be used and the x- and y-sides.

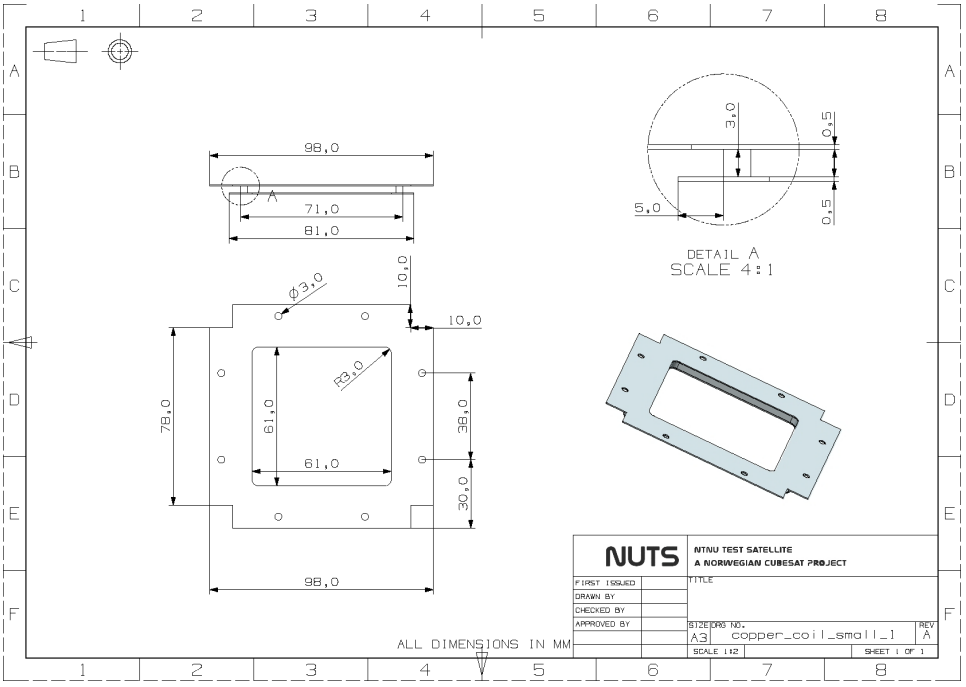

Figure B.2: Mechanical drawing of magnetorquer to be used on the z-side.

==========

Measurement Mass Properties

Displayed Mass Property Values
Volume = 822752.789791495 mm^3
Area = 818187.509097557 mm^2
Mass = 1.200376087 kg
Weight = 11.771678777 N
Radius of Gyration = 92.368722593 mm
Centroid = 0.217424347, 1.849467925, -13.335212988 mm

==========

Detailed Mass Properties
Analysis calculated using accuracy of 0.990000000
Information Units kg - mm

Density = 0.000001459
Volume = 822752.789791495
Area = 818187.509097557
Mass = 1.200376087

First Moments
Mx, My, Mz = 0.260990987, 2.220057071, -16.007270789

Center of Mass
Xcbar, Ycbar, Zcbar = 0.217424347, 1.849467925, -13.335212988

Moments of Inertia (WCS)
Ix, Iy, Iz = 7881.402909432, 7935.008012060, 5102.006880730

Moments of Inertia (Centroidal)
Ixc, Iyc, Izc = 7663.836619764, 7721.490900940, 5097.844210592

Moments of Inertia (Spherical)
I = 10241.585865648

Products of Inertia (WCS)
Iyz, Ixz, Ixy = -16.579474684, -65.516622122, -0.072180012

Products of Inertia (Centroidal)
Iyzc, Ixzc, Ixyc = 13.025459202, -62.036251727, -0.554874470

Radii of Gyration (WCS)
Rx, Ry, Rz = 81.029488479, 81.304581213, 65.194634121

Radii of Gyration (Centroidal)
Rxc, Ryc, Rzc = 79.903251286, 80.203240256, 65.168032909

Radii of Gyration (Spherical)
R = 92.368722593

Principal Axes (Direction vectors relative to the WCS)
Xp(X), Xp(Y), Xp(Z) = 0.004388892, 0.999978556, -0.004860637