UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

Software design for a Cubesat's Orbit Simulator

REPORT

Author: Irene Guitart Rosselló
Director: Javier Gago Barrio
Co-director: Miquel Sureda Anfres

Aeropace Technologies Engineering
Bachelor's Degree Thesis

Dateline: 15th of January, 2020

ESEIAAT
UPC

To my family

Contents

List of Figures

List of Tables

List of Abbreviations

ECI Earth Centered System.

GEO Geostationary Orbit.

GSO Geosyncronous Orbit.

HEO High Earth Orbit.

LEO Low Earth Orbit.

VLEO Very Low Earth Orbit.

MEO Medium Earth Orbit.

NASA National Aeronautics and Space Administration.

ESA European Space Agency

NORAD North American Aerospace Administration.

TLE Two-Line Element.

UTC Coordinated Universal Time.

GMST Greenwich Mean Sidereal Time.

JD Julian Dates

List of Symbols

α_{sun} Incidence of the sun over a *Cubesat's* face.

α_u Angle of umbra condition.

Ω Right Ascension of the Ascending Node.

ω Argument of the periapsis.

i Orbit Inclination.

e Eccentricity.

M Mean anomaly.

n Mean motion.

ν True anomaly.

a Semimajor Axis.

R_E Earth Radius.

R_S Sun Radius.

M_E Earth's Mass.

ω_E Earth's Angular Velocity.

G Gravitational Constant.

μ Standard Gravitational Parameter.

P Orbital Period.

θ_u Angle between umbra vector and satellite to umbra vector.

λ Latitude.

ϕ Longitude.

ξ Scalar product between sun vector and satellite coordinate vector.

T Perifocal Passage.

q Perigee distance.

L Mean Longitude of the Sun.

g Mean Anomaly of the sun.

λ_s Ecliptic Longitude of the Sun.

ϵ Obliquity of the Ecliptic.

Chapter 1

Introduction

1.1 Aim

The main objective of this study is to design a MATLAB software to simulate a constellation of *Cubesats* orbiting around the Earth in LEO or VLEO orbits. It will be estimated the exact position of each satellite and their Ground Track. Moreover, the Solar Array prediction over their faces will be studied.

1.2 Scope

The depth of this work is based on the following points:

- Brief study of the satellites, constellations and their types of orbits.
- Advantages and disadvantages of LEO constellations.
- Deep study in the tracking problem.
- Analyze the Solar Array Architecture.
- Parametrization of all the variables involved in the tracking problem.
- Design of a MATLAB algorithm to simulate orbits which will be able to determine the exact position of a satellite,and its exposition in front of the sun (due to a satellite completly exposed in front the sun is not able to comunicate).

To carry out the states mentioned previously, the following deliveries will be done:

- **Report:** document that will include all relevant information on the development of the project. It includes a theoretical first part, where the essential information needed to be able to understand the work is explained. Next, a practical part, which explains the resolution of the problem. And finally, a part where the final results and conclusions that have been drawn from them are presented.
- **Budget:** attachment that provides the costs of that project.
- **Annex:** final part where the MATLAB code developed is presented and explained.
- **Files** it will be delivered too all the files with the MATLAB code.

1.3 Requirements

To complete the last items, it is necessary:

- A brief introduction about the satellites, their classification and uses. Specifically, what are the *cubesats*.
- The Earth will be considered as a perfect sphere and the satellite's orbit will be circular in this study.

- Understand the time representations to developed an appropriate simulation.
- The *cubesats* studied will be in Low Earth Orbits (LEO) or in Very Low Earth Orbits (VLEO).
- Orbit Perturbations such as the atmospheric drag will be neglected.
- The comunication between *cubesats* will be through TLE messages.
- The coordinates system choosed to represent the satellite's position will be the Earth Centered Inertial System.

1.4 Justification

This study has been proposed for the group of research *DISEN (2017 SGR 1170)*, located at the UPC base in Terrassa. It is included inside of a research project focus on the development of a simulation and emulation platform' of a nanosatellites constellation. It follows up two past bachelor's thesis which studied the visibility problem between the Ground Station and a satellite, and the visibility problem between satellites. The present study completes the problem defining the exact position over the Earth of the satellites and predicting the sun incidence over the *Cubesats* faces. The main purpose of that is to identify where the satellite will be able to communicate or not. The spacecrafts become saturated when the sun is impacting over the face where the communication module is located, thus, it is not operable.

The satellites orbiting near the Earth are exposed to a big number of problems of visibility windows due to the vicinity to the planet. For this reason, this research will help to optimize the communication with LEO spacecrafts. Therefore, *DISEN Group* pretends to develop a network of optical communication taking advantage of the reduced distance to the object. This is translated in a minor cost of energy to launch it and put in to orbit for example.

1.5 Project Plan

This type of projects require a previous plan where the steps to follow to success are defined. Hereunder there is a table with the different tasks that have been estimated before starting the project.

Task Code	Task Description	Preceding Task
0	Brief initial information research	
1	**THEORETICAL PART**	**0**
1.1	Study of the satellites, their constellations and types of orbits	0
1.2	Brief description of *Cubesats*	0
1.3	Advantages and Disadvantages of LEO orbits	0 - 1.1 - 1.5
1.4	Real Live Tracking platforms	0 - 1.3
1.5	Define the time system representations	0 - 1.3
1.6	Define the space environment (Sun and satellite's coordinates)	0 - 1.1
2	**SOFTWARE DEVELOPMENT**	**1**
2.1	Develop an algorithm which solves the satellite's tracking problem	1.4 - 1.5
2.1.1	Develop an algorithm which estimates the sun position	2.1
2.1.2	Give solution to the response of the satellite in front of the sun	2.1
2.2	Transform the code in a simulation	2.1
2.2.1	Include a visuals module to plot in an elegant way the results	2.1- 2.2
3	**VALIDATION**	**2**
3.1	Find cases with unique solutions	2
3.2	Comparison with a professional software a GEO case	3.1
3.3	Comparison with a professional softwer a LEO case	3.1
4	**DRAFTING THE THESIS**	**0 - 1 - 2 - 3**
4.1	Project Charter	0
4.2	Follow - up 1	
4.3	Follow - up 2	
4.4	Follow - up 3	
4.5	Report	0 - 1 - 2 - 3
4.6	Annexes	2
4.7	Files	2 - 3
5	**FINAL PART**	**4**
5.1	Deliver the inform	4
5.2	Final oral presentation	4

Table 1.1: Interdependence relationship among tasks.

Figure 1.1: Proyect Gant.

Chapter 2

State of Art

This first chapter describes the main characteristics of satellites, in particular the *cubesats*. The types of orbits will also be defined emphasizing in the LEO and VLEO orbits, which are the subject of this study. Finally, as the thesis is involved in a project which pretends to develop satellite's pass predictions, a real live tracking software will be presented. It has been very useful to validate the software object of this study.

2.1 Satellites

The word satellite encompasses both celestial bodies and machines orbiting a planet or star. Thus, there are those that are natural satellites, such as the Moon orbiting around the Earth; or an artificial satellite, which are machines created and launched by man.

Currently artificial satellites take part in day a day life, since they lean on telecommunication systems, gps, meteorological prediction... [4]

On October 4, 1957, what would become the beginning of the "Space Race" was successfully completed. The Soviet Union managed to put Sputnik 1 into orbit. This fact revolutionized the planet, making the Soviet Union a pioneer in launching man-made objects into space. [9]

Figure 2.1: Sputnik 1 picture, first artificial satellite put in orbit. [9]

Artificial satellites are classified according to their specific function. However, they all have at least four essential components: an antenna, a power supply, the payload and an altitude control subsystem.

Firstly, the antenna is highly important in order to start communication with the satellite from the ground station. Without it, all the collected data would not be useful and instructions could not be sent to it. Secondly, the power supply, without which the purposes of the satellite could not be carried out. Thirdly, the payload, which, despite not being the essential part that keeps the satellite active, is the cause of its construction. And the last, the

altitude control subsystem for satellite location control.

The classification of the satellites is long and extensive, as they can be divided according to many characteristics (shape, size, location, mission, power supply...). Three classification of satellites are relevant for this study: their size, location and application. [10]

According to their size, the can be divided in:

- **Large Satellites:** more than 1000 kg.
- **Medium Satellites:** between 500 kg and 1000 kg.
- **Minisatellites:** between 100 kg and 500 kg.
- **Microsatellites:** between 10 kg and 100 kg.
- **Nanosatellites:** between 1 kg y 10 kg. The *Cubesats* are included in this group. They will be described in the following section.
- **Picosatellites:** between 0.1 kg and 1 kg.
- **Femtosatellites:** less than 0.1 kg.

Regarding to their location, there are:

- Satellites in low-Earth orbits, known as ***Low Earth Orbit*** **(LEO)**. From 200 to 2000 km high.
- Satellites in medium-earth orbits, known as ***Medium Earth Orbit*** **(MEO)**. This type of orbits are located from 2000 km to just immediately below the Geosynchronous orbit, at approximately 35786 km.
- Satellites in **Geosynchronous Earth orbits (GEO)** o **Geostationary Orbit (GSO)**. In these orbits, the stars have a period equal to Earth's. The difference between them lies in the null inclination of the Geostationary orbit. The semi-major axis of the orbit measures 35786 km, which means the satellites are at that height.
- Satellites in high Earth orbits, known as ***High Earth Orbit*** **(HEO)** , which range from 35786 km on wards.

And depending on their application, there are:

- **Communication:** nowadays, telecommunications take advantage of the satellites for the data transmission. It is probably its greates utility, since telephone, TV, Internet... among others depend on this use.
- **Navigation:** satellites are capable to send very accurate data of position, velocity and time to the users
- **Meteorology:** weather stations use the data collected for the satellites of the current state of the atmosphere.
- **Military** uses.
- **Cientific research** uses.

2.1.1 Cubesats

The word *Cubesat* refers to satellites within the nanosatellite group. It is a simple type of device, as the name implies, they are shaped like a cube scalable to 10 cm of edge. The main module is based on a cube of 10 centimeters of edge, which has a volume of 1 liter and an approximate mass of 1 kilogram, known as 1U. Based on this module can be integrated in different configurations being the 3U (10 cm x 10 cm x 30 cm) one of the most common, allowing a small and modular design.

The reference design of the *Cubesat* was proposed in 1999 by Professor's Jordi Puig-Suari from the *California Polytechnich State University* and Professor's Robert 'Bob' Twiggs from *Stanford*. The goal was to allow graduated students to design, build, test and operate a satellite in space with similar capabilities as the first, Sputnik 1. The *Cubesat* was not intended to become a standard, rather, it was converted in a standard over time. The first *Cubesats* were launched in June 2003, in a Russian Eurockot until by 2008, approximately, 75 *Cubesats* had been placed in

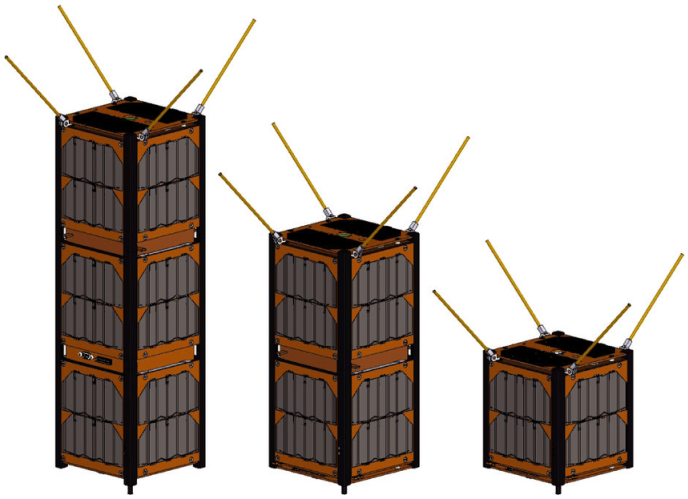


Figure 2.2: Three cubesats, ordered right to left: 1U, 2U y 3U. [17]

orbit.

The great advantage of the technology is the accessibility to space that it grants to different types of entities, from universities in developing countries to private companies, for its low cost. The basic space industry is nourished from the electronics on board on their satellites of components tested and trained to work in extreme environments in space. This robust and reliable design strategy involves some disadvantages in terms of costs and development times, and that's where the *Cubesats* come into play.

Another main advantage from the *Cubesats* is that their development is based in COTS Technologies (*Comercial Of The Shelf*), without being previously qualified for their use in fly. This strategy provides of design flexibility, decrease costs and times of development in exchange of reduce the life expectancy (often not required since its common use is usually in LEO orbits).

The applications for these technologies are very diverse. They are mainly classified as a test bench for technology qualification in space, teaching and training and, finally, scientific missions. Thanks to the participation of various institutions, their development grows daily.

To all of this, the reduction of costs in the integration, launch and orbit phase is joined. As it is a small device compared to the main load, it is usually used by the availability of the various launchers in the usual releases, to attach *Cubesats* to the main load of the same, without the latter having to modify their settings or add fuel charges in exchange.
On the other hand, they also have limitations. One is the low maneuverability of the platforms, as there are currently very few satellites with propulsion inside. The *Cubesats* platforms are based on only ADCS (*Attitude Determination and Control System*).

And, lastly, another disadvantage, is the limited useful rate of downstream data. Specially, this thesis is contained in a project that studies the extension of the range of transmission frequencies to optics, much faster than the ones usually used. [17]

2.2 Orbits

As described in the satellite classification, there are four basic types of orbits: LEO, MEO, GEO/GSO and HEO. It should be clarified, that these are Earth orbits, as there are also satellites orbiting other planets such as Venus, Mars, Uranus or even small planets, asteroids and comets, which are located outside our field of study.

2.2.1 Context

One can not talk about orbits without mentioning the three modernist scientists who allowed us to understand orbital mechanics. The Danish astronomer, Tycho Brahe (1546-1601), is considered the greatest observer of the sky in the period before the invention of the telescope. All his work was inherited by his then fellow, Johannes Kepler (1571–1630), a German astronomer. As a result, Kepler was able to define his three laws of orbital motion:

- **First Kepler's Law or The Law of Ellipses(1609):** the path of the planets about the sun is elliptical in shape, with the center of the sun being located at one focus.

- **Second Kepler's Lay or The Law of Equal Areas(1609):** the line joining the planet to the Sun sweeps out equal areas in equal times.

  The law of the areas is equivalent to the conservation of the angular momentum, it means that, when the planet is farther from the Sun (Aphelion) its speed is less than when it is closer to the Sun (Perihelion

  Aphelion and Perihelion are the exclusive two points in the orbit where the radius and velocity vectors are perpendicular. Therefore, only at these 2 points, the module of the angular momentum can be estimated , L, as the product of the planet's mass by its speed and its distance to the center of the Sun.

  $$L = m \cdot r_a \cdot v_a \tag{2.1}$$

  At any other point in the orbit, different of Aphelion or Perihelion, to compute the angular momentum is more complicated, because the velocity is not perpendicular to the vector radius, and the cross product must be used

  $$L = m \cdot r \times v \tag{2.2}$$

- **Third Kepler's Law or The Law of Harmonies (1618):** the square of the period of any planet is proportional to the cube of its mean distance from the Sun.

  $$\frac{T^2}{a^3} = C \tag{2.3}$$

  where, **T** is the orbital period (time it takes to complete an orbit around the planet), **a** mean distance between the planet and the sun and **C** is a proportional constant.

  The most important law, is probably the third one.

These laws apply to other astronomical bodies that are in a minimum gravitational influence, such as the system formed by the Earth and the Sun.

Later, Isaac Newton (1642–1727) demonstrated that Kepler's laws stemmed from his theory of gravity and that, in general, the orbits of the bodies responding to gravitational force were conical sections. It showed that a couple of bodies follow orbits of dimensions that are inversely proportional to their masses over their common mass center. So when the mass of one of the bodies is negligible respecto to the other body, the rule makes taking the center of mass as the center of the body with greater mass. [6]

2.2.2 Keplerian Elements

Kepler's laws are the foundation for the Keplerian elements which are much better suited for use in estimating a satellite's orbit and position. The comprehension of these elements is crucial to the understanding of the orbit estimator and is therefore presented here. The Keplerian elements, sometimes called classical orbital elements, define an ellipse, orient it about the earth, and place that satellite on the ellipse at a particular time. In the Keplerian model, satellites orbit in an ellipse of constant shape and orientation. The Earth is at one focus of the ellipse, not the center, unless the orbit ellipse is a perfect circle and the two focus coincide with the center [21]. There are six Keplerian elements for an Earth Satellite, and they are:

- **Semimajor axis (a):** half the long axis of the ellipse.

- **Eccentricity (e):** it defines the shape of the orbit. It is a measure on how elongated is the orbit compared to a circular orbit. When is equal to 0, it means a circular orbit; between 0 and 1, elliptical; 1, parabolic; and more than 1 hyperbollic.

- **Orbital Inclination (i):** angle between the orbit plane and the earth's equatorial plane.

- **Right Ascension of the Ascending Node (Ω):** angle measure at the center of the Earth in the equatorial plane from the vernal equinox eastward to the ascending node (i.e the point as which the satellite crosses the equator going from south to north).

- **Argument of perigee (ω):** angle measured at the center of the Earth in the orbit plane from the Ascending node to perigee (closest approach of the orbit of the satellite to the Earth) measured in the directon of the satellite's motion.

- **True anomaly(ν):** defines the position of the orbiting body along the ellipse at a specific time.

For better understanding of the six Keplerian elements, see the following figure:

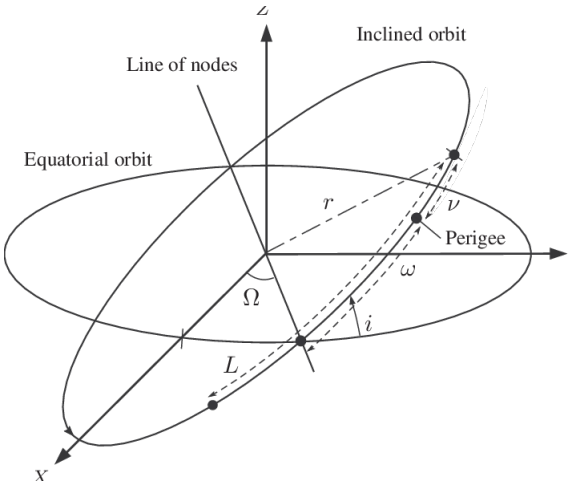


Figure 2.3: Keplerian Elements [21]

In addition, other important parameters are:

- **Parameter or semi-parameter (p):** distance from the primary focus to the orbit. Typically used to descrbe the size of a parabolic orbit because the value of the semi-major axis is infinite.

- **Periapsis distance (q):** distance from the primary body to the closest point of the orbit.

- **Mean anomaly (M):** the fraction of an elliptical orbit's period that has elapsed since the orbiting body passed periapsis.

- **Mean motion (n):** angular speed requiered for a body to complete one orbit, assuming constant speed in a circular orbit which completes in the same time as the variable speed, elliptical orbit.

- **Period (P):** time to complete an orbit - one revolution - around the primary body.

- **Time of perifocal passage(T):** last time the satellite passed through the periapsis. Used as a reference fo the time of interest.

2.2.3 Types of orbits

The classification of the orbits in which artificial satellites orbiting the Earth are located is not unique. There are many ways to classify them. However, one of the most important and used forms is classification based on altitude relative to the Earth's surface to which a body is orbiting it. Following the same classification of the satellites according to the orbit in which they are located, the orbits are divided into four main groups. The following table shows the most important characteristics of each type: [2]

Type of orbit	Height	Period	Use	Inclination	Shape
LEO (0-20.000 Km)					
Sun-syncronized Polar Orbit	150 - 900 Km	98 - 104 min	International Spacial Station,	98º	circular
Non-Inclined Polar Orbit	340 Km	91 min	Meteorology	51.6º	
Non Sun-Synchronized Polar Orbit	450 - 600 Km	90 - 101 min	Earth Scientific Observation	80-94º	
MEO (20.000-35.786 Km)					
Semi-synchronous	20.100 Km	12 hours	Spacial Environment, communication, navigation		circular
GEO					
Geosynchronous	35.786 Km	24 h (23h 56' 04s)	Communication, meteorology, nuclear detection,	0º	circular
Geostationary					
HEO (>35.786 Km)					
Molinya	Varies between 495 Km y 39.587 Km	12h (11h 58min)	Communication	63.4º	ellipse

Table 2.1: Orbit's classification by height.

First, Geosynchronous orbits (GEO) are defined. These orbits have a semi-major axis of approximately 42,164 km from the center of the Earth equivalent to approximately 35,736 km above sea level. A complete revolution has a period equal to the earth's intrinsic period, 24 hours. In GEO you can find most communication and meteorology satellites, which try to avoid the rotation of ground stations located on the Earth's surface. GEO is included in the group of high orbits, which range from 500 km to 39,500 km.

Lower altitude, we have Medium Earth Orbits. Historically, MEO has been used less than GEO or LEO, being the main cause the presence of radiation from Van Allen rings, an area loaded with particles that extends from a distance of 60,000 km from the Earth's surface) that occupy part of its area . In general, only navigation satellites are located in MEO.

Regarding the LEO ones, where the International Space Station is located, they are the first to be found in space, extending from 161 km to 2,000 km altitude. Its two main advantages are that they are very easy to achieve and in addition to staying in them. On the other hand, the short orbital periods they have (between approximately 88 and 127 minutes) make possible to collect a lot of information from a specific place in the same day.

The altitude of the orbit, or the satellite-Earth surface distance, is highly important in the movement of the body around the Earth. Earth's gravity is the greater influence the movement of satellites into Earth orbit has, so the closer they are to Earth, gravity gains ground and causes the satellite to go faster.

GEO and LEO are practically saturated by satellites, in addition with the problem of the space debris. Approximately the 95% of the orbiting objects around the Eart are space debris, in other words, non-functional satellites.

The objects placed in LEO are extremely influenced by the atmospheric drag due to their location in high shapes of the Earth's Atmosphere, inside the Thermosphere (80 km to 500 km). Higher the orbit's attitude is, the atmosphere drag is reduced, creating a disadvantage in the use of LEO orbits.

2.2.4 LEO, advantages and disadvantages

The satellites used in this thesis are located in LEO orbits. That is why it has been considered to look into them deeply. [12]
As it has been explained in the last section, the objects situated in LEO are bound to the difficulties caused by the aerodynamic forces, in particular, the atmospheric drag, as they are at the high shapes of the Earth's atmosphere. However, the advantages are far more extensive.

Satellites that observe our planet, such as remote sensing and climate satellites , often use LEO orbits because from their altitude they can capture accurate images of the Earth's surface. LEO satellite's orbit projects less coverage than GEO's, but LEO satellites are beneficial for communications by providing fast transmission time with little or no delay time. Signals are less susceptible to interference and degradation too. Also if one or more satellites, within a constellation, are disabled, there are a number of other satellites within it that can take and forward the message. On the other hand, starting a complex network is costly, requiring a large number of satellites and technology needed to link the satellite system.

The effectiveness of the satellites in LEO rely on the following factors:

- **Height:** The height above the earth determines the coverage that a satellite can provide. The higher the satellite, the larger the amount of area it can cover, but the longer the transmission time required. Therefore, the low altitude of a LEO satellite reduces transmission time at the cost of lower coverage.
- **Orbital Inclination:**

  The angle between the LEO satellite orbit and the equator also determines the effective coverage area (ensuring the ability to communicate at any point within that area). Satellites orbiting in zero degrees or close to zero degrees relative to the Earth's equator can provide good coverage to equatorial regions even if the polar regions receive zero or little coverage. On the other hand, satellites at larger angles relative to the Earth's equator can reach polar regions, but continuous communications can only be performed by increasing the number of satellites in orbit.
- **Number of Satellites:** The more interconnected LEO satellites we have within a constellation, the coverage area will be larger and a longer communication can be maintained with the satellites; one of the disadvantages is that it increases the cost as the number of satellites increases.
- **Orbits Saturation:** The Low Earth Orbit's zone (LEO), part of the Earth's atmosphere to a high-radiation area known as the Van Allen Belt. They are 900 km away which can accommodate an immense amount of routes. More than 60,000 satellites could then be placed without problems.
- **Space Debris:** Once LEO satellites' are in orbit, a new set of difficulties is presented. The problem is called *Space Debris*, which consist of remnants of preciuous space missions of all sizes, speeds and hazards. The satellites are susceptible to colliding with this space "junk".
- **Satellites lost and substitution :**

  Even if the satellites are not hit by the space debris, they may fall into the atmosphere. Unlike GEO, which, when they finish their lifespan, travels to a parking orbit a few kilometres away than normal, LEOs disintegrate into the atmosphere.
- **Satellites' Visibility:** Assuming these difficulties have been overcome, the matter remains of following the lead and linking with these fast satellites. A LEO satellite is visible for 18-20 min before it disappears on the horizon. This greatly complicates the positioning of the antenna and the work to keep the link active.

  The antenna's problem is solved by a technology called *Phased Array Antenna*. Unlike a normal satellite dish, which mechanically tracks the satellite trail, phased array antennas are self-directed devices that contain several smaller antennas that you can follow to several satellites without physically moving, by means of similar signals received by the antenna array, thus reducing wear, among other advantages.

  The problem of maintaining an active link when the satellite disappears every half hour is solved by keeping at least two satellites in sight at all times. The antenna array is aware of the position of all satellites and initiates a new link before cutting the existing one with the west satellite. In satellite jargon, this is called 'make before break'.

- **Addressing via inter-satellite links:** Another interesting problem is the direction of the signal between two points away from the Earth's surface.

  One possibility is to do it through ground stations, but that leads to loss of the advantage of reduced latency.

  The other possibility, which is the most commonly used, is a satellite-to-satellite address. The disadvantage of this method is, of course, that each satellite must have more communications and tracking hardware (more intelligence) and, therefore, its price will be higher than the case of using ground stations.

  To realize this thesis, the Constellation Iridium, a set of 66 communications satellites that rotate around the Earth in 6 low LEO orbits, has been used at a height of approximately 780 km from Earth. Each of the 6 orbits consists of 11 satellites equidistant from each other. Satellites take 100 minutes to go around the world from pole to pole.

2.3 Live Satellite Tacking Predictors

When the purpose is to establish communication with a satellite, the estimation of its position in the space becomes truly important. Nowadays there are a lot of tools which provide real live tracking of the orbiting spacecrafts and celestial objects in the Solar System. [8] Examples of these tools can be found in websites such as:

- *See A Satellite Tonight* shows you where to look with Google Street View.
- *Uphere.Spacereal* time tracking and predictions. As soon as new satellites are launched this application begins tracking them.
- *ISS Tracker*, since 2015 the most popular ISS Tracker and passes list.
- *Heavens-Above*, called "the most popular website for tracking satellites" in 2007 by *Sky and Telescope* magazine and referenced by *NASA* websites. Linked from both *NASA* and *ESA* websites as a reference for locating satellites and spacecraft. Includes predictions for the ISS, space shuttle and other bright satellites, planets, minor planets, and comets.
- *N2YO* provides real time tracking and pass predictions with orbital paths and footprints overlaid on Google Maps. It features an alerting system that automatically notifies users via SMS and/or email before International Space Station crosses the local sky. The N2YO.com system powers *ESA*'s, *Space.com*'s and many other's satellite tracking web pages.
- *NASA Skywatch*, *Java* based web application. Predicts visible passes for spacecraft, satellites and space debris.

All websites and applications that provide tracking data base their predictions on formula using two-line element sets which describe the satellites and their orbits. The most common websites that provide the two-line element message are:

- *Celestrack* provided by Dr. T.S. Kelso, includes visible objects, openly available.
- *Space Track* maintained by the United States Strategic Command provides orbital information on unclassified satellites requires an account but is available for educational and hobbyist use as well as military, government and spacecraft and payload owners.

The present study has used *N2YO* website to validate the simulator developed. Hereafter, *N2YO* will be briefly described.

2.3.1 *N2YO* for real live satellite tracking

The private company *N2YO* was founded in 2006. This website allows for satellite tracking in an easy formula since it does not necessary any additional software installation. You can search a satellite by introducing its name and live real track it. Satellites of all types can be tracked from many different categories e.g. military, weather, TV, radio,...or, simply, by most popular...or brightest...or just launched. [7]

N2YO is the webmaster's [1]*Ham Radio call sign.* [2]*Amateur radio*, often called ham radio, is a hobby enjoyed by many people throughout the world. An amateur radio operator, ham, or radio amateur uses two-way radio to communicate with other radio amateurs for public service, recreation and self-improvement.

The website displays the location of the satellite with its footprint, as shown in 2.4. It also shows information of the satellite that is currently being tracked. This also uses the TLE to track each satellite.

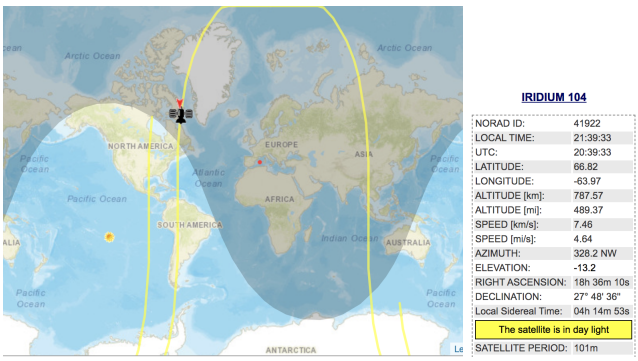


Figure 2.4: *N2YO* Satellite Tracking Website [7]

A very useful property that *N2YO* offers is that it predicts future satellite passes up to ten days in advance.

10-DAY PREDICTIONS by N2YO.COM

Object name IRIDIUM 104
Catalog # 41922, 2017-003F
Observing coord. Lat: 41.22°, Lng: 1.73°
Local time zone GMT +1

Start		Max altitude			End		All passes
Date, Local time	Az	Local time	Az	El	Local time	Az	Mag
13-Jan 02:54	NNE 17°	**03:00**	**ENE 73°**	**14°**	03:06	SE 128°	-
13-Jan 04:34	N 354°	**04:41**	**W 265°**	**69°**	04:48	S 186°	-
13-Jan 15:50	SSE 151°	**15:57**	**E 82°**	**28°**	16:04	NNE 16°	+7.3
13-Jan 17:31	SSW 207°	**17:38**	**W 279°**	**33°**	17:45	N 353°	+7.1
14-Jan 04:00	N 0°	**04:08**	**E 91°**	**65°**	04:15	S 171°	-
14-Jan 05:42	NNW 337°	**05:48**	**W 283°**	**14°**	05:53	SW 229°	-
14-Jan 15:18	SE 134°	**15:24**	**E 79°**	**16°**	15:30	NNE 23°	+8.0
14-Jan 16:57	S 191°	**17:04**	**W 272°**	**60°**	17:11	N 359°	+6.2
15-Jan 03:27	N 6°	**03:34**	**E 82°**	**35°**	03:41	SSE 155°	-
15-Jan 05:08	NNW 344°	**05:14**	**W 277°**	**26°**	05:21	SW 211°	-
15-Jan 16:23	S 176°	**16:31**	**E 92°**	**75°**	16:38	N 5°	+6.0
15-Jan 18:06	SW 234°	**18:12**	**WNW 288°**	**13°**	18:18	NNW 341°	+8.2
16-Jan 02:53	N 13°	**03:00**	**E 76°**	**20°**	03:07	SE 139°	-
16-Jan 04:34	N 351°	**04:41**	**W 272°**	**47°**	04:48	SSW 195°	-
16-Jan 15:50	SSE 160°	**15:57**	**E 84°**	**40°**	16:04	N 11°	+6.8
16-Jan 17:31	SW 216°	**17:38**	**W 283°**	**23°**	17:45	N 349°	+7.5
17-Jan 02:20	NNE 20°	**02:26**	**ENE 70°**	**11°**	02:32	ESE 119°	-
17-Jan 04:00	N 356°	**04:08**	**WNW 291°**	**87°**	04:15	S 180°	-
17-Jan 15:17	SE 145°	**15:24**	**E 81°**	**22°**	15:30	NNE 18°	+7.6
17-Jan 16:57	SSW 200°	**17:04**	**W 276°**	**41°**	17:11	N 355°	+6.7
18-Jan 03:27	N 2°	**03:34**	**E 85°**	**51°**	03:41	SSE 165°	-
18-Jan 05:08	NNW 340°	**05:14**	**W 282°**	**18°**	05:21	SW 221°	-

Figure 2.5: *N2YO* Satellite 10 days prediction tracking [7]

[1]Amateur radio call signs are allocated to amateur radio operators around the world. The call signs are used to legally identify the station or operator, with some countries requiring the station call sign to always be used and others allowing the operator call sign instead.

[2]Amateur radio, also known as ham radio, is the use of radio frequency spectrum for purposes of non-commercial exchange of messages, wireless experimentation, self-training, private recreation, radiosport, contesting, and emergency communication. The term "amateur" is used to specify "a duly authorised person interested in radioelectric practice with a purely personal aim and without pecuniary interest;"(either direct monetary or other similar reward) and to differentiate it from commercial broadcasting, public safety (such as police and fire), or professional two-way radio services (such as maritime, aviation, taxis, etc.).

Chapter 3

The time representation

Sometimes it is hard to understand how a simulation discretize the variable time. It must be expressed as a point in time in order to be used in a simulation algorithm. Moreover, as this projects analyzes the motion of a satellite around the Earth, it must be taken into account that the planet rotates at the same time as the satellites does. Thus, the time also will vary because of it. The next section will explain the astronomical times, it means, the ways to relate a point in time with the Earth's rotation; and then will define the procedure to express the time as a point. [1]

3.1 Astronomical Times

There are two widely used time standards. One is the rotation of the earth, and the other is the frequency of atomic oscillations (mainly the cesium133 atom). The earth's rotation is not uniform. Its rate exhibits both periodic changes and long term drifts on the order of a second per year. Atomic standards are the closest approximations we currently have to a uniform time with accuracies on the order of microseconds per year. Below are exposed the different types of each categories.

Atomic Times	**Earth Rotation Times**
TAI - Interantional Atomic Time	UT1 - Universal Time
UTC - Coordinated Universal Time	UT0
TDT or TT - Terrestrial Dynamic Time	UT2
BDT - Barycentric Dynamic Time	GMST - Greenwich Sidereal Time
	GAST - Greenwich Apparent Sidereal Time
	LMST - Local Mean Sidereal Time
	LST - Local Sidereal TIme

Table 3.1: Astronomical Times

The time discretization used in the simulation will transform a date into Unix time and finally, in order to be consistent with the Earth rotation, will refer the variable in a suitable Astronomical Time required for the computation that is being made. The code will need:

- **UTC:** is the time broadcast by *WWV* and other services. By definition, UTC and TAI have the same rate, but UTC stays close to Mean Solar Time by adding integer numbers of seconds, called leap seconds, from time to time. This keeps solar noon at the same UTC (averaged over the year), even though the rotation of the earth is slowing down. The offset is changed as needed to keep UTC within about 0.9 seconds of earth rotation time, UT1. Leap seconds are typically added once per year at the end of December or June, but they can be added (or subtracted) at other designated times throughout the year. The offset between TAI and UTC is currently 30 seconds, e.g., 20 : 00 : 00 UTC = 20 : 00 : 30 TAI.

$$\text{UTC} = \text{TAI} - (\text{number of leap seconds})$$

- **UT1:** is a measure of the actual rotation of the earth, independent of observing location. UT1 is essentially the same as the now discontinued Greenwich Mean Time (GMT). It is the observed rotation of the earth with

respect to the mean sun corrected for the observer's longitude with respect to the Greenwich Meridian and for the observer's small shift in longitude due to polar motion.

Since the earth's rotation is not uniform, the rate of UT1 is not constant, and its offset from atomic time is continually changing in a not completely predictable way. As of December 1995, UT1 was drifting about 0.8 seconds per year with respect to atomic time (TAI or UTC). Since UTC is intentionally incremented by integer seconds (leap seconds) to stay within 0.9 seconds of UT1, the difference between UT1 and UTC is never greater than this. The difference, $DUT1 = UT1 - UTC$ is monitored by the *International Earth Rotation Service* and published weekly in *IERS Bulletin A* along with predictions for a number of months into the future.

UT1 = UTC + DUT1 (from the IERS Bulletin A)

Note that when a leap second is added to or subtracted from UTC, the value of DUT1 is discontinuous by one second. UT1 is continuous, and UTC is incremented or decremented by integer seconds to stay witin 0.9 seconds of UT1.

- **GMST:** is the measure of the earth's rotation with respect to distant celestial objects. Compare this to UT1, which is the rotation of the earth with respect to the mean position of the sun. One sidereal second is approximately 365.25/366.25 of a UT1 second. In other words, there is one more day in a sidereal year than in a solar year.
  By convention, the reference points for Greenwich Sidereal Time are the Greenwich Meridian and the vernal equinox (the intersection of the planes of the earth's equator and the earth's orbit, the ecliptic). The Greenwich sidereal day begins when the vernal equinox is on the Greenwich Meridian. Greenwich Mean Sidereal Time (GMST) is the hour angle of the average position of the vernal equinox, neglecting short term motions of the equinox due to nutation.

  In conformance with IAU conventions for the motion of the earth's equator and equinox GMST is linked directly to UT1 through the equation

$$GMST = 24110.54841 + 8640184.812866 * T + 0.093104 * T^2 - 0.0000062 * T^3 \tag{3.1}$$

  where T is in Julian centuries from 2000 Jan. 1 12h UT1, next section will describe it.

$$T = d/36525 \tag{3.2}$$

$$d = JD - 2451545.0 \tag{3.3}$$

3.2 Julian Dates

Julian Date is a simply way to name moments, that is simpler than the customary date-plus-time system. Normally, to specify a date and time requires six different numbers - year, month, day, hour, minute and seconds - and comparing two dates takes a terrible amount of math with some extra complications as there are moths with 30 days, others with 31, for example. Julian Dates assign a simple floating-point number to each date-plus-time on the calendar, much easier to do math with.

JD counts the number of days since Jan. 1, 4713 BC 12:00 UTC in the Julian calendar. Sometimes, that is a bit hard to work with, since it is so long ago. For that, *Reduce Julian Date, (JRD)* exists. This counts the number of days since 2000 Jan. 1 12h UT1.

The calculation of *JD* may follow:

$$JD = 2451544,5 + 365Y + 0,25Y - 0,01Y + 0,0025Y + A + D + \frac{1}{24}H + \frac{1}{1441}M + \frac{1}{86400}S \tag{3.4}$$

where

Y = Year

A = correction factor. It is equal to -1 if it is a leap year, otherwise it is equal to zero.

D = Days

H = Hours

M = Minutes

S = Seconds

and the reduced expression may result in

$$RJD = JD - 2451545,0$$

3.3 Unix Time

Unix time (also known as ***Epoch time***, ***POSIX time***,***seconds since the Epoch***, or ***UNIX Epoch time***) is a system for describing a point in time. It is the number of seconds that have elapsed since the Unix epoch, that is the time $00:00:00$ UTC on 1 January 1970, minus leap seconds. Leap seconds are ignored, with a leap second having the same Unix time as the second before it, and every day is treated as if it contains exactly 86400 seconds. Due to this treatment, Unix time is not a true representation of UTC.

It is a suitable format to work in the software and hardware spaces since it allows the express a vector of time as a point. Hardly often is used for developing simulation software and that's why it will be included in this project.

In order to understand it, there is a brief explanation below:

3.3.1 Definition

Two layers of encoding make up Unix time. The first layer encodes a point in time as a scalar real number which represents the number of seconds that have passed since 00:00:00 UTC Thursday, 1 January 1970. The second layer encodes that number as a sequence of bits or decimal digits.

3.3.2 Encoding Time as a number

Unix time is a single signed number which increments every second, without requiring the calculations to determine year, month, day of month, hour and minute required for intelligibility to humans.

The Unix time number is zero at the Unix epoch (00:00:00 UTC on 1 January 1970), and increases by exactly 86400 per day since the epoch. Thus, the equation to compute the unix number may be:

$$Unix_{Num} = n_{dae} \times 86400 \tag{3.5}$$

where n_{dae} refers to the number of days elapsed between the date computed and the epoch. An example would be : 2004-09-16 T 00:00:00Z, 12677 days after the epoch, is represented by the Unix time number $12677 \times 86400 = 1095292800$.

This can be extended backwards from the epoch too, using negative numbers; thus 1957-10-04 T 00:00:00Z, 4472 days before the epoch, is represented by the Unix time number $4472 \times 86400 = 386380800$. This applies within days as well; the time number at any given time of a day is the number of seconds that has passed since the midnight starting that day added to the time number of that midnight.

3.3.3 Leap Seconds

When a leap second occurs, the UTC day is not exactly 86400 seconds long and the Unix time number (which always increases by exactly 86400 each day) experiences a discontinuity. At the end of a day with a negative leap second, which has not yet occurred, the Unix time number would jump up by 1 to the start of the next day. During a positive leap second at the end of a day, which occurs about every year and a half on average, the Unix time number increases continuously into the next day during the leap second and then at the end of the leap second jumps back by 1 (returning to the start of the next day). For example, this is what happened on strictly conforming POSIX.1 systems at the end of 1998:

TAI (1 January 1999)	*UTC (31 December 1998 to 1 January 1999)*	*Unix time*
1999-01-01T00:00:29.75	1998-12-31T23:59:58.75	915 148 798.75
1999-01-01T00:00:30.00	1998-12-31T23:59:59.00	915 148 799.00
1999-01-01T00:00:30.25	1998-12-31T23:59:59.25	915 148 799.25
1999-01-01T00:00:30.50	1998-12-31T23:59:59.50	915 148 799.50
1999-01-01T00:00:30.75	**1998-12-31T23:59:59.75**	**915 148 799.75**
1999-01-01T00:00:31.00	**1998-12-31T23:59:60.00**	**915 148 800.00**
1999-01-01T00:00:31.25	1998-12-31T23:59:60.25	915 148 800.25
1999-01-01T00:00:31.50	1998-12-31T23:59:60.50	915 148 800.50
1999-01-01T00:00:31.75	**1998-12-31T23:59:60.75**	**915 148 800.75**
1999-01-01T00:00:32.00	**1999-01-01T00:00:00.00**	**915 148 800.00**
1999-01-01T00:00:32.25	1999-01-01T00:00:00.25	915 148 800.25
1999-01-01T00:00:32.50	1999-01-01T00:00:00.50	915 148 800.50
1999-01-01T00:00:32.75	1999-01-01T00:00:00.75	915 148 800.75
1999-01-01T00:00:33.00	1999-01-01T00:00:01.00	915 148 801.00
1999-01-01T00:00:33.25	1999-01-01T00:00:01.25	915 148 801.25

Figure 3.1: Unix time across midnight into 1 January 1999 (Positive Leap Second) [1]

Chapter 4

Satellite's Motion Architecture

The following section will explain how to get the satellite's coordinates in order to be able to positioning the object in the space and time. Also, it is a main goal of this project learn how to draw the Spacecraft's Ground Track. Thus, it will be explain the necessary factors needed to achieve it.

4.1 Reference frames

The position of an object in the space can be represented in many coordinates' systems. It is highly important to understand each one of them because of the difference in the center location. This section describes the definitions of the different reference frames used in this study. It is necessary to have them as different measurements and modelling are done in different frames and the rotation from one to another must be unique and well defined. [21]

- **Earth-Centered Inertial (ECI) Reference Frame:** For Newtons laws to be valid, one must have a non accelerating frame. The ECI is such an non accelerating inertial frame. The frame is located in the center of the earth and fixed towards the stars. This reference frame will be denoted I, and the earth rotates around its z-axis. The x-axis points towards the vernal equinox, and the y-axis completes a right hand Cartesian coordinate system.

- **Earth-Centered Earth Fixed (ECEF) Reference Frame:** The frame shares it's origin and z-axis with the ECI frame and is denoted E. The x-axis intersects the earths surface at latitude 0 and longitude 0°. The y-axis completes the right hand system. The ECEF rotates with the earth with a constant angular velocity ω_E , and is therefore not an inertial reference frame, and hence the laws of Newton is not valid.

- **Orbit Reference Frame:** The orbit frame has its origin at the point which the spacecraft has its center. The origin rotate at an angular velocity ω_0 relative to the ECI frame and has its z-axis pointed towards the center of the earth. The x-axis points in the spacecraft's direction of motion tangentially to the orbit. It is important to note that the tangent is only perpendicular to the radius vector in the case of a circular orbit. In elliptic orbits, the x-axis does not align with the satellite's velocity vector. The y-axis completes the right hand system as usual. The satellite attitude is described by roll, pitch and yaw which is the rotation around the x-, y-, and z-axis respectively. The orbit reference frame is denoted O.

- **Body Reference Frame:** The body frame shares it's origin with the orbit frame and is denoted B. The rotation between the orbit frame and the body frame is used to represent the spacecraft's attitude. It's axes are locally defined in the spacecraft, with the origin in the center of gravity or the center of the volume. The nadir side of the spacecraft, intended to point towards the earth, is in the z-axis direction and similar with the other two sides coinciding with the orbit frame.

- **Ecliptic Coordinate System:** The ecliptic coordinate system is a celestial coordinate system commonly used for representing the apparent positions and orbits of Solar System objects. Because most planets (except Mercury) and many small Solar System bodies have orbits with only slight inclinations to the ecliptic, using it as the fundamental plane is convenient. The system's origin can be the center of either the Sun (which would be classified as Heliocentric Coordinate System) or Earth, its primary direction is towards the vernal (March) equinox, and it has a right-hand convention. It may be implemented in spherical or rectangular coordinates.

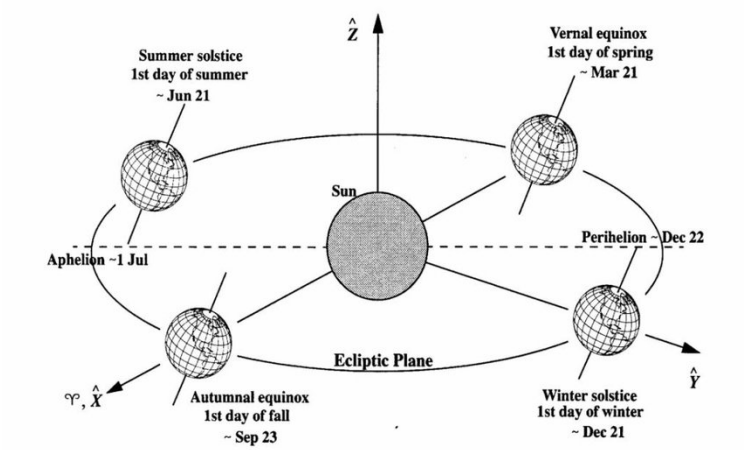


Figure 4.1: Heliocentric Coordinate System [23]

4.2 Satellite's Location

In terms of the Keplerian elements, the spacecraft position may be easy to get. The six parameters are convenient for defining an orbit in space in the inertial system defined by its three axes X, Y and Z. However, it can also be useful to express the location of a moving body in other parameters such as Cartesian or polar. [22]

(a) Satellite Orbit Reference Frame

(b) Satellite Orbit Reference Frame in ECI

Figure 4.2: Reference Frames [16]

Many times of the three-dimensional transformation can be performed as necessary to achieve a desired overall transformation in space around different axes. The following transformation will bring the position from the orbital plane to the inertial frame established in the equatorial plane of the central body. The orbit plane system is transformed by the three Euler angles (ω, i and Ω). Note that $\vec{P}$ is a unit vector directed from the center of the orbit to the perigee, $\vec{W}$ is a unit vector along the momentum axis of the orbit (normal to orbit plan) $h = r \times v$ and $\vec{Q}$, which completes the right orthogonal system, is advanced to $\vec{P}$ by a right angle in the plane and direction of motion.

$$\begin{pmatrix} \vec{P} \\ \vec{Q} \\ \vec{W} \end{pmatrix} = [A_z(\omega)][A_x(i)][A_z(\Omega)] \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \tag{4.1}$$

Where:

$[A_z(\omega)]$ Transformation matrix about z axis by "ω"
$[A_x(i)]$ Transformation matrix about z axis by "i"
$[A_z(\Omega)]$ Transformation matrix about z axis by "Ω"

Down below are presented resultant standard orientation vectors:

$$P_x = \cos\omega_i \cos\Omega_i - \sin\omega_i \sin\Omega_i \cos i_i$$

$$P_y = \cos\omega_i \sin\Omega_i + \sin\omega_i \cos\Omega_i \cos i_i$$

$$P_z = \sin\omega_i \sin\Omega_i$$

$$Q_x = \cos\omega_i \cos\Omega_i - \sin\omega_i \sin\Omega_i \cos i_i$$

$$Q_y = \cos\omega_i \sin\Omega_i + \sin\omega_i \cos\Omega_i \cos i_i$$

$$Q_z = \sin\omega_i \sin\Omega_i$$

$$W_x = \sin i_i \sin\Omega_i$$

$$W_y = \sin i_i \cos\Omega_i$$

$$W_z = \cos i_i$$

Unlike $\vec{P}$, $\vec{Q}$ and $\vec{W}$ tranformation, another effective transformation based on the same three angles, $(\omega + \theta)$, i and Ω by adding the true anomaly(ν) which will allow us position the satellite in its local orbit coordinate system. Adding the true anomaly means following the satellite during its motion. Now, the vectors are expressed as $\vec{R}$, unit vector along the radius vector r, $\vec{W}$, does not change from PQW coordinates (normal to the orbit plan) and $\vec{S}$, which completes the right orthogonal system. The transformation from ECI system to local coordinate orbital system can be expressed as follows:

$$\begin{pmatrix} \vec{R} \\ \vec{S} \\ \vec{W} \end{pmatrix} = [A_z(\omega + \nu)][A_x(i)][A_z(\Omega)] \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \tag{4.2}$$

Where:

$[A_z(\omega + \nu)]$ Transformation matrix about z axis by "$\omega + \nu$"
$[A_x(i)]$ Transformation matrix about z axis by "i"
$[A_z(\Omega)]$ Transformation matrix about z axis by "Ω"

Down below are presented the resultant local orbit vectors:

$$R_x = \cos(\omega_i + \nu_i)\cos\Omega_i - \sin(\omega_i + \nu_i)\sin\Omega_i \cos i_i$$

$$R_y = \cos(\omega_i + \nu_i)\sin\Omega_i + \sin(\omega_i + \nu_i)\cos\Omega_i \cos i_i$$

$$R_z = \sin(\omega_i + \nu_i)\sin\Omega_i$$

$$S_x = \cos(\omega_i + \nu_i)\cos\Omega_i - \sin(\omega_i + \nu_i)\sin\Omega_i \cos i_i$$

$$S_y = \cos(\omega_i + \nu_i)\sin\Omega_i + \sin(\omega_i + \nu_i)\cos\Omega_i \cos i_i$$

$$S_z = \sin(\omega_i + \nu_i)\sin\Omega_i$$

$$W_x = \sin i_i \sin\Omega_i$$

$$W_y = \sin i_i \cos\Omega_i$$

$$W_z = \cos i_i$$

Until now, the position vector of the satellite can be estimated. However, if we want to follow the satellite, we need some method to specify where a satellite is in its orbit. The *true anomaly* ν will allow us to follow the satellite in its orbit. It depends on the *mean anomaly, M*, 360·(Δt/P) deg, where P is the orbital period and Δt is the time since perigee passage of the satellite; thus M = ν for a satellite in circular orbit. The mean anomaly at any time is a trivial calculation and of no phisical interest. The quantity of real interest is the true anomaly, which is difficult to calculate. The *eccentric anomaly, E,* is introduced as an intermediate variable relating to the other two. E is the angle measured at the center of the orbit between the perigee and the projection (as shown in 4.3 of the satellite onto a circular orbit with the same semimajor axis.

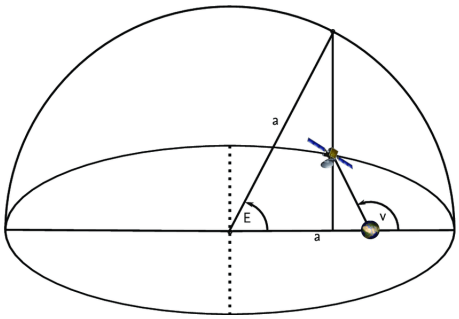


Figure 4.3: Illustration of the eccentric anomaly. [19]

The mean and eccentric anomalies are related by *Kepler's equation* (not related to Kepler's Laws) [20]:

$$M(t) = E(t) - e \sin E(t) \tag{4.3}$$

where e is the eccentricity. E is then related to ν by *Gauss' Equation:*

$$\tan \frac{\nu}{2} = (\frac{1+e}{1-e})^{1/2} \tan \frac{E}{2} \tag{4.4}$$

It may be easy conclude that it will be necessary to solve by iteration methods. The mean anomaly marches uniformly in time, and the future prediction is therefore

$$M(t_0 + t) = M(t_0) + n \cdot t \tag{4.5}$$

and the *Kepler's Equation* may be better expressed as:

$$E(t) = M(t) + e \sin E(t) \tag{4.6}$$

This equation has a unique solution, but is a transcendental equation, and so cannot be inserted and solved directly for E given an arbitrary M. A computational sequence to solve *Kepler's Equation* is explained below.

The iterative method may start analyzing the eccentricity (e) of the orbit:

- If $e_i > 1$ then $n_i = k\sqrt{\frac{\mu}{-a_i^3}}$ and $q_i = a_i(1 - e_i)$
- If $e_i = 1$ then $n_i = k\sqrt{\frac{\mu}{q_i^3}}$
- If $e_i < 1$ then $n_i = k\sqrt{\frac{\mu}{a_i^3}}$ and $q_i = a_i(1 - e_i)$

The next step would be compute the mean anomaly for the i orbit as $M_i = n_i(t - T_i)$, where t is the instant of time and T, the perifocal passage.

Since the solution depends on the eccentricity, there are three different steps to proceed:

- If e_i >1, to solve E_i from Kepler equation of hyperbolic orbit we may use Newton method, so ν_i is compute. Note that E is denoted as F to refer the eccentric anomaly in hyperbolic orbits.
  1. Let $(F_i)_0 = 6M_i$,

2. $(F_i)_{n+1} = (F_i)_n + \frac{M_i - e_i \sinh(F_i)_n + (F_i)_n}{e_i \cosh(F_i)_n - 1}$,
3. If $|(F_i)_{n+1} - (F_i)_n| > 0.000001$, repeat again 2 with $F_i = (F_i)_{n+1}$
4. $\nu_i = \tan^{-1} \frac{-\sinh F_i \sqrt{e_i^2 - 1}}{\cosh F_i - e_i}$ and end.

- If e_i =1, we can solve ν_i using Barkar's equation as follows:
  1. Let $A_i = \frac{3}{2} M_i$,
  2. $B_i = (\sqrt{A_i^2 + 1} + A_i)^{1/3}$,
  3. $C_i = B_i - \frac{1}{B_i}$,
  4. $\nu_i = 2 \tan^{-1} C_i$
- If e_i <1, to solve E_i from Kepler equation of hyperbolic orbit we may use Newton method, so ν_i is compute.
  1. Let $(E_i)_0 = M_i$,
  2. $(E_i)_{n+1} = (E_i)_n + \frac{M_i + e_i \sin(E_i)_n - (E_i)_n}{1 - e_i \cos(E_i)_n}$,
  3. If $|(E_i)_{n+1} - (E_i)_n| > 0.000001$, repeat again 2 with $E_i = (E_i)_{n+1}$
  4. $\nu_i = \tan^{-1} \frac{-\sin E_i \sqrt{1 - e_i^2}}{\cos E_i - e_i}$ and end.

With all these variables calculated we can finally estimate the distance at any time between the main force center and the satellite center as:

$$r_i = \frac{(1 + e_i) q_i}{1 + e_i \cos \nu_i} \tag{4.7}$$

From equation 4.1, we can compute the standard orientation vectors $\vec{P}$ and $\vec{Q}$. Then the position vector results in:

$$\vec{r_i} = \xi_i \vec{P_i} + \eta \vec{Q_i} \tag{4.8}$$

where ξ_i and η_i are the projections of the position radius over the perigee line and its perpendicular (node line if it has inclination). Using the true anomaly it is possible to do that:

$$\xi_i = r_i \cos \nu_i \tag{4.9}$$

$$\eta_i = r_i \sin \nu_i \tag{4.10}$$

4.3 Ground Track

The ground track of a satellite refers to the imaginary line that the spacecraft draws over the Earth's surface. As the Earth is considered a perfect sphere, the spherical trigonometry has to be used in order to get it. This line is well defined by the geodetic coordinates of the satellite, it means, the longitude and the latitude. In terms of the position vector expressed in ECI coordinates, they can be obtained. [18]

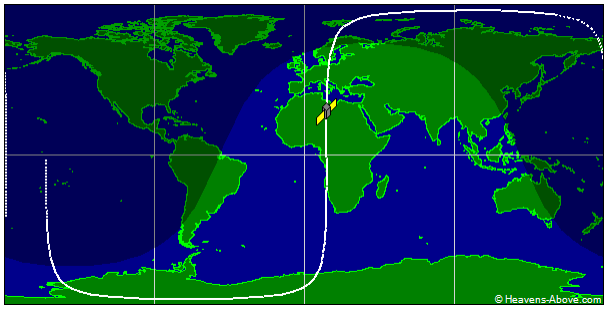


Figure 4.4: Iridium 45 ground track over the Earth surface. [3]

Thanks to equation 4.8, the x, y and z are expressed. After that, let denote ϕ to the latitude and λ to the longitude. The figure 4.5 shows better the next algebra.

Given the position vector $\vec{r_i}$, we can divide:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} \vec{r_1} \\ \vec{r_2} \\ \vec{r_3} \end{pmatrix} \tag{4.11}$$

The longitude, in the satellite orbit reference system, can be easily determined as

$$\lambda_o = 90^\circ - \tan^{-1} \frac{Y}{X} \tag{4.12}$$

As we want to express the results in the ECI system, the longitude may be influenced for the Earth's rotation. Thus, the longitude (λ) expressed in Earth Centered Inertial reference frame is:

$$\lambda(t) = \lambda_o - \omega_E t \tag{4.13}$$

where ω_E is the Earth's rotation $\frac{2\pi}{24 \cdot 60 \cdot 60}$ rad/seg.

To compute the latitude it takes a little bit more of time. In the same way, the geodetic coordinate could be expressed as

$$\phi = 90^\circ - \tan^{-1} \frac{Z \mathrm{d}Z}{r_{ho}} \tag{4.14}$$

where r_{ho} refers to the projection of the radius vector over the equatorial plane that can be compute using Pitagoras' rule,

$$r_{ho} = \sqrt{X \cdot X + Y \cdot Y}$$

The variation of the coordinate Z is computed as follows:

$$Z \mathrm{d}Z = Z + \mathrm{d}Z;$$
$$N_h = \sqrt{r_{ho}^2 + Z \mathrm{d}Z \cdot Z \mathrm{d}Z};$$
$$\sin \lambda = \tfrac{Z \mathrm{d}Z}{N_h};$$

where N_h refers to the distance between the main body center and the center of the satellite and $\mathrm{d}Z$ is the variation in height of the satellite measured from the equatorial plane. All these variables are better understood in the following figure:

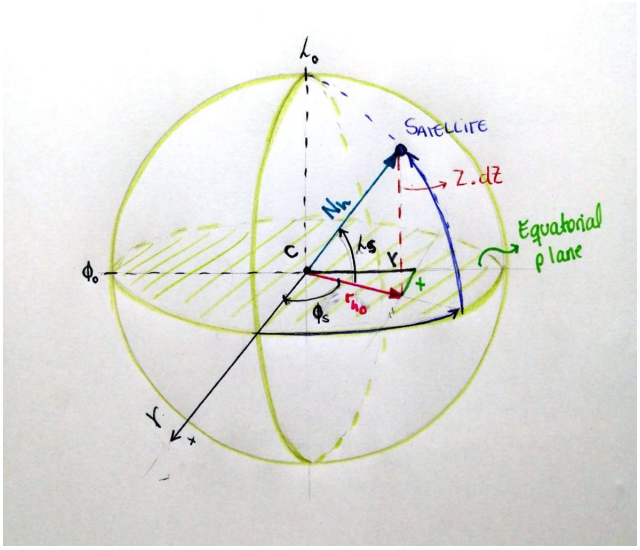


Figure 4.5: Meaning of the equations used to compute the longitude and latitude.

Finally, having considered the Earth as a perfect sphere, the satellite's height results in $h = R_E + N_h$.

Chapter 5

Solar Array Architecture

The other main objective of this study is to develop an algorithm which may predict the Sun incidence over the *Cubesats'* faces. The solar array illumination will be defined in the incoming section.

5.1 The Sun Vector

The first step will be positioning the Sun in the space. To compute the Sun's location with respect to the satellite for any given location at a given time it may be calculate the Sun's position in the ecliptic coordinate system and then convert the result to the ECI system. [13]

For the first step, it is necessary to compute the geodetic coordinates from the Sun, which depend on the time it is being evaluated. The number of days, j, is computed from the epoch referred to as the Julian date, 2.451.545,0:

$$j = JD - 2451545,0 \tag{5.1}$$

where JD is the Julian Date of the date of interest. Then, the mean longitude of the Sun (L), mean anomaly of the Sun (g), and the ecliptic longitude of the Sun (λ_s) are computed:

$$L = 280,460^\circ + 0,9856474^\circ j \tag{5.2}$$

$$g = 357,528^\circ + 0,9856474^\circ j \tag{5.3}$$

$$\lambda = L + 1,915^\circ + 0,020^\circ \sin 2g \tag{5.4}$$

where all the values are in the range of 0 to 360°. The distance to the Sun from the Earth in the unit meter can be approximated as follows:

$$u_{sun} = (1,00014 - 0,01671 \cos g - 0,00014 \cos 2g) \cdot 14600000 \tag{5.5}$$

Other than λ_s and u_{sun}, another parameter to form a complete position of the Sun in the ECI frame is the ecliptic latitude, β. The Sun's ecliptic latitude can be approximated by $\beta = 0$.

Next, the unit sun vector in the equatorial coordinate system is computed as follows:

$$\vec{u_{sun}} = \begin{bmatrix} \cos \lambda_s \\ \cos \epsilon \sin \lambda_s \\ \sin \epsilon \sin \lambda_s \end{bmatrix} \tag{5.6}$$

where ϵ is the obliquity of the ecliptic and it is approximated by:

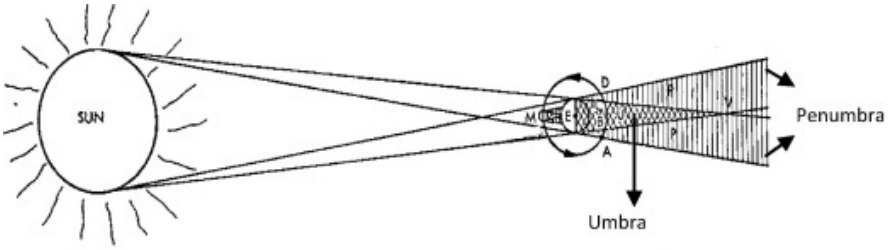
$$\epsilon = 23,439^\circ - 4,00 \cdot 10^{-7} j \tag{5.7}$$

5.2 The Earth's Shadow

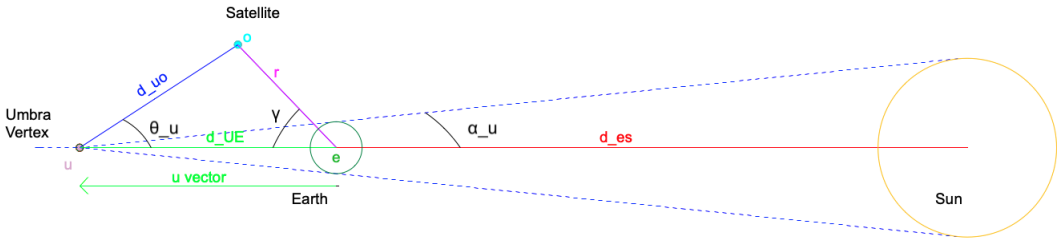
Once the Sun' vector is obtained, it can be studied if the satellite is eclipsed by the Earth's. The Earth and Sun are nearly spherical bodies and the Sun is about 100 times the Earth's size. Therefore, the shadow of the Earth is a cone with the centers of the Sun and Earth. As the vertex is going away from the Sun, the shadow is divided into two main parts as follows in figure 5.1(a) [11].

- The umbra of the shadow which extends about 138,900 km
- The penumbra of the shadow in which the sunlight is partially excluded.

In this project, only the umbra case will be studied. When the satellite's position at the specified time is known, it only takes a bit of trigonometry to know if the satellite lies in the Earth's shadow umbra.



(a) The shadow of the Earth and its two main parts. [11]



(b) The geometry of the eclipse conditions.

Figure 5.1: Eclipse Condition Geometry

The figure 5.1(b) shows that the satellite will be completely eclipsed by the Earth under the next conditions, both at the same time:

1. The distance satellite to Earth (r) is minor to the distance umbra point (d_{UE}). $\longrightarrow r < d_{UE}$
2. The angle associated as θ_u is minor to α_u (see the picture above). $\longrightarrow \theta_u < \alpha_u$

Hence, knowing the position of the sun and of the satellite at any time, the next equations may be used in order to compute the unknowns d_{UE}, θ_u and α_u.

$$d_{UE} = \frac{d_{es} \cdot R_E}{R_S - R_E} \tag{5.8}$$

$$\vec{u} = -d_{UE}\vec{u_{sun}} \tag{5.9}$$

$$\sin \alpha_u = \frac{R_E}{d_{UE}} \tag{5.10}$$

$$\vec{u} \cdot \vec{r} = u \cdot r \cdot \cos\gamma \tag{5.11}$$

$$\sin\theta_u = \frac{r}{d_{UE}} \cdot \sin\gamma \tag{5.12}$$

5.3 The Sun Incidence over the satellite

Finally, the Sun incidence over the spacecraft can be estimated. The angle of incidence between the unit sun vector and the face of the *Cubesat* is expressed as a dot product: [22]

$$\vec{n} \cdot \vec{u_{sun}} = \cos\xi ||n|| \cdot ||u_{sun}|| \tag{5.13}$$

where $\vec{N}$ refers to the unit normal vector to the *Cubesats'* faces. Specifically, they are the position vectors of the satellite. If the sun vector is expressed in ECI coordinate system, they must be $\vec{R}$,$\vec{S}$ and $\vec{W}$ expressed in terms of the keplerian elements $(\omega + \nu)$, i and Ω, as its been presented in equation 4.2.

The sign (positive or negative) of the cosine will determine whether the Sun is facing the spacecraft surface or not. A positive result would mean that the angle between vectors is inside the interval $[0°, 90°]$. On the other hand, if the sign is negative, the resultant angle would be inside the interval $[90°, 180°]$. Let explain the meaning of each sign. If the Sun is facing the surface of the spacecraft, is a vector directed to the face. So, following the define of the scalar product, the angle has to be in the interval of $[90°, 180°]$ or, which is the same, give as a result a negative value of the cosine. In the same way, if the sun does not impacts the surface of the satellites, the direction of the vector is opposite to the face, resulting in an angle between vectors in the range of $[0°, 90°]$, or a positive cosine value. It is better understood in the figure below:

(a) Sun Incidence

(b) No Sun Incidence

Figure 5.2: Solar Array Architecture

Note that the angle given by the scalar product between the vectors (see figure 5.2, angle in blue colour) is not the incidence of the Sun over the satellite's face. Following the clockwise as a the positive values, the Sun incidence would be the difference between 90 ° and the angle given by the scalar product.

$$\xi = \arccos \frac{\vec{n} \cdot \vec{u_{sun}}}{||n|| \cdot ||u_{sun}||} \tag{5.14}$$

$$\alpha_{sun} = 90^{o} - \xi \tag{5.15}$$

Chapter 6

Orbit Simulator Code

The purpose of this project has been to implement an algorithm which could predict with accuracy the position of a satellite and estimate the solar illumination that impacts over its faces.

6.1 Inputs

As well as many codes start, there are a few constants that must be included in the firsts lines.

- Earth Radius $R_E = 6371000m$
- Sun Radius $R_S = 6957000m$
- Earth's Mass $M_E = 5.972 \cdot 10^{24}$
- Earth's Angular Velocity $\omega_E = \frac{2\pi}{86400} = 7,26 \cdot 10^{-5}$
- Earth's Orbit Eccentricity $e_E = 0.016751$
- Gravitational Constant $G = 6.673 \cdot 10^{-11}$
- Standard Gravitational Parameter $\mu = 3.986004418 \cdot 10^{14}$

6.1.1 Satellite's Information

The next step will be obtain the orbital elements of the orbit where is located the satellite. The data format which it will use to send to the ground station this information will be *TLE (Two Line Element)* . This kind of messages refer to, as the name indicates, two lines of text that include the orbital elements of an Earth-orbiting object for a given point in time, called the epoch. Besides, there is another first line(named the line 0) that gives the object designation. Despite not being compulsory, almost always is included.

In order to explain what means each number in each line of data it will be used the following example.

```
IRIDIUM 106
1 41917U 17003A   19362.73345509  .00000066  00000-0  16557-4 0  9990
2 41917  86.3963 355.7388 0002579  93.2452 266.9039 14.34219247154592
```

Figure 6.1: Example of a TLE message. [3]

Line 0 is a twenty-four character name (to be consistent with the name length in the NORAD SATCAT).

Lines 1 and 2 are the standard Two-Line Orbital Element Set Format identical to that used by NORAD and NASA. The format description is well described in the tables 6.1 and 6.2.

Column	Description	Example
01	Line Number	1
03-07	Satellite Number	41917
08	Classification	U
10-11	International Designator (Last two digits of the launch year)	17
12-14	International Designator (Launch number of the year)	003
15-17	International Designator (piece of launch)	A
19-20	Epcoch Year (last two digits of the year)	19
21-32	Epoch (day of the year and fractional portion of the day)	326.73345509
34-43	First Time Derivative of the Mean Motion	-.0000006
45.52	Second Time Derivative of the Mean Motion (decimal point assumed)	00000-0
54-61	BSTAR drag term (decimal point assumed)	16557-4
63	Ephemeris type	0
65-68	Element number	999
69	Checksum (moludo 10)	0

Table 6.1: Parameters of the first line

Column	Description	Example
01	Line Number of Element Data	2
03-07	Satellite Number	4197
09-16	Inclination [Degrees]	86.3963
18-25	Right Ascension of the Ascending Node [Degrees]	355.7388
27-33	Eccentricity (decimal point assumed)	-.0002579
35-42	Argument of Perigee [Degrees]	93.2452
44-51	Mean Anomaly [Degrees]	266.9039
53-63	Mean Motion [Degrees]	14.34219247
64-68	Revolution number at epoch [Revs]	15459
69	Checksum (Module 10)	2

Table 6.2: Parameters of the second line

When the satellite will have sent its orbital data, the code must read it and store it. It has been implemented a function called *lecture* that reads a txt file which contains the TLE data of the satellites constellation and classify each column according to its description. The number of satellites read is obtained too. All this information is saved in a variable *struct trype*. In addition, this function computes the following initial conditions:

- UTC date and Greenwich Sidereal time at the epoch date
- Perifocal Passage (T) [seg] and Perigee distance (q) [m]
- Orbital Period (P) [seg] and Angular velocity (w_s) [rad/seg]
- Longitude of the Ascending node ($LAAN$) [rad]
- Initial position X_o[m] and velocity vectors V_o[m/s]
- Initial longitude ϕ° and latitude λ°

An example of the output information that provides the function *lecture* would be, in the case for two satellites:

```
1
2  ID: {'IRIDIUM'   'IRIDIUM'}
3     designation: {'109'   '102'}
4             PRN: {''   ''}
5               i: [1.5079  1.5079]
6            RAAN: [6.1989  6.1998]
7           omega: [1.6192  1.5423]
```

```
                M: [4.6665  4.7433]
                n: [0.0010  0.0010]
                a: [7.1428e+06  7.1428e+06]
                e: [2.1580e-04  1.9120e-04]
             date: {'29-Dec-2019 21:54:57'  '29-Dec-2019 17:48:21'}
            Bstar: [1.6510e-05  1.9976e-05]
               BC: [4.7537e+03  3.9289e+03]
            epoch: [1.5777e+09  1.5776e+09]
                T: [1.5777e+09  1.5776e+09]
                q: [7.1412e+06  7.1414e+06]
                P: [6.0242e+03  6.0242e+03]
               ws: [0.0010  0.0010]
         deltato: [3.4530e+05  3.4530e+05]
             LAN: [5.0503  6.1300]
              Xo: [3*1 double]
              Vo: [3*1 double]
          RSaveo: [1*3*2 double]
          VSaveo: [1*3*2 double]
            lono: [-70.6315  -8.7666]
            lato: [0.1200  0.1201]
              ho: [7.6470e+05  7.6458e+05]
```

6.2 Algorithm core

The core of the algorithm remains on the estimation of the satellite's groundtrack and the sun incidence over its faces. The next section explains how the theory explained previously has been adapted to a MATLAB language.

6.2.1 Position Module

Once got all the information of the spacecraft required, the location in time can be estimated. Step by step the code implements the equations described in the section 4.2.

```
 %For the i satellite

      a(i) = (mu/n(i)^2)^(1/3);
      q(i) = OrbitData.a(i)*(1-OrbitData.e(i));

%STEP 1: Finding unperturbated mean motion

      if e(i)>= 0
          n(i) = n(i);
      else
         error('Excentricity cannot be a negative value');
      end

%STEP 2: Solving Mean Anomaly
     M(i) = n(i)*(t-T(i)); % Mean anomaly in rad
      M_deg(i) = M(i)*180/pi; %Mean anomaly in deg

%STEP 3: Finding true anomaly
       if e(i)>1
          Fn(i) = 6*M(i)
          error = 1;
       while error > 1e-8
         Fn1(i)= Fn(i)+(M(i)-e(i)*sinh(Fn(i))+Fn(i))/(e(i)*cosh(Fn(i))-1);
```

```
24        error = abs(Fn1(i)-Fn(i));
25           Fn(i) = Fn1(i);
26        end
27
28         f(i) = atan((-sinh(Fn(i))*sqrt(e(i)^2-1))/(cosh(Fn(i))-e(i)))
29
30         elseif e(i) == 1;
31
32         A(i) = (3/2)*M(i);
33         B(i) = (sqrt(A(i)^2+1)+A(i))^(1/3);
34         C(i) = B(i)-1/B(i);
35         f(i) = 2*atan(C(i))
36
37        elseif e(i) <1 && e(i)>=0
38
39        %Convert mean anomaly to true anomaly.
40        %First compute the eccentricity anomaly using Kepler's equation
41
42        if -pi<M(i)<0 || M(i)>pi
43                    Ea = M(i)-e(i);
44            else
45                    Ea = M(i)+e(i);
46            end
47
48        % Define tolerance.
49            tol  = 1e-12;
50
51            test = 999; % Dummy variable.
52        % Implement Newton's method.
53
54        while test > tol
55             E_new = Ea + (M(i)-Ea+e(i)*sin(Ea))/(1-e(i)*cos(Ea));
56            test = abs(E_new - Ea);
57            Ea = E_new;
58          end
59
60        y = sin(Ea)*sqrt(1-e(i)^2)/(1-e(i)*cos(Ea));
61        z = (cos(Ea)-e(i))/(1-e(i)*cos(Ea));
62
63        %Compute true anomaly f
64         f(i) = atan2(y,z) ;
65
66        else
67           error('Excentricity cannot be a negative value');
68        end
69
70  %STEP 4: Finding primary body center to satellite distance
71
72        r(i) = q(i)*(1+e(i))/(1+e(i)*cos(f(i)));
73
74  %STEP 5.1: Finding standard orientation vectors (Orbit Coordinate system) [P,Q,W]
75
76        Px(i) = cos(omega(i))*cos(RAAN(i))-sin(omega(i))*sin(RAAN(i))*cos(i(i));
77        Py(i) = -cos(omega(i))*sin(RAAN(i))-sin(omega(i))*cos(RAAN(i))*cos(i(i));
78        Pz(i) = sin(omega(i))*sin(i(i));
79
```

```
80        P_vector(:,i) = [Px(i) Py(i) Pz(i)];
81
82        Qx(i) = sin(omega(i))*cos(RAAN(i))+cos(omega(i))*sin(RAAN(i))*cos(i(i));
83        Qy(i) = -sin(omega(i))*sin(RAAN(i))+cos(omega(i))*cos(RAAN(i))*cos(i(i));
84        Qz(i) = -cos(omega(i))*sin(i(i));
85
86        Q_vector(:,i) = [Qx(i) Qy(i) Qz(i)];
87
88    %STEP 5.2: Finding standard orientation vectors (Orbit ECI Coordinates system)[R,S,
        W]
89
90        Rx(i) = cos(omega(i)+f(i))*cos(RAAN(i))-sin(omega(i)+f(i))*sin(RAAN(i))*cos(i(
            i));
91        Ry(i) = -cos(omega(i)+f(i))*sin(RAAN(i))-sin(omega(i)+f(i))*cos(RAAN(i))*cos(i
            (i));
92        Rz(i) = sin(omega(i)+f(i))*sin(i(i));
93
94        R_vector(:,i) = [Rx(i) Ry(i) Rz(i)];
95
96        Sx(i) = sin(omega(i)+f(i))*cos(RAAN(i))+cos(omega(i)+f(i))*sin(RAAN(i))*cos(i(
            i));
97        Sy(i) = -sin(omega(i)+f(i))*sin(RAAN(i))+cos(omega(i)+f(i))*cos(RAAN(i))*cos(i
            (i));
98        Sz(i) = -cos(omega(i)+f(i))*sin(i(i));
99
100       S_vector(:,i) = [Sx(i) Sy(i) Sz(i)];
101
102   %Vector W is normal to the orbit plane (same for both reference frames)
103
104       Wx(i) = sin(i(i))*sin(RAAN(i));
105       Wy(i) = sin(i(i))*cos(RAAN(i));
106       Wz(i) = cos(i(i));
107
108       W_vector(:,i) = [Wx(i) Wy(i) Wz(i)];
109
110   %STEP 6: Finding components of the primary body center to satellite vector in the
        orbital plane
111
112       xi(i) = r(i)*cos(f(i));
113       eta(i) = r(i)*sin(f(i));
114
115   %STEP 7: Finding primary body center to satellite vector
116
117       r_fullvector = xi(i)*[Px(i) Py(i) Pz(i)] + eta(i)*[Qx(i) Qy(i) Qz(i)];
118
119       for j = 1:3
120           r_vector(i,j) = r_fullvector(j);
121       end
122
123   %STEP 8: Finding Parameter or Semiparameter
124
125      parameter(i) = a(i)*(1-e(i)^2);
126
127   %STEP 9: Adjust RAAN such that we are consisten with Earth's current orientation.
        This is a conversion to Longitude of the Ascending Node (LAN):
128
```

```
129         LAN(i) = OrbitData.RAAN(i) - GMST; % rad
130         LAN_deg(i) = Orbit.RAAN(i) - GMST*180/pi; %deg
131
132         [X,V] = COE2RV(a(i),e(i),i(i), LAN(i),omega(i), M(i)); %Convert position
                vector (RQW) to ECI coordinates
133
134         RSave(:,i) = X';
135         VSave(:,i) = V';
136
137        [lon(i), lat(i), h(i)] = Geodetic(RSave(:,i));
138
139        lon(step_count,i)=lon(step_count,i)*180/pi-Wt*tsince(step_count)*180/pi;
                    lat(i)=lat(step_count,i)*180/pi;
```

6.2.2 Sun Incidence Module

The second part to solve the problem is to analyze how the sun is facing the surface of the *Cubesat*. As the name suggests, the geometry is a six-sided cube. The coordinates' vectors are orthogonal to each side of the cube. Thus, for each vector may be two solutions:

1. Positive direction of the vector, which has been identified as the front side of the *Cubesat*.
2. Negative direction of the vector, which has been identified as the rear (back) side of the *Cubesat*.

In order to compute the Sun coordinates, each instant of time during the simulation must be converted from *Posixtime* to *Julian Dates*. A function that provides the library of *MATLAB* named *juliandate* has been used. It simply transforms a date given into the corresponding julian number by using the method explain previously in the section 3.2. Once, the date has been obtained, the procedure written in section 5 is implemented.

```
1
2  %STEP 1: Convert the moment of time in question to julian date
3
4  jd= datetime(t,'ConvertFrom','posixtime'); %Express the moment of time as a time
       vector
5  d =  juliandate(jd) - 2451545.0;
6
7  %STEP 2: Estimation of the Sun' coordinates
8
9  L = 280.470 + 0.98565*d; %Mean Longitude of the Sun (degrees)
10 g = 357.528 + 0.9856474*d;  %Mean anomaly of the Sun (degrees)
11 lambda = L + 1.915*sind(g) + 0.020*sind(2*g); %Ecliptic Longitude of the Sun (
       degrees)
12 epsilon = 23.4393 - 0.0000004*d; %Obliquity of the ecliptic (degrees)
13
14 %STEP 3: Compute Sun vector in ECI Coordinate System
15 u_sun(:,i) = (1.00014-0.01671*cosd(g)-0.00014*cosd(2*g))*149600e3*1000; %Distance
       of the Sun from the EARTh in meter
16
17 u_sun_vector(:,i) =[cosd(lambda) ;cosd(epsilon)*sind(lambda); sind(epsilon)*sind(
       lambda)];
18 Solar(i) = norm(u_sun_vector(:,i)); %Verify if the Sun Vector is unitary
19
20 %STEP 4: Estimate the Eclipse Conditions
21
22 d_UE(:,i)=u_sun(:,i)*Rt/(Rs-Rt); %En meters
23
24   alpha_umbra(:,i)=asind(Rt/d_UE(:,i));
```

```
25
26      umbra_v(:,:,i)=-d_UE(step_count,i)*u_sun_vector(:,:,i);
27
28      umbra_vv(:,:,i)=(umbra_v(:,:,i))';
29
30      d_UO(:,i)= sqrt((umbra_vv(:,1,i)-RSave(:,1,i))^2+(umbra_vv(:,2,i)-RSave(:,2,i))
            ^2+(umbra_vv(:,3,i)-RSave(:,3,i))^2);
31
32      cosine(:,i)=dot(umbra_vv(:,:,i),RSave(:,:,i))/(d_UE(:,i)*r(i));
33
34      angle(:,i)=acosd(cosine(:,i));
35
36      theta_u(:,i)=asind((r(i)/d_UE(step_count,i))*sind(angulo(step_count,i)));
37
38      if d_UO(:,i)<d_UE(:,i) && theta_u(:,i)<alpha_umbra(:,i) %Evaluate Eclipse
            Conditions
39
40          shadow(step_count,i)=1; %The satellite is eclipsed
41
42      else  shadow(step_count,i)=0; %The satellite is not eclipsed
43
44      end
45
46
47  %STEP 5: Evaluate the angle between RSW vectors and Sun vector
48
49  %Sun incidence for R
50
51          cos_v_RF(i) = dot(u_sun_vector(:,i),R_vector(:,i)); %Front Face
52
53          v_RF(i) = acosd(cos_v_RF(i));   %Angle between vectors
54
55          solarIncidence_RF(i) = 270 - acosd(cos_v_RF(i));
56
57          cos_v_RB(i) = dot(u_sun_vector(:,i),-R_vector(:,i)); %Rear Face
58
59          v_RB(i) = acosd(cos_v_RB(i));   %Angle between vectors
60
61          solarIncidence_RB(i) = 270 - acosd(cos_v_RB(i));
62
63
64  %Sun incidence for S
65
66          cos_v_SF(i) = dot(u_sun_vector(:,i),S_vector(:,i)); %Front Face
67
68          v_SF(i) = acosd(cos_v_SF(i));    %Angle between vectors
69
70          solarIncidence_SF(i) = 270- acosd(-cos_v_SF(i));
71
72          cos_v_SB(i) = dot(u_sun_vector(:,i),-S_vector(:,i)); %Rear face
73
74          v_SB(i) = acosd(cos_v_SB(i));    %Angle between vectors
75
76          solarIncidence_SB(i) = 270 - acosd(-cos_v_SB(i));
77
78  %Sun incidence for W
```

```
79
80      cos_v_WU(i) = dot(u_sun_vector(:,i),W_vector(:,i)); %Upper Face
81
82      solarIncidence_WU(i) = acosd(cos_v_WU(i));
83
84      cos_v_WD(i) = dot(u_sun_vector(:,i),-W_vector(:,i)); %Lower Face
85
86      solarIncidence_WD(i) = acosd(cos_v_WD(i));
```

6.2.3 Simulation Module

The code has to give a solution for a determined period of time. In MATLAB language, it has to be developed an iterative module able to evaluate each satellite in every instant of time. The time is a variable which is difficult to use due to the impossibility in using a moment of time as a matrix cell, where the information has to be stored. Therefore, the following discretization has been done:

```
1
2  for i=1:num_satellites
3
4          step_count = 1;
5
6     for t=t:increment:t_end %Simulation time and time discretization
7          % Time since the simulation started
8             tsince(step_count) = t-start_time_unix;
9   %CORE ALGORITHM
10
11   step_count = step_count + 1
12
13   end
14 end
```

Let explain the procedure followed. The code analyzes every individual satellite in the entire period of time which starts since a time t previously defined, until a *t_end*, also given. The variable t increases constantly in time where every increase will be taken as an *step_count*. The results will provide matrixes with a number of columns equal to the number of satellites analyzed and a number of rows equal to the steps done until the simulation finishes.

Regarding to the start and end times, the code will ask the user to choose one of the following start options:

- The user defines the date time
- Use local time from the computer (now)
- Start at the epoch date provided by the *TLE message*

After reading the user entry, again will ask how long does the user wants to last the simulation. It must be entered the number of hours that will last from the start time defined. Hence, the end time will be the start time plus the input hours.

When the *lecture* function reads the data, it saves the date as a vector. A suitable way to operate with dates in this kind of problems is computing the transformation of the date to Unix Time as explained in section 3.3. MATLAB provides the *posixtime* function that transforms a given time vector into the corresponding unix time number.

```
1 %% Simulation Time Parameters
2
3 start_time = datetime('now');
4 start_time_unix= posixtime(datetime(start_time)); %Convert start time to posixtime
```

```
5
6  t = start_time_unix; % time [s]
7
8  end_time = datetime('now') + hours(2);
9  end_time_unix= posixtime(datetime(end_time)); %Convert end time to posixtime
10
11 t_end = end_time_unix; % time [s]
12
13 time_divisions = 999;
14 increment = (end_time_unix-start_time_unix)/time_divisions; %Increment of time for
       the simulation
```

Chapter 7

Validation

The code developed gives the user the opportunity to trace a constellation of satellites in time and predict their motion in orbit. Besides, it will check whether the sun is facing the spacecraft or not in order to know if the satellite in question is able to communicate.

This chapter shows the results that can be extracted from the code and how they can be construed.

7.1 Tracking the satellite

In order to validate the software it has been chosen a constellation with a particular type of orbit. The *Intelsat* constellation, a group of satellites related to the telecommunication networks, is compound for many satellites orbiting in equatorial planes. It means that their ground track should be a straight line along the equator. Note that this constellation is placed in a Geostationary Orbit. This study pretends to optimize LEO satellites, not GEO. Although, it has been considered first show results with this type of satellites, due to the strict orbit that they have. After that, it will be presented a LEO case. Based on the data provided for *Celestrak* website we will procedure.

Inside the 69 satellites constellation, it has been analyzed the folowing group of 25:

International Designator	NORAD Catalog Number	Name	Period (minutes)	Inclination (degrees)	Apogee Height (km)	Perigee Height (km)	Eccentricity	Latest Data	TLE Age (days)
1997-007A □ □ □	24732	INTELSAT 26 (IS-26) □	1,436.07	8.81	35,798	35,774	0.0002912	□ □	1.25
1997-026A □ □ □	24812	GALAXY 25 (G-25) □	1,436.09	1.54	35,801	35,772	0.0003419	□ □	0.68
1997-046A □ □ □	24916	INTELSAT 5 (IS-5) □	1,436.10	5.38	35,799	35,775	0.0002828	□ □	0.71
1998-037A □ □ □	25371	INTELSAT 805 (IS-805) □	1,436.09	2.07	35,801	35,772	0.0003408	□ □	1.30
1999-071A □ □ □	26038	GALAXY 11 (G-11) □	1,436.10	0.83	35,789	35,784	0.0000542	□ □	0.77
2000-043A □ □ □	26451	INTELSAT 9 (IS-9) □	1,438.53	5.36	35,851	35,817	0.0004010	□ □	0.61
2000-068A □ □ □	26590	INTELSAT 12 (IS-12) □	1,436.08	2.55	35,813	35,759	0.0006399	□ □	0.58
2000-072A □ □ □	26608	INTELSAT 1R (IS-1R) □	1,436.10	2.33	35,788	35,785	0.0000316	□ □	0.29
2001-019A □ □ □	26766	INTELSAT 10 (IS-10) □	1,436.08	3.52	35,801	35,772	0.0003452	□ □	0.32
2001-024A □ □ □	26824	INTELSAT 901 (IS-901) □	1,442.86	1.52	35,922	35,915	0.0000828	□ □	0.62
2001-039A □ □ □	26900	INTELSAT 902 (IS-902) □	1,439.21	0.23	35,866	35,829	0.0004356	□ □	0.61
2001-052A □ □ □	26985	DIRECTV 4S □	1,450.73	0.31	36,085	36,060	0.0002954	□ □	0.60
2002-007A □ □ □	27380	INTELSAT 904 (IS-904) □	1,436.10	1.07	35,798	35,775	0.0002812	□ □	0.69
2002-016A □ □ □	27403	INTELSAT 903 (IS-903) □	1,436.09	1.90	35,789	35,784	0.0000685	□ □	0.31
2002-023A □ □ □	27426	DIRECTV 5 (TEMPO 1) □	1,436.12	0.02	35,802	35,772	0.0003584	□ □	0.25
2002-027A □ □ □	27438	INTELSAT 905 (IS-905) □	1,436.10	0.89	35,797	35,776	0.0002445	□ □	0.76
2002-030A □ □ □	27445	GALAXY 3C (G-3C) □	1,436.11	0.01	35,794	35,780	0.0001611	□ □	0.31
2002-041A □ □ □	27513	INTELSAT 906 (IS-906) □	1,436.09	0.01	35,795	35,777	0.0002124	□ □	0.25
2003-007A □ □ □	27683	INTELSAT 907 (IS-907) □	1,436.12	0.02	35,797	35,777	0.0002444	□ □	0.69
2003-013B □ □ □	27715	GALAXY 12 (G-12) □	1,436.09	1.11	35,796	35,777	0.0002214	□ □	0.71
2003-034A □ □ □	27854	GALAXY 23 (G-23) □	1,436.10	0.01	35,799	35,774	0.0002871	□ □	0.73
2003-044A □ □ □	27954	GALAXY 13 (HORIZONS 1) □	1,436.08	0.04	35,789	35,784	0.0000527	□ □	0.73
2004-016A □ □ □	28238	DIRECTV 7S □	1,436.10	0.01	35,800	35,774	0.0003072	□ □	0.71
2004-022A □ □ □	28358	INTELSAT 10-02 □	1,436.12	0.01	35,799	35,775	0.0002829	□ □	0.61
2005-008A □ □ □	28626	XM-3 (RHYTHM) □	1,436.12	0.04	35,787	35,787	0.0000095	□ □	0.63

Figure 7.1: Intelsat Satellites: current as of 2020 Jan 04 08:06:46 UTC (Day 004) [3]

The simulation parameters are:

- Start Time : 2020 Jan 04 11:13:23 UTC(+01)
- End Time : 2020 Jan 05 11:13:23 UTC(+01)
- Simulation last : 24h
- Number of Satellites analyzed : 25 (see 7.1)

For a 24h of simulation, the results are the next ones:

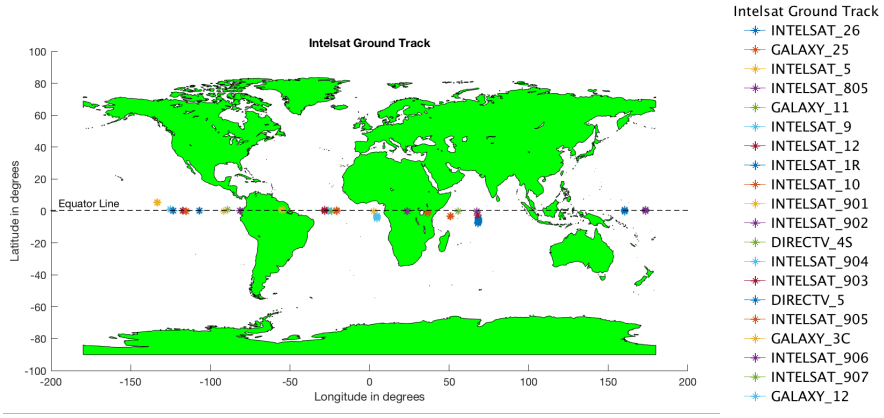
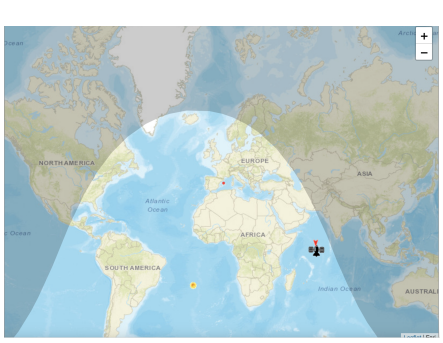


Figure 7.2: Intelsat Constellation Ground Tack for 24 h simulation.

The first impression may lead to a wrong result. It seems that the orbits of the spacecrafts are not well drawn. There is evidence whether the orbits are equatorial or near equatorial. Due to the small inclination of them, the satellites are drawn in a position close to the equator line. But why the picture only shows a point instead a straight line? The reason falls on the orbital period of the Geostationary Satellites. If they move at the same angular velocity as the Earth does, it is clear that they will be positioned over Earth always at the same place. Therefore, the figure 7.1 seems to be in concordance with the reality.

The next step will be to verify if the software estimates with accuracy the exact position of the satellite. This has been validated comparing the real time tracking which provides the *N2YO.com* website with the results that computes the code. For example, *Intelsat 906*, the comparisson can be made between the figures 7.3 and 7.4.



(a) Map Position

NORAD ID:	27513
LOCAL TIME:	14:13:27
UTC:	13:13:27
LATITUDE:	-0.01
LONGITUDE:	64.16
ALTITUDE [km]:	35784.40
ALTITUDE [mi]:	22235.4
SPEED [km/s]:	3.07
SPEED [mi/s]:	1.91
AZIMUTH:	109.0 ESE
ELEVATION:	+11.9
RIGHT ASCENSION:	00h 47m 52s
DECLINATION:	-07° 03' 56"
Local Sidereal Time:	20h 14m 50s
The satellite is in day light	
SATELLITE PERIOD:	1437m

(b) Satellite's current information

Figure 7.3: Live Real Time Satellite *Intelsat 906* Tracking. [7]

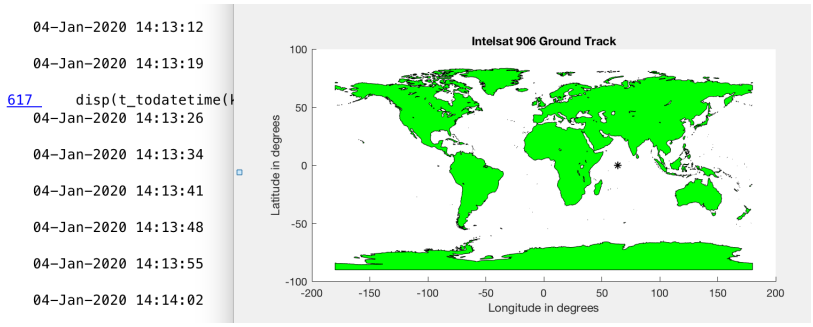


Figure 7.4: Matlab results of the Satellite *Intelsat 906* Tracking.

The figure 7.3 shows the live real time tracking of the satellite that the website provides. It sites the spacecraft in the Indian Ocean over Madagascar island. On the other hand, the Matlab results give the figure 7.4, where the satellites is represented as a black point. The plot places the point as the website does, or at least this is what it seems if one observe and compare both pictures. However, more accurate results will be exposed to verify it.

The algorithm stores the geodetic coordinates at each step count in a vector called *satcoord*. Another form to verify that the code estimates a good approximation of the position is to compare for at the same instant of time the live real time coordinates provided by the website and the ones stored in the *satcoord* vector. The figure 7.3(b) shows that the satellite at 14:13:27 h local time is at 64.16 ° longitude and -0.01 ° latitude. The *satcoord* vector stores at 14:13:26 h local time, just a second before, 63.7918 ° longitude and -0.0072 ° latitude, results practically equal to the real ones. If the relative error is computed, only gives a 0.5 % of error.

Now, a LEO constellation will be presented. In this case, *Geodetic* constellation, compound for scientific satellites, has been chosen. This are:

International Designator	NORAD Catalog Number	Name	Period (minutes)	Inclination (degrees)	Apogee Height (km)	Perigee Height (km)	Eccentricity	Latest Data	TLE Age (days)
1975-010A □ □	07646	STARLETTE □	104.17	49.83	1,107	805	0.0205789	□ □	0.57
1976-039A □ □	08820	LAGEOS 1 □	225.47	109.85	5,947	5,839	0.0044107	□ □	0.65
1986-061A □ □	16908	AJISAI (EGS) □	115.71	50.01	1,496	1,479	0.0011162	□ □	0.27
1992-070B □ □ □	22195	LAGEOS 2 □	222.46	52.67	5,952	5,616	0.0138197	□ □	0.65
1993-061B □ □ □	22824	STELLA □	100.88	98.95	805	796	0.0005709	□ □	0.16
2012-006A □ □ □	38077	LARES □	114.75	69.49	1,449	1,439	0.0006804	□ □	0.30

Figure 7.5: Geodetic Satellites: current as of 2020 Jan 05 08:06:43 UTC (Day 005) [3]

In particular, it will be shown the results for *Starlette*. Since it has a orbital period of 104,17 minutes, the simulation will last at least 2h to get a complete orbit drawn. The simulation parameters are:

- Start Time : 2020 Jan 05 09:58:49 UTC(+01)
- End Time : 2020 Jan 05 12:28:49 UTC(+01)
- Simulation last : 2,5 h
- Number of Satellites analyzed : 1 (*Starlette*)

(a) Satellite's Ground Track

STARLETTE

NORAD ID:	7646
LOCAL TIME:	10:00:04
UTC:	09:00:04
LATITUDE:	-35.35
LONGITUDE:	16.19
ALTITUDE [km]:	829.11
ALTITUDE [mi]:	515.19
SPEED [km/s]:	7.44
SPEED [mi/s]:	4.62
AZIMUTH:	167.9 S
ELEVATION:	-34.5
RIGHT ASCENSION:	19h 55m 52s
DECLINATION:	-79° 23' 53"
Local Sidereal Time:	16h 04m 44s
The satellite is in day light	
SATELLITE PERIOD:	105m

(b) Satellite's current information

Figure 7.6: Live Real Time *Starlette* Satellite Tracking.

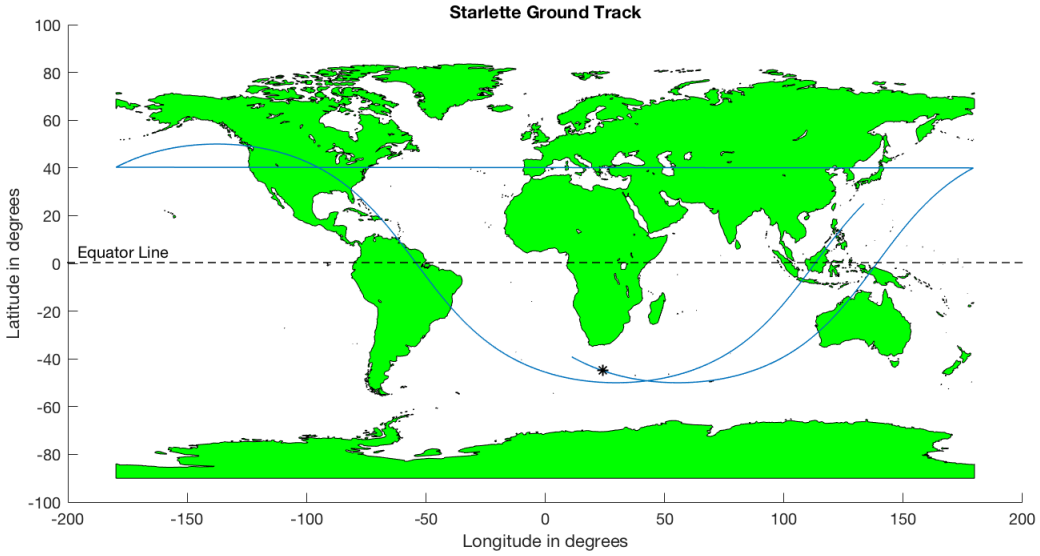


Figure 7.7: Matlab results of the *Starlette* Satellite Tracking. [7]

The figure 7.6 presents the real live tracking of the satellite provided for *N2YO.com* website and the 7.7 the ground track computed by the software. Since it is appreciable how similar are both representations, numerical results will be showed.

Looking to 7.6(b), the satellite is located at 16.19 ° longitude, -35.35 ° latitude and 829.11 km altitude at the local time 10:00:04 h. The vector *satcoord* at the local time 10:00:10 h gives 16.1811 ° longitude, -41.7253 ° latitude and 830 km altitude. The longitude and altitude values are strongly approximated despite the latitude value differs in 6 °. In this case, the software seems to have increased the error around an 18.5 %. The error obtained probably will be cause of not including the atmospheric drag when the coordinates of the satellites are computed. As long as the spacecraft decreases in height, the atmospheric drag influences more in the motion and perturbs the orbit. Nevertheless, if we compare the error for a GEO satellite at almost 35.000 km of altitude and the error for a LEO satellite at 830 km of altitude, it increases in about 15 % when the satellite descends 34.000 km, being the altitude range of LEO satellites 200 - 20.000 km.

7.2 Estimating the Solar Incidence

The next part of the thesis is to estimate whether the sun will influence in the ability of the satellites to communicate. The same constellations as in the tracking validation will be used. First, will be analyzed the *Intelsat* constellation focusing on *Intelsat 906* for its very low inclination. The below hypotesis have been assumed:

- For simplicity, the ***Cubesats 1U* geometry** has been assumed, besides that this project is going to be applied in a *Cubesats* constellation.
- The *Cubesats* are oriented in the way that their **faces are normal to the coordinate vectors**.

In a geocentric coordinate system, the Earth's equatorial plane is inclined to the ecliptic plane, which is the plane of the Earth orbit around the Sun, by an angle of 23.5 °.

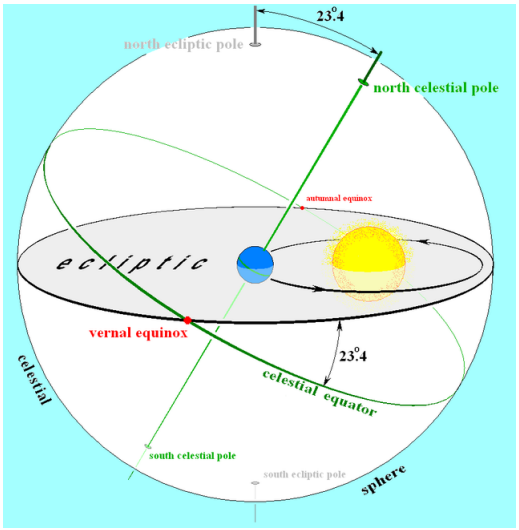


Figure 7.8: Plane of the Earth's orbit around the Sun. [15]

As it has been defined in the section 4.2, the coordinate vectors of the spacecraft are defined in the following way:

- $\vec{R}$ is a vector that points to the perigee of the orbit. Then, it is parallel and inside the plane of the orbit.
- $\vec{S}$ is a vector that is by a right angle, or in other words, follows the direction of motion.
- $\vec{W}$ is a vector that is normal to the orbit plane.

The fact to study a satellite in a GEO Equatorial orbit reduces the solution to a unique one. The vector which is orthogonal to the orbit should result in a constant angle equal to the inclination of the Earth's equatorial plane to the ecliptic, see 7.8. Moreover, this type of satellites remain in day-light except in two particular dates of the year, in spring and fall equinoxes. The eclipse starts slowly. As the Sun travels from one of the Tropics to the equator, the satellite is blocked for a minute or two, at first. Gradually the eclipse increases until the Sun reaches fall or spring equinox and the satellite is blocked for 72 minutes. This is consequence because the orbit of the satellite aligns with the ecliptic plane. As the Sun continues to travel to the other Tropic, the eclipse time becomes smaller and smaller until the spacecraft is again exposed to the Sun the entire day. [5]

Eclipse season occurs twice a year. For station-kept satellites, the spring eclipse season runs from approximately 26 February until 12 or 13 April. The fall eclipse season runs from approximately 30 or 31 August until 15 October. For inclined orbit satellites, the eclipse season starts and ends a little earlier, depending on the satellite's inclination.

Therefore a validation algorithm will be written following the next steps:

1. Check eclipse conditions. The code stores in a variable called *shadow* a 1 if there is eclipse and a 0 if not.

2. If the satellite is not in eclipse, check face by face if the sun impacts or not. That will be identified looking to the cosine between the sun vector and the satellite's coordinates vectors. A positive value, the sun arrays do not impact and the code will store a 0 in the variable *results.*. A negative value, the sun arrays impact.

3. Finally, if it impacts, there will be two solutions:

   - The incidence is between 0° and 30°, thus the satellite is still considered as operable but a little perturbed. The code will store a 1 in the variable *results.*
   - The incidence is between 30° and 90° and the satellite is saturated. The code will store a 2 in the variable *results.*

Below is shown the Matlab code developed, just for one face. The rest follow exactly the same procedure.

```
%Results for the vector directed to the orbit's perigee

if shadow(step_count,i)==0  %Check Eclipse Condition

if cos_v_RF(step_count,i)<0 %Check Sun Impact

 if solarIncidence_RF(step_count,i)<=30 && solarIncidence_RF(step_count,i)>0
 results.RF(step_count,i)=1;
        end

if solarIncidence_RF(step_count,i)>30 && solarIncidence_RF(step_count,i)<90
  results.RF(step_count,i)=2;
        end

 end

if cos_v_RF(step_count,i)>0
  results.RF(step_count,i)=0;
        end

end
```

To identify each face of the *Cubesat*, it has been followed the next figure 7.9. Regarding to $\vec{R}$ and $\vec{S}$, the index F means Front Face, referring to the positive direction of the vector, and B means Back Face, related to the negative direction of the vector. In the same way, U means Upper Face, in order to refer the positive direction of the $\vec{W}$.

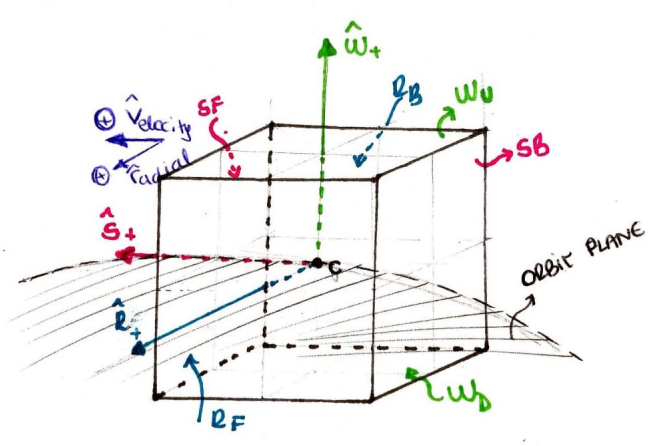


Figure 7.9: Geometry of the Cubesat

As the code computes the angles, it saves the data in a vector named *suncomponents*. It results in a 3 columns matrix for each instant of time that stores the values of the cosine between vectors sun and coordinates; the resultant angle and the calculated incidence angle. Hereunder, a 24 h simulation is done in order to have data of a complete orbit. The first simulation will be in a date where the satellite should not be in eclipse condition. Then it will be forced the simulation to run during the eclipse dates.

- Start Time : 2020 Jan 06 12:48:37 UTC(+01)
- End Time : 2020 Jan 07 12:48:37 UTC(+01)
- Simulation last : 24 h
- Number of Satellites analyzed : 1 (*Inteslat 906*)

The *suncomponents* vector related to the $\vec{W}$ in the positive direction, at the first step count, a middle step count and the last results in:

	Cosine	Angle Between Vectors	**Sun Incidence**	Date Time
First	-0,382895	112,5°	**22,5°**	2020 Jan 06 12:48:37
Middle	-0,381897	112,45°	**22,45°**	2020 Jan 07 02:50:57
Last	-0,380921	112,39°	**22,39°**	2020 Jan 07 12:48:37

Table 7.1: Values for W vector

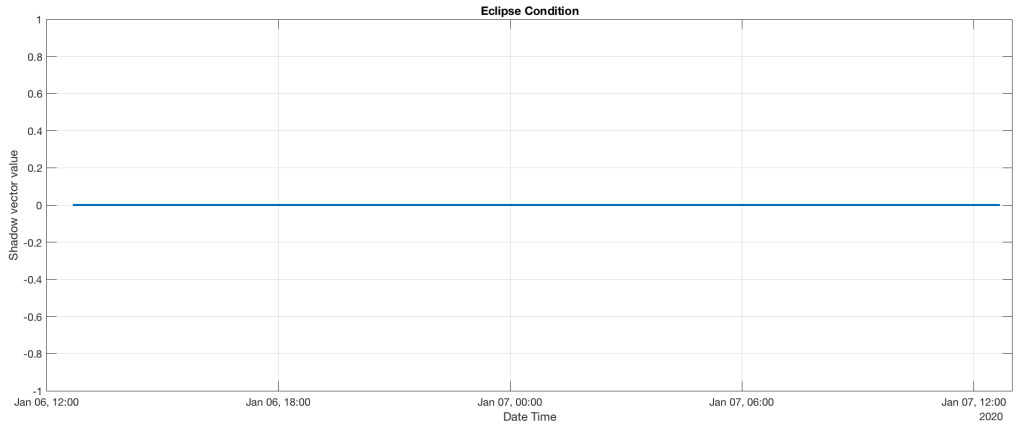


Figure 7.10: Eclipse Condition

A little variation of the incidence can be assumed since the *Intelsat 906* has an orbit inclination of 0.009°. Moreover, during the date of the simulation, the sun declination is not exactly 23,5°, it is 22,52°. The angle computed can be correct assumed.

And the *shadow* variable, as it shows 7.10, gives a null value during the whole simulation. The satellite will be exposed to the sun constantly, it does not mean that is not able to receive data. The $\vec{W}$ won't change, at least in short periods of time, because is normal to the orbit plane. But the other vectors, related to the other faces of the satellite, change their orientation in front of the sun and can be operable. The figure 7.11 shows the time periods when the satellite can receive data or not.

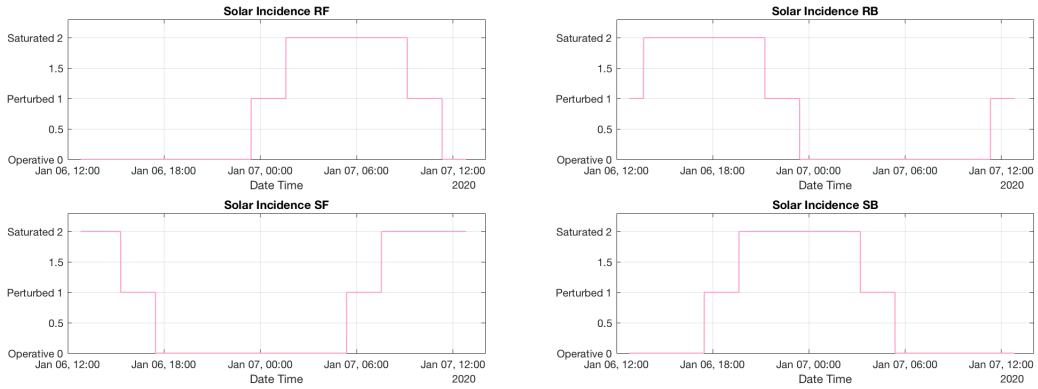


Figure 7.11: Response of the satellite to the sun incidence.

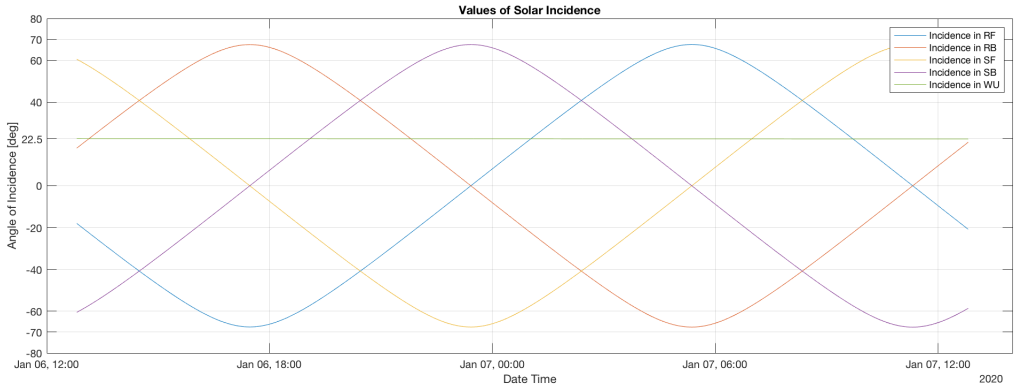


Figure 7.12: Angle values of the solar arrays that impact over the *Cubesat* faces.

As it has been predicted, the satellite is still operable if the antennas are located in the other faces. Finally, in 7.12, we can see how the incidence over each surface of the spacecraft varies in time. The green line represents the results for the positive direction of the normal vector $\vec{W}$, and the other lines are related to the both directions (front or back) of the $\vec{R}$ and $\vec{S}$. All the negative values in this case mean that the sun is not facing the surface. In terms of time of saturation, it seems that for this satellite, the location of the antennas does not affect as the periods of time when is operable is the same.

Until here, seems to work properly. Let's analyze what happens when we run the simulation during the spring equinox. The new simulation parameters are:

- Start Time : 2020 March 20 10:30:30 UTC(+01)
- End Time : 2020 March 21 10:30:30 UTC(+01)
- Simulation last : 24 h
- Number of Satellites analyzed : 1 (*Inteslat 906*)

And now the Eclipse Conditions and the *suncomponents* results for the $\vec{W}$ are :

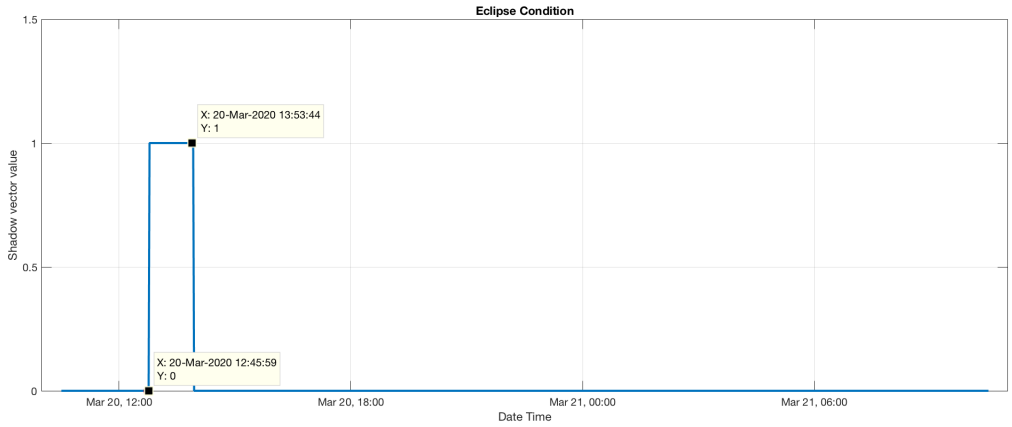


Figure 7.13: Eclipse Condition

	Cosine	Angle Between Vectors	**Sun Incidence**	Date Time
First	0,002310	89,93º	**-0,1324º**	2020 March 20 10:30:33
Middle	0,0030	89,83º	**-0,1747**	2020 March 20 13:34:57
Last	0,0090	89.45º	**-0.5274º**	2020 March 21 12:48:37

Table 7.2: Values for W vector

The satellite becomes eclipsed by the Earth between 12:45:59 h and 13:53:44 h, what it is more less 1h and 10 min, in other words, 70 min. Hence, the code is giving a satisfactory result since the simulation is done when the eclipse should last its maximum time, 72 minutes, during the equinox (20th of March). At this moment, the sun declination will be -0.13755º, thus, the $\vec{W}$ values again are in concordance.

The response of the other surfaces to the solar arrays would be:

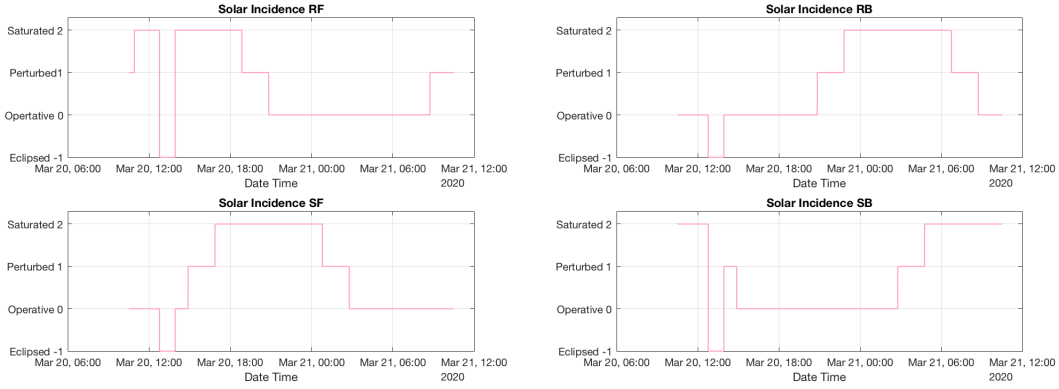


Figure 7.14: Response of the satellite to the Sun Incidence.

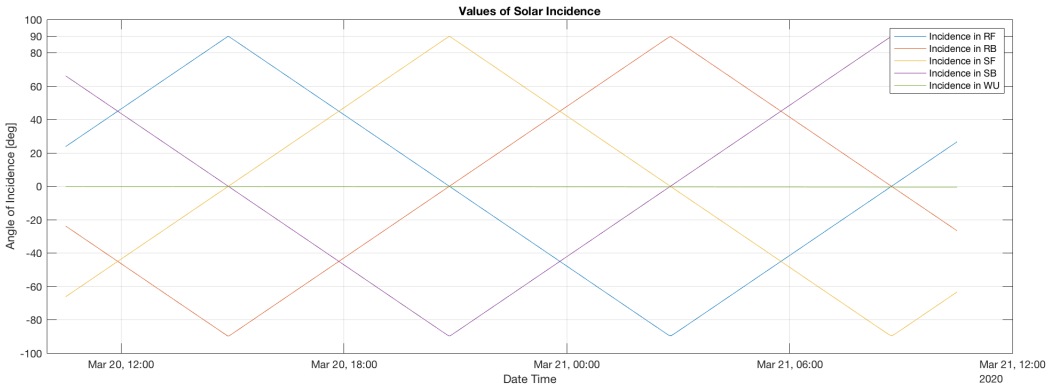


Figure 7.15: Angle values of the solar arrays that impact over the *Cubesat* faces.

In this second case, it is added the period of time of eclipse, stored as -1. During this moment, the satellite is completely operable.

The next simulation will be for the LEO satellite *Starlette*. The new simulation parameters are:

- Start Time : 2020 Jan 12 18:03:51 UTC(+01)
- End Time : 2020 Jan 12 20:33:51 UTC(+01)
- Simulation last : 2,5 h
- Number of Satellites analyzed : 1 (*Starlette*)

After running the code, we can appreciate the following intervals of eclipse:

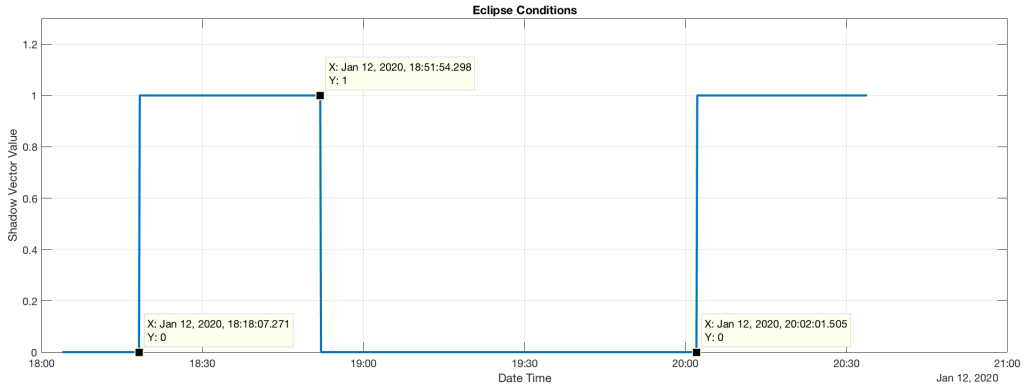


Figure 7.16: Eclipse Condition for *Starlette*

Looking to the figure above, the satellite remains in eclipse during intervals of 33,47 minutes when it has an orbital period of 104,17 minutes. If it doesn't lie on the Earth's shadow during periods of 70,46 minutes, the sum of a eclipse interval and non-eclipse interval should give as a result a complete orbital period.

$$33,47 + 70,46 = 103,46 minutes \tag{7.1}$$

A comparisson with the eclipse times prediction for the website will be done. The following pictures show the approximated timetable.

The website predicts the following:

- Date Time 2020 Jan 12 19:41:20 (UTC +01) = No eclipsed

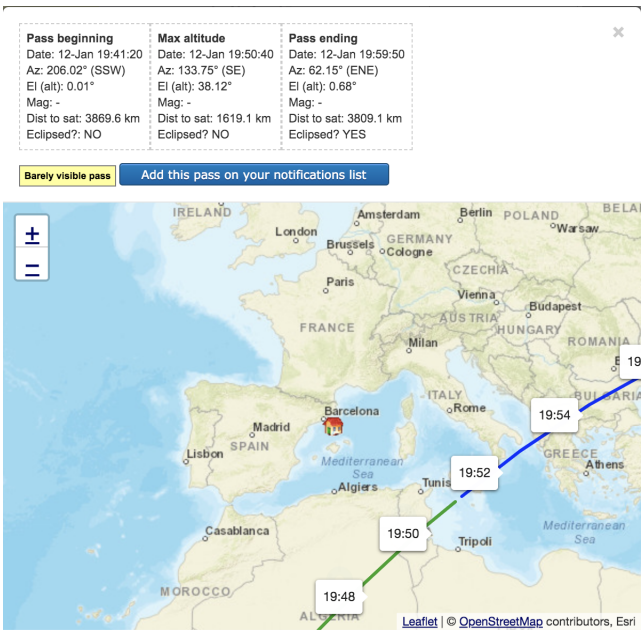


Figure 7.17: *Starlette* Tracking prediction

- Date Time 2020 Jan 12 19:50:40 (UTC +01) = No eclipsed
- Date Time 2020 Jan 12 19:59:50 (UTC +01) = Eclipsed

Looking to the figure 7.16, the satellite becomes eclipsed at 20:02 h (UTC +01). Therefore, it can be concluded that the code estimates a good prediction if the satellite lies on the Earth's shadow or not. These is also a consequence of the accuracy in predicting the satellites' position. Reggarding to the incidence of the solar arrays, the next figures will show the periods of time that will saturate the satellite or not and the variation of the angle of incidence.

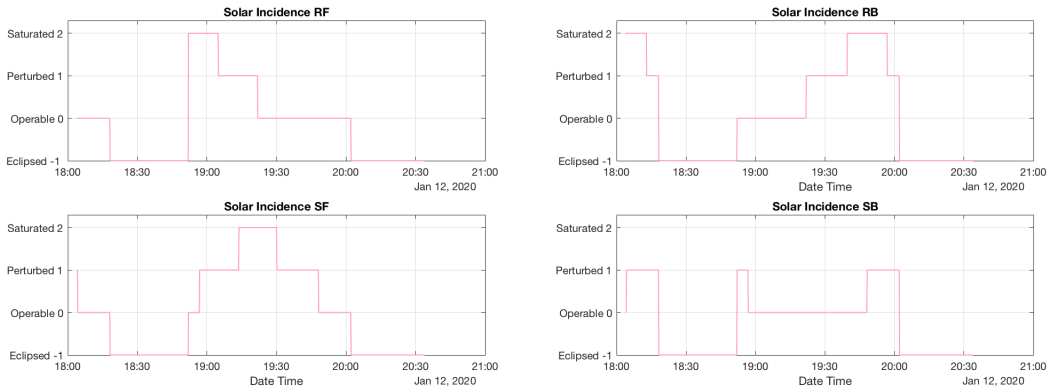


Figure 7.18: Response of the satellite to the Sun Incidence. [7]

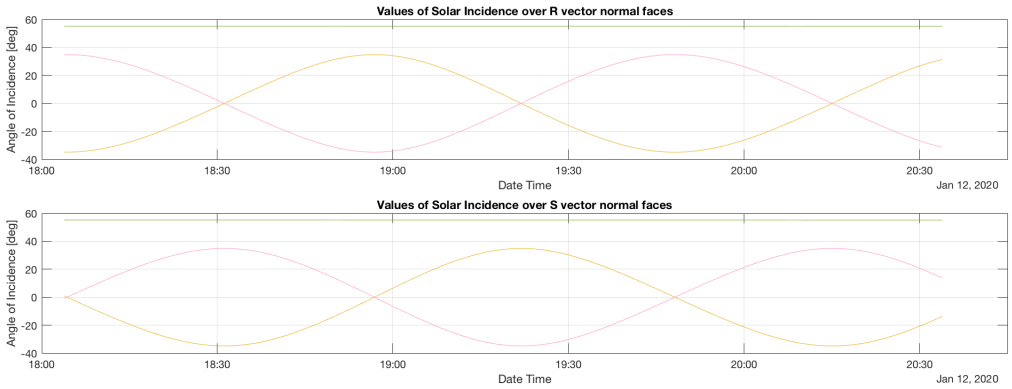


Figure 7.19: Angle values of the solar arrays that impact over the *Cubesat* faces. Lines in orange color refer to the positive direction and lines in pink, refer to the negative direction. Line green refers to the postive $\vec{W}$ vector.

In this case, the response of the satellite to the solar arrays seems to differ between orthogonal faces. Hence, in order to guarantee the longest period of time of communication, the code is also prepared to estimate the time ratios for each condition (saturation, perturbation, operational or eclipsed) during the simulation.

Condition	**RF**(%)	**RB**(%)	**SF**(%)	**SB**(%)	**WU**(%)	**WD**(%)
Eclipsed	43.7	43.7	43.7	43.7	43.7	43.7
Operational	36.4	19.9	21.8	34.5	0	56.3
Perturbed	11.3	18.8	23.6	21.8	0	0
Saturated	8.6	17.6	10.9	0	56.3	0

Table 7.3: Time ratios of the response to the sun incidence over each face of the spacecraft.

And finally if the satellite will be able to receive data during the eclipsed, operational and perturbed conditions, the time ratio results in :

Face	**Total ratio of time operable** (%)
RF	91.4
RB	82.4
SF	89.1
SB	100
WU	43.7
WD	100

Table 7.4: Total ratio of time that the satellite will be operational.

To conclude with *Starlette*, the operational times seem to be quiet good in almost all the faces in the dates of the simulation. The next step would be to choose the face where the satellite will receive and emit data.

Chapter 8

Future work

The present study takes place in the research group *DISEN (2017 SGR 1170)* who pretend to develop a network of optical communications with a *Cubesats'* platform. The software that has been written will serve them to have real time position of their *Cubesats* and whether they are operable or not.

This thesis follows up two projects. One estimates the time visibility between satellite and ground station and the other the time visibility between satellites. By adding the work done in this project, the group will be able to develop a software that must :

1. Check the visibility between the satellites that will communicate.
2. Check if the receptor is operable or not. In case not, the transmitter will have to chose another visible *Cubesat* and check again if its operable.
3. Establish communication.
4. Once the communication will have finish, follow the satellite in its orbit until the ground station detects that is visible for it and can start to downlink the data provided for the transmitter satellite.

In order to improve the algorithm, the next step should be to study the perturbations suffered from the near Earth satellites. The orbital elements of the *LEO* satellites are mainly influenced by the atmospheric drag, the non-sphericity of the Earth and the gravitational force of the Moon. The fact of not considering them reduces the accuracy of the position estimated. [14]

Effects which modify simple Keplerian orbits may be divided into three clases: nongravitational forces, third-body interactions and nonspherical mass distributions. The first two effects may dominate the motion of the spacecraft, as in satellite reentry into the atmosphere or motion about one of the stable [1]Lagrange points. Although the effects of nonspherical mass distributions never dominate the spacecraft motion, they provide the major perturbation, relative to Keplerian orbits, for most intermediate altitude satellites, i.e, those above where the atmosphere plays an important role and below where effects due to the Moon and the Sun become important.

Although the relative importance of these three significant groups of perturbations will depend on the construction of the spacecraft, the details of its orbit, and even the level of solar activity, the general effect of perturbing forces is clear. Atmospheric effects dominate the perturbing forces at altitudes below about 100 km and produce significant long-term perturbations on satellite orbits up to altitudes of about 1000 km. The major effect from the non-spherical symmetry of the Earth is due to the Earth's oblateness, which changes the gravitational potential by about 0.1 % in the vicinity of the Eart. The ratio of the gravitational potential of the Moon to that of the Earth is 0.02 % near the Earth's surface. As the satellite's altitude increases, the effect of oblateness decreases and the effect of the Moon increases; the magnitude of their effect on the gravitational potential is the same at about 8000 km altitude. Lunar and solar perturbations are generally negligible at altitudes below about 700 km.

[1]In celestial mechanics, the Lagrangian points are the points near two large bodies in orbit where a smaller object will maintain its position relative to the large orbiting bodies. At other locations, a small object would go into its own orbit around one of the large bodies, but at the Lagrangian points the gravitational forces of the two large bodies, the centripetal force of orbital motion, and (for certain points) the Coriolis acceleration all match up in a way that cause the small object to maintain a stable or nearly stable position relative to the large bodies.

One last event may be studied in a future work. This thesis only takes into account the Earth eclipses, which are the most representatives when the sun radiation is being studied. Despite that, the Moon also could eclipse the satellite and hide the satellite in front of the sun. In order to get the complete time of the satellite in operable mode, a Lunar Eclipse study should be done.

Chapter 9

Conclusions

This thesis describes a method for the rapid determination of a satellite's position and, then, its response when is exposed to the sun. For any space mission of communication is highly important identify the influence that is inducing the sun over the spacecraft since that it will not be able to receive information from another satellite when is saturated by the sun. Therefore, the present work will directly help in two ways. Before launching the satellites, it can be useful to estimate which face of the *Cubesat* will be better to communicate. By entering the orbit data where the satellite will be placed, a simulation can be done to estimate the time ratios when the satellite is operable. From the other hand, after lunching the satellites, the work will allow the users to identify if the satellite is in operation mode or, on the contrary, is saturated by the sun.

The choice of this study was due to the deeply interest in the space subject. It was a good opportunity to improve the knowledge in the attitude, control and tracking determination of celestial objects. Simultaneously, it would help to learn more about developing algorithms. The amount of work was not little, and the limited amount of time made to prioritize some aspects. However, after finishing the inform I realize how many things I have learnt.

At the beginnings, many difficulties came about. One was to solve the positioning and geometry problem in a generic way that could be extrapolated to any satellite. Once a solution was achieved, the problem was to validate that the feasibility of the software developed in an elegant form. The best form was finding a class of satellites that could only give a unique solution. That's why a constellation of geostationary satellites was chosen although, the research group will use the software for low earth satellites. The website *NY2O.com* was highly useful to contrast my results with the reality and to be able to add a *LEO* case that could be verified with real data.

However, some difficulties were find during the validation. The geostationary satellites was easier as they move slower and are always on the same position over the Earth. When the purpose was to obtain data from a near Earth satellite that changes its position faster, the problem was to finish everything at the same moment. The real time tracking was on the whole time, thus I could not write a part from the validation at one specific hour, and continue in a while, as long as the satellite will have moved. Nevertheless, I believe that the validation chapter has satisfactory results.

On the other hand, verifying the sun incidence was more difficult. Again, the fact of using *GEO* helped in the way that the code could only give one solution as they have defined eclipses dates and times. It was very satisfactory check that the results computed were in concordance with the reality. A more accurate validation may be had done for the *LEO* case, although if the code works properly for one satellite, it should work for the others. The only problem is the hypothesis done at the moment of not considering the orbit perturbations. But this is not included in the study.

After all, the software seems to work properly since the results have been rather accurate. At least, GEO constellations are well approximated. However, it must be improved clearly. As the code simulates satellites in orbits nearer to the Earth, the error becomes higher and higher. Nevertheless, in the way I see it, the results are truly satisfactory.

In conclusion, I will say that I have enjoyed the time spent looking into the study. When something likes, it does not take much to continue doing research and trying until find the solution. From my point of view, I feel satisfied with the worked done and how much I have learnt about the space subject.

Chapter 10

Environmental and security impact

Nowadays, the importance of the environment is becoming highly important. This project is an analytic study and a simulation of the satellites orbits. Hence, it does not have a direct environmental impact. However, it is true that space industry has an important impact in terms of the satellites development and launches. This must be taken in to account before take up space missions.

As the project is related to a group research that wants to build a constellation of LEO satellites, it is worth mentioning the increasing space debris in the atmosphere. Nevertheless, this project is not only closed to LEO satellites. The code developed can be used with all types of satellites in the Solar System.

In relation to the security impact, the only problem to face could be the use in an appropriate way of the computer. Currently, the topic of rely on the computers, has become truly popular nowadays. We get used to work and do everything with them and when we do not have one, we do not know how to proceed. In addition, a day a day use of them can lead to vision problems or back problem due to bad positions, between others.

Bibliography

[1] Astronomical Times. Available at https://www.cv.nrao.edu/{~}rfisher/Ephemerides/times.html, (Accessed on 12/01/2020).

[2] Catalog of earth satellite orbits. Available at https://earthobservatory.nasa.gov/features/OrbitsCatalog (Accessed on 12/01/2020).

[3] CelesTrak: Current NORAD Two-Line Element Sets. Available at https://www.celestrak.com/NORAD/elements/ , (Accessed on 12/01/2020).

[4] Concepto de satélites artificiales. Available at https://concepto.de/satelites-artificiales/, (Accessed on 12/01/2020).

[5] Eclipse Seasons — Intelsat. Available at http://www.intelsat.com/tools-resources/library/satellite-101/eclipse-seasons/, (Accessed on 12/01/2020).

[6] Kepler's three laws. The Physics Classroom. Available at https://www.physicsclassroom.com/class/circles/Lesson-4/Kepler-s-Three-Laws, (Accessed on 12/01/2020).

[7] Real time satellite tracking. https://www.n2yo.com, (Accessed on 12/01/2020).

[8] List of satellite pass predictors. Wikipedia, Jan 2020. Available at https://en.wikipedia.org/wiki/List_of_satellite_pass_predictors, (Accessed on 12/01/2020).

[9] NASA Content Administrator. Sputnik 1. NASA, Nov 2015. Available at https://www.nasa.gov/multimedia/imagegallery/image_feature_924.html,(Accessed on 12/01/2020).

[10] Ronald Charca Choque. SATÉLITE ARTIFICIAL. Technical report.

[11] Alessandro de Iaco Veris. Shadow times of Earth satellites. Rivista italiana di compositi e nanotecnologie, Volume 9(September):pages 7–20, 2014.

[12] Thierry Guillemin. LEO Satellite Constellations: Understanding Strengths and Limitations — Telecom Engine. Available at https://www.telecomengine.com/leo-satellite-constellations-understanding-strengths-and-limitations/, (Accessed on 12/01/2020), 2015.

[13] Syahrim Azhan Ibrahim and Eiki Yamaguchi. Comparison of solar radiation torque and power generation of deployable solar panel configurations on nanosatellites. Aerospace, 6(5), 2019.

[14] Wertz J.R. Spacecraft Attitude Determination and Control. Springer, Dordrecht.

[15] Hiroshi Kinoshita and Shinko Aoki. The definition of the ecliptic. Celestial Mechanics, 31(4):329–338, dec 1983.

[16] Hyung-Chul Lim, Hyo-Choong Bang, and Sang-Jong Lee. Adaptive Backstepping Control for Satellite Formation Flying With Mass Uncertainty. Journal of Astronomy and Space Sciences, 23(4):405–414, 2006.

[17] Elizabeth Mabrouk. What are SmallSats and CubeSats? NASA, Mar 2015.

[18] Meysman Mahooti. Satellite orbits: Models, methods and applications - file exchange - matlab central. Available at https://www.mathworks.com/matlabcentral/fileexchange/55533-satellite-orbits-models-methods-and-applications, (Accessed on 12/01/2020).

[19] Marek Moeckel. High-Performance Propagation of Large Object Populations in Earth Orbits. PhD thesis, 10 2015.

[20] M.A. Sharaf, M.E. Awad, I.A. Hassan, R. Ghoneim, and W.N. Ahmed. Visual contact for two satellites orbits under J 2 -gravity . (1):61–69. NRIAG Journal of Astronomy and Geophysics.

[21] Kristian Svartveit. Attitude determination of the NCUBE satellite. Engineering Cybernetics, page 140, 2003.

[22] M Zahran. In Orbit Performance of LEO Satellite Electrical Power Subsystem - SW Package for Modelling and Simulation Based on MatLab . 7 GUI. 2006(January 2006):379–384, 2006.

[23] N. Zehentner. Kinematic orbit positioning applying the raw observation approach to observe time variable gravity. (May):175, 2016.