Cairo University
Faculty of Engineering
Aerospace Engineering Department

Design and Implementation of satellite

ATTITUDE DETERMINATION AND CONTROL SUBSYSTEM

BY
Ahmed Adel Alreweny
Amr Hossam Eldin Ali
Amr Wahied Ibrahem
Mohamed Adel Anis
Moustafa Sayed Abdel Hamid

Under the Supervision of

Prof. Gamal El-Bayoumi

2017/7/18

“God grant me the serenity to accept the things I cannot change, courage to change the things I can, and wisdom to know the difference”

Acknowledgement

First and foremost, praises to ALLAH, the greatest of all, on whom ultimately we depend for help and guidance.

We would like to express our deep gratitude and appreciation to our project mentors, it has been great privilege to work under the supervision of ***Prof. Gamal Elbayoumi*** for his constructive suggestions and guidance through all this project. Also our appreciation and gratitude to all the teaching staff of the Aerospace engineering department of Cairo University.

We also place on record, our sincere thanks to ***Eng. Mohamed Shouman***, NARSS's supervisor, for the continuous assistance.

Abstract

The project covers the development of a cube-sat attitude determination and control subsystem (ADCS) embedded software and verification of its functionality through different stages including Model-In-The-Loop (MIL), then Software-In-The-Loop (SIL), and finally Hard-Ware-In-The Loop (HIL). Also this project develops a flight simulator with advanced visualizations and interactive user interface built in MATLAB. The role of ADCS is to provide attitude control and stability functions, including the de-tumbling and stabilizing the satellite angular velocity, As well as pointing the satellite payload (camera) in nadir direction during imaging with high accuracy.

ADCS software controls the magnetic torquers which is the only actuation device, with magnetometers and gyro as sensors through different modes of operation including de-tumbling, attitude accusation, imaging and stand-by by means of B-Dot and PD-like controllers. Also, ADCS provides the attitude determination functions for the estimation of the satellite orientation and angular velocities during different modes by utilizing Kalman Filter Algorithm.

While, the Flight Simulator package provides a use friendly GUI for feeding the simulator with different parameters and high visualization capabilities. Then, simulate the orbital and attitude motion of the satellite. Also display all kind of results numerically and through graphs and animations.

Table of Contents

List of Figures

Chapter 1

Introduction

A cube-sat is a type of miniaturized satellite that is made up of multiples of 10×10×10 cm cubic units. Cube-sat have a mass of no more than 1.33 kilograms per unit. Also, cube-sat are most commonly put low Earth orbit (LEO), and many are used to demonstrate spacecraft technologies that are targeted for use in small satellites or that present questionable feasibility and are unlikely to justify the cost of a larger satellite.

Attitude control for cube satellites relies on miniaturizing technology without significant performance degradation. Also, it presents a challenging problem for designers due to their small size and hard limit on power availability. The goal of the attitude determination and control subsystem (ADCS) is to stabilize the spacecraft against all attitude disturbing influences resulting from the environment in the earth orbit in order to point the payload towards a predetermined point on the earth's surface within a specified margin of error.

This project is part of the Egyptian universities cube-sat (Egy_Univer_Sat-1 or EUS-1) main project supervised by the National Authority for Remote Sensing & Space Sciences in Egypt (NARSS). EUS-1 is a 2-uint cube-sat, 1st

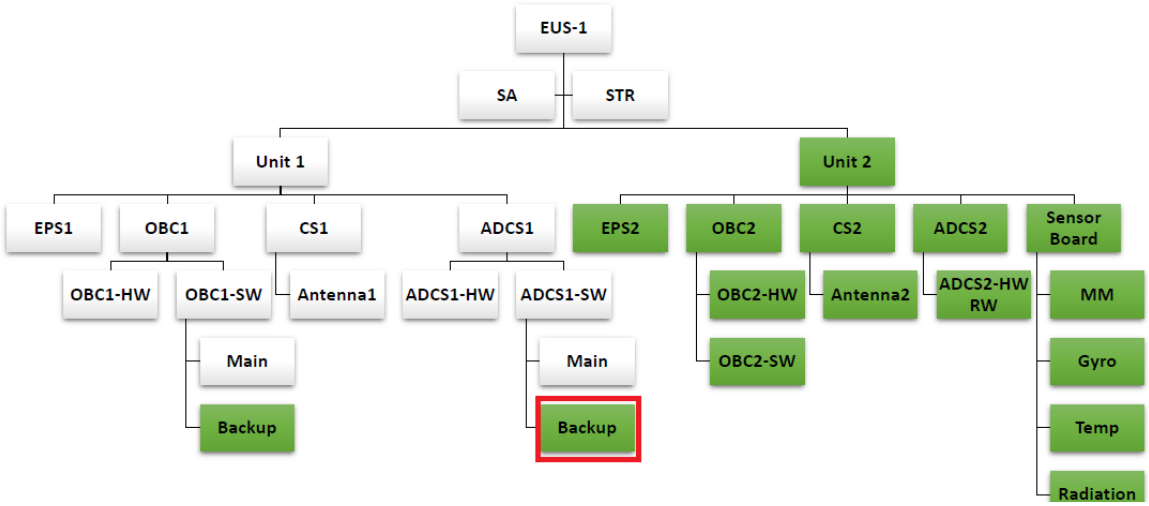


Figure 1| EUS-1Satellite Structure

unit (master unit) will contain space qualified HW and SW to minimize failure probability and Egyptian students will develop a backup SW, however the 2nd unit (slave unit) will contain HW/SW of satellite subsystem developed by students. The EUS-1 satellite structure is shown in Figure below, where all blocks in green are developed and manufactured locally.

The Egyptian students will design and develop the following projects:

- Backup SW of 1st unit:

1. Satellite Onboard Computer OBC1 SW
2. Satellite Attitude control ADCS1 SW
3. Satellite flight control center, FCC SW

- HW/SW of 2nd Unit:

1. Satellite communication subsystem CS2
2. Satellite electrical power subsystem EPS2
3. Satellite sensor board
4. Satellite attitude actuator (RW) ADCS2 HW
5. Satellite onboard computer subsystem OBC2
6. Satellite antenna

Figure 2| 2U CubeSat Satellite

The scope of this project is to design, implement and test the ADCS software for EUS-1 which will be implemented as a backup software (marked with red in fig 1). This software should include attitude control and estimation algorithms for different modes of the satellite operation.

Also to develop a Flight Simulator package provides a use friendly GUI for feeding the simulator with different parameters and high visualization capabilities. Then, simulate the orbital and attitude motion of the satellite. Also display all kind of results numerically and through graphs and animations.
After separation from launcher the satellite goes through different stages in attitude control: First, it spins with high angular rates due to asymmetric forces experienced at ejection form the launcher. So Angular velocity suppression (de-tumbling) mode is activated until the angular velocities reaches 2 orbital angular velocities (0.002 rad/sec), then the satellite enters reorientation section until it reaches the desired attitude, Then it goes to stand-by mode until Imaging order is received, Then Imaging mode is activated which requires the highest accuracy. Fig (3) describes the ADCS mission from launching the satellite to imaging operation. The ADCS software should fulfil the requirements of each segment including accuracy and speed.

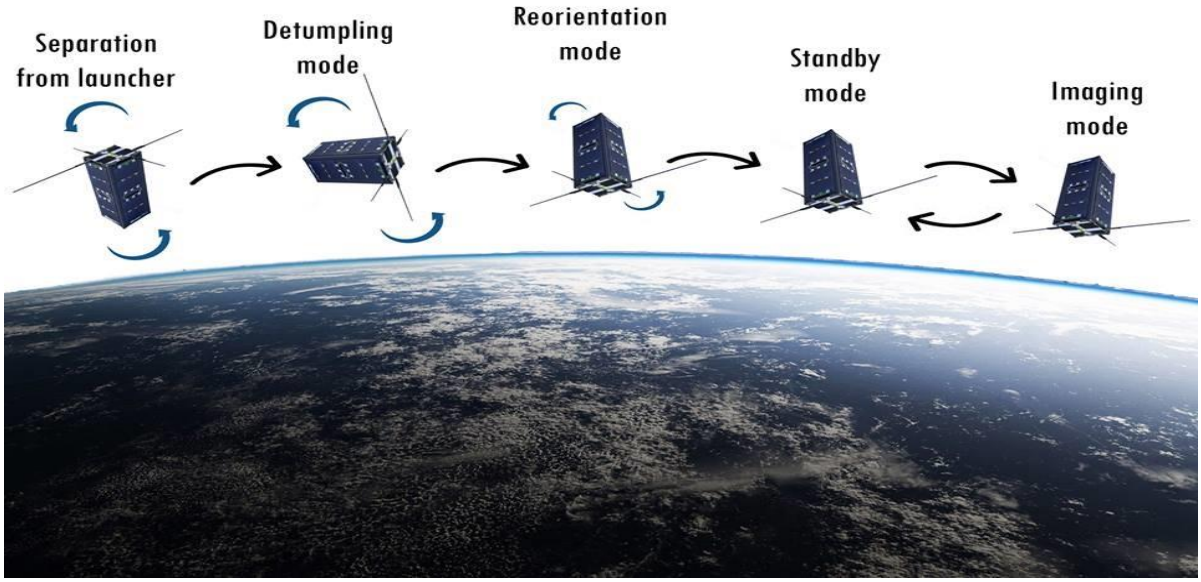


Figure 3| Satellite ADCS Mission

Satellite ADCS Devices

The devices used in the ADCS are integrated in the first unit of the satellite which is bought from GOMSpace (Cube-satellite solutions vendor). And they as the following

1. 3-Axis magnetometer
   The 3-Axis magnetometer to sense Earth's magnetic field in device coordinates
2. Gyroscope
   The gyroscope measures the angular rates in 3-axis
3. Magnetorqures
   To produce actuating moment and has a saturation dipole moment of 0.076 A.m

ADCS software requirements

The goal of the attitude determination and control subsystem (ADCS) is to stabilize the spacecraft against all attitude disturbing influences resulting from the environment in the earth orbit in order to point the payload towards a predetermined point on the earth's surface within a specified margin of error.

These objectives can be translated to the following function of the ADCS:

1) Suppression of angular velocities received by satellite after its separation from launch vehicle with angular velocity $\pm 10\ °/sec$ and given orientation ten orbital periods.
2) During standby mode of operation, AOCS must maintain the satellite in nadir pointing with accuracy not more than 10 ° in the three axes
3) During imaging mode of operation, AOCS must maintain the satellite in nadir pointing with accuracy not more than 5 ° in the three axes
4) Stabilize spacecraft against external disturbances
   - Gravity gradient moment
   - Magnetic moment
   - Aerodynamic moment
   - Solar radiation moment

Constrains on devices operation

During de-tumbling only the magnetometer is operational to save power during this critical stage. And, during stand-by magnetometer only is operational.

That is for the ADCS software while for the Flight simulator it should include model for different space environment parameters including:

1. Earth gravitational field
2. Earth magnetic field
3. Sidereal time
4. Satellite eclipse estimation
5. Sun vector

Also is should include models for the following disturbances:

1. Magnetic disturbance
2. Aerodynamic disturbance
3. Solar disturbance
4. Gravity gradient disturbance

And it also should include sensor model to generate reading for the magnetometer and the gyro to be sent back to the ADCS software while testing.

And actuator model to generate the control torque that the satellite would experience under certain control action.

Also the Flight simulator should include friendly interactive graphical use interface, through which the user could input satellite and simulation parameters.

The following diagram represent the block diagrams and modules utilized in the flight simulator and the ADCS software module, Also it can be considered as a map for the project.

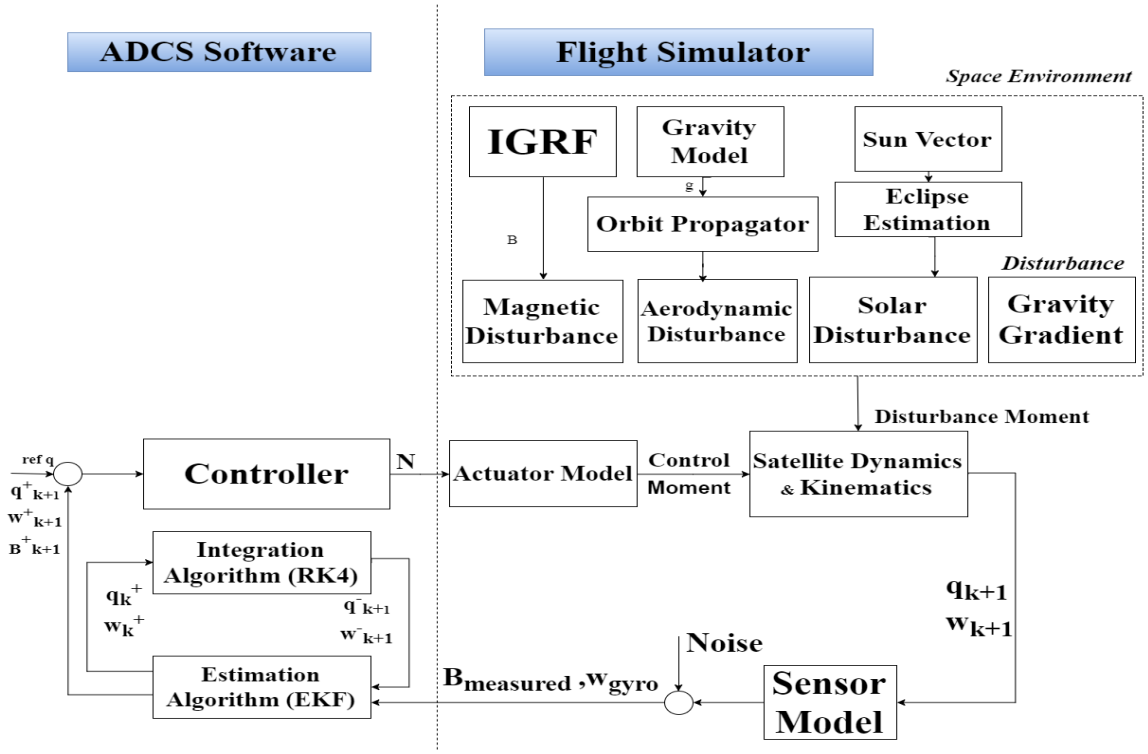


Figure 4| Project Block Diagram

Where it begins with space environment modules, to calculate space parameters to be used in calculating various disturbances acting on the satellite, Also, to calculate measurements fed back to the ADCS software and the actuating torque resulted from the magnetic torques on the satellite as both of measurements and torque depends on the magnetic field of the earth.

Then the second part is the dynamics of the system which include dynamics and kinematics models besides a Runge Kutta 4 integrator to perform system states propagation through time.

The next module is the sensor model which generate various measurements simulations to be fed back to the on-board ADCS software, these measurements are angular velocities and magnetic field vector measured in the device mounted axes. Also for the simulation to resemble the actual environment of satellite operation in add normal random noise to the measurements with specified mean and variance.

All previous modules together with the actuator model which represent saturation in control action and dead zone region and actuator efficiency (Fig(5)). All these modules together represent the computational part of the Flight Simulator which is added to the GUI to form a user friendly package for development and testing of ADCS software for magnetically actuated satellites. A sample of the GUI is examples in (fig (6))

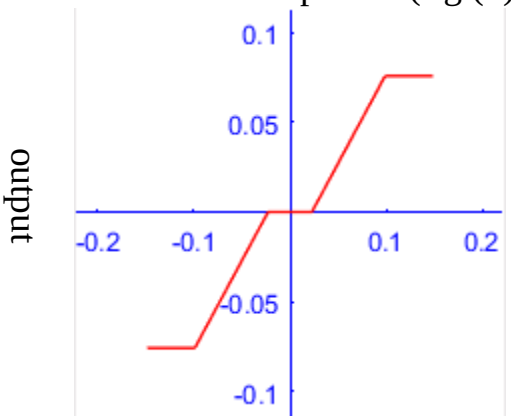


Figure 5| Actuator output diagram

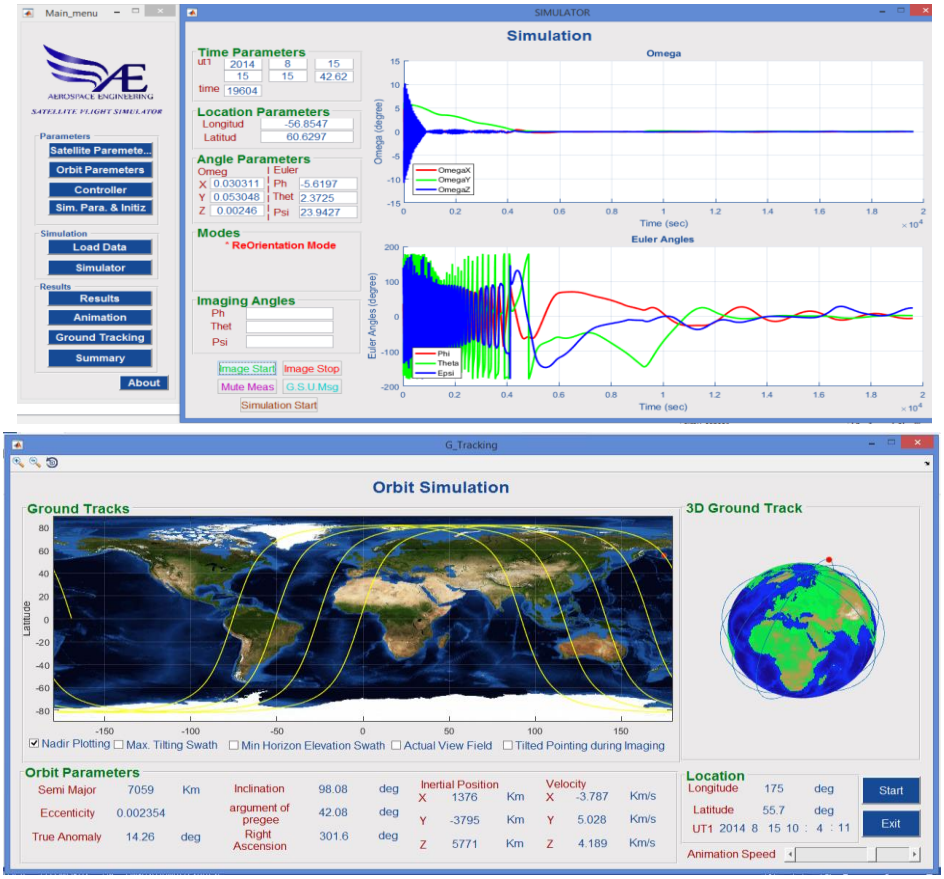


Figure 6| GUI Examples

While the ADCS software includes the attitude estimator which is based on Extended Kalman Filter Algorithm with full dynamics model including all disturbances stated before.

And controller algorithms for different modes of operations and stated before in satellite ADCS mission stages.

This software is written in C programming Language to be implemented on the onboard computer, Also it's required to be tested through different state including:

1. Matlab Model in the loop test (MIL)
2. C software in the loop test (SIL)
3. Finally, Processor in the loop test (PIL)

All the preceding topics is discussed in details throughout this documentation, hoping it would be helpful.

Chapter 2

Axes and Coordinates

2.1 The Different Representations of Three Axes

2.1.1 Direction Cosine Matrix (DCM)

a. Definition

The basic three-axis altitude transformation is based on the **direction cosine matrix**. Any altitude transformation is actually converted to this essential form.

In Fig. 7 The axes 1, 2, and 3 are unit vectors defining an orthogonal, right –handed triad. This triad is chosen as the reference inertial frame. Next, a similar orthogonal triad is attached to the center of mass of moving body, defined by the unit vector u, v, and w.

We Define Matrix [A] as follows:

$$[A] = \begin{bmatrix} u_1 & u_2 & u_3 \\ v_1 & v_2 & v_3 \\ w_1 & w_2 & w_3 \end{bmatrix}$$

In this matrix, u_1, u_2, u_3 are the components of the unit vector **u** along the three axes 1, 2, 3 of the reference orthogonal system: **u** = [u_1 u_2 u_3] ^{T}. In a similar way, **v** and **w** have components v_1, v_2, v_3 and w_1, w_2, w_3 along the same reference axes: **v** = [v_1 v_2 v_3] ^{T} and **w** = [w_1 w_2 w_3] ^{T}.

In other words, it is the cosine of the angle between the two transformed axes as example:u_1 is the cosine of the angle between axes u and 1.

b. Transform Vector from Frame to Another using DCM

The **direction cosine matrix** [A], also called the rotation matrix, has the important property of mapping vectors from the reference frame to the body frame, or Vice versa. Suppose that a vector **a** has components a_1, a_2, a_3 in the reference frame: $a = [\ a_1\ \ a_2\ \ a_3\ ]^T$.

The following matrix vector multiplication expresser the components of the vector **a** in the body frame:

$$[A]a = \begin{bmatrix} u_1 & u_2 & u_3 \\ v_1 & v_2 & v_3 \\ w_1 & w_2 & w_3 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} u \cdot a \\ v \cdot a \\ w \cdot a \end{bmatrix} = \begin{bmatrix} a_u \\ a_v \\ a_w \end{bmatrix} = a_B$$

Where $\mathbf{a_B}$ is the vector **a** mapped into the body frame. Since **u** is a unit vector, it follows that the scalar product $\mathbf{u \bullet a}$ is the component $\mathbf{a_u}$ of the vector **a** along the unit vector **u**. By the same reasoning, the components of the vector **a** on the remaining unit vectors of the body triad are $\boldsymbol{a_v}$ and $\mathbf{a_w}$. [1]

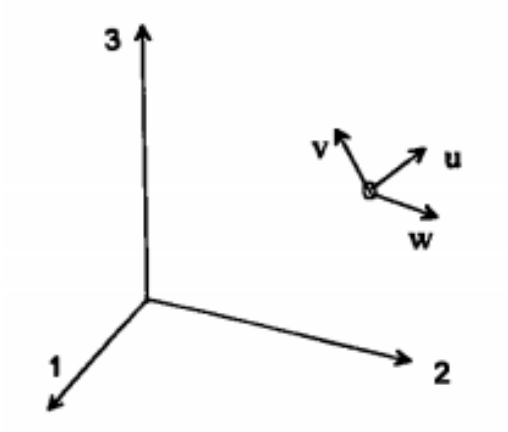


Figure 7| Definition of orientation of the spacecraft axes u, v, w in the reference frame 1, 2, 3.

2.1.2 Euler Angles

a. Definition

The **Euler angles** are three angles introduced by Leonhard Euler to describe the orientation of a rigid body with respect to a fixed coordinate system. They can also represent the orientation of a mobile frame of reference in physics or the orientation of a general basis in 3 dimensional linear algebra.

Any orientation can be achieved by composing three elemental rotations. **Euler angles** can be defined by three of these rotation. **Euler angles** are typically denoted as $\boldsymbol{\phi}$, $\boldsymbol{\theta}$, $\boldsymbol{\psi}$. That $\boldsymbol{\phi}$ is the rotation about X-axis, $\boldsymbol{\theta}$ is rotation about Y-axis, and $\boldsymbol{\psi}$ is rotation about Z-axis.

b. Euler Angles Rotation

The **Euler angle** rotation is defined as successive angular rotations about three orthogonal frame axes. Suppose we define the three orthogonal axes of the body frame by **i, j, k**, and those of the reference frame by **I, J, K.** There is a multitude of order combinations by which the rotation can be performed.

These combination can be successive rotation about each of the three axes **i, j, k**. in which there are six possible orders of rotation 1-2-3, 1-3-2, 2-1-3, 2-3-1, 3-1-2, and 3-2-1. [1]

To facilitate the Work we will deal with only on

$$[A_{213}] = \left[A_{\theta\phi\psi}\right] =$$

$$\begin{bmatrix} c(\psi)\,c(\theta) + s(\psi)\,s(\phi)\,s(\theta) & s(\psi)c(\phi) & -c(\psi)\,s(\theta) + s(\psi)\,s(\phi)\,c(\theta) \\ -s(\psi)\,c(\theta) + c(\psi)\,s(\phi)\,s(\theta) & c(\psi)\,c(\phi) & s(\psi)\,s(\theta) + c(\psi)\,s(\phi)\,c(\theta) \\ c(\phi)\,s(\theta) & -s(\phi) & c(\phi)\,c(\theta) \end{bmatrix} \quad (1)$$

In which, *c* is *cosine and s* is *sine.*

e combination in all of our work which it is **2-1-3.**

The remaining 5 transformations can be found in [1].

2.1.3 Quaternion

a. Definition

It is convenient mathematical notation for representation orientations and rotations of objects in three dimensions. Compared to **Euler angles** they are simpler to compose and avoid the problem of gimbal lock. Compared to **rotation matrices** they are more compact, more numerically stable, and may be more efficient.

The **quaternion's** basic definition is a consequence of the properties of the direction cosine matrix [A]. It is shown by linear algebra that a proper real orthogonal 3 x 3 matrix has at least one Eigen-vector with Eigen-value of unity. This means that, since one of the Eigen-values $\lambda_i (i = 1,2,3)$ is unity, the Eigen-vector is unchanged by the matrix [A].

$$[A]e_1 = 1e_1$$

The Eigen-vector e_1 has the same components along the body axes and along the reference frame axes. The existence of such an Eigen-vector is the analytical demonstration of Euler's famous theorem about rotational displacement "The most general displacement of a rigid body with one point fixed is a rotation about some axis". In this case, the rotation is about the Eigen-vector e_1. It will be demonstrated that any attitude transformation in space by a consecutive rotations about the three orthogonal unit vectors of a coordinate system can be achieved by a single rotation about the Eigen-vector with unity eigenvalue.

The **quaternion** representation of rigid body rotations as follows:

$$\boldsymbol{q = q_4 + iq_1 + jq_2 + kq_3}$$

according to Euler's rotation theorem, any rotation or sequence of rotations of a rigid body or coordinate system about a fixed point is equivalent to a single rotation by a given angle θ about a fixed axis (called the *Euler axis*) that runs through the fixed point.

The **Euler axis** is typically represented by a unit vector $\hat{e}$.Therefore, any rotation in three dimensions can be represented as a combination of a vector **e** and a scalar **θ**.

Quaternions give a simple way to encode this axis–angle representation in four numbers, and can be used to apply the corresponding rotation to a position vector, representing a point relative to the origin. [1]

$$q = \cos\left(\frac{\theta}{2}\right) + (u_x \boldsymbol{i} + u_y \boldsymbol{j} + u_z \boldsymbol{k})\sin\left(\frac{\theta}{2}\right)$$

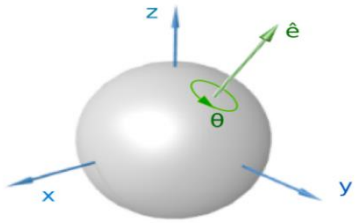


Figure 8| Euler Axis / Angle representation

b. Algebra of Quaternion

- *Conjugation of Quaternion*

The conjugate (or the inverse) of **quaternion** is rotation about opposite the original direction of **Euler axis** with the same value of θ and it's mathematically represented by as follows:

If we have a **quaternion** of $q = q_4 + iq_1 + jq_2 + kq_3$

Then the conjugate of this quantity is $q^* = q_4 - iq_1 - jq_2 - kq_3$

- *Quaternion Multiplication*

Quaternion multiplication is performed in the same manner as the multiplication of complex numbers or algebraic polynomials, except that the order of operations must be taken into account because is not commutative.

The **quaternion** multiplication can be understood as summation of 2 successive rotations and it's mathematically represented as follows:

Consider the product of two quaternions is as follows:

$$q'' = qq' = (q_4 + iq_1 + jq_2 + kq_3)(q'_4 + iq'_1 + jq'_2 + kq'_3)$$

Then it is reduced to

$$q'' = qq' = (-q_1 q_1' - q_2 q_2' - q_3 q'_3 + q_4 q_4')$$

$$+\boldsymbol{i}(q_1 q_4' + q_2 q_3' - q_3 q_2' + q_4 q_1')$$

$$+\boldsymbol{j}(-q_1 q_3' + q_2 q_4' \mp q_1' + q_4 q_2')$$

$$+\boldsymbol{k}(q_1 q_2' - q_2 q_1' + q_3 q_4' + q_4 q_3')$$

Which can be expressed as:

$$q'' = qq' = (q_4 q_4' - q \cdot q' \,,\ q_4 q' + q_4' q + q \times q')$$

For Computational purposes, it is convenient to express **quaternion** multiplication in matrix form .Specifically, let the components of $\boldsymbol{q}$ form a four-vector as follows:

$$\boldsymbol{q} = \begin{bmatrix} \boldsymbol{q_1} \\ \boldsymbol{q_2} \\ \boldsymbol{q_3} \\ \boldsymbol{q_4} \end{bmatrix}$$

In Matrix form,

$$\begin{bmatrix} \boldsymbol{q_1''} \\ \boldsymbol{q_2''} \\ \boldsymbol{q_3''} \\ \boldsymbol{q_4''} \end{bmatrix} = \begin{bmatrix} \boldsymbol{q_4'} & \boldsymbol{q_3'} & \boldsymbol{-q_2'} & \boldsymbol{q_1'} \\ \boldsymbol{-q_3'} & \boldsymbol{q_4'} & \boldsymbol{q_1'} & \boldsymbol{q_2'} \\ \boldsymbol{q_2'} & \boldsymbol{-q_1'} & \boldsymbol{q_4'} & \boldsymbol{q_3'} \\ \boldsymbol{-q_1'} & \boldsymbol{-q_2'} & \boldsymbol{-q_3'} & \boldsymbol{q_4'} \end{bmatrix} \begin{bmatrix} \boldsymbol{q_1} \\ \boldsymbol{q_2} \\ \boldsymbol{q_3} \\ \boldsymbol{q_4} \end{bmatrix}$$

- *Transform Vector from Frame to Another*

The transformation of vector **U,** corresponding to, multiplication by matrix **[A]**:

$$\boldsymbol{U' = AU}$$

The same effect in **quaternion** algebra by the following operation:

$$\boldsymbol{U' = q^{*}Uq}$$

Where q^{*} is the conjugate of **quaternion** "q".

- *The [A] Matrix in Terms of Quaternion*

$$[A(q)] = \begin{bmatrix} q_1^2 - q_2^2 - q_3^2 + q_4^2 & 2(q_1q_2 + q_3q_4) & 2(q_1q_3 - q_2q_4) \\ 2(q_1q_2 - q_3q_4) & -q_1^2 + q_2^2 - q_3^2 + q_4^2 & 2(q_2q_3 + q_1q_4) \\ 2(q_1q_3 + q_2q_4) & 2(q_2q_3 - q_1q_4) & -q_1^2 - q_2^2 + q_3^2 + q_4^2 \end{bmatrix} \quad (2)$$

- *Relative Rotation between 2 Quaternions*

To find the relative rotation between Q_1 and Q_2 is as follows:

$$Q_{difference} = conjugate(Q_1) * Q_2$$

c. Quaternion Transformations

The transformation of **quaternion** from and to **Euler angles** is very important. The representation of angles as **quaternion** will be very difficult to understand or to handle but it is stable and compact for computational process. In the other side **Euler angles** is more understandable to read and easy to handle.

The Transformation is done with respect of order of transformation of **Euler angles** (2-1-3).

- *Transformation from Quaternion to Euler angles*

Using the representation of **[A]** matrix in **quaternion** form (2) and the [A] matrix in **Euler angles** form (1) we can evaluate the **Euler angles**. [2]

That,

$$\phi = \tan^{-1}\left(\frac{-A(q)[3,2]}{\sqrt{1-(A(q)[3,2])^2}}\right)$$

$$\theta = \tan^{-1}\left(\frac{A(q)[3,1]}{A(q)[3,3]}\right)$$

$$\psi = \tan^{-1}\left(\frac{A(q)[1,2]}{A(q)[2,2]}\right)$$

- *Transformation from Euler to Quaternion*

Using the Transformations used in Ref [2].

$$q_4 = \sin\left(\frac{\phi}{2}\right)\sin\left(\frac{\theta}{2}\right)\sin\left(\frac{\psi}{2}\right) + \cos(\frac{\phi}{2})\cos(\frac{\theta}{2})\cos(\frac{\psi}{2})$$

$$q_1 = \sin\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right)\sin\left(\frac{\psi}{2}\right) + \cos(\frac{\phi}{2})\sin(\frac{\theta}{2})\cos(\frac{\psi}{2})$$

$$q_2 = \sin\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right)\cos\left(\frac{\psi}{2}\right) - \cos(\frac{\phi}{2})\sin(\frac{\theta}{2})\sin(\frac{\psi}{2})$$

$$q_3 = -\sin\left(\frac{\phi}{2}\right)\sin\left(\frac{\theta}{2}\right)\cos\left(\frac{\psi}{2}\right) + \cos(\frac{\phi}{2})\cos(\frac{\theta}{2})\sin(\frac{\psi}{2})$$

2.1.4 Comparison between Direction Cosine Matrix & Quaternion & Euler Angles

Parameterization	Advantages	Disadvantages	Common Application
Direction Cosine Matrix	-No Singularities. -No Trigonometric Functions. -Convenient product rule for successive rotations.	-Six redundant parameters.	-In Analysis. -To Transform Vectors from one reference frame to another.
Quaternion	-No Singularities. -No Trigonometric Functions. -Convenient product rule for successive rotations.	-One redundant parameter -No obvious physical interpretation	-Onboard inertial navigation
Euler Angles	-No redundant parameters. -Physical interpretation is clear in some cases.	- Trigonometric Functions. -Singularities at some cases. -No Convenient product rule for successive rotations.	-Analytical Studies. -Input / Output. -Onboard attitude control of 3-axis stabilized spacecraft.

2.2 Coordinate Systems

2.2.1 Definition

One of the first requirements for describing an orbit is to define a suitable reference system. We define a rectangular coordinate system by specifying its origin, fundamental plane, and the preferred direction. In addition, we must specify the sense, or the positive direction.

Most systems have a right-handed sense, the positive directions of each axis form an orthogonal triplet that's oriented with the thumb, index, and middle finger of the right hand. Some systems have a left-handed sense of direction. This may be confusing, so it will be mentioned when a system is left handed.

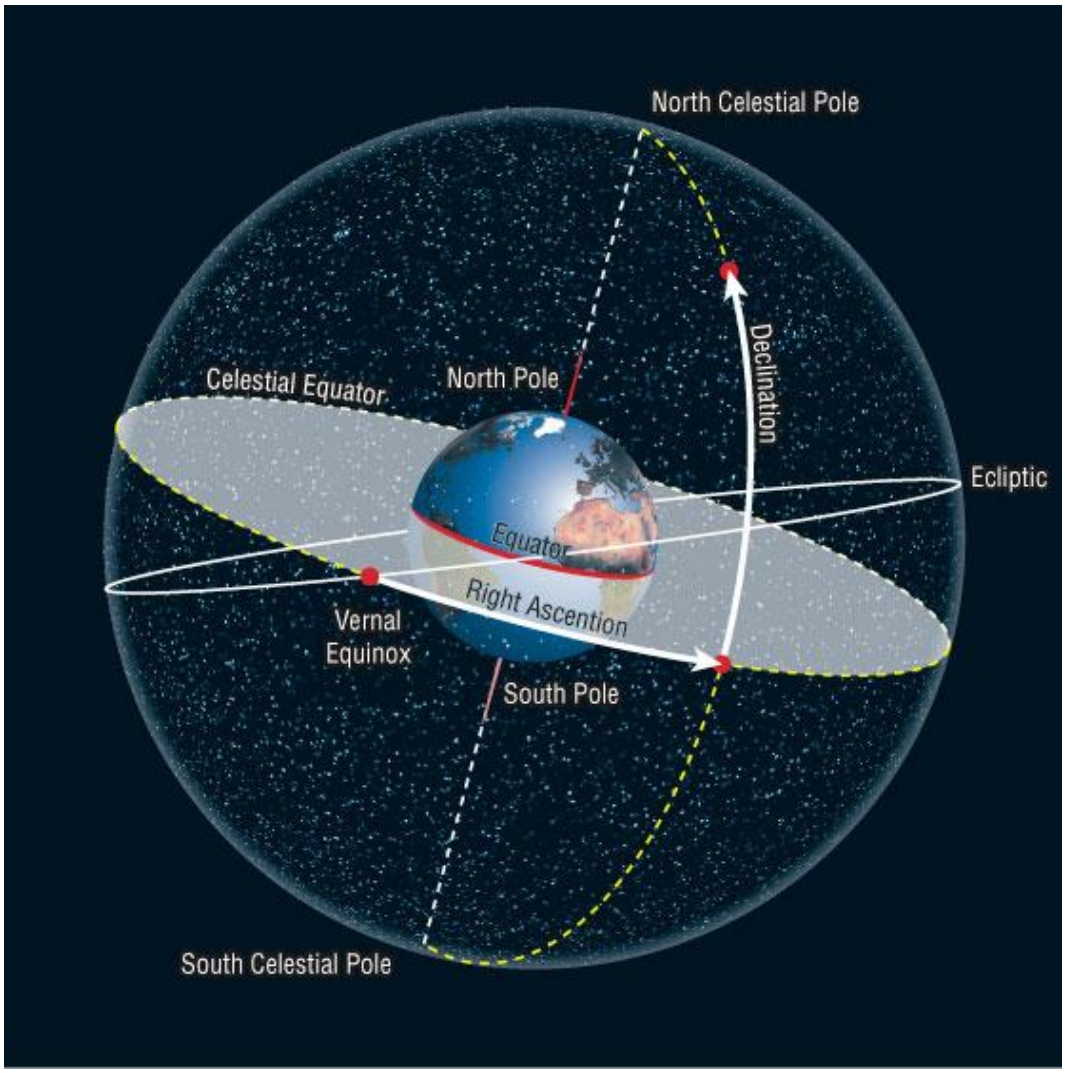


Figure 9 | **Geometry of the Celestial Sphere.**
The celestial sphere is based on an observer's perceived view of objects in space

The origin of the coordinate system helps the naming of each type. In general, there are three common designations: an object's center of mass, the system's center of mass (or *barycenter)* and a rotating system using the barycenter, called the *synodic* system.

a. Celestial sphere

Some concepts transcend all coordinate systems. In particular, because of the great distances to natural celestial objects compared to the Earth's size, these objects appear to be fixed to, or move on the inner surface of, a *celestial sphere* with the observer at the center Fig.(**9**). *Great circles* are the intersections of the celestial sphere with any plane passing through the center of the sphere. The *celestial poles* (north and south) result from the intersection of the Earth's rotational axis and the celestial sphere. The *celestial equator* projects the Earth's equator onto the celestial sphere. *Hour circles* are great circles that are perpendicular to the celestial equator.

This framework allows us to define the direction of objects in space. It also allows us to describe the position of objects closer to the Earth. We uniquely determine the direction of a point on the celestial sphere by two angular coordinates. One coordinate describes the angular distance of the point of interest above or below the fundamental plane, and the other describes the angle from the reference along the celestial equator to the hour circle passing through the object. These coordinates are analogous to latitude and longitude and are called declination and right ascension, respectively.

b. Ecliptic Plane & equinox

The Earth and its orbit around the Sun form the basis for celestial coordinate systems. The ***ecliptic*** is the plane of the Earth's mean orbit about the Sun (we assume it is free of periodic variations).
The term comes from the fact that eclipses of the Moon occur only when the Moon is close enough to this plane and is between the Earth and the Sun. When we view the Sun from the Earth, it appears to move along the ecliptic as shown in Fig. **10**. It doesn't exactly follow the ecliptic because this path is defined as the plane of the Earth's mean orbit.

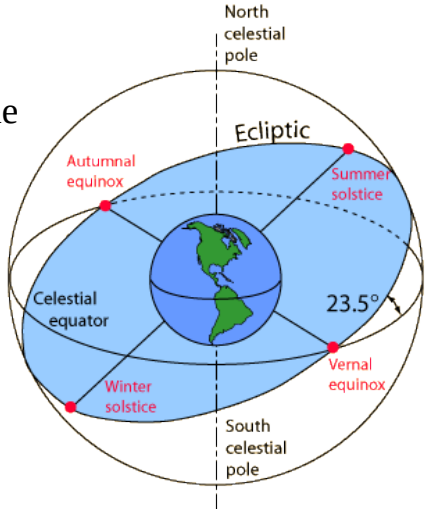


Figure 10 | the ecliptic plane and vernal equinox

The angle between the Earth's mean equator and the ecliptic is called the obliquity of the ecliptic. This angle is about 23.5°, although it does vary slightly over time due to perturbations.

The intersection of the two planes helps us fix a principal direction. The line of intersection is called the line of nodes. The Sun crosses this intersection twice a year. They're called equinoxes*:* one when the Sun is at the ascending node (in the spring about March 21, *vernal equinox)* and one when the Sun is at the descending node (in the fall about September 23, *autumnal equinox).* Remember, the seasons cited are for the **Northern Hemisphere**. When the Sun is at an equinox, the Earth experiences equal times of day and night because the Sun's declination is zero-equinox.

c. Vernal equinox

A formal definition for the ***vernal equinox*** is that it occurs when the Sun's declination is 0° as it changes from negative to positive values. This point differs slightly from the intersection of the ecliptic and the equator because the ecliptic is the mean path of the Sun, not the true (or actual) path. In other words, the vernal equinox occurs at the ascending node of the Sun as viewed from the Earth.

d. Hour Angle

We use hour angles to describe observations of celestial objects relative to a local observer. Hour angles are easiest to visualize by looking at a diagram. They measure the angular distance along the celestial equator of an object, and are analogous to longitude. The hour circle that passes through the observer is 0 hours-it's called the ***primary hour circle***. The *hour angle* of any object is the angle from the primary hour circle to the hour circle of the object. The units are usually hours from 0 to 24. This is a left-handed system, with a sign convention in which angles are measured positively *westward* to the object.
The current hour angle of any object is equal to the elapsed time from when the object was overhead. This definition applies to all objects (Sun, stars, and satellites) for local observers (the *Local Hour Angle*, LHA) and observers at Greenwich (the *Greenwich Hour Angle*, GHA). [3]

$$LHA = GHA + \lambda$$

2.2.2 Earth Based Systems

a. Greenwich System Coordinate (Earth-Centered, Earth-Fixed "ECEF")

ECEF also known as ECR ("Earth-Centered Rotational"), is a geographic coordinate system and Cartesian coordinate system. It represents positions as an X, Y, and Z coordinate. The point (0,0,0) is defined as the center of mass of the earth, hence the name "earth-centered. Its axes are aligned with the international reference pole (IRP) and international reference meridian (IRM) that are fixed with respect to the surface of the earth. The x-axis intersects the sphere of the earth at 0° latitude (the equator) and 0° longitude (prime meridian in Greenwich). This means that ECEF rotates with the earth, and therefore coordinates of a point fixed on the surface of the earth do not change.

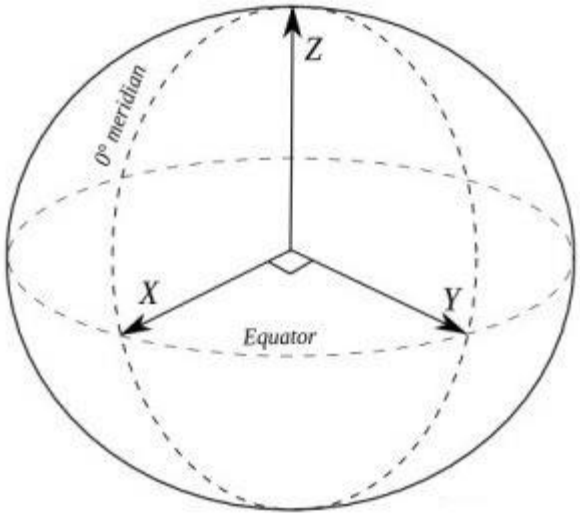


Figure 11 | Greenwich Coordinate Axes

b. Inertial System Coordinate (Earth-Centered inertial "ECI")

Earth-centered inertial (ECI) coordinate frames have their origins at the center of mass of the Earth. ECI frames are called inertial in contrast to the Earth-centered, Earth-fixed (ECEF) frames which rotate in inertial space in order to remain fixed with respect to the surface of the Earth. This means that ECI doesn't rotate with the earth.

The X-axis is fixed towards the vernal equinox which it is a vector from earth towards the sun at specific day (20 March), while The Z-axis is toward to the North Pole of Earth.

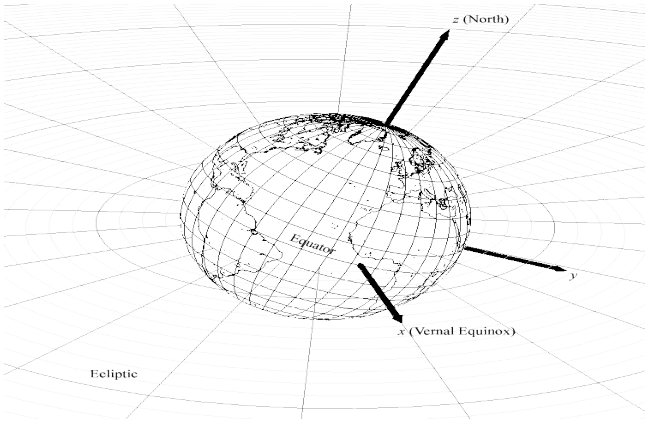


Figure 12 | Inertia Coordinate System

c. Transformation between Greenwich (ECEF) & Inertia (ECI) Coordinate Systems

The Transformation Matrix from Greenwich coordinate system to inertial Coordinate system in terms of Sidereal time "S" is as follows:

$$T^{GI} = \begin{bmatrix} \cos(S) & -\sin(S) & 0 \\ \sin(S) & \cos(S) & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (4.1)$$

Where:

S is the Sidereal Time that, $S = S_0 + W_e\, t$.
S_0 is Initial Sidereal Time.
W_e is the Angular velocity of earth rotation, $W_e = 7.292115 * 10^{-5}$rad/ sec.
t is the Current time , sec.

- *Transformation from Greenwich coordinate system to inertial coordinate system*

To transform a Vector **V** from Greenwich coordinate to inertial coordinate is as follows:

$$Vector^I = T^{GI} * Vector^G$$

- *Transformation from inertial coordinate system to Greenwich coordinate system*

To transform a Vector **V** from inertial coordinate to Greenwich coordinate is as follows:

$$Vector^G = [T^{GI}]^{-1} * Vector^I$$

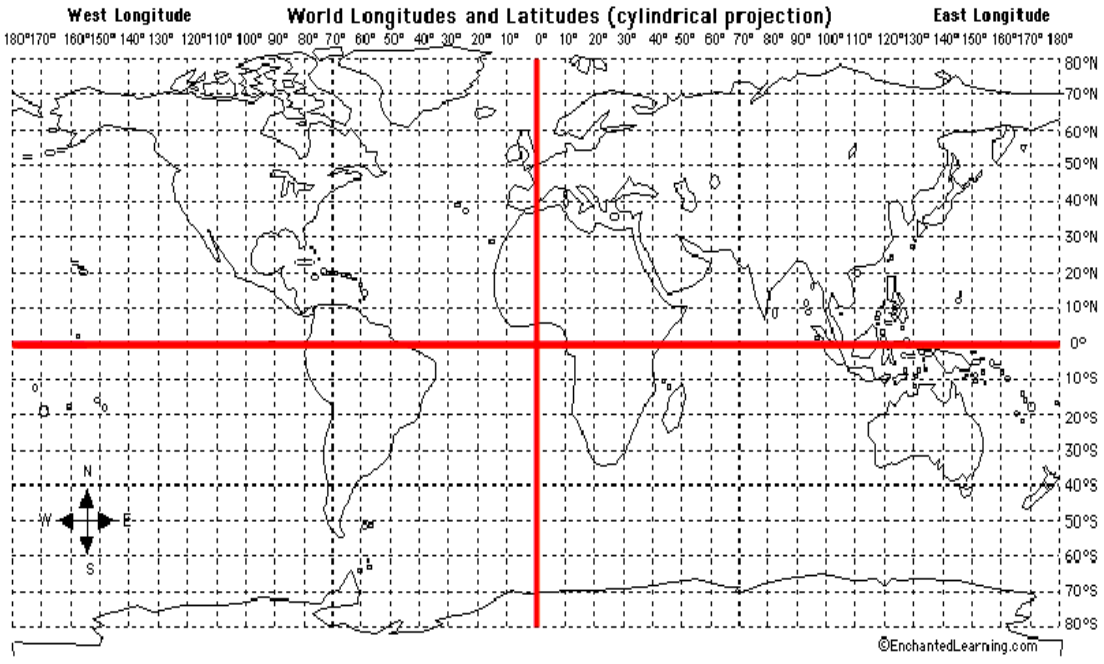
d. Transformation between ECEF to Latitude and Longitude

- *Shape of Earth*

Knowledge of the Earth's shape is essential to many astrodynamics studies including locating ground stations, remote sensing, geodesy, oceanography, plate tectonics, and viewing constraints. We often use a spherical Earth, which is the simplest representation with the same center of gravity and mass as the real Earth. The spherical model is sufficient for many studies, but several other models better represent the Earth's shape. In particular, ellipsoids of revolution (spheroids) can be very accurate.

Geocentric latitude, ϕ_{gc} is the angle measured at the Earth's center from the plane of the equator to the point of interest. ***Geodetic latitude,*** ϕ_{gd} is the angle between the equatorial plane and the normal to the surface of the ellipsoid.

But for simplicity we will use the geocentric latitude.



- *The Mathematical Transformation Algorithm*

A procedure to convert a position vector for a satellite to the corresponding latitude and longitude is the core technique in determining ground tracks. For this problem, we know the orbital elements of the satellite's position vector in the geocentric equatorial system (Earth fixed) but must find the geocentric latitude, longitude, and height of the satellite. The transformation method uses spherical trigonometry. We find the right ascension directly from the Cartesian position vector. Let the equatorial projection of the satellite's position vector be

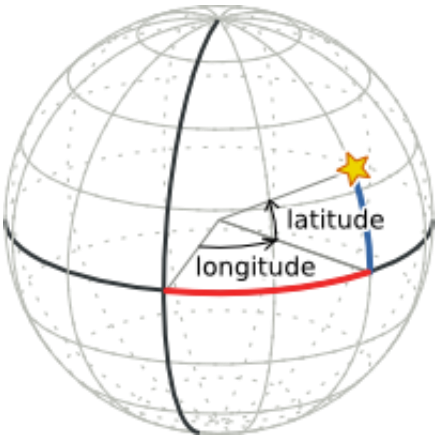
$$r_{\delta_{sat}} = \sqrt{r_i^2 + r_j^2}$$

So,

$$\sin(\alpha) = \frac{r_j}{r_{\delta_{sat}}} \qquad \cos(\alpha) = \frac{r_i}{r_{\delta_{sat}}}$$

$$\lambda = \alpha$$

$$\phi = \tan^{-1}\left(\frac{r_k}{r_{\delta_{sat}}}\right)$$



2.2.3 Satellite based Systems

a. Orbital Coordinate System

The **orbital Coordinate system** is a coordinate system move with the satellite. In which,

The R (**Z-axis for Satellite**) axis points out from the satellite along **the geocentric radius vector.**

The N **(J- axis for Satellite)** is normal to the orbital plane (the plane which created from the position vector and velocity vector).

The T (**I-axis for Satellite)** is normal to the position vector and positive in the direction of the velocity vector.

The **T axis is aligned** with the **velocity vector** only for **circular orbits**.

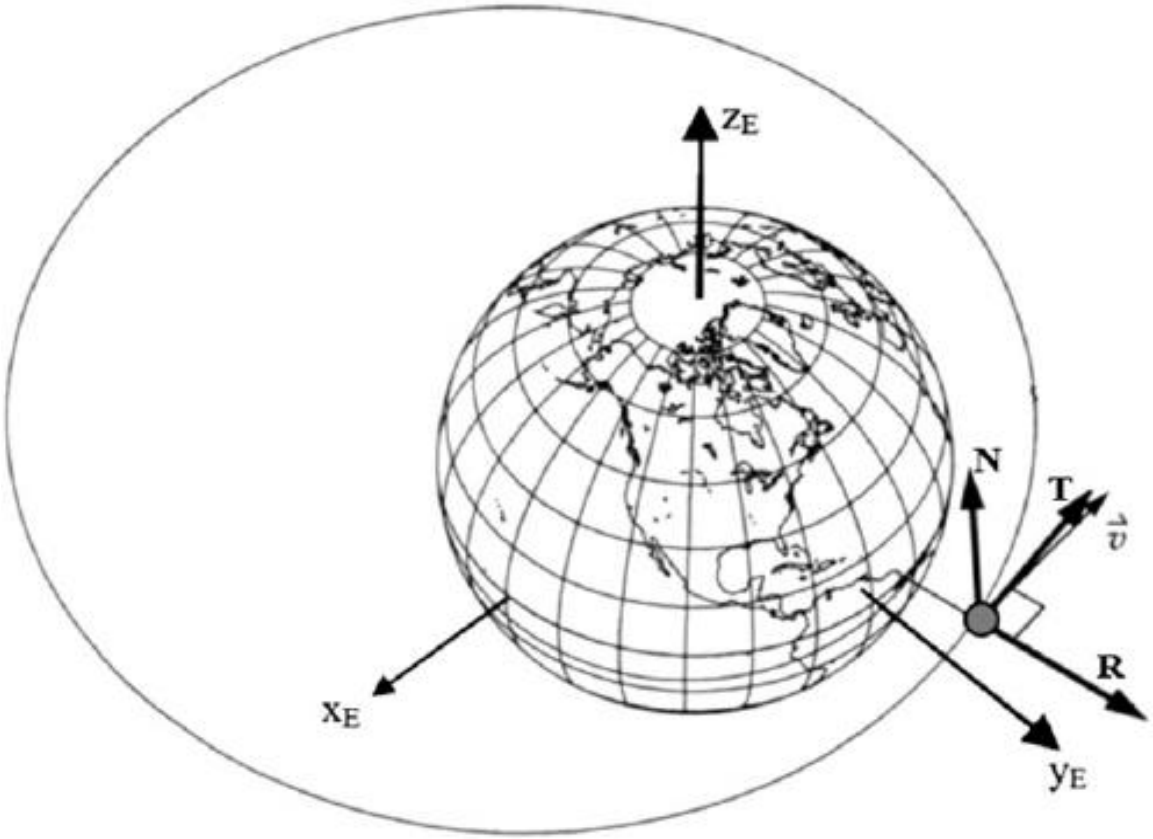


Figure 13 | Orbital Coordinate System

b. Body Coordinate System

We often form equations of motion using a system's center of mass, or Body axes as the origin. We will identify Body systems with a B subscript such as $X_{B,}$ $Y_{B,}$ and Z_B.

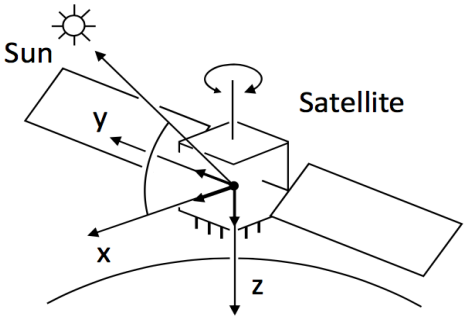


Figure 14 | Body Coordinate System

c. Transformation between Body and Orbital Coordinate Systems

To transform any vector between orbital and body axis, we need the rotation matrix and the required vector to transform (velocity, angular velocity or magnetic field), but we will use the quaternion method to facilitate the operation as follows:

If the required vector to transform is $Vector = [V_x\,\boldsymbol{i} + V_y\,\boldsymbol{j} + V_z\,\boldsymbol{k}\,]$

Transform this to quaternion $Vector_{Quaternion} = [0 + V_x\,\boldsymbol{i} + V_y\,\boldsymbol{j} + V_z\,\boldsymbol{k}\,]$

And the Transformation matrix [T] can be expressed as Quaternion quantity

$$T_{Quaternion} = [T_{scalar} + T_x\,\boldsymbol{i} + T_y\,\boldsymbol{j} + T_z\,\boldsymbol{k}]$$

- ***Transformation from Orbital to Body Coordinate Systems***

The operation is done by performing a Quaternion Multiplication between the Quaternion quantity of vector and transformation.

$$Result_{Quaternion} = Conjugate(T_{Quaternion}) * Vector_{Quaternion} * T_{Quaternion}$$

$$Result_{Quaternion} = [R_{scalar} + R_x\,\boldsymbol{i} + R_y\,\boldsymbol{j} + R_z\,\boldsymbol{k}]$$

Then the Transformed Body system vector,

$$Vector_{Body} = [R_x\,\boldsymbol{i} + R_y\,\boldsymbol{j} + R_z\,\boldsymbol{k}]$$

- ***Transformation from Body to Orbital Coordinate Systems***

The operation is done by performing a Quaternion Multiplication between the Quaternion quantity of vector and transformation.

$$Result_{Quaternion} = T_{Quaternion} * Vector_{Quaternion} * Conjugate(T_{Quaternion})$$

$$Result_{Quaternion} = [R_{scalar} + R_x \, \boldsymbol{i} + R_y \, \boldsymbol{j} + R_z \, \boldsymbol{k}]$$

Then the Transformed Body system vector,

$$Vector_{Body} = [R_x \, \boldsymbol{i} + R_y \, \boldsymbol{j} + R_z \, \boldsymbol{k}]$$

d. Transformation between Satellite based systems and Earth based systems

After we classified the coordinate systems, it is important to know how to transform the vectors from one system related to earth to relate it to satellite or vice versa.

- ***Transformation from Inertial Coordinate System (ECI) to Orbital Coordinate system***

As we have the Position vector **R** and the Velocity Vector **V** of the Satellite related to inertial coordinate system, and the Coordinate of orbital coordinate system ($I_{orbital}$, $J_{orbital}$, $K_{orbital}$) will be as follows:

$$\boldsymbol{K_{Orbital}} = \frac{\boldsymbol{R_{Inertial}}}{|\boldsymbol{R_{inertial}}|}$$

$$\boldsymbol{J_{Orbital}} = \frac{\boldsymbol{V_{Inertial}} \times \boldsymbol{R_{Inertial}}}{|\boldsymbol{V_{Inertial}} \times \boldsymbol{R_{Inertial}}|}$$

$$\boldsymbol{I_{Orbital}} = \boldsymbol{J_{Orbital}} \times \boldsymbol{K_{Orbital}}$$

And to transform any vector such as $Vector_{ics}$ from Inertial Coordinate System to Orbital Coordinate system is as follows:

$$Vector_{ocs} = \begin{bmatrix} \boldsymbol{I_{Orbital}} \\ \boldsymbol{J_{Orbital}} \\ \boldsymbol{K_{Orbital}} \end{bmatrix} * Vector_{ics}$$

Chapter 3

Space Environment

3.1 The Earth's Magnetic Field (IGRF)

Although the general characteristics of the **Earth's magnetic field** have been known for centuries, the first systematic study of the field was initiated by the German mathematician and physicist Karl Gauss' in the early part of the nineteenth century. Since that time, a great deal of data has been accumulated, much of it as a result of spacecraft measurements during the 1960s. Although this body of data has served to increase our ability to accurately describe the field, it has not yet provided the key to the physical processes which produce it or perturb it.

3.1.1 Definition

The Earth's magnetic field is predominantly that of a magnetic dipole such as that produced by a sphere of uniform magnetization or a current loop. The strength of the dipole was $7.96\ x\ 10^{15}$ $(Wb \bullet m)$ in 1975. The "south" end of the dipole was in the northern hemisphere at 78.60° N latitude and 289.55° E longitude and drifting westward at about 0.014 degree/year. The dipole strength is decreasing by 0.05%/year. This secular drift implies a possible field reversal in several thousand years. There is ambiguous evidence of several reversals in the past with time scales of 70,000 to 100,000 years between reversals.

The plane perpendicular to the Earth-centered dipole is called the magnetic equator. The field is weakest there, being about $3\ x10^4$ (nT) at the surface of the Earth. Fig. (**15**) shows the variation in the dipole field strength as a function of altitude at the magnetic equator. The field strength increases by a factor of two as the magnetic latitude increases from 0 degree to 90 degree, as shown in Fig. (**16**). At the geomagnetic equator, the field is horizontal relative to the Earth's surface. At a geomagnetic latitude of about 27 degree, the field is 45 degree down from horizontal.

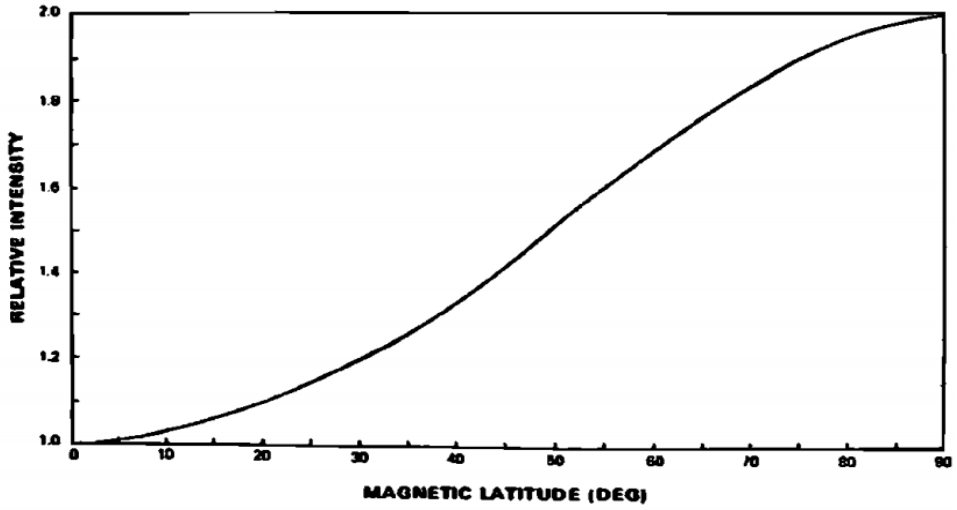


Figure 15| Relative Intensity of the Earth's Magnetic Field as a Function of Magnetic Latitude

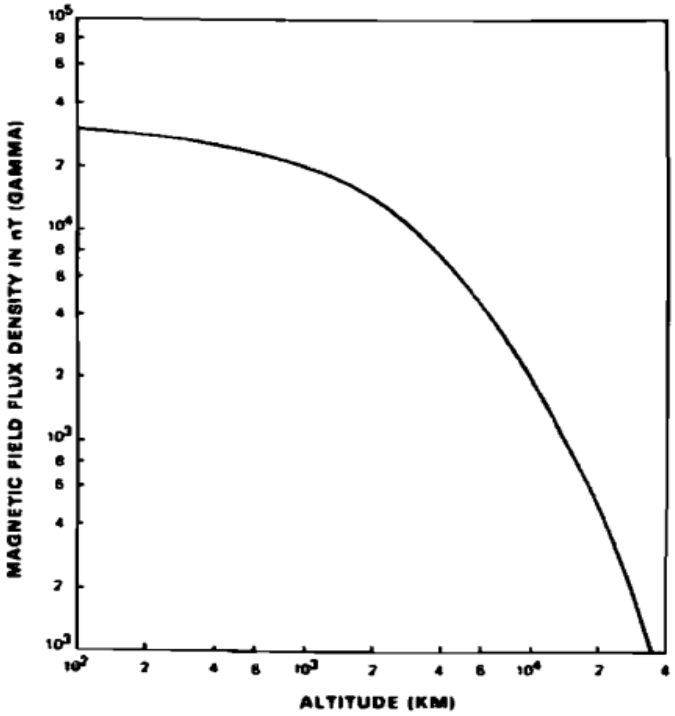


Figure 16 | Earth's Magnetic Field Intensity at the Magnetic Equator as a Function of Altitude.

Plots of the field strength for various altitudes are given in Figs. (**18**). and (**19**). Note that as the altitude increases, the contours become more regular and begin to resemble a dipole field more closely. The low in magnetic intensity at about 25°S, 45°W (called the Brazilian Anomaly) together with the high at about 10°N, 100°E implies that the center of the magnetic dipole is offset from the Earth's center.

In 1975, the eccentric dipole was offset 474.2 km in the direction of 19.5°N, 146.9°E. The eccentric nature of the dipole can be described mathematically as a quadrupole distribution of magnetization. The maximum deviations of the centered dipole model and the quadrupole model from the actual Field of the Earth are shown in **Fig. (17).**

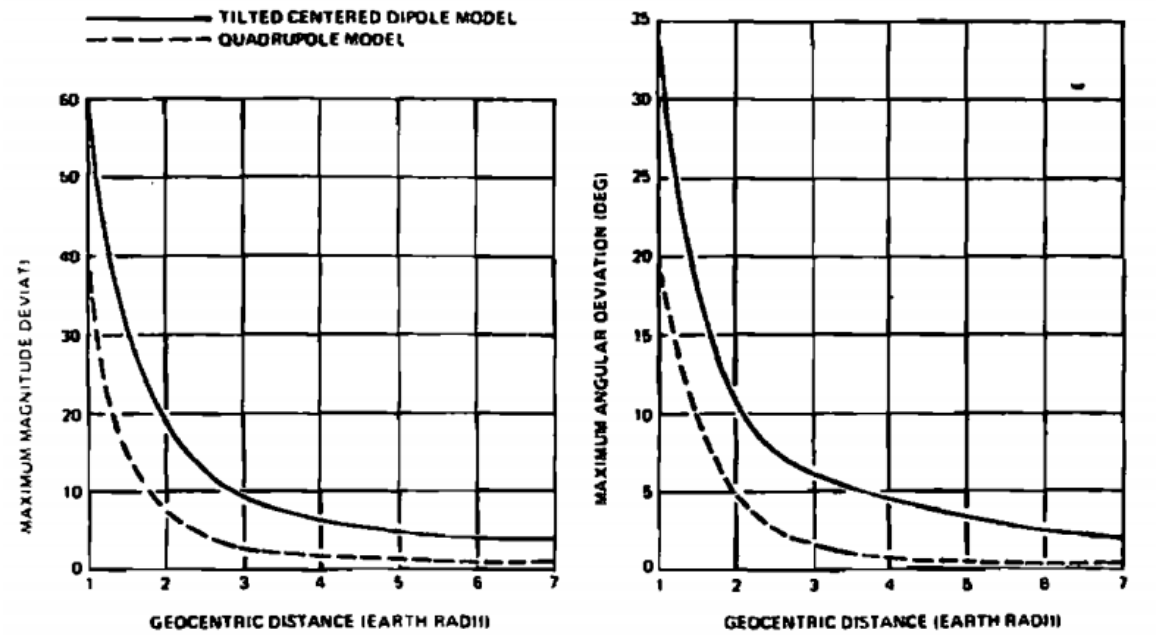


Figure 17 | Maximum Deviations of Approximate Models from the Earth's Magnetic Field [4]

The fact that the field rotates with the Earth is a clear indication that the field originates within the Earth. A coherent dipole field of this nature can be produced either by a uniformly magnetized sphere or by a current loop. However, calculations of the magnetization required lead to values much higher than those observed in the Earth's crust. Magnetization deeper than the crust is unlikely because the Curie point (i.e., the temperature at which a magnetized material loses its magnetization) of iron is reached only 20 km below the Earth's surface.

An alternative theory postulates a dynamo effect in the outer core of the Earth driven by thermal convection currents. Basically, a dynamo is a conductor driven in a magnetic field such that it acts to sustain that field. The theory has been refined to include a primary current which produces the dipole, plus secondary currents or whirlpools near the core-mantle boundary which produce local dipoles. These secondary dipoles are then superimposed to produce the observed multi-pole nature of the field, as well as local anomalies, which are large surface areas where the magnetic field deviates appreciably from the dipole field. The creation and decay of the whirlpools may cause the secular drift. Another theory of the secular drift is that the core is rotating more slowly than the mantle and crust. [4]

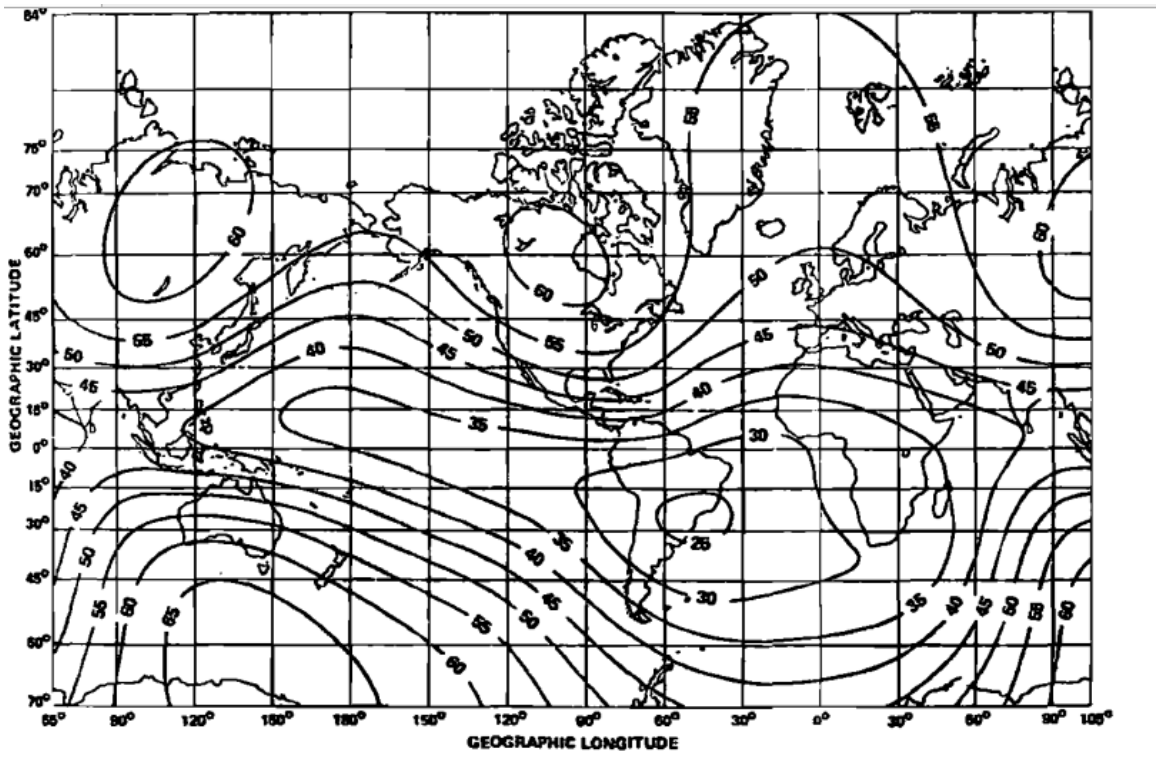


Figure 18 | Total Magnetic Field Intensity at the Earth's Surface

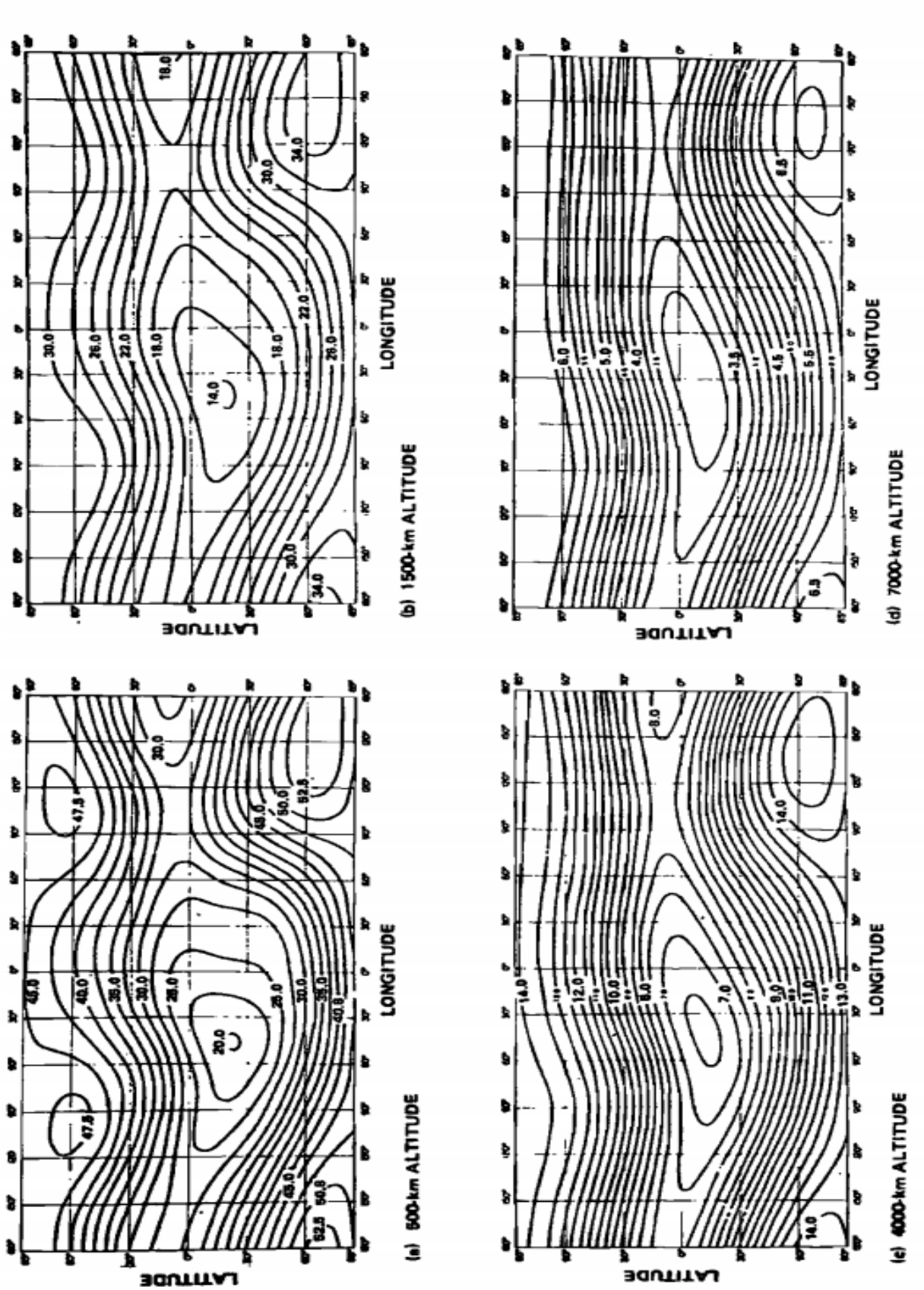


Figure 19 | Magnitude of the Earth's Magnetic Field [4]

3.1.2 **The Mathematical Model**

The field model includes the first-order time derivatives of the coefficients in an attempt to describe the secular variation. Because of the lack of adequate data over a long enough period of time, the accuracy of this (or any) field will degrade with time, that every **IGRF (International Geomagnetic Reference Field) model** is valid for 10 years before and after it has done.

The determination of the Gaussian coefficients is done empirically by doing least square fit to magnetic field data using coefficients as fitting parameters. Data consisting of both magnitude and direction is obtained from series of magnetic observatories. Unfortunately, these observatories are not distributed uniformly so that the data is sparse in some regions of the Earth. More uniformly distributed data is obtained from field magnitude measurements made by satellites. Although there are some theoretical arguments that obtaining coefficients by simple fitting field magnitudes is an ambiguous process, but it appear that it work quite well in practice.

The earth magnetic field T can be calculated using the following formula:

$$T_i = -R_e^2 * Real\left(\sum_{n=1}^{\infty}\sum_{m=0}^{n} R_e^n(g_n^m - j * h_n^m) * d_{n,m} * \frac{\partial_{n,m}}{\partial i}\right)$$

Where

$$i = x, y, z \quad , j = \sqrt{-1}$$

And,
R_e: is the Earth's Radius which is equal to 6378140 (meters)
Real: means the real part of the complex number
g_{nm} , h_{nm}; The Gaussian magnetic field coefficient

The values of $d_{n,m}$, $V_{n,m}$ and $\frac{\partial V_{n,m}}{\partial i}$ can be calculated as follow:

$$d_{1,0} = 1 \qquad d_{1,1} = 1$$

$$n = 2, \ldots\ldots, N_{max}; \quad m = 1, \ldots\ldots, (n-1)$$

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

$$d_{n,0} = 1$$

$$d_{n,n} = d_{n-1,n-1} * 1/\sqrt{(2n-1) * 2n}$$

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

$$V_{n,n} = (2n-1) * \frac{x+jy}{r^2} * V_{n-1,n-1}$$

$$V_{n,m} = \begin{cases} \frac{(2n-1)}{n-m} * \frac{z}{r^2} * V_{n-1,m} \ , & n-m=1 \\ \frac{(2n-1)}{n-m} * \frac{z}{r^2} * V_{n-1,m} - \frac{(n+m-1)}{n-m} * \frac{1}{r^2} * V_{n-2,m} \ , & n-m>1 \end{cases}$$

$$\frac{\partial V_{n,m}}{\partial x} = \begin{cases} -\frac{1}{2} * \left(V_{n+1,1} + V_{n+1,1}^*\right) \ , & m=0 \\ -\frac{1}{2} * \left(V_{n+1,m+1} - \frac{(n-m+2)!}{(n-m)!} V_{n+1,m-1}\right) \ , & m>0 \end{cases}$$

$$\frac{\partial V_{n,m}}{\partial y} = \begin{cases} \frac{i}{2} * \left(V_{n+1,1} - V_{n+1,1}^*\right) \ , & m=0 \\ \frac{i}{2} * \left(V_{n+1,m+1} + \frac{(n-m+2)!}{(n-m)!} V_{n+1,m-1}\right) \ , & m>0 \end{cases}$$

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

Where the complex-conjugate value of function $V_{n+1,1}$ is denoted as $V_{n+1,1}^*$. The calculation of earth magnetic field by mentioned formulas returns the result in (nT).

The values of coefficients g_{nm} , h_{nm} of first harmonic up to 13 orders inclusive, reducing the epoch of 2010 will be used with the help of calculation of coefficients by the formulas.

$$t_x = \frac{T_3 - epoch_3}{365.25} + \frac{T_2 - epoch_2}{12} + T_1 - epoch_1$$

$$g_{nm} = g_{nm}(\text{epoch}) + \dot{g}_{nm} * t_x$$

$$h_{nm} = h_{nm}(\text{epoch}) + \dot{h}_{nm} * t_x$$

Where: T：Initial date of launching satellite.
epoch：Reference time of Gaussian coefficients g_{nm} , h_{nm} calculation.

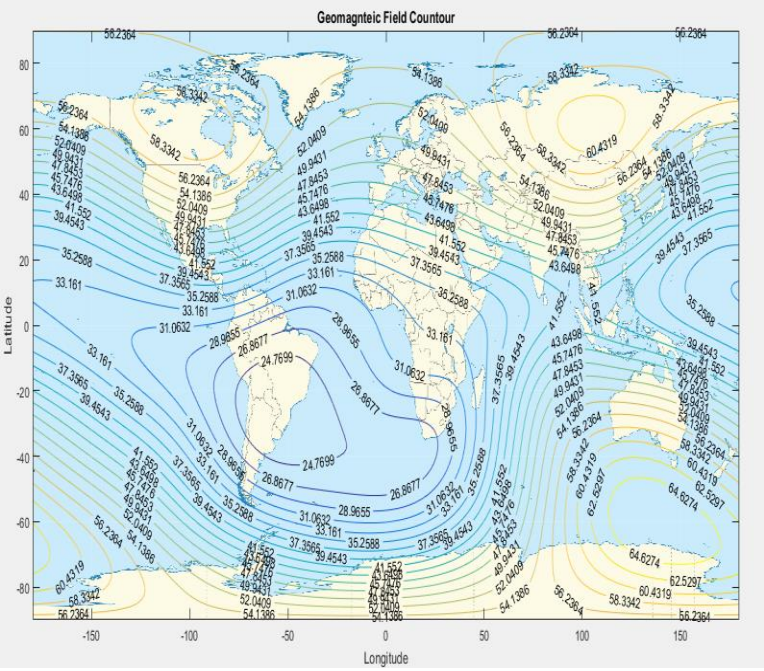
- ***IGRF Spherical Harmonic Coefficients at 2010***

n	m	g_n^m	h_n^m	$\dot{g}_n^m$	$\dot{h}_n^m$	n	m	g_n^m	h_n^m	$\dot{g}_n^m$	$\dot{h}_n^m$
1	0	-29497	0	11.4	0	9	8	-8.4	-6.1	0	0
1	1	-1585.9	4945.1	16.7	-28.8	9	9	-10.1	7	0	0
2	0	-2396.6	0	-11.3	0	10	0	-2	0	0	0
2	1	3026	-2707.7	-3.9	-23	10	1	-6.3	2.8	0	0
2	2	1668.6	-575.4	2.7	-12.9	10	2	0.9	-0.1	0	0
3	0	1339.7	0	1.3	0	10	3	-1.1	4.7	0	0
3	1	-2326.3	-160.5	-3.9	8.6	10	4	-0.2	4.4	0	0
3	2	1231.7	251.7	-2.9	-2.9	10	5	2.5	-7.2	0	0
3	3	634.2	-536.8	-8.1	-2.1	10	6	-0.3	-1	0	0
4	0	912.6	0	-1.4	0	10	7	2.2	-4	0	0
4	1	809	286.4	2	0.4	10	8	3.1	-2	0	0
4	2	166.6	-211.2	-8.9	3.2	10	9	-1	-2	0	0
4	3	-357.1	164.4	4.4	3.6	10	10	-2.8	-8.3	0	0
4	4	89.7	-309.2	-2.3	-0.8	11	0	3	0	0	0
5	0	-231.1	0	-0.5	0	11	1	-1.5	0.1	0	0
5	1	357.2	44.7	0.5	0.5	11	2	-2.1	1.7	0	0
5	2	200.3	188.9	-1.5	1.5	11	3	1.6	-0.6	0	0
5	3	-141.2	-118.1	-0.7	0.9	11	4	-0.5	-1.8	0	0

5	4	-163.1	0.1	1.3	3.7	11	5	0.5	0.9	0	0
5	5	-7.7	100.9	1.4	-0.6	11	6	-0.8	-0.4	0	0
6	0	72.8	0	-0.3	0	11	7	0.4	-2.5	0	0
6	1	68.6	-20.8	-0.3	-0.1	11	8	1.8	-1.3	0	0
6	2	76	44.2	-0.3	-2.1	11	9	0.2	-2.1	0	0
6	3	-141.4	61.5	1.9	-0.4	11	10	0.8	-1.9	0	0
6	4	-22.9	-66.3	-1.6	-0.5	11	11	3.8	-1.8	0	0
6	5	13.1	3.1	-0.2	0.8	12	0	-2.1	0	0	0
6	6	-77.9	54.9	1.8	0.5	12	1	-0.2	-0.8	0	0
7	0	80.4	0	0.2	0	12	2	0.3	0.3	0	0
7	1	-75	-57.8	-0.1	0.6	12	3	1	2.2	0	0
7	2	-4.7	-21.2	-0.6	0.3	12	4	-0.7	-2.5	0	0
7	3	45.3	6.6	1.4	-0.2	12	5	0.9	0.5	0	0
7	4	14	24.9	0.3	-0.1	12	6	-0.1	0.6	0	0
7	5	10.4	7	0.1	-0.8	12	7	0.5	0	0	0
7	6	1.6	-27.7	-0.8	-0.3	12	8	-0.4	0.1	0	0
7	7	4.9	-3.4	0.4	0.2	12	9	-0.4	0.3	0	0
8	0	24.3	0	-0.1	0	12	10	0.2	-0.9	0	0
8	1	8.2	10.9	0.1	0	12	11	-0.8	-0.2	0	0
8	2	-14.5	-20	-0.5	0.2	12	12	0	0.8	0	0
8	3	-5.7	11.9	0.3	0.5	13	0	-0.2	0	0	0
8	4	-19.3	-17.4	-0.3	0.4	13	1	-0.9	-0.8	0	0
8	5	11.6	16.7	0.3	0.1	13	2	0.3	0.3	0	0
8	6	10.9	7.1	0.2	-0.1	13	3	0.4	1.7	0	0
8	7	-14.1	-10.8	-0.5	0.4	13	4	-0.4	-0.6	0	0
8	8	-3.7	1.7	0.2	0.4	13	5	1.1	-1.2	0	0
9	0	5.4	0	0	0	13	6	-0.3	-0.1	0	0
9	1	9.4	-20.5	0	0	13	7	0.8	0.5	0	0
9	2	3.4	11.6	0	0	13	8	-0.2	0.1	0	0
9	3	-5.3	12.8	0	0	13	9	0.4	0.5	0	0
9	4	3.1	-7.2	0	0	13	10	0	0.4	0	0
9	5	-12.4	-7.4	0	0	13	11	0.4	-0.2	0	0
9	6	-0.8	8	0	0	13	12	-0.3	-0.5	0	0
9	7	8.4	2.2	0	0	13	13	-0.3	-0.8	0	0

3.1.3 Validation

To validate our model, we need to compare our work's results with another model results.



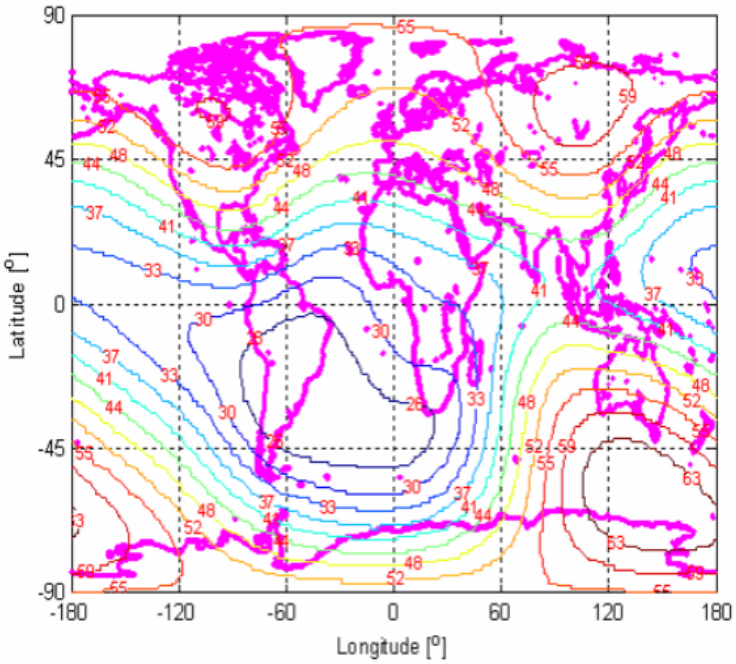
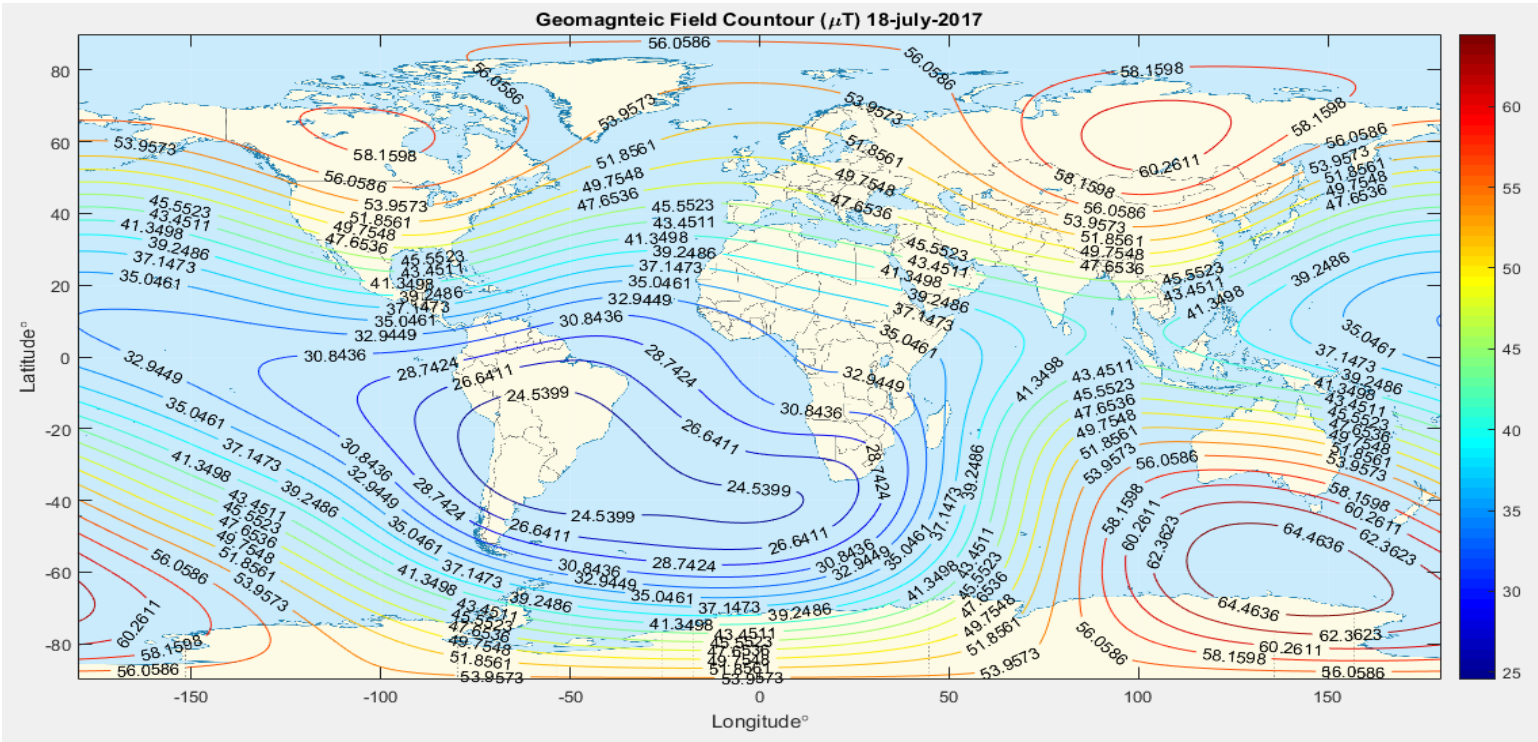


Figure 20 | (on left) the result of our model, (on right) the model of Doctor Tamer Mekky's Model (at [2006 1 1] ,the magnetic field is at μT)



3.2 The Earth Gravitational Field (The Spherical Harmonics Gravity)

3.2.1 Definition

We use the spherical harmonics model to propagate in an orbit as the gravitational acceleration can be integrated to get the position and velocity in the orbit.

Two point masses, ***M*** and **m**, separated by a vector distance **r**, attract each other with a force given by Newton's law of gravitation as:

$$\boldsymbol{F} = \frac{\boldsymbol{G\,M\,m}}{\boldsymbol{r^2}}\,\hat{\boldsymbol{r}}$$

Where ***G: is*** the gravitational constant.

If $\boldsymbol{M_{\oplus}}$ 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, $\boldsymbol{\mu_{\oplus}}$, and the Earth gravitational potential,$\boldsymbol{U}$,by:

$$\boldsymbol{\mu_{\oplus}} = \boldsymbol{G\,M_{\oplus}}$$

$$\boldsymbol{U} = -\frac{\boldsymbol{G\,M_{\oplus}}}{\boldsymbol{r}}$$

So the Force law can be simplified to:

$$\boldsymbol{F} = -\frac{\boldsymbol{\mu_{\oplus}\,m}}{\boldsymbol{r^2}}\,\hat{\boldsymbol{r}}$$

$$\boldsymbol{F} = -\boldsymbol{m}\,\boldsymbol{\nabla U}$$

3.2.2 The Mathematical Model

The earth gravity can be calculated using the following formula in Greenwich coordinate system (GCS):

$$g = \mu \begin{bmatrix} \frac{\partial U}{\partial x} \\ \frac{\partial U}{\partial y} \\ \frac{\partial U}{\partial z} \end{bmatrix}$$

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

Where,

$$i = x, y, z$$

μ : Earth gravitational constant $3.98600441800 * 10^{14}$ $(\frac{\text{meter}^3}{\text{second}^2})$.

R_e : is the Earth Radius which is equal to 6378140 (meters).

C_{nm}, S_{nm}; The earth gravity coefficient.

And these values of $V_{n,m}$ and $\frac{\partial V_{n,m}}{\partial i}$ can be calculated as follow:

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

$$V_{n,n} = (2n-1) * \frac{x + jy}{r^2} * V_{n-1,n-1}$$

$$V_{n,m} = \begin{cases} \frac{(2n-1)}{n-m} * \frac{z}{r^2} * V_{n-1,m} \ , & n - m = 1 \\ \frac{(2n-1)}{n-m} * \frac{z}{r^2} * V_{n-1,m} - \frac{(n+m-1)}{n-m} * \frac{1}{r^2} * V_{n-2,m} \ , & n - m > 1 \end{cases}$$

$$\frac{\partial V_{n,m}}{\partial x} = \begin{cases} -\frac{1}{2} * \left(V_{n+1,1} + V^*_{n+1,1}\right) , & m = 0 \\ -\frac{1}{2} * \left(V_{n+1,m+1} - \frac{(n-m+2)!}{(n-m)!} V_{n+1,m-1}\right) , & m > 0 \end{cases}$$

$$\frac{\partial V_{n,m}}{\partial y} = \begin{cases} \frac{i}{2} * \left(V_{n+1,1} - V^*_{n+1,1}\right) , & m = 0 \\ \frac{i}{2} * \left(V_{n+1,m+1} + \frac{(n-m+2)!}{(n-m)!} V_{n+1,m-1}\right) , & m > 0 \end{cases}$$

9.6889·10-8

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

Where $V^*_{n+1,1}$: is the complex conjugate value of $V_{n+1,1}$.

Table of Earth Gravity Coefficients Cnm, Snm

n	*m*	C_{nm}	S_{nm}	*n*	*m*	C_{nm}	S_{nm}
1	1	1	0	6	2	$-4.5958 \cdot 10^{-7}$	$-6.8485 \cdot 10^{-8}$
3	1	$1.0826 \cdot 10^{-3}$	0	6	3	$9.6889 \cdot 10^{-8}$	$6.4588 \cdot 10^{-8}$
3	3	$-1.5362 \cdot 10^{-6}$	$-8.8149 \cdot 10^{-7}$	6	4	$-1.9302 \cdot 10^{-8}$	$-5.3972 \cdot 10^{-9}$
4	1	$-2.5410 \cdot 10^{-6}$	0	6	5	$-9.0188 \cdot 10^{-10}$	$-3.5344 \cdot 10^{-10}$
4	2	$2.1578 \cdot 10^{-6}$	$2.4127 \cdot 10^{-7}$	6	6	$3.4363 \cdot 10^{-10}$	$-2.1382 \cdot 10^{-9}$
4	3	$2.6584 \cdot 10^{-7}$	$-2.5795 \cdot 10^{-7}$	7	1	$5.5201 \cdot 10^{-7}$	0
4	4	$6.8343 \cdot 10^{-8}$	$2.1311 \cdot 10^{-7}$	7	2	$-5.8780 \cdot 10^{-8}$	$1.3970\ 10^{-8}$
5	1	$-1.6180 \cdot 10^{-6}$	0	7	3	$3.0690 \cdot 10^{-9}$	$-5.0575 \cdot 10^{-7}$
5	2	$-4.9093 \cdot 10^{-7}$	$-4.5670 \cdot 10^{-7}$	7	4	$9.1517 \cdot 10^{-11}$	$6.0242 \cdot 10^{-10}$
5	3	$7.6688 \cdot 10^{-8}$	$1.5021 \cdot 10^{-7}$	7	5	$-3.7866 \cdot 10^{-10}$	$-1.1469\ 10^{-9}$
5	4	$6.2092 \cdot 10^{-8}$	$-7.1254 \cdot 10^{-9}$	7	6	$-1.0899 \cdot 10^{-10}$	$-4.9202 \cdot 10^{-10}$
5	5	$-2.2211 \cdot 10^{-9}$	$7.5348 \cdot 10^{-9}$	7	7	$-6.7881 \cdot 10^{-12}$	$-6.1337 \cdot 10^{-11}$
6	1	$-2.2800 \cdot 10^{-7}$	0				

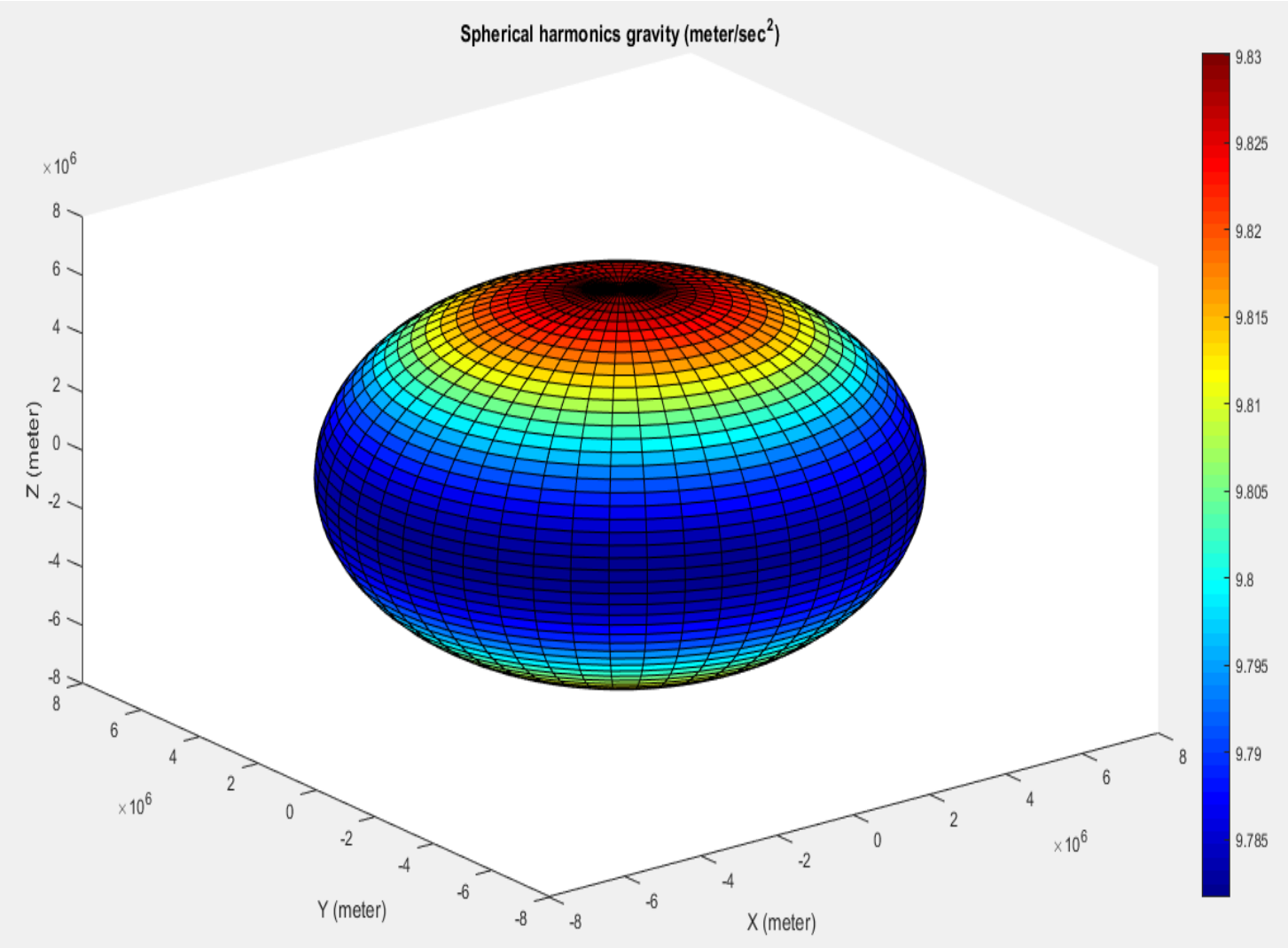
3.2.3 Validation

We can easily validate the spherical harmonies model by using the built in function in matlab (gravityspericalharmonic) which take the input of position and the return the vector of gravity.

And the error percent between our model and the matlab model is

Error in G_x	Error in G_y	Error in G_z
-0.0732 %	-0.0719 %	0.1437 %

a. Results



3.3 Calculating Eclipse Times [5]

3.3.1 Definitions

Generally we think of eclipses with the Sun and Moon. However, it's actually a common stellar phenomenon. We'll introduce the basic concept and focus on solar and lunar applications. Although lunar eclipses aren't of utmost interest in studying Earth-orbiting satellites, the concept of eclipses is useful in complex orbit determination systems, and for lunar observation. Satellites routinely enter eclipse periods during their orbits, and this can cause a great fluctuation in the available solar radiation for the solar arrays. This section presents the fundamentals of eclipse geometries and types, permitting analysis of solar and lunar eclipses.

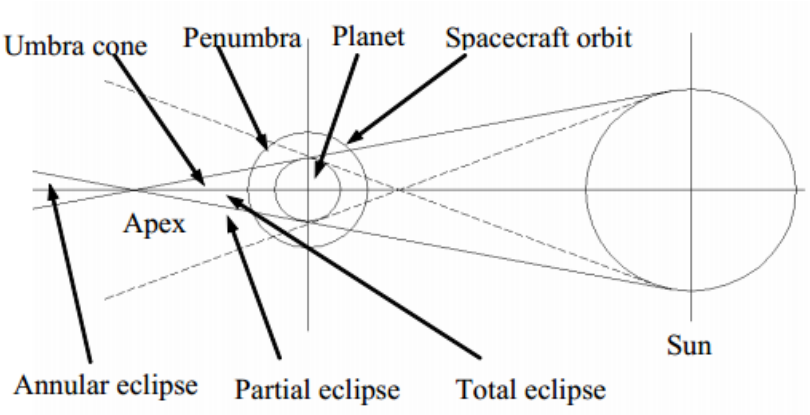


Figure 21 | Eclipse Geometry [5]

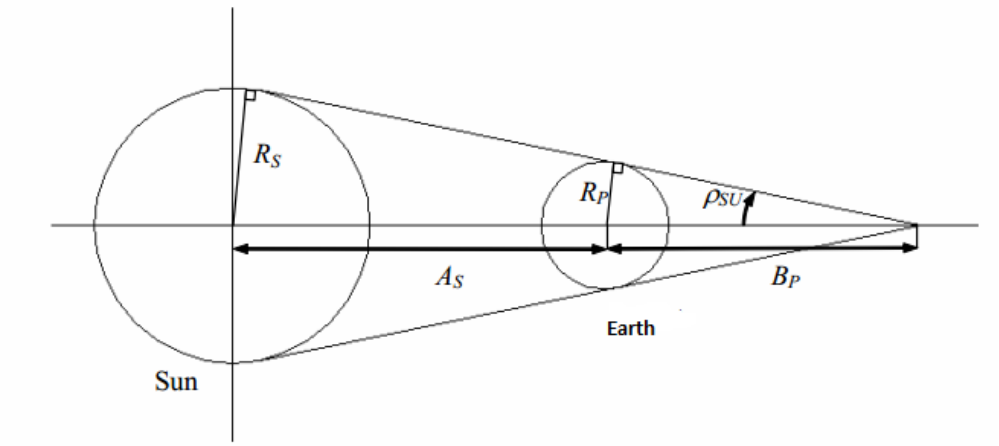


Figure 22 | Sun - Earth Relative Positions [5]

There are 3 types of eclipse will happen to satellite which are

1. Total Eclipse.
2. Partial Eclipse.
3. Annular Eclipse.

That described in Fig. (**21**) with their happening regions.

Fig. (**22** describes The position of Sun relative to earth, and satellite.

In which,

$$\sin(\rho_{SU}) = \frac{R_p}{B_P} = \frac{R_S}{A_S + B_P}$$

Where

R_S: is the Sun Radius ($R_S = 6.9599 \times 10^8$ (meter)).

R_P: is the Planet Radius (Earth) ($R_P = 6371 \times 10\text{^}3$ (meter)).

A_S: is the distance between the sun and the earth ($A_S = 1.49598 \times 10^{11}$)

B_P: is the distance between the earth and satellite (measured by satellite)

3.3.2 Classification of Eclipse

To describe the different eclipse conditions, let V_{SS} be the vector from the spacecraft to the sun, and V_{PS} be the vector from the planet (earth) to spacecraft to planet (earth).

From Figs. (**21**) and (**22**), we can deduce that during the sun-lit portion of the spacecraft orbit, the spacecraft will be in the space between the sun and the planet. If the spacecraft lies in the penumbra (partial eclipse) region, then:

$$\left||V_{SS}|\right| > A_S$$

And if the spacecraft lies in the umbra (Total Eclipse) region, then:

$$A_S < \left||V_{SS}|\right| < A_S + B_P$$

And finally, if the spacecraft lies in the annular eclipse portion, then:

$$\left||V_{SS}|\right| > A_S + B_P$$

The previously derived conditions aren't sufficient to assure various eclipse conditions as viewed by spacecraft. These conditions must be augmented with other set of conditions derived from other views of the scene.

Let θ_{PS} be the angular separation between the planet and the sun, ρ_S be the angular of the sun, and ρ_P be the angular radius of the planet, and all of these angular separation and radii are as viewed from the spacecraft. These angles are given by:

$$\theta_{PS} = \cos^{-1}\left(\frac{V_{SS} \cdot V_{PS}}{\left||V_{SS}|\right| \left||V_{PS}|\right|}\right)$$

$$\rho_S = \sin^{-1}\left(\frac{R_S}{\left||V_{SS}|\right|}\right)$$

$$\rho_P = \sin^{-1}\left(\frac{R_P}{\left||V_{PS}|\right|}\right)$$

Fig. (**23**) represents the condition of no eclipse occurance.

$$(\rho_P + \rho_S) < \theta_{PS}$$

When the sun and the planet become tangent as viewed from the spacecraft as in Fig. (**24**) then the eclipse phenomena is about to appear or disappear and we see that:

$$(\rho_P + \rho_S) = \theta_{PS}$$

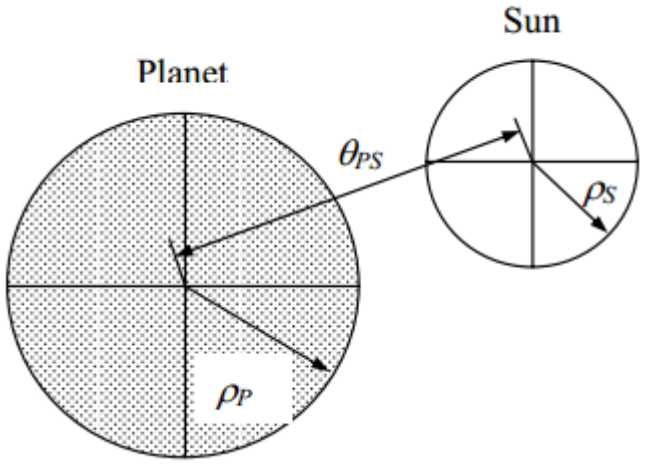


Figure 23 | No Eclipse Condition

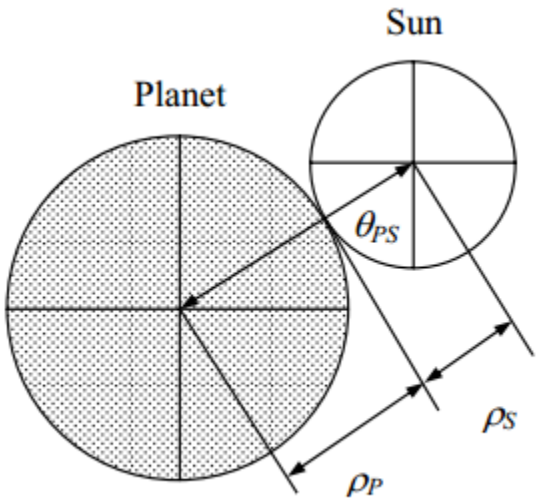


Figure 24 | The Boundary Condition of Eclipse phenomena

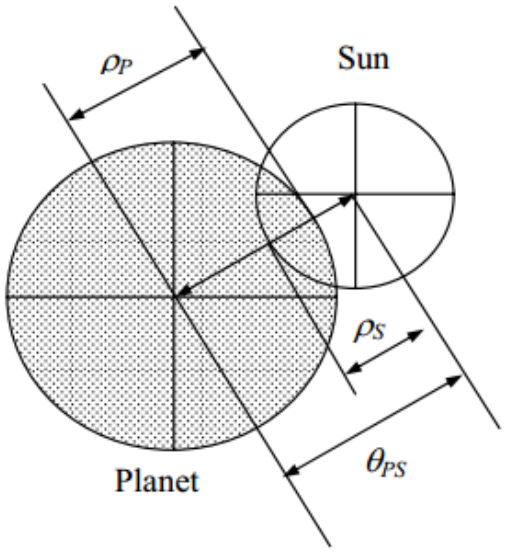


Figure 25 | The Partial Eclipse Phenomena

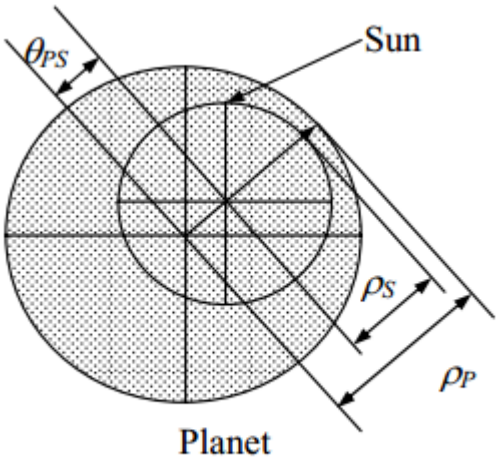


Figure 26 | The Total Eclipse Phenomena

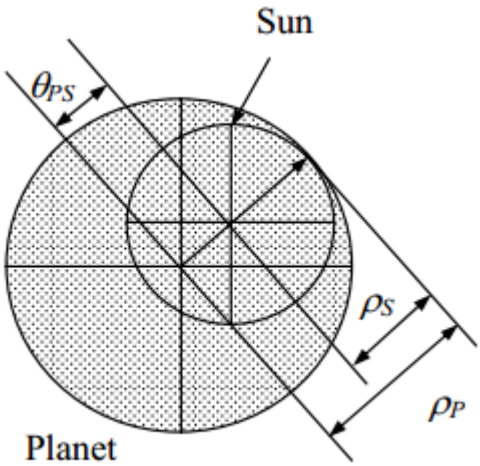


Figure 27 | The Boundary Condition for Eclipse Phenomena

When a part of the sun is blocked by a part of the planet (Earth) disk, as seen in Fig. (**25**) ,then this condition is known as partial eclipse. This condition is indicated by

$$(\rho_P + \rho_S) > \theta_{PS}$$

When the sun disc is compelelty blocked by the planet disc the total eclipse or simple eclipse phenomena occurs, as shown in Fig. (**26**) . The eclipse condition could be derived:

$$(\rho_P - \rho_S) > \theta_{PS}$$

Fig. (**27**) represents the situation at which a portion of the sun lighting disc is about to appear. This means that the spacecraft is in the eclipse region,and is also about to be in partial eclipse region. Mathematically this condtion could be expressed as:

$$(\rho_P - \rho_S) = \theta_{PS}$$

It is important that the last two conditions represent both the total and annular eclipse condition(In the annular eclipse region $\rho_S > \rho_P$ by definition). And to discriminate between the total and annular eclipse phenomena.

The summarized conditions among various eclipse cases

	Angular Separation Condition	Linear Separation Condition
Normal Sun Lighting	$(\rho_P + \rho_S) < \theta_{PS}$	$\left\|\|V_{SS}\|\right\| < A_S$
Partial Eclipse	$(\rho_P - \rho_S) > \theta_{PS}$ $\theta_{PS} < (\rho_P + \rho_S)$	$\left\|\|V_{SS}\|\right\| > A_S$
Total Eclipse	$0 < \theta_{PS} < (\rho_P - \rho_S)$	$A_S < \left\|\|V_{SS}\|\right\| < A_S + B_P$
Annular Eclipse	$0 < (\rho_P - \rho_S) < \theta_{PS}$	$\left\|\|V_{SS}\|\right\| > A_S + B_P$

3.4 Julian Date

An essential concept in astrodynamics is the Julian date. The Julian date, JD, is the interval of time measured in days from the epoch January I, 4713 B.C., 12:00. Joseph .Scaliger made the actual determination in 1582. He combined the solar cycle (28 years), the Metonic cycle (19 years), and the Roman Indication (15 years) to create a Julian period consisting of 7980 Julian years (365.25 days). Because the three cycles were already established, the only common point for these cycles was 4713 B.C.-hence, the epoch value. Interestingly enough, the Julian period was named for Scaliger's father and not Julius Caesar as the name might suggest. The convention to start the JD at noon each day benefits astronomers (who often work at night) because they can make all their observations on a single day. It's also the time that we could best use to reckon solar days. To find the Julian date from a known date and time within the period March I, 1900 to February 28,2100, we use Algorithm 14 (*use 61 seconds if the day contains a leap second). Notice that the year, month, day, hour, minute, and second are known. The TNT relation denotes real truncation. The year must be four digits and not the commonly abbreviated two-digit value. This empirical formula is designed to be accurate for the period described above. Its use is restricted because the JD is continuous (except for JDUTC) whereas the calendar contains periodic steps through the addition of leap years and seconds. Algorithm 14 is valid for any of the time system.

Unless specified, JD usually implies a time based on UT1. The mathematical expression for JD is as follows [3]:

$$JD = 367\,UT1_{year} - INT\left(\frac{7\left(UT1_{year} + INT\left(\frac{UT1_{month}+9}{12}\right)\right)}{4}\right) + INT\left(\frac{275\,UT1_{month}}{9}\right) + UT1_{day} + 1{,}721{,}013.5 + \frac{\left(\frac{UT1_{second}}{60} + UT1_{minute}\right)}{60} + UT1_{hour}$$

The modified Julian date (**MJD**):

$$MJD = JD - 2{,}400{,}000.5$$

For the general formula referencing J2000.0

$$T_{Ut1} = \frac{MJD}{36{,}525}$$

3.5 Sun Position Vector

We often want to find the position vector from the Earth to the Sun for uses such as determining solar-panel illumination and remote sensing. It's often convenient to place a mathematical algorithm inside the particular program using a less precise technique. Some algorithms require us to consider which equator and equinox are referenced (e.g., true-of-date, mean-of-date). We'l1 discuss a simple technique which its results are in a MOD (mean-equator of

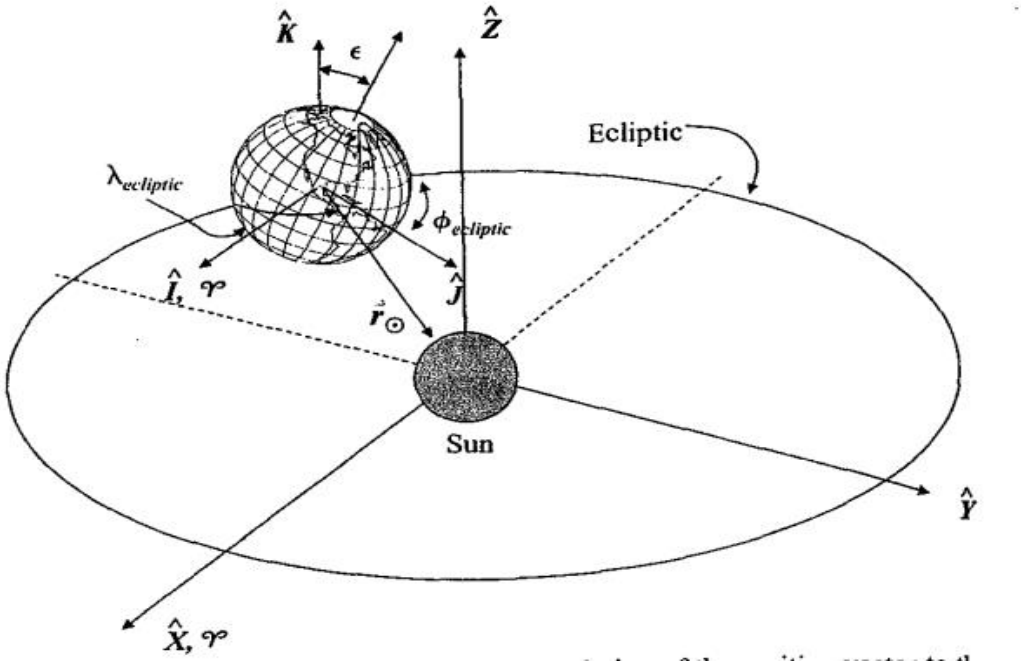


Figure 28 | Sun Vector between the Axis of Sun and Inertial Axis of Earth

date) vector with an accuracy of 0.01°. It's valid from 1950 to 2050 because of the truncation of the expansions. The solution rests in part on expressions using the J2000.0 epoch. [3]

First, we need to find the number of Julian centuries, T_{UT1}, from the epoch. Find the mean longitude of the Sun, $\lambda_{M_\odot}$in a MOD frame.

$$\lambda_{M_\odot} = 280.46\,^\circ + 36{,}000.771\; T_{UT1}$$

Then we need to find the mean anomaly for the sun, $M_\odot$.We show T_{TDB} to be precise, but it's acceptable to use T_{UT1} because this is only a low precision formula.

$$M_\odot = 357.5277233 + 35999.05034\; T_{UT1}$$

After that we need to reduce the both $\lambda_{M_\odot}$ and $M_\odot$ to the range of $0\,^\circ\; to\; 360°$.

Because the Earth's orbit is approximately circular, we can assume the true anomaly is close enough to the longitude for the following expression to hold (remember to convert to units of degrees). The ecliptic latitude of the Sun never exceeds 0.000 333° and is often set to 0.0°.

$$\lambda_{ecliptic} = \lambda_{M_\odot} + 1.914666471° \sin(M_\odot) + 0.019994643 \sin(2M_\odot)$$

$$\phi_{ecliptic} = 0°$$

We can approximate the obliquity of the ecliptic

$$\epsilon = 23.439291° - 0.0130042 T_{TDB}$$

And assuming that $T_{TDB} = T_{UT1}$

Then evaluation the position vector as following:

$$\overrightarrow{r_\odot} = \begin{bmatrix} r_\odot \cos(\lambda_{elliptic}) \\ r_\odot \cos(\epsilon)\sin(\lambda_{elliptic}) \\ r_\odot \sin(\epsilon)\sin(\lambda_{elliptic}) \end{bmatrix}$$

3.6 Sidereal Time

Sidereal time is a direct measure of the Earth's rotation and it's measured positively in the counter-clockwise direction when viewed from the North Pole. Ideally, the observation of a star would suffice for determining sidereal time. The changing instantaneous axis of rotation, causes station locations to continually change. This produces a small difference in the time of meridian transits, depending on the star's declination. Because this effect vanishes at the equator, it's better to use stars with small declinations. Remember we define the vernal equinox to be always on the equator. Thus, we define sidereal time as the hour angle of the vernal equinox relative to the local meridian. Because the vernal equinox is the reference point, the sidereal time associated with the Greenwich meridian is termed **Greenwich Mean Sidereal Time**, θ_{GMST} or GMST. The sidereal time at a particular longitude is called **Local Sidereal Time**, θ_{LST} or LST. In this context, time is an angle measured from the observer's longitude to the equinox. Occasionally, when we use stars, I'll introduce appropriate subscripts. Fig. (**29**) shows the relation between the GMST and LST. [3]

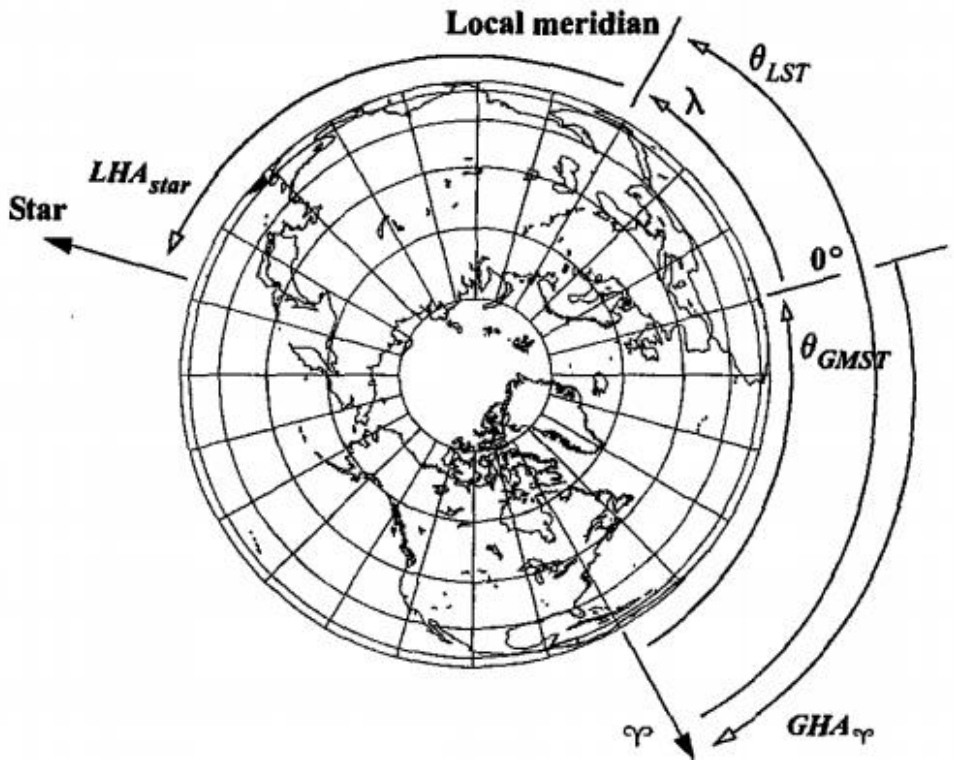


Figure 29 | Geometry for LST and GMST

Be careful when comparing GHA and GMST because hour angles are measured positive in a clockwise direction in a right-handed system, while sidereal time is measured positive in the counter-clockwise direction in that same system. Similar to LHA and GHA, LST and GMST may be expressed in degrees or hours and may have positive or negative values. Also, you may need to check quadrants to correct the angle.

We can convert between GMST and LST at a particular longitude λ, using:

$$\theta_{LST} = \theta_{GMST} + \lambda$$

This formula requires a convention for east and west longitudes.

The concept of sidereal time would be perfect except that the equinox moves very slowly due to precession, the apparent locations of the stars *do* change a little, and different longitude definitions cause small deviations in the exact longitudes of sites. Recall that the equinox results from the intersection of the Earth's equator and the ecliptic, and both planes are moving. This permits a distinction for mean and apparent time. Mean sidereal time, which is most commonly used, refers to a mean equinox that moves only with secular motion (precession). We measure **apparent sidereal time**, **AST**, from the true vernal equinox which includes secular and periodic contributions to the motion of the vernal equinox. The difference between the mean equinox and the apparent equinox [in the plane of the true equator] is the equation of the equinoxes. It gives us the difference between the mean and apparent sidereal times.

Simon Newcomb developed a mathematical expression for the fictitious mean Sun and he defined sidereal time as the hour angle of this fictitious mean Sun. A single expression references Greenwich, providing the **Greenwich mean sidereal time** at midnight, ${\theta_{GMST}}_0^h$, $(0^h 0^m 0^s)$ UTI. Expressions for it at a desired time in seconds and degrees are:

$${\theta_{GMST}}_0^h = 24{,}110.54841^s + 8{,}640{,}184.812866\, T_{UT1} + 0.093104\, T_{UT1}^2 - 6.2 \times 10^{-6} T_{UT1}^3$$

$${\theta_{GMST}}_0^h = 100.4606184^\circ + 36{,}000.770053\, T_{UT1} + 0.00038793\, T_{UT1}^2 - 2.6 \times 10^{-8}\, T_{UT1}^3$$

Where T_{UT1} is the number of Julian centuries elapsed from the epoch J2000.0 and is computed using the Julian day numbers for the date of interest (JD_0^h) and for the epoch J2000.0. Julian day numbers are simply the integer part of the Julian date (JD), i.e., the JD at ($0^h 0^m 0^s$) of the day. Sometimes the last term is ignored because it's required only for very precise calculations (on the order of 1×10^{-8} seconds). The UTI subscript designation is required to be consistent with the precise notation necessary for certain applications. To complete the process, we must add the elapsed UTI time on the day of interest:

$$\theta_{GMST} = \theta_{GMST}{}_0^h + \omega_\oplus \text{ UT1}$$

Where,

$\omega_\oplus$ is the Earth's mean angular rotation in degrees per second.

$UT1$ is the universal time seconds past 0^h.

The final equation of sidereal time which it is function of Julian Centuries is as follows [3]:

$$\boldsymbol{\theta_{GMST} = 67,310.54841^s + \left(876,600^h + 8,640,184.812866^s\right) T_{UT1} + 0.093104\, T_{UT1}^2 - 6.2 \times 10^{-6}\, T_{UT1}^3}$$

Then change this quantity to degree angles as it is calculated as time. As earth rotate about itself 24 hour (day time) =$24 \times 60 \times 60$, and 360 degree.

So,

$$\boldsymbol{\theta^\circ_{GMST} = \theta^s_{GMST} \times \frac{360}{24 \times 60 \times 60}}$$

3.7 Disturbances of Space Environment

To predict the attitude of the spacecraft, it is required to have a model of all the environmental disturbances torques acting on the spacecraft.

There are 4 dominant sources of attitude disturbances torques are Earth's Gravitational force, Earth's magnetic field, solar radiation pressure, and aerodynamic drag.

3.7.1 Gravity Gradient Torque

Any nonsymmetrical object of finite dimensions in orbit is subject to a gravitational torque because of the variation in the Earth's gravitational force over the object. This gravity-gradient torque results from the inverse square gravitational force field; there would be no gravitational torque in a uniform gravitational field. General expressions for the gravity-gradient torque on a satellite of arbitrary shape have been calculated for both spherical and non-spherical earth models. For most applications, it is sufficient to assume a spherical mass distribution for the Earth.

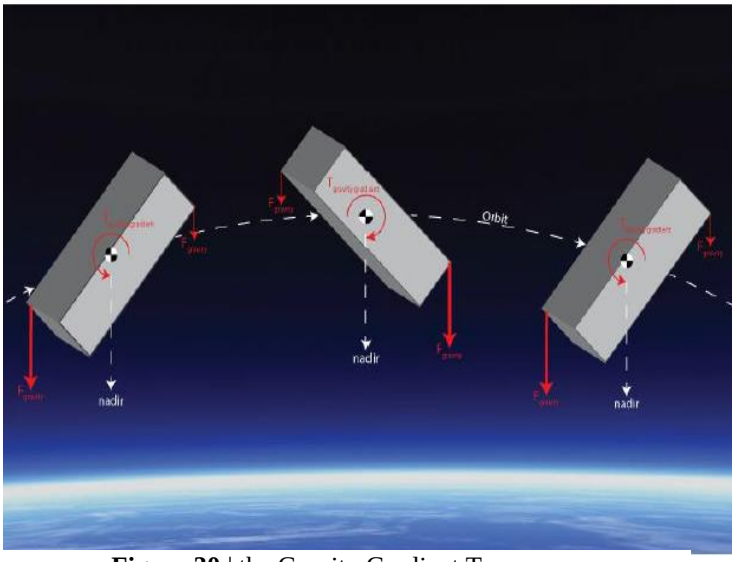


Figure 30 | the Gravity Gradient Torque

The Gravity Gradient Torque can be considered as Disturbance Torque or Stability Torque. It is stability torque if we want to head toward nadir point while it is disturbance moment if we want to point another position but not nadir.

We assume that the spacecraft's moment-of-inertia tensor is known for some arbitrary body reference frame whose origin need not coincide with the spacecraft's center of mass and that the spacecraft is orbiting a spherical Earth.

The gravitational force dF acting on a spacecraft mass element dm; located at a position R_i relative to the geocentric is

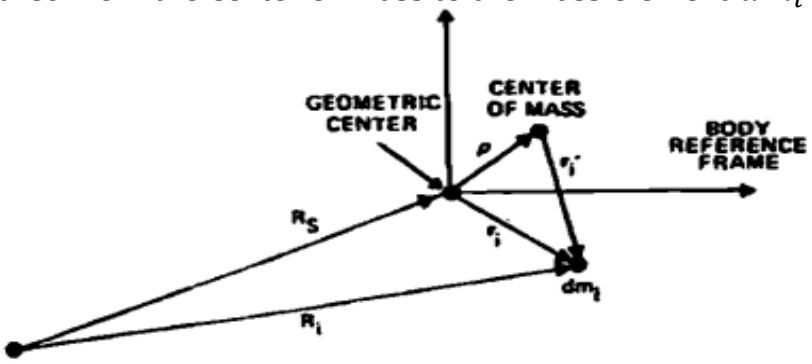
$$dF_i = \frac{-\mu\, R_i\, dm_i}{R_i^3}$$

Where $\mu \equiv GM_{\oplus}$ is the Earth Gravitational constant.

The Torque about the spacecraft's geometric center due to force, $\boldsymbol{dF_i}$ at position, $\boldsymbol{r_i}$, relative to the spacecraft's geometric center is

$$\boldsymbol{dN_i = r_i \times dF_i = (\rho + r') \times dF_i}$$

Where, ρ: is measured from the geometric center to the center of mass. r_i': is measured from the center of mass to the mass element dm_i.



The Gravity Gradient Torque on the entire spacecraft is obtained by integrating the previous equation

$$N_{GG} = \int r_i \times dF_i = \int (\rho + r_i') \times \frac{-\mu\, \boldsymbol{R_i}}{R_i^3} dm_i$$

The geocentric position vector can be expressed as following

$$\boldsymbol{R_i = R_S + r_i = R_S + \rho + r_i'}$$

For a practical artificial satellite $\boldsymbol{R_i = R_S + \rho + r_i' \gg \rho + r_i'}$ **, then**

$$\boldsymbol{R_i^{-3} \approx R_S^{-3}\left[1 - \frac{3R_S(\rho + r_i')}{R_s^2}\right]}$$

The gravity gradient can be written as

$$N_{GG} = \frac{\mu M}{R_S^2}\left(\widehat{\boldsymbol{R_S}} \times \rho\right) + \frac{3\mu}{R_S^3}\int \left(r_i \times \widehat{\boldsymbol{R_S}}\right)\left(r_i \cdot \widehat{\boldsymbol{R_S}}\right) dm_i$$

In which the final form of the **Gravity Gradient Torque** [4]

$$\boldsymbol{N_{GG}} = \frac{\boldsymbol{3\mu}}{\boldsymbol{R_s^3}}\left[\widehat{\boldsymbol{R_S}} \times \left(\boldsymbol{I} \cdot \widehat{\boldsymbol{R_S}}\right)\right]$$

Which can be also expressed as

$$\boldsymbol{G = 3\,\omega_0}(\boldsymbol{er} \times (\boldsymbol{I_{inertia}} * \boldsymbol{er})) \qquad (3)$$

Where **er** is the zenith vector which is equivalent to the third column (transformation from Greenwich to orbital) in the rotation matrix from matrix (1) which can represented (in our case of [2-1-3] order of rotation) by:

3.7.2 Magnetic Disturbance Torque

Magnetic disturbance torques result from the interaction between the spacecraft's residual magnetic field and the geomagnetic field. The primary sources of magnetic disturbance torques are:

1. Spacecraft magnetic moments.
2. Eddy currents.
3. Hysteresis.

The spacecraft's magnetic moment is usually the dominant source of disturbance torques. The spacecraft is usually designed of material selected to make disturbances from the other sources negligible.

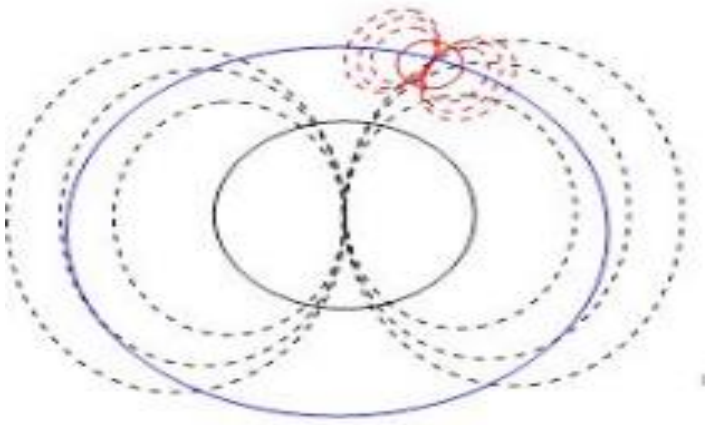
So, the instantaneous magnetic disturbance torque, N_{mag}, due to the spacecraft effective magnetic moment **m** is given by [4]:

$$\boldsymbol{N_{mag} = m \times B}$$

Where **B** is the geocentric magnetic flux density (in Watt / m^2) and **m** is the sum of the individual magnetic moments caused by permanent and induced magnetism and the spacecraft generated current loops.

The residual magnetic field current is measured after the satellite is built.

The value of the **m** (in our case) is [0.0006 , 0.0006 , 0.0006] in the body coordinate system.



3.7.3 Solar Radiation Torque

Radiation incident on a spacecraft's surface produces a force which results in a torque about the spacecraft's center of mass. The surface is subjected to radiation pressure or force per unit area equal to the vector difference between the incident and reflected momentum flux. Because the solar radiation varies as the inverse square of the distance from the Sun, the solar radiation pressure is essentially altitude independent for spacecraft in Earth orbit. The major factors determining the radiation torque on a spacecraft are:

1. The intensity and spectral distribution of the incident radiation.
2. The geometry of the surface and its optical properties.
3. The orientation of the Sun vector relative to the spacecraft.

The major sources of electromagnetic radiation pressure are:

1. Solar illumination.
2. Solar radiation reflected by the Earth and its atmosphere.
3. Radiation emitted from the Earth and its atmosphere.

Of these sources, direct solar radiation is the dominant source and is generally the only one considered. The force produced by the solar wind is also normally negligible relative to the solar radiation pressure. [4]

The mean momentum flux, P, acting on the surface normal to the sun's radiation, is given by:

$$P = \frac{F_e}{C}$$

That F_e is the solar Constant which equal to 1367 (W/m), and C is the speed of the light which equal to 3×10^8 (m/sec).

For most applications, the forces may be modeled adequately by assuming that incident radiation is either absorbed, specularly reflected, diffusely reflected, or some combination of these as shown in Fig. (**31**). Let P be the momentum flux incident on an elemental area **dA** with unit outward normal $\widehat{N}$

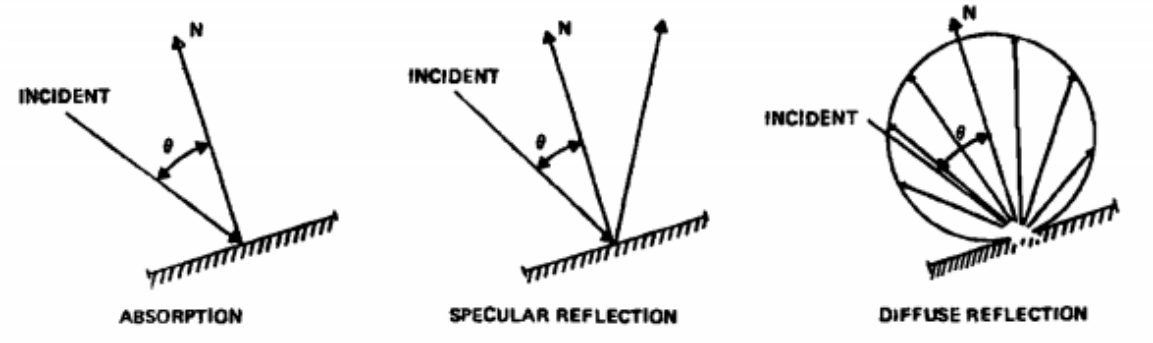


Figure 31 | Absorption and Reflection of Incident Radiation

The differential radiation. force (momentum transferred per unit time) due to that portion of the radiation that is completely absorbed is:

$$\boldsymbol{df_{absorbed} = -P\,C_a \cos(\theta)\,\hat{S}\,dA} \qquad \boldsymbol{(0 < \theta < 90^o)}$$

Where $\hat{\boldsymbol{S}}$ is the unit vector from the spacecraft to the Sun,, θ is the angle between $\hat{\boldsymbol{S}}$ and $\hat{N}$, and C_a is the absorption coefficient. If $\cos(\theta)$ is negative, the surface is not illuminated and will not experience any solar force.

The differential radiation force due to that portion of the radiation which is specularly reflected is:

$$\boldsymbol{df_{specular} = -2\,P\,C_s \cos^2(\theta)\,\hat{N}\,dA} \qquad \boldsymbol{(0 < \theta < 90^o)}$$

Where the reflected radiation is in the direction $\boldsymbol{\left(-\hat{S} + 2\,\hat{N}\,\cos(\theta)\right)}$. the coefficient of specular reflection, $\boldsymbol{C_s}$, is the fraction of the incident radiation that is specularly reflected. For a diffuse surface, the reflected radiation is distributed over all directions with a distribution proportional to $\cos(\phi)$, where ϕ is the angle between the reflected radiation and $\hat{\boldsymbol{N}}$.

The differential radiation force for diffusely reflected radiation is determined by integrating the contribution of the reflected radiation over all angles to obtain:

$$\boldsymbol{df_{diffuse} = P\,C_d\left(-\frac{2}{3}cos(\theta)\hat{N} - cos(\theta)\hat{S}\right)dA} \qquad \boldsymbol{(0 < \theta < 90^o)}$$

Where the coefficient of diffuse reflection,C_d is the fraction of the incident radiation that is diffusely reflected.

Assuming that absorption, specular reflection, and diffuse reflection all play a part (without any transmission), then the total differential radiation force is [4]:

$$\boldsymbol{df_{total} = -P\int \left[(1 - C_s)\widehat{S} + 2(C_s cos(\theta) + \frac{1}{3}C_d)\widehat{N}\right] cos(\theta)\, dA}$$

Where $C_a + C_s + C_d = 1$. For surfaces that are not completely opaque, the incident momentum flux, P, can be modified to account for the radiation that does not impinge or interact with the surface. The differential radiation force can be written to include secondary reflections, but this is normally not a significant factor in the total radiation force.

The Solar radiation torque $\boldsymbol{N_{solar}}$, acting on a spacecraft is given by the general expression:

$$\boldsymbol{N_{solar} = \int R \times df_{total}} \quad (4)$$

Where **R** is the vector from the spacecraft's center of mass to the elemental area **dA**. The integral is over the spacecraft's irradiated surface. Because of the difficulty in evaluating the radiated surface because of the difficulty in evaluating the radiation torque directly from Eq. (4) for arbitrary surfaces, the spacecraft configuration is frequently approximated by a collection of simple geometrical elements (e.g., plane, cylinder, sphere).

The solar radiation force, F_i on each element is determined by evaluating the integral of $\boldsymbol{df_{total}}$ over the exposed surface area, that is:

$$\boldsymbol{F_i = \int df_{total_i}}$$

Which can be simplified to

$$\boldsymbol{N_{solar} = \sum_{i=1}^{n} R_i \times F_i}$$

Where,

R_i: The direction from the satellite's center of mass to the surface element center of geometry.

3.7.4 Aerodynamics Disturbance

The interaction of the upper atmosphere with a satellite's surface produces a torque about the center of mass. For spacecraft below approximately 400 km, the aerodynamic torque is the dominant environmental disturbance torque.

The force due to the impact of atmospheric molecules on the spacecraft surface can be modeled as an elastic impact without reflection. The incident particle's energy is generally completely absorbed. The particle escapes after reaching thermal equilibrium with the surface with a thermal velocity equal to that of the surface molecules. Because this velocity is substantially less than that of the incident molecules, the impact can be modeled as if the incident particles lose their entire energy on collision. [4]

The force $\boldsymbol{F_{Aero}}$ on a surface area $\boldsymbol{A}$, with outward normal $\boldsymbol{\hat{N}}$, is given by:

$$\boldsymbol{F_{Aero} = -\frac{1}{2}\, C_D\, \rho\, V^2 \left(\hat{N} \cdot \hat{V}\right) \hat{V}\, A}$$

Where
$\boldsymbol{\rho}$: Atmospheric density calculated with different density models.

$\boldsymbol{V}$: Euclidean norm of orbital velocity vector.

$\boldsymbol{C_D}$: Drag coefficient.

$\boldsymbol{\hat{N}}$: Outward normal unit vector for each satellite surface element.

$\boldsymbol{\hat{V}}$: Unit vector in the direction of the translational velocity, V.

The aerodynamic disturbance moments are the effect of drag forces caused by satellite velocity, called orbital velocity, multiplied by the distance between point of force and satellite center of mass. It is given by:

$$\sum R_C(i) \times F(i)$$

WhereR_C:T
he direction from the satellite’s center of mass to the surface element center of geometry.

Chapter 4

Dynamics and Kinematics Model

4.1 Derivation of dynamics and kinematics model

4.1.1 Dynamics equation derivation

The dynamic equation of motion can be derived through applying Newton's second law for rotational motion, where a moment, acting on a body about its center of mass, equals the time rate of change of its angular momentum. In this section we will examine the rotational motion of a body caused by the applied moment.

$$\dot{H}^I = M_{ext}$$

This is the well-known Euler's moment equation. In this equation, the Subscript "I" indicates a derivative in the inertial frame, while the subscript "B" indicates a derivative in the rotating body frame.

Recalling that the derivative of the angular momentum is the external torques and applying the attitude matrix rotations to the last eq., yields

$$M_{ext} = \dot{H}^B + \omega \times H^B$$

$$\dot{H}^B = -\phi(\omega)H^B + M_{ext}$$

Where M_{ext} is the sum of external torques acting on the satellite. Such as controlling torques M_c, gravity gradient M_{gg} and the other external disturbance M_d (i.e. aero drag, magnetic disturbance….)

$$M_{ext} = M_{gg} + M_c + M_d$$

Where H^B is angular momentum in Body coordinate system .

$$\dot{\omega}_{bi} = J^{-1}(M_d + M_c + M_{gg} - \omega_{bi} \times J\,\omega_{bi})$$

4.1.2 Kinematics equation derivation

Let the quaternion q represents orientation of the rigid body with respect to the reference system at time t, and q'' represents the orientation with respect to the reference system at time $t + \Delta t$. We shall denote these by $q(t)$ and $q(t + \Delta t)$, respectively. Then q' specifies the orientation of the BCS triad $(\hat{u}, \hat{v}, \hat{w})$ at time $t + \Delta t$ relative to the position occupied at time t.

$$q_1' = e_u \sin\left(\frac{\Delta\phi}{2}\right)$$
$$q_2' = e_v \sin\left(\frac{\Delta\phi}{2}\right)$$
$$q_3' = e_w \sin\left(\frac{\Delta\phi}{2}\right)$$
$$q_o' = \cos\left(\frac{\Delta\phi}{2}\right)$$

Where e_u, e_v, e_w are the components of the rotation axis unit vector along the triad $(\hat{u}, \hat{v}, \hat{w})$ at time t (because this is the reference system for q') and $\Delta\phi$ is the rotation in time Δt.Thus, the propagation of the attitude from t to $t + \Delta t$ can be written as:

$$\boldsymbol{q(t + \Delta t) = q(t) \circ q'} \qquad \textbf{(5)}$$

Where:

$$q(t) = \begin{bmatrix} q_o \\ q_1 \\ q_2 \\ q_3 \end{bmatrix}$$

For the case of general attitude motion, it is convenient to convert the above equation to differential equation (function of time). In this case, Δt is infinitesimal and $\Delta\phi = \omega\Delta t$, where ω is the magnitude of the instantaneous angular velocity of the rigid body. We use small angle approximations [4]:

$$\cos\left(\tfrac{\Delta\phi}{2}\right) \approx 1 \qquad , \qquad sin\left(\tfrac{\Delta\phi}{2}\right) \approx \tfrac{\Delta\phi}{2} \approx \tfrac{|\omega_{BO}|\Delta t}{2}$$

Thus,

$$q' = \begin{bmatrix} 1 \\ \frac{e_u|\omega_{BO}|\Delta t}{2} \\ \frac{e_v|\omega_{BO}|\Delta t}{2} \\ \frac{e_w|\omega_{BO}|\Delta t}{2} \end{bmatrix}$$

Then the equation (**6**) can be represented as:

$$q(t+\Delta t) = q(t) \circ \begin{bmatrix} 1 \\ \frac{e_u|\omega_{BO}|\Delta t}{2} \\ \frac{e_v|\omega_{BO}|\Delta t}{2} \\ \frac{e_w|\omega_{BO}|\Delta t}{2} \end{bmatrix}$$

$$= q_o - \frac{\Delta t}{2}\boldsymbol{q}.{\omega_{BO}}^T + \frac{\Delta t}{2} q_o {\omega_{BO}}^T + \boldsymbol{q} + \frac{\Delta t}{2}\boldsymbol{q} \times \omega_{BO}$$

Where:

$\boldsymbol{q}$**:** is the vector part of the quaternion $q(t)$

The differential equation of the $q(t)$ is then defined as:

$$\dot{q} = \lim_{\Delta t \to 0} \frac{q(t+\Delta t) - q(t)}{\Delta t}$$

Substituting by $q(t+\Delta t)\ and\ q(t)$ and taking the limit yields to the known kinematic differential equation:

$$q^{\dot{BO}} = \frac{1}{2}(-\boldsymbol{q}.{\omega_{BO}}^T + q_o\omega_{BO} + \boldsymbol{q} \times \omega_{BO}) = \frac{1}{2} q^{BO} \circ \Omega_{\mathrm{BO}} \qquad (7)$$

Where:
Ω_{BO} : the angular velocity of the BCS relative to the OCS in the quaternion form and can be written as:

$$\Omega_{BO} = \begin{bmatrix} 0 \\ \omega_{BO_x} \\ \omega_{BO_y} \\ \omega_{BO_z} \end{bmatrix}$$

The above kinematics differential equation is represented using relative angular velocity between satellite (BCS) and orbital frame of reference (OCS). However, the dynamic model of satellite is calculated using absolute angular velocity of the satellite ω_{BI}, so it is more convenient to introduce the satellite kinematics through absolute angular. This can be performed using the following formula:

$$\omega_{BO} = \omega_{\mathrm{BI}} - \omega_o en$$

Where:
en: is the second column of the rotation matrix from OCS to BCS.

Substituting the above equation in the kinematic equation leads to:

$$\frac{1}{2} q^{BO} \circ \Omega_{\mathrm{BO}} = \frac{1}{2} q^{BO} \circ \begin{bmatrix} 0 \\ \omega_{BO} \end{bmatrix} = \frac{1}{2} q^{BO} \circ \begin{bmatrix} 0 \\ \omega_{\mathrm{BI}} - \omega_o en \end{bmatrix}$$

$$\dot{q^{BO}} = 0.5 * (q^{BO} \circ \Omega_{BI} - \Omega_o \circ q^{BO})$$

$$\boldsymbol{\dot{q^{BO}} = 0.5 * \left(q^{BO} \circ \begin{bmatrix} 0 \\ \omega_{BI_x} \\ \omega_{BI_y} \\ \omega_{BI_z} \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ \omega_o \\ 0 \end{bmatrix} \circ q^{BO} \right)} \qquad \textbf{(8)}$$

Chapter 5

Attitude Estimation

5.1 Overview

The control algorithm depends on the system states (i.e. quaternion and angular velocity) to compute a suitable control action. Sometimes, some of these states could not be measured due to prohibiting obstacles, or may be measured with noise. In these cases, the measured states should not be used directly to determine the control action. Alternatively, an estimation algorithm is used to calculate more accurate states which are then used by the controller.

The purpose of the estimation algorithm is to determine a state vector estimate $\hat{x}_k$ with its associated covariance $\hat{P}_k$ .Therefore, this estimate must satisfy certain optimality condition. In the least squares sense, it must minimize the quantity $E\{(\hat{x}_k - x_k)^T(\hat{x}_k - x_k)\}$, where E is the expectation and x_k is the real states. In this study, we consider the Extended Kalman Filter (EKF) as the estimation algorithm and is to be discussed in this section.

Kalman filtering, also known as linear quadratic estimation (LQE), is an algorithm that uses a series of measurements observed over time, containing statistical noise and other inaccuracies, and produces estimates of unknown variables that tend to be more precise than those based on a single measurement alone, by using Bayesian inference and estimating a joint probability distribution over the variables for each timeframe. The filter is named after Rudolf E. Kalman, one of the primary developers of its theory. It is simply a set of mathematical equations that provides a recursive means of estimating the state of a process while minimizing any error in the system. The extended Kalman filter (EKF) is the nonlinear version of the Kalman filter which linearizes about an estimate of the current mean and covariance. In the case of well-defined transition models, the EKF has been considered. [5] [6]

5.2 Kalman filter algorithms

5.2.1 Basic KF

Assuming that a random process to be estimated can be modeled in the form:

$$x_{k+1} = \phi_k x_k + w_k \quad (9)$$

The observation (measurement) of the process is assumed to occur at discrete points in time in accordance with the linear relationship:

$$z_k = H_k x_k + v_k \quad (10)$$

Where:

- x_k= (n×1) process state vector at time t_k.
- ϕ_k=(n×n) matrix relating x_k to x_{k+1} in the absence of a forcing function (if x_k is a sample of continuous process, ϕ_k is the usual state transition matrix).
- w_k= (n×1) vector assumed to be a white sequence with known covariance structure (process noise).
- z_k= (m×1) vector measurement at time t_k.
- H_k= (m×n) matrix giving the ideal (noiseless) connection between the measurement and the state vector at time t_k.
- v_k= (m×1) measurement error assumed to be a white sequence with known covariance structure and having zero cross-correlation with the w_k sequence.

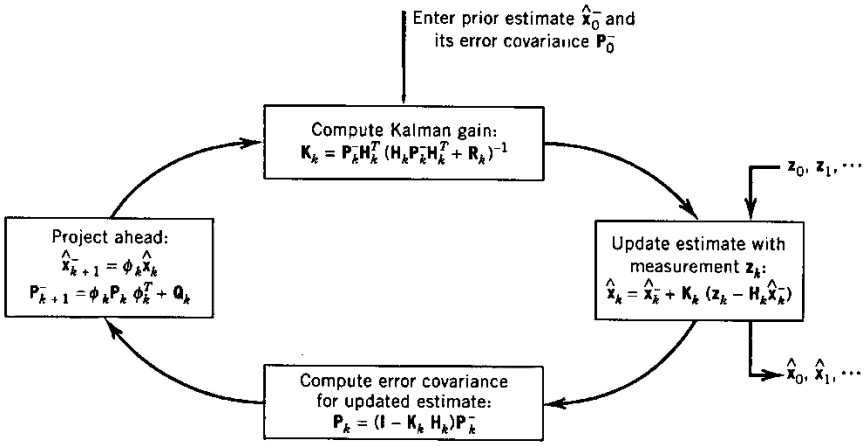


Figure 32| Kalman Filter loop

The above figure shows the algorithm of the basic linear KF mentioned in equations (11) and (12). The EKF algorithm is to be discussed next section [7].

5.2.2 EKF

In the extended Kalman filter, the state transition and observation models don't need to be linear functions of the state but it may instead be differentiable functions.

$$\hat{x}_{k+1} = f(x_k, u_k) + w_k$$
$$\hat{z}_k = h(x_k) + v_k$$

Where w_k and v_k are the process and observation noises which are both assumed to be zero mean multivariate Gaussian noises with covariance Q_k and R_k respectively. u_k is the control vector.

The function f can be used to compute the predicted state from the previous estimate and similarly the function h can be used to compute the predicted measurement from the predicted state. However, f and h cannot be applied to the covariance directly. Instead a matrix of partial derivatives (the Jacobian) is computed.

The basic equation of the extended Kalman filter are:

- Initialization
  → Initial state covariance matrix P_o^-
  → Initial predicted states $\hat{x}_o^-$

- Correction

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1}$$
$$\hat{x}_k^+ = \hat{x}_k^- + K_k(z_k - \hat{z}_k)$$
$$P_k^+ = (I - K_k H_k) P_k^- (I - K_k H_k)^T + K_k R_k K_k^T$$

- Prediction

$$\hat{x}_{k+1}^- = f(\hat{x}_k^-, u_k)$$
$$P_{k+1}^- = F_k P_k^+ F_k^T + Q_k$$

Where:

$f(.)$: Is **a nonlinear function of the state vector** x.

F_k: Is **the state transition matrix computed from:**

$$F = (I + A\delta t) \text{ and } A = \frac{\partial f}{\partial x}|_{x=x_{k-1}^{+}}$$

5.2.3 Alternative EKF based on the error model

The dynamic model is basically nonlinear as illustrated in the previous discussion. The technique of nonlinear problems as dynamics and kinematics equations of satellite in space faced with difficulty of valid linear approach for all regions of motion, so a linearized technique is used to be applied to the (EKF) algorithm to avoid the nonlinear dynamic model complexity. This linearized model is based on error between actual states and estimated states instead of the real states and can be valid for great band of maneuvers as it is not based on the small angle approximation which is not valid for satellite modes of operation. In the following line, the derivation of the error model is introduced:

The new system model can be written in the standard formula of the linear time invariant system:

$$\dot{x} = Fx + Gu$$
$$y = Cx + Du$$

Where:
x: State vector
y: Output vector
F: State matrix (relating the states and its derivatives)
G: Input matrix (relating the states and the inputs)
C: Output matrix (relating the outputs and the states)
D: Direct transmission matrix (relating outputs and inputs)

The new states depend on the error difference between the existing states and estimated states.

$$M = \tilde{q} \circ q^{-}$$
$$\Delta\omega = \omega_E^{-} - \omega_E$$

Where:
M= quaternion represents the error between estimated and measured quaternions and can be written as:

$M = \begin{pmatrix} \vec{\mu} \\ \mu_o \end{pmatrix}$ Where μ_o and $\vec{\mu}$ are the scalar and vector part successively.

$\Delta\omega$= vector represents the error between estimated and measured angular velocities.
q^- = estimated quaternion.
$\tilde{q}$ = measured quaternion conjugate
ω_E^-= Estimated angular velocity (BI)
ω_E= Measured angular velocity (BI)

And Δx which is the new state vector is to be:

$$\Delta x = \begin{pmatrix} \vec{\mu} \\ \Delta\omega \end{pmatrix}$$

a. Dynamics Equations

Dynamics equations can be represented as:

$$\dot{\omega}_E = -J^{-1}\big(\omega_E \times (J\,\omega_E + H_{wo}) - 3\omega_o^2(er \times J * er)\big) + M_{control} \qquad (13)$$

Inertia tensor can be approximated in principal axes in stable region for gravity gradient configuration:
$J_y > J_x > J_z$ and $J_{xy} = J_{xz} = J_{yz} = 0$
H_{wo}=0 as there is no reaction wheel.

$$J = \begin{bmatrix} J_x & 0 & 0 \\ 0 & J_y & 0 \\ 0 & 0 & J_z \end{bmatrix} \text{ and } J^{-1} = \begin{bmatrix} \frac{1}{J_x} & 0 & 0 \\ 0 & \frac{1}{J_y} & 0 \\ 0 & 0 & \frac{1}{J_z} \end{bmatrix}$$

$$J^{-1}\big(\omega_E \times (J\,\omega_E)\big) = \begin{bmatrix} \beta_x\omega_2\omega_3 \\ \beta_y\omega_1\omega_3 \\ \beta_z\omega_1\omega_2 \end{bmatrix} = 0.5 \begin{bmatrix} 0 & \beta_x\omega_3 & \beta_x\omega_2 \\ \beta_y\omega_3 & 0 & \beta_y\omega_1 \\ \beta_z\omega_2 & \beta_z\omega_1 & 0 \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix}$$

Where :

$$\beta = \begin{bmatrix} \beta_x \\ \beta_y \\ \beta_z \end{bmatrix} = \begin{bmatrix} \frac{J_z - J_y}{J_x} \\ \frac{J_x - J_z}{J_y} \\ \frac{J_y - J_x}{J_z} \end{bmatrix}$$

Hence,

$$J^{-1}(\omega_E \times J\omega_E) = -0.5G(\beta, \omega_E)\omega_E$$

Where:

$$G(x,y) = \begin{bmatrix} 0 & x_1y_3 & x_1y_2 \\ x_2y_3 & 0 & x_2y_1 \\ x_3y_2 & x_3y_1 & 0 \end{bmatrix}$$

Similarly for $J^{-1}(er \times J\, er) = -0.5\, G(\beta, er)er$

The previous derivation leads to equation (14) in the following form:

$$\dot{\omega}_E = 0.5(G(\beta, \omega_E)\omega_E - 3\omega_o^2 G(\beta, er)er) + M_{control}$$

Similarly:

$$\dot{\omega}_E^- = 0.5(G(\beta, \omega_E^-)\omega_E^- - 3\omega_o^2 G(\beta, er^-)er^-) + M_{control}^-$$

Hence:

$$\begin{aligned} \Delta\dot{\omega} &= \dot{\omega}_E^- - \dot{\omega}_E \\ &= 0.5(G(\beta, \omega_E^-)\omega_E^- - 3\omega_o^2 G(\beta, er^-)er^-) \\ &\quad - 0.5(G(\beta, \omega_E)\omega_E - 3\omega_o^2 G(\beta, er)er) + \Delta m \end{aligned}$$

as:

$$\omega_E = \omega_E^- - \Delta\omega$$

$$\begin{aligned} &0.5(G(\beta, \omega_E^-)\omega_E^- - G(\beta, \omega_E)\omega_E) \\ &\quad = 0.5\big(G(\beta, \omega_E^-)\omega_E^- - G(\beta, \omega_E^- - \Delta\omega)(\omega_E^- - \Delta\omega)\big) \\ &\quad = 0.5(G(\beta, \omega_E^-)\omega_E^- - G(\beta, \omega_E^-)\omega_E^- + G(\beta, \Delta\omega)\omega_E^- \\ &\quad + G(\beta, \omega_E^-)\Delta\omega - G(\beta, \Delta\omega)\Delta\omega) \end{aligned}$$

Also as:

$$G(\beta, \Delta\omega)\omega_E^- = G(\beta\,, \omega_E^-)\Delta\omega$$

$$\begin{aligned} &0.5(G(\beta, \omega_E^-)\omega_E^- - G(\beta, \omega_E)\omega_E) \\ &\quad = G(\beta, \omega_E^-)\Delta\omega - 0.5G(\beta, \Delta\omega)\Delta\omega \\ &\quad = G(\beta, \omega_E^- - 0.5\Delta\omega)\Delta\omega \end{aligned}$$

Similarly:

$$0.5(G(\beta, er^-)er^- - G(\beta, er)er) = \mathrm{G}(\beta, er^- - 0.5\Delta er^-)\Delta er^-$$

Where:

$$\Delta er^- = er^- - er = er^- - \tilde{q} \circ J_3 \circ q = er^- - M \circ \tilde{q}^- \circ J_3 \circ q^- \circ \widetilde{M}$$
$$= er^- - M \circ er^- \circ \widetilde{M}$$

Where:

$$M \circ er^- \circ \widetilde{M} = er^- - 2\mu_o er^- \times \mu - 2\mu \times er^- \times \mu$$
$$\Delta er^- = 2\mu_o er^- \times \mu + 2\mu \times er^- \times \mu$$

Linearization about operating point of error states $\Delta x = 0$

$$\Delta\dot{\omega} = (\mathrm{G}(\beta, \omega_{\mathrm{E}}^- - 0.5\Delta\omega)\Delta\omega - 6\omega_o^2\, \mathrm{G}(\beta, er^- - 0.5\Delta er^-)\Delta er^- + \Delta m$$

$$\boldsymbol{\Delta\dot{\omega} = \mathbf{G}(\beta, \omega_{\mathbf{E}}^-)\Delta\omega - \mathbf{G}(0.5\Delta\omega)\Delta\omega - 6\omega_o^2 G(\beta, er^-)\Delta er^- + 6\omega_o^2 G(\beta, \Delta er^-)\Delta er^- + \Delta m} \quad \textbf{(15)}$$

The linearization technique is calculated by using Jacobian formula of:

$$F(x, y) = \frac{\partial F}{\partial x}\Delta x + \frac{\partial F}{\partial y}\Delta y$$

Which can be derived for dynamics equations with new states to:

$$\Delta\dot{\omega} = \frac{\partial\Delta\dot{\omega}}{\partial\Delta\omega}\Delta\omega + \frac{\partial\Delta\dot{\omega}}{\partial\vec{\mu}}\vec{\mu} + \frac{\partial\Delta\dot{\omega}}{\partial\Delta m}\Delta\mathrm{m}$$

The Δm term can be considered as the input, so we end up with the first two terms only in the error-states dynamics equation.

$$F_{\omega\omega} = \left|\frac{\partial\Delta\dot{\omega}}{\partial\Delta\omega}\right|_{\Delta x=0} = \mathrm{G}(\beta, \omega_{\mathrm{E}}^-) - \mathrm{G}(\beta, 0.5\Delta\omega) - \frac{\partial\mathrm{G}(\beta, 0.5\Delta\omega)}{\partial\Delta\omega}\Delta\omega$$

For

$\Delta x = 0 \rightarrow \Delta\omega = 0 \rightarrow G(\beta, 0.5\Delta\omega) = 0$ and this leads to:

$$\boldsymbol{F_{\omega\omega} = G(\beta, \omega_E^-)} \quad \textbf{(16)}$$

$$F_{\omega\mu} = \left|\frac{\partial \Delta\dot{\omega}}{\partial \vec{\mu}}\right|_{\Delta x=0} = \frac{\partial}{\partial \vec{\mu}}(-6\omega_o^2 G(\beta, er^-)\Delta er^- + 6\omega_o^2 G(\beta, \Delta er^-)\Delta er^-)$$

$$= \frac{\partial}{\partial \vec{\mu}}\left(6\omega_o^2 G(\beta, \Delta er^- - er^-)\right)$$

$$= -3\omega_o^2 \left| G(\beta, er^- - 0.5\Delta er^-) * \frac{\partial \Delta er^-}{\partial \vec{\mu}} - \frac{\partial G(\beta, er^- - 0.5\Delta er^-)}{\partial \vec{\mu}} \Delta er^- \right|_{\Delta x=0}$$

Where

$$|\Delta er^-|_{\vec{\mu}=0} = \left| 2\mu_o \phi(er^-) + \frac{\partial(2\phi(\mu).er^-)}{\partial \vec{\mu}} \times \mu + (2\phi(\mu).er^-) \times 1 \right|_{\vec{\mu}=0} = 2\phi(er^-)$$

$$\boldsymbol{F_{\omega\mu} = -6\omega_o^2 G(\beta, er^-)\phi(er^-)} \quad \textbf{(17)}$$

b. Kinematics Equations

From the equation of quaternion error $M = \tilde{q} \circ q^-$, the kinematics equation with error states can be derived by as following:

$$\frac{dM}{dt} = \frac{d\tilde{q}}{dt} \circ q^- + \tilde{q} \circ \frac{dq^-}{dt}$$

$$\dot{M} = 0.5 * (\tilde{q} \circ \omega_E - \omega_0 \circ \tilde{q}) \circ q^- + 0.5\tilde{q} \circ (q^- \omega_E^- - \omega_0 \circ q^-)$$

$$\dot{M} = 0.5 * (\tilde{q} \circ \omega_E \circ q^- - \omega_0 \circ \tilde{q} \circ q^- + \tilde{q} \circ q^- \circ \omega_E^- - \tilde{q} \circ \omega_0 \circ q^-)$$

But:

$$\tilde{q} \circ \omega_E \circ q^- = -\omega_E \circ \tilde{q} \circ q^-$$

$$\omega_0 \circ \tilde{q} \circ q^- = -\tilde{q} \circ \omega_0 \circ q^-$$

So:

$$\dot{M} = 0.5 * (-\omega_E \circ \tilde{q} \circ q^- + \tilde{q} \circ q^- \circ \omega_E^-)$$
$$\dot{M} = 0.5 * (-\omega_E \circ M + M \circ \omega_E^-)$$

Where:

$$\omega_E = \omega_E^- - \Delta\omega$$
$$\dot{M} = 0.5 * (-(\omega_E^- - \Delta\omega) \circ M + M \circ \omega_E^-)$$
$$\dot{M} = 0.5 * (M \circ \omega_E^- - \omega_E^- \circ M + \Delta\omega \circ M)$$

Where:

$$M \circ \omega_E^- = \begin{bmatrix}\mu_1\\ \mu_2\\ \mu_3\\ \mu_0\end{bmatrix} \circ \begin{bmatrix}\omega_1\\ \omega_2\\ \omega_3\\ 0\end{bmatrix} = \begin{bmatrix}\mu_0\omega_1 + \mu_2\omega_3 - \mu_3\omega_2\\ \mu_0\omega_2 + \mu_3\omega_1 - \mu_1\omega_3\\ \mu_0\omega_3 + \mu_1\omega_2 - \mu_2\omega_1\\ -\mu_1\omega_1 - \mu_2\omega_2 - \mu_3\omega_3\end{bmatrix}$$

$$\omega_E^- \circ M = \begin{bmatrix}\omega_1\\ \omega_2\\ \omega_3\\ 0\end{bmatrix} \circ \begin{bmatrix}\mu_1\\ \mu_2\\ \mu_3\\ \mu_0\end{bmatrix} = \begin{bmatrix}-\mu_0\omega_1 - \mu_2\omega_3 + \mu_3\omega_2\\ \mu_0\omega_2 - \mu_3\omega_1 + \mu_1\omega_3\\ \mu_0\omega_3 - \mu_1\omega_2 + \mu_2\omega_1\\ -\mu_1\omega_1 - \mu_2\omega_2 - \mu_3\omega_3\end{bmatrix}$$

Hence

$$M \circ \omega_E^- - \omega_E^- \circ M = 2\begin{bmatrix}\mu_2\omega_3 - \mu_3\omega_2\\ \mu_3\omega_1 - \mu_1\omega_3\\ \mu_1\omega_2 - \mu_2\omega_1\\ 0\end{bmatrix} = 2\begin{bmatrix}\vec{\mu} \times \omega_E^-\\ 0\end{bmatrix}$$

Cross product can be replaced by dot product using ϕ function:

$$M \circ \omega_E^- - \omega_E^- \circ M = \begin{bmatrix}2\phi(\vec{\mu})\omega_E^-\\ 0\end{bmatrix} = \begin{bmatrix}-2\phi(\omega_E^-)\\ 0\end{bmatrix}\begin{bmatrix}\vec{\mu}\\ 0\end{bmatrix}$$

Similar to $\omega_E^- \circ M$, $\Delta\omega \circ M$ can be calculated.

$$\dot{M} = \begin{bmatrix}\dot{\vec{\mu}}\\ \dot{\mu}_o\end{bmatrix} = \begin{bmatrix}-\phi(\omega_E^- - 0.5 * \Delta\omega) & 0.5 * \Delta\omega\\ -0.5 * \Delta\omega^T & 0\end{bmatrix}\begin{bmatrix}\vec{\mu}\\ \mu_o\end{bmatrix}$$

As $\vec{\mu}, \Delta\omega$ are our new states, the equations can be written as:

$$\dot{\vec{\mu}} = \phi(\omega_E^- - 0.5 * \Delta\omega)\vec{\mu} + 0.5 * \Delta\omega * \mu_o$$

$$\dot{\vec{\mu}} = \boldsymbol{\phi(\omega_E^-) * \vec{\mu} - 0.5 * \phi(\Delta\omega) * \vec{\mu} + 0.5\mu_o\Delta\omega} \quad \textbf{(18)}$$

$$\dot{\vec{\mu}} = \left|\frac{\partial\dot{\vec{\mu}}}{\partial\vec{\mu}}\right|_{\vec{\mu}=0} * \vec{\mu} + \left|\frac{\partial\dot{\vec{\mu}}}{\partial\vec{\mu}}\right|_{\vec{\mu}=0} * \vec{\mu} + \left|\frac{\partial\dot{\vec{\mu}}}{\partial\Delta\omega}\right|_{\Delta\omega=0} * \Delta\omega$$

$$\boldsymbol{F_{\mu\mu} = \left|\frac{\partial\dot{\vec{\mu}}}{\partial\vec{\mu}}\right|_{\Delta x=0} = |\phi(\omega_E^-) - 0.5 * \phi(\Delta\omega)|_{\Delta x=0}} \quad \textbf{(19)}$$

$$= -\boldsymbol{\phi(\omega_E^-)}$$

$$\boldsymbol{F_{\mu\omega} = \left|\frac{\partial\dot{\vec{\mu}}}{\partial\Delta\omega}\right|_{\Delta x=0} = |0.5 * \phi(\vec{\mu}) + 0.5\mu_o|_{\Delta x=0} = 0.5 * I(3,3)} \quad \textbf{(20)}$$

Eventually, the F matrix is:

$$\boldsymbol{F = \begin{bmatrix} F_{\mu\mu} & F_{\mu\omega} \\ F_{\omega\mu} & F_{\omega\omega} \end{bmatrix}} \quad \textbf{(21)}$$

And the state vector is to be:

$$\Delta x = \begin{bmatrix} \vec{\mu} \\ \Delta\omega \end{bmatrix}$$

Measurement output which depends on the new states is represented by the difference between measurement outputs and estimated outputs equivalent to estimated states and given by:

$$\begin{aligned} Z = B_f - B_E^- &= \tilde{q} \circ B_o \circ q + \zeta^B - B_E^- \\ &= M \circ \tilde{q}^- \circ B_o \circ q^- \circ \widetilde{M} + \zeta^B - B_E^- \\ &= M \circ B_E^- \circ \widetilde{M} + \zeta^B - B_E^- \end{aligned}$$

Where:

$$M \circ B_E^- \circ \widetilde{M} = B_E^- + 2\mu_o B_E^- \times \mu - 2\mu \times B_E^- \times \mu$$

$$Z = -2\mu_o B_E^- \times \mu - 2\mu \times B_E^- \times \mu + \zeta^B$$

Linearization technique:

$$\begin{aligned} Z &= \left|\frac{\partial Z}{\partial \vec{\mu}}\right|_{\vec{\mu}=0} \vec{\mu} \\ &= -\left|2\mu_o \phi(B_E^-) + \frac{\partial(2\phi(\mu).B_E^-)}{\partial \vec{\mu}} \times \mu + 2\phi(\mu).B_E^-\right|_{\vec{\mu}=0} \vec{\mu} \\ &= -2\phi(B_E^-)\vec{\mu} \end{aligned}$$

As the angular velocities are measured directly, the corresponding C matrix elements are identity.

$$\boldsymbol{C} = \begin{bmatrix} -2\boldsymbol{\phi}(\boldsymbol{B_E^-}) & \mathbf{0} \\ \mathbf{0} & \boldsymbol{I}(\mathbf{3},\mathbf{3}) \end{bmatrix} \quad \textbf{(22)}$$

And $\boldsymbol{D} = \mathbf{0}$

The measurement model will be explained in more details in the next section.

****

c. EKF applied to the error-states model

The number of sensors used in attitude estimation process differs according to the mode of operation. There are two versions required for the EKF. These two models are to be explained in this section

1. **Full-Kalman model**

During the Imaging mode of operation, the maximum accuracy is required. Therefore, both of the magnometer and gyro measurements are observed.

i. Measurement Model

The model of the linearization of the system deals with the measurement equations by obtaining the characteristic of the noise represented in the following form:

- **Magnometer:**

$$B_{mm}^{b} = B_0 + 2q_0 B_0 \times q_v - 2\, q_v \times B_0 \times q_v + \xi_{dBw}$$

- **Gyro:**

$$\omega_{gy} = \omega_{act} + \xi^{\omega}$$

Where:

B_{mm}^{b}:The measured magnetic field in the BCS

B_0: The geomagnetic field in the OCS

q_v: The vector part of the quaternion that specifies the position of the BCS relative to the OCS.

q_0: The scalar part of the same quaternion.

ξ_{dBw} : The magnometer sensor noise and can be separated to:

$$\xi_{dBw} = dB + \xi^{B}$$

Where

$$dB = \text{Magnometer bias}, \quad \xi^{B} = N\left(0\,, R_K^{B}\right) = \text{sesnor noise}$$

ξ^{ω}: The Gyro sensor noise and can be represented as: $\xi^{\omega} = N(0\,, R_K^{\omega})$

Using linear relation at a point $\Delta x = 0$, measurements y can be represented in this form:

$$y = H\,\Delta x + \xi$$

Where:

$$H = \begin{bmatrix} -2\phi(B_E^-) & 0 \\ 0 & E3 \end{bmatrix}$$

Where:

B_E^-: is the estimated magnetic field and can be represented as:

$$B_E^- = B_O + 2q_0{}^- B_O \times q_v{}^- - 2\, q_v{}^- \times B_O \times q_v{}^-$$

B_O: is the orbital magnetic field.

ii. Linear Dynamics Model

The linear dynamics model can be written in the form:

$$\Delta\dot{x} = F\Delta x + \zeta$$

Where:

$$\Delta\dot{x} = \begin{bmatrix} \mu \\ \Delta\omega \end{bmatrix}$$

$$F = \begin{bmatrix} -\phi(\omega_E^-) & \frac{1}{2}E_3 \\ -6\omega_0^2 * G(\beta, er^-) * \phi(er^-) & G(\beta, \omega_E^-) \end{bmatrix}$$

ζ: The process noise. $N(0, Q)$

iii. Y vector

$$y_{m_{6\times1}} = \begin{bmatrix} \left(\frac{B_E^-}{\|B_E^-\|} - \frac{B_{mm}^B}{\|B_{mm}^B\|}\right) \\ \omega_E^- - \omega_{gy} \end{bmatrix}$$

Where y is to be explained in the algorithm below.

2. B-Kalman model

Only MM measurements will be observed during the operation of the SB mode. The reason of this is to save power as the highest accuracy is not required in this mode, so the need of the B-kalman model arises which deals only with the MM readings to perform its algorithm.

i. Measurement Model

The model of the linearization of the system deals with the measurement equations by obtaining the characteristic of the noise represented in the following form:

- **Magnometer:**

$$B_{mm}^{b} = B_0 + 2q_0 B_0 \times q_v - 2\, q_v \times B_0 \times q_v + \xi_{dBw}$$

Using linear relation at a point $\Delta x = 0$, measurements y can be represented in this form:

$$y = H\,\Delta x + \xi$$

Where:

$$H = -2\phi(B_E^-)$$

ii. Linear Dynamics Model

The linear dynamics model can be written in the form:

$$\Delta\dot{x} = F\Delta x + \zeta$$

Where:

$$\Delta\dot{x} = \mu$$

$$F = -\phi(\omega_E^-)$$

ζ: The process noise. $N(0, Q)$

iii. Y vector

$$y_{m_{3\times1}} = \left[\left(\frac{B_E^-}{\|B_E^-\|} - \frac{B_{mm}^B}{\|B_{mm}^B\|}\right)\right]$$

****

Structure of the EKF

- Full Kalman

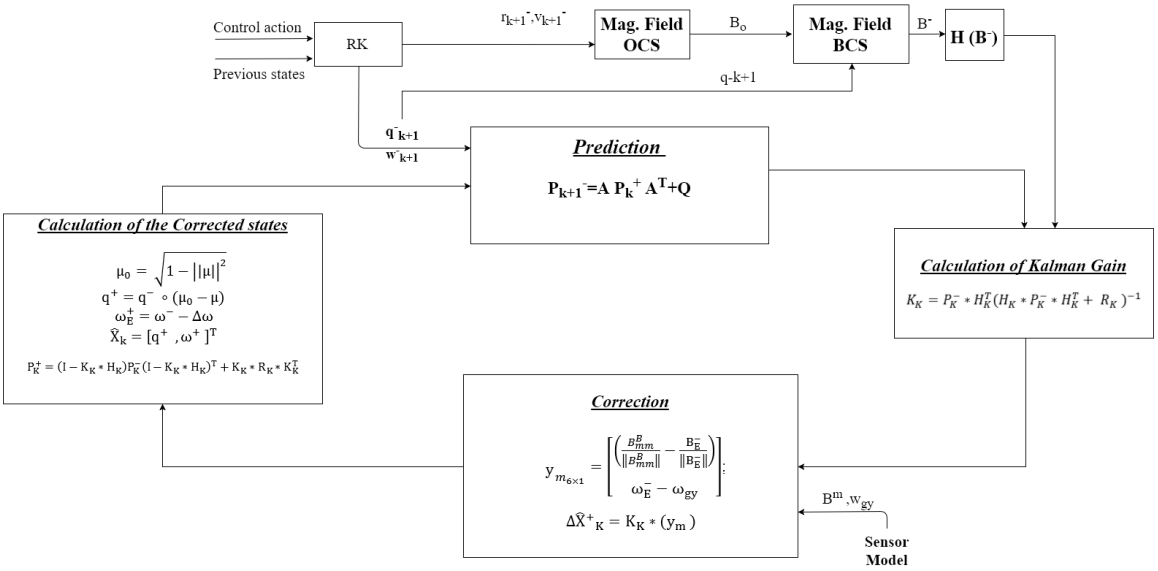


Figure 33| Full EKF algorithm

- B-Kalman

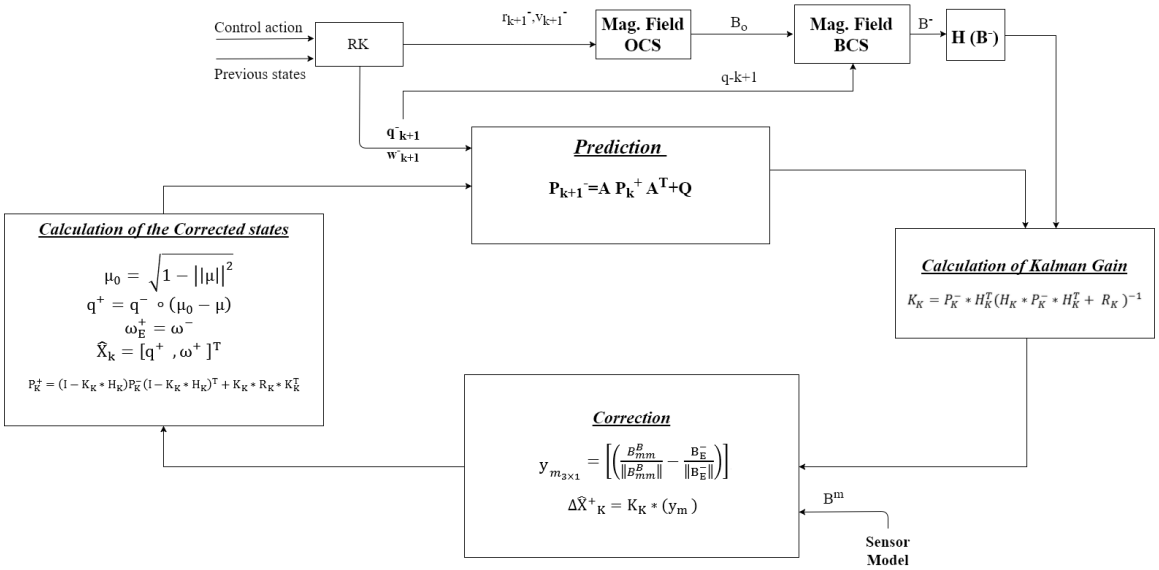


Figure 34| B- EKF algorithm

Kalman filter Procedure

1- Calculate Kalman's gain

Kalman gains will be calculated according to the following formula:

$$K_K = P_K^- * H_K^T (H_K * P_K^- * H_K^T + R_K)^{-1}$$

Where:

K_K : is an observer gain.

P_K^- : Estimated error covariance matrix.

H_K : Linearized error measurement matrix.

R_K : Measurement noise covariance matrix.

2- Calculate the error signal between estimated and measured output due to estimated states

$$y_{m_{6\times1}} = \begin{bmatrix} \left(\frac{B_{mm}^B}{\|B_{mm}^B\|} - \frac{B_E^-}{\|B_E^-\|}\right) \\ \omega_E^- - \omega_{gy} \end{bmatrix}$$; For n=6

$$y_{m_{3\times1}} = \left[\left(\frac{B_{mm}^B}{\|B_{mm}^B\|} - \frac{B_E^-}{\|B_E^-\|}\right)\right]$$; For n=3

y_m : is output error value.

B_E^- : Estimated magnetic field in BCS.

B_{mm}^B : Magnetometer reading in BCS.

r_{meas} : Sun vector xcalculated from Sun sensor reading.

ω_E^- : Estimated angular velocities.

ω_{gy} : Gyro reading.

$$y_m = y_m(1:n)$$

Where n is the number of sensors used in attitude estimation process, and it differs according to the mode of operation. It can be:

$$n = \begin{cases} 6 & IM\ (MM, Gyro) \\ 3 & SM\ (MM) \end{cases}$$

3- Calculate the corrected state vector (error model)

The corrected error states as quaternion and angular velocities are calculated by multiplying Kalman gains with measurements error, as stated in the following equation:

$$\Delta\hat{X}^{+}{}_{K} = K_K * (y_m)$$

4- Calculate the corrected quaternion and angular velocity

To calculate the corrected quaternion and angular velocities using corrected error values, we use the following formula.

$$\mu_0 = \sqrt{1 - ||\mu||^2}$$
$$q^+ = q^- \circ (\mu_0 - \mu)$$
$$\omega_E^+ = \omega^- - \Delta\omega$$
$$\hat{X}_k = [q^+ , \omega^+]^T$$

Where
μ_0: is the scalar part of the error quaternion.
μ: is the vector part of the error quaternion.
q^+: Corrected quaternion.
q^-: Predicted quaternion.
ω_E^+: Corrected angular velocity.
ω^-: Predicted angular velocity.

5- Calculate the corrected error covariance matrix

$$P_K^+ = (I - K_K * H_K)P_K^-(I - K_K * H_K)^T + K_K * R_K * K_K^T$$

6- Update state vector and the covariance matrix

The updated states can be calculated using the onboard dynamics and disturbance model explained previously in this document.
To update the covariance matrix, the following relation can be used:

$$F = \begin{bmatrix} -\phi(\omega_E^+) & \frac{1}{2}E_3 \\ -6\omega_0^2 * G(\beta, er^+) * \phi(er^+) & E_3 \end{bmatrix}$$

$$A = e^{F\Delta t}$$

$$P_{k+1}{}^- = A * P_k{}^+ * A^T + Q$$

Kalman filter Test

In order to test the full EKF algorithm, the simulation was run with the following initial conditions and the results are shown. This sample of the results was collected during the 12000 seconds just after detumbling mode where the Full-Kalman is in operation.

- Initialization

$$P_o^- = \begin{bmatrix} (100 \times 10^{-8})^2 * I(3,3) & zeros(3,3) \\ zeros(3,3) & 1 \times 10^{-10} * I(3,3) \end{bmatrix}_{6\times6}$$

$$q_o^- = [1 \quad 0 \quad 0 \quad 0]$$

$$\omega_o^- = [0 \quad \omega_o \quad 0]$$

Where: ω_o is the orbital angular velocity.

Magnometer Measurement Noise:

$$\sigma_B = 5 \times 10^{-7} \text{ Tesla}$$

Gyro Measurement Noise:

$$\sigma_\omega = 5 \times 10^{-5} \text{ rad/sec}$$

- Simulation results

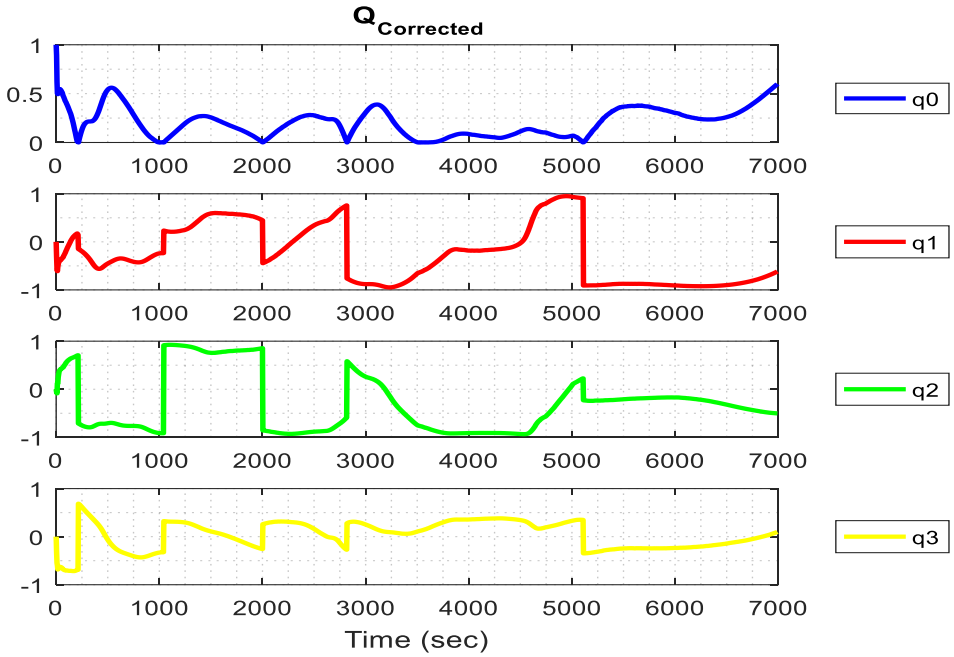


Figure 35| Corrected quaternion

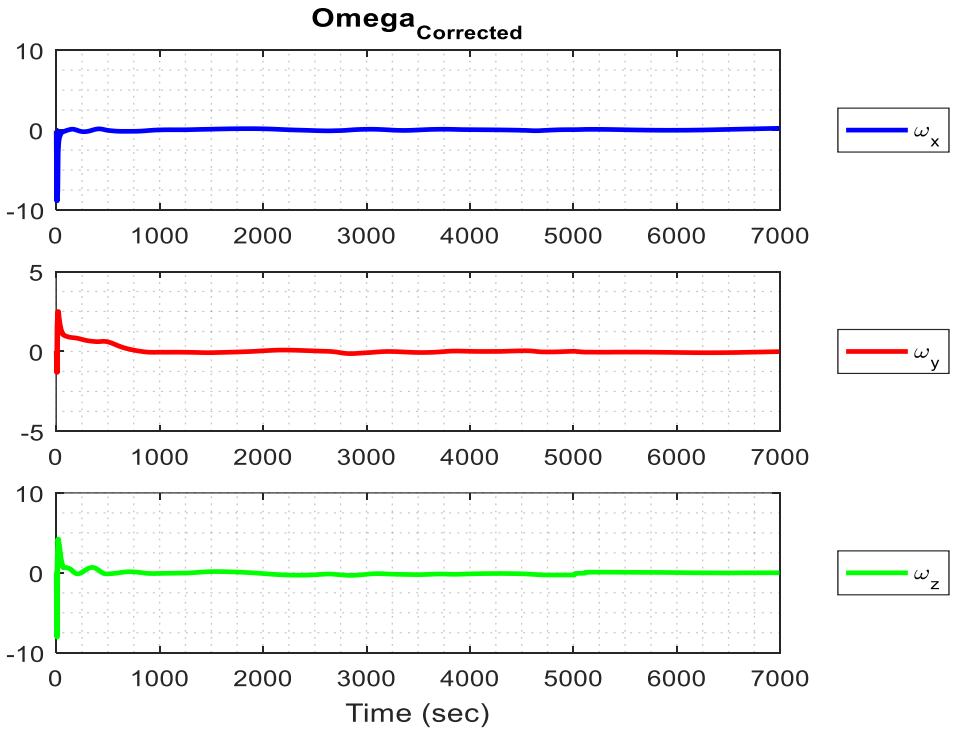


Figure 36| Corrected angular velocity

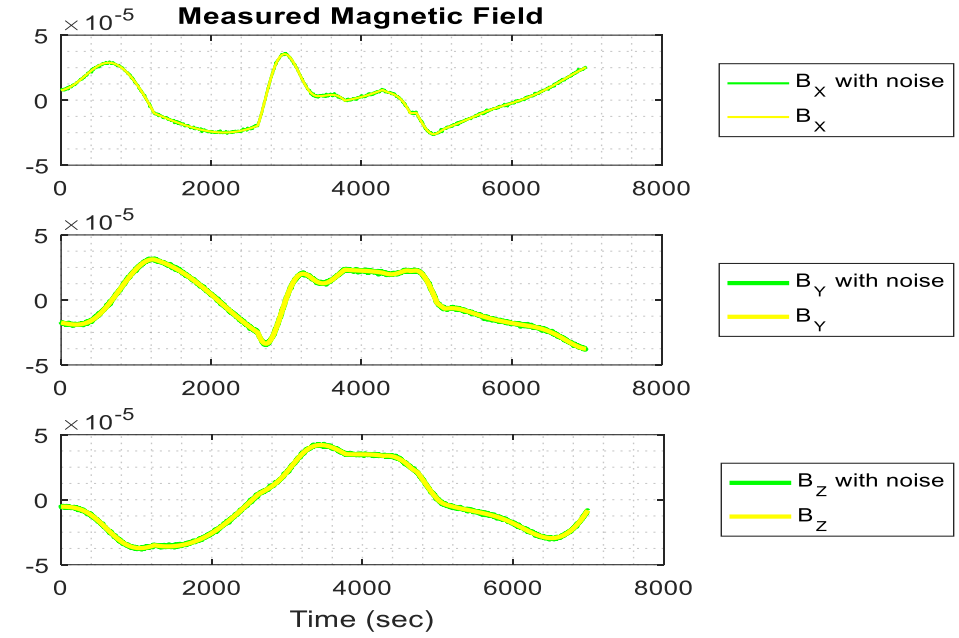


Figure 38| Measured Magnetic Field

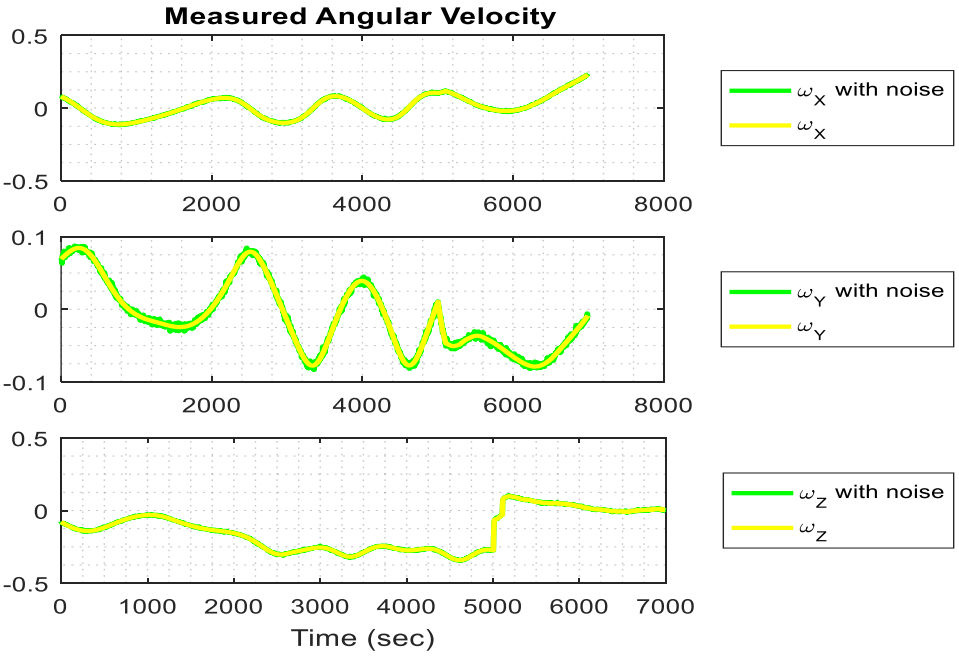


Figure 37| Measured angular velocity

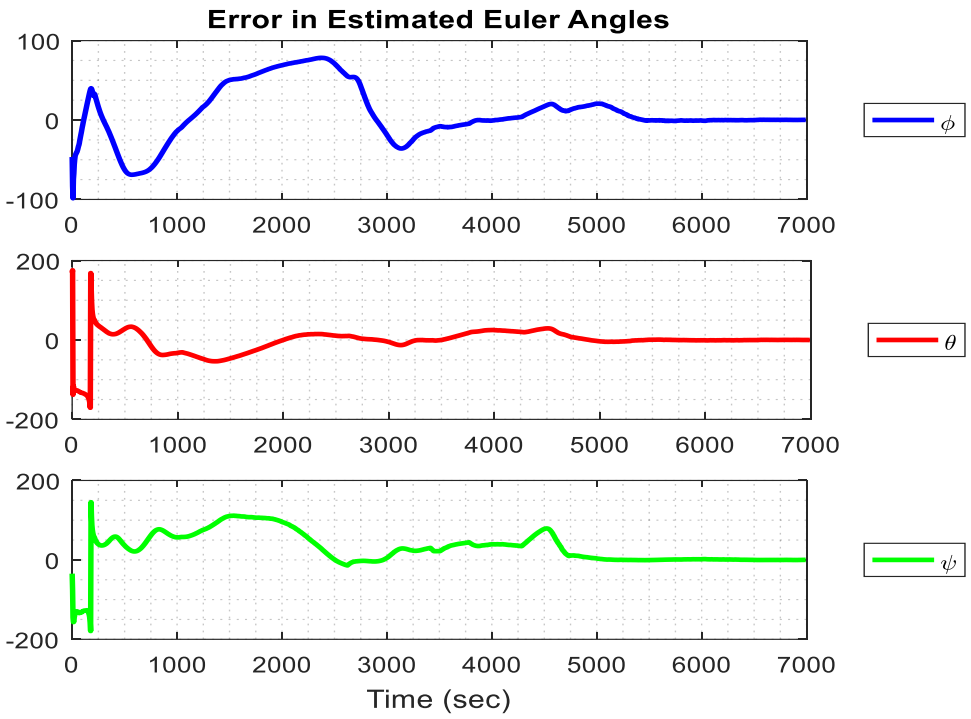


Figure 39| Error in estimated euler angles

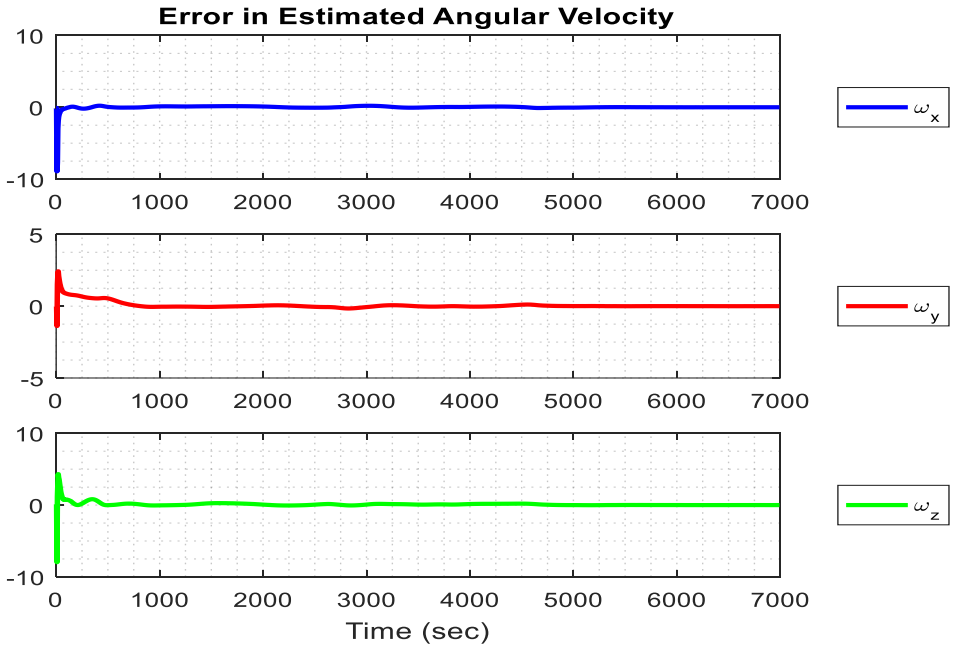


Figure 40| Error in estimated angular velocity

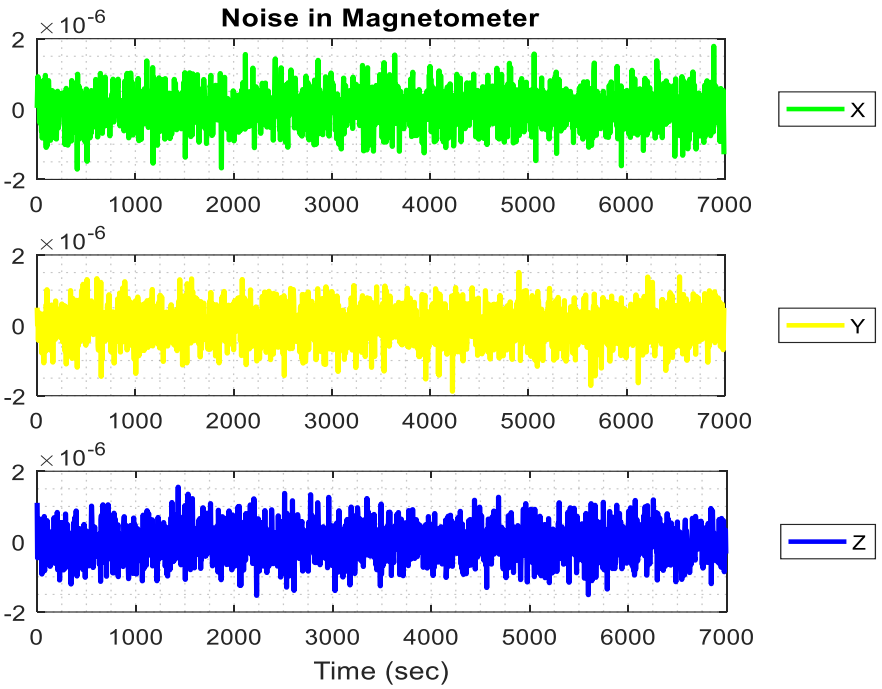


Figure **41**| Noise in MM

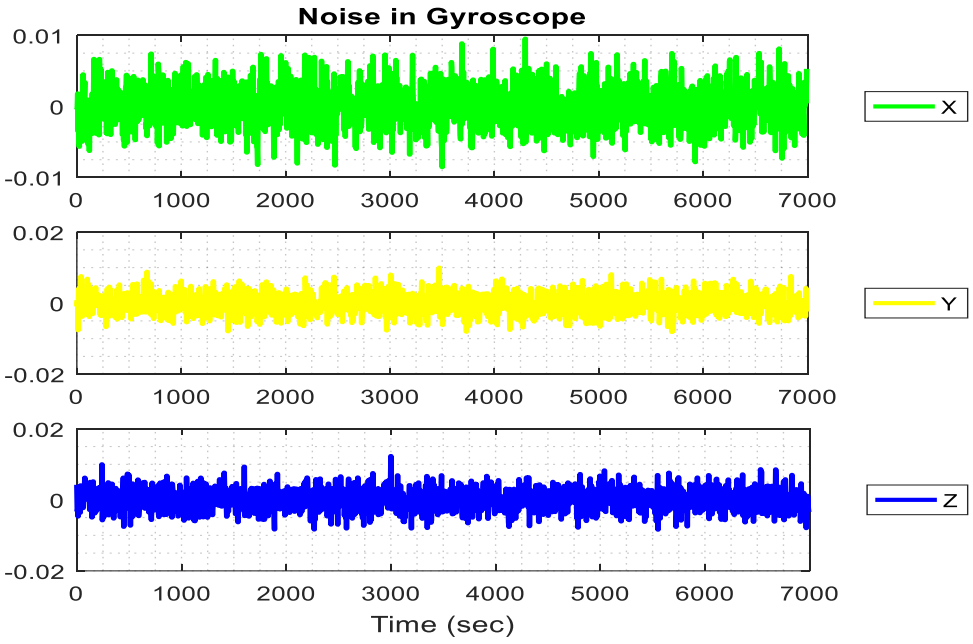


Figure **42**: Noise in Gyroscope

Chapter 5

Attitude Control Algorithms

6.1 Introduction

The main tasks of ADCS are to control the angular rotation of satellite starting from separation from launcher then attitude acquisition and then keep satellite stabilization at nadir pointing .Recently magnetic actuator become one of the most used actuators in spacecraft attitude control. Generally magnetic control algorithms used in attitude control are divide to angular suppression algorithms, in addition to attitude acquisition and stabilization algorithms.

Different control algorithms are required due to the difference in angular velocities and angles together with the nonlinearity of the system described in the dynamics and kinematics chapter (4), Also due to the difference in measurement and attitude estimation available in each segment of the mission.

For example, during de-tumbling large actuation power is required also the other subsystems don't begin their function normally, and the system encounter a critical stage, so minimizing the required power during this critical stage is essential. This leads to restriction on the operation of devices that requires high power like rate gyro, so in small satellites the measuring the angular velocities is not a likely thing.

Also during de-tumbling it's required to minimize the computational power, so there is restriction on heavy algorisms of attitude determination, do it's not likely to use controllers built on estimating the attitude angles or quaternion.

That means the need for different control algorisms, to be utilized during the mission of controlling the attitude of a cube sat.

In this section different control algorisms are investigated and tested to fulfill the requirements and to determine advantages and dis-advantages of them, then in next section the integration between them in the onboard software is described.

6.2 Attitude Magnetic control concept. [8]

Magnetic actuator was used because it is a low power consumption, small mass, low cost and reliability in attitude control. In this case control problem of the satellite involves angular velocity suppression, attitude acquisition and finally attitude stabilization will be solved by magnetic actuator only.

The main concept of magnetic attitude control of satellite is to generate dipole moment L. This dipole moment reacts with the earth magnetic field B generating control torque T_c used to control the satellite rotation.

$$T_c = L \times B \tag{23}$$

The satellite actuated by a set of magnetorqure (MT) has a serious limitation, The mechanical torque, produced by the interaction of the geomagnetic field and dipole moment generated by the MT, is always perpendicular to the geomagnetic field vector. Thus, the direction parallel to the geomagnetic field vector is not controllable. The geomagnetic field changes it's orientation in the OCS when the satellite moves in orbit. This implies that yaw is not controllable over the poles but controllable over the equator and roll is not controllable over the equator but controllable over the poles. See fig (**43**)

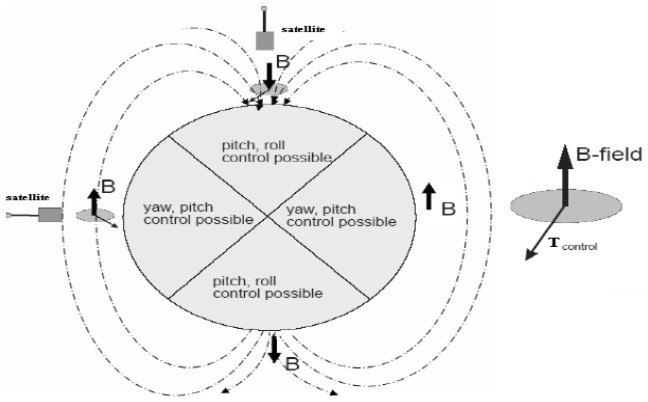


Figure 43| Magnetic Field Controllability

So in case of completely magnetic controlled satellites another source of torque is required to keep the satellite stable, where the steepest is by gravity gradient torque for nadir pointing satellites, and this is the way we utilized.

6.3 Algorithms Used For Angular Velocity Suppression

The objective of the angular velocity suppression or detumbling controller is to suppress the high angular velocity of satellite obtained due to separation from launcher. Commonly there are two methods used for satellite angular suppression, angular velocity feedback and B-dot technique.

6.3.1 Detumbling controller based on B-dot

The first and most important attitude control task to be executed after orbital insertion of a satellite is stabilizing its angular rate, i.e. detumbling. This procedure should be done by a robust and failsafe system which does not depend on "very complex" systems being operational like e.g. attitude estimation filters. A very simple solution to detumbling using magnetic actuation is the $\dot{B}$ (B-dot) algorithm.

The principle of a $\dot{B}$-controller is to minimize the derivative of the magnetic field vector measured by a magnetometer. As the spacecraft orbits the earth, the magnetic field vector in the spacecraft orbit frame (OCS) changes depending on the position of the spacecraft. However, the dominant rate of change in direction of the field vector is caused by the tumbling of the satellite as it may tumble with angular rates much larger than the orbital rate ($w_{orbit} = 0.0011\ rad/s$). Minimizing the change in the measured field vector by means of actuation causes the spacecraft to approach an angular rate close to the orbital rate which is achieved by forcing the derivative of the measured B-field $\dot{B}$, to zero.

The control law for $\dot{B}$ is nice and neat and can be written as eq(24)

$$L_c = -K\,\dot{B} \tag{24}$$

Where L_C is the required magnetic dipole moment from the actuator and $\dot{B}$ is the time derivative of the magnetic field vector measured in the body coordinats, and K is the controller gains, and the –ve sign to actuate in an opposite direction to the rotation.

The reason that L is obtained directly without the need for a cross multiplication as will be seen in other control algorithm is that the derivative of the B-field is perpendicular to the field vector.

And can be written as

$$\dot{B} \cong B \times \omega$$

Where ω is the angular velocity of the space craft relative to the inertial frame given the assumption that the direction and magnitude of the B-field with respect to the orbit coordinate system (OCS), is constant. This assumption leads to the conclusion that the rate of change of the B-field in the spacecraft fixed frame (BCS) is mainly due to the rotation of the spacecraft.

- ***Estimating B-dot***

One problem with the $\dot{B}$-algorithm is that $\dot{B}$ cannot be directly measured by the magnetometer and differentiating its output may give peaks of unwanted noise.

So a continuous time filter that estimates the rate of change of the B-vector can be realized and simply multiplying its output by the controller gain C gives the controller output to the magnetorquers as a dipole moment fig (44)

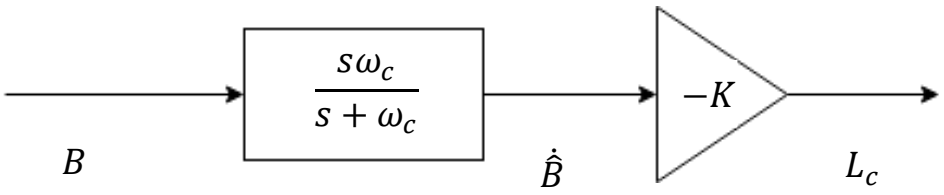


Figure 44| B-dot Filter

Where $\hat{\dot{B}}$ is the estimated one after the filter, Then the TF of the filter would be

$$H_{cont} = \frac{\hat{\dot{B}}}{B} = \frac{s\omega_c}{s+\omega_c} \quad (25)$$

In reality, the controller will be implemented on a computer and therefor needs to be discrete. Using pole-zero matching and gain matching the transfer function of the $\dot{B}$-estimator can be written in the z-domain as shown in the following

$\text{Poles}_{\text{continuous}} = -\omega_c$ $\quad$ $\text{Poles}_{\text{discrete}} = e^{-\omega_c Ts}$

$\text{Zero}_{\text{continuous}} = 0$ $\quad$ $\text{Zero}_{\text{discrete}} = 1$

$$H_{disc}(z) = k\frac{z-1}{z-e^{-\omega_c Ts}}$$

The gain is matched in center of the bandwidth, that is at $\omega_0 = \frac{\omega_c}{2}$ and the gain correction K is computed as follows:

$$k = \frac{|H_{cont}(s)|}{|H_{disc}(z)|}\ |s = j\omega_0, z = e^{-j\omega_0 Ts} \quad (26)$$

Where Bode plot of this TF is as the Following for $\omega_c = 0.7$

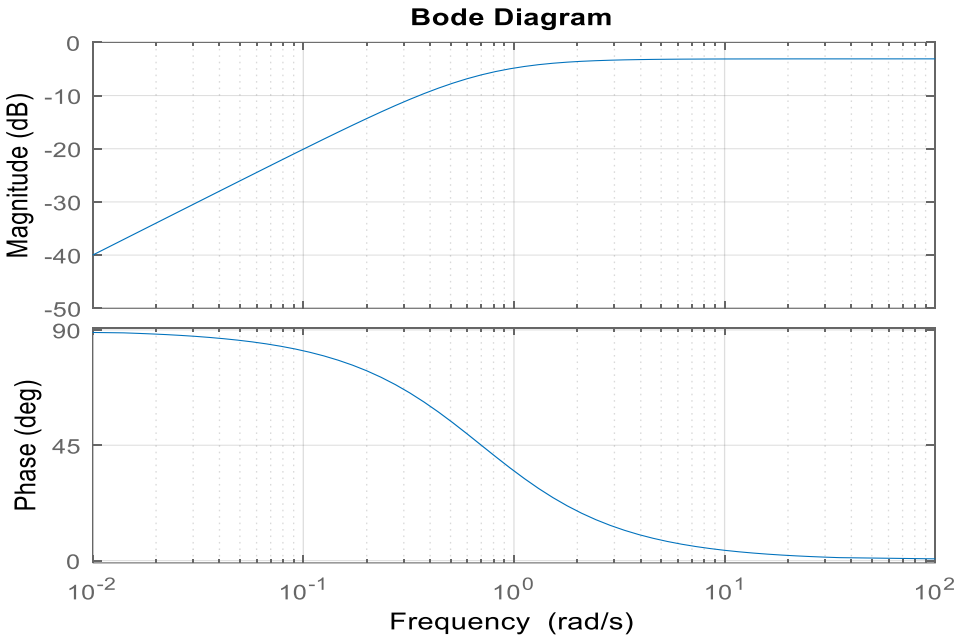
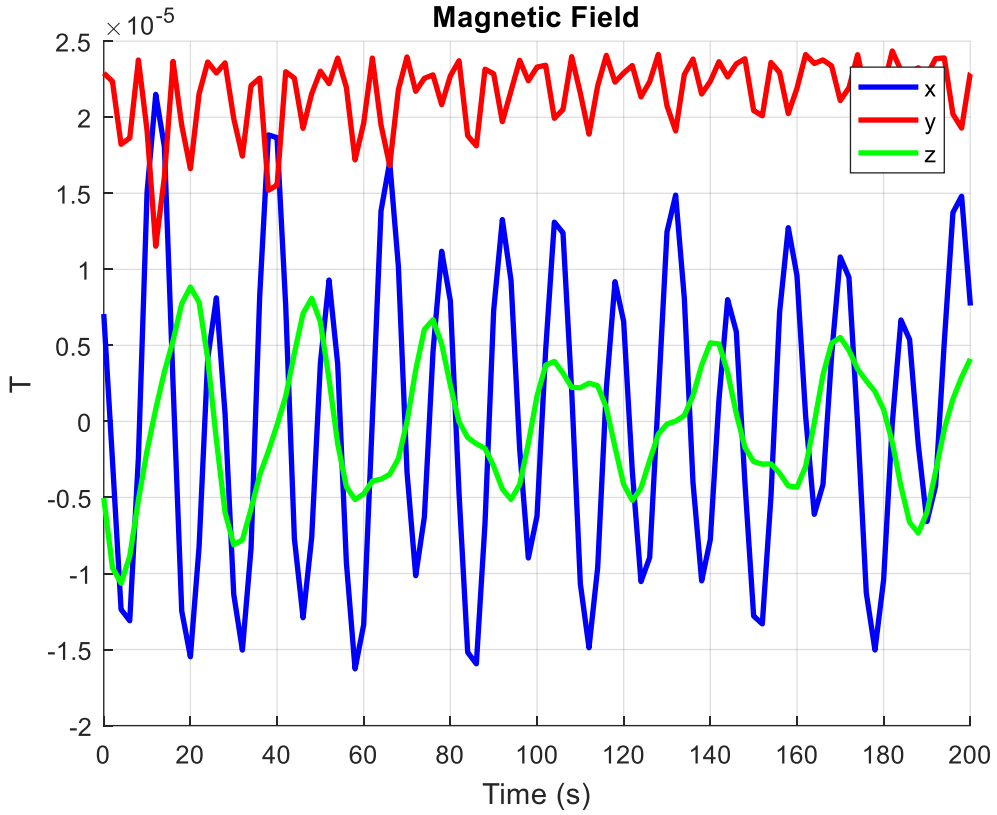


Figure 45|Bode plot

The cut of frequency can be determined from simulating the satellite motion and determine the maximum frequency of the magnetic field at angular velocity equals the design limit of the satellite during separation from the launcher.

this figure shows how B changes with $\omega = [10,10,10]\ deg/sec$



From the figure the maximum frequency for B is 0.785 rad/sec

So we have determined the cutoff frequency but the correction gain is still remaining, form eq (26)

$$k = \frac{\left|\frac{\omega_c^2}{2} j\right| \left|e^{-j\frac{\omega_c}{2}Ts} - e^{-\omega_c Ts}\right|}{\left|\frac{\omega_c}{2} j + \omega_c\right| \left|e^{-j\frac{\omega_c}{2}Ts} - 1\right|}$$

After substituting the complex exponential and solving for the magnitudes, the resulted equation is as the following:

$$K = \frac{\omega_c}{\sqrt{5}} \sqrt{\frac{2 - 2\cos\frac{\omega_c Ts}{2}}{1 - 2\cos\frac{\omega_c Ts}{2}\ e^{-\omega_c Ts} + e^{-2\omega_c Ts}}}$$

- ***Stability of the B-dot controller***

The stability using the control law in (1) can be proven by the Lyapunov direct method. Since the stability criteria for detumbling implies that the rotational kinetic energy of the satellite should converge to zero[1], a Lyapunov candidate function is the energy equation which is + ve definite

$$E_{rot} = 0.5\ \omega^T I \omega \qquad (27)$$

Where $\dot{E}$= the work done by the actuation, which is

$$\dot{E_{rota}} = \omega^T M_{control}$$

From eqs (43,24)

$$\begin{aligned} \dot{E} &= \omega^T\left(-K\dot{B} \times B\right) \\ &= -K\omega^T(\dot{B} \times B) \end{aligned}$$

the cross product can be replaced by cross product matrix S

$$\begin{aligned} \dot{E} &= -K\omega^T\left(S\left(\dot{B}\right)B\right) \\ &= K\omega^T\left(S(B)\dot{B}\right) \\ &= K\left(S(B)\dot{B}\right)^T\omega \\ &= -K\dot{B}^T S(B)^T\omega \\ &= -K\dot{B}^T(B \times \omega) \\ &= -K\dot{B}^T\dot{B} \\ &= -K|\dot{B}^2| \end{aligned}$$

This equation describes the change of rotational kinetic energy of the spacecraft when applying the $\dot{B}$ control law. This equation is negative definite thus proving that energy is dissipated from the system during detumbling. Lyapunov analysis also shows that energy dissipation in the detumbling phase is proportional to k $\dot{B}$ which means that angular rates are reduced rapidly after initiating B-dot control and slowly converging over time.

[1] This is an assumption as the orbital rate of the satellite is not considered.

- ***Stability of estimated B-dot***

To study the stability of the system after adding the filter, the control moment equation is obtained from the estimated $\hat{\dot{B}}$, so the rate of change of energy become

$$\dot{E} = -K\hat{\dot{B}}^T\dot{B}$$

In order for this equation to stay negative definite the vector dot product $\hat{\dot{B}}^T\dot{B}$ must be negative at all times which can only be ensured if the absolute angle between $\hat{\dot{B}}$ and $\dot{B}$ is less that 90◦. If the angle is more than 90◦ then the dot product becomes negative and the equation becomes positive so the kinetic energy rises. In this case the Lyapunov analysis does not prove stability.

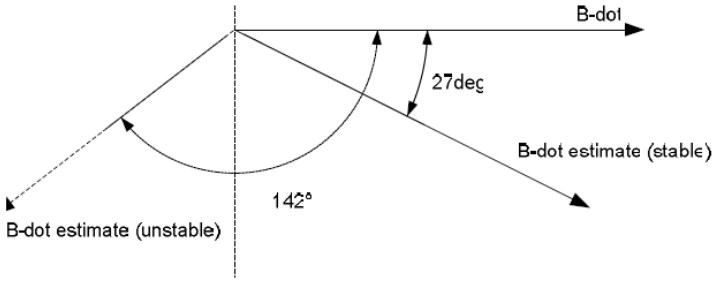


Figure 46| stability of $\hat{\dot{B}}$

By investigating eq (25)

$$\frac{\hat{\dot{B}}}{\frac{\dot{B}}{s}} = \frac{\omega_c S}{S + \omega_c}$$

$$\text{So, } \frac{\hat{\dot{B}}}{\dot{B}} = \frac{\omega_c}{S+\omega_c}$$

This Transfer function has a phase in the interval [0,90]. The interpretation of this is that when the frequency of change of $\dot{B}$ is large the angle between $\hat{\dot{B}}$ and $\dot{B}$ increases and stability becomes marginal; i.e. the phase margin decreases.

The frequency of $\dot{B}$ is related to the size of $\ddot{B}$. The following equations expresses $\ddot{B}$ with the assumption that the geomagnetic field is constant in the inertial coordinate system to start with in order to simplify the equations.

$$\dot{B} = B \times \omega$$
$$\ddot{B} = \dot{B} \times \omega + B \times \dot{\omega}$$

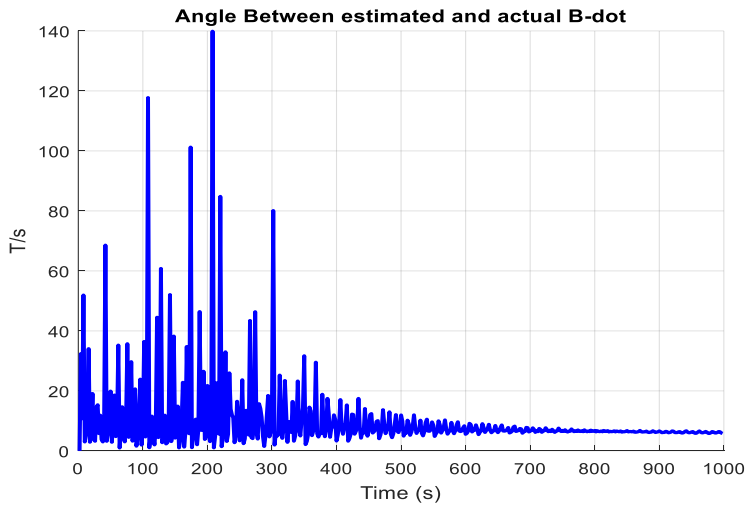
from dynamics equation. $\dot{\omega} = I^{-1}(M_{control} - \omega \times I.\omega)$

$$\ddot{B} = \dot{B} \times \omega + B \times I^{-1}(M_{control} - \omega \times I.\omega)$$

It is clear that the phase increases with the actuation torque which is proportional to the controller gain K and the size of $\dot{B}$ and inverse proportional to the inertia of the satellite.

So the controller gain must be checked to satisfy a certain phase margin, as there will be a contribution to the rate of $\dot{B}$ from the local variations in the B-field caused by the change of position of the spacecraft (i.e. its position in orbit). Measurement noise and pure time delay from the magnetometer may add even more phase to the estimated $\dot{B}$.

In our case after the simulation with $\omega_0 = [10,10,10]$, the angle between the estimated and actual magnetic fields is:



There are few peaks at very high angular velocities, but they didn't affect the simulation stability or results, and any attempt to decrease the controller gains will increase detumbling time. While all other angles are between $40^0 and\ 10^o$ with phase margin of mean 40^o.

- ***B-dot Algorithm Results***
- ***Angular Velocity***

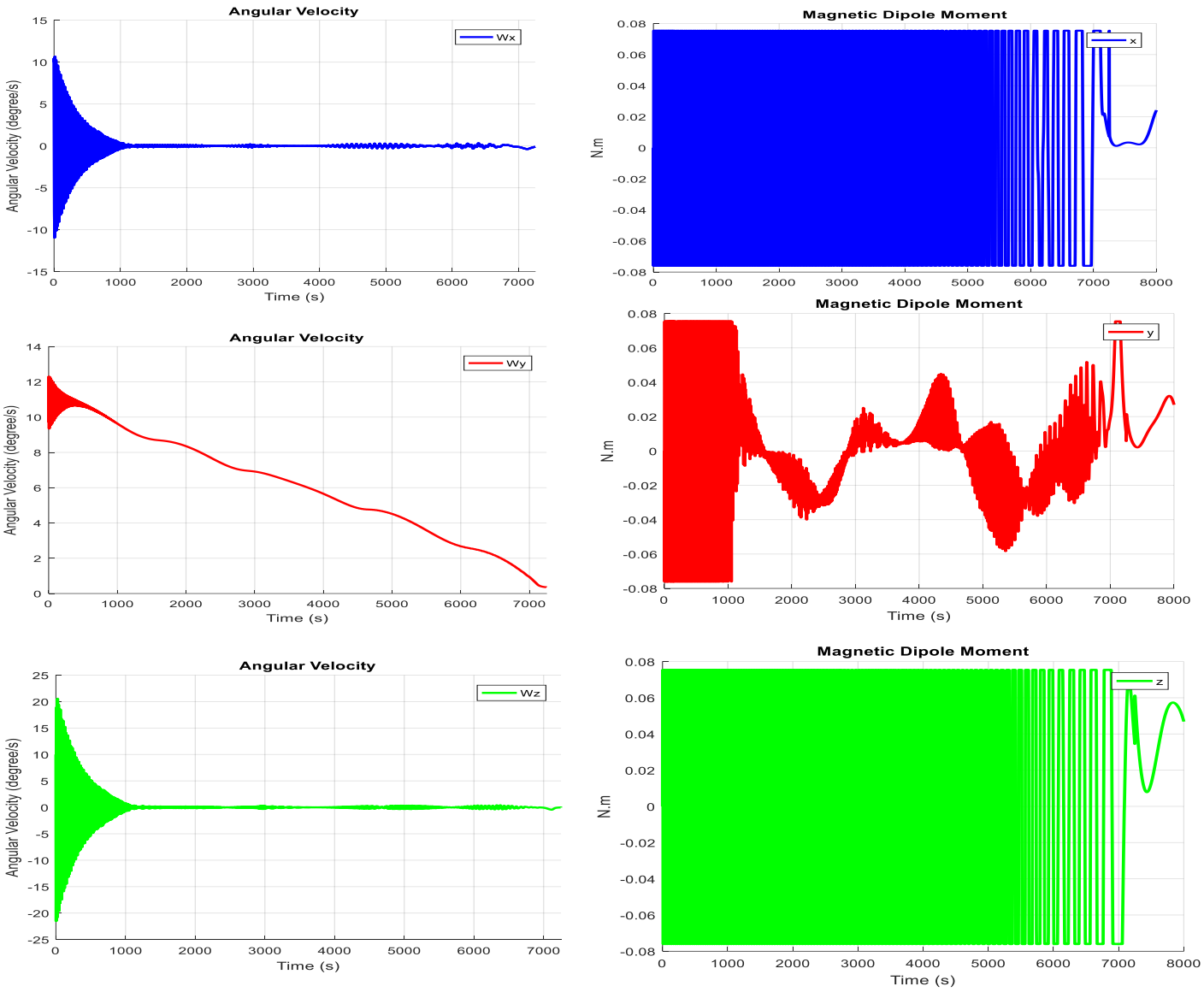


Figure 47| Angular velocity suppression by B-dot ($\omega_x, \omega_y, \omega_z$)

From the curves it's evident that complete angular velocity suppression occurred in time of 7250 sec which represent 1.34 orbits (for orbital parameters for EUS1 described in another section) from the beginning with maximum allowable angular velocity of [10,10,10] degree / sec in BCS.

Also figures of magnetic dipole moment show that it reached the saturation limit of the device which is acceptable during detumbling, however L_y is some how smaller because it depends on ω_x *and* ω_z which already damped quickly.

6.4 Attitude Acquisition and Stabilization

After angular velocity suppression, satellite may have arbitrary orientation (BCS of the satellite may not co-onside with the OCS). Therefore, it is required to make attitude acquisition or reorient the satellite in order to make the satellite nadir pointing (i.e. to make BCS of the satellite co-onside with the OCS), after that it is required to make attitude stabilization or keeping the satellite at nadir pointing. To make attitude acquisition and stabilization, the used control algorithm must receive information about the satellite attitude and angular velocity.

So measurements only will not give the full data about satellite orientation required for controlling the satellite, Therefore attitude determination algorithm should be utilized, In our case, **Extended Kalman Filter** algorism described in another section is utilized during all simulations.

For small attitude maneuvers a linearized model of the satellite dynamics and kinematics can be obtained in Euler angles representation, then a simple first order controller can be designed to control this simple second order system. Any linear controller design technique can be used to obtain certain response characteristics like damping ration (ζ) and natural frequency (ω_n).

The problem is less trivial when large attitude maneuvers are considered, for three principal reasons. First, the simplified dynamics of the linearized model does not hold for large attitude maneuvers. Second, there is a control problem with regard to saturation and that is, the maximum achievable torques. Also,. For the problem of controllability discussed before due to the fact that the control action depends on the magnetic field of the earth These control difficulties make the design of the controller in conventional way not a good choose. The best way in dealing with such case is by running Full Model non linear simulation to determine the best controller to satisfy the requirments.

Also Euler angles can't be used for another reason that, for large Euler attitude angles, the attitude kinematics equations can become singular. For example, in the Euler angle rotation, the Euler kinematics equations become singular. As we shall see, this drawback can be alleviated by using more effective kinematics expressions for the attitude control laws, all these expressions is derived from the idea of direction cosine matrix, can be developed to euler axis representation then quaternion representation.

In our case as we use the dynamics equation in form of quaternion and for it's lower computational requirements than direction cosine matrix. So, the control law in quaternion for is the most suitable. In the next section we are going to derive the concept of using quaternion in control law, Also the derivation should begin with the control law for the direction cosine matrix then develop to Euler axis then the quaternion form. All the next concept derivations are presented in [1]

6.4.1 PD Like Controller in Quaternion Form [1]

- ***Control Command Law Using the Direction Cosine Error Matrix***

Suppose that the attitude of the satellite is expressed in terms of the direction cosine matrix $[A_s]$ relative to the reference frame in which the attitude maneuver is to be commanded and achieved. Suppose then that a vector a has the components a_1, a_2, a_3.in the reference frame a = [a_1, a_2, a_3], and that the satellite is to be maneuvered so that its final direction cosine matrix will coincide with a known and defined matrix $[A_T]$, called the target matrix.

According to matrix transformation

$$a_s = [A_s]a$$
$$a_T = [A_T]a$$
$$so, a_s = [A_s][A_T]^{-1}a_T = [A_s][A_T]^T a_T = [A_E]a_T \qquad (28)$$

Where the matrix $[A_E]$ is the attitude error matrix, the required is to make the error matrix equals identity, the $[A_s]$=$[A_T]$, which means the satellite reached the required attitude, this is done by zeroing the off diagonal elements.

$$[A_E] = \begin{bmatrix} a_{11s} & a_{12s} & a_{13s} \\ a_{21s} & a_{22s} & a_{23s} \\ a_{31s} & a_{32s} & a_{33s} \end{bmatrix} \begin{bmatrix} a_{11T} & a_{21T} & a_{31T} \\ a_{12T} & a_{22T} & a_{32T} \\ a_{12T} & a_{23T} & a_{33T} \end{bmatrix} = \begin{bmatrix} a_{11E} & a_{12E} & a_{13E} \\ a_{21E} & a_{22E} & a_{23E} \\ a_{31E} & a_{32E} & a_{33E} \end{bmatrix}$$

From this direction cosine matrix multiplication the resultant error matrix is symmetric

And by taking one of the off diagonal terms to investigate the meaning of zeroing it and by referencing to fig (),

for example: $a_{12E} = X_s \,.\, Y_T$

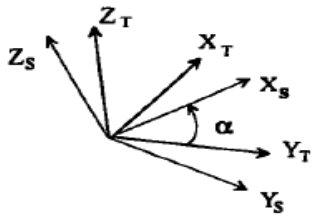


Figure 48|Geometric Interpretation of zeroing the off diagonal element (Sidi)

To make this dot product equals zero, the angle between X_s and Y_T must equals zero, his can be made by rotating the satellite around around it's body Z axis

So the control law would be

$$T_{cX} = K_{pX}\, a_{23E} + K_{dX} P$$
$$T_{cY} = K_{pY}\, a_{13E} + K_{dY}\, q$$
$$T_{cZ} = K_{pZ}\, a_{12E} + K_{dZ}\, r$$

Where p, q, r are the angular velocities in body coordinates

Also another control action law can be obtained from the definition of euler axis described in chapter (2) , so that the control torque tends to rotate the satellite around the Euler axis of rotation.

Then the resulted control law would be

$$T_{cX} = -K_{pX}\,(a_{32E} - a_{23E}) + K_{dX} P \qquad (29)$$
$$T_{cY} = -K_{pY}(a_{13E} - a_{23E}) + K_{dY}\, q$$
$$T_{cZ} = -K_{pZ}\,(a_{21E} - a_{12E}) + K_{dZ}\, r$$

This control law minimizing the angular path that will be traversed by the satellite in its angular motion.

The back draw of this control law is that it requires the computation of all 6 elements of the error direction cosine matrix, the quaternion form of this control law solves this problem.

- ***Quaternion Form***

We also can find the error direction cosine matrix from the quaternion of the body axes and target axes, from the transformations between quaternion and direction cosine matrix. And from eq (28)

$$A(q_E) = [A(q_s)][A(q_T)]^{-1}$$

Which is equivalent to quaternion multiplication $q_E = \tilde{q} \circ q$

And from the relations between direction cosine matrix and quaternion

$$q_4 = 0.5\sqrt{1 + a_{11} + a_{22} + a_{33}}$$
$$q_1 = \frac{0.25(a_{23} - a_{32})}{q_4}$$
$$q2 = \frac{0.25(a_{31} - a_{13})}{q_4}$$
$$q_3 = \frac{0.25(a_{12} - a_{21})}{q_4}$$

By using these transformations and eq (29) the control law would be

$$T_{cX} = -2K_{pX}\, q_{1E} q_{4E} + K_{dX} P$$
$$T_{cY} = -2K_{pY}\, q_{2E} q_{4E} + K_{dY}\, q$$
$$T_{cZ} = -2K_{pZ}\ \ q_{3E} q_{4E} + K_{dZ}\, r$$

This is the equation represented in reference [1]

Also multiplying by q_4 may case the system to settle at 180 degree not 0 as required for nadir pointing because $= \cos\frac{\alpha}{2}$ where α is euler rotation angle.

These control law produce needed control torque, however it should be converted to dipole moment to be sent to the actuators as required current.

This conversion it done through the following

$$T_c = L \times B$$

Multiply both sides by cross product with B

$$B \times T_c = B \times L \times B$$

By using this vector identity

$$A \times (B \times C) = B(A \,.\, C) - C(A \,.\, B)$$

We can get L as

$$L = \frac{B \times T_C}{|B|^2}$$

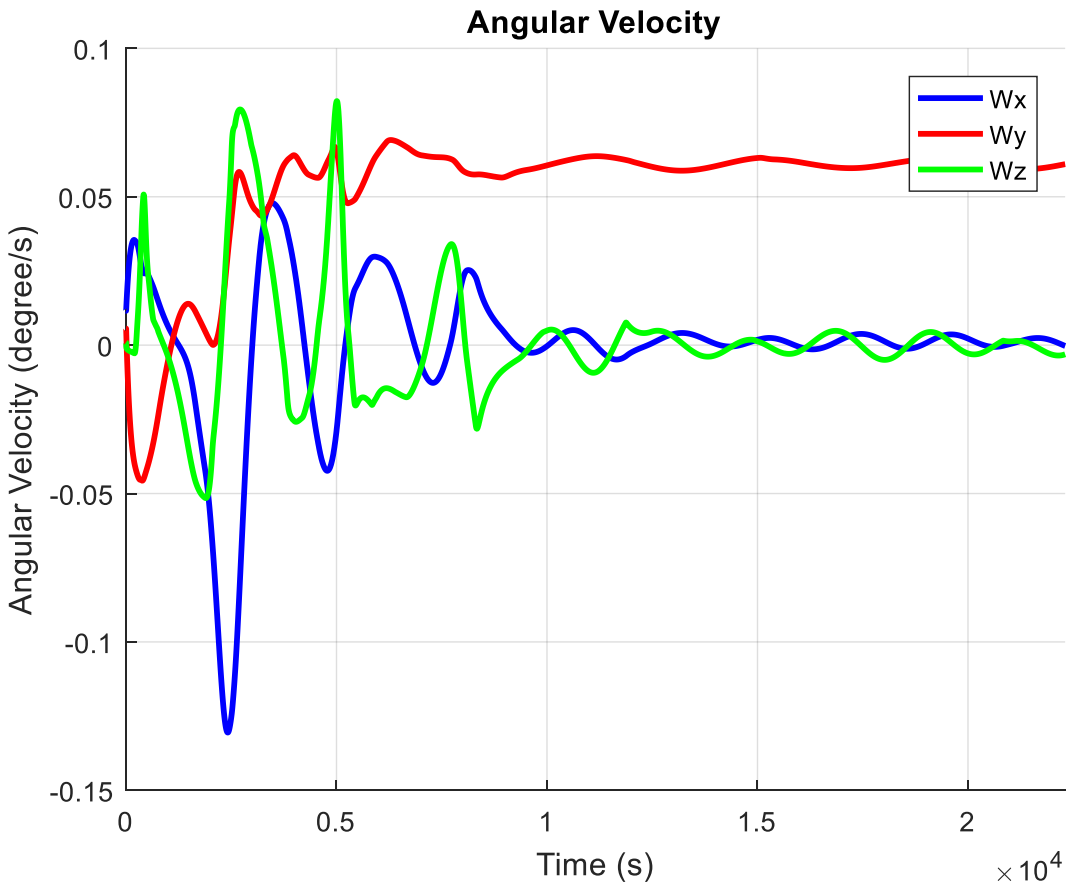
The product dipole moment is normal on the magnetic field as required.

6.5 Attitude Acquisition and Stabilization

The following simulation results are done for EUS 1 parameters provided in a previous section with initial conditions of angular velocities $\omega_0 = [1,1,1]$ deg/sec and for Euler angles $\phi, \theta, \psi = [65,150,-60]$ degree.

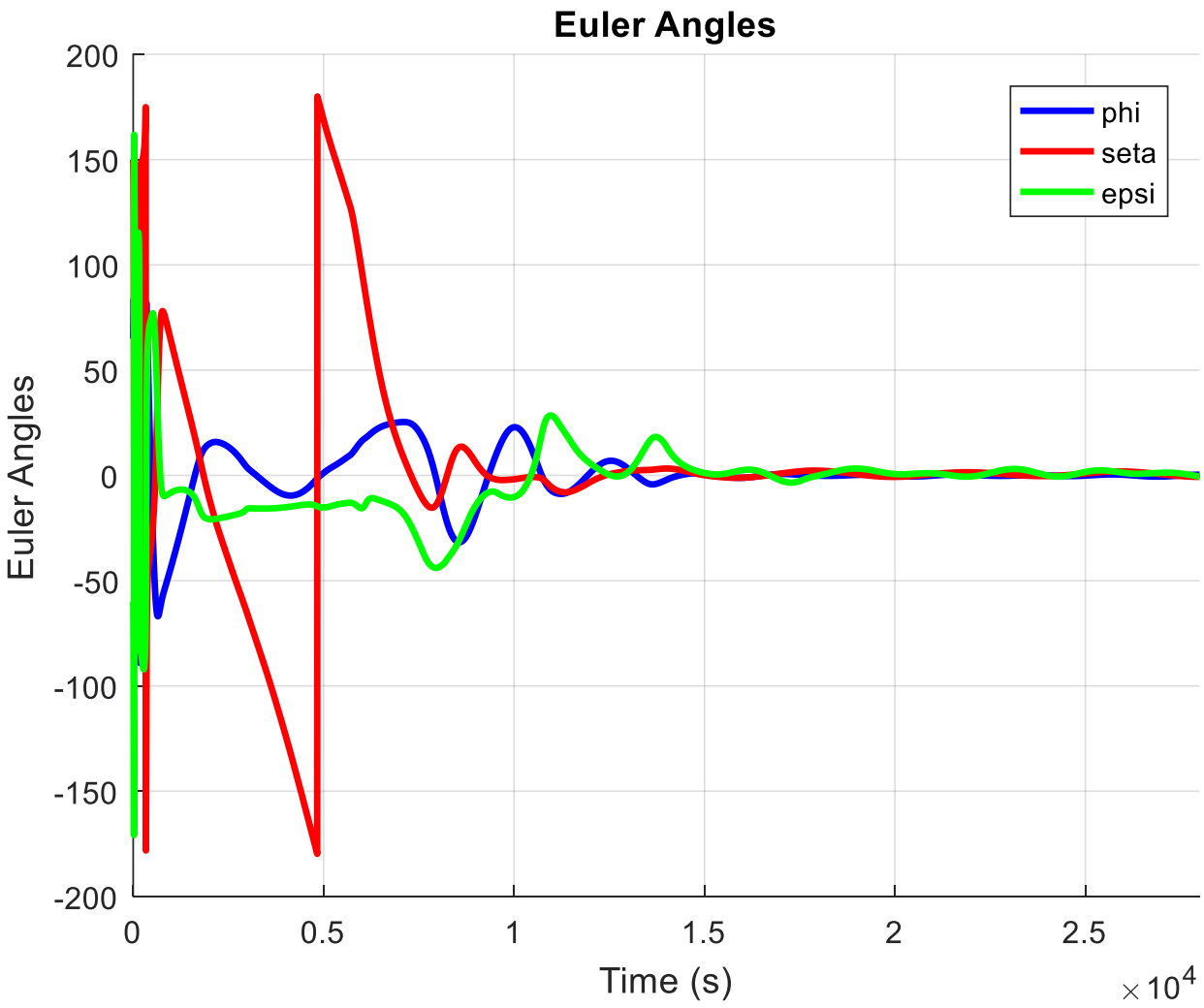
Attitude estimation algorism used here is Extended Kalman Filter described in a another section.

- ***Angular Velocity Response***



It's clear that the angular velocity settles on 0 for $\omega_x \; and \; \omega_z$ and ω_y settles on the orbit angular velocity, and oscillation of 0.004 deg/sec

- ***Euler Angles Response***



Where the angles oscillates with magnitude 1.2 degree.

Chapter 7

ADCS Software

The following figure represents the ADCS software algorithm used in the satellite, it depends on the following Parameters:

- Previous Control Action: To be used in the propagation of the states using Runge Kutta technique and produce the predicted states used in the Estimation Algorithm
- Universal Time: Also used for the calculations of the IGRF and the Gravity Model used in Runge Kutta for the Propagation of the States
- Previous Corrected States: Used in the determination of the Current mode of operation of the satellite
- Sensor Measurement: Used in the Estimation algorithm (EKF) as well as determining the current mode of operation

The ADCS is required to provide the in-orbit attitude control and determination functions. The attitude control function includes the stabilization and control of the satellite to the prerequisite attitude acquisition, while the attitude determination function is to facilitate the estimated attitude information.

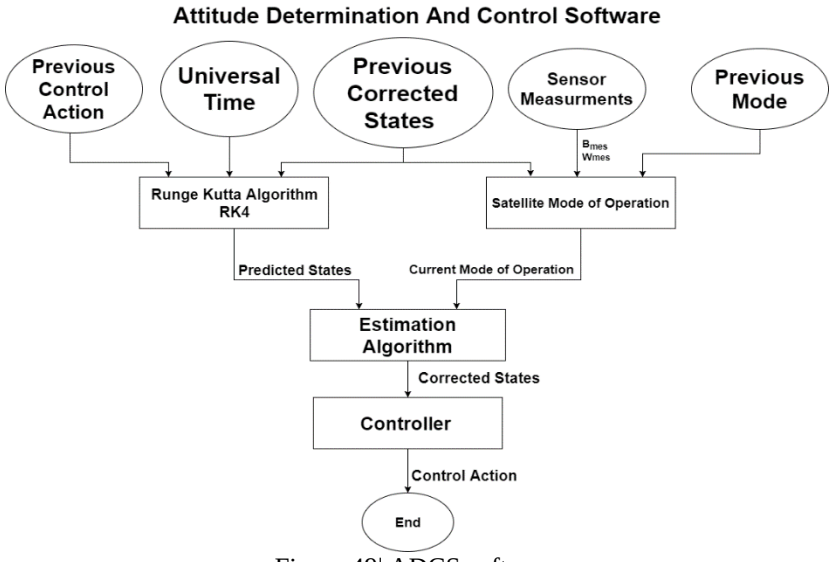


Figure 49| ADCS software

7.1 Satellite Modes of Operation

7.1.1 Detumbling Mode:

This mode is the phase just after leaving the launch vehicle in orbit. Initially, the spacecraft will be unstable prior to spinning, the main function of this mode is to suppress the satellite angular velocity from maximum $10^{\circ}/s$ down to two orbital angular velocity using three Magnetorquers and 3-axis magnetometer. In this mode Bdot controller is used to provide attitude stabilization based on the magnetometer readings only and no Estimation algorithm could be used due to the high-power consumption of the actuator in stabilizing the satellite.

- ***Attitude Determination Requirements***
  - Spin rate between $0.05^0/s$ ~ $0.06^0/s$
  - Accuracy: Spin rate must be sensed within $0.001^0/s$ accuracy
- ***Sensors***
  - Magnetometer
- ***Controller***
  - Bdot Control Algorithm
- ***Estimation Algorithm***
  - None

7.1.2 idle Mode:

This is also called preparation mode and it is directly after the detumbling mode and it lasts about 3000 secs, in this mode the satellite begins the estimations algorithms (Full – Kalman) to converge to a more reliable value of states like the angular velocity and quaternion to further used in the Re-Orientation mode. In the IDLE mode, a velocity feedback algorithm based on the gyro reading is used to maintain low angular rate and to prepare the satellite for the Re-Orientation mode. In this mode, both the Gyro and the Magnetometer would be online.

- ***Attitude Determination Requirements***
  - Provide More reliable readings
- ***Sensors***
  - Magnetometer
  - Gyro
- ***Controller***
  - Velocity Feedback Control Algorithm
- ***Estimation Algorithm***
  - Full- Kalman

7.1.3 Re-Orientation Mode:

ADCS will estimate angels using magnetometer and gyros to Re-Orient the satellite to the require attitude acquisition (Nadir pointing by default unless the Ground station sent different commands). In this mode, a Full -kalman estimation algorithm is used in the presence of a PD control algorithm to Re-Orient the satellite in a stable manner. Both the Gyro and Magnetometer shall be used.

- ***Attitude Determination Requirements***
  - All attitudes (meaning that any random attitude shall be sensed)
  - Spin rate between $0.5^0/s$ ~ $0.6^0/s$
  - Euler Angles: Less than 10^0 from the required attitude acquisition
- ***Sensors***
  - Magnetometer
  - Gyro
- ***Controller***
  - PD Control Algorithm

- ***Estimation Algorithm***
  - Full – Kalman

7.1.4 Standby Mode:

This will be the common mode through more than 80% of the satellite mission life, In this mode a B-Kalman is used as an estimation algorithm instead of the Full-Kalman for lower power consumption, a PD controller is used to stabilize the satellite at the attitude acquisition with a pointing accuracy of less than 10^0.

- ***Attitude Determination Requirements***
  - Spin rate between $0.5^0/s$ ~ $0.6^0/s$
  - Euler Angles: Less than 5^0 from the required attitude acquisition
- ***Sensors***
  - Magnetometer
- ***Controller***
  - PD Control Algorithm
- ***Estimation Algorithm***
  - B- Kalman

7.1.5 Pre-Imaging Mode:

This mode is quite similar to the standby algorithm except that it uses a more powerful estimation algorithm (Full- Kalman) and the gyro to provide a more accurate reading to the PD control algorithm so that the satellite can transition to the Imaging mode as fast possible. In other words, this mode is used to minimize the time spent in the standby mode after when the satellite receives telemetry from the ground station indicating an Imaging request.

- ***Attitude Determination Requirements***
  - Spin rate between $0.5^0/s$ ~ $0.6^0/s$
  - Euler Angles: Less than 5^0 from the required attitude acquisition
- ***Sensors***
  - Magnetometer
  - Gyro
- ***Controller***
  - PD Control Algorithm
- ***Estimation Algorithm***
  - Full- Kalman

7.1.6 Imaging Mode:

This mode is used to provide attitude acquisition with high pointing accuracy adequate for imaging process (i.e. accuracy less than ±5°), This accuracy is provided by the use of the powerful estimation algorithm (Full – Kalman) and a PD controller in the presence of the Magnetometer and gyro sensors.

- ***Attitude Determination Requirements***
  - Spin rate between $0.5^0/s$ ~ $0.6^0/s$
  - Euler Angles: Less than 2^0 from the required attitude acquisition
- ***Sensors***
  - Magnetometer
  - Gyro
- ***Controller***
  - PD Control Algorithm
- ***Estimation Algorithm***
  - Full- Kalman

7.1.7 Satellite Modes of operation Flow chart

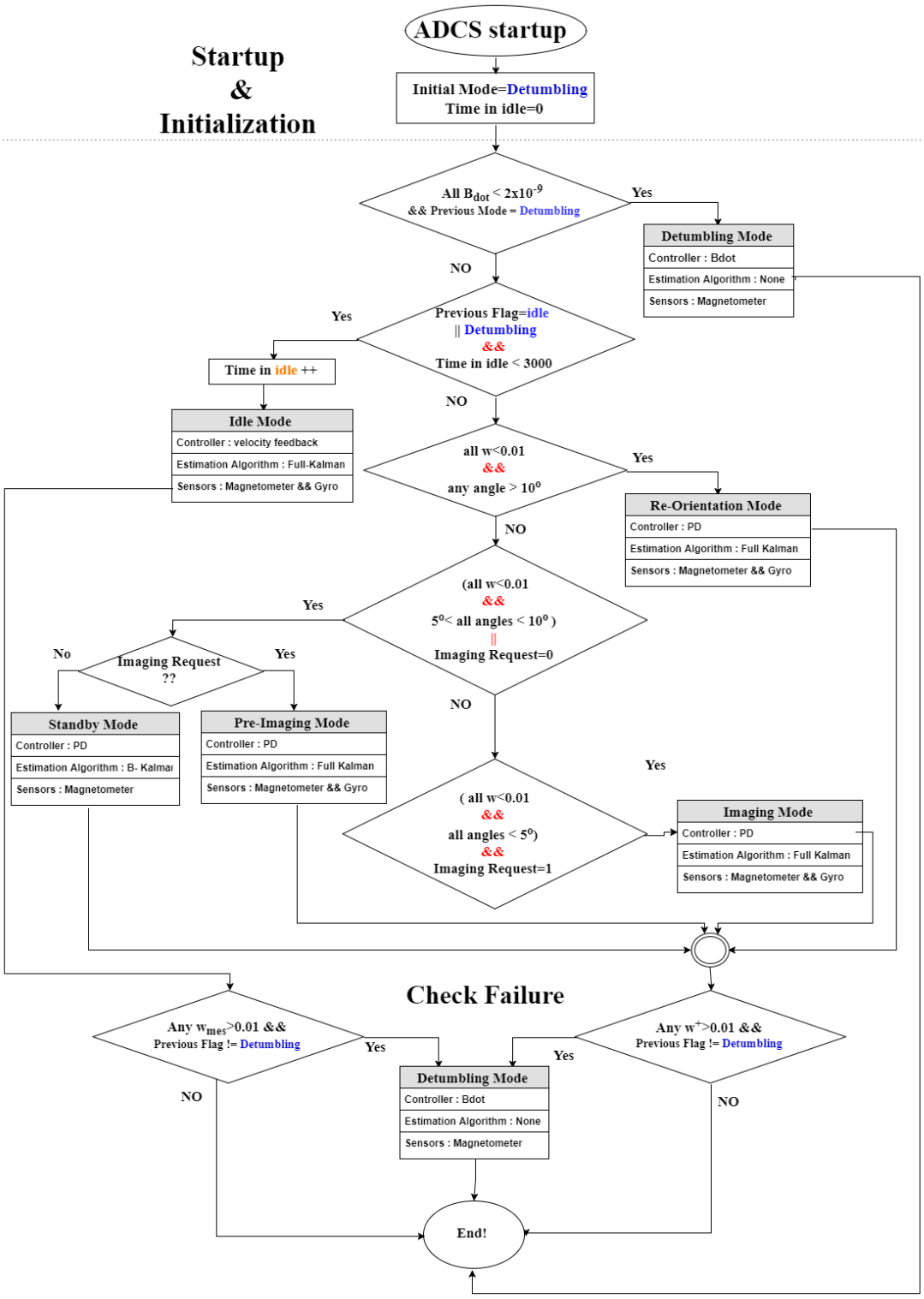


Figure 50| Satellite mode of operation

7.2 Runge Kutta Algorithm (RK4)

RK is one of the most well-known ways to solve initial value problems like what we are dealing with in this model. Runge Kutta of 4th order which is based on a routine called the Euler Method used in temporal discretization for the approximate solutions of ordinary differential equations.

The Runge-Kutta ('RK') methods were originally developed by the German mathematicians Carle Runge (1856-1927) and Martin Kutta (1867-1944). In the explicit, single step RK methods, y_{i+1} at $t_i + h$ is obtained from $y_i$at t_i by the formula

$$y_{i+1} = y_i + h\phi(t_i, y_i, h)$$

The increment function ϕ is an average of the derivative $\frac{dy}{dt}$ over the time interval t_i to $t_i + h$. This average is obtained by evaluating the derivative $f(t, y)$ at several points or "stages" within the time interval. The order of an RK method reflects the accuracy to which ϕ us computed, compared to a Taylor series expansion. An RK method of order p is called an RK_pmethod. An RK_pmethod is as accurate in computing y_i from the previous equation as is the p^{th} order Taylor series.

$$y(t_i + h) = y_i + c_1 h + c_2 h^2 + \cdots c_p h^p$$

The higher the RK order, the more stages there are and the more accurate is. The number of stages equals the order of the RK method if the order is <5.

The RK4 method is a fourth-order method which means that the local truncation error is on the order of O(h5), while the total accumulated error is on the order of O(h4).

It can be formulated in vector form as following:

$$\tilde{t}_m = t_i + a_m h \quad m = 1, 2, \cdots, s$$

$$\tilde{\mathbf{f}}_m = \mathbf{f}(\tilde{t}_m, \tilde{\mathbf{y}}_m) \quad m = 1, 2, \cdots, s$$

$$\tilde{\mathbf{y}}_m = \mathbf{y}_i + h \sum^{m-1} b_{mn} \tilde{\mathbf{f}}_n \quad m = 1, 2, \cdots, s$$

$$\mathbf{y}_{i+1} = \mathbf{y}_i + h \sum_{m=1}^{s} c_m \tilde{\mathbf{f}}_m$$

Where in case of RK4

$$\{a\} = \begin{Bmatrix} 0 \\ 1/2 \\ 1/2 \\ 1 \end{Bmatrix} \quad [b] = \begin{bmatrix} 0 & 0 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \{c\} = \begin{Bmatrix} 1/6 \\ 1/3 \\ 1/3 \\ 1/6 \end{Bmatrix}$$

Runge Kutta Algorithm

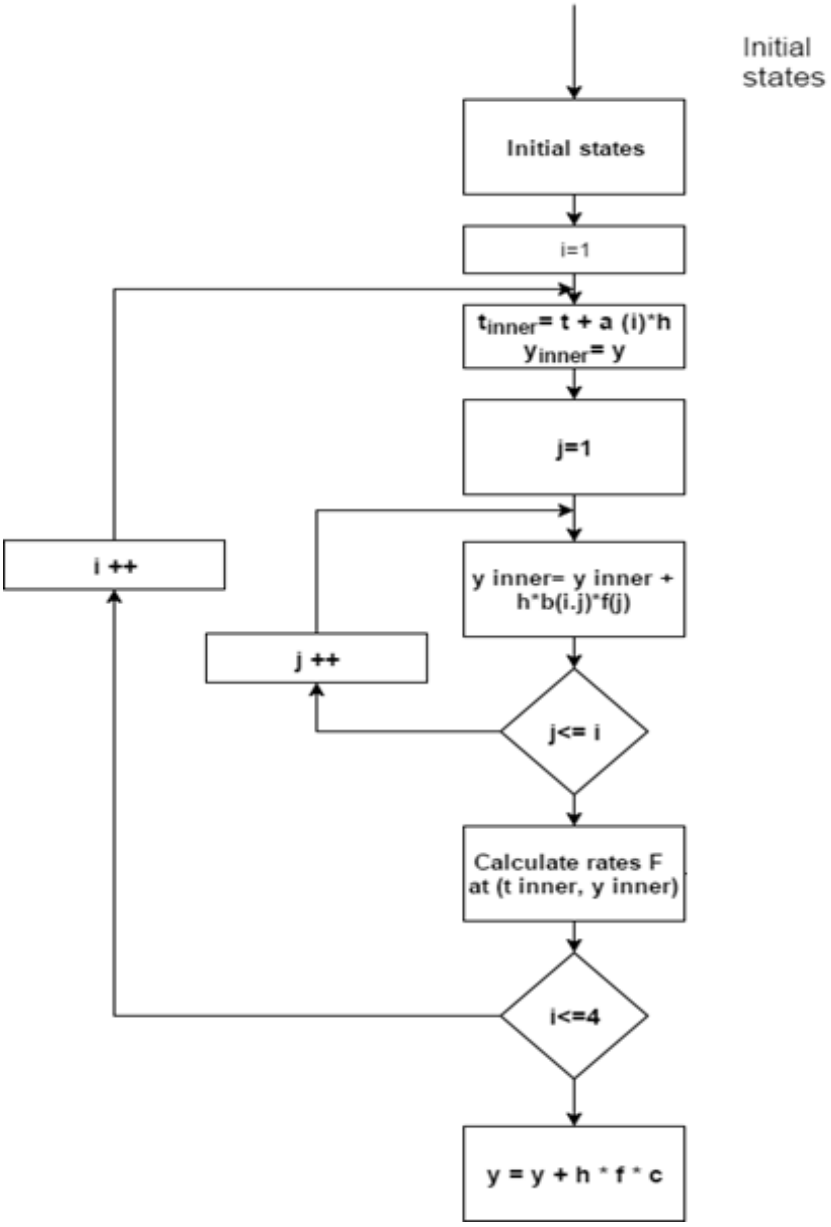


Figure 51|RK flow chart

7.3 Differential Equation

The differential equation is used to calculate the rates of the states used in Runge Kutta technique (RK4) to propagate the state to the next time step.

Firstly it gets the Sidereal Time through the Position vector in Greenwich Coordinates and the universal time to be used in the calculations of the IGRF model, then the resulting magnetic field is transformed to the body coordinates to be used in the calculations of the Magnetic disturbance, Secondly, we get the Sun vector of Earth using the Universal model and subtract it from the Position vector of the satellite to the satellite sun vector , then it is transformed to body axes to be used in solar disturbance calculation. The rate of change of the linear velocity can be obtained using the gravity model (Spherical harmonics) after transforming it to inertial coordinates. Furthermore, the rate of change of the angular velocity and Quaternion can be obtained by the use of the satellite dynamics and Kinematics equations.

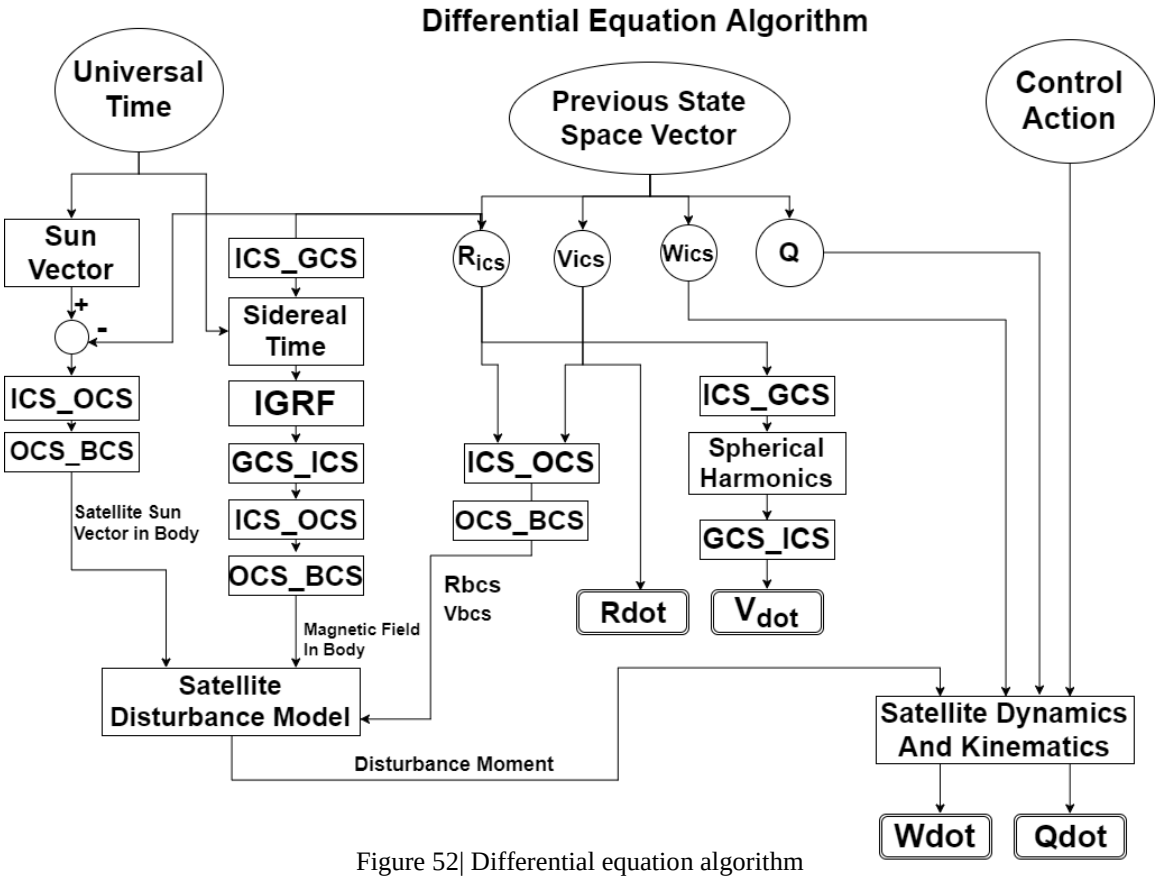


Figure 52| Differential equation algorithm

7.4 Estimation Algorithms

The observer mode of operation is used to determine which Estimation Algorithm to be used to get more reliable readings depending on the current mode of operation of the satellite

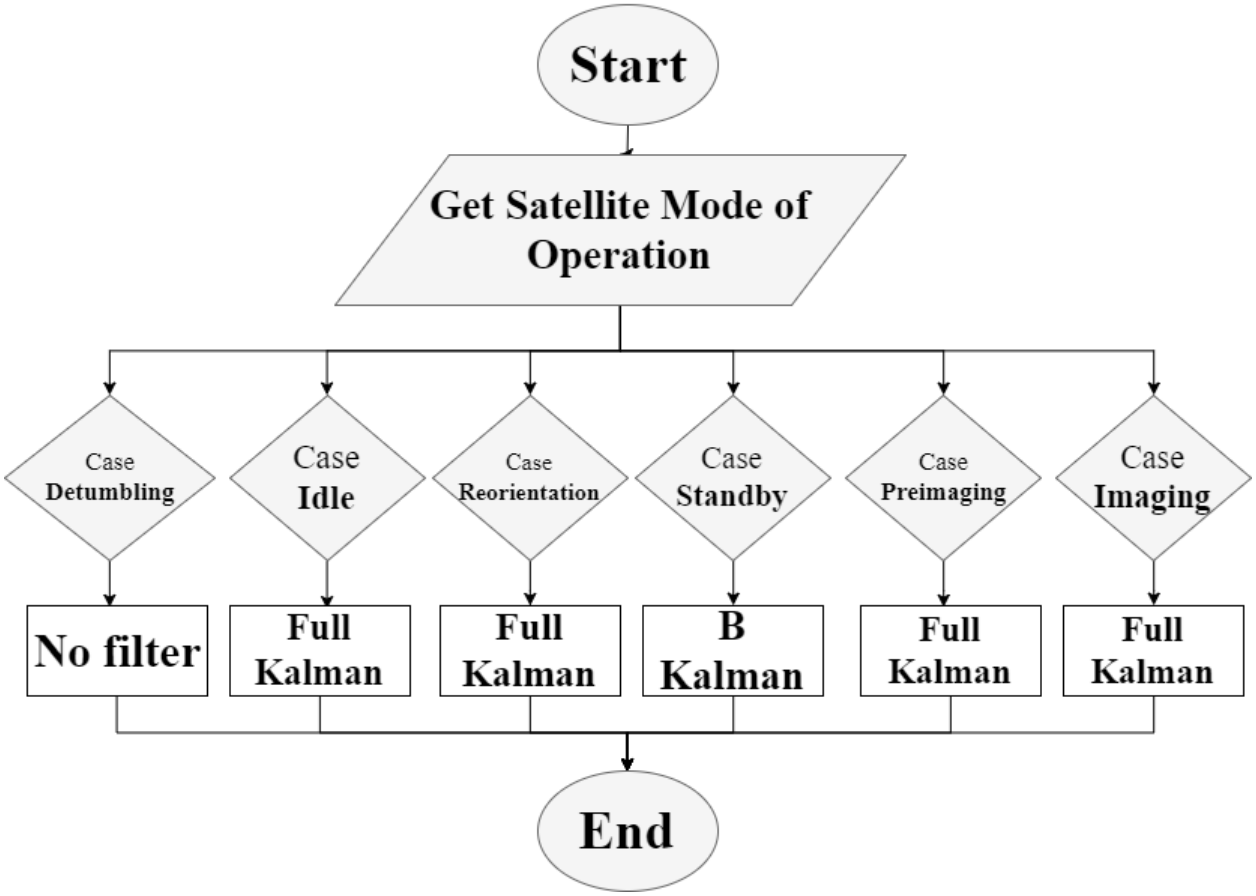


Figure 53| Estimation mode of operation

7.5 Controller

The Controller mode of operation is used to determine the suitable control Algorithm to provide stabilization and attitude acquisition of the satellite depending on the current mode of operation of the satellite

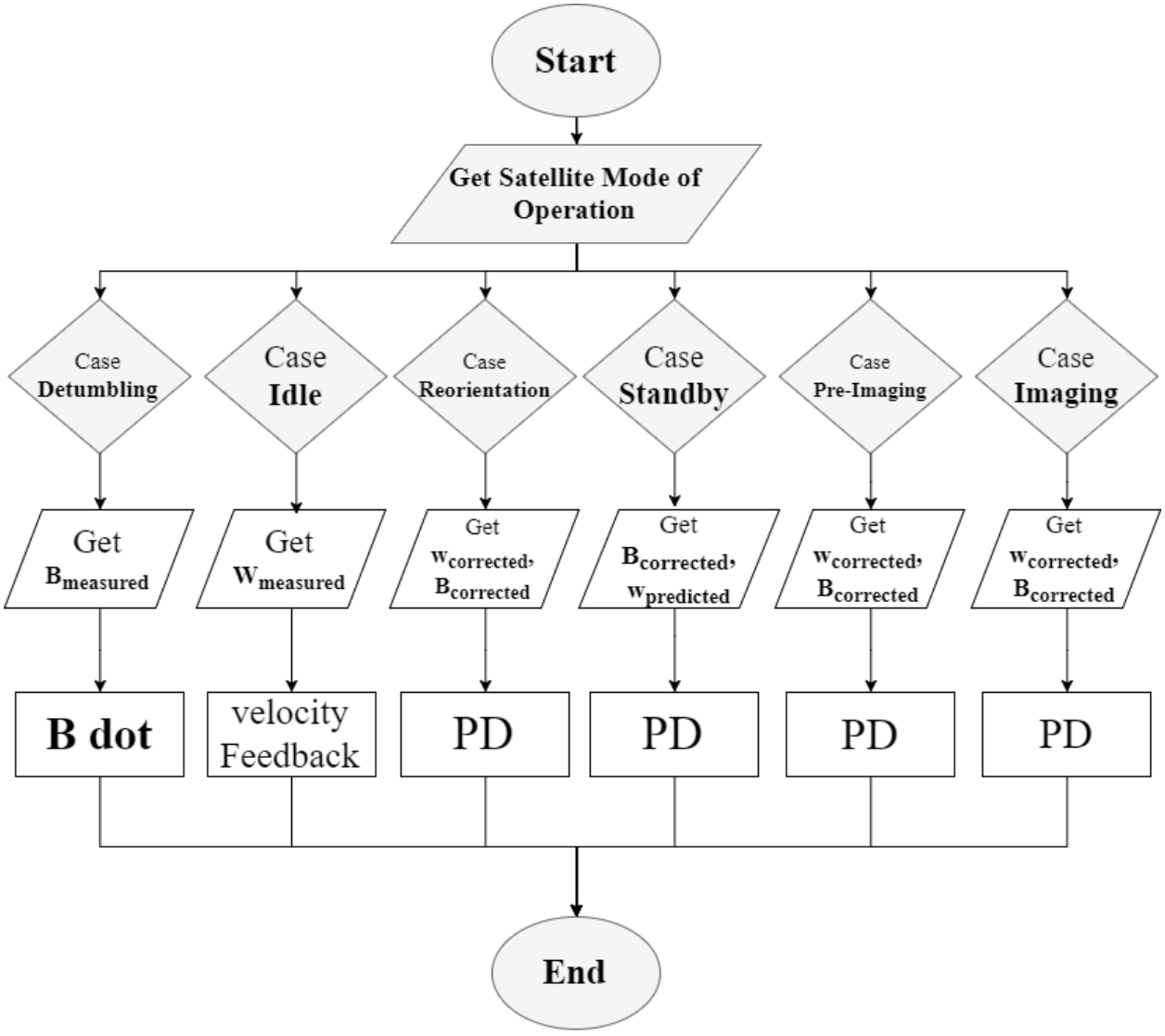


Figure 54| Controller mode of operation

7.6 Software in the Loop (SIL)

SIL is on a mathematical model, in theory, to complete the preliminary microsatellite ADCS design. The SIL platform contains a dynamic simulator and a real-time microcontroller, as well as some interfacing circuitry. The dynamic simulator is capable of performing simulation of the space environment, orbit dynamics, attitude dynamics, and sensor/actuator models. The controller/estimator of ADCS is a realization of the embedded microcontroller for attitude determination and control

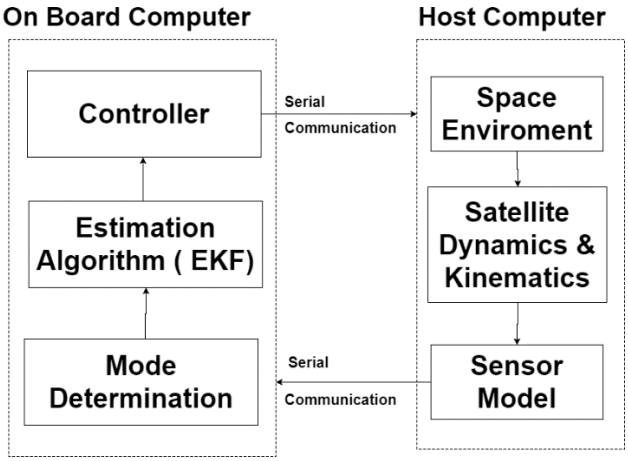


Figure 55| (SIL) algorithm

The SIL test of microsatellite ADCS is conducted to verify the function of each module. To this end, the MATLAB with is used to facilitate the simulation

SIL system is divided into two parts: one is for the host computer and the other is the on-board computer, Serial Communication between the Host computer and on-board computer with used for satellite attitude simulation tests. Host computer simulates microsatellite dynamic equations of motion, in orbit on the environment and attitude measurement sensing element mathematical model. The on-board computer not only is a real-time hardware system to replace the satellite attitude controller but also simulated satellite attitude control element mathematical model. Host computer is connected with the on-

board computer through the Serial connection into a loop, they are linked for the satellite attitude (angle, angular velocity) and a control command. The SIL test can be conducted in a subsystem level or closed-loop system level. In SIL testing, the system under consideration is composed of six functional blocks as depicted in the figure. The function of each block is described as follows

- Space Environment: it represents the Attitude disturbance the satellite is exposed to and the Gravitational Pull of earth as well as its magnetic field
- Satellite Dynamics and Kinematics: It provides three main functions including the computation of satellite attitude and angular rates, satellite position and velocity, and coordinate frame transformation
- Sensor Model: Two types of sensors are modelled and simulated in this block, including magnetometer and gyro.
- Mode Determination: To determine which is the current mode of operation of the satellite
- Estimation Algorithm: This block provides the function of attitude estimation by using the Extended Kalman Filter
- Controller: This block executes the attitude control function

After the simulation of ADCS in software, the ADCS algorithm has to be implemented in the ADCS on board microcontroller. To fully develop a verifiable ADCS algorithm, a processor-in-the-loop (PIL) test is developed. PIL test is a way to bridge the gap between the simulation results and the final system construction. It increases the realism of the simulation and provides access to hardware features that are currently not available in the software-only simulation. PIL enables the testing of the actual control and estimation codes running on the microcontroller in the verification environment

Chapter 8

Full Mission Simulation Results

This simulation is used to test the serviceability of the ADCS software described in above section, and make sure that it satisfy the requirement of accuracy and time of each reaching the required attitude after separation from the launcher.

8.1.1 Satellite and simulation parameters

Inertia parameters					
Ixx	0.06 Kg.m^2	Ixy	0		
Iyy	0.08 Kg.m^2	Iyz	0		
Izz	0.004 Kg.m^2	Ixz	0		
Satellite Dimensions					
x	10 cm	y	10 cm	z	20 cm
Orbit parameters					
Semi major axis	7046.1 Km	eccentricity	0		
inclination	98.085°	Right ascension	301.643°		
Argument of perigee	291.1406°	True anomaly	68.859°		
Actuator parameters					
Maximum dipole moment	0.076 A.m^2	dead zone	0	efficiency	1
Initials					
UT1	2014, 08, 15 09:48:58.62				
Angular velocity	[10,10,10] degree / sec				
Euler angles	$\phi = 65°,$ $\theta = -150°,$ $\psi = 140°$				
Disturbance parameters					
Residual magnetic field	0.0006 A.m^2	Aerodynamic drag coeff.	2		
Simulation time	50,000 sec	Time step	4 sec		

8.1.2 Simulation Results

- ***Angular velocity***

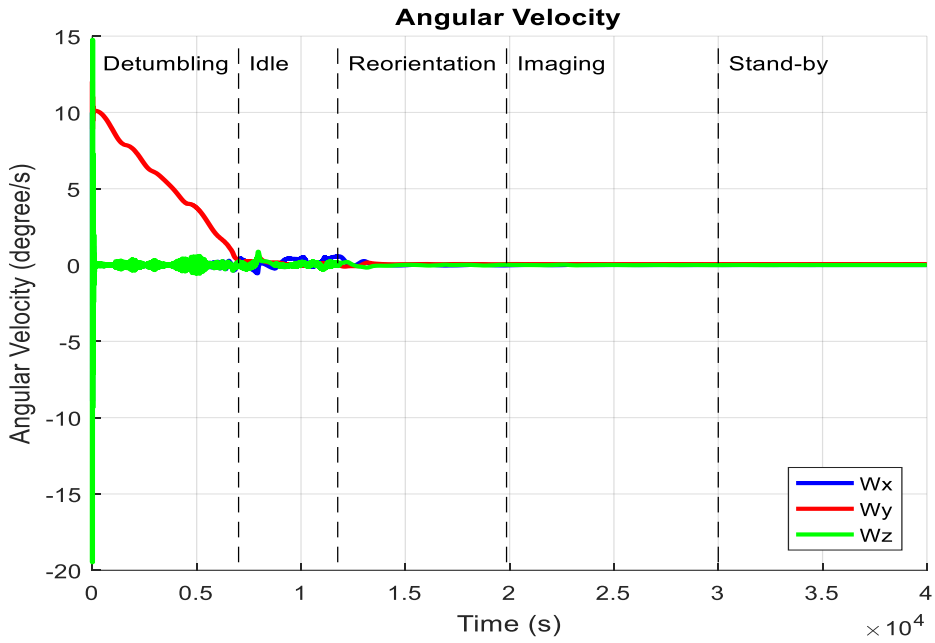


Figure 56| angular velocity Response

- ***Euler angles***

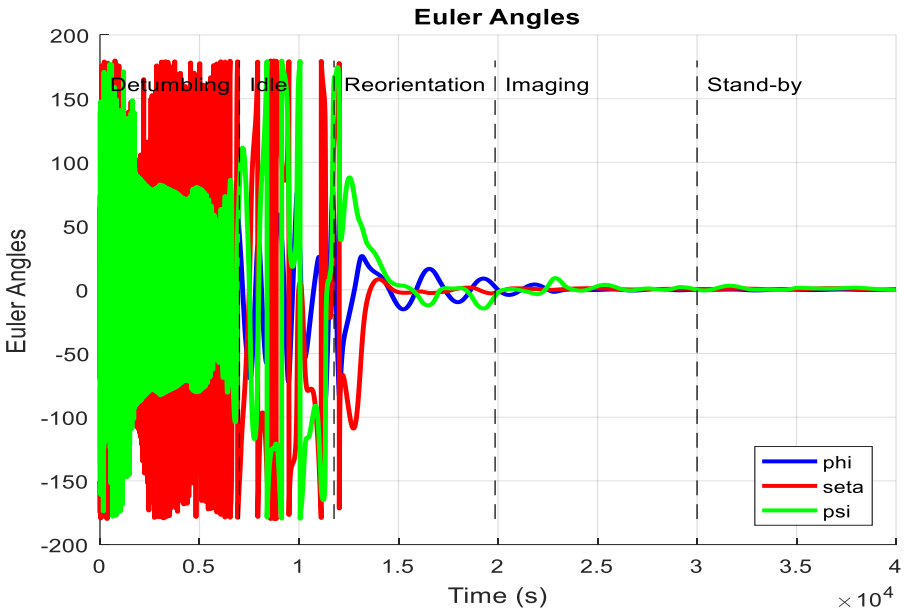


Figure 57| Euler angles response

- ***Quaternion***

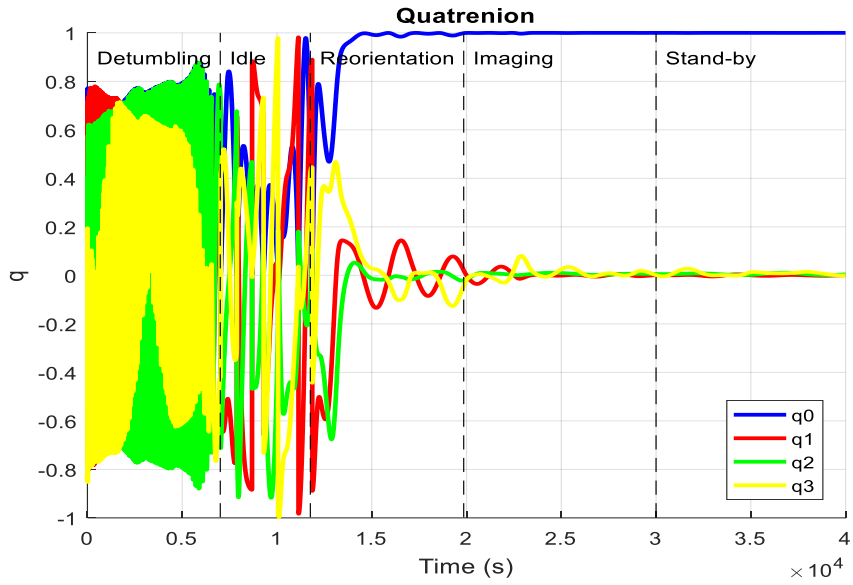


Figure 58| Quaternion Response

- ***Control Moment***

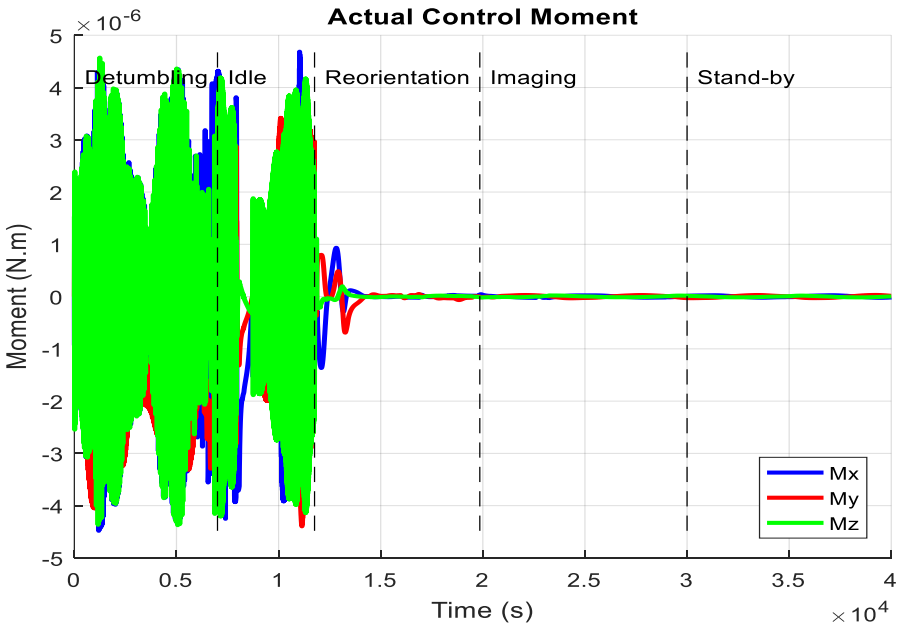


Figure 59| control moment response

- ***Angular Velocities Estimation error***

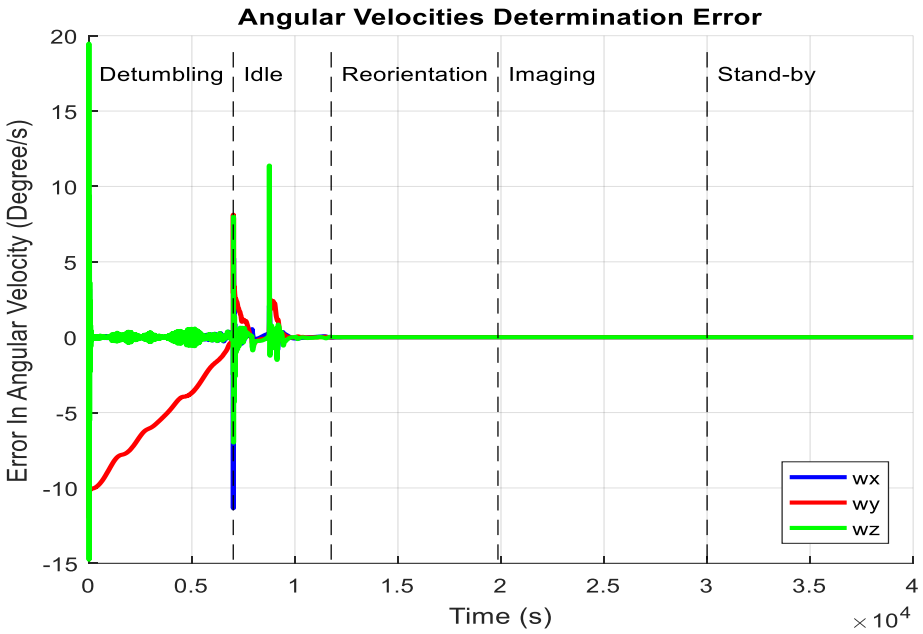


Figure 60| Angular Velocity estimation error

- ***Euler angle estimation error***

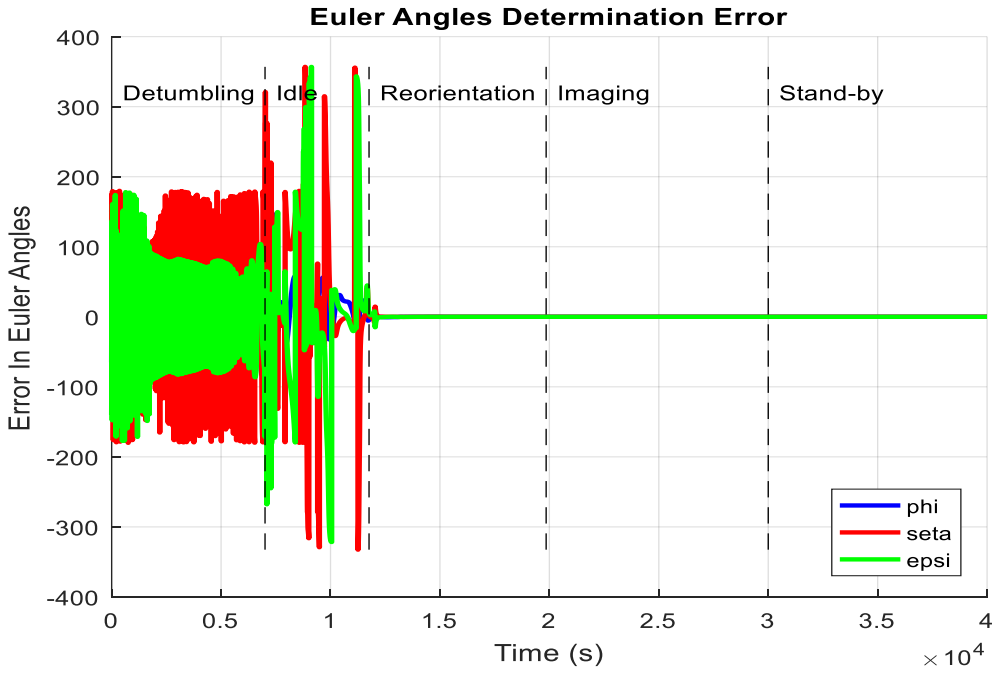


Figure 61| Euler angles estimation error

- ***Required control action (dipole moment)***

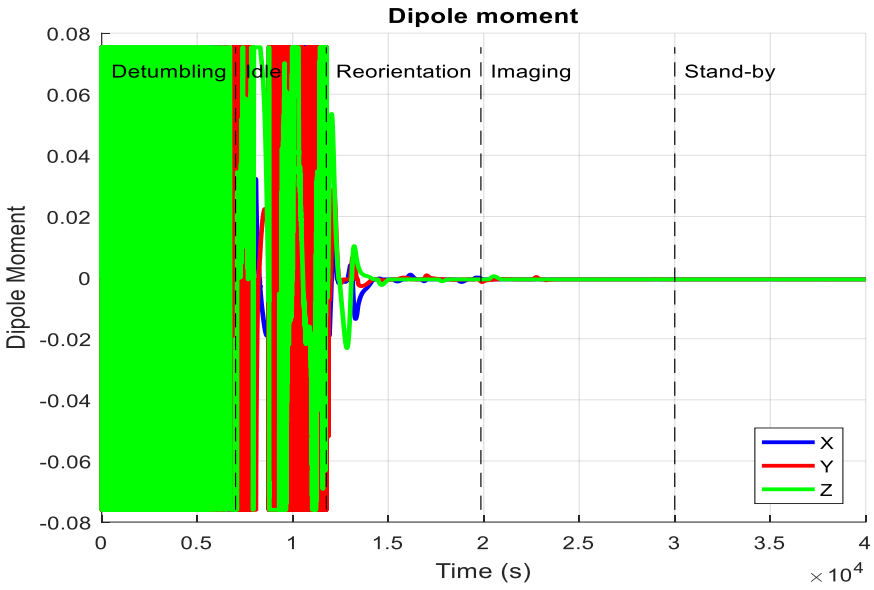


Figure 62| control action (dipole moment)

From these results it's clear that the ADCS software model performed as required and stabilized the satellite from it's limiting angular velocities, then it went through idle mode to initialize the estimator, then through attitude accusation, until it reached imaging and stabled on it until stop imajing order is received then it returned back to stand by mode.

By investigating these results we can find how our ADCS software trace the system requirements presented in the introduction.

From the previous results we can find the following summary

1. Detumbling time after 7008 sec
2. Reorientation time after 19852 sec
3. Imaging attitude max error < 1 degree
4. Stand-by attitude max error < 1.5 degree

8.1.3 Requirements tracing

parameter	requirement	resulted
Time to achieve reorientation from initial angular velocity of 10 deg/sec in all direction	10 orbital periods (54,000) sec	3.67 orbits
Max error during imaging	<5°	1 °
Max error during stand by	<10 °	1.5°

8.2 Software in the loop testing results

A software testing is also conducted to make sure that the Embedded C software works properly as the matlab model code of the ADCS software.

After running the simulation with the C code it showed error of 10^{-8} than the matlab simulation after simulation time of 50,000 sec.

Chapter 9

Satellite Flight Simulator

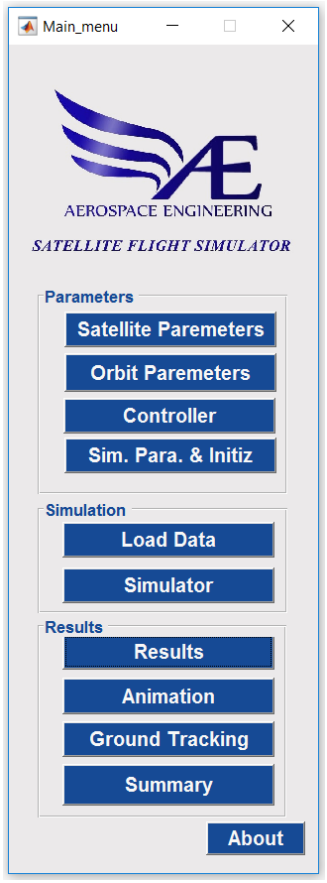


Figure 63| Flight Simulator in Main Menu

Satellite Flight Simulator has been designed for CubeSat. A CubeSat (U-class spacecraft) is a type of miniaturized satellite for space research that is made up of multiples of 10×10×10 cm cubic units.

In the following, a User Guide document for the Graphical User Interface of the satellite calculation, simulation, Animation and results summary. It's consider a user friendly step-by-step for easy usage of the application.

Graphical User Interface (GUI) of the Satellite Flight Simulator of Aerospace Engineering Department, Cairo University is based on MatLab (The MathWorks, Inc.).

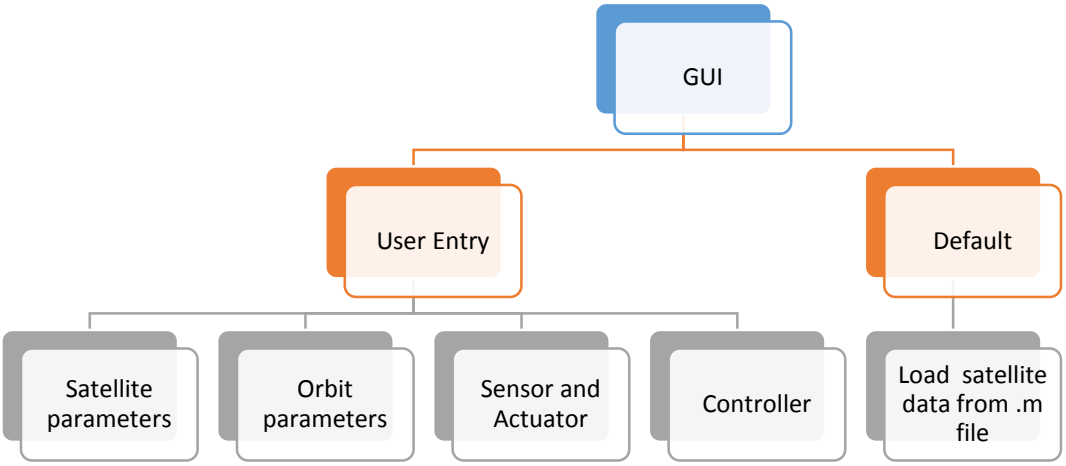


Figure 64| Parameters Entry strategy

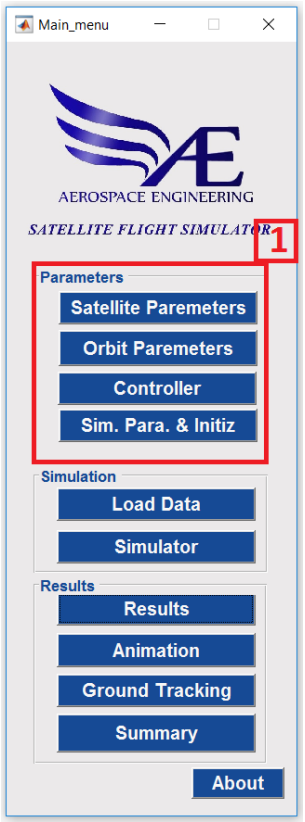


Figure 65| Satellite Parameters in main menu figure

9.1 Characteristics of the Satellite, Orbit and Initialization:

9.1.1 Satellite Parameters:

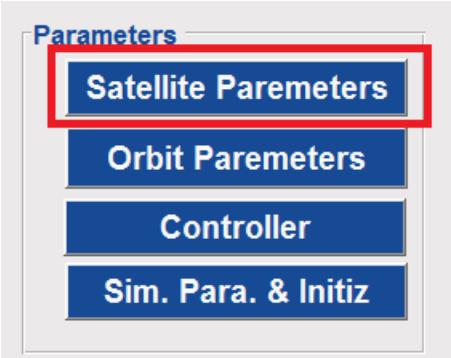


Figure 66| Satellite Parameters button

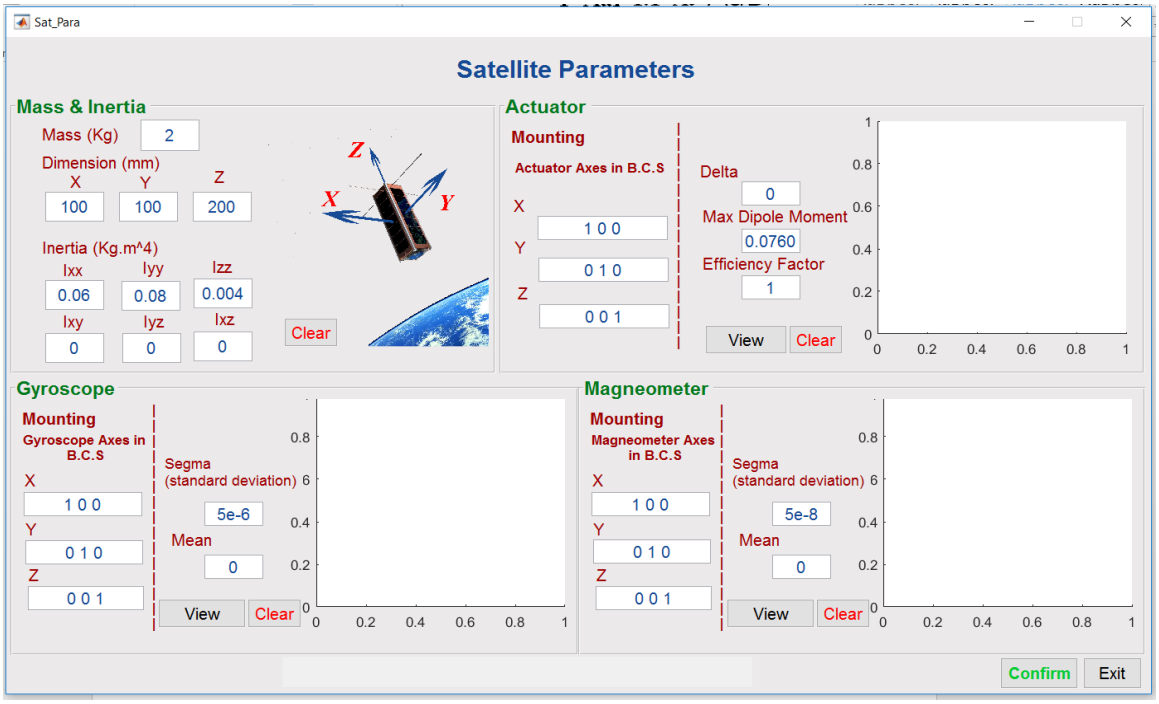


Figure 67| Satellite Parameters figure

Satellite Parameters can be categorized into:

- Mass, Dimension and Inertia
- Actuator Characteristics
- Gyroscope Characteristics
- Magnetometer Characteristics

1.1.1- Mass and Inertia:

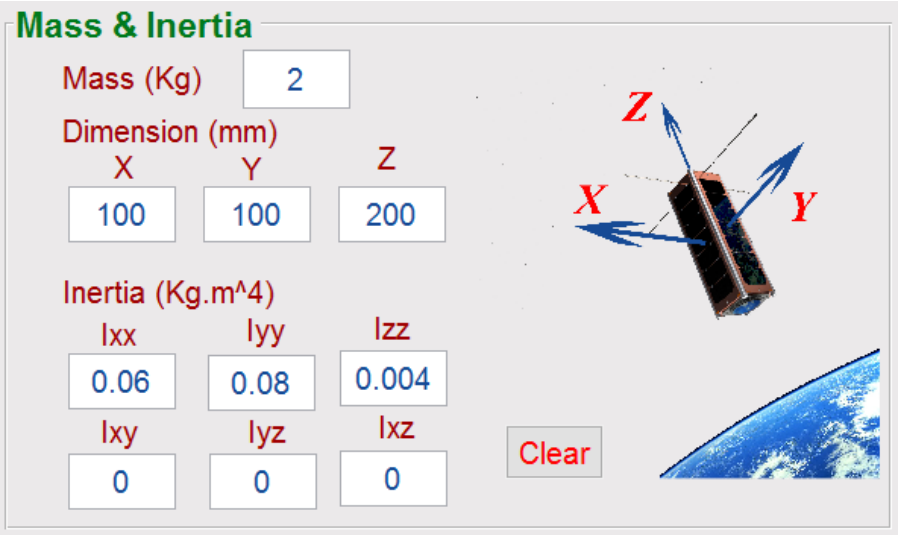


Figure 68| Mass and Inertia panel

1.1.1-a- Satellite Mass:

It represents the satellite total mass, containing all subsystems. Mass must be in unit of Kg.

1.1.1-b- Satellite dimensions:

It represents the size of the satellite (External dimensions), from edge to edge. Dimensions must be in unit of mm^4. The three dimensions are: in X direction, in Y direction and in Z direction. The following figure shows details of the dimension:

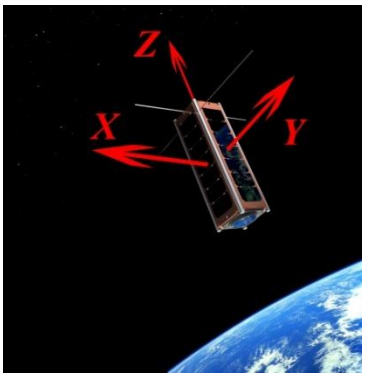


Figure 69| Satellite Dimension

1.1.1-c- Satellite Inertia:

It represents the Moment of Inertia of the Satellite ($I_{xx}, I_{yy}, I_{zz}, I_{xy}, I_{yz}, I_{xz}$), in unit of ($Kg.m^4$). For symmetry, ($I_{xy} = 0, I_{xz} = 0, I_{yz} = 0$).

For calculation of the satellite's moment of inertia:

$$I_{xx} = \int (y^2 + z^2)dm = \int (y^2 + z^2)dydz \qquad I_{yy} = \int (x^2 + z^2)dm = \int (x^2 + z^2)dxdz$$

$$I_{zz} = \int (x^2 + y^2)dm = \int (x^2 + y^2)dxdy \qquad I_{xy} = \int xydm = \int xydxdy$$

$$I_{yz} = \int yzdm = \int yzdydz \qquad I_{xz} = \int xzdm = \int xzdxdz$$

1.1.2- Actuator

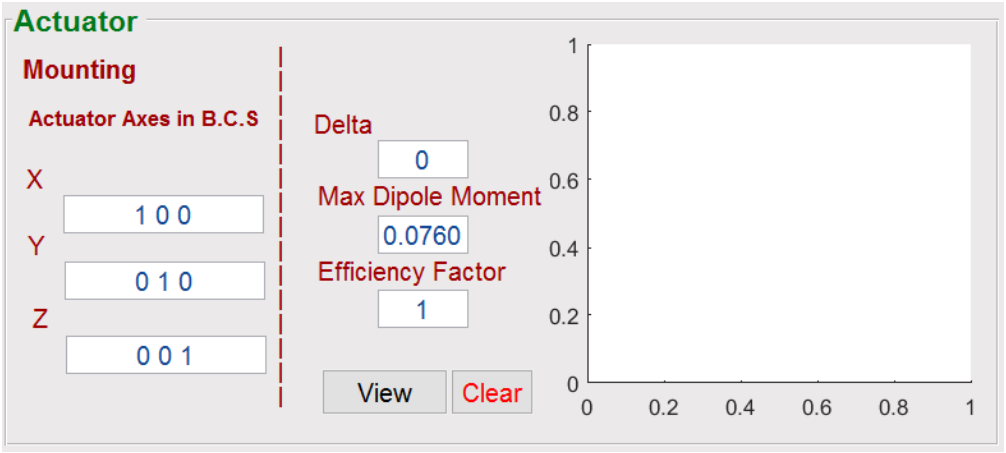


Figure 70| Actuator Panel

1.1.2-a- Mounting Matrix of Actuator:

It represents the unit matrix of the axes conventions of the actuator in Body Coordinates System. Axes conventions are standardized ways of establishing the location and orientation of coordinate axes for use as a frame of reference. Axes convention (Mounting Matrix) is 3×3 matrix, each row vector represents the convention of the actuator coordinates system in the direction in Body Coordinates System. Each vector should be entered in elements shape separated by space.

1.1.2-b- Delta:

It represents the dead zone of the characteristics equation of the actuator. It can be considered as a percentage of the maximum Dipole Moment. It must be in range of 0→1.

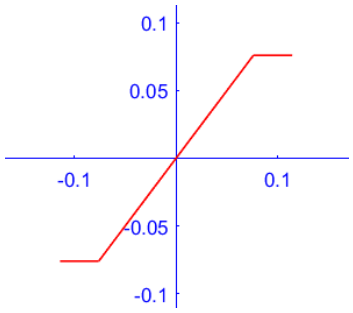


Figure 71| Zero dead zone

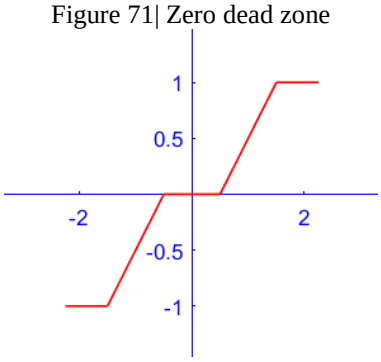


Figure 72| 50% dead zone

1.1.2-c- Maximum Dipole Moment:

It represents the saturation value of the magnetic torquer, it's unit is in $A.m^2$

1.1.2-d- Efficiency Factor

It represent a ratio of the moment generated in the sensor material to the magnetic field surround the sensor. It can be simplified to the ratio of the output to the input. It must be in range of 0→1. Higher efficiency factor, higher accuracy to measure the magnetic field.

1.1.3- Gyroscope Sensor:

The characteristics of the Gyroscope follow the normal distribution. View button for displaying the graph of the normal distribution of the Gyroscope.

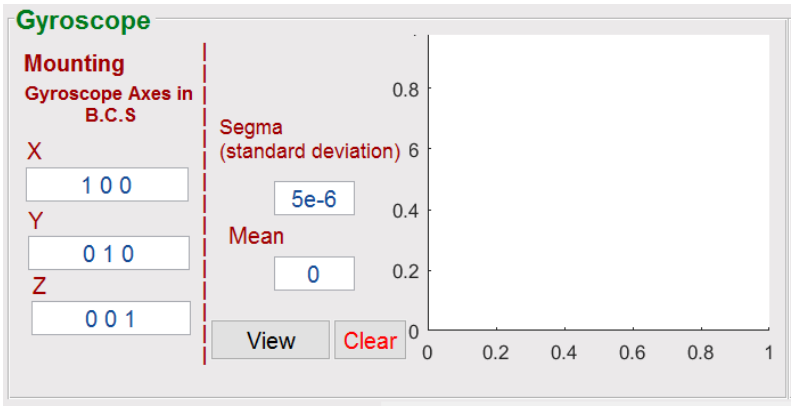


Figure 73| Gyroscope panel

1.1.3-a- Mounting Matrix of Gyroscope

It represents the unit matrix of the axes conventions of the Gyroscope in Body Coordinates System. Axes convention (Mounting Matrix) is 3×3 matrix, each row vector represents the convention of the gyroscope coordinates system in the direction in Body Coordinates System. Each vector should be entered in elements shape separated by space.

1.1.3-b- Sigma (Standard Deviation σ):

It represents the noise the Gyroscope affected by. Deviation is an expression of how the variable is distributed around the mean value. The standard deviation is the deviation converted to a standard format where the mean has the value "0" and the deviation has the value "1".

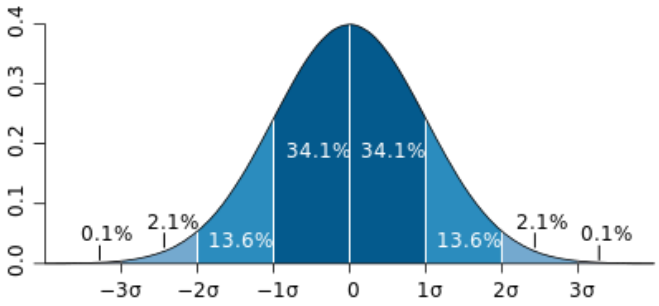


Figure 74| Normal distribution

https://en.wikipedia.org/wiki/Standard_deviation

1.1.3-c- Mean (μ):

It represents the expected value of the random variables. As the standard deviation therefore is simply a scaling variable that adjusts how broad the curve will be, though it also appears in the normalizing constant, so $\mu = 0$.

1.1.4- Magnetometer Sensor:

The characteristics of the Magnetometer follow the normal distribution. View button for displaying the graph of the normal distribution of the Magnetometer.

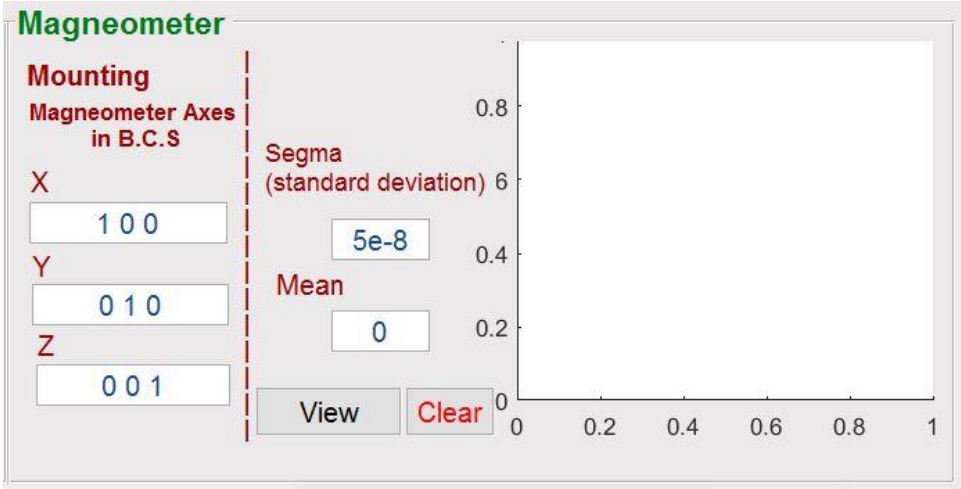


Figure 75| Magnetometer panel

1.1.3-a- Mounting Matrix of Magnetometer

It represents the unit matrix of the axes conventions of the Magnetometer in Body Coordinates System. Axes convention (Mounting Matrix) is 3×3 matrix, each row vector represents the convention of the Magnetometer coordinates system in the direction in Body Coordinates System. Each vector should be entered in elements shape separated by space.

1.1.3-b- Sigma (Standard Deviation σ):

It represents the noise the Magnetometer affected by.

1.1.3-c- Mean (μ):

It represents the expected value of the random variables.

9.1.2 Orbit Parameters:

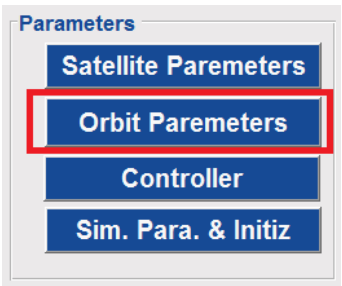


Figure 76| Orbit Parameters button

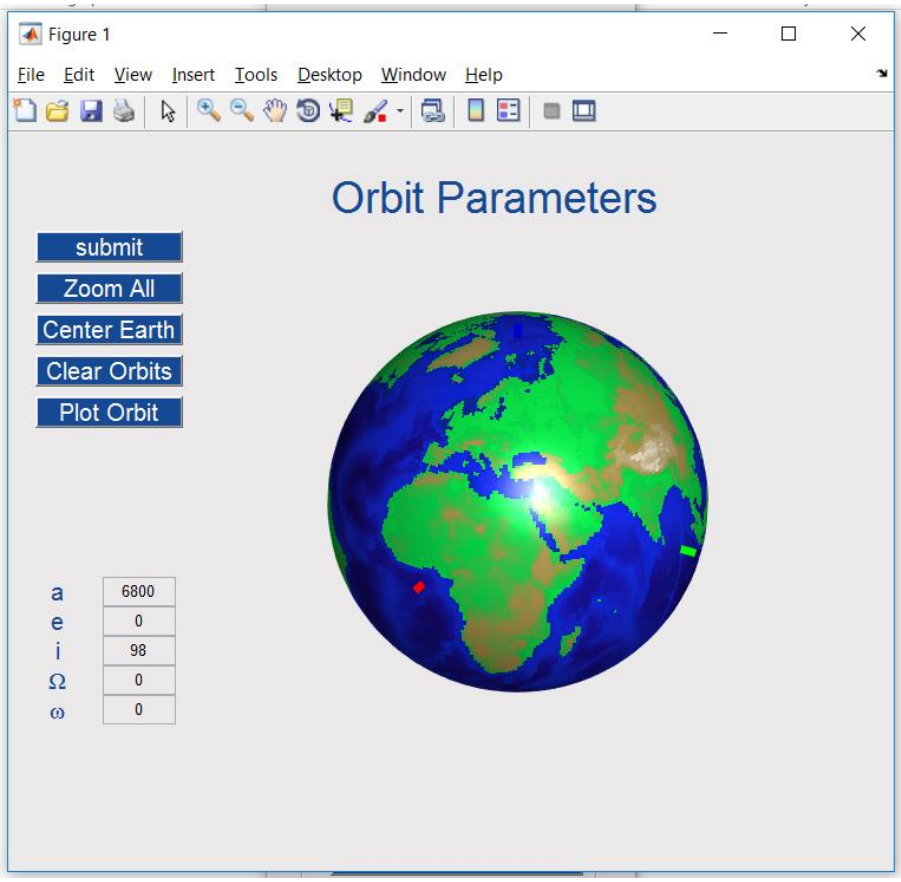


Figure 77| Orbit Parameters figure

Orbit Parameters can be categorized into:

- Orbit parameters entry
- Figure Buttons
- Earth (3-D) orbit Simulation

1.2.1- Orbit Parameters Entry (Kepler Parameters):

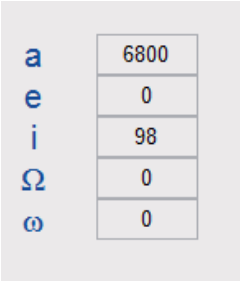


Figure 78| Orbit parameters entry

1.2.1-a- Semi-Major axis:

The semi-major axis of a hyperbola is, depending on the convention, plus or minus one half of the distance between the two branches. Thus it is the distance from the center to either vertex (turning point) of the hyperbola. Semi Major axis must be in unit of Km.

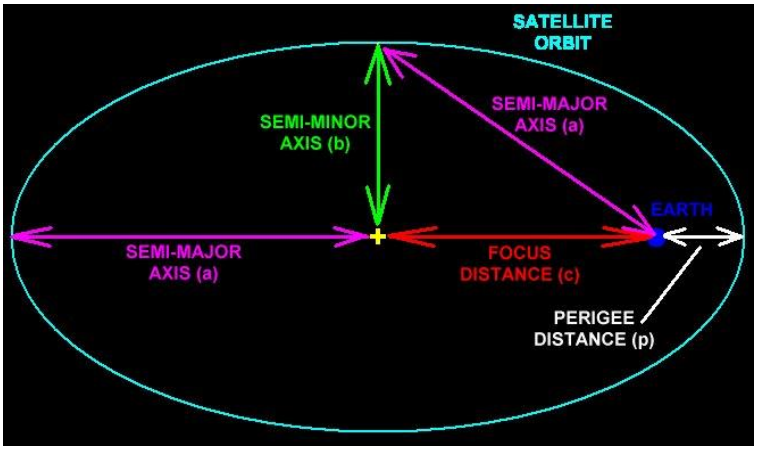


Figure 79| Semi Major Axis

http://www.castor2.ca/03_Mechanics/01_Basics/01_Parameters/index.html

1.2.1-b- Eccentricity (e):

It represents the orbital eccentricity of an astronomical object is a parameter that determines the amount by which its orbit around another body deviates from a perfect circle. It has no unit.

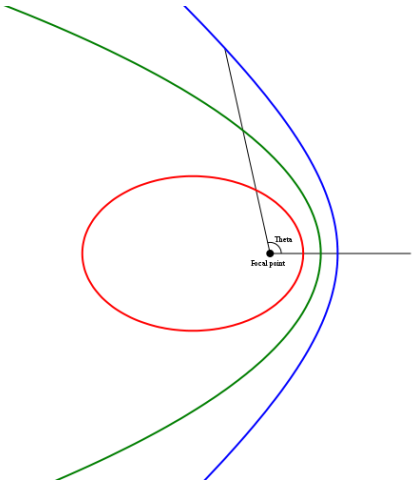


Figure 80| elliptic (red) (e = 0.7), parabolic (green) (e = 1), hyperbolic orbit (blue) (e = 1.3)

https://en.wikipedia.org/wiki/File:Kepler_orbits.svg

1.2.1-c- Orbit Inclination (i):

It represents the measurements of tilt of an object's orbit around a celestial body. It is expressed as the angle between a reference plane and the orbital plane or axis of direction of the orbiting object. Orbit Inclination must be in unit of degree (°).

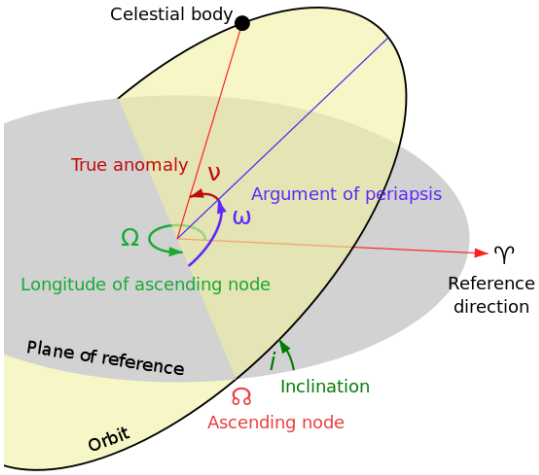


Figure 81| Orbit Inclination

https://en.wikipedia.org/wiki/Orbital_inclination#/media/File:Orbit1.svg

1.2.1-d- Right Ascension (Ω):

It represents the angular distance measured eastward along the celestial equator from the vernal equinox to the hour circle of the point in question. Right Ascension must be in unit of degree (°).

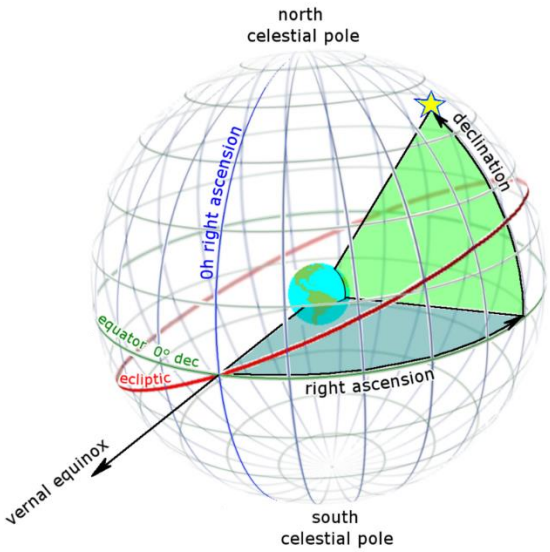


Figure 82| Right Ascension

https://en.wikipedia.org/wiki/Right_ascension#/media/File:Ra_and_dec_on_celestial_sphere.png

1.2.1-e- Argument of perigee(ω):

It represents the angle within the satellite orbit plane that is measured from the Ascending Node (ω) to the perigee point (p) along the satellite's direction of travel. Argument of perigee must be in unit of degree (°).

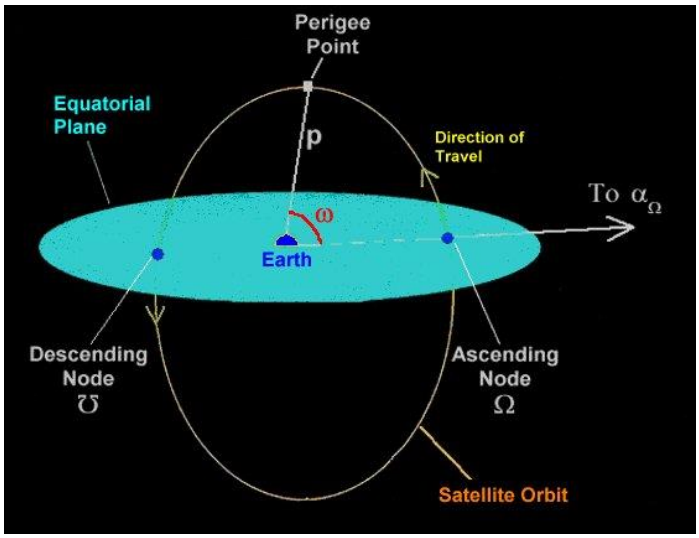


Figure 83| Argument of Perigee

http://www.castor2.ca/03_Mechanics/02_Elements/05_Arg/index.html

1.2.2- Figure buttons:

It was made for ease of use of the figure to figure the orbit, rotate the earth, back to the center of the earth and send the data to the main menu figure.

Figure 84| Figure buttons

1.2.2-a- Submit:

It enables sending the orbit parameters to the main menu as a vector of data. After pressing "Submit", Orbit parameters figure will be closed automatically. So, Press submit when orbit parameters equal the data of your own satellite.

1.2.2-b- Zoom All:

It enables focusing on the earth and the orbit for ease access of the orbit. If your satellite is High Earth Orbit (HEO) or Medium Earth Orbit (MEO), you can use the option of zoom all to view the overall of the orbit and the earth without cut-off or crop.

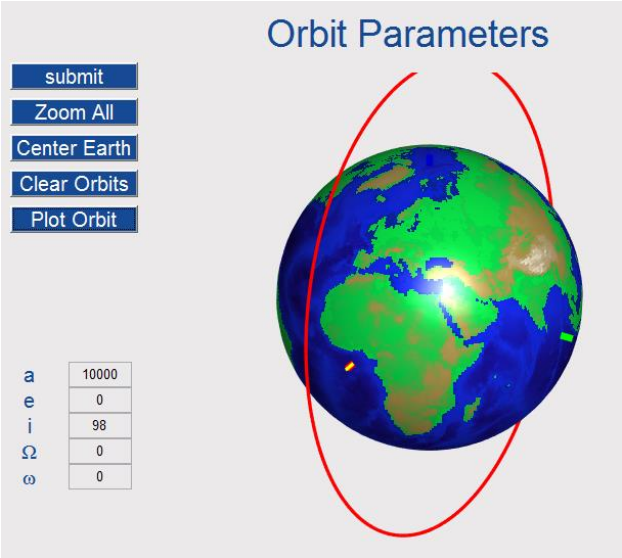


Figure 85| Orbit plotting without using "Zoom All"

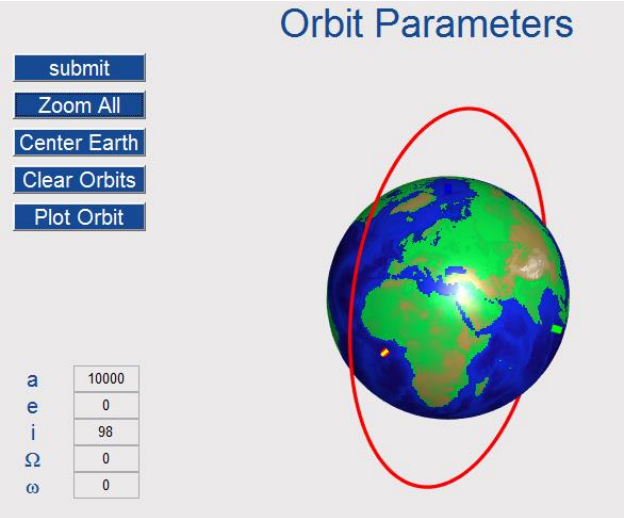


Figure 86| Orbit plotting using "Zoom All"

1.2.2-c- Center Earth:

To return to the origin view of the earth, “Center Earth” option enable that. Center Earth is the point of intersection between earth surface and X-axis.

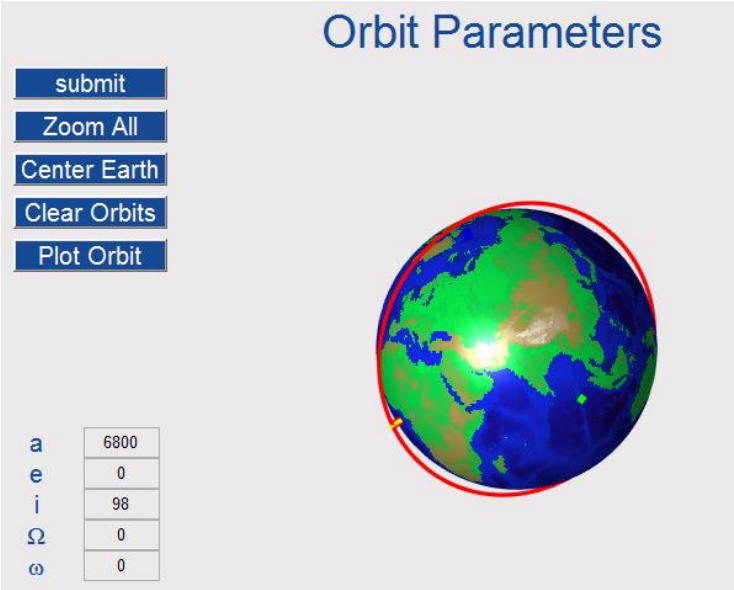


Figure 87| Orbit Parameters figure after using rotation option

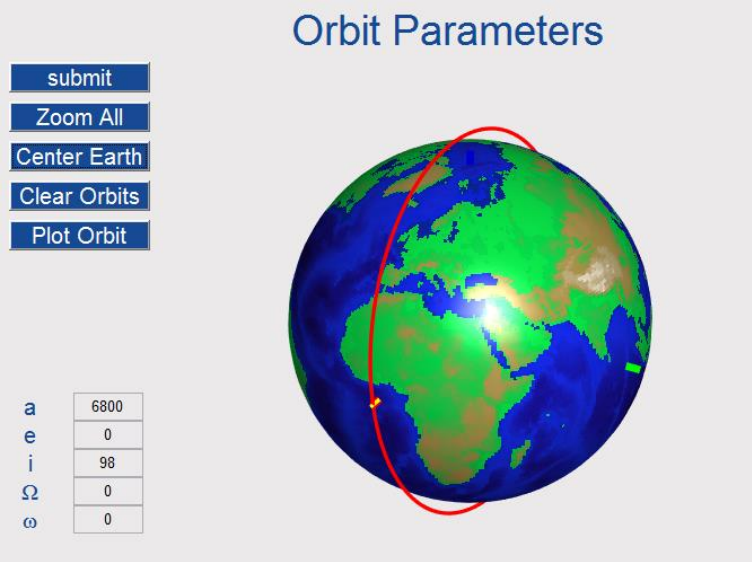


Figure 88| Orbit Parameters figure after using "Center Earth" button

1.2.2-d- Clear Orbits:

It enables the option of clear the figure of the plotted orbits. Several orbits plotting can be used for viewing different usage of each orbit.

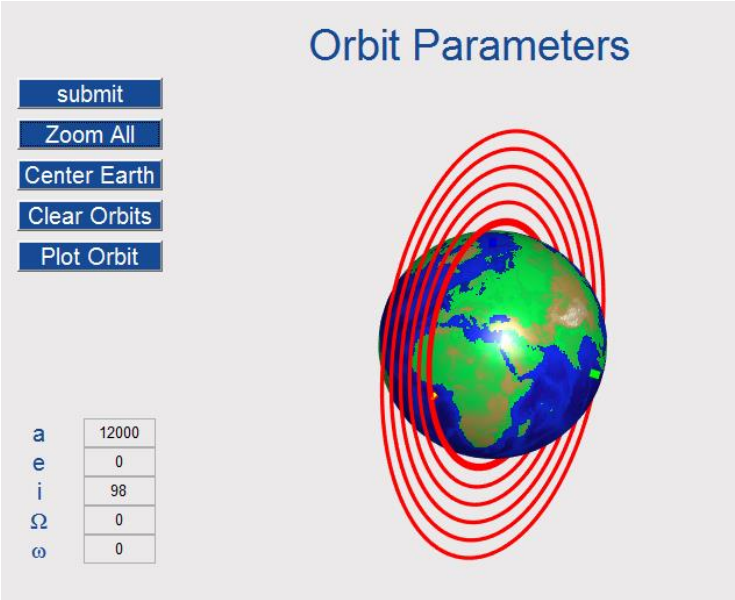


Figure 89| several orbits plotted

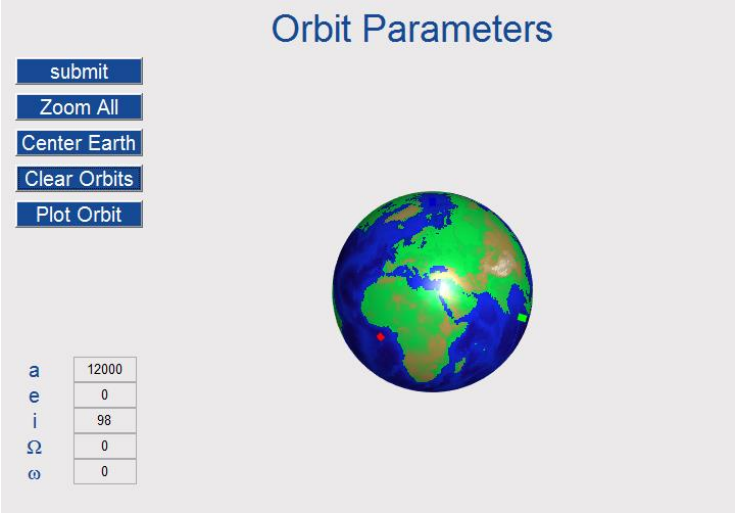


Figure 90| Clear the figure of plotted orbits

1.2.2-e- Plot Orbit:

To show the orbit around the earth with the same scaling of the earth.

1.2.3- Earth Orbit Simulation

It represents the main axes in the figure. Main axes can be sorted into 2 main groups:

- Earth axes
- Orbit axes

1.2.3-a- Earth axes:

It's included into:

- X-direction: which has red color on the following figure
- Y-direction: which has green color
- Z-direction: which has blue color

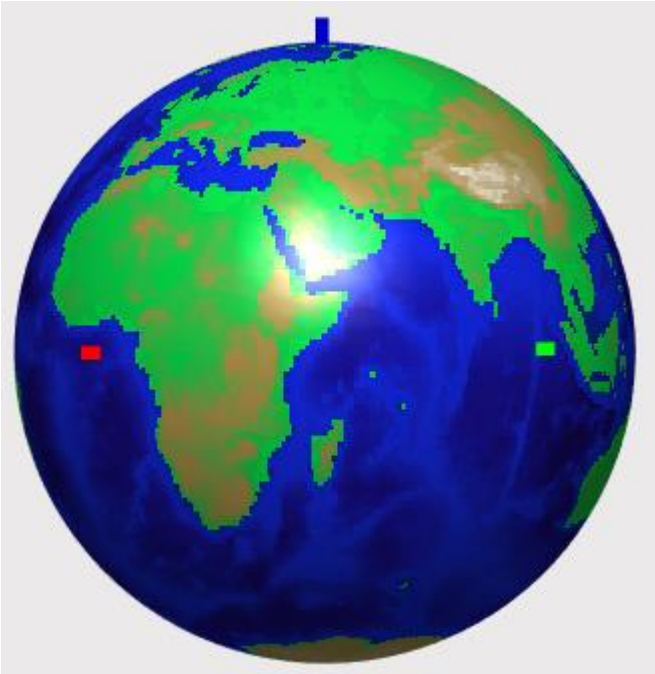


Figure 91| Earth Main Axes

1.2.3-b- Orbit axes:

It can be included into:

- Ascending node: in red color on the orbit path
- Axes of Perigee: in yellow color
- In direction of orbit momentum: in blue color

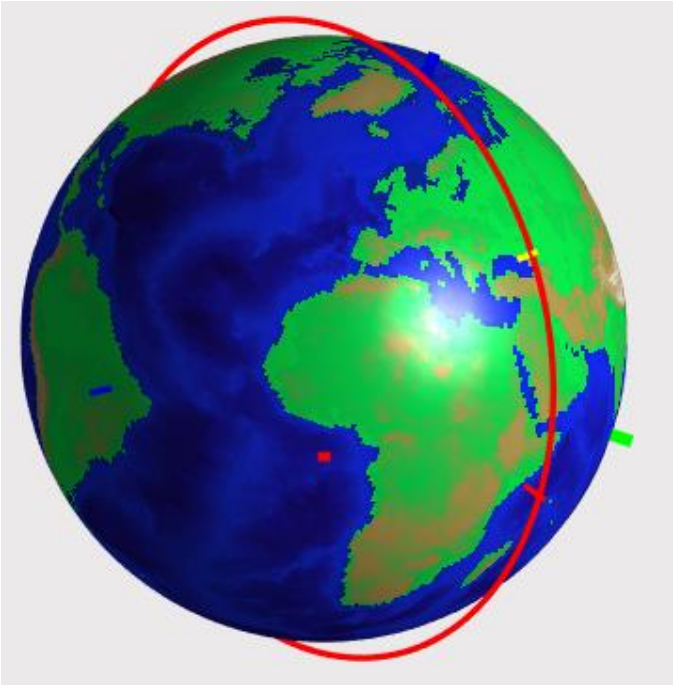


Figure 92| Orbit main axes

9.1.3 Controller

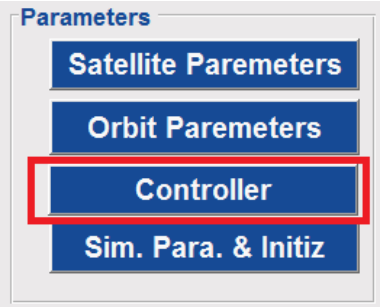


Figure 93| Controller button in main menu

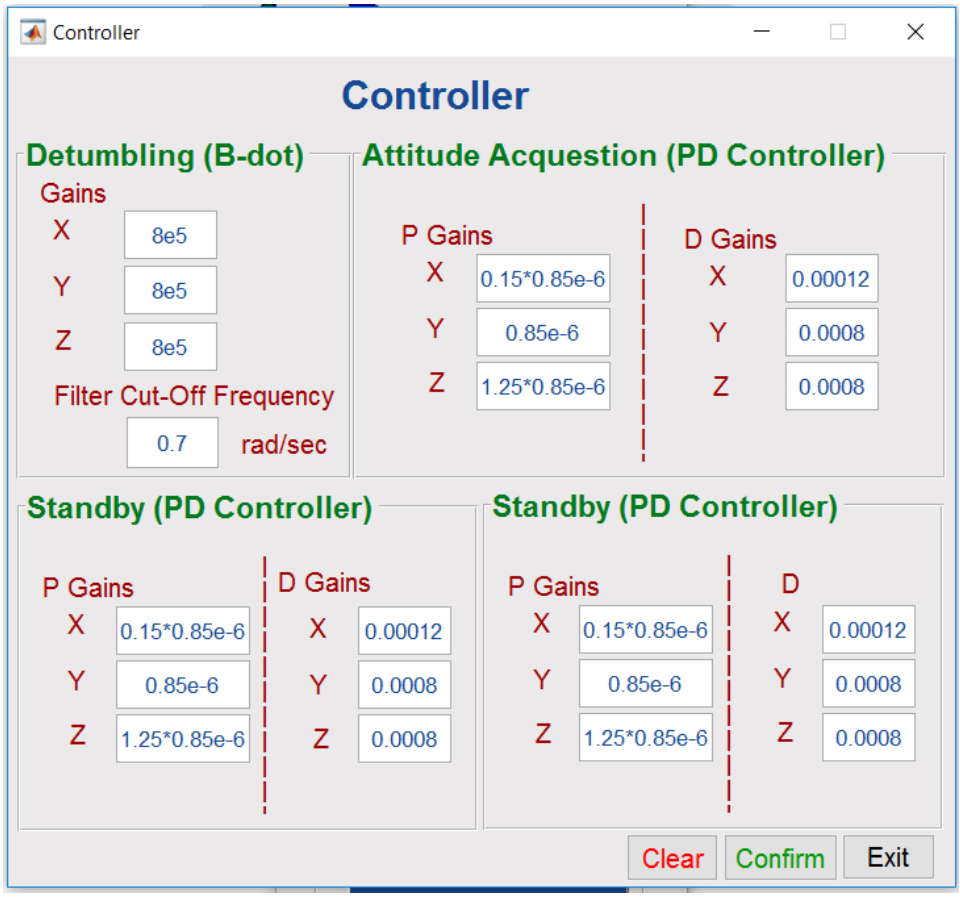


Figure 94| Controller figure

Controller can be categorized into:

- Detumbling (B-dot)
- Attitude Acquisition (PD Controller)
- Standby (PD Controller)
- Imaging (PD Controller)

1.3.1- Detumbling (B-dot)

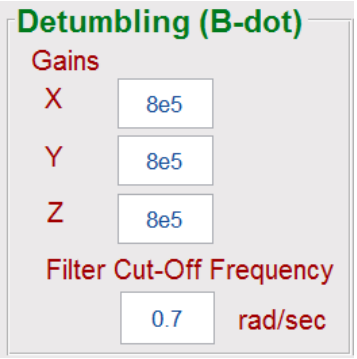


Figure 95| Detumbling panel

1.3.1-a- Gains:

It represents the gains of the B-dot controller in the axes set; X, Y, Z respectively.

1.3.1-b- Filter Cut-off Frequency:

It represents the characteristics of the filtering system. Cut-Off Frequency must be on unit of rad/sec.

1.3.2- Attitude Acquisition (PD Controller):

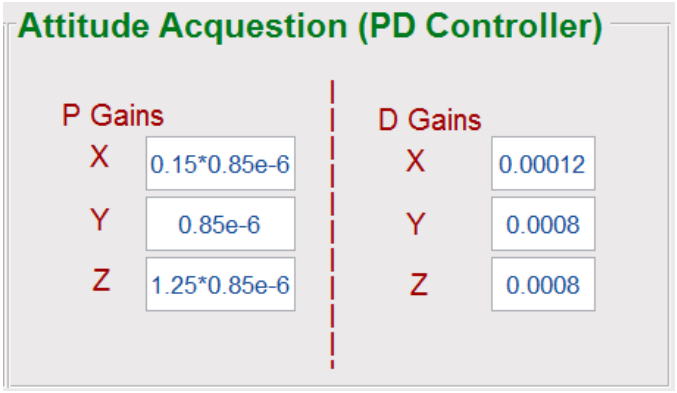


Figure 96| Attitude Acquisition

1.3.2-a- P Controller Gains:

It represents the gains of the P controller of the Attitude Acquisition in the axes set; X, Y, Z respectively.

1.3.2-b- D Controller Gains:

It represents the gains of the D controller of the Attitude Acquisition in the axes set; X, Y, Z respectively.

1.3.3- Standby (PD Controller):

Figure 97| Attitude Acquisition

1.3.3-a- P Controller Gains:

It represents the gains of the P controller of the Standby in the axes set; X, Y, Z respectively.

1.3.3-b- D Controller Gains:

It represents the gains of the D controller of the Standby in the axes set; X, Y, Z respectively.

1.3.4- Standby (PD Controller):

Figure 98| Attitude Acquisition

1.3.4-a- P Controller Gains:

It represents the gains of the P controller of the Imaging in the axes set; X, Y, Z respectively.

1.3.4-b- D Controller Gains:

It represents the gains of the D controller of the Imaging in the axes set; X, Y, Z respectively.

9.1.4 Simulation Parameters and Initialization:

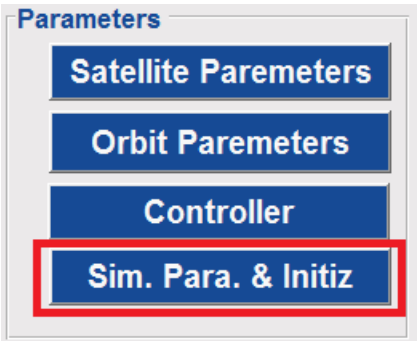


Figure 99| Simulation button in main menu

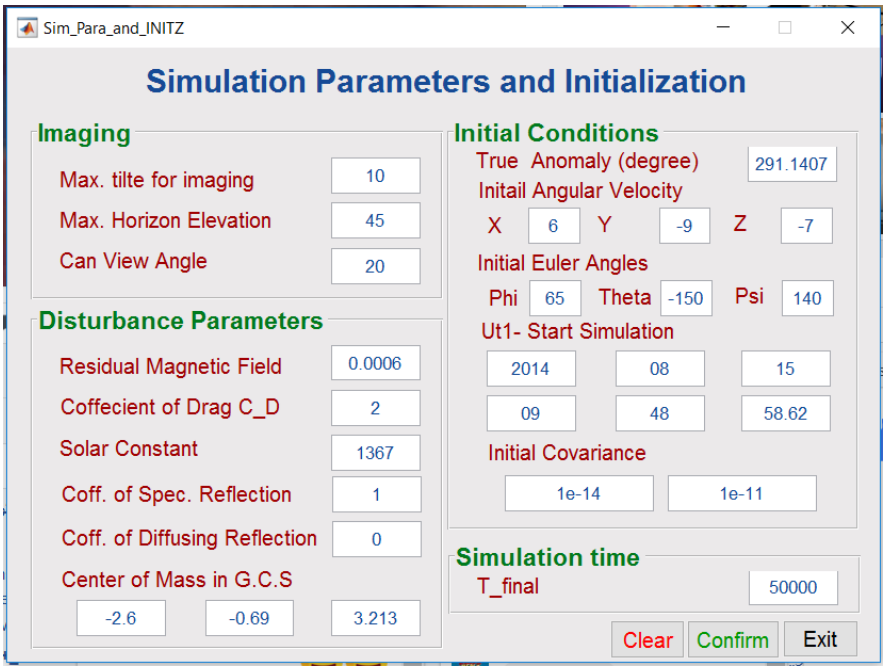


Figure 100| Simulation Parameters and Initialization

Simulation can be categorized into:

- Imaging
- Disturbance Parameters
- Initial Conditions
- Simulation time

1.4.1- Imaging:

It represents the data of the imaging mode for the required mission.

Figure 101| Imaging panel

1.4.1-a- Maximum Tilt for Imaging:

It represents the maximum roll tilting the satellite can achieve, Tilt angle must be in unit of degree.

1.4.1-b- Minimum Horizon Elevation:

It represents the minimum elevation angle from the horizon required for imaging with good quality, Horizon Elevation must be in unit of degree.

1.4.1-c- Can View Angle:

It represents Cam View angle must be in unit of degree.

1.4.2- Disturbance Parameters:

It represents the data of the disturbance of the environment.

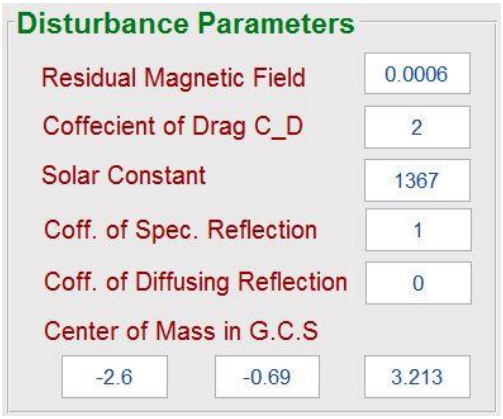


Figure 102| Disturbance parameters panel

1.4.2-a- Residual Magnetic Field:

It represents residual magnetism of the satellite.must be in unit of A.m^2.

1.4.2-b- Coefficient of Drag (C_D):

It represents Aerodynamic Drag coefficient. Coefficient of drag has no unit.

1.4.2-c- Solar Constant:

It represents solar radiation intensity. must be in unit of watt/m^2.

1.4.2-d- Coefficient of Specular Reflection:

It represents amount of radiation reflected specularly from the surface. has no units

1.4.2-e- Coefficient of Diffusing Reflection:

It represents amount of radiation reflected diffusivily from the surface. Has no units.

1.4.2-f- Center of Mass in G.C.S:

It represents the position vector of the center of gravity relative. must be in unit of to the geometric axis. Must be in mm

1.4.3- Initial Conditions:

It represents the data of the initial conditions of the satellite

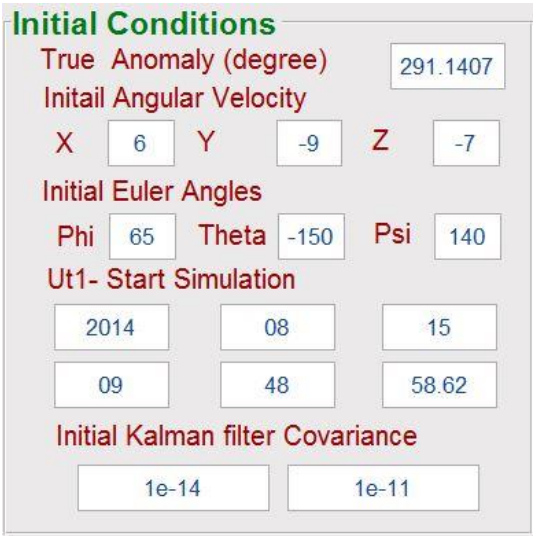


Figure 103| Initial Conditions panel

1.4.3-a- True Anomaly:

It represents the initial position of the satellite in it' orbit, True Anomaly must be in unit of degree.

1.4.3-b- Initial Angular Velocity:

It represents the rate of rotation of the satellite about X, Y, Z axes respectively. Angular velocity must be in unit of $degree/sec$.

1.4.3-c- Initial Euler Angles:

It represents rotation angles of the satellite in X, Y, Z axes $(\varphi, \theta, \boldsymbol{\psi})$ respectively. Euler Angles must be in unit of $degree$.

1.4.2-d- UT1 Start Simulation:

It represents the time of starting of simulation. It consists of 6 elements: Year, Month, Day, Hour, Minutes, and Seconds, respectively.

1.4.2-e- Initial Kalman Filter Covariance:

It represents the two initial value of the Kalman filter covariance. P_P_1 is the diagonal matrix of the two initial value of Kalman Filter Covariance.

1.4.2-f- Center of Mass in G.C.S:

It represents the position of the center of mass of the satellite relative to the original position of the C.G (center of symmetry shape). Center of Mass of the satellite must be in unit of mm.

1.4.4- Simulation time

Figure 104| Simulation Panel

It represents the total simulation time. It must be in unit of sec. To have all modes, simulation time should be more than 20000 sec.

2- Calculation and Simulation

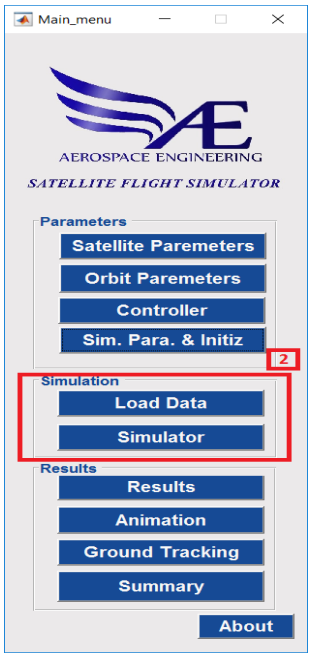


Figure 105| Simulation panel in main menu figure

It consists of:

- Load Data: For load data from .m file.
- Simulation: to calculate and then simulate the data in user-friendly viewing.

2.1- Load Data:

The following table shows the sequence of the data loading in .m file

Mass	0	Kg
X-dimension	100	mm
Y-dimension	100	mm
Z-dimension	200	mm
I_{xx}	0.06	$Kg.mm^$
I_{yy}	0.08	$Kg.mm^$
I_{zz}	0.004	$Kg.mm^$
I_{xy}	0	$Kg.mm^$
I_{yz}	0	$Kg.mm^$
I_{xz}	0	$Kg.mm^$
Max Dipole Moment	0.0760	
Efficiency factor	1	
Semi-major axis	7046141.784	M
B-dot → X	0.8e6	
B-dot → Y	0.8e6	
B-dot → Z	0.8e6	
Cut-off frequency	0.7	rad/sec
P Controller → X	0.85*0.15e-6	

P Controller → Y	0.85e-6	
P Controller → Z	0.85*1.25e-6	
Standard Deviation of Gyroscope	5e-6	rad/sec
Standard Deviation of Magnetometer	5e-7	$\frac{rad}{sec}$
P_P_initial	$\begin{matrix} 1e-14 & 0 & 0 & & \\ 0 & 1e-14 & 0 & & 0 \\ 0 & 0 & 1e-14 & & \\ & & & 1e-11 & 0 \\ & 0 & & 0 & 1e-11 \\ & & & 0 & 0 \end{matrix}$	

2.2- Simulation:

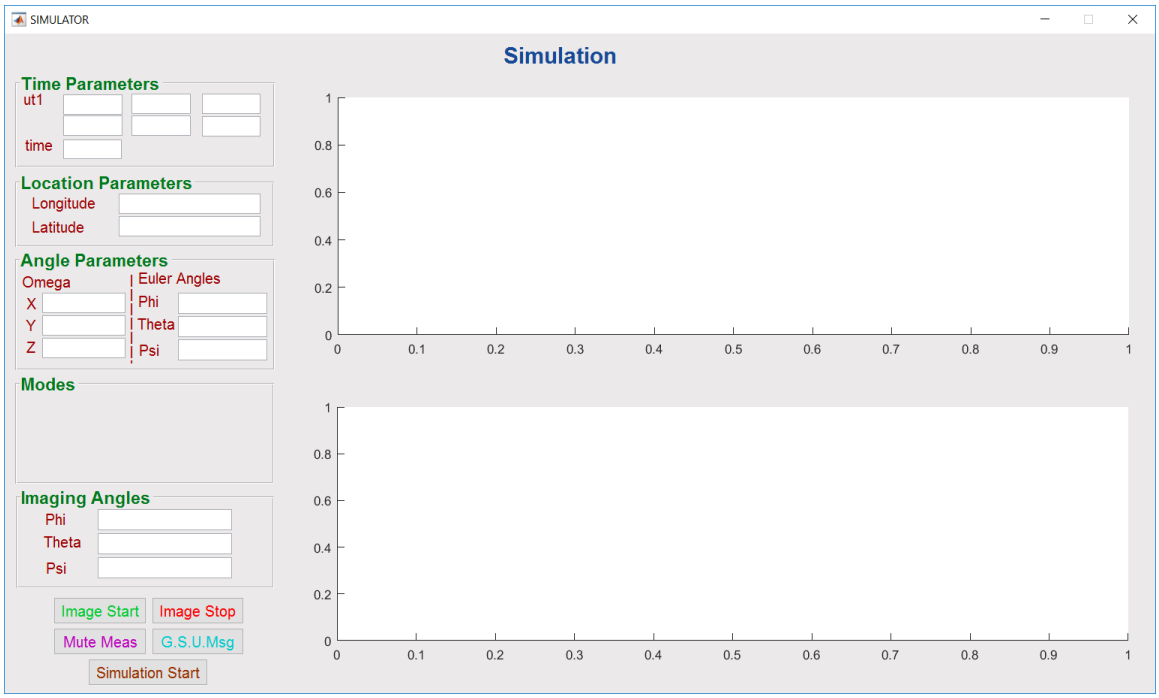


Figure 106| Simulation figure

It consists of:

- Time parameters
- Location parameters
- Angle parameters
- Modes
- Imaging angles

2.2.1- Time Parameters:

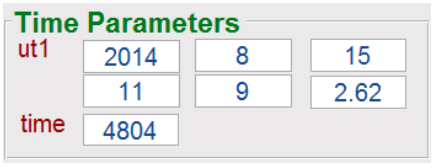


Figure 107| time parameters panel

It shows UT1 in 6 textbox; Year, Month, Day, Hour, Minute, Second respectively and time elapsed from the start of simulation.

2.2.2- Time Parameters:

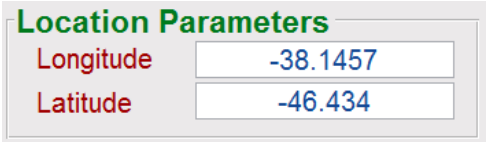


Figure 108| Location parameters panel

It shows the longitude and latitude of the satellite in orbit. Longitude is given as an angular measurement ranging from 0° at the Prime Meridian to +180° eastward and −180° westward. Latitude is an angle which ranges from 0° at the Equator to 90° (North or South) at the poles.

2.2.3- Angle Parameters:

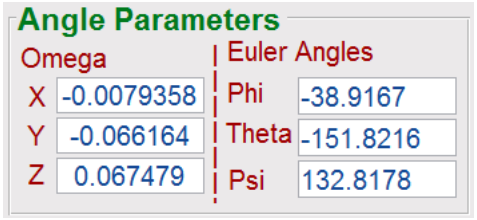


Figure 109| Angle parameters panel

It shows the angular velocities in X, Y, Z, and Euler angles in X, Y, Z respectively.

2.2.4- Modes of operation:

Figure 110| Modes panel

It shows which mode the satellite in. There are 6 modes of operation:

No.	Name of the mode	Color
1	Ideal Mode	Red
2	Detumbling Mode	Red
3	Reorientation Mode	Red
4	Standby Mode	Yellow
5	PreImaging Mode	Red
6	Imaging Mode	Green

2.2.5- Imaging angles:

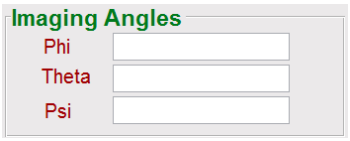


Figure 111| Imaging angles

It represents the Euler angles that the satellite should rotate to achieve imaging mission. Imaging angles must be in $degree$.

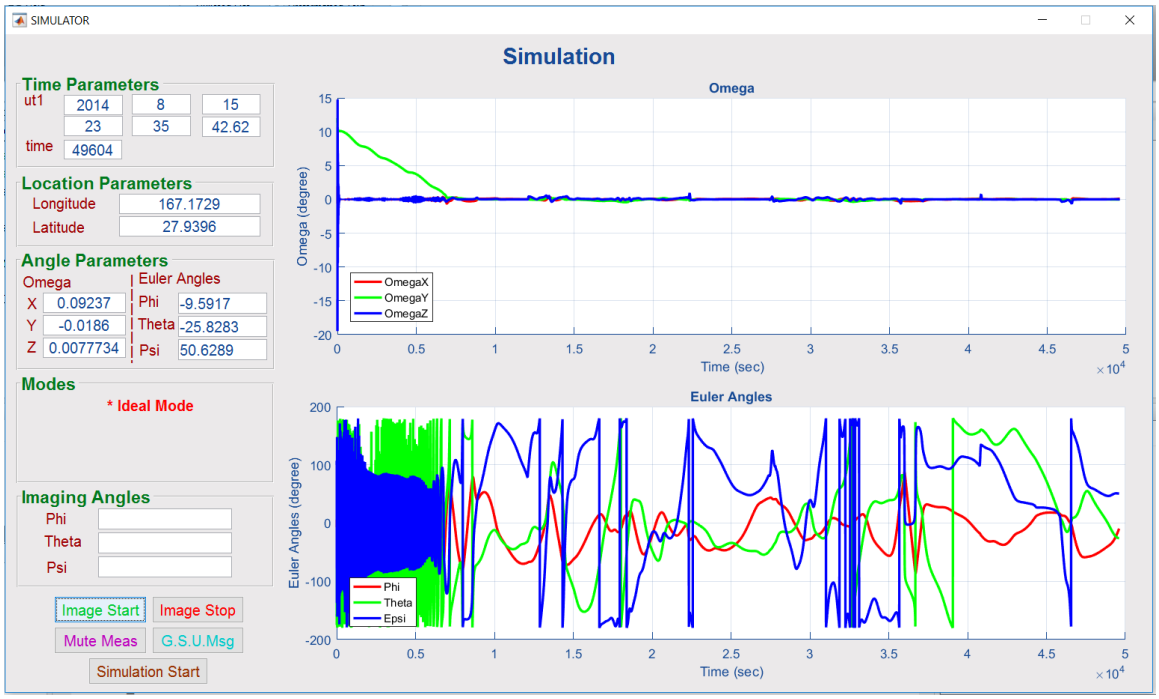


Figure 112| Simulation finished

3- Results

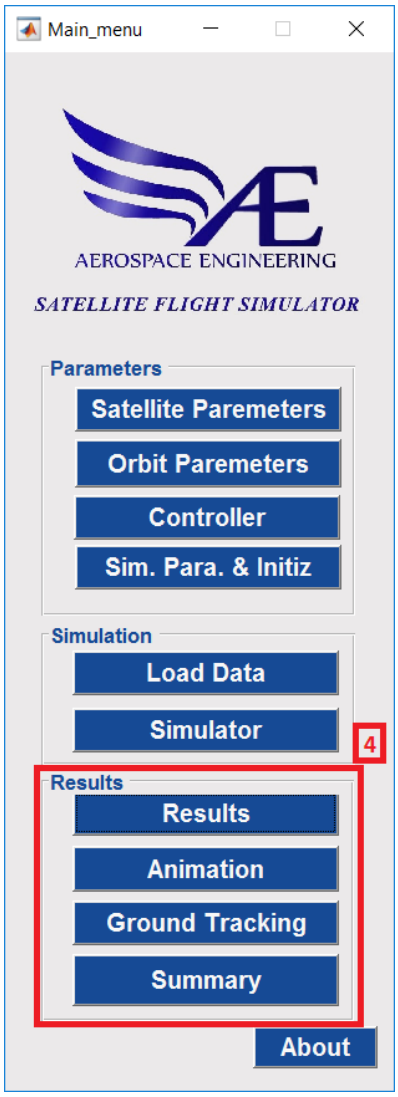


Figure 113| Results panel in main menu figure

It consists of:

- Results
- Animation
- Ground Tracking
- Summary

3.1- Results:

It represents the data displaying in graphs and plotting. It's better than showing data in tables. It gives an indication to the change of parameters during simulation.

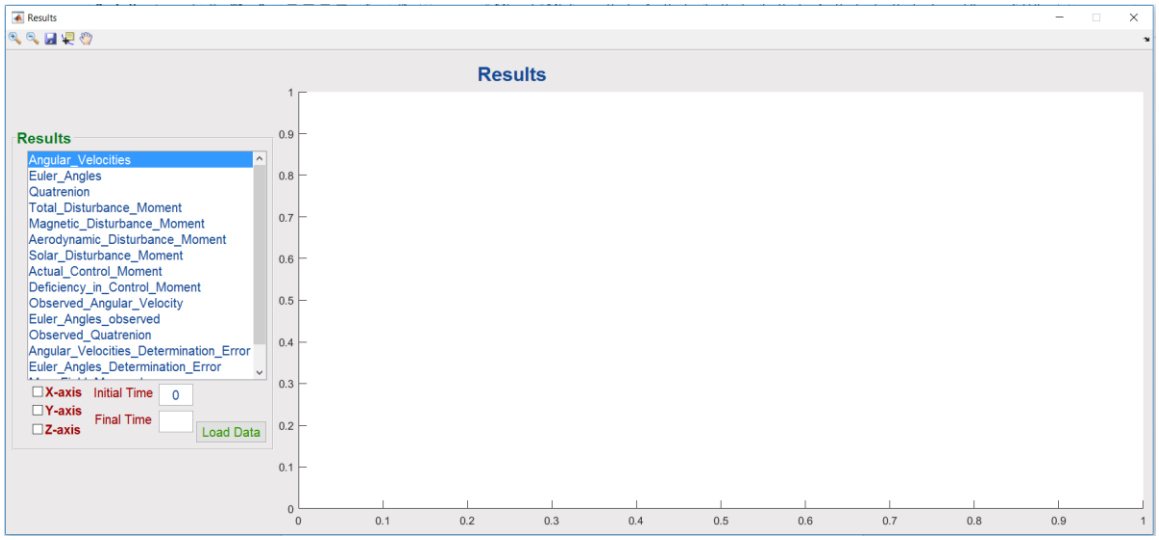


Figure 114| Results figure

Features of results figure:

- Display data in each axis separately.
- Focus on limited time (Start point → End point).
- Zoom in and Zoom out
- Saving the figure to picture or vector
- Data cursor
- Handle movement

Load Data: User should press "Load Data" button to load the data from main menu figure's handles to the varargin of "Results" figure. Before pressing the button, Initial time and Final time should be enter if not the default needed. Default time extend from 0 → the simulation time.

Graphs that can be displayed in results figure are:

Angular velocities	Euler angles
Quaternion	Total disturbance moment
Magnetic disturbance moment	Aerodynamic disturbance moment
Solar disturbance moment	Actual control moment
Deficiency in control moment	Observed angular velocity
Observed Euler angles	Observed quaternion
Angular velocity determination error	Euler angles determination error
Measured magnetic field	Measured B-dot

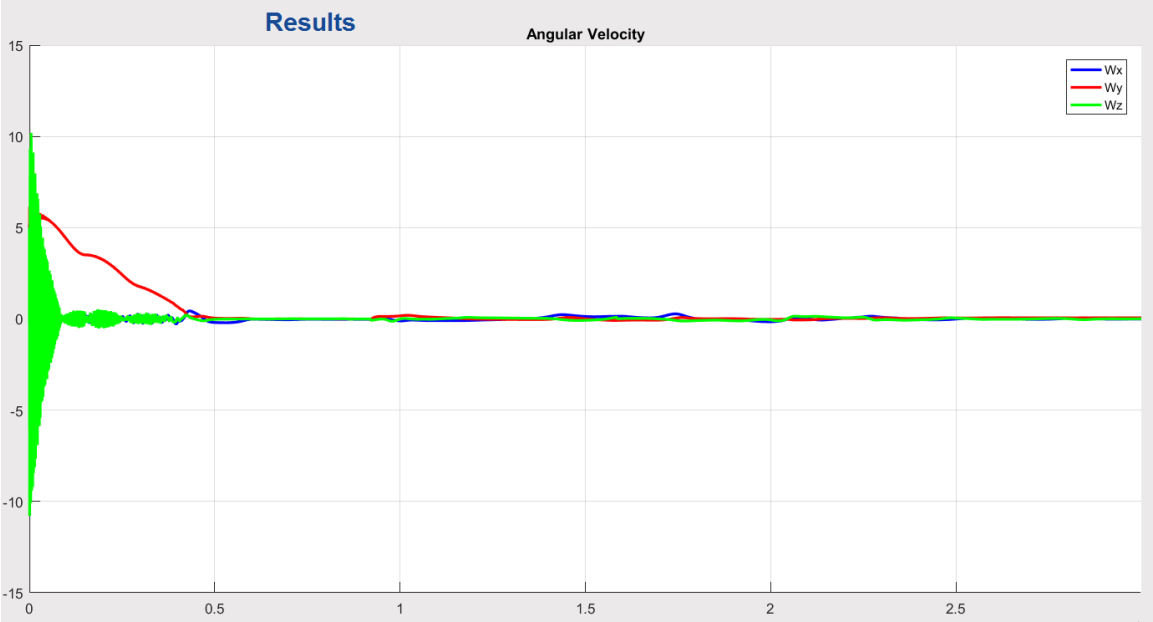


Figure 115| Angular velocity

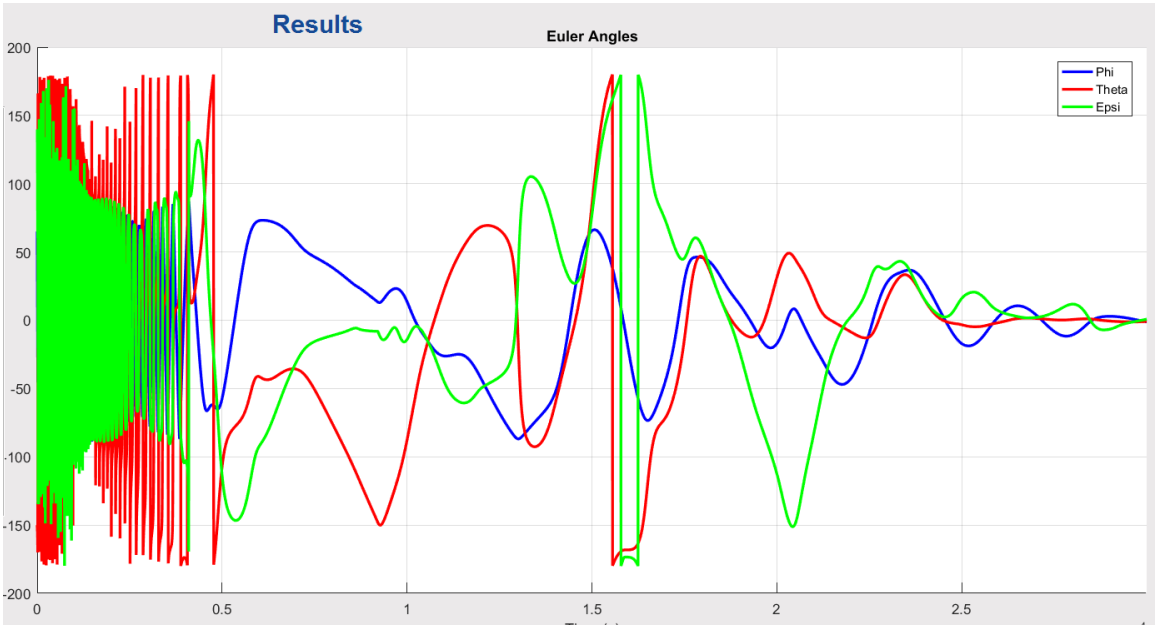


Figure 116| Euler angles

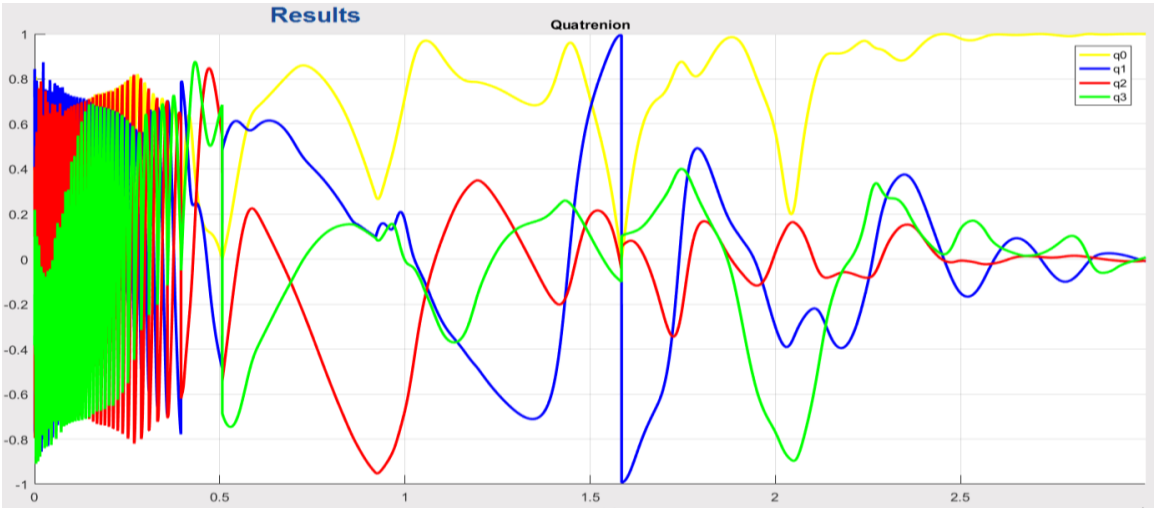


Figure 117| Quaternion

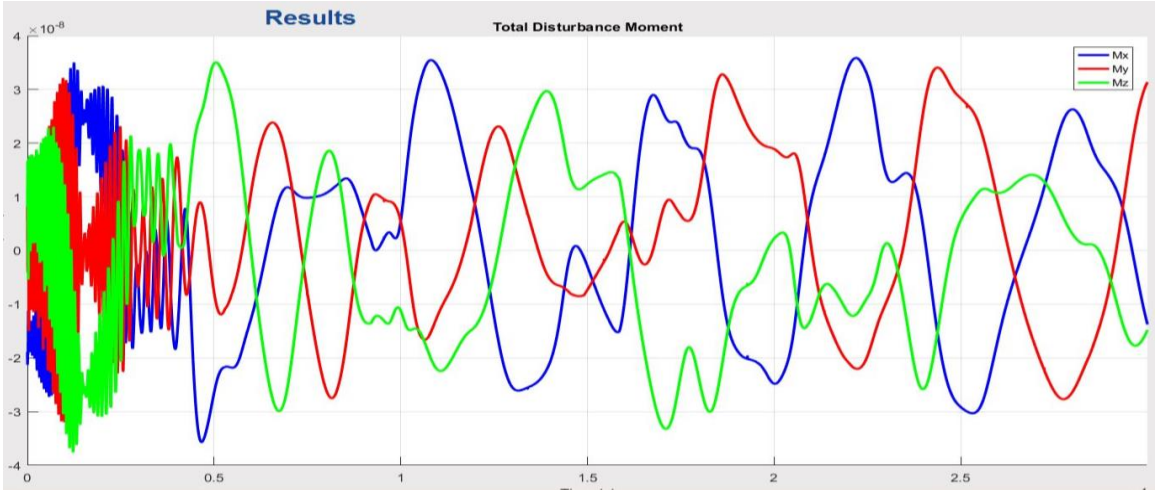


Figure 118| Total Disturbance moment

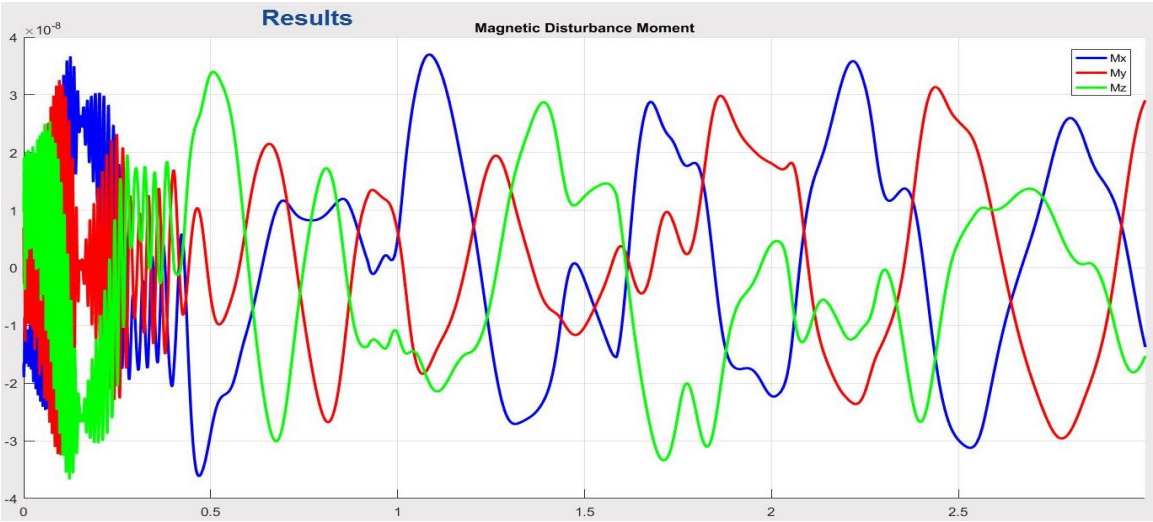


Figure 119| Magnetic disturbance moment

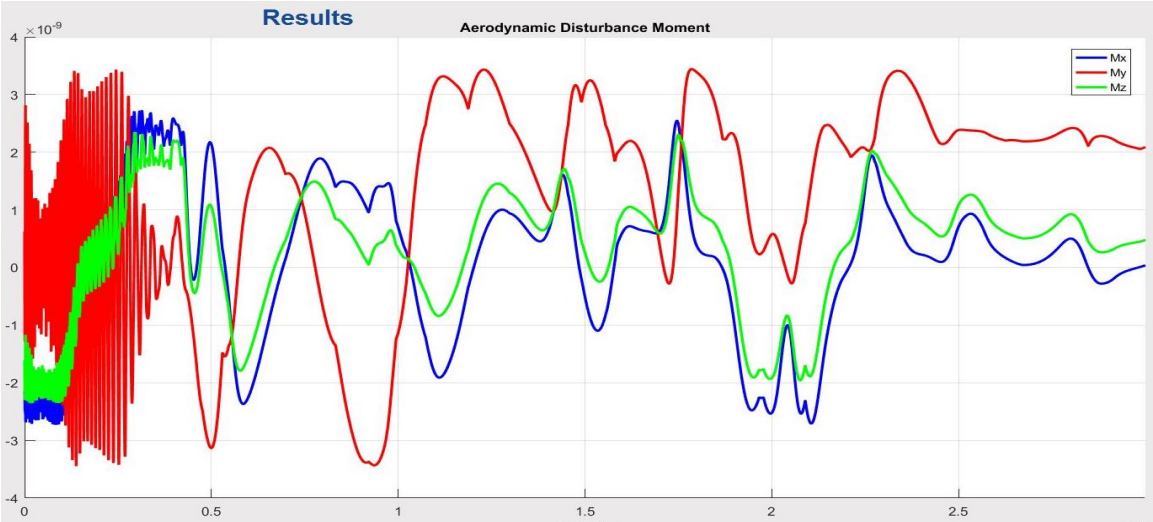


Figure 120| Aerodynamic disturbance moment

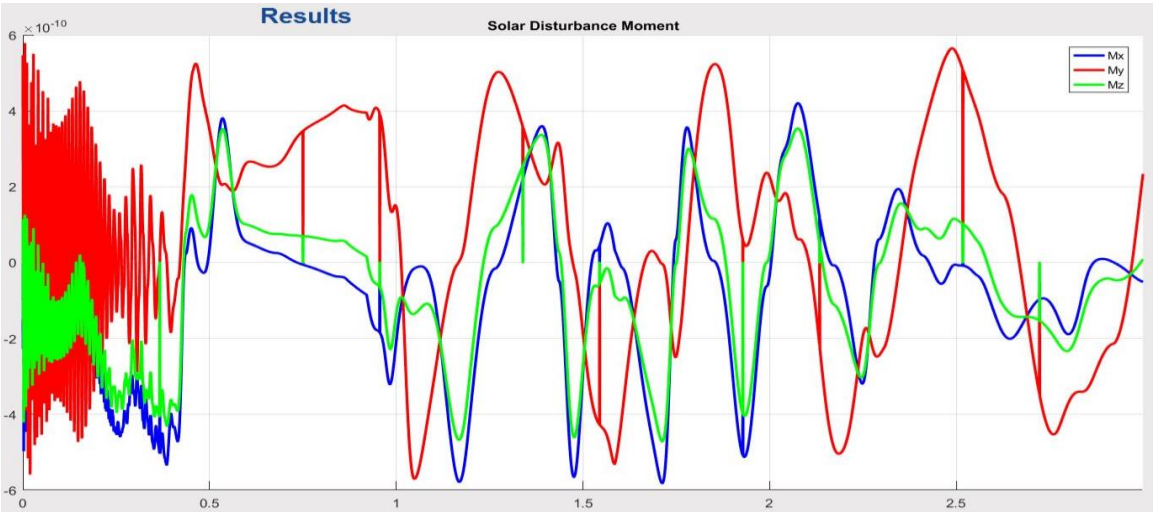


Figure 121| Solar disturbance moment

Figure 122| Actual control moment

Figure 123| Deficiency in control moment

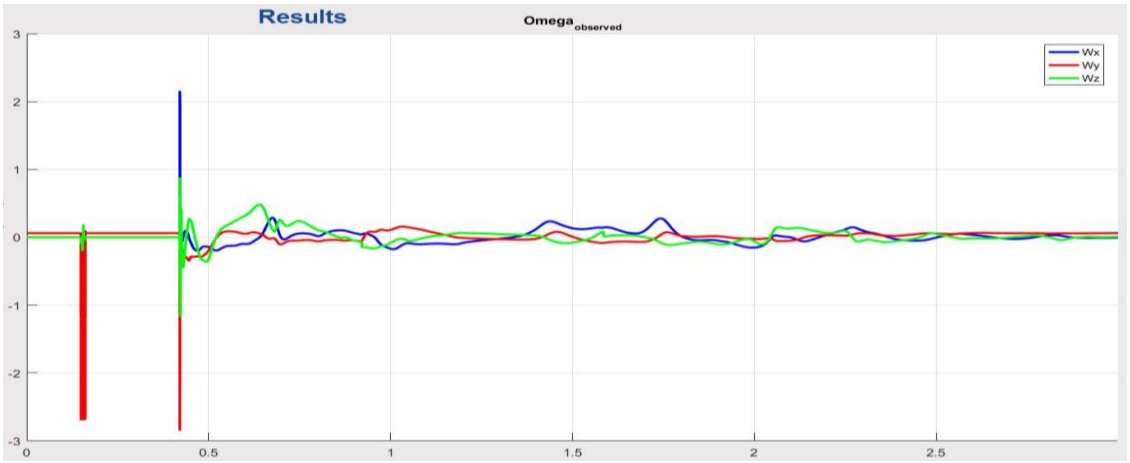


Figure 124| Observed Angular velocities

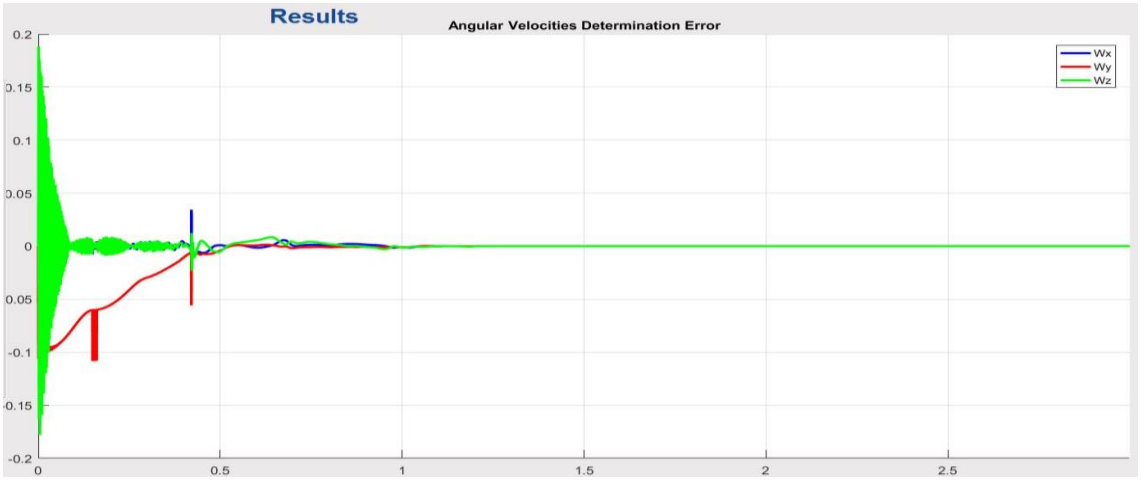


Figure 125| Angular velocities determination error

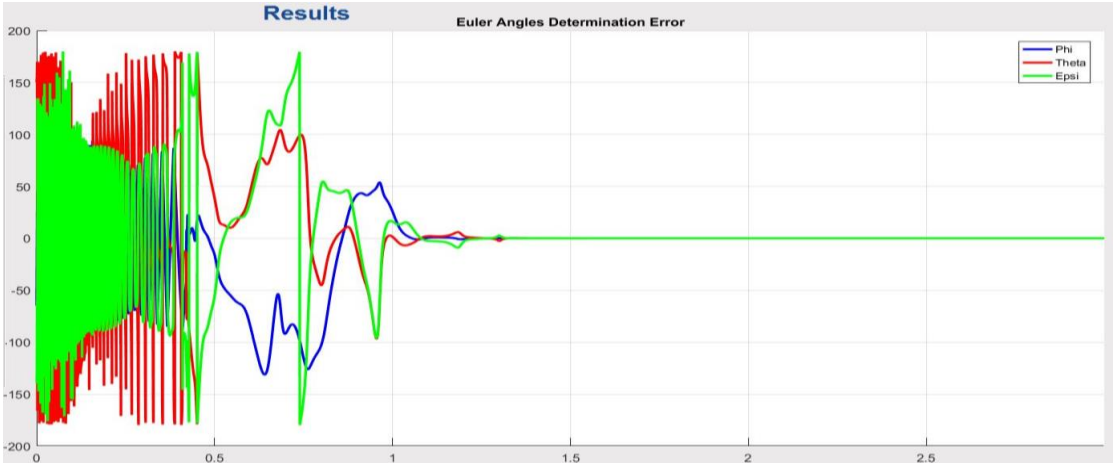


Figure 126| Euler angles determination error

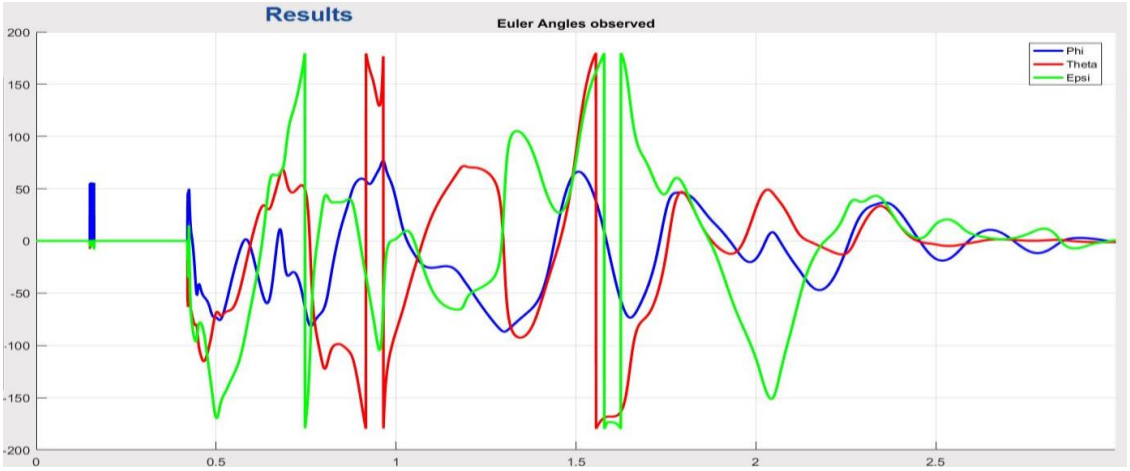


Figure 127| Observed Euler angles

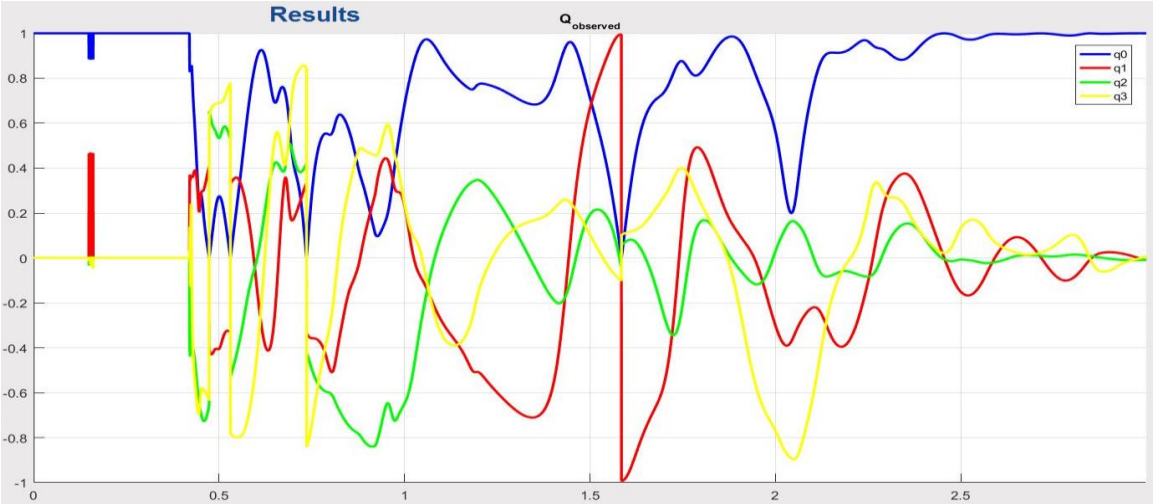


Figure 128| Observed quaternion

3.2- Animation:

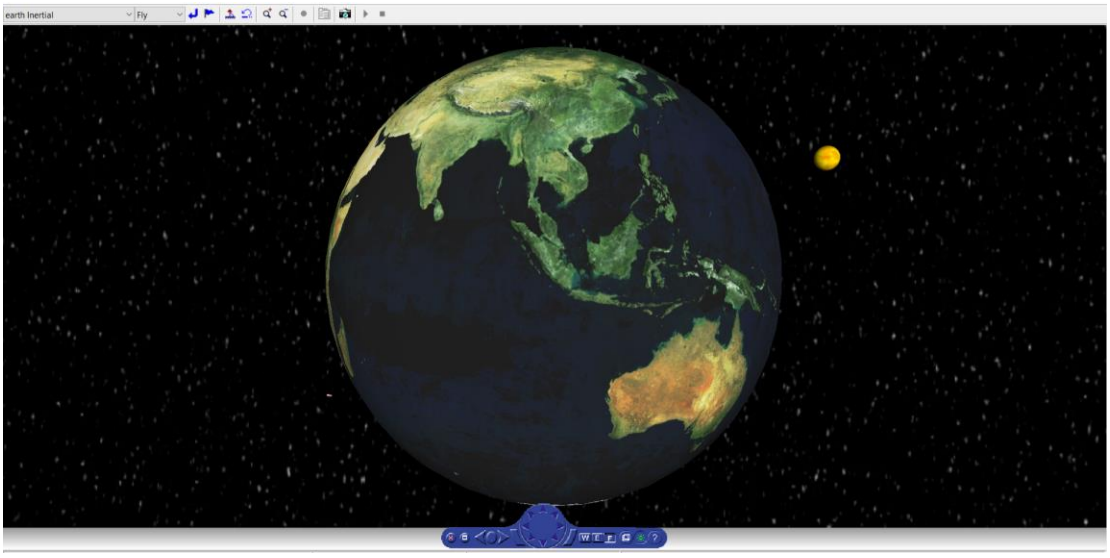


Figure 129| Animation VR

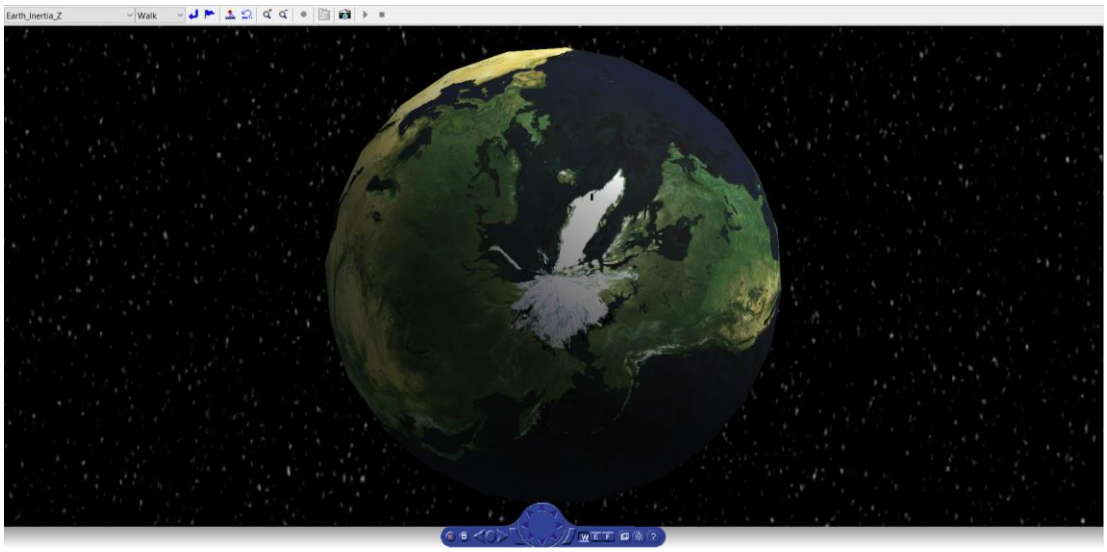


Figure 130| Animation VR

It refers to the process of making the illusion of motion and the illusion of change by means of the rapid succession of sequential images that minimally differ from each other. Satellite animation consists of:

- 3D Virtual world: VR programmed attached to MatLab is used.
- Translation and Rotation programming code: sends a sequence of commands for the required animation.

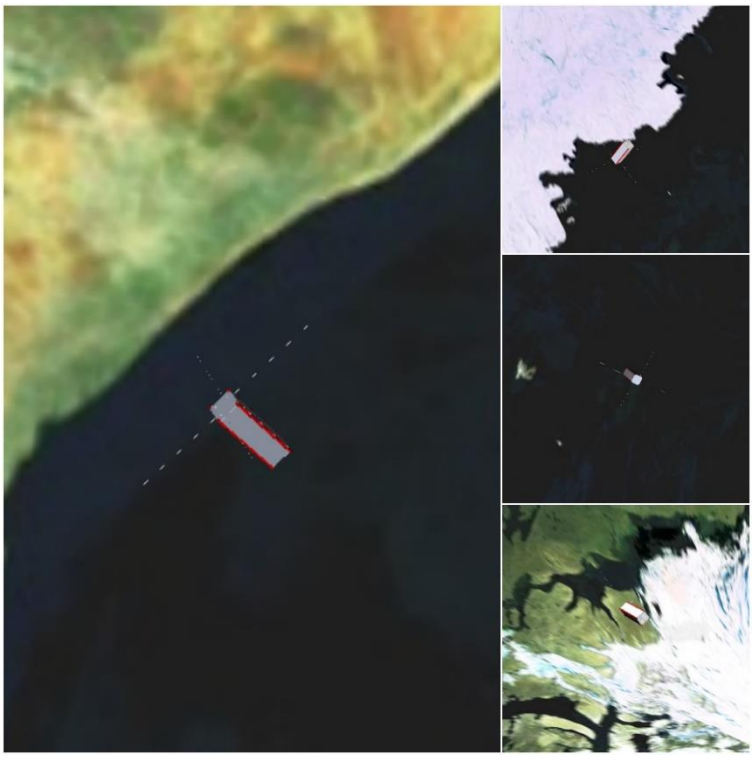


Figure 131| Animation

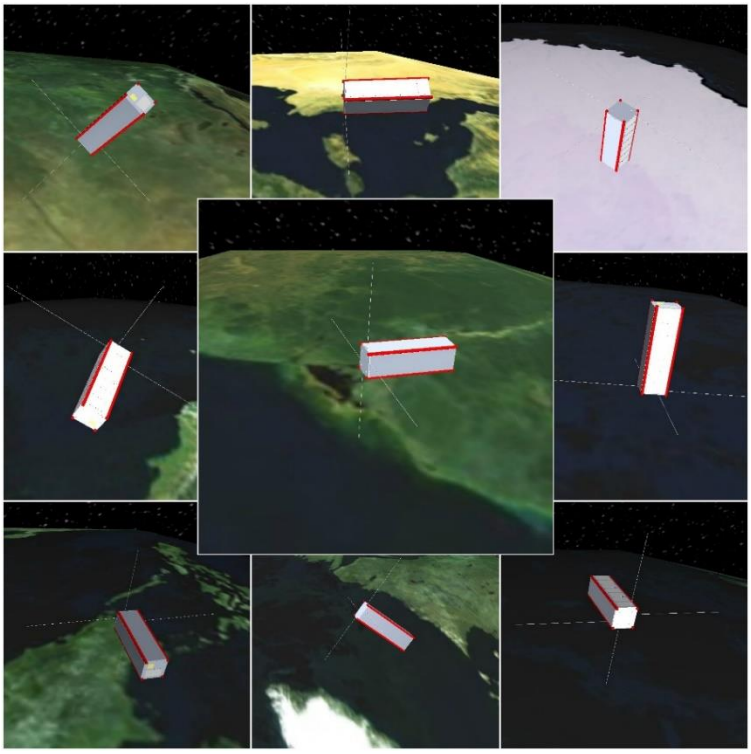


Figure 132| Animation

3.3- Ground Track

Satellite ground track is the projection of the sub-satellite point (Point where a straight line drawn from a satellite to the center of the Earth intersects the Earth's surface) on the rotating earth. It may be represented on a 3D spherical map or a 2D planar map of the earth.

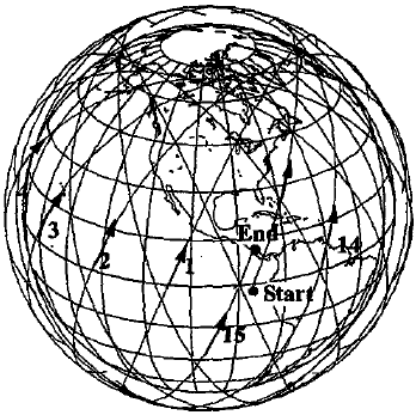


Figure 133| 3D Ground Track (Vallado)

Ground tracks are used in evaluating satellite coverage of a certain region on the earth, Moreover in our case it's used to estimate and evaluate the coverage with satellite tilting capabilities, Also to determine the pointing accuracy on the ground arose from the attitude deviations which minimizing is the main objective of ADCS.

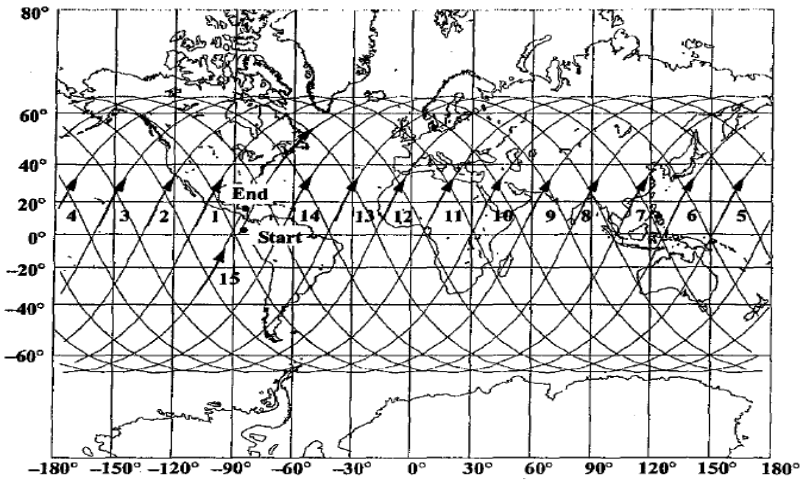


Figure 134| 2D Ground Track (Vallado)

After simulating the satellite motion in its orbit around the earth by utilizing the orbit propagator algorithm, obtaining the ground track depends on transforming the position vector of the satellite relative to Earth Center Inertial (ECI) axis to Earth Centered Earth Fixed (ECEF) axis, then obtaining the geocentric latitude and longitude to be plotted on the map this is discussed later.

This is for determining the nadir ground track, for other tracks like:

- Swath ground tracks for minimum elevation from the horizon
- Swath tracks for the tilted pointing
- Actual ground pointing with deviated attitude
- Also for determining the actual view field during imaging

Globe Projections[2]

A map projection is a systematic transformation of the latitudes and longitudes of locations from the surface of a sphere (Earth) into locations on a plane. All map projections necessarily distort the surface in some fashion. Depending on the purpose of the map, some distortions are acceptable and others are not; therefore, different map projections exist in order to preserve some properties of the sphere-like body at the expense of other properties.

The map projection used in this project is Equirectangular Projection (plate carrée map), which is former by using a spherical model of the earth and project it's surface on a gigantic cylindrical surface with axis parallel to the earth axis of rotation and applying scale to keep constant spacing between meridians and parallels

The projection maps meridians to vertical straight lines of constant spacing, and circles of latitude to horizontal straight lines of constant spacing.

Because of the distortions introduced by this projection, it has little use in navigation mapping and finds its main use in thematic mapping because of the particularly simple relationship between the position of an image on the map and its corresponding geographic location on Earth.

[2] Appendix of map projections classifications is included at appendices

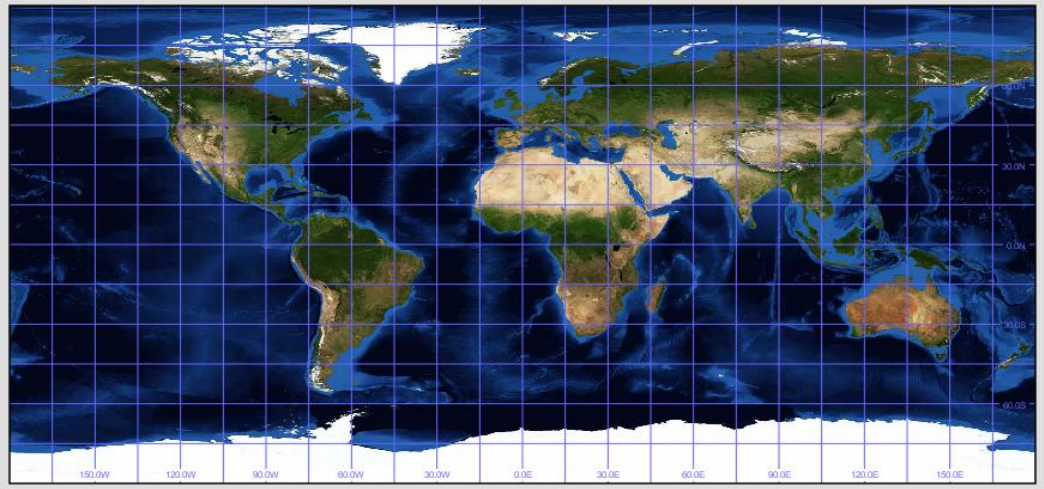


Figure 135| Equirectangular Projection

The transformation used for this projection is very simple,

$$x = (\lambda - \lambda_0)cos\phi_0 \quad , \quad (30)$$
$$y = (\phi - \phi_0)$$

Where:

λ is the longitude of the location to project.
ϕ is the latitude of the location to project.
ϕ_0 the standard parallels (north and south of the equator) where the scale of the projection is true in our case it's 0 degree latitude.
λ_0 is the central meridian of the map in our case it's Greenwich longitude.
x is the horizontal coordinate of the projected location on the map.
y is the vertical coordinate of the projected location on the map.

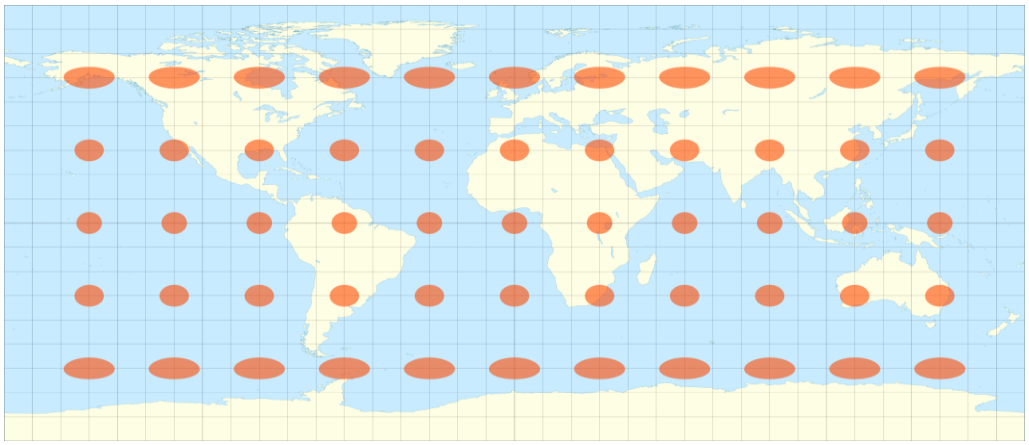


Figure 136| Equirectangular projection with Tissot's indicatrix of deformation

This map projection preserve the distance between points along the meridians (equidistant), Also it doesn't preserve the shape and directions (not conformal), nor the area. The deformation is clear in changing the shape of the circles at the poles from that at the equator, also the areas are deformed.

Nadir Ground Track

The easiest ground track to plot is the nadir ground track, as it depends only on the position vector of the satellite relative to the center of the Earth in (ECEF) coordinates system, which can be obtained from using the coordinate transformation (ECI►ECEF) described in transformations section which model the rotation of the earth around its axis. Where ECI position vector is calculated by orbit propagator module.

Then this position vector is transformed to longitude and latitude through the transformation (ECEF► Longitude & Latitude) to be plotted on the Equirectangular map projection by using (Eq 11111111).

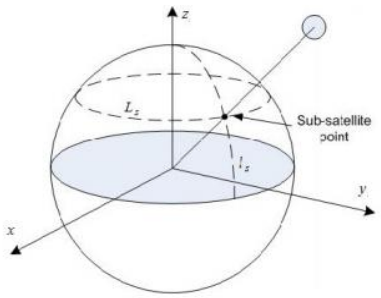


Figure 137| Sub Satellite Point

This nadir pointing ground track shows the path of the sub satellite point on the 1111 earth (Figure 137).

Min Horizon Elevation Swath

The swath is "the area on the surface of the Earth around the ground track that the satellite can observe as it passes overhead.

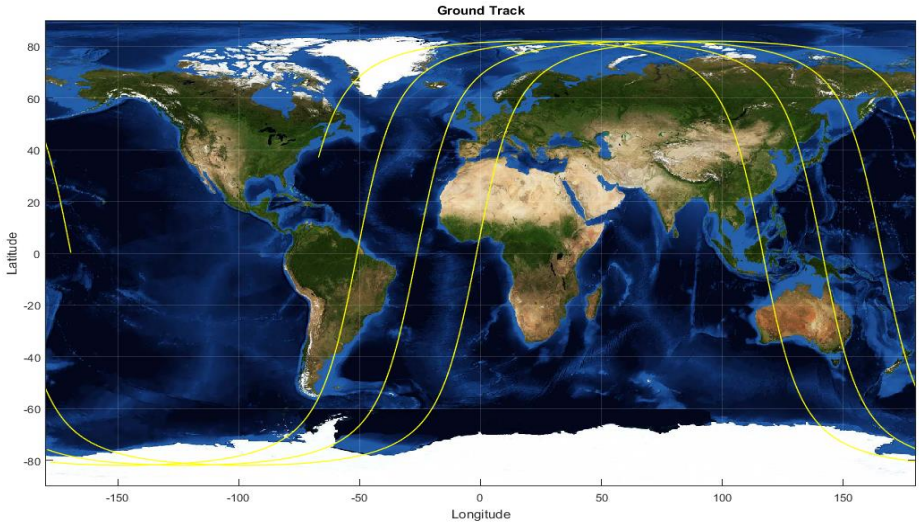


Figure 138| Nadir Ground Track

Minimum elevation swath track is the region which can be imaged or covered for communication by the satellite until the limit of minimum horizon elevation, beyond this limit the imaging wouldn't be clear due to large deviation of the satellite camera or antenna from the normality to the ground.

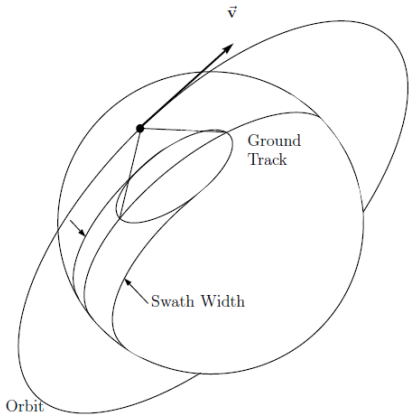


Figure 139| Swath Ground Track

Also, the instantaneous access area (IAA) is the area enclosed by the small circle on the sphere of the Earth, centered at the SSP and extending to the horizon as seen by the spacecraft. Two angles are evident in this geometry: the Earth central angle λ_0 and the Earth angular radius ρ. These are the two non-right angles of a right triangle whose vertices are the center of the Earth, the spacecraft, and the Earth horizon as seen by the spacecraft. The two angles are related by:

$$\lambda_0 + \rho = 90° \quad (31)$$

Generally, a spacecraft can point an instrument at any point within its IAA. However, near the horizon, a foreshortening takes place that distorts the view. The operational effects of this distortion are handled by introducing a minimum elevation angle, ϵ_{min} that reduces the usable IAA. Figure 140 shows the relationship between elevation angle ϵ, Earth central angle λ, Earth angular radius ρ , nadir angle η, and range D, H is the position vector of the satellite.

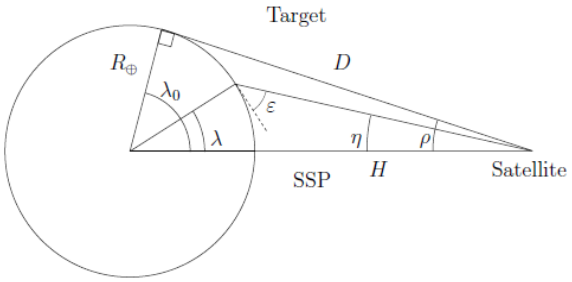


Figure 140| Geometry of Earth Viewing

Summary of Trigonometric Equations used in determining these angles are given below

$$\sin\rho = \frac{Re}{Re + H} \tag{32}$$

$$\cos\epsilon = \frac{\sin\eta}{\sin\rho}$$

$$\lambda = 90 - \epsilon - \eta$$

We have all the information necessary to calculate λ.

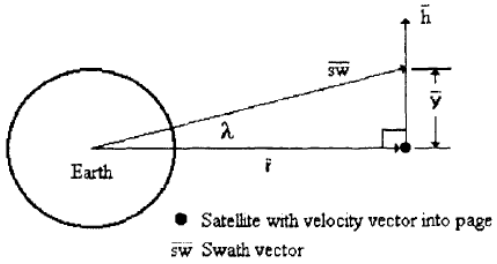


Figure 141| Satellite with velocity vector

Now we need a way to find the location of the swath for a given satellite position. Figure 141 below shows the geometry of the earth central angle, λ with a satellite "flying into the page." Where ‘h’ is the angular momentum direction of the orbit.

If we can find the vector $\bar{y}$ in Figure 141, we can simply add $\bar{y}$ to $\bar{r}$ to get the swath vector. If we subtract y from r we would get the swath vector on the other side as well. Since the velocity vector, $\bar{V}$ is into the page, we can form the vector $\bar{h}$ by taking the cross product of $\bar{r}$ and $\bar{V}$.

$$y = |\bar{r}| \tan \lambda \quad (33)$$

$$\bar{y} = y \frac{\bar{h}}{|\bar{h}|}$$

$$\bar{h} = \bar{r} \, X \, \bar{V}$$

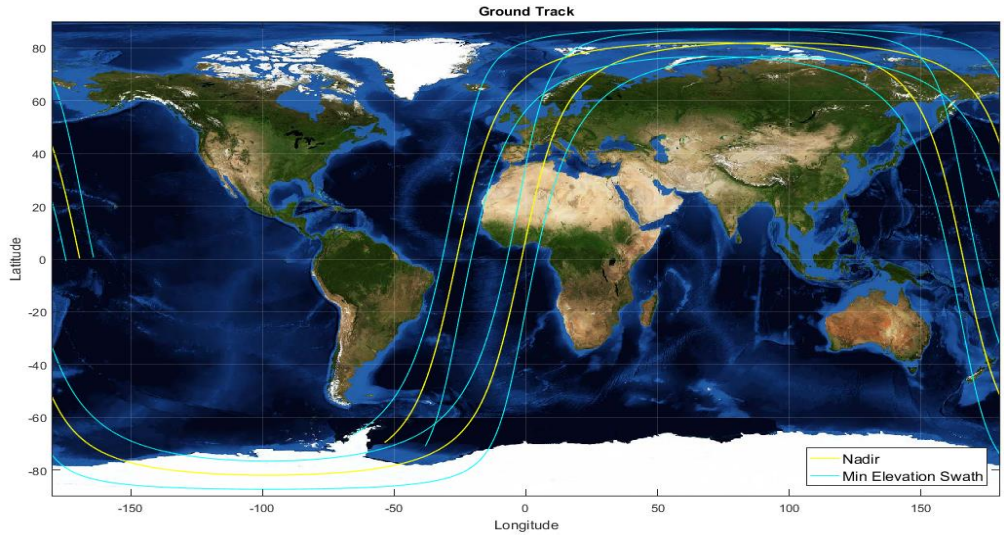


Figure 142| Min Elevation Swath ϵ=45°

Using Eq (33) we can form two swath vectors by adding or subtracting $\bar{y}$ to or from $\bar{r}$. Since both of these vectors are in the ECI frame, they should be transformed to ECEF then to Longitude and latitude using transformation algorisms described.

Max Tilting Swath

Another swath ground track is the max tilting swath which represent the coverage of the imaging payload is the satellite is tilted with max tilting angle around it's velocity vector (in $\bar{h}$, momentum vector and nadir plane).

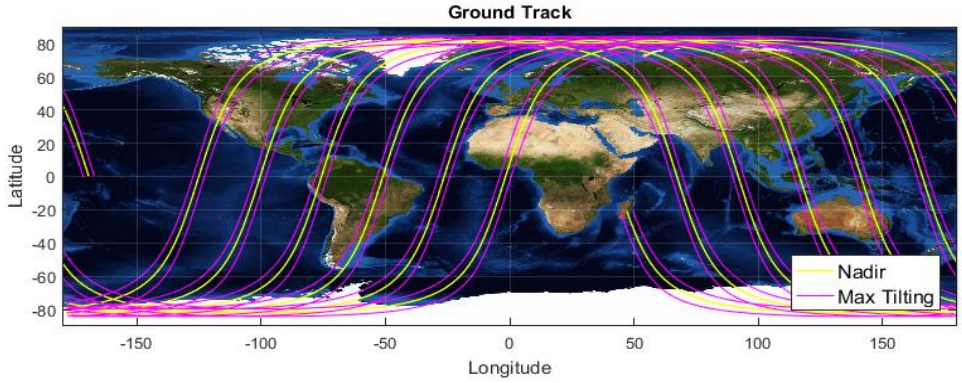


Figure 143| Max Tilting Ground Track for 20°

By referring to Figure 140, this swath can be obtained by assigning the nadir angle $\eta = max\ tilting\ angle$, then solving Eq 32 for ϵ, then λ. Then the same previous algorism in determining the swath vector it utilized by finding the magnitude of $\bar{y}$ and use it to generate a vector in the direction of angular momentum $\bar{h}$ and add and subtract it to $\bar{r}$ to get 2 swath vector then transformed from ECI to ECEF then longitude and latitude.

Tilted pointing during Imaging Ground Track

Then the satellite is in imaging position, it's camera is intended to be pointing to the nadir, However due to attitude deviations, it's pointing is deviated.

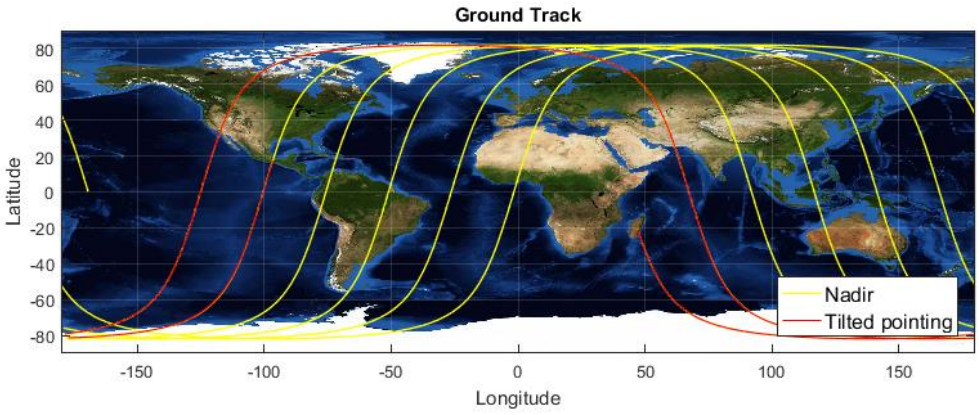


Figure 144| Tilted Pointing During Imaging

To draw a ground track for that deviated pointing of the satellite payload (camera), we should have information regarding the position of the satellite in its orbit and also its attitude at each point. Both of this data is available from the simulation. The question now is how to determine the point on the ground at which the satellite is pointing with its tilted attitude.

The following algorithm present a good way to determine such point by referring to Figure 140.

1. First the camera direction is known in the body coordinate system (BCS), and form knowing the attitude quaternion, we can perform transformation to orbit coordinate system (OCS).
2. Now we have 2 vector in OCS which are the camera vector and the nadir vector, so we can find the angle between them in the plane containing them by dot product. By referring to Figure 140, This angle is η
3. Through eq 32, we can get the angle λ
4. Through applying sin law we can get the length of the camera line that intersects the earth

$$L = \frac{R_e}{\sin \eta} * \sin \lambda$$

5. We now know the length of the vector and it's direction from step 1, so we can get this vector in OCS, then is transformed to ECI then to ECEF

Actual View Field Swath Ground Track

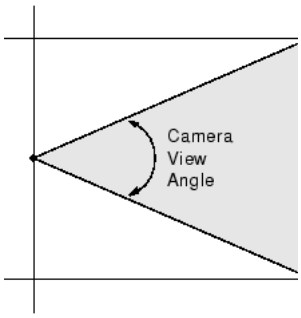


Figure 145| camera View Angle

If the view angle of the camera (see Figure 145) is known we can draw a swath ground track of the actual view field of the camera while taking into consideration the attitude deviation from the nadir pointing.

This swath ground track is formed from the points of intersection of 2 vectors in the plane which include the camera pointing vector obtained in the previous algorithm and the orbit momentum vector in OCS. These 2 vectors are tilted to the camera pointing vector with angle equals to half the camera view angle.

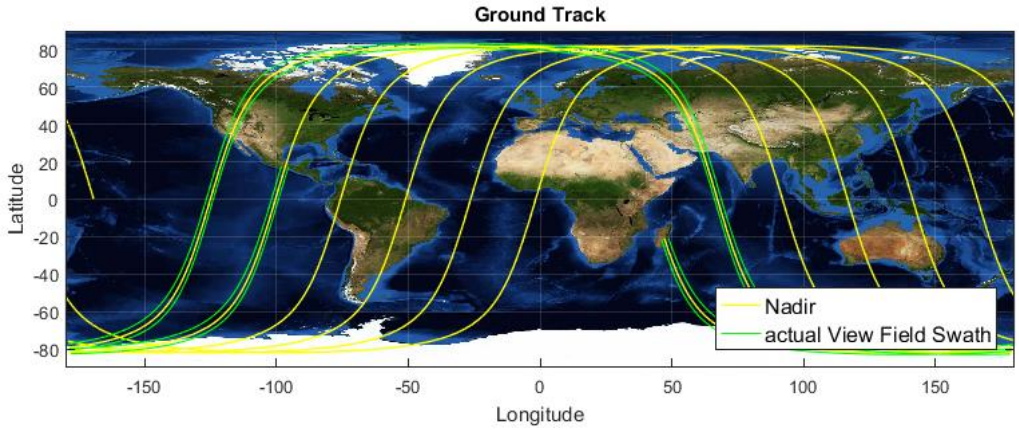


Figure 146| Actual View Field Swath

Instantaneous Actual Field of View

Also, we can find the Actual field of view of the satellite camera as it passes over the earth. This is done by finding the ellipse at which the camera view cone intersects with the earth surface. If the satellite was perfectly pointing toward the nadir, then this ellipse would be a circle, but in our case the pointing of the satellite is deviated by rotation in two directions which are (pitch and roll).

The result is that the intersection between the camera view cone and the earth is an ellipse, this ellipse is not centered at the center of the cone and also none of its axes is the principal axis. So to find this ellipse we need to get the most general equation of ellipse in the 2-D plane.

This equation is found to take the form [9]

$$Ax^2 + Bxy + cy^2 - (2Ah + kB)x - (2Ck + Bh)y + (Ah^2 + Bhk + cK^2 - 1) = 0$$

This equation is in 5 unknown ellipse parameters, to find these parameters we need to find 5 point on the ellipse we want, these points are found in a similar manner as the two points needed in the previous algorithm of plotting the actual view field swath ground track.

After the 5 points are found, the preceding equation can form a system of 5 linear equations to be solved together, then the ellipse parameters are obtained.

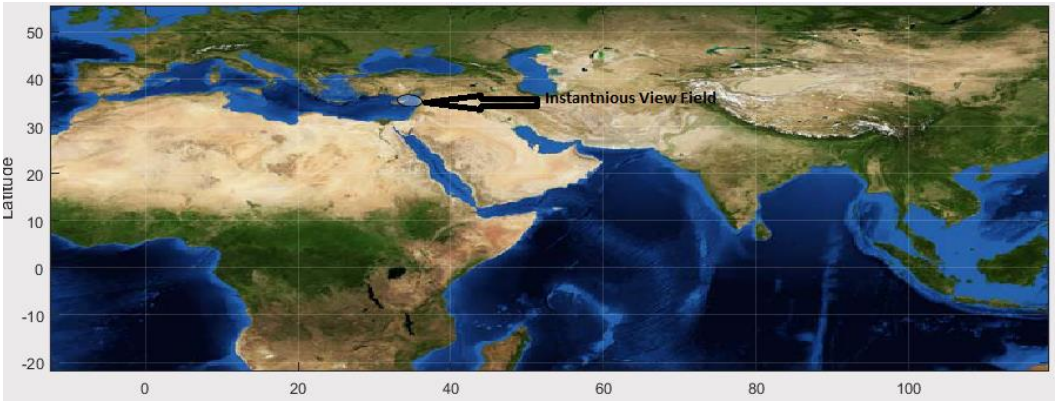


Figure 147| Instantaneous View Field

By this means we can plot the actual field of view of the satellite as it passes over the earth.

Figure 148| Ground Tracking

It consider a graphical representation for the orbit, Orbit parameters and location. It consists of:

- Plotting
- Orbit parameters
- Inertial Position
- Velocity
- Location
- Animation speed control

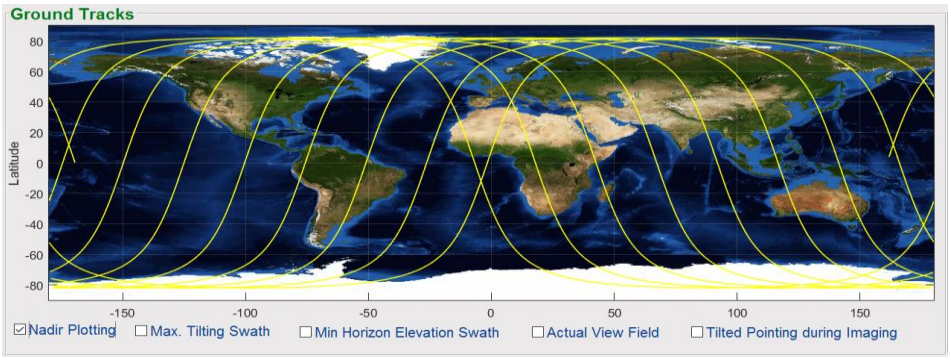


Figure 149| Nadir Plotting

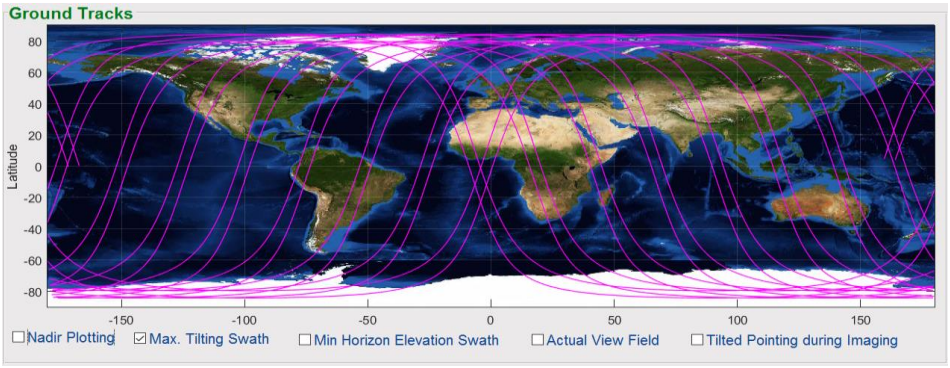


Figure 150| Max. Tilting Swath

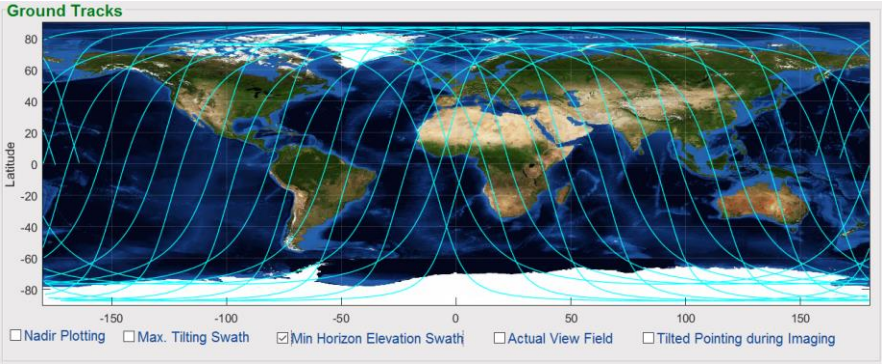


Figure 151| Max. Horizon Elevation Swath

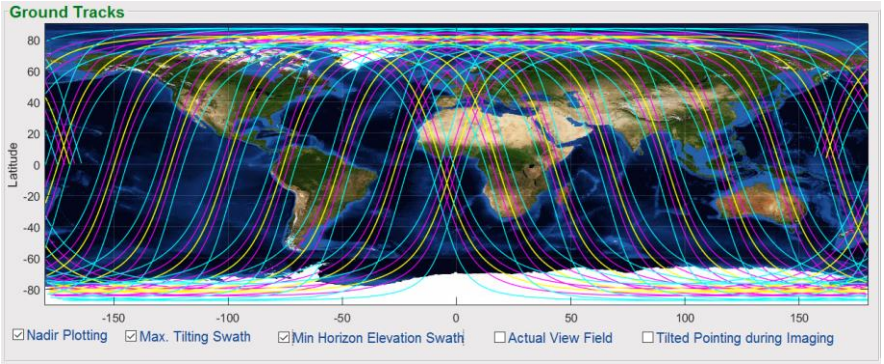


Figure 152| Ground Track plotting

3.3.2- Orbit Parameters:

Figure 153| orbit parameters displaying

It considers a representation of the data of the orbit; Semi major axes, Eccentricity, True Anomaly, Inclination, Argument of perigee, Ascension, Inertial position and Velocity.

3.3.3- 3D Ground Track:

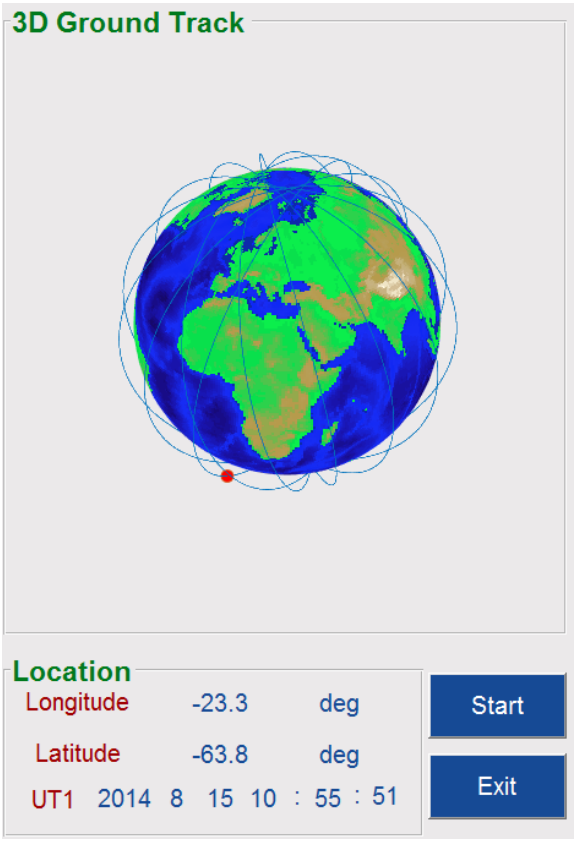


Figure 154| orbit animation in 3D

It displays the orbit of the satellite in 3-D model around the earth. Longitude, Latitude and UT1 has been displayed in numerical way.

3.4- Summary

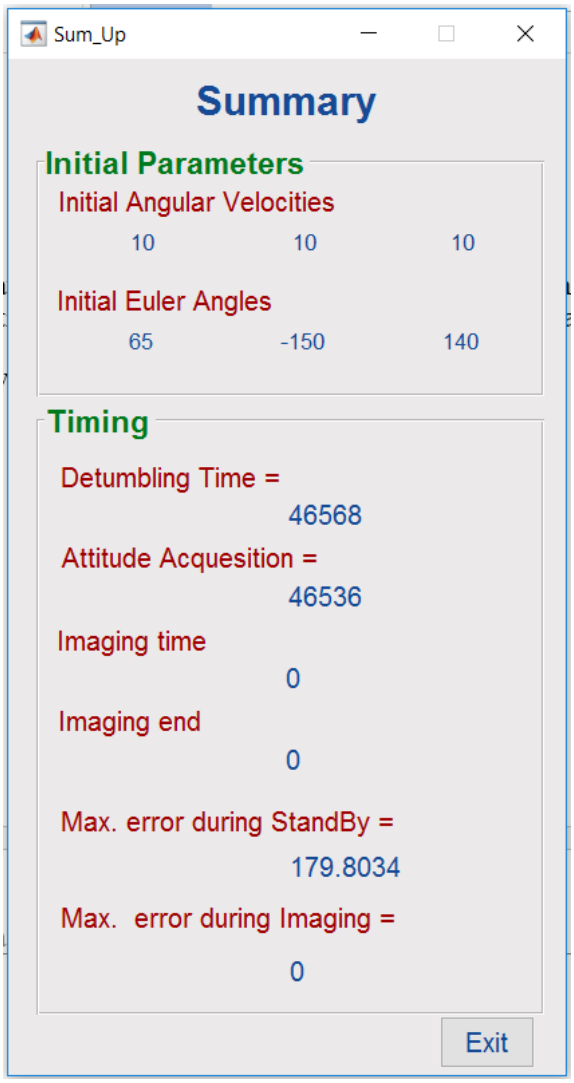


Figure 155| Summary Panel

It sums up the data of the mission. It considers a quick indication that the satellite mission achieved or not. It consists of:

- Initial parameters
- Timing

3.4.1- Initial parameters:

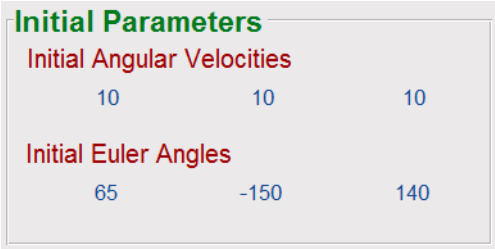


Figure 156| Initial parameters panel

It shows the initial conditions that entered at the simulation parameters and Initializations figure or loaded from Load_Data.m file. Initial parameters are shown in $degree$.

3.4.2- Timing:

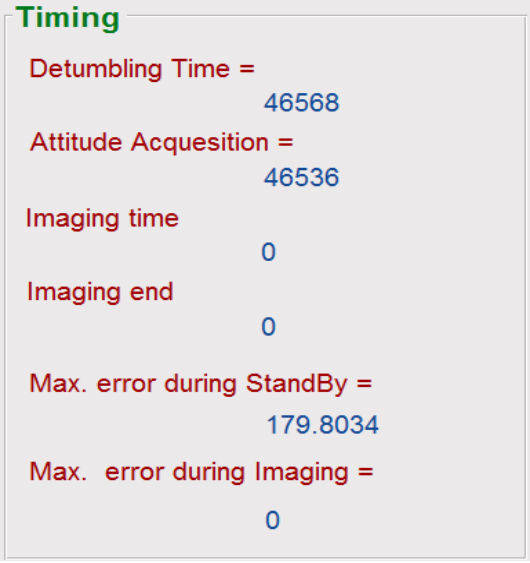


Figure 157| Timing panel

It displays the summary of the satellite flight simulator. It consists of:

- Detumbling time
- Attitude acquisition
- Imaging time
- Imaging end
- Max error during Standby
- Max error during Imagi

Chapter **10**

Conclusion and Future Work Recommendation

10.1 Conclusion

In this project, the satellite attitude determination and control techniques are studied and the verification results are presented. The ADCS is designed in C programming language with the aid of flight simulator and being tested in the Software in the loop simulation for the verification of the ADCS model and resulted in an Error of about 10^{-8} from the MATLAB code. The results of the Software in the loop simulation show that the attitude estimation errors are within 0.01° for Euler Angles and within 0.06 rad/s for the Angular velocity and the EKF error can converge after 3000 seconds (0.6 orbital period) even though the initial attitude error is large. The designed ADCS can fulfill the mission requirement by achieving attitude determination error of less than 2° in less than 6 orbital periods (about 5400 sec) although the requirement was 10 orbital periods.

10.2Future Work

In the future, some additional subjects will be investigated. The sensors and magnetometer should undergo calibration tests in order to increase the precision of the simulated sensor model. The Extended Kalman filter (EKF) was superseded by unscented Kalman filter that can be used as the algorithm for attitude estimation. Furthermore, A Test Bench for the Satellite should be developed as those test benches offer many prospects for ground testing of Cube satellites and their subsystems, and first of all Attitude Determination and Control System (ADCS). The difficulty of ground satellite tests is to create conditions simulating space environment with given accuracy while limiting the effects occurred on ground facilities. For Cube satellites, the accuracy of ground tests has to be very high due to reduced capability of actuators and high sensitivity to external forces. The main challenge of this research is to find a design solution allowing a Cube satellites to rotate freely around three axes with minimum disturbances from gravity and friction forces. We propose two ways to deal with these difficulties and avoid any limit in rotational motion. The first one is the active compensation of parasitic moments acting on Cube satellite during the tests. The second one consists of an air bearing sphere structure providing a contactless suspension for all possible angular position of the Cube satellites. In prospect, both approaches can be used together in one test bench to improve its efficiency.

Also, this test bench should utilize Helmholtz coil to generate a magnetic field similar to that of the earth, that magnetic field would be used in generating the actuating torque on the satellite and to simulate magnetometer measurements.

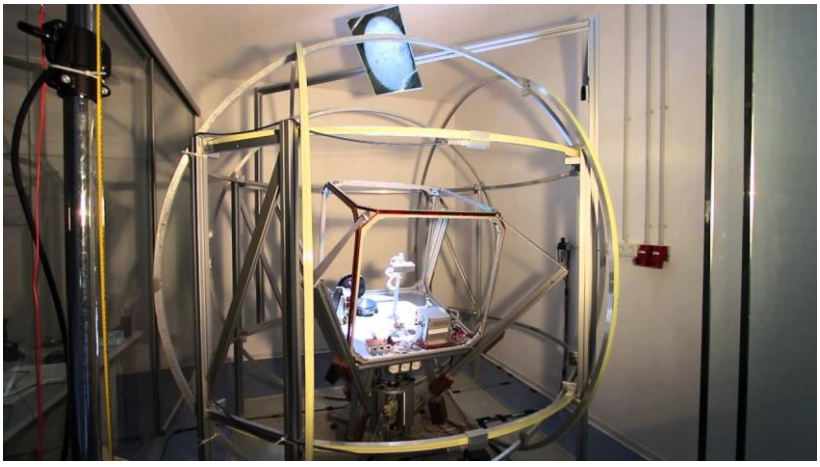


Figure 158| Helmholtz Test Ben

References

[1] *Sidi, M. J. (2006). Spacecraft dynamics and control: a practical engineering approach. Cambridge: Cambridge Univ. Pr..*

[2] *Euler Angles,Quaternions,and Transformation Matrices (by Lyndon B. Johnson Space Center).*

[3] *Vallado, D. A. (2007). Fundamentals of astrodynamics and applications. New York: Springer..*

[4] *Wertz, J. R. (2002). Spacecraft attitude determination and control. Dordrecht: Kluwer Academic..*

[5] Mekky, T. (n.d.). New algorithms of nonlinear spacecraft attitude control via attitude, angular velocity, and orbit estimation based on the earth's magnetic field. Faculty of Engineering at Cairo University..

[6] Julier, S., & Uhlmann, J. (2004). Unscented Filtering and Nonlinear Estimation. Proceedings of the IEEE, 92(3), 401-422. doi:10.1109/jproc.2003.823141.

[7] Brown, R. G., & Hwang, P. Y. (1992). Introduction to Random signals and applied Kalman filtering by Robert G. Brown. New York: J. Wiley..

[8] A.Farrag. A Combined Attitude Magnetic Controller for Remote Sensing.

[9] D. Kalman, "Mathematical Assosiation of America," [Online]. Available: https://www.maa.org/external_archive/joma/Volume8/Kalman/General.html.

Appendix

1. C Functions used for the ADCS Software Code

Function Name	AttitudeDeterminationAndControlSoftware
Description	This is the Main on Board Software function that performs Attitude Determination and Control of the Satellite.
C Arguments	Inputs: PreviousModeOfOperation → Pointer to Enum TimeSinceSimulation → Pointer to variable TimeStep → Pointer to variable UniversalTime → Pointer to Array ControlMoment → Pointer to Array CurrentObserverStateSpaceVector → Pointer to Array NextStateSpaceVector → Pointer to Array NextMeasuredMagneticFieldInBody → Pointer to Array NextMeasuredAngularVelocityInInertial → Pointer to Array EstimatedErrorFullCovarianceMatrix → Pointer to Array RequiredAngleAcquisition → Pointer to Array ImagingRequestType → Pointer to Enum TimeInIdealMode → Pointer to Variable SatelliteParametersObject → Pointer to Structure Outputs: CorrectedStateSpaceVectorNextLoop → Pointer to Array CorrectedErrorFullCovarianceMatrix → Pointer to Array NextEstimatedErrorFullCovarianceMatrix → Pointer to Array CurrentModeOfOperation → Pointer to Enum DipoleMoment → Pointer to Array

****

Function Name	SatelliteModeOfOperation
Description	This Function is used to determine the Current Mode of Operation of the satellite which may be needed to choose the suitable Filter, controller and Sensors for this mode
C Arguments	Inputs: PreviousModeOfOperation → Pointer to Variable TimeInIdealMode → Pointer to Variable TimeStep → Pointer to Variable ImagingRequestType → Pointer to Enum SimulationStateSpaceVector → Pointer to Array RequiredAngleAcquisition → Pointer to Array SatelliteParametersObject → Pointer to Structure Outputs: CurrentModeOfOperation → Pointer to Enum

****

Function Name	ObserverModeOfOperation
Description	This function is used to Determine which type of Filtering techniques to be used based on the Current mode of operation of the Satellite, and outputs the Corrected states
C Arguments	Inputs: CurrentModeOfOperation → Pointer to Enum PreviousModeOfOperation → Pointer to Enum TimeSinceSimulation → Pointer to Variable TimeStep → Pointer to Variable UniversalTime → Pointer to Array ControlMoment → Pointer to Array CorrectedStateSpaceVector → Pointer to Array MeasuredMagneticFieldInBody → Pointer to Array MeasuredAngularVelocityInInertial → Pointer to Array EstimatedErrorFullCovarianceMatrix → Pointer to Array SatelliteParametersObject → Pointer to Structure Outputs: CorrectedQuaternion → Pointer to Array CorrectedAngularVelocityInInertial → Pointer to Array CorrectedErrorFullCovarianceMatrix → Pointer to Array NextEstimatedErrorFullCovarianceMatrix → Pointer to Array CorrectedPositionInInertial → Pointer to Array CorrectedLinearVelocityInInertial → Pointer to Array

****

Function Name	ControllerModeOfOperation
Description	This function is used to Determine which type of Control Algorithm to be used based on the Current mode of operation of the Satellite, and outputs the Control Action Dipole Moment
C Arguments	Inputs: CurrentModeOfOperation → Pointer to Enum TimeSinceSimulation → Pointer to Variable TimeStep → Pointer to Variable CorrectedAngularVelocityInInertial → Pointer to Array CorrectedQuaternion → Pointer to Array MeasuredMagneticFieldInBody → Pointer to Array RequiredAngleAcquisition → Pointer to Array SatelliteParametersObject → Pointer to Structure Outputs: DipoleMoment

****

Function Name	RungeKutta
Description	This function uses the Runge kutta Integration Technique and the differential equation to propagate the state space vector through time
C Arguments	Inputs: TimeStep| → Pointer to Variable TimeSinceSimulation → Pointer to Variable CorrectedStateSpaceVector → Pointer to Array UniversalTime → Pointer to Array ControlMoment → Pointer to Array SatelliteParametersObject → Pointer to Structure Outputs: EstimatedStateSpaceDiffrentiationVector → Pointer to Array EstimatedStateSpaceVetor → Pointer to Array

****

Function Name	KalmanFilter
Description	This Function is used to Estimate and Correct the states measured by sensors which is subjected to Noise Depending on the Filtering technique choose by the observer mode of operation
C Arguments	Inputs: KalmanFilterType → Pointer to Enum MeasuredMagneticFieldInBody → Pointer to Array MeasuredAngularVelocityInInertial → Pointer to Array EstimatedMagneticFieldInOrbital → Pointer to Array EstimatedQuaternion → Pointer to Array EstimatedAngularVelocityInInertial → Pointer to Array EstimatedErrorCovarianceMatrix → Pointer to Array SatelliteParametersObject → Pointer to Structure Outputs: CorrectedQuaternion → Pointer to Array CorrectedAngularVelocityInInertial → Pointer to Array CorrectedErrorFullCovarianceMatrix → Pointer to Array NextEstimatedErrorFullCovarianceMatrix → Pointer to Array

****

Function Name	DiffrentialEquation
Description	This function is used to calculate the Rates of all the state space vector based on the Dynamics and Kinematics Equations
C Arguments	Inputs: TimeSinceSimulation → Pointer to Variable StateSpaceVector → Pointer to Array UniversalTime → Pointer to Array ControlMoment → Pointer to Array SatelliteParametersObject → Pointer to Structure Outputs: StateSpaceDiffrentiationVector → Pointer to Array

****

Function Name	SiderealTime
Description	This Function is used to Calculate the Greenwich Mean Sidereal Time
C Arguments	Inputs: UniversalTime → Pointer to Array ObserverLocalLongitude → Pointer to Variable Outputs: GreenwichMeanSiderealTime → Pointer to Variable LocalSiderealTime → Pointer to Variable

****

Function Name	GeomagneticField
Description	This Function is used to Calculate the magnetic field of earth In Greenwich Coordinates
C Arguments	Inputs: PositionVectorInGreenwish → Pointer to Array UniversalTime → Pointer to Array Outputs: MagneticFieldIntensityInGreenwish → Pointer to Array

****

Function Name	SphericalHarmonics
Description	This Function is used to Calculate the Gravity Vector of earth in Greenwish Coordinates
C Arguments	Inputs: PositionVectorInGreenwish → Pointer to Array Outputs: GravityVectorInGreenwish → Pointer to Array

****

Function Name	SunVectorInInertialCoordinates
Description	This Function is used to Calculate the Sun Vector in Inertial Coordinates to be used in calculating the Solar Disturbance
C Arguments	Inputs: UniversalTime → Pointer to Array Outputs: EarthSunVectorInInertial → Pointer to Array SunRightAscension → Pointer to Variable SunDeclination → Pointer to Variable

****

Function Name	SetSatelliteParameters
Description	This Function is used by the user to set all the Parameters related to the satellite like Mass, Moment of Inertia , Dimensions …etc
C Arguments	Inputs: Outputs: SatelliteParametersObject → Pointer to Structure

****

Function Name	AttitudeDisturbance
Description	This Function is used to Calculate the Total disturbances on the satellite by adding the Solar, Aerodynamics and Magnetic Disturbances
C Arguments	Inputs: MagneticFieldIntensityInBody → Pointer to Array LinearVelocityVectorBody → Pointer to Array SatelliteSunVectorInBody → Pointer to Array PositionVectorInBody → Pointer to Array SatelliteParametersObject → Pointer to Structure Outputs: TotalAttitudeDisturbanceMoment → Pointer to Array

****

Function Name	MagneticDisturbance
Description	This Function is used to Calculate the magnetic disturbances on the satellite due to the permanent Magnetism of the Satellite
C Arguments	Inputs: MagneticFieldInBodyCoordinates → Pointer to Array SatelliteParametersObject → Pointer to Structure Outputs: MagneticDisturbanceMoment → Pointer to Array

****

Function Name	AerodynamicDisturbance
Description	This is used to Calculate the Aerodynamic disturbances on the satellite due to the Drag on the satellite in its low Earth orbit
C Arguments	Inputs: SatelliteVelocityInBodyCoordinates → Pointer to Array SatelliteParametersObject → Pointer to Structure Outputs: AerodynamicDisturbanceMoment → Pointer to Array

****

Function Name	SolarDisturbance
Description	This is used to Calculate the Solar disturbances on the satellite due to incident Radiation on the surface of the Satellite
C Arguments	Inputs: SunVectorInBodyCoordinates → Pointer to Array EarthVectorInBodyCoordinates → Pointer to Array SatelliteParametersObject → Pointer to Structure Outputs: SolarDisturbanceMoment → Pointer to Array

****

Function Name	EclipseClassification
Description	This Function used to Calssify the Type of Eclipse the Satellite experience
C Arguments	Inputs: SunVectorInBodyCoordinates → Pointer to Array EarthVectorInBodyCoordinates → Pointer to Array Outputs: EclipseType → Pointer to Enum

****

Function Name	**`JulianCenturiesElapsedSinceJ200Epoch`**
Description	This is used to Calculate the Julian Centuries since Epoch J2000 to be used in Siderial time calculations
C Arguments	Inputs: UniversalTime → Pointer to Array Outputs: JulianCenturiesElapsed → Pointer to Array

****

Function Name	SphericalHarmonics
Description	This function is used to Calculate the Gravity Vector of earth on the Satellite in Greenwich Coordinates
C Arguments	Inputs: PositionVectorInGreenwish → Pointer to Array Outputs: GravityVectorInGreenwish → Pointer to Array

****

Function Name	SatelliteDynamicsAndKinematics
Description	This Function is used to Calculate The derivatives of the Angular velocity and the Quaternion using the Dynamics and Kinematics Equations Respectively
C Arguments	Inputs: AngularVelocityVectorInInertial → Pointer to Array Quaternion → Pointer to Array TotalAttitudeDisturbanceMoment → Pointer to Array ControlMoment → Pointer to Array SatelliteParametersObject → Pointer to Structure Outputs: AngularVelocityDiffrentiation → Pointer to Array QuaternionDiffrentiation → Pointer to Array

****

Function Name	AngularVelocitySuppression
Description	This Function is used to Calculate the dipole moment used for Angular Suppression used in Detumbling Mode of the satellite using $\dot{B}$ Technique
C Arguments	Inputs: TimeSinceSimulaion → Pointer to Variable TimeStep → Pointer to Variable MagneticFieldInBodyCoordinates → Pointer to Array SatelliteParametersObject → Pointer to Structure Outputs: DipoleMoment → Pointer to Array

****

Function Name	AttitudeAcquisitionAndStabilization
Description	This Function is used to Calculate the dipole moment and the Corrected States used for Attitude Acquisition And Stabilization Using PD-Technique
C Arguments	Inputs: TimeSinceSimulaion → Pointer to Variable AngularVelocityBodyToInertial → Pointer to Array Quaternion → Pointer to Array MagneticFieldInBodyCoordinates → Pointer to Array RequiredAngleAcquisition → Pointer to Array SatelliteParametersObject → Pointer to Structure Outputs: DipoleMoment → Pointer to Array

****

Function Name	VelocityFeedbackController
Description	This function is used to calculate the Dipole Moment required to stabilize the Satellite during the preparation Mode of the Kalman Filter
C Arguments	Inputs: MagneticFieldInBody → Pointer to Array AngularVelocityInInertial → Pointer to Array Outputs: DipoleMoment → Pointer to Array

****

Library Name	MathematicalLibrary
Description	This library contains the Mathematical Functions that are used throughout the code
C Functions	ComplexAddition ComplexMultiplication VectorNorm VectorNormalization DotProduct CrossProduct QuaternionNormalization QuaternionMagnitude QuaternionConjugate QuaternionMultiplication MatrixMultiplication MatrixInverse MatrixDeterminate Factorial IdentityMatrix Sign MatrixTranspose MatrixExponential Absolute

****

Library Name	AxesTransformation
Description	This library contains Functions that is used to transform any vector from different Axes coordination's like Body Coordinates, Orbital Coordinates, Inertial Coordinates and Greenwich Coordinates
C Functions	BCS_OCS → From Body to Orbital OCS_BCS → From Orbital to Body GSC_ICS → From Greenwich to Inertial ICS_GSC → From Inertial to Greenwich ICS_OCS → From Inertial to Orbital

****

Library Name	QuaternionTransformation
Description	This library contains Functions that is used to transform any Quaternion to Euler angles and any Euler Angles to Quaternion
C Functions	QuaternionToEulerAngles EulerAnglesToQuaternion

****

ملخص المشروع

يغطي المشروع جانب تطوير وبناء برنامج مدمج (embedded software) مع نظام متكامل للتحقق من كفاءة وعمل هذا البرنامج وذلك عن طريق مراحل مختلفة تتضمن (Model-In-The-Loop) و (Software-In-The-Loop) و أخيراً(Hard-Ware-In-The Loop) وذلك لـ "نظام تحديد الوجهة والتحكم بها لقمر صناعي ADCS " من فئة كيوبسات (Cube-Sat).

بالإضافة إلى ذلك ، تم بناء برنامج محاكاة مزود بوسائل عرض متطورة ونافذة للمستخدم قابلة للتفاعل وذلك باستخدام برنامج MATLAB. الهدف الرئيس للـ ADCS هو ضمان التحكم والاتزان للقمر الصناعي، سواء كان ذلك في مرحلة ال (de-tumbling) حيث يتم كبح السرعة الدورانية للقمر ، أو في مرحلة توجيه الكاميرا المحمولة على القمر الصناعي نحو هدف محدد على سطح الأرض ، وذلك أثناء ما يسمى بمرحلة التصوير (imaging).

يتحكم برنامج الـ ADCS بمنشئات العزم المغناطيسي (magnetic torquers)، والتي هي الجزء الميكانيكي الوحيد المسؤول عن إنشاء الحركة الميكانيكة على القمر، وذلك بمساعدة مستشعرات الحركة الدورانية (Gyro) ومستشعرات المجال المغناطيسي (Magnetometers) خلال مختلف أوضاع تشغيل القمر بما في ذلك ال (de-tumbling, attitude accusation, imaging and stand-by) ، ويتم ذلك باسخدام أنظمة تحكم مختلفة مثل (B-Dot and PD-like). علاوة على ذلك ، يستخدم البرنامج لوغاريتم ال Extended Kalman Filter وذلك بهدف توقع وجهة القمر وسرعته الدورانية بغرض استخدامها في لوغاريتم التحكم.

إضافة إلى ذلك، يستخدم برنامج المحاكاة واجهة مستخدم سهلة التعامل وذلك بهدف إدخال مختلف البيانات التي يحتاجها برنامج ال ADCS لبدء العمل. بعد ذلك ، يقوم برنامج المحاكاة بمحاكاة حركة القمر الصناعي أثناء دورانه في مداره وعرض مختلف أنواع النواتج سواء في صورة رقمية أو مرئية.

جامعة القاهرة
كلية الهندسة
قسم الطيران والفضاء

تصميم وتنفيذ

نظام تحديد الوجهة والتحكم بها لقمر صناعي

تنفيذ
أحمد عادل أحمد الرويني
عمرو حسام الدين علي
عمرو وحيد إبراهيم
محمد عادل أنيس
مصطفى سيد عبد الحميد

تحت إشراف

أ.د. جمال البيومي