Development of the Software Package of The Attitude Determination and Control Subsystem in a Cube Satellite.

Zeyad Mahmoud Manaa
Mohammad Reda Elbalshy
Rana Aymen Elsemary
Walid Ibrahim Elbalshy
Abdelrahman Ahmed Kotb

A thesis presented for the Bachelor's degree fulfilment udner the supervision of Egyptian Space Agency and Zewail City.

مدينة زويل
ZEWAIL CITY
ESTABLISHED 2000
INAUGURATED 2011

Aerospace Department
University of Science and Technology
Zewail city
Egypt

Development of the Software Package of The Attitude Determination and Control Subsystem in a Cube Satellite.

Zeyad Manaa, Mohammad Elbalshy, Rana Elsemary, Abdelrahman Kotb, Walid Elbalshy

Abstract

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

PREFACE AND ACKNOWLEDGMENTS.

CONTENTS

1 Attitude Parameterization of Rigid Body 9

1.1 Direction Cosine Matrix . . . 9

1.2 Euler angles . . . 10

1.3 Quaternions . . . 11

1.3.1 Definition . . . 12

1.3.2 Quaternions Equality, Addition, and Subtraction . . . 12

1.3.3 Quaternions Multiplication . . . 13

1.3.4 Quaternion Complex Conjugate and the Norm . . . 13

1.3.5 Natural power and the exponential of pure quaternions . . . 14

1.3.6 Rotation Operator . . . 15

2 Coordinate Systems and Reference Frames 17

2.1 Earth-Based Reference Frames . . . 17

2.1.1 Earth Centered Earth Fixed Frame . . . 17

2.1.2 Earth Centered Inertial Frame . . . 17

2.1.3 The Geodetic Coordinate System . . . 18

2.2 Satellite-Based Reference Frames . . . 19

2.2.1 Orbit reference frame . . . 19

2.2.2 Satellite body reference frame . . . 19

3 Space Environment 20

3.1 Geomagnetic Field – IGRF . . . 20

3.1.1 Geomagnetic Field Definition . . . 21

3.1.2 Mathematical Model . . . 21

3.1.3 MATLAB Model Implementation . . . 23

3.1.4 Results of The MATLAB Model . . . 25

3.2 Geo-Gravitational Field – Spherical Harmonics Gravity . . . 26

3.2.1 Definition . . . 26

3.2.2 Mathematical Model . . . 26

3.2.3 MATLAB Model Implementation . . . 27

3.3 Eclipse Time . . . 28

3.3.1 Definition . . . 28

3.3.2 Beta Angle Calculations . . . 29

3.4 Julian Date 29
3.5 Sun Position Vector 30
3.6 Sidereal Time 31

4 Space Perturbations 33
4.1 Gravity Gradient Disturbance 33
4.2 Aerodynamic Disturbance 34
4.3 Magnetic Field Residual Disturbance 35

5 Equations of Motion 36
5.1 Kinematic Equations of Motion 36
5.1.1 Time Derivative of Quaternions 36
5.1.2 The Omega operator 37
5.2 Angular Momentum and Moment of Inertia Tensor 38
5.3 Dynamics Equation of Motion 38
5.3.1 Euler's Moment Equation 38
5.4 Controllability 39
5.5 Numerical Scheme 39
5.5.1 RK1 – Euler's Method 40
5.5.2 RK2 – Midpoint Heun's Method 40
5.5.3 RK4 – Method 40
5.5.4 General Runge-Kutta method 42
5.5.5 Adaptive Step Size Runge-Kutta Method 42

6 State Estimation 45
6.1 Attitude Estimation 45
6.2 Mathematical Model 46
6.2.1 Rate Gyro and Magnetometer Models 46
6.3 Kalman Filter Design 47
6.3.1 Linearized process model 47
6.3.2 Initialization 47
6.3.3 Correction 47
6.3.4 Prediction 48

7 Control Algorithm 49
7.1 Literature Review 49
7.2 Detumbling 50
7.2.1 B-dot Controller 50
7.2.2 Simulation Setup 50
7.2.3 Simulation Results 52

LIST OF FIGURES

1.1 Body frame with respect to inertial reference frame. 10
1.2 Rotation sequence. 11
1.3 Vector $\boldsymbol{v}$ rotated with angle θ to be vector $\boldsymbol{w}$ 15

2.1 The reference frames used in this thesis. 18

3.1 Worldwide Distribution of Geomagnetic Observatories 20
3.2 Magnetic intensity as function of altitude and latitude. 21
3.3 Total Geomagnetic Field contours in nT as depicted by MATLAB simulation. 25
3.4 Umbra and penumbra regions from earth's shadow 29
3.5 Solar beta angle definition 29

4.1 Coordinate System for the Calculation of Gravity-Gradient Torque . 34

5.1 Comparison between RK1:4 and Ode45 41
5.2 Error between RK1:4 and Ode45 42
5.3 Comparison between RK4, RKF45 and Ode45 44

6.1 block diagram for the estimation problem formulation 46

7.1 A block diagram of the simulation framework together with the detumbling controller. 51
7.2 The controlled angular rate ω of the satellite for the case of detumbling 52
7.3 Zoom into controlled angular rate ω of the satellite 52
7.4 The controlled Euler angels of the satellite for the case of detumbling 53

LIST OF TABLES

3.1 g_n^m Values as provided in IGRF-13 Generation 23
3.2 h_n^m Values as provided in IGRF-13 Generation 23
3.3 IN/OUT parameters of calculate_d_nm.m 24
3.4 IN/OUT parameters of calculate_v.m 24
3.5 IN/OUT parameters of calc_partial_v.m 24
3.6 IN/OUT parameters of read_coef.m 24
3.7 IN/OUT parameters of magneticFieldModel.m 25
3.8 IN/OUT parameters of calc_v.m 27
3.9 IN/OUT parameters of calc_partial_v.m 27
3.10 *MATLAB function* – reading $g_{n,m}$, $h_{n,m}$, $\dot{g}_{n,m}$, $\dot{h}_{n,m}$ 28
3.11 IN/OUT parameters of calculate_gravity.m 28
3.12 IN/OUT parameters of Julian_day.m 30
3.13 IN/OUT parameters of calc_sun_pos_vector.m 31

7.1 Cube-sat simulation parameters 51

NOMENCLATURE

Acronyms

ACS	Attitude Control System.
ADS	Attitude Determination System.
ADS	Attitude Determination System.
ADCS	Attitude Determination and Control System.
EKF	Extended Kalman Filter.
DCM	Direction Cosine Matrix.
EoM	Equation of Motion.
IGRF	International Geomagnetic Reference Field.
JD	Julian Date.
KF	Kalman Filter.
LEO	Low Earth Orbit.
MoD	Mean-equator of Date.
ODE	Ordinary Differential Equation.
RK	Runge-Kutta.
RKF	Runge-Kutta-Fehlberg.
UKF	Unscented Kalman Filter.
UT	Universal Time.

Reference Frames

BRF	Body Reference Frame.
ECI	Earth Centered Inertial reference frame.
ECEF	Earth Centered Earth Fixed reference frame.
ORF	Orbit Reference Frame.

CHAPTER 1

ATTITUDE PARAMETERIZATION OF RIGID BODY

1.1 Direction Cosine Matrix

Three-dimensional space, denoted $\mathbb{R}^3$, can be coordinated in a manner that is entirely analogous to how coordinates in $\mathbb{R}^2$ are introduced. $\mathbb{R}^3$ specifies an arbitrary but fixed point, which we call the origin. Three mutually perpendicular lines passing through this origin are designated as the *x-axis,y-axis,and z-axis* respectively. Each of these axes is a real number line, with the origin at the zero point. These axes are oriented so that a positive or right-handed coordinate frame is formed. A right-handed coordinate frame is one in which the right hand's fingers point positively along the *x-axis*.

Given such a 3D coordinate system, points are represented by triplets of real numbers (x, y, z). The origin point O is represented in this frame by the point $(0, 0, 0)$. Any given point in this coordinate system can be represented as $P(x, y, z)$ which also can be represented as vector v from the origin O to the point P.

In $\mathbb{R}^3$, a rotation vector or axis that remains fixed during the rotation (such as x, y and z axes) is needed to rotate about. In this case, the rotation matrix $[\mathbf{A}]$ can be described as a 3×3 matrix. Suppose that a vector has coordinates (x_1, y_1, z_1) relative to the *xyz* coordinate frame. If the coordinate frame is rotated with an angle θ about the *x-axis*. Let the coordinates (x_2, y_2, z_2) denotes P in the rotated frame. As the rotation occurs about *z-axis*, the z component will not be affected by the rotation.

$$z_1 = z_2$$

x_2 and y_2 are determined by the following equations

$$\begin{aligned} x_2 &= \cos(\theta)x_1 + \sin(\theta)y_1 + 0z_1 \\ y_2 &= -\sin(\theta)x_1 + y_1\cos(\theta) + 0z_1 \\ z_2 &= 0x_1 + 0y_1 + z_1 \end{aligned} \tag{1.1}$$

This can be represented as

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} \tag{1.2}$$

Now, the rotation in $\mathbb{R}^3$ by angle θ about the *z-axis* can be carried out using the rotation matrix represented

$$R(\theta) = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1.3}$$

This matrix is also called the direction cosine matrix.

1.2 Euler angles

A rigid body in space has a coordinate frame attached to it, which is frequently located at the center of mass. This frame is known as the body frame or the local frame. The body frame can be used to describe the position, orientation, and motion of the body in relation to a fixed reference frame known as the Inertial frame as in figure 1.1.

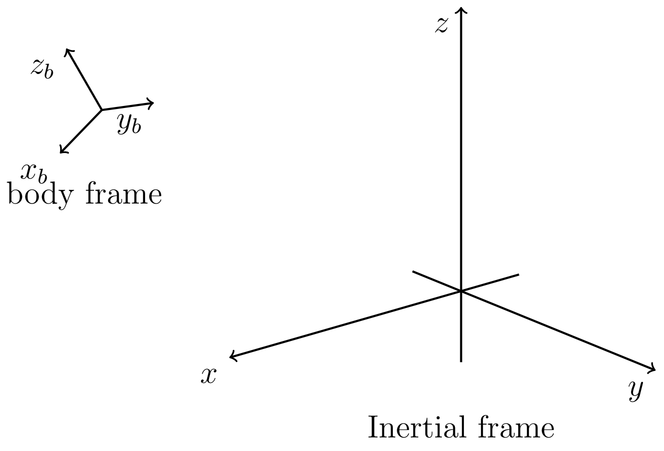


Figure 1.1: Body frame with respect to inertial reference frame.

There are six degrees of freedom for the rigid body: its position in the world frame is given by the *x,y,and z* coordinates of its center of mass such as the origin of the local frame, and its orientation defined by three angles of rotation of its body frame relative to the world frame One option for describing this rotation is to use three angles, one for each of the world frame's *x-,y-,and z-axes.* From the perspective of the rotating body, using three angles about the body frame's axes is a more convenient option. The last three angles are known as Euler angles. Depending on the axes around which the rotations are performed, there are several conventions for Euler angles. The *XYZ* Euler angles are introduced here. The body frame is initially oriented in the same way as the world frame.

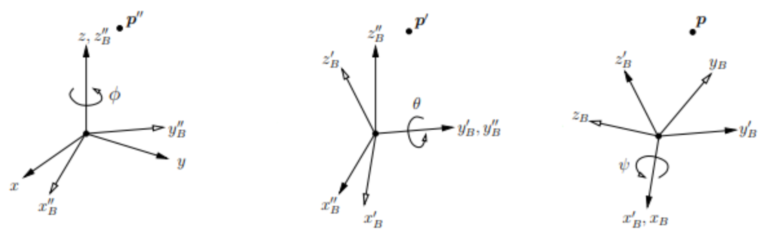


Figure 1.2: Rotation sequence.

Three consecutive different rotations are carried out to move from the world frame to the final position. The first rotation is about the body *x–axis* by an angle ψ, the second rotation is about the body *y-axis* by an angle θ and the third rotation is about the body *z–axis* by an angle ϕ. ϕ, θ and ψ are the Euler angles in this case.

$$\begin{aligned} \boldsymbol{p}' &= R_x(\psi)\boldsymbol{p} \\ \boldsymbol{p}'' &= R_y(\theta)R_x(\psi)\boldsymbol{p} \\ R_z(\phi)\boldsymbol{p}'' &= R_z(\phi)R_y(\theta)R_x(\psi)\boldsymbol{p} \\ R_{xyz}(\psi,\theta,\phi) &= R_z(\phi)R_y(\theta)R_x(\psi) \end{aligned} \tag{1.4}$$

The total rotation matrix is defined as

$$R_{xyz}(\psi,\theta,\phi) = \begin{bmatrix} \cos(\phi) & -\sin(\phi) & 0 \\ \sin(\phi) & \cos(\phi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\psi) & -\sin(\psi) \\ 0 & \sin(\psi) & \cos(\psi) \end{bmatrix}$$

$$R_{xyz}(\psi,\theta,\phi) = \begin{bmatrix} c\phi c\theta & c\phi s\theta s\psi - s\phi c\psi & c\phi s\theta c\psi + s\phi s\psi \\ s\phi c\theta & s\phi s\theta s\psi + c\phi c\psi & s\phi s\theta c\psi - c\phi s\psi \\ -s\theta & c\theta s\psi & c\theta c\psi \end{bmatrix} \tag{1.5}$$

This is defined as the rotation matrix DCM.

1.3 Quaternions

W. R. Hamilton [1] is credited with the invention of quaternions in 1843. According to legend, Hamilton was strolling with his wife Helen at the Royal Irish Academy when the thought of adding a fourth dimension to multiply triples hit him. He etched the newfound quaternion equations

$$\hat{\boldsymbol{i}}^2 = \hat{\boldsymbol{j}}^2 = \hat{\boldsymbol{k}}^2 = \hat{\boldsymbol{i}}\hat{\boldsymbol{j}}\hat{\boldsymbol{k}} = -1 \tag{1.6}$$

into the stone as he and his wife passed the Brougham bridge of the Royal Canal. His problem was to find a similar analogy to what is known for the complex number in $\mathbb{R}^2$. The Product of complex numbers in $\mathbb{R}^2$ geometrically provoked into the rotation of vectors in the plane. Hamilton needed to find such a way in $\mathbb{R}^3$ space using real number triplets. His way discussed in earlier equation managed to have such an implication [2].

1.3.1 Definition

Let an expression of the form

$$q = q_0 + q_1\hat{\imath} + q_2\hat{\jmath} + q_3\hat{k} \quad \Leftrightarrow \quad q = q_0 + \boldsymbol{q} \tag{1.7}$$

Be called a quaternion where q_1, q_2, q_3, q_4 are the four constituents of the quaternion q, denote any real quantities, positive or negative or null, but $\hat{\imath}, \hat{\jmath}, \hat{k}$ are symbols of three imaginary quantities, which we shall call imaginary units, and are assumed to be unconnected by any linear relation with each other.

From the definition we see that we can set in complex numbers $\mathbb{Z}$, and thus real $\mathbb{R}$ and imaginary numbers $\mathbb{I}$, in the quaternion Hamilton space $\mathbb{H}$, in the sense that real, imaginary and complex numbers are implicitly quaternions

$$q = q_0 \in \mathbb{R} \subset \mathbb{H}, \quad q = q_1\hat{\imath} \in \mathbb{I} \subset \mathbb{H}, \quad q = q_0 + q_1\hat{\imath} \in \mathbb{Z} \subset \mathbb{H} \tag{1.8}$$

Similarly, we can define numbers in $\mathbb{H}$'s three-dimensional imaginary subspace. They can be called pure quaternions, and the space of pure quaternions is $\mathbb{H}_p = Im(\mathbb{H})$

$$q = q_1\hat{\imath} + q_2\hat{\jmath} + q_3\hat{k} \in \mathbb{H}_p \subset \mathbb{H} \tag{1.9}$$

A quaternion can be defined as an order pair scalar-vector

$$\langle q_0, \boldsymbol{q} \rangle \tag{1.10}$$

Quaternions are defined frequently as a 4-vector q as

$$\boldsymbol{q} \triangleq \begin{bmatrix} q_0 \\ \boldsymbol{q} \end{bmatrix} = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \tag{1.11}$$

This allows us to utilize matrix algebra for quaternion operations. We might allow ourselves to blend notations on occasion by manipulating the symbol (=).

1.3.2 Quaternions Equality, Addition, and Subtraction

If there is another quaternion

$$q' = q'_0 + q'_1\hat{\imath} + q'_2\hat{\jmath} + q'_3\hat{k} \quad \Leftrightarrow \quad q' = q'_0 + \boldsymbol{q'} \tag{1.12}$$

the presumption that these two quaternions are equal,

$$q' = q \tag{1.13}$$

shall be considered to entail four independent equations between their elements, namely, the four equations listed below

$$q'_1 = q_1, \quad q'_2 = q_2, \quad q'_3 = q_3, \quad q'_4 = q_4 \tag{1.14}$$

It will then be natural to state that the formula affects the addition and subtraction of quaternions as

$$q \pm q' = (q'_0 \pm q_0) + (q'_1 \pm q_1)\,\hat{\imath} + (q'_2 \pm q_2)\,\hat{\jmath} + (q'_3 \pm q_3)\,\hat{k} \Leftrightarrow q \pm q' = (q'_0 \pm q_0) + (\boldsymbol{q} + \boldsymbol{q}') \tag{1.15}$$

or, to say with different way, by the rule that the sums or differences of the constituents of any two quaternions are the components of the sum or difference of those two quaternions.

1.3.3 Quaternions Multiplication

As in $\mathbb{R}^3$, the product of a scalar and a quaternion is defined in the same way as vectors are defined. If c is a scalar and q is a quaternion. The result is the product of the quaternion q and the scalar c is

$$cq = cq_0 + cq_1\hat{\imath} + cq_2\hat{\jmath} + cq_3\hat{k} \quad \Leftrightarrow \quad q = cq_0 + c\boldsymbol{q} \tag{1.16}$$

Simply multiply the quaternion's components by the scalar to multiply the quaternion by the scalar. It's worth noticing that the result is another quaternion.

Given the fundamental set of rules proposed by Hamilton

$$\begin{aligned} \hat{\boldsymbol{i}}^2 = \hat{\boldsymbol{j}}^2 = \hat{\boldsymbol{k}}^2 = \hat{\boldsymbol{i}}\hat{\boldsymbol{j}}\hat{\boldsymbol{k}} = -1 \\ \hat{\boldsymbol{j}}\hat{\boldsymbol{j}} = \hat{\boldsymbol{k}} = -\hat{\boldsymbol{j}}\hat{\boldsymbol{i}} \\ \hat{\boldsymbol{j}}\hat{\boldsymbol{k}} = \hat{\boldsymbol{i}} = -\hat{\boldsymbol{k}}\hat{\boldsymbol{J}} \\ \hat{\boldsymbol{k}}\hat{\boldsymbol{i}} = \hat{\boldsymbol{j}} = -\hat{\boldsymbol{i}}\hat{\boldsymbol{k}} \end{aligned} \tag{1.17}$$

It will also be natural to state that the product $qq^{'}$, which is the result of multiplying q_1 as a multiplier into $q^{'}$ as a multiplicand, can be formulated as

$$\begin{aligned} qq' &= \left(q_0 + q_1\hat{\imath} + q_2\hat{\jmath} + q_3\hat{k}\right)\left(q_0' + q_1'\hat{\imath} + q_2'\hat{\jmath} + q_3'\hat{k}\right) \\ &= (q_0q_0') - (q_1q_1' + q_2q_2' + q_3q_3') + q_0\left(q_1'\hat{\imath} + q_2'\hat{\jmath} + q_3'\hat{k}\right) \\ &+ q_0'\left(q_1\hat{\imath} + q_2\hat{\jmath} + q_3\hat{k}\right) + \hat{\imath}\,(q_2q_3' - q_3q_2') + \hat{\jmath}\,(q_3q_1' - q_1q_3') + \hat{k}\,(q_1q_2' - q_2q_1') \end{aligned} \tag{1.18}$$

$$qq' = \begin{bmatrix} q_0q_0' - q_1q_1' - q_2q_2' - q_3q_3' \\ q_0q_1' + q_0'q_1 + q_2q_3' - q_3q_2' \\ q_0q_2' + q_0'q_2 + q_3q_1' - q_1q_3' \\ q_0q_3' + q_0'q_3 + q_1q_2' - q_2q_1' \end{bmatrix} = \begin{bmatrix} q_0q_0' - \boldsymbol{q}^\top\boldsymbol{q}' \\ q_0\boldsymbol{q}' + q_0'\boldsymbol{q} + \boldsymbol{q}\times\boldsymbol{q}' \end{bmatrix} \tag{1.19}$$

Utilizing the inner and cross product of two vector in $\mathbb{R}^3$ in order to rewrite the above expression in more concise form

$$qq' = q_0q_0' - \boldsymbol{q}\cdot\boldsymbol{q}' + q_0\boldsymbol{q} + q_0'\boldsymbol{q} + \boldsymbol{q}\times\boldsymbol{q}' \tag{1.20}$$

1.3.4 Quaternion Complex Conjugate and the Norm

The complex conjugate of a quaternion, an important algebraic notion relevant to quaternions as well as conventional complex numbers, will be beneficial in what follows. Let q be a quaternion defined in (1.7). The conjugate of q will be donated as $\bar{q}$ and is defined as

$$\bar{q} = q_0 - q_1\hat{\imath} - q_2\hat{\jmath} - q_3\hat{k} \quad \Leftrightarrow \quad \bar{q} = q_0 - \boldsymbol{q} \tag{1.21}$$

From the definition we consequently have the following

$$\begin{aligned} \overline{(\bar{q})} = \overline{(q_0 - \boldsymbol{q})} = (q_0 + \boldsymbol{q}) = q, \\ \bar{q} + q = 2q_0 \end{aligned}$$

The norm of a quaternion q, denoted by

$$\|q\| \triangleq \sqrt{q\bar{q}} \tag{1.22}$$

$$\begin{aligned} \bar{q}q &= (q_0 - \boldsymbol{q})(q_0 + \boldsymbol{q}) \\ &= q_0q_0 - (-\boldsymbol{q})\cdot\boldsymbol{q} + q_0\boldsymbol{q} + (-\boldsymbol{q})q_0 + (-\boldsymbol{q})\times\boldsymbol{q} = q_0^2 + \boldsymbol{q}\cdot\boldsymbol{q} \\ &= q_0^2 + \boldsymbol{q}\cdot\boldsymbol{q} = q_0^2 + q_1^2 + q_2^2 + q_3^2 = q\bar{q} \end{aligned} \tag{1.23}$$

$$\|q\| = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2} \in \mathbb{R} \tag{1.24}$$

A quaternion is named a unit or normalized quaternion if it has a norm equals to 1. If a quaternion has norm 1 each of its components must have absolute value less than or equal to one [2].

1.3.5 Natural power and the exponential of pure quaternions

The quaternion exponential function is a quaternion-based version of the exponential function. It is defined as the totally convergent power series, much like in the real exponential case [3].

$$e^q = \sum_{k=0}^{\infty} \frac{1}{k!} q^k \in \mathbb{H} \tag{1.25}$$

More remarkably, the exponential representation of a pure quaternion $\boldsymbol{v} = v_1\hat{\imath} + v_2\hat{\jmath} + v_3\hat{}$ is a new quaternion defined by

$$e^v = \sum_{k=0}^{\infty} \frac{1}{k!} v^k \subset \mathbb{I} \tag{1.26}$$

Let us define q^n, $n \in \mathbb{N}$, as the *n-th* power of q using the quaternion product. Then, if $\boldsymbol{v}$ is a pure quaternion and we let $\boldsymbol{v} = \hat{\boldsymbol{a}}\theta$, with $\theta = \|\boldsymbol{v}\| \in \mathbb{R}$ and $\hat{\boldsymbol{a}}$ unitary, we get from the definition of quaternion product the cyclic pattern

$$v^2 = -\theta^2, \quad v^3 = -\widehat{\boldsymbol{a}}\theta^3, \quad v^4 = \theta^4, \quad v^5 = \widehat{\boldsymbol{a}}\theta^5 \tag{1.27}$$

From the cyclic pattern retrieved in (1.27), and grouping the scalar and vector terms in the series

$$e^v = e^{\hat{\boldsymbol{a}}\theta} = \left(1 - \frac{\theta^2}{2!} + \frac{\theta^4}{4!} + \cdots\right) + \left(\hat{\boldsymbol{a}}\theta - \frac{\hat{\boldsymbol{a}}\theta^3}{3!} + \frac{\hat{\boldsymbol{a}}\theta^5}{5!} + \cdots\right) \tag{1.28}$$

and recognize in them, correspondingly, the series expansion of $cos\theta$, and $sin\theta$

$$e^v = e^{\hat{a}\theta} = \cos\frac{\theta}{2} + \hat{\boldsymbol{a}}\sin\frac{\theta}{2} = \begin{bmatrix} \cos\frac{\theta}{2} \\ \hat{\boldsymbol{a}}\sin\frac{\theta}{2} \end{bmatrix} \tag{1.29}$$

which establishes a stunning extension of the Euler formula, $e^{i\theta} = cos\theta + isin\theta$, defined for $\mathbb{I}$. Notice that since $\|e^v\|^2 = cos^2\theta{+}sin^2\theta = 1$, the exponential representation of a pure quaternion is a unit quaternion. Consequently, for any unit quaternion

$$q = q_0 + \boldsymbol{q} = cos\frac{\theta}{2} + \hat{\boldsymbol{a}}sin\frac{\theta}{2} \tag{1.30}$$

1.3.6 Rotation Operator

For any vector $v \in \mathbb{R}^3 \subset \mathbb{H}$ the action of the operator

$$\mathcal{L}(\boldsymbol{v}) = q\boldsymbol{v}\bar{q} \tag{1.31}$$

on $\boldsymbol{v}$ is equivalent to a rotation of the vector through an angle θ about $\hat{u}$ as the axis of rotation.

The analysis to prove this theorem will be based on getting the expression of the vector $\boldsymbol{w}$, the image of vector $\boldsymbol{v}$ after rotation about the unit quaternion, shown in the figure 1.3, geometrically, then getting an expression using quaternion will be called the quaternion rotation operator.

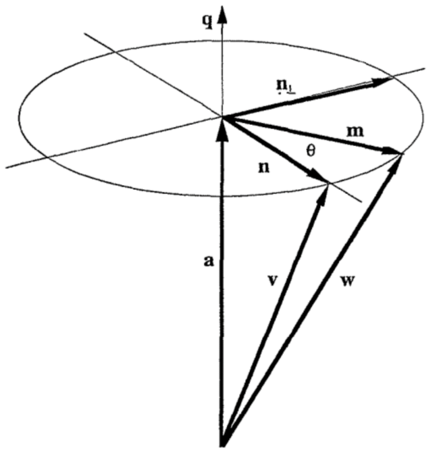


Figure 1.3: Vector $\boldsymbol{v}$ rotated with angle θ to be vector $\boldsymbol{w}$

$$\boldsymbol{w} = \widehat{\boldsymbol{a}} + \boldsymbol{m} \tag{1.32}$$

For getting the expression of the vector $\hat{\boldsymbol{a}}$, the projection of $\boldsymbol{v}$ on the direction of $\hat{\boldsymbol{a}}$

$$\begin{gathered}\boldsymbol{a} = (v.\widehat{\boldsymbol{a}}).\widehat{\boldsymbol{a}} \\ \boldsymbol{n} = \boldsymbol{v} - \widehat{\boldsymbol{a}} = \boldsymbol{v} - (\boldsymbol{v} \cdot \widehat{\boldsymbol{a}}) \cdot \widehat{\boldsymbol{a}} \\ \boldsymbol{n}_\perp = \widehat{\boldsymbol{a}} \times \boldsymbol{v} \\ \boldsymbol{m} = \boldsymbol{n}\cos(\theta) + \boldsymbol{n}_\perp \sin(\theta) = \cos(\theta)(\boldsymbol{v} - (\boldsymbol{v} \cdot \widehat{\boldsymbol{a}}) \cdot \widehat{\boldsymbol{a}}) + \sin(\theta)(\widehat{\boldsymbol{a}} \times \boldsymbol{v})\end{gathered}$$

Therefore,

$$\boldsymbol{w} = (\boldsymbol{v}.\widehat{\boldsymbol{a}}) \cdot \widehat{\boldsymbol{a}} + \cos(\theta)(\boldsymbol{v} - (\boldsymbol{v} \cdot \boldsymbol{a}) \cdot \widehat{\boldsymbol{a}}) + \sin(\theta)(\widehat{\boldsymbol{a}} \times \boldsymbol{v}) \tag{1.33}$$

Equation (1.33) will be revisited after getting the expression of $\mathcal{L}(\boldsymbol{v}) = q\boldsymbol{v}\bar{q}$. Extending the analysis to calculate the value of the operator $\mathcal{L}(\boldsymbol{v}) = q\boldsymbol{v}\bar{q}$.

$$qv = -\sin\left(\frac{\theta}{2}\right)(\widehat{\boldsymbol{a}} \cdot \boldsymbol{v}) + \left(\boldsymbol{v} + \sin\left(\frac{\theta}{2}\right)(\widehat{\boldsymbol{a}} \times \boldsymbol{v})\right) \tag{1.34}$$

Given,

$$\bar{q} = \cos\left(\frac{\theta}{2}\right) - \sin\left(\frac{\theta}{2}\right)\widehat{\boldsymbol{a}} \tag{1.35}$$

$$\begin{aligned}
\mathcal{L}(\boldsymbol{v}) = q\boldsymbol{v}\bar{q} &= 0 + \sin^2\left(\frac{\theta}{2}\right)(\widehat{\boldsymbol{a}}\cdot\boldsymbol{v})\widehat{\boldsymbol{a}} + \cos^2\left(\frac{\theta}{2}\right)\boldsymbol{v} \\
&+ \cos\left(\frac{\theta}{2}\right)\sin\left(\frac{\theta}{2}\right)(\widehat{\boldsymbol{a}}\times\boldsymbol{v}) + \sin\left(\frac{\theta}{2}\right)\widehat{\boldsymbol{a}}\times\left(\cos\left(\frac{\theta}{2}\right)\boldsymbol{v} + \sin\left(\frac{\theta}{2}\right)(\widehat{\boldsymbol{a}}\times\boldsymbol{v})\right) \\
&= \sin^2\left(\frac{\theta}{2}\right)(\widehat{\boldsymbol{a}}\cdot\boldsymbol{v})\widehat{\boldsymbol{a}} + \cos^2\left(\frac{\theta}{2}\right)\boldsymbol{v} + \sin(\theta)(\widehat{\boldsymbol{a}}\times\boldsymbol{v}) + \sin^2\left(\frac{\theta}{2}\right)(\widehat{\boldsymbol{a}}\times(\widehat{\boldsymbol{a}}\times\boldsymbol{v}))
\end{aligned}$$

Using the identity of

$$\boldsymbol{x}\times(\boldsymbol{y}\times\boldsymbol{c}) = (\boldsymbol{x}.\boldsymbol{c})\boldsymbol{y} + (\boldsymbol{x}.\boldsymbol{y})\boldsymbol{c} \tag{1.36}$$

$$\mathcal{L}(\boldsymbol{v}) = qv\bar{q} = \left[\cos^2\left(\frac{\theta}{2}\right) - \sin^2\left(\frac{\theta}{2}\right)\right]\boldsymbol{v} + \sin(\theta)(\widehat{\boldsymbol{a}}\times\boldsymbol{v}) + 2\sin^2\left(\frac{\theta}{2}\right)(\widehat{\boldsymbol{a}}(\widehat{\boldsymbol{a}}\cdot\boldsymbol{v}) - \boldsymbol{v})$$

$$+\sin^2\left(\frac{\theta}{2}\right)\widehat{\boldsymbol{a}}(\widehat{\boldsymbol{a}}\cdot\boldsymbol{v})$$

$$\mathcal{L}(\boldsymbol{v}) = q\boldsymbol{v}\bar{q} = \cos(\theta)\boldsymbol{v} + \sin(\theta)(\widehat{\boldsymbol{a}}\times\boldsymbol{v}) + (1-\cos(\theta))\widehat{\boldsymbol{a}}(\widehat{\boldsymbol{a}}\cdot\boldsymbol{v}) = \boldsymbol{w} \tag{1.37}$$

Equation (1.33) is exactly the same as equation (1.37) which verifies the action of the operator.

CHAPTER 2

COORDINATE SYSTEMS AND REFERENCE FRAMES

This section defines the reference frames used throughout this thesis. They are introduced to describe rigid body rotation and to define vectors in $\mathbb{R}^3$. A reference frame is a term used to describe a right-handed three-dimensional Cartesian coordinate system that is described by three mutually perpendicular unit vectors. Calculations are made easier by using multiple reference frames that have been carefully placed. They are introduced in figure 2.1. They are divided into two Geo-based frames and two satellite based frames.

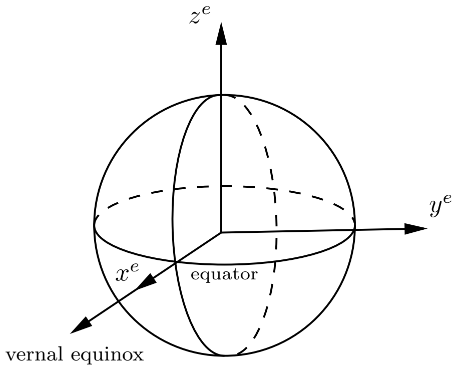
2.1 Earth-Based Reference Frames

2.1.1 Earth Centered Earth Fixed Frame

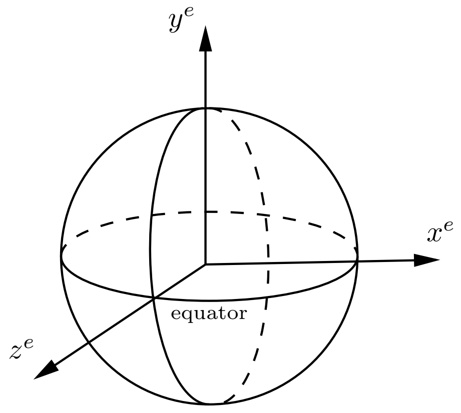
The origin of ECEF system is the earth's center. This frame of reference rotates with the earth which means if there's any fixed point on the surface of the earth, its coordinates will not change. The x^eaxis is passing through the Equator and Greenwich Meridian intersection. The z^e axis is passing by the North Pole. The y^e axis is orthogonal to x^e axis and z_e axis. The configuration is shown in figure 2.1b.

2.1.2 Earth Centered Inertial Frame

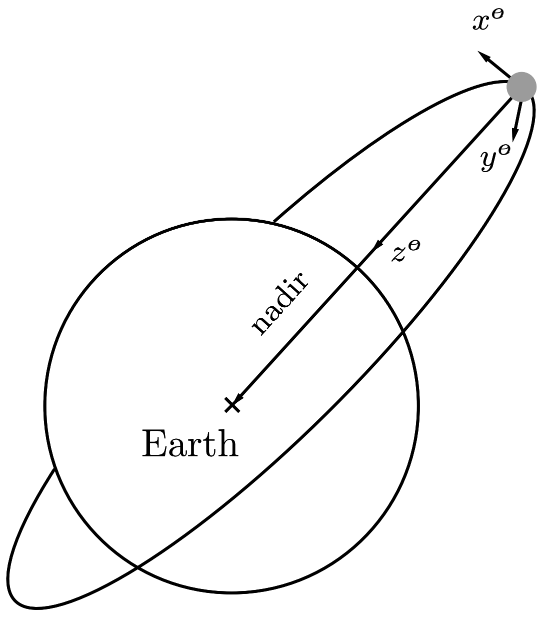
ECI coordinate system is a reference system where the origin is at the earth's center of mass. Unlike the ECEF frame the ECI frame doesn't rotate with the earth. The x^i axis is fixed towards the mean vernal equinox. The z^i axis corresponds to the mean north celestial pole of epoch J2000. The y^i axis is the vector product of the z^i and x^i axes. The configuration is shown in figure 2.1a



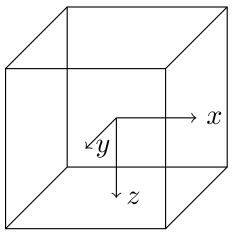
(a) Earth centered inertial reference frame



(b) Earth centered Earth fixed reference frame



(c) Orbit reference frame



(d) Satellite body reference frame

Figure 2.1: The reference frames used in this thesis.

2.1.3 The Geodetic Coordinate System

The geodetic coordinate system; also called the Latitude and Longitude coordinate system. It consists of the geodetic grid for the planet which is mainly parallel East/West lines representing the latitude and North/South lines representing the longitude which intersect at the poles. The lines of the Latitude and Longitude are ordered by the angle they subtend relative to a certain reference.

The latitude angle ϕ is the angle between the plane of the equator and the point of interest. The zero reference for the latitude is the Equator. The longitude geodetic location does not have an intuitive understanding of distance because the longitude lines are not parallel. The zero reference is the Greenwich Meridian.

2.2 Satellite-Based Reference Frames

2.2.1 Orbit reference frame

The ORF maintains its orientation with respect to the Earth and tracks the satellite in its orbit. The attitude of the satellite with respect to this coordinate system is also referred to as roll, pitch, and yaw. The z^o axis points out from the satellite along the geocentric radius vector. The y^o axis is normal to the orbital plane which is created from the position vector and velocity vector. The x^o axis is normal to the position vector and positive in the direction of the velocity vector.it is also aligned with the velocity vector only for circular orbits. The configuration is shown in figure 2.1c.

2.2.2 Satellite body reference frame

This coordinate system is used to define ADCS measurements. The origin of this system is the center of mass. The axis are chosen as the x^b points through the front of the satellite. The y^b points to the right of the x^b and is perpendicular to it. The z^b is pointing down the satellite and perpendicular to the $x^b - y^b$ plane.

CHAPTER 3

SPACE ENVIRONMENT

3.1 Geomagnetic Field – IGRF

Although the broad properties of the Geomagnetic field have been known for centuries, the first systematic study of the field was begun in the early nineteenth century by German mathematician and physicist Gauss as found in Wertz [4]. Jeffrey [5] stating that around 170 observatories networks around the globe are used to report geomagnetic data. Throughout 120 observatories now produce and frequently report digital data with a one-minute collection cadence or better; figure 3.1 below depicts their locations around the world. The remaining 50 or so observatories rely on outdated, analogue equipment or provide data years after it has been collected.

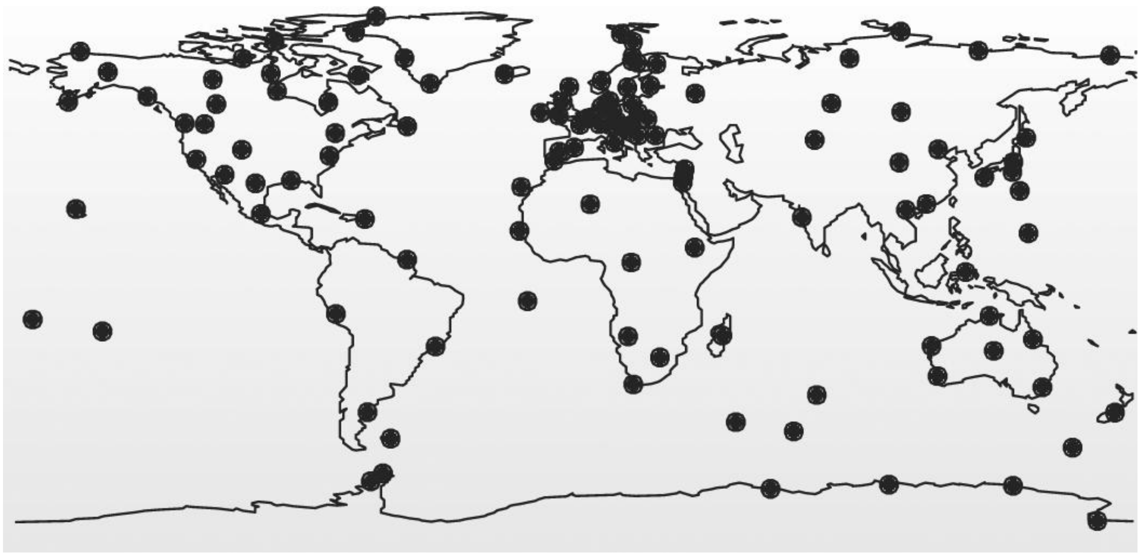


Figure 3.1: Worldwide Distribution of Geomagnetic Observatories

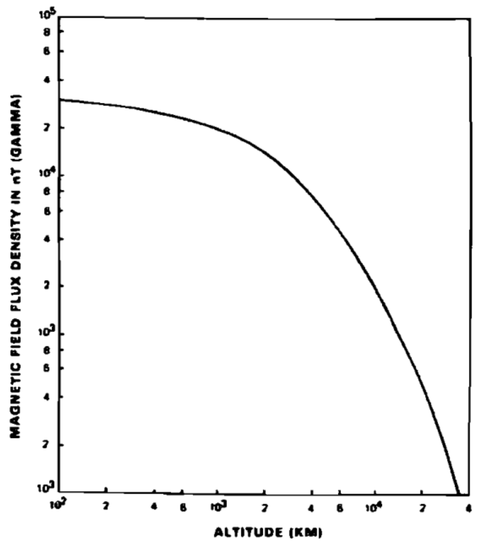
Wertz [4] elaborated that despite the fact that this collection of data has improved our capacity to precisely characterise the field, it still lacks the key to the physical processes that make or disturb it. As a result, we shall discuss the observed occurrences in this part and, when feasible, present compelling justifications for their existence.

3.1.1 Geomagnetic Field Definition

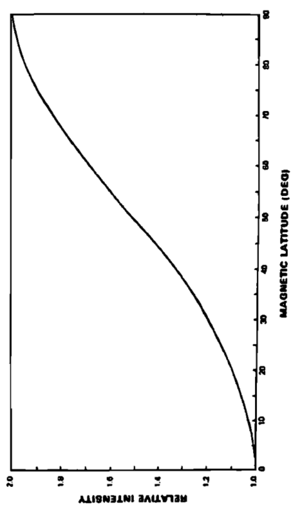
The geomagnetic field is primarily that of a magnetic dipole, like that produced by a sphere of uniform magnetization or a current loop. In 1975, the dipole's strength was 7.96×10^{15} Wb.m. The dipole's "south" end was located at 78.60° N latitude and 289.55° E longitude in the northern hemisphere, migrating westward at a rate of around 0.014 degrees each year. The dipole strength is dropping at a rate of 0.05 percent every year.

The magnetic equator is the plane perpendicular to the Earth-centered dipole. The field is weakest there, with a strength of around 3×10^4 nT near the Earth's surface. At the magnetic equator, figure 3.2a below depicts the fluctuation in dipole field intensity as a function of height [4].

As the magnetic latitude increases from 0 to 90 degrees, the field strength increases by a factor of two, as seen in figure 3.2b. The field is horizontal relative to the Earth's surface at the geomagnetic equator. The field is 45 degrees down from horizontal at a geomagnetic latitude of around 27 degrees [4].



(a) Fluctuation in dipole field intensity as a function of height.



(b) Field strength as function of magnetic latitude.

Figure 3.2: Magnetic intensity as function of altitude and latitude.

3.1.2 Mathematical Model

The International Geomagnetic Reference Field (IGRF) is a set of spherical harmonic coefficients that may be entered into a mathematical model to explain the big, time-varying component of the Earth's core magnetic field between 1900 A.D. and the present. Under the auspices of the Multinational Association of Geomagnetism and Aeronomy (IAGA) Working Group V-MOD, an international task force of scientists creates and maintains the IGRF [6].

The IGRF defines the primary geomagnetic field $B(x, y, z, t)$ created by internal sources largely within the Earth's core. The IGRF is valid on and above the Earth's surface, where the primary geomagnetic field may be characterised by the following equation.

$$\mathbf{B}_\lambda(x, y, z, t) = -R_\oplus^2 \times Real \left\{ \sum_{n=1}^{\infty} \sum_{m=0}^{n} R_\oplus^n (g_n^m - h_n^m i) \frac{\partial V_{n,m}}{\partial \lambda} d_{n,m} \right\} \tag{3.1}$$

Where,

$$\lambda = x, y, z.$$
$$i = \sqrt{-1}.$$
$$R_\oplus \triangleq \text{Earth's radius.}$$
$$g_n,\ h_n \triangleq \text{The Gaussian magnetic field coefficients.}$$

The values for $d_{n,m}$ and $V_{n,m}$ can be calculated as following.

$$d_{1,0} = 1,\ d_{1,1} = 1,\ d_{0,0} = 1 \tag{3.2}$$

$$d_{n,n} = \frac{d_{n-1,n-1}}{\sqrt{(2n-1) \times 2n}} \tag{3.3}$$

$$d_{n,m} = \sqrt{2 \times \frac{(n-m)!}{(n+m)!}} \tag{3.4}$$

$$V_{n,n} = (2n-1) \times \frac{(x+iy) \times V_{n-1,n-1}}{r^2} \tag{3.5}$$

$$V_{n,m} = \begin{cases} \dfrac{(2n-1) \times z \times V_{n-1,m}}{(n-m) \times r^2}, & \text{if } n-m = 1 \\ \dfrac{(2n-1) \times z \times V_{n-1,m}}{(n-m) \times r^2} - \dfrac{(n+m-1) \times V_{n-2,m}}{(n-m) \times r^2}, & \text{if } n-m > 1 \end{cases} \tag{3.6}$$

$$\frac{\partial V_{n,m}}{\partial x} = \begin{cases} -\dfrac{1}{2} \times \left(V_{n+1,m} + \overline{V}_{n+1,1} \right), & \text{if } m = 0 \\ -\dfrac{1}{2} \times \left(V_{n+1,m+1} - \dfrac{(n-m+2)!}{(n-m)!} V_{n+1,m-1} \right), & \text{if } m > 1 \end{cases} \tag{3.7}$$

$$\frac{\partial V_{n,m}}{\partial y} = \begin{cases} -\dfrac{i}{2} \times \left(V_{n+1,m} - \overline{V}_{n+1,1} \right), & \text{if } m = 0 \\ -\dfrac{i}{2} \times \left(V_{n+1,m+1} + \dfrac{(n-m+2)!}{(n-m)!} V_{n+1,m-1} \right), & \text{if } m > 1 \end{cases} \tag{3.8}$$

$$\frac{\partial V_{n,m}}{\partial z} = \frac{(n-m+1)!}{(n-m)!} V_{n+1,m} \tag{3.9}$$

Where the complex-conjugate value of function $V_{n+1,1}$ is denoted as $\overline{V}_{n+1,1}$.

The values of the coefficients gn,m and hn,m of the first harmonic up to 13 orders inclusive, are retrieved from the International Geomagnetic Reference Field

13th edition [6]. Using the coefficients presented in the IGRF 13th edition, the upcoming coefficients for the consecutive years can be calculated using the secular variation by the following equations.

$$g_n^m(t) = g_n^m(T_t) + (t - T_t)\dot{g}_n^m(T_t) \tag{3.10}$$

$$h_n^m(t) = h_n^m(T_t) + (t - T_t)\dot{h}_n^m(T_t) \tag{3.11}$$

Where $g_n^m(T_t)$ and $h_n^m(T_t)$ are the Gauss coefficients at the time t preceding epoch T_t. The model epochs in IGRF-13 are given in precise multiples of 5 years beginning in 1900 and ending in 2020 (see Table 3.1 and 3.2 for 2020 values), hence $T_t \leqslant t < T_t{+}5$. The specifications for $T_t < 2020$.

n/m	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	-29404.8	-1450.9	0	0	0	0	0	0	0	0	0	0	0	0
2	-2499.6	2982.0	1677.0	0	0	0	0	0	0	0	0	0	0	0
3	1363.2	-2381.2	1236.2	525.7	0	0	0	0	0	0	0	0	0	0
4	903.0	809.5	86.3	-309.4	48.0	0	0	0	0	0	0	0	0	0
5	-234.3	363.2	187.8	-140.7	-151.2	13.5	0	0	0	0	0	0	0	0
6	66.0	65.5	72.9	-121.5	-36.2	13.5	-64.7	0	0	0	0	0	0	0
7	80.6	-76.7	-8.2	56.5	15.8	6.4	-7.2	9.8	0	0	0	0	0	0
8	23.7	9.7	-17.6	-0.5	-21.1	15.3	13.7	-16.5	-0.3	0	0	0	0	0
9	5.0	8.4	2.9	-1.5	-1.1	-13.2	1.1	8.8	-9.3	-11.9	0	0	0	0
10	-1.9	-6.2	-0.1	1.7	-0.9	0.7	-0.9	1.9	1.4	-2.4	-3.8	0	0	0
11	3.0	-1.4	-2.5	2.3	-0.9	0.3	-0.7	-0.1	1.4	-0.6	0.2	3.1	0	0
12	-2.0	-0.1	-0.1	1.3	-1.2	0.7	0.3	0.5	-0.3	-0.5	0.1	-1.1	-0.3	0
13	0.1	-0.9	0.5	0.7	-0.3	0.8	0.0	0.8	0.0	0.4	0.1	0.5	-0.5	-0.4

Table 3.1: g_n^m Values as provided in IGRF-13 Generation

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	4652.5	0	0	0	0	0	0	0	0	0	0	0	0
2	0	-2991.6	-734.6	0	0	0	0	0	0	0	0	0	0	0
3	0	-82.1	241.9	-543.4	0	0	0	0	0	0	0	0	0	0
4	0	281.9	-158.4	199.7	-349.7	0	0	0	0	0	0	0	0	0
5	0	47.7	208.3	-121.2	32.3	98.9	0	0	0	0	0	0	0	0
6	0	-19.1	25.1	52.8	-64.5	8.9	68.1	0	0	0	0	0	0	0
7	0	-51.5	-16.9	2.2	23.5	-2.2	-27.2	-1.8	0	0	0	0	0	0
8	0	8.4	-15.3	12.8	-11.7	14.9	3.6	-6.9	2.8	0	0	0	0	0
9	0	-23.4	11	9.8	-5.1	-6.3	7.8	0.4	-1.4	9.6	0	0	0	0
10	0	3.4	-0.2	3.6	4.8	-8.6	-0.1	-4.3	-3.4	-0.1	-8.8	0	0	0
11	0	0	2.5	-0.6	-0.4	0.6	-0.2	-1.7	-1.6	-3	-2	-2.6	0	0
12	0	-1.2	0.5	1.4	-1.8	0.1	0.8	-0.2	0.6	0.2	-0.9	0	0.5	0
13	0	-0.9	0.6	1.4	-0.4	-1.3	-0.1	0.3	-0.1	0.5	0.5	-0.4	-0.4	-0.6

Table 3.2: h_n^m Values as provided in IGRF-13 Generation

3.1.3 MATLAB Model Implementation

calculate_d_nm.m

MATLAB-function to compute the value of $d_{n,m}$. The function takes no input and the output is described in table 3.3.

Table 3.3: IN/OUT parameters of calculate_d_nm.m

calculate_d_nm.m: calculates the value of $d_{n,m}$ function		
IN/OUT	**Var. Name**	**Description**
OUT	d_nm	14-by-14 matrix storing the values of d_nm

calculate_v.m

MATLAB-function to compute the value of V. The IN/OUT parameters are described in table 3.4.

Table 3.4: IN/OUT parameters of calculate_v.m

calculate_v.m: calculates the value of $d_{n,m}$ function		
IN/OUT	**Var. Name**	**Description**
OUT	v	14-by-14 matrix storing the values of V
IN	x	x coordinate of the point of interest
IN	y	y coordinate of the point of interest
IN	z	z coordinate of the point of interest

calc_partial_v.m

MATLAB-function to compute $\partial V/\partial\lambda$. The IN/OUT parameters are described in table 3.5.

Table 3.5: IN/OUT parameters of calc_partial_v.m

calc_partial_v.m: calculates the partial derivative of v with respect to x, y, and z.		
IN/OUT	**Var. Name**	**Description**
OUT	partial_v_x	the derivative of V with respect to x
OUT	partial_v_y	the derivative of V with respect to y
OUT	partial_v_z	the derivative of V with respect to z
IN	v	14-by-14 matrix containing the values of v

read_coef.m

MATLAB-function to read the coefficients of the IGRF model. The function takes no input and the output is described in table 3.6.

Table 3.6: IN/OUT parameters of read_coef.m

read_coef.m: reads the IGRF Coefficients		
IN/OUT	**Var. Name**	**Description**
OUT	g_nm	the values of g_{nm}
OUT	h_nm	the values of h_{nm}
OUT	g_nm_dot	the values of $\dot{g}_{n,m}$
OUT	h_nm_dot	the values of $\dot{h}_{n,m}$

magneticFieldModel.m

MATLAB-function to model the IGRF mathematical representation discussed in section 3.1.2. It computes the magnetic field components in $x, y,$ and, z directions as well as the total magnetic field in ECI. The IN/OUT attributes is described in table 3.7.

Table 3.7: IN/OUT parameters of magneticFieldModel.m

magneticFieldModel.m: Modeling the geo-magnetic field		
IN/OUT	**Var. Name**	**Description**
OUT	B_x	the geomagnetic field component in x direction
OUT	B_y	the geomagnetic field component in y direction
OUT	B_z	the geomagnetic field component in z direction
OUT	total	the total geomagnetic field
IN	lat	the latitude of the point of interest
IN	lon	the longitude of the point of interest
IN	lat	the altitude of the point of interest

3.1.4 Results of The MATLAB Model

A MATLAB code is made for this mathematical model in order to get the earths' magnetic field contours at given altitude and time. Figure 3.3 shows the Total Field at the ellipsoid surface for epoch 2020 at earths' surface.

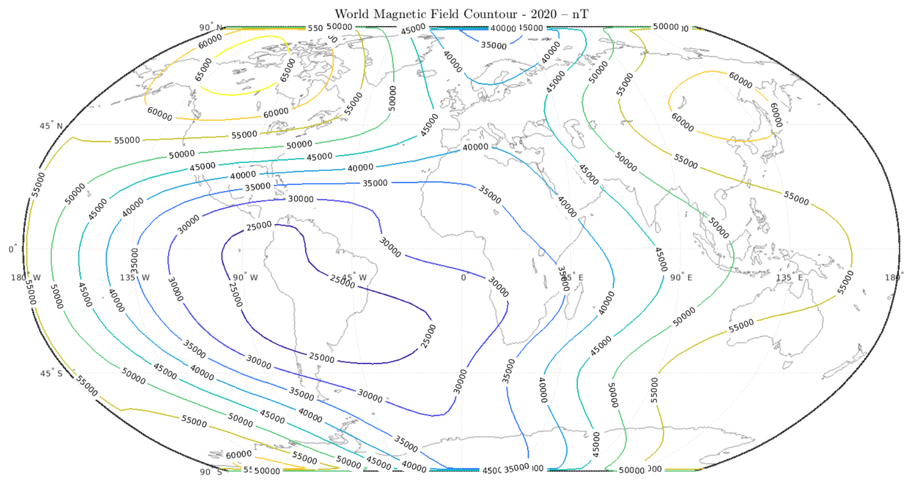


Figure 3.3: Total Geomagnetic Field contours in nT as depicted by MATLAB simulation.

3.2 Geo-Gravitational Field – Spherical Harmonics Gravity

3.2.1 Definition

Two point masses, M and m, separated by a vector distance r, are attracted to one other by a force described by Newton's law of gravity

$$F = G\frac{Mm}{r^2}\bar{r} \tag{3.12}$$

If $M_\oplus$ is the mass of the Earth and m is the mass of the body whose motion we wish to follow, then it is convenient to define the geocentric gravitational constant, $\mu_\oplus$, and the Earth gravitational potential, $\boldsymbol{U}$ by

$$\mu_\oplus = GM_\oplus$$
$$\boldsymbol{U} = -\frac{GM_\oplus}{r},$$

The force equation becomes

$$\boldsymbol{F} = -\frac{\mu_\oplus m}{r^2}\bar{\boldsymbol{r}} = -m\nabla\boldsymbol{U} \tag{3.13}$$

3.2.2 Mathematical Model

For this Model a MATLAB code is carried out and can be found in appendix **??**.

$$\frac{\partial U}{\partial \lambda} = \text{real}\left(\sum_{n=1}^{\infty}\sum_{m=0}^{n} R_e^n\left(C_{nm} - j * S_{nm}\right) * \frac{\partial V_{n,m}}{\partial \lambda}\right) \tag{3.14}$$

Where

$$\lambda = x, y, z.$$
$$i = \sqrt{-1}.$$
$$R_\oplus \triangleq \text{Earth's radius.}$$
$$C_{nm},\ S_{nm} \triangleq \text{The Earth's gravity coefficients.}$$

$$V_{0,0} = \frac{1}{r} \tag{3.15}$$

$$V_{n,n} = (2n-1) \times \frac{(x+iy) \times V_{n-1,n-1}}{r^2} \tag{3.16}$$

$$V_{n,m} = \begin{cases} \dfrac{(2n-1) \times z \times V_{n-1,m}}{(n-m) \times r^2}, & \text{if } n-m = 1 \\ \dfrac{(2n-1) \times z \times V_{n-1,m}}{(n-m) \times r^2} - \dfrac{(n+m-1) \times V_{n-2,m}}{(n-m) \times r^2}, & \text{if } n-m > 1 \end{cases} \tag{3.17}$$

$$\frac{\partial V_{n,m}}{\partial x} = \begin{cases} -\dfrac{1}{2} \times \left(V_{n+1,m} + \overline{V}_{n+1,1}\right), & \text{if } m = 0 \\ -\dfrac{1}{2} \times \left(V_{n+1,m+1} - \dfrac{(n-m+2)!}{(n-m)!}V_{n+1,m-1}\right), & \text{if } m > 1 \end{cases} \tag{3.18}$$

$$\frac{\partial V_{n,m}}{\partial x} = \begin{cases} -\frac{i}{2} \times \left(V_{n+1,m} - \overline{V}_{n+1,1}\right), & \text{if } m = 0 \\ -\frac{i}{2} \times \left(V_{n+1,m+1} + \frac{(n-m+2)!}{(n-m)!} V_{n+1,m-1}\right), & \text{if } m > 1 \end{cases} \tag{3.19}$$

$$\frac{\partial V_{n,m}}{\partial z} = \frac{(n-m+1)!}{(n-m)!} V_{n+1,m} \tag{3.20}$$

The values of each C_{nm} and S_{nm} are both retrieved from [4]. The gravity vector will be

$$\boldsymbol{g} = \mu_{\oplus} \begin{bmatrix} \frac{\partial U}{\partial x} \\ \frac{\partial U}{\partial y} \\ \frac{\partial U}{\partial z} \end{bmatrix} \tag{3.21}$$

3.2.3 MATLAB Model Implementation

calculate_v.m

MATLAB-function to compute the value of V. The IN/OUT parameters are described in table 3.8.

Table 3.8: IN/OUT parameters of calc_v.m

calculate_v.m: calculates the value of $d_{n,m}$ function		
IN/OUT	**Var. Name**	**Description**
OUT	v	7-by-7 matrix storing the values of V
IN	x	x coordinate of the point of interest
IN	y	y coordinate of the point of interest
IN	z	z coordinate of the point of interest

calculate_partial_v.m

MATLAB-function to compute $\partial V/\partial\lambda$. The IN/OUT parameters are described in table 3.9.

Table 3.9: IN/OUT parameters of calc_partial_v.m

calc_partial_v.m: calculates the partial derivative of v with respect to x, y, and z.		
IN/OUT	**Var. Name**	**Description**
OUT	partial_v_x	the derivative of V with respect to x
OUT	partial_v_y	the derivative of V with respect to y
OUT	partial_v_z	the derivative of V with respect to z
IN	v	7-by-7 matrix containing the values of v

read_coef.m

MATLAB-function to read the coefficients of the IGRF model. The function takes no input and the output is described in table 3.10.

Table 3.10: *MATLAB function* – reading $g_{n,m}$, $h_{n,m}$, $\dot{g}_{n,m}$, $\dot{h}_{n,m}$

read_coef.m: reads the IGRF Coefficients		
IN/OUT	**Var. Name**	**Description**
OUT	g_nm	the values of g_{nm}
OUT	h_nm	the values of h_{nm}
OUT	g_nm_dot	the values of $\dot{g}_{n,m}$
OUT	h_nm_dot	the values of $\dot{h}_{n,m}$

calculate_gravity.m

MATLAB-function to calculate the earth's gravity. It computes the gravity field components in x, y, and, z directions as well as the total gravity. The IN/OUT attributes is described in table 3.11.

Table 3.11: IN/OUT parameters of calculate_gravity.m

calculate_gravity.m: Modeling the gravity field		
IN/OUT	**Var. Name**	**Description**
OUT	g_x	the gravity field component in x direction
OUT	g_y	the gravity field component in y direction
OUT	g_z	the gravity field component in z direction
OUT	g$_tot$	the total gravity field
IN	x	the x coordinate in ECI of the point of interest
IN	y	the y coordinate in ECI of the point of interest
IN	z	the z coordinate in ECI of the point of interest

3.3 Eclipse Time

3.3.1 Definition

When a satellite orbiting the earth passes into a shadow zone, depriving the solar arrays of sunlight, the time duration for the orbit in this shadow zone change with the variation of the orbit type. This period of time is referred to as eclipse time, and the shadow zone is referred to as eclipse region. In the eclipse time, a secondary power source is used to provide power to the satellite's variant subsystems. Batteries are usually used as this secondary power source. Thus, calculating eclipse time is critical for battery sizing and maintaining a steady heat balance within the satellite.

The eclipse time was calculated using the assumption that the planet is spherical in shape. The Umbra and the Penumbra, as seen in figure **??**, are two separate conical projections of a planet's shadow. For ease of calculation, the umbra area has been regarded as a cylindrical projection of Earth. For low-altitude circular orbits, the assumptions are quite accurate, but for high-altitude or highly elliptical orbits, they may result in considerable errors.

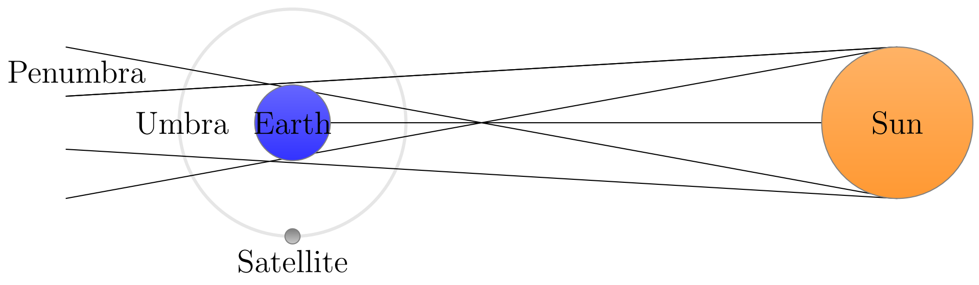


Figure 3.4: Umbra and penumbra regions from earth's shadow

The angle between the solar vector and its projection onto the orbital plane is known as the orbital beta angle β. The angle between the unit solar vector $\boldsymbol{s}$ and the unit vector normal to the orbital plane $\hat{\boldsymbol{n}}$ must be determined in order to calculate the beta angle.

3.3.2 Beta Angle Calculations

The orbital beta angle can be calculated by the following equation

$$\Phi = \beta + \frac{\pi}{2} \tag{3.22}$$

The configuration is shown in figure 3.5 retrieved from [7].

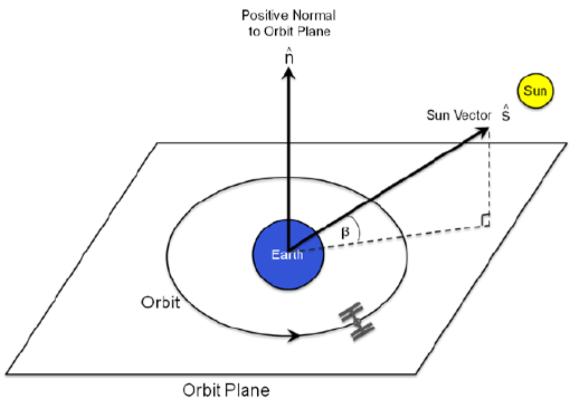


Figure 3.5: Solar beta angle definition

The unit solar vector $\boldsymbol{s}$ and unit normal vector $\hat{\boldsymbol{n}}$ can be obtained using two Euler angle transformations. The solar vector $\boldsymbol{s}$, which points towards the sun in the celestial inertial coordinate system, is determined by two parameters, namely, Right Ascension of the Sun Γ and Declination of the Sun ϵ.

Similarly, a vector n is defined as a vector that points normal to the orbital plane and is determined by two parameters: orbital inclination i and Right Ascension of Ascending Node Ω.

3.4 Julian Date

It is the time interval measured from epoch January 1, 4713 B.c., 12:00.

J_0 represents the Julian day number at 0 h UT (which is halfway into the Julian day). The Julian day is provided at any other UT by

$$JD = J_0 + UT \tag{3.23}$$

Several Algorithms for calculation J_0 exist in literature, One of the simplest formulas is found in Boulet in [8]. The algorithm of calculating the JD is shown as follows

$$J_0 = 367 \cdot y - \text{INT}\left\{\frac{7\left[y + \text{INT}\left(\frac{m+9}{12}\right)\right]}{4}\right\} + \text{INT}\left(\frac{275 \cdot m}{9}\right) + d + 1,721,013.5 \tag{3.24}$$

$INT(x)$ means to obtain the integer portion of x only without rounding. Where y, d, and m, are numbers lying in the following ranges:

$$1901 \le y \le 2099$$
$$1 \le m \le 12$$
$$1 \le d \le 31$$

The algorithm implemented in MATLAB can be found in table 3.12.

Table 3.12: IN/OUT parameters of Julian_day.m

Julian_day.m: Calculating the julian day		
IN/OUT	**Var. Name**	**Description**
OUT	J0	Julian day at 0 hr UT (Universal Time)
IN	day	range: 1 – 31
IN	month	range: 1 – 12
IN	year	range: range: 1901 to 2099

3.5 Sun Position Vector

Position vectors for the sun is needed to assess disturbance forces on satellites. When constructing a mission for sensor observation, sunrise or sunset circumstances are required. We may generate techniques for determining the local time of dawn and sunset using equations employed in coordinate reduction. This section describes useful methods for doing certain tasks [9].

A basic approach whose findings are in the form of a MoD (mean-equator of date) vector with an accuracy of 0.01° will be discussed and implemented. Because of the shortening of the expansions, it is valid from 1950 to 2050. The answer is based in part on equations that use the J2000.0 epoch [9]. The Algorithm can be implemented as following as proposed by [9]

1. Calculate the number of Julian centuries

$$T_0 = \frac{J_0 - 2,451,545}{36,525} \tag{3.25}$$

2. Get the value of the mean longitude of the sun

$$\lambda_{M\odot} = 280.460^{\circ} + 36,000T_0 \tag{3.26}$$

3. Get the ecliptic latitude of the Sun by

$$\lambda_{\text{ecliptic}} = \lambda_{M\odot} + 1.914666471\sin(M_\odot) + 0.0119994643\sin(2M_\odot) \text{ where } M_\odot = 357.291092^\circ + 35,000.05034T_{TDB} \tag{3.27}$$

4. Let $T_{TDB} = T_0$

5. Find the position magnitude using the yielded equation

$$r_\odot = 1.000140612 - 0.016708617\cos(M_\odot) - 0.000139589\cos(2M_\odot) \tag{3.28}$$

6. Find approximate value for the obliquity of the ecliptic using

$$\varepsilon = 23.439291^\circ - 0.0130042T_0 \tag{3.29}$$

7. The position vector in geocentric MoD is

$$\boldsymbol{r}_\odot = r_\odot\cos(\lambda_{\text{ecliptic}})\hat{I} + r_\odot\cos(\varepsilon)\sin(\lambda_{\text{ecliptic}})\hat{J} + r_\odot\sin(\varepsilon)\sin(\lambda_{\text{ecliptic}})\widehat{K} \tag{3.30}$$

8. The position vector in MoD is

$$\boldsymbol{r}_\odot = \begin{bmatrix} r_\odot\cos(\lambda_{\text{ecliptic}}) \\ r_\odot\cos(\varepsilon)\sin(\lambda_{\text{ecliptic}}) \\ r_\odot\sin(\varepsilon)\sin(\lambda_{\text{ecliptic}}) \end{bmatrix} AU \tag{3.31}$$

The algorithm implemented in MATLAB can be found in table 3.13.

Table 3.13: IN/OUT parameters of calc_sun_pos_vector.m

calc_sun_pos_vector.m: Calculating the sun position ector		
IN/OUT	**Var. Name**	**Description**
OUT	r	Sun position vector
IN	T0	Julian centuries
IN	promt	Km or AU

3.6 Sidereal Time

A solar day is the amount of time it takes for the sun to return to its original position which comprises 24 hours. The universal time is measured by the passage of the sun through Greenwich meridian. This meridian is zero degrees terrestrial longitudinal. The sun lies on this meridian at noon UT.

A solar day is the amount of time it takes for the sun to return to its original position which comprises 24 hours. The universal time is measured by the passage of the sun through Greenwich meridian. This meridian is zero degrees terrestrial longitudinal. The sun lies on this meridian at noon UT.

The rotation of the earth relative to the fixed stars (i.e., the celestial sphere) is used to calculate sidereal time. One sidereal day is the time it takes for a distant star to return to the same location overhead to lie on the same meridian. It consists of Sidereal hours. The sidereal day is slightly shorter than the solar day due to the earth's orbit around the sun. A sidereal day has a length of 23 hours and 56 minutes. The planet spins 360 degrees in a sidereal day and 360.986 degrees in a solar day. Knowing the position of a point on the earth in relation to the geocentric equatorial frame at any given time necessitates knowing its local sidereal time. The local sidereal time of a site is calculated by first determining the Greenwich sidereal time θ_G (the Greenwich meridian's sidereal time) and then adding (or subtracting) the site's east longitude [10]. The concept of Julian day is used in algorithms for determining sidereal time.

The Greenwich sidereal time θ_{G0} at $0hUT$ can be given as proposed by [11] as

$$\theta_{G0} = 100.4606184 + 36,000.77004T_0 + 0.000387933T_0^2 - 2.583\left(10^{-8}\right)T_0^3\big) \tag{3.32}$$

If the formula results in a value greater than 360° or smaller than 0°, multiple of 360 must added or subtracted from the value to adjust it to the range. Then, The Greenwich sidereal time θ_G can be calculated from by

$$\theta_G = \theta_{G0} + 360.98564724 * \frac{UT}{24} \tag{3.33}$$

The number of degrees an earth rotates within the day is given from

$$360.98564724 * \frac{UT}{24} \tag{3.34}$$

Finally, a site's local sidereal time Λ is computed by adding its east longitude L by the Greenwich sidereal time.

$$\theta = \theta_G + \Lambda \tag{3.35}$$

CHAPTER 4

SPACE PERTURBATIONS

For determining and controlling the attitude of the spacecraft, a brief study should be done to examine any disturbance in the space environment that may generate extra torque to change the position and orientation of the spacecraft.

A satellite's attitude is influenced by a myriad of different disturbances. These can include both internal disturbances caused by the satellite and external disturbances. Because the disturbances affect the satellite as torques, they are also referred to as torques in this section.

Aerodynamic drag in the upper earth's atmosphere, the magnetic and gravitational field, solar radiation pressure, and pressure from micrometeorites impacts are the primary external disturbances. However, in orbits around the Earth, the last mentioned is unlikely to take place [4]. Moreover, the solar radiation induced torque has low order of magnitude when compared with the aforementioned perturbations. These dominant regions are influenced by the geometry and mass distribution of the satellite in question. The external disturbances relevant to cubesat are described in detail in the following sections and are based on [4, 12].

The total external disturbance torque in the BRF is given by

$$\mathbf{M}_{\text{ext.}} = \mathbf{M}_{\text{drag}} + \mathbf{M}_{\text{GG}} + \mathbf{M}_{\text{mrs.}} \tag{4.1}$$

4.1 Gravity Gradient Disturbance

The components of the gravity vector change along the orbit of the satellite due to its non-symmetric nature and finite dimensions. The variation in the gravity components results in a gravitational torque applied on the body which makes a rotational effect on the body.

The gravitational force element $d\mathbf{F}_i$ acting on mass element dm_i has a position vector $\mathbf{R}_i$ relative to the geocenter is defined as

$$d\mathbf{F}_i = -\frac{\mu \mathbf{R}_i}{R_i} dm_i; \quad \mu \triangleq GM_{\oplus} \tag{4.2}$$

The torque induced from this element force $d\mathbf{F}_i$ about the spacecraft's geometric center, at position $\mathbf{r}_i$ relative to spacecraft's geometric center can be expressed as

$$d\mathbf{N}_i = \mathbf{r}_i \times d\mathbf{F}_i = (\boldsymbol{\rho} + \mathbf{r}') \times d\mathbf{F}_i \tag{4.3}$$

as shown in figure 4.1.

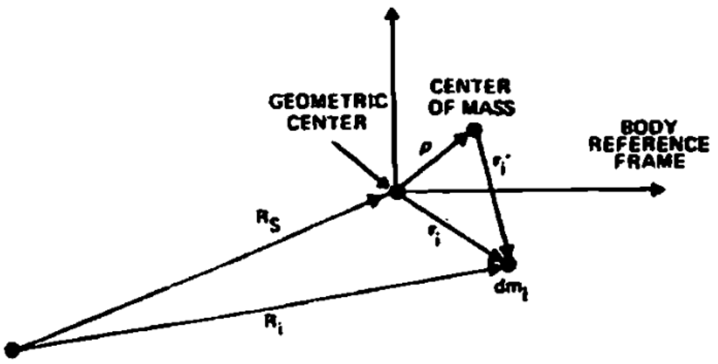


Figure 4.1: Coordinate System for the Calculation of Gravity-Gradient Torque

The total gravity-gradient torque acting on the entire spacecraft is then given by equation

$$\mathbf{M}_{GG} = \int \mathbf{r}_i \times \mathrm{d}\mathbf{F}_i = \int (\boldsymbol{\rho} + \mathbf{r}_i') \times \frac{-\mu \mathbf{R}_i}{R_i^3}\, \mathrm{d}m_i \tag{4.4}$$

The geocentric position vector for the ith mass element can be expressed in terms of the geocentric position vector of the origin of the body reference frame, $\mathbf{R}_s$, as

$$\mathbf{R}_i = \mathbf{R}_s + \mathbf{r}_i = \mathbf{R}_s + \boldsymbol{\rho} + \mathbf{r}'_i \tag{4.5}$$

For a real satellite situation $\mathbf{R}_i = \mathbf{R}_S + \boldsymbol{\rho} + \mathbf{r}_i' \gg \boldsymbol{\rho} + \mathbf{r}_i'$, then

$$\mathbf{R}_i^{-3} \approx \mathbf{R}_s^{-3} \left[1 - \frac{3\boldsymbol{R}_s \left(\boldsymbol{\rho} + \boldsymbol{r}_i' \right)}{R_s^2} \right] \tag{4.6}$$

The gravity gradient can be written as

$$\mathbf{M}_{GG} = \frac{\mu M}{R_S^2} \left(\boldsymbol{R}_s \times \boldsymbol{\rho} \right) + \frac{3\mu}{R_s^3} \int \left(\mathbf{r}_i \times \boldsymbol{R}_s \right) \left(\mathbf{r}_i \cdot \boldsymbol{R}_s \right) dm_i \tag{4.7}$$

According to [4] equation 4.7 can be simplified as

$$\mathbf{M}_{GG} = \frac{3\mu}{R_s^3} \left(\mathbf{R}_s \times ([\mathbf{I}] \cdot \mathbf{R_s}) \right) \tag{4.8}$$

4.2 Aerodynamic Disturbance

Aerodynamic drag is caused by atmospheric molecules colliding with the satellite's surface. For spacecraft orbiting below approximately 400 km, the aerodynamic torque disturbance is the dominant one [4].

The force, $d\mathbf{F}_{\text{drag}}$, on a surface element dA, with outward normal $\hat{\mathbf{N}}$, is given by

$$d\mathbf{F}_{\text{drag}} = -\frac{1}{2} C_D \rho \left\| \mathbf{V}_{sat} \right\|^2 \left(\hat{\mathbf{N}} \cdot \hat{\mathbf{V}}_{sat} \right) \hat{\mathbf{V}}_{sat} dA \tag{4.9}$$

The drag coefficient C_D is usually lay between 1 and 2. A value of 2 is a good estimate for C_D, in cases where the aerodynamic coefficient is undetermined.

It is important to consider that the atmospheric density is not constant with altitude. Vallado [9] presented an exponential model for density which is valid for the altitudes varying from 0 to 1000 Km.

$$\rho = \rho_o \, \text{EXP} \left[-\frac{h_{ellp} - h_o}{H} \right] \tag{4.10}$$

Where the data concerning this equation is given in [9].
Equation 4.9 becomes

$$\mathbf{F}_{\text{drag}} = -\frac{1}{2} C_D \rho \left\| \mathbf{V}_{sat} \right\|^2 \left(\hat{\mathbf{N}} \cdot \hat{\mathbf{V}}_{sat} \right) \hat{\mathbf{V}}_{sat} A \tag{4.11}$$

and the total aerodynamic disturbance is found to be

$$\mathbf{M}_{\text{drag}} = \sum \mathbf{R} \times \mathbf{F}_{\text{drag}} \tag{4.12}$$

4.3 Magnetic Field Residual Disturbance

residual magnetic torque $\mathbf{M}_{\text{mrs}}$ is caused by the interaction of the satellite's magnetic moment and the geomagnetic field, which is given as [13]

$$\mathbf{M}_{\text{mrs}} = \mathbf{m} \times \mathbf{B} \tag{4.13}$$

where:
$\mathbf{m}$ is the effective magnetic dipole moment of the satellite.
$\mathbf{B}$ is the Earth's magnetic field.
For the case here $\mathbf{m} = \begin{bmatrix} 0.001 & 0.001 & 0.001 \end{bmatrix}$ Am^2

CHAPTER 5

EQUATIONS OF MOTION

5.1 Kinematic Equations of Motion

The kinematic equations of motion and the dynamic equations of motion are the two types of equations of motion in attitude dynamics. Kinematics is the study of motion in general, regardless of the forces that derive it. The kinematic equations of motion are a set of first-order differential equations that describe how the attitude parameters change with time. The dynamic equations of motion will be developed, which express the time dependence of $\boldsymbol{\omega}$.

5.1.1 Time Derivative of Quaternions

Let $q(t)$ be a unit quaternion function of time, and $\boldsymbol{\omega}(t)$ the angular velocity determined by $q(t)$. The derivative of $q(t)$ is

$$\dot{\mathbf{q}} = \frac{1}{2} q\boldsymbol{\omega} \tag{5.1}$$

At $t + \Delta t$, the rotation is described by $q(t + \Delta t)$. This is after some extra rotation during Δt achieved on the frame that has already experienced a rotation described by $q(t)$. This extra rotation is about the instantaneous axis $\widehat{\boldsymbol{\omega}} = \frac{\boldsymbol{\omega}}{\|\boldsymbol{\omega}\|}$ over the angle $\Delta\theta = \|\boldsymbol{\omega}\|\Delta t$. With the aid off equation 1.30, it can be expressed as

$$\Delta\mathbf{q} = \cos\frac{\theta}{2} + \widehat{\boldsymbol{\omega}} \sin\frac{\theta}{2} \tag{5.2}$$

$$\Delta\mathbf{q} = \cos\frac{\|\boldsymbol{\omega}\|\Delta t}{2} + \frac{\boldsymbol{\omega}}{\|\boldsymbol{\omega}\|} \sin\frac{\|\boldsymbol{\omega}\|\Delta t}{2} \tag{5.3}$$

Inferring,

$$\begin{aligned}\mathbf{q}(t+\Delta t) &= \Delta\mathbf{q}\mathbf{q}(t)\\ \mathbf{q}(t+\Delta t) - q(t) &= \left(\cos\frac{\|\boldsymbol{\omega}\|\Delta t}{2} + \frac{\boldsymbol{\omega}}{\|\boldsymbol{\omega}\|}\sin\frac{\|\boldsymbol{\omega}\|\Delta t}{2}\right)\mathbf{q} - \mathbf{q}\\ &= \left(-2\sin^2\frac{\|\boldsymbol{\omega}\|\Delta t}{4} + \frac{\boldsymbol{\omega}}{\|\boldsymbol{\omega}\|}\sin\frac{\|\boldsymbol{\omega}\|\Delta t}{2}\right)\mathbf{q}\end{aligned} \tag{5.4}$$

Divide by Δt

$$\frac{\mathbf{q}(t+\Delta t)-\mathbf{q}(t)}{\Delta t} = \frac{\left(-2\sin^2\frac{\|\boldsymbol{\omega}\|\Delta t}{4} + \frac{\boldsymbol{\omega}}{\|\boldsymbol{\omega}\|}\sin\frac{\|\boldsymbol{\omega}\|\Delta t}{2}\right)\mathbf{q}}{\Delta t} \tag{5.5}$$

$$\dot{\mathbf{q}} = \lim_{\Delta t\to 0}\frac{q(t+\Delta t)-\mathbf{q}(t)}{\Delta t} = \lim_{\Delta t\to 0}\frac{1}{\Delta t}\left(-2\sin^2\frac{\|\omega\|\Delta t}{4} + \frac{\omega}{\|\omega\|}\sin\frac{\|\omega\|\Delta t}{2}\right)\mathbf{q} \tag{5.6}$$

$$= \frac{\omega}{\|\boldsymbol{\omega}\|\Delta t\to 0}\lim_{\Delta t}\frac{1}{\Delta t}\left(\sin\frac{\|\omega\|\Delta t}{2}\right)q = \frac{\boldsymbol{\omega}}{\|\boldsymbol{\omega}\|}\left(\frac{\|\omega\|\Delta t}{2}\right)\mathbf{q} \tag{5.7}$$

$$\boxed{\dot{\mathbf{q}} = \frac{1}{2}\boldsymbol{\omega}\mathbf{q}}$$

Which matches equation 5.1. Proposing $\boldsymbol{\omega}$ as a pure quaternion $\boldsymbol{\omega} = \begin{bmatrix} 0 & \omega_1 & \omega_2 & \omega_3 \end{bmatrix}^T \in \mathbb{H}$ and from quaternion multiplication discussed earlier

$$\dot{\mathbf{q}} = \frac{1}{2}\begin{bmatrix} -\omega_1 q_1 - \omega_2 q_2 - \omega_3 q_3 \\ \omega_1 q_0 + \omega_2 q_3 - \omega_3 q_2 \\ \omega_2 q_0 + \omega_3 q_1 - \omega_1 q_3 \\ \omega_3 q_0 + \omega_1 q_2 - \omega_2 q_1 \end{bmatrix} \tag{5.8}$$

5.1.2 The Omega operator

Let the omega operator $\boldsymbol{\Omega}(\boldsymbol{\omega})$ be

$$\boldsymbol{\Omega}(\boldsymbol{\omega}) = \begin{bmatrix} 0 & -\boldsymbol{\omega}^T \\ \boldsymbol{\omega} & \lfloor\boldsymbol{\omega}\rfloor_\times \end{bmatrix} \tag{5.9}$$

Where the expression $\lfloor\boldsymbol{\omega}\rfloor_\times$expands a vector $\boldsymbol{\omega} \in \mathbb{R}^3 \subset \mathbb{H}$ into a skew-symmetric matrix as

$$\lfloor\boldsymbol{\omega}\rfloor_\times \triangleq \begin{bmatrix} 0 & -\omega_2 & \omega_1 \\ \omega_2 & 0 & -\omega_0 \\ -\omega_1 & \omega_0 & 0 \end{bmatrix} \tag{5.10}$$

So, the $\boldsymbol{\Omega}(\boldsymbol{\omega})$ will be

$$\boldsymbol{\Omega}(\boldsymbol{\omega}) = \begin{bmatrix} 0 & -\omega_1 & -\omega_2 & -\omega_3 \\ \omega_1 & 0 & \omega_2 & -\omega_1 \\ \omega_2 & -\omega_2 & 0 & \omega_0 \\ \omega_3 & \omega_1 & -\omega_0 & 0 \end{bmatrix} \tag{5.11}$$

An equivalent matrix expression for the derivative of the quaternion is

$$\dot{\mathbf{q}} = \frac{1}{2}\mathbf{\Omega}(\boldsymbol{\omega})\mathbf{q} = \frac{1}{2}\begin{bmatrix} 0 & -\omega_1 & -\omega_2 & -\omega_3 \\ \omega_1 & 0 & -\omega_3 & \omega_2 \\ \omega_2 & \omega_3 & 0 & -\omega_1 \\ \omega_3 & -\omega_2 & \omega_1 & 0 \end{bmatrix}\begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} -\omega_1 q_1 - \omega_2 q_2 - \omega_3 q_3 \\ \omega_1 q_0 - \omega_3 q_2 + \omega_2 q_3 \\ \omega_2 q_0 + \omega_3 q_1 - \omega_1 q_3 \\ \omega_3 q_0 - \omega_2 q_1 + \omega_1 q_2 \end{bmatrix} \tag{5.12}$$

This definition simplifies the quaternion multiplication as it only involves simple matrix algebra.

5.2 Angular Momentum and Moment of Inertia Tensor

Consider the case of a rigid body traveling in an inertial frame. The transnational motion of the body's center of mass combined with a rotational motion of the body along some axis via its center of mass can be used to characterize this motion.

The angular momentum $\boldsymbol{h}$ about the center of mass of a rigid body consists of m_n objects is

$$\boldsymbol{h} = \sum_{m_n} \boldsymbol{r_n} \times (\boldsymbol{\omega} \times \boldsymbol{r_n})\, m_n \tag{5.13}$$

If the vector part of $\boldsymbol{\omega} = \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}^T$ is the angular velocity vector describing the motion of the rigid body, the angular momentum vector is defined as, see [14].

$$\boldsymbol{h} = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix}\begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = [\mathbf{I}]\boldsymbol{\omega} \tag{5.14}$$

$[\mathbf{I}]$ is the inertia tensor.

5.3 Dynamics Equation of Motion

5.3.1 Euler's Moment Equation

The angular momentum equation of a rigid body about its center of mass is simply given as

$$\boldsymbol{M} = \dot{\boldsymbol{h}} \tag{5.15}$$

Where $\boldsymbol{h}$ is the angular momentum vector of a rigid body about its mass center, $\boldsymbol{M}$ is the external moment acting on the body about its mass center.

We will utilize the well-known operator equation operating on a given vector A in the following analysis.

$$\frac{d}{dt}\boldsymbol{A}_I = \frac{d}{dt}\boldsymbol{A}_B + \boldsymbol{\omega} \times \boldsymbol{A} \tag{5.16}$$

This simply says that the rate of change of the vector $\boldsymbol{A}$ in the stationary coordinate system I *"inertial" in our instance* equals the rate of change of the vector $\boldsymbol{A}$ in the rotating coordinate system *B-"body"* plus the vector product $\boldsymbol{\omega} \times \boldsymbol{A}$. This relation will be used along the discussion

The rotational equation of motion of a rigid body about its center of mass is then written as

$$\boldsymbol{M} = \boldsymbol{h}_n = \left\{\frac{d\boldsymbol{h}}{dt}\right\}_B + \boldsymbol{\omega}^{B/I} \times \boldsymbol{h} \tag{5.17}$$

For simplicity let $\boldsymbol{\omega}^{B/I} = \boldsymbol{\omega}$ and using $\boldsymbol{h} = [\mathbf{I}] \cdot \boldsymbol{\omega}$

$$\boldsymbol{M} = \boldsymbol{h}_n = \left\{\frac{d([\mathbf{I}]\boldsymbol{\omega})}{dt}\right\}_B + \boldsymbol{\omega} \times [\mathbf{I}] \cdot \boldsymbol{\omega} = \left\{\frac{d[\mathbf{I}]}{dt}\right\}_B \cdot \boldsymbol{\omega} + [\mathbf{I}] \cdot \left\{\frac{d\boldsymbol{\omega}}{dt}\right\}_B + \boldsymbol{\omega} \times [\mathbf{I}] \cdot \boldsymbol{\omega} \tag{5.18}$$

Where $\left\{\frac{d[\mathrm{n}]}{dt}\right\}_B = 0$ and $\left\{\frac{d\boldsymbol{\omega}}{dt}\right\}_B = \left\{\frac{d\boldsymbol{\omega}}{dt}\right\}_I = \dot{\boldsymbol{\omega}}$

$$\boldsymbol{M} = [\mathbf{I}] \cdot \dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times [\mathbf{I}] \cdot \boldsymbol{\omega} \tag{5.19}$$

Which is called Euler's rotational equation of motion expressed in vector arrangement [12]. That can be manipulated into

$$[\mathbf{I}] \cdot \dot{\boldsymbol{\omega}} = \boldsymbol{M} - \boldsymbol{\omega} \times [\mathbf{I}] \cdot \boldsymbol{\omega} \tag{5.20}$$

$$[\mathbf{I}]^{-1}[\mathbf{I}] \cdot \dot{\boldsymbol{\omega}} = [\mathbf{I}]^{-1}(\boldsymbol{M} - \boldsymbol{\omega} \times [\mathbf{I}] \cdot \boldsymbol{\omega}) \tag{5.21}$$

$$\dot{\boldsymbol{\omega}} = [\mathbf{I}]^{-1}(\boldsymbol{M} - \boldsymbol{\omega} \times [\mathbf{I}] \cdot \boldsymbol{\omega}) \tag{5.22}$$

5.4 Controllability

The satellite controlled by a group of magnetorquers has a significant limitation. The mechanical torque resulting from the interaction of the geomagnetic field and the magnetic field generated by the magnetorquers is always perpendicular to the vector of the geomagnetic field. As a result, the direction parallel to the geomagnetic field vector cannot be controlled.

5.5 Numerical Scheme

The aim is to integrate nonlinear differential equations on the form

$$\dot{\mathbf{x}} = f(t, \mathbf{x}) \tag{5.23}$$

over a finite time interval Δt, to convert them to a difference equation,

$$\mathbf{x}(t + \Delta t) = \mathbf{x}(t) + \int_t^{t+\Delta t} f(\tau, \mathbf{x}(\tau))d\tau \tag{5.24}$$

or alternatively, if we assume that $t_n = n\Delta t$ and $\mathbf{x}_n \triangleq \mathbf{x}_n(t_n)$,

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \int_t^{t+\Delta t} f(\tau, \mathbf{x}(\tau))d\tau \tag{5.25}$$

The RK methods are a popular family of methods. These methods estimate the derivative over the interval using several iterations, and then use this derivative to integrate over the step Δt.
Several RK methods are presented in the sections that follow, ranging from the most basic to the most general.

5.5.1 RK1 – Euler's Method

The Euler method is based on the assumption that the derivative $f(x)$ is constant over the interval, and thus

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \Delta t \cdot f(t_n, \mathbf{x}_n) \tag{5.26}$$

Computing the derivative at the initial point can be generalized as follows

$$k_1 = f(t_n, \mathbf{x}_n) \tag{5.27}$$

Then, the integrated final value at the end point can be calculated from

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \Delta t \cdot k_1 \tag{5.28}$$

5.5.2 RK2 – Midpoint Heun's Method

The midpoint method assumes that the derivative is at the interval's midpoint and computes the value of $\mathbf{x}$ at this midpoint in one iteration.

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \Delta t \cdot f\left(t_n + \frac{1}{2}\Delta t, \mathbf{x}_n + \frac{1}{2}\Delta t \cdot f\left(t_n, \mathbf{x}_n\right)\right) \tag{5.29}$$

The midpoint method is explained in two steps as follows. First, using k_1 as previously defined, use the Euler method to integrate until the midpoint.

$$k_1 = f\left(t_n, \mathbf{x}_n\right) \tag{5.30}$$

$$\mathbf{x}\left(t_n + \frac{1}{2}\Delta t\right) = \mathbf{x}_n + \frac{1}{2}\Delta t \cdot k_1 \tag{5.31}$$

Then, this value can be used to calculate the midpoint derivative value k_2 which leads to the integrated value.

$$k_2 = f\left(t_n + \frac{1}{2}\Delta t, \mathbf{x}\left(t_n + \frac{1}{2}\Delta t\right)\right) \tag{5.32}$$

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \Delta t \cdot k_2 \tag{5.33}$$

5.5.3 RK4 – Method

This is commonly referred to as the Runge-Kutta method. It assumes $f(.)$ evaluation values at the beginning, middle, and end of the interval. And it computes the integral in four stages or iterations, with four derivatives, $k_1, .., k_4$, obtained sequentially. These derivatives, or slopes, are then weighted to obtain a 4^{th}-order estimate of the derivative in the interval. The RK-4 method is better specified as a small algorithm rather than a one-step formula, as are the previous two methods. The RK-4 integration step is as follows

$$\mathbf{x}_{\mathrm{n+1}} = \mathbf{x}_{\mathrm{n}} + \frac{\Delta t}{6} \cdot (k_1 + 2k_2 + 2k_3 + k_4) \tag{5.34}$$

The slopes k_1, k_2, k_3, k_4 are calculated as follows

$$k_1 = f\left(t_n, \mathbf{x}_\mathrm{n}\right) \tag{5.35}$$

$$k_2 = f\left(t_n + \frac{1}{2}\Delta t, \mathbf{x}_\mathrm{n} + \frac{\Delta t}{2}k_1\right) \tag{5.36}$$

$$k_3 = f\left(t_n + \frac{1}{2}\Delta t, \mathbf{x}_\mathrm{n} + \frac{\Delta t}{2}k_3\right) \tag{5.37}$$

$$k_2 = f\left(t_n + \Delta t, \mathbf{x}_\mathrm{n} + \Delta t.k_3\right) \tag{5.38}$$

The various slopes are interpreted as follows:

1. k_1 is the slope at the beginning of the interval (Euler's method).
2. k_2 is the slope at the midpoint of the interval (midpoint method).
3. k_3 is the slope at the midpoint again.
4. k_4 is the slope at the end of the interval.

The algorithms are all implemented in MATLAB can be found in Appendix **??**

Testing RK1 to RK4

The algorithm is tested on the van der Pol equation is a second-order ODE

$$\ddot{y} + \mu\left(1 - y^2\right)\dot{y} + y = 0 \tag{5.39}$$

The output is then compared with MATLAB ode45 routine for solving non-stiff ODE, the results obtained in figure 5.3.

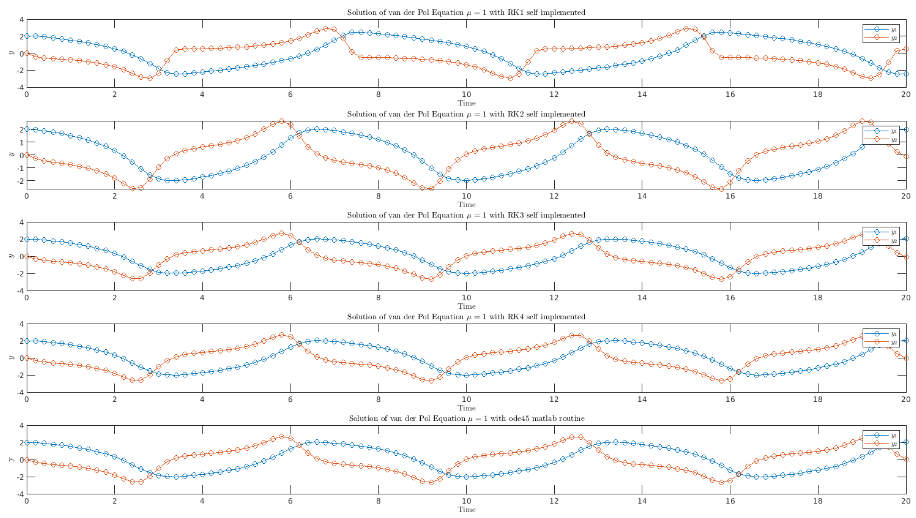


Figure 5.1: Comparison between RK1:4 and Ode45

The error obtained between two regimes are provided in figure 5.2.

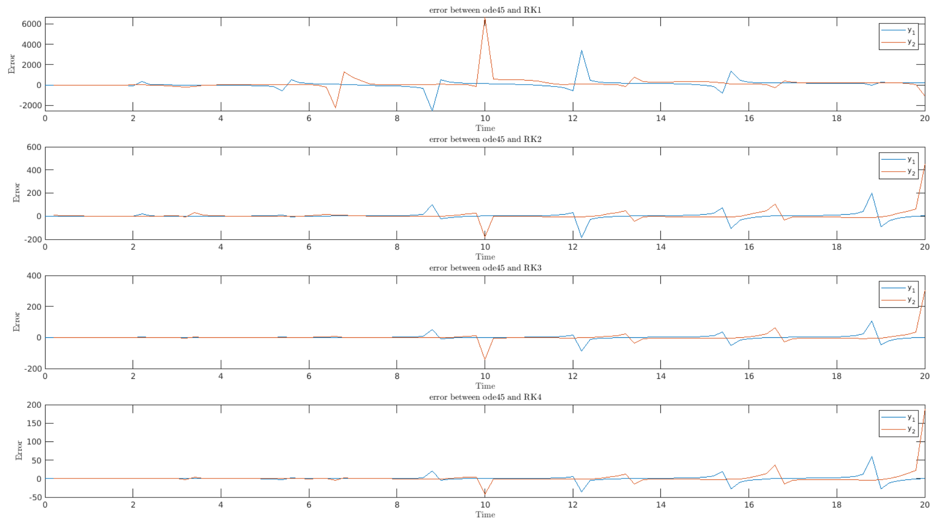


Figure 5.2: Error between RK1:4 and Ode45

It is obvious how RK4 is the best to be used.

5.5.4 General Runge-Kutta method

More elaborated RK methods are possible. They aim at either reduce the error and/or increase stability. They take the general form

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \Delta t \sum_{i=1}^{s} b_n k_n \tag{5.40}$$

where

$$k_n = f\left(t_n + \Delta t \cdot c_n, \mathbf{x}_n + \Delta t \sum_{j=1}^{s} a_{ij} k_j\right) \tag{5.41}$$

5.5.5 Adaptive Step Size Runge-Kutta Method

There are methods for automatically adjusting the step size. They entail combining two adjacent-order RK methods into one and estimating the truncation error in the lower order solution using the difference between the higher and lower order solutions. To keep the truncation error within bounds, the step size h is adjusted [10].

The integrating of RK4 into RK5 to generate the RKF4(5) method is a common example. The F is added to recognise E. Fehlberg's contribution to this RK method extension. The procedure is divided into six phases, and the Fehlberg coefficients are calculated at each stage [15].

$$\mathbf{a} = \begin{bmatrix} 0 \\ 1/4 \\ 3/8 \\ 12/13 \\ 1 \\ 1/2 \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1/4 & 0 & 0 & 0 & 0 \\ 3/32 & 9/32 & 0 & 0 & 0 \\ 1932/2197 & -7200/2197 & 7296/2197 & 0 & 0 \\ 439/216 & -8 & 3680/513 & -845/4104 & 0 \\ -8/27 & 2 & -3544/2565 & 1859/4104 & -11/40 \end{bmatrix}$$

(5.42)

$$\mathbf{c}^* = \begin{bmatrix} 25/216 \\ 0 \\ 1408/2565 \\ 2197/4104 \\ -1/5 \\ 0 \end{bmatrix} \quad \mathbf{c} = \begin{bmatrix} c16/135 \\ 0 \\ 6656/12825 \\ 28561/56430 \\ -9/50 \\ 2/55 \end{bmatrix} \tag{5.43}$$

Using asterisks to indicate that RK4 is the lower order of the two

$$\mathbf{x}_{n+1}^* = \mathbf{x}_n + \Delta t \left(c_1^* k_1 + c_2^* k_2 + c_3^* k_3 + c_4^* k_4 + c_5^* k_5 + c_6^* k_6\right) \quad \text{Low order solution } (RK4) \tag{5.44}$$

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \Delta t \left(c_1 k_1 + c_2 k_2 + c_3 k_3 + c_4 k_4 + c_5 k_5 + c_6 k_6\right) \quad \text{High order solution } (RK5) \tag{5.45}$$

Using equations 5.40 and 5.41, the derivatives of the six stages are

$$\begin{aligned} k_1 &= f\left(t_i, \mathbf{x}_i\right) \\ k_2 &= f\left(t_i + a_2 \Delta t, \mathbf{x}_i + \Delta t b_{21} k_1\right) \\ k_3 &= f\left(t_i + a_3 \Delta t, \mathbf{x}_i + \Delta t \left[b_{31} k_1 + b_{32} k_2\right]\right) \\ k_4 &= f\left(t_i + a_4 \Delta t, \mathbf{x}_i + \Delta t \left[b_{41} k_1 + b_{42} k_2 + b_{43} k_3\right]\right) \\ k_5 &= f\left(t_i + a_5 \Delta t, \mathbf{x}_i + \Delta t \left[b_{51} k_1 + b_{52} k_2 + b_{53} k_3 + b_{54} k_4\right]\right) \\ k_6 &= f\left(t_i + a_6 \Delta t, \mathbf{x}_i + \Delta t \left[b_{61} k_1 + b_{62} k_2 + b_{63} k_3 + b_{64} k_4 + b_{65} k_5\right]\right) \end{aligned} \tag{5.46}$$

The truncation vector **e** is the difference between the higher order RK5 $\mathbf{x_{n+1}}$ and the lower order RK4 $\mathbf{x^*_{n+1}}$

$$\begin{aligned} \mathbf{e} &= \mathbf{x}_{n+1} - \mathbf{x}_{n+1}^* \\ &= \Delta t \left[\left(c_1 - c_1^*\right) k_1 + \left(c_2 - c_2^*\right) k_2 + \left(c_3 - c_3^*\right) k_3 + \left(c_4 - c_4^*\right) k_4 + \left(c_5 - c_5^*\right) k_5 + \left(c_6 - c_6^*\right) k_6\right] \end{aligned} \tag{5.47}$$

The scalar truncation error **e** is the largest value of the absolute values of the components of **e**

$$e = \text{ maximum of the set } \left(\left|e_1\right|, \left|e_2\right|, \left|e_3\right|, \cdots, \left|e_N\right|\right) \tag{5.48}$$

We established a tolerance *tol* that the truncation error could not exceed. Rather than using the same Δt for each step of the numerical integration process, we can vary the step size to keep the error **e** from exceeding *tol.* A straightforward strategy for adaptive step size control is to update h after each time step using a formula derived in, for example, Bond and Allman in [16].

$$h_{\text{new}} = h_{\text{old}} \left(\frac{tol}{e} \right)^{\frac{1}{p+1}} \tag{5.49}$$

Where p is the lower order in RKF–p(p+1). A factor ε is added when $e < tol$, ε may be 0.8 or 0.9.

$$h_{\text{new}} = h_{\text{old}} \cdot \varepsilon \left(\frac{tol}{e} \right)^{\frac{1}{p+1}} \tag{5.50}$$

The algorithm implemented in MATLAB can be found in Appendix **??**.

The output of the same equation given in 5.39 is then compared with MATLAB ode45 routine for solving non-stiff ODE along alos with RK4 algorithm, the results obtained in figure 5.3.

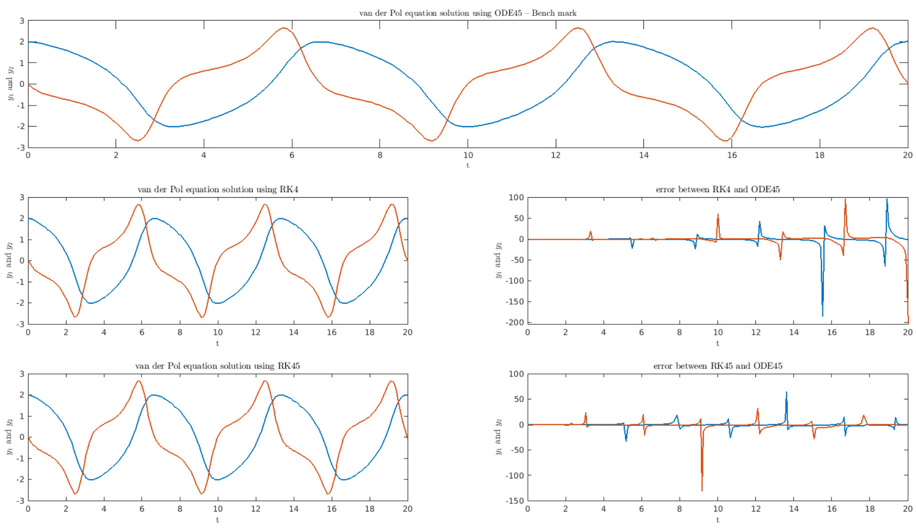


Figure 5.3: Comparison between RK4, RKF45 and Ode45

The edge of this variable step size numerical integration scheme is that it takes much less computational time compared to the regular RK family.

CHAPTER 6

STATE ESTIMATION

6.1 Attitude Estimation

In this part of our project, we tend to obtain accurate attitude estimates that will help us in computing a suitable control action. In our system we have Gyro and magnetometer sensors that provide us with measurements of angular rates, magnetic field, and acceleration components. To have accurate estimates to the actual internal states, we use Extended Kalman Filter (EKF) to combine the system model with the measurements. For the altitude to be estimated we need to define and implement a State observer. State observer is the mathematical model that represents the real system and gives the internal state estimates depending on input and output signals.

$$\hat{\mathbf{x}}_k = f(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{w}_k \tag{6.1}$$

$$\hat{\mathbf{z}}_k = h(\mathbf{x}_k) + \boldsymbol{\nu}_k \tag{6.2}$$

where:
$\hat{\mathbf{x}}_k$: is the internal state vector.
$\mathbf{u}_k$: is input signals at time k.
$f(\mathbf{x}_k, \mathbf{u}_k)$: is the function that relates the states at time $k+1$ to the previous state $\hat{\mathbf{x}}_k$ and input $\mathbf{u}_k$).
$\hat{\mathbf{z}}_k$: is the system output.
$h(\mathbf{x}_k)$: relates the internal states to the output.
$\mathbf{w}_k$, $\boldsymbol{\nu}_k$: are the noises which are both assumed to be zero mean multivariate Gaussian noises with covariance $\mathbf{P}_k$ and $\mathbf{Q}_k$ respectively. The block diagram for the problem is defined in 6.1.

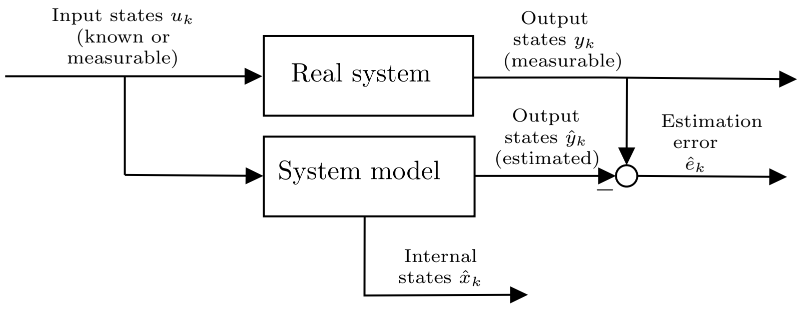

Input states u_k (known or measurable)
Real system
Output states y_k (measurable)
System model
Output states $\hat{y}_k$ (estimated)
Estimation error $\hat{e}_k$
−
Internal states $\hat{x}_k$

Figure 6.1: block diagram for the estimation problem formulation

The estimation error $\mathbf{e}_k$ is the difference between the real system output and the estimated output. A very common way to minimize the variance of this estimation error is the Kalman Filters. The standard Kalman filter is a recursive filter that works in two steps:

1. predicting the altitude based on the mathematical models.
2. updating this altitude based on the data we get from sensors.

For each new data from the sensor, is complained with the estimated state of the previous time step $\mathbf{x}_{k-1}$ to have a new update for the altitude $\mathbf{x}$ (state estimate) and the error covariance matrix (the accuracy of the estimate $\mathbf{P}$) this is called the time update step. The second step is the state correction. It mainly depends on the state vector covariance matrix $\mathbf{P}_k$.

$$\hat{\mathbf{x}}_0^+ = \mathbb{E}(\mathbf{x}_0) \tag{6.3}$$
$$\mathbf{P}_0^+ = \mathbb{E}\left\{(\mathbf{x}_0 - \hat{\mathbf{x}}_0^+)(\mathbf{x}_0 - \hat{\mathbf{x}}_0^+)^{\mathrm{T}}\right\} \tag{6.4}$$

KF works as a least square error optimizer, this means that the system we have must be a linear system [17].For a discrete-time nonlinear dynamic system -like ours- we use a modified version of the KF which is the EKF. EKF works by linearizing the system under investigation and this is around its current state, then we force this filter to use this linearized version of our system as a model. EKF is used as the first-order or second-order approximating estimator. However, If the system is highly nonlinear, the EKF may diverge thus we use the Unscented Kalman Filter. For our problem we will use the EKF as the UKF is more computationally-extensive [18]. Another advantage of the EKF is the simple derivations of Jacobian matrices. However, the linearization that is only a first order approximation of the non-linear equations is considered to be a disadvantage of the EKF.

6.2 Mathematical Model

For the real system model provided in figure 6.1, the mathematical formulation for such block is provided in chapter 5.

6.2.1 Rate Gyro and Magnetometer Models

It is assumed that measurements related to the state are made or that the state is directly measured during the standard estimation procedure. Bias and white noise are common sources of contamination in measurements. The angular velocity measurement, for example, can be expressed as

$$\boldsymbol{\omega}_{\text{measured}} = \boldsymbol{\omega} + \mathbf{b} + \boldsymbol{\eta}_w \tag{6.5}$$

The same case for the magnetometer model, it can be expressed as

$$\mathbf{B}_{\text{measured}} = \mathbf{B} + \mathbf{b} + \boldsymbol{\eta}_b \tag{6.6}$$

where $\boldsymbol{\eta}_b$ and $\boldsymbol{\eta}_w$ are standard Gaussian white noise vectors with zero mean.

6.3 Kalman Filter Design

To design EKF, we need to discretize the process model to be used on the microcontroller for the future purposes. To implement this we will use a zero-order hold approach

$$f(t) = f(k_t) \quad t_k \leq t \leq t_{k+1} \tag{6.7}$$

6.3.1 Linearized process model

Linearization is a very important step in EKF. We need to linearize the nonlinear state equations by taking the error between actual states and estimated states instead of the real states where

$$\mathbf{F}_k = \frac{\partial \mathbf{f}_k}{\partial \mathbf{x}}|_{\mathbf{x}=\hat{\mathbf{x}}} \tag{6.8}$$

$$\mathbf{H}_k = \frac{\partial \mathbf{h}_k}{\partial \mathbf{x}}|_{\mathbf{x}=\hat{\mathbf{x}}} \tag{6.9}$$

6.3.2 Initialization

This initialization stage can be done using the expected values of the state and covariance.

$$\hat{\mathbf{x}} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}^T$$

$$\mathbf{P}_o^- = \begin{bmatrix} (100 \times 10^{-8})^2 * \mathbf{I}(3,3) & \text{zeros}(3,3) \\ \text{zeros}(3,3) & 1 \times 10^{-10} * \mathbf{I}(3,3) \end{bmatrix}_{6\times 6}$$

Initialize the posterior estimate and $\mathbf{P}$ as

$$\hat{\mathbf{x}}_{0k}^+ = \hat{\mathbf{x}}_k^-$$
$$\mathbf{P}_{0k}^+ = \mathbf{P}_k^-$$

6.3.3 Correction

This step includes calculating Kalman gains K and corrects the state vector and the covariance matrix.

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}_k^T \left(\mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{R}_k\right)^{-1}$$
$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k \left(\mathbf{z}_k - \hat{\mathbf{z}}_k\right)$$
$$\mathbf{P}_k^+ = \left(\mathbf{I} - \mathbf{K}_k \mathbf{H}_k\right) \mathbf{P}_k^- \left(\mathbf{I} - \mathbf{K}_k \mathbf{H}_k\right)^T + \mathbf{K}_k \mathbf{R}_k \mathbf{K}_k^T$$

6.3.4 Prediction

$$\hat{\mathbf{x}}_{k+1}^{-} = f\left(\hat{\mathbf{x}}_{k}^{-}, \mathbf{u}_k\right) \tag{6.10}$$

$$\mathbf{P}_{k+1}^{-} = \mathbf{F}_k \mathbf{P}_k^{+} \mathbf{F}_k^{T} + \mathbf{Q}_k \tag{6.11}$$

CHAPTER 7

CONTROL ALGORITHM

7.1 Literature Review

Magnetic coils have been employed as a simple and efficient torque production method for attitude and momentum control since the dawn of the space age [19]. Their performance is based on the interaction of the magnetic field generated by the coils and the earth's magnetic field, and so provides a straightforward solution to the problem of creating torques on board a satellite [19, 4]. As a result, the torques that can be applied to the spacecraft for attitude control are confined to lie in the plane perpendicular to the magnetic field vector. In instance, 3 axis magnetic stabilization is only feasible if the orbit under consideration sees a variation in the magnetic field sufficient to ensure the spacecraft's stability [20]. Sidi [14] proposed a solution in which a magnetic coil is replaced with a reaction wheel in any of the body frame's axes. Therefore the spacecraft becomes fully-actuated.

High angular velocity can be created during launch vehicle deployment or occur spontaneously in orbit owing to disturbance torques. Consequently, one important mission after the satellite gets launched into space is to detumble its angular velocity [21]. When the angular rate of a satellite in low Earth orbit (LEO) drops to 0.13 degree/s or below, it is considered detumbled [22]. Detumbling is a necessary procedure since it comes before any other operation on the satellite that requires some degree of attitude control. There are a variety of detumbling controllers available, and several of them have been tested in [23] for a CubeSat. In terms of maneuver time, precision, control effort, robustness, and implementation. Controllers based on magnetometer feedback, gyroscope feedback, sun sensor feedback, and passive control are compared. The magnetometer feedback B-dot controller was determined to be the best and easiest to implement controller for a CubeSat.

There are other modes for the satellite to keep it pointing or aligned to specific axes as discussed in [24]. Some methods and approaches to the problem of Earth-pointing attitude control for a spacecraft using magnetic actuators are provided, which ensures virtually global closed loop stability of the desired relative attitude equilibrium for the spacecraft. As proposed by [19, 25] PD controller provides attitude regulation in the case of three controllers in the three axes. The nonlinear H-infinity state feedback attitude control method is designed for large spacecraft maneuvers and more convenient with control systems based on thrusters as discussed in [26].

Additionally, robust controllers have been developed to overcome inaccuracies in modeling and noise effects [24]. Moreover, [27] devised yet another robust control technique based on a nonlinear H-infinity control mechanism. This approach entails solving Hamilton-Jacobi-Isaacs inequalities, which effectively determine feedback gains for the whole state feedback control problem, allowing the spacecraft to be stabilized in the presence of uncertainties and disturbances. Variable structure control is robust to generic model parameter errors, but it comes at the expense of potentially high control activity. Adaptive controllers update the model during operation based on measured performances [24]. An adaptive scheme which estimates external torques by tracking a Lyapunov function has been developed by [28]. This method has been shown to be very robust in the presence of spacecraft modeling errors and disturbances.

Controlling a satellite without full state feedback is a more difficult problem. Methods for estimating unavailable states using a filter algorithm and methods for developing control laws directly from output feedback are the two basic approaches used to solve this problem [24].

7.2 Detumbling

7.2.1 B-dot Controller

The torque produced by magnetic torquers is given by

$$\mathbf{L} = \mathbf{m} \times \mathbf{B} \tag{7.1}$$

where $\mathbf{m}$ is the torquer-generated commanded magnetic dipole moment and $\mathbf{B}$ is the local geomagnetic field in body-frame coordinates. The magnetometers' magnetic dipole moment is controlled to detumble the satellite from its arbitrary initial spinning by

$$\mathbf{m} = -\frac{k}{\|\mathbf{B}\|}\dot{\mathbf{B}} \tag{7.2}$$

where k is a scalar gain [24].

7.2.2 Simulation Setup

A MATLAB simulation framework with the configuration of the block diagram in figure 7.1 is created. An orbital propagator, an attitude propagator, a function to calculate attitude disturbance torques, and a magnetometer function are all part of this framework. It should be simple to change orbital, satellite, and hardware parameters. Each of the functions shown in figure 7.1 will be supplied along with this thesis.

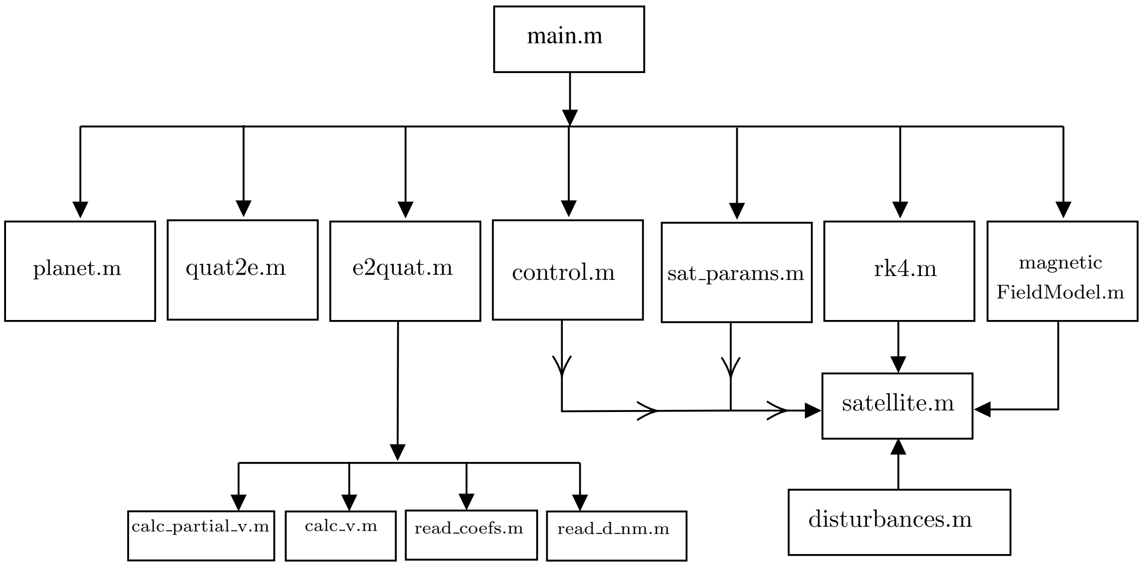


Figure 7.1: A block diagram of the simulation framework together with the detumbling controller.

The simulation in this section is carried out for the given parameters in table 7.1. The satellite motion is simulated for orbits and the results obtained can be found in section 7.2.3.

Parameter	Value
Semimajor Axis (Km)	9122
Eccentricty	0.1
Inclination (deg)	120
Altitude (Km)	450
I_x $(Kg.m^2)$	0.1
I_y $(Kg.m^2)$	0.2
Iz $(Kg.m^2)$	0.3
ω_{x0} (deg)	5
ω_{y0} (deg)	5
ω_{z0} (deg)	5

Table 7.1: Cube-sat simulation parameters

7.2.3 Simulation Results

The b-dot control law in equation 7.1 was simulated for a small satellite in low earth orbit for 3 orbits, the satellite and the orbit parameters are given in table 7.1. The initial angular rate is set to $\omega_0 = \begin{bmatrix} 5 & 5 & 5 \end{bmatrix}$ deg/sec. The controller gain is set to $k = 90 \times 10^3$. Figure 7.2 shows the response of $\boldsymbol{\omega}$ to the control torque, figure 7.3 shows a zoom into the angular rate which entails that the controller managed to detumble the satellite to around 0.15 deg/sec in 6000 sec.

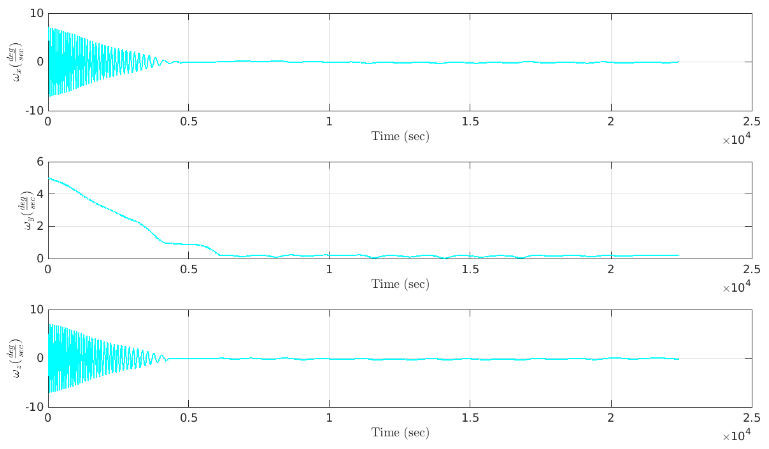


Figure 7.2: The controlled angular rate ω of the satellite for the case of detumbling

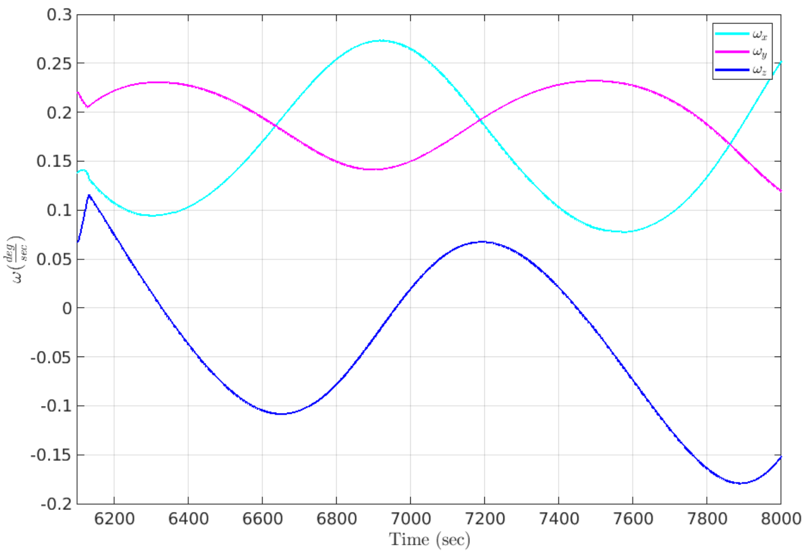


Figure 7.3: Zoom into controlled angular rate ω of the satellite

Figure 7.4 shows the behaviour of the satellite described by Euler angles.

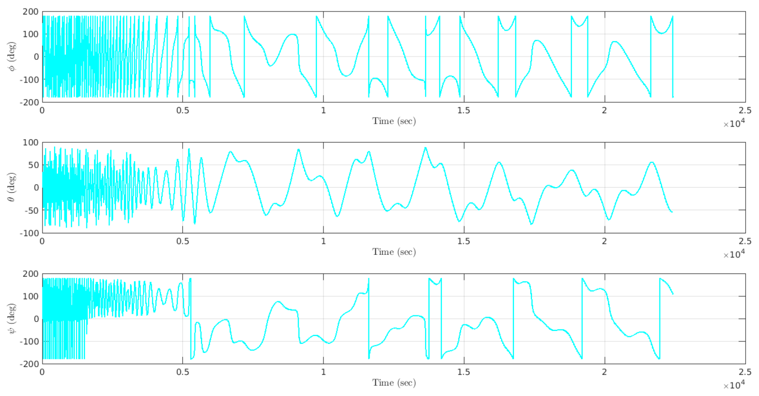


Figure 7.4: The controlled Euler angels of the satellite for the case of detumbling

BIBLIOGRAPHY

[1] W. R. Hamilton, "On quaternions; or on a new system of imaginaries in algebra," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 33, no. 219, pp. 58–60, 1848.

[2] J. B. Kuipers, *Quaternions and rotation sequences: a primer with applications to orbits, aerospace, and virtual reality.* Princeton university press, 1999.

[3] J. Sola, "Quaternion kinematics for the error-state kalman filter," *arXiv preprint arXiv:1711.02508*, 2017.

[4] J. R. Wertz, *Spacecraft attitude determination and control.* Springer Science & Business Media, 2012, vol. 73.

[5] J. J. Love, "Magnetic monitoring of earth and space," *Physics Today*, vol. 61, no. 2, p. 31, 2008.

[6] P. Alken, E. Thébault, C. D. Beggan, H. Amit, J. Aubert, J. Baerenzung, T. Bondar, W. Brown, S. Califf, A. Chambodut *et al.*, "International geomagnetic reference field: the thirteenth generation," *Earth, Planets and Space*, vol. 73, no. 1, pp. 1–25, 2021.

[7] S. Bhatt, A. Svecz, A. Alaniz, J.-W. Jang, L. Nguyen, and P. Spanos, "Thermally-constrained fuel-optimal iss maneuvers," in *Guidance and Control Conference*, no. JSC-CN-32749, 2015.

[8] D. L. Boulet, "Methods of orbit determination for the microcomputer," *Richmond*, 1991.

[9] D. A. Vallado, *Fundamentals of astrodynamics and applications.* Springer Science & Business Media, 2001, vol. 12.

[10] H. Curtis, *Orbital mechanics for engineering students.* Butterworth-Heinemann, 2013.

[11] P. K. Seidelmann, *Explanatory supplement to the astronomical almanac.* University Science Books, 2006.

[12] B. Wie, *Space vehicle dynamics and control.* Aiaa, 1998.

[13] P. C. Hughes, *Spacecraft attitude dynamics.* Courier Corporation, 2012.

[14] M. J. Sidi, *Spacecraft dynamics and control: a practical engineering approach.* Cambridge university press, 1997, vol. 7.

[15] E. Fehlberg, *Low-order classical Runge-Kutta formulas with stepsize control and their application to some heat transfer problems.* National aeronautics and space administration, 1969, vol. 315.

[16] V. R. Bond and M. C. Allman, *Modern Astrodynamics: Fundamentals and perturbation methods.* Princeton University Press, 1996, vol. 51.

[17] M. Z. Hasan, A. Ahmed, A. H. Abdullah, S. Yaacob, S. B. Yaakob, M. H. Idris, and M. A. M. Said, "Review on attitude estmation algorithm of attitude determination system," 2006.

[18] K. Fujii, "Extended kalman filter," *Refernce Manual*, pp. 14–22, 2013.

[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] M. Lovera, "Periodic attitude control for satellites with magnetic actuators: an overview," *IFAC Proceedings Volumes*, vol. 34, no. 12, pp. 113–118, 2001.

[21] J. Yadav and T. Goyal, "Investigation of instabilities in detumbling algorithms," in *2020 IEEE Aerospace Conference.* IEEE, 2020, pp. 1–8.

[22] B. Ø. Andresen, C. Grøn, R. H. Knudsen, C. Nielsen, K. K. Sørensen, and D. Taagaard, "Attitude control system for aausat-ii," *Institute of Electronic Systems, Aalborg University*, 2005.

[23] A. Pignède, "Detumbling of the ntnu test satellite," *Project thesis, Norwegian University of Science and Technology, Department of Engineering Cybernetics*, 2014.

[24] F. L. Markley and J. L. Crassidis, "Fundamentals of spacecraft attitude determination and control," 2014.

[25] M. Lovera and A. Astolfi, "Global magnetic attitude control of spacecraft in the presence of gravity gradient," *IEEE transactions on aerospace and electronic systems*, vol. 42, no. 3, pp. 796–805, 2006.

[26] L. Jonsson, "Simulations of satellite attitude maneuvers: Detumbling and pointing," 2019.

[27] J. L. Crassidis, S. R. Vadali, and F. L. Markley, "Optimal variable-structure control tracking of spacecraft maneuvers," *Journal of Guidance, Control, and Dynamics*, vol. 23, no. 3, pp. 564–566, 2000.

[28] H. Schaub, M. R. Akella, and J. L. Junkins, "Adaptive control of nonlinear attitude motions realizing linear closed loop dynamics," *Journal of Guidance, Control, and Dynamics*, vol. 24, no. 1, pp. 95–100, 2001.